ABSTRACT

THOMPSON, KYLE B. Aerothermodynamic Design Sensitivities for a Reacting Gas Flow Solver on an Unstructured Mesh Using a Discrete Adjoint Formulation. (Under the direction of Hassan Hassan and Peter Gnoffo.)

Approach is described to efficiently compute aerothermodynamic design sensitivities using a decoupled approach. In this approach, the species continuity equations are decoupled from the mixture continuity, momentum, and total energy equations for the Roe flux difference splitting scheme in both the flow and adjoint solvers. This decoupling simplifies the implicit system, so that the flow solver can be made significantly more efficient, with very little penalty on overall scheme robustness. Most importantly, the computational cost of the point implicit relaxation is shown to scale linearly with the number of species for the decoupled system, whereas the fully coupled approach scales quadratically. Also, the decoupled method significantly reduces the cost in wall time and memory in comparison to the fully coupled approach.

A design optimization of a re-entry vehicle with a annular nozzle on the forebody is completed based on this approach. The sensitivities of the drag coefficient and surface temperature with respect to a plenum inboard the vehicle were computed and verified against complex-variable finite-difference.

© Copyright 2016 by Kyle B. Thompson

All Rights Reserved

Aerothermodynamic Design Sensitivities for a Reacting Gas Flow Solver on an Unstructured Mesh Using a Discrete Adjoint Formulation

by
Kyle B. Thompson

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Aerospace Engineering

Raleigh, North Carolina

2016

APPROVED BY:

Hassan Hassan Co-chair of Advisory Committee	Peter Gnoffo Co-chair of Advisory Committee
Jack Edwards	Hong Luo

John Griggs

DEDICATION

To my parents and friends.

BIOGRAPHY

The author was born on December 11th, 1990, in Willow Springs, North Carolina. He recieved a BS in Aerospace Engineering from North Carolina State University in 2012, and subsequently recieved his MS in Aerospace Engineering from North Carolina State University in 2014. After recieving he MS, he began work in the Aerothermodynamics branch of NASA Langley Research Center, via the Pathways program. He completed this disseration on adjoint-based optimization while working at NASA Langley Research Center.

ACKNOWLEDGEMENTS

I would like thank my parents for all of their encouragement over the years, and their commitment to seeing I be given the best opportunity to make the most of myself. I would like to thank my advisor for instilling in me a diligence to improve myself and for mentoring me through many challenges. I would like to thank the Entry Systems Modeling Project within NASA's Game Changing Development Program for their funding and support of this research. Finally, I would like to recognize the FUN3D team at NASA Langley Research Center, for their support and availability to discuss many challenging problems I have encountered during the course of my time at NASA.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

The usability of hypersonic solvers on complex geometries is often limited by the extreme problem size associated with high energy physics. The additional equations required in reacting gas simulations lead to large Jacobians that scale quadratically in size to the number of governing equations. This leads to a significant increase in the memory required to store the flux linearizations and the computational cost of the point solver. As reacting gas CFD solvers are used to solve increasingly more complex problems, this onerous quadratic scaling of computational cost and Jacobian size will ultimately surpass the current limits of hardware and time constraints on achieving a flow solution[2].

The proposed method is based heavily upon the work of Candler et al.[1]. In that work, it was shown that quadratic scaling between the cost of solving the implicit system and adding species mass equations can be reduced from quadratic to linear scaling by decoupling the species mass equations from the mixture mass, momentum, and energy equations and solving the two systems sequentially. In the aforementioned work, the scheme was derived for a modified form of the Steger-Warming flux vector splitting method[9], whereas the work presented here is derived for the Roe flux difference splitting (FDS) scheme[12].

A primary motivator for the presented work is to prepare for the implementation of an adjoint capability in a reacting gas solver that employs the Roe scheme. Developing an adjoint implementation for non-equilibrium flows is an extremely challenging problem, because deriving exact Jacobians for a reacting gas system is particularly difficult. Decoupling the system also simplifies the derivation of exact Jacobians. If the species mass equations are decoupled, the mixture mass, momentum, and energy flux Jacobians can be easily derived[10], and a weighting scheme can be used to correct the non-uniqueness of the pressure linearization[14]. For the decoupled species fluxes, we derive an exact linearization in the presented work. Future work will include an adjoint-based error estimation capability that leverages these exact linearizations.

Chapter 2

Numerical Solution of Flow Equations

2.1 Fully-Coupled Point Implicit Method

All work presented here is for the inviscid conservation equations, but it can be extended to include viscous terms. For an inviscid, multi-species mixture, the governing equations in vector form are:

$$\frac{\partial \mathbf{U}}{\partial t} + \nabla \cdot \mathbf{F} = \mathbf{W} \tag{2.1}$$

or, in semi-discrete form,

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{1}{V} \sum_{f} (\mathbf{F} \cdot \mathbf{S})^f = \mathbf{W} \tag{2.2}$$

summing over all faces, f, in the domain, where V is the cell volume, $\mathbf{W}$ is the chemical source term vector, and $\mathbf{S}$ is the face outward normal vector. The vectors of conserved variables and fluxes are:

$$\mathbf{U} = \begin{pmatrix} \rho_1 \\ \vdots \\ \rho_{ns} \\ \rho u \\ \rho v \\ \rho w \\ \rho E \end{pmatrix}, \quad \mathbf{F} = \begin{pmatrix} \rho_1 \overline{U} \\ \vdots \\ \rho_{ns} \overline{U} \\ \rho u \overline{U} + p s_x \\ \rho u \overline{U} + p s_y \\ \rho u \overline{U} + p s_z \\ (\rho E + p) \overline{U} \end{pmatrix} \tag{2.3}$$

where $\overline{U}$ is the outward pointing normal velocity, and E is the total energy of the mixture per unit mass, defined as:

$$E = \sum c_s e_s + \frac{u^2 + v^2 + w^2}{2} \tag{2.4}$$

where e_s is the internal energy of species s. By using the Roe FDS scheme,

$$\begin{aligned} \mathbf{F}^{n+1} &\approx \mathbf{F}^n + \tfrac{\partial \mathbf{F}}{\partial \mathbf{U}} \delta \mathbf{U}^n \\ \mathbf{W}^{n+1} &\approx \mathbf{W}^n + \tfrac{\partial \mathbf{W}}{\partial \mathbf{U}} \delta \mathbf{U}^n \end{aligned} \tag{2.5}$$

where $\delta \mathbf{U}^n = \mathbf{U}^{n+1} - \mathbf{U}^n$. By using an implicit time integration, the implicit scheme becomes:

$$\frac{\delta \mathbf{U}^n}{\Delta t} + \frac{1}{V} \sum_f (\frac{\partial \mathbf{F}^f}{\partial \mathbf{U^L}} \delta \mathbf{U}^L + \frac{\partial \mathbf{F}^f}{\partial \mathbf{U^R}} \delta \mathbf{U}^R)^n \mathbf{S}^f - \frac{\partial \mathbf{W}}{\partial \mathbf{U}} \delta \mathbf{U}^n = -\frac{1}{V} \sum_f (\mathbf{F}^f \cdot \mathbf{S}^f)^n + \mathbf{W}^n \tag{2.6}$$

or, put more simply:

$$A \delta \mathbf{U}^n = \mathbf{b} \tag{2.7}$$

where A is the Jacobian matrix of the fully coupled system, and $\mathbf{b}$ is the residual vector. For a point implicit relaxation scheme, the Jacobian matrix can be split into its diagonal and off-diagonal elements, with the latter moved to the RHS:

$$A = O + D \tag{2.8}$$

Each matrix element is a square $(ns+4) \times (ns+4)$ matrix. One method of solving this system is a Red-Black Gauss-Seidel scheme[13], where matrix coefficients with even indices are updated first and, subsequently, the coefficients with odd indices are updated. This red-black ordering enables better vectorization in solving the linear system. The computational work for the Gauss-Seidel scheme is dominated by matrix-vector multiplications of elements of O with $\delta \mathbf{U}$, which are $O(N^2)$ operations, where $N = ns + 4$. In the next section, it is shown that decoupling the system reduces these matrix-vector multiplications to $O(N^2 + M)$ operations, where $N = ns$ and $M = ns$.

2.2 Decoupled Point Implicit Method

If the species mass equations are replaced by a single mixture mass equation, the mixture equations can be separated from the species mass equations and the conserved variables become

$$\mathbf{U}' = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E \end{pmatrix} \quad \hat{\mathbf{U}} = \begin{pmatrix} \rho_1 \\ \vdots \\ \rho_{ns} \end{pmatrix} \tag{2.9}$$

Solving the flux vector is performed in two sequential steps. The mixture fluxes are first solved as

$$\frac{\partial \mathbf{U}'}{\partial t} + \frac{1}{V}\sum_f (\mathbf{F}' \cdot \mathbf{S})^f = 0 \tag{2.10}$$

followed by the species fluxes as

$$\frac{\partial \hat{\mathbf{U}}}{\partial t} + \frac{1}{V}\sum_f (\hat{\mathbf{F}} \cdot \mathbf{S})^f = \hat{\mathbf{W}} \tag{2.11}$$

Point relaxation uses Red-Black Gauss-Seidel to update the conserved variables in $\mathbf{U}'$ and all associated auxiliary variables, such as temperature, pressure, speed of sound, etc. This is done by holding the thermo-chemical state constant, and will always result in the relaxation of a five-equation system. This does trade an implicit relationship between the mixture and species equations for an explicit one; thus, this decoupling can have an impact on the stability of the scheme, especially due to the non-linearity of the chemical source term[11].

The solution of the species mass equations takes a different form. Based on the work of Candler et al.[1], the decoupled variables can be rewritten in terms of mass fraction, as follows:

$$\delta\hat{\mathbf{U}}^n = \rho^{n+1}\hat{\mathbf{V}}^{n+1} - \rho^n\hat{\mathbf{V}}^n = \rho^{n+1}\delta\hat{\mathbf{V}}^n + \hat{\mathbf{V}}^n\delta\rho^n \tag{2.12}$$

where $\hat{\mathbf{V}} = (c_1, \ldots, c_{ns})^T$, and $c_s = \rho_s/\rho$ the mass fraction of species s. While the derivation of the species mass equations is different for the Roe FVS scheme from that of Steger-Warming proposed by Candler et al.[1], the final result takes a similar form:

$$\hat{F}_{\rho_s} = c_s F'_\rho + (c_s^L - \tilde{c}_s)\rho^L\lambda^+ + (c_s^R - \tilde{c}_s)\rho^R\lambda^- \tag{2.13}$$

where F'_ρ is the total mass flux computed previously using all $\mathbf{U}'$ variables, and $\tilde{}$ denotes a Roe-averaged quantity. Likewise, linearizing the species mass fluxes with respect to the $\hat{\mathbf{V}}$ variables

yields

$$\hat{\mathbf{F}}^{n+1} = \hat{\mathbf{F}}^{n} + \frac{\partial \hat{\mathbf{F}}}{\partial \hat{\mathbf{V}}^{L}} \delta \hat{\mathbf{V}}^{L} + \frac{\partial \hat{\mathbf{F}}}{\partial \hat{\mathbf{V}}^{R}} \delta \hat{\mathbf{V}}^{R} \tag{2.14}$$

$$\frac{\partial \hat{\mathbf{F}}}{\partial \hat{\mathbf{V}}^{L}} = wF_{\rho} + (1-w)\rho^{L}\lambda^{+} - w\rho^{R}\lambda^{-} \tag{2.15}$$

$$\frac{\partial \hat{\mathbf{F}}}{\partial \hat{\mathbf{V}}^{R}} = (1-w)F_{\rho} + (w-1)\rho^{L}\lambda^{+} + w\rho^{R}\lambda^{-} \tag{2.16}$$

A full derivation of Eqs. (2.13-2.16), along with the definition of w, is included in Appendix A. The chemical source term is linearized in the same manner as the fully coupled scheme; however, the updated $\mathbf{U}'$ variables are used to evaluate the Jacobian, and the chain rule is applied to linearize $\hat{\mathbf{W}}$ with respect to the species mass fractions:

$$\hat{\mathbf{W}}^{n+1} = \hat{\mathbf{W}}^{n} + \left.\frac{\partial \hat{\mathbf{W}}}{\partial \mathbf{U}}\right|_{\mathbf{U}'} \frac{\partial \mathbf{U}}{\partial \hat{\mathbf{V}}} \tag{2.17}$$

For simplicity of notation, we define

$$C = \left.\frac{\partial \hat{\mathbf{W}}}{\partial \mathbf{U}}\right|_{\mathbf{U}'} \frac{\partial \mathbf{U}}{\partial \hat{\mathbf{V}}} \tag{2.18}$$

The decoupled system to be solved becomes:

$$\begin{aligned} \rho^{n+1}\frac{\delta \hat{\mathbf{V}}^{n}}{\Delta t} + \frac{1}{V}\sum_{f}\left(\frac{\partial \hat{\mathbf{F}}^{f}}{\partial \hat{\mathbf{V}}^{L}}\delta \hat{\mathbf{V}}^{L} + \frac{\partial \hat{\mathbf{F}}^{f}}{\partial \hat{\mathbf{V}}^{R}}\delta \hat{\mathbf{V}}^{R}\right)^{n,n+1}\mathbf{S}^{f} - C^{n,n+1}\delta \mathbf{V}^{n} \\ = -\frac{1}{V}\sum_{f}(\hat{\mathbf{F}}^{n,n+1}\cdot\mathbf{S})^{f} + \mathbf{W}^{n,n+1} - \hat{\mathbf{V}}^{n}\frac{\delta \rho^{n}}{\Delta t} - R_{\rho} \end{aligned} \tag{2.19}$$

$$R_{\rho} = -\frac{1}{V}\sum_{f}\sum_{s}(\hat{F}_{\rho_s}^{n,n+1}\cdot\mathbf{S}) \tag{2.20}$$

where R_{ρ} is included to preserve the constraint that the mass fractions sum to unity, i.e., $\sum_{s} c_s = 1$, $\sum_{s} \delta c_s = 0$.

2.3 Cost and Memory Savings of the Decoupled Implicit Problem

In decoupling the species equations, the most significant savings comes from the source term linearization being purely node-based[6]. Solving the mean flow equations is conducted in the same manner as the fully coupled system. All entries in the Jacobian A_m are linearizations of

the mixture equation fluxes, which results in 5×5 matrices. All entries in the Jacobian A_d are linearizations of the species mass fluxes, which results in $ns \times ns$ matrices. Because there is no interdependence of species, except through the chemical source term, all contributions due to linearizing the convective flux are purely diagonal $ns \times ns$ matrices. Via Eq. (2.8), we decompose A_d into its diagonal and off-diagonal elements, resulting in the following linear system:

$$\begin{pmatrix} \square & & & & \\ & \ddots & & & \\ & & \square & & \\ & & & \ddots & \\ & & & & \square \end{pmatrix} \begin{pmatrix} \delta\hat{\mathbf{V}}_1 \\ \vdots \\ \delta\hat{\mathbf{V}}_i \\ \vdots \\ \delta\hat{\mathbf{V}}_{nodes} \end{pmatrix} = \begin{pmatrix} \hat{b}_1 \\ \vdots \\ \hat{b}_i \\ \vdots \\ \hat{b}_{nodes} \end{pmatrix} - \begin{pmatrix} (\sum_{j=1}^{N_{nb}} [\diagdown] \delta\hat{\mathbf{V}}_j)_1 \\ \vdots \\ (\sum_{j=1}^{N_{nb}} [\diagdown] \delta\hat{\mathbf{V}}_j)_i \\ \vdots \\ (\sum_{j=1}^{N_{nb}} [\diagdown] \delta\hat{\mathbf{V}}_j)_{nodes} \end{pmatrix} \tag{2.21}$$

where $\square$ represents a dense $ns \times ns$ matrix, $[\diagdown]$ represents a diagonal matrix, and $\delta\hat{\mathbf{V}}_j$ is the decoupled variable update on the node j that neighbors node i, where N_{nb} is the number of nodes neighboring node i. Thus, the non-zero entries in the off-diagonal matrix can be reduced from diagonal matrices to vectors. This results in significant savings in both computational cost and memory, as the only quadratic operation left in solving the implicit system is dealing with the diagonal entries in the Jacobian. Because the off-diagonal entries significantly outnumber the diagonal entries, we can expect nearly linear scaling in cost with the number of species. If compressed row storage[3] is used to only store non-zero off-diagonal entries, the relative memory savings in the limit of a large number of species for the Jacobian is given by

$$\begin{aligned} \textit{Relative Memory Cost} &= \frac{size(A_d)}{size(A)} \\ &= \lim_{ns \to \infty} \frac{(ns^2 + 5^2)(N_{nodes}) + (ns + 5^2)(N_{nz})}{(ns + 4)^2 (N_{nodes} + N_{nz})} \\ &= \frac{N_{nodes}}{N_{nodes} + N_{nz}} \end{aligned} \tag{2.22}$$

where N_{nodes} is the number of nodes, and N_{nz} is the number of non-zero off-diagonal entries stored using compressed row storage. For a structured grid, each node has six neighbors in 3D, i.e., $N_{nz} = 6N_{nodes}$; therefore, we can expect the Jacobian memory required to decrease by a factor of seven using this decoupled scheme. Interestingly, for a grid that is not purely hexahedral, $N_{nz} > 6N_{nodes}$; thus, this decoupled scheme provides higher relative memory savings on unstructured grids than structured grids when using compressed row storage.

Chapter 3

Numerical Solution of Adjoint Equations

This section details the derivation of the adjoint equations to be solved in conjuction with the primal flow equations. The primary goal of this research is to compute sensitivities of aerodynamic and aerothermodynamic quantities to design variables. To achieve this, the sensitivity to the primal flow equation formulation must first be solved. For a discrete adjoint formulation, this requires a solution of costate variables, relating a change in the flow equation residual to a change in the function of interest. Because of the large number of equations required in a reacting gas solver, the adjoint solver will suffer from the quadratic scaling in computational cost and memory required similarly to the primal flow solver. To mitigate this, a decoupled scheme is derived that is consistent with the decoupled flow solver.

3.1 Discrete Adjoint Derivation

The derivation for the discrete adjoint begins with forming the Lagrangian as

$$L(\mathbf{D}, \mathbf{Q}, \mathbf{X}, \mathbf{\Lambda}) = f(\mathbf{D}, \mathbf{Q}, \mathbf{X}) + \mathbf{\Lambda}^T \mathbf{R}(\mathbf{D}, \mathbf{Q}, \mathbf{X}) \tag{3.1}$$

Where $\mathbf{R}$ is the residual of the flow equations. Differentiating with respect to the design variables $\mathbf{D}$ yields

$$\frac{\partial L}{\partial \mathbf{D}} = \left\{ \frac{\partial f}{\partial \mathbf{D}} + \left[\frac{\partial \mathbf{X}}{\partial \mathbf{D}}\right]^T \frac{\partial f}{\partial \mathbf{X}} \right\} + \left[\frac{\partial \mathbf{Q}}{\partial \mathbf{D}}\right]^T \left\{ \frac{\partial f}{\partial \mathbf{Q}} + \left[\frac{\partial \mathbf{R}}{\partial \mathbf{Q}}\right]^T \mathbf{\Lambda} \right\} + \left\{ \left[\frac{\partial \mathbf{R}}{\partial \mathbf{D}}\right]^T + \left[\frac{\partial \mathbf{X}}{\partial \mathbf{D}}\right]^T \left[\frac{\partial \mathbf{R}}{\partial \mathbf{X}}\right]^T \right\} \mathbf{\Lambda} \tag{3.2}$$

To eliminate the dependence of conserved variables $\mathbf{Q}$ on the design variables, we solve the adjoint equation

$$\left[\frac{\partial \mathbf{R}}{\partial \mathbf{Q}}\right]^T \mathbf{\Lambda} = -\frac{\partial f}{\partial \mathbf{Q}} \tag{3.3}$$

Where the Lagrange multipliers (also known as costate variables), $\mathbf{\Lambda}$ are the cost function dependence on the residual

$$\mathbf{\Lambda} = -\frac{\partial f}{\partial \mathbf{R}} \tag{3.4}$$

This can ultimately be used to error estimation and sensitivity analysis for design optimization. With the second term in eq. (3.2) eliminated, the derivative of the Lagrangian becomes

$$\frac{\partial L}{\partial \mathbf{D}} = \left\{ \frac{\partial f}{\partial \mathbf{D}} + \left[\frac{\partial \mathbf{X}}{\partial \mathbf{D}}\right]^T \frac{\partial f}{\partial \mathbf{X}} \right\} + \left\{ \left[\frac{\partial \mathbf{R}}{\partial \mathbf{D}}\right]^T + \left[\frac{\partial \mathbf{X}}{\partial \mathbf{D}}\right]^T \left[\frac{\partial \mathbf{R}}{\partial \mathbf{X}}\right]^T \right\} \mathbf{\Lambda} \tag{3.5}$$

By solving the adjoint equation in Eq. 3.3) to obtain the costate variable vector, $\mathbf{\Lambda}$, we can now use a non-linear optimizer to determine the optimum set of design variables, $\mathbf{D}^*$. This can be done using **SNOPT[5]**, **KSOPT[8]**, or **NPSOL[4]** in FUN3D, as well as a host of other non-linear optimizers.

3.2 Block Jacobi Adjoint Decoupling

It is possible to decoupled the adjoint equations in a fashion similiar to that done to the primal flow equations. In this decoupled adjoint formulation, the conserved variables are split identically to flow equations, with the fully-coupled vector of conserved variables

$$\mathbf{U} = \begin{pmatrix} \rho_1 \\ \vdots \\ \rho_{ns} \\ \rho \mathbf{u} \\ \rho E \end{pmatrix} \tag{3.6}$$

split into

$$\mathbf{U}' = \begin{pmatrix} \rho \\ \rho \mathbf{u} \\ \rho E \end{pmatrix}, \qquad \hat{\mathbf{U}} = \begin{pmatrix} \rho_1 \\ \vdots \\ \rho_{ns} \end{pmatrix} \tag{3.7}$$

With this splitting, the mixture equations for single point in the global system of the decoupled flow solve can be written as

$$\frac{V}{\Delta t}\mathbf{I}+\begin{pmatrix} \frac{\partial \mathbf{R}_{\rho}}{\partial \rho} & \frac{\partial \mathbf{R}_{\rho}}{\partial \rho \mathbf{u}} & \frac{\partial \mathbf{R}_{\rho}}{\partial \rho E} \\ \frac{\partial \mathbf{R}_{\rho \mathbf{u}}}{\partial \rho} & \frac{\partial \mathbf{R}_{\rho \mathbf{u}}}{\partial \rho \mathbf{u}} & \frac{\partial \mathbf{R}_{\rho \mathbf{u}}}{\partial \rho E} \\ \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho} & \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho \mathbf{u}} & \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho E} \end{pmatrix}\begin{pmatrix} \Delta \rho \\ \Delta \rho \mathbf{u} \\ \Delta \rho E \end{pmatrix}=\begin{pmatrix} \mathbf{R}_{\rho} \\ \mathbf{R}_{\rho \mathbf{u}} \\ \mathbf{R}_{\rho E} \end{pmatrix} \tag{3.8}$$

Likewise, the species mass equations for a single point can be written as

$$\frac{V}{\Delta t}\mathbf{I}+\begin{pmatrix} \frac{\partial \mathbf{R}_{\rho_1}}{\partial c_1} & \cdots & \frac{\partial \mathbf{R}_{\rho_1}}{\partial c_{ns}} \\ \vdots & \ddots & \vdots \\ \frac{\partial \mathbf{R}_{\rho_{ns}}}{\partial c_1} & \cdots & \frac{\partial \mathbf{R}_{\rho_{ns}}}{\partial c_{ns}} \end{pmatrix}\begin{pmatrix} \Delta c_1 \\ \vdots \\ \Delta c_{ns} \end{pmatrix}=\begin{pmatrix} \mathbf{R}_{\rho_1} \\ \vdots \\ \mathbf{R}_{\rho_{ns}} \end{pmatrix} \tag{3.9}$$

Examining Eq.s (3.8-3.9) shows that there are clearly some physical dependencies being omitted, namely $\frac{\partial \mathbf{R}_{\rho_s}}{\partial \rho}$, $\frac{\partial \mathbf{R}_{\rho}}{\partial c_s}$, $\frac{\partial \mathbf{R}_{\rho \mathbf{u}}}{\partial c_s}$, and $\frac{\partial \mathbf{R}_{\rho E}}{\partial c_s}$. It has been found[1] that omitting this dependencies does not hinder convergence the primal flow solver; however, because the adjoint requires an exact linearization of the converged steady-state solution, these must be accounted for in the decoupled adjoint formulation.

The next step is to reconcile the split conserved variables, $\mathbf{U}'$ and $\hat{\mathbf{U}}$, with the conserved variable vector $\mathbf{Q}$ in the discrete adjoint formulation given in Eq. 3.3. The most intuitive and staightforward way to do this is to forgo solving for the species mass ρ_s in lieu of the species mass fraction c_s. Thus, $\mathbf{Q}$ can be expressed as

$$\mathbf{Q}=\begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E \\ c_1 \\ \vdots \\ c_{ns} \end{pmatrix} \tag{3.10}$$

This allows the linearizations in Eq.s (3.8-3.9) to be used in the adjoint formulation, by augmenting them with the previously omitted linearizations. Replacing $\frac{\partial \mathbf{R}}{\partial \mathbf{Q}}$ with the fully-coupled

system, the adjoint system becomes

$$\begin{pmatrix} \frac{\partial \mathbf{R}_\rho}{\partial \rho}^T & \frac{\partial \mathbf{R}_{\rho\mathbf{u}}}{\partial \rho}^T & \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho}^T & \frac{\partial \mathbf{R}_{\rho_s}}{\partial \rho}^T \\ \frac{\partial \mathbf{R}_\rho}{\partial \rho\mathbf{u}}^T & \frac{\partial \mathbf{R}_{\rho\mathbf{u}}}{\partial \rho\mathbf{u}}^T & \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho\mathbf{u}}^T & \frac{\partial \mathbf{R}_{\rho_s}}{\partial \rho\mathbf{u}}^T \\ \frac{\partial \mathbf{R}_\rho}{\partial \rho E}^T & \frac{\partial \mathbf{R}_{\rho\mathbf{u}}}{\partial \rho E}^T & \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho E}^T & \frac{\partial \mathbf{R}_{\rho_s}}{\partial \rho E}^T \\ \frac{\partial \mathbf{R}_\rho}{\partial c_s}^T & \frac{\partial \mathbf{R}_{\rho\mathbf{u}}}{\partial c_s}^T & \frac{\partial \mathbf{R}_{\rho E}}{\partial c_s}^T & \frac{\partial \mathbf{R}_{\rho_s}}{\partial c_s}^T \end{pmatrix} \begin{pmatrix} \Lambda_\rho \\ \Lambda_{\rho\mathbf{u}} \\ \Lambda_{\rho E} \\ \Lambda_{c_s} \end{pmatrix} = - \begin{pmatrix} \frac{\partial f}{\partial \rho} \\ \frac{\partial f}{\partial \rho\mathbf{u}} \\ \frac{\partial f}{\partial \rho E} \\ \frac{\partial f}{\partial c_s} \end{pmatrix} \tag{3.11}$$

Thus the jacobian in Eq. 3.11 is the completed one of Eq.s (3.8-3.9). While this is useful, the advantage of decoupling the species equations from the mixture equations was to speed up the linear solver and save memory. Solving Eq. 3.11 is roughly equivalent to solving the fully-coupled system of equations, which undermines both of these goals; so, an alternative solution strategy must be formulated. If a block jacobi scheme is employed, the system can be decoupled once again as

$$\begin{pmatrix} \frac{\partial \mathbf{R}_\rho}{\partial \rho}^T & \frac{\partial \mathbf{R}_{\rho\mathbf{u}}}{\partial \rho}^T & \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho}^T \\ \frac{\partial \mathbf{R}_\rho}{\partial \rho\mathbf{u}}^T & \frac{\partial \mathbf{R}_{\rho\mathbf{u}}}{\partial \rho\mathbf{u}}^T & \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho\mathbf{u}}^T \\ \frac{\partial \mathbf{R}_\rho}{\partial \rho E}^T & \frac{\partial \mathbf{R}_{\rho\mathbf{u}}}{\partial \rho E}^T & \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho E}^T \end{pmatrix} \begin{pmatrix} \Lambda_\rho \\ \Lambda_{\rho\mathbf{u}} \\ \Lambda_{\rho E} \end{pmatrix} = - \begin{pmatrix} \frac{\partial f}{\partial \rho} \\ \frac{\partial f}{\partial \rho\mathbf{u}} \\ \frac{\partial f}{\partial \rho E} \end{pmatrix} - \begin{pmatrix} \frac{\partial \mathbf{R}_{\rho_s}}{\partial \rho}^T \\ \frac{\partial \mathbf{R}_{\rho_s}}{\partial \rho\mathbf{u}}^T \\ \frac{\partial \mathbf{R}_{\rho_s}}{\partial \rho E}^T \end{pmatrix} \Lambda_{c_s} \tag{3.12}$$

$$\frac{\partial \mathbf{R}_{\rho_s}}{\partial c_s}^T \Lambda_{c_s} = -\frac{\partial f}{\partial c_s} - \frac{\partial \mathbf{R}_\rho}{\partial c_s}^T \Lambda_\rho - \frac{\partial \mathbf{R}_{\rho\mathbf{u}}}{\partial c_s}^T \Lambda_{\rho\mathbf{u}} - \frac{\partial \mathbf{R}_{\rho E}}{\partial c_s}^T \Lambda_{\rho E} \tag{3.13}$$

Adding a time-like derivative to the adjoint equations, the solution of the costate variables, Λ, can be time marched similar to the primal flow solver

$$\left[\frac{V}{\Delta t}\mathbf{I} + \frac{\partial \mathbf{R}}{\partial \mathbf{Q}}^T \right] \Delta\Lambda = -\frac{\partial f}{\partial \mathbf{Q}} - \frac{\partial \mathbf{R}}{\partial \mathbf{Q}}^T \Lambda \tag{3.14}$$

Thus, the first system of equations in Eq. 3.12 becomes

$$\left[\frac{V}{\Delta t}\mathbf{I} + \begin{pmatrix} \frac{\partial \mathbf{R}_\rho}{\partial \rho}^T & \frac{\partial \mathbf{R}_{\rho\mathbf{u}}}{\partial \rho}^T & \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho}^T \\ \frac{\partial \mathbf{R}_\rho}{\partial \rho\mathbf{u}}^T & \frac{\partial \mathbf{R}_{\rho\mathbf{u}}}{\partial \rho\mathbf{u}}^T & \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho\mathbf{u}}^T \\ \frac{\partial \mathbf{R}_\rho}{\partial \rho E}^T & \frac{\partial \mathbf{R}_{\rho\mathbf{u}}}{\partial \rho E}^T & \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho E}^T \end{pmatrix}\right] \begin{pmatrix} \Delta\Lambda_\rho \\ \Delta\Lambda_{\rho\mathbf{u}} \\ \Delta\Lambda_{\rho E} \end{pmatrix} = \\ - \begin{pmatrix} \frac{\partial f}{\partial \rho} \\ \frac{\partial f}{\partial \rho\mathbf{u}} \\ \frac{\partial f}{\partial \rho E} \end{pmatrix} - \begin{pmatrix} \frac{\partial \mathbf{R}_\rho}{\partial \rho}^T & \frac{\partial \mathbf{R}_{\rho\mathbf{u}}}{\partial \rho}^T & \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho}^T \\ \frac{\partial \mathbf{R}_\rho}{\partial \rho\mathbf{u}}^T & \frac{\partial \mathbf{R}_{\rho\mathbf{u}}}{\partial \rho\mathbf{u}}^T & \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho\mathbf{u}}^T \\ \frac{\partial \mathbf{R}_\rho}{\partial \rho E}^T & \frac{\partial \mathbf{R}_{\rho\mathbf{u}}}{\partial \rho E}^T & \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho E}^T \end{pmatrix} \begin{pmatrix} \Lambda_\rho \\ \Lambda_{\rho\mathbf{u}} \\ \Lambda_{\rho E} \end{pmatrix} - \begin{pmatrix} \frac{\partial \mathbf{R}_{\rho_s}}{\partial \rho}^T \\ \frac{\partial \mathbf{R}_{\rho_s}}{\partial \rho\mathbf{u}}^T \\ \frac{\partial \mathbf{R}_{\rho_s}}{\partial \rho E}^T \end{pmatrix} \Lambda_{c_s} \tag{3.15}$$

and the second system in Eq. 3.13 becomes

$$\left(\frac{V}{\Delta t}\mathbf{I} + \frac{\partial \mathbf{R}_{\rho_s}}{\partial c_s}^T\right) \Delta\Lambda_{c_s} = -\frac{\partial f}{\partial c_s} - \frac{\partial \mathbf{R}_{\rho_s}}{\partial c_s}^T \Lambda_{c_s} - \frac{\partial \mathbf{R}_\rho}{\partial c_s}^T \Lambda_\rho - \frac{\partial \mathbf{R}_{\rho\mathbf{u}}}{\partial c_s}^T \Lambda_{\rho\mathbf{u}} - \frac{\partial \mathbf{R}_{\rho E}}{\partial c_s}^T \Lambda_{\rho E} \tag{3.16}$$

The LHS of Eq.s (3.15-3.16) are the same first-order approximate jacobians that were used to solve the primal flow equations; therefore, all of the benefits of the diagonal block matricies that are exploited in the primal flow solver to reduce the linear solver cost and overall memory now apply to the adjoint.

Chapter 4

Design Optimization

Design optimization is a wide field that encompasses methods generally falling into two categories: local gradient-based optimization, and heuristic global optimization. Local gradient-based optimization techniques focus on the determining an optimality condition by evaluating a function and its gradients. Provided certain conditions are met, it can be proven that the optimization procedure will find a local minimum or maximum on a bounded domain. Examples of local gradient-based optimization methods include steepest-descent[cite??], conjugate-gradient[cite??], and Augmented Lagrangian Method [cite??]. A heuristic global optimization seeks to find the global extrema of a function. Although these methods are powerful, because of their heuristic nature they are not guaranteed to find the absolute optimum condition and are not the focus this research.

In the field of optimization, the function of interest is referred to as the "cost function" or "objective function". Optimization methods seek to minimize this function; therefore, if the intent is to find the maximum value of the function, it should be formulated as the negative of the original. This section focuses on the implementation of the cost function components and design variables used in the optimization, as well as the implementation of the optimizer that is used.

4.1 Cost Function Definition

The cost function (or objective function) as formulated in FUN3D is a composite, weighted function

$$f = \sum_{j=1}^{N_{func}} w_j \left(C_j - C_{j^*}\right)^{p_j} \tag{4.1}$$

Where w_j, C_{j^*}, and p_j are the weight, target, and power of cost function component j. C_j is the component value, which is evaluated at each flow solution. For this particular optimization

problem the cost function is defined as

$$f = w_1 \left(T_{RMS}\right)^2 + w_2 \left(C_D - C_D^*\right)^2 \tag{4.2}$$

The component weights were determined heuristically, to normalize the changes in drag coefficient, C_D, and surface temperature Root-Mean-Square (RMS) T_{RMS}. The drag coefficient is defined as

$$C_D = \sum_i^{N_{faces}} \frac{2\left(p_i - p_\infty\right) n_{x_i}}{\rho_\infty V_\infty S_{ref}} \tag{4.3}$$

where p_i is the average pressure at face i. RMS of surface temperature is defined as

$$\sqrt{\frac{\sum_i^{N_{faces}} \left(T_{RMS} A_i\right)^2}{\sum_i^{N_{faces}} \left(A_i\right)^2}} \tag{4.4}$$

The area-weighted RMS of surface temperature was chosen over a simple area-weighted average of surface temperature, because the stagnation temperature is generally much higher than temperature elsewhere on a vehicle forebody in hypersonic flows. The squaring of temperature in the RMS will therefore give greater weight to the stagnation temperature in the design.

REFERENCES

[1] Graham V. Candler, Pramod K. Subbareddy, and Ioannis Nompelis. Decoupled implicit method for aerothermodynamics and reacting flows. *AIAA Journal*, 51(5):1245–1254, 2015/04/23 2013.

[2] Paul F. Fischer. *Scaling Limits for PDE-Based Simulation (Invited)*. American Institute of Aeronautics and Astronautics, 2015/10/21 2015.

[3] Alan George and Joseph W. Liu. *Computer Solution of Large Sparse Positive Definite*. Prentice Hall Professional Technical Reference, 1981.

[4] Philip E. Gill, Elizabeth Wong, Walter Murray, and Michael Saunders. *User's Guide for NPSOL 5.0: A FORTRAN Package for Nonlinear Programming*, July 1998.

[5] Philip E. Gill, Elizabeth Wong, Walter Murray, and Michael Saunders. *User's Guide for SNOPT Version 7.4: Software Large-Scale Nonlinear Programming*, January 2015.

[6] R. N.; Gnoffo, P. A.; Gupta and J. L. Shinn. Conservation equations and physical models for hypersonic air flows in thermal and chemical nonequilibrium. Technical Paper 2867, NASA, 1989.

[7] Ami Harten. High resolution schemes for hyperbolic conservation laws. *Journal of Computational Physics*, 49(3):357–393, 1983.

[8] P.W. Jansen and R.E. Perez. Constrained structural design optimization via a parallel augmented lagrangian particle swarm optimization approach. *Computers & Structures*, 89(13–14):1352 – 1366, 2011.

[9] Robert W. MacCormack and Graham V. Candler. The solution of the navier-stokes equations using gauss-seidel line relaxation. *Computers and Fluids*, 17(1):135–150, 1989.

[10] Hiroaki Nishikawa. Implementing a real-gas roe solver into implementing a real-gas roe solver into cfl3d, April 2004.

[11] Chul Park. *Assessment of two-temperature kinetic model for ionizing air*. American Institute of Aeronautics and Astronautics, 2015/10/23 1987.

[12] P.L Roe. Approximate riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics*, 43(2):357 – 372, 1981.

[13] Y. Saad. *Iterative Methods for Sparse Linear Systems*, pages 391–392. Society for Industrial and Applied Mathematics, 2003.

[14] Jian-Shun Shuen, Meng-Sing Liou, and Bram Van Leer. Inviscid flux-splitting algorithms for real gases with non-equilibrium chemistry. *Journal of Computational Physics*, 90(2):371–395, 1990.

APPENDIX

Appendix A

Derivations

A.1 Decoupled Flux Derivation

For the Roe flux difference splitting scheme, the species mass fluxes are given by

$$F_{\rho_s} = \frac{\rho_s^L \overline{U}^L + \rho_s^R \overline{U}^R}{2} - \frac{\tilde{c}_s(\lambda_1 dv_1 + \lambda_2 dv_2) + \lambda_3 dv_{3_s}}{2} \tag{A.1}$$

$$dv_1 = \frac{p^R - p^L + \tilde{\rho}\tilde{a}(\overline{U}^R - \overline{U}^L)}{\tilde{a}^2} \tag{A.2}$$

$$dv_2 = \frac{p^R - p^L - \tilde{\rho}\tilde{a}(\overline{U}^R - \overline{U}^L)}{\tilde{a}^2} \tag{A.3}$$

$$dv_{3_s} = \frac{\tilde{a}^2(\rho_s^R - \rho_s^L) - \tilde{c}_s(p^R - p^L)}{\tilde{a}^2} \tag{A.4}$$

$$\lambda_1 = \mid \overline{\mathbf{U}} + \tilde{a} \mid, \quad \lambda_2 = \mid \overline{\mathbf{U}} - \tilde{a} \mid, \quad \lambda_3 = \mid \overline{\mathbf{U}} \mid \tag{A.5}$$

where the ˜ notation signifies a Roe-averaged quantity, given by:

$$\tilde{\mathbf{U}} = w\tilde{\mathbf{U}}^L + (1-w)\tilde{\mathbf{U}}^R \tag{A.6}$$

$$w = \frac{\tilde{\rho}}{\tilde{\rho} + \rho^R} \tag{A.7}$$

$$\tilde{\rho} = \sqrt{\rho^R \rho^L} \tag{A.8}$$

The species mass fluxes must sum to the total mass flux; thus, the total mixture mass flux is given as

$$F_\rho = \sum_s F_{\rho_s} = \frac{\rho^L \overline{U}^L + \rho^R \overline{U}^R}{2} - \frac{\tilde{c}_s(\lambda_1 dv_1 + \lambda_2 dv_2) + \lambda_3 dv_3}{2} \tag{A.9}$$

$$dv_3 = \frac{\tilde{a}^2(\rho^R - \rho^L) - (p^R - p^L)}{\tilde{a}^2} \tag{A.10}$$

Multiplying Eq. (A.9) by the Roe-averaged mass fraction and substituting it into Eq. (A.1) results in:

$$F_{\rho_s} = \tilde{c}_s F_\rho + \frac{(c_s^L - \tilde{c}_s)\rho^L(\overline{U}^L + \mid \tilde{U} \mid)}{2} + \frac{(c_s^R - \tilde{c}_s)\rho^R(\overline{U}^R - \mid \tilde{U} \mid)}{2} \tag{A.11}$$

It should be noted here that the Roe-averaged normal velocity, $\tilde{U}$, requires an entropy correction in the presence of strong shocks[7]. This correction has no dependence on the species mass fractions; therefore, it does not change the form of the Jacobian for this decoupled scheme. The notation can be further simplified by defining the normal velocities as follows:

$$\lambda^+ = \frac{\overline{U}^L + \mid \tilde{U} \mid}{2}, \quad \lambda^- = \frac{\overline{U}^R - \mid \tilde{U} \mid}{2} \tag{A.12}$$

Finally, substituting Eq. (A.12) into Eq. (A.11) yields the final result for calculating the species flux in the decoupled system:

$$F_{\rho_s} = \tilde{c}_s F_\rho + (c_s^L - \tilde{c}_s)\rho^L \lambda^+ + (c_s^R - \tilde{c}_s)\rho^R \lambda^- \tag{A.13}$$

Forming the convective contributions to the Jacobians is straightforward. Because the $\mathbf{U}'$ level variables are constant, only the left, right, and Roe-averaged state mass fractions vary. Differentiating Eq. (A.13) with respect to the mass fraction, c_s, the left and right state contributions are

$$\frac{\partial F_{\rho_s}}{\partial c_s^L} = wF_\rho + (1-w)\rho^L \lambda^+ - w\rho^R \lambda^- \tag{A.14}$$

$$\frac{\partial F_{\rho_s}}{\partial c_s^R} = (1-w)F_\rho + (w-1)\rho^L \lambda^+ + w\rho^R \lambda^- \tag{A.15}$$

Because there is no dependence between species in decoupled convective formulation, the Jacobian block elements are purely diagonal for the convective contributions, of the form

$$\begin{pmatrix} \frac{\partial F_{\rho_1}}{\partial c_1} & & 0 \\ & \ddots & \\ 0 & & \frac{\partial F_{\rho_{ns}}}{\partial c_{ns}} \end{pmatrix} \tag{A.16}$$

A.2 Quadratic Interpolation Between Thermodynamic Curve Fits

We seek to blend the two thermodynamic curve fits in such a way that we maintain c_0 continuity in both specific heat (C_p) and enthalpy (h). To accomplish this, a quadratic function must be used, of the form

$$aT^2 + bT + c = C_p \tag{A.17}$$

The coefficients a, b, and c are determined by solving the system that results from the boundary value problem

$$\begin{cases} a{T_1}^2 + bT_1 + c = C_{p_1} \\ a{T_2}^2 + bT_2 + c = C_{p_2} \\ a\frac{\left({T_2}^3 - {T_1}^3\right)}{3} + b\frac{\left({T_2}^2 - {T_1}^2\right)}{2} + c\left(T_2 - T_1\right) = h_2 - h_1 \end{cases} \tag{A.18}$$

Where the x_1 and x_2 subscripts describe the left and right states, respectively. Solving the linear system, the coefficients are

$$\begin{cases} fa = \frac{3\left(C_{p_2} + C_{p_1}\right)}{(T_2 - T_1)^2} - \frac{6(h_2 - h_1)}{(T_2 - T_1)^3} \\ b = -\frac{2\left[(C_{p_2} + 2C_{p_1})T_2 + (2C_{p_2} + C_{p_1})T_1\right]}{(T_2 - T_1)^2} + \frac{6(T_2 + T_1)(h_2 - h_1)}{(T_2 - T_1)^3} \\ c = \frac{C_{p_1}T_2(T_2 + 2T_1) + C_{p_2}T_1(T_1 + 2T_2)}{(T_2 - T_1)^2} - \frac{6T_1T_2(h_2 - h_1)}{(T_2 - T_1)^3} \end{cases} \tag{A.19}$$

This can be simplified to

$$\begin{cases} a = 3B - A \\ b = \frac{-2(C_{p_1}T_2 + C_{p_2}T_2)}{(T_2 - T_1)^2} + (T_2 + T_1)(A - 2B) \\ c = \frac{C_{p_1}{T_2}^2 + C_{p_2}{T_1}^2}{(T_2 - T_1)^2} + T_1T_2(2B - A) \end{cases} \tag{A.20}$$

$$A = \frac{6(h_2 - h_1)}{(T_2 - T_1)^3} \tag{A.21}$$

$$B = \frac{C_{p_2} + C_{p_1}}{(T_2 - T_1)^2} \tag{A.22}$$

Note that this does not ensure that entropy will be continuous across curve fits.