ABSTRACT

THOMPSON, KYLE B. A North Carolina State University Sample LaTeX Thesis with a Title So Long it Needs a Line Break. (Under the direction of Hassan Hassan and Peter Gnoffo.)

This is my abstract.

A North Carolina State University Sample LaTeX Thesis with a Title So Long it Needs a Line Break

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 in a small town . . .

ACKNOWLEDGEMENTS

I would like to thank my advisor for his help.

TABLE OF CONTENTS

List of Tables . . . . **vi**

List of Figures . . . . **vii**

Chapter 1 Introduction **1**
1.1 Flow Solver Decoupled Approach 1
1.1.1 Introduction 1
1.1.2 Background: Fully-Coupled Point Implicit Method 2
1.1.3 Background: Decoupled Point Implicit Method 3
1.1.4 Cost and Memory Savings of the Decoupled Implicit Problem 5

Chapter 2 Adjoint Solver **7**
2.1 Adjoint Derivation 7
2.2 Block Jacobi Adjoint Decoupling 8

Chapter 3 Design Optimization **12**
3.1 Introduction 12
3.2 Cost Function Definition 12

References **14**

Appendix **15**
Appendix A Derivations 16
A.1 Decoupled Flux Derivation 16

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Flow Solver Decoupled Approach

1.1.1 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[6], whereas the work presented here is derived for the Roe flux difference splitting (FDS) scheme[9].

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[7], and a weighting scheme can be used to correct the non-uniqueness of the pressure linearization[11]. 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.

1.1.2 Background: 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:

$$\frac{\partial \mathbf{U}}{\partial t} + \nabla \cdot \mathbf{F} = \mathbf{W} \tag{1.1}$$

or, in semi-discrete form,

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{1}{V}\sum_{f}(\mathbf{F} \cdot \mathbf{S})^{f} = \mathbf{W} \tag{1.2}$$

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:

$$\mathbf{U} = \begin{pmatrix} \rho_1 \\ \vdots \\ \rho_{ns} \\ \rho u \\ \rho v \\ \rho w \\ \rho E \end{pmatrix}, \quad \mathbf{F} = \begin{pmatrix} \rho_1 \overline{U} \\ \vdots \\ \rho_{ns} \overline{U} \\ \rho u \overline{U} + p s_x \\ \rho u \overline{U} + p s_y \\ \rho u \overline{U} + p s_z \\ (\rho E + p)\overline{U} \end{pmatrix} \tag{1.3}$$

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

$$E = \sum c_s e_s + \frac{u^2 + v^2 + w^2}{2} \tag{1.4}$$

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

$$\begin{aligned} \mathbf{F}^{n+1} &\approx \mathbf{F}^{n} + \tfrac{\partial \mathbf{F}}{\partial \mathbf{U}} \delta \mathbf{U}^{n} \\ \mathbf{W}^{n+1} &\approx \mathbf{W}^{n} + \tfrac{\partial \mathbf{W}}{\partial \mathbf{U}} \delta \mathbf{U}^{n} \end{aligned} \tag{1.5}$$

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

$$\frac{\delta \mathbf{U}^n}{\Delta t} + \frac{1}{V}\sum_{f}(\frac{\partial \mathbf{F}^f}{\partial \mathbf{U^L}}\delta \mathbf{U}^L + \frac{\partial \mathbf{F}^f}{\partial \mathbf{U^R}}\delta \mathbf{U}^R)^n \mathbf{S}^f - \frac{\partial \mathbf{W}}{\partial \mathbf{U}}\delta \mathbf{U}^n = -\frac{1}{V}\sum_{f}(\mathbf{F}^f \cdot \mathbf{S}^f)^n + \mathbf{W}^n \tag{1.6}$$

or, put more simply:

$$A\delta\mathbf{U}^n = \mathbf{b} \tag{1.7}$$

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:

$$A = O + D \tag{1.8}$$

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[10], 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 $M = ns$.

1.1.3 Background: 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

$$\mathbf{U}' = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E \end{pmatrix} \quad \hat{\mathbf{U}} = \begin{pmatrix} \rho_1 \\ \vdots \\ \rho_{ns} \end{pmatrix} \tag{1.9}$$

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

$$\frac{\partial \mathbf{U}'}{\partial t} + \frac{1}{V}\sum_f (\mathbf{F}'\cdot\mathbf{S})^f = 0 \tag{1.10}$$

followed by the species fluxes as

$$\frac{\partial \hat{\mathbf{U}}}{\partial t} + \frac{1}{V}\sum_f (\hat{\mathbf{F}}\cdot\mathbf{S})^f = \hat{\mathbf{W}} \tag{1.11}$$

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[8].

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:

$$\delta\hat{\mathbf{U}}^n = \rho^{n+1}\hat{\mathbf{V}}^{n+1} - \rho^n\hat{\mathbf{V}}^n = \rho^{n+1}\delta\hat{\mathbf{V}}^n + \hat{\mathbf{V}}^n\delta\rho^n \tag{1.12}$$

where $\hat{\mathbf{V}} = (c_1, \dots, c_{ns})^T$, 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:

$$\hat{F}_{\rho_s} = c_s F'_\rho + (c_s^L - \tilde{c}_s)\rho^L\lambda^+ + (c_s^R - \tilde{c}_s)\rho^R\lambda^- \tag{1.13}$$

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

$$\hat{\mathbf{F}}^{n+1} = \hat{\mathbf{F}}^n + \frac{\partial\hat{\mathbf{F}}}{\partial\hat{\mathbf{V}}^L}\delta\hat{\mathbf{V}}^L + \frac{\partial\hat{\mathbf{F}}}{\partial\hat{\mathbf{V}}^R}\delta\hat{\mathbf{V}}^R \tag{1.14}$$

$$\frac{\partial\hat{\mathbf{F}}}{\partial\hat{\mathbf{V}}^L} = wF_\rho + (1-w)\rho^L\lambda^+ - w\rho^R\lambda^- \tag{1.15}$$

$$\frac{\partial\hat{\mathbf{F}}}{\partial\hat{\mathbf{V}}^R} = (1-w)F_\rho + (w-1)\rho^L\lambda^+ + w\rho^R\lambda^- \tag{1.16}$$

A full derivation of Eqs. (1.13-1.16), 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:

$$\hat{\mathbf{W}}^{n+1} = \hat{\mathbf{W}}^n + \left.\frac{\partial\hat{\mathbf{W}}}{\partial\mathbf{U}}\right|_{\mathbf{U}'}\frac{\partial\mathbf{U}}{\partial\hat{\mathbf{V}}} \tag{1.17}$$

For simplicity of notation, we define

$$C = \left.\frac{\partial\hat{\mathbf{W}}}{\partial\mathbf{U}}\right|_{\mathbf{U}'}\frac{\partial\mathbf{U}}{\partial\hat{\mathbf{V}}} \tag{1.18}$$

The decoupled system to be solved becomes:

$$\begin{aligned}
&\rho^{n+1}\frac{\delta\hat{\mathbf{V}}^n}{\Delta t} + \frac{1}{V}\sum_f\left(\frac{\partial\hat{\mathbf{F}}^f}{\partial\hat{\mathbf{V}}^L}\delta\hat{\mathbf{V}}^L + \frac{\partial\hat{\mathbf{F}}^f}{\partial\hat{\mathbf{V}}^R}\delta\hat{\mathbf{V}}^R\right)^{n,n+1}\mathbf{S}^f - C^{n,n+1}\delta\mathbf{V}^n \\
&= -\frac{1}{V}\sum_f(\hat{\mathbf{F}}^{n,n+1}\cdot\mathbf{S})^f + \mathbf{W}^{n,n+1} - \hat{\mathbf{V}}^n\frac{\delta\rho^n}{\Delta t} - R_\rho
\end{aligned} \tag{1.19}$$

$$R_\rho = -\frac{1}{V}\sum_f\sum_s(\hat{F}^{n,n+1}_{\rho_s}\cdot\mathbf{S}) \tag{1.20}$$

where R_ρ is included to preserve the constraint that the mass fractions sum to unity, i.e., $\sum\limits_s c_s = 1$, $\sum\limits_s \delta c_s = 0$.

1.1.4 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[4]. 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. (1.8), we decompose A_d into its diagonal and off-diagonal elements, resulting in the following linear system:

$$\begin{pmatrix} \square & & & & \\ & \ddots & & & \\ & & \square & & \\ & & & \ddots & \\ & & & & \square \end{pmatrix}\begin{pmatrix} \delta\hat{\mathbf{V}}_1 \\ \vdots \\ \delta\hat{\mathbf{V}}_i \\ \vdots \\ \delta\hat{\mathbf{V}}_{nodes}\end{pmatrix} = \begin{pmatrix} \hat{b}_1 \\ \vdots \\ \hat{b}_i \\ \vdots \\ \hat{b}_{nodes}\end{pmatrix} - \begin{pmatrix} (\sum_{j=1}^{N_{nb}}[\diagdown]\delta\hat{\mathbf{V}}_j)_1 \\ \vdots \\ (\sum_{j=1}^{N_{nb}}[\diagdown]\delta\hat{\mathbf{V}}_j)_i \\ \vdots \\ (\sum_{j=1}^{N_{nb}}[\diagdown]\delta\hat{\mathbf{V}}_j)_{nodes}\end{pmatrix} \tag{1.21}$$

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

$$\begin{aligned} Relative\ Memory\ Cost &= \frac{size(A_d)}{size(A)} \\ &= \lim_{ns\to\infty} \frac{(ns^2+5^2)(N_{nodes}) + (ns+5^2)(N_{nz})}{(ns+4)^2(N_{nodes}+N_{nz})} \\ &= \frac{N_{nodes}}{N_{nodes}+N_{nz}} \end{aligned} \tag{1.22}$$

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 2

Adjoint Solver

2.1 Adjoint Derivation

The FUN3D adjoint derivation is given in Eric Nielson's PhD Thesis[cite??]. Starting with forming the Lagrangian as

$$L(\mathbf{D}, \mathbf{Q}, \mathbf{X}, \mathbf{\Lambda}) = f(\mathbf{D}, \mathbf{Q}, \mathbf{X}) + \mathbf{\Lambda}^T \mathbf{R}(\mathbf{D}, \mathbf{Q}, \mathbf{X}) \tag{2.1}$$

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

$$\frac{\partial L}{\partial \mathbf{D}} = \left\{ \frac{\partial f}{\partial \mathbf{D}} + \left[\frac{\partial \mathbf{X}}{\partial \mathbf{D}}\right]^T \frac{\partial f}{\partial \mathbf{X}} \right\} + \left[\frac{\partial \mathbf{Q}}{\partial \mathbf{D}}\right]^T \left\{ \frac{\partial f}{\partial \mathbf{Q}} + \left[\frac{\partial \mathbf{R}}{\partial \mathbf{Q}}\right]^T \mathbf{\Lambda} \right\} + \left\{ \left[\frac{\partial \mathbf{R}}{\partial \mathbf{D}}\right]^T + \left[\frac{\partial \mathbf{X}}{\partial \mathbf{D}}\right]^T \left[\frac{\partial \mathbf{R}}{\partial \mathbf{X}}\right]^T \right\} \mathbf{\Lambda} \tag{2.2}$$

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

$$\left[\frac{\partial \mathbf{R}}{\partial \mathbf{Q}}\right]^T \mathbf{\Lambda} = -\frac{\partial f}{\partial \mathbf{Q}} \tag{2.3}$$

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

$$\mathbf{\Lambda} = -\frac{\partial f}{\partial \mathbf{R}} \tag{2.4}$$

This can ultimately be used to error estimation and sensitivity analysis for design optimization. With the second term in eq. (2.2) eliminated, the derivative of the Lagrangian becomes

$$\frac{\partial L}{\partial \mathbf{D}} = \left\{ \frac{\partial f}{\partial \mathbf{D}} + \left[\frac{\partial \mathbf{X}}{\partial \mathbf{D}}\right]^T \frac{\partial f}{\partial \mathbf{X}} \right\} + \left\{ \left[\frac{\partial \mathbf{R}}{\partial \mathbf{D}}\right]^T + \left[\frac{\partial \mathbf{X}}{\partial \mathbf{D}}\right]^T \left[\frac{\partial \mathbf{R}}{\partial \mathbf{X}}\right]^T \right\} \mathbf{\Lambda} \tag{2.5}$$

By solving the adjoint equation in Eq. 2.3) 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 **PORT** or **KSOPT** in FUN3D, as well as a host of other non-linear optimizers.

2.2 Block Jacobi Adjoint Decoupling

The primal flow equations are solved using the decoupled scheme based on the work by Candler, et. al. In doing this the conserved variables are split from

$$\mathbf{U} = \begin{pmatrix} \rho_1 \\ \vdots \\ \rho_{ns} \\ \rho u \\ \rho v \\ \rho w \\ \rho E \end{pmatrix} \tag{2.6}$$

into

$$\mathbf{U}' = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E \end{pmatrix}, \qquad \hat{\mathbf{U}} = \begin{pmatrix} \rho_1 \\ \vdots \\ \rho_{ns} \end{pmatrix} \tag{2.7}$$

By doing this, the flow equations can be solved implicitly using an approximate, first-order jacobian to drive the flow solution to a converged steady-state. In practice, this is done by essentially formulating

$$\frac{V}{\Delta t}\mathbf{I} + \begin{pmatrix} \frac{\partial \mathbf{R}_\rho}{\partial \rho} & \frac{\partial \mathbf{R}_\rho}{\partial \rho \mathbf{u}} & \frac{\partial \mathbf{R}_\rho}{\partial \rho E} \\ \frac{\partial \mathbf{R}_{\rho \mathbf{u}}}{\partial \rho} & \frac{\partial \mathbf{R}_{\rho \mathbf{u}}}{\partial \rho \mathbf{u}} & \frac{\partial \mathbf{R}_{\rho \mathbf{u}}}{\partial \rho E} \\ \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho} & \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho \mathbf{u}} & \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho E} \end{pmatrix} \begin{pmatrix} \Delta \rho \\ \Delta \rho \mathbf{u} \\ \Delta \rho E \end{pmatrix} = \begin{pmatrix} \mathbf{R}_\rho \\ \mathbf{R}_{\rho \mathbf{u}} \\ \mathbf{R}_{\rho E} \end{pmatrix} \tag{2.8}$$

to solve for the mixture flow variables, and formulating

$$\frac{V}{\Delta t}\mathbf{I}+\begin{pmatrix}\frac{\partial \mathbf{R}_{\rho_1}}{\partial c_1} & \cdots & \frac{\partial \mathbf{R}_{\rho_1}}{\partial c_{ns}} \\ \vdots & \ddots & \vdots \\ \frac{\partial \mathbf{R}_{\rho_{ns}}}{\partial c_1} & \cdots & \frac{\partial \mathbf{R}_{\rho_{ns}}}{\partial c_{ns}}\end{pmatrix}\begin{pmatrix}\Delta c_1 \\ \vdots \\ \Delta c_{ns}\end{pmatrix}=\begin{pmatrix}\mathbf{R}_{\rho_1} \\ \vdots \\ \mathbf{R}_{\rho_{ns}}\end{pmatrix} \tag{2.9}$$

Note that the correction terms required to maintain $\sum_{s=1}^{ns} c_s = 1$ and $\sum_{s=1}^{ns} \delta c_s = 0$ are omitted in Eq. 2.9 only for brevity in this explanation. Examining Eq.s (2.8-2.9) 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 demonstrated that omitting this dependencies does not hinder convergence the primal solver; however, the adjoint requires an exact linearization of the converged steady-state solution.

The discrete adjoint formulation is given as

$$\left[\frac{\partial \mathbf{R}}{\partial \mathbf{Q}}\right]^T \mathbf{\Lambda} = -\frac{\partial f}{\partial \mathbf{Q}} \tag{2.10}$$

where $\mathbf{Q}$ is a vector of conserved variables, and f is the cost function (i.e. lift, drag, etc.). There is now an apparent need to reconcile the two sets of conserved variables in Eq. 2.7 with $\mathbf{Q}$. 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

$$\mathbf{Q}=\begin{pmatrix}\rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E \\ c_1 \\ \vdots \\ c_{ns}\end{pmatrix} \tag{2.11}$$

This allows the linearizations derived in Eq.s (2.8-2.9) 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 system, the adjoint system becomes

$$\begin{pmatrix} \frac{\partial \mathbf{R}_\rho}{\partial \rho}^T & \frac{\partial \mathbf{R}_\rho}{\partial \rho \mathbf{u}}^T & \frac{\partial \mathbf{R}_\rho}{\partial \rho E}^T & \frac{\partial \mathbf{R}_\rho}{\partial c_s}^T \\ \frac{\partial \mathbf{R}_{\rho \mathbf{u}}}{\partial \rho}^T & \frac{\partial \mathbf{R}_{\rho \mathbf{u}}}{\partial \rho \mathbf{u}}^T & \frac{\partial \mathbf{R}_{\rho \mathbf{u}}}{\partial \rho E}^T & \frac{\partial \mathbf{R}_{\rho \mathbf{u}}}{\partial c_s}^T \\ \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho}^T & \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho \mathbf{u}}^T & \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho E}^T & \frac{\partial \mathbf{R}_{\rho E}}{\partial c_s}^T \\ \frac{\partial \mathbf{R}_{\rho_s}}{\partial \rho}^T & \frac{\partial \mathbf{R}_{\rho_s}}{\partial \rho \mathbf{u}}^T & \frac{\partial \mathbf{R}_{\rho_s}}{\partial \rho E}^T & \frac{\partial \mathbf{R}_{\rho_s}}{\partial c_s}^T \end{pmatrix} \begin{pmatrix} \Lambda_\rho \\ \Lambda_{\rho \mathbf{u}} \\ \Lambda_{\rho E} \\ \Lambda_{c_s} \end{pmatrix} = - \begin{pmatrix} \frac{\partial f}{\partial \rho} \\ \frac{\partial f}{\partial \rho \mathbf{u}} \\ \frac{\partial f}{\partial \rho E} \\ \frac{\partial f}{\partial c_s} \end{pmatrix} \tag{2.12}$$

Thus the jacobian in Eq. 2.12 is the completed one of Eq.s (2.8-2.9). While this is useful, the point of decoupling the species equations from the mixture equations was to speed up the linear solver and save memory. Solving Eq. 2.12 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

$$\begin{pmatrix} \frac{\partial \mathbf{R}_\rho}{\partial \rho}^T & \frac{\partial \mathbf{R}_\rho}{\partial \rho \mathbf{u}}^T & \frac{\partial \mathbf{R}_\rho}{\partial \rho E}^T \\ \frac{\partial \mathbf{R}_{\rho \mathbf{u}}}{\partial \rho}^T & \frac{\partial \mathbf{R}_{\rho \mathbf{u}}}{\partial \rho \mathbf{u}}^T & \frac{\partial \mathbf{R}_{\rho \mathbf{u}}}{\partial \rho E}^T \\ \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho}^T & \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho \mathbf{u}}^T & \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho E}^T \end{pmatrix} \begin{pmatrix} \Lambda_\rho \\ \Lambda_{\rho \mathbf{u}} \\ \Lambda_{\rho E} \end{pmatrix} = - \begin{pmatrix} \frac{\partial f}{\partial \rho} \\ \frac{\partial f}{\partial \rho \mathbf{u}} \\ \frac{\partial f}{\partial \rho E} \end{pmatrix} - \begin{pmatrix} \frac{\partial \mathbf{R}_\rho}{\partial c_s}^T \\ \frac{\partial \mathbf{R}_{\rho \mathbf{u}}}{\partial c_s}^T \\ \frac{\partial \mathbf{R}_{\rho E}}{\partial c_s}^T \end{pmatrix} \Lambda_{c_s} \tag{2.13}$$

$$\frac{\partial \mathbf{R}_{\rho_s}}{\partial c_s}^T \Lambda_{c_s} = -\frac{\partial f}{\partial c_s} - \frac{\partial \mathbf{R}_{\rho_s}}{\partial \rho}^T \Lambda_\rho - \frac{\partial \mathbf{R}_{\rho_s}}{\partial \rho \mathbf{u}}^T \Lambda_{\rho \mathbf{u}} - \frac{\partial \mathbf{R}_{\rho_s}}{\partial \rho E}^T \Lambda_{\rho E} \tag{2.14}$$

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

$$\left[\frac{V}{\Delta t}\mathbf{I} + \frac{\partial \mathbf{R}}{\partial \mathbf{Q}}^T \right] \Delta\Lambda = -\frac{\partial f}{\partial \mathbf{Q}} - \frac{\partial \mathbf{R}}{\partial \mathbf{Q}}^T \Lambda \tag{2.15}$$

Thus, the first system in Eq. 2.13 becomes

$$\left[\frac{V}{\Delta t}\mathbf{I} + \begin{pmatrix} \frac{\partial \mathbf{R}_\rho}{\partial \rho}^T & \frac{\partial \mathbf{R}_\rho}{\partial \rho\mathbf{u}}^T & \frac{\partial \mathbf{R}_\rho}{\partial \rho E}^T \\ \frac{\partial \mathbf{R}_{\rho\mathbf{u}}}{\partial \rho}^T & \frac{\partial \mathbf{R}_{\rho\mathbf{u}}}{\partial \rho\mathbf{u}}^T & \frac{\partial \mathbf{R}_{\rho\mathbf{u}}}{\partial \rho E}^T \\ \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho}^T & \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho\mathbf{u}}^T & \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho E}^T \end{pmatrix}\right] \begin{pmatrix} \Delta\Lambda_\rho \\ \Delta\Lambda_{\rho\mathbf{u}} \\ \Delta\Lambda_{\rho E} \end{pmatrix} = \\ - \begin{pmatrix} \frac{\partial f}{\partial \rho} \\ \frac{\partial f}{\partial \rho\mathbf{u}} \\ \frac{\partial f}{\partial \rho E} \end{pmatrix} - \begin{pmatrix} \frac{\partial \mathbf{R}_\rho}{\partial \rho}^T & \frac{\partial \mathbf{R}_\rho}{\partial \rho\mathbf{u}}^T & \frac{\partial \mathbf{R}_\rho}{\partial \rho E}^T \\ \frac{\partial \mathbf{R}_{\rho\mathbf{u}}}{\partial \rho}^T & \frac{\partial \mathbf{R}_{\rho\mathbf{u}}}{\partial \rho\mathbf{u}}^T & \frac{\partial \mathbf{R}_{\rho\mathbf{u}}}{\partial \rho E}^T \\ \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho}^T & \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho\mathbf{u}}^T & \frac{\partial \mathbf{R}_{\rho E}}{\partial \rho E}^T \end{pmatrix} \begin{pmatrix} \Lambda_\rho \\ \Lambda_{\rho\mathbf{u}} \\ \Lambda_{\rho E} \end{pmatrix} - \begin{pmatrix} \frac{\partial \mathbf{R}_\rho}{\partial c_s}^T \\ \frac{\partial \mathbf{R}_{\rho\mathbf{u}}}{\partial c_s}^T \\ \frac{\partial \mathbf{R}_{\rho E}}{\partial c_s}^T \end{pmatrix} \Lambda_{c_s} \tag{2.16}$$

and the second system in Eq. 2.14 becomes

$$\left(\frac{V}{\Delta t}\mathbf{I} + \frac{\partial \mathbf{R}_{\rho_s}}{\partial c_s}^T\right)\Delta\Lambda_{c_s} = -\frac{\partial f}{\partial c_s} - \frac{\partial \mathbf{R}_{\rho_s}}{\partial c_s}^T\Lambda_{c_s} - \frac{\partial \mathbf{R}_{\rho_s}}{\partial \rho}^T\Lambda_\rho - \frac{\partial \mathbf{R}_{\rho_s}}{\partial \rho\mathbf{u}}^T\Lambda_{\rho\mathbf{u}} - \frac{\partial \mathbf{R}_{\rho_s}}{\partial \rho E}^T\Lambda_{\rho E} \tag{2.17}$$

This is advantageous, because the jacobians on the left hand side (LHS) can be the first-order approximate jacobians that were used to solve the primal flow equations; hence, all of the benefits of the diagonal block matricies that are exploited to reduce the linear solver cost and overall memory now apply to the adjoint.

Chapter 3

Design Optimization

3.1 Introduction

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.

3.2 Cost Function Definition

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

$$f = \sum_{j=1}^{N_{func}} w_j \left(C_j - C_{j^*}\right)^{p_j} \tag{3.1}$$

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. Examples of component values include integrated aerodynamic or aero-thermodynamic quantities.

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] 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.

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

[6] 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.

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

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

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

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

[11] 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 $\tilde{}$ 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[5]. 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

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