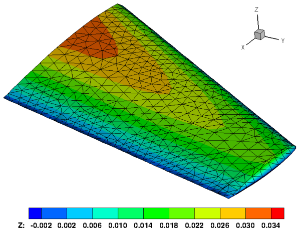
# Aerothermodynamic Design Sensitivities for a Reacting Gas Flow Solver on an Unstructured Mesh Using a Discrete Adjoint Formulation

Kyle B. Thompson

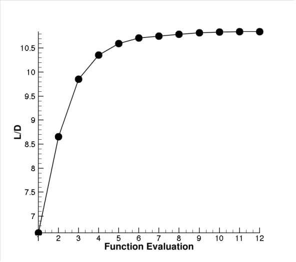
April 15, 2016

# Outline

# Introduction and Motivation




Shape Design Mesh Movement




Lift/Drag Objective Function

- Exploring design space using high-fidelity CFD is challenging
- zero-order methods (sampling) are prohibitively expensive
- Need to be intelligent about techniques for evaluating sensitivity to design parameters
- Gradient-based optimization much more efficient than sampling, but requires calculating sensitivity derivatives

# Introduction and Motivation - Design

- How to compute sensitivity of many design variables?
- Direct differentiation approach
  - Navier-Stokes equations can be directly differentiated to yield sensitivity derivatives necessary for gradient-based optimization
  - Finite difference requires a minimum of one flow solution for each design variable sensitivity
  - Prohibitively expensive for large number of design parameters
- Adjoint approach
  - Solve adjoint equations in addition to Navier Stokes flow equations to obtain sensitivity derivatives
  - One flow and adjoint solution needed for each cost function, regardless of number of design variables
  - Considerably more efficient than direct differentiation approach for large number of design parameters

# Introduction and Motivation - Design

- Adjoint-based design optimization has recieved considerable attention in compressible, perfect gas CFD solvers, but very little in reacting flow solvers
- Difficulty of adjoint approach lies in implementating exact linearizations for 2nd-order flux construction scheme
- Particularly difficult for reacting flows, due to
  - complexity of linearizing the additional equations for multi-species chemical kinetics
  - Serious memory and computational cost concerns
- Decoupling equations can significantly mitigate cost and memory overhead for large number of conservation equations

# Introduction and Motivation - Decoupled Approach

- Reacting gas simulations require solving a large number of conservation equations
- Memory concerns
  - Size of Jacobians scales quadratically with number species in gas mixture
  - Solving system of equations in a tightly-coupled fashion can be limited by memory constraints
- Cost concerns
  - Cost of solving the linear system scales quadratically with number of species in gas mixture
- Efficiently solving adjoint problem is a primary motivator
  - Solving adjoint system particularly costly if linear solver is slow
  - Can be necessary to store jacobian twice $\rightarrow$ large memory overhead

# Introduction and Motivation - Decoupled Approach

- Loosely-coupled solvers have become popular in the combustion community.
    - Decouple species conservation equations from meanflow equations, and solve two smaller systems

$$\underset{(4+ns)\times(4+ns)}{\begin{pmatrix} \square & \square & \cdots & \square \\ \square & \square & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \square & \cdots & \cdots & \square \end{pmatrix}} \rightarrow \underset{5\times5}{\begin{pmatrix} \square & \cdots & \square \\ \vdots & \ddots & \vdots \\ \square & \cdots & \square \end{pmatrix}} \text{ and } \underset{ns\times ns}{\begin{pmatrix} \square & \boxbslash & \cdots & \boxbslash \\ \boxbslash & \square & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \boxbslash & \cdots & \cdots & \square \end{pmatrix}}$$

- Candler, et al. originally derived this for Steger-Warming scheme, this work extends to Roe FDS scheme
    - Candler, G. V., Subbareddy, P. K., and Nompelis, I. "Decoupled Implicit Method for Aerothermodynamics and Reacting Flows." *AIAA Journal*, Vol. 51, no. 5, pp. 1245-1254.

# Fully-Coupled Point Implicit Method

- All work presented is for inviscid flows in chemical non-equilibrium, using a one-temperature model, but is extendable to viscous flows.
- Beginning with the semi-discrete form

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

$$\mathbf{U} = \begin{pmatrix} \rho_1 \\ \vdots \\ \rho_{ns} \\ \rho u \\ \rho v \\ \rho w \\ \rho E \end{pmatrix}, \quad \mathbf{F} \cdot \mathbf{S} = \begin{pmatrix} \rho_1 \overline{U} \\ \vdots \\ \rho_{ns} \overline{U} \\ \rho u \overline{U} + p s_x \\ \rho u \overline{U} + p s_y \\ \rho u \overline{U} + p s_z \\ (\rho E + p) \overline{U} \end{pmatrix} S, \quad \mathbf{W} = \begin{pmatrix} \dot{\rho}_1 \\ \vdots \\ \dot{\rho}_{ns} \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

# Fully-Coupled Point Implicit Method

- Using the Roe FDS scheme to compute the inviscid flux at the face, $\mathbf{F}^f$, and linearizing the system results in

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

- Which can be thought of more simply as

$$\mathbf{Au} = \mathbf{b}$$

$$\mathbf{A} \rightarrow \begin{array}{c}(4+ns) \times (4+ns) \\ \text{Jacobian Block}\end{array}$$

$$\mathbf{b} \rightarrow \begin{array}{c}(4+ns) \times 1 \\ \text{Residual}\end{array}$$

# Fully-Coupled Point Implicit Method

- Constructing the Jacobian in a fully-coupled fashion results in large, dense block matricies
- Using a stationary iterative method (i.e., Gauss-Seidel, SSOR, etc.), work is dominated by matrix-vector products

$$\text{Cost} \rightarrow O((4 + ns)^2)$$

- Leads to onerous quadratic scaling with respect to number of species

# Decoupled Point Implicit Method

- The main idea is to separate the meanflow and species composition equations, adding a new equation for the total mixture density
- Leads to two sets of conserved variables

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

Meanflow Species Composition

# Decoupled Point Implicit Method

- The fluxes are solved in two sequential steps
  - The mixture fluxes are first solved as

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

  - Followed by the species fluxes

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

- Since the mixture density was determined in the first step, step two actually solves for the species mass fractions

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

$$\hat{\mathbf{V}} = (c_1, \ldots, c_{ns})^T, c_s = \rho_s/\rho$$

# Decoupled Point Implicit Method

- The Roe FDS scheme species mass fluxes can be rewritten as

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

$$\frac{\partial \hat{\mathbf{F}}_{\rho_s}}{\partial c_s^L} = w\mathbf{F}_{\rho} + (1-w)\rho^L\lambda^+ - w\rho^R\lambda^-$$

$$\frac{\partial \hat{\mathbf{F}}_{\rho_s}}{\partial c_s^R} = (1-w)\mathbf{F}_{\rho} + (w-1)\rho^L\lambda^+ + w\rho^R\lambda^-$$

- Jacobian Approximations

$$\text{Step 1:} \quad \left.\frac{\partial \mathbf{F}}{\partial \mathbf{U}'}\right|_{\hat{\mathbf{V}}} = \underset{c_s = \text{Constant}}{5 \times 5 \text{ Roe FDS Jacobian}}$$

$$\text{Step 2:} \quad \left.\frac{\partial \mathbf{F}}{\partial \hat{\mathbf{V}}}\right|_{\hat{\mathbf{U}}'} = \begin{pmatrix} \frac{\partial F_{\rho_1}}{\partial c_1} & & 0 \\ & \ddots & \\ 0 & & \frac{\partial F_{\rho_{ns}}}{\partial c_{ns}} \end{pmatrix}$$

# Decoupled Point Implicit Method

- Chemical source term linearized via

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

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

- Full system to be solved in step two

$$\rho^{n+1}\frac{\delta \hat{\mathbf{V}}^{n}}{\Delta t} + \frac{1}{V}\sum_f \left(\frac{\partial \hat{\mathbf{F}}^f}{\partial \mathbf{V}^L}\delta \mathbf{V}^L + \frac{\partial \hat{\mathbf{F}}^f}{\partial \hat{\mathbf{V}}^R}\delta \hat{\mathbf{V}}^R\right)^{n,n+1}\mathbf{S}^f - \mathbf{C}^{n,n+1}\delta \mathbf{V}^n$$

$$= -\frac{1}{V}\sum_f (\hat{\mathbf{F}}^{n,n+1}\cdot \mathbf{S})^f + \mathbf{W}^{n,n+1} - \hat{\mathbf{V}}^n \frac{\delta \rho^n}{\Delta t} - R_\rho$$

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

- $R_\rho$ is included to preserve $\sum_s c_s = 1$, $\sum_s \delta c_s = 0$.

# Derivation of Discrete Adjoint Formulation

- The derivation of the adjoint approach to compute design sensitivities begins with forming the Lagrangian and differentiating with respect to the design variables

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

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

$$+ \left\{ \left[\frac{\partial \mathbf{R}}{\partial \mathbf{D}}\right]^T + \left[\frac{\partial \mathbf{X}}{\partial \mathbf{D}}\right]^T \left[\frac{\partial \mathbf{R}}{\partial \mathbf{X}}\right]^T \right\} \mathbf{\Lambda}$$

$\mathbf{D}$ = design variables  $f$ = cost function

$\mathbf{Q}$ = flow variables  $\mathbf{R}$ = flow residual

$\mathbf{X}$ = computational grid  $\mathbf{\Lambda}$ = costate variables

# Derivation of Discrete Adjoint Formulation

- The derivation of the adjoint approach to compute design sensitivities begins with forming the Lagrangian and differentiating with respect to the design variables

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

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

$$+\left\{\left[\frac{\partial \mathbf{R}}{\partial \mathbf{D}}\right]^T + \left[\frac{\partial \mathbf{X}}{\partial \mathbf{D}}\right]^T \left[\frac{\partial \mathbf{R}}{\partial \mathbf{X}}\right]^T\right\}\mathbf{\Lambda}$$

$\mathbf{D}$ = design variables — $f$ = cost function

$\mathbf{Q}$ = flow variables — $\mathbf{R}$ = flow residual

$\mathbf{X}$ = computational grid — $\mathbf{\Lambda}$ = costate variables

# Derivation of Discrete Adjoint Formulation

- Need to eliminate flow variable dependence on design variables, $\frac{\partial \mathbf{Q}}{\partial \mathbf{D}}$
- Adjoint equation

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

- Solve for $\mathbf{\Lambda}$ and compute sensitivity derivatives

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

# Fully Coupled Iterative Method

- Adjoint problem is a linear system

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

- Can be solved with Krylov method (.i.e GMRES), but time marching similar to flow solver shown to be more robust

$$\left( \frac{V}{\Delta t} \mathbf{I} + \frac{\partial \mathbf{R}_1}{\partial \mathbf{Q}}^T \right) \Delta \Lambda = - \frac{\partial f}{\partial \mathbf{Q}} - \frac{\partial \mathbf{R}_2}{\partial \mathbf{Q}}^T \Lambda$$

- Straightforward to formulate, but cost and memory requirements scale quadratically with number of species

# Decoupled Iterative Method

- Rewrite conserved variables similar to decoupled flow solver

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

# Decoupled Iterative Method

- Rewrite conserved variables similar to decoupled flow solver

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

- Recognize that there is an analogue to the species mass equation decoupling used in the flow solver
- Linear system can be decomposed as block jacobi scheme

# Decoupled Iterative Method

- Separate into two systems and solve as block jacobi scheme

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

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

# Cost and Memory Savings of the Decoupled Implicit Problem

- Most significant savings comes from the source term linearization being purely node-based
  - Convective contributions to block Jacobians are diagonal
  - Source term jacobian is dense block Jacobian
  - In the global system (w/chemistry), all off-diagonal block jacobians are diagonal

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

- Matrix-vector products $\rightarrow$ inner products: $O(ns^2) \rightarrow O(ns)$

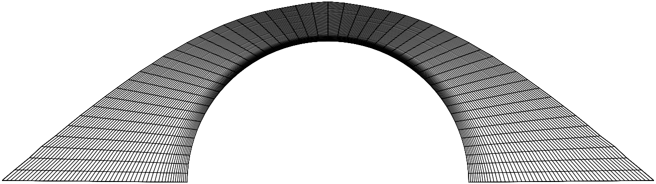
# Cost and Memory Savings of the Decoupled Implicit Problem

- Comparing size of Jacobian systems, using Compressed Row Storage

$$\mathbf{A}_d = \text{Decoupled system Jacobians}$$
$$\mathbf{A} = \text{Fully-coupled system Jacobians}$$

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

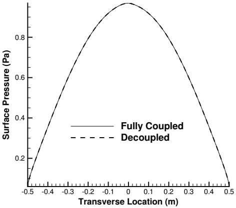
# Numerical Results: 2D Cylinder

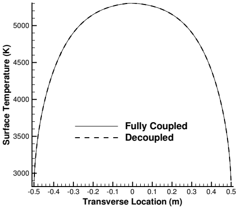


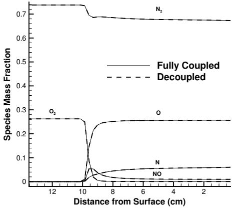
- Fully-coupled and decoupled methods both implemented in the Generic Gas Path of FUN3D
- Tested on 2D cylinder case
  - $V_\infty = 5000\ m/s$, $\rho_\infty = 0.001\ kg/m^3$, and $T_\infty = 200\ K$

# Numerical Results: 2D Cylinder

- Verification of implementation
  - 5-species air model: N, $N_2$, O, $O_2$, and NO with five reactions

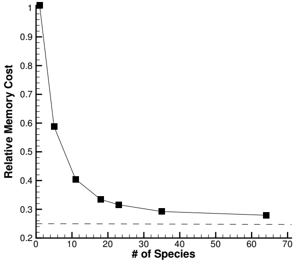


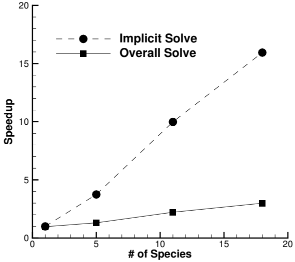




  - Surface pressure and surface temperature agree discretely to 8 significant figures
  - Mass fractions on stagnation line agree to 4 significant figures

# Numerical Results: 2D Cylinder



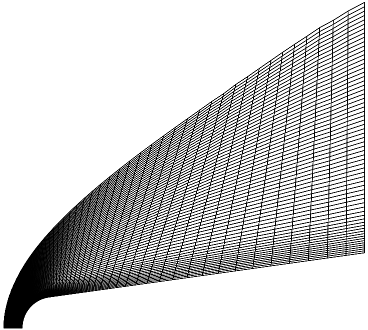




- On structured grids $N_{nbrs} \approx 6N_{nodes}$
  - Half precision off-diagonal $N_{nbrs} = \frac{6N_{nodes}}{2}$

$$Memory\ \ Cost \approx \frac{N_{nodes}}{N_{nodes} + N_{nbrs}} = \frac{N_{nodes}}{N_{nodes} + 6N_{nodes}/2} = \frac{1}{4}$$

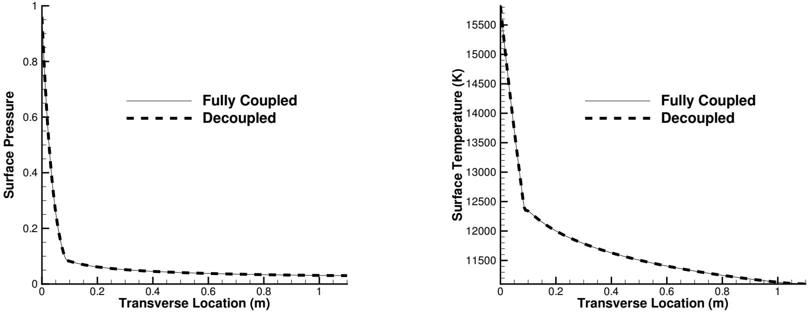
  - Linear speedup in solver: $\frac{O(N^2)}{O(N)} = O(N)$

# Numerical Results: Axisymmetric Spherically Capped Cone



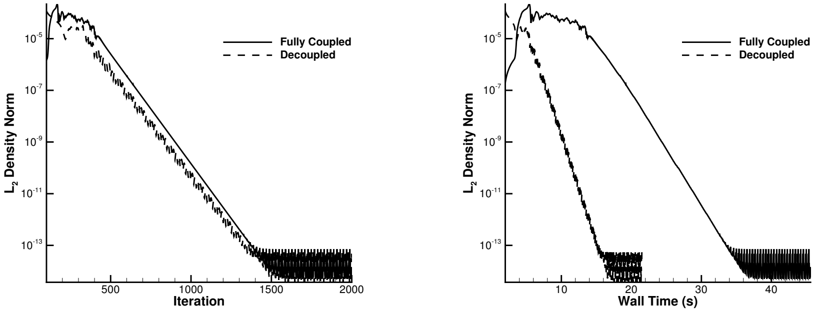
- Verify that the decoupled scheme is robust at high velocities
  - $V_\infty = 15000\ m/s$, $\rho_\infty = 0.001\ kg/m^3$, $T_\infty = 200\ K$.
  - 11-species air model N, $N_2$, O, $O_2$, NO, $N^+$, $N_2^+$, $O^+$, $O_2^+$, $NO^+$, and electrons, with 22 possible reactions.

# Numerical Results: Axisymmetric Spherically Capped Cone



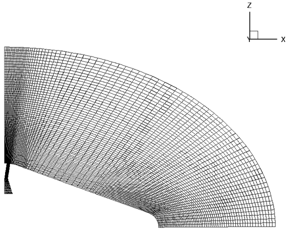

- Surface pressure and surface temperature agree discretely to 8 significant figures

# Numerical Results: Axisymmetric Spherically Capped Cone

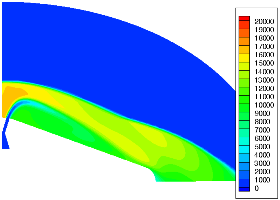



- Necessary to scale source term magnitude by $0.001 \leq w \leq 1$ for the first 500 iterations, due to extreme reaction rates
- Both schemes converge in a similar number of iterations
- Decoupled scheme $\approx$ 2x faster

# Design Problem: Hypersonic SRP Vehicle




Annular nozzle mesh




Temperature Contour

- Apply adjoint to design Reaction Control System (RCS) jet system to shape shock interation for maximum drag and minimum surface temperature
- This annular nozzle configuration has been shown to have a steady solution for inviscid flow

# Design Problem: Parameterization

- Design sensitivities given by

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

- Define cost functions $=> f$
  - Total vehicle drag (with and without jet thrust contribution)
  - Total vehicle surface temperature (in lieu of heating, since these are inviscid simulations)
- Define design parameters $=> \mathbf{D}$
  - Plenum pressure
  - Plenum temperature
  - Jet placement and geometry
- Define mesh parameters $=> \mathbf{X}$
  - Custom grid generation utility
  - Faciliate all grid dependencies by wrapping in complex variables

# Concluding Remarks

- Decoupling the species equations yield impressive benefits at minimal cost in robustness
  - 2 times faster and 1/3 required memory for both 2D Cylinder and Sphere-Cone 11-species cases
  - Convergence issues at very high velocities can be offset by scaling source term as solution progresses
- Decoupling appears promising for adjoint work
  - Preliminary testing has shown that memory overhead in adjoint is significantly reduced with decoupled scheme
  - Can expect similar speedup in adjoint solve
- Design problem provides good testbed for a truly unique hypersonic application
  - Parameterization is well understood and defined
  - Non-linearity of design space is a concern
  - Optimal steady solution exists?

# Acknowledgements


- Thanks the FUN3D team at NASA Langley Research Center, for their support in integrating aspects of the compressible gas path into the reacting gas path of FUN3D.
- Thanks to the Entry Systems Modeling Project within the NASA Game Changing Development Program for their funding and support of this research.

# Backup: Derivation

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

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

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

$$dv_2 = \frac{p^R - p^L - \tilde{\rho}\tilde{a}(\mathbf{U}^R - \mathbf{U}^L)}{\tilde{a}^2}$$

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

$$\lambda_1 = \mid \tilde{\mathbf{U}} + \tilde{a} \mid, \quad \lambda_2 = \mid \tilde{\mathbf{U}} - \tilde{a} \mid, \quad \lambda_3 = \mid \tilde{\mathbf{U}} \mid$$

# Backup: Derivation

- The ˜ notation signifies a Roe-averaged quantity

$$\tilde{\mathbf{U}} = wU^L + (1 - w)\mathbf{U}^R$$

$$w = \frac{\tilde{\rho}}{\tilde{\rho} + \rho^R}$$

$$\tilde{\rho} = \sqrt{\rho^R \rho^L}$$

- The species mass fluxes must sum to the total mass flux

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

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

# Backup: Derivation

- Substituting back into species mass flux equation

$$F_{\rho_s} = \tilde{c}_s F_\rho + \frac{(c_s^L - \tilde{c}_s)\rho^L(\mathbf{U}^L + \mid \tilde{\mathbf{U}} \mid)}{2} + \frac{(c_s^R - \tilde{c}_s)\rho^R(\mathbf{U}^R - \mid \tilde{\mathbf{U}} \mid)}{2}$$

- This can be simplified to yield a form similar to that derived by Candler, et. al for the Steger-Warming scheme

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

$$\lambda^+ = \frac{\mathbf{U}^L + \mid \tilde{\mathbf{U}} \mid}{2}, \quad \lambda^- = \frac{\mathbf{U}^R - \mid \tilde{\mathbf{U}} \mid}{2}$$

# Backup: Derivation

- Differentiating with respect to the mass fraction, $c_s$, the left and right state contributions are

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

- Again, where $w$ is the Roe-averaged density weighting

$$w = \frac{\tilde{\rho}}{\tilde{\rho} + \rho^R}, \quad \tilde{\rho} = \sqrt{\rho^R\rho^L}$$