# A Newton-Krylov Algorithm for Hypersonic Flows
## Performance Demonstration and Application

K. A. Damm

Centre for Hypersonics, School of Mechanical and Mining Engineering,
University of Queensland, Brisbane, Queensland 4072, Australia

22nd July 2021

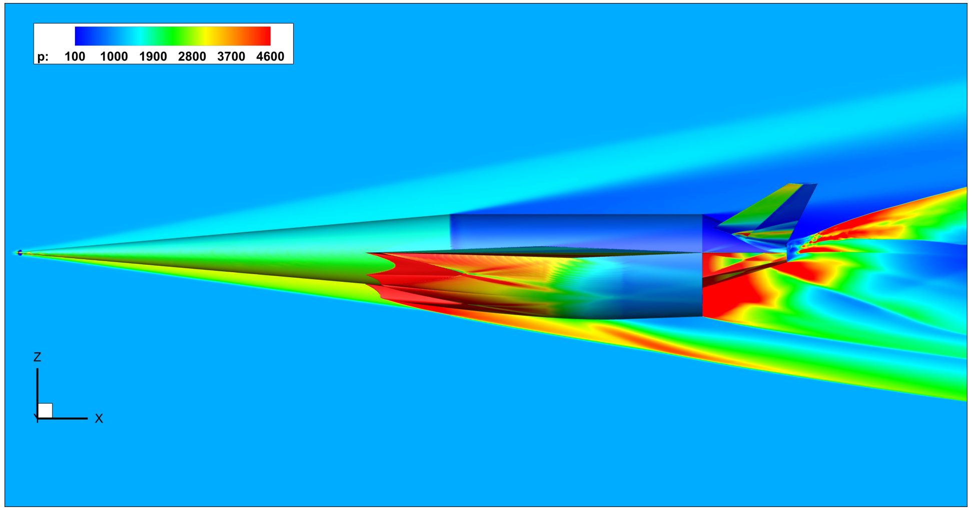
# Overview

- Motivation: Hypersonic vehicle design via numerical optimization
- Newton-Krylov methods
- **Eilmer4** overview
- Demonstrative examples
- Application: BoLT-II
- Grand challenge: HIFiRE-7
- Future Work

# Hypersonic Vehicle Design

- Flow around a vehicle is **complex**:
    - -- Shock-shock interactions
    - -- Shock boundary layer interactions
    - -- Separated regions of flow
    - -- Thermochemical nonequilibrium

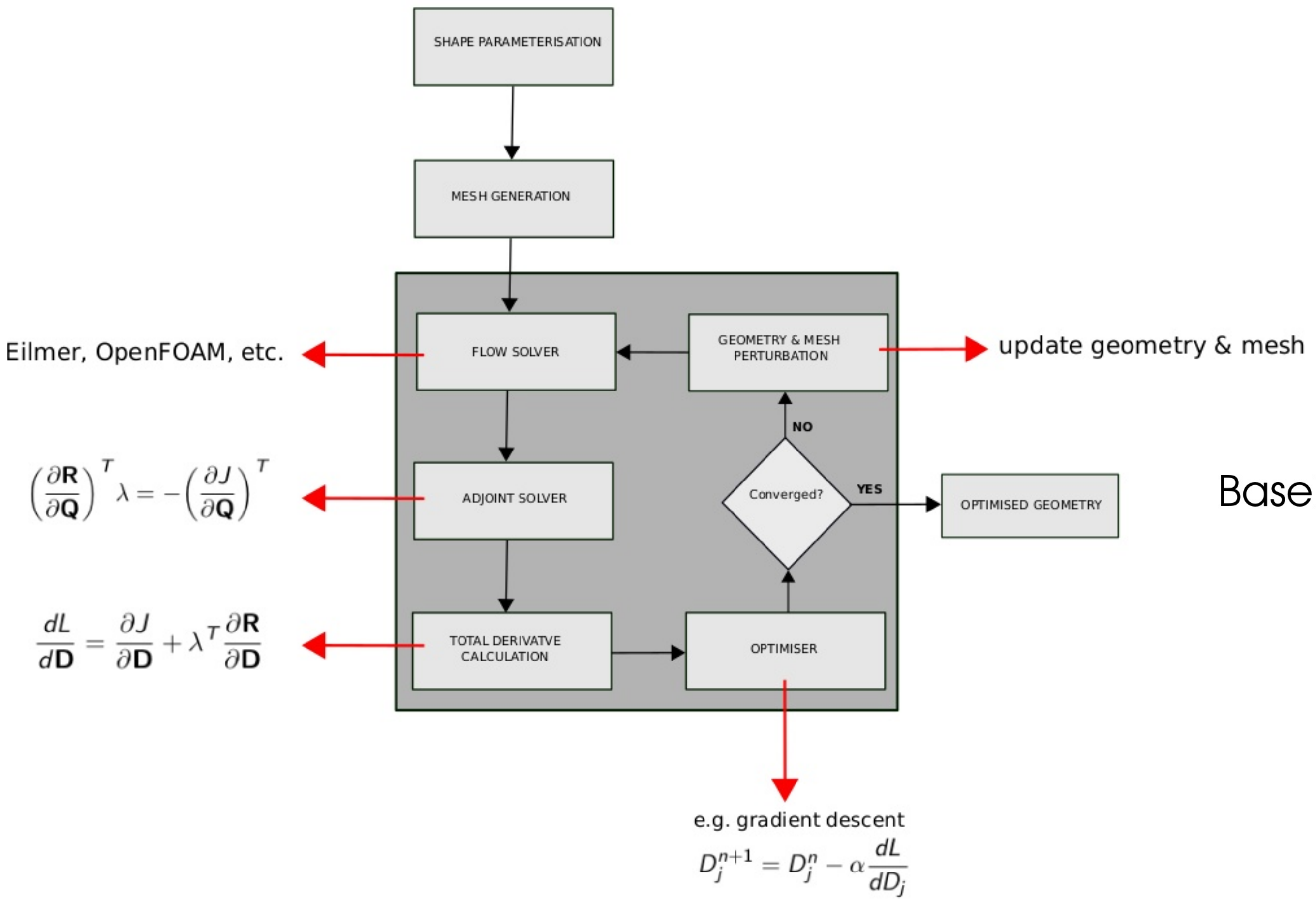
- **High-fidelity CFD** to resolve flow physics

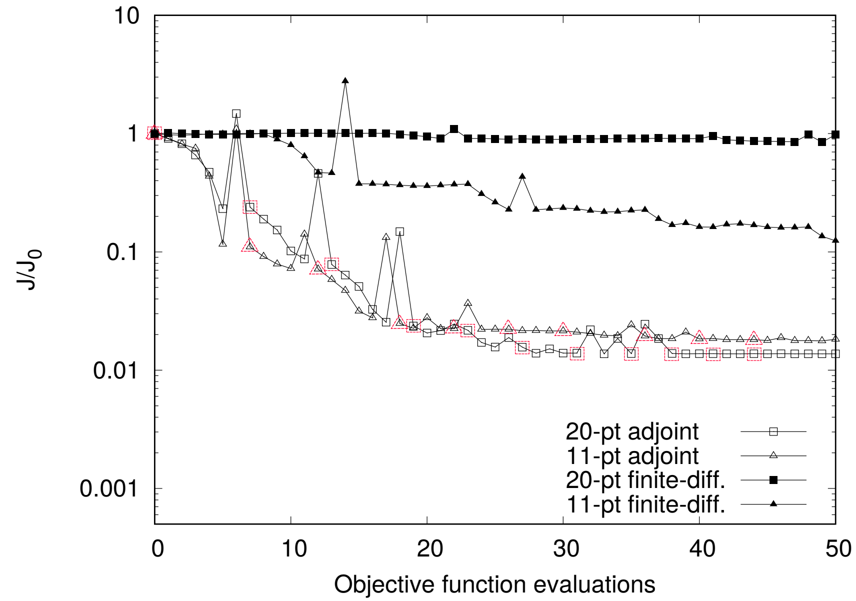




Source: Alex Ward (Hypersonix)

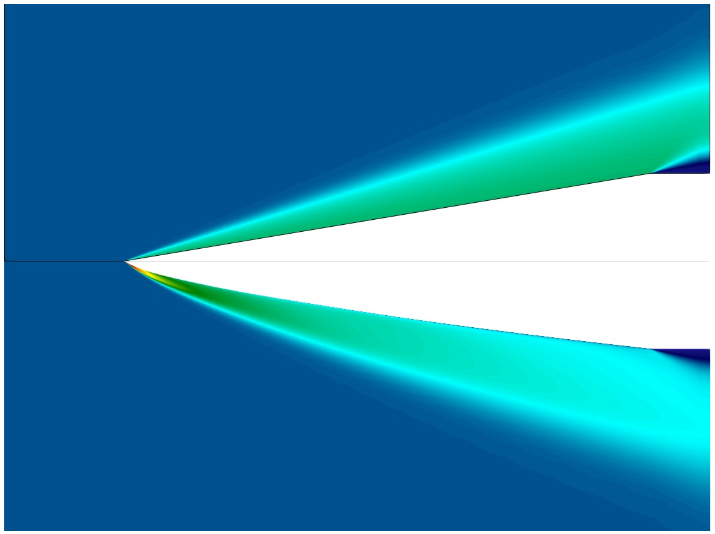
# Hypersonic Vehicle Design Optimization

- Example: minimum-drag slender body of revolution
  - -- **Many flow solutions** to achieve converged design
  - -- Adjoint-based method requires **deep** convergence




Baseline

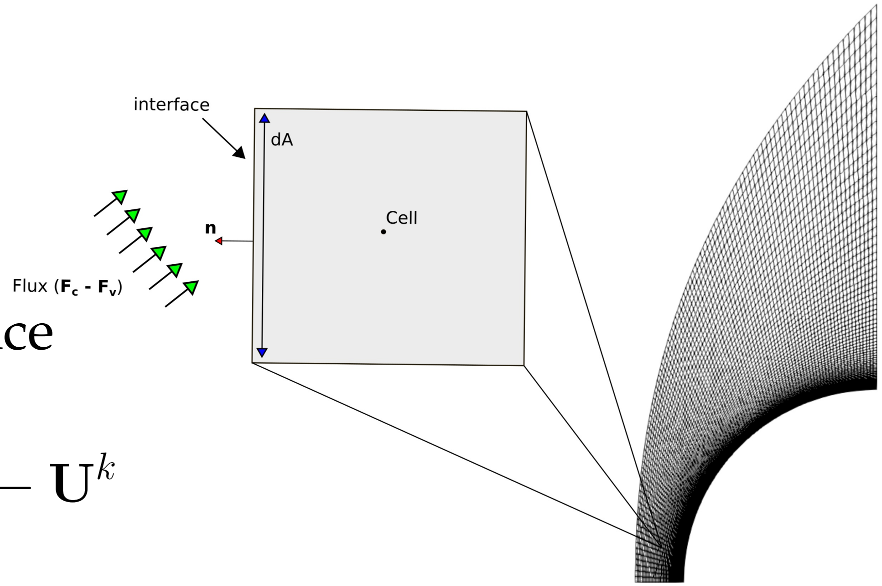


Optimized

# Newton-Krylov Methods

Residual function defined as

$$\frac{d\mathbf{U}}{dt} = \mathbf{R}(\mathbf{U}) = -\frac{1}{V}\sum_{faces}(\overline{F_c} - \overline{F_v}) \cdot \hat{n}\, dA + \mathbf{S}$$




Fully discrete form written using a backward difference

$$\frac{\Delta\mathbf{U}^k}{\Delta t} = \mathbf{R}(\mathbf{U}^{k+1}), \quad \Delta\mathbf{U}^k = \mathbf{U}^{k+1} - \mathbf{U}^k$$

Since we don't know $\mathbf{R}(\mathbf{U}^{k+1})$, we linearise in time

$$\frac{\Delta\mathbf{U}^k}{\Delta t} = \mathbf{R}(\mathbf{U}^k) + \frac{\partial\mathbf{R}(\mathbf{U}^k)}{\partial\mathbf{U}^k}\Delta\mathbf{U}^k$$

This is then rearranged to recover the implicit-Euler time marching iterate

$$\mathbf{J}(\mathbf{U}^k)\Delta\mathbf{U}^k = \left[\frac{1}{\Delta t}\mathbf{I} - \frac{\partial\mathbf{R}(\mathbf{U}^k)}{\partial\mathbf{U}^k}\right]\Delta\mathbf{U}^k = \mathbf{R}(\mathbf{U}^k), \quad \mathbf{U}^{k+1} = \mathbf{U}^k + \Delta\mathbf{U}^k$$

Note: as $\frac{1}{\Delta t}$ approaches 0, Newton's method is recovered

# Newton-Krylov Methods

Solving the linear system

$$\mathbf{J}(\mathbf{U}^k)\Delta\mathbf{U}^k = \mathbf{R}(\mathbf{U}^k) \rightarrow \mathbf{Ax} = \mathbf{b}$$

ALGORITHM **6.9** *GMRES*

1. Compute $r_0 = b - Ax_0$, $\beta := \|r_0\|_2$, and $v_1 := r_0/\beta$
2. For $j = 1, 2, \ldots, m$ Do:
3. Compute $w_j := Av_j$
4. For $i = 1, \ldots, j$ Do:
5. $h_{ij} := (w_j, v_i)$
6. $w_j := w_j - h_{ij} v_i$
7. EndDo
8. $h_{j+1,j} = \|w_j\|_2$. If $h_{j+1,j} = 0$ set $m := j$ and go to 11
9. $v_{j+1} = w_j / h_{j+1,j}$
10. EndDo
11. Define the $(m+1) \times m$ Hessenberg matrix $\bar{H}_m = \{h_{ij}\}_{1 \le i \le m+1, 1 \le j \le m}$.
12. Compute $y_m$ the minimizer of $\|\beta e_1 - \bar{H}_m y\|_2$ and $x_m = x_0 + V_m y_m$.

- The matrix only ever appears as a matrix-vector product in Krylov algorithms.
- The result of the matrix-vector product is a vector.
- Use the Frechet derivative to compute the vector result directly without forming the matrix.

Source: Saad (2003)

$$\mathbf{Jv} = \left[\mathbf{R}(\mathbf{U} + \epsilon\mathbf{v}) - \mathbf{R}(\mathbf{U})\right]/\epsilon$$

***We use a complex step variant**

# Newton-Krylov Methods

- **Benefits** of the **Newton-Krylov** approach:
  -- able to treat **R(U)** as a **black box**
  -- good for high speed flows (i.e. grids with high aspect ratio cells)
  -- works for both structured and unstructured grids
  -- avoid the need to derive and code implicit boundary conditions
  -- easily parallelized and scales well in parallel
  -- efficient on memory, in particular in 3D

- **DISCLAIMER:** GMRES requires a **preconditioning** step for fast convergence!!
  -- popular methods: Jacobi, SGS/SSOR (LU-SGS), **ILU**
  -- most require **approximate matrix** to be constructed
  -- we use forward-mode AD via a **complex-step derivate** approach
  -- can **freeze matrix** over several steps to amortize cost

# Compressible Flow Governing Equations

*Conservation of mass:*

$$\frac{\partial}{\partial t}\rho + \nabla \cdot \rho \mathbf{u} = 0 \tag{1}$$

*Conservation of species mass:*

$$\frac{\partial}{\partial t}\rho_i + \nabla \cdot \rho_i \mathbf{u} = -(\nabla \cdot \mathbf{J}_i) + \dot{\omega}_i \tag{2}$$

*Conservation of momentum:*

$$\frac{\partial}{\partial t}\rho \mathbf{u} + \nabla \cdot \rho \mathbf{u}\mathbf{u} = -\nabla p - \nabla \cdot \left\{ -\mu(\nabla \mathbf{u} + (\nabla \mathbf{u})^{\dagger}) + \tfrac{2}{3}\mu(\nabla \cdot \mathbf{u})\boldsymbol{\delta} \right\} \tag{3}$$

*Conservation of total energy:*

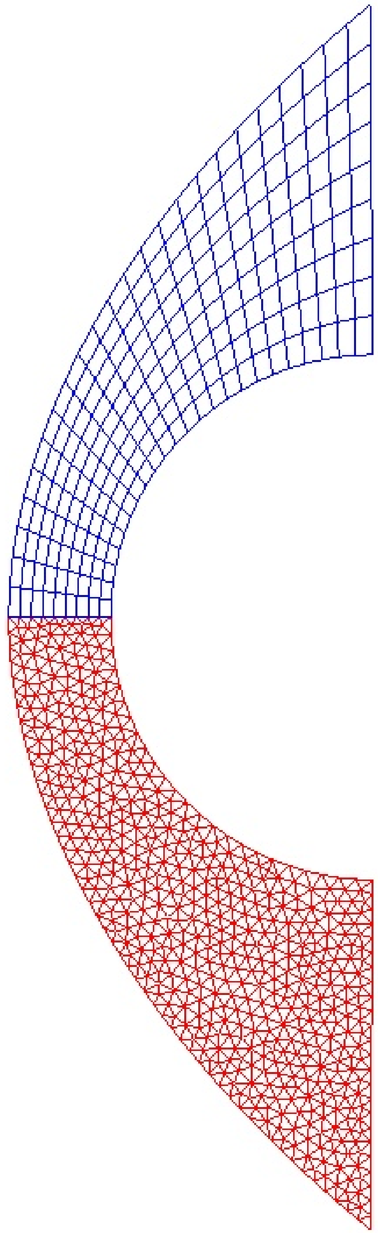
$$\frac{\partial}{\partial t}\rho E + \nabla \cdot (e + \frac{p}{\rho})\mathbf{u} = \nabla \cdot [k \nabla T + \sum_{s=1}^{N_v} k_{v,s} \nabla T_{v,s}] + \nabla \cdot \left[ \sum_{i=1}^{N_s} h_i \mathbf{J}_i \right] - \left( \nabla \cdot \left[ \left\{ -\mu(\nabla \mathbf{u} + (\nabla \mathbf{u})^{\dagger}) + \tfrac{2}{3}\mu(\nabla \cdot \mathbf{u})\boldsymbol{\delta} \right\} \cdot \mathbf{u} \right] \right) - Q_{\mathrm{rad}} \tag{4}$$

*Conservation of vibrational energy:*

$$\frac{\partial}{\partial t}\rho_i e_{v,i} + \nabla \cdot \rho_i e_{v,i} \mathbf{u} = \nabla \cdot [k_{v,i} \nabla T_{v,i}] - \nabla \cdot e_{v,i} \mathbf{J}_i + Q_{T-V_i} + Q_{V-V_i} + Q_{\mathrm{Chem}-V_i} - Q_{\mathrm{rad}_i} \tag{5}$$

# Spatial Discretization

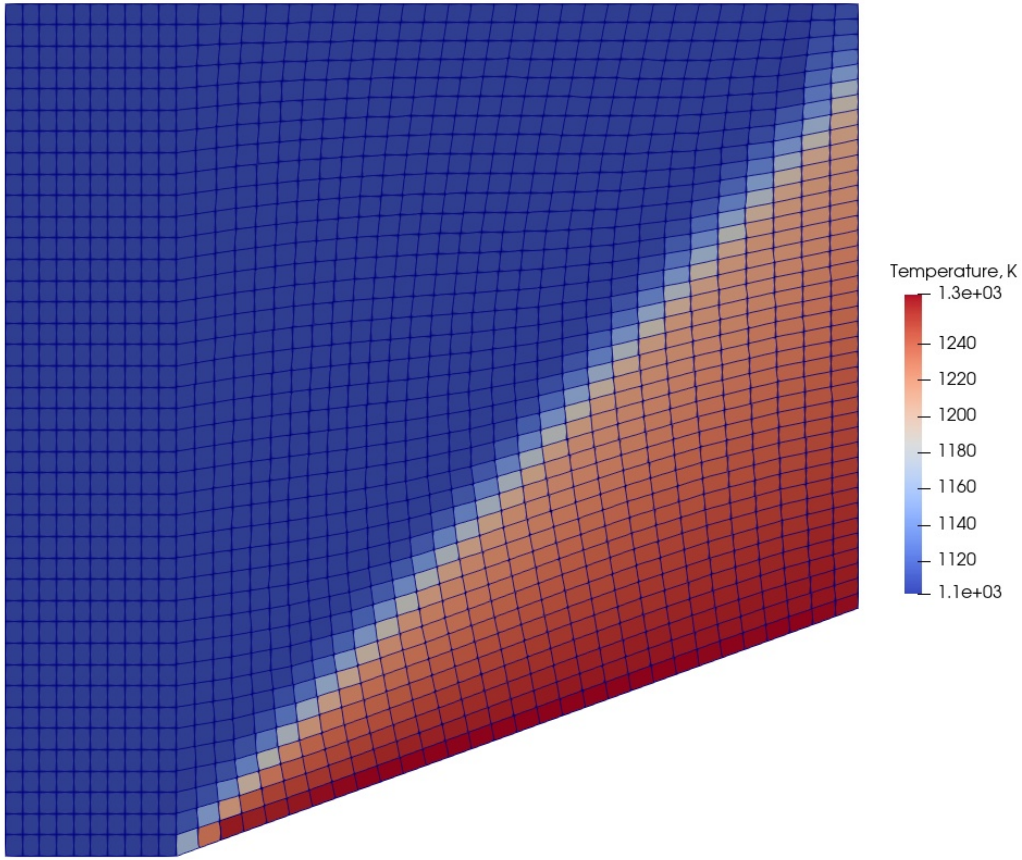
- **Convective Fluxes**
  - -- **Flux calculators**
    - + EFM, AUSMDV, HLLC, LDFSS, Hanel, HLLE, Roe, ASF
  - -- **Structured Grids**
    - + Piecewise parabolic reconstruction - $O(h^3)$
    - + Modified Van Albada limiter
  - -- **Unstructured Grids**
    - + Least-squares reconstruction - $O(h^2)$
    - + Venkatakrishnan limiter
    - + Limiter freezing available

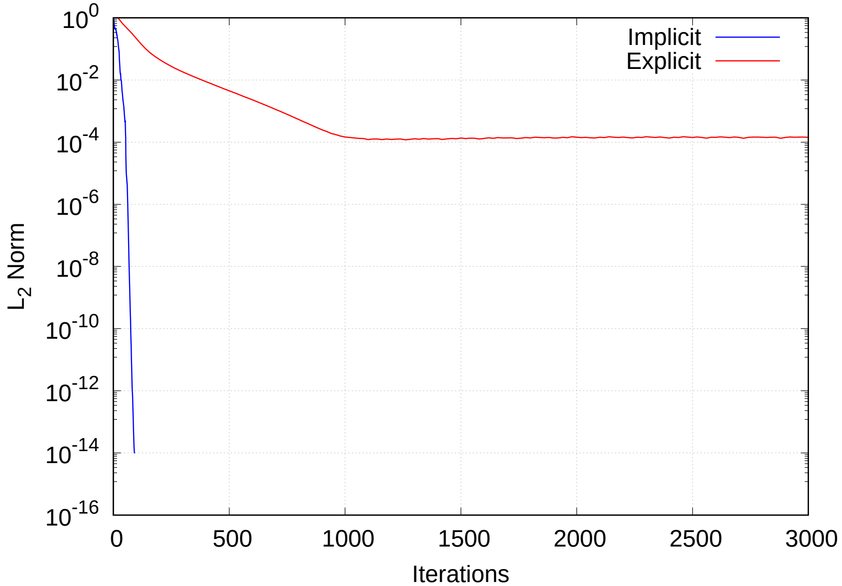


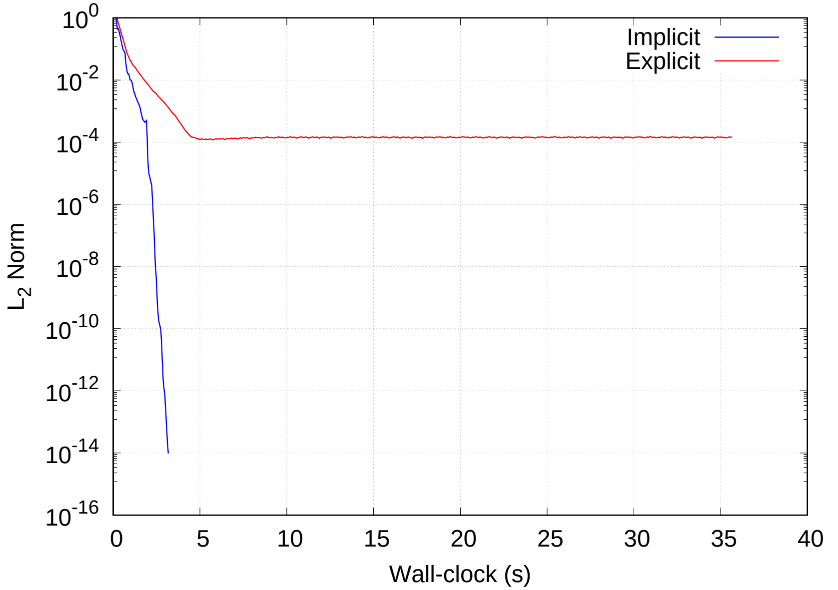
- **Viscous Fluxes**
  - -- **Augmented-face face-tangent method**
    - + Least-squares method to reconstruct gradients at cell center
    - + Special averaging using gradients, flowstates, and cell geometry
    - + available with structured and unstructured grids
    - + retains high spatial order for multi-block simulations

# Example #1: Inviscid Cone

- **Flow Condition:** Mach 1.5 single-species air
- **Geometry:** 20 degree cone (2D axisymmetric)
- **Numerics:** AUSMDV with $O(h^2)$ spatial reconstruction
- **CFL schedule**: 1.0 to $1\text{x}10^6$ (automatic growth)
- Solving **Euler** equations

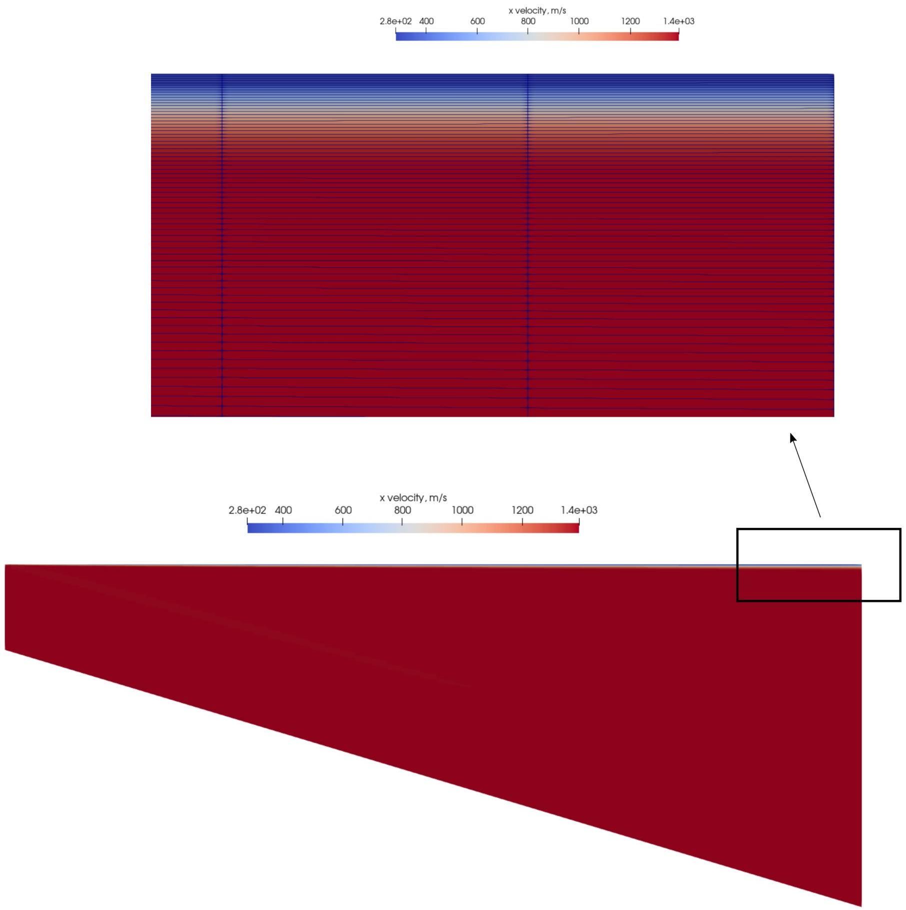


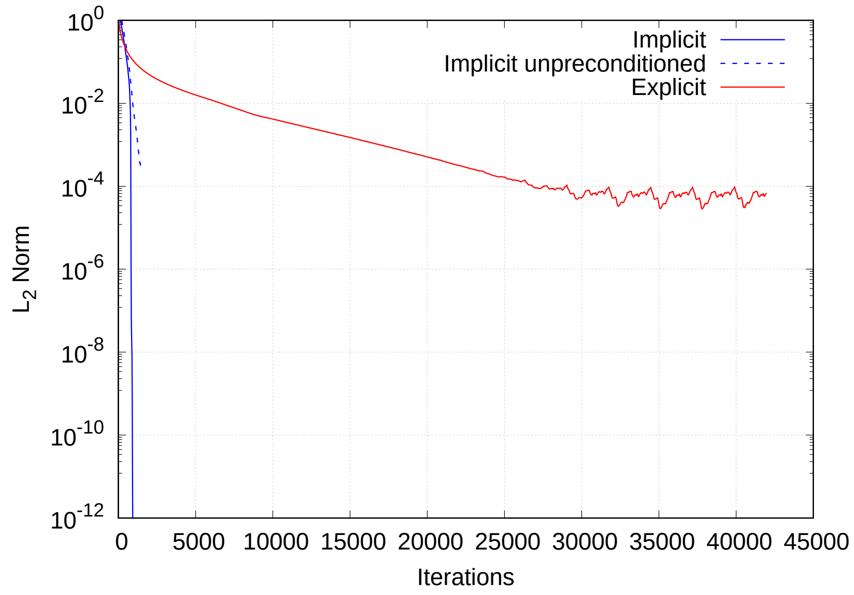



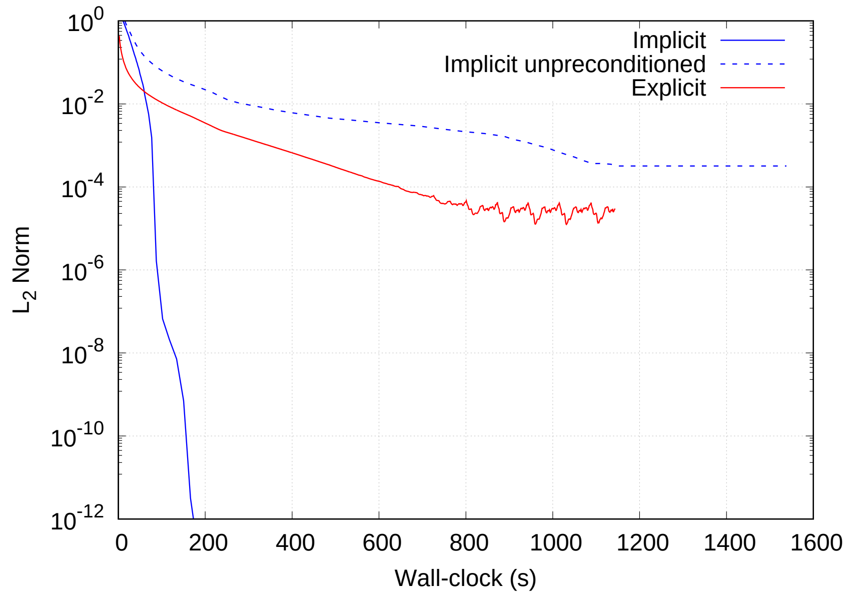



# Example #2: Laminar Flat Plate

- **Flow Condition:** Mach 4 single-species air
- **Geometry:** 2D flat plate
- **Numerics:** AUSMDV with $O(h^2)$ spatial reconstruction
- **CFL schedule**: 0.1 to $1\text{x}10^6$ (automatic growth)
- Solving **Navier-Stokes** equations

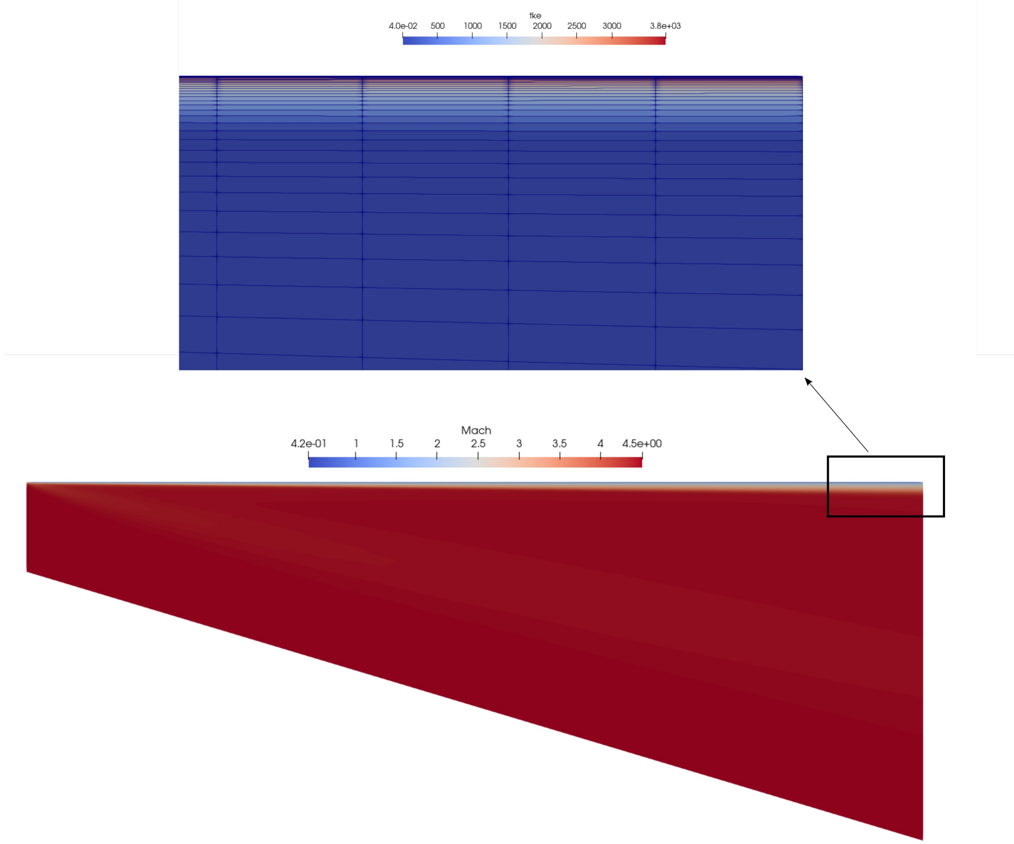


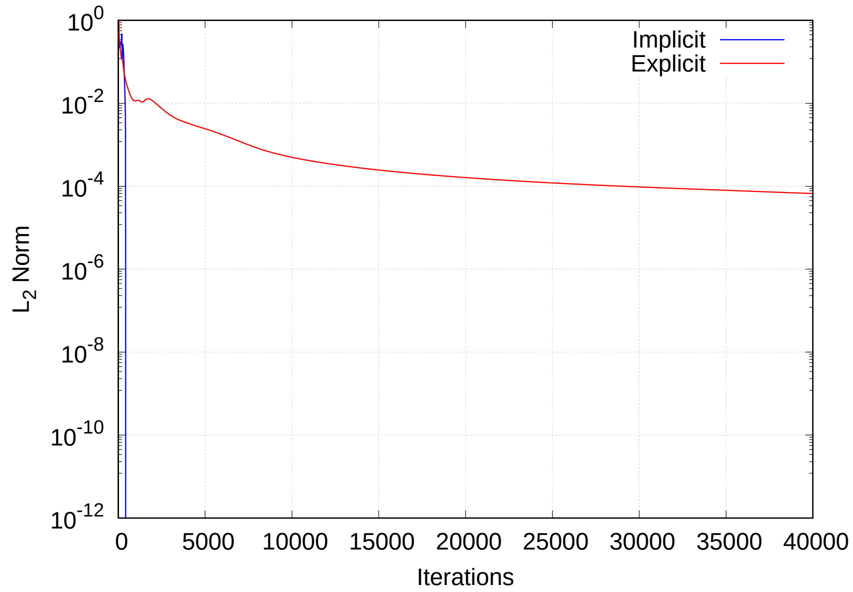



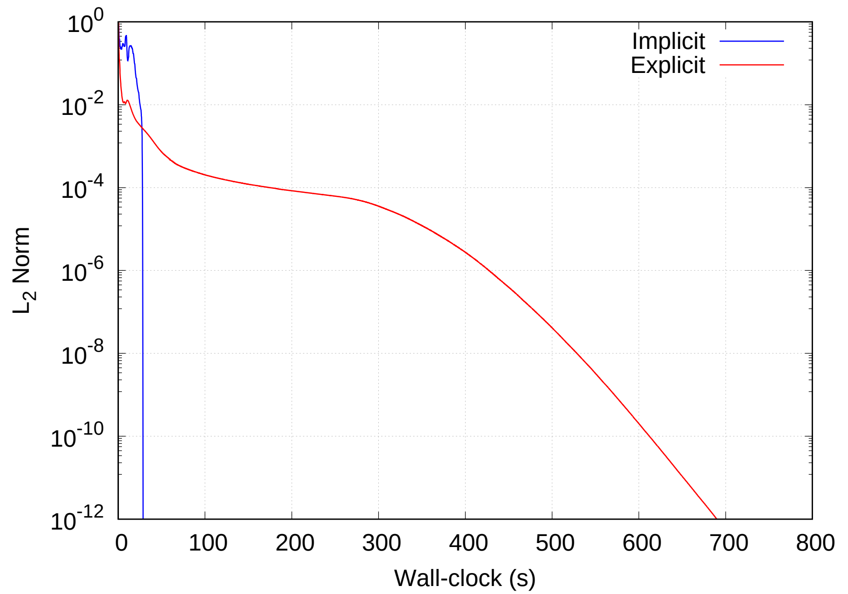



# Example #3: Turbulent Flat Plate

- **Flow Condition:** Mach 4.5 single-species air
- **Geometry:** 2D flat plate model from Mabey (1976)
- **Numerics:** AUSMDV with O(h) spatial reconstruction
- **CFL schedule**: 0.1 to 1x$10^6$ (automatic growth)
- Solving **RANS** equations
- Employed **k-omega** two-equation turbulence model

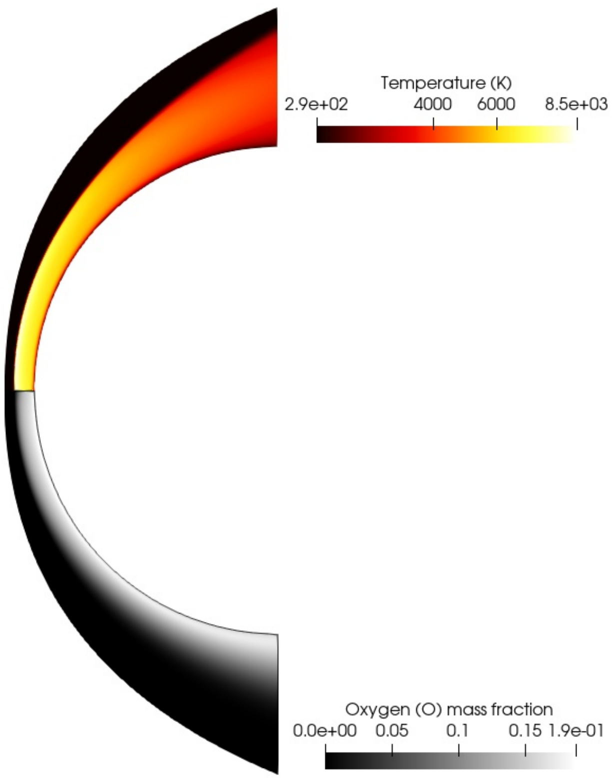


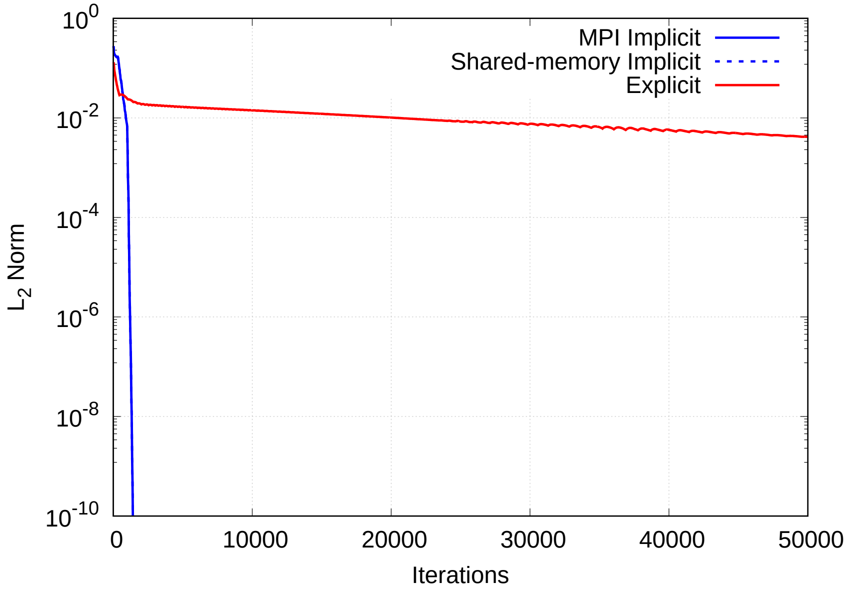



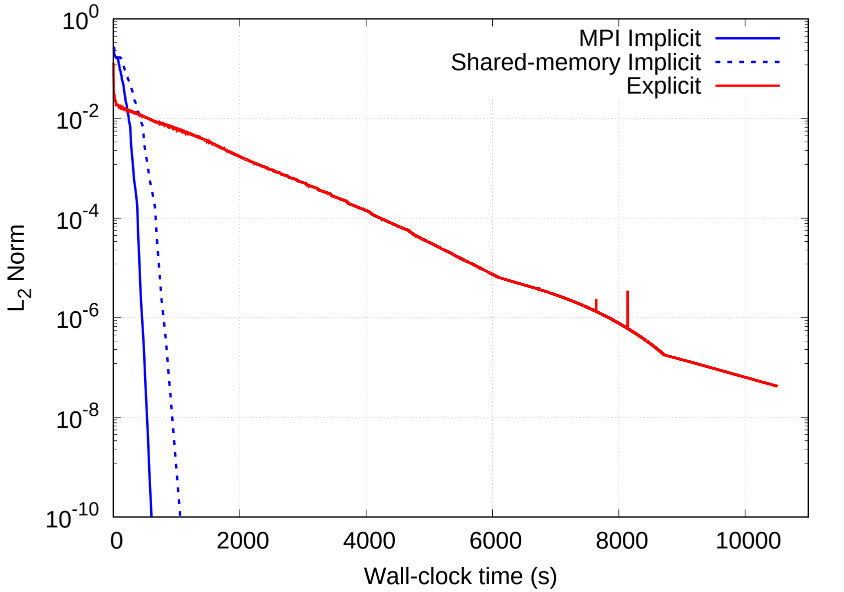



# Example #4: Laminar Reacting Flow over a Sphere

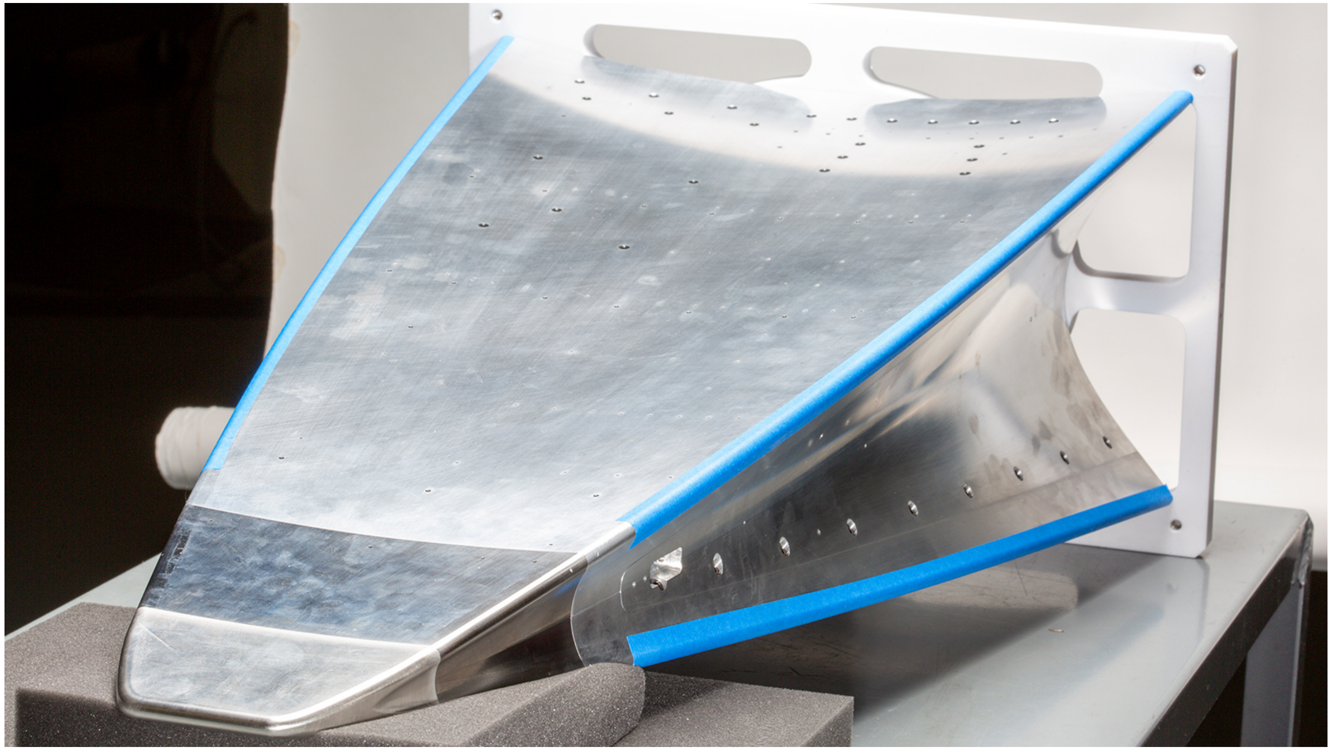
- **Flow Condition:** Mach 14 air
- **Geometry:** Sphere (2D) model from Lobb (1964)
- **Numerics:** Hanel with O(h) spatial reconstruction
- **CFL schedule**: 1 to 1000 (aggressive schedule)
- Solving **Navier-Stokes** equations
- **Finite-rate chemistry:** 5s\6r Gupta et al. (1990) model



# **Application:** BoLT-II Project Simulations

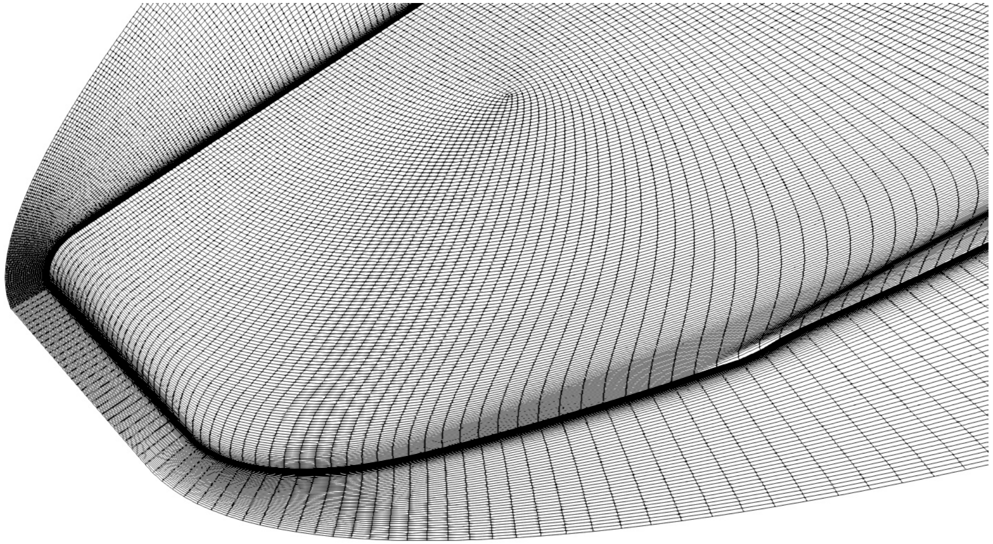


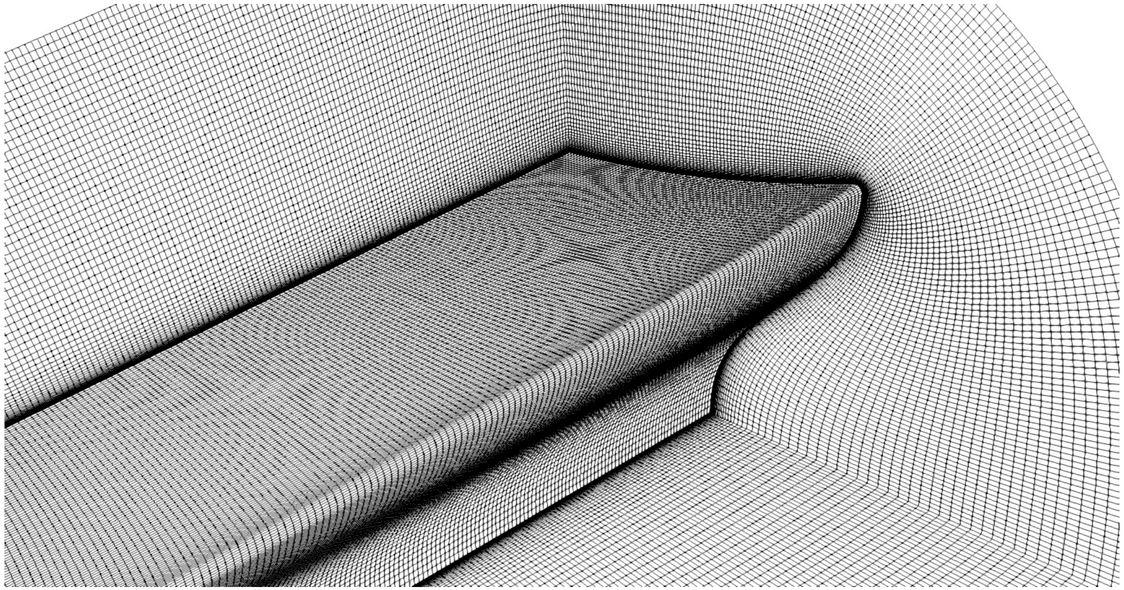
Source: AFRL/Johns Hopkins APL

- **Bo**undary-**L**ayer **T**ransition program sponsored by AFRL/AFOSR
- **Project goal:** Provide database for natural boundary layer transition
- **Ground tests**, **simulations**, flight experiment (later this year)
- **Application #1:** high-fidelity steady-state simulations to feed into **DNS** work
- **Application #2:** assist in T4 tunnel **experimental design**

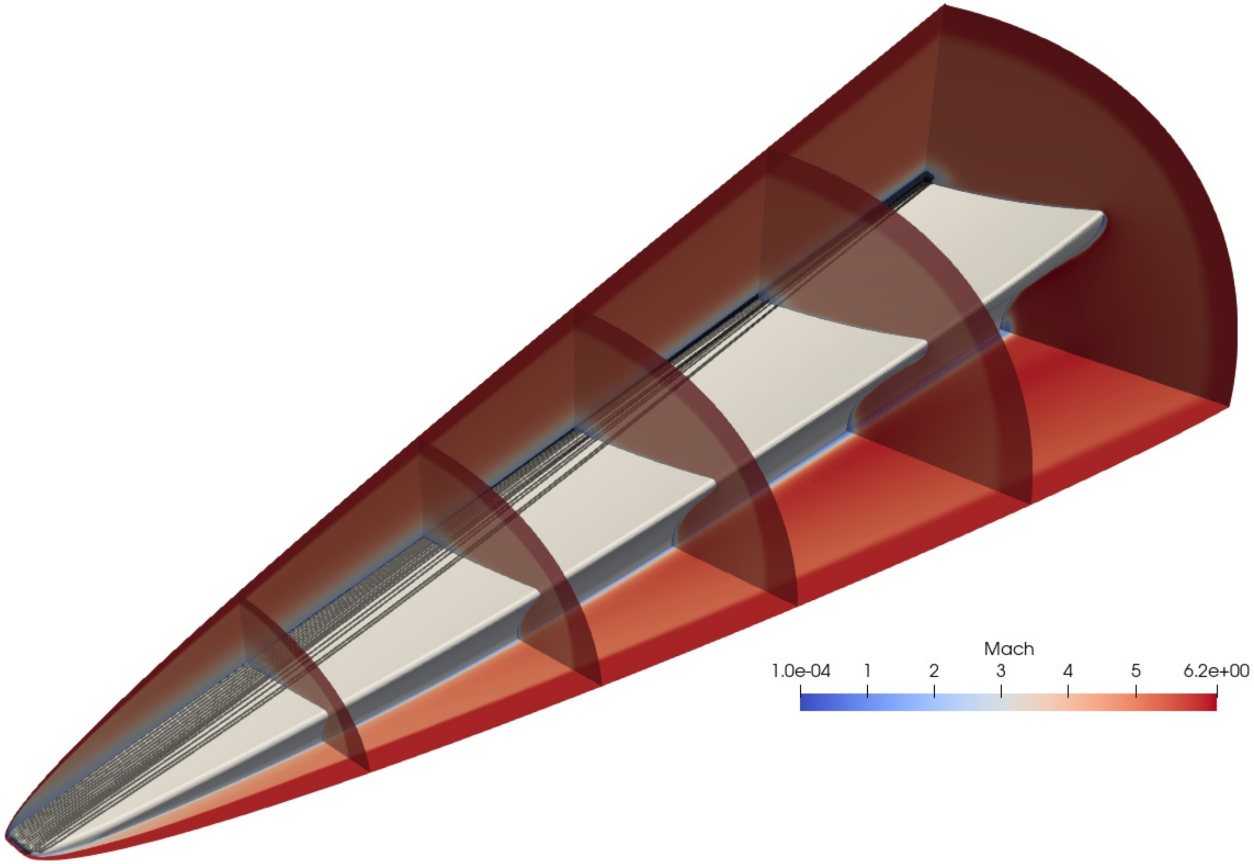
# **Application:** BoLT-II High-fidelity Laminar Simulation

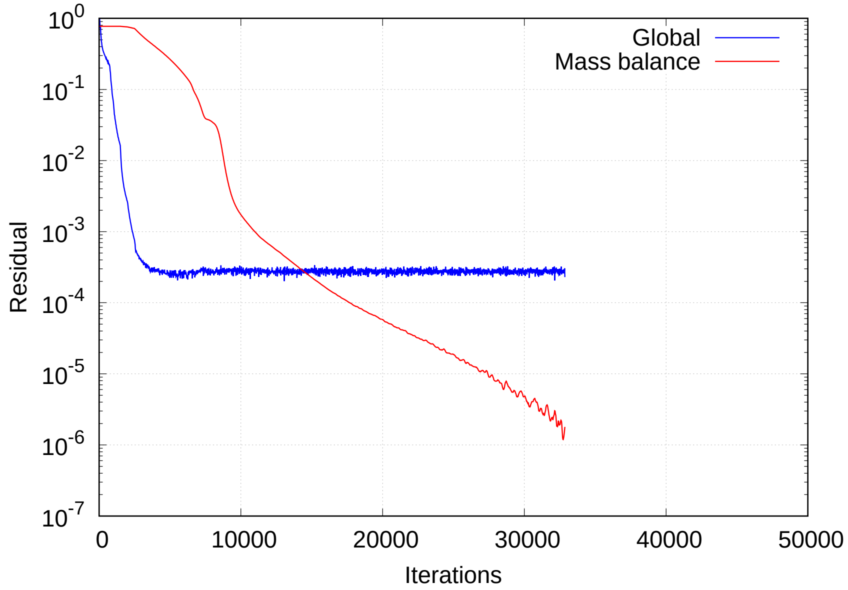
- **Flow Condition:** Mach 6 (tunnel condition) single-species air
- **Geometry:** 1/3 scale BoLT-II tunnel model
- **Numerics:** blended Hanel-AUSMDV with $O(h^2)$ spatial reconstruction
- **CFL schedule**: 0.001 to 1000 (conservative schedule)
- Solving **Navier-Stokes** equations
- 6.5 million cell (**GridPro**) structured elements stored in unstructured grid format
- Grid partitioned into 480 blocks using Eilmer4 **METIS** wrapper

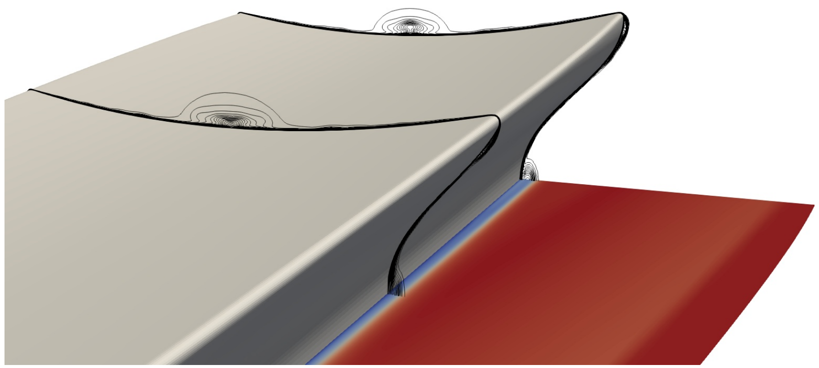




# **Application:** BoLT-II High-fidelity Laminar Simulation

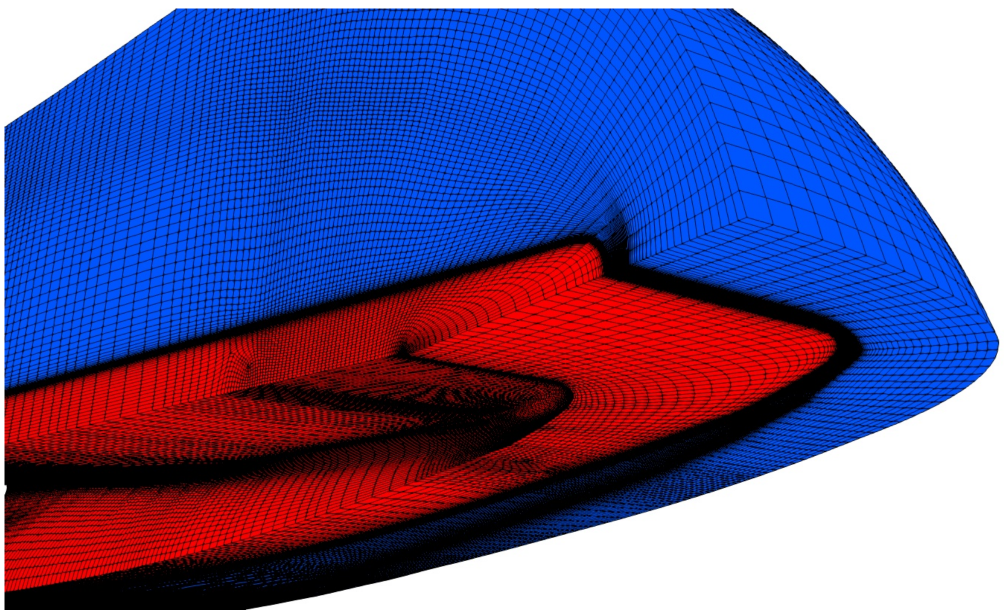


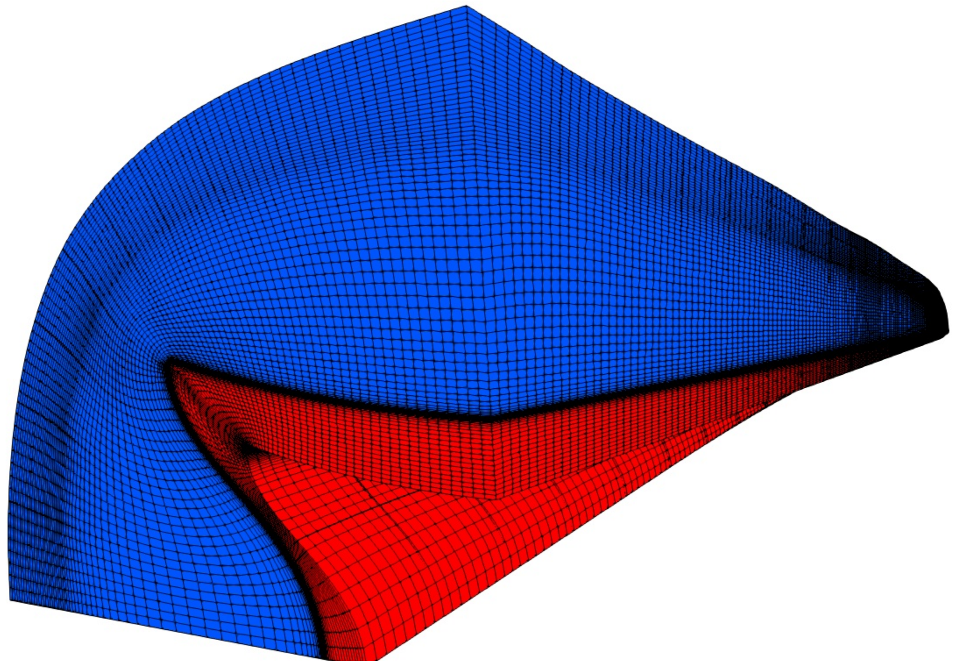



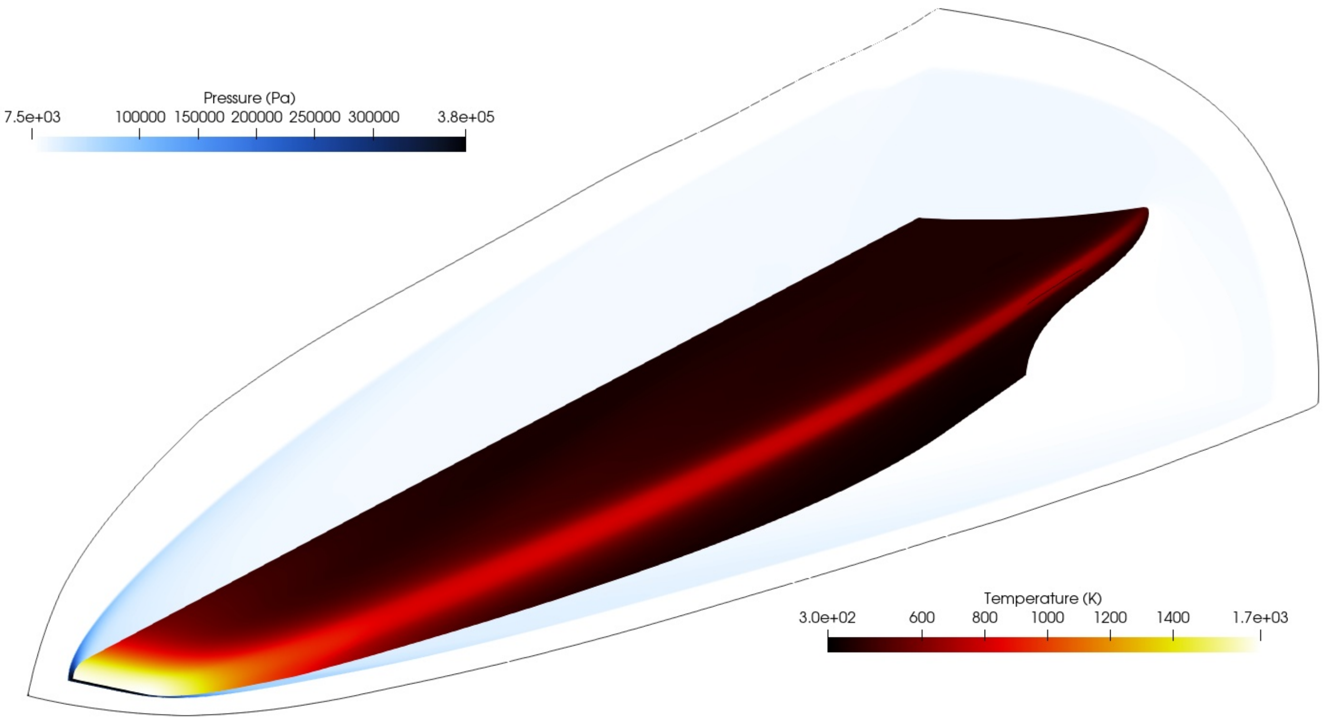
# **Application:** BoLT-II CHT Simulation

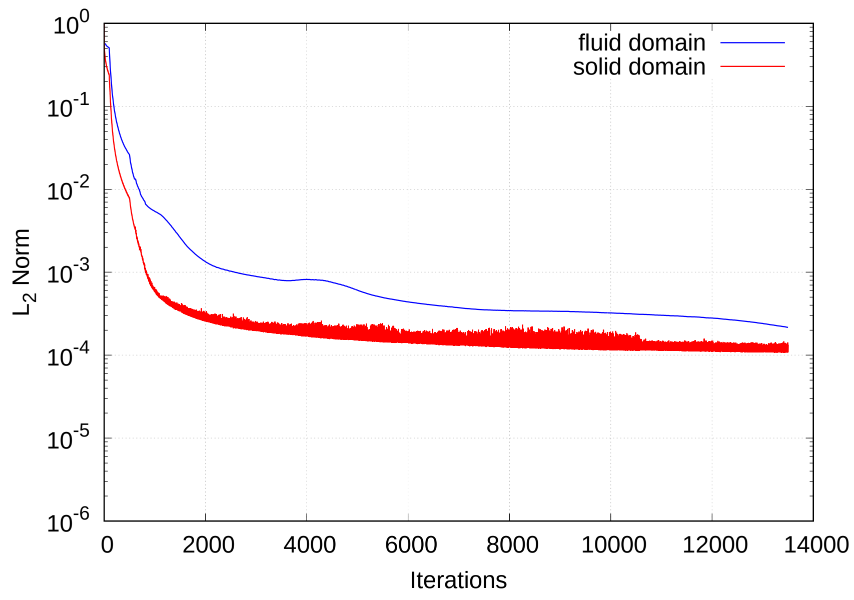
- **Flow Condition:** Mach 6 (tunnel condition) single-species air
- **Geometry:** 1/3 scale BoLT-II tunnel model
- **Numerics:** AUSMDV with O($h^3$) spatial reconstruction
- **CFL schedule**: 0.1 to 1000 (conservative schedule)
- Solving **Navier-Stokes** equations in **fluid domain**
- Solving **energy** equation in **solid domain**
- 1.2 million cell (**GridPro**) structured elements stored in structured grid format

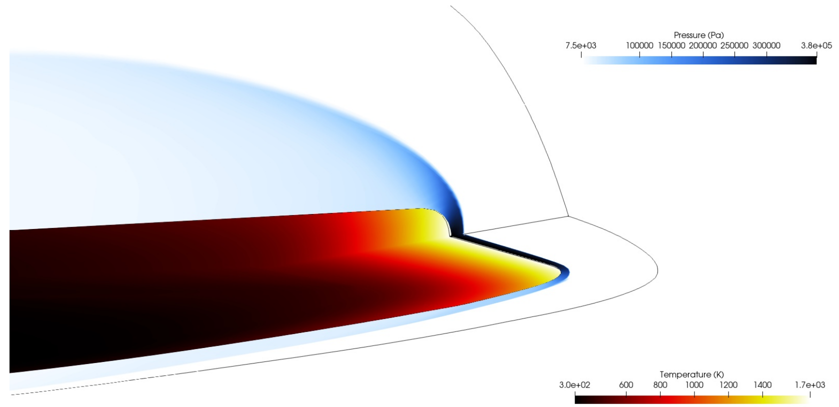




# **Application:** BoLT-II CHT Simulation

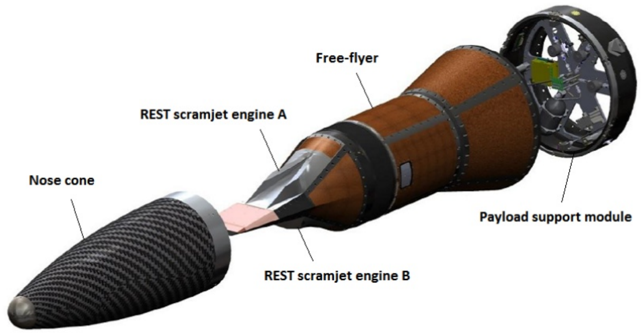


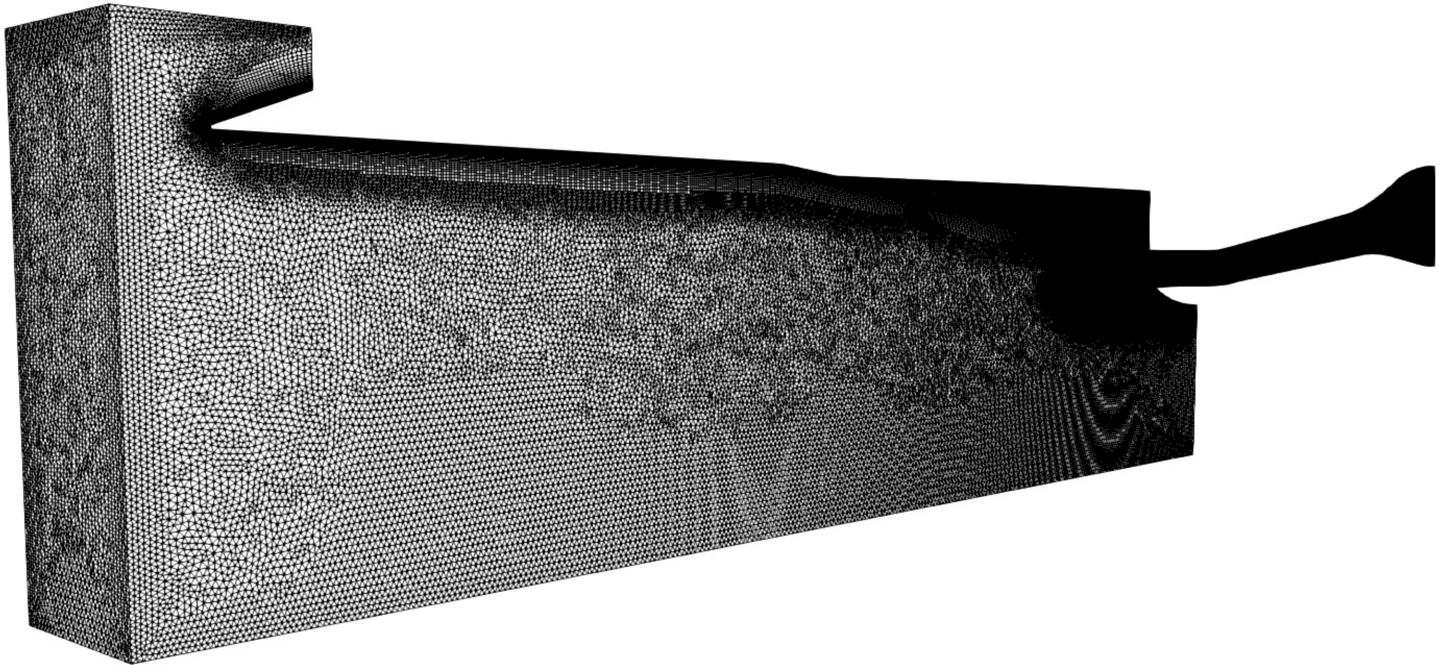



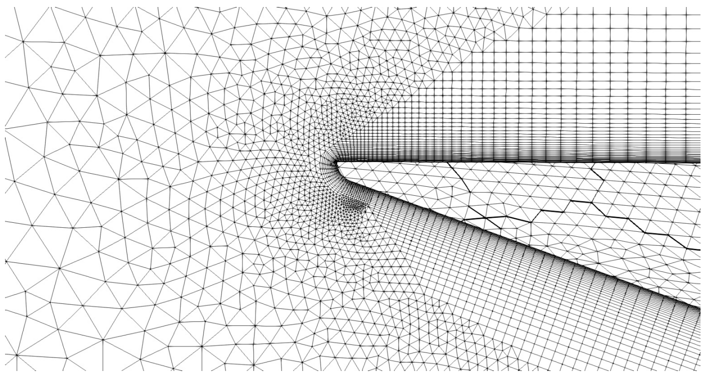
# **Grand Challenge:** HIFiRE-7 Simulation

- **Flow Condition:** Mach 7.8 (tunnel condition) single-species air
- **Geometry:** 75% scale HIFiRE-7 flowpath model
- **Numerics:** Hanel with O(h) spatial reconstruction
- **CFL schedule**: 1 to 2000 (conservative schedule)
- Solving **Euler** equations
- 45 million cell (**Pointwise**) unstructured grid (c\o NASA)
- Grid partitioned into 768 blocks using Eilmer4 **METIS** wrapper

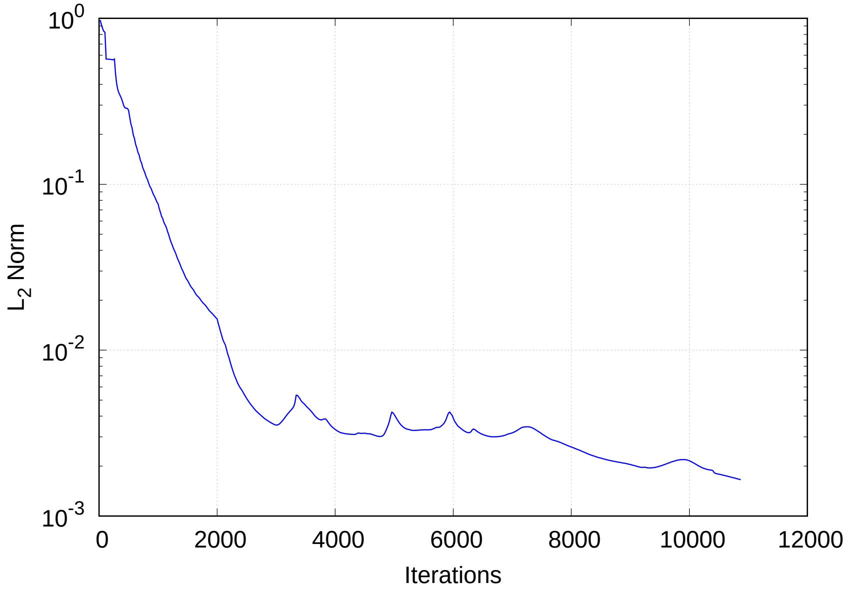


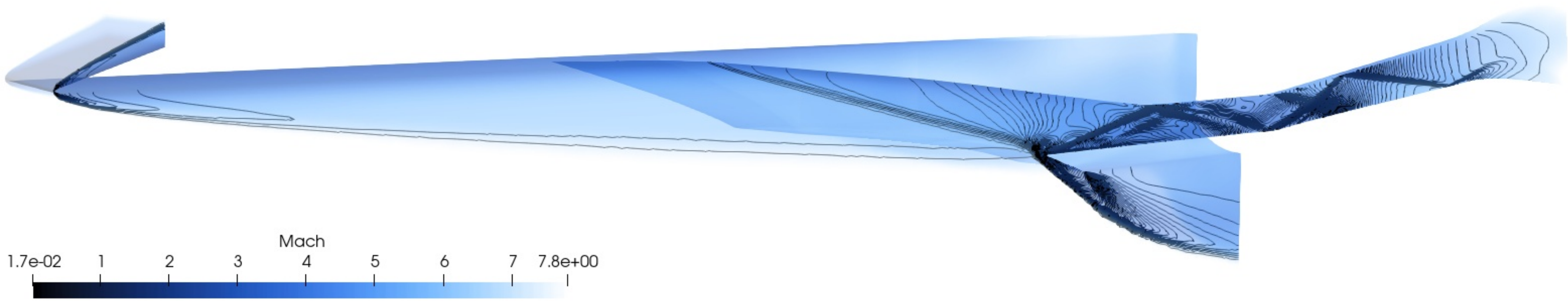

Source: Chan et al. (2014)





# **Grand Challenge:** HIFiRE-7 Simulation



# Future Work

- **Newton-Krylov acclerator**:
  -- Evaluate performance of **new preconditioners**: Jacobi, SGS, SGS relaxation
  -- Compare performance to in-house **matrix-based SGS relaxation** solver

- **Design optimization**:
  -- **Extend adjoint solver** to incorporate:
    + finite-rate chemistry
    + two-temperature modelling
  -- Explore application of optimizer to flows in **thermochemical nonequilibrium**