Automated High Fidelity Simulation

David Darmofal, Bob Haimes
Krzysztof Fidkowski, Todd Oliver, Laslo Diosady
Garrett Barter, James Modisette, David Walfisch

Massachusetts Institute of Technology
Department of Aeronautics and Astronautics

Mar 21, 2007

Outline

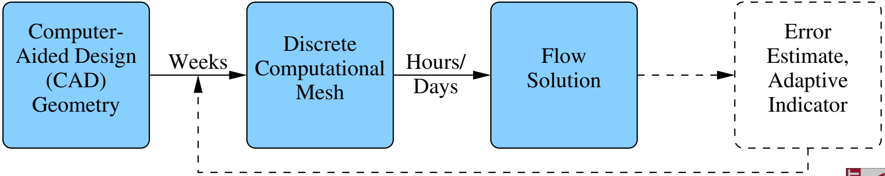
Outline

Computational Fluid Dynamics (CFD)

CFD in aerospace engineering

- Numerous codes in private and public sector
- Actively used in design and analysis
- Supplements or replaces expensive wind-tunnel tests
- Reduces design cycle time
- Allows for innovative or non-standard designs and test conditions

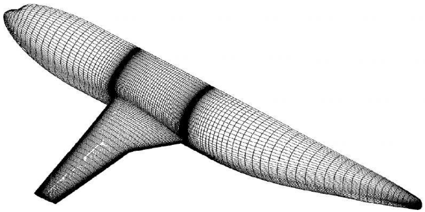
Typical Design Process



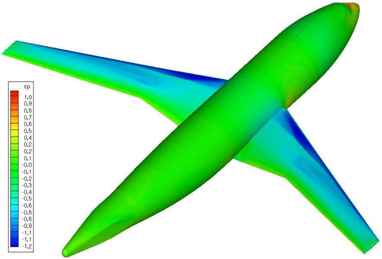
Practical Application

AIAA Drag Prediction Workshop III

- Wing-body geometry, $M = 0.75$, $C_L = 0.5$, $Re = 5 \times 10^6$
- Fine grids: ~ 25 million nodes
- Run on today's state of the art CFD codes

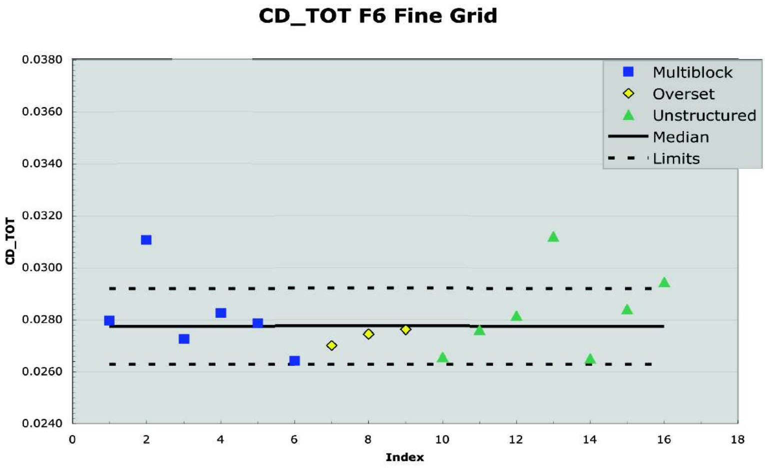


Sample surface mesh



Flow solution

AIAA Drag Prediction Workshop Results



1 drag count ($.0001\,C_D$) $\approx$ 4-8 passengers (Boeing 747-400)

CFD is Not Yet Fully Mature

Current methods are not sufficient for engineering-required accuracy

Increased computer power may not be the answer:

- Much more complex geometries exist
- New physical models (e.g. LES, DNS)
- Complex problems increasingly rely on user involvement and expertise (e.g. meshing, adaptation)
- Lack of solution quality control leads to skepticism about the reliability of CFD answers

Key Issues

- Automation
- Robustness

State of the Art in CFD

Discretization

- Finite Volume with accuracy between first and second order
- Second order Discretization requires extended stencil

Meshing

- Requires user experience and *time* (e.g. multiblock)
- Lack of robustness for complex geometries
- CAD information often neglected after initial mesh generation (Cartesian method is exception)

Error Estimation and Adaptation

- Error estimation rarely performed; codes often calibrated on test suite and then used for similar cases
- Manual or feature-based adaptation

Next-Generation CFD: One Approach

Project X

Research initiative at Aerospace Computational Design Laboratory aimed at developing the next generation CFD capability.

Goal:

Engineering accuracy in a reasonable amount of time and in an automated manner.

Key Features:

- Higher-order discretization using Discontinuous Galerkin finite element method
- Solution-based adaptivity
- Direct interface to Computer-Aided Design (CAD) models

Overview

- Initiated in MIT ACDL in 2002
- Team software development
    - ~ 6 developers at any given time
    - Automatic code archiving (CVS/Subversion)
    - Nightly build-and-test suite
- Over 100,000 lines of code: flow solver, post-processing, multiple equation sets, meshing, parallelization, etc.

DG Discretization

Compressible Navier-Stokes Equations

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot \underbrace{\mathcal{F}_i(\mathbf{u})}_{\text{Inviscid Flux}} - \nabla \cdot \underbrace{\mathcal{F}_v(\mathbf{u}, \nabla \mathbf{u})}_{\text{Viscous Flux}} = 0$$

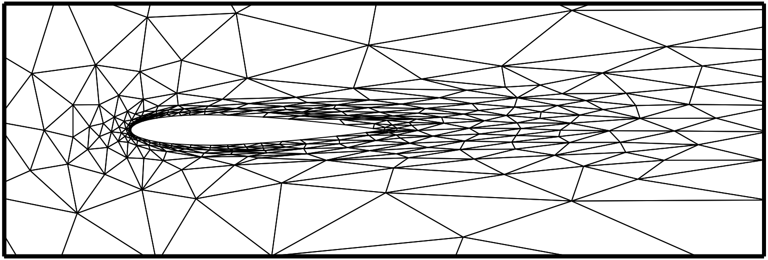
- K Conservative variables: $\mathbf{u} = [\rho, \rho u, \rho v, \rho w, \rho E]$
- Solution/test space: $\mathcal{V}_H = [V_H^p]^K$,
  $V_H^p = \{v \in L^2(\Omega) : v|_\kappa \in P^p(\kappa) : \forall \kappa \in T_H\}$
- Roe inviscid flux; 2^{nd} form of Bassi and Rebay for elliptic term
- Discrete semi-linear form: $\mathcal{R}_H(\mathbf{u}_H, \mathbf{v}_H) = 0, \quad \forall \mathbf{v}_H \in \mathcal{V}_H$

Motivating features:

- High-order accuracy, compact stencil, ease of implementation

Outline

Output-Based Adaptation



$C_D = 565.7$ counts

- How accurate is this value?
- Where is more resolution necessary to improve the accuracy?
- How should that resolution be added?

Implementation for high-order DG

- Output error estimation and localization
  - Requires solution of linear adjoint problem
  - Captures propagation effects of hyperbolic problems
  - Adapting on several key outputs often produces an adequate multi-purpose solution
- Automated anisotropic h-adaptation
  - Anisotropy detection via extension of Hessian analysis to $p > 1$
  - Goal-oriented mesh optimization
  - Re-meshing at every adaptation iteration

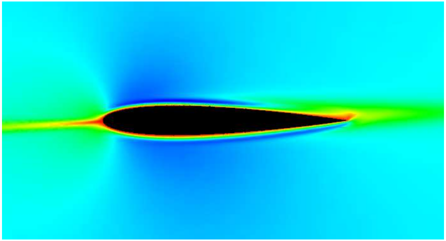
Output Error Estimation: The Adjoint

Given governing PDE and an output:

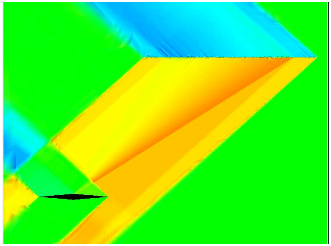
$$\nabla \cdot \mathbf{F}(\mathbf{u}) = \mathbf{g}(\mathbf{x}), \qquad \mathcal{J}(\mathbf{u}(\mathbf{g}))$$

The *adjoint*, ψ, is a Green's function relating the source, $\mathbf{g}$, to the output, $\mathcal{J}$:

$$\mathcal{J}(\mathbf{u}(\mathbf{g})) - \mathcal{J}(\mathbf{u}(\mathbf{0})) = \int_{\Omega} \psi \mathbf{g}(\mathbf{x})$$



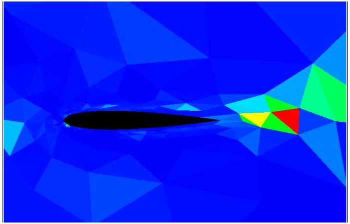
x-momentum drag adjoint: viscous case



y-momentum pres. integral adjoint: supersonic

Output Error Estimation: Local Error Indicator

Extensive previous work:
Pierce+Giles+Suli (2000),
Becker+Rannacher (2001),
Hartmann+Houston (2002),
Barth+Larson (2002)
Minor implementation differences



Error indicator for viscous case

$$\mathcal{J}(\mathbf{u}) - \mathcal{J}(\mathbf{u}_H) \approx \underbrace{\mathcal{R}_H(\mathbf{u}_H, \psi - \psi_H)}_{\text{Primal Residual}} \approx \underbrace{\mathcal{R}_H^{\psi}(\mathbf{u}_H; \mathbf{u} - \mathbf{u}_H, \psi_H)}_{\text{Adjoint Residual}}$$

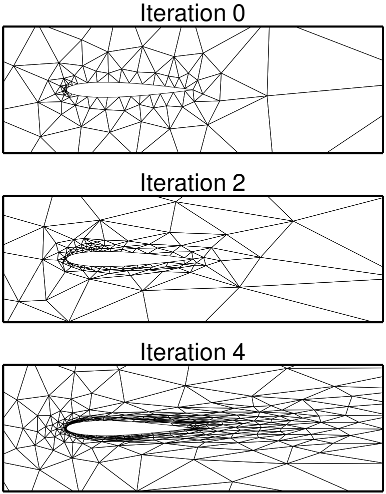
$\mathbf{u} - \mathbf{u}_H$ and $\psi - \psi_H$ estimated via reconstruction on enriched space.

Elemental Error Indicator:

$$\epsilon_\kappa = \frac{1}{2}\left(\left|\mathcal{R}_h(\mathbf{u}_H, (\psi_h - \psi_H)|_\kappa)\right| + \left|\mathcal{R}_h^{\psi}(\mathbf{u}_H; (\mathbf{u}_h - \mathbf{u}_H)|_\kappa, \psi_H)\right|\right)$$

Anisotropic Adaptation

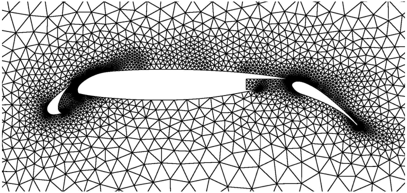
Idea: refine elements with high error; coarsen elements with low error



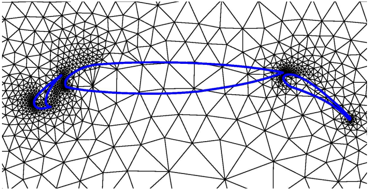
- Use *a priori* output error estimate to relate element error to size request: $\epsilon_\kappa \sim h_\kappa^{p+1}$
- Detect anisotropy by measuring $p+1$st order derivatives of a scalar quantity (e.g. Mach number)
- Optimize mesh size to meet requested tolerance and to satisfy error equidistribution
- Meshing: BAMG (anisotropic) in 2d, TetGen (isotropic) in 3d
- *Left:* NACA 0012, $M = 0.5$, $Re = 5000$, $p = 2$ adapted on drag

Outline

What Are Cut Cells?



Boundary-conforming mesh



Simplex cut-cell mesh

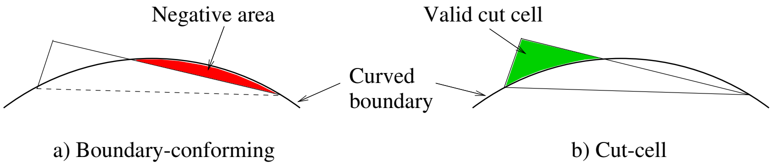
Features

- Cut-cell meshes do not conform to geometry boundary
- Solution only exists inside the computational domain
- Premise: metric-driven meshing of a simple convex volume (e.g. box) is straightforward

The Cut-Cell Advantage

Boundary-conforming mesh generation

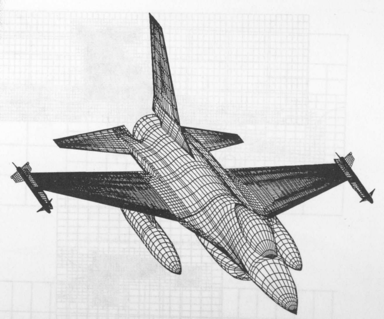
- Common bottleneck in geometry-to-solution process
- Difficult (not robust) for complex 3d geometries
- Prone to failure on curved boundaries



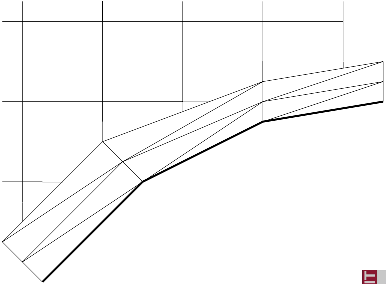
Cut-cells

- Naturally handle curved boundaries and complex geometries
- Burden of robustness transferred to computational geometry
- Fully-automated mesh generation is possible

- 1986 – Boeing's TRANAIR: FEM for 3D Full Potential Equations; adaptation on geometry and user input; integration via Stoke's theorem. Still in use today.
- 1995 – Karman's SPLITFLOW (Lockheed): 3D RANS; required prismatic boundary layer mesh; outer flow via Cartesian cut cells.



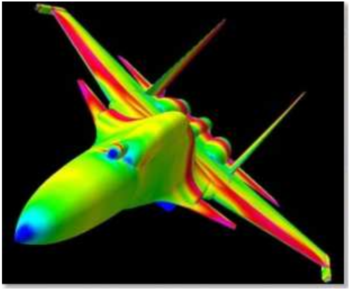
TRANAIR

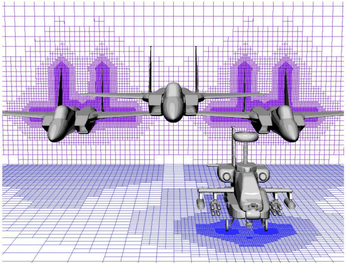


Splitflow

History: Recent Work

- 1991 to present – MGAERO by Analytical Methods, Inc: finite difference for 3D Euler; multigrid, uniform grids.
- 1993+ – Application of adaptive refinement to Cartesian method for Euler; DeZeeuw, Powell, Coirier.
- 1999 to present – Cart3d: Mike Aftosmis *et al* , NASA; finite volume for 3D Euler; adaptively refined grids.


MGAERO


Cart3d

Why Simplex Cut Cells?

Objective: A robust, automated mesher and efficient meshes

Cartesian cut-cell method

- Robust and automatic grid generation
- Inability to adapt anisotropically

Simplex (triangles, tetrahedra) cut-cell method

- Robust and automatic grid generation
- Ability to adapt anisotropically in any direction
- Not as lean as Cartesian method

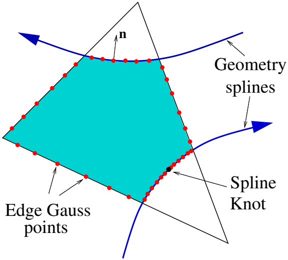
Integration

Requirements

- 1d Embedded boundary face integration (on splines)
- 1d Cut edge integration
- 2d Cut element integration

1d Edge Integrations

- Gauss points on each interval
- Normals on splines analytically available
- Currently not integrating through spline knots

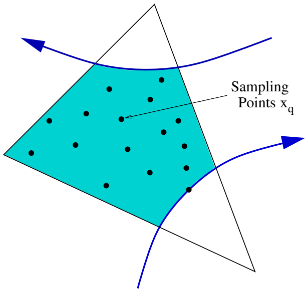


Area Integration

Goal

Pre-computed sampling points, $\mathbf{x}_q$, and weights, w_q for integrating arbitrary $f(\mathbf{x})$ to a desired order:

$$\int_{\kappa} f(\mathbf{x})d\mathbf{x} \approx \sum_q w_q f(\mathbf{x}_q)$$



Key Idea

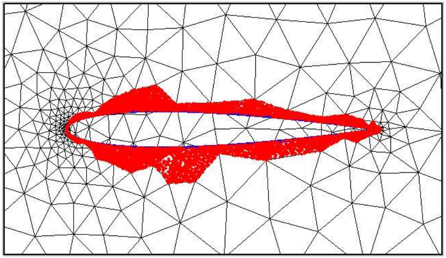
Project $f(\mathbf{x})$ onto space of high-order basis functions, $\zeta_{\mathbf{i}}(\mathbf{x})$:

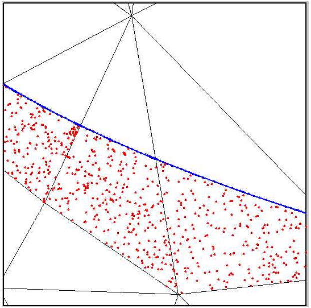
$$f(\mathbf{x}) \approx \sum_{\mathbf{i}} F_{\mathbf{i}} \zeta_{\mathbf{i}}(\mathbf{x})$$

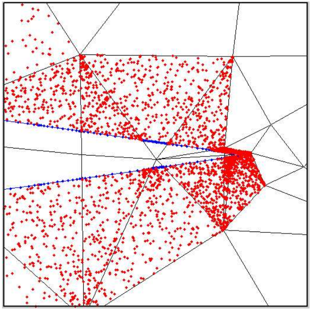
Choose $\zeta_{\mathbf{i}}(\mathbf{x})$ to allow for simple computation of $\int_{\kappa} \zeta_{\mathbf{i}}(\mathbf{x})d\mathbf{x}$.

Example: Quadrature Points

- NACA 0012
- 12 Gauss points per cut edge and spline segment
- Over 200 interior sampling points per element

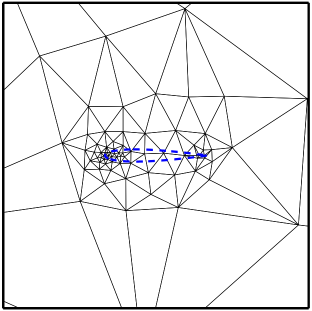




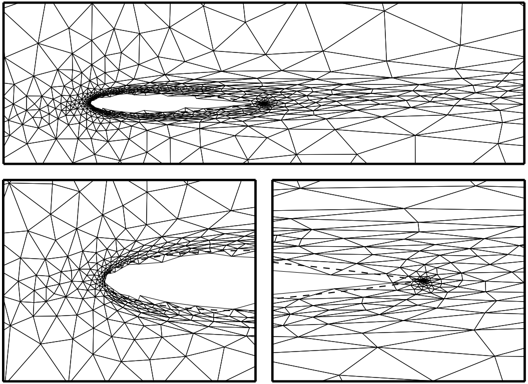


Drag Adaptation in a Viscous Case

NACA 0012, $M = 0.5$, $Re = 5000$, $\alpha = 2^o$

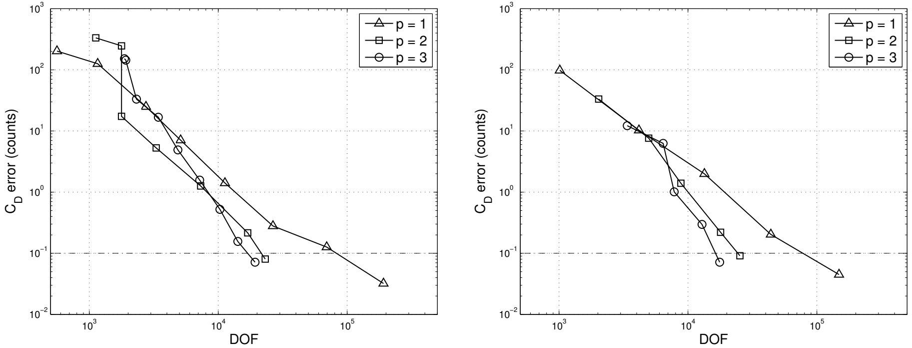


Initial mesh, adapted to geometry



$p = 3$ Adapted cut-cell mesh

Degree of freedom (DOF) vs. drag output error for $p = 1$ to $p = 3$.
Requested tolerance is 0.1 drag counts (horizontal line).
Cut-cell and boundary-conforming convergence rates are similar.



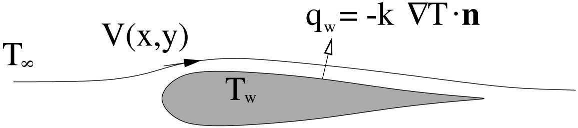
Boundary-conforming meshes

Cut-cell meshes

High Peclet Number Convection-Diffusion

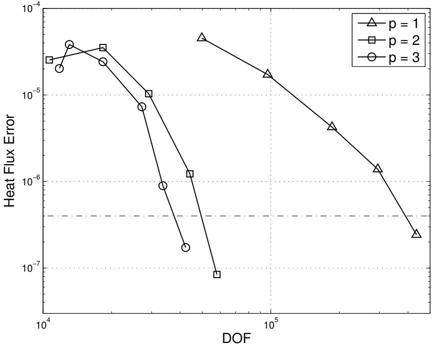
- **Purpose**: Test robustness of cut-cells + adaptation for highly anisotropic boundary layer meshes
- RANS still under development
- Can simulate behavior with convection diffusion at high *Pe*

$$\nabla \cdot (\mathbf{V}T) - \nabla \cdot (k \nabla T) = 0, \qquad Pe = \frac{V_\infty L}{k}$$



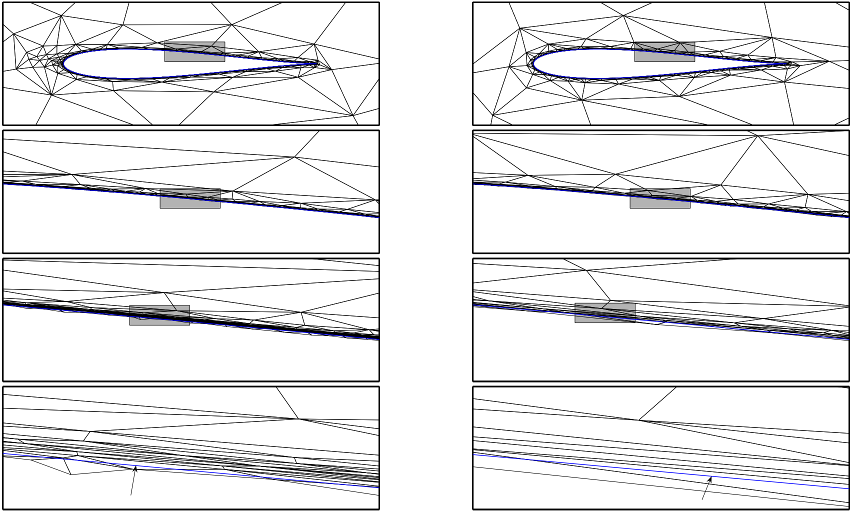
$Pe = 4 \times 10^8$: Error Convergence

- $Pe = 4 \times 10^8$ simulates turbulent inner layer at $Re \sim 10^6$
- Heat flux output: $\mathcal{J} = \int_{\text{airfoil}} q_w ds$
- Error tolerance is 1% of true heat flux



- $p = 1$ requires a factor of 10 more degrees of freedom than $p = 3$
- $p = 2$ performance is similar to $p = 3$

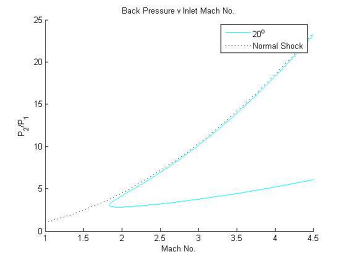
$Pe = 4 \times 10^8$: Adapted Meshes



$p = 2$ $\quad$ $p = 3$

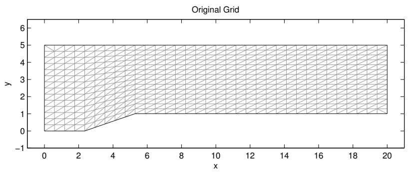
Oblique Shocks in an Inlet

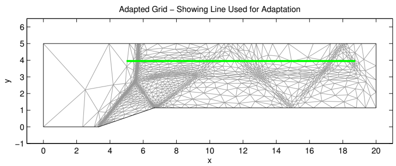
- For a 16.120, Compressible Flows, class project ProjectX was used to study the affect of back pressure on the primary oblique shock which forms in an inlet
- Experimental and Analytical results show that for many mach numbers and compression ramp angles there are two oblique shock solutions for the flow

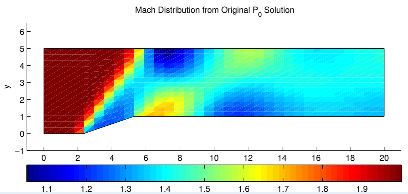


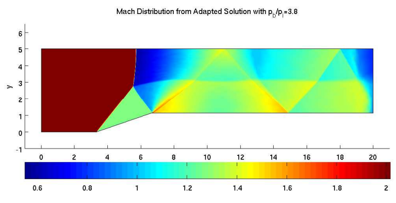
  - The purpose of the project was to study the transition between the week oblique shock, the strong oblique shock and finally the normal shock
  - A rough structured grid was first created in Matlab and then using anisotropic grid adaptation the mesh was refined

Oblique Shocks in an Inlet









Outline

Extension to Three Dimensions

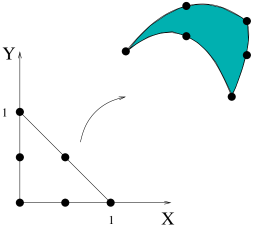
- Cut-cell mesh generation becomes more difficult:
  - Geometry representation is not as straightforward as in 2D
  - Harder intersection problem: volume-surface instead of area-line
  - Integration rules needed on geometry surface, cut faces, and cut elements
- However, generating 3D boundary-conforming meshes is much more difficult compared to 2D:
  - Meshing around intricate 3D geometries is not trivial
  - No robust automated technique for anisotropic adaptation
- Cut cell difficulties are related to computational geometry. An automated meshing technique is possible.

Geometry Definition and Intersection

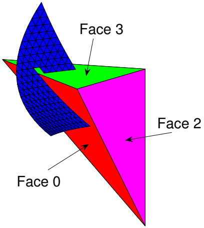
- Quadratic patches, 6 nodes per patch
- Patch surface ($\mathbf{x}$) is given analytically:

$$\mathbf{x} = \sum_j \phi(\mathbf{X})_j \mathbf{x}_j,$$

  $\mathbf{X} = [X, Y]$: patch ref coords
- Watertight representation (no holes)
- Intersection between a plane and a quadratic patch is a conic section (ellipse, hyperbola, etc.) in (X, Y)
- Robustness of cutting algorithm relies on robustness of conic-conic intersections



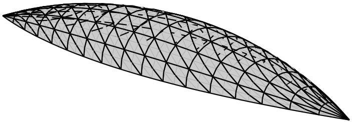
Patch reference space



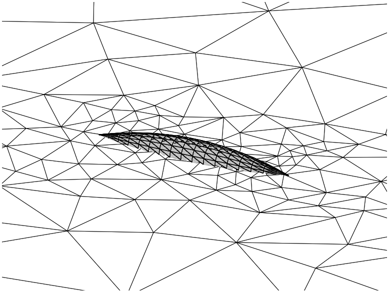
Tet-patch intersection

Flow Around a Football

- Inviscid, $M = 0.3$ flow around a body of revolution
- Model half the geometry



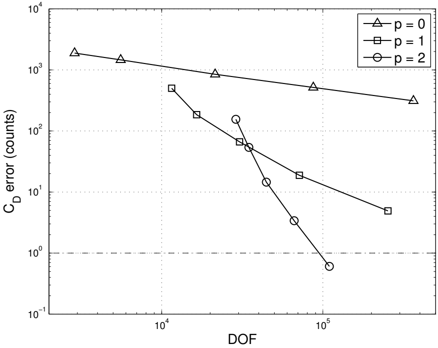
Surface representation: 256 quadratic patches



Initial background mesh: 576 elements

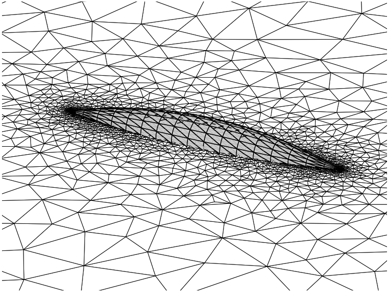
Football: Error Convergence

- Isotropic adaptation on drag, with error tolerance of 1 drag count
- C_D measured using frontal cross-sectional area

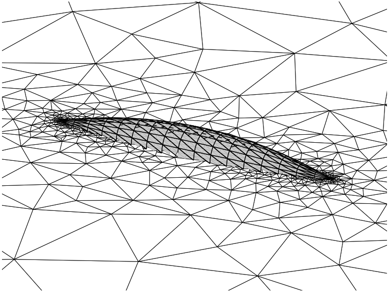


- $p = 0$ and $p = 1$ not converged due to memory limitations (serial runs)
- $p = 0$ is not practical for accurate computation
- $p = 2$ convergence is much better than $p = 1$

Football: Adapted Meshes

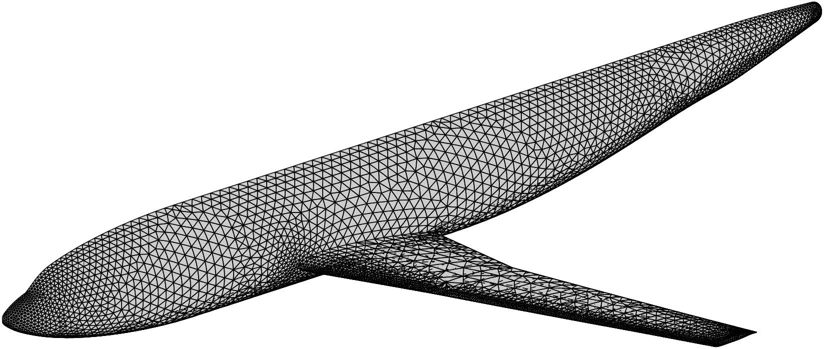


$p = 1$: 66304 elements: error = 4.0 drag counts



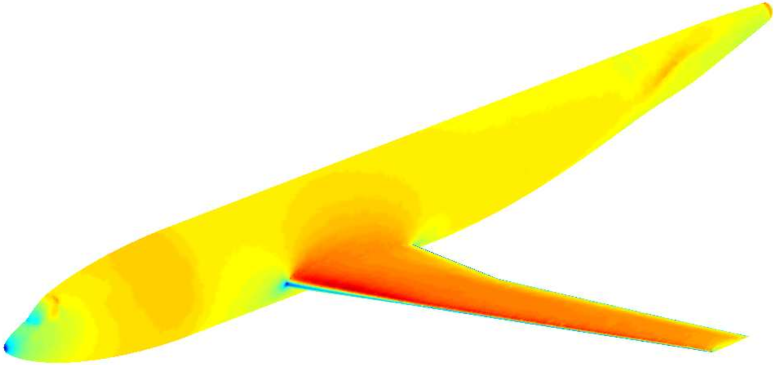
$p = 2$: 11354 elements: error = 0.6 drag counts

Wing-Body Geometry



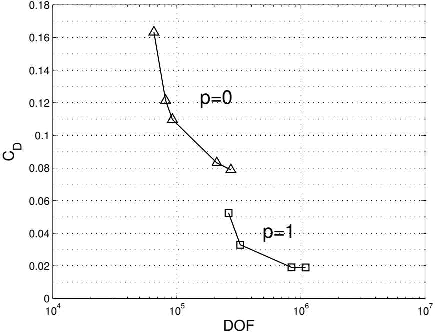
- Geometry from Drag Prediction Workshop
- 10,000 quadratic surface patches
- 300,000 tetrahedron mesh from isotropically adaptation on drag

Wing-Body Geometry Solution

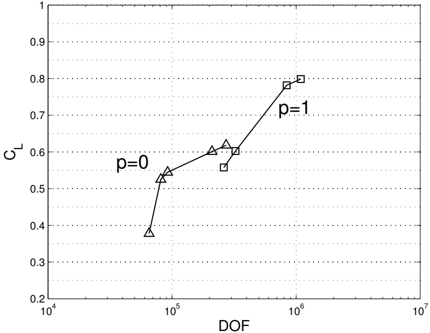


- Inviscid $M_\infty = 0.1$ flow, $p = 1$ interpolation
- Mach number contours shown

Wing-Body Drag and Lift Comparison



Drag



Lift

- $p = 1$ solution achieve higher accuracy in fewer DOF's

Outline

Unsteady Adaptation

Objective

A robust, automated adaptive method to control error in engineering outputs for simulations of essentially steady, aerodynamic flows

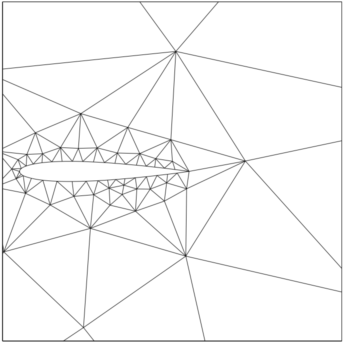
Typical Output-Based Adaptation Framework

- Generate initial triangulation of domain and initial solution guess
- While error > err tol
  - Compute steady state solution
  - Compute error estimate using steady state solution
  - If error < err tol, quit. Otherwise, adapt mesh.

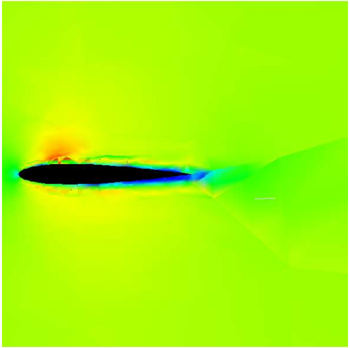
Problems

- Algorithm fails if flow solver cannot find steady state solution
- Unrealistic robustness requirement placed on flow solver
- In situations where flow as small unsteadiness, (linear) adjoint problem is often unstable.

Steady Adaptation Failure Example



Initial mesh (zoom)

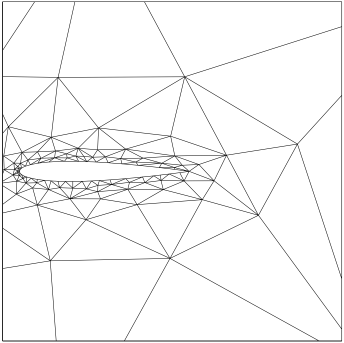


Mach Number, $p = 2$, Initial mesh

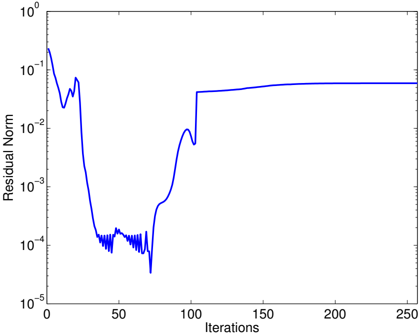
Case Information

- Laminar flow over NACA 0012
- $M_\infty = 0.5$, $Re = 2 \times 10^5$, $\alpha = 0^o$
- Adaptation output = Drag
- Start from 165 element mesh

Steady Adaptation Failure Example



Failure mesh (zoom)



Residual history on failure mesh

Failure

- Solver cannot find steady state solution after 2 adaptation iterations
- Adaptation cannot continue without steady state solution

Proposed Solution: Unsteady Adaptation

Idea

- Add time dependence to equations of interest
- Define output to be time-averaged output over a single time step,

$$\mathcal{J}(u) = \frac{1}{\Delta t} \int_{t}^{t+\Delta t} J(u,t)\, dt.$$

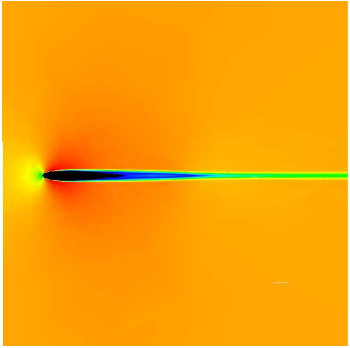
- Raise Δt as solution converges to steady state
- As $\Delta t \to \infty$, error estimate is identical to steady state case.

Proposed Solution: Unsteady Adaptation

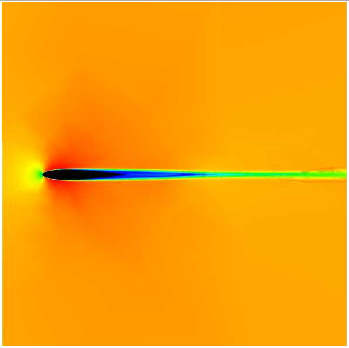
Basic Algorithm

- Generate initial triangulation, time step, and primal state
- While (steady residual > res tol) and (error > err tol),
  - March primal/adjoint forward/backward in time
  - Estimate error for current time step
  - Compute new mesh, $\mathcal{T}_{new}$, based on error estimate and tolerance
  - Modify Δt based on maintaining a physically realizable update
  - Continue

Test Case: Laminar Airfoil



Mach Number, $p = 2$, Final mesh

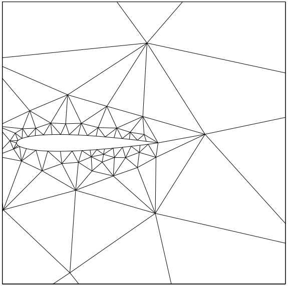


Mach Number, $p = 3$, Final mesh

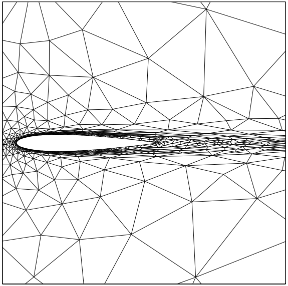
Case Information

- Solve compressible Navier-Stokes
- NACA 0012
- $M_\infty = 0.5$, $Re = 2 \times 10^5$, $\alpha = 0^o$
- Adaptation output = Drag

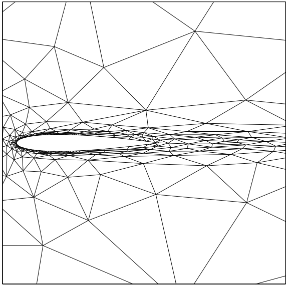
Initial and Final Meshes



Initial mesh,
165 elements

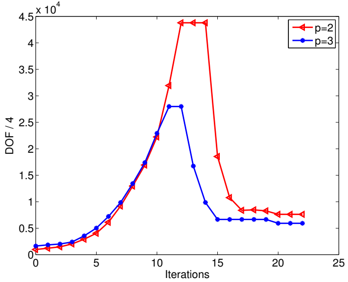


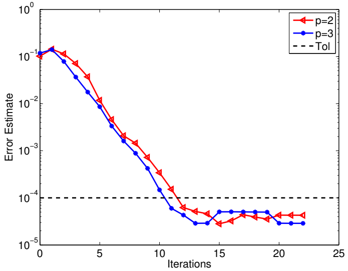
Final mesh, $p = 2$,
1272 elements



Final mesh, $p = 3$,
594 elements

Adaptation History





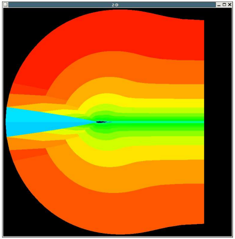
Observations

- Require many more DOFs during run than required for final steady state
  - $p = 2$: $N_{DOF,max} / N_{DOF,final}$ = 5.74
  - $p = 3$: $N_{DOF,max} / N_{DOF,final}$ = 4.71
- Need to adapt even when error tolerance met
  - Must be able to bring N_{DOF} back down

Outline

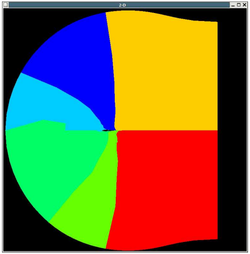
1. Introduction
   - Motivation
   - Approach
2. Output-Based Adaptation
3. Cut Cells in Two Dimensions
4. Cut Cells in Three Dimensions
5. Improving Robustness through Unsteady Adaptation
6. **Other Work in Progress**
7. Plans
8. Back-up Slides

On-going Work: Parallel solution algorithms

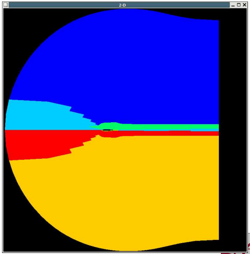


Lines of Maximum Coupling

- Lines of maximum coupling are important for good solver performance
- Repartitioning according to lines ensures lines are the same in both parallel and serial cases

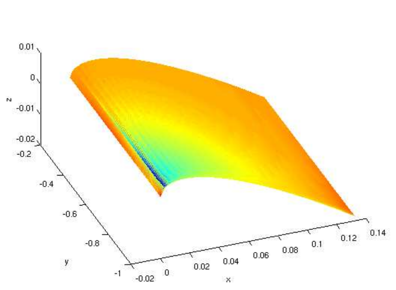


Basic Partitioning

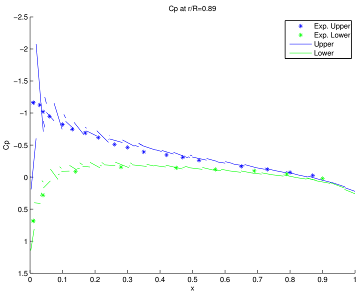


Line Repartitioning

On-going Work: Rotorcraft Applications

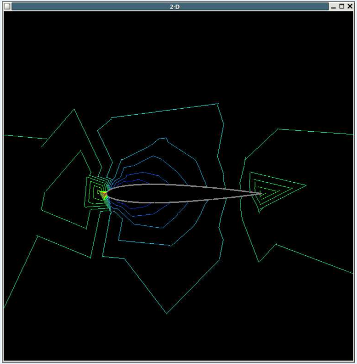


C_p Distribution over the Top Rotor Surface

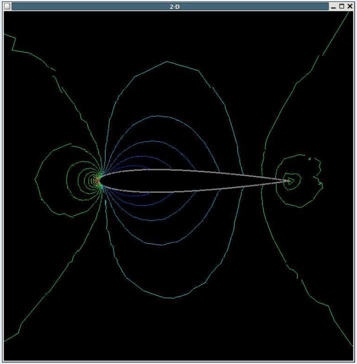


C_p Distribution at $\frac{r}{R} = 0.89$ Compared to Experimental Data

On-going Work: Others



Visualization on original mesh



Adaptive visualization

- Higher-order visualization (David W.)
- Hypersonic applications (Garrett, Loretta, NASA JSC)
- Supersonic aircraft aerodynamic design (MIT, Boeing, NASA Langley)

Outline

Plans for 2007

- Adaptive Reynolds-averaged Navier-Stokes in 2-D
- Three-dimensional anisotropic mesh adaptation
- Assist Boeing in testing and using 3-D Euler capability
- NASA-funded Boeing-MIT collaboration to apply methodology to supersonic aircraft design
- Improve computational speed

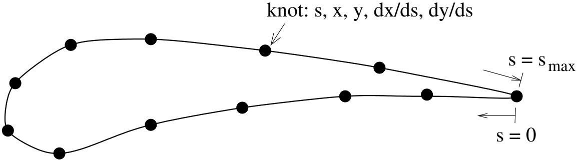
Questions?

Outline

2d Geometry Definition

Cubic splines

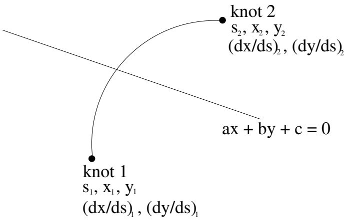
- Efficient treatment of curved boundaries
- Slope continuity at spline knots
- Corners possible with multiple splines

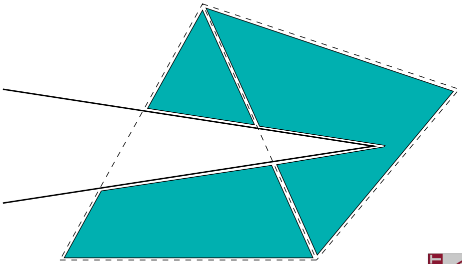


Intersection Problem

Implementation

- Analytic intersections between splines and edges: cubic equation
- Split triangles treated as separate elements
- Triangles completely contained inside geometry removed from mesh structure
- Integration rules on cut cells/edges calculated in preprocessing





Set $\zeta_{\mathbf{i}} \equiv \partial_k G_{\mathbf{i}k}$ and use the divergence theorem:

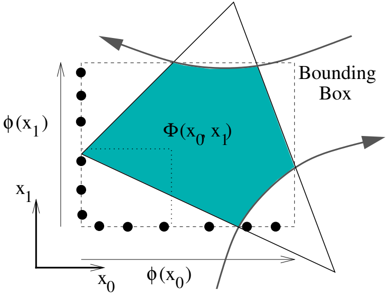
$$\int_\kappa \zeta_{\mathbf{i}} d\mathbf{x} = \int_\kappa \partial_k G_{\mathbf{i}k} d\mathbf{x} = \int_{\partial\kappa} G_{\mathbf{i}k} n_k ds$$

Integrals over $\partial\kappa$ using 1d edge formulas.

$$G_{\mathbf{i}k} = x_k \Phi_{\mathbf{i}}(\mathbf{x}), \quad \Phi_{\mathbf{i}}(\mathbf{x}) = \prod_k \phi_{i_k}(x_k),$$

$$\mathbf{x} = [x_k], \quad \mathbf{i} = [i_k]$$

- ϕ_i are 1d Lagrange basis functions with nodes at Gauss points on element bounding box.



Projection

Projection $f(\mathbf{x}) \approx \sum_{\mathbf{i}} F_{\mathbf{i}} \zeta_{\mathbf{i}}(\mathbf{x})$ minimizes the least-squares error:

$$E^2 = \sum_q \left[\sum_{\mathbf{i}} F_{\mathbf{i}} \zeta_{\mathbf{i}}(\mathbf{x}_q) - f(\mathbf{x}_q) \right]^2$$

The solution vector, $F_{\mathbf{i}}$, is found using QR factorization:

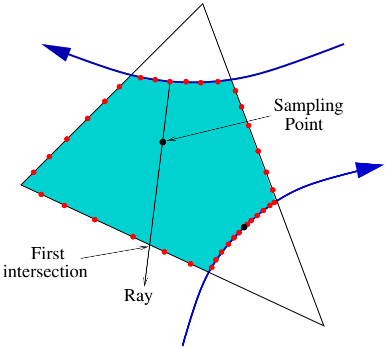
$$F_{\mathbf{i}} = (R^{-1})_{\mathbf{ij}} (Q^T)_{\mathbf{j}q} f(\mathbf{x}_q), \quad \text{where} \quad \zeta_{\mathbf{i}}(\mathbf{x}_q) = Q_{q\mathbf{j}} R_{\mathbf{ji}}.$$

Integrating over κ leads to an expression for the quadrature weights:

$$\int_\kappa f(\mathbf{x}) d\mathbf{x} \approx \sum_{\mathbf{i}} F_{\mathbf{i}} \int_\kappa \zeta_{\mathbf{i}}(\mathbf{x}) d\mathbf{x} = \sum_q f(\mathbf{x}_q) \underbrace{Q_{q\mathbf{j}} (R^{-T})_{\mathbf{ji}} \int_\kappa \zeta_{\mathbf{i}}(\mathbf{x}) d\mathbf{x}}_{w_q}$$

Sampling Point Selection

- $\mathbf{x}_q$ must lie inside the cut cell to keep the integrand evaluations physical for non-linear problems
- Choosing $\mathbf{x}_q$ randomly via ray-casting:



- Clusters of sampling points are undesirable in terms of QR conditioning $\Rightarrow$ use *oversampling*

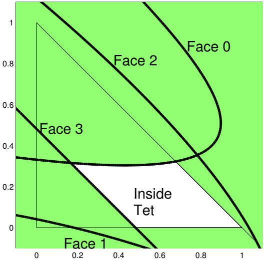
Integration

Requirements

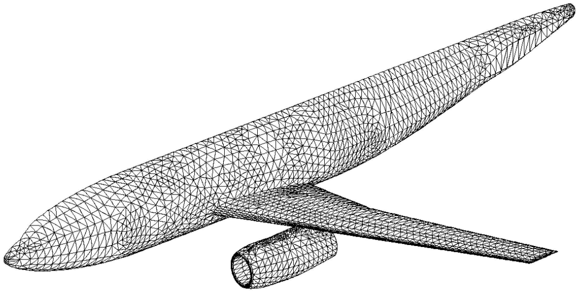
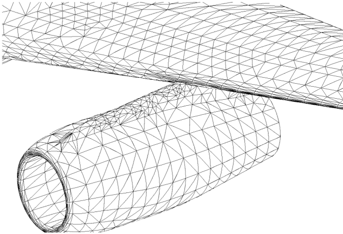
- 2d Embedded boundary face integration (on patches)
- 2d Cut face integration
- 3d Cut element integration

Methodology

- Gauss points on 1d edges of 2d faces
- Sampling point speckling for face integration (as in 2d)
- 3d extension of point speckling for cut elements

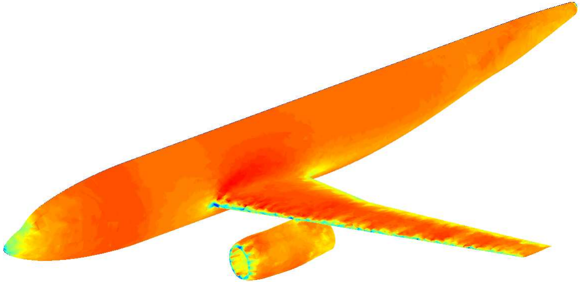


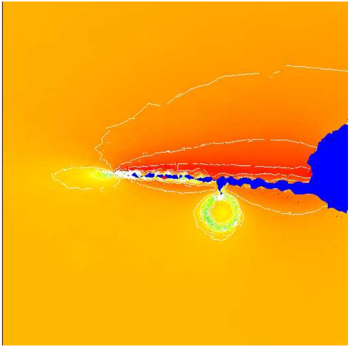
Wing-Body-Nacelle

- Geometry from Drag Prediction Workshop II
- 10,000 quadratic surface patches
- Trial run on geometry-adapted background mesh (150,000 tets)

Preliminary Flow Solution





- Inviscid $M_\infty = 0.08$ flow, $p = 1$ interpolation
- Mach number contours shown

Recommended Solver Parameters

Solver

- **Solver = Newton** Default nonlinear solver is Newton's method. Available solvers include Block, Line, VMultigrid, FMultigrid, Newton, Adjoint
- **LinearSolver = GMRES** Default linear solver for each Newton step is preconditioned GMRES

GMRES

- **Preconditioner = ILU** Default preconditioner is block incomplete LU factorization. Available preconditioners are Block, Line, ILU, and MG. ILU is recommended for $p = 0$ and strong shock cases ($M > 2$). MG is recommended for $p \geq 1$.
- **GMRES_MaxInner** Number of GMRES vectors, ideally as large as possible without running out of memory.
- **GMRES_MaxOuter** Number of GMRES restarts

Recommended Solver Parameters

Multigrid

- **Smoother = ILU** Available Multigrid smoothers include Block, Line and ILU.
- **CoarseOrder = 0** Polynomial order of coarsest multigrid level
- **VDownIter = 1** Number of pre-smoothing iterations
- **VUpIter = 1** Number of post-smoothing iterations
- **VCoarseIter = 5** Number of smoothing iterations on coarsest level

Other useful Solver Flags

- **ResidualCriterionFlag = True** Linear System in only solved to a limited tolerance each Newton iteration when the nonlinear residual is still large.
- **RepartitionInterval = 20** Grid repartitioning according to lines to improve solver performance for Line and ILU. Default is -1 (no repartitioning) since repartitioning does not yet work for cut cells.

Error Estimation

Basics

- In terms of elemental residuals [Becker and Rannacher, 2001],

$$\mathcal{J}(u) - \mathcal{J}(u_h) = -\sum_{\kappa \in T_h} r_h(u_h, (\psi - \psi_h)|_\kappa) = -\sum_{\kappa \in T_h} r_h^*(\psi_h, (u - u_h)|_\kappa)$$

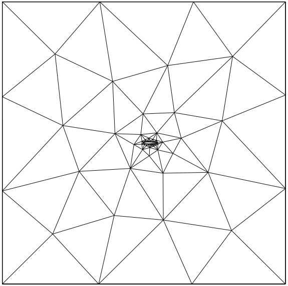
- In general, u and ψ are unknown
- Let $\tilde{u}$ and $\tilde{\psi}$ be approximations of u and ψ based on u_h and ψ_h
- Choose error indicator to be

$$\epsilon = \sum_\kappa \epsilon_\kappa = \sum_{\kappa \in T_h} \frac{1}{2} |r_h(u_h, (\tilde{\psi} - \psi_h)|_\kappa)| + \frac{1}{2} |r_h^*(\psi_h, (\tilde{u} - u_h)|_\kappa)|$$

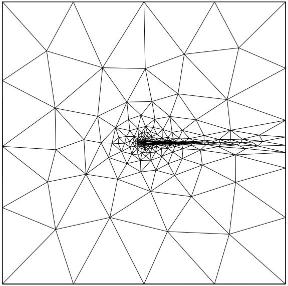
Key Point

Error estimate formally the same for steady and unsteady problems

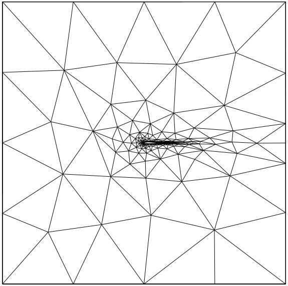
Initial and Final Meshes



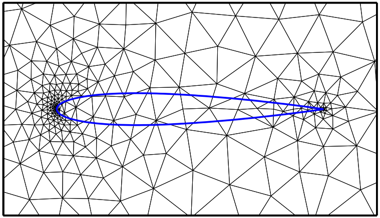
Initial mesh,
165 elements



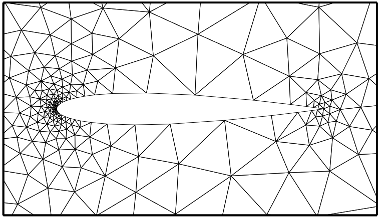
Final mesh, $p = 2$,
1272 elements



Final mesh, $p = 3$,
594 elements



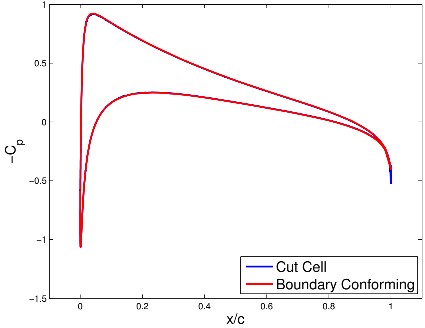
Cut-cell mesh



Boundary-conforming mesh

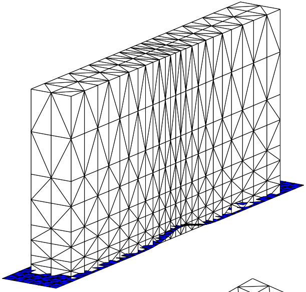
- $M = 0.5$, $\alpha = 3^o$
- $p = 2$ interpolation

C_p comparison

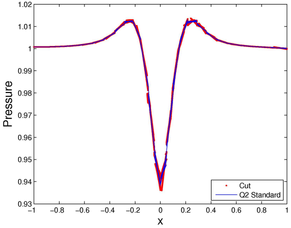


Example: Flow Solution

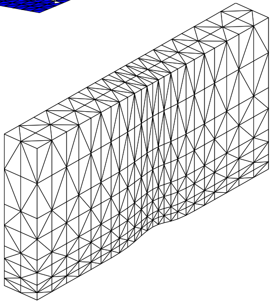
Cut-cell mesh:



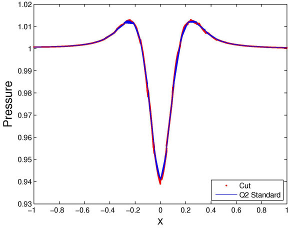
$p = 1$



Boundary-conforming mesh:



$p = 2$



Parallel Status

Rigorously Tested Features

- **Euler and Navier-Stokes Equations** Both Euler and Navier-Stokes equations sets have been thoroughly tested in parallel
- **Shock Capturing** All 2D shock capturing techniques have been thoroughly tested in parallel.
- **Linear Solvers** All linear solver/preconditioner and multigrid/smoothers combinations work reliably in parallel.

Parallel Status

Moderately Tested Features

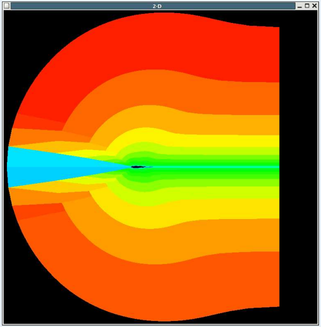
- **Cut Cells** Cut cells appear to be working correctly in serial. Cutting is performed serially on the root processor, possibly limiting the size of mesh which may be used.
- **Adaptation** 2D Adaptation has been moderately tested and appears to be working correctly in parallel. Grid adaptation is performed by calling BAMG from the root processor.
- **RANS** 2D RANS features available in serial appear to be working correctly in parallel
- **Grid Repartitioning** Grid repartitioning according to lines for Line and ILU preconditioners has been moderately tested in parallel.
- **Nonlinear Solvers** have been moderately tested
- **Petsc Solvers** The PETSc version of the code, ProjectXPetsc, using the PETSc matrix, vector and linear solver formats has been somewhat tested serially and in parallel.

Parallel Status

Un-Tested or not working features

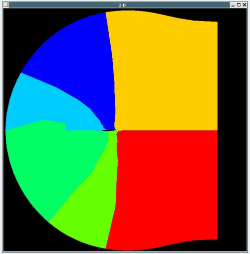
- **Grid Repartitioning for Cut Cells** Grid repartitioning for cut cells to have yet to be implemented. Hopefully this will be implemented in the next couple of weeks.
- **Periodic BCs** Periodic boundary conditions have been recently broken in parallel. This bug should be tracked down and fixed within the next couple of days.
- **Lean Nonlinear Solvers** Lean-Block and Lean-Line solvers will not be implemented in parallel any time in the near future.

Line Repartitioning

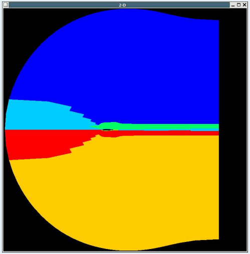


Lines of Maximum Coupling

- Lines of maximum coupling are important for good solver performance
- Repartitioning according to lines ensures lines are the same in both parallel and serial cases

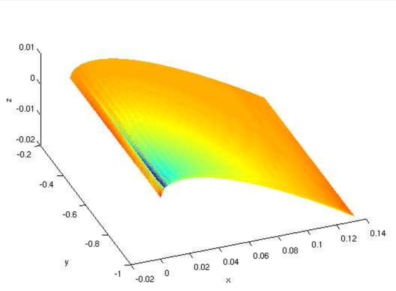


Basic Partitioning

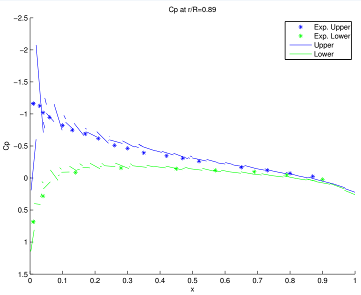


Line Repartitioning

Helicopter Rotor in Hover



C_p Distribution over the Top Rotor Surface

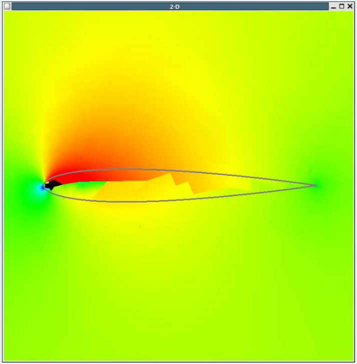


C_p Distribution at $\frac{r}{R} = 0.89$ Compared to Experimental Data

- Calculation uses rotating relative reference frame and periodic boundaries
- Current implementation is based on a Q1 embedded boundary
- Future work consists of getting similar results at a lower cost using 3-D cut cells

Cutcell visualization

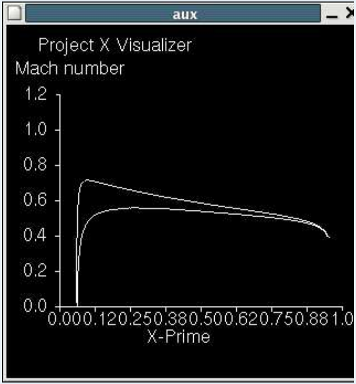
Cutcell solution



- Solution is visualized everywhere on the mesh.
- Embedded surface is added, in order to see the object boundaries.
- Solution inside the body can be neglected.

Cutcell visualization

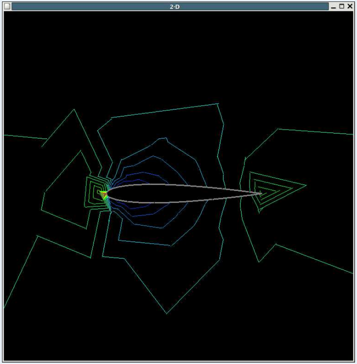
Surface



- Solution distribution on the surface can be plotted.
- Surface distribution plot uses values from quadrature points and interpolates them.

Visualization-Error based adaptation

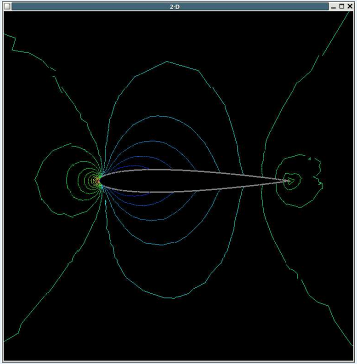
Visualization on non adapted grid



- Visualization uses linear interpolation between vertices.
- For Higher-Order solutions the grid must be adapted for visualization in order to reduce the error introduced from the linear interpolation.
- User can define maximum visualization error.

Visualization-Error based adaptation

Visualization on adapted grid



- Grid only adapted in locations where the error was greater than the user defined value. (.005)
- Isolines are smoother and the visualization is a more accurate representation of the actual P5 solution.

Improvements to the State of the Art

Discretization

- High-order accurate Finite Element, with compact stencil
- One option: Discontinuous Galerkin (DG)

Meshing

- Robust and automated to reduce design cycle time
- Direct link to CAD
- One option: Cut-cells

Error Estimation and Adaptation

- Output error estimation for confidence in engineering values
- Anisotropic mesh adaptation for practical resolution of boundary layers and wakes.
- Automated adaptive solution method