Output-based Adaptive Methods for Large-Scale Aerodynamics Simulations

Krzysztof Fidkowski
University of Michigan, USA

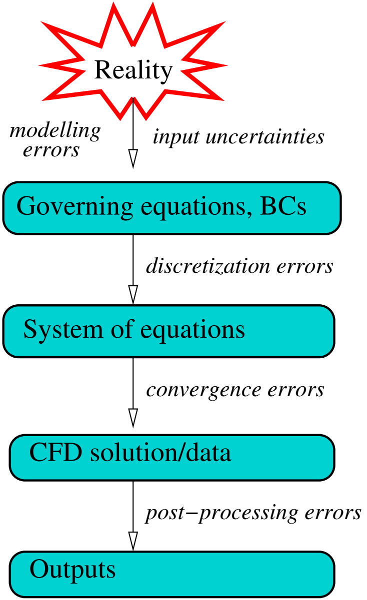
Jameson, Roe, van-Leer Symosium
San Diego, CA

June 22-23, 2013

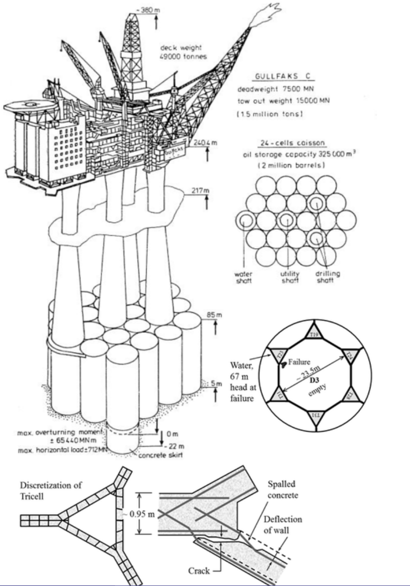
Outline

Outline

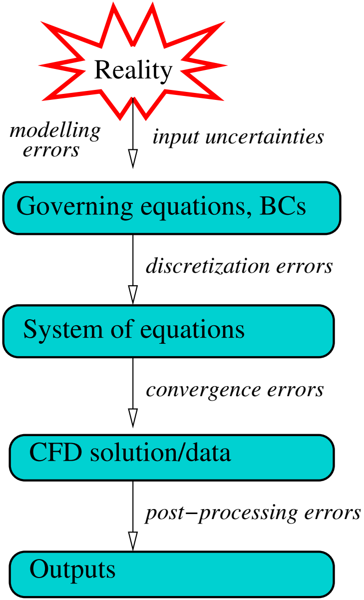
The computer is not always right



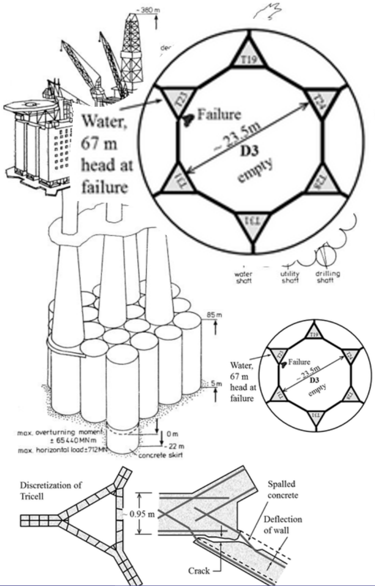
Sleipner Platform A Failure (1991)



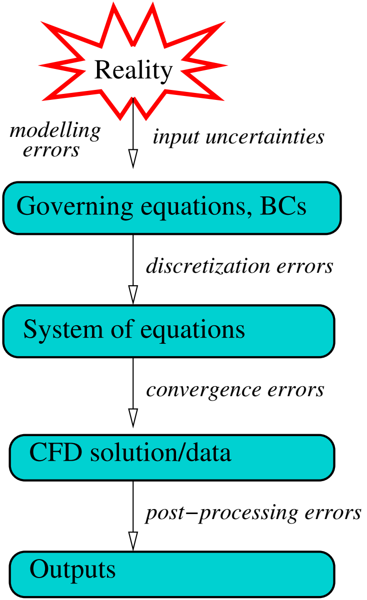
The computer is not always right



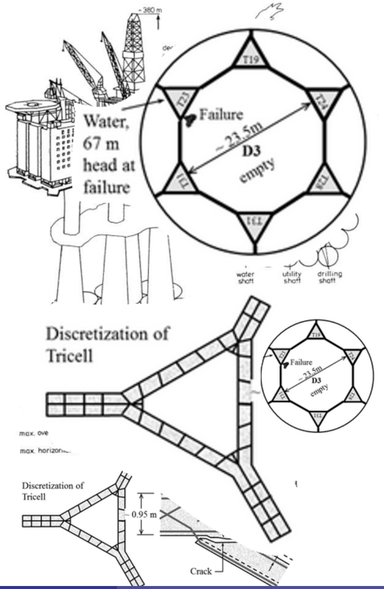
Sleipner Platform A Failure (1991)



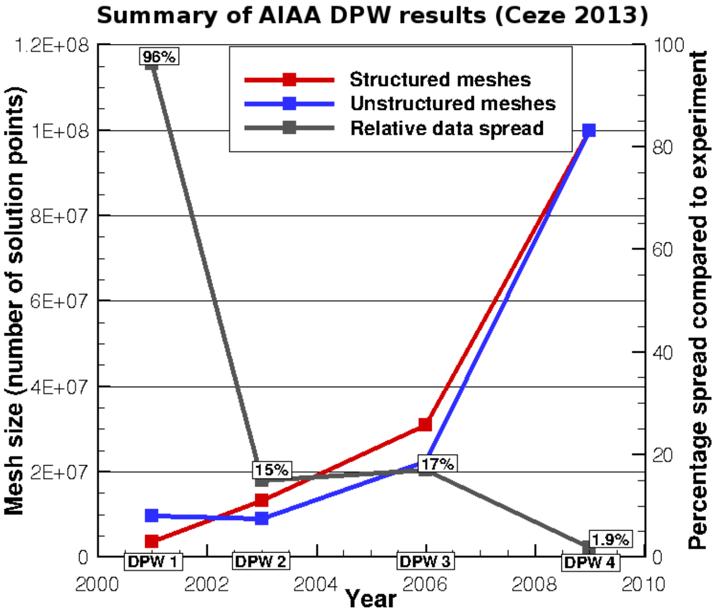
The computer is not always right



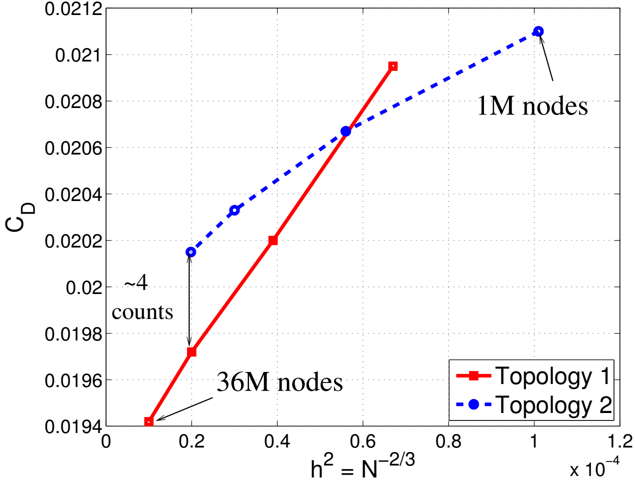
Sleipner Platform A Failure (1991)



Discretization errors are important



Uniform refinement can be misleading

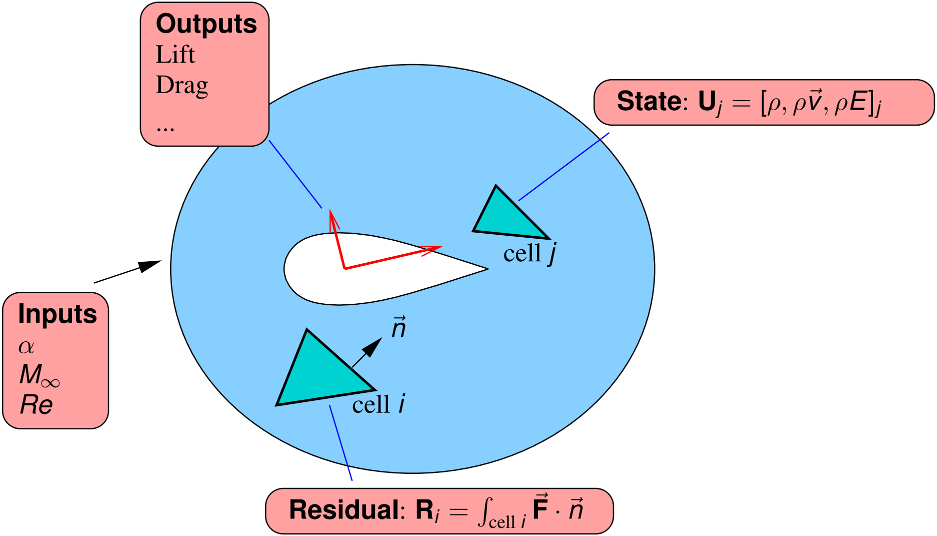


DPW III wing-alone case: $M_\infty = 0.76$, $Re = 5 \times 10^6$. Same code but two different initial meshes (Mavriplis, 2007).

Outline

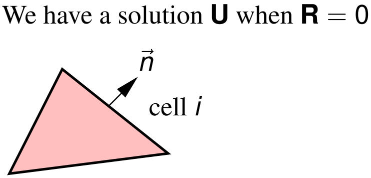
Some definitions

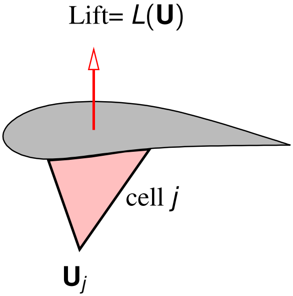
Consider flow over an airfoil:



Output sensitivity to residuals: the adjoint

The lift adjoint $\mathbf{\Psi}_i$ is the sensitivity of lift to residual sources in cell i.

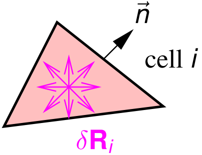




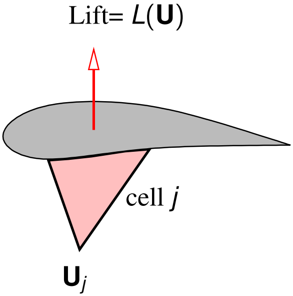
Output sensitivity to residuals: the adjoint

The lift adjoint $\boldsymbol{\Psi}_i$ is the sensitivity of lift to residual sources in cell i.

We have a solution $\mathbf{U}$ when $\mathbf{R} = 0$

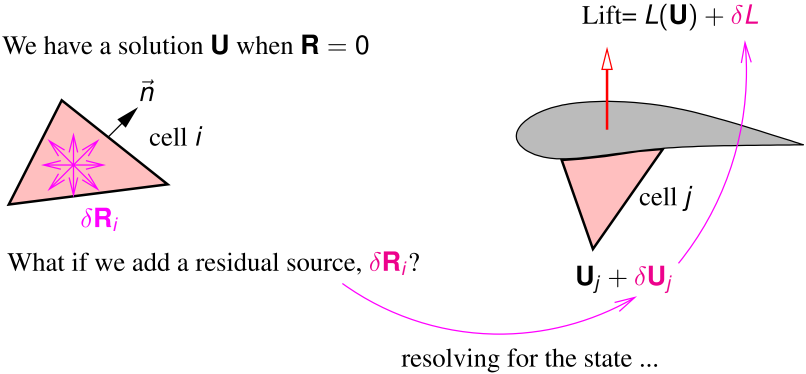


What if we add a residual source, $\delta\mathbf{R}_i$?



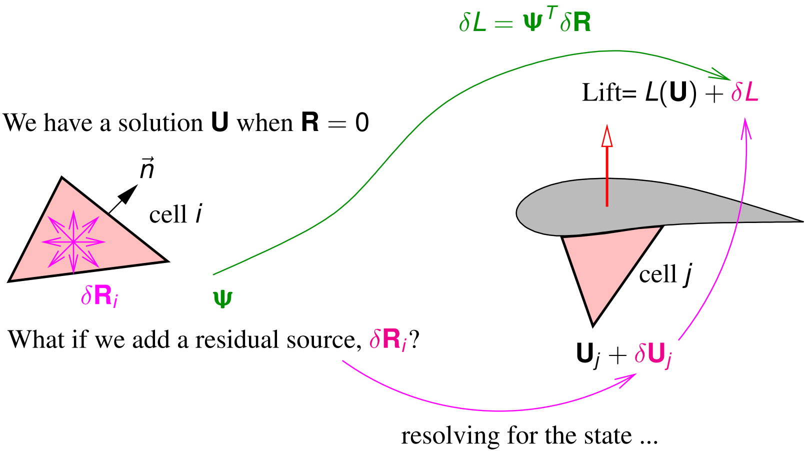
Output sensitivity to residuals: the adjoint

The lift adjoint $\mathbf{\Psi}_i$ is the sensitivity of lift to residual sources in cell i.

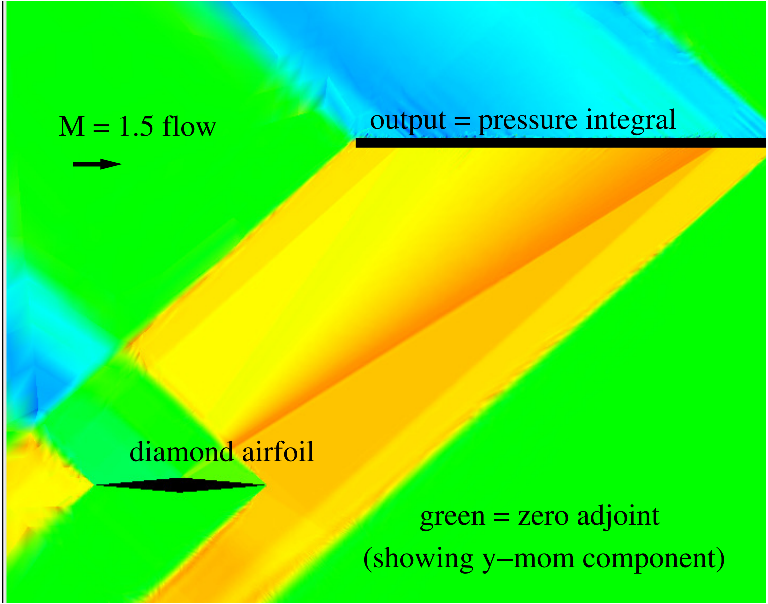


Output sensitivity to residuals: the adjoint

The lift adjoint $\boldsymbol{\Psi}_i$ is the sensitivity of lift to residual sources in cell i.

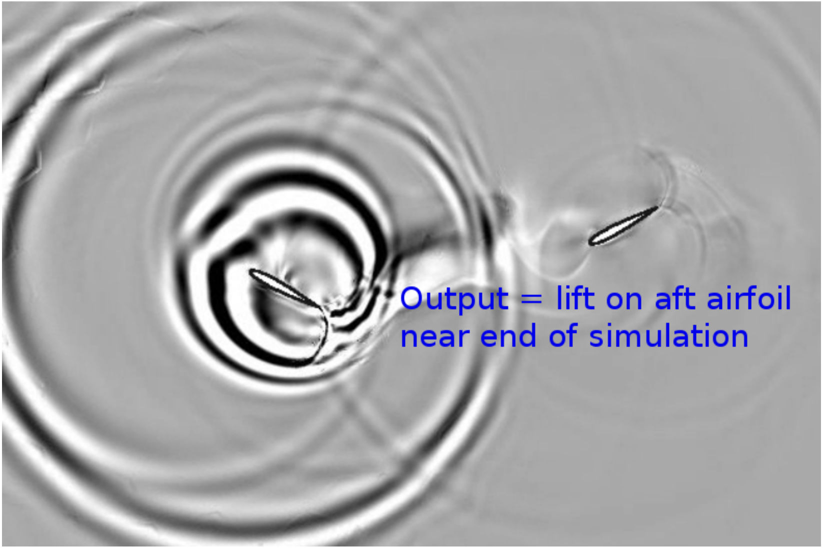


Sample steady adjoint solution



Sample unsteady adjoint solution

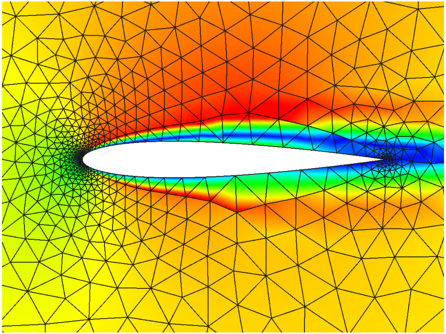
Two pitching+plunging airfoils in low-Re flow



Where do the residuals come from?

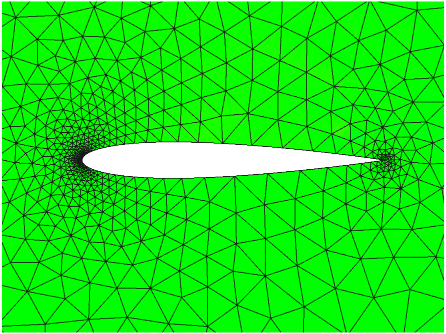
- A finer mesh or higher order discretization can uncover residuals in a converged solution
- Example from DG FEM:

Coarse space state, $\mathbf{U}_H$



$p_H = 1$

Coarse space residual, $\mathbf{R}_H(\mathbf{U}_H)$

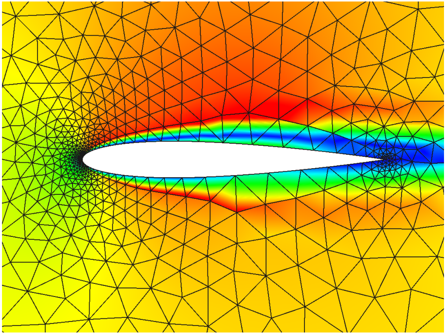


Zero as expected

Where do the residuals come from?

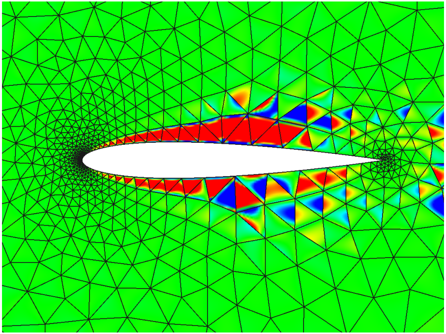
- A finer mesh or higher order discretization can uncover residuals in a converged solution
- Example from DG FEM:

Injected state, $\mathbf{U}_h^H$



$p_h = 2$

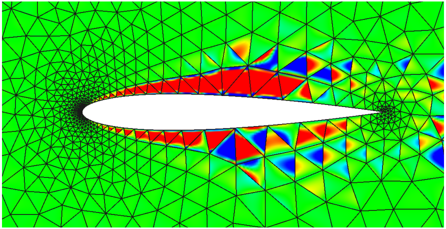
Fine space residual, $\mathbf{R}_h(\mathbf{U}_h^H)$



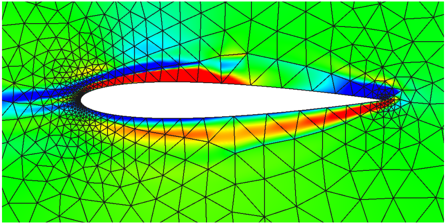
Nonzero: new info

The adjoint-weighted residual

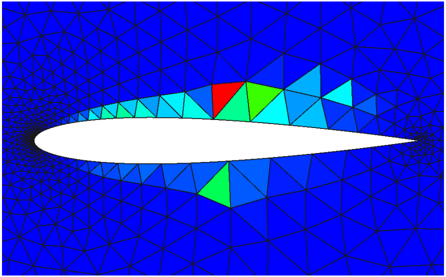
Fine space residual, $\mathbf{R}_h(\mathbf{U}_h^H)$



Fine space adjoint, $\mathbf{\Psi}_h$



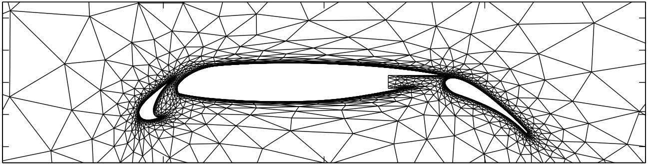
Error indicator, $\epsilon_i = |\mathbf{\Psi}_{h,i}^T \mathbf{R}_{h,i}(\mathbf{U}_h^H)|$



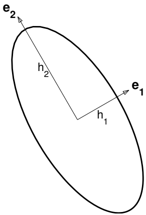
Output error: $\delta J \approx -\mathbf{\Psi}_h^T \mathbf{R}_h$

Idea: adapt where ϵ_i is high, to reduce residual there.

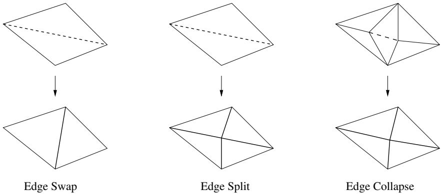
Meshing and adaptation strategies



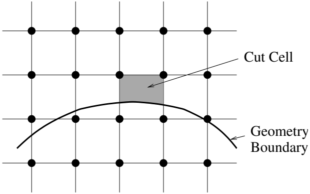
Metric-based anisotropic mesh regeneration (e.g. BAMG software)



Riemannian ellipse

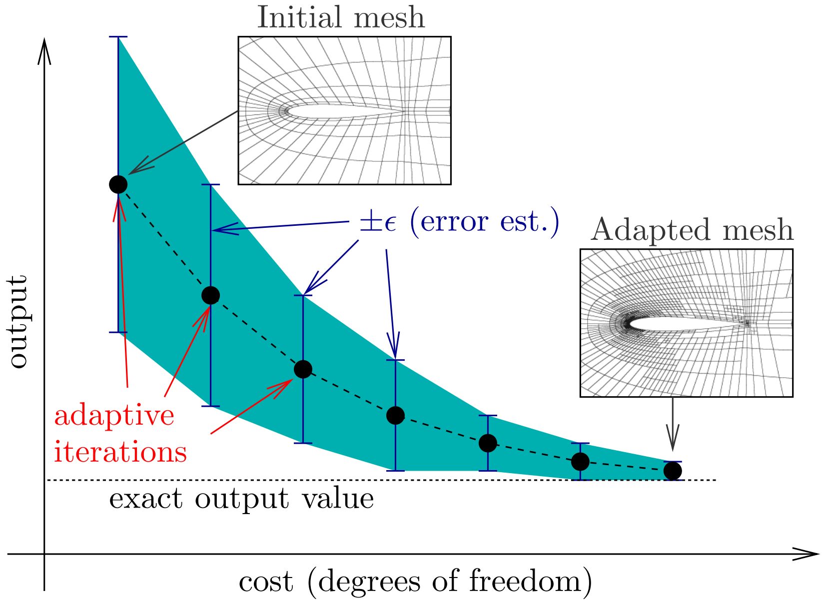


Local mesh operators, and direct optimization



Cut-cell meshes: Cartesian and simplex

Typical adaptive result

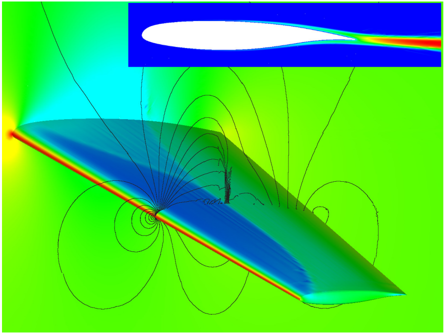


Outline

A steady-state example

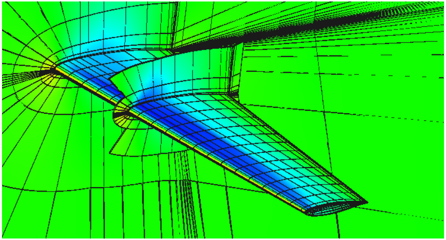
DPW III wing-alone case: $M_\infty = 0.76$, $Re = 5 \times 10^6$

- In-house DG FEM code
- Initial mesh: cubic hex elements generated by agglomeration of linear multiblock meshes (first element $y^+ \approx 1$)
- Artificial viscosity shock capturing
- Spalart-Allmaras turbulence model with negative $\tilde{\nu}$ modification [Oliver & Allmaras]
- Drag-adaptive simulation using *hp* discrete choice algorithm (Ceze + Fidkowski, 2013)

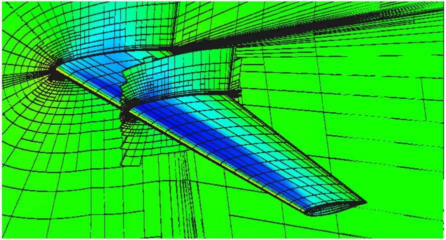


Contours of c_p and $\tilde{\nu}$

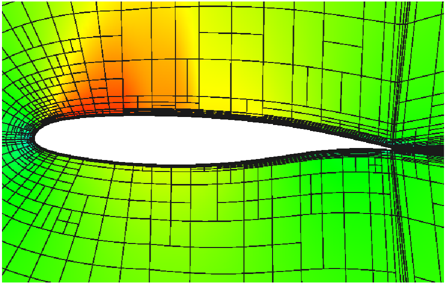
DPW wing: adapted meshes



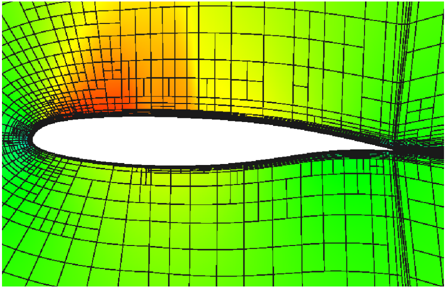
Original mesh, with c_p contours



7th drag-adapted mesh

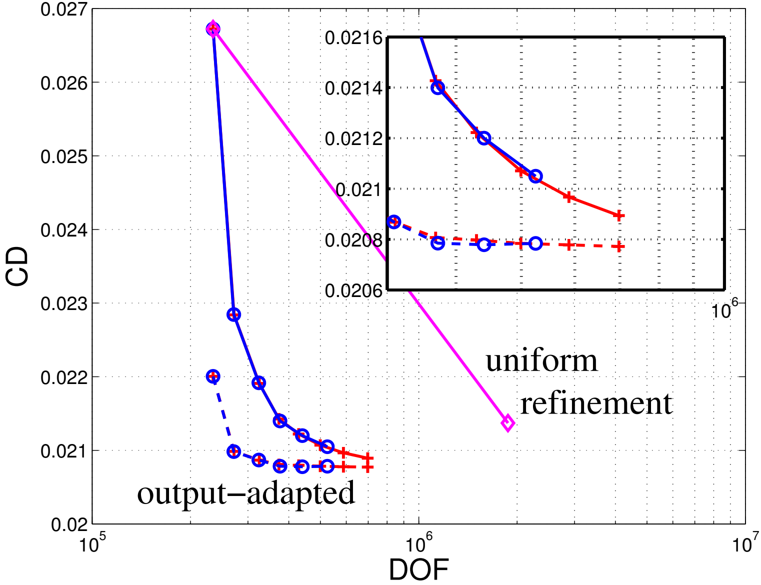


Mach/mesh using DOF cost

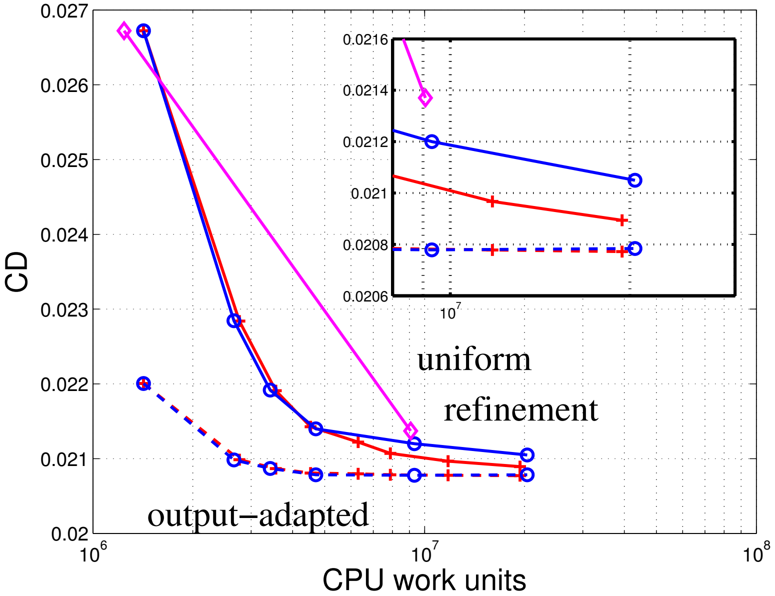


Mach/mesh using non-zero entries cost

DPW wing: comparison to uniform refinement



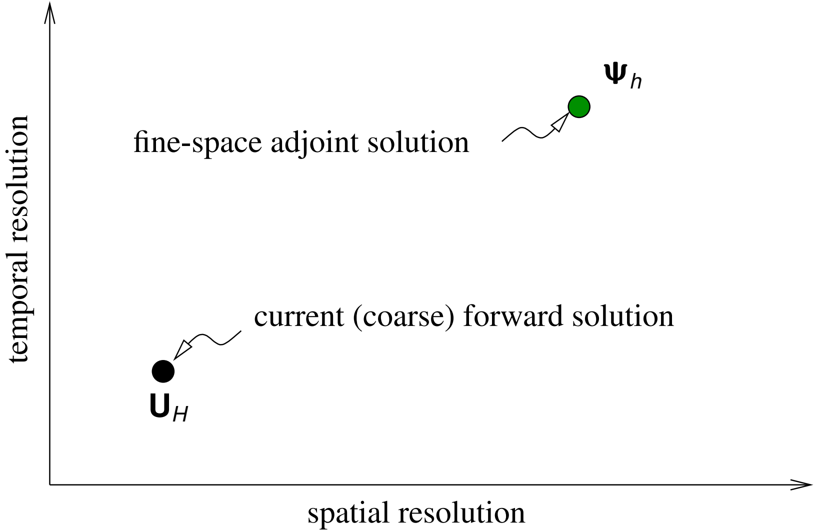
DPW wing: comparison to uniform refinement



Outline

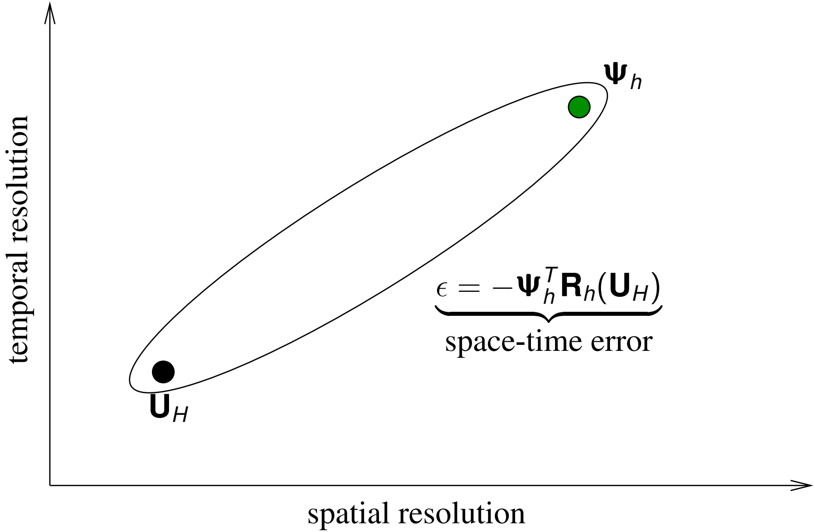
Extension to unsteady problems

- The adjoint becomes more expensive
- Adaptation is trickier – need to measure space-time anisotropy



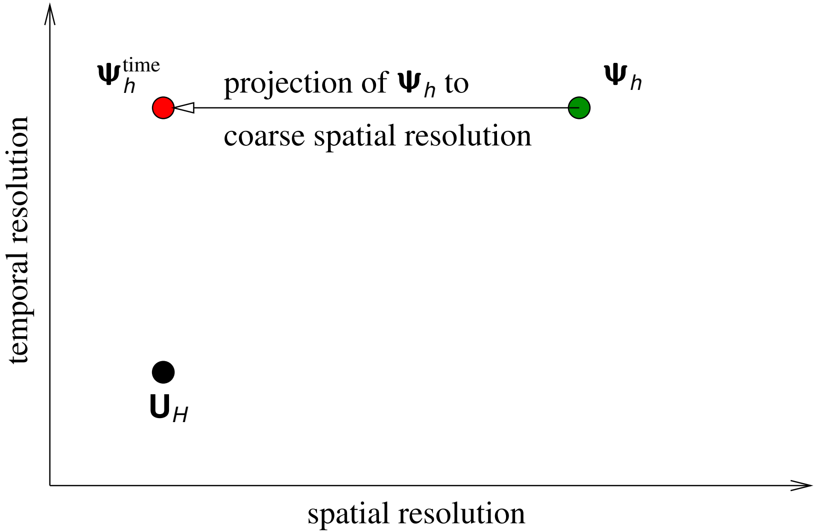
Extension to unsteady problems

- The adjoint becomes more expensive
- Adaptation is trickier – need to measure space-time anisotropy



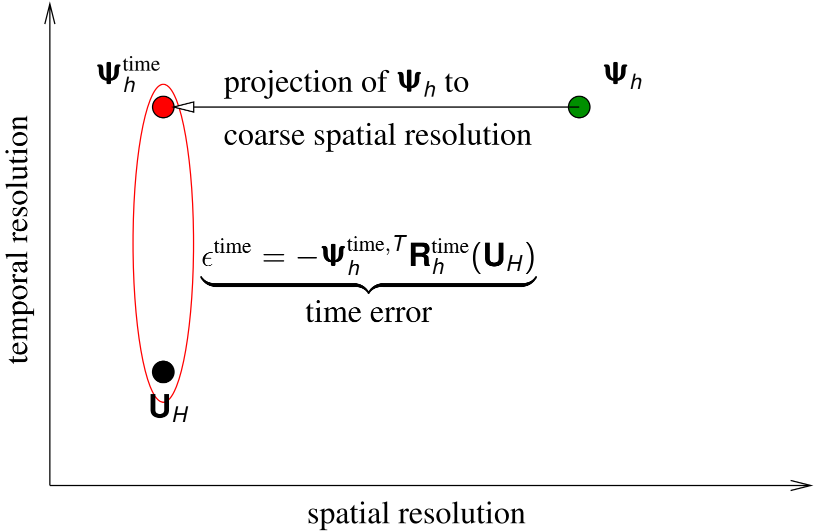
Extension to unsteady problems

- The adjoint becomes more expensive
- Adaptation is trickier – need to measure space-time anisotropy



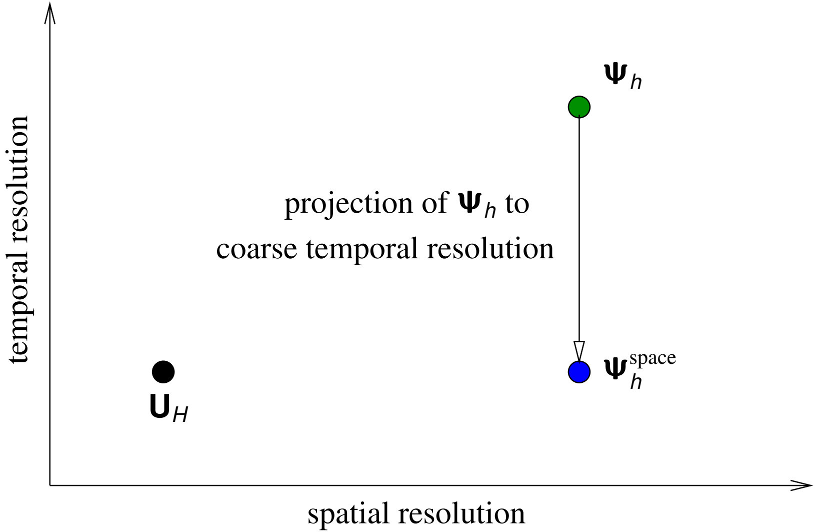
Extension to unsteady problems

- The adjoint becomes more expensive
- Adaptation is trickier – need to measure space-time anisotropy



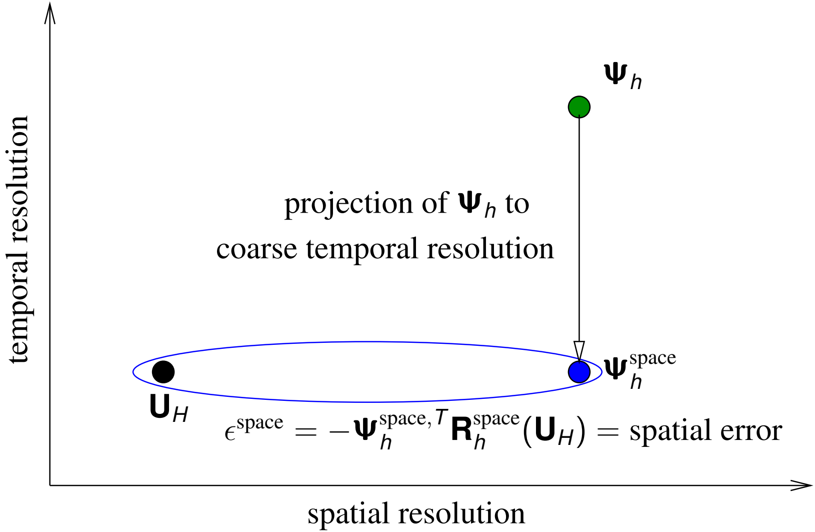
Extension to unsteady problems

- The adjoint becomes more expensive
- Adaptation is trickier – need to measure space-time anisotropy

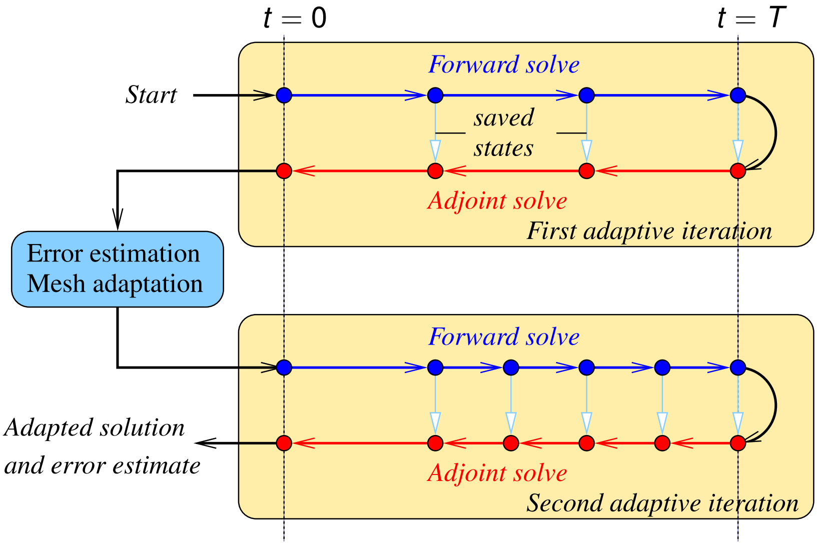


Extension to unsteady problems

- The adjoint becomes more expensive
- Adaptation is trickier – need to measure space-time anisotropy



Adaptive process for unsteady problems



Three-dimensional flapping

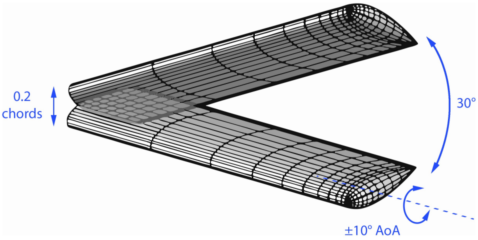
We apply the adaptive strategy to a three-dimensional flapping simulation.

Flow parameters

$Re = 500, \quad M_{inf} = 0.3, \quad Str = 0.4, \quad A_{stroke} = \pm 30^\circ, \quad A_{pitch} = \pm 10^\circ$

Case parameters

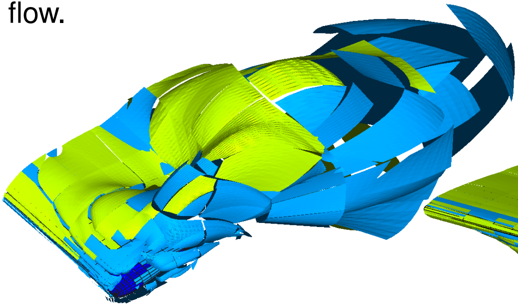
- Farfield at 20+ chords
- DG1 time scheme
- The order p is kept between 0 and 5
- $f_{growth} = 30\%$
- $f_{coarsen} = 5\%$



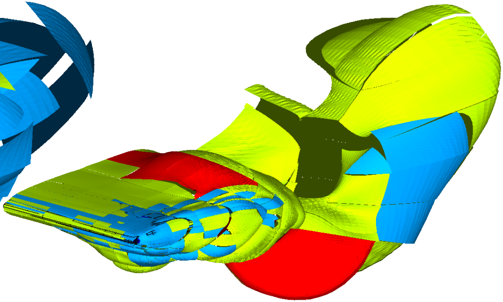
Output: Lift integrated over final 5% of simulation time.

Adapted spatial meshes

Orders (0 to 3) plotted on entropy isosurfaces for two snapshots of the flow.

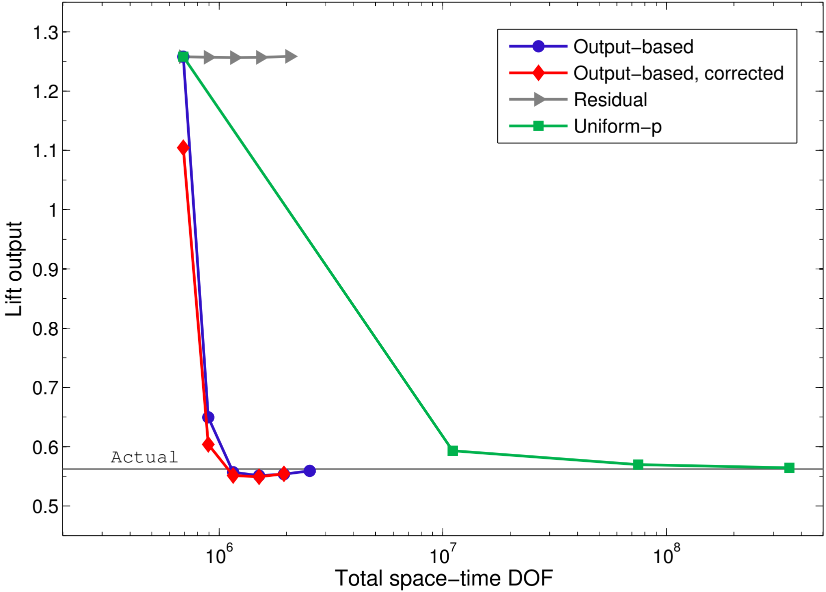


$t = 2.25T$

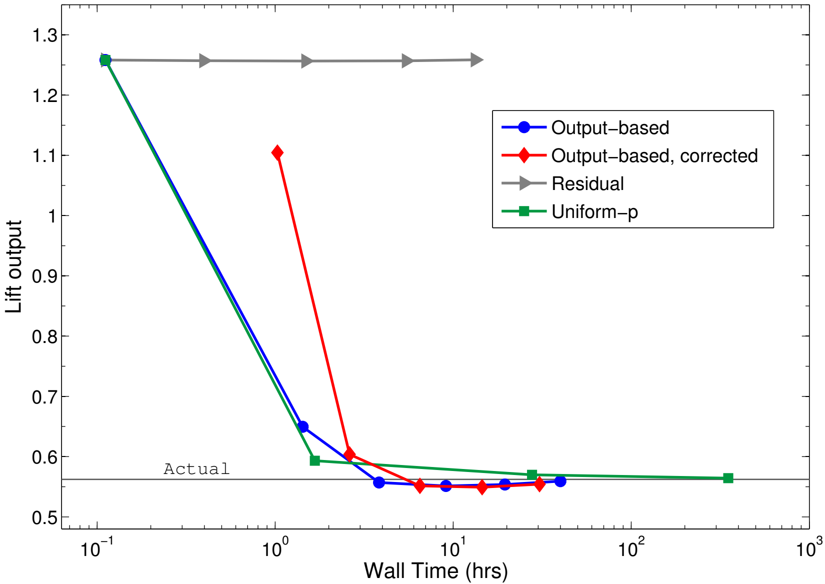


$t = 2.8T$

Output convergence versus DOF



Output convergence versus CPU time

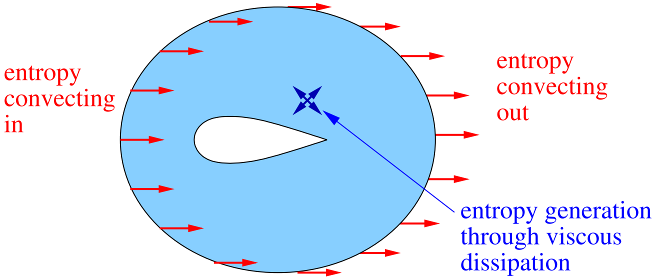


Outline

A "free" adjoint

- An adjoint implementation is not trivial
- But we often do have a "free" adjoint: **the entropy variables**
  - For U = entropy function, $\mathbf{v} = U_{\mathbf{u}}$ is the entropy variable vector
  - The state $\mathbf{v}$ satisfies an adjoint equation!
  - The corresponding output is

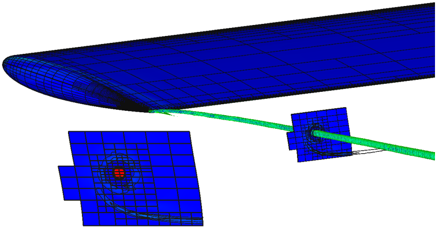
J = net entropy outflow $-$ net physical entropy generation



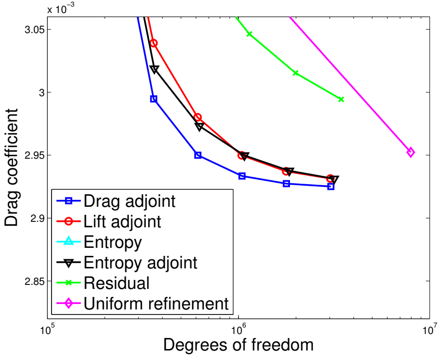
- The adjoint-weighted residual becomes the *entropy residual*

Adapting on the entropy residual

h-Refinement on a rectangular wing in subsonic inviscid flow:



Trailing vortex in a mesh adapted on the entropy adjoint



Convergence of drag compared to output adjoints

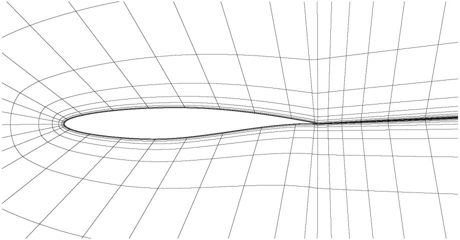
But we lack an error estimate for an engineering output ... or do we?

We can predict drag error!

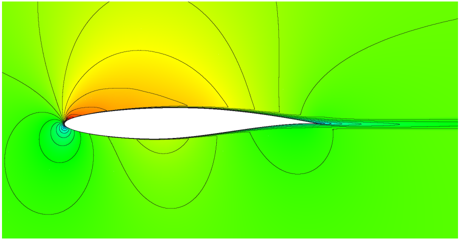
- Under a few assumptions (e.g. 2D), the approximate Oswatitsch formula gives drag:

$$D_{\mathrm{osw}} \approx \frac{u_\infty}{\gamma R M_\infty^2} \boxed{\int_{S_\infty} \Delta s \, \rho \vec{V} \cdot \vec{n} \, dS}$$

- Thermodynamically equivalent to near and far-field measures
- Numerically, values will differ since flow is approximated
- Example: turbulent flow over an RAE airfoil ($Re = 6 \times 10^6$)

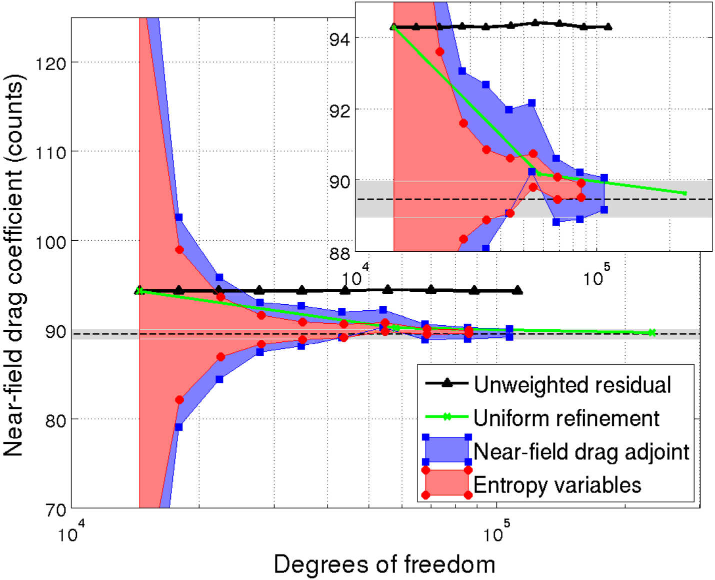


Initial mesh: 1610 elements



Mach number contours

Adapting on the Oswatitsch formula



Outline

Summary

- Output-based methods can improve efficiency and robustness of CFD in aerospace applications
- Adaptation provides tailored meshes for simulations of practical interest
- Error estimation and adaptation extend to unsteady systems
- Our methods allow us to refine space and time meshes separately by gleaning anisotropy from the error indicator
- For sufficiently-fine error tolerances, output-based adaptation *saves CPU time*
- “Almost” output-based methods, e.g. entropy adjoint, offer cheaper alternatives for a variety of situations

Acknowledgments

- Students and post-docs
  - Steve Kast
  - Marco Ceze
- Funding: Air Force, Department of Defense, Department of Energy, University of Michigan

— Thank you —