High Order Discontinuous Galerkin Methods for Fluid and Solid Mechanics

Per-Olof Persson

Department of Mathematics
Massachusetts Institute of Technology

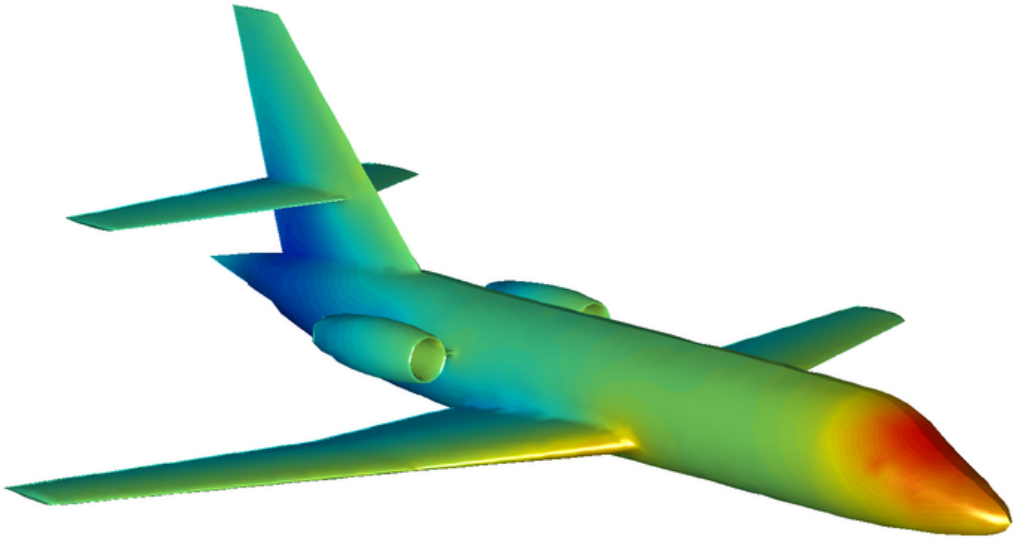
Collaborators: J. Peraire, J. Bonet, M. Drela, D. J. Willis,
N. C. Nguyen, E. Israeli, A. Uranga, A. Mohnot, J. P. Whitney

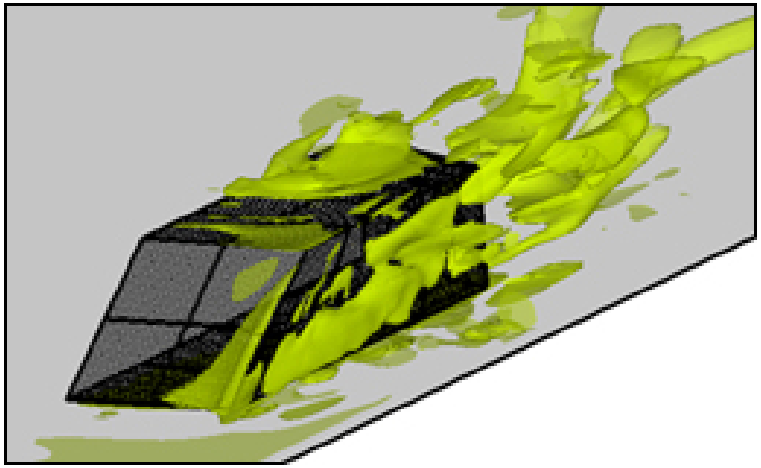
Applied Math Colloquium
Department of Mathematics, MIT
March 3, 2008

Outline

Motivation

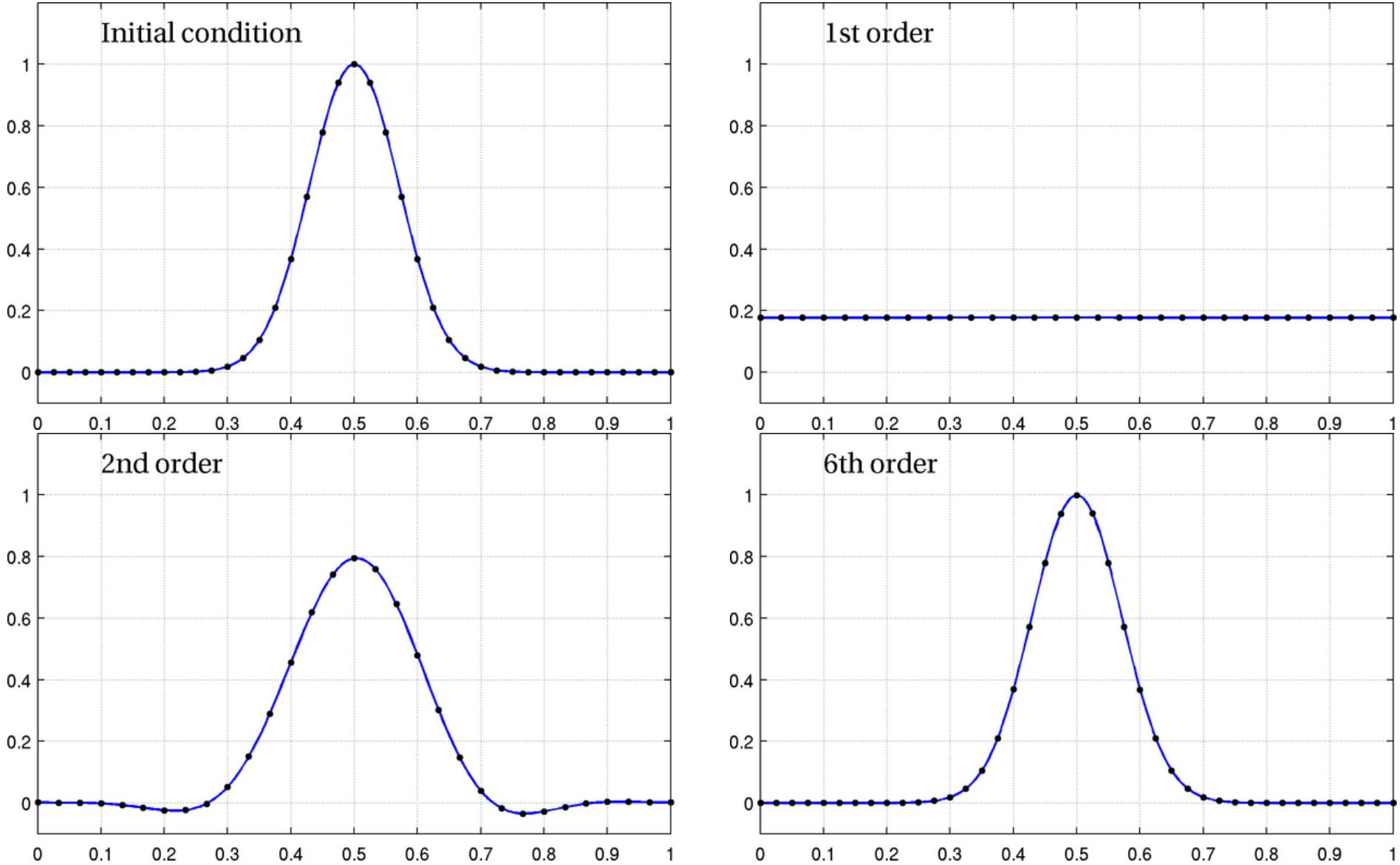
- Need for higher fidelity predictions in computational mechanics
  - DNS/LES/DES applications
  - Accurate RANS for engineering applications (drag prediction, rotor dynamics, fluid/structure interaction, flapping flight)
  - Computational aeroacoustics (direct solution of compressible flow, accurate computation of noise sources)
  - Other problems involving wave propagation, multiple scale phenomena, and non-linear interactions





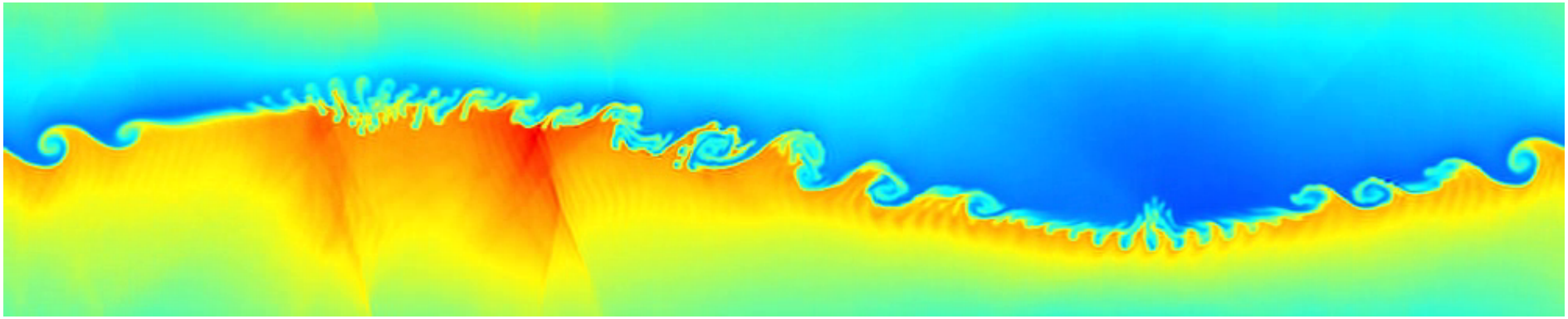
Example: Wave propagation

- Scalar convection equation $u_t + u_x = 0$, fixed resolution



Example: Aeroacoustics and K-H Instability

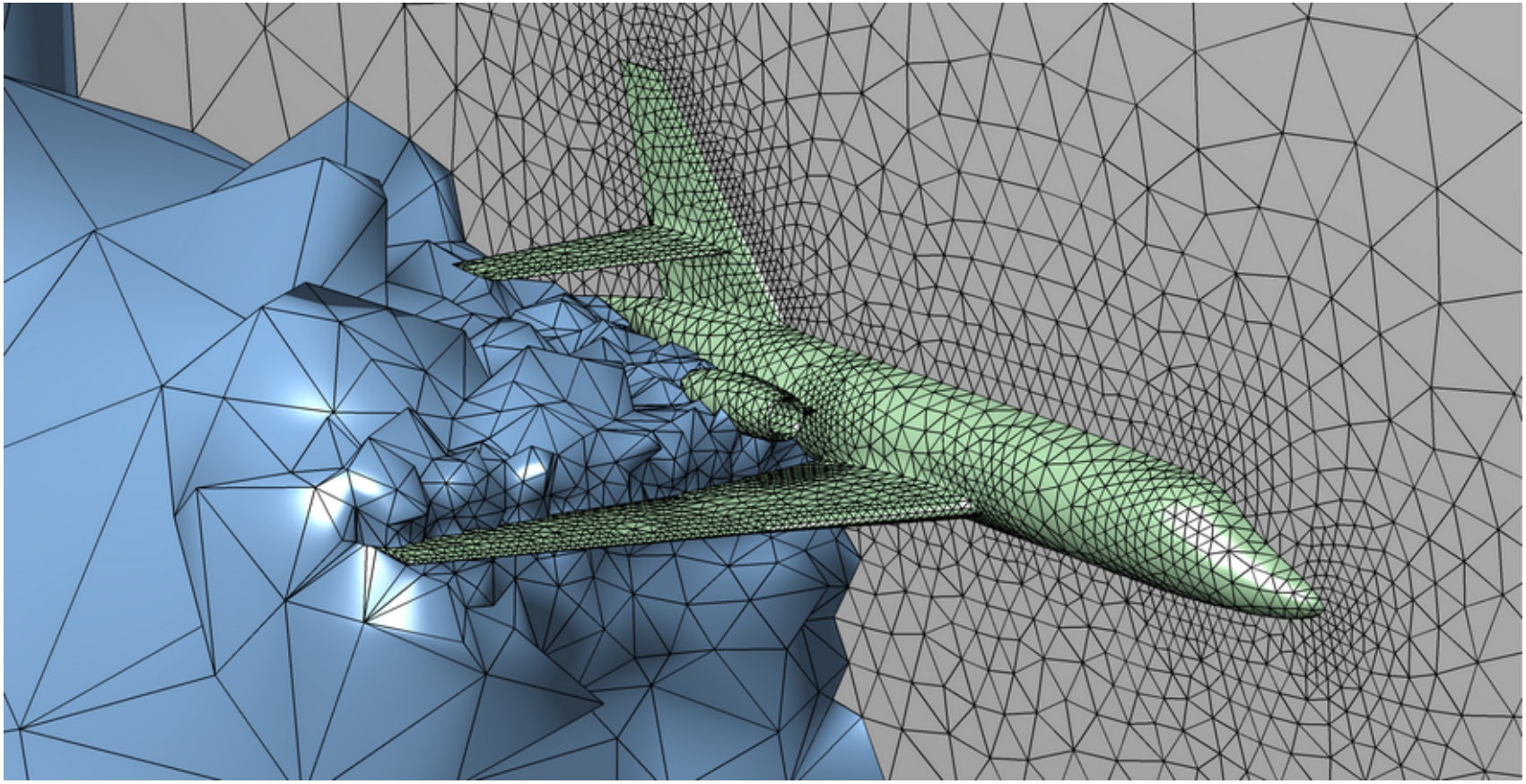
- Inspired by calculations of Munz et al [03] (linearized)
- Nonlinear behavior: Large scale acoustic wave interacts with small scale flow features, leading to vorticity generation
- $p = 7$ (8th order accuracy), 140-by-28 square elements



Mesh Generation for Complex Geometries

Unstructured meshes *required for realistic problems*

- Geometric and adaptive flexibility, accurate boundary treatment
- Tetrahedral meshes give robustness and automation



Motivation

- Fundamental properties of Discontinuous Galerkin (DG) methods:

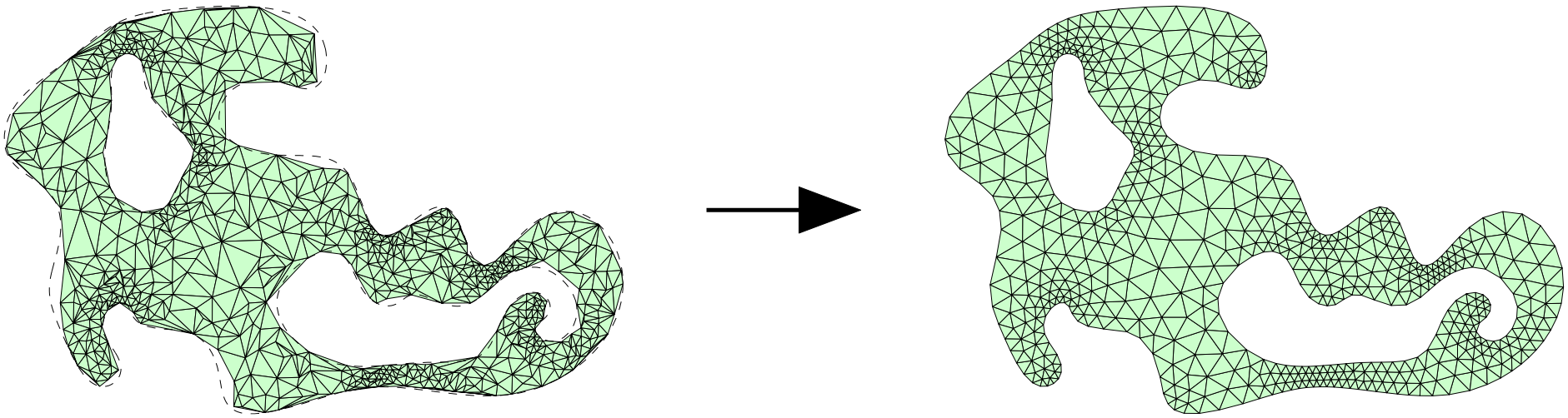
	FVM	FDM	FEM	DG
1) High-order/Low dispersion	✗	✓	✓	✓
2) Unstructured meshes	✓	✗	✓	✓
3) Stability for conservation laws	✓	✓	✗	✓

- However, several problems to resolve:
  - High CPU/memory requirements (compared to FVM or H-O FDM)
  - Low tolerance to under-resolved features
  - High-order geometry representation and mesh generation
- *The challenge is to make DG competitive for real-world problems*

Outline

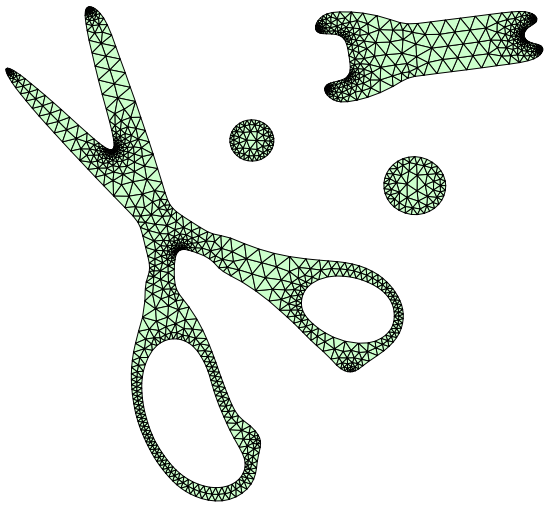
The DistMesh Mesh Generator

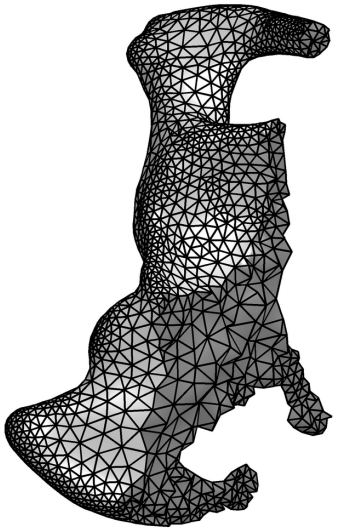
- High quality meshes obtained using the *DistMesh* algorithm [Persson Ph.D. thesis, 2005]
  1. Start with *any* topologically correct initial mesh, for example random node distribution and Delaunay triangulation
  2. Move nodes to find force equilibrium in edges
     - Project boundary nodes using *implicit geometry* $\phi(\boldsymbol{x})$
     - Update element connectivities with Delaunay

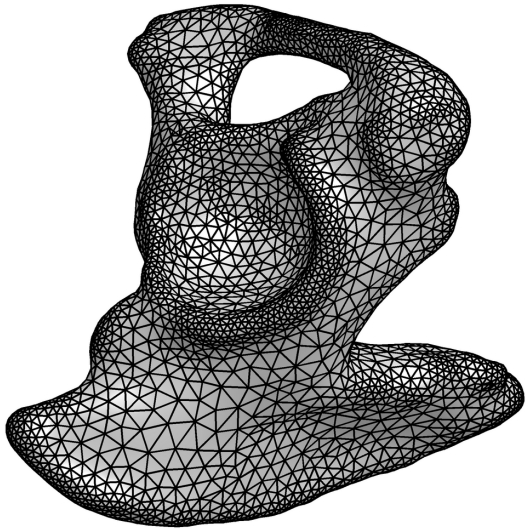


DistMesh Applications

- High quality meshes from images and MRI/CT scans



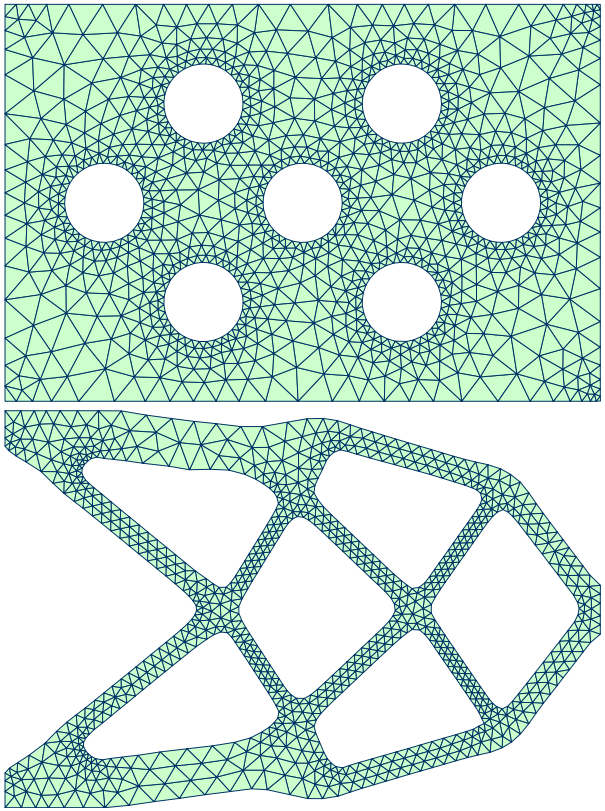




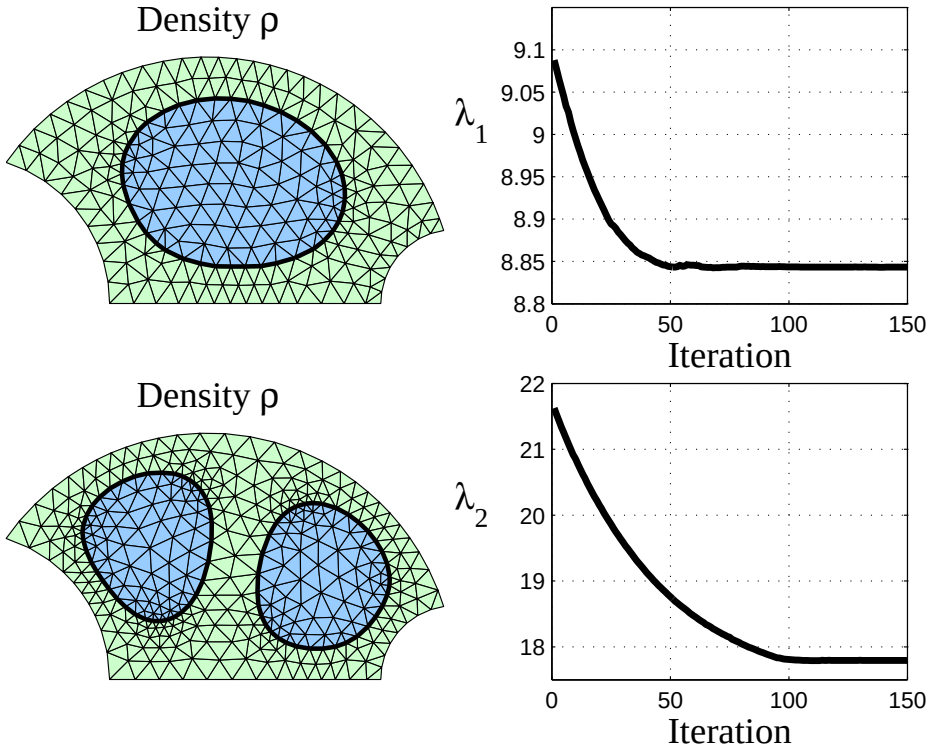
DistMesh Applications

- Shape optimization by combined levelset/finite element method

Structural design
(compliance minimization)

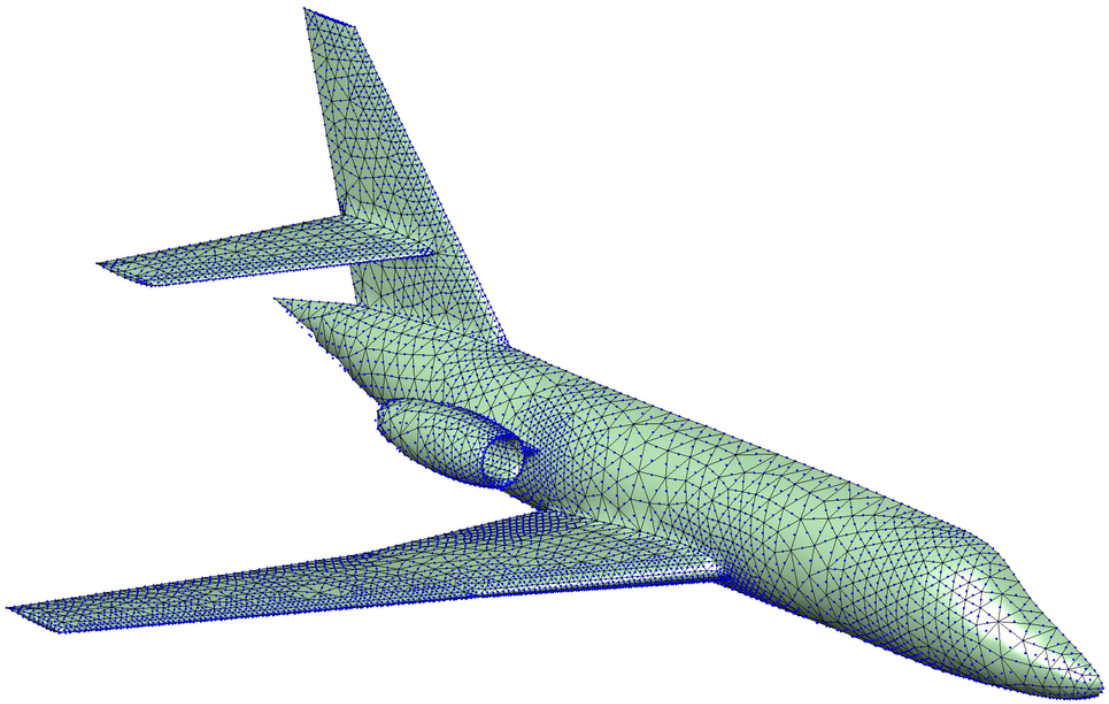


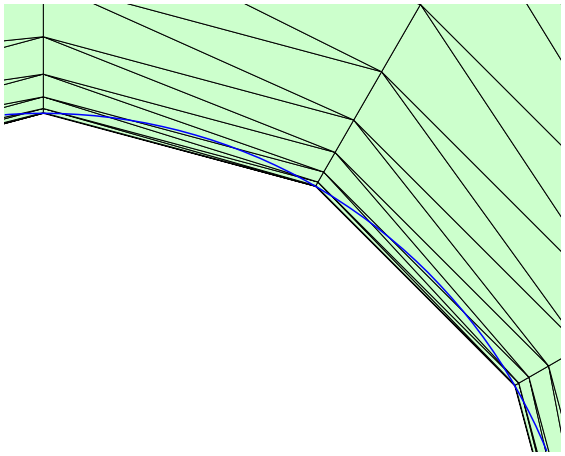
Vibration control
(eigenvalue minimization)

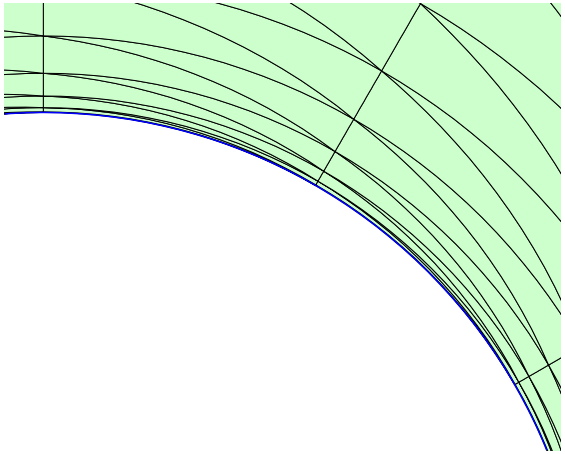


High-Order Curved Mesh Generation

- Open research topic: Unstructured curved mesh generation
- High quality meshes make it easier to avoid inversion
- DistMesh approach for automatic curving (ongoing work)







Outline

The Discontinuous Galerkin Method

- (Reed/Hill 1973, Lesaint/Raviart 1974, Cockburn/Shu 1989-, etc)
- Consider non-linear hyperbolic system in conservative form:

$$\boldsymbol{u}_t + \nabla \cdot \mathcal{F}_i(\boldsymbol{u}) = 0$$

- Triangulate domain Ω into elements $\kappa \in T_h$
- Seek approximate solution $\boldsymbol{u}_h$ in space of element-wise polynomials:

$$\mathcal{V}_h^p = \{\boldsymbol{v} \in L^2(\Omega) : \boldsymbol{v}|_\kappa \in P^p(\kappa) \ \forall \kappa \in T_h\}$$

- Multiply by test function $\boldsymbol{v}_h \in \mathcal{V}_h^p$ and integrate over element κ:

$$\int_\kappa \left[(\boldsymbol{u}_h)_t + \nabla \cdot \mathcal{F}_i(\boldsymbol{u}_h)\right] \boldsymbol{v}_h \, d\boldsymbol{x} = 0$$

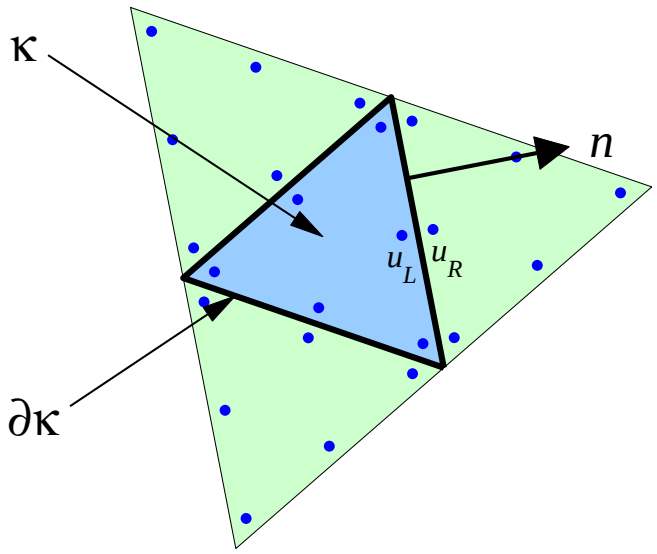
The Discontinuous Galerkin Method

- Integrate by parts:

$$\int_\kappa [(\boldsymbol{u}_h)_t]\, \boldsymbol{v}_h \, d\boldsymbol{x} - \int_\kappa \mathcal{F}_i(\boldsymbol{u}_h) \nabla \boldsymbol{v}_h \, d\boldsymbol{x} + \int_{\partial\kappa} \hat{\mathcal{F}}_i(\boldsymbol{u}_h^+, \boldsymbol{u}_h^-, \hat{\boldsymbol{n}}) \boldsymbol{v}_h^+ \, ds = 0$$

  with numerical flux function $\hat{\mathcal{F}}_i(\boldsymbol{u}_L, \boldsymbol{u}_R, \hat{\boldsymbol{n}})$ for left/right states $\boldsymbol{u}_L$,$\boldsymbol{u}_R$ in direction $\hat{\boldsymbol{n}}$ (Godunov, Roe, Osher, Van Leer, Lax-Friedrichs, etc)

- Global problem: Find $\boldsymbol{u}_h \in \mathcal{V}_h^p$ such that this weighted residual is zero for all $\boldsymbol{v}_h \in \mathcal{V}_h^p$
- Error $= \mathcal{O}(h^{p+1})$ for smooth solutions



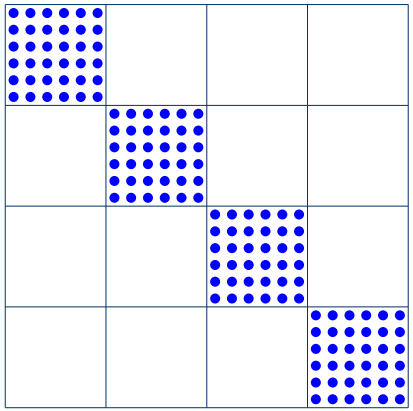
The DG Method – Observations

- Reduces to the finite volume method for $p = 0$:

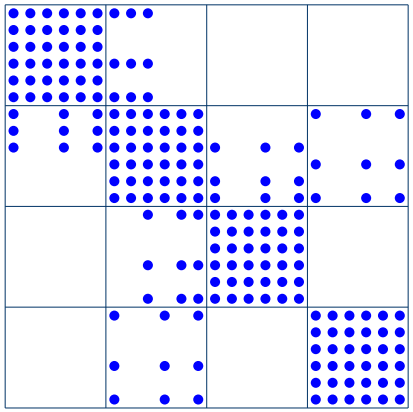
$$(\boldsymbol{u}_h)_t A_\kappa + \int_{\partial\kappa} \hat{\mathcal{F}}_i(\boldsymbol{u}_h^+, \boldsymbol{u}_h^-, \hat{\boldsymbol{n}})\, ds = 0$$

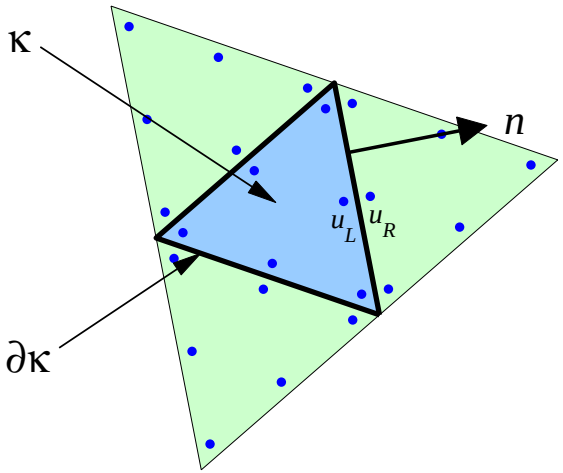
- Boundary conditions enforced naturally for any degree p
- Block-diagonal mass matrix (no overlap between basis functions)
- Block-wise compact stencil – neighboring elements connected

Mass Matrix



Jacobian





Outline

Viscous Discretization

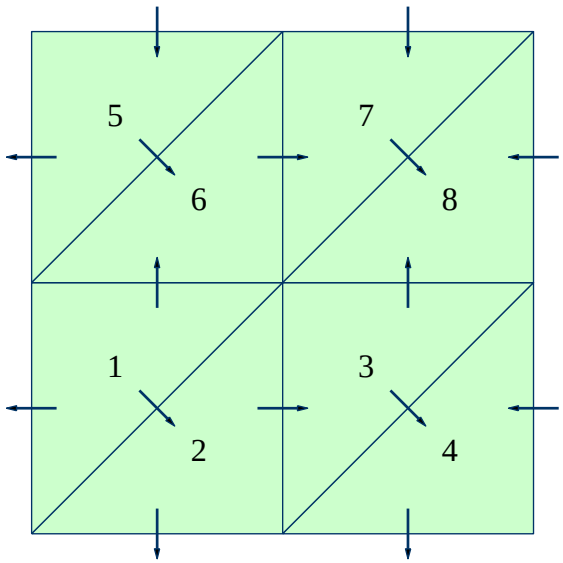
- General approach for second derivatives:
  - Write as system of first order equations:

$$\boldsymbol{u}_t + \nabla \cdot \mathcal{F}_i(\boldsymbol{u}) - \nabla \cdot \mathcal{F}_v(\boldsymbol{u}, \boldsymbol{\sigma}) = 0$$
$$\boldsymbol{\sigma} - \nabla \boldsymbol{u} = 0$$

  - Discretize using DG, choose appropriate numerical fluxes $\hat{\boldsymbol{\sigma}}$, $\hat{u}$
- Various schemes have been proposed:
  - *BR1* [Bassi/Rebay 97]: Averaging, unstable and non-compact
  - *BR2* [Bassi/Rebay 98]: Different lifting operator for each edge, compact connectivities, similar to Interior Penalty (IP)
  - *LDG* [Cockburn/Shu 98]: Upwind/Downwind, non-compact
  - *CDG* [Peraire/Persson 07]:
    Modification of LDG for local dependence – sparse and compact

The Local DG Method

- Consider Poisson problem $-\nabla \cdot (\kappa \nabla u) = f$
- Write as system of first order equations,



$$-\nabla \cdot \boldsymbol{\sigma} = f$$

$$\boldsymbol{\sigma} = \kappa \nabla u$$

- Use numerical inter-element fluxes

$$\hat{\boldsymbol{\sigma}} = \{\boldsymbol{\sigma}_h\} - C_{11}[u_h] + \boldsymbol{C}_{12}[\boldsymbol{\sigma}_h]$$

$$\hat{u} = \{u_h\} - \boldsymbol{C}_{12} \cdot [u_h]$$

where $\{\cdot\}, [\cdot]$ denote averaging and difference

- In particular, choosing $\boldsymbol{C}_{12} = 1$ or -1 depending on a *switch* for each edge, will upwind/downwind $\hat{\boldsymbol{\sigma}}, \hat{u}$

The Local DG Scheme

- Solving for the variables $\boldsymbol{\sigma}_h$ gives

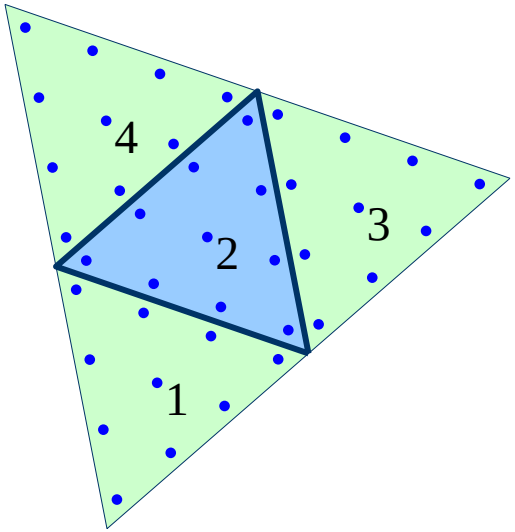
$$\boldsymbol{\sigma}_h = \kappa \nabla_h u_h + \bar{\boldsymbol{\sigma}}_h$$

where

$$\bar{\boldsymbol{\sigma}}_h = \kappa r([u_h]) + \kappa l(\boldsymbol{C}_{12} \cdot [u_h]) + \text{ boundary terms}$$

and $r(\phi)$ and $l(q)$ are *lifting operators* (essentially L_2-projections)

- In general, this introduces non-local couplings since the lifting operators involve all element edges



The Compact DG Scheme

- In the CDG scheme, we split the lifting operators into sums of edge-wise lifting operators $r^e(\phi)$, $l^e(q)$, and set

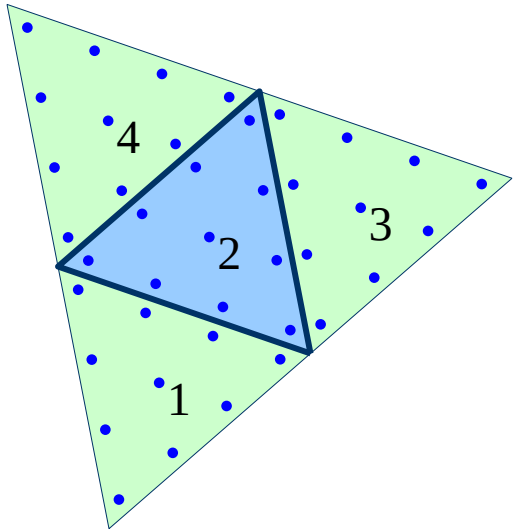
$$\hat{\boldsymbol{\sigma}} = \{\boldsymbol{\sigma}_h^e\} - C_{11}[u_h] + \boldsymbol{C}_{12}[\boldsymbol{\sigma}_h^e]$$

$$\hat{u} = \{u_h\} - \boldsymbol{C}_{12} \cdot [u_h]$$

where $\boldsymbol{\sigma}_h^e = \kappa \nabla_h u_h + \bar{\boldsymbol{\sigma}}_h^e$, with

$$\bar{\boldsymbol{\sigma}}_h^e = \kappa r^e([u_h]) + \kappa l^e(\boldsymbol{C}_{12} \cdot [u_h]) + \text{ boundary terms}$$

- Since only the lifting operator corresponding to the current edge is used, only neighboring elements are connected



Error Estimates

- In primal form, the LDG scheme becomes (ignoring bnd terms):

$$\int_\Omega \kappa(r([u]) + l(\boldsymbol{C}_{12} \cdot [u])) \cdot (r([v]) + l(\boldsymbol{C}_{12} \cdot [v]))\, dx =$$
$$\sum_{e \in \mathcal{E}_i} \sum_{f \in \mathcal{E}_i} \int_\Omega \kappa(r^e([u]) + l^e(\boldsymbol{C}_{12} \cdot [u])) \cdot (r^f([v]) + l^f(\boldsymbol{C}_{12} \cdot [v]))\, dx$$

- The CDG scheme excludes some terms that are indefinite:

$$\sum_{e \in \mathcal{E}_i} \int_\Omega \kappa(r^e([u]) + l^e(\boldsymbol{C}_{12} \cdot [u])) \cdot (r^e([v]) + l^e(\boldsymbol{C}_{12} \cdot [v]))\, dx =$$
$$\sum_{e \in \mathcal{E}_i} \sum_{f \in \mathcal{E}_i} \delta_{ef} \int_\Omega \kappa(r^e([u]) + l^e(\boldsymbol{C}_{12} \cdot [u])) \cdot (r^f([v]) + l^f(\boldsymbol{C}_{12} \cdot [v]))\, dx$$

- Non-compact terms are eliminated but the scheme remains stable

Error Estimates

- Coercivity and boundedness for the CDG scheme same as for LDG, leading to a-*priori* estimates:

$$|||u - u_h||| \leq Ch^p |u|_{p+1,\Omega}$$

and

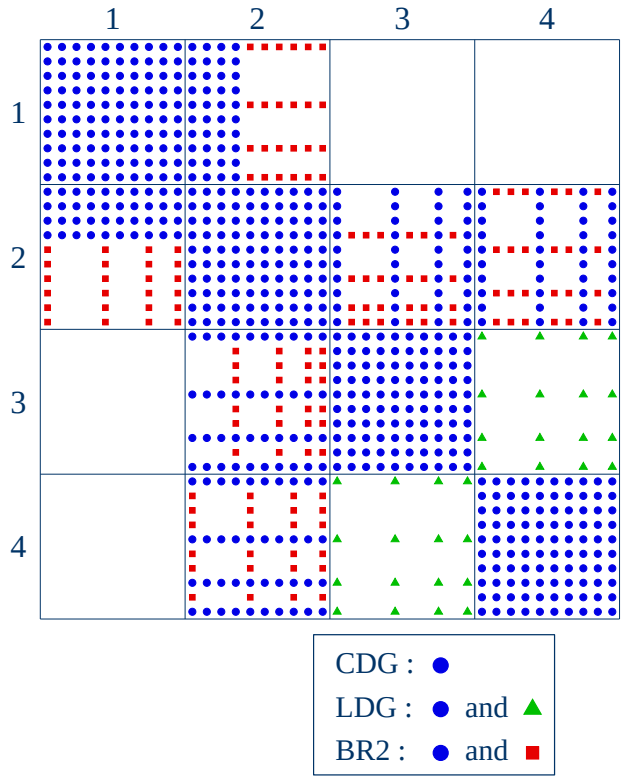
$$||u - u_h||_{0,\Omega} \leq Ch^{p+1} |u|_{p+1,\Omega}$$

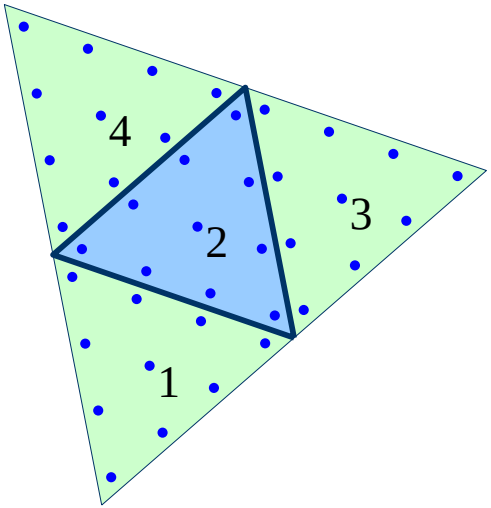
with the norm

$$|||v|||^2 = \sum_{K \in \mathcal{T}_h} |v|_{1,K}^2 + \sum_{e \in \mathcal{E}_i} ||r_e([v])||_{0,\Omega}^2 + \sum_{e \in \partial\Omega_D} ||r_D(v)||_{0,\Omega}^2$$

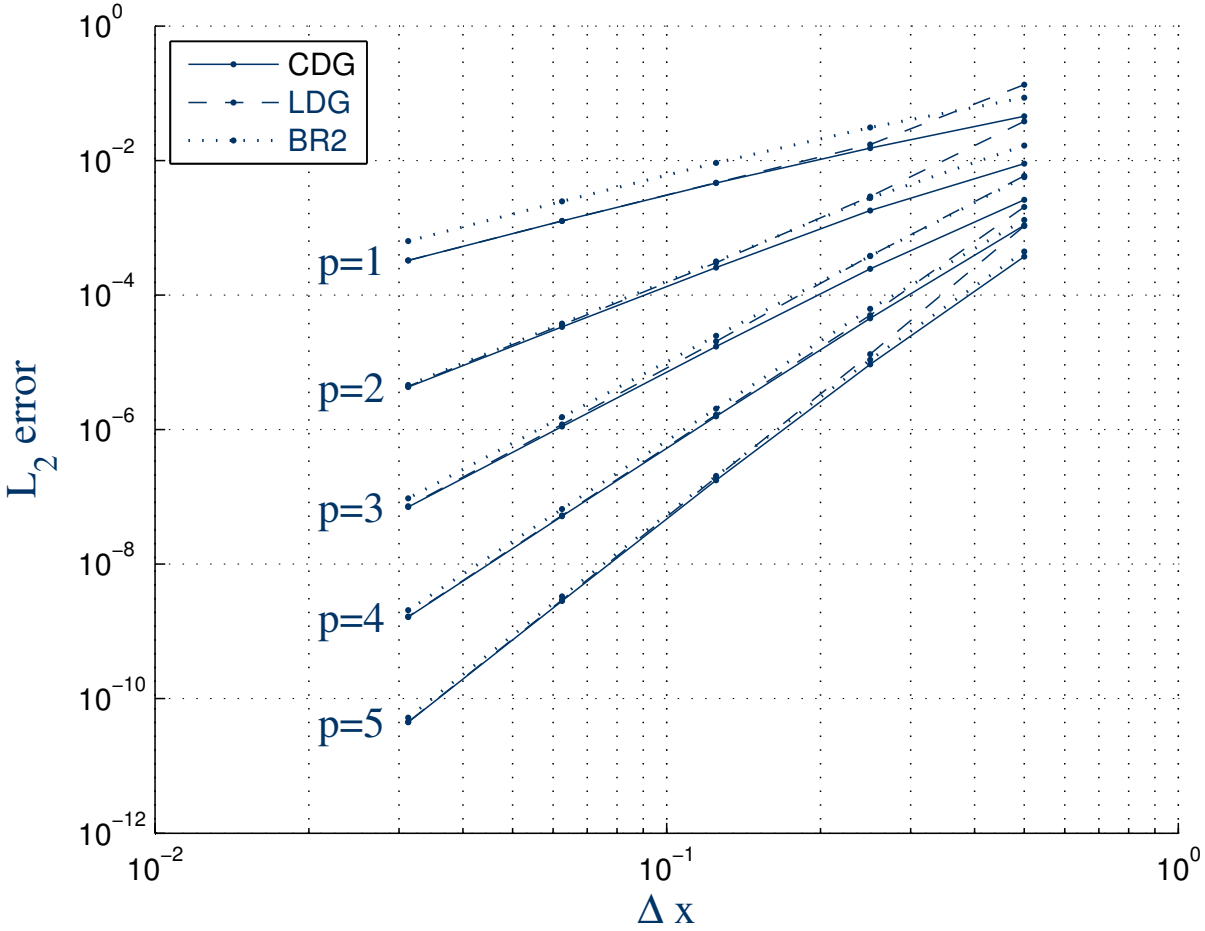
- Assumes $C_{11} = \mathcal{O}(h^{-1})$, but is observed numerically for $C_{11} = 0$

The CDG Method – Summary



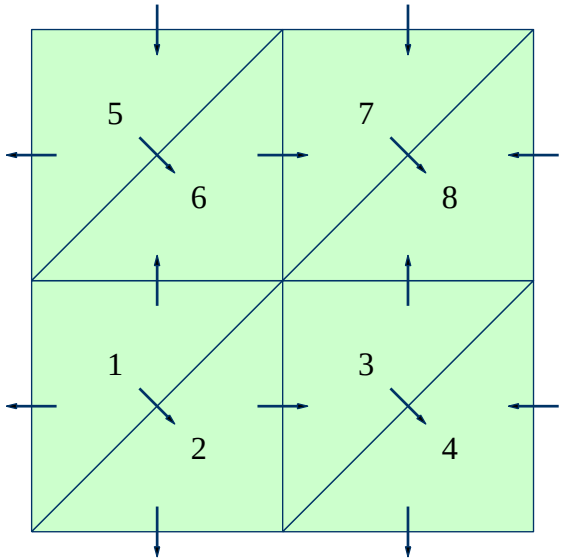


- Element-wise compact stencil
- Less connectivities than LDG/BR2/IP
- More accurate than LDG and BR2



Switches and Null-space Dimensions

- Unlike the LDG scheme, the CDG scheme appears to be stable for $C_{11} = 0$ and an *inconsistent switch* such as highest element number
- Simple test [Sherwin et al 05]: Poisson problem, periodic boundary conditions, expected nullspace dimension = 1

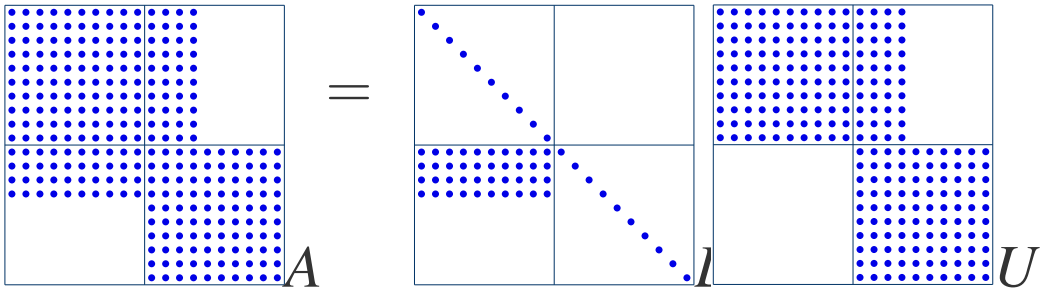


Nullspace dimension

Polynomial order p		1	2	3	4	5	6	7
Consistent switch	CDG	1	1	1	1	1	1	1
	LDG	1	1	1	1	1	1	1
Natural switch	CDG	1	1	1	1	1	1	1
	LDG	3	4	5	6	7	8	9

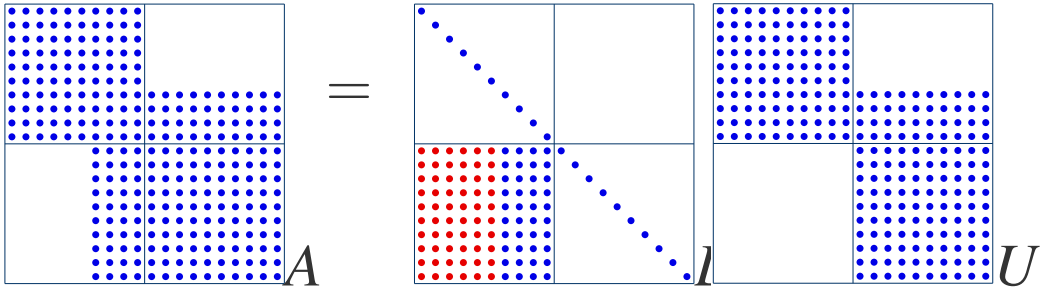
ILU and Switch Orientation

- Orientation of lower-triangular blocks important for ILU sparsity
- Take advantage of CDG's insensitivity to orientation



Switch 1:

Same LU storage



Switch 2:

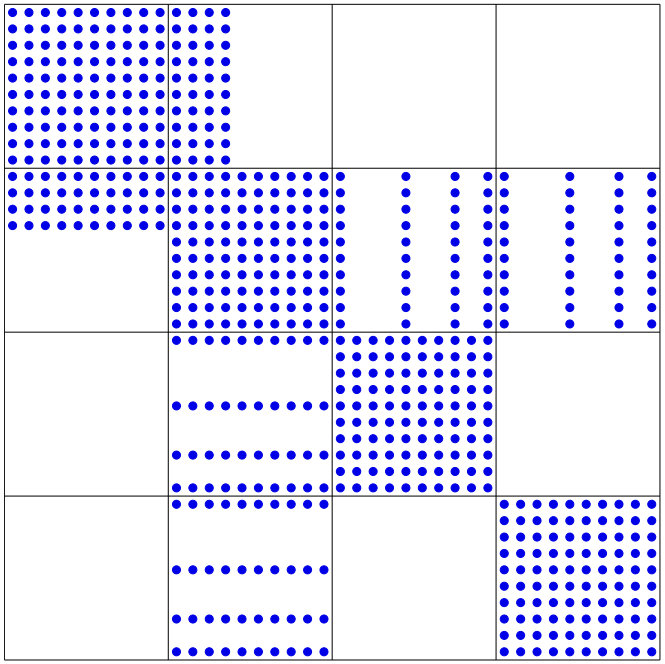
More LU storage

Switches and Null-space Dimensions

- No additional non-zeros in block-ILU(0) factorization using CDG
- Dense lower-triangular blocks using BR2 / IP

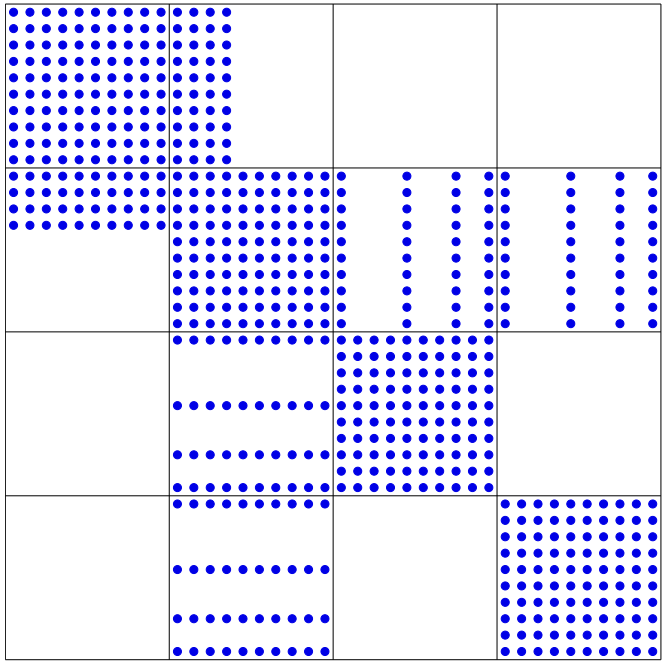
CDG

Stiffness Matrix



640 non-zeros

Block ILU(0)



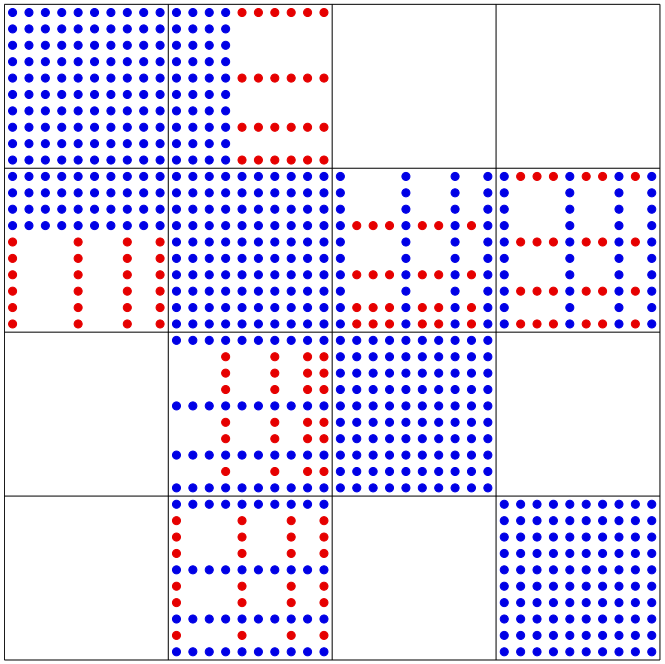
640 non-zeros

Switches and Null-space Dimensions

- No additional non-zeros in block-ILU(0) factorization using CDG
- Dense lower-triangular blocks using BR2 / IP

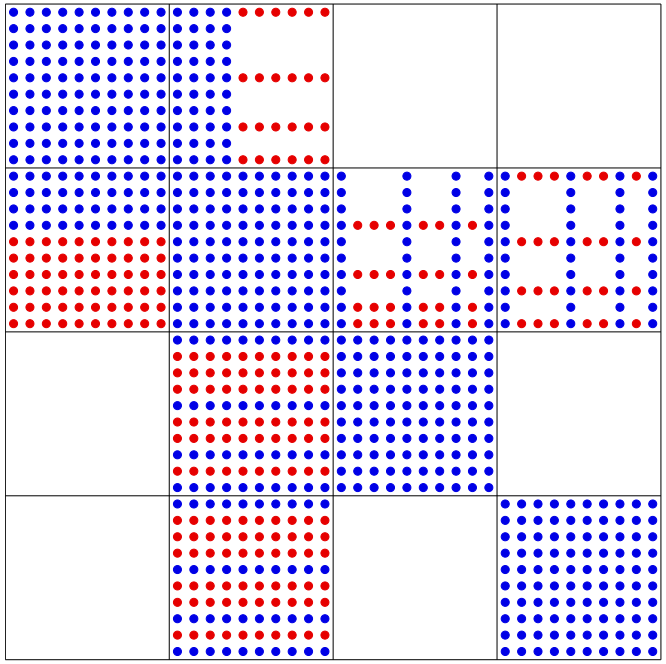
BR2 / IP

Stiffness Matrix



784 non-zeros

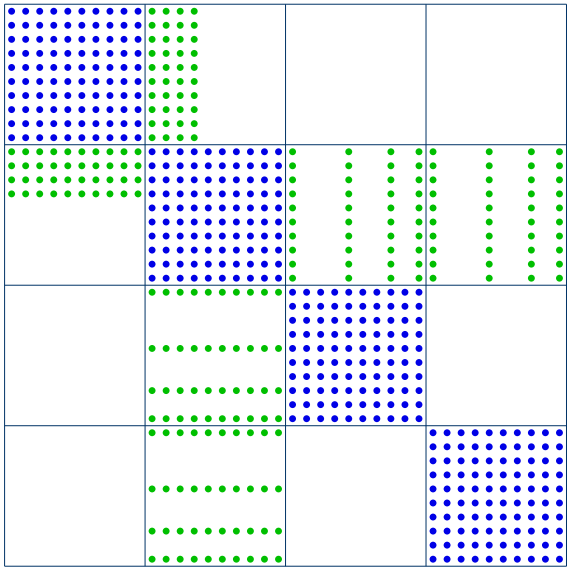
Block ILU(0)



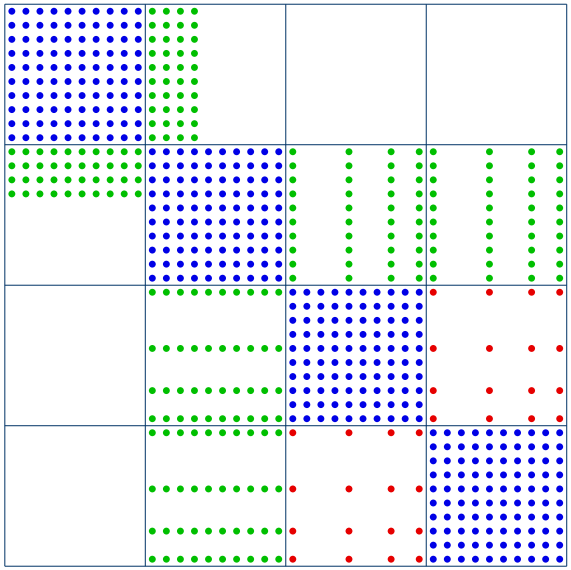
892 non-zeros

Matrix Representation

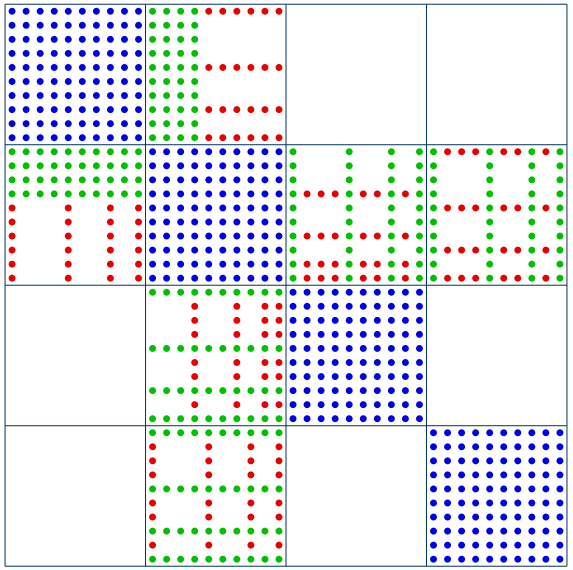
- Block matrix representation *fundamental for high performance*
  - Solver algorithms based on blocks
  - Up to 10 times higher performance with optimized BLAS
- Compact stencil $\Longrightarrow$ Matrix structure given by mesh connectivities
- Hard to store LDG/BR2/IP efficiently



CDG – 2 arrays



LDG – 3 arrays + struct

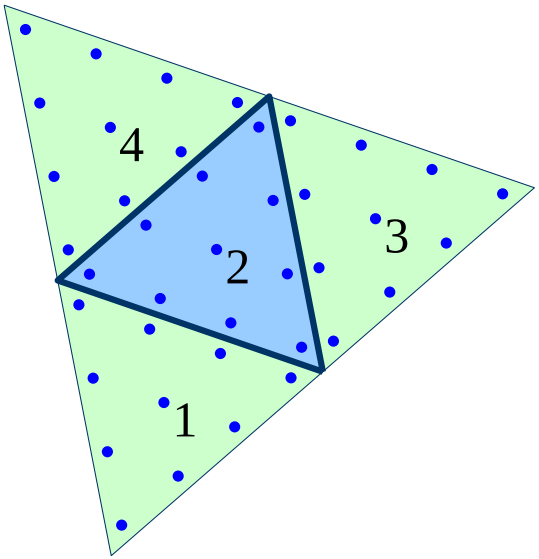


BR2 / IP – 3 arrays

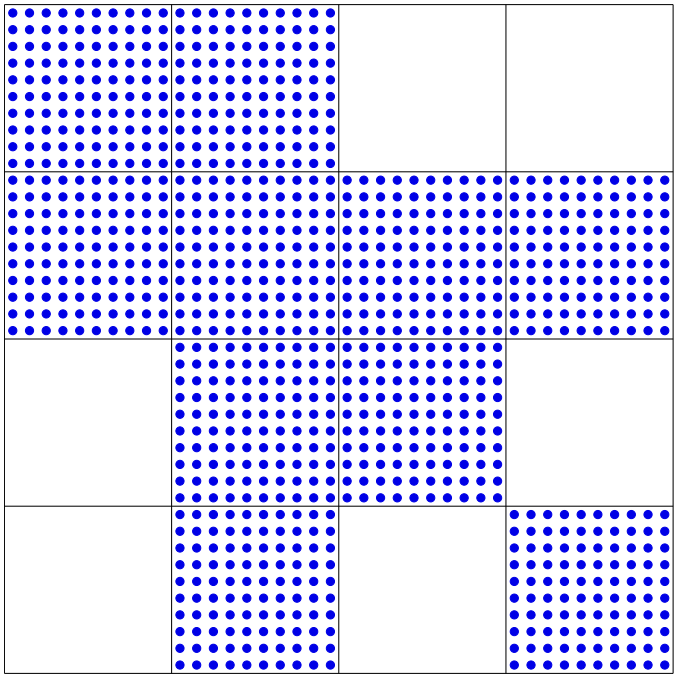
Outline

Jacobian Sparsity Structure

- Simple model of sparsity pattern: Connectivity of dual mesh, with large complete graphs $\boldsymbol{A}_{ij}$ in each cell
- Sizes of submatrices $\boldsymbol{A}_{ij}$ typically $>$ 100 DOFs
- Easy to obtain full BLAS performance! (e.g. in matvec or LU)



Block structure



Preconditioners for Krylov Methods

- Performance of Krylov methods greatly improved by preconditioning
- Various standard methods for block matrices:
  - Block diagonal: $\boldsymbol{A} \approx \mathrm{blockdiag}(\boldsymbol{A})$
    - Poor in general
  - Block Incomplete LU: $\boldsymbol{A} \approx \tilde{\boldsymbol{L}}\tilde{\boldsymbol{U}}$
    - Good for convection (with the right ordering, more later)
  - p-multigrid (low-degree) preconditioner
    - Good for diffusion
- Highly efficient combination [Persson/Peraire 06]
  - Use block ILU(0) as post-smoother for coarse scale correction
  - Combines advantages of ILU and low-degree preconditioners
  - Cheap general purpose preconditioner

Incomplete LU Factorization

- For our low-degree dual meshes, it is uncommon that an element k is a neighbor of both element j and i when j is a neighbor of i
- This simplifies the Block-ILU(0) algorithm:

$\tilde{\boldsymbol{U}} \leftarrow \boldsymbol{A}, \tilde{\boldsymbol{L}} \leftarrow \boldsymbol{I}$

for $j = 1$ **to** $n - 1$

  for neighbors $i > j$ of j

    $\tilde{\boldsymbol{L}}_{ij} = \tilde{\boldsymbol{U}}_{ij} \tilde{\boldsymbol{U}}_{jj}^{-1}$

    $\tilde{\boldsymbol{U}}_{ii} \leftarrow \tilde{\boldsymbol{U}}_{ii} - \tilde{\boldsymbol{L}}_{ik} \tilde{\boldsymbol{U}}_{ki}$

  end for

end for

- This means $\tilde{\boldsymbol{U}}_{ij} = \boldsymbol{A}_{ij}$ when $j > i$, or $\tilde{\boldsymbol{U}}$ only differs from $\boldsymbol{A}$ in the diagonal blocks
- Store only diagonal blocks of $\tilde{\boldsymbol{U}}$!

Minimum Discarded Fill Element Ordering

- Performance of ILU(0) highly dependent of ordering
- Greedy algorithm for element ordering [Persson/Peraire 07]:
  At step j, if j' is chosen next, we would discard the fill

$$\Delta\tilde{\boldsymbol{U}}_{ik}^{(j,j')} = -\tilde{\boldsymbol{U}}_{ij'}\tilde{\boldsymbol{U}}_{j'j'}^{-1}\tilde{\boldsymbol{U}}_{j'k}, \quad \text{for neighbors } i \geq j, k \geq j \text{ of element } j'$$

- Choose the j' that minimizes the norm of the discarded fill

$$w^{(j,j')} = \|\Delta\tilde{\boldsymbol{U}}^{(j,j')}\|_{\mathrm{F}}$$

- Some simplifications, min-heap data structure $\Longrightarrow \mathcal{O}(n \log n)$ cost
- Similar to the *Minimum Degree* algorithm, but considering the magnitude of the fill instead of just the size
- Related to the *Minimum Discarded Fill* algorithm [D'Azevedo, Forsyth, Tang 92]

Minimum Discarded Fill Element Ordering

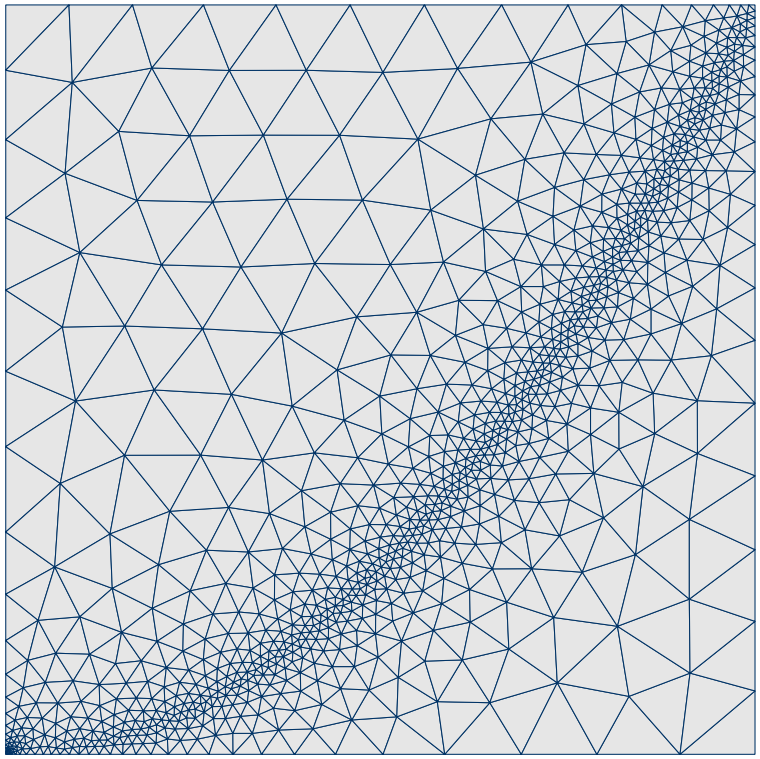
- Compute MDF ordering p for block-matrix $\boldsymbol{A}$:

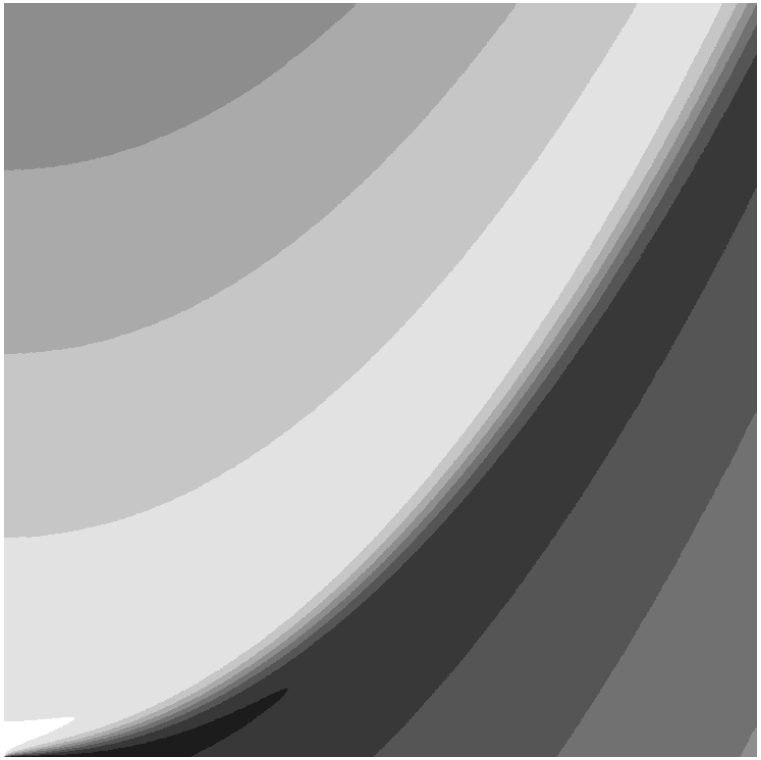
$\boldsymbol{B} \leftarrow (\tilde{\boldsymbol{A}}^{\mathrm{J}})^{-1}\boldsymbol{A}$	*Pre-multiply by block diagonal*
$C_{ij} \leftarrow \|\boldsymbol{B}_{ij}\|_{\mathrm{F}}$	*Reduce each block to scalar*
for $k = 1, \ldots, n$	*Compute all weights*
$\quad \Delta C \leftarrow 0$	
$\quad$ **for** neighbors i, j of element k, $i \neq j$	
$\quad\quad \Delta C_{ij} \leftarrow C_{ik} C_{kj}$	*Discarded fill matrix*
$\quad$ **end for**	
$\quad w_k \leftarrow \|\Delta C\|_{\mathrm{F}}$	*Fill weight*
end for	
for $i = 1, \ldots, n$	*Main loop*
$\quad p_i \leftarrow \mathrm{argmin}_j w_j$	*Pivot with smallest fill-in*
$\quad w_{p_i} \leftarrow \infty$	*Do not choose p_i again*
$\quad$ **for** neighbors k of p_i not yet numbered	*Update weights*
$\quad\quad$ Recompute w_k, only considering	
$\quad\quad$ neighbors not yet numbered	
$\quad$ **end for**	
end for	

Effect of Ordering on Convection-Diffusion

- Convection-Diffusion model problem, with $(\alpha, \beta) = (1, 2x)$, $\varepsilon \geq 0$:

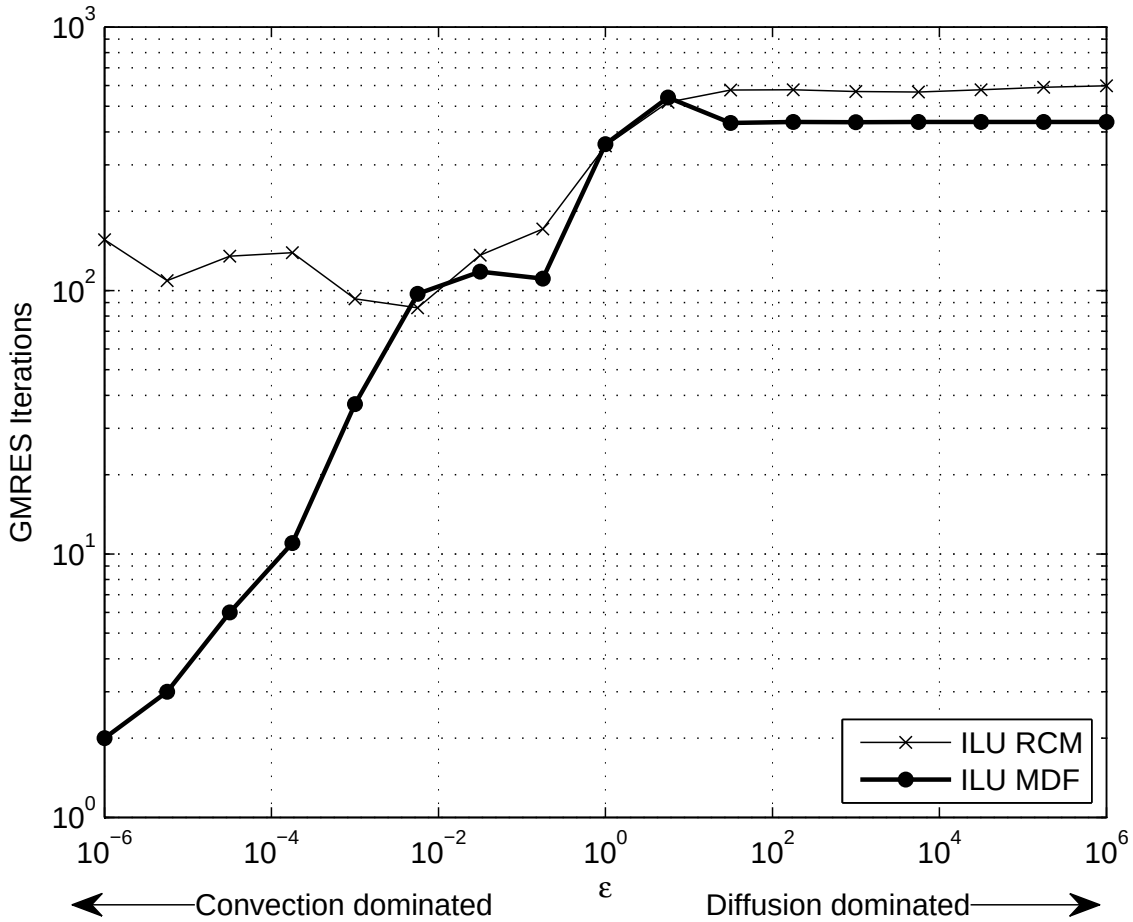
$$\frac{\partial u}{\partial t} + \nabla \cdot \begin{bmatrix} \alpha u \\ \beta u \end{bmatrix} - \nabla \cdot \begin{bmatrix} \varepsilon u_x \\ \varepsilon u_y \end{bmatrix} = 0$$





Effect of Ordering on Convection-Diffusion

- Reverse Cuthill-McKee vs. Minimum Discarded Fill ordering
- MDF makes ILU perfect for convection
- Ordering less important for diffusion, ILU remains poor



The ILU(0)-p1 Preconditioner

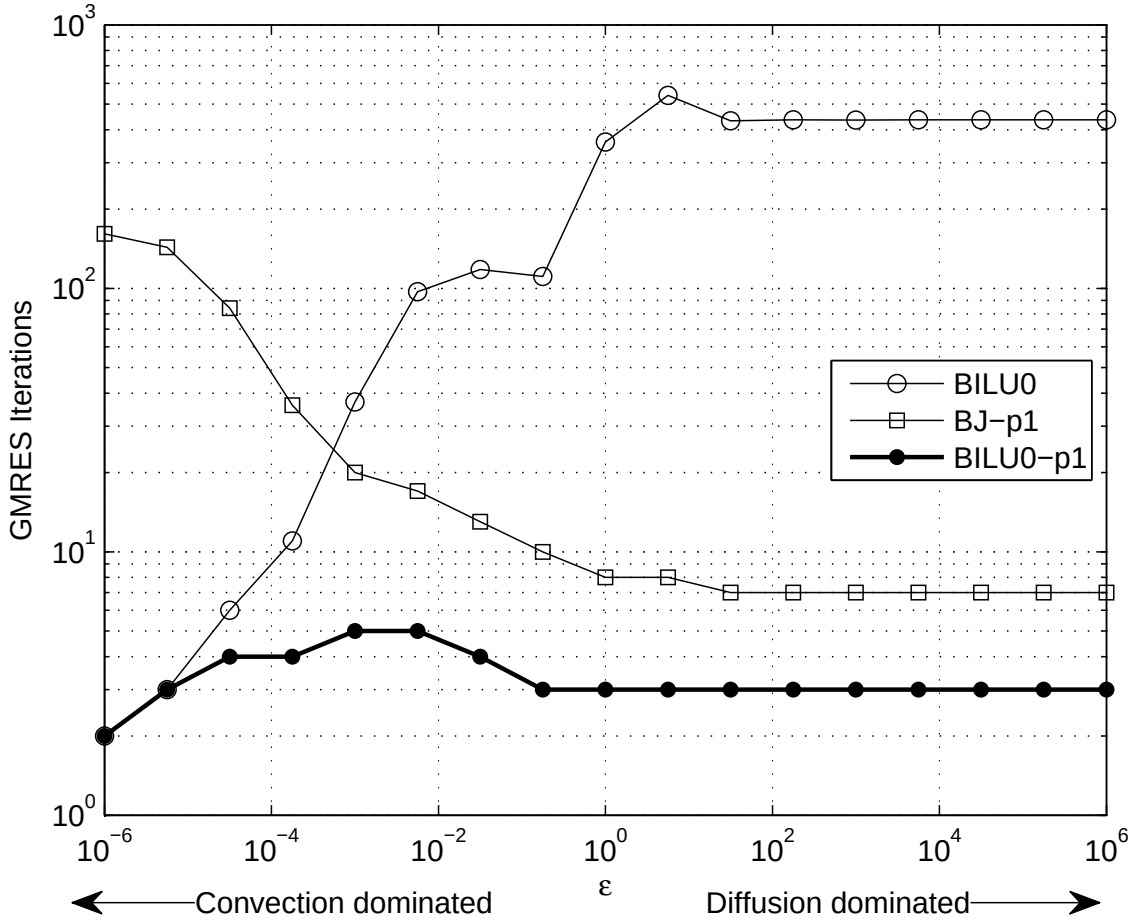
- Combination of block ILU and multigrid [Persson/Peraire 06]
- Coarse scale correction + post-smoothing by $\tilde{\boldsymbol{A}} = \tilde{\boldsymbol{L}}\tilde{\boldsymbol{U}}$:

0. $\boldsymbol{A}^{(0)} = \boldsymbol{P}^T\boldsymbol{A}\boldsymbol{P}$	*Precompute coarse operator, block wise*
1. $\boldsymbol{b}^{(0)} = \boldsymbol{P}^T\boldsymbol{b}$	*Restrict residual element/component wise*
2. $\boldsymbol{A}^{(0)}\boldsymbol{u}^{(0)} = \boldsymbol{b}^{(0)}$	*Solve coarse scale problem*
3. $\boldsymbol{u} = \boldsymbol{P}\boldsymbol{u}^{(0)}$	*Prolongate solution element/component wise*
4. $\boldsymbol{u} = \boldsymbol{u} + \alpha\tilde{\boldsymbol{A}}^{-1}(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{u})$	*Apply smoother* $\tilde{\boldsymbol{A}}$ *with damping* α

- Restriction/prolongation operator $\boldsymbol{P}$ block diagonal, based on orthogonal Koornwinder polynomials
- Coarse scale problem solved directly (2-D problems, serial) or iteratively by GMRES or h-multigrid

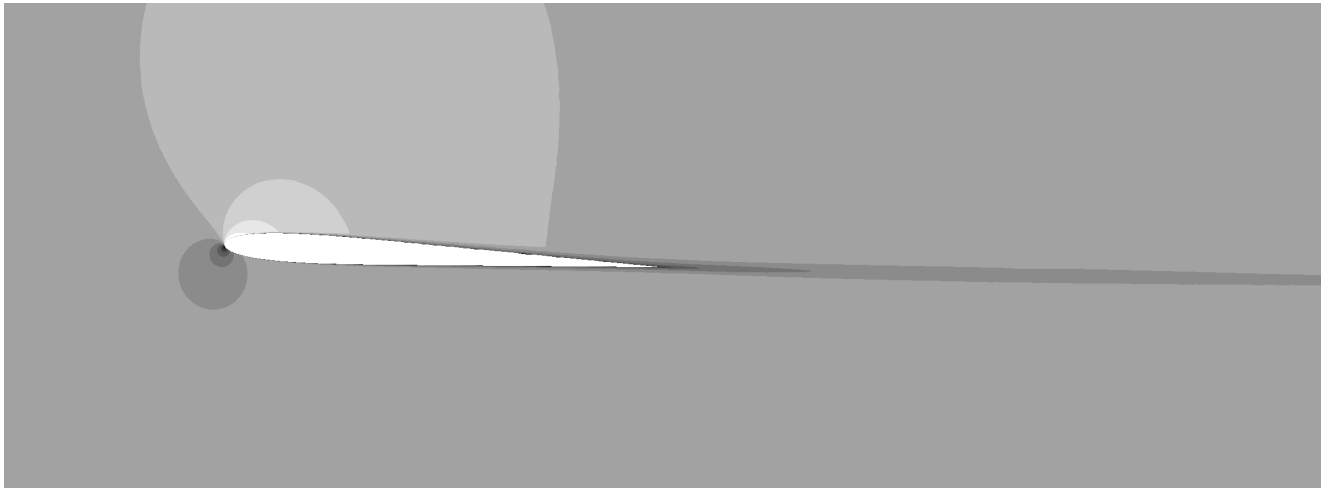
The ILU-p1 Preconditioner - Convection-Diffusion

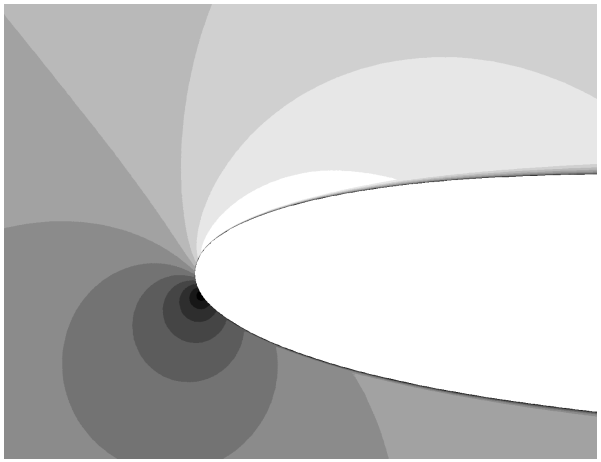
- Block ILU perfect for convection, multigrid perfect for diffusion
- Block ILU-smoothed multigrid (BILU0-p1) almost perfect for any ε

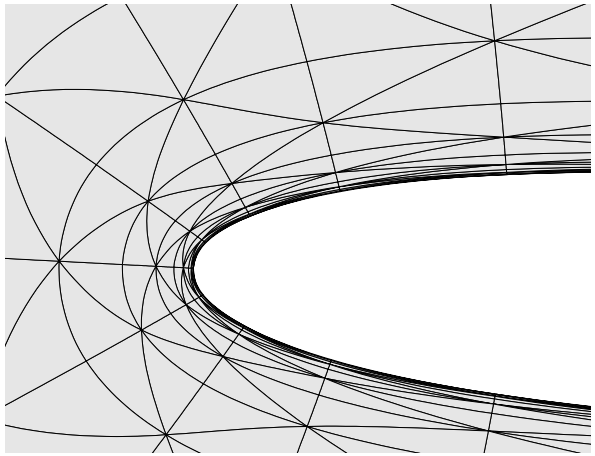


Effect of Ordering on Navier-Stokes

- Model Navier-Stokes problem for wide range of Reynolds numbers







The ILU-p1 Preconditioner – Navier-Stokes

- Convergence for various Reynolds numbers, Δt, and M
- $\times =$ No convergence after 1,000 iterations

Problem	Parameters		Preconditioner/Iterations		
	Δt	M	Block Jacobi-p1	Block GS-p1	Block ILU0-p1
Inviscid	10^{-3}	0.2	19	11	4
	10^{-1}	0.2	85	49	6
	∞	0.2	142	72	9
	10^{-3}	0.01	112	67	6
	10^{-1}	0.01	$\times$	532	10
	∞	0.01	$\times$	$\times$	16
Laminar Re=1,000	10^{-3}	0.2	34	19	4
	10^{-1}	0.2	597	225	5
	∞	0.2	$\times$	$\times$	7
	10^{-3}	0.01	66	33	5
	10^{-1}	0.01	619	207	9
	∞	0.01	$\times$	748	12

The ILU-p1 Preconditioner – Navier-Stokes

- Convergence for various Reynolds numbers, Δt, and M
- $\times =$ No convergence after 1,000 iterations

Problem	Parameters		Preconditioner/Iterations		
	Δt	M	Block Jacobi-p1	Block GS-p1	Block ILU0-p1
Laminar	10^{-3}	0.2	19	11	4
Re=20,000	10^{-1}	0.2	220	113	8
	∞	0.2	$\times$	$\times$	20
	10^{-3}	0.01	90	38	6
	10^{-1}	0.01	$\times$	735	16
	∞	0.01	$\times$	$\times$	35
RANS	10^{-3}	0.2	56	28	7
Re=10^6	10^{-1}	0.2	$\times$	$\times$	25
	∞	0.2	$\times$	$\times$	18
	10^{-3}	0.01	231	110	9
	10^{-1}	0.01	$\times$	$\times$	16
	∞	0.01	$\times$	$\times$	28

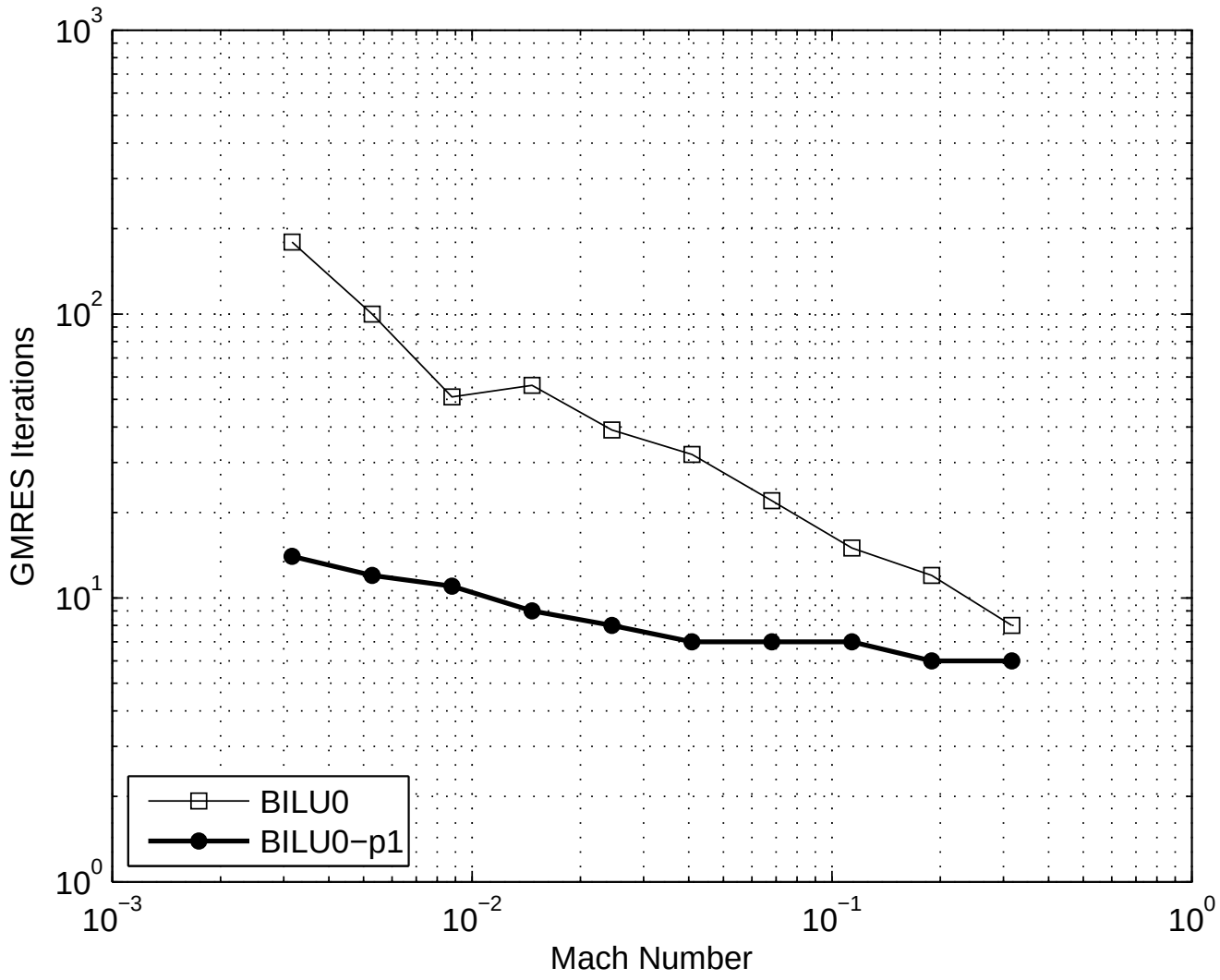
Effect of Ordering on Navier-Stokes

- Good element ordering critical for Block ILU/multigrid solver

Problem	Element Ordering		
	Random	RCM	MDF
Inviscid	51	27	12
Laminar, Re=1,000	200	135	12
Laminar, Re=20,000	197	139	27
RANS, Re=10^6	98	99	18

Mach Number Dependence

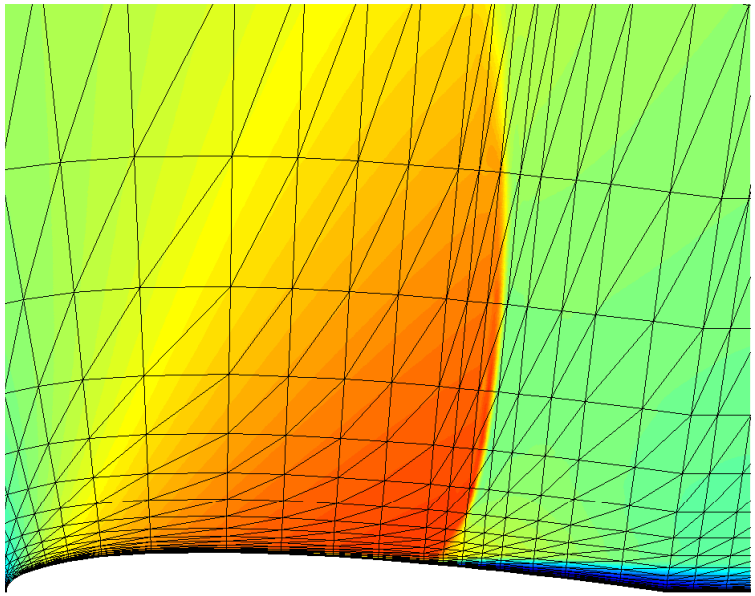
- Excellent convergence for Mach number $M << 1$ with BILU0-p1



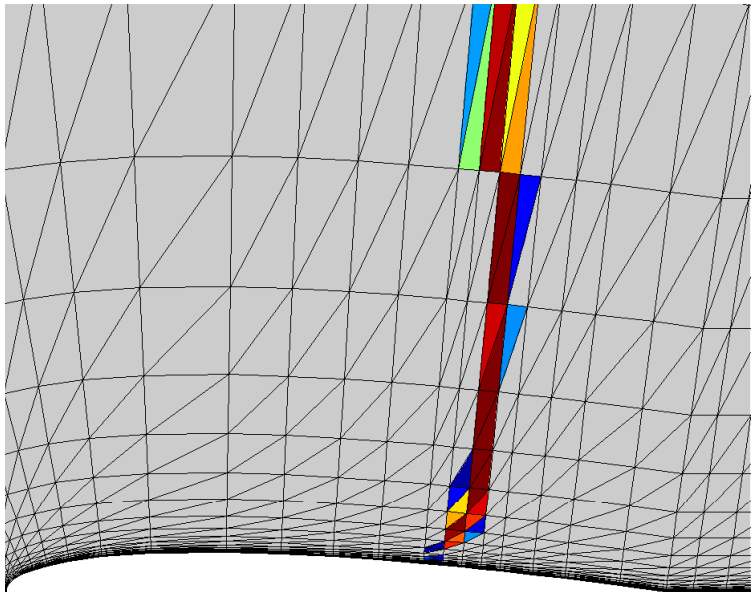
Outline

Artificial Viscosity for Underresolved Features

- Cannot resolve all solution features (shocks, RANS, singularities)
- Low dissipation makes DG sensitive to underresolution
- Detect by sensors and add viscosity [Persson/Peraire 06,07]
- Enables shock capturing with sub-cell resolution and robust solution of Spalart-Alamaras RANS model



Mach



Sensor

Shock Sensor

- Regularity of solution determined from the decay rate of expansion coefficients in orthogonal basis
- Example: Periodic Fourier case: $f(x) = \sum_{k=-\infty}^{\infty} g_k e^{ikx}$
  If $f(x)$ has m continuous derivatives $\rightarrow$ $|g_k| \sim k^{-(m+1)}$
- For simplices: Expand solution in orthonormal Koornwinder basis:

$$u = \sum_{i=1}^{N(p)} u_i \psi_i, \quad \hat{u} = \sum_{i=1}^{N(p-1)} u_i \psi_i, \quad s_e = \log_{10}\left(\frac{(u-\hat{u}, u-\hat{u})_e}{(u,u)_e}\right)$$

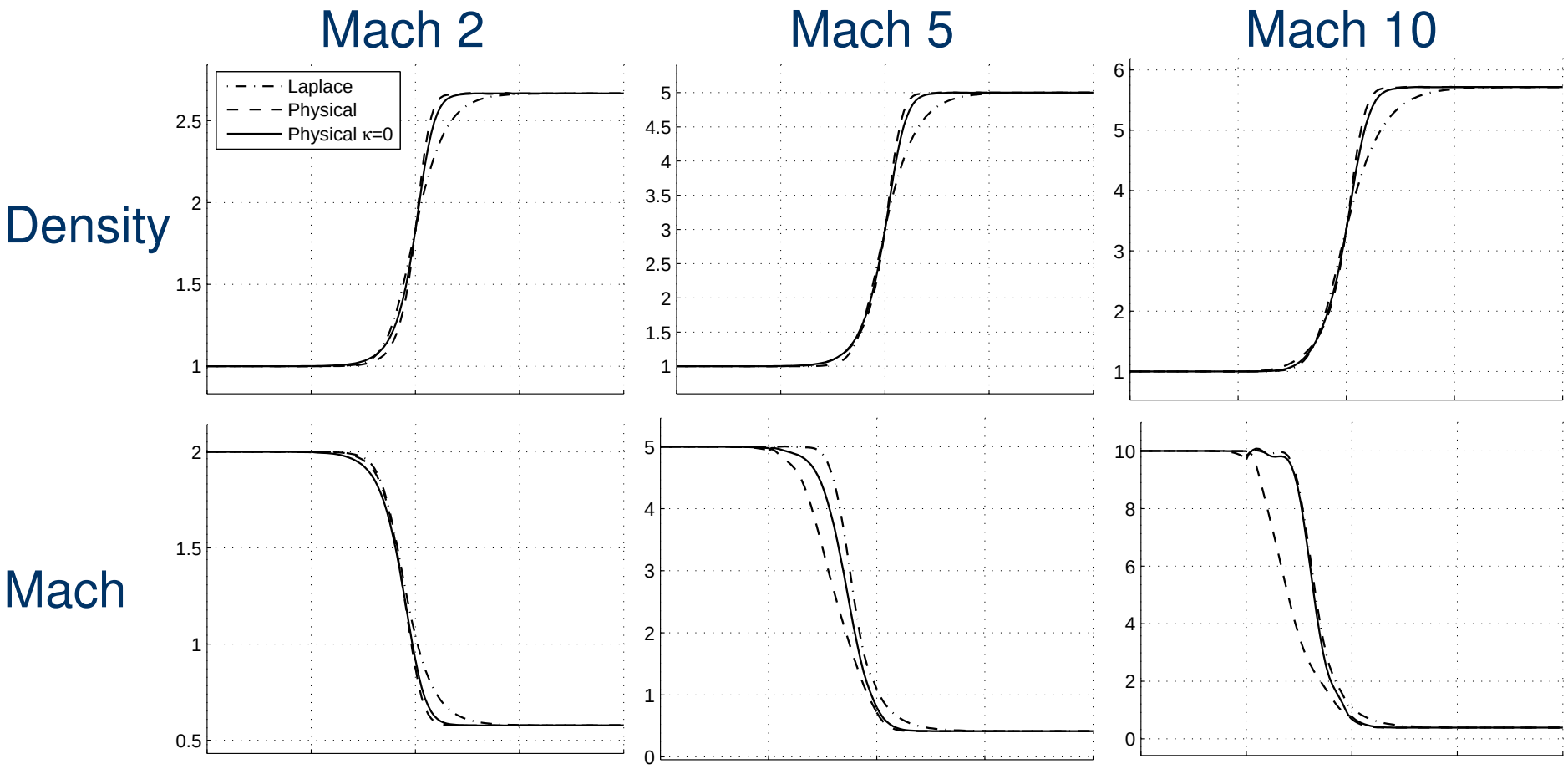
- Determine elemental piecewise constant ε_e

$$\varepsilon_e = \begin{cases} 0 & \text{if} \quad s_e < s_0 - \kappa \\ \frac{\varepsilon_0}{2}\left(1 + \sin\frac{\pi(s_e - s_0)}{2\kappa}\right) & \text{if} \quad s_0 - \kappa \leq s_e \leq s_0 + \kappa \\ \varepsilon_0 & \text{if} \quad s_e > s_0 + \kappa \end{cases} .$$

where $\varepsilon_0 \sim h/p$, $s_0 \sim 1/p^4$ and κ empirical

Euler Equations – Artificial Viscosity Models

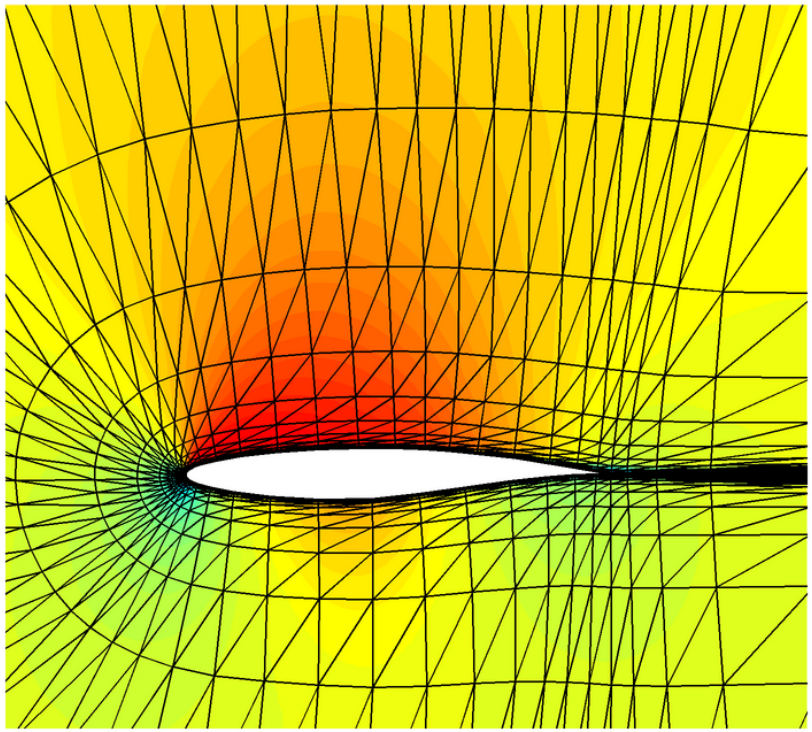
- Laplacian: $\frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot \mathcal{F}(\boldsymbol{u}) = \nabla \cdot (\epsilon \nabla \boldsymbol{u})$
- Physical: $\frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot \mathcal{F}(\boldsymbol{u}) = \nabla \cdot \mathcal{F}_v(\boldsymbol{u}; \mathrm{Re}, \mathrm{Pr})$

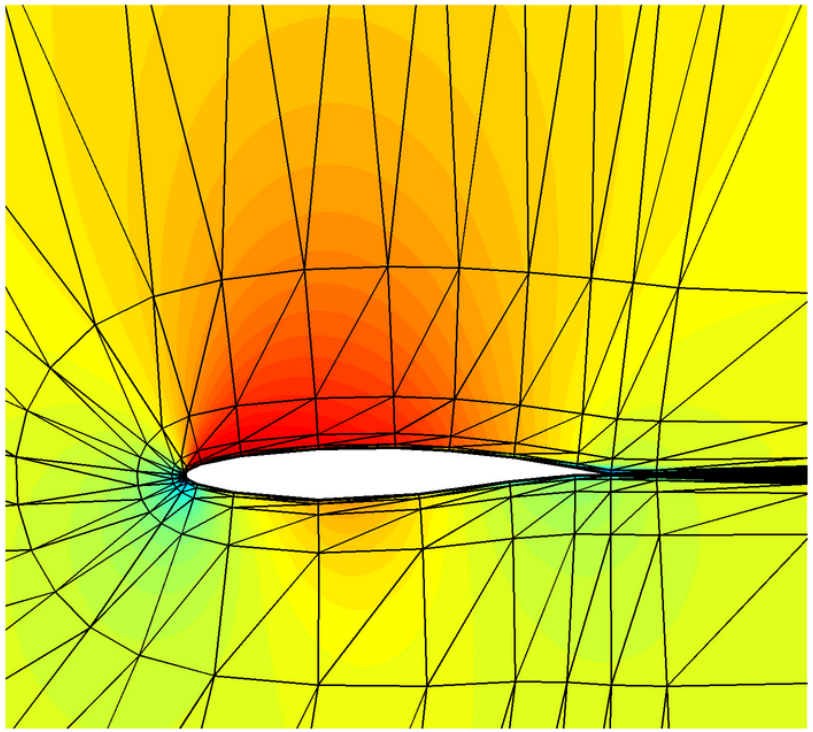


Example: RAE2822

- Turbulent RANS flow ($M = 0.675, \alpha = 2.31^o, \mathrm{Re} = 6.5 \cdot 10^6$)
- p-converged solution, fixed resolution h/p

$p = 2$ (constant h/p) $p = 4$





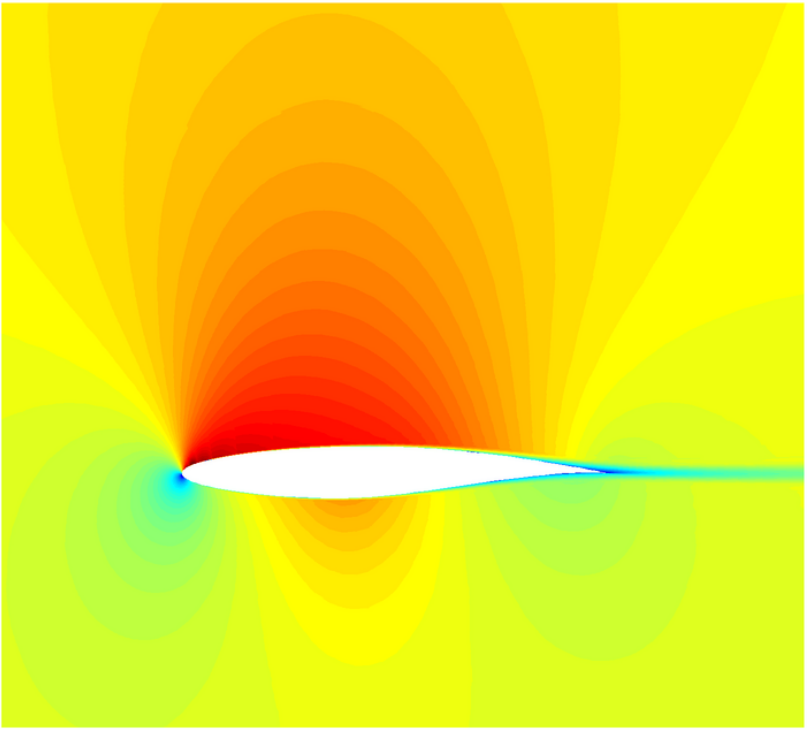
$C_L = 0.6144 \quad C_D = 0.0104$

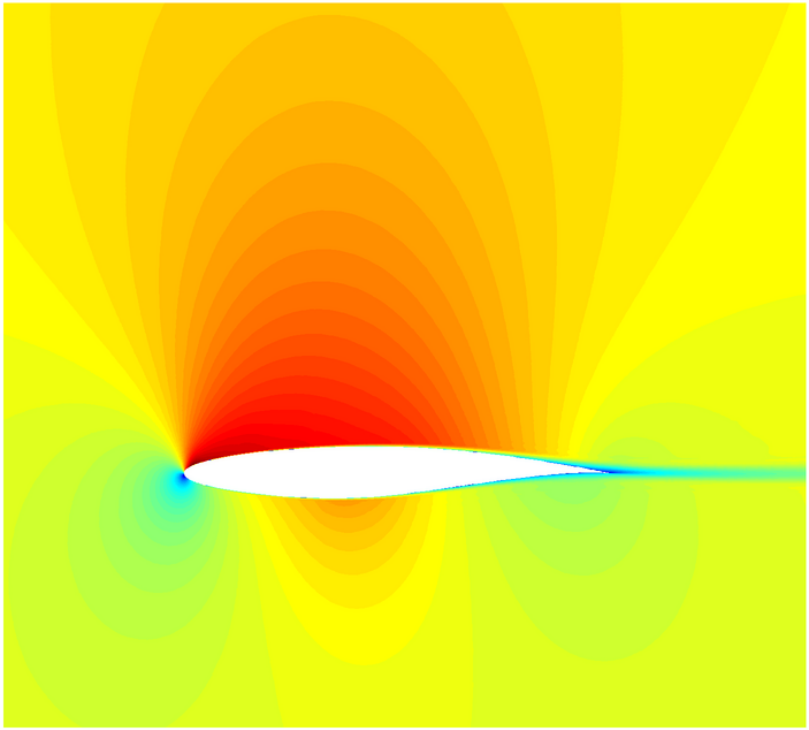
$C_L = 0.6131 \quad C_D = 0.0103$

Example: RAE2822

- Turbulent RANS flow ($M = 0.675, \alpha = 2.31^o, \mathrm{Re} = 6.5 \cdot 10^6$)
- p-converged solution, fixed resolution h/p

$p = 2$ (constant h/p) $p = 4$





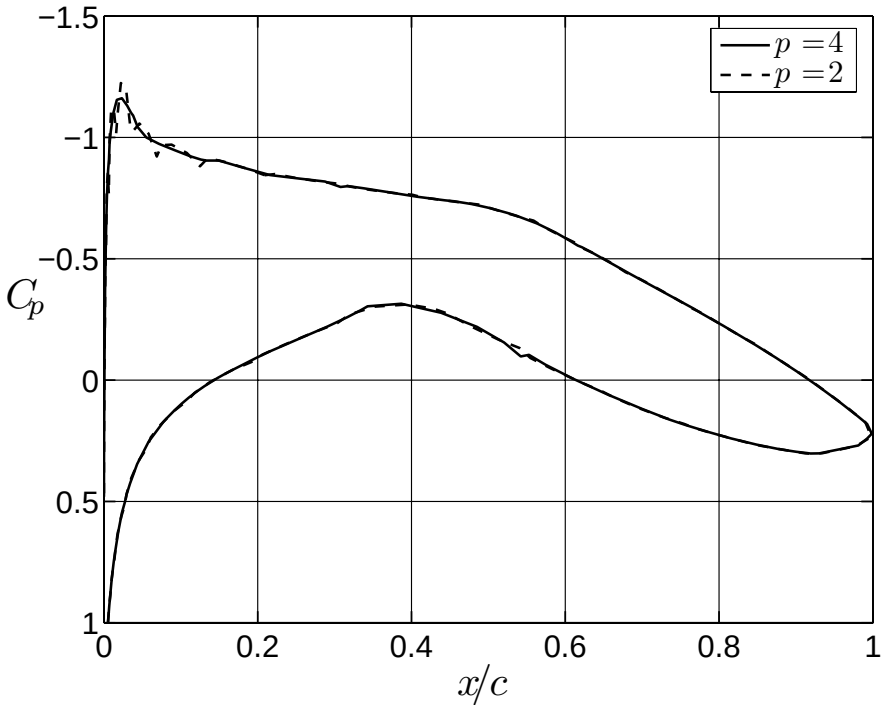
$C_L = 0.6144 \quad C_D = 0.0104$

$C_L = 0.6131 \quad C_D = 0.0103$

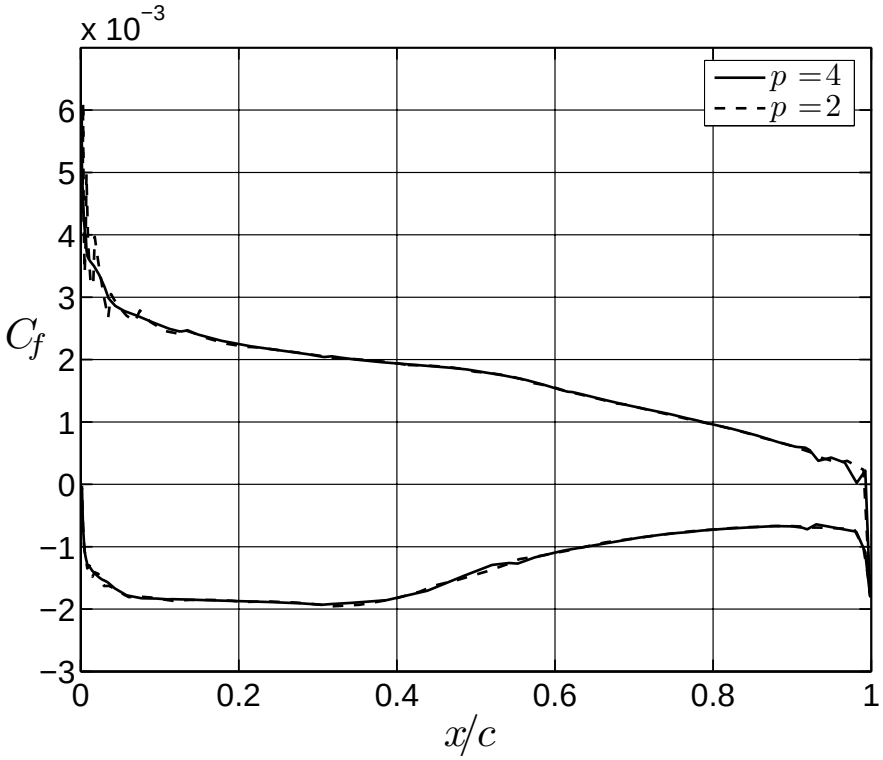
Example: RAE2822

- Highly accurate boundary forces even with coarse meshes

C_p



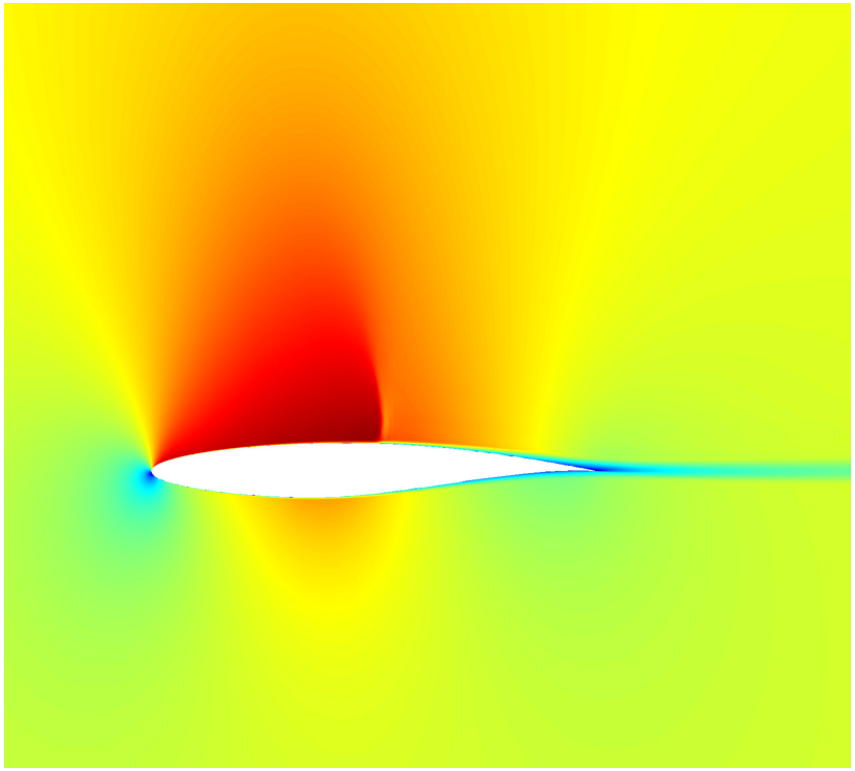
C_f

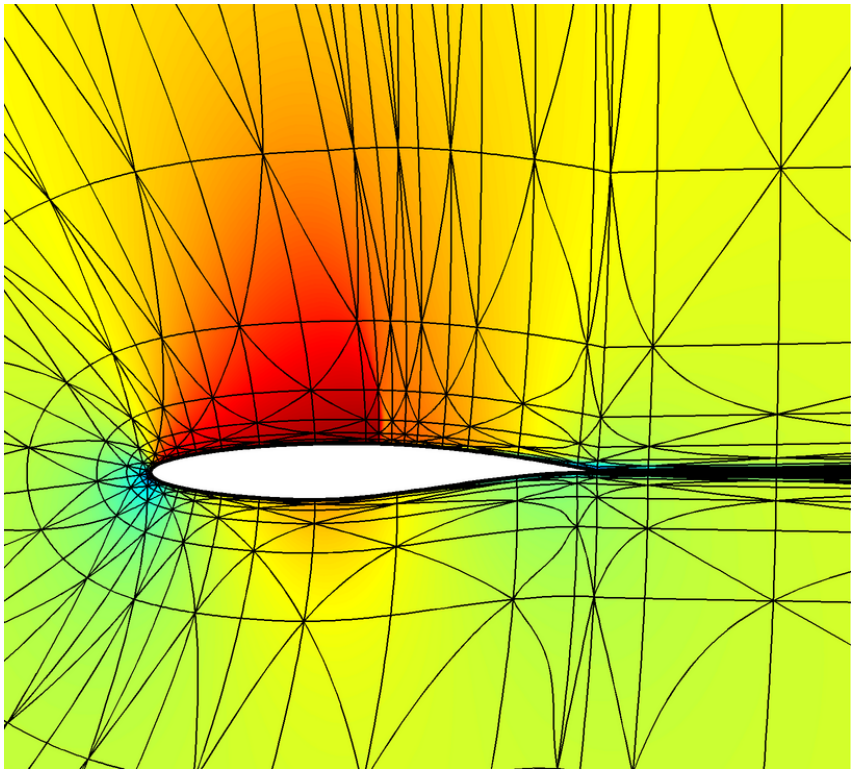


Example: RAE2822, Transonic

- Transonic flow ($M = 0.729, \mathrm{Re} = 6.5 \cdot 10^6$)
- Sub-cell resolution of shocks

$p = 4$

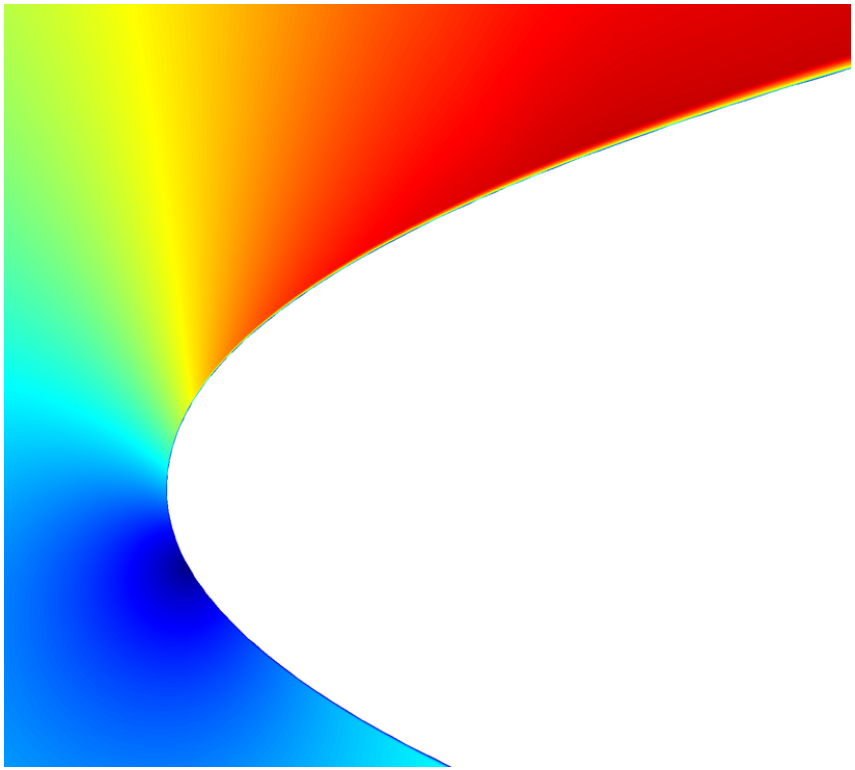


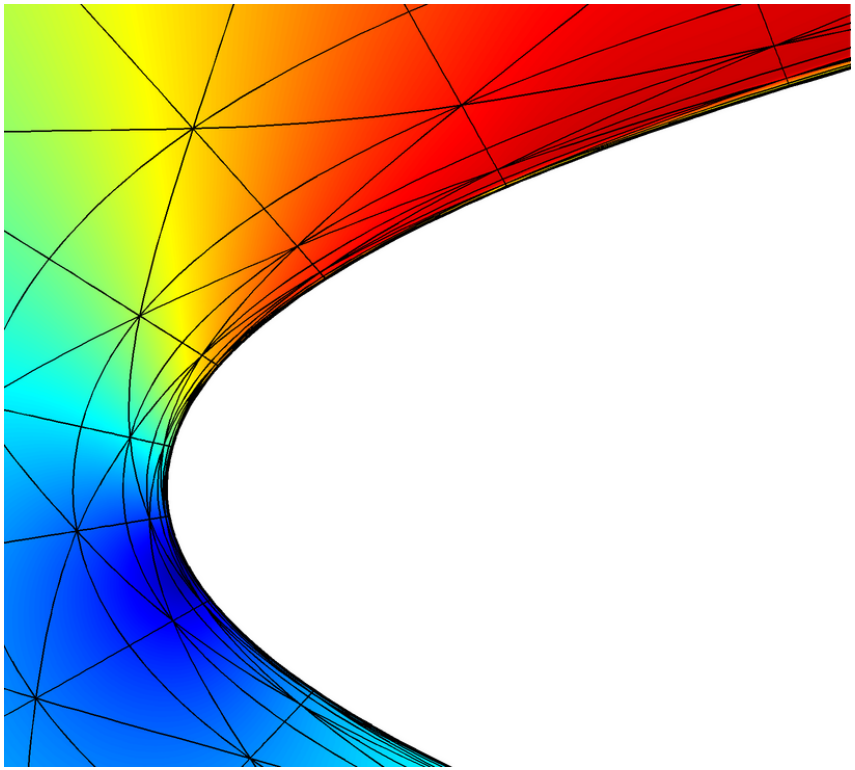


Example: RAE2822, Transonic

- Transonic flow ($M = 0.729, \mathrm{Re} = 6.5 \cdot 10^6$)
- Sub-cell resolution of shocks

$p = 4$





Outline

Methods for Deforming Domains

- Many ALE formulations for unstructured meshes [Venkatasubban 95], [Lovtev et al 99], [Farhat/Geuzaine 04], [Ahn/Kallinderis 07])
  - Equations discretized on a deforming grid, time-dependent metric
  - At most third order accuracy in space and time demonstrated
- Alternative approach for finite differences [Visbal/Gaitonde 02]
  - Map from fixed reference domain to time-varying physical domain
  - Correction for the Geometric Conservation Law (GCL) with sources
- Our method: Mapping approach in a DG setting, with a conservative formulation for satisfying the GCL
  - Guaranteed stability
  - Arbitrary orders of accuracy in space and time

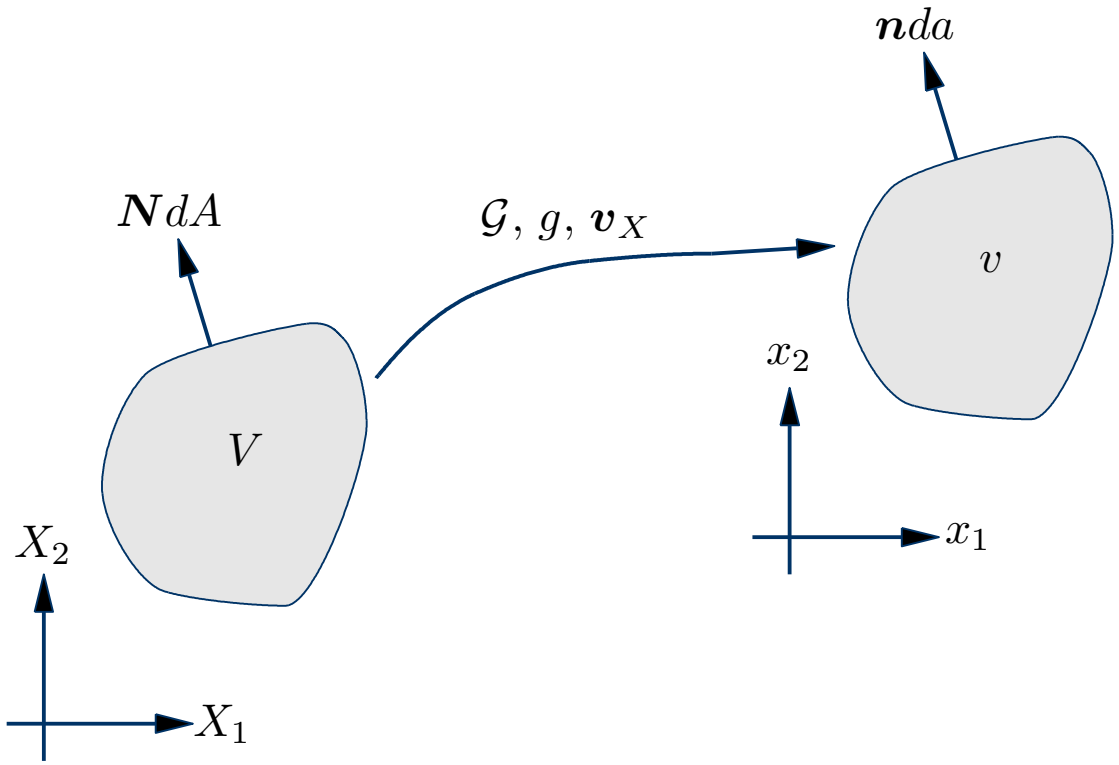
ALE Formulation

- Map from reference domain V to physical deformable domain $v(t)$
- A point $\boldsymbol{X}$ in V is mapped to a point $\boldsymbol{x}(t) = \boldsymbol{G}(\boldsymbol{X}, t)$ in $v(t)$
- Introduce the *mapping deformation gradient* $\boldsymbol{G}$ and the *mapping velocity* $\boldsymbol{v}_X$ as

$$\boldsymbol{G} = \boldsymbol{\nabla}_X \boldsymbol{G}$$

$$\boldsymbol{v}_X = \left.\frac{\partial \boldsymbol{G}}{\partial t}\right|_X$$

and set $g = \det(\boldsymbol{G})$



Transformed Equations

- The system of conservation laws in the physical domain $v(t)$

$$\left.\frac{\partial \boldsymbol{U}_x}{\partial t}\right|_x + \boldsymbol{\nabla}_x \cdot \boldsymbol{F}_x(\boldsymbol{U}_x, \boldsymbol{\nabla}_x \boldsymbol{U}_x) = 0$$

can be written in the reference configuration V as

$$\left.\frac{\partial \boldsymbol{U}_X}{\partial t}\right|_X + \boldsymbol{\nabla}_X \cdot \boldsymbol{F}_X(\boldsymbol{U}_X, \boldsymbol{\nabla}_X \boldsymbol{U}_X) = 0$$

where

$$\boldsymbol{U}_X = g\boldsymbol{U}_x \ , \qquad \boldsymbol{F}_X = g\boldsymbol{G}^{-1}\boldsymbol{F}_x - \boldsymbol{U}_X\boldsymbol{G}^{-1}\boldsymbol{v}_X$$

and

$$\boldsymbol{\nabla}_x \boldsymbol{U}_x = \boldsymbol{\nabla}_X(g^{-1}\boldsymbol{U}_X)\boldsymbol{G}^{-T} = (g^{-1}\boldsymbol{\nabla}_X \boldsymbol{U}_X - \boldsymbol{U}_X \boldsymbol{\nabla}_X(g^{-1}))\boldsymbol{G}^{-T}$$

- *Proof.* See [Persson/Peraire/Bonet 07]

Geometric Conservation Law

- A constant solution in $v(t)$ is not necessarily a solution in V, due to inexact integration of the Jacobian g
  - The time evolution of g is

$$\left.\frac{\partial g}{\partial t}\right|_X - \boldsymbol{\nabla}_X \cdot (g\boldsymbol{G}^{-1}\boldsymbol{v}_X) = 0 \ ,$$

    which in general is non-zero
- Visbal and Gaitonde added source terms to cancel the errors
- Our approach solves instead the conservative system

$$\left.\frac{\partial(\bar{g}g^{-1}\boldsymbol{U}_X)}{\partial t}\right|_X - \boldsymbol{\nabla}_X \cdot \boldsymbol{F}_X = \boldsymbol{0}$$

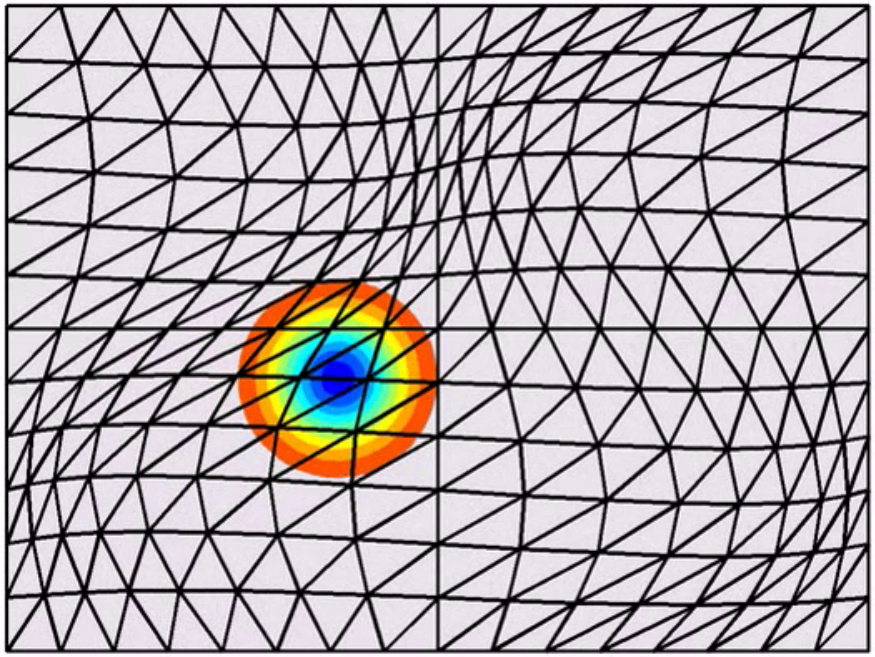
$$\left.\frac{\partial \bar{g}}{\partial t}\right|_X - \boldsymbol{\nabla}_X \cdot (g\boldsymbol{G}^{-1}\boldsymbol{v}_X) = 0$$

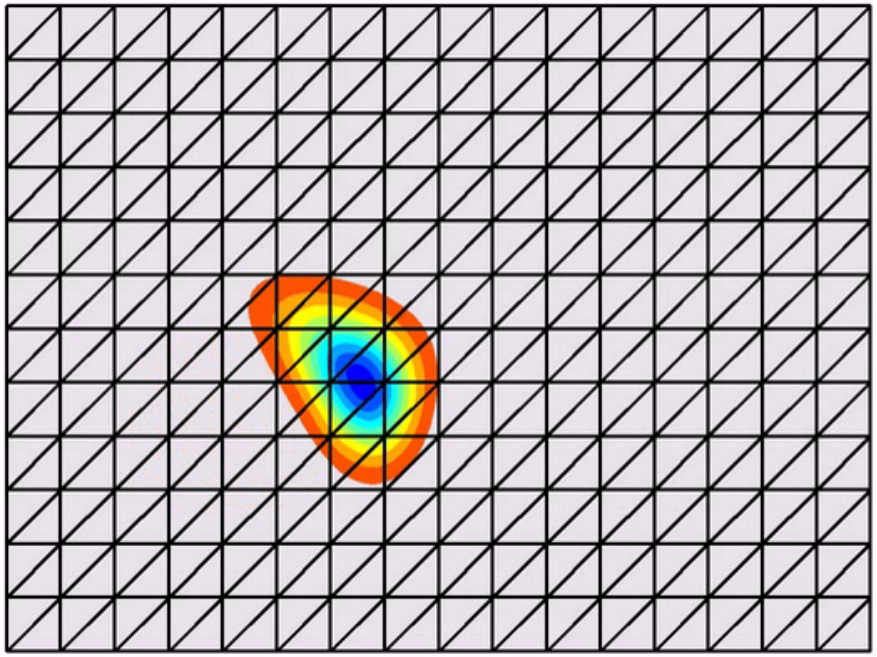
Example: Euler Vortex

- Propagate an Euler vortex on a variable domain with

$$x(\xi,\eta,t) = \xi + 2.0\sin(2\pi\xi/20)\sin(\pi\eta/7.5)\sin(1.0\cdot 2\pi t/t_0)$$
$$y(\xi,\eta,t) = \eta + 1.5\sin(2\pi\xi/20)\sin(\pi\eta/7.5)\sin(2.0\cdot 2\pi t/t_0)$$

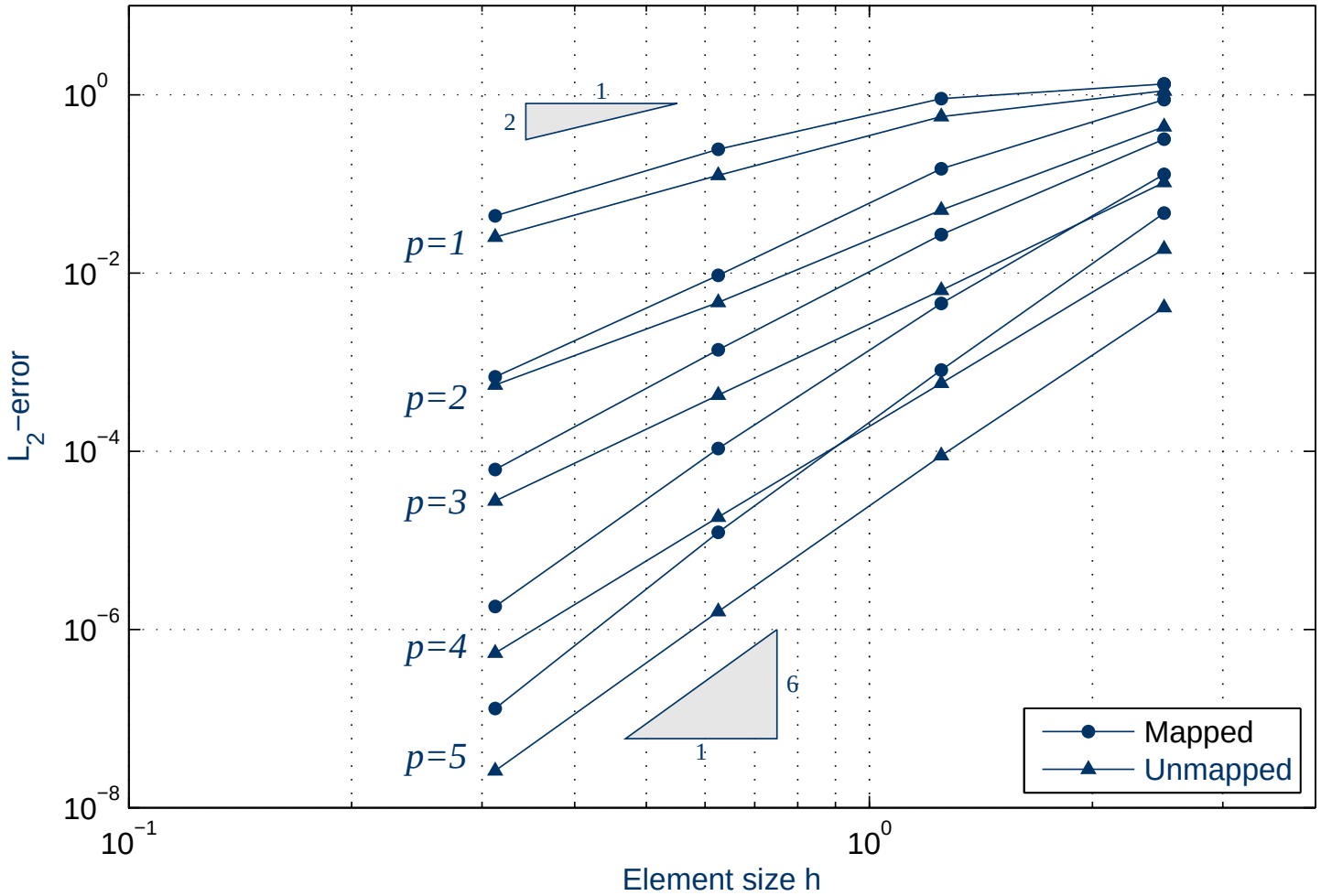
- Mapped scheme – Everything is computed on the reference mesh





Example: Euler Vortex, Convergence

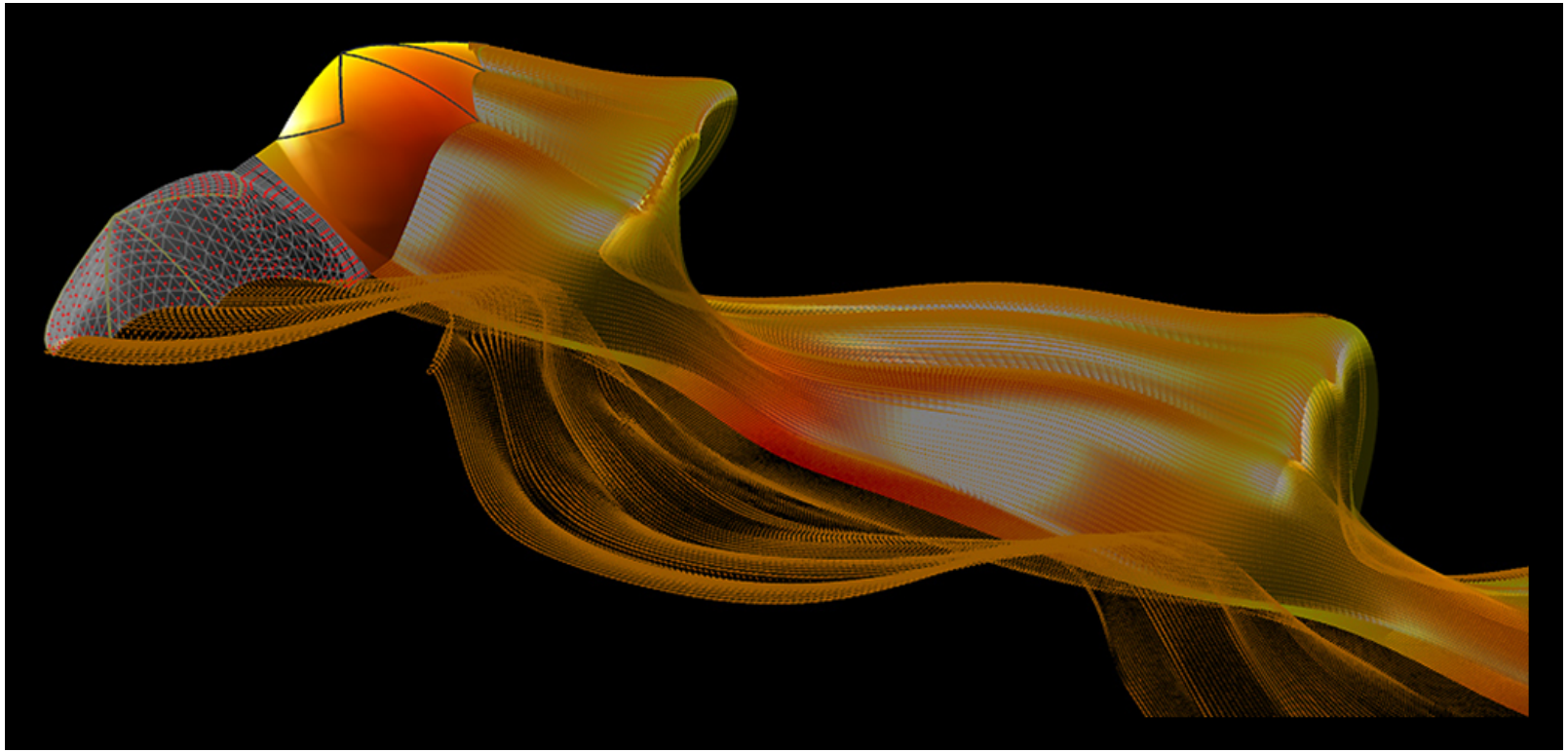
- Optimal order of convergence $O(h^{p+1})$ for mapped scheme



Outline

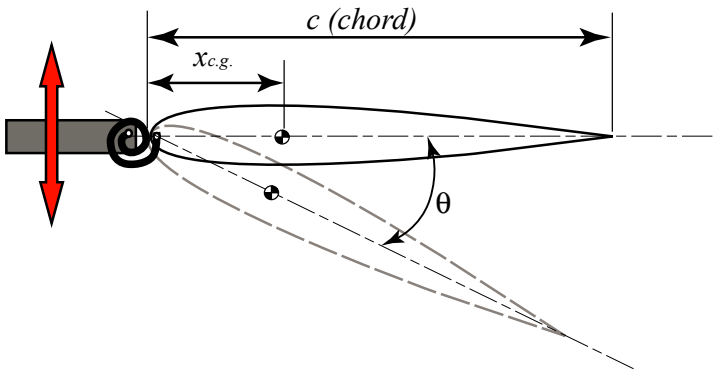
Biologically-Inspired Flapping Flight

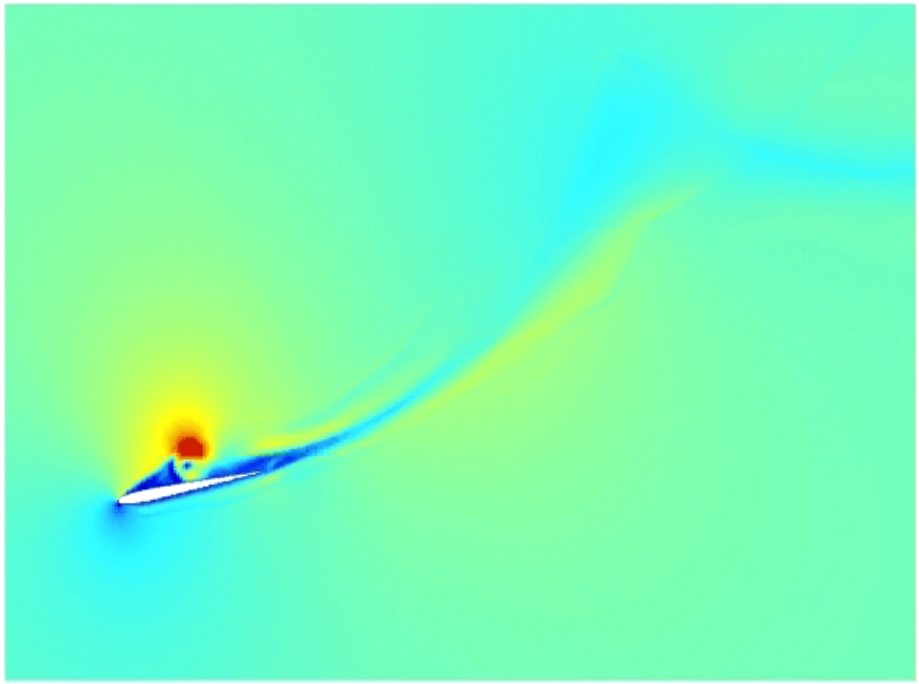
- Development of computational tools for studying flapping flight
- Challenging problems: Deforming domains, fluid-structure interaction, transitional flows, etc



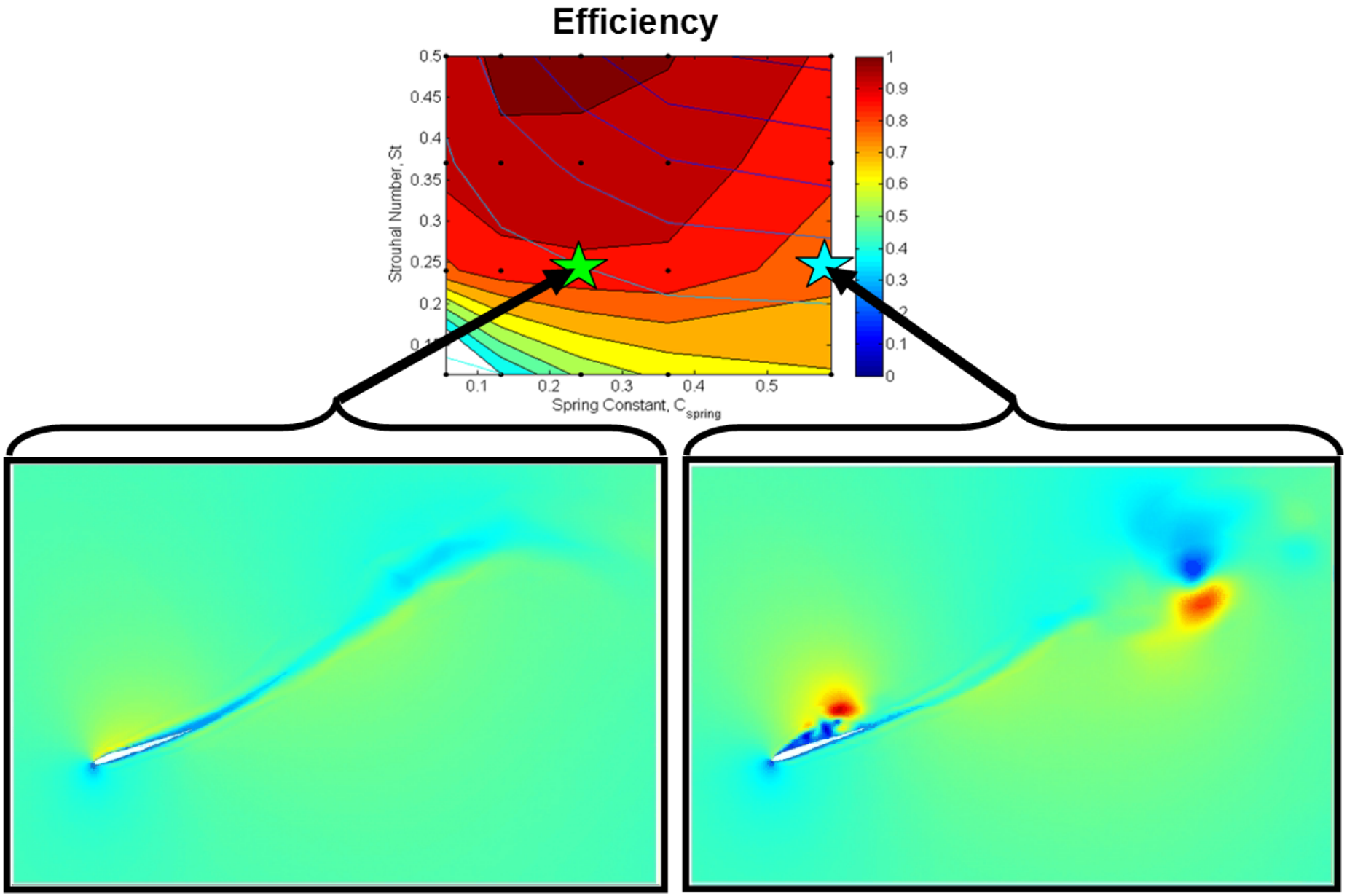
Example: Pitching Airfoil

- Airfoil attached to translating and heaving point by torsional spring
- Fluid properties: $\mathrm{Re} = 5000$, $M = 0.2$
- Forced vertical motion $r_z(t) = r_0 \sin \omega t$ (at leading edge)
- Moment equation: $I\ddot{\theta} + C\theta - S\ddot{r}_z(t) + M_{\mathrm{aero}} = 0$
  - I moment of inertia, C spring stiffness, $S = mx_c$ static unbalance
  - M_{aero} moment from fluid



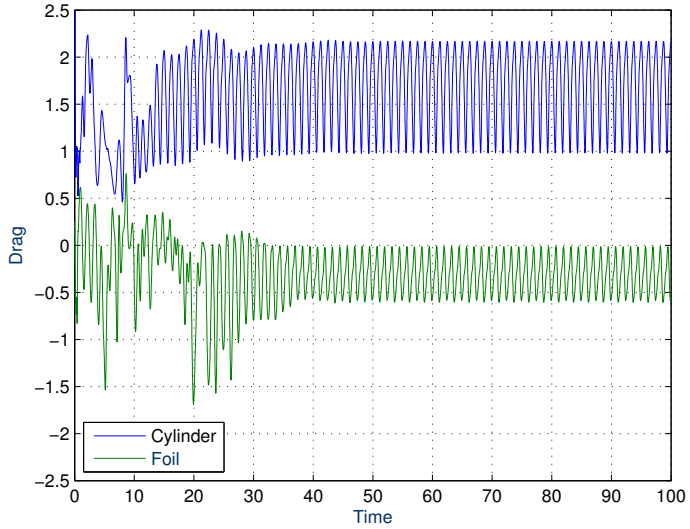


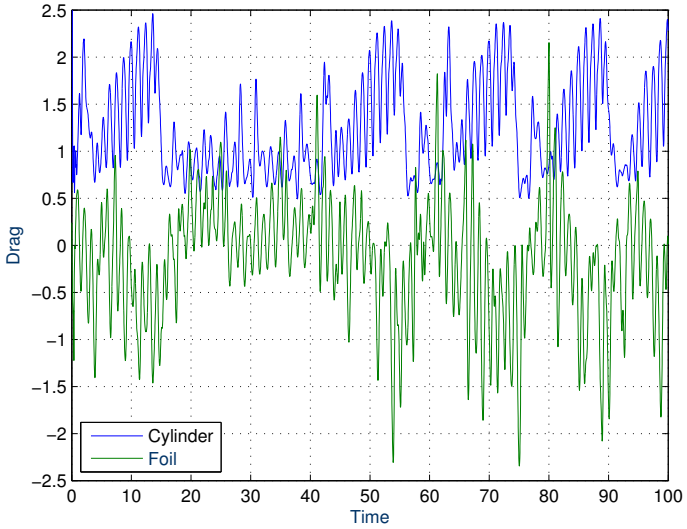
Example: Pitching Airfoil/Flapper Design

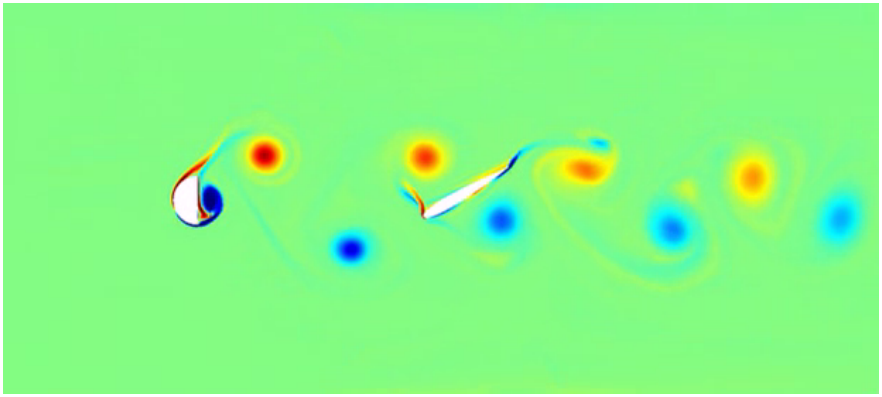


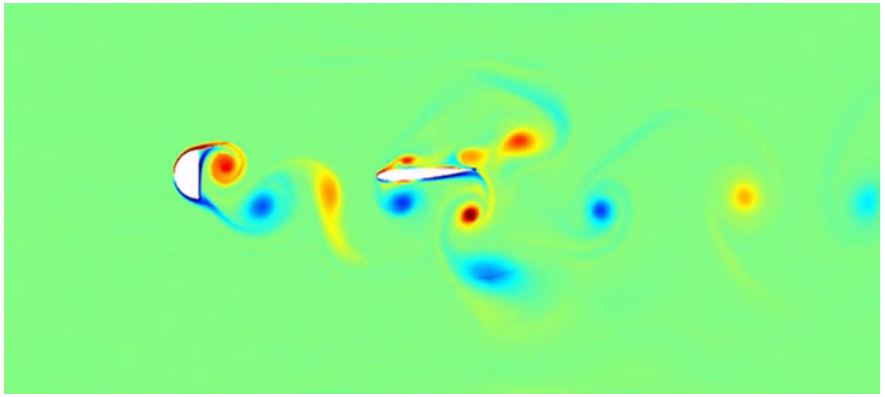
Example: Heaving and Pitching Foil in Wake

- NACA 0012 foil heaving and pitching in wake of D-section cylinder
- Both oscillate $y(t) = A\sin(2\pi ft)$, foil pitching $\theta = a\sin(2\pi ft + \pi/2)$
- Based on experimental study [Gopalkrishnan et al 94]



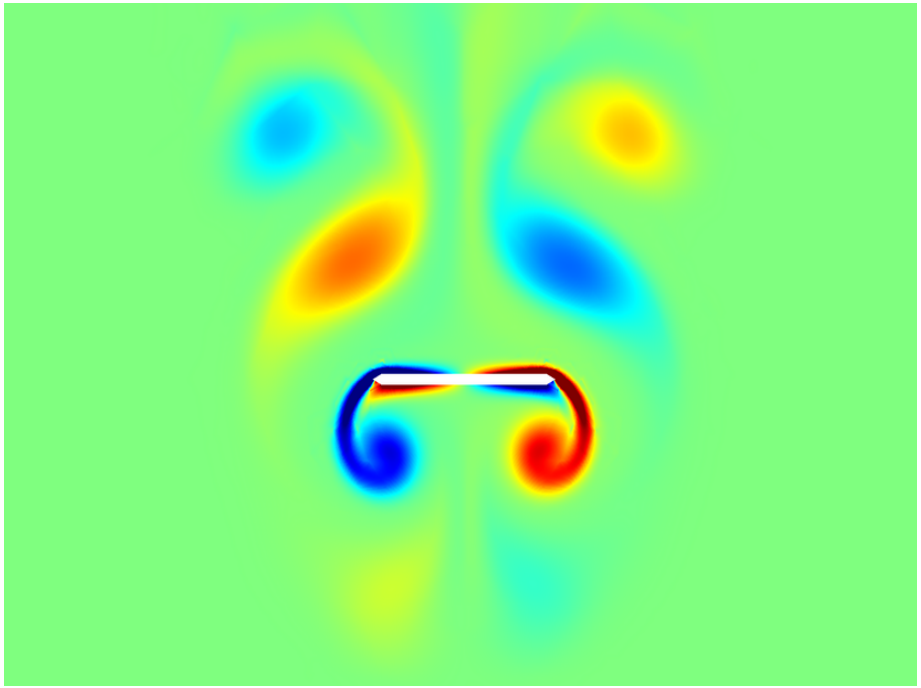


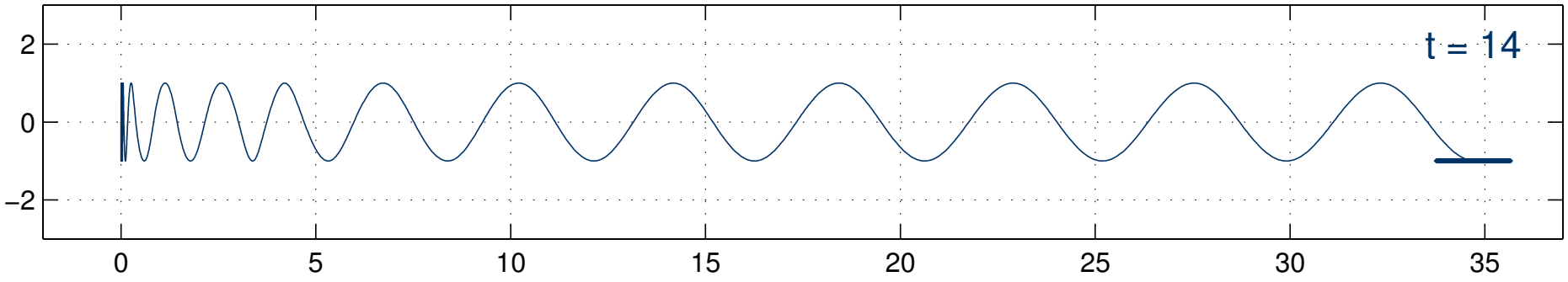




Example: Locomotion of Free Flapping Body

- Oscillating plate, unconstrained horizontally [Vandenberghe et al 04, Alben/Shelley 05]
- Instability breaks symmetry and forces plate into motion

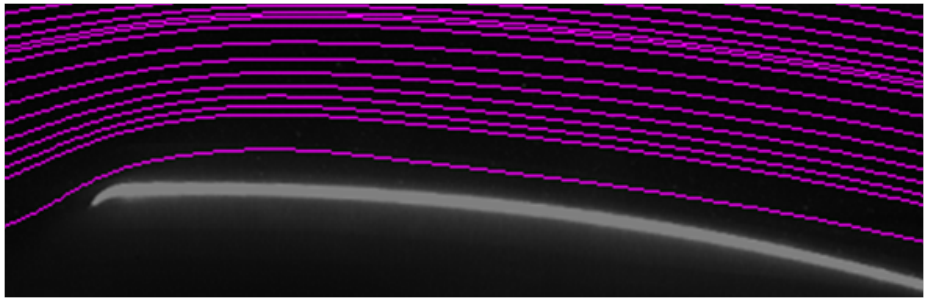




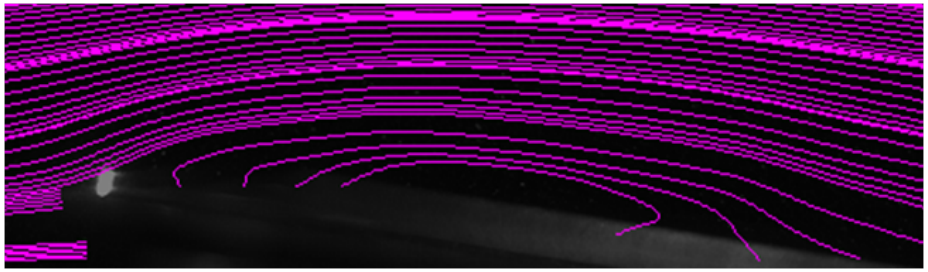
Example: Fluid-Structure Interaction

- Interaction between fluid and a hyperelastic membrane
- Compliancy can alleviate leading edge separation

Experiment (A. Song, Brown U)

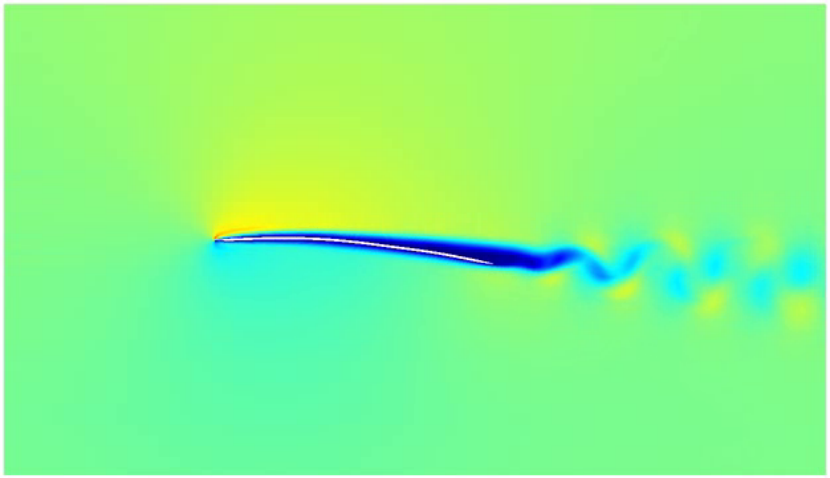


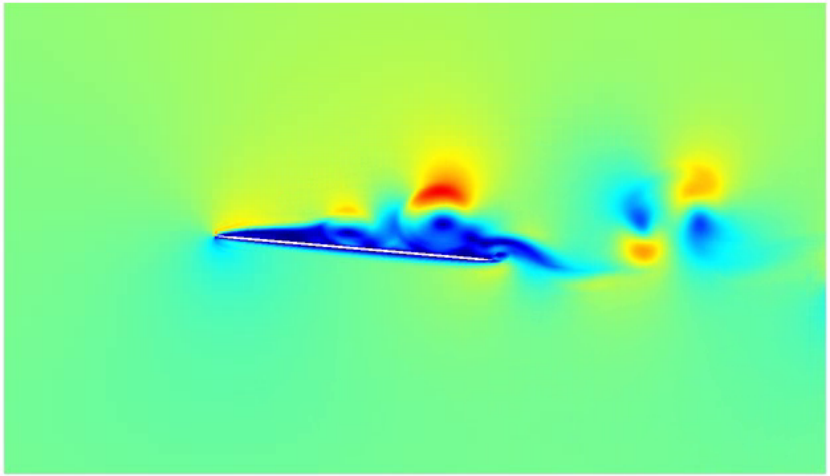
Compliant membrane



Rigid flat plate

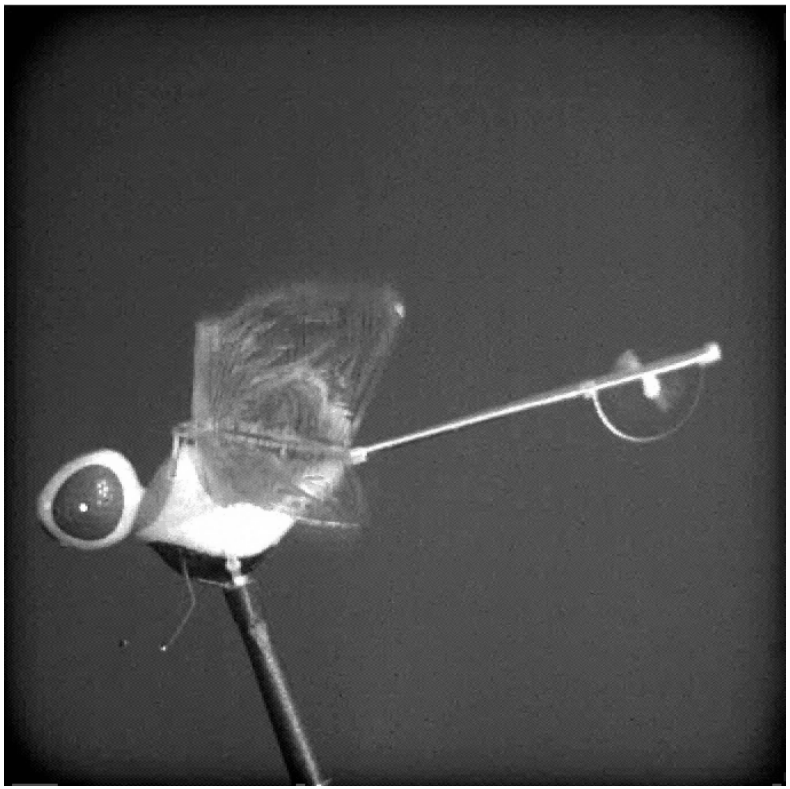
Fluid/membrane simulation



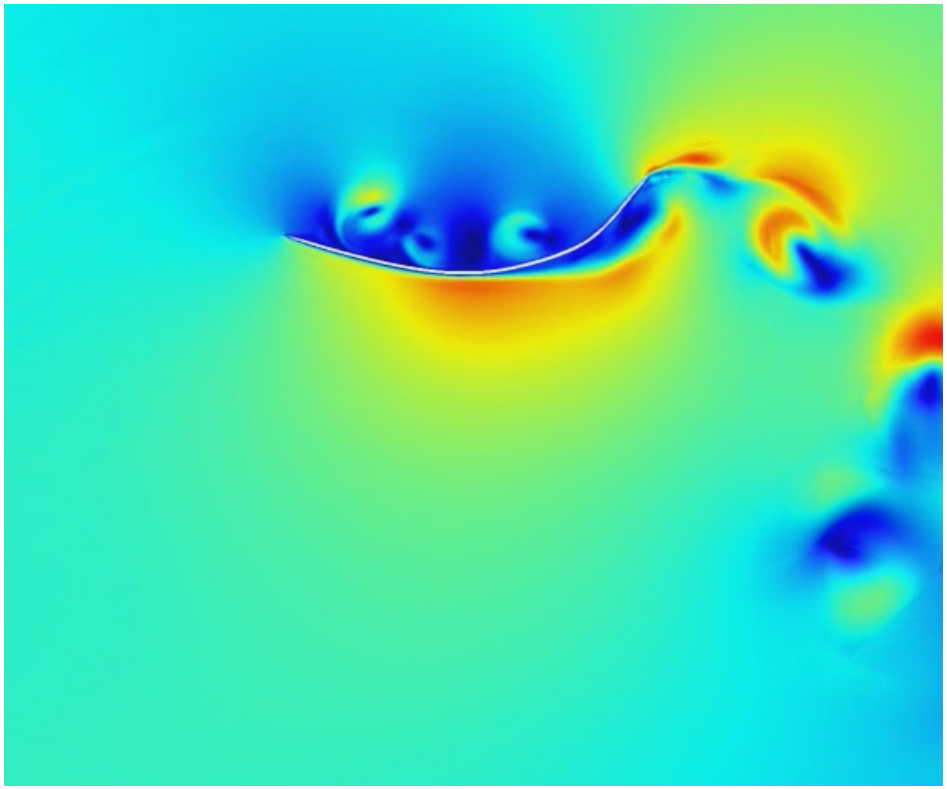


Example: Dragonfly, Compliant Wings

Experiment (A. Song, Brown U)



Fluid/membrane simulation



Outline

Lagrangian DG Formulation for Solid Dynamics

- Governing equations, conservation of linear momentum:

$$\frac{\partial \boldsymbol{p}}{\partial t} - \nabla \cdot \boldsymbol{P} = \rho_0 \boldsymbol{b}$$

  with momentum $\boldsymbol{p}$, first Piola-Kirchhoff stress tensor $\boldsymbol{P}(\boldsymbol{F})$, deformation tensor $\boldsymbol{F} = \partial \boldsymbol{x} / \partial \boldsymbol{X}$
- Hyperelastic neo-Hookean model for $\boldsymbol{P}(\boldsymbol{F})$
- Many potential benefits:
  - Arbitrary orders of accuracy
  - Discontinuous elements less sensitive to locking
  - DG stabilization for ALE
  - Block-diagonal mass matrix $\Longrightarrow$ efficient explicit methods

Lagrangian DG Formulation for Solid Dynamics

- Formulation 1: Material points $\boldsymbol{x}$ and momentum $\boldsymbol{p}$:

$$\frac{\partial \boldsymbol{x}}{\partial t} = \boldsymbol{p}/\rho_0$$
$$\frac{\partial \boldsymbol{p}}{\partial t} - \nabla \cdot \boldsymbol{P}(\boldsymbol{F}) = \rho_0 \boldsymbol{b}$$

  - Cheap, polynomial spaces, CDG scheme for second derivatives
  - Non-conservative, treatment of shocks unclear

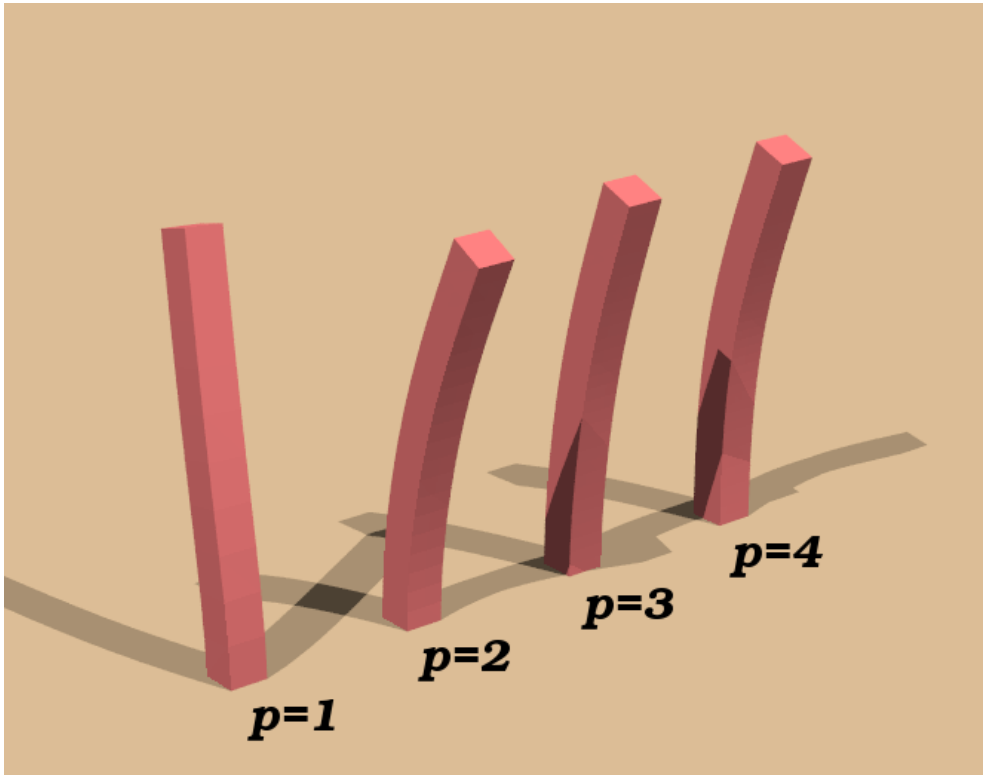
- Formulation 2: Momentum $\boldsymbol{p}$ and deformation gradient $\boldsymbol{F}$:

$$\frac{\partial \boldsymbol{p}}{\partial t} - \nabla \cdot \boldsymbol{P}(\boldsymbol{F}) = \rho_0 \boldsymbol{b}$$
$$\frac{\partial \boldsymbol{F}}{\partial t} - \nabla \cdot \left(\frac{\boldsymbol{p}}{\rho_0} \otimes \boldsymbol{I} \right) = 0$$

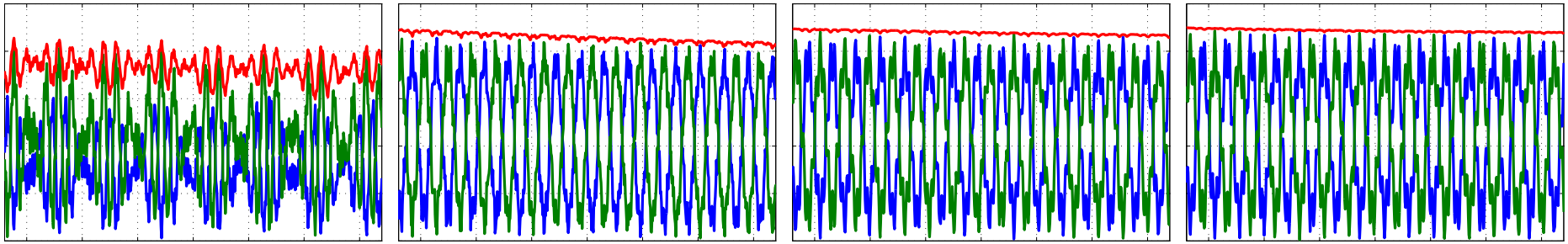
  - Conservative formulation, borrow shock capturing from CFD
  - Expensive, needs curl-free spaces for $\boldsymbol{F}$

Oscillating Beam, Energy Conservation

- Energy conservation not explicitly enforced by scheme
- Therefore, it is a good indicator of accuracy



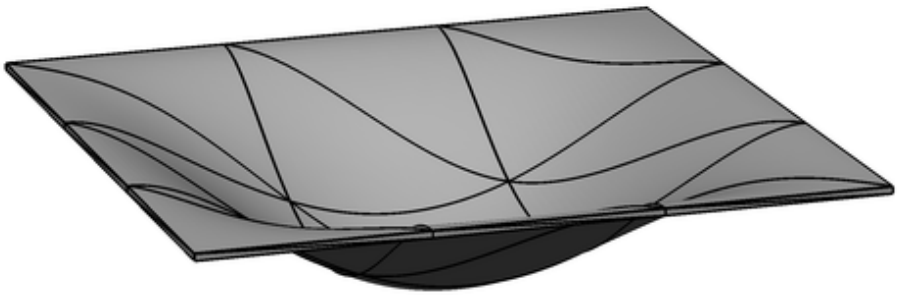
Kinetic, potential, and total energy

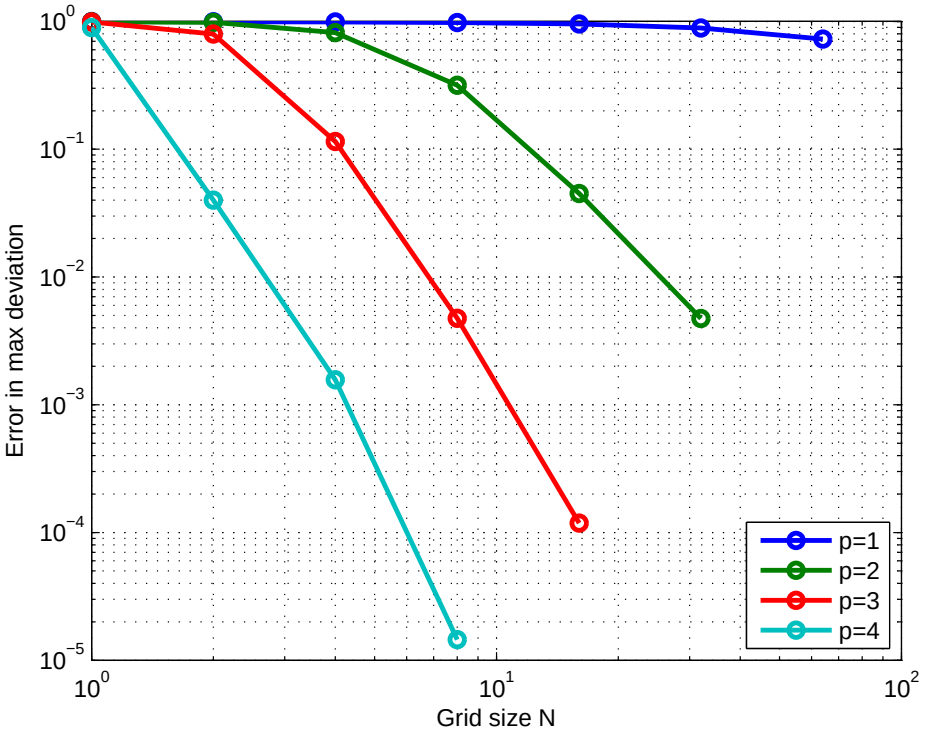


$p = 1$ $p = 2$ $p = 3$ $p = 4$

Volumetric Modeling of Thin Structures

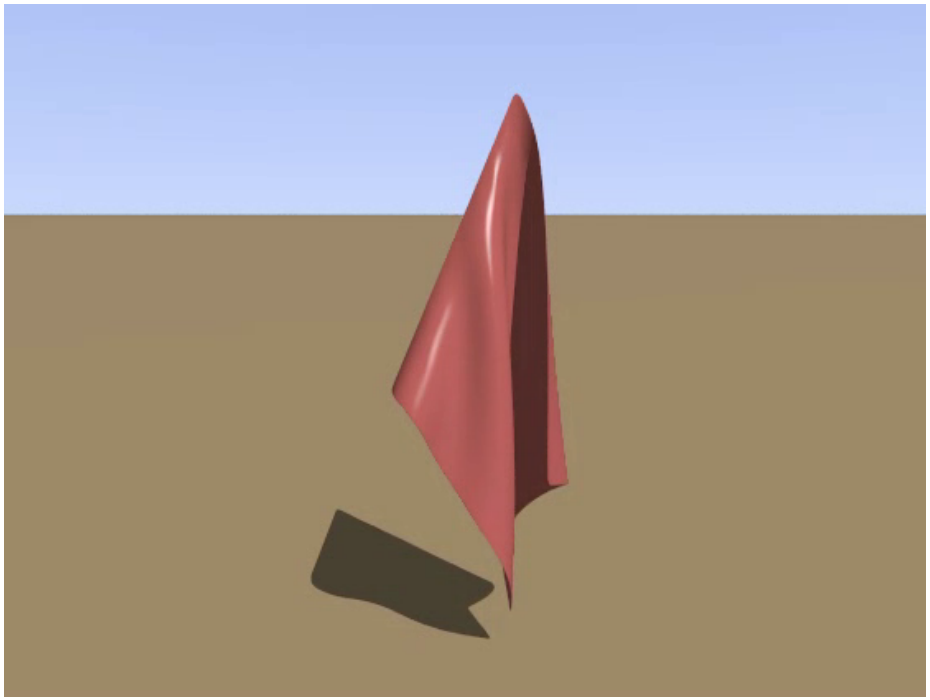
- The high order discretizations allow for highly stretched elements
- Model plate problem, single element across thickness
- Discontinuous basis functions less sensitive to locking



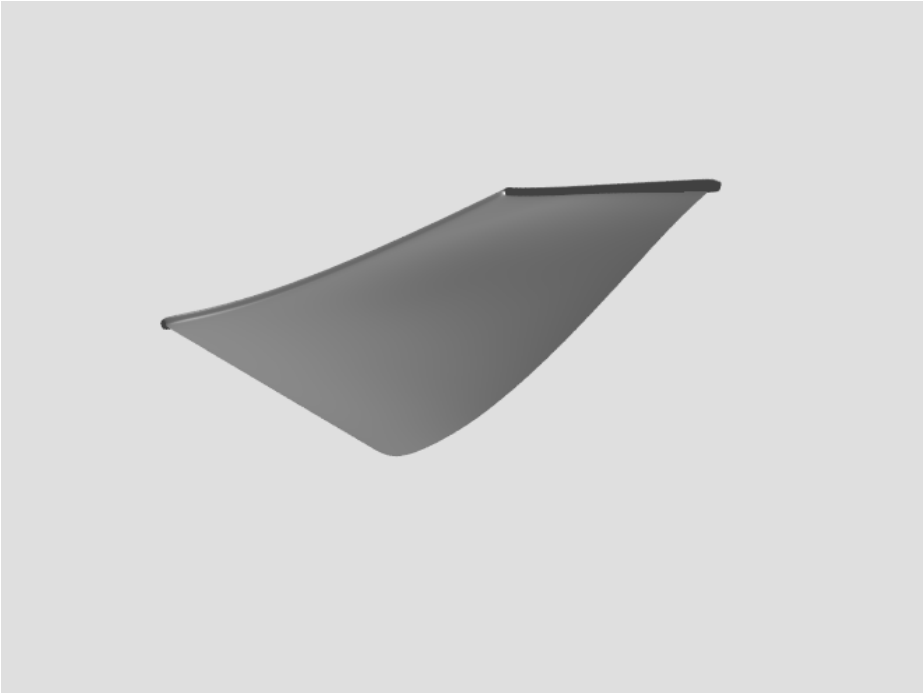


Volumetric Modeling of Thin Structures

- Automatic treatment of coupling between beams/membranes
  - No special models required
  - Only a question of generating the stretched meshes



Thin elastic membrane



Coupling of thin beams/membrane

Conclusions

- Important steps toward practical DG solver for realistic problems:
  - Efficient viscous discretization (the CDG method)
  - General purpose multigrid/block ILU preconditioner
  - Robustness by sensors and artificial viscosity, for RANS and shocks
  - Optimal accuracy for deformable domains by mapping approach
  - High order Lagrangian DG formulation for solid dynamics
- Current work: New sparser discretizations, 3-D curved mesh generation, orderings for parallel ILU, algebraic/geometric multigrid for coarse scale problem, extensions for LES/DES, coupling of DG fluid/structure formulations, applications in flapping flight, aeroacoustics, and transonic/supersonic flows