High-Order Discontinuous Galerkin Methods for Fluid and Solid Mechanics

Per-Olof Persson

Department of Mathematics, University of California, Berkeley
Mathematics Department, Lawrence Berkeley National Laboratory

with M. Zahr, W. Pazner, B. Froehle, A. Shi, L. Wang, M. Franco, M. Fortunato

Computing Sciences Summer Program 2023
Berkeley Lab

June 22, 2023

Outline

1. Introduction and Motivation

2. Numerical Schemes – Discretization and Solvers
   - The Discontinuous Galerkin Method
   - Curved Mesh Generation
   - Time-Stepping and Parallel Implicit Solvers

3. Methods for Deforming Domains
   - High-Order ALE Formulation
   - Time-Dependent PDE-Constrained Optimization
   - DistMesh and Moving Meshes
   - Optimization-Based High-Order Shock Tracking

Outline

1. Introduction and Motivation

2. Numerical Schemes – Discretization and Solvers
   - The Discontinuous Galerkin Method
   - Curved Mesh Generation
   - Time-Stepping and Parallel Implicit Solvers

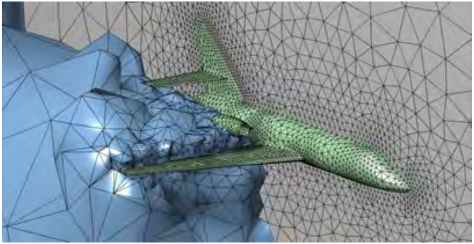
3. Methods for Deforming Domains
   - High-Order ALE Formulation
   - Time-Dependent PDE-Constrained Optimization
   - DistMesh and Moving Meshes
   - Optimization-Based High-Order Shock Tracking

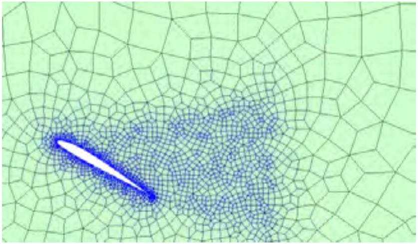
Motivation

- Need for higher fidelity predictions in computational mechanics
  - Turbulent flows, wave propagation, multiscale phenomena, non-linear interactions
- Many practical applications involve time-varying geometries
  - Fluid/structure interaction, flapping flight, wind turbines, rotor-stator flows
- Goal: Develop *robust*, *efficient*, and *accurate* high-order methods based on fully unstructured meshes

Why Unstructured Meshes?

- Complex *geometries* need flexible element topologies
- Complex *solution fields* need spatially variable resolution
- Fully automated mesh generators for CAD geometries are based on unstructured simplex elements
- Real-world simulation software dominated by unstructured mesh discretization schemes

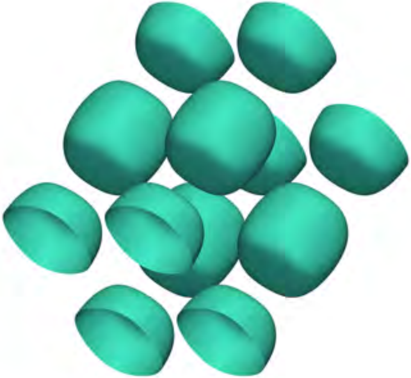




Why high-order accurate methods?

- DNS / LES of Compressible Taylor-Green Vortex
- Challenge problem from 3rd High-Order Workshop
- Transitional flow, turbulent decay

$t = 0$

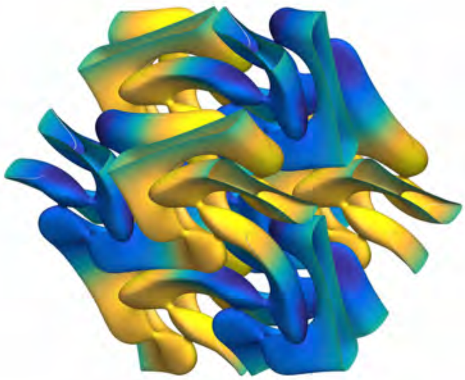


Vorticity magnitude contours, colored by helicity

Why high-order accurate methods?

- DNS / LES of Compressible Taylor-Green Vortex
- Challenge problem from 3rd High-Order Workshop
- Transitional flow, turbulent decay

$t = 2$

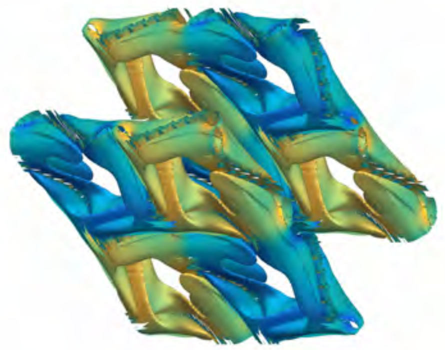


Vorticity magnitude contours, colored by helicity

Why high-order accurate methods?

- DNS / LES of Compressible Taylor-Green Vortex
- Challenge problem from 3rd High-Order Workshop
- Transitional flow, turbulent decay

$t = 4$

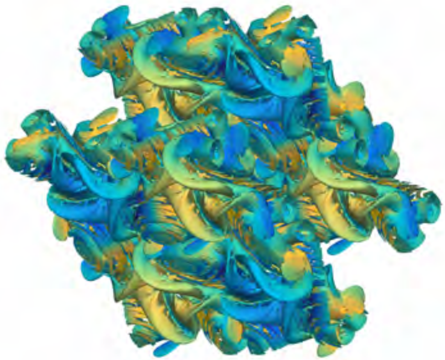


Vorticity magnitude contours, colored by helicity

Why high-order accurate methods?

- DNS / LES of Compressible Taylor-Green Vortex
- Challenge problem from 3rd High-Order Workshop
- Transitional flow, turbulent decay

$t = 6$

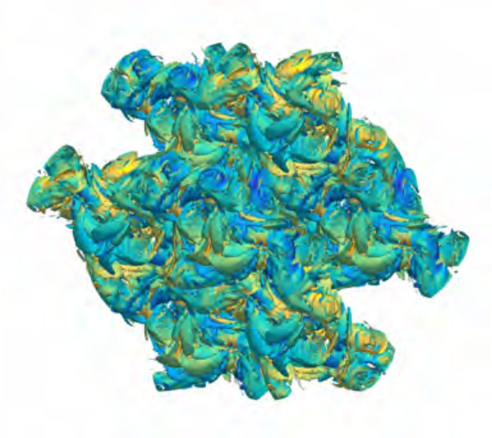


Vorticity magnitude contours, colored by helicity

Why high-order accurate methods?

- DNS / LES of Compressible Taylor-Green Vortex
- Challenge problem from 3rd High-Order Workshop
- Transitional flow, turbulent decay

$t = 8$

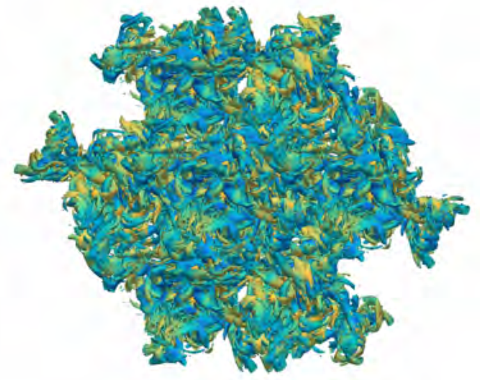


Vorticity magnitude contours, colored by helicity

Why high-order accurate methods?

- DNS / LES of Compressible Taylor-Green Vortex
- Challenge problem from 3rd High-Order Workshop
- Transitional flow, turbulent decay

$t = 10$

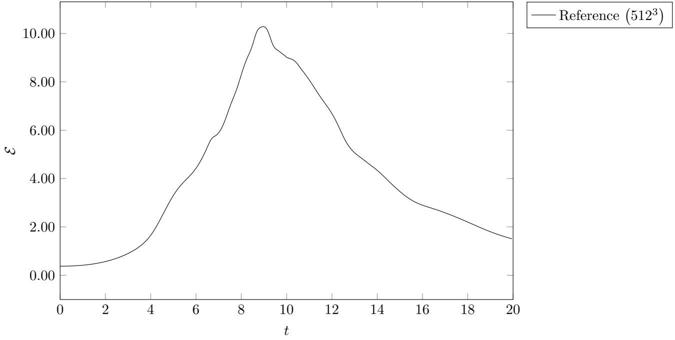


Vorticity magnitude contours, colored by helicity

Motivation: High degree p

- Spectral accuracy as $p \to \infty$
- Accurate wave propagation
- Low numerical dissipation, long time integration

Example: Compressible Taylor-Green Vortex



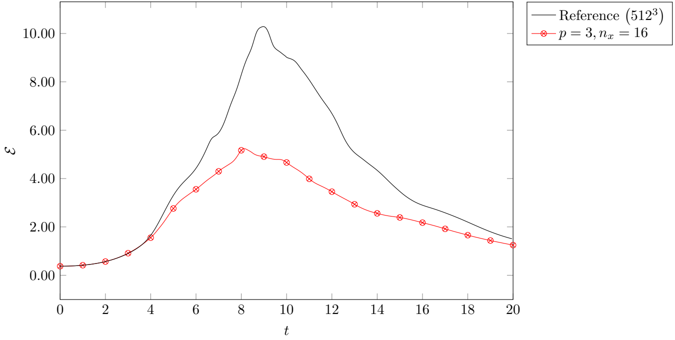
DOFs: 512^3

$$\mathcal{E}(t) = \frac{1}{\rho_0|\Omega|} \int_\Omega \rho \frac{\omega \cdot \omega}{2} \, dx$$

Motivation: High degree p

- Spectral accuracy as $p \to \infty$
- Accurate wave propagation
- Low numerical dissipation, long time integration

Example: Compressible Taylor-Green Vortex



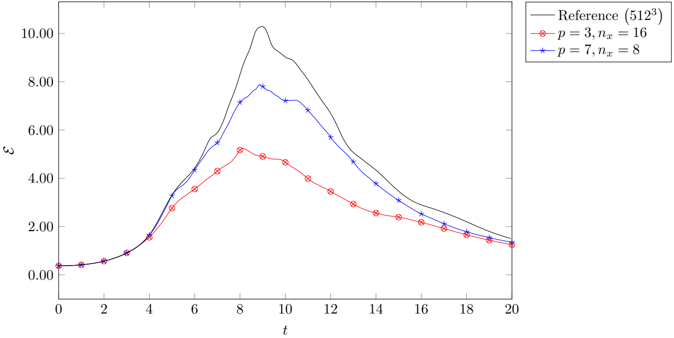
DOFs: 64^3

$$\mathcal{E}(t) = \frac{1}{\rho_0 |\Omega|} \int_\Omega \rho \frac{\omega \cdot \omega}{2} \, dx$$

Motivation: High degree p

- Spectral accuracy as $p \to \infty$
- Accurate wave propagation
- Low numerical dissipation, long time integration

Example: Compressible Taylor-Green Vortex



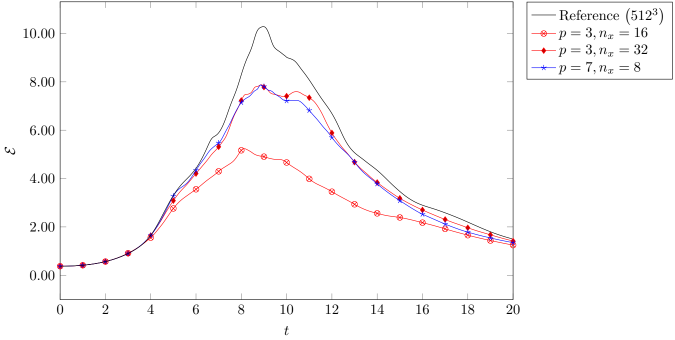
DOFs: 64^3

$$\mathcal{E}(t) = \frac{1}{\rho_0 |\Omega|} \int_\Omega \rho \frac{\omega \cdot \omega}{2} \, dx$$

Motivation: High degree p

- Spectral accuracy as $p \to \infty$
- Accurate wave propagation
- Low numerical dissipation, long time integration

Example: Compressible Taylor-Green Vortex



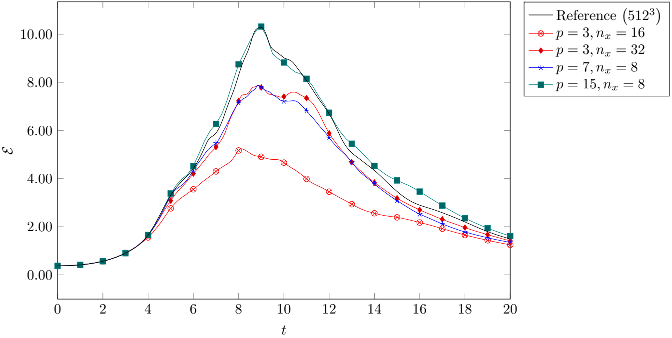
DOFs: 128^3

$$\mathcal{E}(t) = \frac{1}{\rho_0 |\Omega|} \int_\Omega \rho \frac{\omega \cdot \omega}{2} \, dx$$

Motivation: High degree p

- Spectral accuracy as $p \to \infty$
- Accurate wave propagation
- Low numerical dissipation, long time integration

Example: Compressible Taylor-Green Vortex

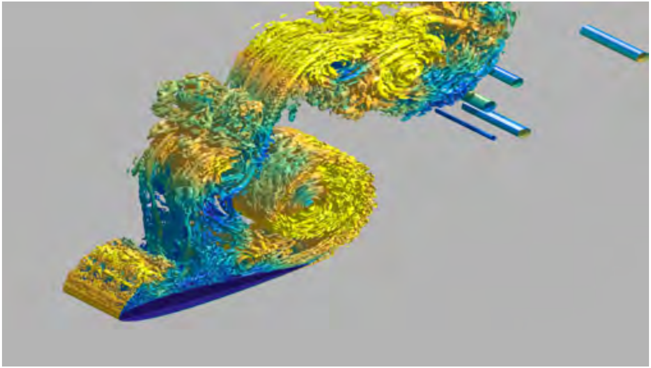


DOFs: 128^3

$$\mathcal{E}(t) = \frac{1}{\rho_0 |\Omega|} \int_\Omega \rho \frac{\omega \cdot \omega}{2} \, dx$$

Real-world applications: Large Eddy Simulation

- Turbulent flow problems are inherenty difficult due to interactions between small and large scales
- Large Eddy Simulation (LES) and high-order methods are widely belived to be part of the future state-of-the-art simulation tools



Outline

1. Introduction and Motivation
2. Numerical Schemes – Discretization and Solvers
   - The Discontinuous Galerkin Method
   - Curved Mesh Generation
   - Time-Stepping and Parallel Implicit Solvers
3. Methods for Deforming Domains
   - High-Order ALE Formulation
   - Time-Dependent PDE-Constrained Optimization
   - DistMesh and Moving Meshes
   - Optimization-Based High-Order Shock Tracking

Outline

1. Introduction and Motivation

2. Numerical Schemes – Discretization and Solvers
   - The Discontinuous Galerkin Method
   - Curved Mesh Generation
   - Time-Stepping and Parallel Implicit Solvers

3. Methods for Deforming Domains
   - High-Order ALE Formulation
   - Time-Dependent PDE-Constrained Optimization
   - DistMesh and Moving Meshes
   - Optimization-Based High-Order Shock Tracking

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:

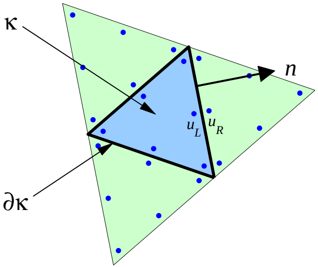
$$\int_\kappa [(\boldsymbol{u}_h)_t + \nabla \cdot \mathcal{F}_i(\boldsymbol{u}_h)]\, \boldsymbol{v}_h \, d\boldsymbol{x} = 0$$

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



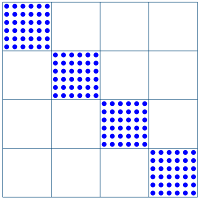
The DG Method – Observations

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

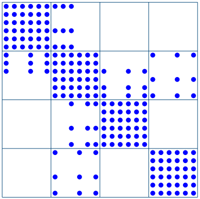
$$(\boldsymbol{u}_h)_t A_\kappa + \int_{\partial\kappa} \hat{\mathcal{F}}_i(\boldsymbol{u}_h^+, \boldsymbol{u}_h^-, \hat{\boldsymbol{n}})\, ds = 0$$

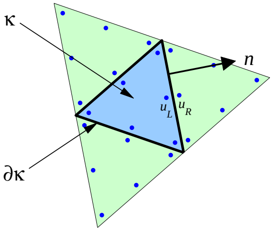
- Boundary conditions enforced naturally for any degree p
- Block-diagonal mass matrix (no overlap between basis functions)
- Block-wise compact stencil – neighboring elements connected

Mass Matrix



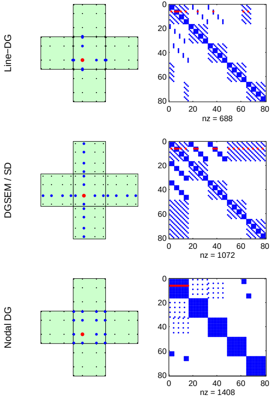
Jacobian





Sparse discretizations, the Line-DG method

- Most high-order CFD performed with $p \leq 3$
- Drastically improved sparsity required for higher p
- The Line-DG method (Persson 2012) achieves an optimal sparsity pattern without under-integration
- This directly speeds up explicit schemes and matrix-vector computations



- **The main issue is how to precondition** – standard preconditioners such as element-based Jacobi / ILU destroy sparsity, but they are required for performance
- ***Goal: Develop efficient sparsity preserving approximate block preconditioners***

A Face Upwinded Spectral Element Method (FUSE) for Conservation Laws

Scientific Achievement

A new stabilization scheme for high-order continuous Spectral Element Methods which is provable convergent up to any order.

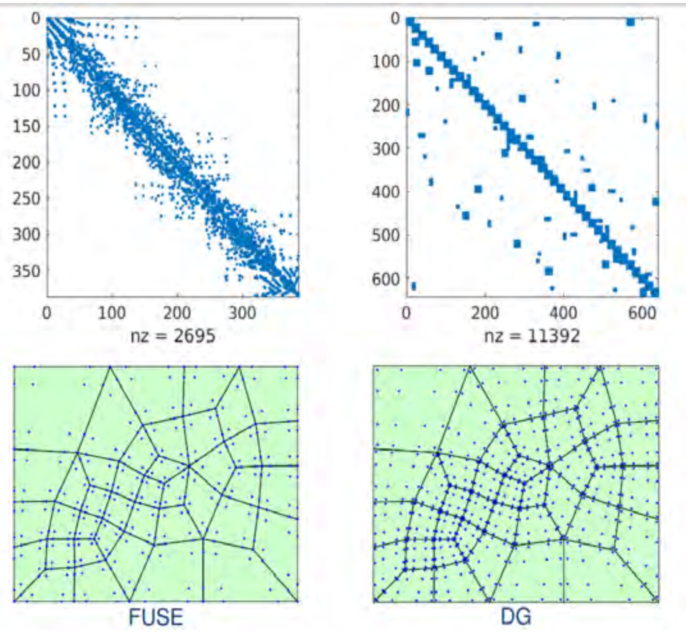
Significance and Impact

The work has the potential to drastically improve the performance of high-order methods, which are widely believed to be required for accurate predictions of turbulent flows and problems with waves and non-linear interactions.

Technical Approach

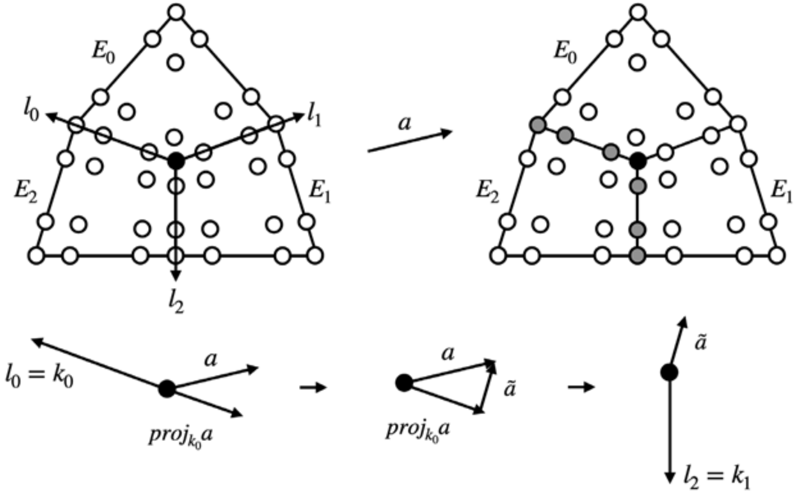
- Most stabilized schemes for fluids and other conservation laws are based on discontinuous formulations (e.g., the discontinuous Galerkin method)
- A remarkably simple way to stabilize continuous methods: Inspired by finite difference methods, choose the full upwind stencil only for face nodes
- Provably high-order convergent for a non-standard node distribution
- In addition, a line-based sparsity patterns bring the Jacobian cost from $\mathcal{O}(p^D)$ to $\mathcal{O}(pD)$, for polynomial degree p in D dimensions

PI(s)/Facility Lead(s): Per-Olof Persson, LBNL Math Group
ASCR Program: Base Math
ASCR PM: Steven Lee
Publication(s) for this work:
Y. Pan, P.-O. Persson, "A Face-Upwinded Spectral Element Method on Unstructured Quadrilateral Meshes," *Journal of Computational Physics* (in review)
Y. Pan, P.-O. Persson, "A Stabilized Face-Upwinded High-Order Method for Incompressible Flows," *Proc. of 2023 AIAA AVIATION,* June 2023.



Sparsity pattern for FUSE vs DG, at polynomial degree 3. Due to continuous fields and line-based sparsity patterns, the Jacobian matrices are more than 4 times cheaper. This effect increases in 3D and for higher degrees.

Face-Upwinding for unstructured grids. For general non-linear roblems, the upwinding directions depend on the solution. Also note the line-based sparsity pattern.

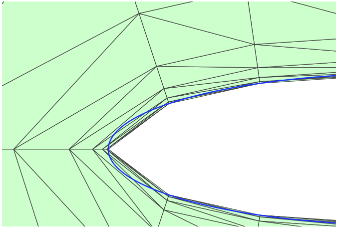


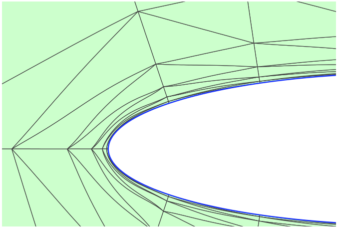
Outline

1. Introduction and Motivation
2. Numerical Schemes – Discretization and Solvers
   - The Discontinuous Galerkin Method
   - Curved Mesh Generation
   - Time-Stepping and Parallel Implicit Solvers
3. Methods for Deforming Domains
   - High-Order ALE Formulation
   - Time-Dependent PDE-Constrained Optimization
   - DistMesh and Moving Meshes
   - Optimization-Based High-Order Shock Tracking

Curved Mesh Generation

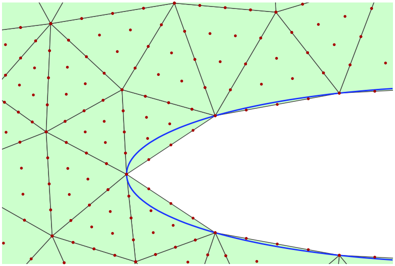
- Automatic generation of non-inverted curved elements largely an unresolved problem
- In general this is a global problem, affecting many elements except for simple isotropic 2-D meshes
- In [Persson/Peraire AIAA 2009], we proposed a *non-linear solid mechanics* approach, where the mesh is considered an elastic deformable solid
- In [Fortunato/Persson JCP 2016], we developed a high-order unstructured formulation for the classical *Winslow equations*



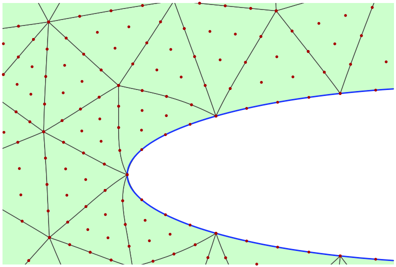


Curved Mesh Generation using Solid Mechanics

- The initial, straight-sided mesh corresponds to undeformed solid
- External forces come from the true boundary data
- Solving for a force equilibrium gives the deformed, curved, boundary conforming mesh
- Bottom-up approach can be used to obtain the boundary data



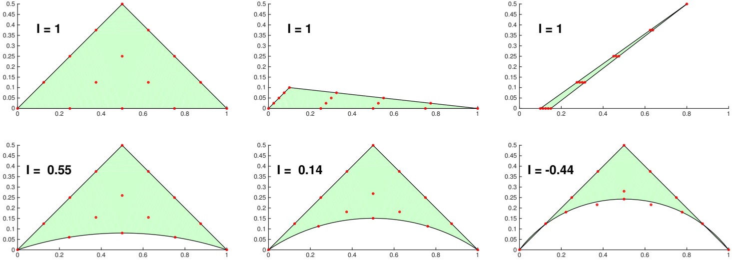
Reference domain, initial configuration

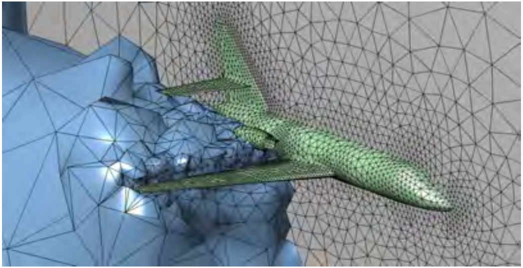


Equilibrium solution, final curved mesh

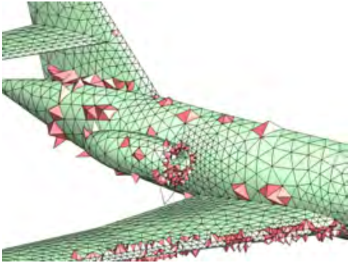
Tetrahedral Mesh of Falcon Aircraft

- Measure element distortion using *scaled Jacobians*





Tetrahedral mesh of Falcon Aircraft



Elements with $I < 0.5$

Outline

1 Introduction and Motivation

2 Numerical Schemes – Discretization and Solvers
- The Discontinuous Galerkin Method
- Curved Mesh Generation
- Time-Stepping and Parallel Implicit Solvers

3 Methods for Deforming Domains
- High-Order ALE Formulation
- Time-Dependent PDE-Constrained Optimization
- DistMesh and Moving Meshes
- Optimization-Based High-Order Shock Tracking

Temporal Discretization: DIRK

- Diagonally Implicit RK (DIRK) are implicit Runge-Kutta schemes defined by lower triangular Butcher tableau $\rightarrow$ **decoupled implicit stages**
- Overcomes issues with high-order BDF and IRK
  - Limited accuracy of A-stable BDF schemes (2nd order)
  - High cost of general implicit RK schemes (coupled stages)

$$\boldsymbol{u}^{(0)} = \boldsymbol{u}_0(\boldsymbol{\mu})$$

$$\boldsymbol{u}^{(n)} = \boldsymbol{u}^{(n-1)} + \sum_{i=1}^{s} b_i \boldsymbol{k}_i^{(n)}$$

$$\boldsymbol{u}_i^{(n)} = \boldsymbol{u}^{(n-1)} + \sum_{j=1}^{i} a_{ij} \boldsymbol{k}_j^{(n)}$$

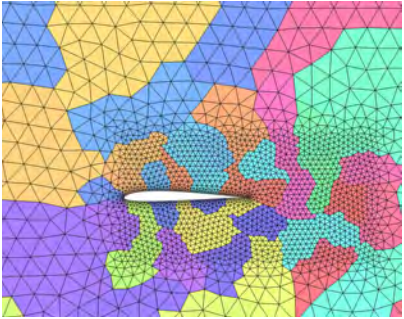
$$\mathbb{M}\boldsymbol{k}_i^{(n)} = \Delta t_n \boldsymbol{r}\left(\boldsymbol{u}_i^{(n)},\ \boldsymbol{\mu},\ t_{n-1} + c_i \Delta t_n\right)$$

$$\begin{array}{c|cccc}
c_1 & a_{11} & & & \\
c_2 & a_{21} & a_{22} & & \\
\vdots & \vdots & \vdots & \ddots & \\
c_s & a_{s1} & a_{s2} & \cdots & a_{ss} \\
\hline
 & b_1 & b_2 & \cdots & b_s
\end{array}$$

Butcher Tableau for DIRK scheme

Preconditioning for Newton-Krylov Solvers

- Implicit solvers typically required because of CFL restrictions from viscous effects, low Mach numbers, and adaptive/anisotropic grids
- Jacobian matrices are large even at $p = 2$ or $p = 3$, however:
  - They are required for non-trivial preconditioners
  - They are very expensive to recompute
- Block-ILU(0) preconditioners and Minimum Discarded Fill (MDF) element ordering [Persson/Peraire 2008]
- Distributed parallel solvers developed in [Persson '09]
- IMEX schemes for geometrically induced stiffness (e.g. boundary layers) [Persson 2011]

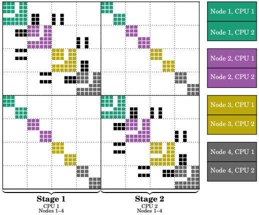


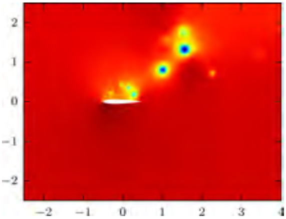
Stage-Parallel Implicit Runge-Kutta Methods

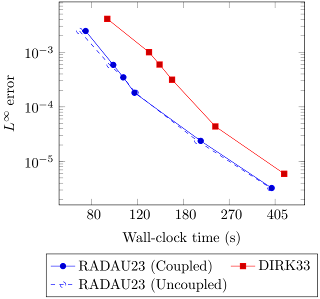
- s-stage Implicit Runge-Kutta (IRK) Method for $\boldsymbol{M}\frac{\partial \boldsymbol{u}}{\partial t} = \boldsymbol{f}(\boldsymbol{u})$:

$$\boldsymbol{M}\boldsymbol{k}_i = \boldsymbol{f}\left(t_0 + \Delta t c_i, \boldsymbol{u}_0 + \Delta t \sum_{j=1}^{s} a_{ij}\boldsymbol{k}_j\right), \quad \boldsymbol{u}_1 = \boldsymbol{u}_0 + \Delta t \sum_{i=1}^{s} b_i \boldsymbol{k}_i$$

- Solve for stage solutions $\boldsymbol{W} = (A \otimes I_n)\boldsymbol{K}$ for increased sparsity, precondition by stage-uncoupled shifted block ILU(0)
- Perfect stage parallelization, high communication on shared memory







W. Pazner and P.-O. Persson, *Stage-parallel fully implicit Runge-Kutta solvers for discontinuous Galerkin fluid simulations*. J. Comp. Phys., Vol. 335, pp. 700-717, April 2017.

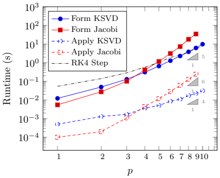
Approximate tensor-product preconditioners

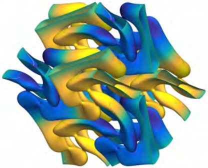
- Develop implicit DG methods with *linear complexity* in the polynomial degree p *per DOF*, that is, $\mathcal{O}(p^{d+1})$
- For element-wise inverses, find best approximation with KSVDs:

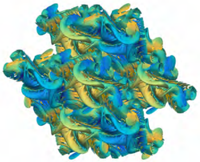
$$\text{2D:} \quad P = A_1 \otimes B_1 + A_2 \otimes B_2$$

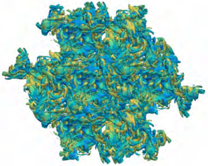
$$\text{3D:} \quad P = A_1 \otimes B_1 \otimes C_1 + A_1 \otimes B_2 \otimes C_2$$

- Use Schur factorization (or matrix diagonalization) for inversion
- Asymptotically superior, lower run-time already at $p > 3$









W. Pazner and P.-O. Persson, *Approximate tensor-product preconditioners for very high order discontinuous Galerkin methods*, in review, J. Comput. Phys.

W. Pazner and P.-O. Persson, *High-Order DNS and LES Simulations Using an Implicit Tensor-Product Discontinuous Galerkin Method*. Proc. of the 23rd AIAA Computational Fluid Dynamics Conference, June 2017. AIAA-2017-3948
(Winner, AIAA CFD 2017 Student Paper Competition)

Outline

1. Introduction and Motivation
2. Numerical Schemes – Discretization and Solvers
   - The Discontinuous Galerkin Method
   - Curved Mesh Generation
   - Time-Stepping and Parallel Implicit Solvers
3. Methods for Deforming Domains
   - High-Order ALE Formulation
   - Time-Dependent PDE-Constrained Optimization
   - DistMesh and Moving Meshes
   - Optimization-Based High-Order Shock Tracking

Outline

1. Introduction and Motivation

2. Numerical Schemes – Discretization and Solvers
   - The Discontinuous Galerkin Method
   - Curved Mesh Generation
   - Time-Stepping and Parallel Implicit Solvers

3. Methods for Deforming Domains
   - High-Order ALE Formulation
   - Time-Dependent PDE-Constrained Optimization
   - DistMesh and Moving Meshes
   - Optimization-Based High-Order Shock Tracking

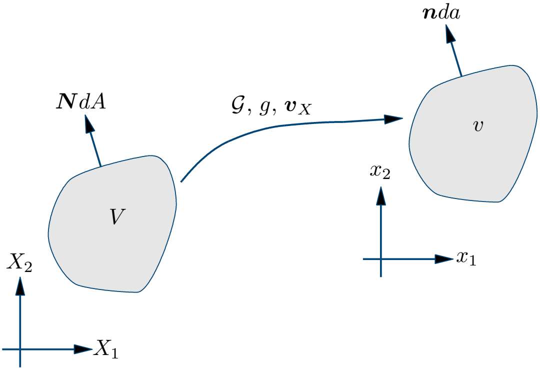
ALE Formulation for Deforming Domains

- Use mapping-based ALE formulation for moving domains [Visbal/Gaitonde '02], [Persson/Bonet/Peraire '09]
- Map from reference domain V to physical deformable domain $v(t)$
- Introduce the *mapping deformation gradient* $\boldsymbol{G}$ and the *mapping velocity* $\boldsymbol{v}_X$ as

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

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

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



- Transform equations to account for the motion

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

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

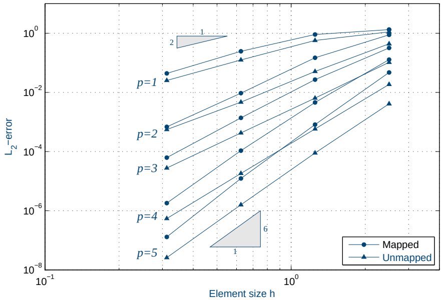
  and

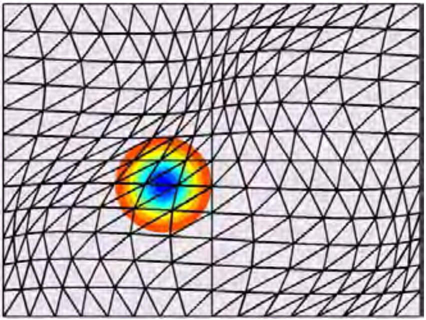
$$\boldsymbol{\nabla}_x \boldsymbol{U}_x = \boldsymbol{\nabla}_X(g^{-1}\boldsymbol{U}_X)\boldsymbol{G}^{-T} = (g^{-1}\boldsymbol{\nabla}_X \boldsymbol{U}_X - \boldsymbol{U}_X \boldsymbol{\nabla}_X(g^{-1}))\boldsymbol{G}^{-T}$$

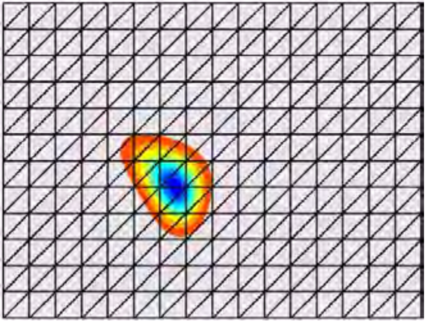
- Details in [Persson/Bonet/Peraire ’09], including how to satisfy the so-called Geometric Conservation Law (GCL)

ALE Formulation for Deforming Domains

- Mapping-based formulation gives arbitrarily high-order accuracy in space and time







Vertical Axis Wind Turbines

- Recent interest in vertical axis wind turbines (VAWT):
  - 2D airfoils, easy to manufacture, supportable at both ends
  - Omnidirectional (good in gusty, low wind, e.g. close to ground)
  - Lower blade speeds – lower noise and impact
  - Can be packed close together in wind farms
- Numerical simulations can help overcome remaining challenges:
  - Lower theoretical (and practical) efficiency than HAWTs
  - Sensitive to design conditions
  - Structural problems, fatigue and catastrophic failure

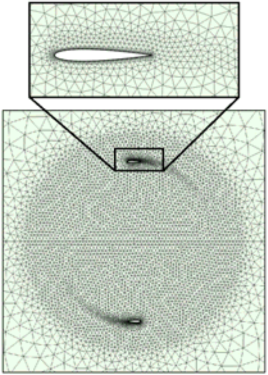
Windterra ECO 1200 1Kw VAWT

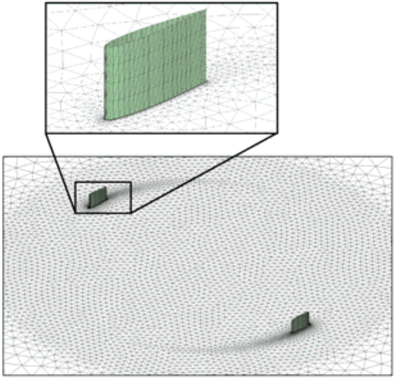
VAWT – Mathematical Model and Discretization

- Solve the Navier-Stokes equations in a rotating frame:

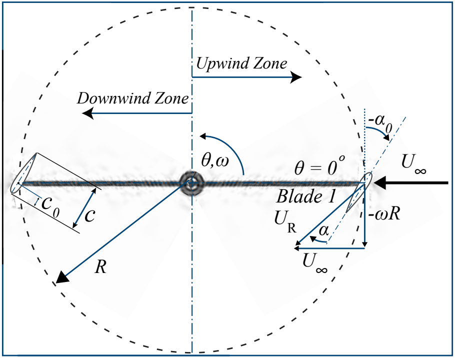
$$\mathcal{G}(X, Y, t) = \begin{bmatrix} \cos \omega t & -\sin \omega t \\ \sin \omega t & \cos \omega t \end{bmatrix} \begin{bmatrix} X \\ Y \end{bmatrix}$$

- Hybrid boundary layer/unstructured mesh, element degree $p = 3$

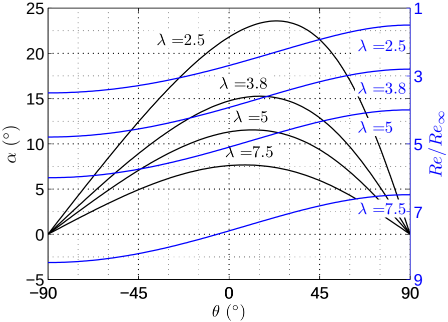




Range of angle-of-attack (α) based on TSR (λ)



Bird's-eye-view of 2-bladed VAWT

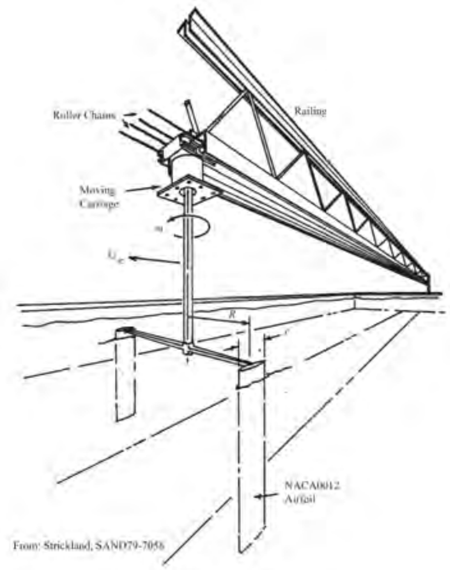


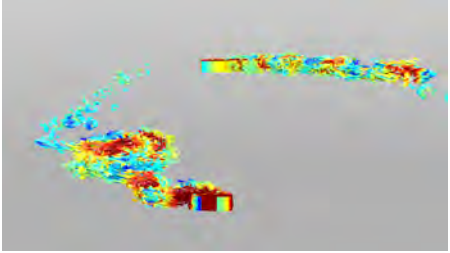
Variation of angle-of-attack at various TSR as a function of azimuthal angle

$$\lambda = \frac{\omega R}{U_\infty}$$

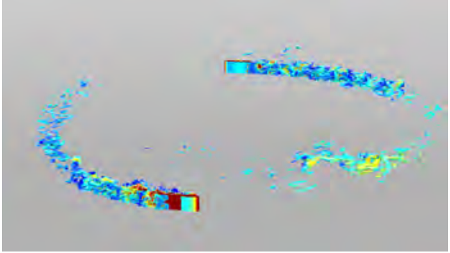
Sandia National Laboratories’ Low Re VAWT

Sandia National Labs tow tank experiment from 1979





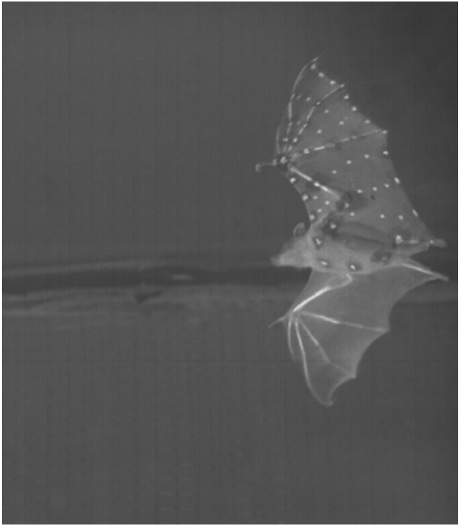
TSR = 2.5



TSR = 5.0

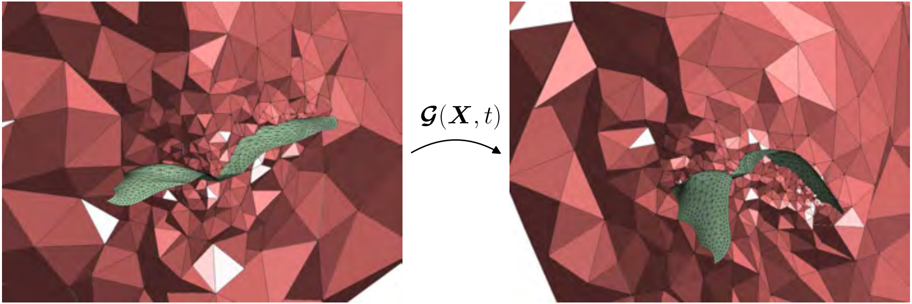
Bio-Inspiration for Flapping Wing MAVs

- Develop high-order accurate simulation capabilities that capture the complex physics in flapping flight
- Use the computational tools for increased understanding and to design optimized flapping kinematics



Domain Mapping

- Highly complex wing motion from measured data
- Construct mapping $\mathcal{G}(\boldsymbol{X}, t)$ *numerically* by nonlinear solid mechanics approach [Persson '09]
- A reference mesh (left) is deformed elastically to smoothly align with the prescribed wing motion (right)
- Grid velocity $\boldsymbol{v}_X = \frac{\partial \mathcal{G}}{\partial t}\Big|_X$ defined consistently with DIRK scheme



Outline

1. Introduction and Motivation

2. Numerical Schemes – Discretization and Solvers
   - The Discontinuous Galerkin Method
   - Curved Mesh Generation
   - Time-Stepping and Parallel Implicit Solvers

3. Methods for Deforming Domains
   - High-Order ALE Formulation
   - Time-Dependent PDE-Constrained Optimization
   - DistMesh and Moving Meshes
   - Optimization-Based High-Order Shock Tracking

Discretization of PDE-Constrained Optimization

- *Continuous* PDE-constrained optimization problem

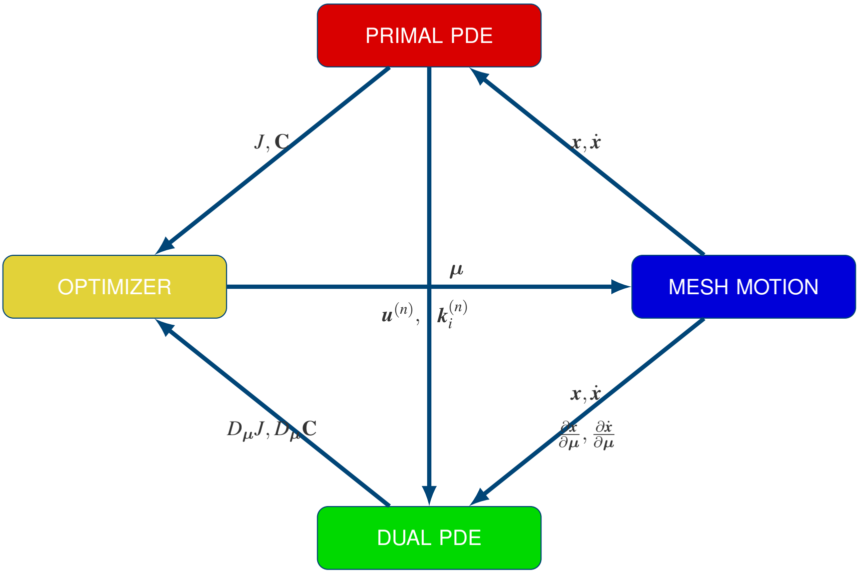
$$\begin{aligned} \underset{\boldsymbol{U},\ \boldsymbol{\mu}}{\text{minimize}} \quad & \mathcal{J}(\boldsymbol{U}, \boldsymbol{\mu}) \\ \text{subject to} \quad & \boldsymbol{C}(\boldsymbol{U}, \boldsymbol{\mu}) \leq 0 \\ & \frac{\partial \boldsymbol{U}}{\partial t} + \nabla \cdot \boldsymbol{F}(\boldsymbol{U}, \nabla \boldsymbol{U}) = 0 \ \text{ in } \ v(\boldsymbol{\mu}, t) \end{aligned}$$

- *Fully discrete* PDE-constrained optimization problem

$$\begin{aligned} \underset{\substack{\boldsymbol{u}^{(0)},\ \dots,\ \boldsymbol{u}^{(N_t)} \in \mathbb{R}^{N_{\boldsymbol{u}}}, \\ \boldsymbol{k}_1^{(1)},\ \dots,\ \boldsymbol{k}_s^{(N_t)} \in \mathbb{R}^{N_{\boldsymbol{u}}}, \\ \boldsymbol{\mu} \in \mathbb{R}^{n_{\boldsymbol{\mu}}}}}{\text{minimize}} \quad & J(\boldsymbol{u}^{(0)},\ \dots,\ \boldsymbol{u}^{(N_t)},\ \boldsymbol{k}_1^{(1)},\ \dots,\ \boldsymbol{k}_s^{(N_t)},\ \boldsymbol{\mu}) \\ \text{subject to} \quad & \mathbf{C}(\boldsymbol{u}^{(0)},\ \dots,\ \boldsymbol{u}^{(N_t)},\ \boldsymbol{k}_1^{(1)},\ \dots,\ \boldsymbol{k}_s^{(N_t)},\ \boldsymbol{\mu}) \leq 0 \\ & \boldsymbol{u}^{(0)} - \boldsymbol{u}_0(\boldsymbol{\mu}) = 0 \\ & \boldsymbol{u}^{(n)} - \boldsymbol{u}^{(n-1)} + \sum_{i=1}^{s} b_i \boldsymbol{k}_i^{(n)} = 0 \\ & \mathbb{M}\boldsymbol{k}_i^{(n)} - \Delta t_n \boldsymbol{r}\left(\boldsymbol{u}_i^{(n)},\ \boldsymbol{\mu},\ t_i^{(n-1)}\right) = 0 \end{aligned}$$

Generalized Reduced-Gradient Approach

Optimizer drives, Primal returns QoI values, Dual returns QoI gradients



Adjoint Method to Compute QoI Gradients

- Consider the *fully discrete* output functional $F(\boldsymbol{u}^{(n)}, \boldsymbol{k}_i^{(n)}, \boldsymbol{\mu})$
  - Represents either the **objective** function or a **constraint**
- The *total derivative* with respect to the parameters $\boldsymbol{\mu}$, required in the context of gradient-based optimization, takes the form

$$D_{\boldsymbol{\mu}} F = \frac{\partial F}{\partial \boldsymbol{\mu}} + \sum_{n=0}^{N_t} \frac{\partial F}{\partial \boldsymbol{u}^{(n)}} \frac{\partial \boldsymbol{u}^{(n)}}{\partial \boldsymbol{\mu}} + \sum_{n=1}^{N_t} \sum_{i=1}^{s} \frac{\partial F}{\partial \boldsymbol{k}_i^{(n)}} \frac{\partial \boldsymbol{k}_i^{(n)}}{\partial \boldsymbol{\mu}}$$

- The sensitivities, $\frac{\partial \boldsymbol{u}^{(n)}}{\partial \boldsymbol{\mu}}$ and $\frac{\partial \boldsymbol{k}_i^{(n)}}{\partial \boldsymbol{\mu}}$, are expensive to compute, requiring the solution of $n_{\boldsymbol{\mu}}$ linear evolution equations
- **Adjoint method**: alternative method for computing $D_{\boldsymbol{\mu}} F$ that require one linear evolution evoluation equation for each quantity of interest, F

Fully Discrete Adjoint Equations: Dissection

- **Linear** evolution equations solved **backward** in time
- **Primal** state $\boldsymbol{u}_i^{(n)}$ required at each stage of dual problem
- Heavily dependent on **chosen output**

$$\boldsymbol{\lambda}^{(N_t)} = \frac{\partial F}{\partial \boldsymbol{u}^{(N_t)}}^T$$

$$\boldsymbol{\lambda}^{(n-1)} = \boldsymbol{\lambda}^{(n)} + \frac{\partial F}{\partial \boldsymbol{u}^{(n-1)}}^T + \sum_{i=1}^{s} \Delta t_n \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}} \left(\boldsymbol{u}_i^{(n)},\ \boldsymbol{\mu},\ t_{n-1} + c_i \Delta t_n\right)^T \boldsymbol{\kappa}_i^{(n)}$$

$$\mathbb{M}^T \boldsymbol{\kappa}_i^{(n)} = \frac{\partial F}{\partial \boldsymbol{u}^{(N_t)}}^T + b_i \boldsymbol{\lambda}^{(n)} + \sum_{j=i}^{s} a_{ji} \Delta t_n \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}} \left(\boldsymbol{u}_j^{(n)},\ \boldsymbol{\mu},\ t_{n-1} + c_j \Delta t_n\right)^T \boldsymbol{\kappa}_j^{(n)}$$

- Gradient reconstruction via dual variables

$$D_{\boldsymbol{\mu}} F = \frac{\partial F}{\partial \boldsymbol{\mu}} + \boldsymbol{\lambda}^{(0)\,T} \frac{\partial \boldsymbol{u}_0}{\partial \boldsymbol{\mu}} + \sum_{n=1}^{N_t} \Delta t_n \sum_{i=1}^{s} \boldsymbol{\kappa}_i^{(n)\,T} \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\mu}} (\boldsymbol{u}_i^{(n)},\ \boldsymbol{\mu},\ t_i^{(n)})$$

Energetically Optimal Flapping, Thrust Constraint

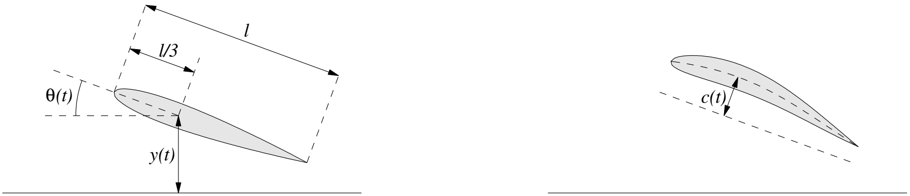
$$\underset{\boldsymbol{\mu}}{\text{minimize}} \quad -\int_{2T}^{3T}\int_{\boldsymbol{\Gamma}} \boldsymbol{f}\cdot\dot{\boldsymbol{x}}\,dS\,dt$$

$$\text{subject to} \quad \int_{2T}^{3T}\int_{\boldsymbol{\Gamma}} \boldsymbol{f}\cdot\boldsymbol{e}_1\,dS\,dt = q$$

$$\boldsymbol{U}(\boldsymbol{x},0) = \bar{\boldsymbol{U}}(\boldsymbol{x})$$

$$\frac{\partial \boldsymbol{U}}{\partial t} + \nabla\cdot\boldsymbol{F}(\boldsymbol{U},\nabla\boldsymbol{U}) = 0$$

- Isentropic, compressible, Navier-Stokes
- Re = 1000, M = 0.2
- $y(t)$, $\theta(t)$, $c(t)$ parametrized via periodic cubic splines
- Black-box optimizer: SNOPT



Airfoil schematic, kinematic description

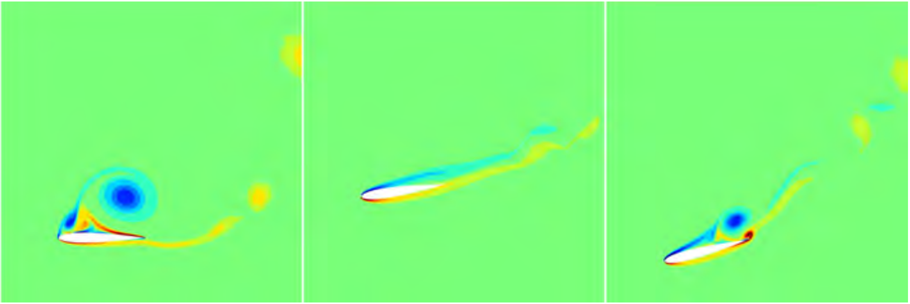
Optimal Control - Fixed Shape

Fixed Shape, Optimal Rigid Body Motion (RBM), Varied x-Impulse

Energy = 9.4096
x-impulse = -0.1766

Energy = 0.45695
x-impulse = 0.000

Energy = 4.9475
x-impulse = -2.500



Initial Guess

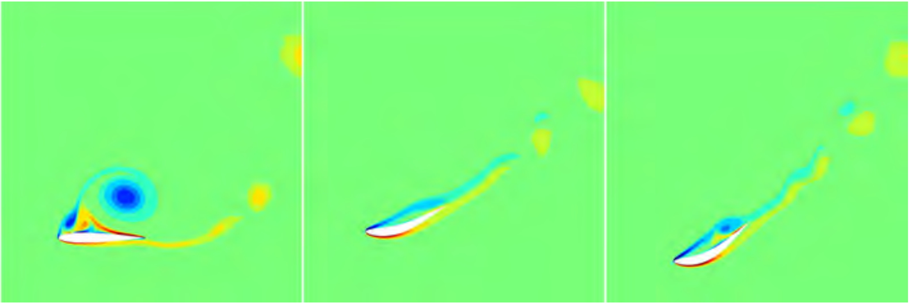
Optimal RBM
$J_x = 0.0$

Optimal RBM
$J_x = -2.5$

Optimal Control, Time-Morphed Geometry

Optimal Rigid Body Motion (RBM) and Time-Morphed Geometry (TMG), Varied x-Impulse

Energy = 9.4096	Energy = 0.45027	Energy = 4.6182
x-impulse = -0.1766	x-impulse = 0.000	x-impulse = -2.500



Initial Guess	Optimal RBM/TMG $J_x = 0.0$	Optimal RBM/TMG $J_x = -2.5$

Adjoint Method for Periodic PDE-Constraints

- Following identical procedure as for non-periodic case, the adjoint equations corresponding to the periodic conservation law are

$$\boldsymbol{\lambda}^{(N_t)} = \boldsymbol{\lambda}^{(0)} + \frac{\partial F}{\partial \boldsymbol{u}^{(N_t)}}^T$$

$$\boldsymbol{\lambda}^{(n-1)} = \boldsymbol{\lambda}^{(n)} + \frac{\partial F}{\partial \boldsymbol{u}^{(n-1)}}^T + \sum_{i=1}^{s} \Delta t_n \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}} \left(\boldsymbol{u}_i^{(n)},\ \boldsymbol{\mu},\ t_{n-1} + c_i \Delta t_n\right)^T \boldsymbol{\kappa}_i^{(n)}$$

$$\mathbb{M}^T \boldsymbol{\kappa}_i^{(n)} = \frac{\partial F}{\partial \boldsymbol{u}^{(N_t)}}^T + b_i \boldsymbol{\lambda}^{(n)} + \sum_{j=i}^{s} a_{ji} \Delta t_n \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}} \left(\boldsymbol{u}_j^{(n)},\ \boldsymbol{\mu},\ t_{n-1} + c_j \Delta t_n\right)^T \boldsymbol{\kappa}_j^{(n)}$$

- Dual problem is also periodic
- Solve *linear, periodic* problem using Krylov shooting method

Energetically optimal flapping in three-dimensions

Energy = 1.4459e-01

Thrust = -1.1192e-01

Energy = 3.1378e-01

Thrust = 0.0000e+00

Energetically optimal flapping vs. required thrust

Energy = 0.21935
Thrust = 0.0000

Optimal $T_x = 0$

Energy = 3.00404
Thrust = 1.5000

Optimal
$T_x = 1.5$

Energy = 6.2869
Thrust = 2.5000

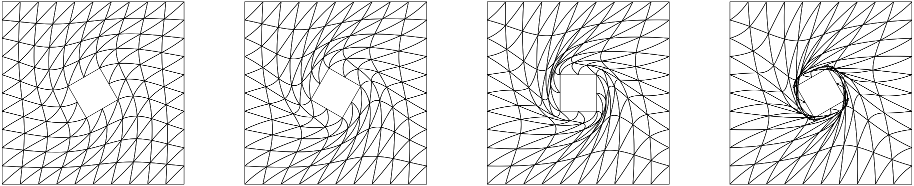
Optimal
$T_x = 2.5$

Outline

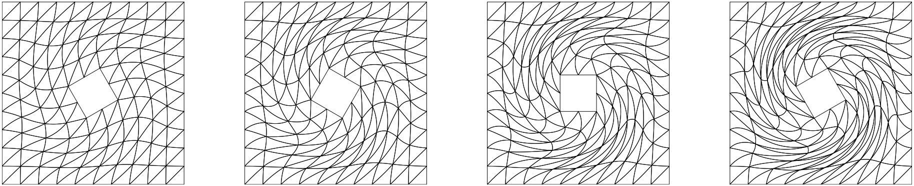
1. Introduction and Motivation
2. Numerical Schemes – Discretization and Solvers
   - The Discontinuous Galerkin Method
   - Curved Mesh Generation
   - Time-Stepping and Parallel Implicit Solvers
3. Methods for Deforming Domains
   - High-Order ALE Formulation
   - Time-Dependent PDE-Constrained Optimization
   - DistMesh and Moving Meshes
   - Optimization-Based High-Order Shock Tracking

Domains with Large Deformations

- For large deformations, it is in general not possible to deform the meshes smoothly – *remeshing required*
- For efficient numerical schemes, use *local* mesh operations



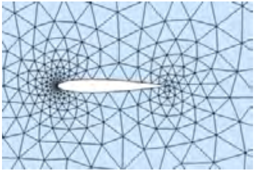
Radial basis functions



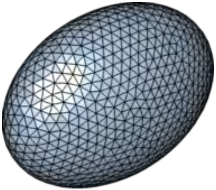
Nonlinear elasticity

The DistMesh Mesh Generator

- High quality meshes obtained using the *DistMesh* algorithm [Persson, Ph.D. thesis, ’05]
  1. Start with *any* topologically correct initial mesh
  2. Move nodes to find force equilibrium in edges
     - Project boundary nodes using *implicit geometry* $\phi(\boldsymbol{x})$
     - Update element connectivities with Delaunay



- Excellent properties:
  - Very simple (1 page of MATLAB)
  - Implicit geometries $\rightarrow$ No CAD required
  - Very high element qualities
  - Moving meshes/deforming domains



- Widely used:
  - Numerous books and courses
  - Rewritten in C, C++, C#, Fortran 77/90, Python, Mathematica, Octave

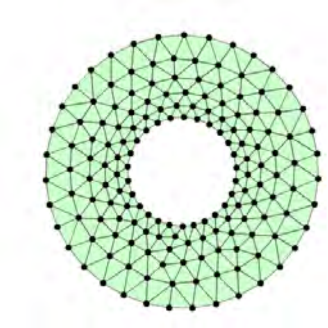
The DistMesh Mesh Generator

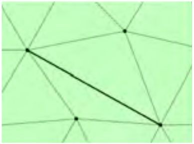
- Spring-based non-linear compressive force analogy for mesh motion

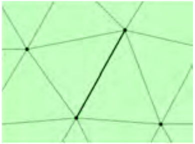
$$\boldsymbol{p}^{(n+1)} = \boldsymbol{p}^{(n)} + \delta \sum_i \boldsymbol{F}_i$$

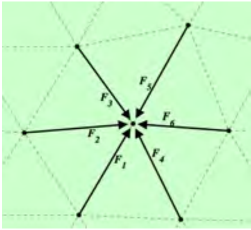
$$|\boldsymbol{F}_i(l)| = \begin{cases} k(l - l_0) & \text{if } l \geq l_0, \\ 0 & \text{if } l < l_0, \end{cases}$$

- Perform topological transformations ("edge flips") to improve element connectivities

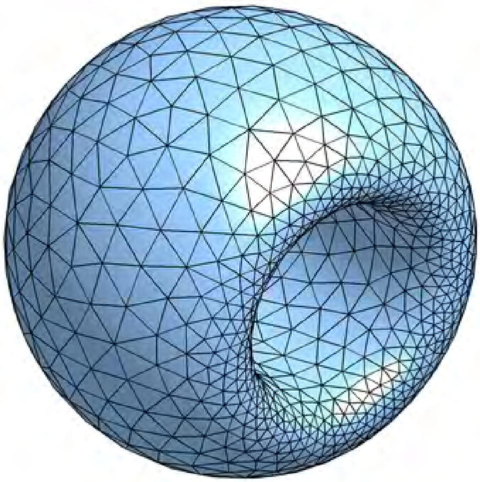






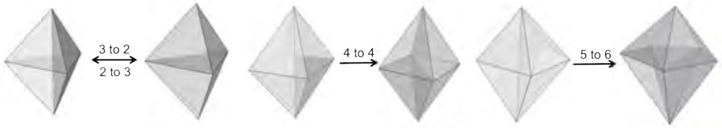


The DistMesh Mesh Generator on Surfaces

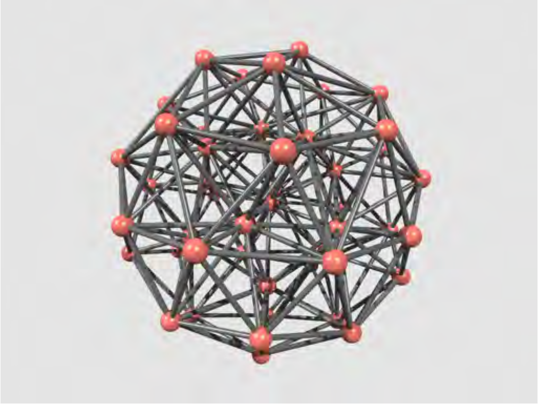


Element flips and DistMesh in 3D

- Local element flips for 3D tetrahedra:

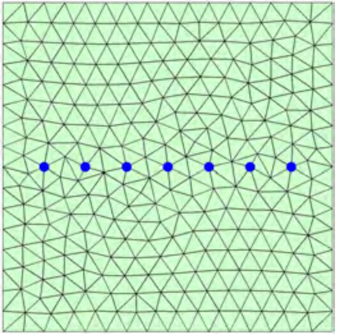


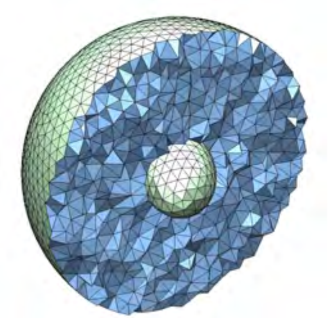
- Restricts the topology changes to a small number of elements



Moving Meshes

- In addition to generating high-quality initial meshes, the DistMesh algorithm is excellent for iterative generation of moving meshes
- The resulting mesh sequence involves two types of operations:
  1. Smooth node movements
  2. *Localized* element topology updates
- This allows for integration with efficient numerical schemes





Deep Reinforcement Learning for Optimal Block Mesh Generation

Scientific Achievement

A machine learning approach for optimal block mesh generation, using reinforcement learning to improve an initial Delaunay mesh using local topological mesh operations.

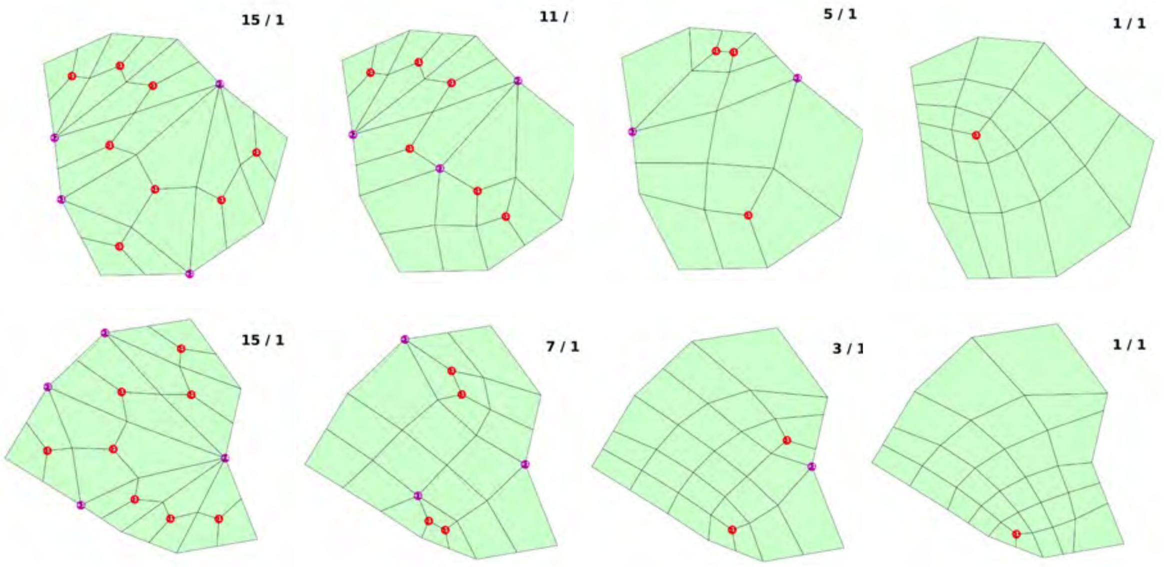
Significance and Impact

Mesh generation remains one of the major bottlenecks in many numerical simulations, e.g. in fluid dynamics. Well-shaped block meshes are ideal for most methods, including finite elements and (mapped) multi-block finite difference methods. This work is a major step towards automating the generation of the topological blocks for arbitrary geometries.

Technical Approach

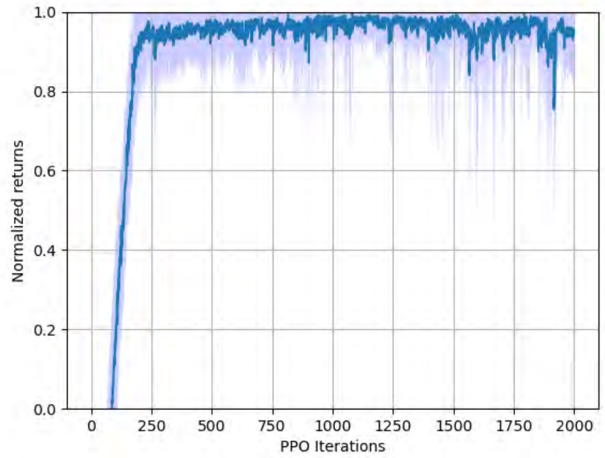
- Define an appropriate "game", where the moves are local topological operations and the score is based on the optimality of the mesh
- Use a half-edge mesh structure to define a CNN-type network which extends to fully unstructured quadrilateral meshes
- Train on random geometries, using the PPO algorithm on GPUs
- Consistently produces close-to-optimal meshes

PI(s)/Facility Lead(s): Per-Olof Persson, LBNL Math Group
ASCR Program: Base Math
ASCR PM: Steven Lee
Publication(s) for this work:
A. Narayanan, Y. Pan, P.-O. Persson, "Learning Topological Operations on Meshes," in review.



Top: Block meshing of two simple polygons. The initial meshes are split Delaunay triangles. The resulting block meshes have optimal regularities.

Right: The training using the PPO method, achieving about 99% returns.



Outline

1. Introduction and Motivation

2. Numerical Schemes – Discretization and Solvers
   - The Discontinuous Galerkin Method
   - Curved Mesh Generation
   - Time-Stepping and Parallel Implicit Solvers

3. Methods for Deforming Domains
   - High-Order ALE Formulation
   - Time-Dependent PDE-Constrained Optimization
   - DistMesh and Moving Meshes
   - Optimization-Based High-Order Shock Tracking

Optimization-Based High-Order Shock Tracking

(on separate slides)