Integrating computational physics and numerical optimization to address challenges in science, engineering, and medicine

Matthew J. Zahr
Assistant Professor
Department of Aerospace and Mechanical Engineering
University of Notre Dame

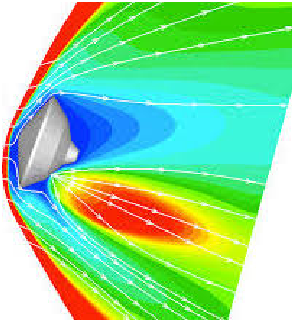
Applied and Computational Mathematics and Statistics (ACMS) Seminar
University of Notre Dame, Notre Dame, IN
April 5, 2021

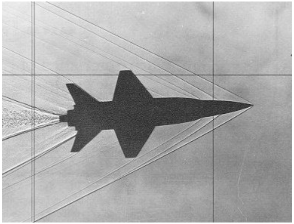
Students: T. Huang, M. Mirhoseini, C. Naudet, T. Wen
Collaborators: D.Z. Huang, P.-O. Persson, A. Shi, J. Töger

Discontinuities often arise in engineering systems, particularly in those involving compressible flows: shock waves, contact lines

Supersonic and transonic flow around commercial planes and fighter jets

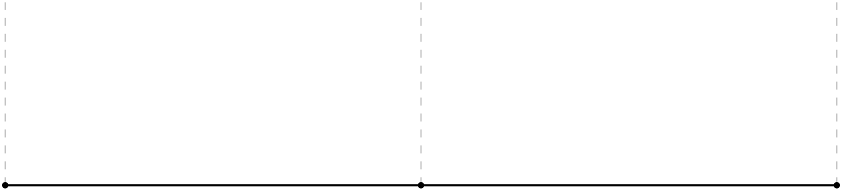
Hypersonics, e.g., re-entry of vehicles in atmosphere, and scramjets





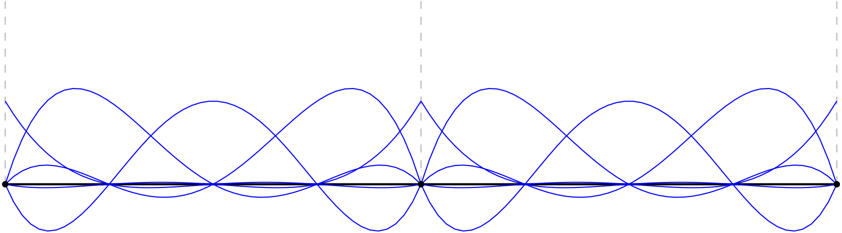
Other applications with discontinuities: fracture, problems with interfaces

Numerical methods for resolving shocks



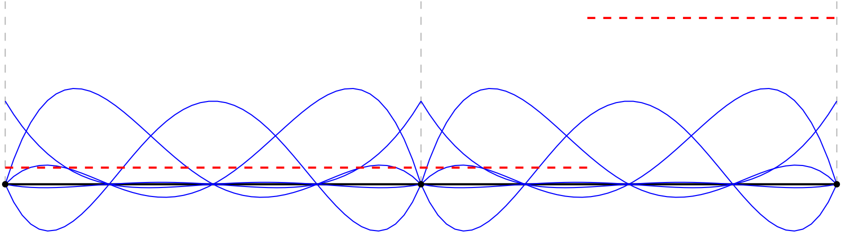
Fundamental issue: approximate discontinuity with polynomial basis

Numerical methods for resolving shocks



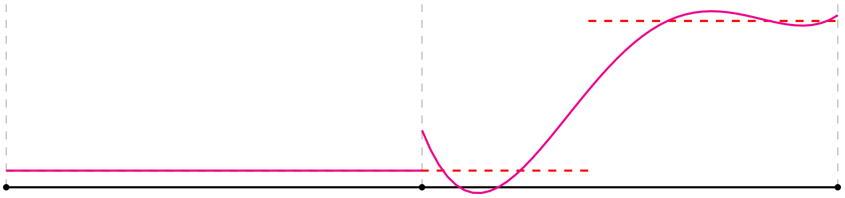
Fundamental issue: approximate discontinuity with polynomial basis

Numerical methods for resolving shocks



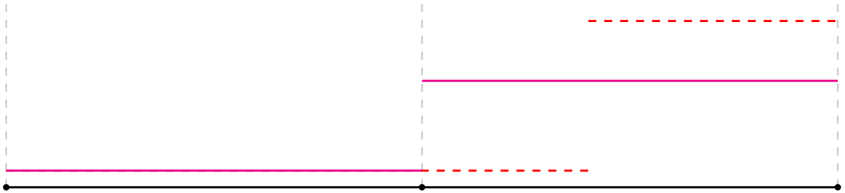
Fundamental issue: approximate discontinuity with polynomial basis

Numerical methods for resolving shocks



Fundamental issue: approximate discontinuity with polynomial basis

Numerical methods for resolving shocks

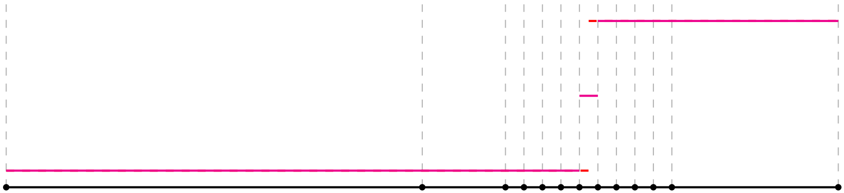


Fundamental issue: approximate discontinuity with polynomial basis

Exising solutions: **limiting**, artificial viscosity

Drawbacks: order reduction, local refinement

Numerical methods for resolving shocks

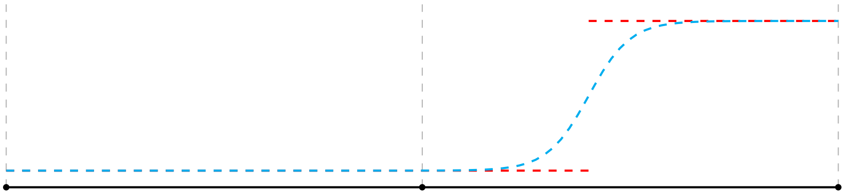


Fundamental issue: approximate discontinuity with polynomial basis

Exising solutions: **limiting**, artificial viscosity

Drawbacks: order reduction, local refinement

Numerical methods for resolving shocks

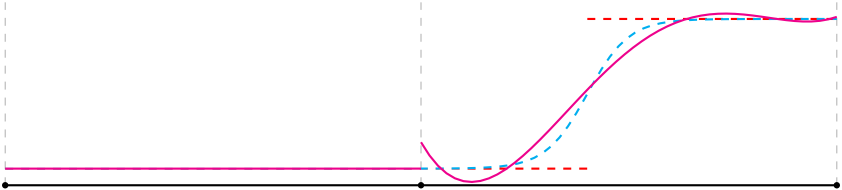


Fundamental issue: approximate discontinuity with polynomial basis

Exising solutions: limiting, **artificial viscosity**

Drawbacks: order reduction, local refinement

Numerical methods for resolving shocks

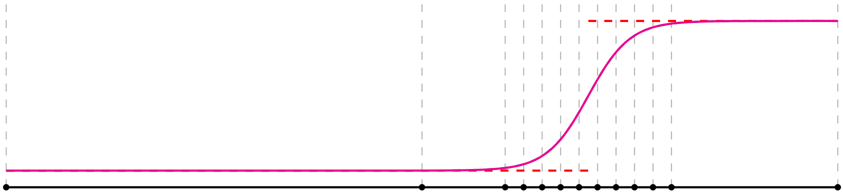


Fundamental issue: approximate discontinuity with polynomial basis

Exising solutions: limiting, **artificial viscosity**

Drawbacks: order reduction, local refinement

Numerical methods for resolving shocks

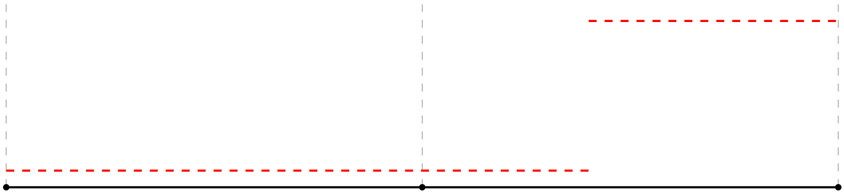


Fundamental issue: approximate discontinuity with polynomial basis

Exising solutions: limiting, **artificial viscosity**

Drawbacks: order reduction, local refinement

Numerical methods for resolving shocks



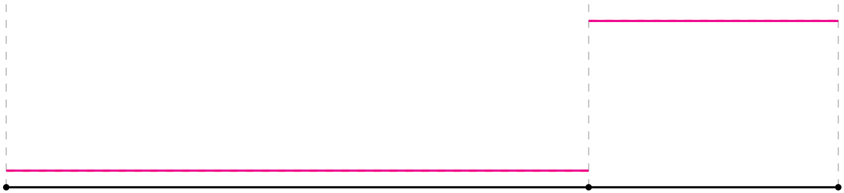
Fundamental issue: approximate discontinuity with polynomial basis

Exising solutions: limiting, artificial viscosity

Drawbacks: order reduction, local refinement

Shock tracking/fitting: align features of solution basis with features in the solution using optimization formulation and solver

Numerical methods for resolving shocks



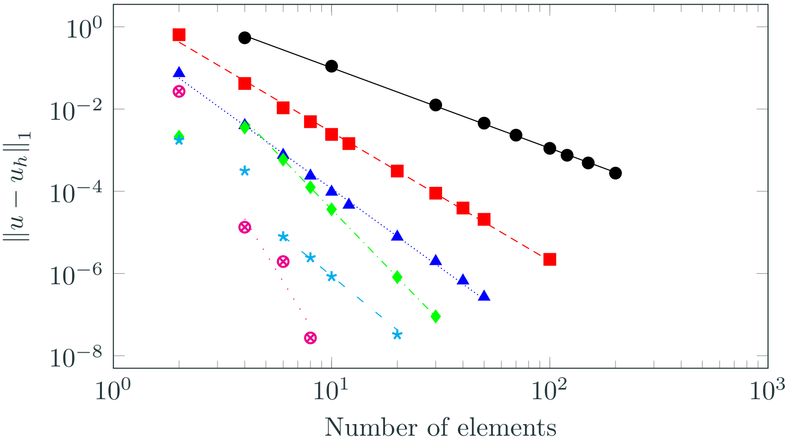
Fundamental issue: approximate discontinuity with polynomial basis

Exising solutions: limiting, artificial viscosity

Drawbacks: order reduction, local refinement

Shock tracking/fitting: align features of solution basis with features in the solution using optimization formulation and solver

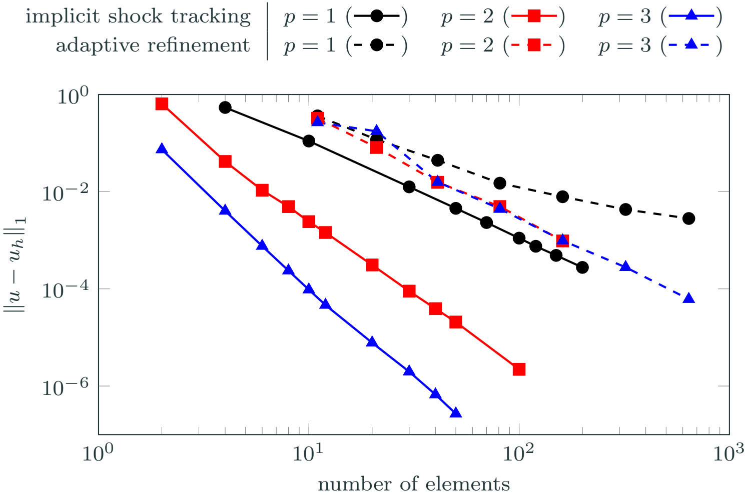
Why tracking: Recover optimal $\mathcal{O}(h^{p+1})$ convergence rates



Convergence of implicit shock tracking (Burgers' equation) with polynomial degrees $p = 1$ (●), $p = 2$ (■), $p = 3$ (▲), $p = 4$ (♦), $p = 5$ (∗), $p = 6$ (⊗).

Key observation: Optimal convergence rates ($\mathcal{O}(h^{p+1})$) attainable, even for discontinuous solutions.

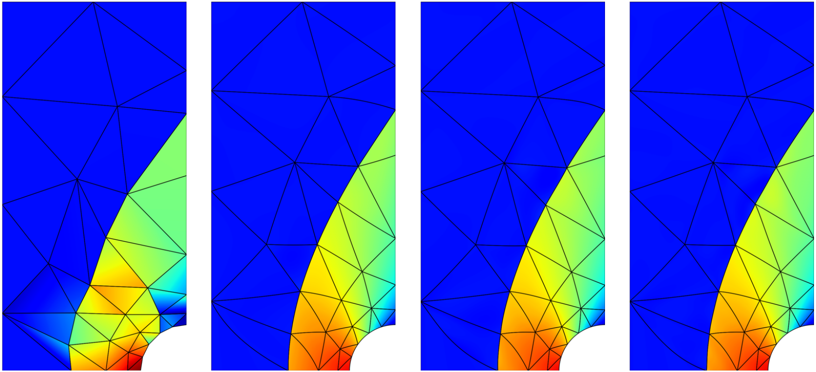
Why high-order tracking: Benefits more dramatic than low-order



Convergence of implicit shock tracking (Burgers' equation): implicit shock tracking (solid) vs. adaptive mesh refinement (dashed).

Key observation: Accuracy improvement of tracking approach relative to (specialized) adaptive mesh refinement is more exaggerated for high-order approximations: $\mathcal{O}(10^1)$ for $p = 1$ and $\mathcal{O}(10^6)$ for $p = 3$.

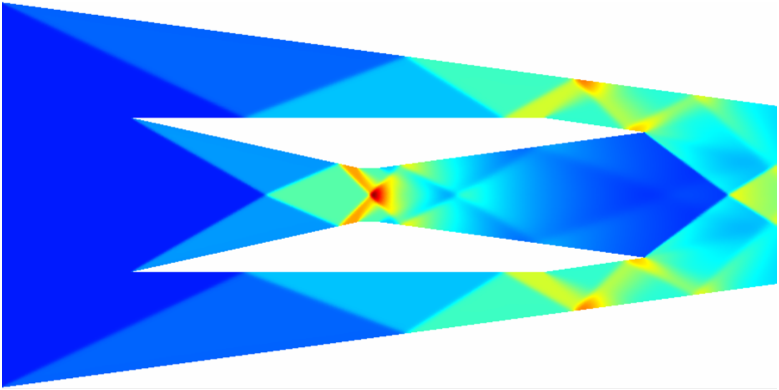
Why high-order tracking: Accurate solutions on coarse meshes



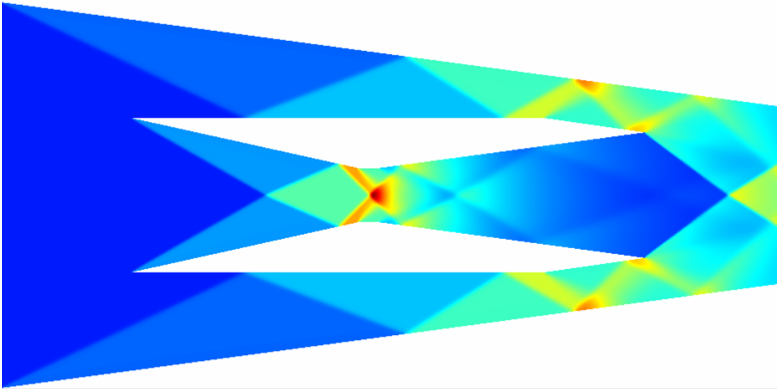
Density of supersonic flow ($M = 2$) past a cylinder using implicit shock tracking with $p = 1$ to $p = 4$ (left to right) DG discretization.

Key observation: High-order tracking enables accurate resolution of 2D supersonic flow with 48 elements; the error in the stagnation enthalpy is $\mathcal{O}(10^{-4})$ for $p = 2$ (1152 DoF).

Why not tracking: Difficult for complex discontinuity surfaces



Why not tracking: Difficult for complex discontinuity surfaces

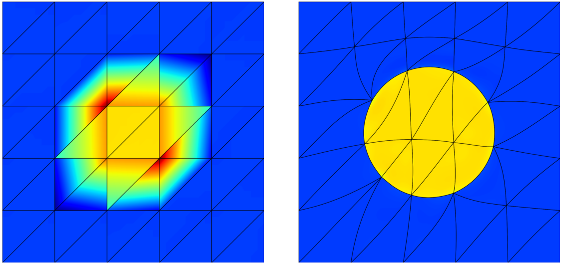


Implicit shock tracking

Aims to overcome the difficulty of explicitly meshing the unknown shock surface, e.g., HOIST [Zahr, Persson; 2018], MDG-ICE [Corrigan, Kercher, Kessler; 2019]

Implicit tracking for stable, high-order resolution of discontinuities

Goal: Align element faces with (unknown) discontinuities to perfectly capture them and approximate smooth regions to high-order



Non-aligned   Discontinuity-aligned

High-Order Implicit Shock Tracking (HOIST)[1]

- Discontinuous Galerkin discretization: inter-element jumps, high-order
- Discontinuity-aligned mesh: solution of optimization problem constrained by the discrete PDE $\implies$ **implicit tracking**
- Full space solver that converges the solution and mesh simultaneously to ensure solution of PDE never required on non-aligned mesh

[1] [Zahr, Persson; 2018], [Zahr, Shi, Persson; 2020]

Discontinuous Galerkin discretization of conservation law

Inviscid conservation law:

$$\nabla \cdot F(U) = 0 \quad \text{in } \Omega$$

Element-wise finite-dimensional weak form of conservation law:

$$r^K_{h,p'}(U_{h,p}) := \int_{\partial K} \psi^+_{h,p'} \cdot \mathcal{H}(U^+_{h,p}, U^-_{h,p}, n)\, dS - \int_K F(U_{h,p}) : \nabla \psi_{h,p'}\, dV,$$

where $\mathcal{V}_{h,p'}$ is the test space, $\mathcal{V}_{h,p}$ is the trial space, $\mathcal{H}$ is the numerical flux function, h is element size, and p/p' is the polynomial degree.

Introduce basis for polynomial spaces to obtain discrete residuals

$$\boldsymbol{r}(\boldsymbol{u}, \boldsymbol{x}) \quad (p' = p), \qquad \boldsymbol{R}(\boldsymbol{u}, \boldsymbol{x}) \quad (p' = p + 1),$$

where $\boldsymbol{u}$ is the discrete state vector and $\boldsymbol{x}$ are the coordinates of the mesh nodes.

Implicit shock tracking: constrained optimization formulation

We formulate the problem of tracking discontinuities with the mesh as the solution of an optimization problem constrained by the discrete PDE (DG discretization)

$$\begin{aligned} \underset{\boldsymbol{u},\boldsymbol{x}}{\text{minimize}} \quad & f(\boldsymbol{u},\boldsymbol{x}) := \frac{1}{2}\left\|\boldsymbol{F}(\boldsymbol{u},\boldsymbol{x})\right\|_2^2 \\ \text{subject to} \quad & \boldsymbol{r}(\boldsymbol{u},\boldsymbol{x}) = \boldsymbol{0}. \end{aligned}$$

The objective function *balances* tracking and mesh quality

$$\boldsymbol{F}(\boldsymbol{u},\boldsymbol{x}) = \begin{bmatrix} \boldsymbol{R}(\boldsymbol{u},\boldsymbol{x}) \\ \kappa \boldsymbol{R}_{\text{msh}}(\boldsymbol{x}) \end{bmatrix}$$

$\boldsymbol{r}(\boldsymbol{u},\boldsymbol{x}) = \boldsymbol{0}$ (DG equation), $\boldsymbol{u}$ (discrete state vector), $\boldsymbol{x}$ (coordinates of mesh nodes)

$\boldsymbol{R}$ (tracking term): penalizes the DG residual in the *enriched test space*

$\boldsymbol{R}_{\text{msh}}$ (mesh term): accounts for the distortion of each high-order element

κ: mesh distortion penalization parameter

Implicit shock tracking: sequential quadratic programming solver

Define $\boldsymbol{z} = (\boldsymbol{u}, \boldsymbol{x})$ and use interchangeably. To solve the optimization problem, we define a sequence $\{\boldsymbol{z}_k\}$ updated as

$$\boldsymbol{z}_{k+1} = \boldsymbol{z}_k + \alpha_k \Delta \boldsymbol{z}_k.$$

Implicit shock tracking: sequential quadratic programming solver

Define $\boldsymbol{z} = (\boldsymbol{u}, \boldsymbol{x})$ and use interchangeably. To solve the optimization problem, we define a sequence $\{\boldsymbol{z}_k\}$ updated as

$$\boldsymbol{z}_{k+1} = \boldsymbol{z}_k + \alpha_k \Delta \boldsymbol{z}_k.$$

The step direction $\Delta \boldsymbol{z}_k$ is defined as the solution of the quadratic program (QP) approximation of the tracking problem centered at $\boldsymbol{z}_k$

$$\begin{aligned} \underset{\Delta \boldsymbol{z} \in \mathbb{R}^{N_{\boldsymbol{z}}}}{\text{minimize}} \quad & \boldsymbol{g}_{\boldsymbol{z}}(\boldsymbol{z}_k)^T \Delta \boldsymbol{z} + \frac{1}{2} \Delta \boldsymbol{z}^T \boldsymbol{B}_{\boldsymbol{z}}(\boldsymbol{z}_k, \hat{\boldsymbol{\lambda}}(\boldsymbol{z}_k)) \Delta \boldsymbol{z} \\ \text{subject to} \quad & \boldsymbol{r}(\boldsymbol{z}_k) + \boldsymbol{J}_{\boldsymbol{z}}(\boldsymbol{z}_k) \Delta \boldsymbol{z} = \boldsymbol{0}, \end{aligned}$$

where

$$\boldsymbol{g}_{\boldsymbol{z}}(\boldsymbol{z}) = \frac{\partial f}{\partial \boldsymbol{z}}(\boldsymbol{z})^T, \quad \boldsymbol{J}_{\boldsymbol{z}}(\boldsymbol{z}) = \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{z}}(\boldsymbol{z}), \qquad \boldsymbol{B}_{\boldsymbol{z}}(\boldsymbol{z}, \boldsymbol{\lambda}) \approx \frac{\partial^2 \mathcal{L}}{\partial \boldsymbol{z} \partial \boldsymbol{z}}(\boldsymbol{z}, \boldsymbol{\lambda}),$$

$$\mathcal{L}(\boldsymbol{z}, \boldsymbol{\lambda}) = f(\boldsymbol{z}) - \boldsymbol{\lambda}^T \boldsymbol{r}(\boldsymbol{z}) \qquad \text{(Lagrangian)}$$

$$\hat{\boldsymbol{\lambda}}(\boldsymbol{z}) = \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}}(\boldsymbol{z})^{-T} \frac{\partial f}{\partial \boldsymbol{u}}(\boldsymbol{z})^T \qquad \text{(Lagrange mulitplier estimate)}$$

Implicit shock tracking: sequential quadratic programming solver

The solution of the quadratic program leads to the following linear system

$$\begin{bmatrix} \boldsymbol{B}_{\boldsymbol{uu}}(\boldsymbol{z}_k, \hat{\boldsymbol{\lambda}}(\boldsymbol{z}_k)) & \boldsymbol{B}_{\boldsymbol{ux}}(\boldsymbol{z}_k, \hat{\boldsymbol{\lambda}}(\boldsymbol{z}_k)) & \boldsymbol{J}_{\boldsymbol{u}}(\boldsymbol{z}_k)^T \\ \boldsymbol{B}_{\boldsymbol{ux}}(\boldsymbol{z}_k, \hat{\boldsymbol{\lambda}}(\boldsymbol{z}_k))^T & \boldsymbol{B}_{\boldsymbol{xx}}(\boldsymbol{z}_k, \hat{\boldsymbol{\lambda}}(\boldsymbol{z}_k)) & \boldsymbol{J}_{\boldsymbol{x}}(\boldsymbol{z}_k)^T \\ \boldsymbol{J}_{\boldsymbol{u}}(\boldsymbol{z}_k) & \boldsymbol{J}_{\boldsymbol{x}}(\boldsymbol{z}_k) & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \Delta\boldsymbol{u}_k \\ \Delta\boldsymbol{x}_k \\ \boldsymbol{\eta}_k \end{bmatrix} = - \begin{bmatrix} \boldsymbol{g}_{\boldsymbol{u}}(\boldsymbol{z}_k) \\ \boldsymbol{g}_{\boldsymbol{x}}(\boldsymbol{z}_k) \\ \boldsymbol{r}(\boldsymbol{z}_k) \end{bmatrix},$$

where

$$\boldsymbol{g}_{\boldsymbol{u}}(\boldsymbol{z}) = \frac{\partial f}{\partial \boldsymbol{u}}(\boldsymbol{z})^T, \quad \boldsymbol{J}_{\boldsymbol{u}}(\boldsymbol{z}) = \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}}(\boldsymbol{z}), \quad \boldsymbol{g}_{\boldsymbol{x}}(\boldsymbol{z}) = \frac{\partial f}{\partial \boldsymbol{x}}(\boldsymbol{z})^T, \quad \boldsymbol{J}_{\boldsymbol{x}}(\boldsymbol{z}) = \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{x}}(\boldsymbol{z}),$$

the approximate Hessian of the Lagrangian is taken as

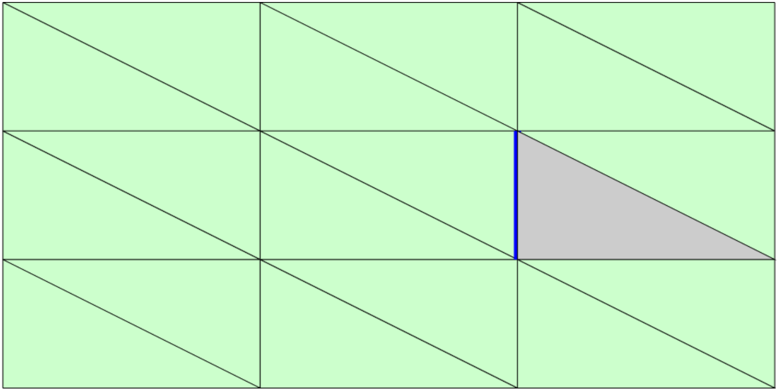
$$\begin{aligned} \boldsymbol{B}_{\boldsymbol{uu}}(\boldsymbol{z}, \boldsymbol{\lambda}) &= \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{u}}(\boldsymbol{z})^T \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{u}}(\boldsymbol{z}), \quad \boldsymbol{B}_{\boldsymbol{ux}}(\boldsymbol{z}, \boldsymbol{\lambda}) = \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{u}}(\boldsymbol{z})^T \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{x}}(\boldsymbol{z}), \\ \boldsymbol{B}_{\boldsymbol{xx}}(\boldsymbol{z}, \boldsymbol{\lambda}) &= \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{x}}(\boldsymbol{z})^T \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{x}}(\boldsymbol{z}) + \gamma \boldsymbol{D}, \end{aligned}$$

and $\boldsymbol{\eta}_k$ are the Lagrange multipliers of the QP and $\boldsymbol{D}$ is a mesh regularization matrix (linear elasticity stiffness).

Practical considerations: shock-aware element collapse

Despite measures to keep mesh well-conditioned, best option may be to *remove* element from the mesh: tag elements for removal based on volume and minimum edge length, collapse shortest edge

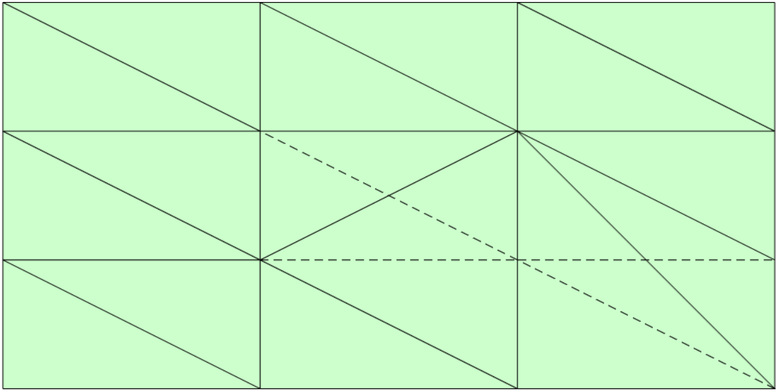
- Well-defined for simplices of any order in any dimension



Practical considerations: shock-aware element collapse

Despite measures to keep mesh well-conditioned, best option may be to *remove* element from the mesh: tag elements for removal based on volume and minimum edge length, collapse shortest edge

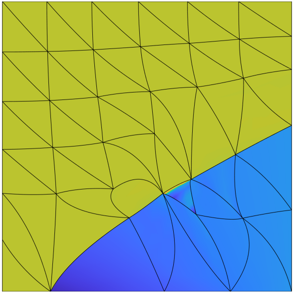
- Well-defined for simplices of any order in any dimension



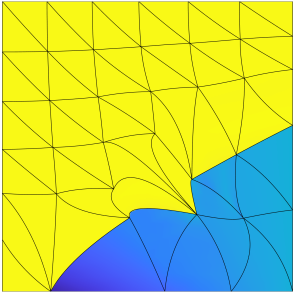
Practical considerations: shock-aware element collapse

Despite measures to keep mesh well-conditioned, best option may be to *remove* element from the mesh: tag elements for removal based on volume and minimum edge length, collapse shortest edge

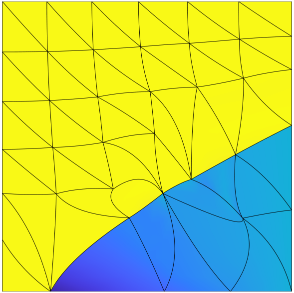
- Well-defined for simplices of any order in any dimension
- Must preserve boundaries and shock



before collapse



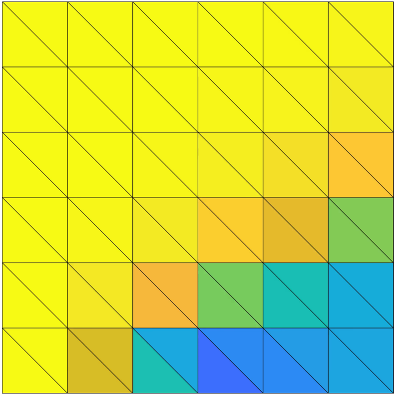
ignore shock



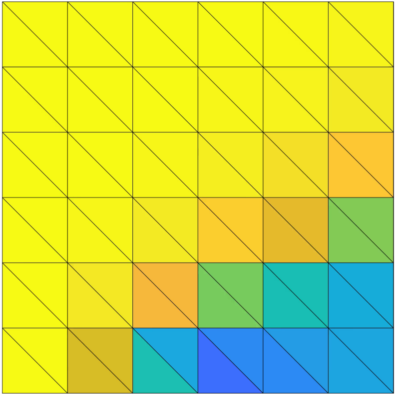
shock-aware

Practical considerations: solution re-initialization

- High-order solutions can become oscillatory, which leads to poor SQP steps (requiring many line search iterations)
- Overcome by replacing element-wise solution with the element-wise average (oscillatory element identified using Persson-Peraire indicator)
- Without re-initialization, must hope oscillatory elements get collapsed



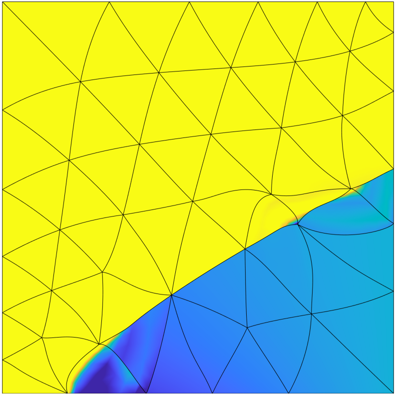
without re-initialization



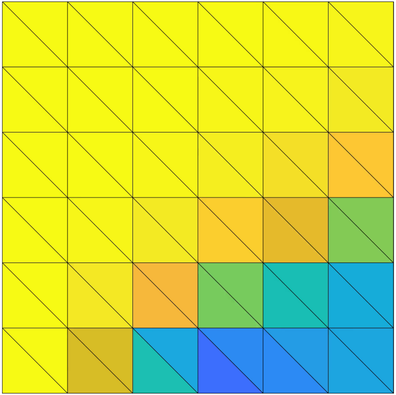
with re-initialization

Practical considerations: solution re-initialization

- High-order solutions can become oscillatory, which leads to poor SQP steps (requiring many line search iterations)
- Overcome by replacing element-wise solution with the element-wise average (oscillatory element identified using Persson-Peraire indicator)
- Without re-initialization, must hope oscillatory elements get collapsed



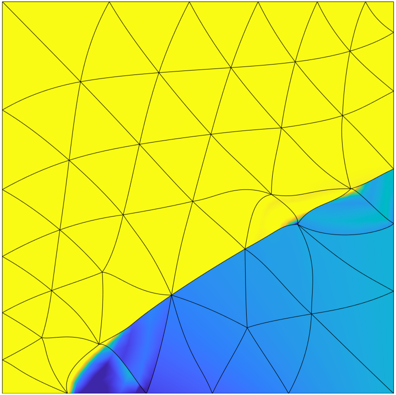
without re-initialization



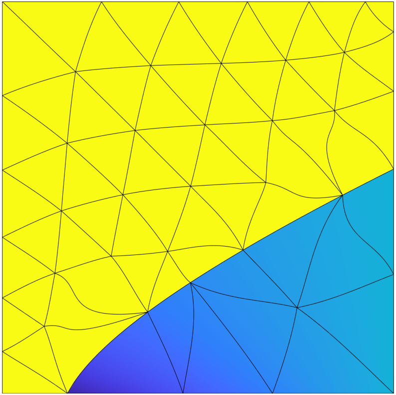
with re-initialization

Practical considerations: solution re-initialization

- High-order solutions can become oscillatory, which leads to poor SQP steps (requiring many line search iterations)
- Overcome by replacing element-wise solution with the element-wise average (oscillatory element identified using Persson-Peraire indicator)
- Without re-initialization, must hope oscillatory elements get collapsed



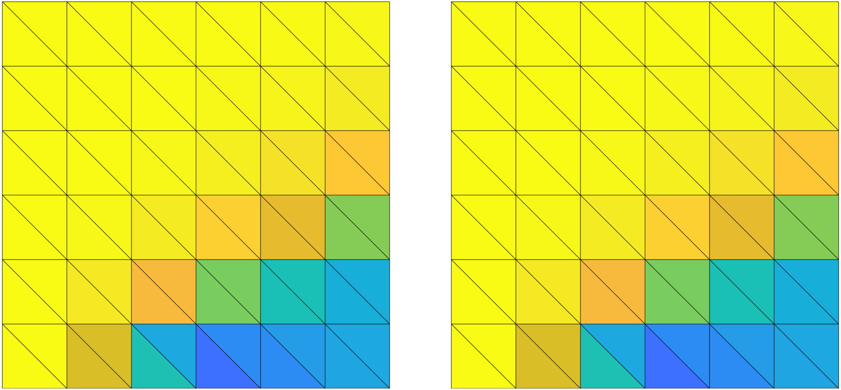
without re-initialization



with re-initialization

HOIST optimization problem is *non-convex* so initialization of $\boldsymbol{u}$, $\boldsymbol{x}$ is critical

- $\boldsymbol{x}_0$: directly from mesh generation
- $\boldsymbol{u}_0$: DG$(p = 0)$ solution on mesh $\boldsymbol{x}_0$

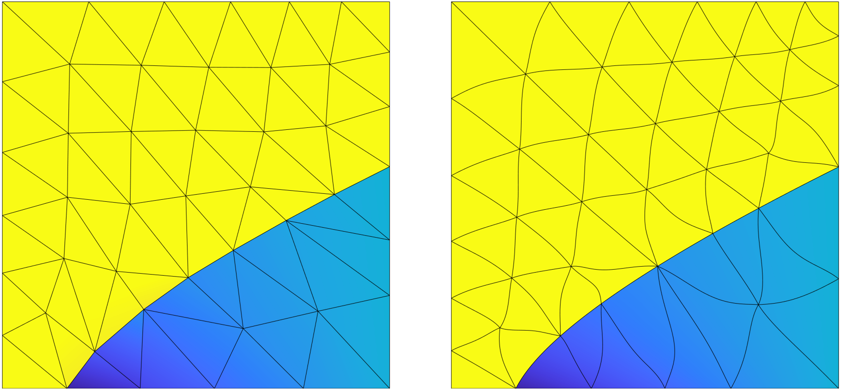


Reference mesh, $p = 0$ DG solution

Practical considerations: initialization

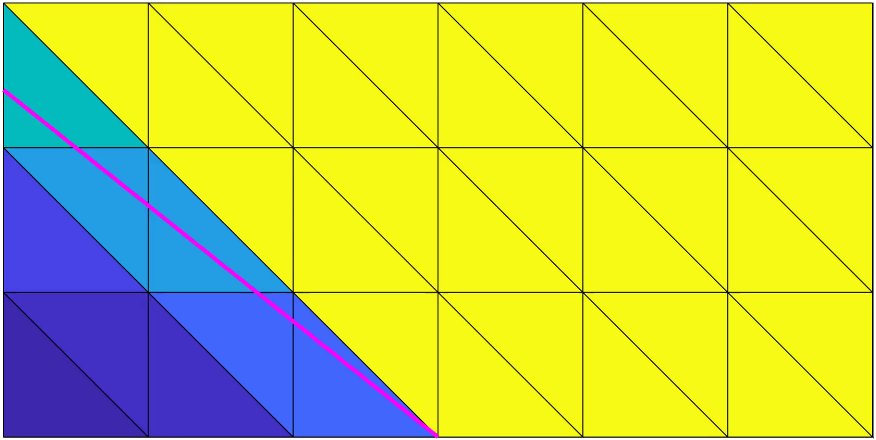
HOIST optimization problem is *non-convex* so initialization of $\boldsymbol{u}$, $\boldsymbol{x}$ is critical

- $\boldsymbol{x}_0$: directly from mesh generation
- $\boldsymbol{u}_0$: DG($p = 0$) solution on mesh $\boldsymbol{x}_0$



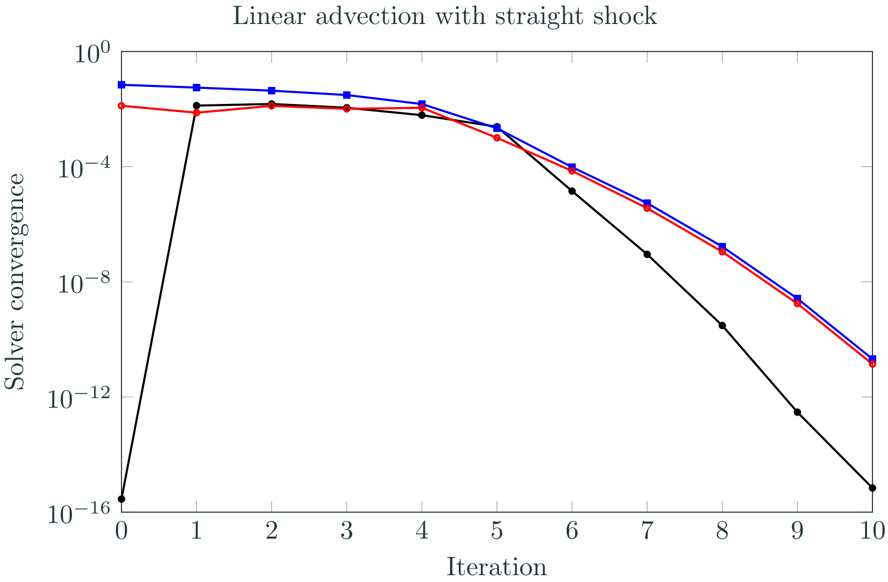
$p = 1$ (*left*) and $p = 4$ (*right*) tracking solution

Linear advection (2D), straight shock



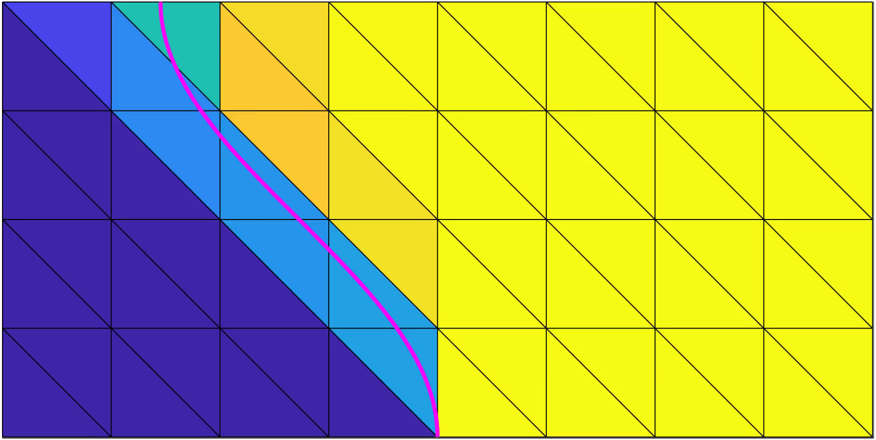
$p = 0$ space for solution, $q = 1$ space for mesh

Newton-like convergence when solution lies in DG subspace



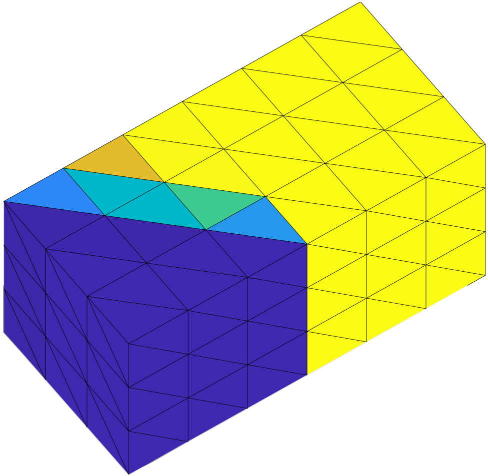
$\|\boldsymbol{r}(\boldsymbol{z})\|$ (—●—), $\|\boldsymbol{R}(\boldsymbol{z})\|$ (—■—), $\left\|\nabla_{\boldsymbol{x}}\mathcal{L}(\boldsymbol{z},\hat{\boldsymbol{\lambda}}(\boldsymbol{z}))\right\|$ (—○—)

Linear advection (2D), trigonometic shock



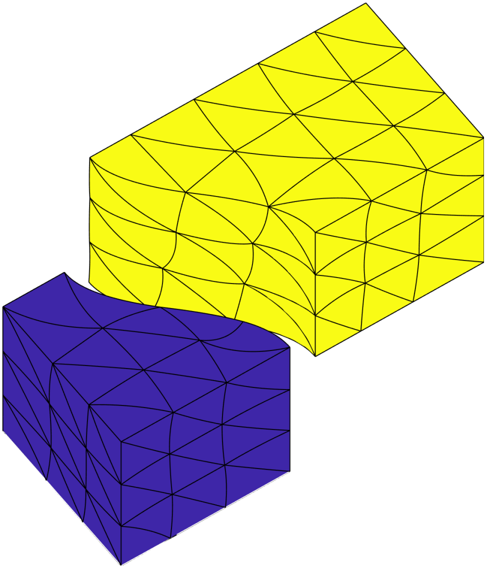
$p = 0$ space for solution, $q = 2$ space for mesh

Linear advection (3D), trigonometric shock



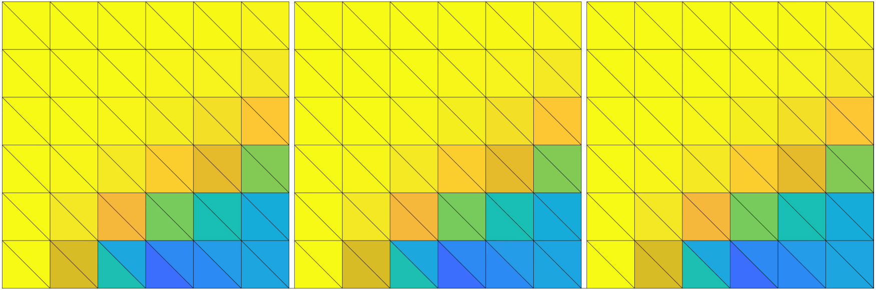
$p = 0$ space for solution, $q = 2$ space for mesh

Linear advection (3D), trigonometric shock

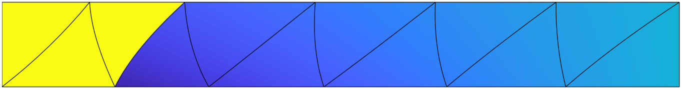


$p = 0$ space for solution, $q = 2$ space for mesh

Burgers' equation, accelerating shock

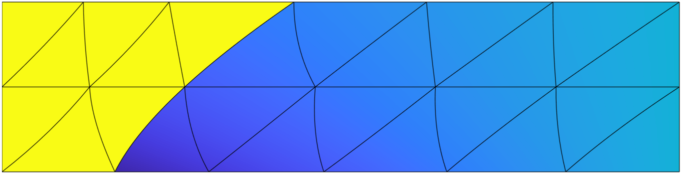


$p = q = 1$ $\qquad$ $p = q = 2$ $\qquad$ $p = q = 3$



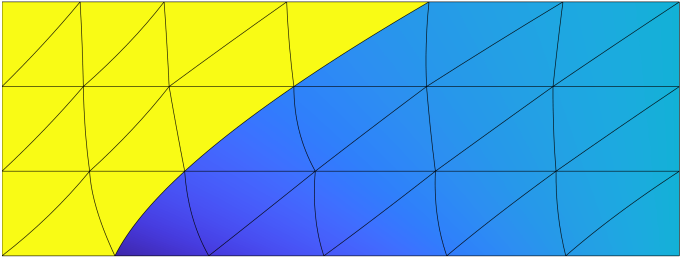
Observation: Monolithic space-time formulation not always practical; use implicit shock tracking over sequence of space-time slabs.

Burgers' equation, accelerating shock: space-time slabs



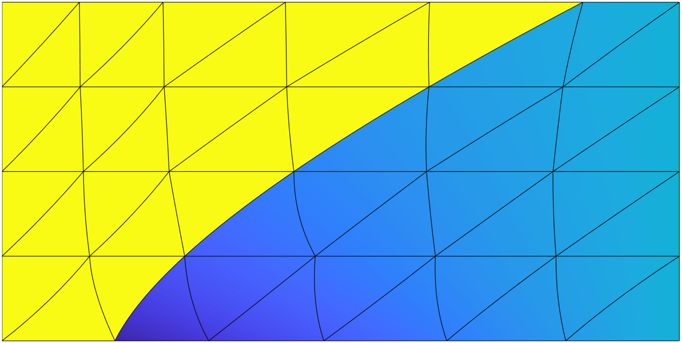
Observation: Monolithic space-time formulation not always practical; use implicit shock tracking over sequence of space-time slabs.

Burgers' equation, accelerating shock: space-time slabs



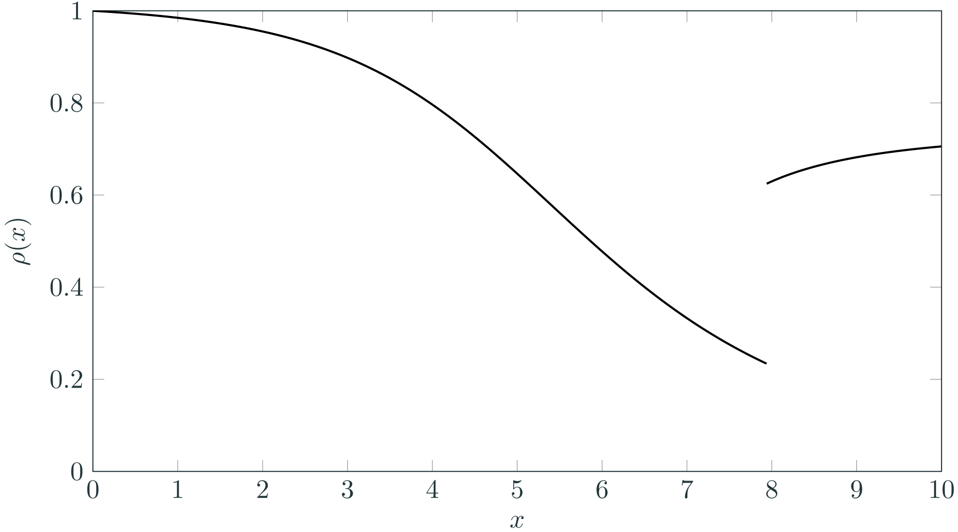
Observation: Monolithic space-time formulation not always practical; use implicit shock tracking over sequence of space-time slabs.

Burgers' equation, accelerating shock: space-time slabs



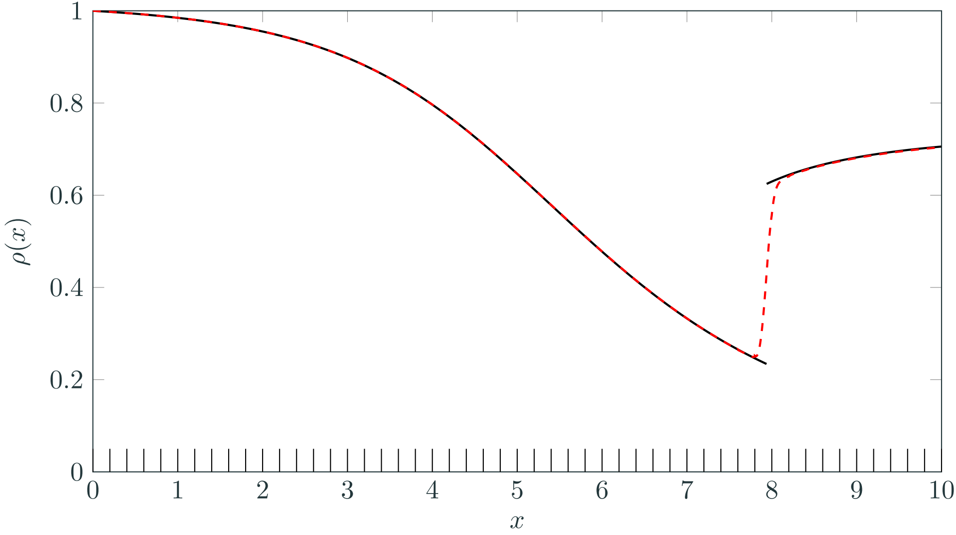
Observation: Monolithic space-time formulation not always practical; use implicit shock tracking over sequence of space-time slabs.

Inviscid flow through area variation: HOIST vs capturing ($p = 4$)



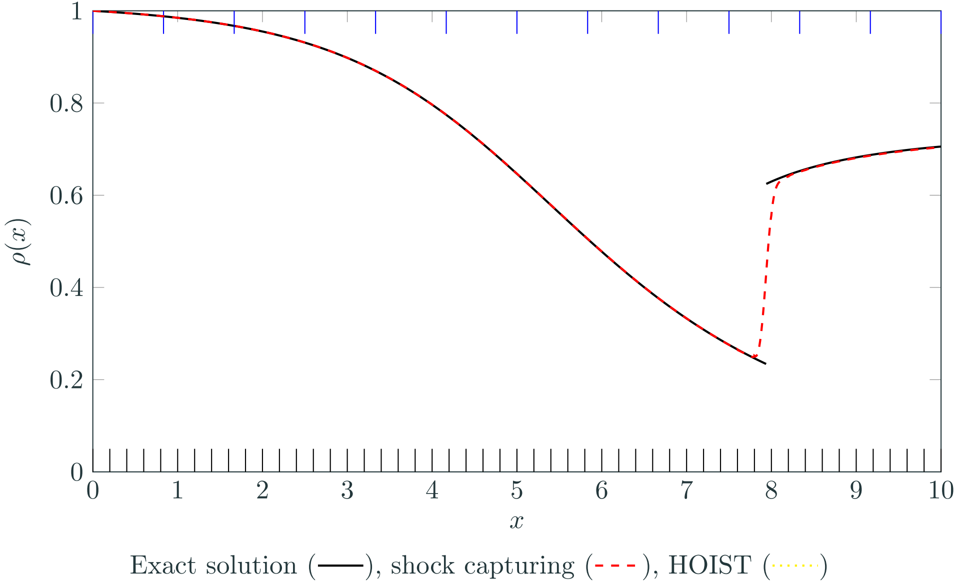
Exact solution (——), shock capturing (- - -), HOIST (⋯⋯)

Inviscid flow through area variation: HOIST vs capturing ($p = 4$)

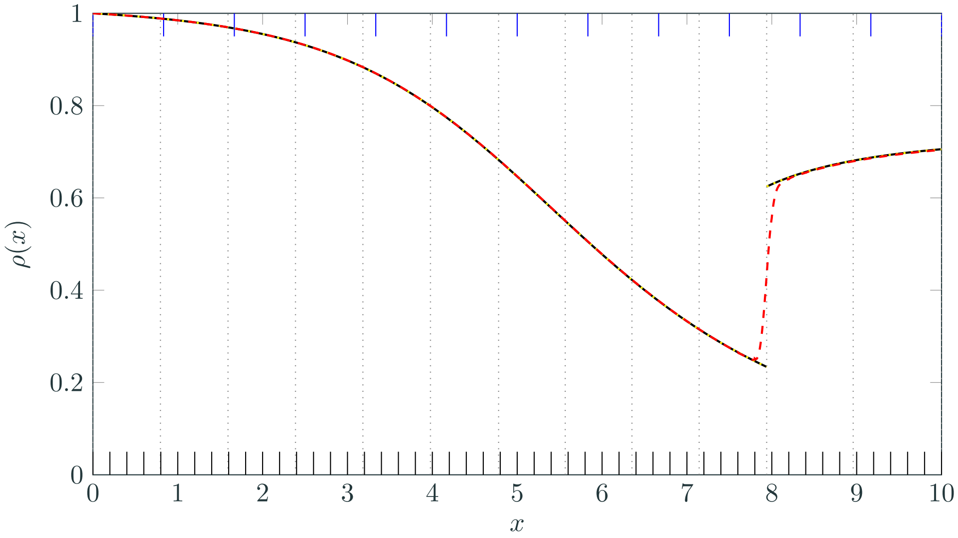


Exact solution (——), shock capturing (- - -), HOIST (······)

Inviscid flow through area variation: HOIST vs capturing ($p = 4$)

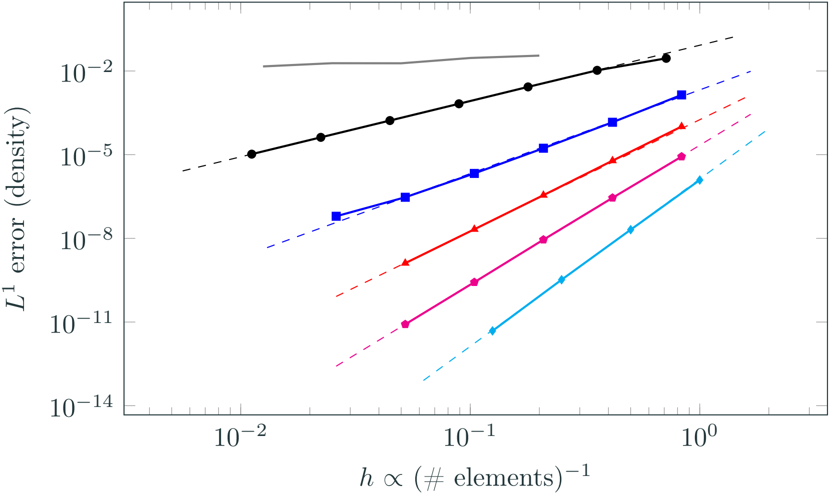


Inviscid flow through area variation: HOIST vs capturing ($p = 4$)



Exact solution (——), shock capturing (- - -), HOIST (⋯⋯)

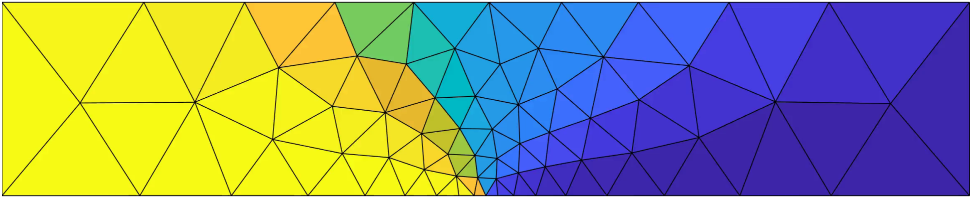
Inviscid flow through area variation: h-convergence



Shock capturing: $p = 4$ (gray line); HOIST: $p = 1$ (black, circle), $p = 2$ (blue, square), $p = 3$ (red, triangle), $p = 4$ (magenta, pentagon), $p = 5$ (cyan, diamond); dashed line indicates optimal convergence rate ($\mathcal{O}(h^{p+1})$)

Observation: Shock capturing limited to sub-first-order convergence rate; HOIST achieves optimal convergence rates ($\mathcal{O}(h^{p+1})$) and high accuracy per DoF

Unsteady, inviscid flow, space-time: Sod shock tube



$p = 2$, $q = 1$

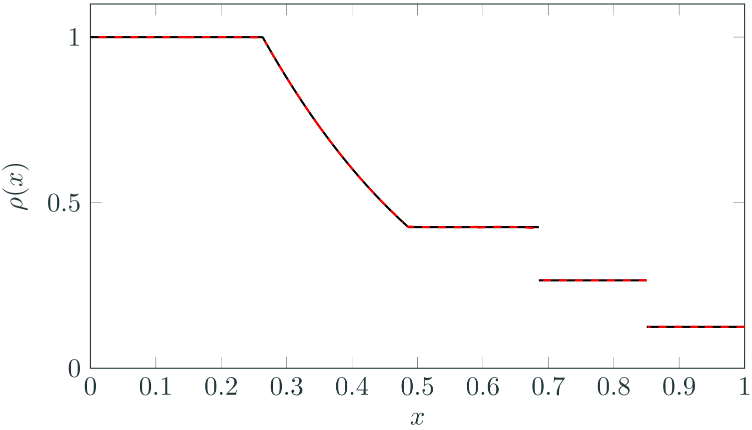
Observation: Tracks multiple features including discontinuities and derivative jumps; stronger features "easier" to track (track earlier in process).

Unsteady, inviscid flow, space-time: Sod shock tube

$p = 2$, $q = 1$

Observation: Tracks multiple features including discontinuities and derivative jumps; stronger features "easier" to track (track earlier in process).

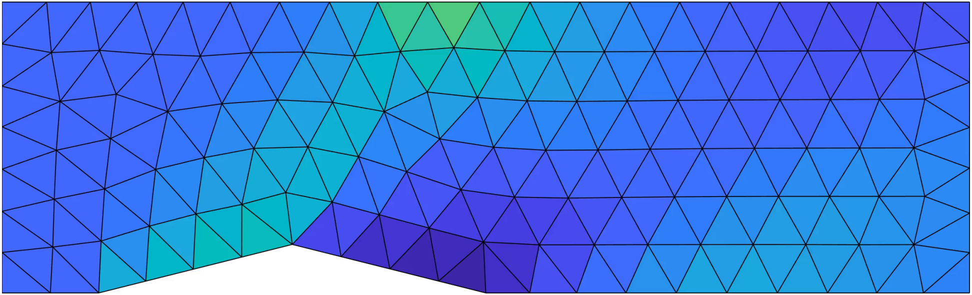
Unsteady, inviscid flow, space-time: Sod shock tube



Exact (——), $p = 2$ ($q = 1$) HOIST (- - -)

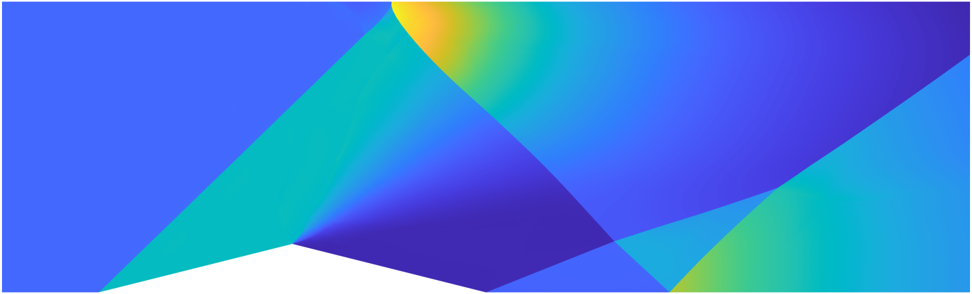
Observation: Tracks multiple features including discontinuities and derivative jumps; stronger features "easier" to track (track earlier in process).

2D Supersonic flow: $M = 2$ flow over diamond



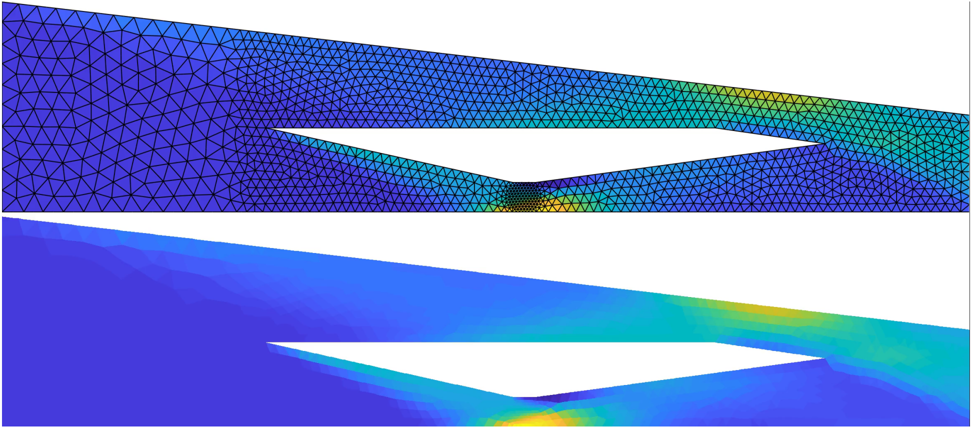
$p = q = 2$

2D Supersonic flow: $M = 2$ flow over diamond



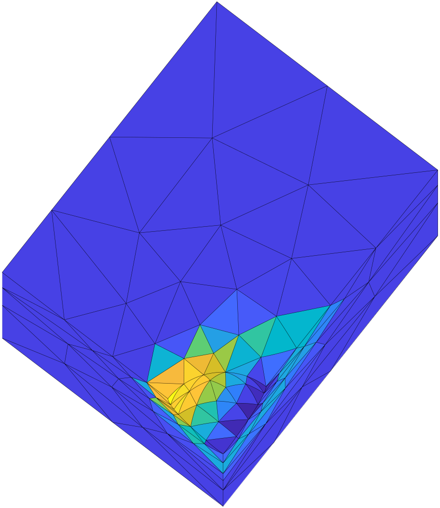
$p = q = 2$

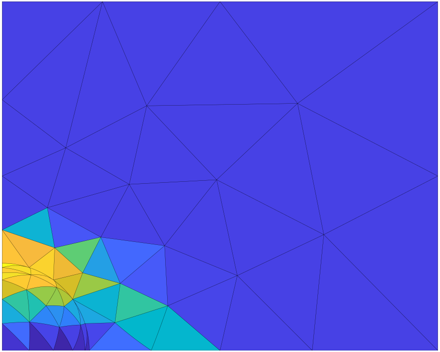
2D Hypersonic flow: $M = 5$ flow through scramjet



$p = q = 2$

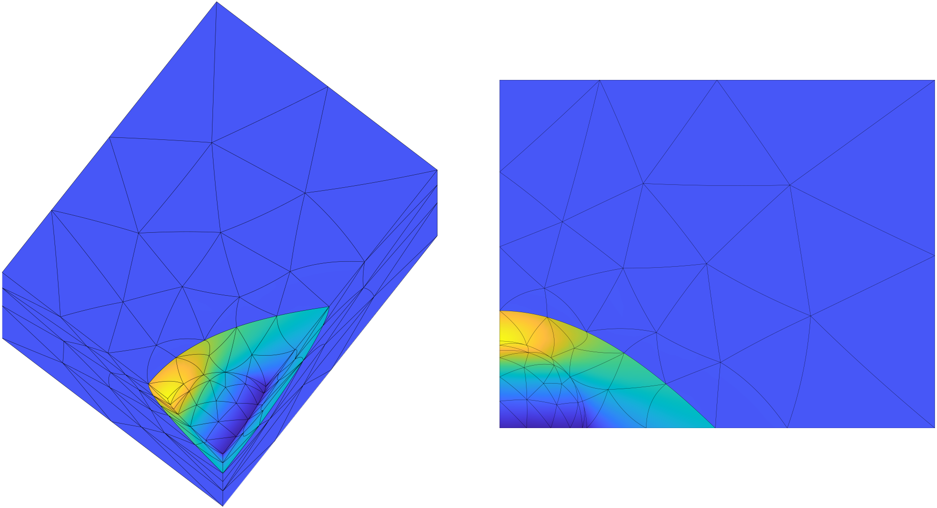
3D Supersonic flow: $M = 2$ flow over sphere





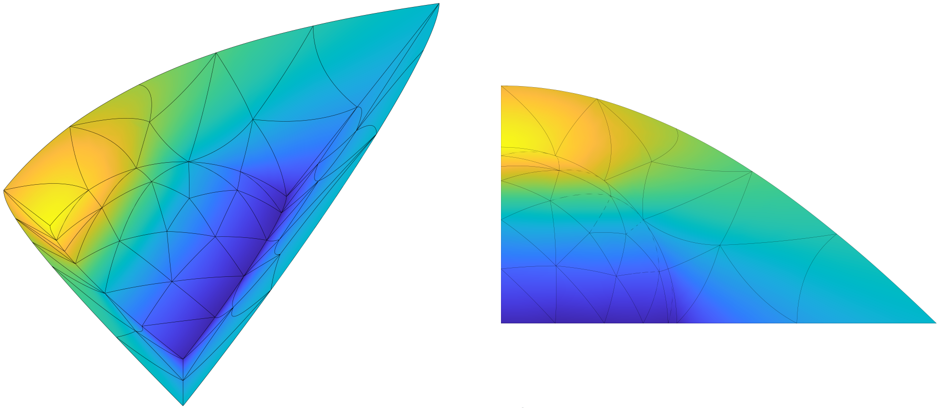
$p = q = 2$

3D Supersonic flow: $M = 2$ flow over sphere



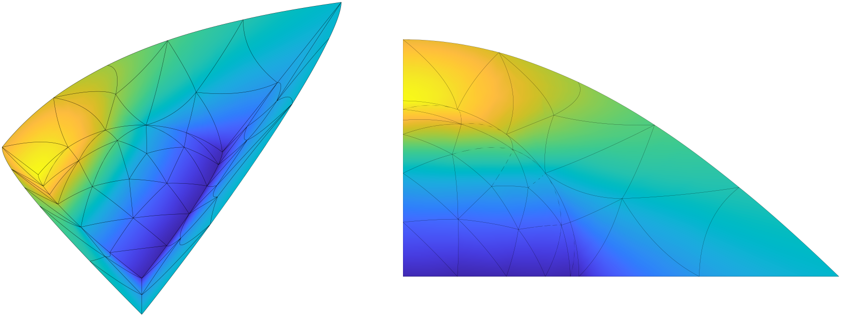
$p = q = 2$

3D Supersonic flow: $M = 2$ flow over sphere



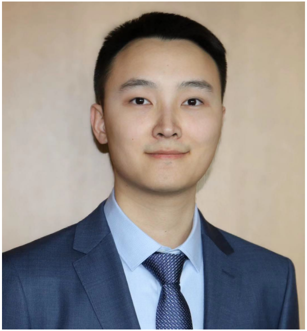
$p = q = 2$

- **Implicit tracking:** formulate tracking as optimization problem over $(\boldsymbol{u}, \boldsymbol{x})$
- Highly *accurate solutions* on coarse meshes, *optimal convergence* rates
- High-order methods exaggerate accuracy benefits of tracking discontinuities
- Traditional barrier to tracking (explicitly meshing unknown discontinuity surface) replaced with solving constrained optimization problem

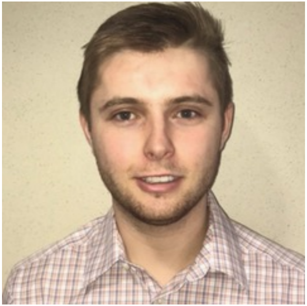


Acknowledgments

- DOE Grant DE-AC02-05CH1123 (Alvarez fellowship)
- AFOSR Grant FA9550-20-1-0236 (F. Fahroo)



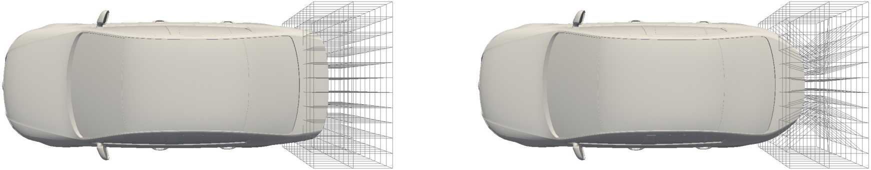
Tianci Huang (ND)
robust solvers



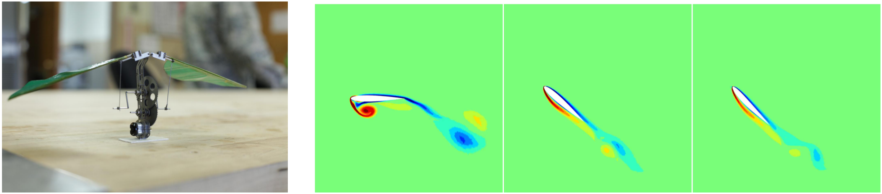
Charles Naudet (ND)
space-time slabs

PDE optimization is ubiquitous in science and engineering

Design: Find system that optimizes performance metric, satisfies constraints



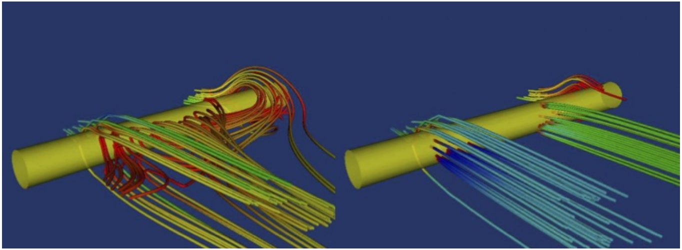
Aerodynamic shape design of automobile



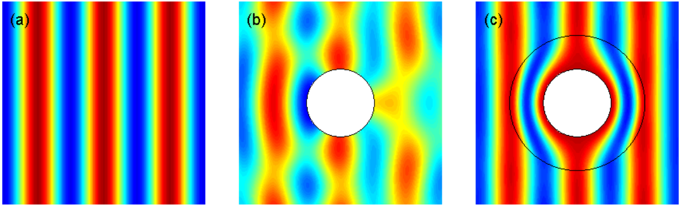
Optimal flapping motion of micro aerial vehicle

PDE optimization is ubiquitous in science and engineering

Control: Drive system to a desired state



Boundary flow control



Metamaterial cloaking – electromagnetic invisibility

PDE-constrained optimization formulation

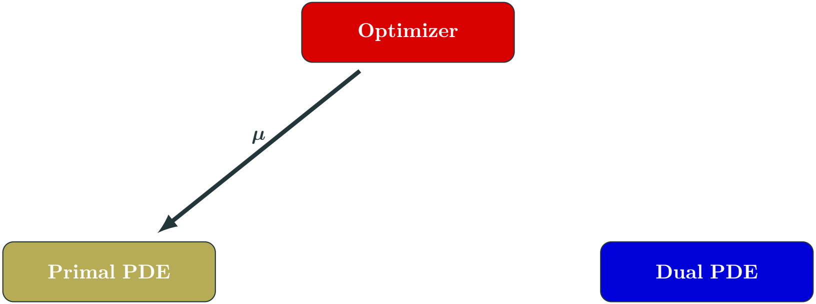
Goal: Find the solution of the *PDE-constrained optimization* problem

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

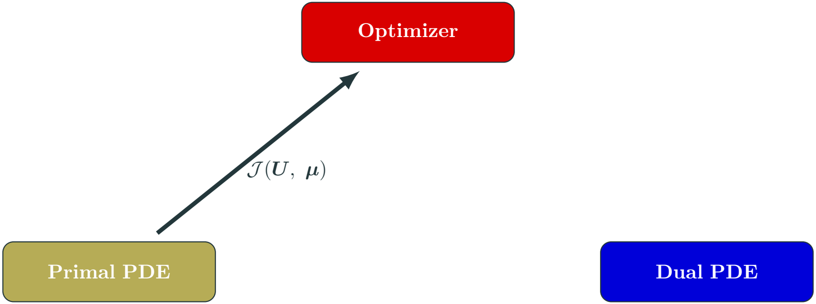
$\boldsymbol{U}$	:	PDE solution
$\boldsymbol{\mu}$	:	design/control parameters
$\mathcal{J}(\boldsymbol{U}, \boldsymbol{\mu})$	:	objective function
$\boldsymbol{C}(\boldsymbol{U}, \boldsymbol{\mu})$	:	constraints
$\boldsymbol{F}(\boldsymbol{U}, \nabla \boldsymbol{U})$	:	conservation law flux function

Nested approach to PDE-constrained optimization

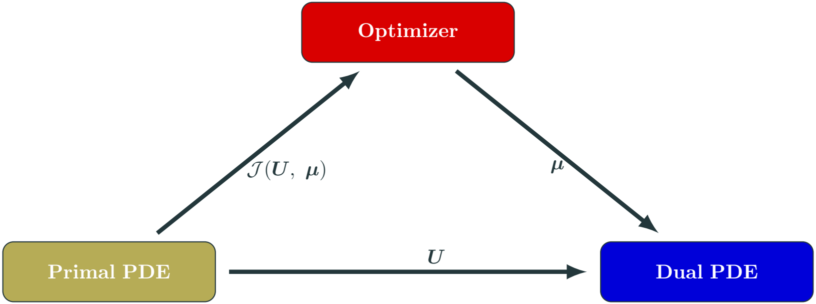
Nested approach to PDE-constrained optimization



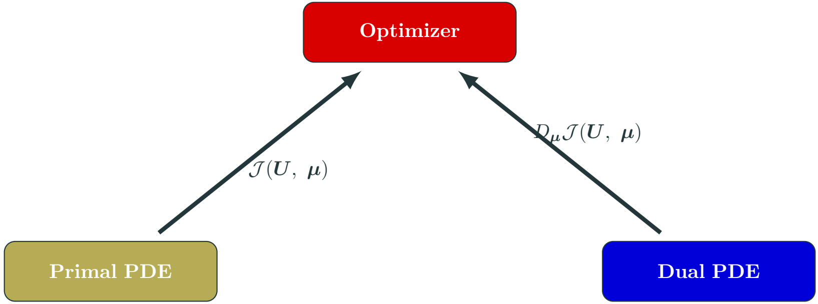
Nested approach to PDE-constrained optimization



Nested approach to PDE-constrained optimization

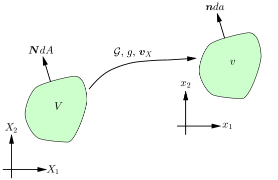


Nested approach to PDE-constrained optimization



Arbitrary Lagrangian-Eulerian formulation: Map, $\mathcal{G}(\cdot, \boldsymbol{\mu}, t)$, from physical $v(\boldsymbol{\mu}, t)$ to reference V

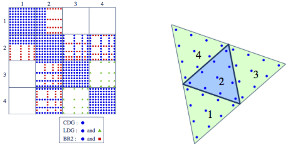
$$\left.\frac{\partial \boldsymbol{U}_{\boldsymbol{X}}}{\partial t}\right|_{\boldsymbol{X}} + \nabla_{\boldsymbol{X}} \cdot \boldsymbol{F}_{\boldsymbol{X}}(\boldsymbol{U}_{\boldsymbol{X}},\ \nabla_{\boldsymbol{X}} \boldsymbol{U}_{\boldsymbol{X}}) = 0$$



Mapping-Based ALE

Space discretization: discontinuous Galerkin

$$\boldsymbol{M}\frac{\partial \boldsymbol{u}}{\partial t} = \boldsymbol{r}(\boldsymbol{u}, \boldsymbol{\mu}, t)$$



DG Discretization

Time discretization: diagonally implicit RK

$$\boldsymbol{u}_n = \boldsymbol{u}_{n-1} + \sum_{i=1}^{s} b_i \boldsymbol{k}_{n,i}$$

$$\boldsymbol{M}\boldsymbol{k}_{n,i} = \Delta t_n \boldsymbol{r}\left(\boldsymbol{u}_{n,i},\ \boldsymbol{\mu},\ t_{n,i}\right)$$

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

Butcher Tableau for DIRK

Quantity of interest: solver-consistency

$$F(\boldsymbol{u}_0, \ldots, \boldsymbol{u}_{N_t}, \boldsymbol{k}_{1,1}, \ldots, \boldsymbol{k}_{N_t,s}, \boldsymbol{\mu})$$

Adjoint method to efficiently compute gradients of QoI

Fully discrete output function i.e., either **objective** or a **constraint**

$$F(\boldsymbol{\mu}) = F(\boldsymbol{u}_0, \ldots, \boldsymbol{u}_n, \boldsymbol{k}_{1,1}, \ldots, \boldsymbol{k}_{N_t,s}, \boldsymbol{\mu})$$

Adjoint method to efficiently compute gradients of QoI

Fully discrete output function i.e., either **objective** or a **constraint**

$$F(\boldsymbol{\mu}) = F(\boldsymbol{u}_0, \ldots, \boldsymbol{u}_n, \boldsymbol{k}_{1,1}, \ldots, \boldsymbol{k}_{N_t,s}, \boldsymbol{\mu})$$

Total derivative with respect to parameters $\boldsymbol{\mu}$

$$DF = \frac{\partial F}{\partial \boldsymbol{\mu}} + \sum_{n=0}^{N_t} \frac{\partial F}{\partial \boldsymbol{u}_n} \frac{\partial \boldsymbol{u}_n}{\partial \boldsymbol{\mu}} + \sum_{n=1}^{N_t} \sum_{i=1}^{s} \frac{\partial F}{\partial \boldsymbol{k}_{n,i}} \frac{\partial \boldsymbol{k}_{n,i}}{\partial \boldsymbol{\mu}}$$

However, the sensitivities, $\frac{\partial \boldsymbol{u}_n}{\partial \boldsymbol{\mu}}$ and $\frac{\partial \boldsymbol{k}_{n,i}}{\partial \boldsymbol{\mu}}$, are expensive to compute, requiring the solution of $n_{\boldsymbol{\mu}}$ linear evolution equations

Adjoint method to efficiently compute gradients of QoI

Fully discrete output function i.e., either **objective** or a **constraint**

$$F(\boldsymbol{\mu}) = F(\boldsymbol{u}_0, \ldots, \boldsymbol{u}_n, \boldsymbol{k}_{1,1}, \ldots, \boldsymbol{k}_{N_t,s}, \boldsymbol{\mu})$$

Total derivative with respect to parameters $\boldsymbol{\mu}$

$$DF = \frac{\partial F}{\partial \boldsymbol{\mu}} + \sum_{n=0}^{N_t} \frac{\partial F}{\partial \boldsymbol{u}_n} \frac{\partial \boldsymbol{u}_n}{\partial \boldsymbol{\mu}} + \sum_{n=1}^{N_t} \sum_{i=1}^{s} \frac{\partial F}{\partial \boldsymbol{k}_{n,i}} \frac{\partial \boldsymbol{k}_{n,i}}{\partial \boldsymbol{\mu}}$$

However, the sensitivities, $\frac{\partial \boldsymbol{u}_n}{\partial \boldsymbol{\mu}}$ and $\frac{\partial \boldsymbol{k}_{n,i}}{\partial \boldsymbol{\mu}}$, are expensive to compute, requiring the solution of $n_{\boldsymbol{\mu}}$ linear evolution equations

Adjoint method

Alternative method for computing DF that does not require sensitivities

Dissection of fully discrete adjoint equations

- **Linear** evolution equations solved **backward** in time
- **Primal** state/stage, $\boldsymbol{u}_{n,i}$ required at each state/stage of dual problem
- Heavily dependent on **chosen ouput**

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

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

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

Gradient reconstruction via dual variables

$$DF = \frac{\partial F}{\partial \boldsymbol{\mu}} + \boldsymbol{\lambda}_0{}^T \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{\mu}}(\boldsymbol{\mu}) + \sum_{n=1}^{N_t} \Delta t_n \sum_{i=1}^{s} \boldsymbol{\kappa}_{n,i}{}^T \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\mu}}(\boldsymbol{u}_{n,i},\ \boldsymbol{\mu},\ t_{n,i})$$

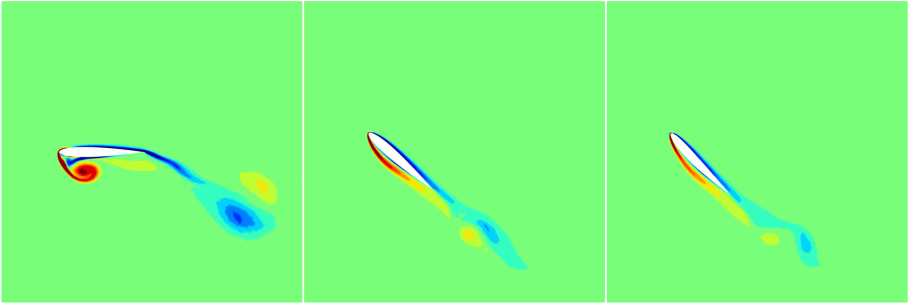
[Zahr and Persson, 2016]

Energetically optimal flapping flight

Energy $= 9.4096$
Thrust $= 0.1766$

Energy $= 4.9476$
Thrust $= 2.500$

Energy $= 4.6182$
Thrust $= 2.500$



Initial Guess

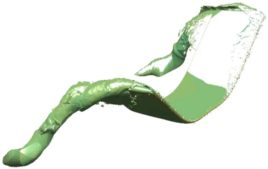
Optimal RBM
$T_x = 2.5$

Optimal RBM/TMG
$T_x = 2.5$

Energetically optimal flapping in three dimensions

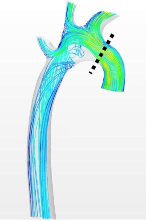
Energy = 1.4459e-01
Thrust = -1.1192e-01

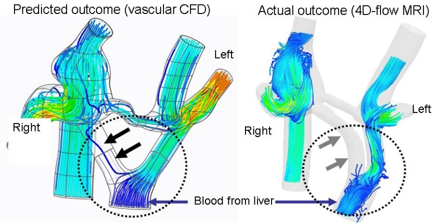
Energy = 3.1378e-01
Thrust = 0.0000e+00

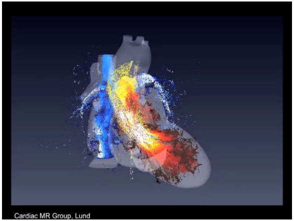


In vivo medical imaging insufficient for many applications

- Detailed *in vivo* imaging of the human body using MRI holds great potential for scientific discovery and impact in health care
- Limited by a fundamental trade-off: resolution, image quality, scan time
- Resolution: 1-3mm, 25-100ms in 10-20 minute scan
- Insufficient for many applications: involving infants, while exercising



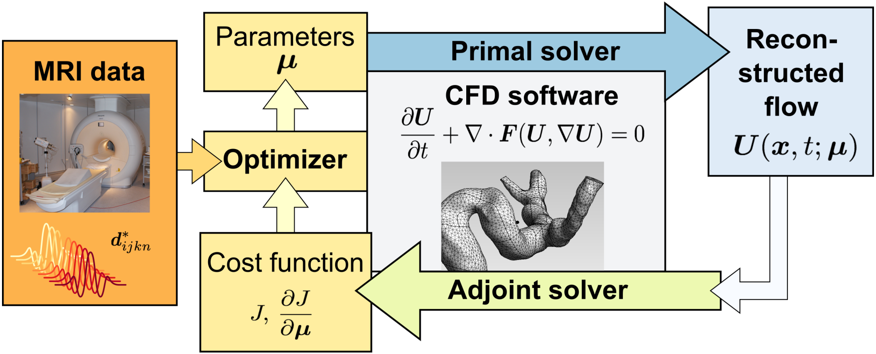




Goal: visualize *in vivo* flow with high-resolution and accurately compute clinically relevant quantities from quick scans

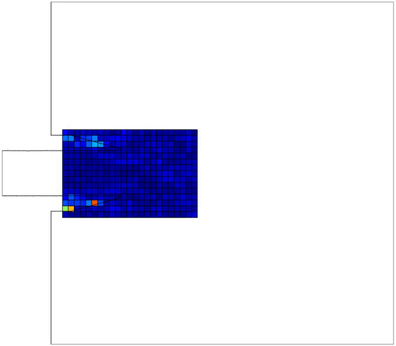
Approach: determine CFD parameters (material properties, boundary conditions) such that the simulation matches MRI data using optimization

Simulation-based imaging (SBI) workflow

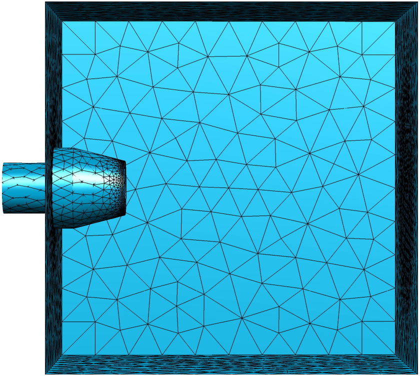


High-quality reconstruction with experimental data: pulsatile flow

CFD-based reconstruction from quick, low-resolution scan matches laser PIV measurements better than slow, high-resolution scan

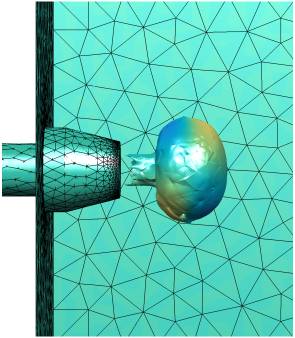


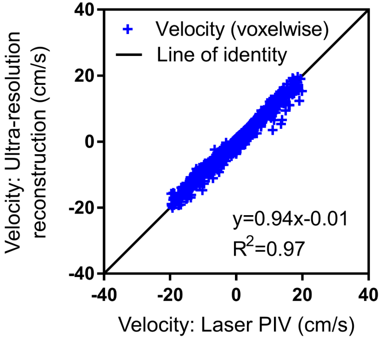
MRI data



Reconstructed flow

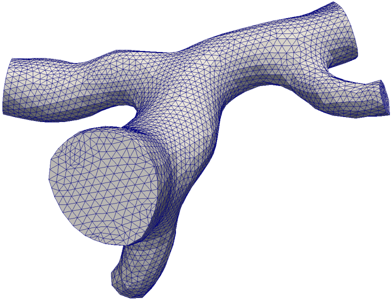
Laser PIV validation of simulation-based flow reconstruction



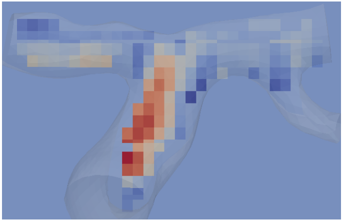


Flow visualization (*left*) and quantitative comparison with experimental data shows excellent reconstruction accuracy (*right*)

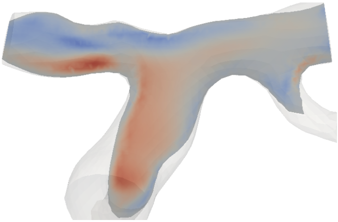
In vivo test of simulation-based flow reconstruction



Patient-specific mesh of brain vessel network (Circle of Willis)



MRI voxel velocity data on 2D spatial slice at time instance



SBI reconstruction

PDE-constrained optimization: Virtually all expense emanates from primal, dual PDE solves

$$\underset{\boldsymbol{u},\boldsymbol{\mu}}{\text{minimize}}\ \mathcal{J}(\boldsymbol{u},\boldsymbol{\mu}) \quad \text{subject to} \quad \boldsymbol{r}(\boldsymbol{u},\boldsymbol{\mu}) = 0$$

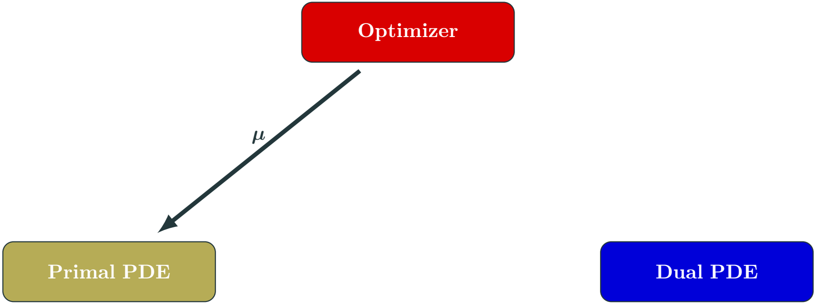
Optimizer

Primal PDE

Dual PDE

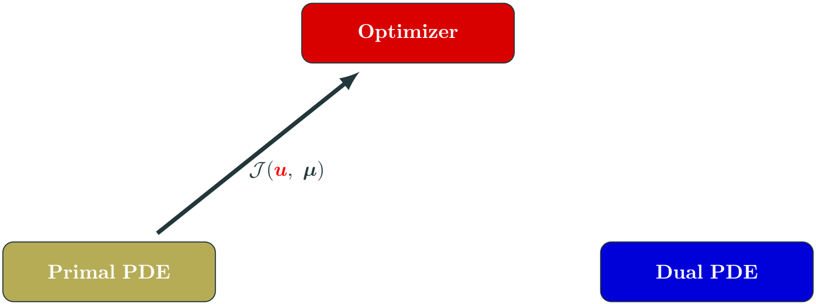
PDE-constrained optimization: Virtually all expense emanates from primal, dual PDE solves

$$\underset{\boldsymbol{u},\boldsymbol{\mu}}{\text{minimize}}\ \mathcal{J}(\boldsymbol{u},\boldsymbol{\mu}) \quad \text{subject to} \quad \boldsymbol{r}(\boldsymbol{u},\boldsymbol{\mu}) = 0$$



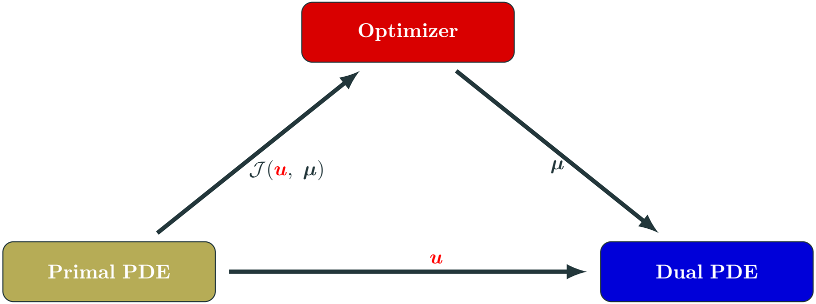
PDE-constrained optimization: Virtually all expense emanates from primal, dual PDE solves

$$\underset{\boldsymbol{u},\boldsymbol{\mu}}{\text{minimize}}\ \mathcal{J}(\boldsymbol{u},\boldsymbol{\mu}) \ \ \text{subject to} \ \ \boldsymbol{r}(\boldsymbol{u},\boldsymbol{\mu}) = 0$$



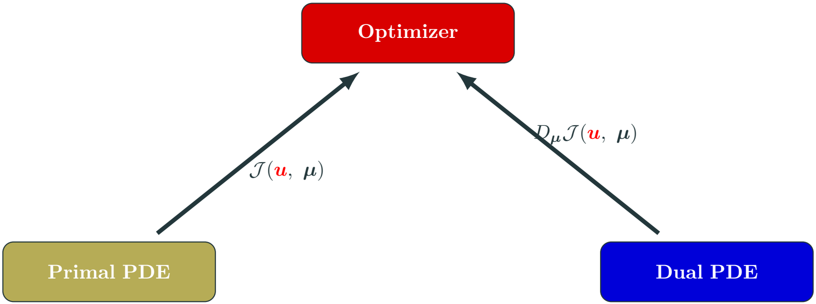
PDE-constrained optimization: Virtually all expense emanates from primal, dual PDE solves

$$\underset{\boldsymbol{u},\boldsymbol{\mu}}{\text{minimize}}\ \mathcal{J}(\boldsymbol{u},\boldsymbol{\mu})\ \ \text{subject to}\ \ \boldsymbol{r}(\boldsymbol{u},\boldsymbol{\mu}) = 0$$



PDE-constrained optimization: Virtually all expense emanates from primal, dual PDE solves

$$\underset{\boldsymbol{u},\boldsymbol{\mu}}{\text{minimize}}\ \mathcal{J}(\boldsymbol{u},\boldsymbol{\mu})\quad \text{subject to}\quad \boldsymbol{r}(\boldsymbol{u},\boldsymbol{\mu})=0$$



Proposed approach: managed inexactness

Replace expensive PDE with inexpensive approximation model

- **Reduced-order models** used for *inexact PDE evaluations*
- **Partially converged solutions** used for *inexact PDE evaluations*

$$\underset{\boldsymbol{\mu}\in\mathbb{R}^{n_{\boldsymbol{\mu}}}}{\text{minimize}} \quad F(\boldsymbol{\mu}) \qquad \longrightarrow \qquad \underset{\boldsymbol{\mu}\in\mathbb{R}^{n_{\boldsymbol{\mu}}}}{\text{minimize}} \quad m(\boldsymbol{\mu})$$

[2] Must be *computable* and apply to general, nonlinear PDEs

Proposed approach: managed inexactness

Replace expensive PDE with inexpensive approximation model

- **Reduced-order models** used for *inexact PDE evaluations*
- **Partially converged solutions** used for *inexact PDE evaluations*

$$\underset{\boldsymbol{\mu}\in\mathbb{R}^{n_{\boldsymbol{\mu}}}}{\text{minimize}} \quad F(\boldsymbol{\mu}) \qquad \longrightarrow \qquad \underset{\boldsymbol{\mu}\in\mathbb{R}^{n_{\boldsymbol{\mu}}}}{\text{minimize}} \quad m(\boldsymbol{\mu})$$

Manage inexactness with trust region method

- Embedded in globally convergent **trust region** framework
- **Error indicators**[2] to account for *all* sources of inexactness
- **Refinement** of approximation model using *greedy algorithms*

$$\underset{\boldsymbol{\mu}\in\mathbb{R}^{n_{\boldsymbol{\mu}}}}{\text{minimize}} \quad F(\boldsymbol{\mu}) \qquad \longrightarrow \qquad \begin{aligned} &\underset{\boldsymbol{\mu}\in\mathbb{R}^{n_{\boldsymbol{\mu}}}}{\text{minimize}} && m_k(\boldsymbol{\mu}) \\ &\text{subject to} && \|\boldsymbol{\mu}-\boldsymbol{\mu}_k\| \leq \Delta_k \end{aligned}$$

[2] Must be *computable* and apply to general, nonlinear PDEs

Trust region ingredients for global convergence

Approximation model

$$m_k(\boldsymbol{\mu})$$

Error indicator

$$\|\nabla F(\boldsymbol{\mu}) - \nabla m_k(\boldsymbol{\mu})\| \leq \xi \varphi_k(\boldsymbol{\mu}), \qquad \xi > 0$$

Adaptivity

$$\varphi_k(\boldsymbol{\mu}_k) \leq \kappa_\varphi \min\{\|\nabla m_k(\boldsymbol{\mu}_k)\|\,,\, \Delta_k\}$$

Global convergence

$$\boxed{\liminf_{k\to\infty} \ \|\nabla F(\boldsymbol{\mu}_k)\| = 0}$$

Trust region method with inexact gradients [Kouri et al., 2013]

1: **Model update**: Choose model m_k such that error indicator φ_k satisfies

$$\varphi_k(\boldsymbol{\mu}_k) \leq \kappa_\varphi \min\{\|\nabla m_k(\boldsymbol{\mu}_k)\|\,,\ \Delta_k\}$$

2: **Step computation**: Approximately solve the trust region subproblem

$$\hat{\boldsymbol{\mu}}_k \quad = \quad \underset{\boldsymbol{\mu}\in\mathbb{R}^{n_{\boldsymbol{\mu}}}}{\arg\min}\ m_k(\boldsymbol{\mu}) \quad \text{subject to} \quad \|\boldsymbol{\mu} - \boldsymbol{\mu}_k\| \leq \Delta_k$$

3: **Step acceptance**: Compute actual-to-predicted reduction

$$\rho_k = \frac{F(\boldsymbol{\mu}_k) - F(\hat{\boldsymbol{\mu}}_k)}{m_k(\boldsymbol{\mu}_k) - m_k(\hat{\boldsymbol{\mu}}_k)}$$

$$\textbf{if} \quad \rho_k \geq \eta_1 \quad \textbf{then} \quad \boldsymbol{\mu}_{k+1} = \hat{\boldsymbol{\mu}}_k \quad \textbf{else} \quad \boldsymbol{\mu}_{k+1} = \boldsymbol{\mu}_k \quad \textbf{end if}$$

4: **Trust region update**:

$$\begin{array}{llll}
\textbf{if} & \rho_k \leq \eta_1 & \textbf{then} & \Delta_{k+1} \in (0, \gamma\,\|\hat{\boldsymbol{\mu}}_k - \boldsymbol{\mu}_k\|] & \textbf{end if}\\
\textbf{if} & \rho_k \in (\eta_1, \eta_2) & \textbf{then} & \Delta_{k+1} \in [\gamma\,\|\hat{\boldsymbol{\mu}}_k - \boldsymbol{\mu}_k\|\,, \Delta_k] & \textbf{end if}\\
\textbf{if} & \rho_k \geq \eta_2 & \textbf{then} & \Delta_{k+1} \in [\Delta_k, \Delta_{\max}] & \textbf{end if}
\end{array}$$

Source of inexactness/efficiency: projection-based model reduction

- Model reduction ansatz: *state vector lies in low-dimensional subspace*

$$\boldsymbol{u} \approx \boldsymbol{\Phi}\boldsymbol{u}_r$$

  - $\boldsymbol{\Phi} = \begin{bmatrix} \boldsymbol{\phi}^1 & \cdots & \boldsymbol{\phi}^{k_{\boldsymbol{u}}} \end{bmatrix} \in \mathbb{R}^{n_{\boldsymbol{u}} \times k_{\boldsymbol{u}}}$ is the reduced (trial) basis ($n_{\boldsymbol{u}} \gg k_{\boldsymbol{u}}$)
  - $\boldsymbol{u}_r \in \mathbb{R}^{k_{\boldsymbol{u}}}$ are the reduced coordinates of $\boldsymbol{u}$

- Substitute into $\boldsymbol{r}(\boldsymbol{u}, \boldsymbol{\mu}) = 0$ and project onto columnspace of a test basis $\boldsymbol{\Phi} \in \mathbb{R}^{n_{\boldsymbol{u}} \times k_{\boldsymbol{u}}}$ to obtain a square system

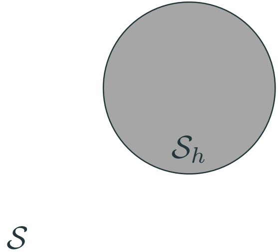
$$\boldsymbol{\Phi}^T \boldsymbol{r}(\boldsymbol{\Phi}\boldsymbol{u}_r, \boldsymbol{\mu}) = 0$$

Connection to finite element method: hierarchical subspaces

$$\mathcal{S}$$

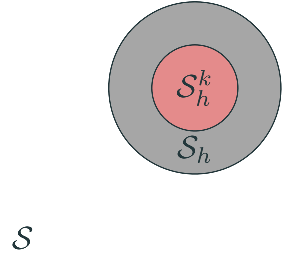
- $\mathcal{S}$ - infinite-dimensional trial space

Connection to finite element method: hierarchical subspaces



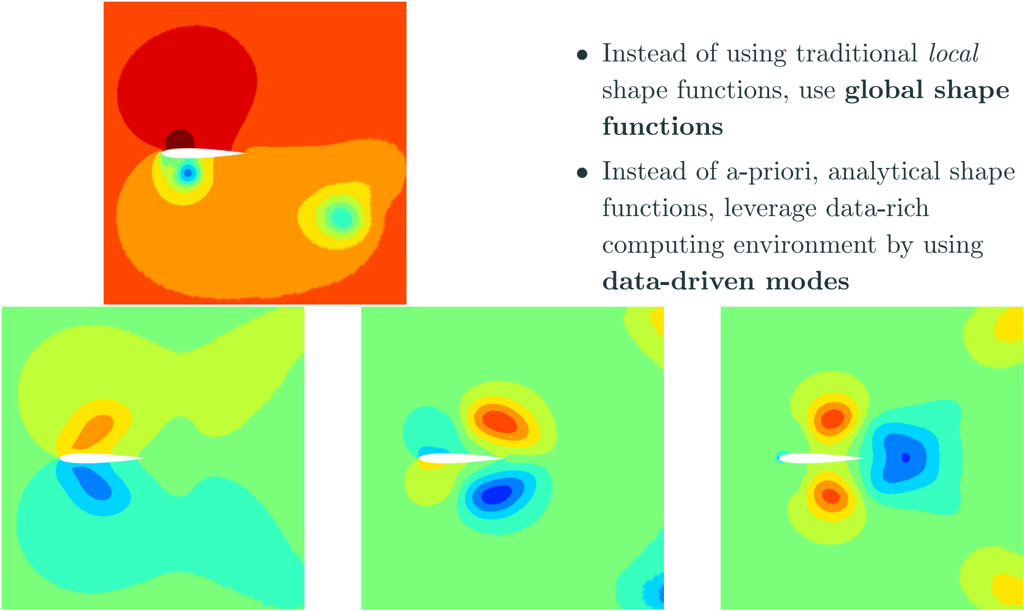
- $\mathcal{S}$ - infinite-dimensional trial space
- $\mathcal{S}_h$ - (large) finite-dimensional trial space

Connection to finite element method: hierarchical subspaces



- $\mathcal{S}$ - infinite-dimensional trial space
- $\mathcal{S}_h$ - (large) finite-dimensional trial space
- $\mathcal{S}_h^k$ - (small) finite-dimensional trial space
- $\mathcal{S}_h^k \subset \mathcal{S}_h \subset \mathcal{S}$

Few global, data-driven basis functions v. many local ones



- Instead of using traditional *local* shape functions, use **global shape functions**
- Instead of a-priori, analytical shape functions, leverage data-rich computing environment by using **data-driven modes**

Trust region method: ROM approximation model

Approximation models based on reduced-order models

$$m_k(\boldsymbol{\mu}) = \mathcal{J}(\boldsymbol{\Phi}_k \boldsymbol{u}_r(\boldsymbol{\mu}),\ \boldsymbol{\mu})$$

Error indicators from residual-based error bounds

$$\varphi_k(\boldsymbol{\mu}) = \left\|\boldsymbol{r}(\boldsymbol{\Phi}_k \boldsymbol{u}_r(\boldsymbol{\mu}),\ \boldsymbol{\mu})\right\|_{\boldsymbol{\Theta}} + \left\|\boldsymbol{r}^{\boldsymbol{\lambda}}(\boldsymbol{\Phi}_k \boldsymbol{u}_r(\boldsymbol{\mu}),\ \boldsymbol{\Phi}_k \boldsymbol{\lambda}_r(\boldsymbol{\mu}),\ \boldsymbol{\mu})\right\|_{\boldsymbol{\Theta}^{\boldsymbol{\lambda}}}$$

Adaptivity to refine basis at trust region center

$$\boldsymbol{\Phi}_k = \begin{bmatrix} \boldsymbol{u}(\boldsymbol{\mu}_k) & \boldsymbol{\lambda}(\boldsymbol{\mu}_k) & \texttt{POD}(\boldsymbol{U}_k) & \texttt{POD}(\boldsymbol{V}_k) \end{bmatrix}$$
$$\boldsymbol{U}_k = \begin{bmatrix} \boldsymbol{u}(\boldsymbol{\mu}_0) & \cdots & \boldsymbol{u}(\boldsymbol{\mu}_{k-1}) \end{bmatrix} \qquad \boldsymbol{V}_k = \begin{bmatrix} \boldsymbol{\lambda}(\boldsymbol{\mu}_0) & \cdots & \boldsymbol{\lambda}(\boldsymbol{\mu}_{k-1}) \end{bmatrix}$$

Interpolation property of minimum-residual reduced-order models $\implies \varphi_k(\boldsymbol{\mu}_k) = 0$

Trust region method: ROM approximation model

Approximation models based on reduced-order models

$$m_k(\boldsymbol{\mu}) = \mathcal{J}(\boldsymbol{\Phi}_k \boldsymbol{u}_r(\boldsymbol{\mu}),\, \boldsymbol{\mu})$$

Error indicators from residual-based error bounds

$$\varphi_k(\boldsymbol{\mu}) = \|\boldsymbol{r}(\boldsymbol{\Phi}_k \boldsymbol{u}_r(\boldsymbol{\mu}),\, \boldsymbol{\mu})\|_{\boldsymbol{\Theta}} + \left\|\boldsymbol{r}^{\boldsymbol{\lambda}}(\boldsymbol{\Phi}_k \boldsymbol{u}_r(\boldsymbol{\mu}),\, \boldsymbol{\Phi}_k \boldsymbol{\lambda}_r(\boldsymbol{\mu}),\, \boldsymbol{\mu})\right\|_{\boldsymbol{\Theta}^{\boldsymbol{\lambda}}}$$

Adaptivity to refine basis at trust region center

$$\boldsymbol{\Phi}_k = \begin{bmatrix} \boldsymbol{u}(\boldsymbol{\mu}_k) & \boldsymbol{\lambda}(\boldsymbol{\mu}_k) & \texttt{POD}(\boldsymbol{U}_k) & \texttt{POD}(\boldsymbol{V}_k) \end{bmatrix}$$

$$\boldsymbol{U}_k = \begin{bmatrix} \boldsymbol{u}(\boldsymbol{\mu}_0) & \cdots & \boldsymbol{u}(\boldsymbol{\mu}_{k-1}) \end{bmatrix} \qquad \boldsymbol{V}_k = \begin{bmatrix} \boldsymbol{\lambda}(\boldsymbol{\mu}_0) & \cdots & \boldsymbol{\lambda}(\boldsymbol{\mu}_{k-1}) \end{bmatrix}$$

Interpolation property of minimum-residual reduced-order models $\implies \varphi_k(\boldsymbol{\mu}_k) = 0$

$$\boxed{\liminf_{k\to\infty} \ \|\nabla \mathcal{J}(\boldsymbol{u}(\boldsymbol{\mu}_k),\, \boldsymbol{\mu}_k)\| = 0}$$

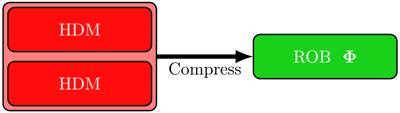
Trust region framework for optimization with ROMs

Schematic

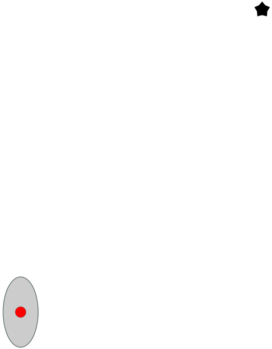
$\boldsymbol{\mu}$-space

Breakdown of Computational Effort

Trust region framework for optimization with ROMs



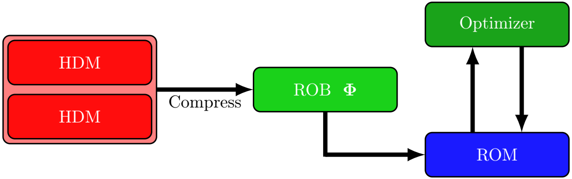
Schematic



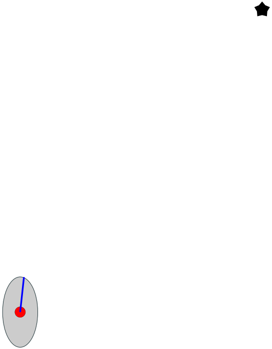
$\boldsymbol{\mu}$-space

Breakdown of Computational Effort

Trust region framework for optimization with ROMs



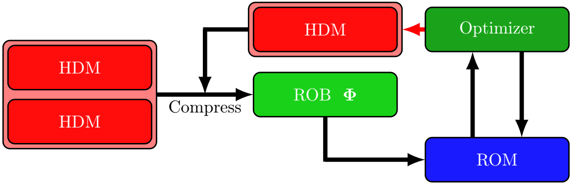
Schematic



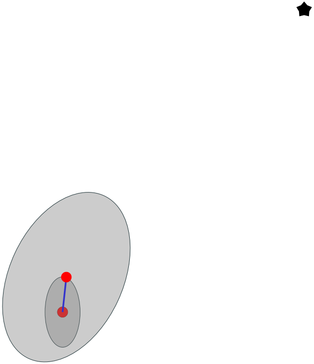
$\boldsymbol{\mu}$-space

Breakdown of Computational Effort

Trust region framework for optimization with ROMs



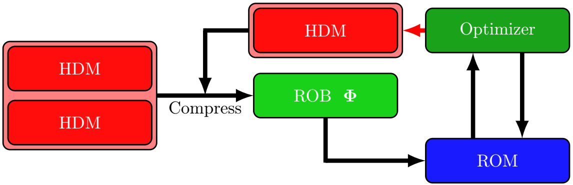
Schematic



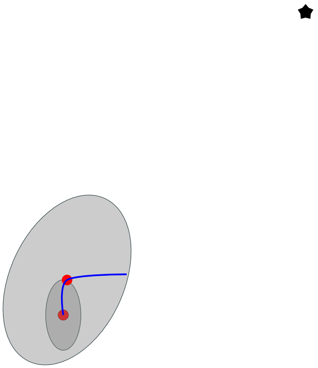
$\boldsymbol{\mu}$-space

Breakdown of Computational Effort

Trust region framework for optimization with ROMs



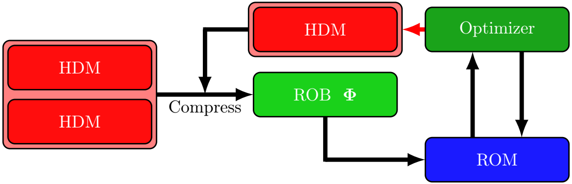
Schematic



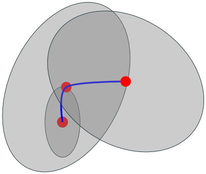
$\boldsymbol{\mu}$-space

Breakdown of Computational Effort

Trust region framework for optimization with ROMs



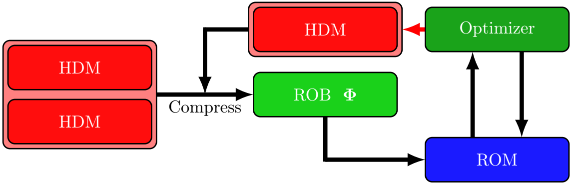
Schematic



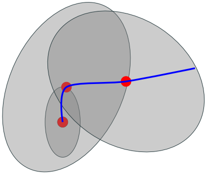
$\boldsymbol{\mu}$-space

Breakdown of Computational Effort

Trust region framework for optimization with ROMs



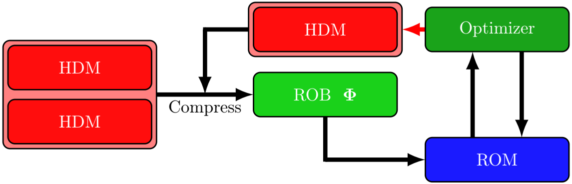
Schematic



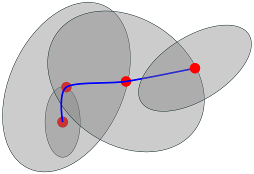
$\boldsymbol{\mu}$-space

Breakdown of Computational Effort

Trust region framework for optimization with ROMs



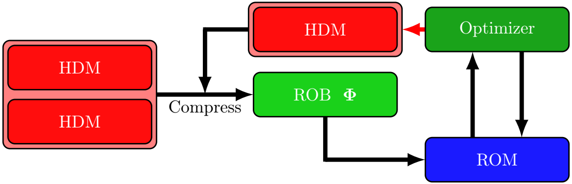
Schematic



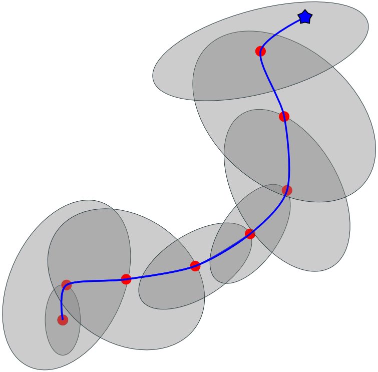
μ-space

Breakdown of Computational Effort

Trust region framework for optimization with ROMs



Schematic

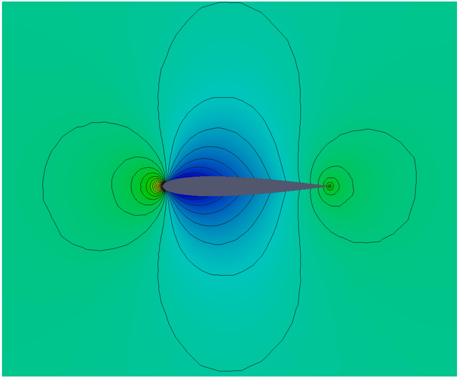


$\boldsymbol{\mu}$-space

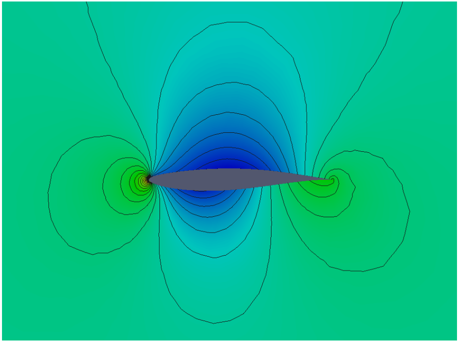
Breakdown of Computational Effort

Compressible, inviscid airfoil design

Pressure discrepancy minimization (Euler equations)



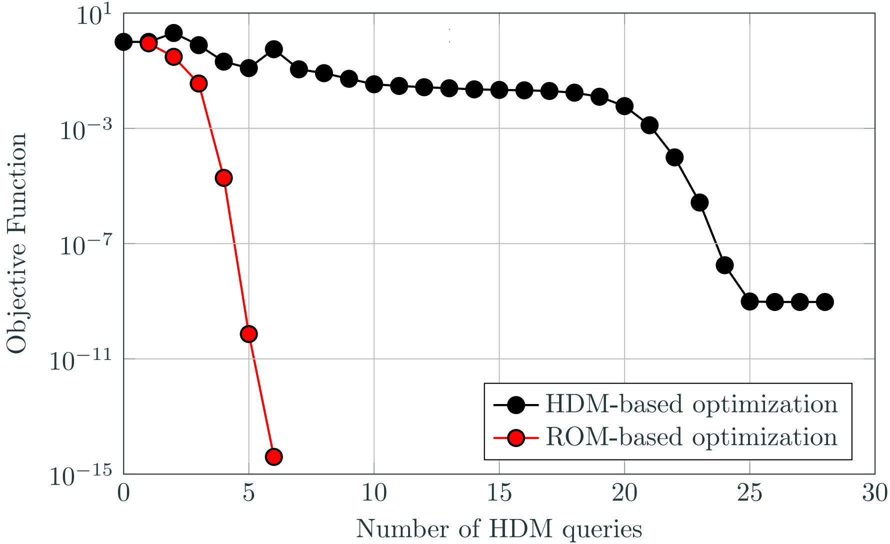
NACA0012: Initial



RAE2822: Target

Pressure field for airfoil configurations at $M_\infty = 0.5$, $\alpha = 0.0^\circ$

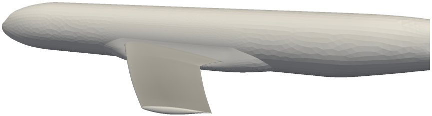
Proposed method: $4\times$ fewer HDM queries



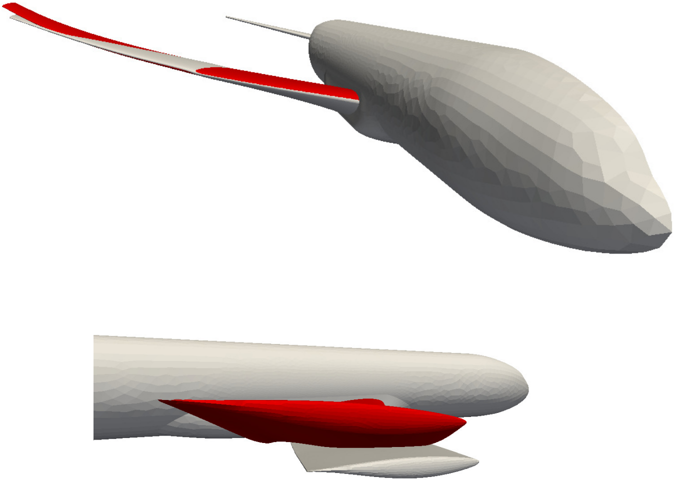
Shape optimization of aircraft in turbulent flow

- **Flow**: $M = 0.85 \quad \alpha = 2.32^\circ \quad Re = 5 \times 10^6$
- **Equations**: RANS with Spalart-Allmaras
- **Solver**: Vertex-centered finite volume method
- **Mesh**: **11.5M** nodes, **68M** tetra, **69M** DOF
- **Shape**: 4 parameters (length, sweep, dihedral, twist)

$$\begin{aligned} \underset{\boldsymbol{\mu} \in \mathbb{R}^4}{\text{minimize}} \quad & -L_z(\boldsymbol{\mu})/L_x(\boldsymbol{\mu}) \\ \text{subject to} \quad & L_z(\boldsymbol{\mu}) = \bar{L}_z \end{aligned}$$

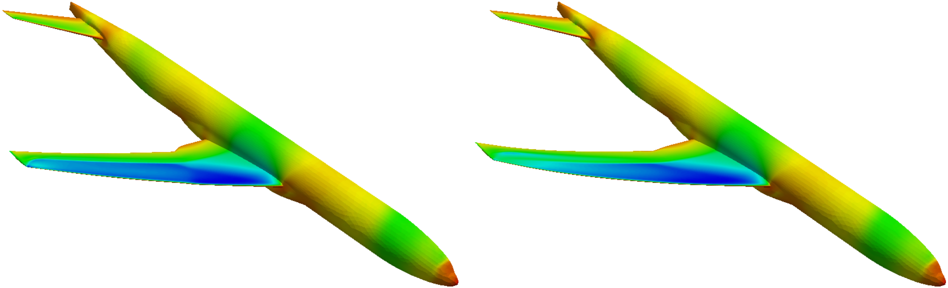


Optimized shape: reduction in 2.2 drag counts



Baseline (gray) and optimized shape (red) – 2× magnification

Optimized shape: reduction in 2.2 drag counts

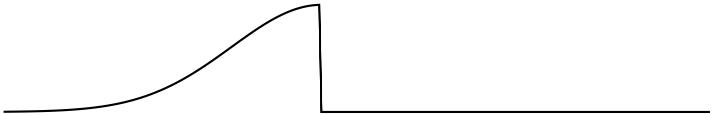


Baseline (left) and optimized (right) shape – colored by C_p

Performance: ROM-TR method obtains same solution (to 4 digits of accuracy) as HDM-only optimization and only requires about 60% of the computation time.

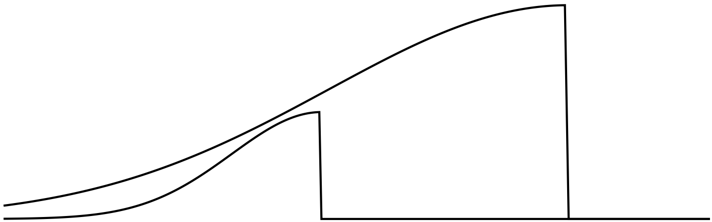
Conclusion: Very promising results considering ROMs have notoriously poor prediction capabilities for problems with moving shocks/discontinuities.

Reduction of conservation laws with parametrized discontinuities



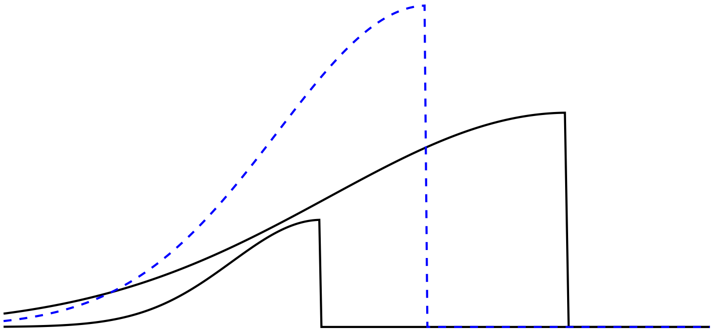
Fundamental issue: linear subspace approximation ill-suited for advection-dominated features (slowly decay Kolmogorov n-width)

Reduction of conservation laws with parametrized discontinuities



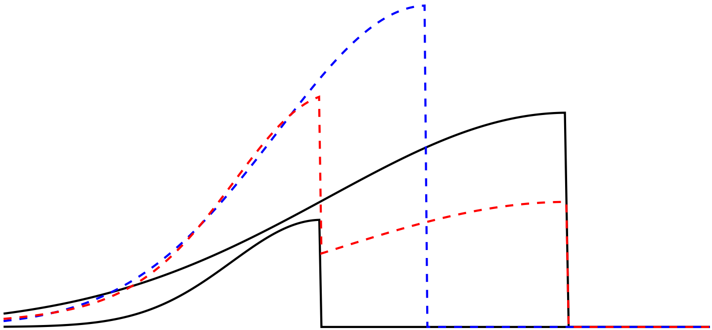
Fundamental issue: linear subspace approximation ill-suited for advection-dominated features (slowly decay Kolmogorov n-width)

Reduction of conservation laws with parametrized discontinuities



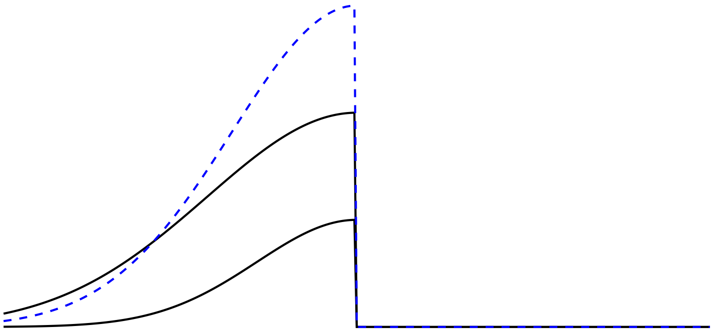
Fundamental issue: linear subspace approximation ill-suited for advection-dominated features (slowly decay Kolmogorov n-width)

Reduction of conservation laws with parametrized discontinuities



Fundamental issue: linear subspace approximation ill-suited for advection-dominated features (slowly decay Kolmogorov n-width)

Reduction of conservation laws with parametrized discontinuities

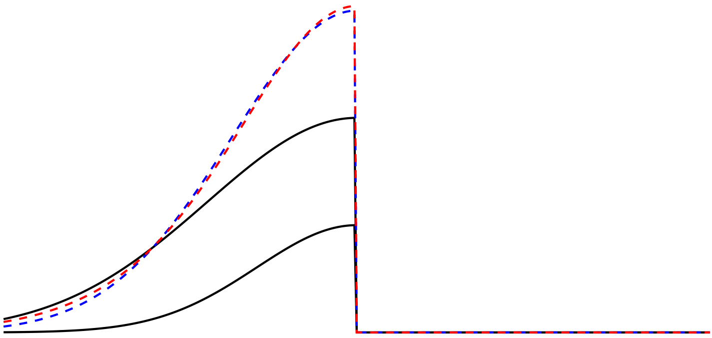


Fundamental issue: linear subspace approximation ill-suited for advection-dominated features (slowly decay Kolmogorov n-width)

Proposed solution:

- apply parameter-dependent domain mapping to align features

Reduction of conservation laws with parametrized discontinuities

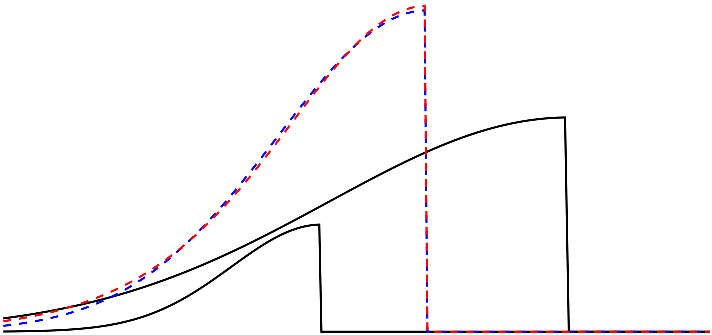


Fundamental issue: linear subspace approximation ill-suited for advection-dominated features (slowly decay Kolmogorov n-width)

Proposed solution:

- apply parameter-dependent domain mapping to align features
- use linear subspace in reference domain to reduce dimension

Reduction of conservation laws with parametrized discontinuities



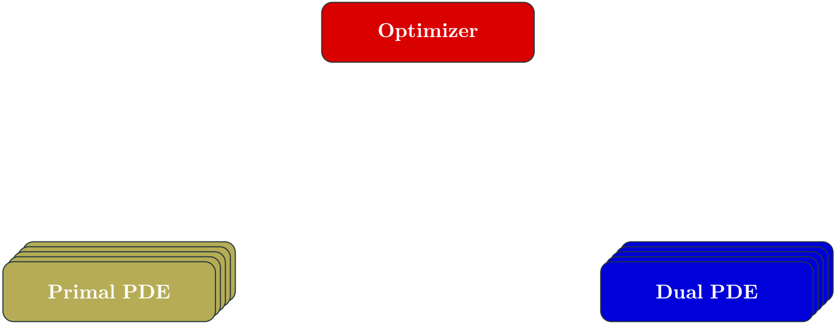
Fundamental issue: linear subspace approximation ill-suited for advection-dominated features (slowly decay Kolmogorov n-width)

Proposed solution:

- apply parameter-dependent domain mapping to align features
- use linear subspace in reference domain to reduce dimension
- push forward to physical domain

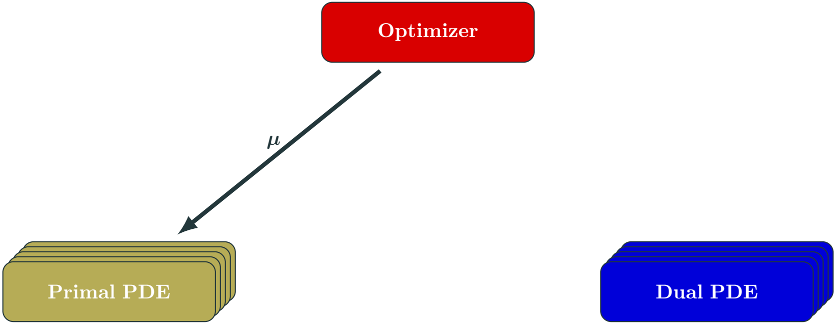
PDE-constrained optimization under uncertainty: Ensemble of primal, dual PDE solves required at every optimization iteration

$$\underset{\boldsymbol{u},\boldsymbol{\mu}}{\text{minimize}}\ \mathbb{E}\left[\mathcal{J}(\boldsymbol{u},\boldsymbol{\mu},\cdot)\right] \quad \text{subject to} \quad \boldsymbol{r}(\boldsymbol{u},\boldsymbol{\mu},\boldsymbol{\xi}) = 0, \quad \forall \boldsymbol{\xi}$$



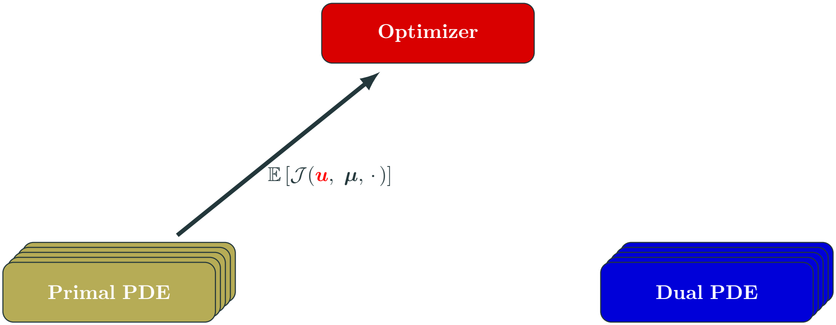
PDE-constrained optimization under uncertainty: Ensemble of primal, dual PDE solves required at every optimization iteration

$$\underset{\boldsymbol{u},\boldsymbol{\mu}}{\text{minimize}}\ \mathbb{E}\left[\mathcal{J}(\boldsymbol{u},\boldsymbol{\mu},\cdot)\right] \quad \text{subject to} \quad \boldsymbol{r}(\boldsymbol{u},\boldsymbol{\mu},\boldsymbol{\xi}) = 0, \quad \forall \boldsymbol{\xi}$$



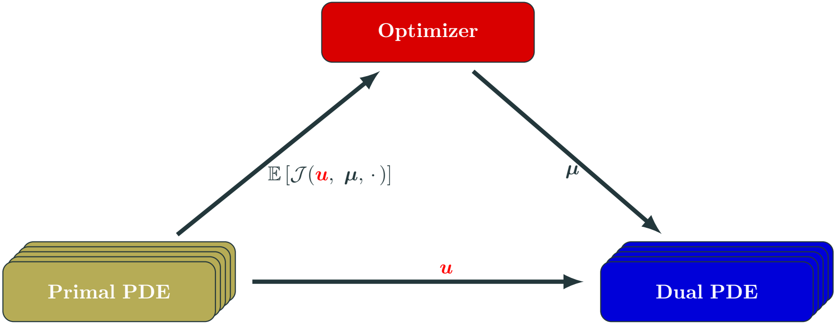
PDE-constrained optimization under uncertainty: Ensemble of primal, dual PDE solves required at every optimization iteration

$$\underset{\boldsymbol{u},\boldsymbol{\mu}}{\text{minimize}}\ \mathbb{E}\left[\mathcal{J}(\boldsymbol{u},\boldsymbol{\mu},\cdot)\right] \quad \text{subject to} \quad \boldsymbol{r}(\boldsymbol{u},\boldsymbol{\mu},\boldsymbol{\xi}) = 0, \quad \forall \boldsymbol{\xi}$$



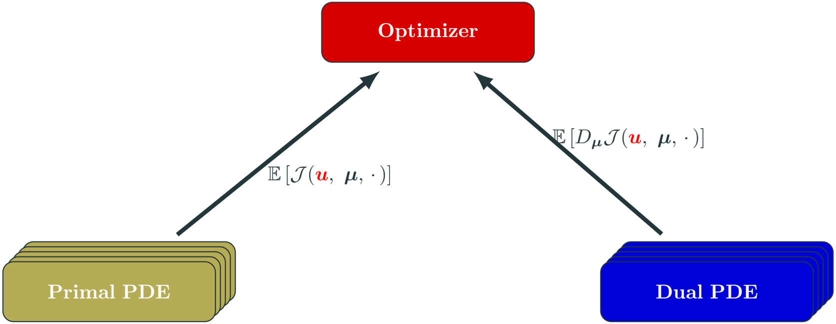
PDE-constrained optimization under uncertainty: Ensemble of primal, dual PDE solves required at every optimization iteration

$$\underset{\boldsymbol{u},\boldsymbol{\mu}}{\text{minimize}}\ \mathbb{E}\left[\mathcal{J}(\boldsymbol{u},\boldsymbol{\mu},\cdot)\right] \quad \text{subject to} \quad \boldsymbol{r}(\boldsymbol{u},\boldsymbol{\mu},\boldsymbol{\xi}) = 0, \quad \forall \boldsymbol{\xi}$$



PDE-constrained optimization under uncertainty: Ensemble of primal, dual PDE solves required at every optimization iteration

$$\underset{\boldsymbol{u},\boldsymbol{\mu}}{\text{minimize}}\ \mathbb{E}\left[\mathcal{J}(\boldsymbol{u},\boldsymbol{\mu},\cdot)\right] \quad \text{subject to} \quad \boldsymbol{r}(\boldsymbol{u},\boldsymbol{\mu},\boldsymbol{\xi}) = 0, \quad \forall \boldsymbol{\xi}$$



Proposed approach: managed inexactness

Replace expensive PDE with inexpensive approximation model

- **Anisotropic sparse grids** used for *inexact integration* of risk measures
- **Reduced-order models** used for *inexact PDE evaluations*

$$\underset{\boldsymbol{\mu}\in\mathbb{R}^{n_{\boldsymbol{\mu}}}}{\text{minimize}} \quad F(\boldsymbol{\mu}) \qquad \longrightarrow \qquad \underset{\boldsymbol{\mu}\in\mathbb{R}^{n_{\boldsymbol{\mu}}}}{\text{minimize}} \quad m(\boldsymbol{\mu})$$

Proposed approach: managed inexactness

Replace expensive PDE with inexpensive approximation model

- **Anisotropic sparse grids** used for *inexact integration* of risk measures
- **Reduced-order models** used for *inexact PDE evaluations*

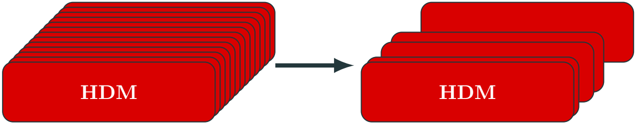
$$\underset{\boldsymbol{\mu}\in\mathbb{R}^{n_{\boldsymbol{\mu}}}}{\text{minimize}} \quad F(\boldsymbol{\mu}) \qquad \longrightarrow \qquad \underset{\boldsymbol{\mu}\in\mathbb{R}^{n_{\boldsymbol{\mu}}}}{\text{minimize}} \quad m(\boldsymbol{\mu})$$

Proposed approach: managed inexactness

Replace expensive PDE with inexpensive approximation model

- **Anisotropic sparse grids** used for *inexact integration* of risk measures
- **Reduced-order models** used for *inexact PDE evaluations*

$$\underset{\boldsymbol{\mu}\in\mathbb{R}^{n_{\boldsymbol{\mu}}}}{\text{minimize}} \quad F(\boldsymbol{\mu}) \qquad \longrightarrow \qquad \underset{\boldsymbol{\mu}\in\mathbb{R}^{n_{\boldsymbol{\mu}}}}{\text{minimize}} \quad m(\boldsymbol{\mu})$$



Proposed approach: managed inexactness

Replace expensive PDE with inexpensive approximation model

- **Anisotropic sparse grids** used for *inexact integration* of risk measures
- **Reduced-order models** used for *inexact PDE evaluations*

$$\underset{\boldsymbol{\mu}\in\mathbb{R}^{n_{\boldsymbol{\mu}}}}{\text{minimize}} \quad F(\boldsymbol{\mu}) \qquad \longrightarrow \qquad \underset{\boldsymbol{\mu}\in\mathbb{R}^{n_{\boldsymbol{\mu}}}}{\text{minimize}} \quad m(\boldsymbol{\mu})$$

First source of inexactness: anisotropic sparse grids

Stochastic collocation using anisotropic sparse grid nodes to approximate integral with summation

$$\begin{aligned} \underset{\boldsymbol{u}\in\mathbb{R}^{n_{\boldsymbol{u}}},\ \boldsymbol{\mu}\in\mathbb{R}^{n_{\boldsymbol{\mu}}}}{\text{minimize}} \quad & \mathbb{E}[\mathcal{J}(\boldsymbol{u},\, \boldsymbol{\mu},\, \cdot\,)] \\ \text{subject to} \quad & \boldsymbol{r}(\boldsymbol{u},\, \boldsymbol{\mu},\, \boldsymbol{\xi}) = 0 \quad \forall \boldsymbol{\xi} \in \boldsymbol{\Xi} \end{aligned}$$

$$\Downarrow$$

$$\begin{aligned} \underset{\boldsymbol{u}\in\mathbb{R}^{n_{\boldsymbol{u}}},\ \boldsymbol{\mu}\in\mathbb{R}^{n_{\boldsymbol{\mu}}}}{\text{minimize}} \quad & \mathbb{E}_{\mathcal{I}}[\mathcal{J}(\boldsymbol{u},\, \boldsymbol{\mu},\, \cdot\,)] \\ \text{subject to} \quad & \boldsymbol{r}(\boldsymbol{u},\, \boldsymbol{\mu},\, \boldsymbol{\xi}) = 0 \quad \forall \boldsymbol{\xi} \in \boldsymbol{\Xi}_{\mathcal{I}} \end{aligned}$$

[Kouri et al., 2013, Kouri et al., 2014]

Two sources of inexactness

Stochastic collocation of the reduced-order model over anisotropic sparse grid nodes used to approximate integral with cheap summation

$$\begin{aligned} \underset{\boldsymbol{u}\in\mathbb{R}^{n_{\boldsymbol{u}}},\ \boldsymbol{\mu}\in\mathbb{R}^{n_{\boldsymbol{\mu}}}}{\text{minimize}} \quad & \mathbb{E}[\mathcal{J}(\boldsymbol{u},\, \boldsymbol{\mu},\, \cdot\,)] \\ \text{subject to} \quad & \boldsymbol{r}(\boldsymbol{u},\, \boldsymbol{\mu},\, \boldsymbol{\xi}) = 0 \quad \forall \boldsymbol{\xi} \in \boldsymbol{\Xi} \end{aligned}$$

$$\Downarrow$$

$$\begin{aligned} \underset{\boldsymbol{u}\in\mathbb{R}^{n_{\boldsymbol{u}}},\ \boldsymbol{\mu}\in\mathbb{R}^{n_{\boldsymbol{\mu}}}}{\text{minimize}} \quad & \mathbb{E}_{\mathcal{I}}[\mathcal{J}(\boldsymbol{u},\, \boldsymbol{\mu},\, \cdot\,)] \\ \text{subject to} \quad & \boldsymbol{r}(\boldsymbol{u},\, \boldsymbol{\mu},\, \boldsymbol{\xi}) = 0 \quad \forall \boldsymbol{\xi} \in \boldsymbol{\Xi}_{\mathcal{I}} \end{aligned}$$

$$\Downarrow$$

$$\begin{aligned} \underset{\boldsymbol{u}_r\in\mathbb{R}^{k_{\boldsymbol{u}}},\ \boldsymbol{\mu}\in\mathbb{R}^{n_{\boldsymbol{\mu}}}}{\text{minimize}} \quad & \mathbb{E}_{\mathcal{I}}[\mathcal{J}(\boldsymbol{\Phi}\boldsymbol{u}_r,\, \boldsymbol{\mu},\, \cdot\,)] \\ \text{subject to} \quad & \boldsymbol{\Phi}^T\boldsymbol{r}(\boldsymbol{\Phi}\boldsymbol{u}_r,\, \boldsymbol{\mu},\, \boldsymbol{\xi}) = 0 \quad \forall \boldsymbol{\xi} \in \boldsymbol{\Xi}_{\mathcal{I}} \end{aligned}$$

- **Optimization problem:**

$$\underset{\boldsymbol{\mu} \in \mathbb{R}^{n_{\boldsymbol{\mu}}}}{\text{minimize}} \quad \int_{\boldsymbol{\Xi}} \rho(\boldsymbol{\xi}) \left[\int_0^1 \frac{1}{2}(u(\boldsymbol{\mu}, \boldsymbol{\xi}, x) - \bar{u}(x))^2 \, dx + \frac{\alpha}{2} \int_0^1 z(\boldsymbol{\mu}, x)^2 \, dx \right] d\boldsymbol{\xi}$$

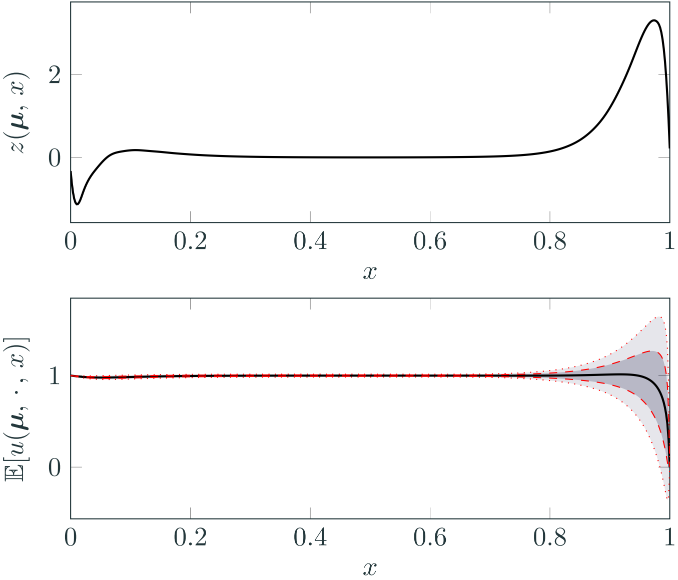
where $u(\boldsymbol{\mu}, \boldsymbol{\xi}, x)$ solves

$$-\nu(\boldsymbol{\xi}) \partial_{xx} u(\boldsymbol{\mu}, \boldsymbol{\xi}, x) + u(\boldsymbol{\mu}, \boldsymbol{\xi}, x) \partial_x u(\boldsymbol{\mu}, \boldsymbol{\xi}, x) = z(\boldsymbol{\mu}, x) \quad x \in (0, 1), \quad \boldsymbol{\xi} \in \boldsymbol{\Xi}$$
$$u(\boldsymbol{\mu}, \boldsymbol{\xi}, 0) = d_0(\boldsymbol{\xi}) \qquad u(\boldsymbol{\mu}, \boldsymbol{\xi}, 1) = d_1(\boldsymbol{\xi})$$

- **Target state:** $\bar{u}(x) \equiv 1$
- **Stochastic Space:** $\boldsymbol{\Xi} = [-1, 1]^3$, $\rho(\boldsymbol{\xi}) d\boldsymbol{\xi} = 2^{-3} d\boldsymbol{\xi}$

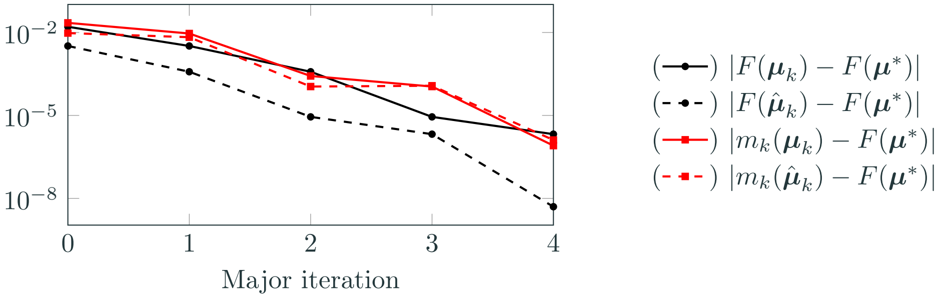
$$\nu(\boldsymbol{\xi}) = 10^{\boldsymbol{\xi}_1 - 2} \qquad d_0(\boldsymbol{\xi}) = 1 + \frac{\boldsymbol{\xi}_2}{1000} \qquad d_1(\boldsymbol{\xi}) = \frac{\boldsymbol{\xi}_3}{1000}$$

- **Parametrization:** $z(\boldsymbol{\mu}, x)$ – cubic splines with 51 knots, $n_{\boldsymbol{\mu}} = 53$



Optimal control and corresponding mean state (——) ± one (- - -) and two (·······) standard deviations

Global convergence without pointwise agreement

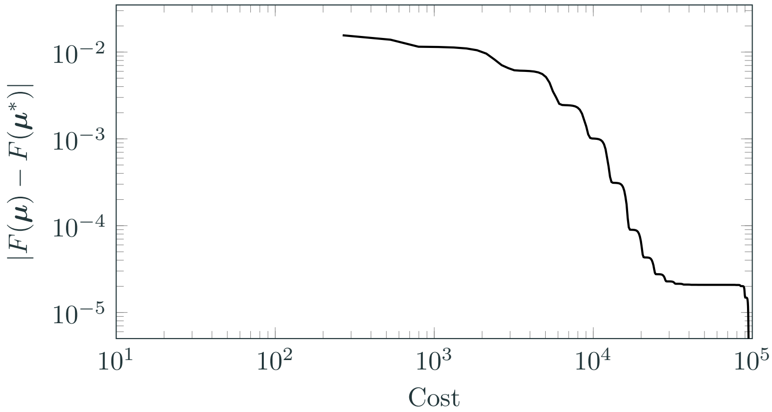


$F(\boldsymbol{\mu}_k)$	$m_k(\boldsymbol{\mu}_k)$	$F(\hat{\boldsymbol{\mu}}_k)$	$m_k(\hat{\boldsymbol{\mu}}_k)$	$\|\nabla F(\boldsymbol{\mu}_k)\|$	ρ_k	Success?
6.6506e-02	7.2694e-02	5.3655e-02	5.9922e-02	2.2959e-02	1.0257e+00	1.0000e+00
5.3655e-02	5.9593e-02	5.0783e-02	5.7152e-02	2.3424e-03	9.7512e-01	1.0000e+00
5.0783e-02	5.0670e-02	5.0412e-02	5.0292e-02	1.9724e-03	9.8351e-01	1.0000e+00
5.0412e-02	5.0292e-02	5.0405e-02	5.0284e-02	9.2654e-05	8.7479e-01	1.0000e+00
5.0405e-02	5.0404e-02	5.0403e-02	5.0401e-02	8.3139e-05	9.9946e-01	1.0000e+00
5.0403e-02	5.0401e-02	-	-	2.2846e-06	-	-

Convergence history of trust region method built on two-level approximation

Significant reduction in cost, even if (largest) ROM only $10\times$ faster than HDM

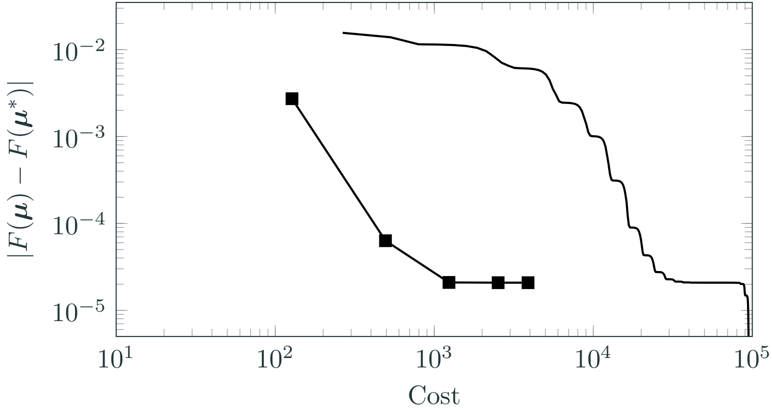
$$\text{Cost} = \text{nHdmPrim} + 0.5 \times \text{nHdmAdj} + \tau^{-1} \times (\text{nRomPrim} + 0.5 \times \text{nRomAdj})$$



5-level isotropic SG (——), dimension-adaptive SG [Kouri et al., 2014] ( ), and proposed ROM/SG for $\tau = 1$ ( ), $\tau = 10$ ( ), $\tau = 100$ ( ), $\tau = \infty$ ( )

Significant reduction in cost, even if (largest) ROM only $10\times$ faster than HDM

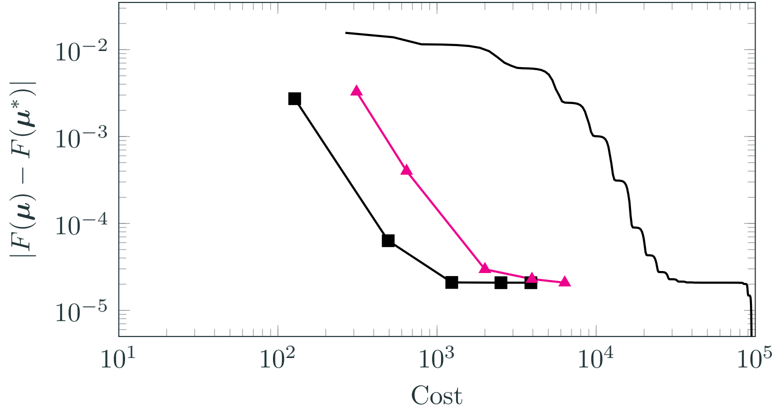
$$\text{Cost} = \text{nHdmPrim} + 0.5 \times \text{nHdmAdj} + \tau^{-1} \times (\text{nRomPrim} + 0.5 \times \text{nRomAdj})$$



5-level isotropic SG (——), dimension-adaptive SG [Kouri et al., 2014] (–■–), and proposed ROM/SG for $\tau = 1$ ( ), $\tau = 10$ ( ), $\tau = 100$ ( ), $\tau = \infty$ ( )

Significant reduction in cost, even if (largest) ROM only $10\times$ faster than HDM

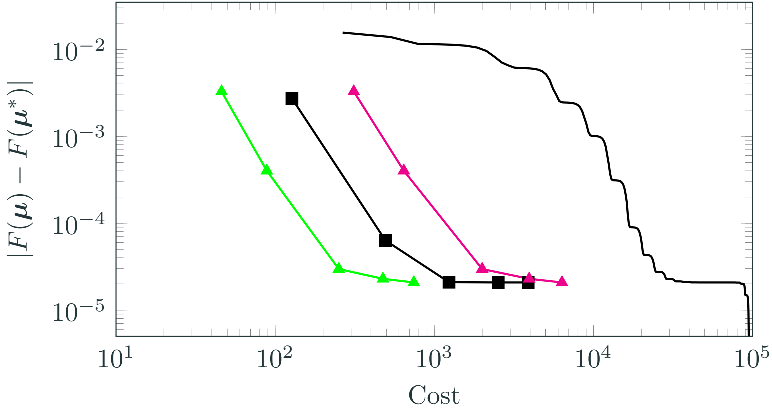
$$\text{Cost} = \text{nHdmPrim} + 0.5 \times \text{nHdmAdj} + \tau^{-1} \times (\text{nRomPrim} + 0.5 \times \text{nRomAdj})$$



5-level isotropic SG (——), dimension-adaptive SG [Kouri et al., 2014] (—■—), and proposed ROM/SG for $\tau = 1$ (—▲—), $\tau = 10$ ( ), $\tau = 100$ ( ), $\tau = \infty$ ( )

Significant reduction in cost, even if (largest) ROM only $10\times$ faster than HDM

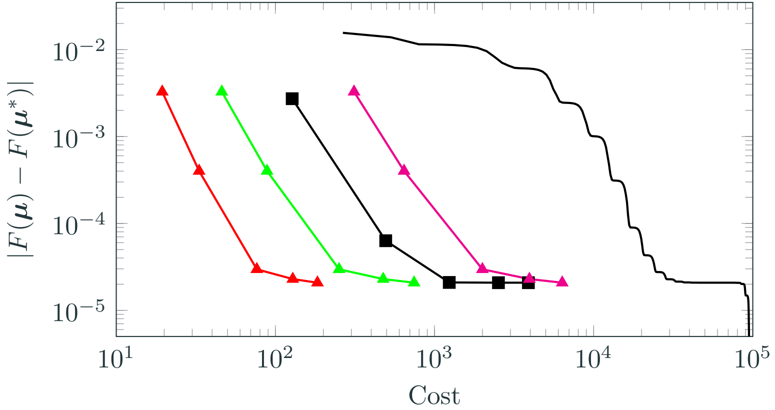
$$\text{Cost} = \text{nHdmPrim} + 0.5 \times \text{nHdmAdj} + \tau^{-1} \times (\text{nRomPrim} + 0.5 \times \text{nRomAdj})$$



5-level isotropic SG (——), dimension-adaptive SG [Kouri et al., 2014] (—■—), and proposed ROM/SG for $\tau = 1$ (—▲—), $\tau = 10$ (—▲—), $\tau = 100$ ( ), $\tau = \infty$ ( )

Significant reduction in cost, even if (largest) ROM only $10\times$ faster than HDM

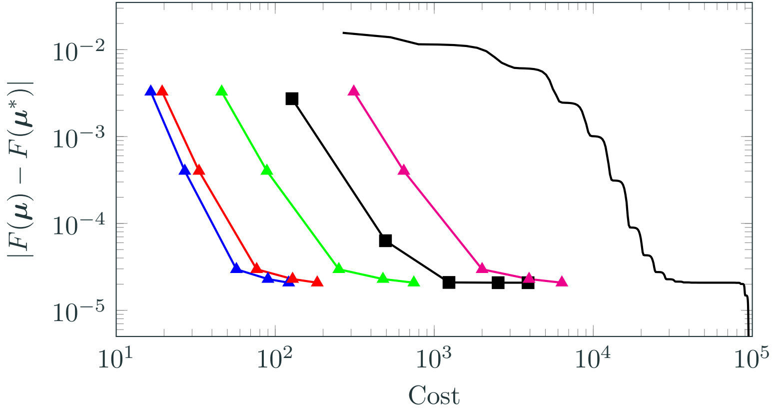
$$\text{Cost} = \text{nHdmPrim} + 0.5 \times \text{nHdmAdj} + \tau^{-1} \times (\text{nRomPrim} + 0.5 \times \text{nRomAdj})$$



5-level isotropic SG (——), dimension-adaptive SG [Kouri et al., 2014] (—■—), and proposed ROM/SG for $\tau = 1$ (—▲—), $\tau = 10$ (—▲—), $\tau = 100$ (—▲—), $\tau = \infty$ ( )

Significant reduction in cost, even if (largest) ROM only $10\times$ faster than HDM

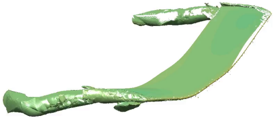
$$\text{Cost} = \text{nHdmPrim} + 0.5 \times \text{nHdmAdj} + \tau^{-1} \times (\text{nRomPrim} + 0.5 \times \text{nRomAdj})$$

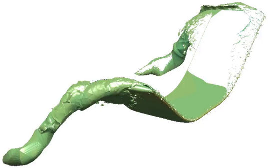


5-level isotropic SG (—), dimension-adaptive SG [Kouri et al., 2014] (—■—), and proposed ROM/SG for $\tau = 1$ (—▲—), $\tau = 10$ (—▲—), $\tau = 100$ (—▲—), $\tau = \infty$ (—▲—)

High- and reduced-order methods for PDE optimization

- Developed **fully discrete adjoint method** for **high-order** numerical discretizations of PDEs and QoIs
- Treatment of **parametrized time domain** (optimal frequency)
- Explicit enforcement of **time-periodicity constraints**
- Extension to **multiphysics** (fluid-structure interaction, particle-laden flow, ...)
- **Acceleration** via rigorous **multi-fidelity** framework that uses reduced-order models, partially converged solutions, and sparse grids
- **Applications**: optimal flapping flight, energy harvesting, data assimilation





Acknowledgments

- DOE Grant DE-AC02-05CH1123 (Alvarez fellowship)
- AFOSR Grant FA9550-20-1-0236 (F. Fahroo)

Tianshu Wen (ND)
ROM/TR optimization

Marzieh Mirhoseini
ROM for convection-dominated

Barter, G. E. (2008).
Shock capturing with PDE-based artificial viscosity for an adaptive, higher-order discontinuous Galerkin finite element method.
PhD thesis, M.I.T.

Huang, D. Z., Persson, P.-O., and Zahr, M. J. (2018).
High-order, linearly stable, partitioned solvers for general multiphysics problems based on implicit-explicit Runge-Kutta schemes.
Computer Methods in Applied Mechanics and Engineering, 346:674 – 706.

Kouri, D. P., Heinkenschloss, M., Ridzal, D., and van Bloemen Waanders, B. G. (2013).
A trust-region algorithm with adaptive stochastic collocation for PDE optimization under uncertainty.
SIAM Journal on Scientific Computing, 35(4):A1847–A1879.

Kouri, D. P., Heinkenschloss, M., Ridzal, D., and van Bloemen Waanders, B. G. (2014).
Inexact objective function evaluations in a trust-region algorithm for PDE-constrained optimization under uncertainty.
SIAM Journal on Scientific Computing, 36(6):A3011–A3029.

Wang, J., Zahr, M. J., and Persson, P.-O. (6/5/2017 – 6/9/2017).
Energetically optimal flapping flight based on a fully discrete adjoint method with explicit treatment of flapping frequency.
In *Proc. of the 23rd AIAA Computational Fluid Dynamics Conference*, Denver, Colorado. American Institute of Aeronautics and Astronautics.

Zahr, M. J. and Persson, P.-O. (1/8/2018 – 1/12/2018b).
An optimization-based discontinuous Galerkin approach for high-order accurate shock tracking.
In *Proc. of the AIAA Science and Technology Forum and Exposition (SciTech2018)*, Kissimmee, Florida. American Institute of Aeronautics and Astronautics.

Zahr, M. J. and Persson, P.-O. (2016).
An adjoint method for a high-order discretization of deforming domain conservation laws for optimization of flow problems.
Journal of Computational Physics, 326(Supplement C):516 – 543.

Zahr, M. J. and Persson, P.-O. (2018a).
An optimization-based approach for high-order accurate discretization of conservation laws with discontinuous solutions.
Journal of Computational Physics, 365:105 – 134.

Zahr, M. J., Persson, P.-O., and Wilkening, J. (2016).
A fully discrete adjoint method for optimization of flow problems on deforming domains with time-periodicity constraints.
Computers & Fluids, 139:130 – 147.

SQP solver: regularization matrix $\boldsymbol{D}$

The mesh regularization matrix $\boldsymbol{D}$ is taken as the stiffness matrix of the linear elliptic PDE

$$\nabla \cdot (k \nabla v_i) = 0 \quad \text{in } \Omega$$

for $i = 1, \ldots, d$. The coefficient is constant over each element and inversely proportional to the element volume

$$k(x) = \frac{\min\limits_{K' \in \mathcal{E}_{h,q}} |K'|}{|K|}, \quad x \in K$$

for each element K in the mesh: *critical* to maintain well-conditioned search directions for meshes where element size varies significantly.

SQP solver: step length (α_k)

The step length, $\alpha_k \in (0, 1]$, is selected using a backtracking line search to ensure *sufficient decrease* of a merit function $\varphi_k : \mathbb{R} \to \mathbb{R}$

$$\varphi_k(\alpha_k) \leq \varphi_k(0) + c\alpha_k \varphi_k'(0), \quad c \in (0, 1).$$

We use the ℓ_1 merit function

$$\varphi_k(\alpha) := f(\boldsymbol{z}_k + \alpha \Delta \boldsymbol{z}_k) + \mu \left\| \boldsymbol{r}(\boldsymbol{z}_k + \alpha \Delta \boldsymbol{z}_k) \right\|_1$$

where $\mu > \left\| \hat{\boldsymbol{\lambda}}(\boldsymbol{z}_k) \right\|_\infty$ because it is “exact”, i.e., any minimizer of the original optimization problem is a minimizer of φ_k.

SQP solver: termination criteria

The termination criteria for the solver is based on the Karush-Kuhn-Tucker (KKT) conditions: $\boldsymbol{z}^\star$ is a solution if there exist Lagrange multipliers $\boldsymbol{\lambda}^\star$ such that

$$\nabla_{\boldsymbol{u}} \mathcal{L}(\boldsymbol{z}^\star, \boldsymbol{\lambda}^\star) = \mathbf{0}, \quad \nabla_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{z}^\star, \boldsymbol{\lambda}^\star) = \mathbf{0}, \quad \boldsymbol{r}(\boldsymbol{z}^\star) = \mathbf{0}$$

Our choice for the Lagrange multiplier estimate $\hat{\boldsymbol{\lambda}}(\boldsymbol{z})$ ensure

$$\nabla_{\boldsymbol{u}} \mathcal{L}(\boldsymbol{z}, \hat{\boldsymbol{\lambda}}(\boldsymbol{z})) = \mathbf{0}$$

and therefore termination is based on the remaining KKT conditions

$$\left\| \nabla_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{z}, \hat{\boldsymbol{\lambda}}(\boldsymbol{z})) \right\| < \epsilon_1, \qquad \|\boldsymbol{r}(\boldsymbol{z})\| < \epsilon_2,$$

where $\epsilon_1, \epsilon_2 > 0$ are convergence tolerances.

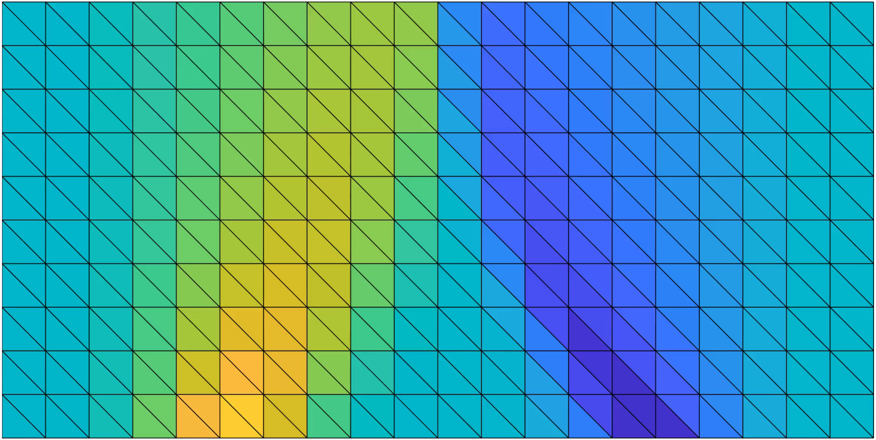
Burgers' equation, accelerating shock: h convergence

Convergence of solution error (E_u) along line $x = 0.8$ and shock surface error (E_Γ)

p	q	$\|\mathcal{E}_h\|$	h	E_u	$m(E_u)$	E_Γ	$m(E_\Gamma)$
1	1	38	1.45e-01	2.72e-02	-	2.32e-03	-
1	1	152	7.25e-02	7.18e-03	1.92	1.09e-03	1.09
1	1	598	3.66e-02	1.91e-03	1.93	1.93e-04	2.53
1	1	2392	1.83e-02	4.69e-04	2.03	3.92e-05	2.30
2	2	38	1.45e-01	5.68e-03	-	4.83e-05	-
2	2	152	7.25e-02	9.64e-05	5.88	2.70e-07	7.48
2	2	608	3.63e-02	6.36e-06	3.92	1.20e-08	4.49
2	2	2432	1.81e-02	8.66e-07	2.88	7.70e-10	3.96
3	3	32	1.58e-01	1.57e-03	-	2.06e-05	-
3	3	128	7.91e-02	1.62e-05	6.60	3.37e-07	5.93
3	3	512	3.95e-02	4.37e-07	5.21	5.90e-09	5.84
3	3	2040	1.98e-02	3.31e-08	3.73	1.87e-10	5.00

Observation: Optimal convergence rates ($\mathcal{O}(h^{p+1})$) obtained for solution error; faster rates obtained for shock surface.

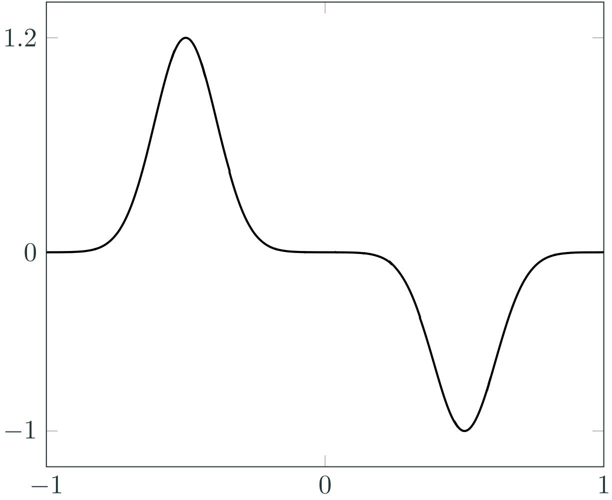
Burgers' equation, shock formation and intersection (space-time)



$p = q = 3$

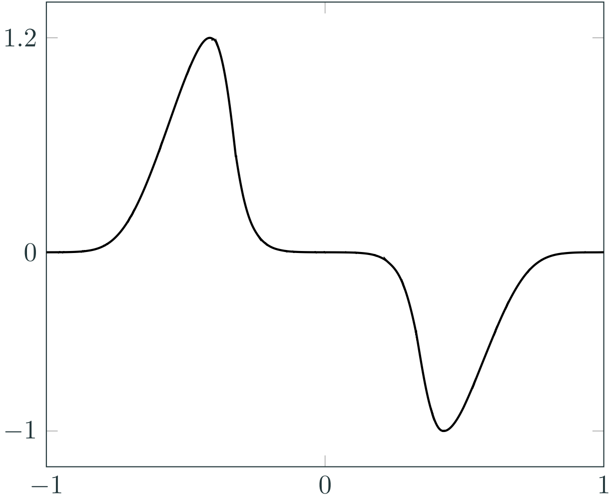
Observation: Triple point where shocks merge is tracked. Insufficient resolution to fully capture shock formation; approximate with discontinuity.

Burgers' equation, shock formation and intersection (time slices)



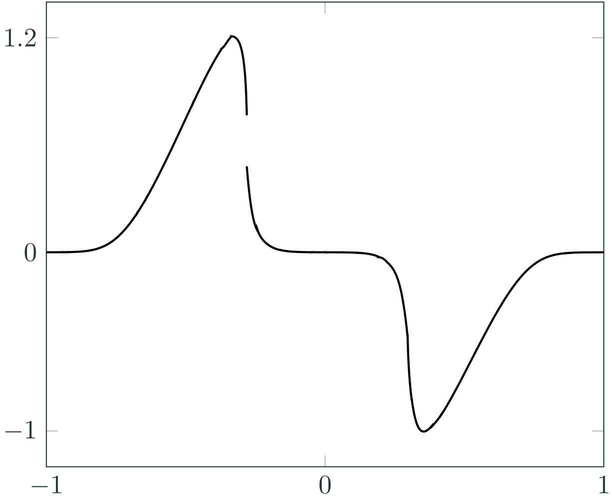
Observation: Triple point where shocks merge is tracked. Insufficient resolution to fully capture shock formation; approximate with discontinuity.

Burgers' equation, shock formation and intersection (time slices)



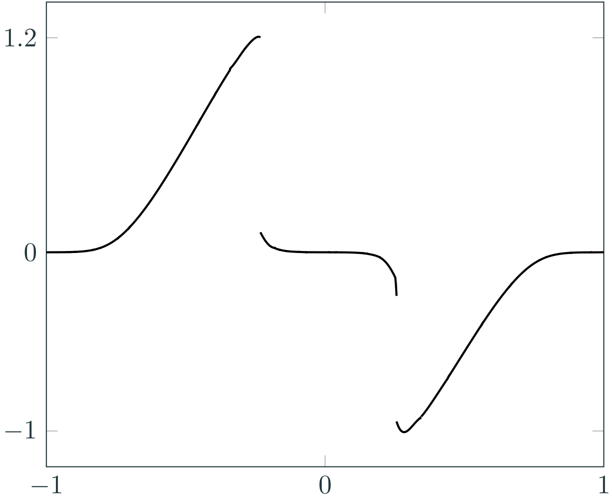
Observation: Triple point where shocks merge is tracked. Insufficient resolution to fully capture shock formation; approximate with discontinuity.

Burgers' equation, shock formation and intersection (time slices)



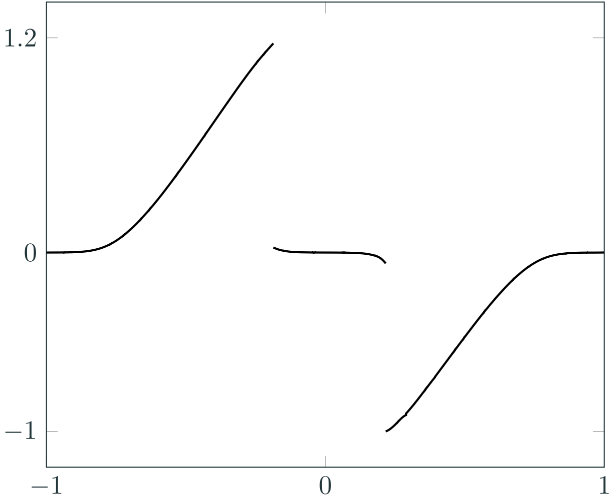
Observation: Triple point where shocks merge is tracked. Insufficient resolution to fully capture shock formation; approximate with discontinuity.

Burgers' equation, shock formation and intersection (time slices)



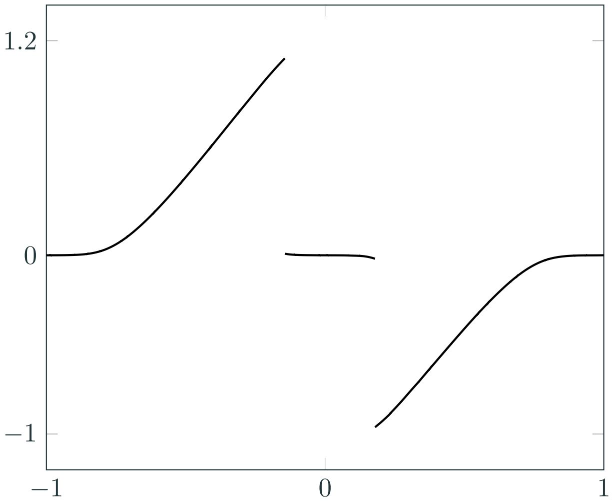
Observation: Triple point where shocks merge is tracked. Insufficient resolution to fully capture shock formation; approximate with discontinuity.

Burgers' equation, shock formation and intersection (time slices)



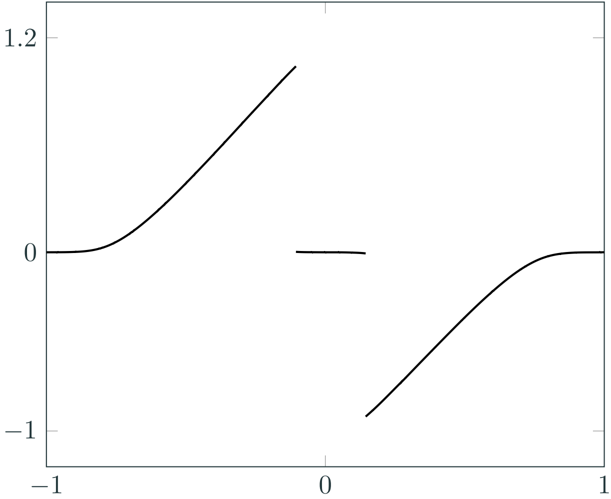
Observation: Triple point where shocks merge is tracked. Insufficient resolution to fully capture shock formation; approximate with discontinuity.

Burgers' equation, shock formation and intersection (time slices)



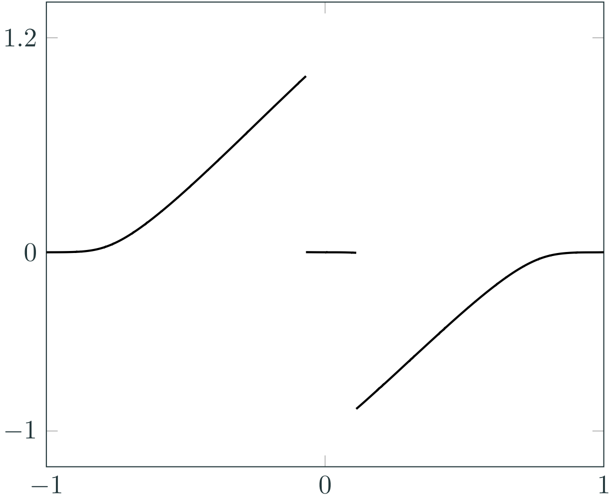
Observation: Triple point where shocks merge is tracked. Insufficient resolution to fully capture shock formation; approximate with discontinuity.

Burgers' equation, shock formation and intersection (time slices)



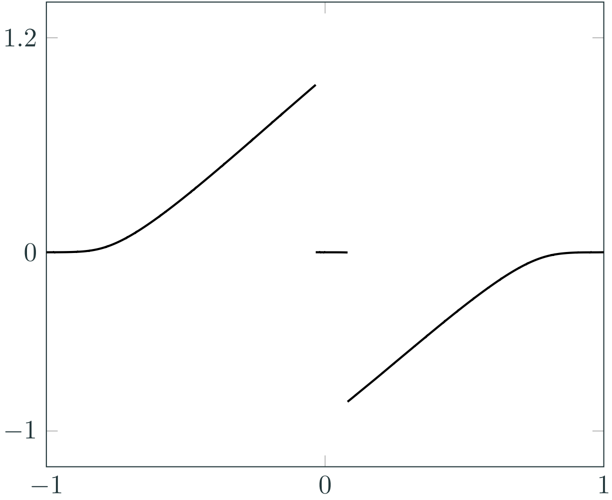
Observation: Triple point where shocks merge is tracked. Insufficient resolution to fully capture shock formation; approximate with discontinuity.

Burgers' equation, shock formation and intersection (time slices)



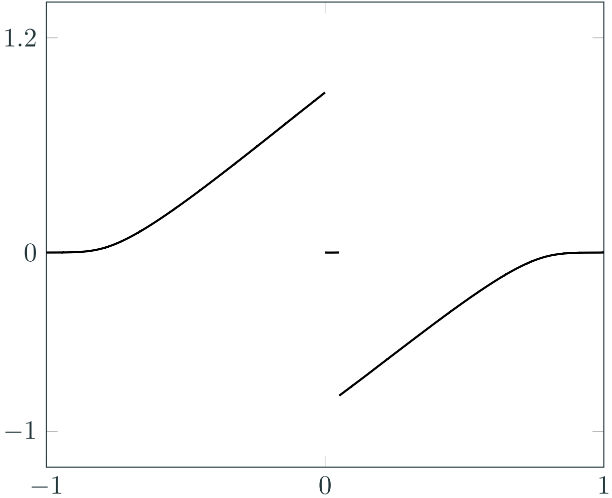
Observation: Triple point where shocks merge is tracked. Insufficient resolution to fully capture shock formation; approximate with discontinuity.

Burgers' equation, shock formation and intersection (time slices)



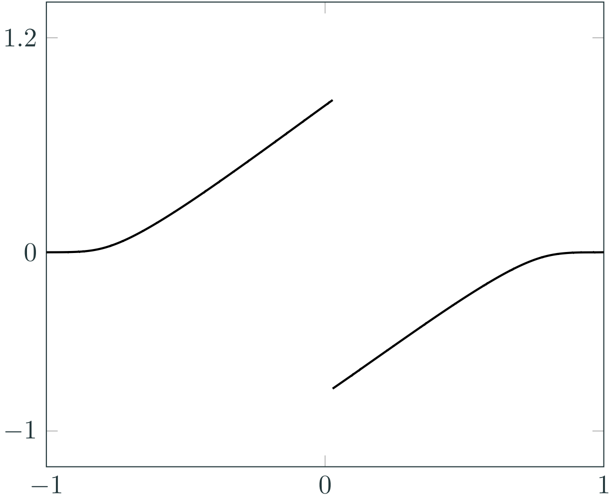
Observation: Triple point where shocks merge is tracked. Insufficient resolution to fully capture shock formation; approximate with discontinuity.

Burgers' equation, shock formation and intersection (time slices)



Observation: Triple point where shocks merge is tracked. Insufficient resolution to fully capture shock formation; approximate with discontinuity.

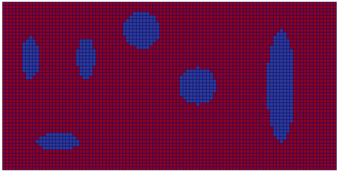
Burgers' equation, shock formation and intersection (time slices)

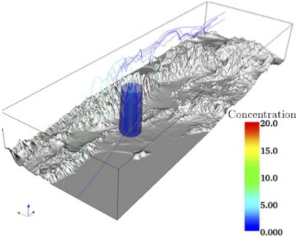


Observation: Triple point where shocks merge is tracked. Insufficient resolution to fully capture shock formation; approximate with discontinuity.

PDE optimization is ubiquitous in science and engineering

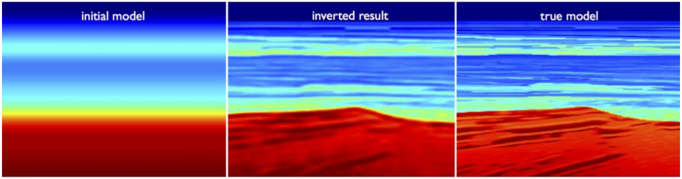
Inverse problems: Infer the problem setup given solution observations





Material inversion: find inclusions from acoustic, structural measurements

Source inversion: find source of contaminant from downstream measurements



Full waveform inversion: estimate subsurface of crust from acoustic measurements

High-order 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

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

Discrete adjoint equations can be derived from an algebraic manipulation to save computations

Let $\boldsymbol{u}(\boldsymbol{\mu})$ be the solution of $\boldsymbol{r}(\cdot, \boldsymbol{\mu}) = 0$

$$\boldsymbol{r}(\boldsymbol{\mu}) = \boldsymbol{r}(\boldsymbol{u}(\boldsymbol{\mu}), \boldsymbol{\mu}) = 0, \qquad F(\boldsymbol{\mu}) = F(\boldsymbol{u}(\boldsymbol{\mu}), \boldsymbol{\mu})$$

Discrete adjoint equations can be derived from an algebraic manipulation to save computations

Let $\boldsymbol{u}(\boldsymbol{\mu})$ be the solution of $\boldsymbol{r}(\cdot, \boldsymbol{\mu}) = 0$

$$\boldsymbol{r}(\boldsymbol{\mu}) = \boldsymbol{r}(\boldsymbol{u}(\boldsymbol{\mu}), \boldsymbol{\mu}) = 0, \qquad F(\boldsymbol{\mu}) = F(\boldsymbol{u}(\boldsymbol{\mu}), \boldsymbol{\mu})$$

The total derivative of $\boldsymbol{r}$ leads to the sensitivity equations

$$D\boldsymbol{r} = \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\mu}} + \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}} \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{\mu}} = 0 \implies \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{\mu}} = -\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}}^{-1} \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\mu}}$$

Discrete adjoint equations can be derived from an algebraic manipulation to save computations

Let $\boldsymbol{u}(\boldsymbol{\mu})$ be the solution of $\boldsymbol{r}(\cdot, \boldsymbol{\mu}) = 0$

$$\boldsymbol{r}(\boldsymbol{\mu}) = \boldsymbol{r}(\boldsymbol{u}(\boldsymbol{\mu}), \boldsymbol{\mu}) = 0, \qquad F(\boldsymbol{\mu}) = F(\boldsymbol{u}(\boldsymbol{\mu}), \boldsymbol{\mu})$$

The total derivative of $\boldsymbol{r}$ leads to the sensitivity equations

$$D\boldsymbol{r} = \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\mu}} + \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}} \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{\mu}} = 0 \implies \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{\mu}} = -\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}}^{-1} \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\mu}}$$

The total derivative of F

$$DF = \frac{\partial F}{\partial \boldsymbol{\mu}} + \frac{\partial F}{\partial \boldsymbol{u}} \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{\mu}}$$

Discrete adjoint equations can be derived from an algebraic manipulation to save computations

Let $\boldsymbol{u}(\boldsymbol{\mu})$ be the solution of $\boldsymbol{r}(\cdot, \boldsymbol{\mu}) = 0$

$$\boldsymbol{r}(\boldsymbol{\mu}) = \boldsymbol{r}(\boldsymbol{u}(\boldsymbol{\mu}), \boldsymbol{\mu}) = 0, \qquad F(\boldsymbol{\mu}) = F(\boldsymbol{u}(\boldsymbol{\mu}), \boldsymbol{\mu})$$

The total derivative of $\boldsymbol{r}$ leads to the sensitivity equations

$$D\boldsymbol{r} = \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\mu}} + \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}} \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{\mu}} = 0 \implies \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{\mu}} = -\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}}^{-1} \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\mu}}$$

The total derivative of F

$$DF = \frac{\partial F}{\partial \boldsymbol{\mu}} + \frac{\partial F}{\partial \boldsymbol{u}} \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{\mu}} = \frac{\partial F}{\partial \boldsymbol{\mu}} - \frac{\partial F}{\partial \boldsymbol{u}} \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}}^{-1} \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\mu}}$$

Discrete adjoint equations can be derived from an algebraic manipulation to save computations

Let $\boldsymbol{u}(\boldsymbol{\mu})$ be the solution of $\boldsymbol{r}(\cdot, \boldsymbol{\mu}) = 0$

$$\boldsymbol{r}(\boldsymbol{\mu}) = \boldsymbol{r}(\boldsymbol{u}(\boldsymbol{\mu}), \boldsymbol{\mu}) = 0, \qquad F(\boldsymbol{\mu}) = F(\boldsymbol{u}(\boldsymbol{\mu}), \boldsymbol{\mu})$$

The total derivative of $\boldsymbol{r}$ leads to the sensitivity equations

$$D\boldsymbol{r} = \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\mu}} + \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}} \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{\mu}} = 0 \implies \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{\mu}} = -\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}}^{-1} \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\mu}}$$

The total derivative of F

$$DF = \frac{\partial F}{\partial \boldsymbol{\mu}} + \frac{\partial F}{\partial \boldsymbol{u}} \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{\mu}} = \frac{\partial F}{\partial \boldsymbol{\mu}} - \frac{\partial F}{\partial \boldsymbol{u}} \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}}^{-1} \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\mu}} = \frac{\partial F}{\partial \boldsymbol{\mu}} - \boldsymbol{\lambda}^T \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\mu}}$$

Discrete adjoint equations can be derived from an algebraic manipulation to save computations

Let $\boldsymbol{u}(\boldsymbol{\mu})$ be the solution of $\boldsymbol{r}(\cdot, \boldsymbol{\mu}) = 0$

$$\boldsymbol{r}(\boldsymbol{\mu}) = \boldsymbol{r}(\boldsymbol{u}(\boldsymbol{\mu}), \boldsymbol{\mu}) = 0, \qquad F(\boldsymbol{\mu}) = F(\boldsymbol{u}(\boldsymbol{\mu}), \boldsymbol{\mu})$$

The total derivative of $\boldsymbol{r}$ leads to the sensitivity equations

$$D\boldsymbol{r} = \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\mu}} + \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}}\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{\mu}} = 0 \implies \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{\mu}} = -\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}}^{-1}\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\mu}}$$

The total derivative of F

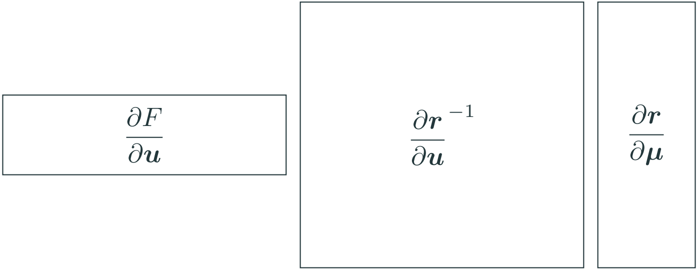
$$DF = \frac{\partial F}{\partial \boldsymbol{\mu}} + \frac{\partial F}{\partial \boldsymbol{u}}\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{\mu}} = \frac{\partial F}{\partial \boldsymbol{\mu}} - \frac{\partial F}{\partial \boldsymbol{u}}\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}}^{-1}\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\mu}} = \frac{\partial F}{\partial \boldsymbol{\mu}} - \boldsymbol{\lambda}^T\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\mu}}$$

Algebraic equations leads to adjoint equations

$$\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}}^T \boldsymbol{\lambda} = \frac{\partial F}{\partial \boldsymbol{u}}^T$$

Sensitivity vs. adjoint method to compute gradient of F

$$\frac{\partial F}{\partial \boldsymbol{u}} \, \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}}^{-1} \, \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\mu}}$$



Sensitivity vs. adjoint method to compute gradient of F

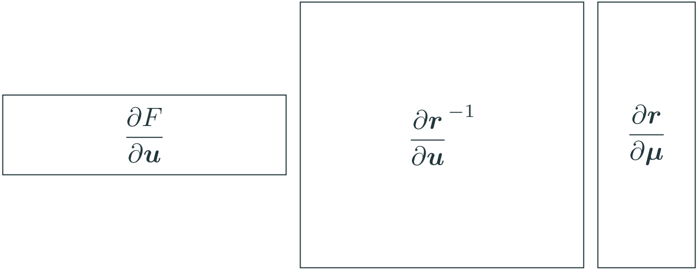
$$\frac{\partial F}{\partial \boldsymbol{u}} \; \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}}^{-1} \; \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\mu}}$$

$$\boxed{\frac{\partial F}{\partial \boldsymbol{u}}} \; \boxed{\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{\mu}}}$$

Sensitivity method requires $n_{\boldsymbol{\mu}}$ linear solves and $n_F n_{\boldsymbol{\mu}}$ inner products ($\mathbb{R}^{n_{\boldsymbol{u}}}$)

Sensitivity vs. adjoint method to compute gradient of F

$$\frac{\partial F}{\partial \boldsymbol{u}} \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}}^{-1} \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\mu}}$$



Sensitivity method requires $n_{\boldsymbol{\mu}}$ linear solves and $n_F n_{\boldsymbol{\mu}}$ inner products ($\mathbb{R}^{n_{\boldsymbol{u}}}$)

Sensitivity vs. adjoint method to compute gradient of F

$$\frac{\partial F}{\partial \boldsymbol{u}} \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}}^{-1} \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\mu}}$$

$$\boxed{\boldsymbol{\lambda}^T} \; \boxed{\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\mu}}}$$

Sensitivity method requires $n_{\boldsymbol{\mu}}$ linear solves and $n_F n_{\boldsymbol{\mu}}$ inner products ($\mathbb{R}^{n_{\boldsymbol{u}}}$)

Adjoint method requires n_F linear solves and $n_F n_{\boldsymbol{\mu}}$ inner products ($\mathbb{R}^{n_{\boldsymbol{u}}}$)

Adjoint equation derivation: outline

- Define **auxiliary** PDE-constrained optimization problem

$$\begin{aligned}
\underset{\substack{\boldsymbol{u}_0,\ \ldots,\ \boldsymbol{u}_{N_t} \in \mathbb{R}^{N_{\boldsymbol{u}}},\\ \boldsymbol{k}_{1,1},\ \ldots,\ \boldsymbol{k}_{N_t,s} \in \mathbb{R}^{N_{\boldsymbol{u}}}}}{\text{minimize}} \quad & F(\boldsymbol{u}_0,\ \ldots,\ \boldsymbol{u}_{N_t},\ \boldsymbol{k}_{1,1},\ \ldots,\ \boldsymbol{k}_{N_t,s},\ \boldsymbol{\mu}) \\
\text{subject to} \quad & \boldsymbol{R}_0 = \boldsymbol{u}_0 - \boldsymbol{g}(\boldsymbol{\mu}) = 0 \\
& \boldsymbol{R}_n = \boldsymbol{u}_n - \boldsymbol{u}_{n-1} - \sum_{i=1}^{s} b_i \boldsymbol{k}_{n,i} = 0 \\
& \boldsymbol{R}_{n,i} = \boldsymbol{M}\boldsymbol{k}_{n,i} - \Delta t_n \boldsymbol{r}\left(\boldsymbol{u}_{n,i},\ \boldsymbol{\mu},\ t_{n,i}\right) = 0
\end{aligned}$$

- Define **Lagrangian**

$$\mathcal{L}(\boldsymbol{u}_n,\ \boldsymbol{k}_{n,i},\ \boldsymbol{\lambda}_n,\ \boldsymbol{\kappa}_{n,i}) = F - \boldsymbol{\lambda}_0{}^T \boldsymbol{R}_0 - \sum_{n=1}^{N_t} \boldsymbol{\lambda}_n{}^T \boldsymbol{R}_n - \sum_{n=1}^{N_t} \sum_{i=1}^{s} \boldsymbol{\kappa}_{n,i}{}^T \boldsymbol{R}_{n,i}$$

- The solution of the optimization problem is given by the **Karush-Kuhn-Tucker (KKT) sytem**

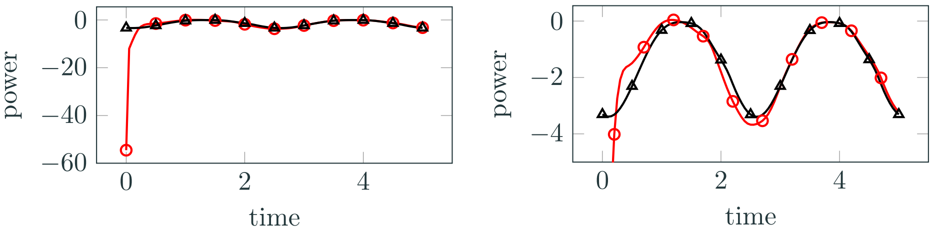
$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{u}_n} = 0, \quad \frac{\partial \mathcal{L}}{\partial \boldsymbol{k}_{n,i}} = 0, \quad \frac{\partial \mathcal{L}}{\partial \boldsymbol{\lambda}_n} = 0, \quad \frac{\partial \mathcal{L}}{\partial \boldsymbol{\kappa}_{n,i}} = 0$$

Extension: constraint requiring time-periodicity [Zahr et al., 2016]

Optimization of *cyclic* problems requires finding time-periodic solution of PDE; necessary for physical relevance and avoid transients that may lead to crash

$$
\begin{aligned}
\underset{\boldsymbol{U},\ \boldsymbol{\mu}}{\text{minimize}} \quad & \mathcal{J}(\boldsymbol{U}, \boldsymbol{\mu}) \\
\text{subject to} \quad & \color{red}{\boldsymbol{U}(\boldsymbol{x}, 0) = \boldsymbol{U}(\boldsymbol{x}, T)} \\
& \frac{\partial \boldsymbol{U}}{\partial t} + \nabla \cdot \boldsymbol{F}(\boldsymbol{U}, \nabla \boldsymbol{U}) = 0
\end{aligned}
$$

$$
\begin{aligned}
\boldsymbol{\lambda}_{N_t} &= \color{red}{\boldsymbol{\lambda}_0} + \frac{\partial F}{\partial \boldsymbol{u}_{N_t}}^T \\
\boldsymbol{\lambda}_{n-1} &= \boldsymbol{\lambda}_n + \frac{\partial F}{\partial \boldsymbol{u}_{n-1}}^T + \sum_{i=1}^{s} \Delta t_n \frac{\partial \boldsymbol{r}_{n,i}}{\partial \boldsymbol{u}}^T \boldsymbol{\kappa}_{n,i} \\
\boldsymbol{M}^T \boldsymbol{\kappa}_{n,i} &= \frac{\partial F}{\partial \boldsymbol{u}_{N_t}}^T + b_i \boldsymbol{\lambda}_n + \sum_{j=i}^{s} a_{ji} \Delta t_n \frac{\partial \boldsymbol{r}_{n,i}}{\partial \boldsymbol{u}}^T \boldsymbol{\kappa}_{n,j}
\end{aligned}
$$

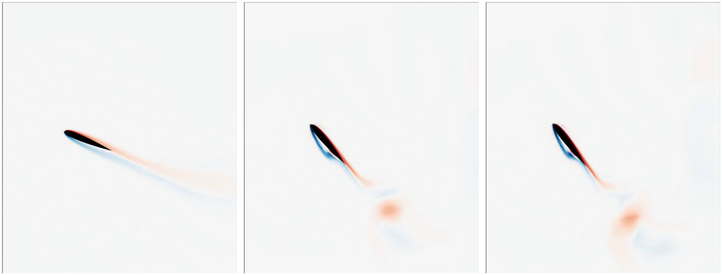


Time history of power on airfoil of flow initialized from steady-state (red line with circle markers) and from a time-periodic solution (black line with triangle markers)

Extension: Parametrized time domain [Wang et al., 2017]

Parametrization of time domain, e.g., flapping frequency, leads to parametrization of time discretization in fully discrete setting

$$T(\boldsymbol{\mu}) = N_t \Delta t \implies N_t = N_t(\boldsymbol{\mu}) \text{ or } \Delta t = \Delta t(\boldsymbol{\mu})$$

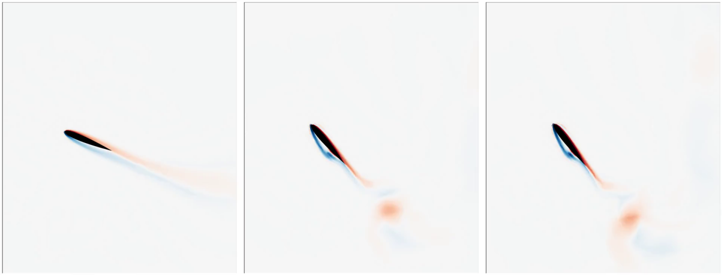


Extension: Parametrized time domain [Wang et al., 2017]

Parametrization of time domain, e.g., flapping frequency, leads to parametrization of time discretization in fully discrete setting

$$T(\boldsymbol{\mu}) = N_t \Delta t \implies N_t = N_t(\boldsymbol{\mu}) \text{ or } \Delta t = \Delta t(\boldsymbol{\mu})$$

Choose $\Delta t = \Delta t(\boldsymbol{\mu})$ to avoid discrete changes



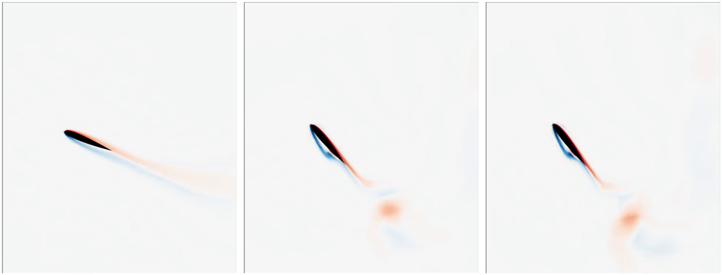
Extension: Parametrized time domain [Wang et al., 2017]

Parametrization of time domain, e.g., flapping frequency, leads to parametrization of time discretization in fully discrete setting

$$T(\boldsymbol{\mu}) = N_t \Delta t \implies N_t = N_t(\boldsymbol{\mu}) \text{ or } \Delta t = \Delta t(\boldsymbol{\mu})$$

Choose $\Delta t = \Delta t(\boldsymbol{\mu})$ to avoid discrete changes

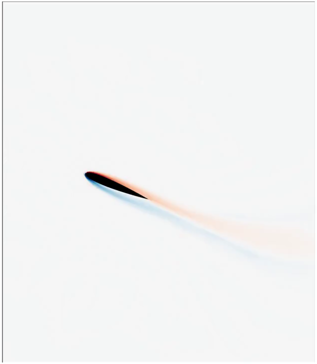
Does not change adjoint equations themselves, only reconstruction of gradient from adjoint solution



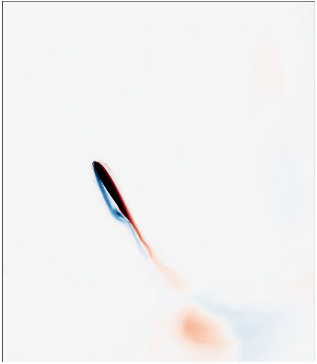
Energetically optimal flapping vs. required thrust

Energy $= 1.8445$	Energy $= 0.21934$	Energy $= 6.2869$
Thrust $= 0.06729$	Thrust $= 0.0000$	Thrust $= 2.5000$

Initial Guess

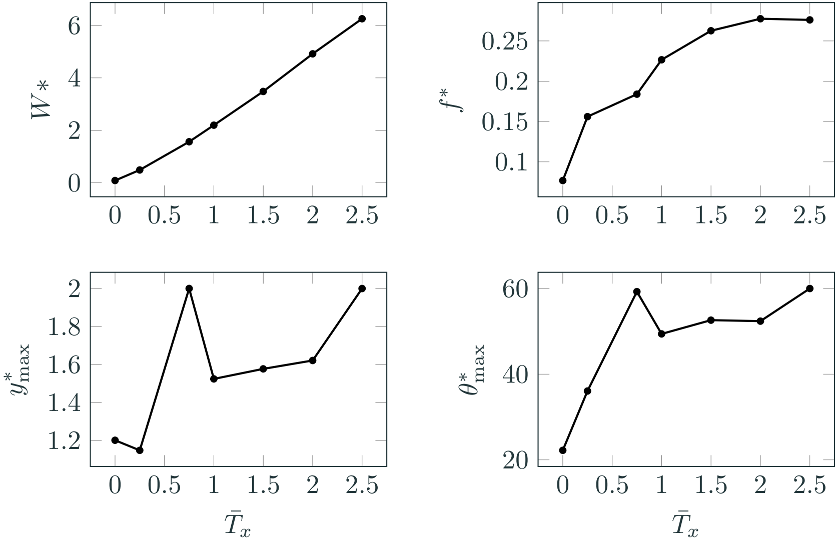


Optimal
$T_x = 0$



Optimal
$T_x = 2.5$

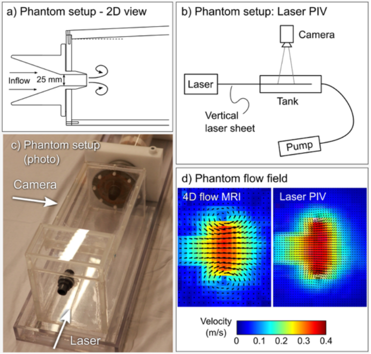
Energetically optimal flapping vs. required thrust: QoI



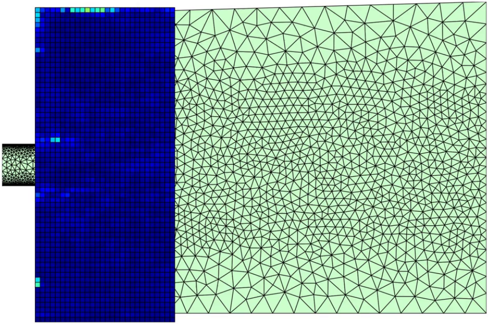
The optimal flapping energy (W^*), frequency (f^*), maximum heaving amplitude ($y^*_{\max}$), and maximum pitching amplitude ($\theta^*_{\max}$) as a function of the thrust constraint $\bar{T}_x$.

High-resolution *in vivo* images through optimization

Goal: visualize *in vivo* flow with high-resolution and accurately compute clinically relevant quantities from quick scans



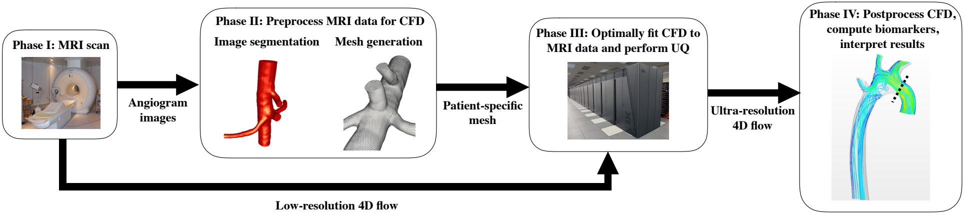
Experimental setup



Noisy, low-resolution MRI data

Approach: determine CFD parameters (material properties, boundary conditions) such that the simulation matches MRI data using optimization

Simulation-based imaging (SBI) workflow



MRI optimization formulation that respects scanner physics

$$\boxed{\underset{\boldsymbol{\mu}}{\text{minimize}} \quad \sum_{i=1}^{n_{xyz}} \sum_{n=1}^{n_t} \frac{\alpha_{i,n}}{2} \left\| \boldsymbol{d}_{i,n}(\boldsymbol{U}(\boldsymbol{\mu})) - \boldsymbol{d}^*_{i,n} \right\|_2^2}$$

$\boldsymbol{d}^*_{i,n}$: MRI measurement taken in voxel i at the nth time sample

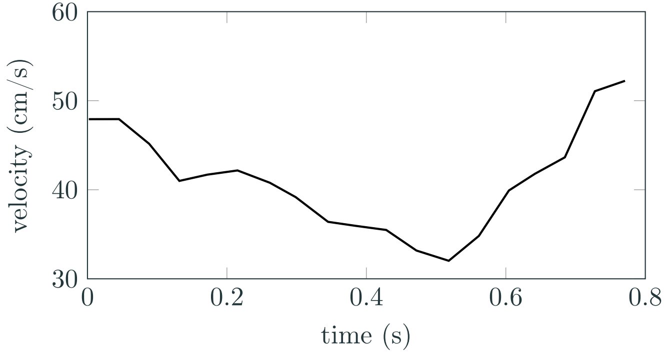
$\boldsymbol{d}_{i,n}(\boldsymbol{U})$: computational representation of $\boldsymbol{d}^*_{i,n}$

$$\begin{aligned}
\boldsymbol{d}_{i,n}(\boldsymbol{U},\, \boldsymbol{\mu}) &= \int_0^T \int_V w_{i,n}(\boldsymbol{x},\, t) \cdot \boldsymbol{U}(\boldsymbol{x},\, t)\, dV\, dt \\
w_{i,n}(\boldsymbol{x},\, t) &= \chi_s(\boldsymbol{x};\, \boldsymbol{x}_i,\, \Delta\boldsymbol{x})\chi_t(t;\, t_n,\, \Delta t) \\
\chi_t(s;\, c,\, w) &= \frac{1}{1 + e^{-(s-(c-0.5w))/\sigma}} - \frac{1}{1 + e^{-(s-(c+0.5w))/\sigma}} \\
\chi_s(\boldsymbol{x};\, \boldsymbol{c},\, \boldsymbol{w}) &= \chi_t(x_1;\, c_1,\, w_1)\chi_t(x_2;\, c_2,\, w_2)\chi_t(x_3;\, c_3,\, w_3)
\end{aligned}$$

$\boldsymbol{x}_i$ center of ith MRI voxel, $\Delta\boldsymbol{x}$ size of MRI voxel

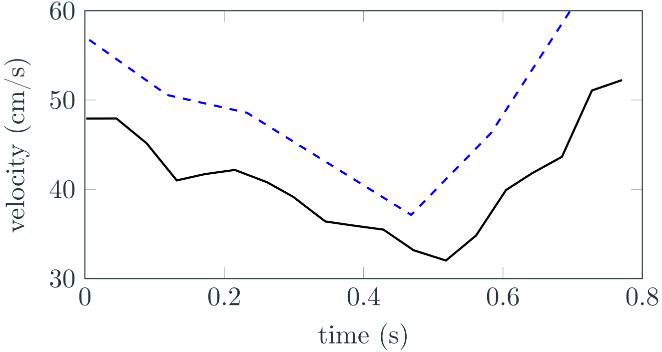
t_n time instance of nth MRI sample, Δt sampling interval in time

Quantitative comparison of 4D flow and SBI reconstruction



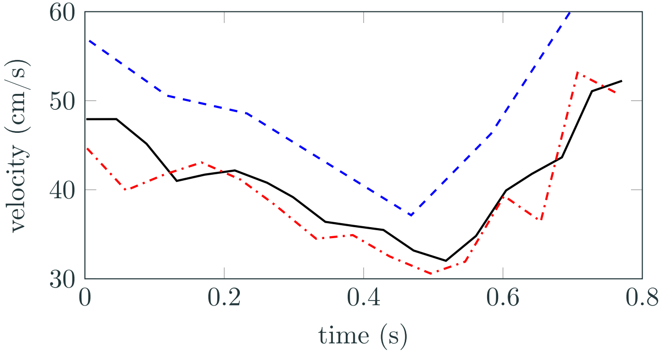
The reconstructed flow field (-·-·-) provides better agreement to accurate velocity measurements (——) on a 2D section than the 4D flow MRI measurements (- - -)

Quantitative comparison of 4D flow and SBI reconstruction



The reconstructed flow field (-·-·-) provides better agreement to accurate velocity measurements (——) on a 2D section than the 4D flow MRI measurements (- - -)

Quantitative comparison of 4D flow and SBI reconstruction



The reconstructed flow field (red dash-dotted line) provides better agreement to accurate velocity measurements (black solid line) on a 2D section than the 4D flow MRI measurements (blue dashed line)

Extension: Multiphysics problems [Huang et al., 2018]

For problems that involve the interaction of multiple types of physical phenomena, *no changes required* if monolithic system considered

$$\boldsymbol{M}_0\dot{\boldsymbol{u}}_0 = \boldsymbol{r}_0(\boldsymbol{u}_0,\, \boldsymbol{c}_0(\boldsymbol{u}_0,\, \boldsymbol{u}_1))$$
$$\boldsymbol{M}_1\dot{\boldsymbol{u}}_1 = \boldsymbol{r}_1(\boldsymbol{u}_1,\, \boldsymbol{c}_1(\boldsymbol{u}_0,\, \boldsymbol{u}_1))$$

Extension: Multiphysics problems [Huang et al., 2018]

For problems that involve the interaction of multiple types of physical phenomena, *no changes required* if monolithic system considered

$$\boldsymbol{M}_0 \dot{\boldsymbol{u}}_0 = \boldsymbol{r}_0(\boldsymbol{u}_0,\, \boldsymbol{c}_0(\boldsymbol{u}_0,\, \boldsymbol{u}_1))$$
$$\boldsymbol{M}_1 \dot{\boldsymbol{u}}_1 = \boldsymbol{r}_1(\boldsymbol{u}_1,\, \boldsymbol{c}_1(\boldsymbol{u}_0,\, \boldsymbol{u}_1))$$

However, to solve in partitioned manner and achieve high-order, split as follows and apply implicit-explicit Runge-Kutta

$$\boldsymbol{M}_0 \dot{\boldsymbol{u}}_0 = \boldsymbol{r}_0(\boldsymbol{u}_0,\, \tilde{\boldsymbol{c}}_0) \quad + \quad (\boldsymbol{r}_0(\boldsymbol{u}_0,\, \boldsymbol{c}_0(\boldsymbol{u}_0,\, \boldsymbol{u}_1)) - \boldsymbol{r}_0(\boldsymbol{u}_0,\, \tilde{\boldsymbol{c}}_0))$$
$$\boldsymbol{M}_1 \dot{\boldsymbol{u}}_1 = \boldsymbol{r}_1(\boldsymbol{u}_1,\, \tilde{\boldsymbol{c}}_1) \quad + \quad (\boldsymbol{r}_1(\boldsymbol{u}_1,\, \boldsymbol{c}_1(\boldsymbol{u}_0,\, \boldsymbol{u}_1)) - \boldsymbol{r}_1(\boldsymbol{u}_1,\, \tilde{\boldsymbol{c}}_1))$$

Extension: Multiphysics problems [Huang et al., 2018]

For problems that involve the interaction of multiple types of physical phenomena, *no changes required* if monolithic system considered

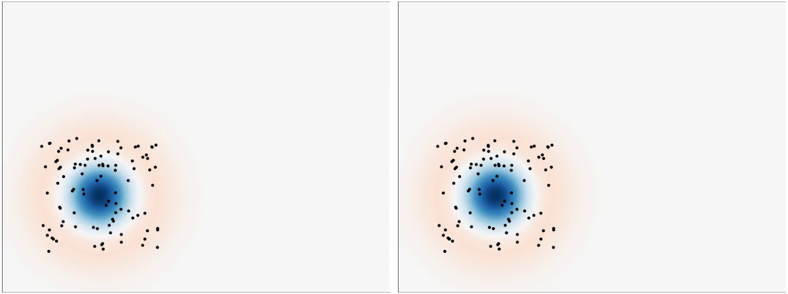
$$\begin{aligned} \boldsymbol{M}_0 \dot{\boldsymbol{u}}_0 &= \boldsymbol{r}_0(\boldsymbol{u}_0,\, \boldsymbol{c}_0(\boldsymbol{u}_0,\, \boldsymbol{u}_1)) \\ \boldsymbol{M}_1 \dot{\boldsymbol{u}}_1 &= \boldsymbol{r}_1(\boldsymbol{u}_1,\, \boldsymbol{c}_1(\boldsymbol{u}_0,\, \boldsymbol{u}_1)) \end{aligned}$$

However, to solve in partitioned manner and achieve high-order, split as follows and apply implicit-explicit Runge-Kutta

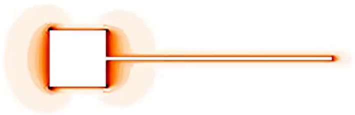
$$\begin{aligned} \boldsymbol{M}_0 \dot{\boldsymbol{u}}_0 &= \boldsymbol{r}_0(\boldsymbol{u}_0,\, \tilde{\boldsymbol{c}}_0) \quad + \quad (\boldsymbol{r}_0(\boldsymbol{u}_0,\, \boldsymbol{c}_0(\boldsymbol{u}_0,\, \boldsymbol{u}_1)) - \boldsymbol{r}_0(\boldsymbol{u}_0,\, \tilde{\boldsymbol{c}}_0)) \\ \boldsymbol{M}_1 \dot{\boldsymbol{u}}_1 &= \boldsymbol{r}_1(\boldsymbol{u}_1,\, \tilde{\boldsymbol{c}}_1) \quad + \quad (\boldsymbol{r}_1(\boldsymbol{u}_1,\, \boldsymbol{c}_1(\boldsymbol{u}_0,\, \boldsymbol{u}_1)) - \boldsymbol{r}_1(\boldsymbol{u}_1,\, \tilde{\boldsymbol{c}}_1)) \end{aligned}$$

Adjoint equations inherit explicit-implicit structure

High-order method for general multiphysics problems with unconditional linear stability



Particle-laden flow



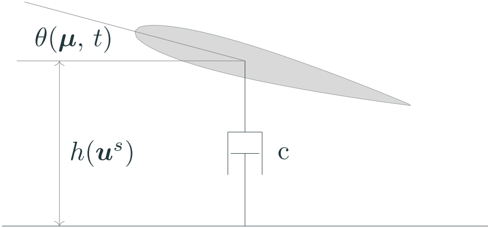
Fluid-structure interaction

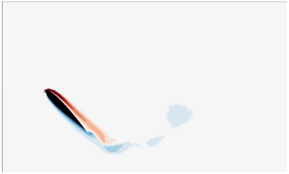
Optimal energy harvesting from foil-damper system

Goal: Maximize energy harvested from foil-damper system

$$\underset{\boldsymbol{\mu}}{\text{maximize}} \quad \frac{1}{T}\int_0^T \left(c\dot{h}^2(\boldsymbol{u}^s) - M_z(\boldsymbol{u}^f)\dot{\theta}(\boldsymbol{\mu},\, t)\right) dt$$

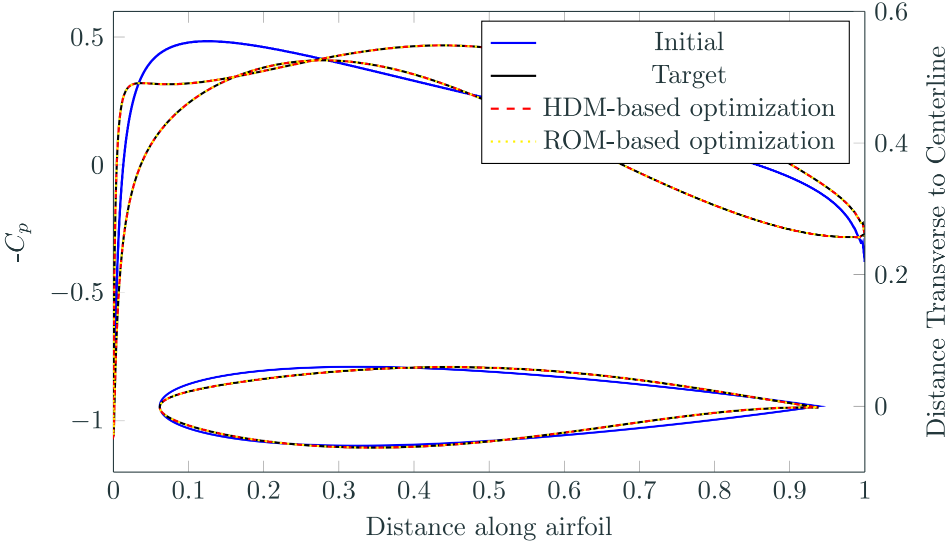
- Fluid: Isentropic Navier-Stokes on deforming domain (ALE)
- Structure: Force balance in y-direction between foil and damper
- Motion driven by *imposed* $\theta(\boldsymbol{\mu},\, t) = \mu_1 \cos(2\pi f t)$



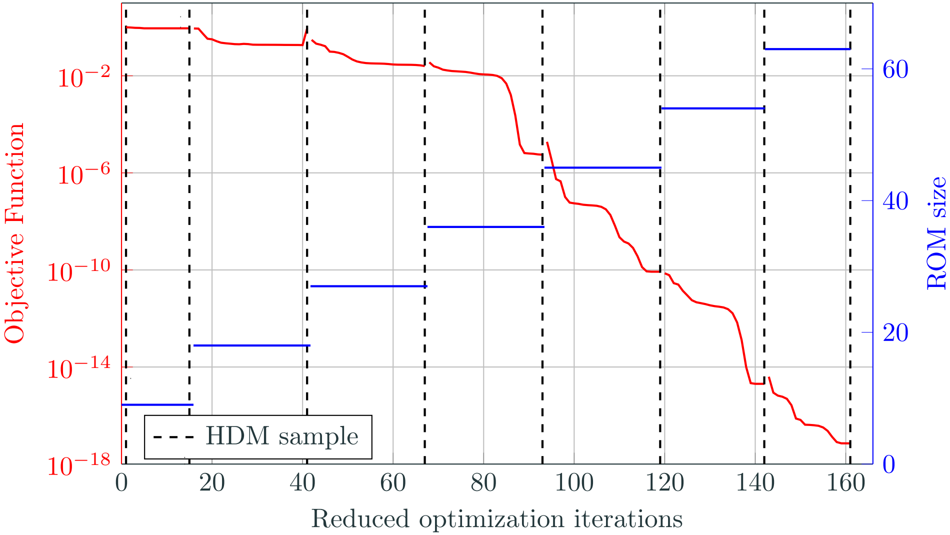


$\mu_1^* \approx 45^\circ$

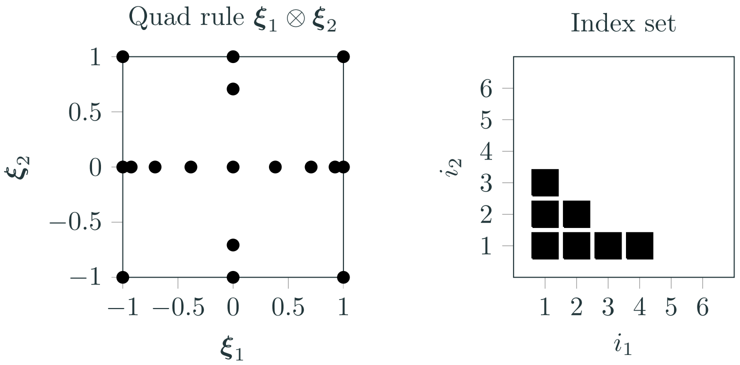
Proposed method: recovers target airfoil



At the cost of ROM queries

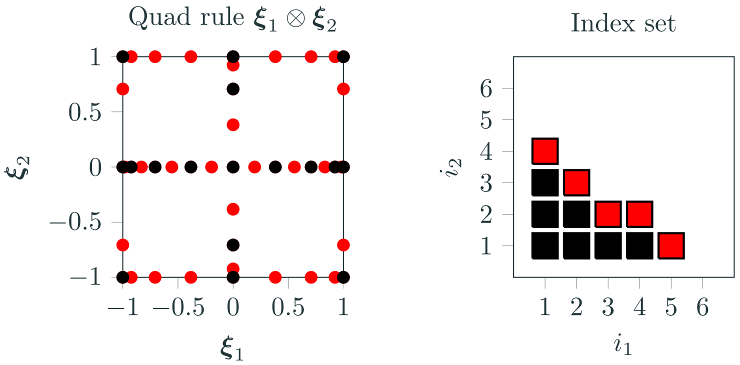


Source of inexactness: anisotropic sparse grids



Index set ($\mathcal{I}$) – ●　　Neighbors ($\mathcal{N}(\mathcal{I})$) – ●

Source of inexactness: anisotropic sparse grids



Index set ($\mathcal{I}$) – ●　　　Neighbors ($\mathcal{N}(\mathcal{I})$) – ●

Trust region ingredients for global convergence

$$\underset{\boldsymbol{\mu}\in\mathbb{R}^{n_{\boldsymbol{\mu}}}}{\text{minimize}} \quad F(\boldsymbol{\mu}) \qquad \longrightarrow \qquad \begin{aligned} &\underset{\boldsymbol{\mu}\in\mathbb{R}^{n_{\boldsymbol{\mu}}}}{\text{minimize}} && m_k(\boldsymbol{\mu}) \\ &\text{subject to} && \|\boldsymbol{\mu}-\boldsymbol{\mu}_k\| \leq \Delta_k \end{aligned}$$

Approximation models

$$m_k(\boldsymbol{\mu}),\ \psi_k(\boldsymbol{\mu})$$

Error indicators

$$\begin{aligned} \|\nabla F(\boldsymbol{\mu}) - \nabla m_k(\boldsymbol{\mu})\| &\leq \xi \varphi_k(\boldsymbol{\mu}) && \xi > 0 \\ |F(\boldsymbol{\mu}_k) - F(\boldsymbol{\mu}) + \psi_k(\boldsymbol{\mu}) - \psi_k(\boldsymbol{\mu}_k)| &\leq \sigma\theta_k(\boldsymbol{\mu}) && \sigma > 0 \end{aligned}$$

Adaptivity

$$\begin{aligned} \varphi_k(\boldsymbol{\mu}_k) &\leq \kappa_\varphi \min\{\|\nabla m_k(\boldsymbol{\mu}_k)\|\,,\ \Delta_k\} \\ \theta_k(\hat{\boldsymbol{\mu}}_k)^\omega &\leq \eta \min\{m_k(\boldsymbol{\mu}_k) - m_k(\hat{\boldsymbol{\mu}}_k),\ r_k\} \end{aligned}$$

Trust region method with inexact gradients and objective

1: **Model update**: Choose model m_k and error indicator φ_k

$$\varphi_k(\boldsymbol{\mu}_k) \leq \kappa_\varphi \min\{\|\nabla m_k(\boldsymbol{\mu}_k)\|\,,\ \Delta_k\}$$

2: **Step computation**: Approximately solve the trust region subproblem

$$\hat{\boldsymbol{\mu}}_k \quad = \quad \underset{\boldsymbol{\mu}\in\mathbb{R}^{n_{\boldsymbol{\mu}}}}{\arg\min}\ m_k(\boldsymbol{\mu}) \quad \text{subject to} \quad \|\boldsymbol{\mu} - \boldsymbol{\mu}_k\| \leq \Delta_k$$

3: **Step acceptance**: Compute approximation of actual-to-predicted reduction

$$\rho_k = \frac{\psi_k(\boldsymbol{\mu}_k) - \psi_k(\hat{\boldsymbol{\mu}}_k)}{m_k(\boldsymbol{\mu}_k) - m_k(\hat{\boldsymbol{\mu}}_k)}$$

if $\rho_k \geq \eta_1$ **then** $\boldsymbol{\mu}_{k+1} = \hat{\boldsymbol{\mu}}_k$ **else** $\boldsymbol{\mu}_{k+1} = \boldsymbol{\mu}_k$ **end if**

4: **Trust region update**:

if $\rho_k \leq \eta_1$ **then** $\Delta_{k+1} \in (0, \gamma\,\|\hat{\boldsymbol{\mu}}_k - \boldsymbol{\mu}_k\|)]$ **end if**

if $\rho_k \in (\eta_1, \eta_2)$ **then** $\Delta_{k+1} \in [\gamma\,\|\hat{\boldsymbol{\mu}}_k - \boldsymbol{\mu}_k\|\,,\Delta_k]$ **end if**

if $\rho_k \geq \eta_2$ **then** $\Delta_{k+1} \in [\Delta_k, \Delta_{\max}]$ **end if**

Trust region ingredients for global convergence

Approximation models

$$m_k(\boldsymbol{\mu}),\ \psi_k(\boldsymbol{\mu})$$

Error indicators

$$\|\nabla F(\boldsymbol{\mu}) - \nabla m_k(\boldsymbol{\mu})\| \leq \xi \varphi_k(\boldsymbol{\mu}) \qquad \xi > 0$$

$$|F(\boldsymbol{\mu}_k) - F(\boldsymbol{\mu}) + \psi_k(\boldsymbol{\mu}) - \psi_k(\boldsymbol{\mu}_k)| \leq \sigma \theta_k(\boldsymbol{\mu}) \qquad \sigma > 0$$

Adaptivity

$$\varphi_k(\boldsymbol{\mu}_k) \leq \kappa_\varphi \min\{\|\nabla m_k(\boldsymbol{\mu}_k)\|\,,\ \Delta_k\}$$

$$\theta_k(\hat{\boldsymbol{\mu}}_k)^\omega \leq \eta \min\{m_k(\boldsymbol{\mu}_k) - m_k(\hat{\boldsymbol{\mu}}_k),\ r_k\}$$

Global convergence

$$\boxed{\liminf_{k\to\infty}\ \|\nabla F(\boldsymbol{\mu}_k)\| = 0}$$

Trust region method: ROM/SG approximation model

Approximation models built on two sources of inexactness

$$\begin{aligned} m_k(\boldsymbol{\mu}) = &\quad \mathbb{E}_{\mathcal{I}_k}\left[\mathcal{J}(\boldsymbol{\Phi}_k \boldsymbol{u}_r(\boldsymbol{\mu},\,\cdot\,),\,\boldsymbol{\mu},\,\cdot\,)\right] \\ \psi_k(\boldsymbol{\mu}) = &\quad \mathbb{E}_{\mathcal{I}'_k}\left[\mathcal{J}(\boldsymbol{\Phi}'_k \boldsymbol{u}_r(\boldsymbol{\mu},\,\cdot\,),\,\boldsymbol{\mu},\,\cdot\,)\right] \end{aligned}$$

Error indicators that account for both sources of error

$$\begin{aligned} \varphi_k(\boldsymbol{\mu}) &= \alpha_1 \mathcal{E}_1(\boldsymbol{\mu};\,\mathcal{I}_k,\,\boldsymbol{\Phi}_k) + \alpha_2 \mathcal{E}_2(\boldsymbol{\mu};\,\mathcal{I}_k,\,\boldsymbol{\Phi}_k) + \alpha_3 \mathcal{E}_4(\boldsymbol{\mu};\,\mathcal{I}_k,\,\boldsymbol{\Phi}_k) \\ \theta_k(\boldsymbol{\mu}) &= \beta_1(\mathcal{E}_1(\boldsymbol{\mu};\,\mathcal{I}'_k,\,\boldsymbol{\Phi}'_k) + \mathcal{E}_1(\boldsymbol{\mu}_k;\,\mathcal{I}'_k,\,\boldsymbol{\Phi}'_k)) + \beta_2(\mathcal{E}_3(\boldsymbol{\mu};\,\mathcal{I}'_k,\,\boldsymbol{\Phi}'_k) + \mathcal{E}_3(\boldsymbol{\mu}_k;\,\mathcal{I}'_k,\,\boldsymbol{\Phi}'_k)) \end{aligned}$$

Reduced-order model errors

$$\begin{aligned} \mathcal{E}_1(\boldsymbol{\mu};\,\mathcal{I},\,\boldsymbol{\Phi}) &= \mathbb{E}_{\mathcal{I}\,\cup\,\mathcal{N}(\mathcal{I})}\left[\|\boldsymbol{r}(\boldsymbol{\Phi}\boldsymbol{u}_r(\boldsymbol{\mu},\,\cdot),\,\boldsymbol{\mu},\,\cdot\,)\|\right] \\ \mathcal{E}_2(\boldsymbol{\mu};\,\mathcal{I},\,\boldsymbol{\Phi}) &= \mathbb{E}_{\mathcal{I}\,\cup\,\mathcal{N}(\mathcal{I})}\left[\left\|\boldsymbol{r}^{\boldsymbol{\lambda}}(\boldsymbol{\Phi}\boldsymbol{u}_r(\boldsymbol{\mu},\,\cdot),\,\boldsymbol{\Phi}\boldsymbol{\lambda}_r(\boldsymbol{\mu},\,\cdot\,),\,\boldsymbol{\mu},\,\cdot\,)\right\|\right] \end{aligned}$$

Sparse grid truncation errors

$$\begin{aligned} \mathcal{E}_3(\boldsymbol{\mu};\,\mathcal{I},\,\boldsymbol{\Phi}) &= \mathbb{E}_{\mathcal{N}(\mathcal{I})}\left[|\mathcal{J}(\boldsymbol{\Phi}\boldsymbol{u}_r(\boldsymbol{\mu},\,\cdot\,),\,\boldsymbol{\mu},\,\cdot\,)|\right] \\ \mathcal{E}_4(\boldsymbol{\mu};\,\mathcal{I},\,\boldsymbol{\Phi}) &= \mathbb{E}_{\mathcal{N}(\mathcal{I})}\left[\|\nabla\mathcal{J}(\boldsymbol{\Phi}\boldsymbol{u}_r(\boldsymbol{\mu},\,\cdot\,),\,\boldsymbol{\mu},\,\cdot\,)\|\right] \end{aligned}$$

Final requirement for convergence: Adaptivity

With the approximation model, $m_k(\boldsymbol{\mu})$, and gradient error indicator, $\varphi_k(\boldsymbol{\mu})$

$$
\begin{aligned}
m_k(\boldsymbol{\mu}) &= \ \mathbb{E}_{\mathcal{I}_k}\left[\mathcal{J}(\boldsymbol{\Phi}_k \boldsymbol{u}_r(\boldsymbol{\mu}, \cdot), \boldsymbol{\mu}, \cdot)\right] \\
\varphi_k(\boldsymbol{\mu}) &= \alpha_1 \mathcal{E}_1(\boldsymbol{\mu}; \mathcal{I}_k, \boldsymbol{\Phi}_k) + \alpha_2 \mathcal{E}_2(\boldsymbol{\mu}; \mathcal{I}_k, \boldsymbol{\Phi}_k) + \alpha_3 \mathcal{E}_4(\boldsymbol{\mu}; \mathcal{I}_k, \boldsymbol{\Phi}_k)
\end{aligned}
$$

the sparse grid $\mathcal{I}_k$ and reduced-order basis $\boldsymbol{\Phi}_k$ must be constructed such that the gradient condition holds

$$
\varphi_k(\boldsymbol{\mu}_k) \leq \kappa_\varphi \min\{\|\nabla m_k(\boldsymbol{\mu}_k)\|, \Delta_k\}
$$

Define dimension-adaptive greedy method to target each source of error such that the stronger conditions hold

$$
\begin{aligned}
\mathcal{E}_1(\boldsymbol{\mu}_k; \mathcal{I}, \boldsymbol{\Phi}) &\leq \frac{\kappa_\varphi}{3\alpha_1} \min\{\|\nabla m_k(\boldsymbol{\mu}_k)\|, \Delta_k\} \\
\mathcal{E}_2(\boldsymbol{\mu}_k; \mathcal{I}, \boldsymbol{\Phi}) &\leq \frac{\kappa_\varphi}{3\alpha_2} \min\{\|\nabla m_k(\boldsymbol{\mu}_k)\|, \Delta_k\} \\
\mathcal{E}_4(\boldsymbol{\mu}_k; \mathcal{I}, \boldsymbol{\Phi}) &\leq \frac{\kappa_\varphi}{3\alpha_3} \min\{\|\nabla m_k(\boldsymbol{\mu}_k)\|, \Delta_k\}
\end{aligned}
$$

Adaptivity: Dimension-adaptive greedy method

while $\mathcal{E}_4(\boldsymbol{\Phi}, \mathcal{I}, \boldsymbol{\mu}_k) > \frac{\kappa_\varphi}{3\alpha_3} \min\{\|\nabla m_k(\boldsymbol{\mu}_k)\|, \Delta_k\}$ **do**

Refine index set: Dimension-adaptive sparse grids

$$\mathcal{I}_k \leftarrow \mathcal{I}_k \cup \{\mathbf{j}^*\} \qquad \text{where} \qquad \mathbf{j}^* = \underset{\mathbf{j} \in \mathcal{N}(\mathcal{I}_k)}{\arg\max}\, \mathbb{E}_{\mathbf{j}}\left[\|\nabla \mathcal{J}(\boldsymbol{\Phi}\boldsymbol{u}_r(\boldsymbol{\mu}, \cdot), \boldsymbol{\mu}, \cdot)\|\right]$$

Adaptivity: Dimension-adaptive greedy method

while $\mathcal{E}_4(\boldsymbol{\Phi}, \mathcal{I}, \boldsymbol{\mu}_k) > \frac{\kappa_\varphi}{3\alpha_3} \min\{\|\nabla m_k(\boldsymbol{\mu}_k)\|, \Delta_k\}$ **do**

<u>**Refine index set**</u>: Dimension-adaptive sparse grids

$$\mathcal{I}_k \leftarrow \mathcal{I}_k \cup \{\mathbf{j}^*\} \qquad \text{where} \qquad \mathbf{j}^* = \underset{\mathbf{j}\in\mathcal{N}(\mathcal{I}_k)}{\arg\max}\ \mathbb{E}_{\mathbf{j}}\left[\|\nabla \mathcal{J}(\boldsymbol{\Phi}\boldsymbol{u}_r(\boldsymbol{\mu}, \cdot), \boldsymbol{\mu}, \cdot)\|\right]$$

<u>**Refine reduced-order basis**</u>: Greedy sampling

while $\mathcal{E}_1(\boldsymbol{\Phi}, \mathcal{I}, \boldsymbol{\mu}_k) > \frac{\kappa_\varphi}{3\alpha_1} \min\{\|\nabla m_k(\boldsymbol{\mu}_k)\|, \Delta_k\}$ **do**

$$\boldsymbol{\Phi}_k \leftarrow \begin{bmatrix} \boldsymbol{\Phi}_k & \boldsymbol{u}(\boldsymbol{\mu}_k, \boldsymbol{\xi}^*) & \boldsymbol{\lambda}(\boldsymbol{\mu}_k, \boldsymbol{\xi}^*) \end{bmatrix}$$

$$\boldsymbol{\xi}^* = \underset{\boldsymbol{\xi}\in\boldsymbol{\Xi}_{\mathbf{j}^*}}{\arg\max}\ \rho(\boldsymbol{\xi}) \left\|\boldsymbol{r}(\boldsymbol{\Phi}_k \boldsymbol{u}_r(\boldsymbol{\mu}_k, \boldsymbol{\xi}), \boldsymbol{\mu}_k, \boldsymbol{\xi})\right\|$$

end while

Adaptivity: Dimension-adaptive greedy method

while $\mathcal{E}_4(\boldsymbol{\Phi}, \mathcal{I}, \boldsymbol{\mu}_k) > \frac{\kappa_\varphi}{3\alpha_3} \min\{\|\nabla m_k(\boldsymbol{\mu}_k)\|, \Delta_k\}$ **do**

Refine index set: Dimension-adaptive sparse grids

$$\mathcal{I}_k \leftarrow \mathcal{I}_k \cup \{\mathbf{j}^*\} \qquad \text{where} \qquad \mathbf{j}^* = \underset{\mathbf{j}\in\mathcal{N}(\mathcal{I}_k)}{\arg\max}\ \mathbb{E}_{\mathbf{j}}\left[\|\nabla \mathcal{J}(\boldsymbol{\Phi}\boldsymbol{u}_r(\boldsymbol{\mu}, \cdot), \boldsymbol{\mu}, \cdot)\|\right]$$

Refine reduced-order basis: Greedy sampling

while $\mathcal{E}_1(\boldsymbol{\Phi}, \mathcal{I}, \boldsymbol{\mu}_k) > \frac{\kappa_\varphi}{3\alpha_1} \min\{\|\nabla m_k(\boldsymbol{\mu}_k)\|, \Delta_k\}$ **do**

$$\boldsymbol{\Phi}_k \leftarrow \begin{bmatrix} \boldsymbol{\Phi}_k & \boldsymbol{u}(\boldsymbol{\mu}_k, \boldsymbol{\xi}^*) & \boldsymbol{\lambda}(\boldsymbol{\mu}_k, \boldsymbol{\xi}^*) \end{bmatrix}$$

$$\boldsymbol{\xi}^* = \underset{\boldsymbol{\xi}\in\boldsymbol{\Xi}_{\mathbf{j}^*}}{\arg\max}\ \rho(\boldsymbol{\xi}) \left\|\boldsymbol{r}(\boldsymbol{\Phi}_k\boldsymbol{u}_r(\boldsymbol{\mu}_k, \boldsymbol{\xi}), \boldsymbol{\mu}_k, \boldsymbol{\xi})\right\|$$

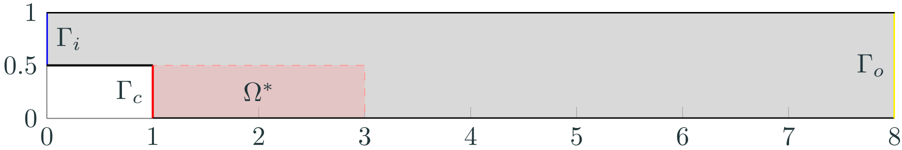
end while

while $\mathcal{E}_2(\boldsymbol{\Phi}, \mathcal{I}, \boldsymbol{\mu}_k) > \frac{\kappa_\varphi}{3\alpha_2} \min\{\|\nabla m_k(\boldsymbol{\mu}_k)\|, \Delta_k\}$ **do**

$$\boldsymbol{\Phi}_k \leftarrow \begin{bmatrix} \boldsymbol{\Phi}_k & \boldsymbol{u}(\boldsymbol{\mu}_k, \boldsymbol{\xi}^*) & \boldsymbol{\lambda}(\boldsymbol{\mu}_k, \boldsymbol{\xi}^*) \end{bmatrix}$$

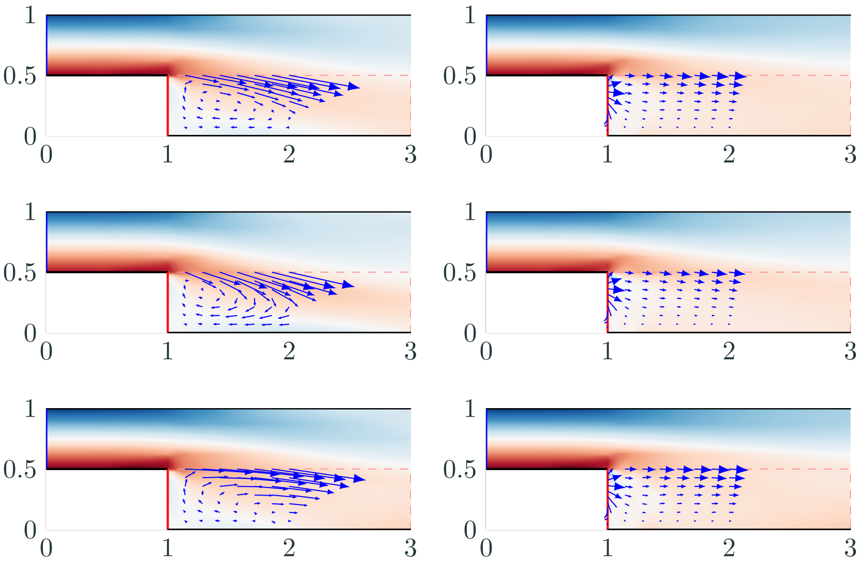
$$\boldsymbol{\xi}^* = \underset{\boldsymbol{\xi}\in\boldsymbol{\Xi}_{\mathbf{j}^*}}{\arg\max}\ \rho(\boldsymbol{\xi}) \left\|\boldsymbol{r}^{\boldsymbol{\lambda}}(\boldsymbol{\Phi}_k\boldsymbol{u}_r(\boldsymbol{\mu}_k, \boldsymbol{\xi}), \boldsymbol{\Phi}_k\boldsymbol{\lambda}_r(\boldsymbol{\mu}_k, \boldsymbol{\xi}), \boldsymbol{\mu}_k, \boldsymbol{\xi})\right\|$$

Optimal boundary control of incompressible Navier-Stokes



Geometry and boundary conditions for backward facing step. Boundary conditions: viscous wall (—), parametrized inflow (—), stochastic inflow (—), outflow (—). Vorticity magnitude minimized in red shaded region.

Optimal boundary control and statistics



The mean flow $\bar{u}(x, \boldsymbol{\mu})$ (*top*) and standard deviation offsets $\bar{u}_-(x, \boldsymbol{\mu})$ (*center*), $\bar{u}_+(x, \boldsymbol{\mu})$ (*bottom*) corresponding to the uncontrolled, $\boldsymbol{\mu} = 0$, (*left*) and controlled flow (*right*). Boundary control along Γ_c effectively eliminates the re-circulation region.

Global convergence without pointwise agreement

$F(\boldsymbol{\mu}_k)$	$m_k(\boldsymbol{\mu}_k)$	$F(\hat{\boldsymbol{\mu}}_k)$	$m_k(\hat{\boldsymbol{\mu}}_k)$	$\lVert\nabla F(\boldsymbol{\mu}_k)\rVert$	ρ_k	Success?
1.0740e+00	1.0805e+00	8.4412e-01	8.6172e-01	1.8723e+00	1.0000e+00	1.0000e+00
8.4412e-01	8.4351e-01	7.4896e-01	7.4628e-01	1.3292e+00	1.0000e+00	1.0000e+00
7.4896e-01	7.3757e-01	7.3766e-01	7.2654e-01	3.3224e-01	8.6570e-01	1.0000e+00
7.3766e-01	7.3429e-01	7.3601e-01	7.3204e-01	1.1425e-01	7.3229e-01	1.0000e+00
7.3601e-01	7.3250e-01	7.3548e-01	7.3207e-01	7.9688e-02	1.2288e+00	1.0000e+00
7.3548e-01	7.3207e-01	-	-	1.4001e-02	-	-

Convergence history of trust region method built on two-level approximation

One to two order of magnitude reduction in HDM evaluations

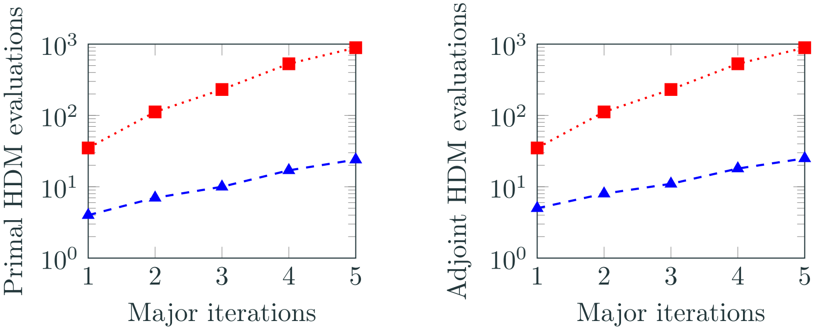
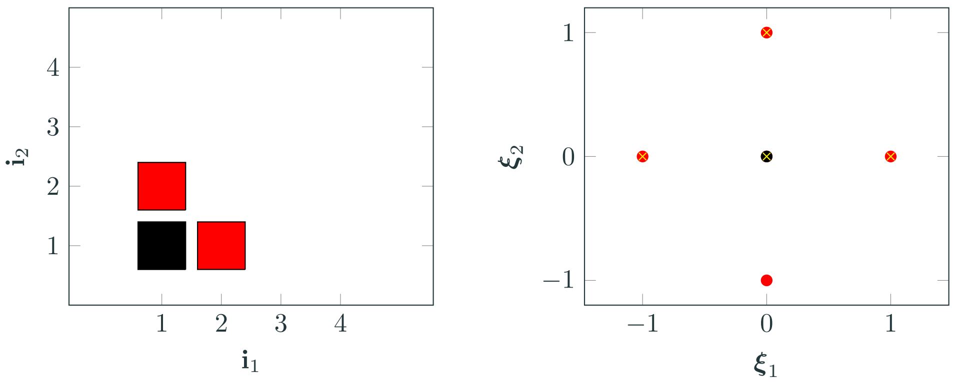
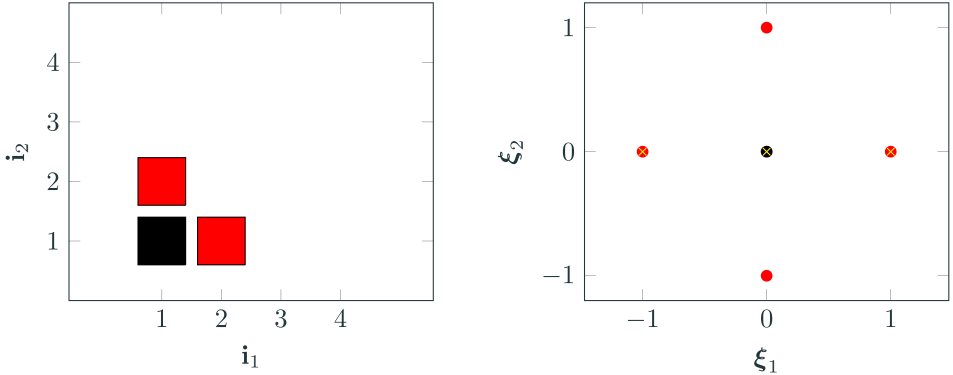


Figure 3: *Cumulative* number of HDM primal and adjoint evaluations as the major iterations in the various trust region algorithms progress: dimension-adaptive sparse grid [Kouri et al., 2014] (···■···) and proposed method (- ▲ -).

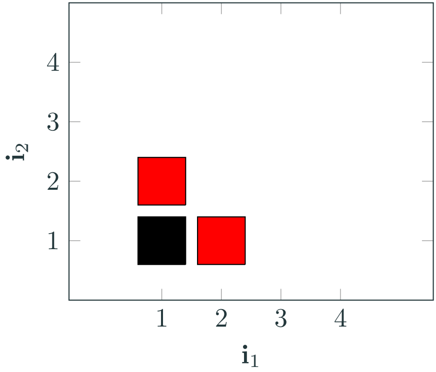
Adaptation of sparse grid and reduced basis

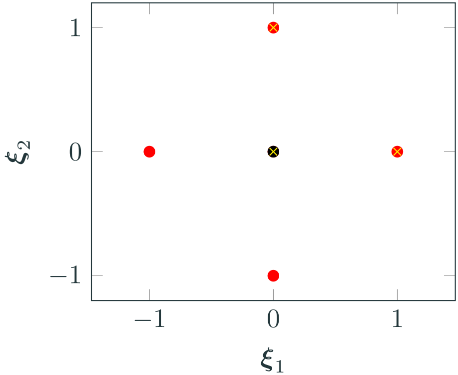


Adaptation of sparse grid and reduced basis

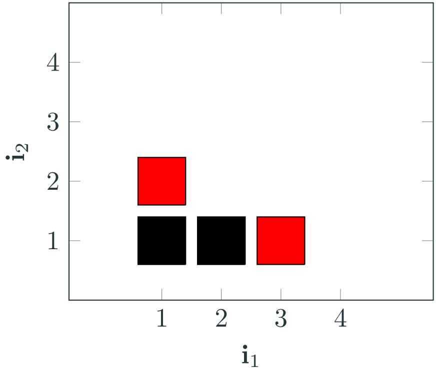


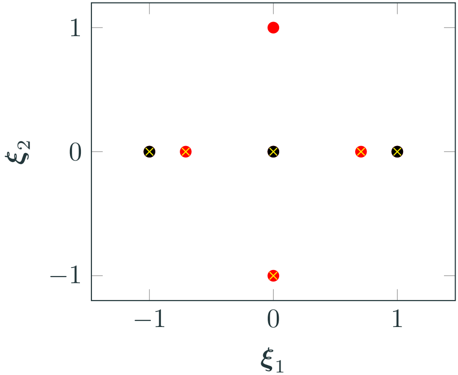
Adaptation of sparse grid and reduced basis



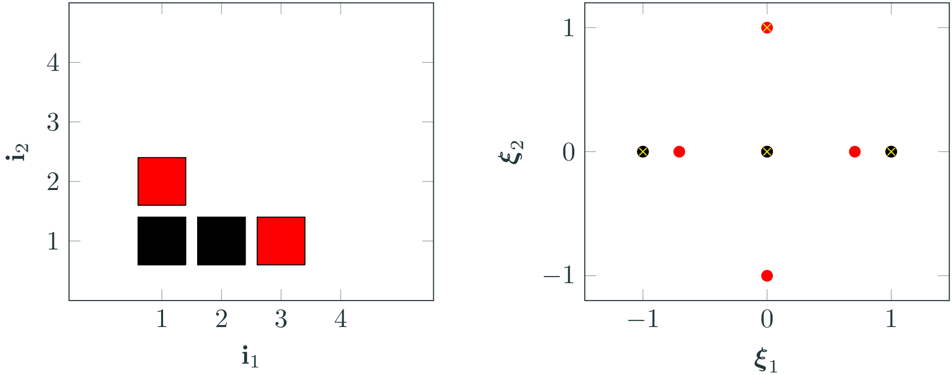


Adaptation of sparse grid and reduced basis





Adaptation of sparse grid and reduced basis



Adaptation of sparse grid and reduced basis