Integrated computational physics and numerical optimization

Matthew J. Zahr
Luis W. Alvarez Postdoctoral Fellow
Mathematics Group
Computational Research Division
Lawrence Berkeley National Laboratory

UCB/LBNL Applied Mathematics Seminar
University of California, Berkeley, CA
September 6, 2018

Collaborators: Daniel Huang, Per-Olof Persson, Johannes Töger, Jingyi Wang

Integrating computational physics and numerical optimization

Optimize physics

Optimize numerics

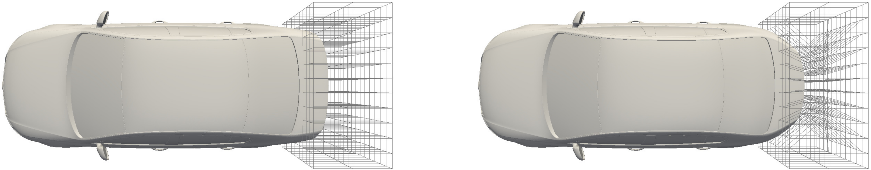
Integrating computational physics and numerical optimization

Optimize physics

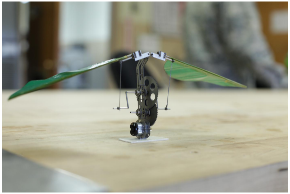
Optimize numerics

PDE optimization is ubiquitous in science and engineering

Design: Find system that optimizes performance metric, satisfies constraints



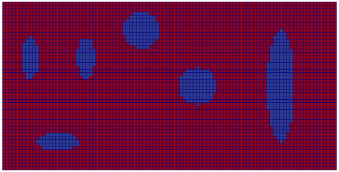
Aerodynamic shape design of automobile

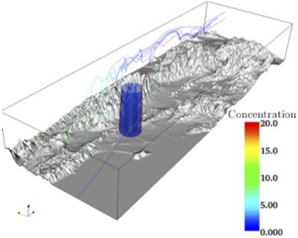


Optimal flapping motion of micro aerial vehicle

PDE optimization is ubiquitous in science and engineering

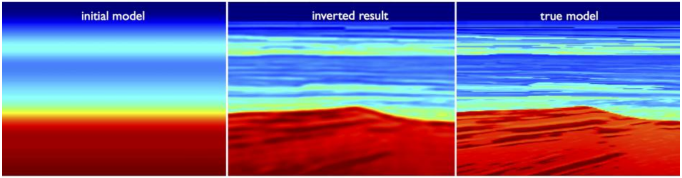
Inverse problems: Infer the problem setup given solution observations





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

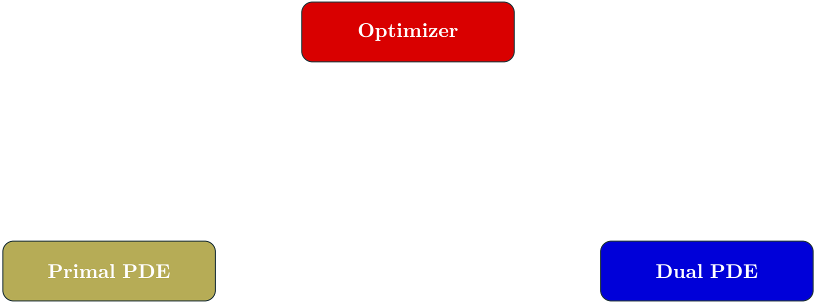
Unsteady PDE-constrained optimization formulation

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

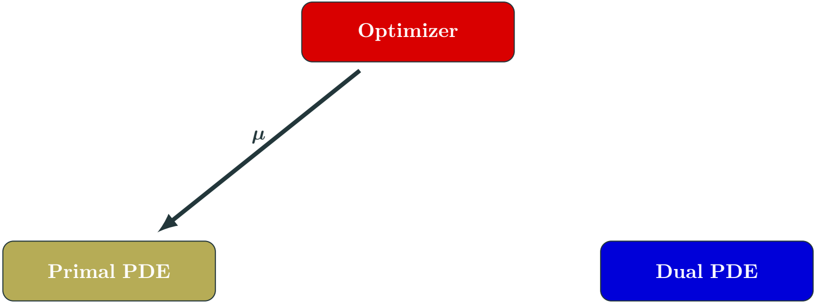
$$\begin{aligned} \underset{\boldsymbol{U},\ \boldsymbol{\mu}}{\text{minimize}} \quad & \mathcal{J}(\boldsymbol{U}, \boldsymbol{\mu}) \\ \text{subject to} \quad & \boldsymbol{C}(\boldsymbol{U}, \boldsymbol{\mu}) \leq 0 \\ & \frac{\partial \boldsymbol{U}}{\partial t} + \nabla \cdot \boldsymbol{F}(\boldsymbol{U},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

$\boldsymbol{U}(\boldsymbol{x}, t)$	PDE solution
$\boldsymbol{\mu}$	design/control parameters
$\mathcal{J}(\boldsymbol{U}, \boldsymbol{\mu}) = \int_{T_0}^{T_f} \int_{\boldsymbol{\Gamma}} j(\boldsymbol{U}, \boldsymbol{\mu}, t) \, dS \, dt$	objective function
$\boldsymbol{C}(\boldsymbol{U}, \boldsymbol{\mu}) = \int_{T_0}^{T_f} \int_{\boldsymbol{\Gamma}} \mathbf{c}(\boldsymbol{U}, \boldsymbol{\mu}, t) \, dS \, dt$	constraints

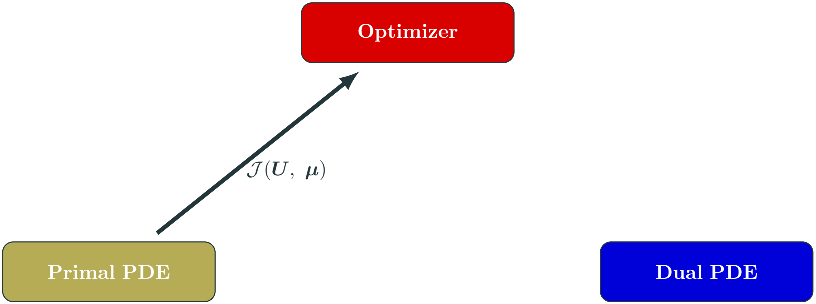
Nested approach to PDE-constrained optimization



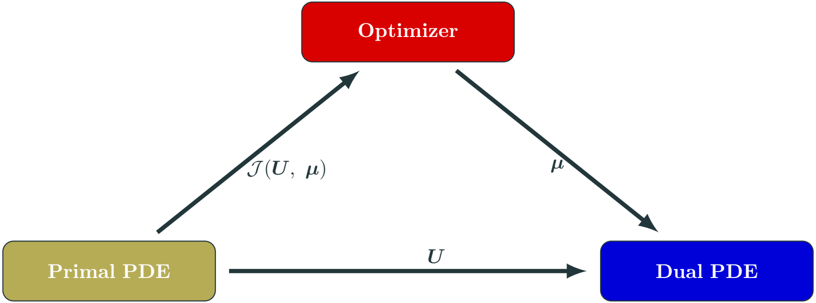
Nested approach to PDE-constrained optimization



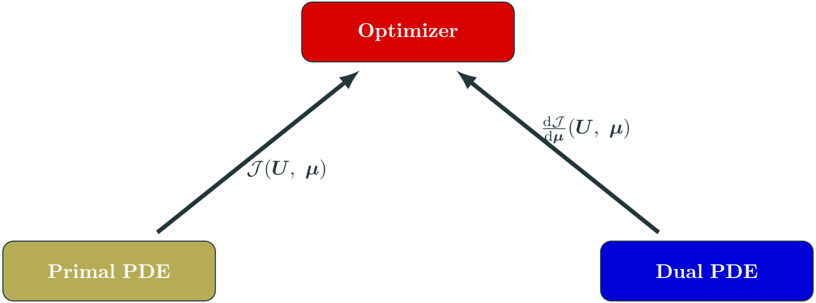
Nested approach to PDE-constrained optimization



Nested approach to PDE-constrained optimization



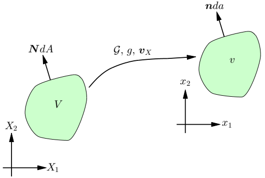
Nested approach to PDE-constrained optimization



Highlights of globally high-order discretization

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

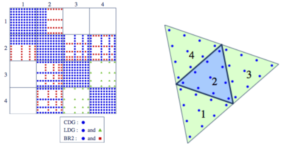
$$\left.\frac{\partial \boldsymbol{U}_{\boldsymbol{X}}}{\partial t}\right|_{\boldsymbol{X}} + \nabla_{\boldsymbol{X}} \cdot \boldsymbol{F}_{\boldsymbol{X}}(\boldsymbol{U}_{\boldsymbol{X}},\ \nabla_{\boldsymbol{X}} \boldsymbol{U}_{\boldsymbol{X}}) = 0$$



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]

Optimal rigid body motion (RBM), time-morph geometry (TMG)

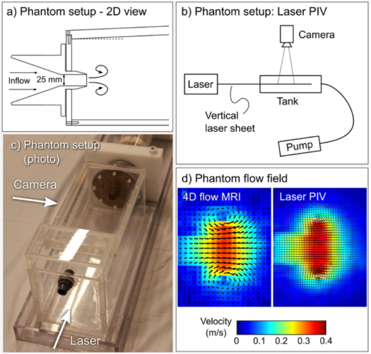
Initial Guess	Optimal RBM	Optimal RBM/TMG
Energy $= 9.4096$	Energy $= 4.9476$	Energy $= 4.6182$
Thrust $= 0.1766$	Thrust $= 2.500$	Thrust $= 2.500$
	$T_x = 2.5$	$T_x = 2.5$

Energetically optimal flapping in three dimensions

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

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

Super-resolution MR images through optimization



Experimental setup

Noisy, low-resolution MRI data

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

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

MRI optimization formulation that respects scanner physics

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

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

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

MRI optimization formulation that respects scanner physics

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

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

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

$$\boldsymbol{d}_{i,n}(\boldsymbol{U},\, \boldsymbol{\mu}) = \int_0^T \int_V w_{i,n}(\boldsymbol{x},\, t) \cdot \boldsymbol{U}(\boldsymbol{x},\, t)\, dV\, dt$$

$$w_{i,n}(\boldsymbol{x},\, t) = \chi_s(\boldsymbol{x};\, \boldsymbol{x}_i,\, \Delta\boldsymbol{x})\chi_t(t;\, t_n,\, \Delta t)$$

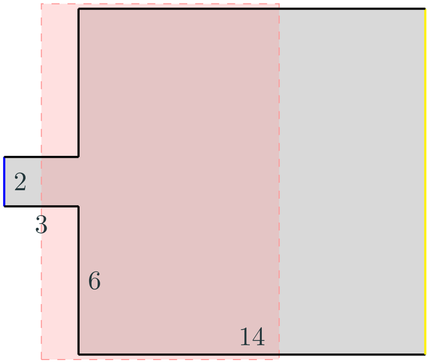
$$\chi_t(s;\, c,\, w) = \frac{1}{1 + e^{-(s-(c-0.5w))/\sigma}} - \frac{1}{1 + e^{-(s-(c+0.5w))/\sigma}}$$

$$\chi_s(\boldsymbol{x};\, \boldsymbol{c},\, \boldsymbol{w}) = \chi_t(x_1;\, c_1,\, w_1)\chi_t(x_2;\, c_2,\, w_2)\chi_t(x_3;\, c_3,\, w_3)$$

$\boldsymbol{x}_i$ 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

Model problem with synthetic data



Viscous wall (——), parametrized inflow (——), and outflow (——).

MRI data collected in the red shaded region.

High-quality reconstruction from coarse MRI grid (space: 24×36, time: 10) and low noise (3%)

Reconstructed flow

Synthetic MRI data $\boldsymbol{d}^*_{i,n}$ (top) and computational representation of MRI data $\boldsymbol{d}_{i,n}$ (bottom)

High-quality reconstruction from fine MRI grid (space: 40×60, time: 20) and high noise (10%)

Reconstructed flow

Synthetic MRI data $\boldsymbol{d}^*_{i,n}$ (top) and computational representation of MRI data $\boldsymbol{d}_{i,n}$ (bottom)

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

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 $T_x = 2.5$

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

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

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

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

Adjoint equations inherit explicit-implicit structure

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

Particle-laden flow

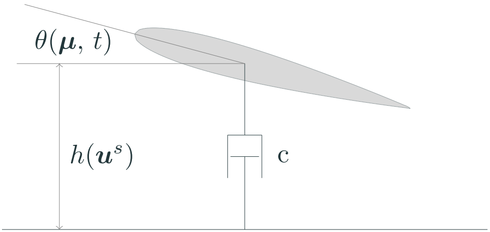
Fluid-structure interaction

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* $\theta(\boldsymbol{\mu},\, t) = \mu_1 \cos(2\pi f t)$



$\mu_1^* \approx 45°$

High-order methods for PDE-constrained optimization

- Developed **fully discrete adjoint method** for **high-order** numerical discretizations of PDEs and QoIs
- Used to compute **gradients** of QoI for use in gradient-based numerical optimization method
- Treatment of **parametrized time domain** (optimal frequency)
- Explicit enforcement of **time-periodicity constraints**
- Extension to **multiphysics** (fluid-structure interaction, particle-laden flow, ...)
- **Applications**: optimal flapping flight, energy harvesting, data assimilation

Integrating computational physics and numerical optimization

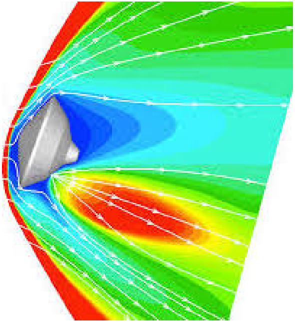
Optimize physics

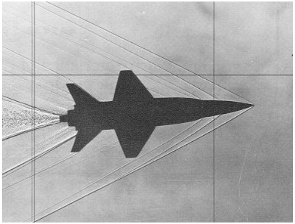
Optimize numerics

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

Supersnoic and transonic flow around commercial planes and fighter jets

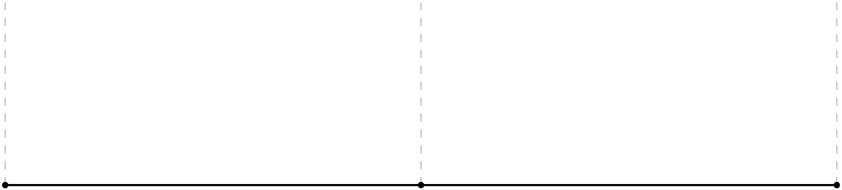
Hypersonics, e.g., re-entry of vehicles in atmosphere, and scramjets





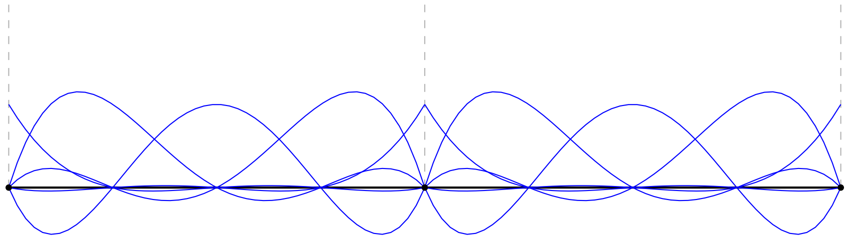
Other applications with discontinuities: fracture, problems with interfaces

State-of-the-art numerical methods for resolving shocks



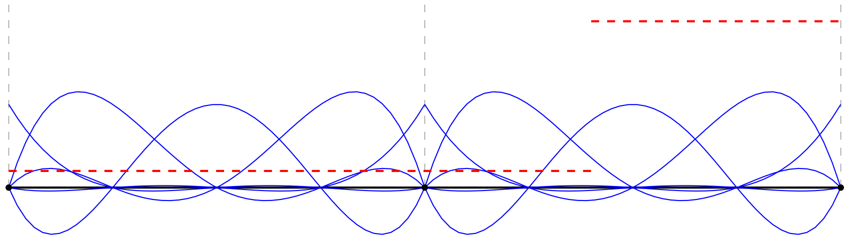
Fundamental issue: approximate discontinuity with polynomial basis

State-of-the-art numerical methods for resolving shocks



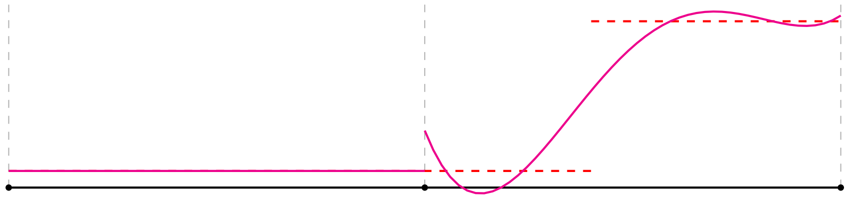
Fundamental issue: approximate discontinuity with polynomial basis

State-of-the-art numerical methods for resolving shocks



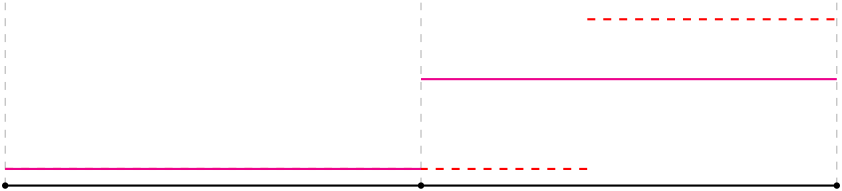
Fundamental issue: approximate discontinuity with polynomial basis

State-of-the-art numerical methods for resolving shocks



<u>Fundamental issue</u>: approximate discontinuity with polynomial basis

State-of-the-art numerical methods for resolving shocks

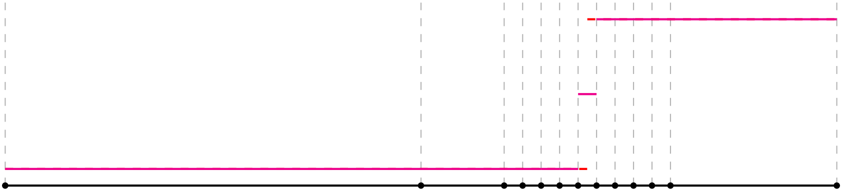


Fundamental issue: approximate discontinuity with polynomial basis

Exising solutions: **limiting**, artificial viscosity

Drawbacks: order reduction, local refinement

State-of-the-art numerical methods for resolving shocks

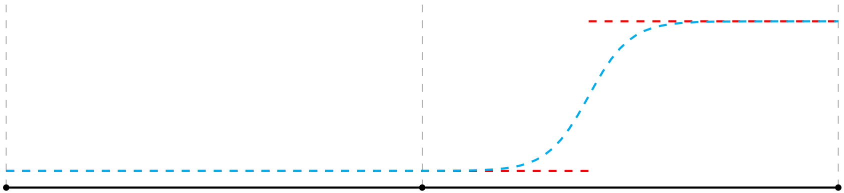


Fundamental issue: approximate discontinuity with polynomial basis

Exising solutions: **limiting**, artificial viscosity

Drawbacks: order reduction, local refinement

State-of-the-art numerical methods for resolving shocks

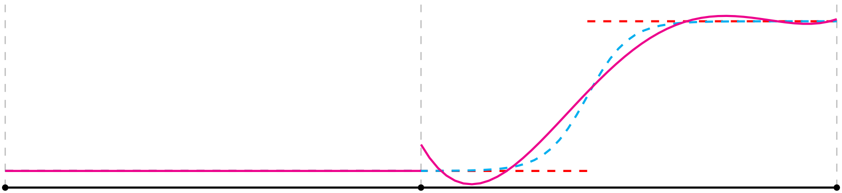


Fundamental issue: approximate discontinuity with polynomial basis

Exising solutions: limiting, **artificial viscosity**

Drawbacks: order reduction, local refinement

State-of-the-art numerical methods for resolving shocks

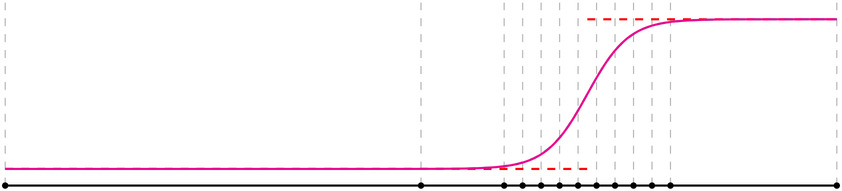


Fundamental issue: approximate discontinuity with polynomial basis

Exising solutions: limiting, **artificial viscosity**

Drawbacks: order reduction, local refinement

State-of-the-art numerical methods for resolving shocks

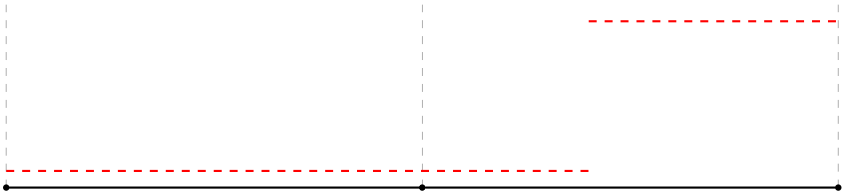


Fundamental issue: approximate discontinuity with polynomial basis

Exising solutions: limiting, **artificial viscosity**

Drawbacks: order reduction, local refinement

State-of-the-art numerical methods for resolving shocks



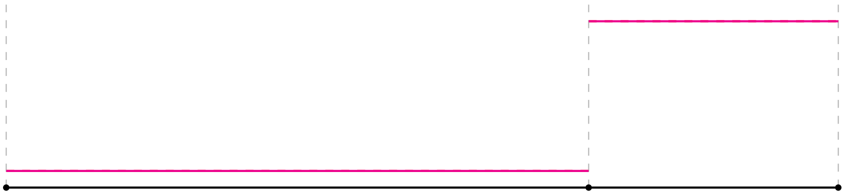
Fundamental issue: approximate discontinuity with polynomial basis

Exising solutions: limiting, artificial viscosity

Drawbacks: order reduction, local refinement

Proposed solution: align features of solution basis with features in the solution using optimization formulation and solver

State-of-the-art numerical methods for resolving shocks



Fundamental issue: approximate discontinuity with polynomial basis

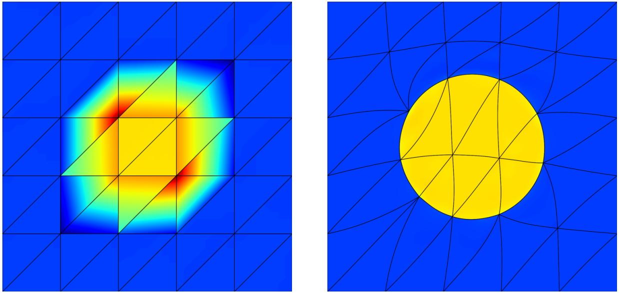
Exising solutions: limiting, artificial viscosity

Drawbacks: order reduction, local refinement

Proposed solution: align features of solution basis with features in the solution using optimization formulation and solver

Tracking method for stable, high-order resolution of discontinuities

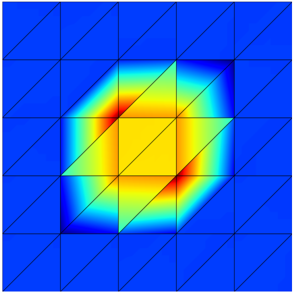
Goal: Align element faces with (unknown) discontinuities to perfectly capture them and approximate smooth regions to high-order



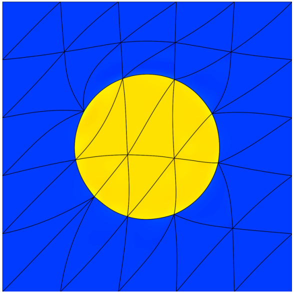
Non-aligned Discontinuity-aligned

Tracking method 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

Ingredients

- Discontinuous Galerkin discretization: inter-element jumps, high-order
- Optimization formulation that penalizes local instabilities in the solution and enforces the discrete PDE
- Full space solver that converges the solution and mesh simultaneously to ensure solution of PDE never required on non-aligned mesh

Discontinuity-tracking as PDE-constrained optimization problem

$$\begin{aligned} \underset{\boldsymbol{u},\,\boldsymbol{x}}{\text{minimize}} \quad & f(\boldsymbol{u},\,\boldsymbol{x}) \\ \text{subject to} \quad & \boldsymbol{r}(\boldsymbol{u},\,\boldsymbol{x}) = 0 \end{aligned}$$

Discontinuity-tracking as PDE-constrained optimization problem

$$\begin{array}{ll} \underset{\boldsymbol{u},\, \boldsymbol{x}}{\text{minimize}} & f(\boldsymbol{u},\, \boldsymbol{x}) \\ \text{subject to} & \boldsymbol{r}(\boldsymbol{u},\, \boldsymbol{x}) = 0 \end{array}$$

Objective function

Must **obtain minimum** when mesh face aligned with shock and **monotonically** decreases to minimum in neighborhood of radius $\mathcal{O}(h/2)$ about discontinuity

Discontinuity-tracking as PDE-constrained optimization problem

$$\begin{aligned} \underset{\boldsymbol{u},\,\boldsymbol{x}}{\text{minimize}} \quad & f(\boldsymbol{u},\,\boldsymbol{x}) \\ \text{subject to} \quad & \boldsymbol{r}(\boldsymbol{u},\,\boldsymbol{x}) = 0 \end{aligned}$$

Objective function

Must **obtain minimum** when mesh face aligned with shock and **monotonically** decreases to minimum in neighborhood of radius $\mathcal{O}(h/2)$ about discontinuity

Optimization approach

Cannot use **nested** approach where constraint $\boldsymbol{r}(\boldsymbol{u},\,\boldsymbol{x}) = 0$ is eliminated because discrete PDE cannot be solved unless $\boldsymbol{x} = \boldsymbol{x}^*$ $\implies$ **full space** approach required

Transformed conservation law from deformation of physical domain

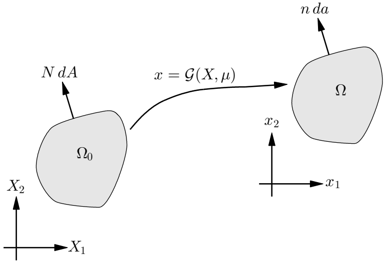
Consider physical domain as the result of a μ-parametrized diffeomorphism applied to some reference domain Ω_0

$$\Omega = \mathcal{G}(\Omega_0, \mu)$$

Re-write conservation law on reference domain

$$\nabla \cdot \mathcal{F}(U) = 0 \quad \text{in} \;\; \mathcal{G}(\Omega_0, \mu) \qquad \Longrightarrow \qquad \nabla_X \cdot F(u,\, \mu) = 0 \quad \text{in} \;\; \Omega_0,$$

$$u = g_\mu U, \quad F(u,\, \mu) = g_\mu \mathcal{F}(g_\mu^{-1} u) G_\mu^{-T}, \quad G_\mu = \frac{\partial}{\partial X}\mathcal{G}(X, \mu), \quad g_\mu = \det G_\mu$$



Mapping between reference and physical domains

Discontinuous Galerkin discretization of conservation law

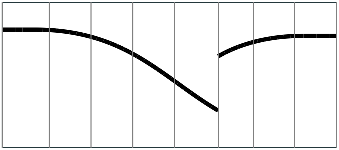
Element-wise weak form of transformed conservation law

$$\int_{\partial K} \psi \cdot F(u,\, \mu) N \, dA - \int_K F(u,\, \mu) : \nabla_X \psi \, dV = 0$$

Global weak form and introduction of numerical flux

$$\sum_{K \in \mathcal{E}_{h,p}} \int_{\partial K} \psi \cdot F^*(u,\, \mu,\, N) \, dA - \int_{\Omega_0} F(u,\, \mu) : \nabla_X \psi \, dV = 0$$

Strict requirements on numerical flux since inter-element jumps will not tend to zero on shock surface



Fully discrete transformed conservation law in terms of the discrete state vector $\boldsymbol{u}$ and coordinates of physical mesh $\boldsymbol{x}$

$$\boldsymbol{r}(\boldsymbol{u},\, \boldsymbol{x}) = 0$$

Objective function: penalize oscillations and mesh distortion

Consider a discontinuity indicator that aims to penalize oscillations in finite-dimensional solution

$$f_{shk}(\boldsymbol{u},\, \boldsymbol{x}) = h_0^{-2} \sum_{K \in \mathcal{E}_{h,p}} \int_{\mathcal{G}(K,\boldsymbol{x})} \left|\left| u_{h,p} - \bar{u}_{h,p}^{K} \right|\right|_{\boldsymbol{W}}^{2} \, dV,$$

$$\bar{u}_{h,p}^{K} = \frac{1}{|\mathcal{G}(K,\, \boldsymbol{x})|} \int_{\mathcal{G}(K,\boldsymbol{x})} u_{h,p}\, dV, \qquad |\mathcal{G}(K,\, \boldsymbol{x})| = \int_{\mathcal{G}(K,\boldsymbol{x})} dV, \qquad h_0 = |\Omega_0|^{1/d}$$

Objective function: penalize oscillations and mesh distortion

Consider a discontinuity indicator that aims to penalize oscillations in finite-dimensional solution

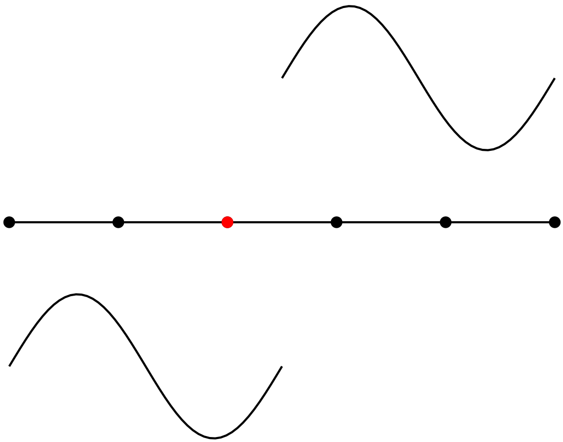
$$f_{shk}(\boldsymbol{u}, \boldsymbol{x}) = h_0^{-2} \sum_{K \in \mathcal{E}_{h,p}} \int_{\mathcal{G}(K, \boldsymbol{x})} \left|\left| u_{h,p} - \bar{u}_{h,p}^K \right|\right|_{\boldsymbol{W}}^2 \, dV,$$

$$\bar{u}_{h,p}^K = \frac{1}{|\mathcal{G}(K, \boldsymbol{x})|} \int_{\mathcal{G}(K, \boldsymbol{x})} u_{h,p} \, dV, \qquad |\mathcal{G}(K, \boldsymbol{x})| = \int_{\mathcal{G}(K, \boldsymbol{x})} dV, \qquad h_0 = |\Omega_0|^{1/d}$$

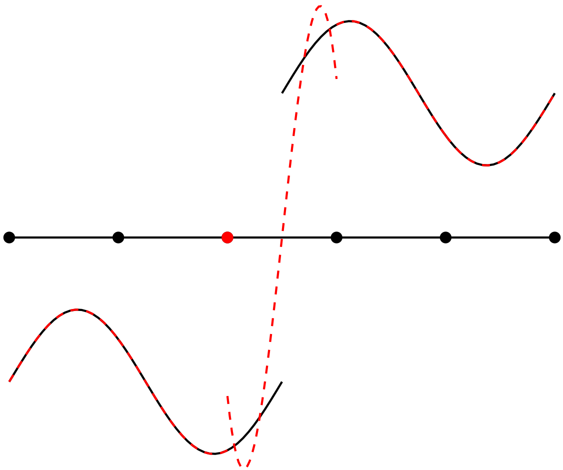
Construct objective function as weighted combination between discontinuity indicator and mesh distortion metric

$$f(\boldsymbol{u}, \boldsymbol{x}; \alpha) = f_{shk}(\boldsymbol{u}, \boldsymbol{x}) + \alpha f_{msh}(\boldsymbol{x})$$

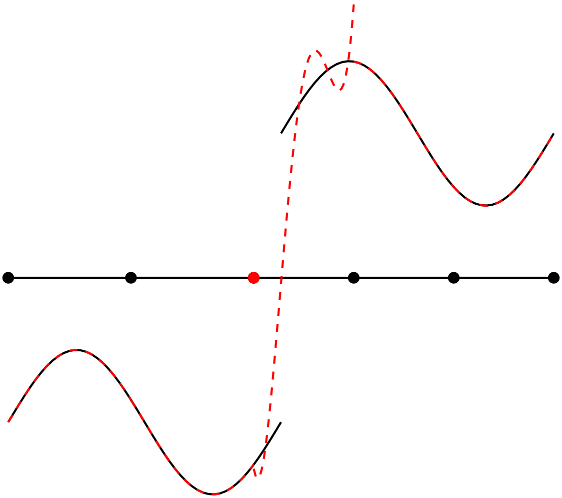
One-dimensional mesh parametrization and objective function test



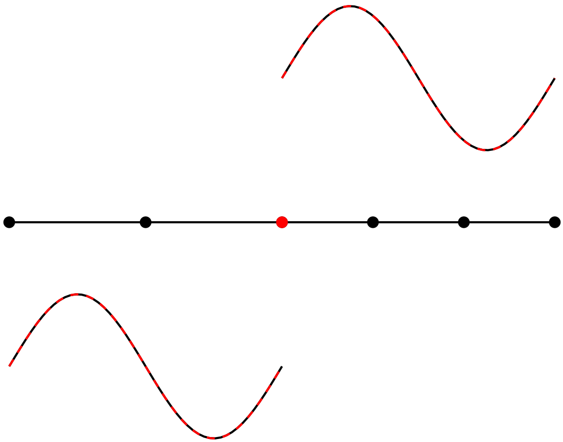
One-dimensional mesh parametrization and objective function test



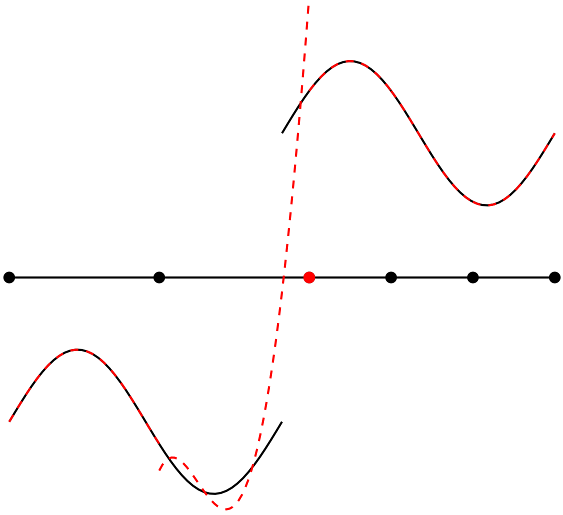
One-dimensional mesh parametrization and objective function test



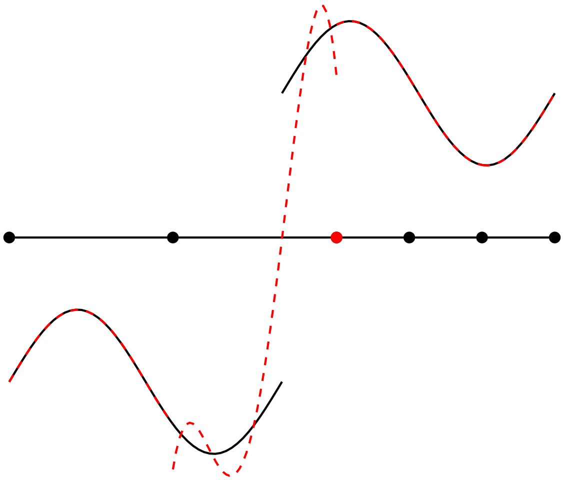
One-dimensional mesh parametrization and objective function test



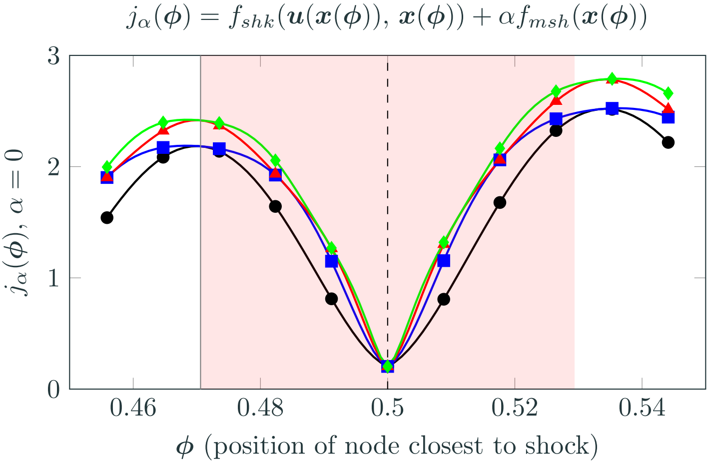
One-dimensional mesh parametrization and objective function test



One-dimensional mesh parametrization and objective function test

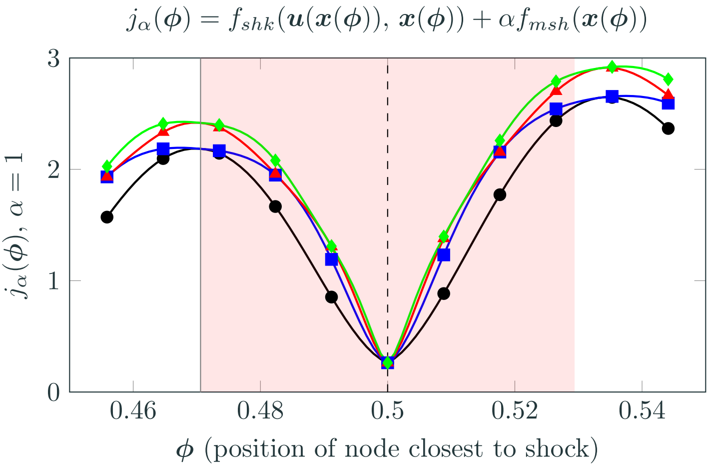


Objective function monotonically approaches minimum as mesh aligns with discontinuity, regardless of p, for a range of α



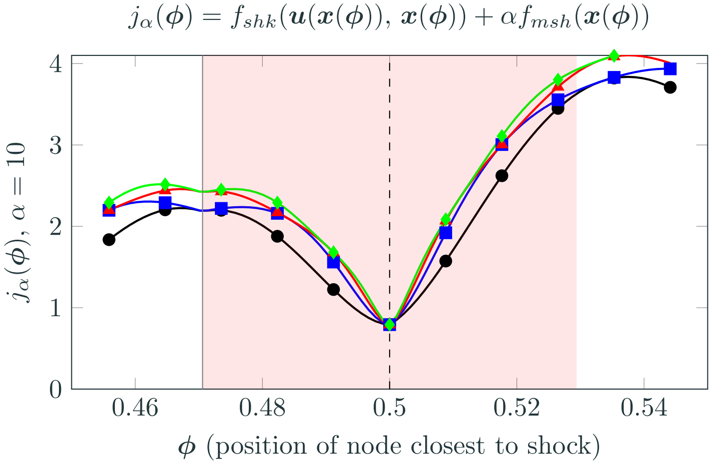
Objective function as an element face is smoothly swept across discontinuity (- - -): $p = 1$ (—●—), $p = 2$ (—■—), $p = 3$ (—▲—), $p = 4$ (—◆—).

Objective function monotonically approaches minimum as mesh aligns with discontinuity, regardless of p, for a range of α



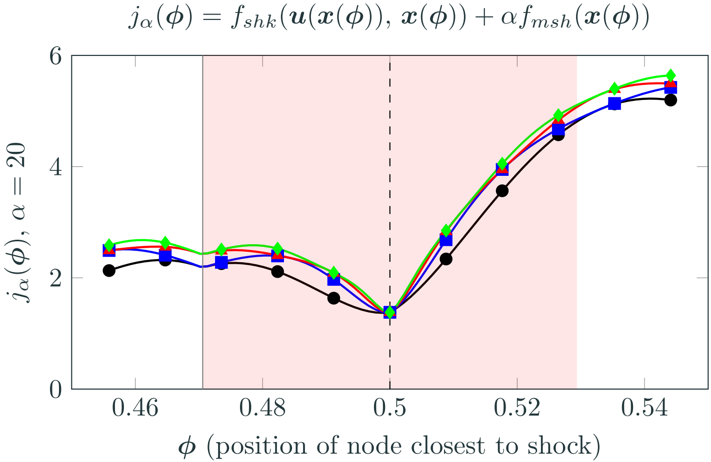
Objective function as an element face is smoothly swept across discontinuity (- - -):
$p = 1$ (—●—), $p = 2$ (—■—), $p = 3$ (—▲—), $p = 4$ (—◆—).

Objective function monotonically approaches minimum as mesh aligns with discontinuity, regardless of p, for a range of α



Objective function as an element face is smoothly swept across discontinuity (- - -): $p = 1$ (—●—), $p = 2$ (—■—), $p = 3$ (—▲—), $p = 4$ (—◆—).

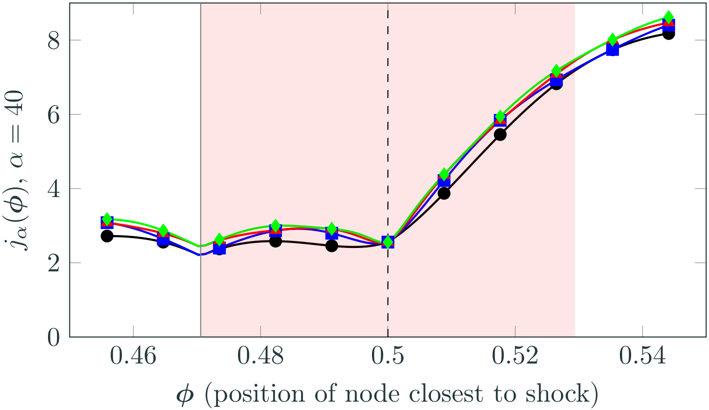
Objective function monotonically approaches minimum as mesh aligns with discontinuity, regardless of p, for a range of α



Objective function as an element face is smoothly swept across discontinuity (- - -): $p = 1$ (—●—), $p = 2$ (—■—), $p = 3$ (—▲—), $p = 4$ (—◆—).

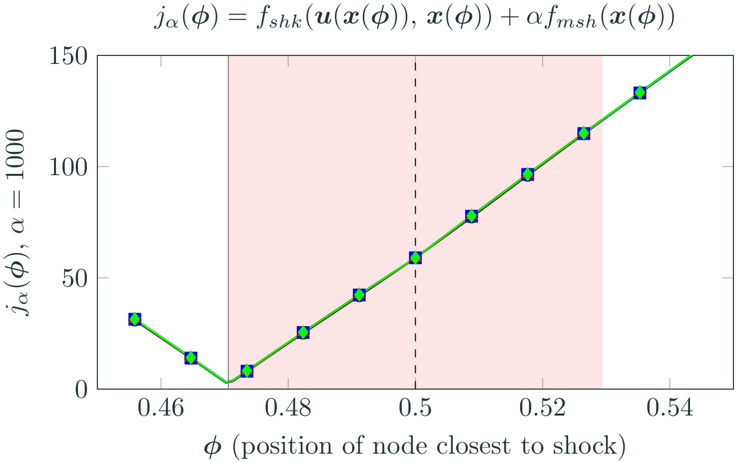
Objective function monotonically approaches minimum as mesh aligns with discontinuity, regardless of p, for a range of α

$$j_\alpha(\boldsymbol{\phi}) = f_{shk}(\boldsymbol{u}(\boldsymbol{x}(\boldsymbol{\phi})), \boldsymbol{x}(\boldsymbol{\phi})) + \alpha f_{msh}(\boldsymbol{x}(\boldsymbol{\phi}))$$



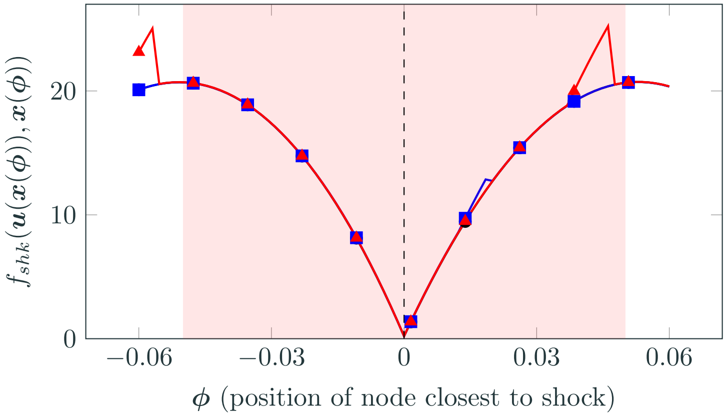
Objective function as an element face is smoothly swept across discontinuity (- - -): $p = 1$ (—●—), $p = 2$ (—■—), $p = 3$ (—▲—), $p = 4$ (—◆—).

Objective function monotonically approaches minimum as mesh aligns with discontinuity, regardless of p, for a range of α



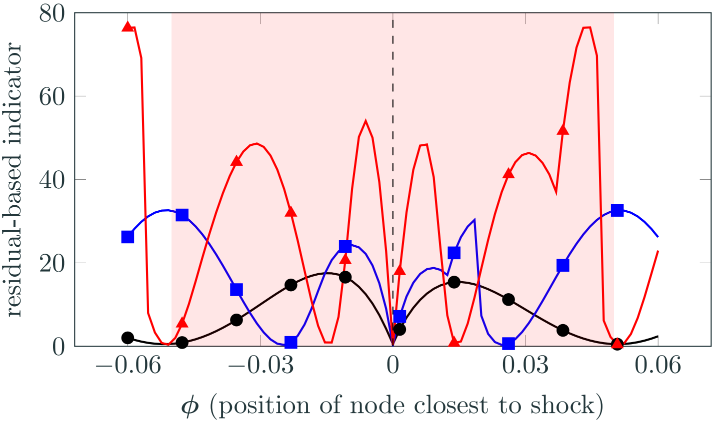
Objective function as an element face is smoothly swept across discontinuity (- - -): $p = 1$ (—●—), $p = 2$ (—■—), $p = 3$ (—▲—), $p = 4$ (—◆—).

Proposed discontinuity indicator is monotonic and attains minimum at discontinuity, whereas other indicators are not monotonic



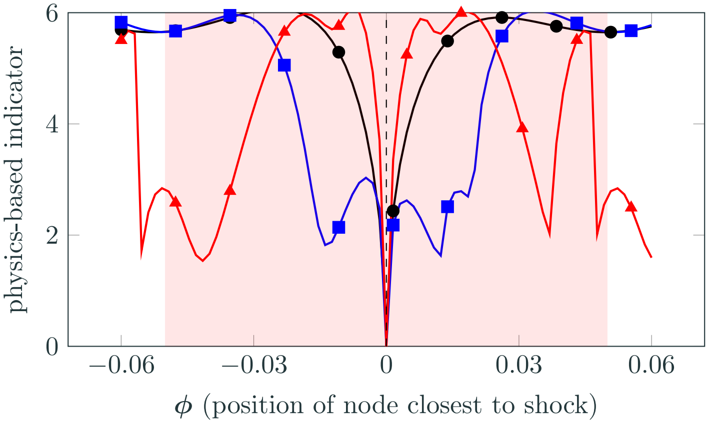
Objective function as an element face is smoothly swept across discontinuity (- - -): $p = 1$ (—●—), $p = 2$ (—■—), $p = 3$ (—▲—).

Proposed discontinuity indicator is monotonic and attains minimum at discontinuity, whereas other indicators are not monotonic



Objective function as an element face is smoothly swept across discontinuity (- - -): $p = 1$ (—●—), $p = 2$ (—■—), $p = 3$ (—▲—).

Proposed discontinuity indicator is monotonic and attains minimum at discontinuity, whereas other indicators are not monotonic



Objective function as an element face is smoothly swept across discontinuity (- - -): $p = 1$ (—●—), $p = 2$ (—■—), $p = 3$ (—▲—).

Cannot use nested approach to PDE optimization because it requires solving $\boldsymbol{r}(\boldsymbol{u}, \boldsymbol{x}) = 0$ for $\boldsymbol{x} \neq \boldsymbol{x}^*$ $\implies$ crash

Full space approach: $\boldsymbol{u} \rightarrow \boldsymbol{u}^*$ and $\boldsymbol{x} \rightarrow \boldsymbol{x}^*$ *simultaneously*

[1] usually requires globalization such as linesearch or trust-region

Cannot use nested approach to PDE optimization because it requires solving $\boldsymbol{r}(\boldsymbol{u}, \boldsymbol{x}) = 0$ for $\boldsymbol{x} \neq \boldsymbol{x}^*$ $\implies$ crash

Full space approach: $\boldsymbol{u} \rightarrow \boldsymbol{u}^*$ and $\boldsymbol{x} \rightarrow \boldsymbol{x}^*$ *simultaneously*

Define Lagrangian

$$\mathcal{L}(\boldsymbol{u}, \boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{u}; \boldsymbol{x}) - \boldsymbol{\lambda}^T \boldsymbol{r}(\boldsymbol{u}; \boldsymbol{x})$$

First-order optimality (KKT) conditions for full space optimization problem

$$\nabla_{\boldsymbol{u}} \mathcal{L}(\boldsymbol{u}^*, \boldsymbol{x}^*, \boldsymbol{\lambda}^*) = \mathbf{0}, \qquad \nabla_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{u}^*, \boldsymbol{x}^*, \boldsymbol{\lambda}^*) = \mathbf{0}, \qquad \nabla_{\boldsymbol{\lambda}} \mathcal{L}(\boldsymbol{u}^*, \boldsymbol{x}^*, \boldsymbol{\lambda}^*) = \mathbf{0}$$

Apply (quasi-)Newton method[1] to solve nonlinear KKT system for $\boldsymbol{u}^*, \boldsymbol{x}^*, \boldsymbol{\lambda}^*$

[1] usually requires globalization such as linesearch or trust-region

Implementation mostly requires standard terms in implicit code

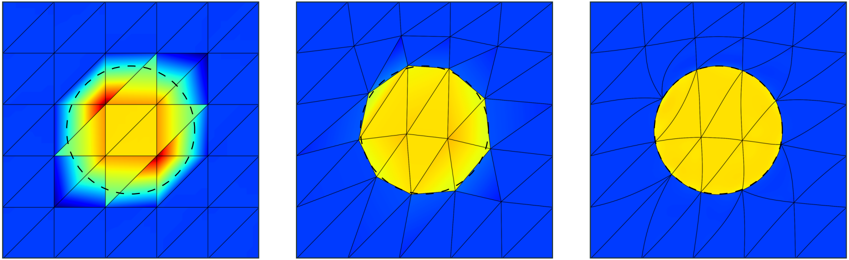
Gradient-based optimizers for the tracking optimization problem will require

$$\begin{aligned} &f(\boldsymbol{u},\,\boldsymbol{x}), \qquad &&\frac{\partial f}{\partial \boldsymbol{u}}(\boldsymbol{u},\,\boldsymbol{x}), \qquad &&\frac{\partial f}{\partial \boldsymbol{x}}(\boldsymbol{u},\,\boldsymbol{x}),\\ &\boldsymbol{r}(\boldsymbol{u},\,\boldsymbol{x}), \qquad &&\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}}(\boldsymbol{u},\,\boldsymbol{x}), \qquad &&\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{x}}(\boldsymbol{u},\,\boldsymbol{x})\end{aligned}$$

- $\boldsymbol{r}$ and $\partial_{\boldsymbol{u}}\boldsymbol{r}$ required by standard implicit solvers
- Same terms required for reduced space approach

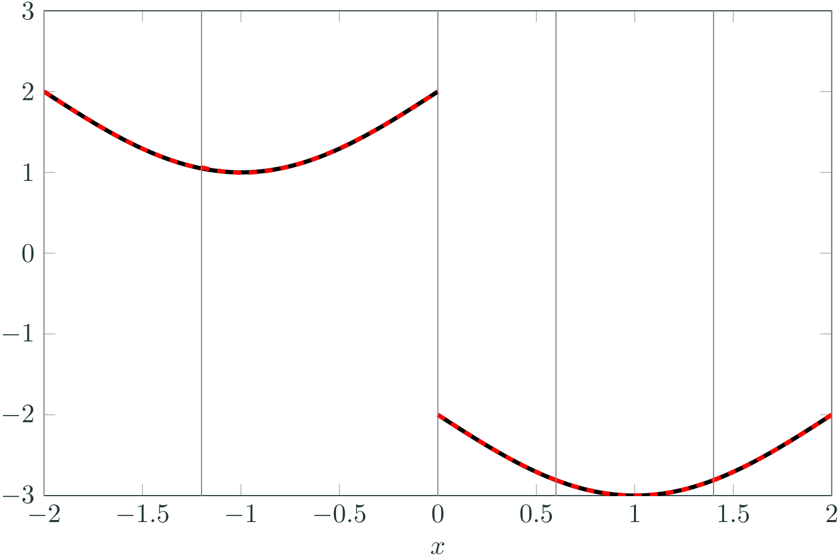
L^2 projection of discontinuous function on DG basis

$$\eta(x) = \begin{cases} 2, & x^2 + y^2 < r^2 \\ 1, & x^2 + y^2 > r^2 \end{cases}$$



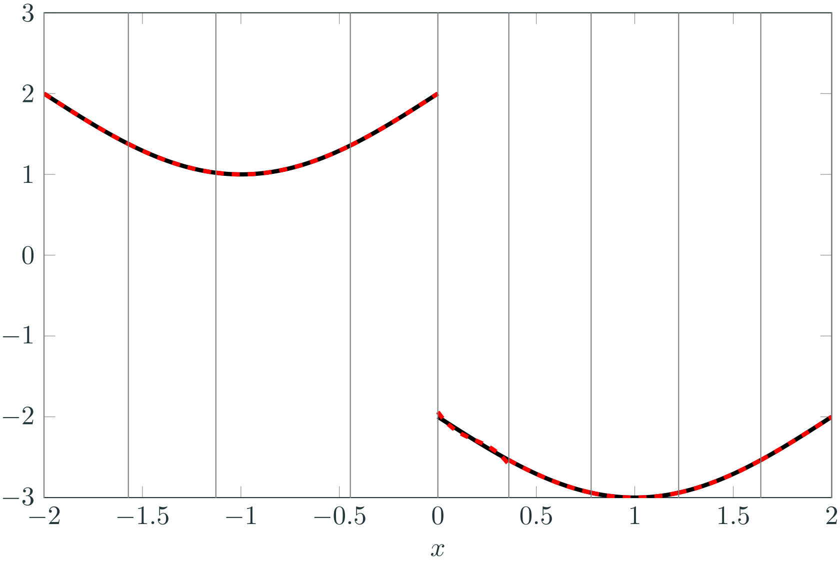
Non-aligned (*left*) vs. discontinuity-aligned mesh with linear (*middle*) and cubic (*right*) elements

Resolution of modified Burgers' equation with few elements



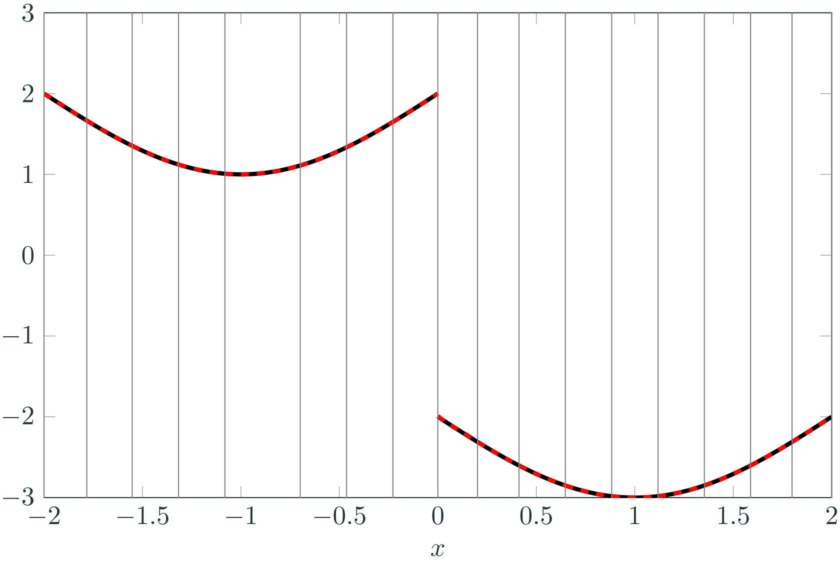
Exact solution (▬▬), tracking solution (- - -) and mesh (——) for $p = 3$

Resolution of modified Burgers' equation with few elements



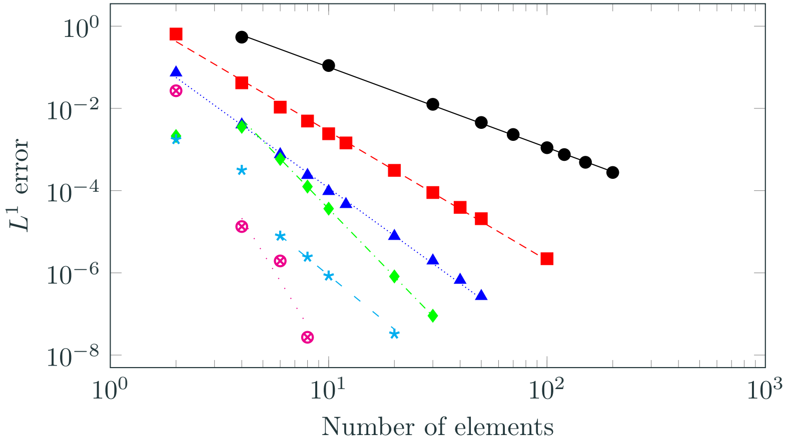
Exact solution (▬), tracking solution (- - -) and mesh (—) for $p = 3$

Resolution of modified Burgers' equation with few elements



Exact solution (▬), tracking solution (- - -) and mesh (—) for $p = 3$

$\mathcal{O}(h^{p+1})$ convergence rates demonstrated for Burgers' equation

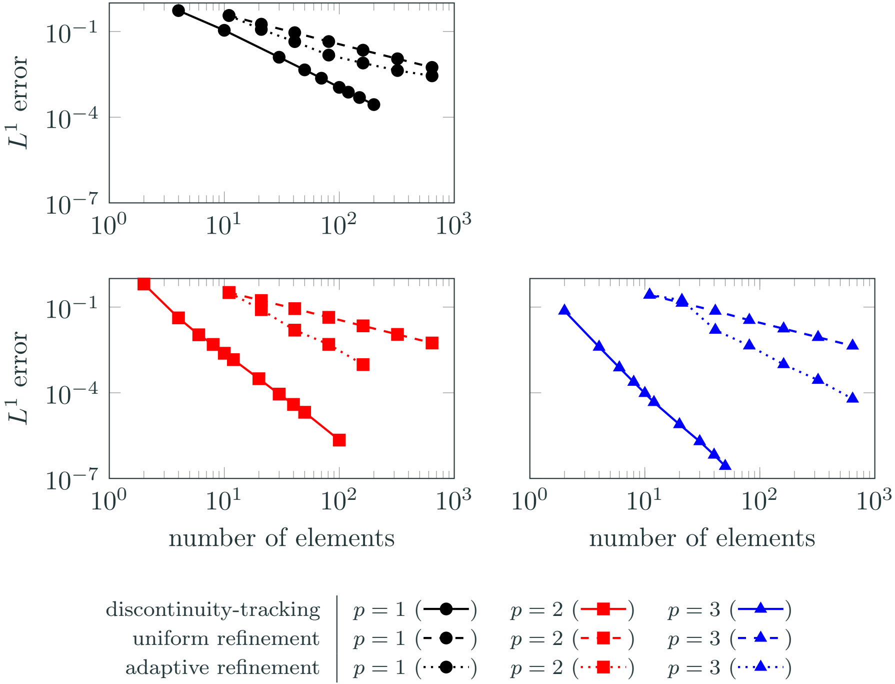


$p = 1$ (●), $p = 2$ (■), $p = 3$ (▲), $p = 4$ (◆), $p = 5$ (∗), $p = 6$ (⊗)

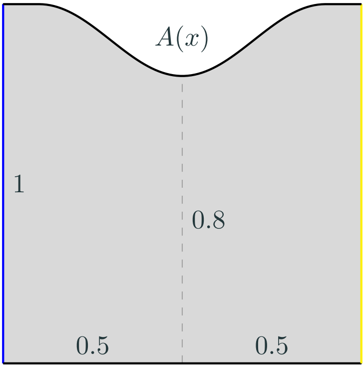
The slopes of the best-fit lines to the data points in the asymptotic regime are:

$\angle -2.0$ (———), $\angle -3.1$ (- - - -), $\angle -3.9$ (·········), $\angle -5.5$ (-·-·-), $\angle -4.4$ (– –), $\angle -8.7$ (· · ·)

Convergence: tracking vs. uniform/adaptive refinement

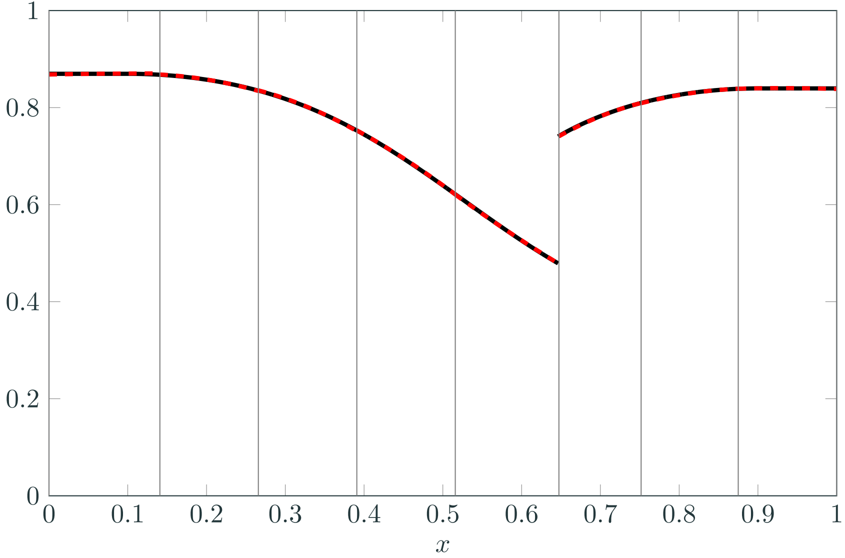


Nozzle flow: quasi-1d Euler equations



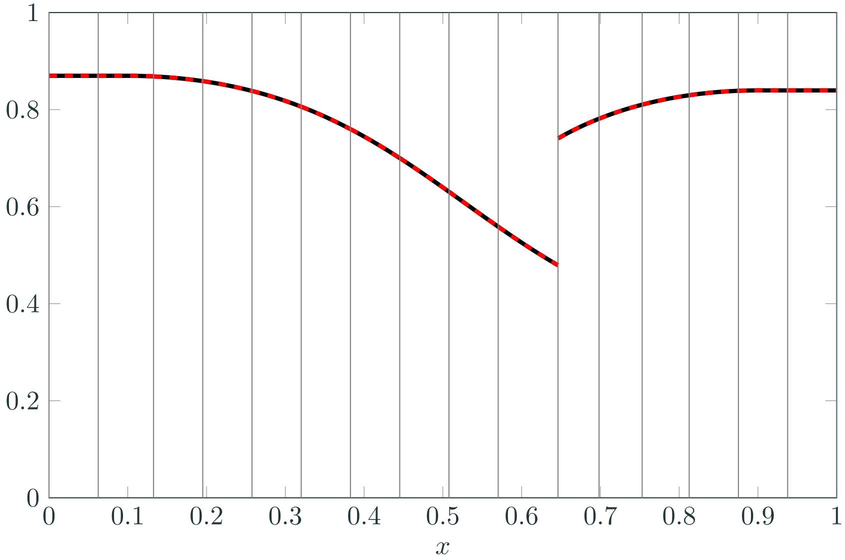
Inviscid wall (——), inflow (——), outflow (——)

Resolution of quasi-1d Euler equations with few elements



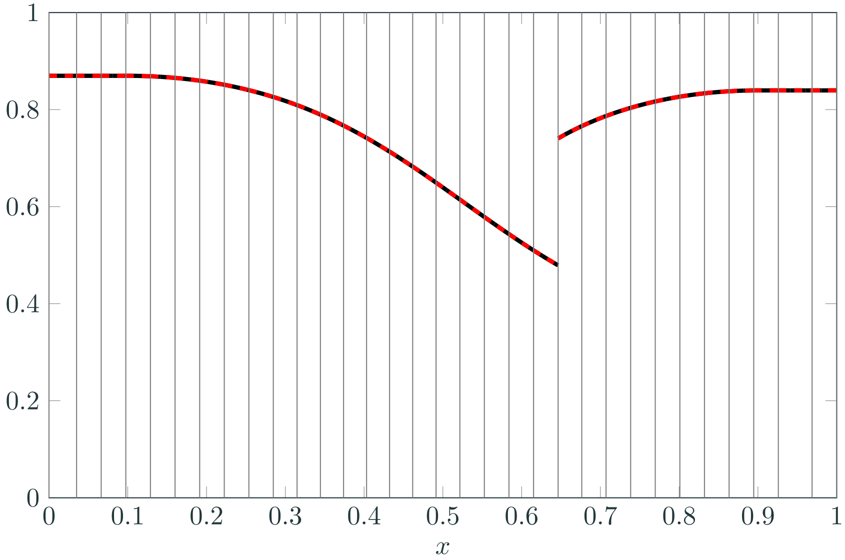
Exact solution (▬▬), tracking solution (- - -) and mesh (——) for $p = 3$

Resolution of quasi-1d Euler equations with few elements



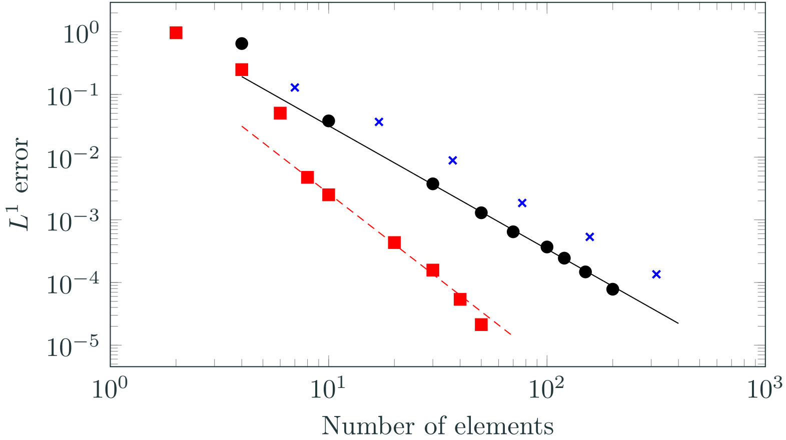
Exact solution (——), tracking solution (- - -) and mesh (——) for $p = 3$

Resolution of quasi-1d Euler equations with few elements



Exact solution (▬▬), tracking solution (- - -) and mesh (——) for $p = 3$

$\mathcal{O}(h^{p+1})$ convergence rates demonstrated for nozzle flow

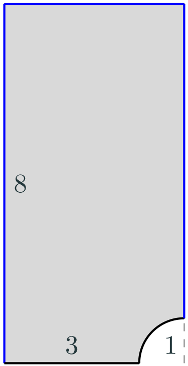


$p = 1$ (●), $p = 2$ (■)

Slope of best-fit line: $\angle -2.0$ (——), $\angle -2.7$ (- - - -)

Reference second-order method ($p = 1$) with adaptive mesh refinement (×)

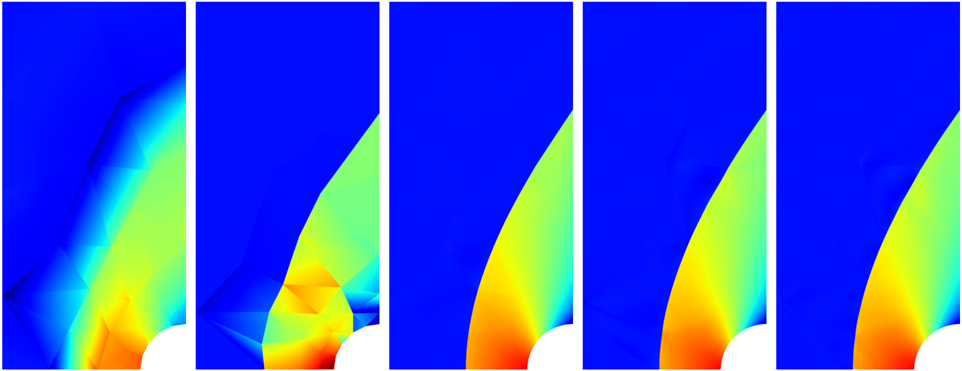
Supersonic flow ($M = 2$) around cylinder: 2D Euler equations



Inviscid wall/symmetry condition (——) and farfield (——)

Resolution of 2D supersonic flow with 48 elements

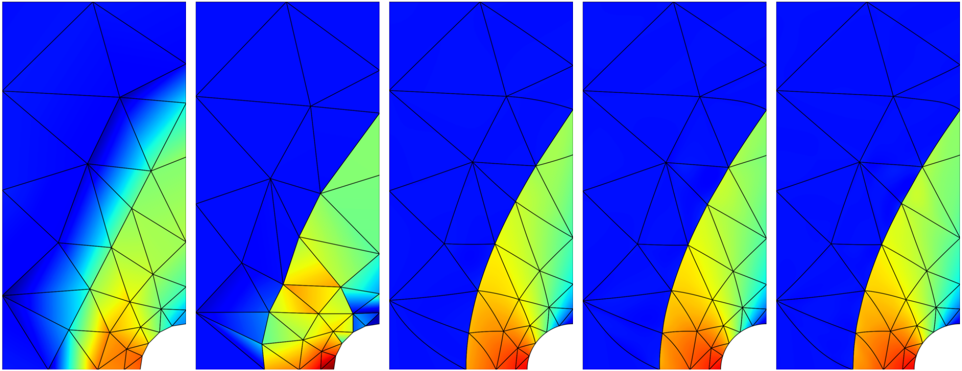
Density (ρ)



Left: Solution on non-aligned mesh with 48 linear elements and added viscosity (initial guess for shock tracking method). *Remaining*: solution using shock tracking framework corresponding to mesh with 48 $p = 1$, $p = 2$, $p = 3$, $p = 4$ elements.

Resolution of 2D supersonic flow with 48 elements

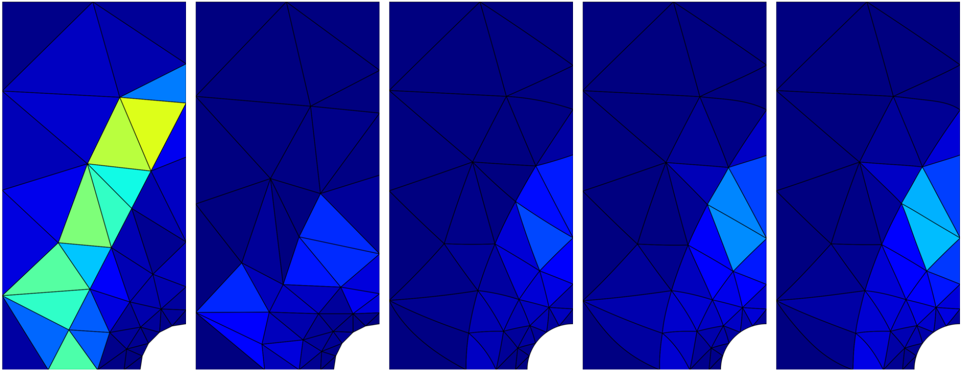
Density (ρ)



Left: Solution on non-aligned mesh with 48 linear elements and added viscosity (initial guess for shock tracking method). *Remaining*: solution using shock tracking framework corresponding to mesh with 48 $p = 1$, $p = 2$, $p = 3$, $p = 4$ elements.

Resolution of 2D supersonic flow with 48 elements

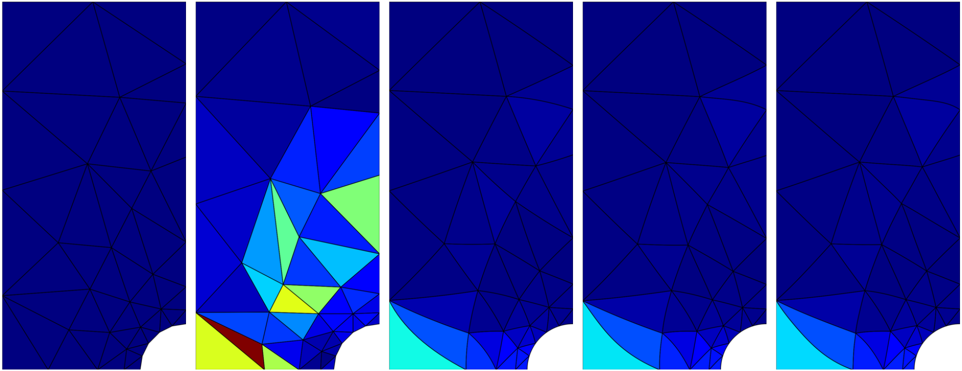
Shock tracking objective (f_{shk})



Left: Solution on non-aligned mesh with 48 linear elements and added viscosity (initial guess for shock tracking method). *Remaining*: solution using shock tracking framework corresponding to mesh with 48 $p = 1$, $p = 2$, $p = 3$, $p = 4$ elements.

Resolution of 2D supersonic flow with 48 elements

Distortion metric (f_{msh})



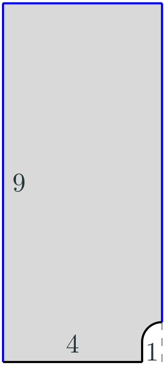
Left: Solution on non-aligned mesh with 48 linear elements and added viscosity (initial guess for shock tracking method). *Remaining*: solution using shock tracking framework corresponding to mesh with 48 $p = 1$, $p = 2$, $p = 3$, $p = 4$ elements.

Convergence to optimal solution and mesh

Discontinuity-tracking performance summary

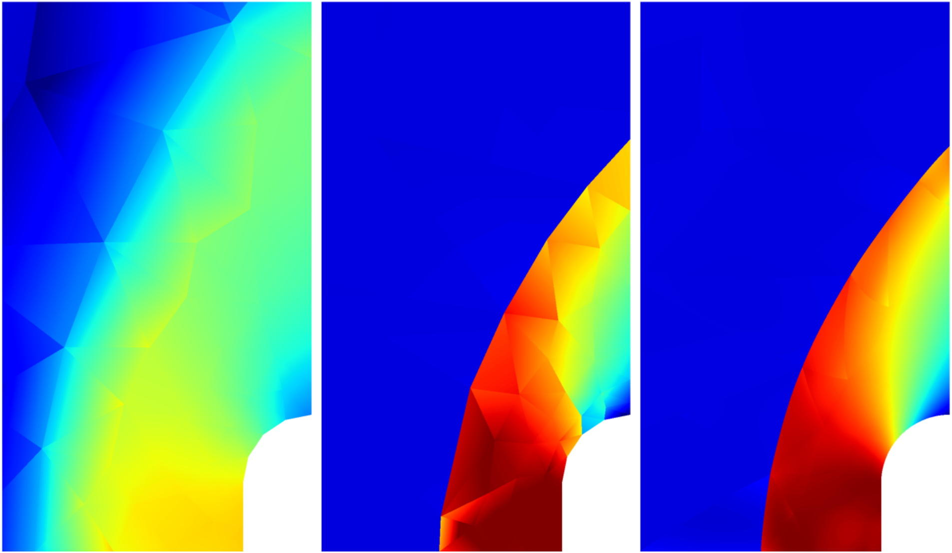
Polynomial order (p)	1	2	3	4
Degrees of freedom ($N_{\boldsymbol{u}}$)	576	1152	1920	2880
Enthalpy error (e_H)	0.0106	0.000462	0.00151	0.000885
Stagnation pressure error (e_p)	0.0711	0.00479	0.0112	0.000616

Supersonic flow ($M = 4$) around blunt body: 2D Euler equations



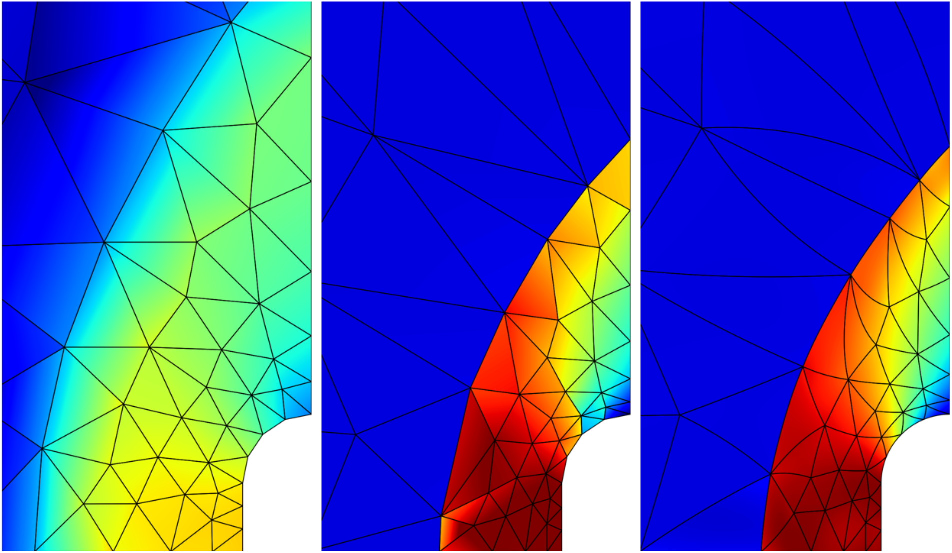
Inviscid wall/symmetry condition (——) and farfield (——)

Resolution of 2D supersonic flow with 102 quadratic elements



Left: Solution (density) on non-aligned mesh with 102 linear elements and added viscosity (initial guess for shock tracking method). *Middle/right*: solution using shock tracking framework corresponding to mesh with 102 linear (*middle*) and quadratic (*right*) elements.

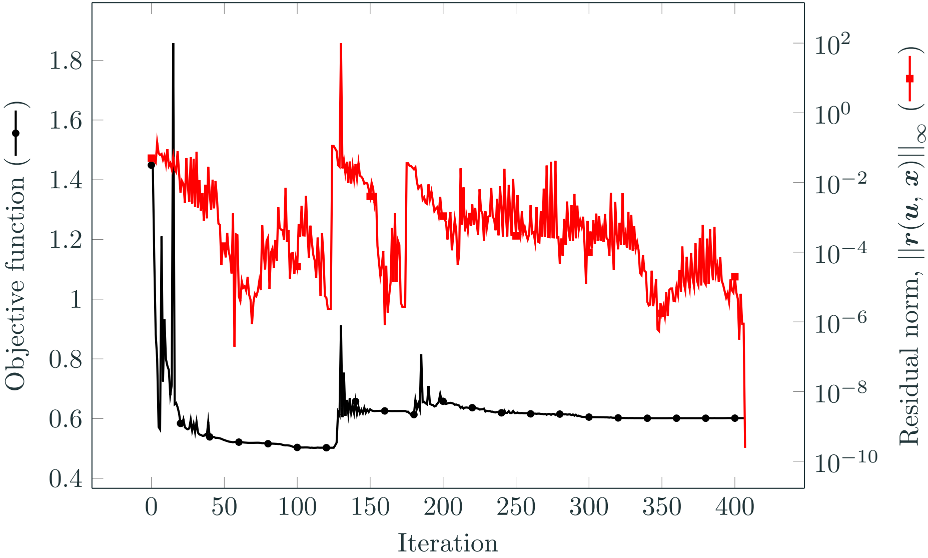
Resolution of 2D supersonic flow with 102 quadratic elements



Left: Solution (density) on non-aligned mesh with 102 linear elements and added viscosity (initial guess for shock tracking method). *Middle/right*: solution using shock tracking framework corresponding to mesh with 102 linear (*middle*) and quadratic (*right*) elements.

Convergence to optimal solution and mesh

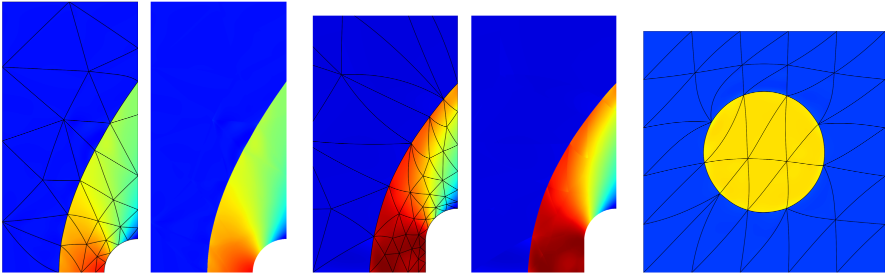
Solver simultaneously minimizes objective and solves PDE



Convergence of residual and objective function

Conclusions and future work

- Introduced high-order shock tracking method based on DG discretization and PDE-constrained optimization formulation
- Key innovations: *objective function* that monotonically approaches a minimum as mesh face aligns with shock and *full space solver*
- Optimal convergence $\mathcal{O}(h^{p+1})$ rates obtained and used to resolve a number of transonic and supersonic flows on very coarse meshes
- Future work
    - **numerical flux** consistent with *integral form* (jumps do not tend to 0)
    - **solver** that exploits *problem structure* and incorporates *homotopy*
    - **local topology changes** to reduce iterations and improve mesh quality



Mach 2 flow around cylinder (*left*), Mach 4 flow around blunt body (*middle*), and L^2 projection of discontinuous function (*right*).

References I

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.

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.

References II

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 *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.

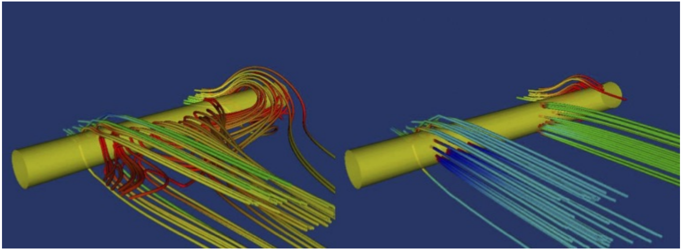
Zahr, M. J. and Persson, P.-O. (2018a).
An optimization-based approach for high-order accurate discretization of conservation laws with discontinuous solutions.
Journal of Computational Physics, 365:105 – 134.

References III

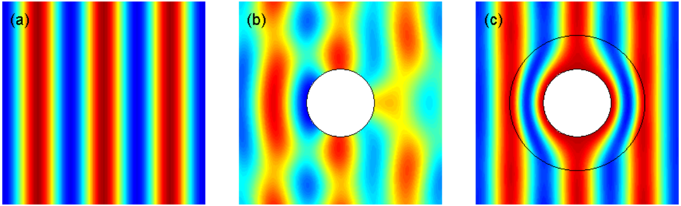
Zahr, M. J., Persson, P.-O., and Wilkening, J. (2016).
A fully discrete adjoint method for optimization of flow problems on deforming domains with time-periodicity constraints.
Computers & Fluids, 139:130 – 147.

PDE optimization is ubiquitous in science and engineering

Control: Drive system to a desired state



Boundary flow control



Metamaterial cloaking – electromagnetic invisibility

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

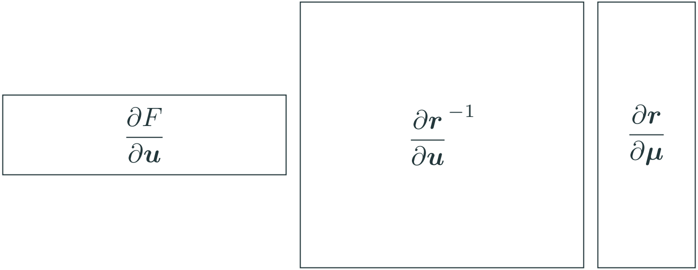
$$DF = \frac{\partial F}{\partial \boldsymbol{\mu}} + \frac{\partial F}{\partial \boldsymbol{u}}\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{\mu}} = \frac{\partial F}{\partial \boldsymbol{\mu}} - \frac{\partial F}{\partial \boldsymbol{u}}\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}}^{-1}\frac{\partial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

Algebraic equations leads to adjoint equations

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

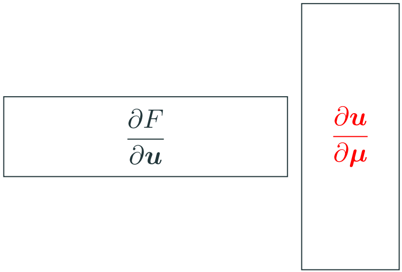
Sensitivity vs. adjoint method to compute gradient of F

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



Sensitivity vs. adjoint method to compute gradient of F

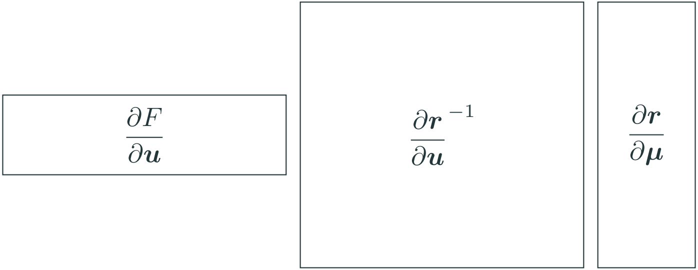
$$\frac{\partial F}{\partial \boldsymbol{u}} \; \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}}^{-1} \; \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\mu}}$$



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

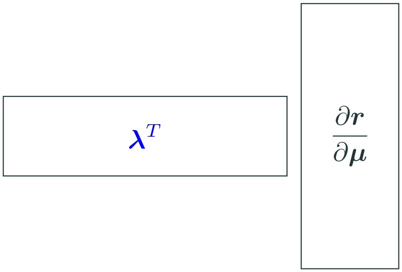
$$\frac{\partial F}{\partial \boldsymbol{u}} \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}}^{-1} \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\mu}}$$



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

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



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**

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

High-quality reconstruction from coarse MRI grid (space: 24×36, time: 20) and low noise (3%)

Reconstructed flow

Synthetic MRI data $\boldsymbol{d}^*_{i,n}$ (top) and computational representation of MRI data $\boldsymbol{d}_{i,n}$ (bottom)

High-quality reconstruction from fine MRI grid (space: 40×60, time: 20) and low noise (3%)

Reconstructed flow

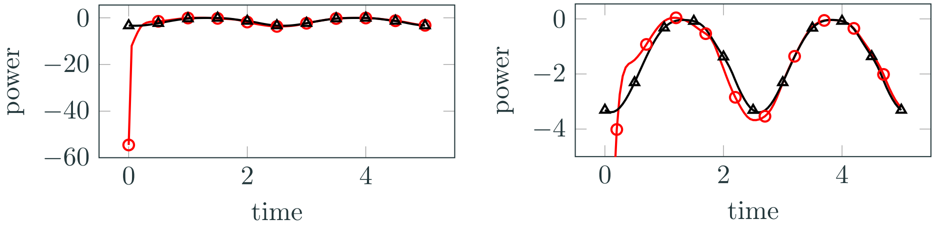
Synthetic MRI data $\boldsymbol{d}^*_{i,n}$ (top) and computational representation of MRI data $\boldsymbol{d}_{i,n}$ (bottom)

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

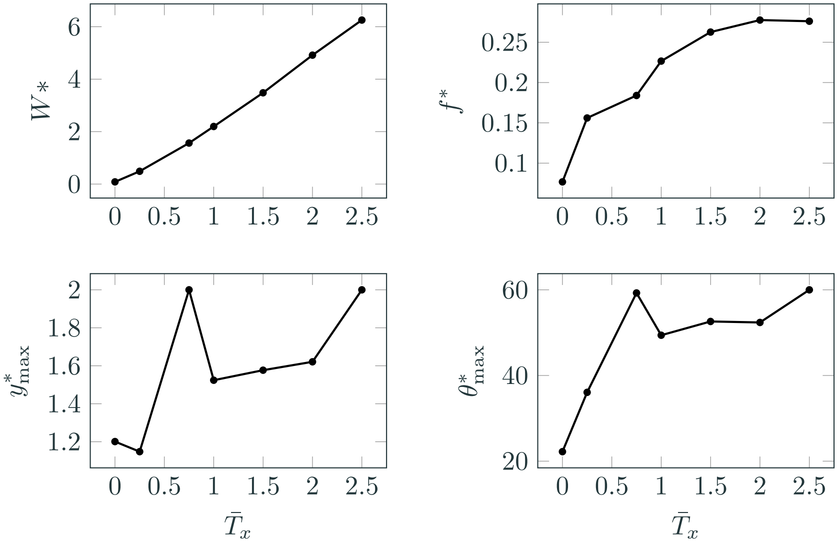
$$\begin{aligned}
\underset{\boldsymbol{U},\ \boldsymbol{\mu}}{\text{minimize}} \quad & \mathcal{F}(\boldsymbol{U}, \boldsymbol{\mu}) \\
\text{subject to} \quad & \boldsymbol{U}(\boldsymbol{x}, 0) = \boldsymbol{U}(\boldsymbol{x}, T) \\
& \frac{\partial \boldsymbol{U}}{\partial t} + \nabla \cdot \boldsymbol{F}(\boldsymbol{U},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

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



Time history of power on airfoil of flow initialized from steady-state (red circles) and from a time-periodic solution (black triangles)

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 $\bar{T}_x$.

Initial guess for optimization: $\boldsymbol{u}_0$, ϕ_0

- Initial guess for $\boldsymbol{u}$ and $\boldsymbol{\phi}$ critical given the non-convex nonlinear optimization formulation of our shock tracking method
- *Homotopy*: define a sequence of shock tracking problems where the solution of problem j is used to initialize problem $j+1$
- Sequence of problems chosen using homotopy in *polynomial order* and Mach number (for high Mach flows)
- For initial problem in homotopy sequence:
  - $\boldsymbol{\phi}_0$ chosen such that resulting mesh is identical to the reference mesh
  - $\boldsymbol{u}_0$ chosen as the solution of the discrete conservation law with enough added viscosity ν

$$\boldsymbol{r}_\nu(\boldsymbol{u},\, \boldsymbol{x}(\boldsymbol{\phi}_0)) = 0$$

Modified Burgers' equation with discontinuous source term

Inviscid, modified one-dimensional Burgers' equation with a discontinuous source term from [Barter, 2008]

$$\frac{\partial}{\partial x}\left(\frac{1}{2}u^2\right) = \beta u + f(x), \quad \text{for } x \in \Omega = (-2,\, 2),$$

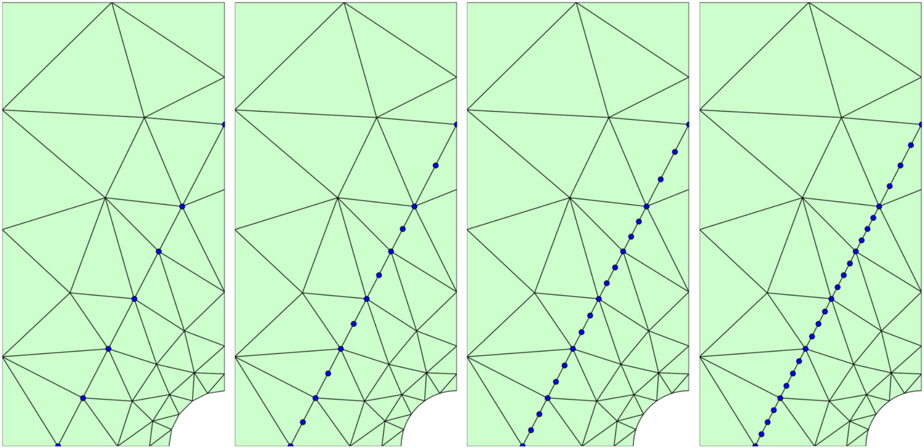
where $u(-2) = 2$, $u(2) = -2$, $\beta = -0.1$ and

$$f(x) = \begin{cases} (2 + \sin(\frac{\pi x}{2}))(\frac{\pi}{2}\cos(\frac{\pi x}{2}) - \beta), & x < 0 \\ (2 + \sin(\frac{\pi x}{2}))(\frac{\pi}{2}\cos(\frac{\pi x}{2}) + \beta), & x > 0 \end{cases}$$

Analytical solution

$$u(x) = \begin{cases} 2 + \sin(\frac{\pi x}{2}), & x < 0 \\ -2 - \sin(\frac{\pi x}{2}), & x > 0 \end{cases}$$

High-order meshes and parametrization



Reference domain and mesh with 48 elements and polynomial orders $p = 1$ (*left*), $p = 2$ (*middle left*), $p = 3$ (*middle right*), and $p = 4$ (*right*). The blue circles identify parametrized nodes.