Accurate Methods for Computing High Mach Number Reactive Flow

Matthew J. Zahr and Joseph M. Powers

Department of Aerospace and Mechanical Engineering
University of Notre Dame, USA

Lawrence Livermore National Laboratory

Livermore, California

24 June 2020

Motivation

We investigate numerical methods to accurately simulate high Mach number flow with shocks and exothermic reaction:

- Shock-capturing: (WENO, Roe, etc), less than first order convergence,
- Shock-fitting: high accuracy and potential high order convergence; limited to simple topologies and algorithmically complicated,
- Wavelet adaptive methods: high accuracy, appropriate for multiscale, algorithmically complicated,
- Implicit shock-tracking: high accuracy and potential high order convergence, finite element based.

We find, relative to WENO, implicit shock tracking yields remarkable improvement in accuracy on a standard verification problem in oblique detonation.

WENO shock-capturing: looks good, but converges at $< O(\Delta x)$

- 1,986 journal articles on WENO (Shu, et al, 1996), cited 35,489 times!
- WENO shock-capturing "looks" better than Lax-Wendroff, etc. shock-capturing in the "picture norm".
- But, WENO converges at $O(\Delta x^{1-1/(r+1)})$, where r is the rate of convergence for smooth problems.
- Our $r = 5$ WENO converges at $O(\Delta x^{5/6})$ for shocks.

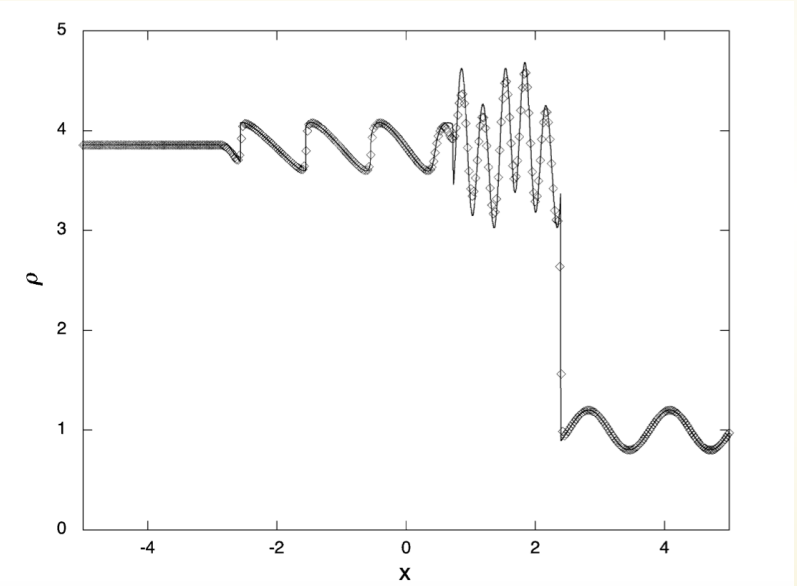


Table 9
L_1 density error norms and convergence rates, r_c, for WENO5 ($\epsilon = 10^{-6}$) as implemented in Ref. [1] and WENO5M ($\epsilon = 10^{-40}$) methods applied to Sod's [12] shock tube problem

Δx	L_1		r_c	
	WENO5	WENO5M	WENO5	WENO5M
1/100	6.88×10^{-3}	6.35×10^{-3}	–	–
1/200	3.67×10^{-3}	3.34×10^{-3}	0.907	0.925
1/400	2.01×10^{-3}	1.84×10^{-3}	0.866	0.860
1/800	1.03×10^{-3}	9.37×10^{-4}	0.970	0.975
1/1600	5.11×10^{-4}	4.59×10^{-4}	1.008	1.030
1/3200	2.76×10^{-4}	2.48×10^{-4}	0.889	0.885
1/6400	1.38×10^{-4}	1.23×10^{-4}	0.998	1.018
1/12800	7.27×10^{-5}	6.40×10^{-5}	0.927	0.938
1/25600	3.89×10^{-5}	3.42×10^{-5}	0.900	0.906
1/51200	2.18×10^{-5}	1.91×10^{-5}	0.837	0.837
1/102400	1.20×10^{-5}	1.09×10^{-5}	0.858	0.817

Henrick, Aslam, Powers, *Journal of Computational Physics*, 2005.

WENO with shock fitting: looks good and converges at $O(\Delta x^5)$

- Algorithmically complicated shock-fitting requires exact solution of shock jump conditions and mapping of the domain to shock-attached coordinates.
- It enables high accuracy and convergence rates.
- Our $r = 5$ WENO and shock-fitting converges at $O(\Delta x^{4.907})$ for a problem with shock and reaction.

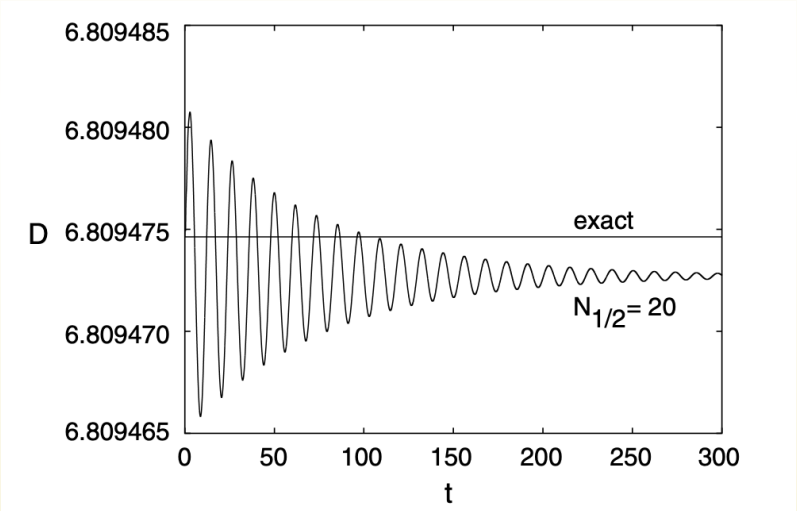


Table 7
Convergence rates of the limit cycle period for $E = 28.2$

Δx	Period	r_c
1/20	35.86111963	–
1/40	35.859442127	4.2648
1/80	35.859529390	4.6210
1/160	35.859532936	4.9407
1/320	35.859533052	–

Henrick, Aslam, Powers, *Journal of Computational Physics*, 2006.

Pseudo-spectral with shock fitting: looks good and converges spectrally!

- Shock-fitting may be coupled with a pseudo-spectral method for very high accuracy and convergence rates.
- Solution here for an inert blunt body re-entry problem at $M = 3.5$.
- Algorithmically complicated!
- Karhunen-Loéve-based reduced-order model for shape optimization

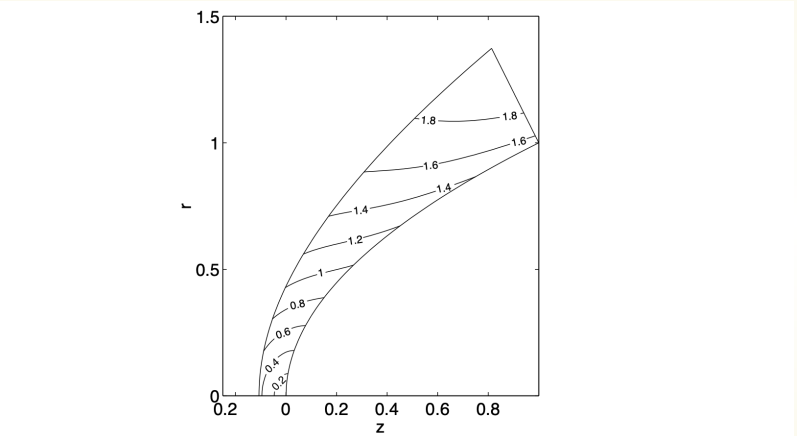
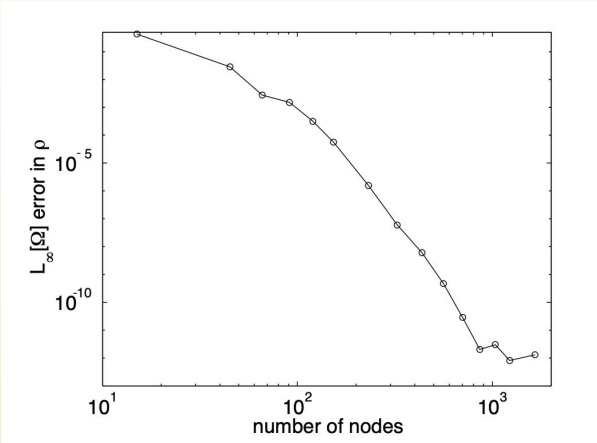


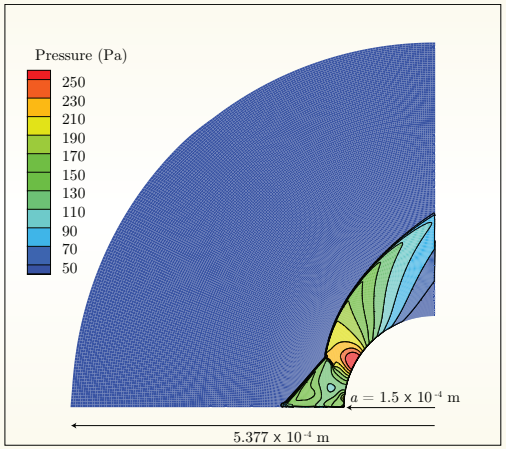
Fig. 6. Contours of Mach number for flow over the blunt body for $b = 0.5$, $M_\infty = 3.5$, 17×9 grid.

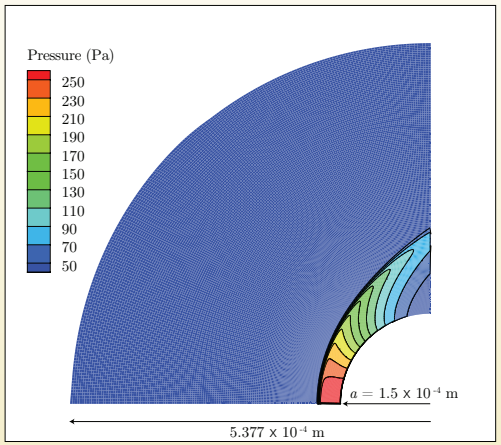


Brooks and Powers, *Journal of Computational Physics*, 2004.

Physical diffusion can eliminate carbuncle instability, but must resolve thin zones

- Roe-based shock-capturing may induce non-physical carbuncle instabilities for Euler equation solution.
- Inclusion of physical viscosity in Navier-Stokes solution stabilizes the flow.
- Solution here from OpenFOAM for an inert blunt body re-entry problem at $M = 5.73$.
- Resolution of thin zones is expensive!

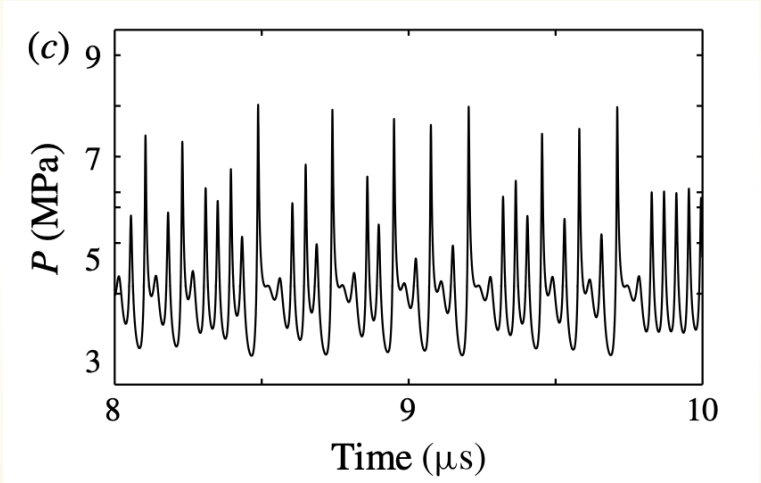




Powers, Bruns, Jemcov, AIAA-2015-0579, 2015.

Adaptive wavelet method can capture viscous shock and detailed kinetics reaction

- Wavelet basis can efficiently capture multiscale signals.
- Solutions must be continuous, so shocks require viscosity.
- Paolucci, Powers, et al., 2001, initiated this method for realistic combustion flows with detailed kinetics.
- Nonlinear chaotic dynamics resolved.
- Multidimensional viscous detonations in $H_2 - air$ resolved.
- Algorithmically complicated!



Romick, Aslam, Powers, *Journal of Fluid Mechanics*; Romick, Ph.D. Dissertation, 2015.

Implicit shock tracking: a relatively new method

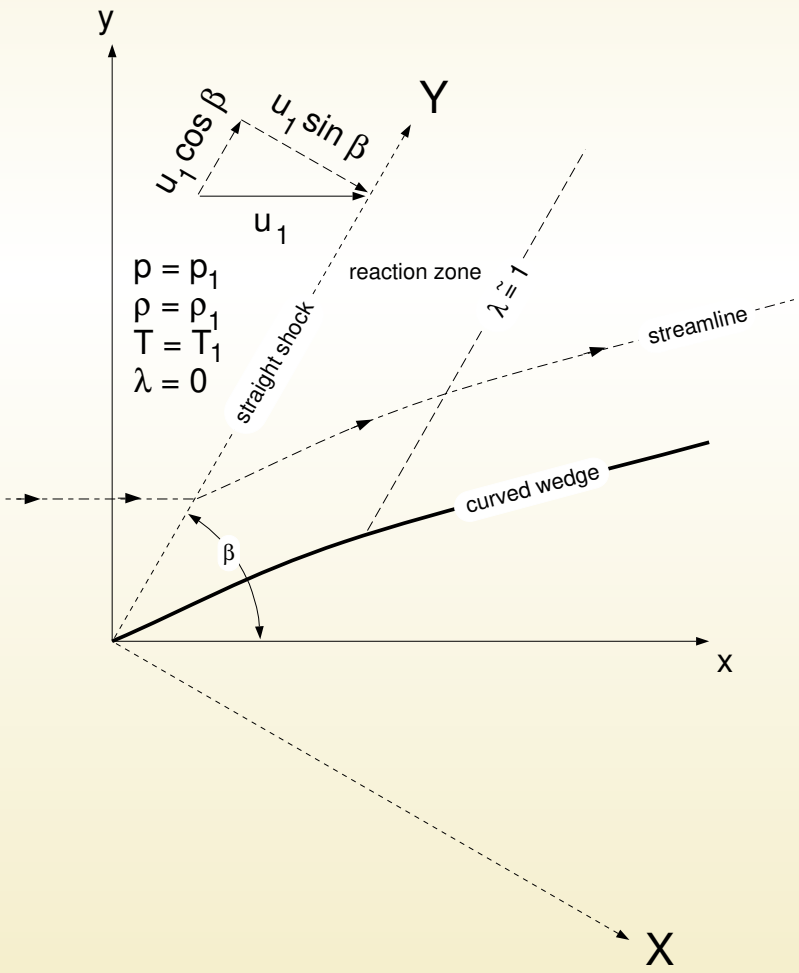
- The new method of implicit shock tracking is a competitive and often highly advantageous method, as will be described shortly.
- To quantify its advantage, we will test it against the *widely used* WENO shock-capturing method on a benchmark verification problem in two-dimensional steady detonation.
- Powers, J. M., and Aslam, T. D., 2006, "Exact Solution for Multidimensional Compressible Reactive Flow for Verifying Numerical Algorithms," *AIAA Journal*, 44(2): 337-344.
- We develop this useful solution next.

Partial review of oblique detonation

- Samaras, D. G., "Gas Dynamic Treatment of Exothermic and Endothermic Discontinuities," *Canadian Journal of Research A*, Vol. 26, No. 1, 1948, pp. 1-21.
- Gross, R. A., "Oblique Detonation Waves," *AIAA Journal*, Vol. 1, No. 5, 1963, pp. 1225-1227.
- Pratt, D. T., Humphrey, J. W., and Glenn, D. E., "Morphology of Standing Oblique Detonation Waves," *Journal of Propulsion and Power*, Vol. 7, No. 5, 1991, pp. 837-645.
- Lee, R. S., "A Unified Analysis of Supersonic Nonequilibrium Flow over a Wedge: I. Vibrational Nonequilibrium," *AIAA Journal*, Vol. 4, No. 1, 1966, pp. 30-37.
- Powers, J. M., and Stewart, D. S., "Approximate Solutions for Oblique Detonations in the Hypersonic Limit," *AIAA Journal*, Vol. 30, No. 3, 1992, pp. 726-736.
- Grismer, M. J., and Powers, J. M., "Comparisons of Numerical Oblique Detonation Solutions with an Asymptotic Benchmark," *AIAA Journal*, Vol. 30, No. 12, 1992, pp. 2985-2987.
- Powers, J. M., and Gonthier, K. A., "Reaction Zone Structures for Strong, Weak Overdriven, and Weak Underdriven Oblique Detonations," *Physics of Fluids A*, Vol. 4, No. 9, 1992, pp. 2082-2089.
- Grismer, M. J., and Powers, J. M., "Numerical Predictions of Oblique Detonation Stability Boundaries," *Shock Waves*, Vol. 6, No. 3, 1996, pp. 147-156.

Oblique detonation schematic

- Straight shock.
- Curved wedge.
- Orthogonal coordinate system aligned with shock.



Powers and Aslam, *AIAA Journal*, 2006.

Model: reactive Euler equations

- two-dimensional,
- steady,
- inviscid,
- irrotational,
- one-step kinetics with zero activation energy,
- calorically perfect ideal gases with identical molecular masses and specific heats

Model: reactive Euler PDEs

$$
\begin{aligned}
\frac{\partial}{\partial X}(\rho U) + \frac{\partial}{\partial Y}(\rho V) &= 0, \\
\frac{\partial}{\partial X}\left(\rho U^2 + p\right) + \frac{\partial}{\partial Y}(\rho U V) &= 0, \\
\frac{\partial}{\partial X}(\rho U V) + \frac{\partial}{\partial Y}\left(\rho V^2 + p\right) &= 0, \\
\frac{\partial}{\partial X}\left(\rho U\left(e + \frac{1}{2}(U^2+V^2) + \frac{p}{\rho}\right)\right) & \\
+\frac{\partial}{\partial Y}\left(\rho V\left(e + \frac{1}{2}(U^2+V^2) + \frac{p}{\rho}\right)\right) &= 0, \\
\frac{\partial}{\partial X}(\rho U\lambda) + \frac{\partial}{\partial Y}(\rho V\lambda) &= \alpha\rho(1-\lambda)H(T-T_i), \\
e &= \frac{1}{\gamma-1}\frac{p}{\rho} - \lambda q, \\
p &= \rho R T.
\end{aligned}
$$

Model reductions: PDEs→ODEs

Assume no Y variation, so

$$\begin{aligned}
\frac{d}{dX}\left(\rho U\right) &= 0, \\
\frac{d}{dX}\left(\rho U^2 + p\right) &= 0, \\
\frac{d}{dX}\left(\rho U V\right) &= 0, \\
\frac{d}{dX}\left(\rho U\left(e + \frac{1}{2}(U^2+V^2) + \frac{p}{\rho}\right)\right) &= 0, \\
\frac{d}{dX}\left(\rho U \lambda\right) &= \alpha\rho(1-\lambda)H(T-T_i).
\end{aligned}$$

Model reductions: ODEs→DAEs

$$
\begin{aligned}
\rho U &= \rho_1 u_1 \sin\beta, \\
\rho U^2 + p &= \rho_1 u_1^2 \sin^2\beta + p_1, \\
V &= u_1 \cos\beta, \\
\frac{\gamma}{\gamma-1}\frac{p}{\rho} - \lambda q + \frac{1}{2}\left(U^2 + u_1^2\cos^2\beta\right) &= \frac{\gamma}{\gamma-1}\frac{p_1}{\rho_1} + \frac{1}{2}u_1^2, \\
\frac{d\lambda}{dX} &= \alpha\frac{1-\lambda}{U}H(T-T_i).
\end{aligned}
$$

ZND reaction zone structure ODE supplemented with extended Rankine-Hugoniot algebraic conditions.

Model reductions: inversion of algebraic relations

with $\mathcal{M}_1 \equiv M_1 \sin\beta$,

$$\rho(\lambda) = \frac{\rho_1(\gamma+1)\mathcal{M}_1^2}{1+\gamma\mathcal{M}_1^2 \pm \sqrt{(1+\gamma\mathcal{M}_1^2)^2 - (\gamma+1)\mathcal{M}_1^2\left(2+\frac{\gamma-1}{\gamma}\frac{2\lambda q}{RT_1}+(\gamma-1)\mathcal{M}_1^2\right)}},$$

$$U(\lambda) = \frac{\rho_1 u_1 \sin\beta}{\rho(\lambda)},$$

$$p(\lambda) = p_1 + \rho_1^2 u_1^2 \sin^2\beta\left(\frac{1}{\rho_1} - \frac{1}{\rho(\lambda)}\right),$$

$$T(\lambda) = \frac{p_1}{\rho(\lambda)R} + \frac{\rho_1^2 u_1^2 \sin^2\beta}{\rho(\lambda)R}\left(\frac{1}{\rho_1} - \frac{1}{\rho(\lambda)}\right),$$

$$q \le \frac{\gamma R T_1(\mathcal{M}_1^2-1)^2}{2(\gamma^2-1)\mathcal{M}_1^2}, \qquad \text{CJ limitation.}$$

$+$ shocked; $-$ unshocked. Take the shocked branch.

Reaction zone structure solution

$$\frac{d\lambda}{dX} = \frac{\alpha}{\rho_1 u_1 \sin\beta}\,\rho(\lambda)(1-\lambda), \qquad \lambda(0) = 0,$$

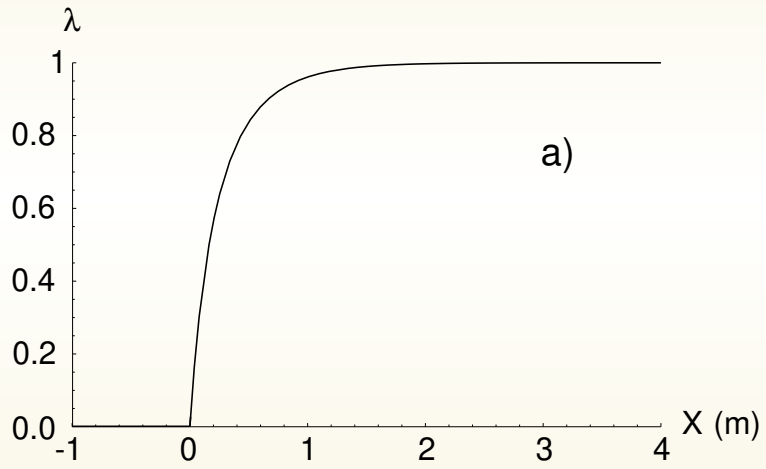
$$X(\lambda) = a_1 \left(2a_3 \left(\sqrt{1 - a_4\lambda} - 1 \right) + \ln \left(\left(\frac{1}{1-\lambda} \right)^{a_2} \left(\frac{\left(1 - \sqrt{\frac{1-a_4\lambda}{1-a_4}} \right) \left(1 + \sqrt{\frac{1}{1-a_4}} \right)}{\left(1 + \sqrt{\frac{1-a_4\lambda}{1-a_4}} \right) \left(1 - \sqrt{\frac{1}{1-a_4}} \right)} \right)^{a_3\sqrt{1-a_4}} \right) \right),$$

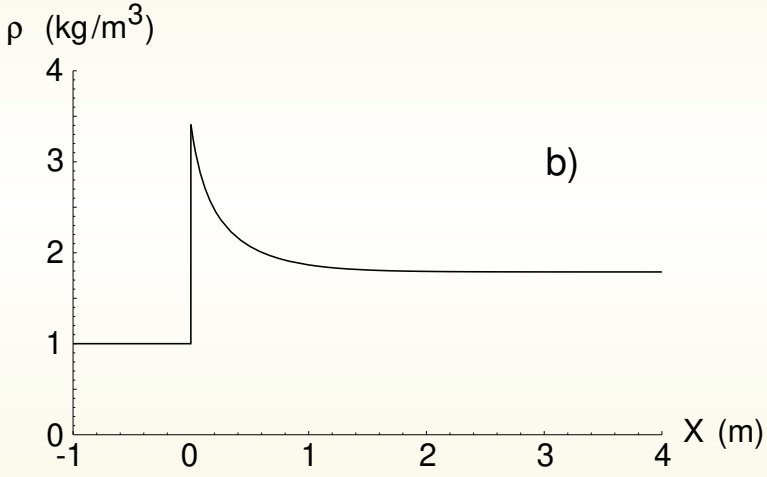
$$\begin{aligned}
a_1 &= \frac{1}{(\gamma+1)\mathcal{M}_1} \frac{\sqrt{\gamma R T_1}}{\alpha}, \\
a_2 &= 1 + \gamma\mathcal{M}_1^2, \\
a_3 &= \mathcal{M}_1^2 - 1, \\
a_4 &= 2\frac{\mathcal{M}_1^2}{\left(\mathcal{M}_1^2 - 1\right)^2} \frac{\gamma^2 - 1}{\gamma} \frac{q}{RT_1}.
\end{aligned}$$

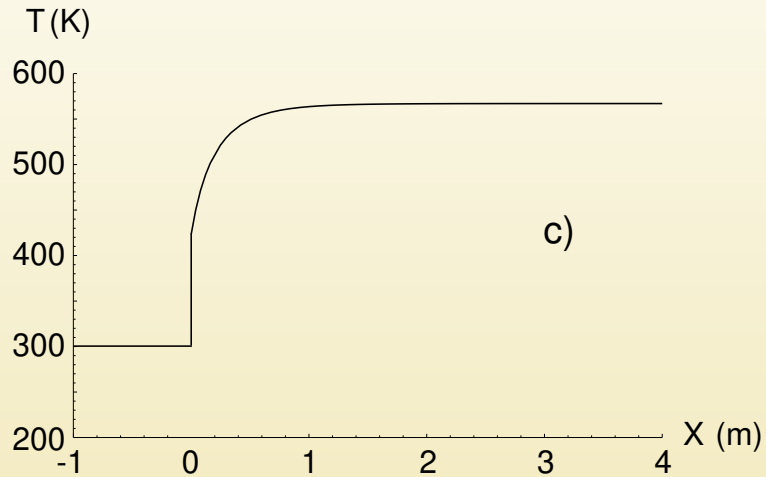
Parametric values

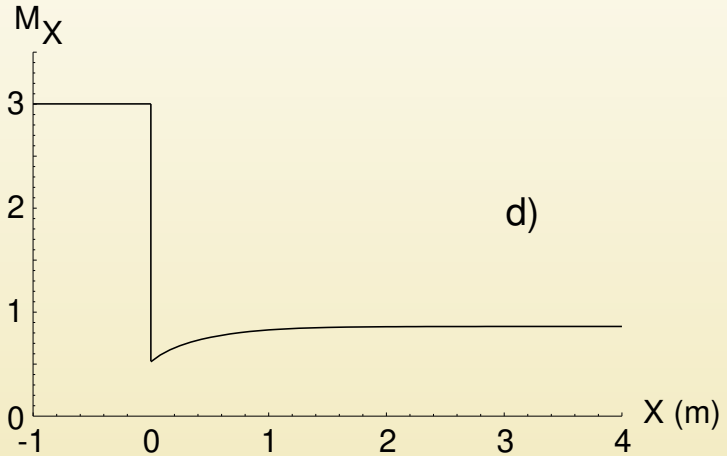
Independent Parameter	Units	Value
R	J/kg/K	287
α	1/s	1000
β	rad	$\pi/4$
γ	-	6/5
T_1	K	300
M_1	-	3
ρ_1	$\mathrm{kg/m^3}$	1
q	J/kg	300000
T_i	K	131300/363

Reaction zone structure normal to shock



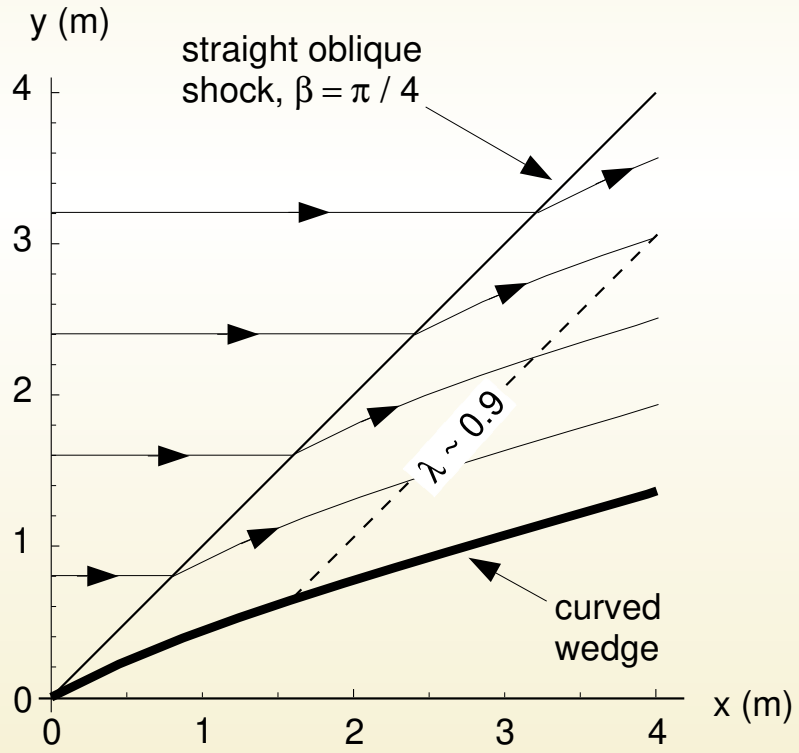






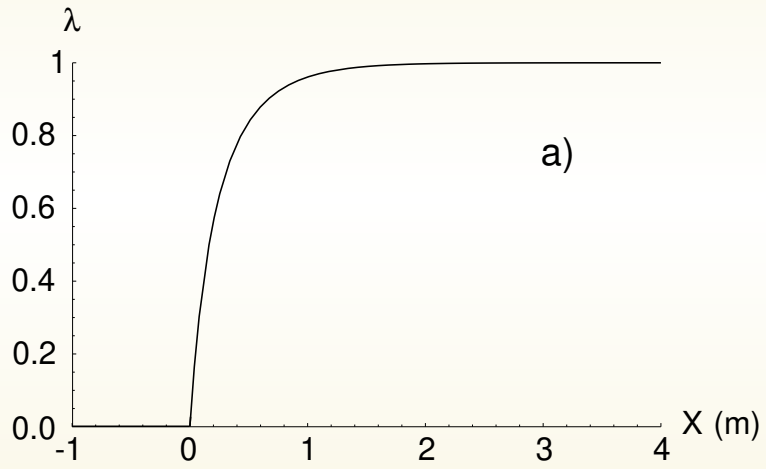
Exact solution: streamlines

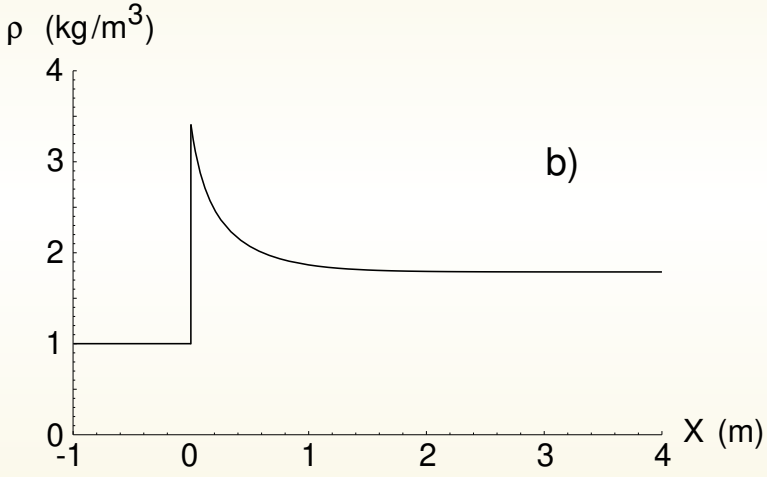
- Curved streamlines identical to wedge contour.
- Streamline curvature approaches zero as reaction completes.

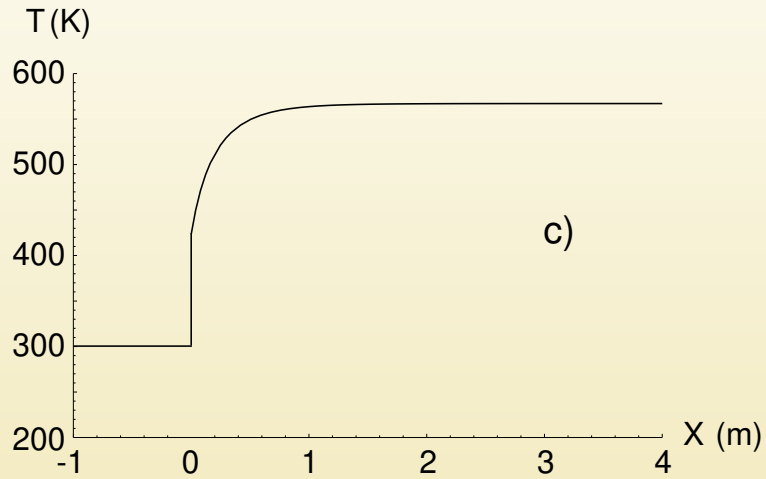


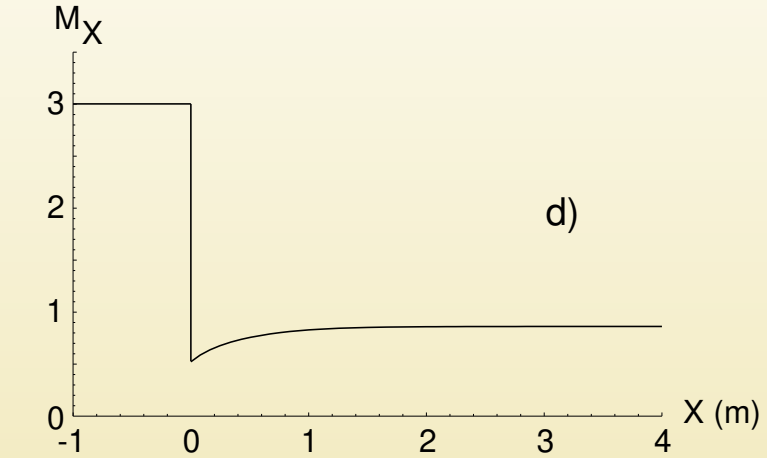
Powers and Aslam, *AIAA Journal*, 2006.

Reaction zone structure normal to shock







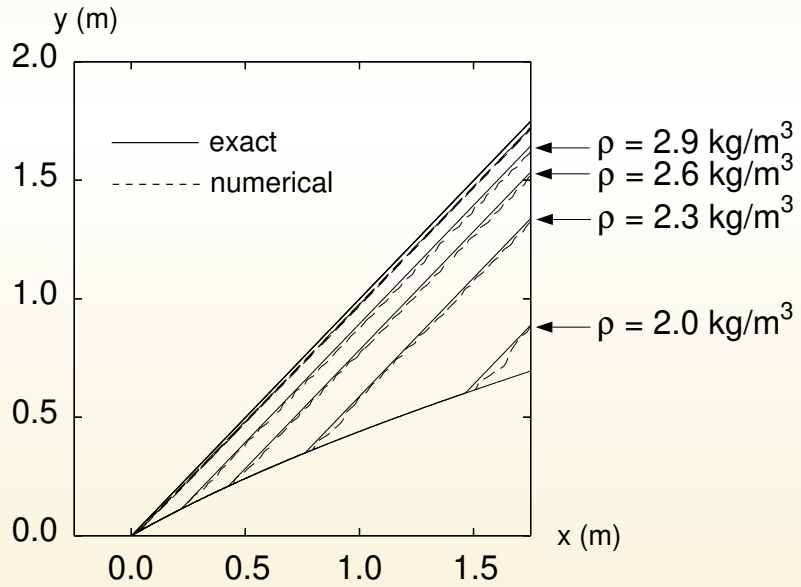


Verification of WENO shock capturing

- Algorithm of Xu, Aslam, and Stewart, 1997, *CTM*.
- Uniform Cartesian grid.
- Embedded internal boundary with level set representation.
- Nominally fifth order weighted essentially non-oscillatory (WENO) discretization.
- Non-decomposition based Lax-Friedrichs solver.
- Third order Runge-Kutta time integration.

Exact versus WENO solution

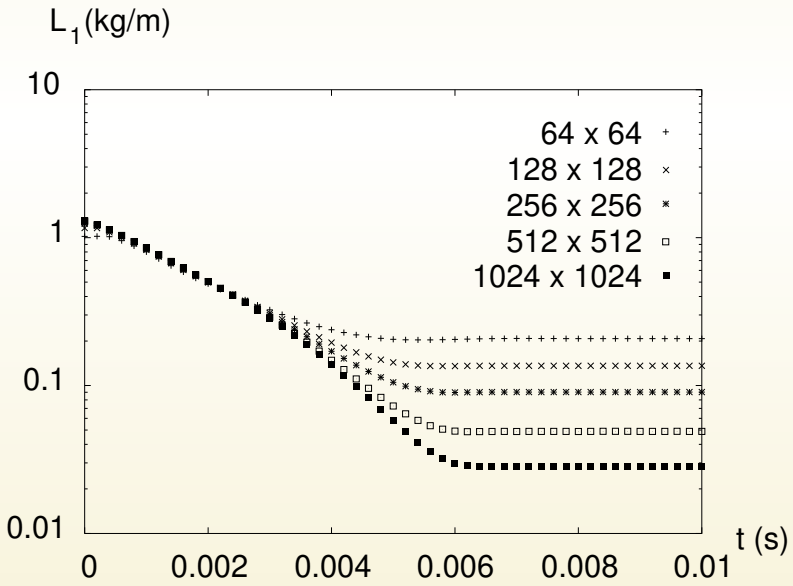
- 256×256 uniform numerical grid.
- good agreement in picture norm.
- numerical solution stable.



Powers and Aslam, *AIAA Journal*, 2006.

Iterative convergence to steady state: various grids

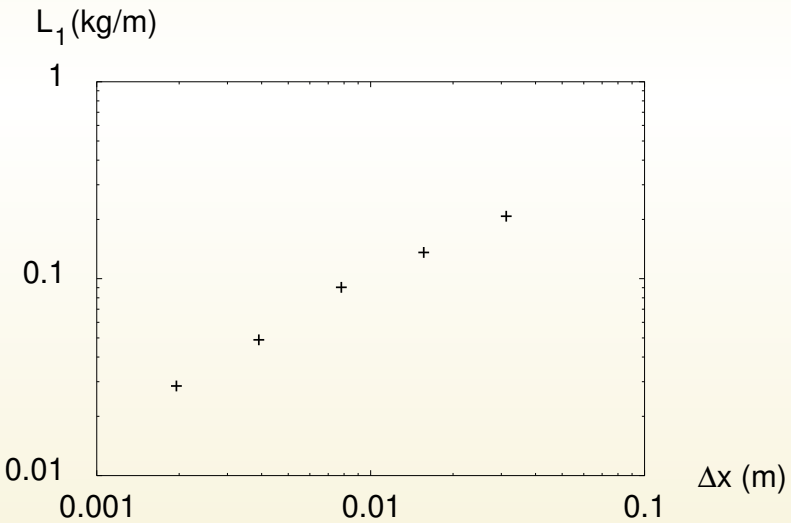
- Coarse grids relax quickly; fine grids relax slowly.
- All grids iteratively converge to steady state.
- Iterative convergence is distinct from grid convergence.



Powers and Aslam, *AIAA Journal*, 2006.

Grid convergence

- Convergence rate: $O\left(\Delta x^{0.779}\right)$.
- Both shock capturing and embedded boundary induce the low convergence rate.



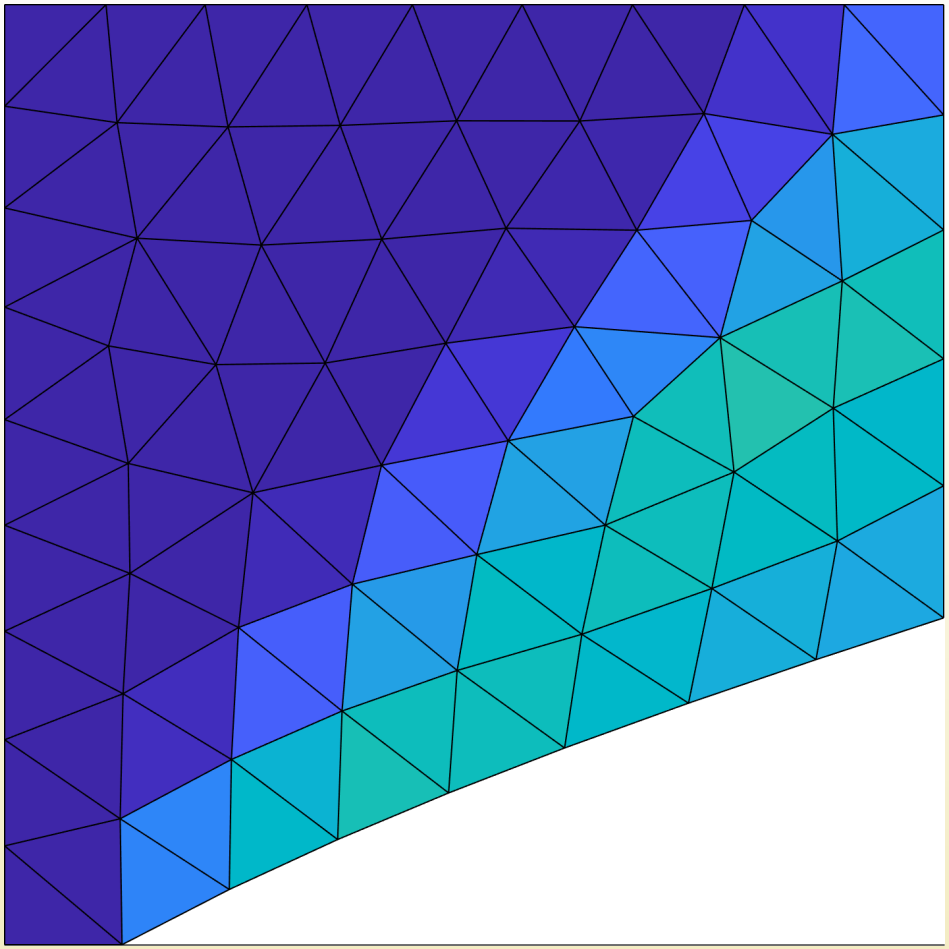
Powers and Aslam, *AIAA Journal*, 2006.

Conclusions

- Modeling flows with thin zones and surfaces of discontinuity is challenging.
- Most common shock-capturing methods converge at less than first order.
- Heroic measures are often needed to achieve high order convergence
- Implicit shock tracking.....

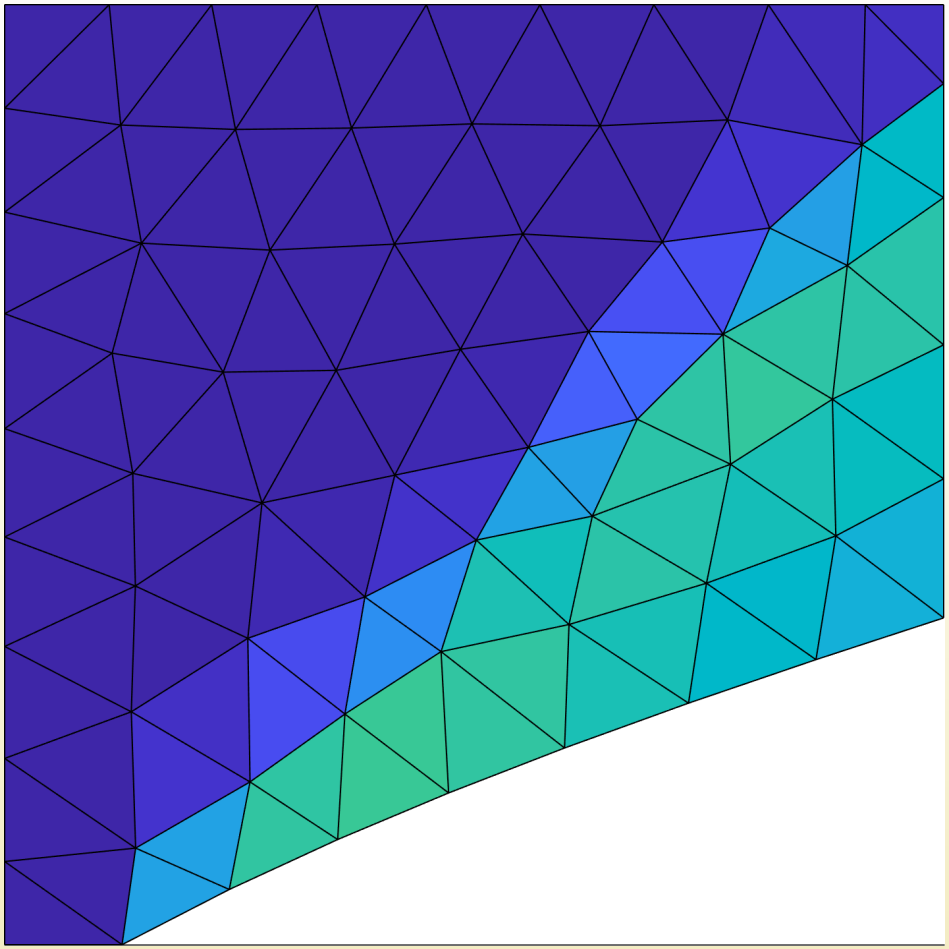
High-order implicit tracking: reacting inviscid flow

Original mesh: 100 triangular elements (unstructured)



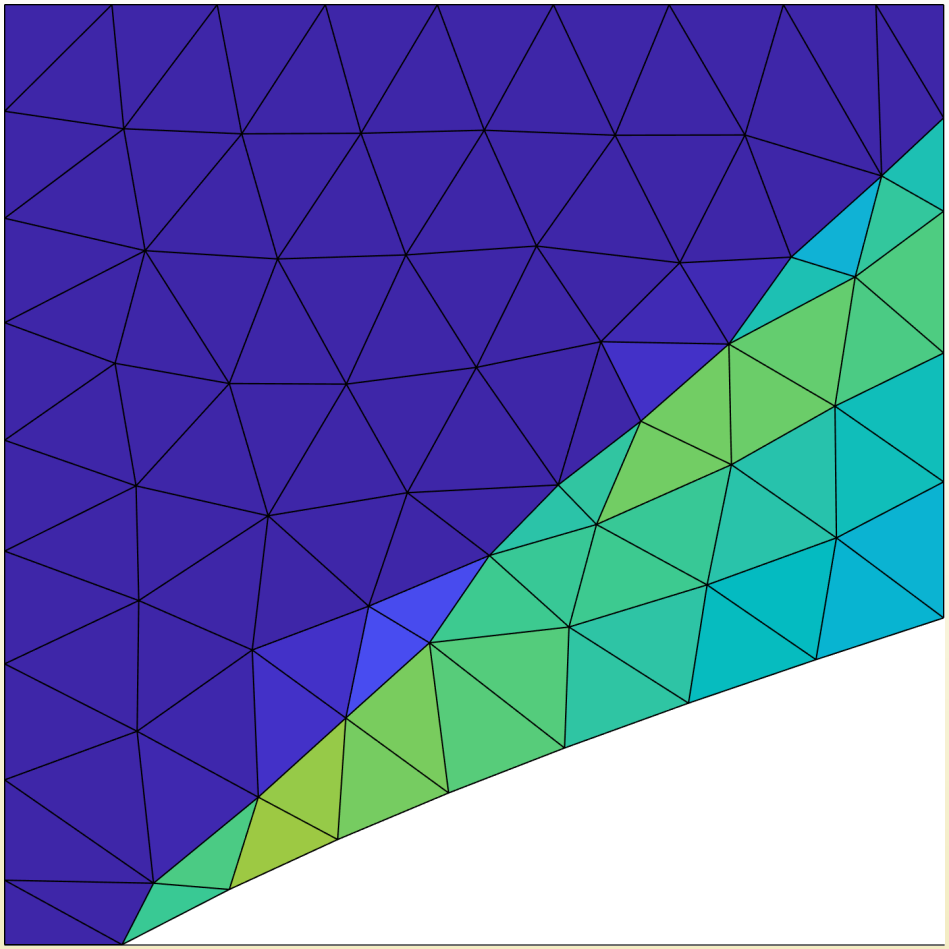
High-order implicit tracking: reacting inviscid flow

Original mesh: 100 triangular elements (unstructured)



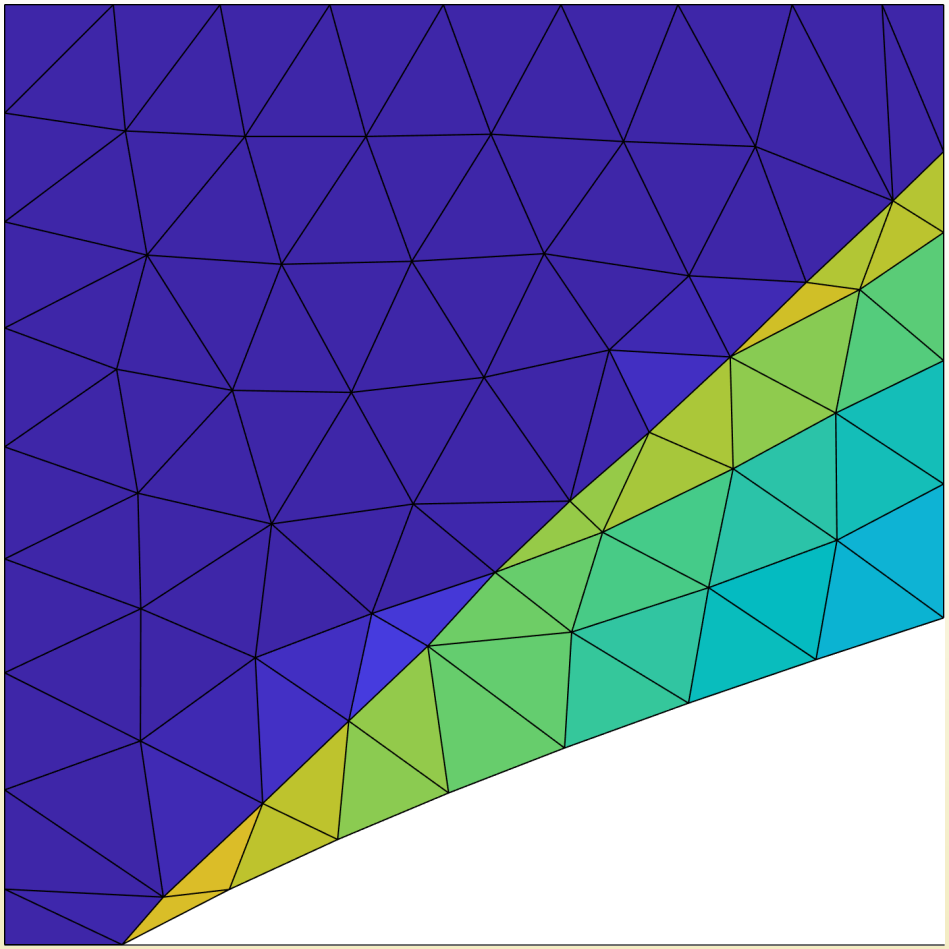
High-order implicit tracking: reacting inviscid flow

Original mesh: 100 triangular elements (unstructured)



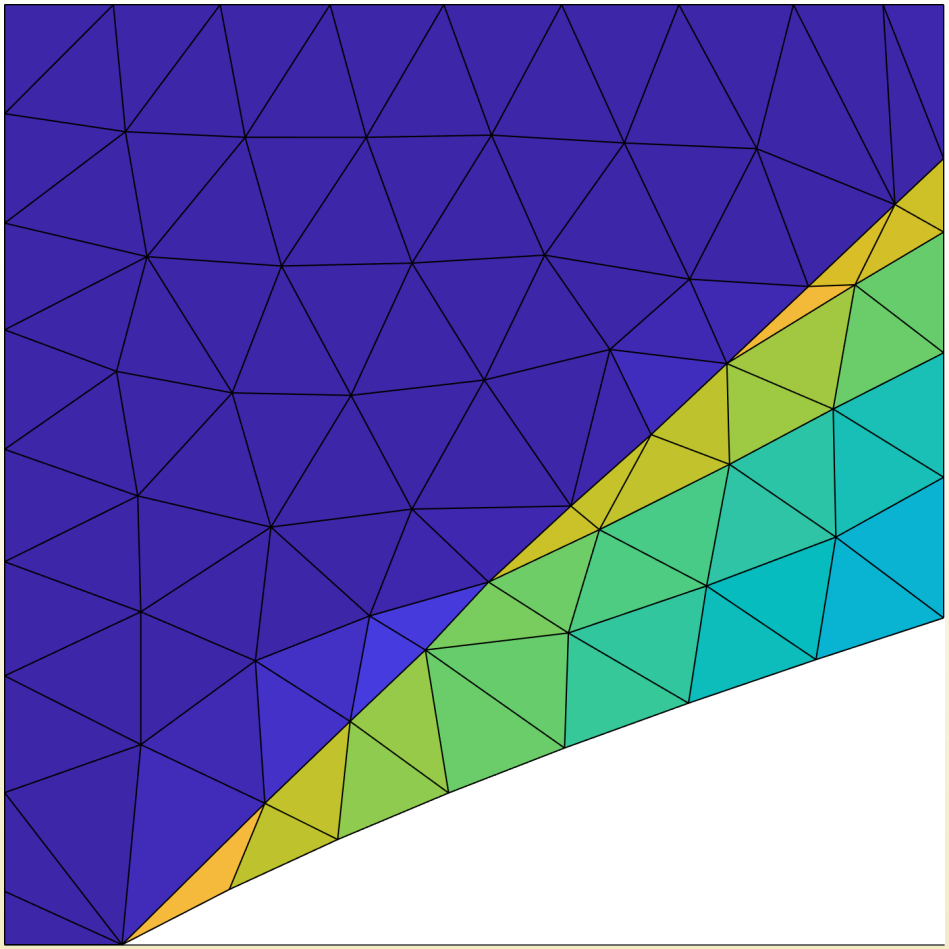
High-order implicit tracking: reacting inviscid flow

Original mesh: 100 triangular elements (unstructured)



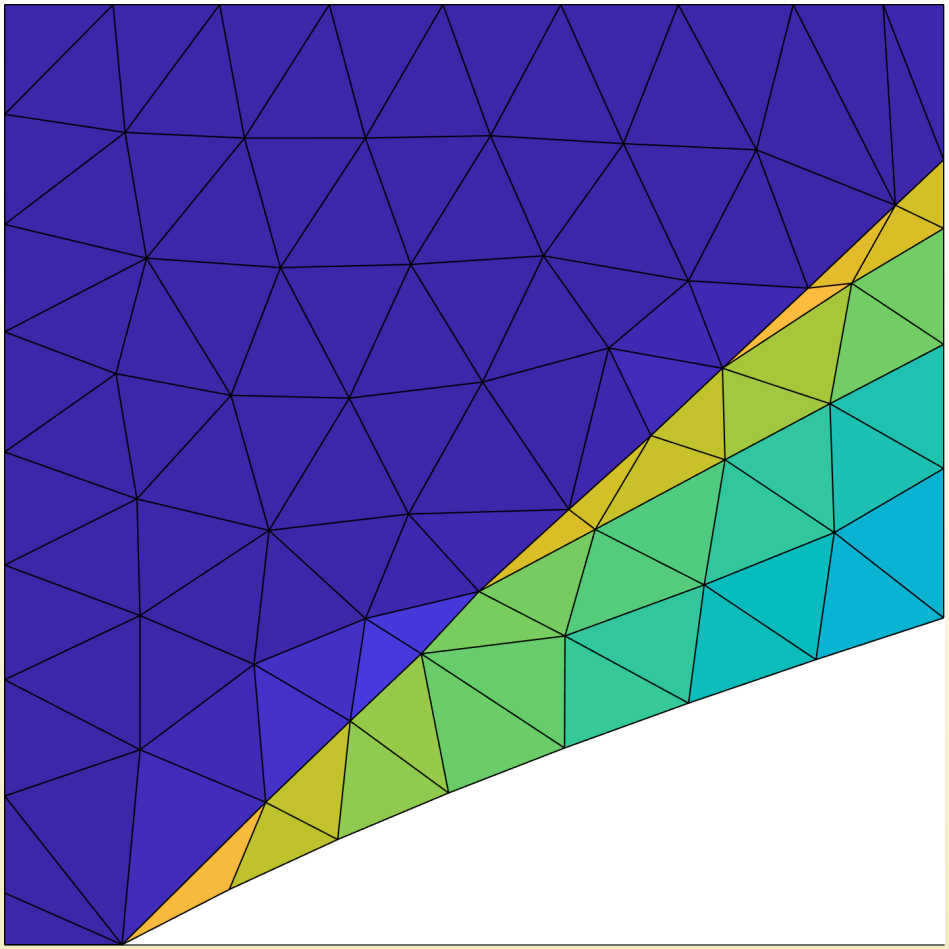
High-order implicit tracking: reacting inviscid flow

Original mesh: 100 triangular elements (unstructured)



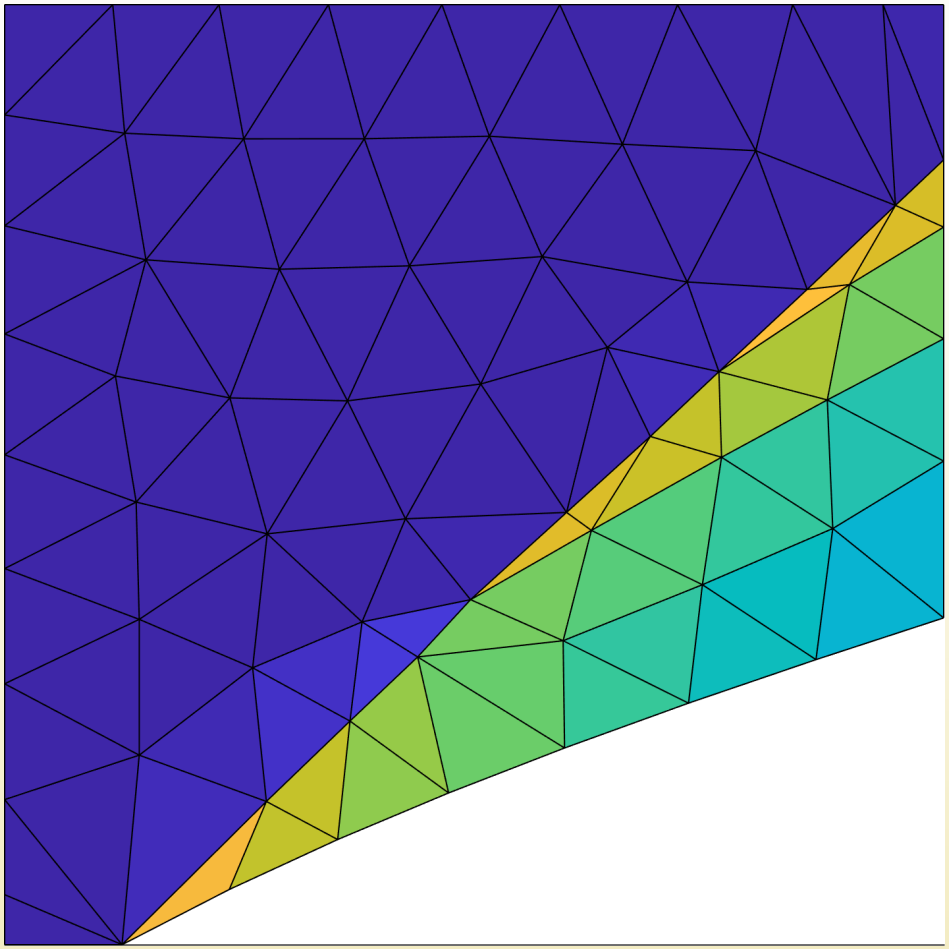
High-order implicit tracking: reacting inviscid flow

Original mesh: 100 triangular elements (unstructured)



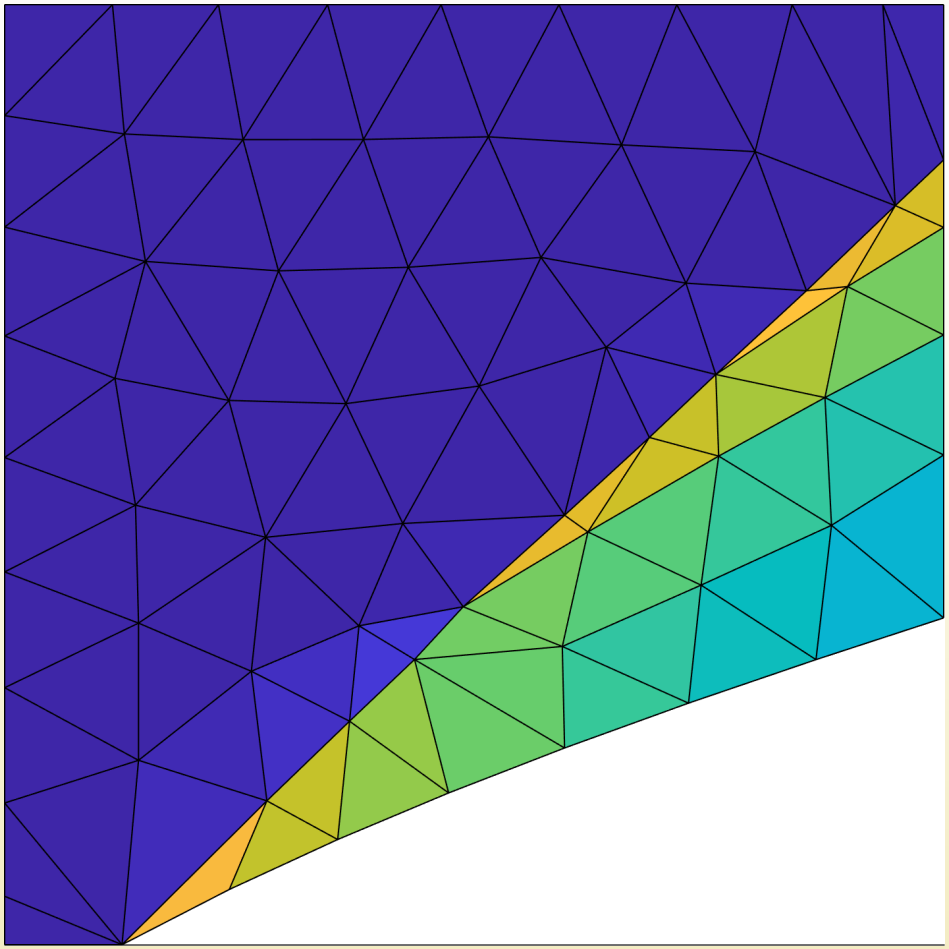
High-order implicit tracking: reacting inviscid flow

Original mesh: 100 triangular elements (unstructured)



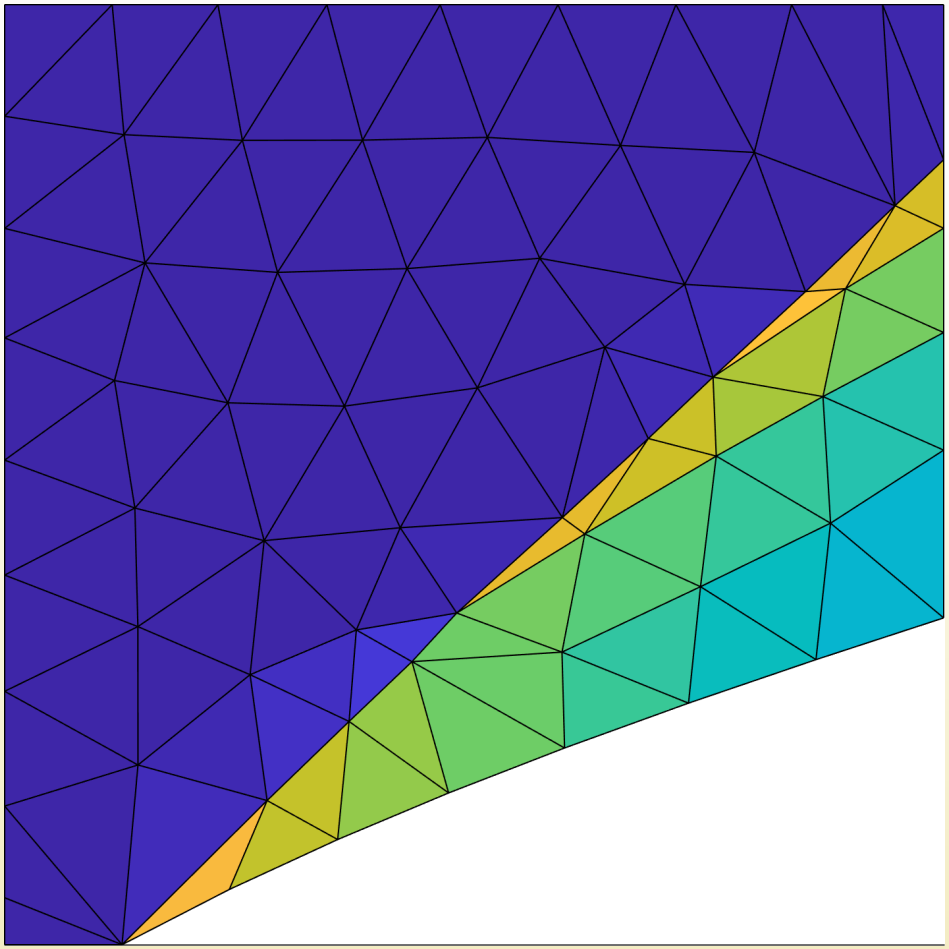
High-order implicit tracking: reacting inviscid flow

Original mesh: 100 triangular elements (unstructured)



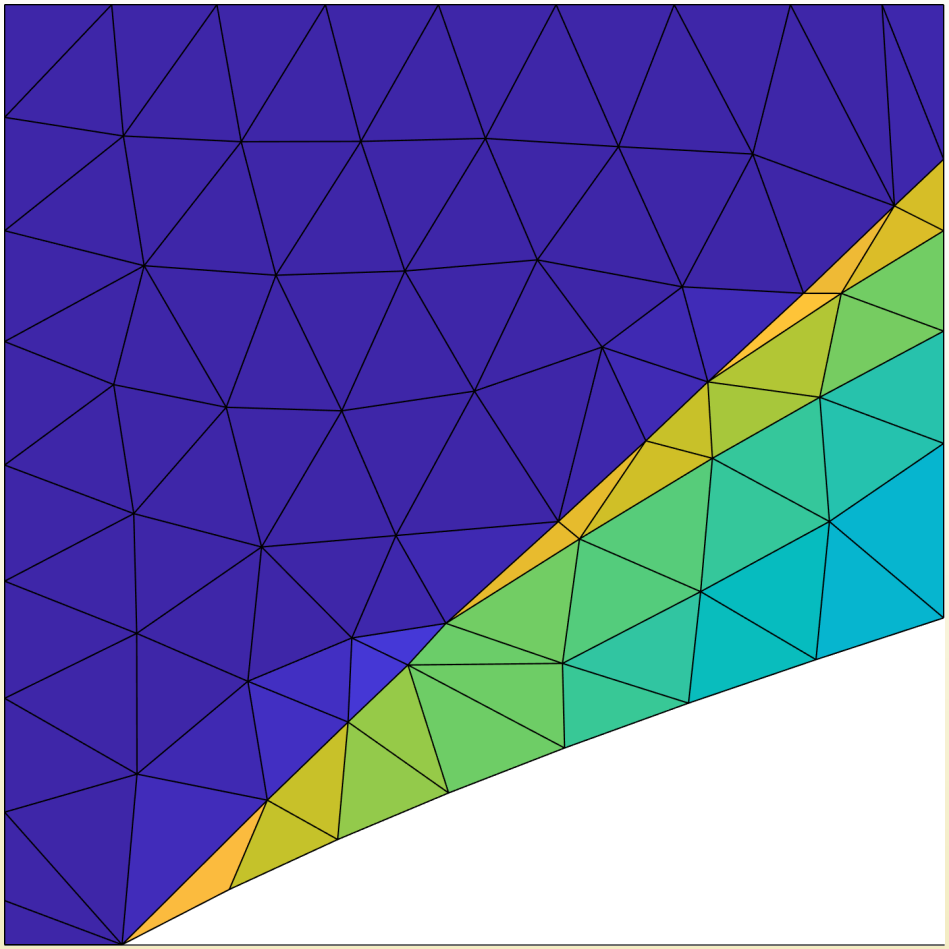
High-order implicit tracking: reacting inviscid flow

Original mesh: 100 triangular elements (unstructured)



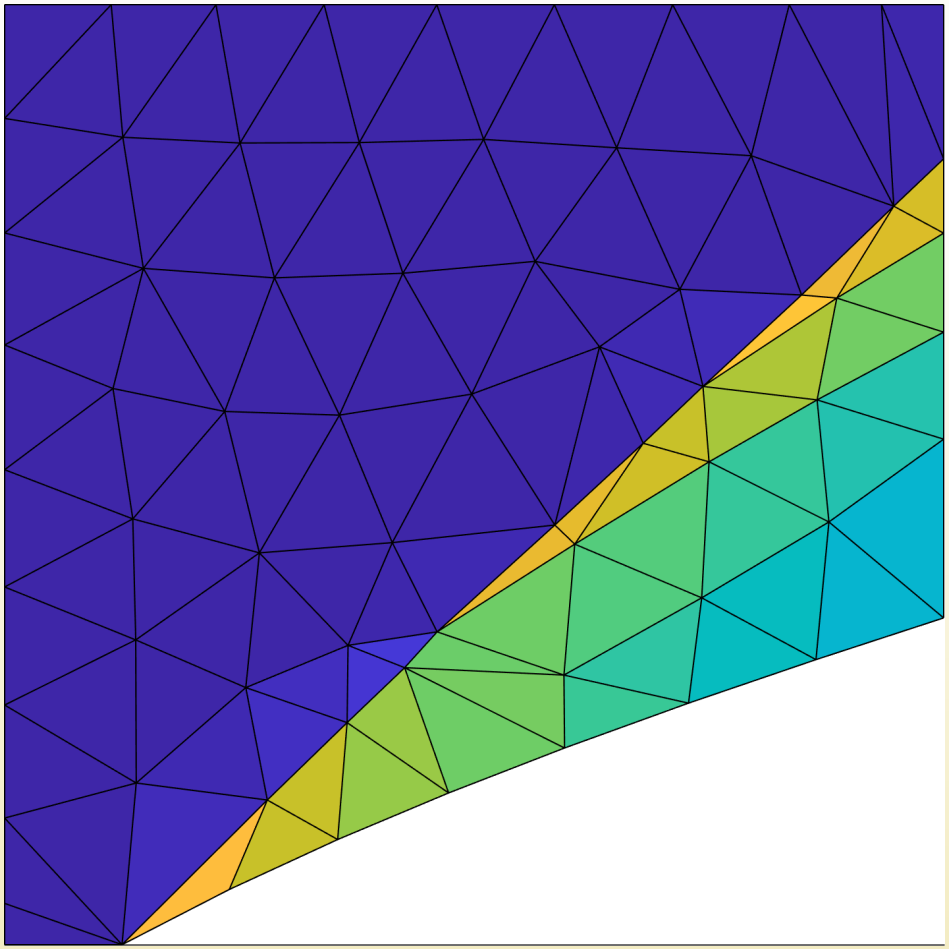
High-order implicit tracking: reacting inviscid flow

Original mesh: 100 triangular elements (unstructured)



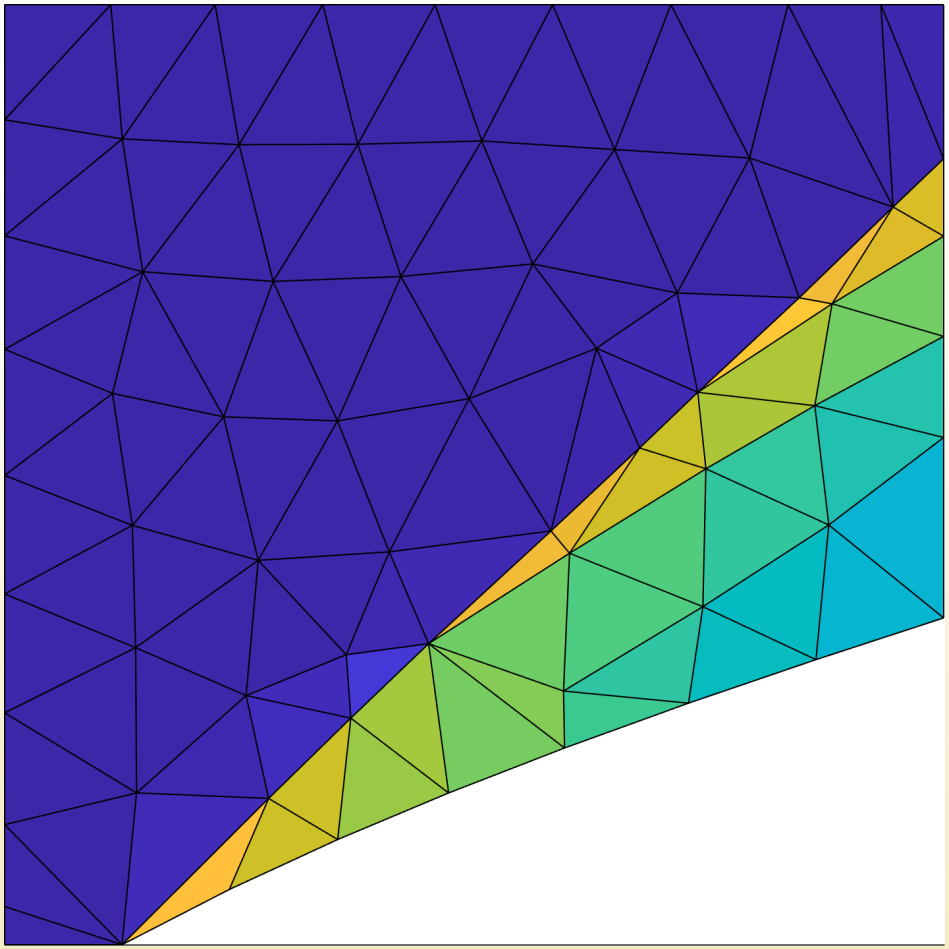
High-order implicit tracking: reacting inviscid flow

Original mesh: 100 triangular elements (unstructured)



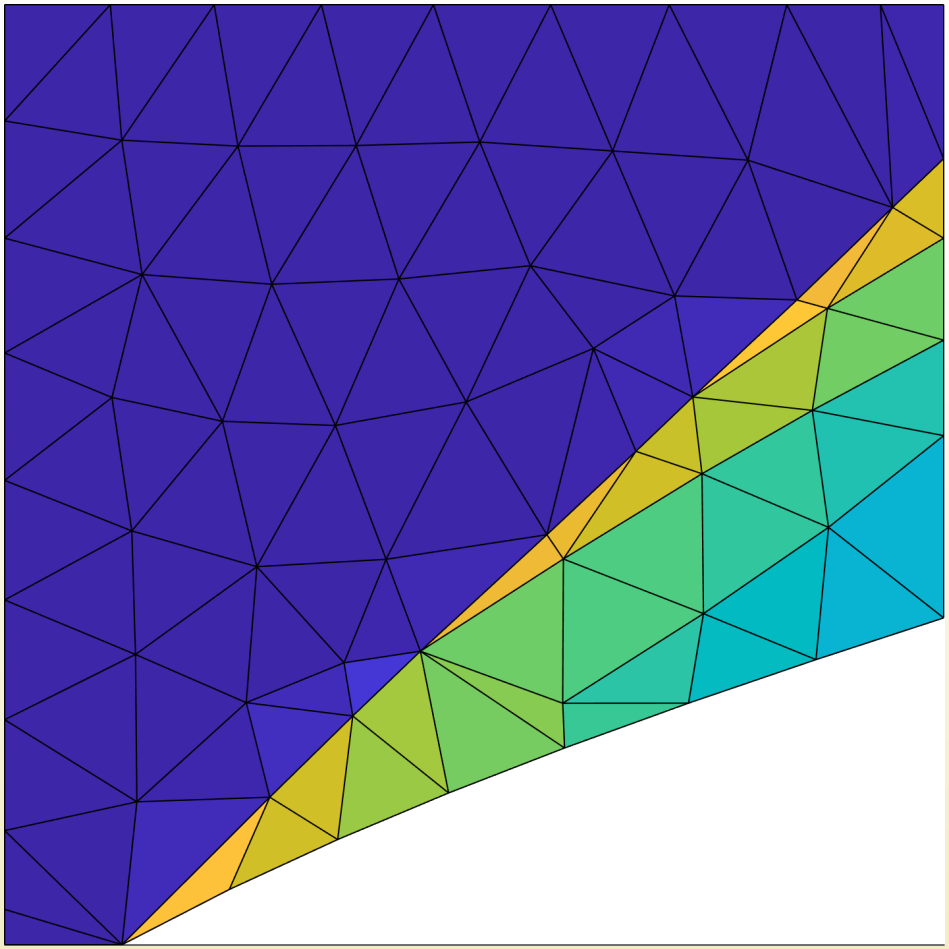
High-order implicit tracking: reacting inviscid flow

Original mesh: 100 triangular elements (unstructured)



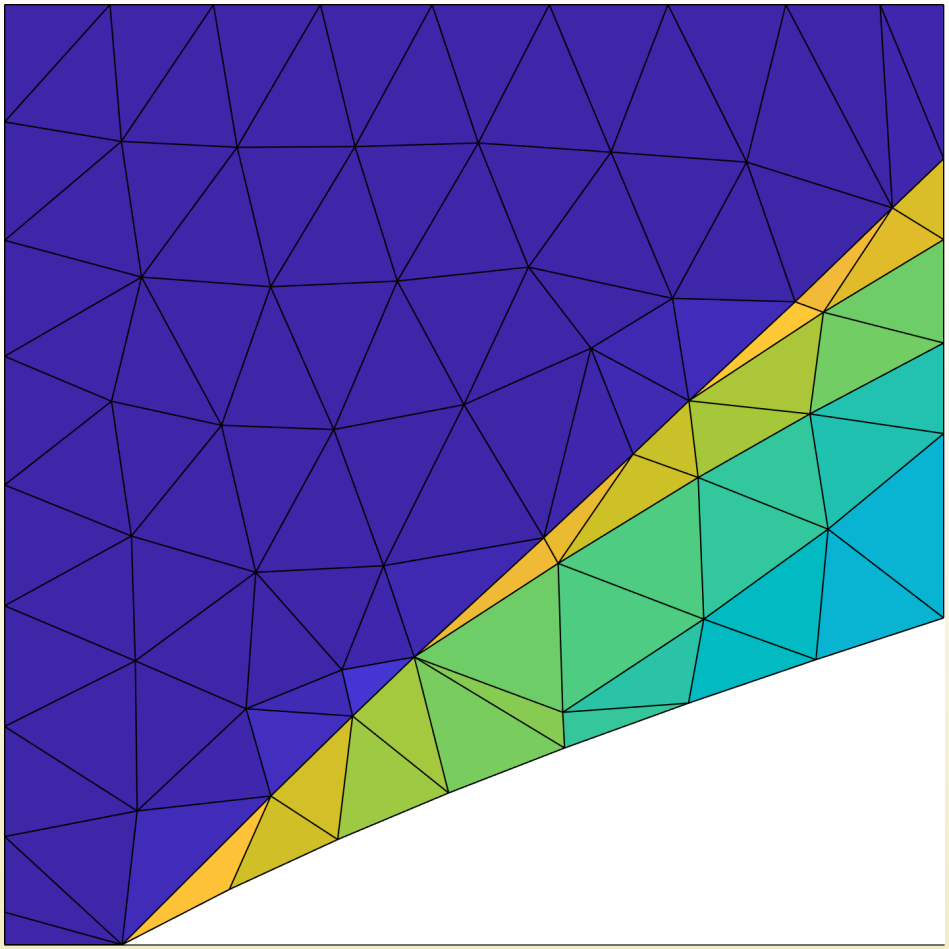
High-order implicit tracking: reacting inviscid flow

Original mesh: 100 triangular elements (unstructured)



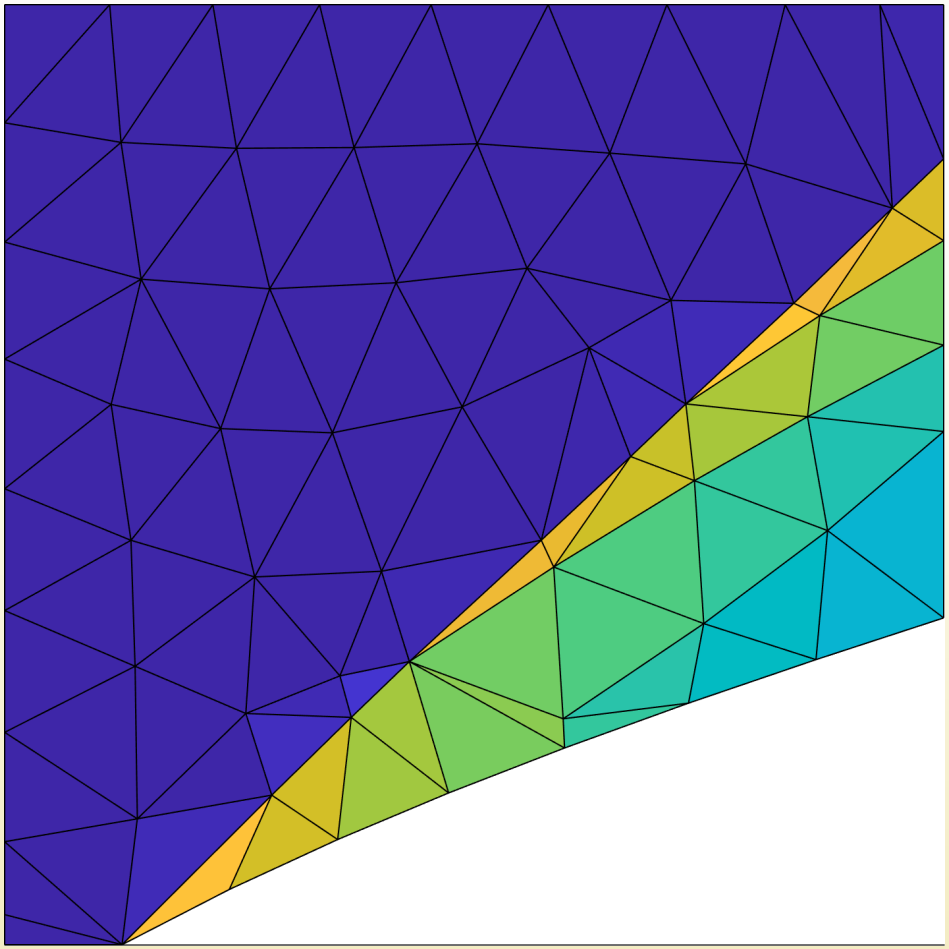
High-order implicit tracking: reacting inviscid flow

Original mesh: 100 triangular elements (unstructured)



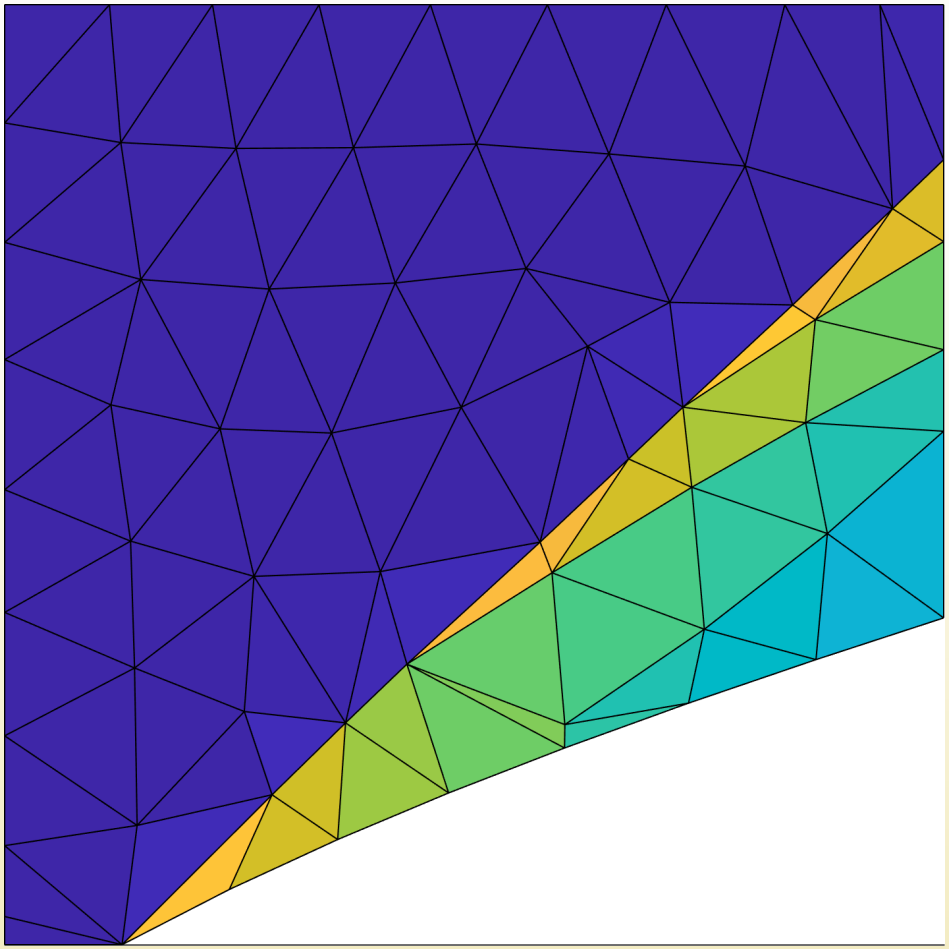
High-order implicit tracking: reacting inviscid flow

Original mesh: 100 triangular elements (unstructured)



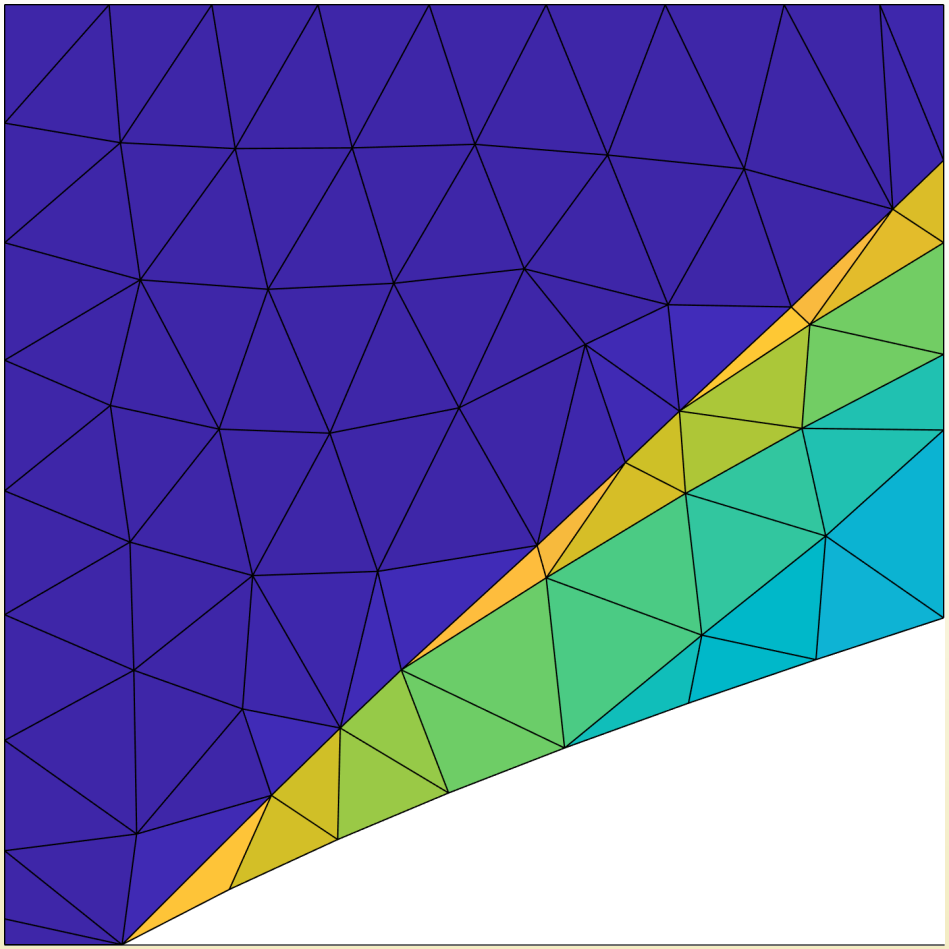
High-order implicit tracking: reacting inviscid flow

Original mesh: 100 triangular elements (unstructured)



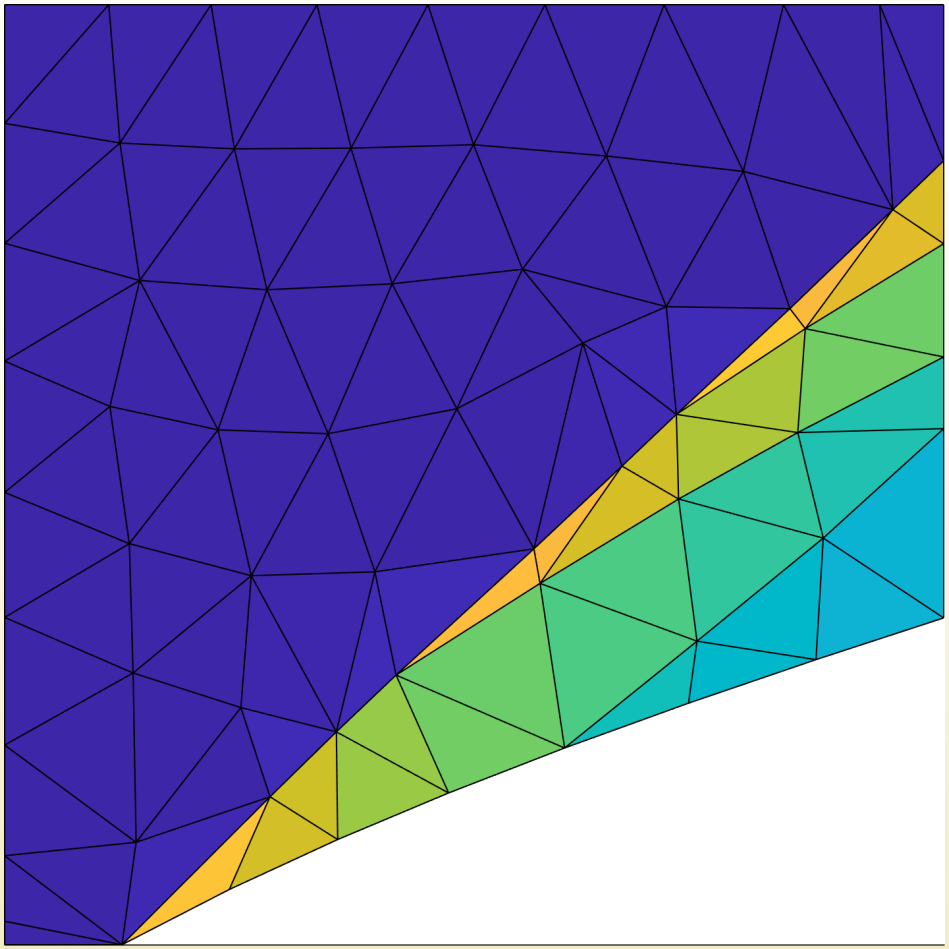
High-order implicit tracking: reacting inviscid flow

Original mesh: 100 triangular elements (unstructured)



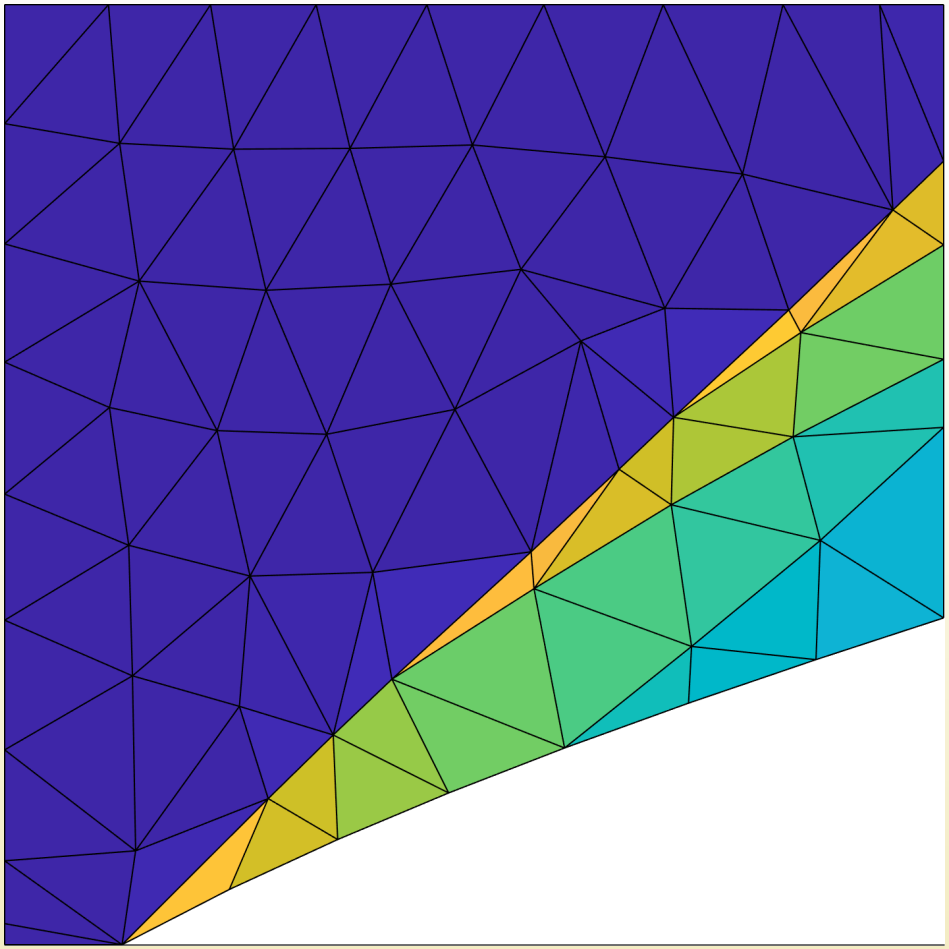
High-order implicit tracking: reacting inviscid flow

Original mesh: 100 triangular elements (unstructured)



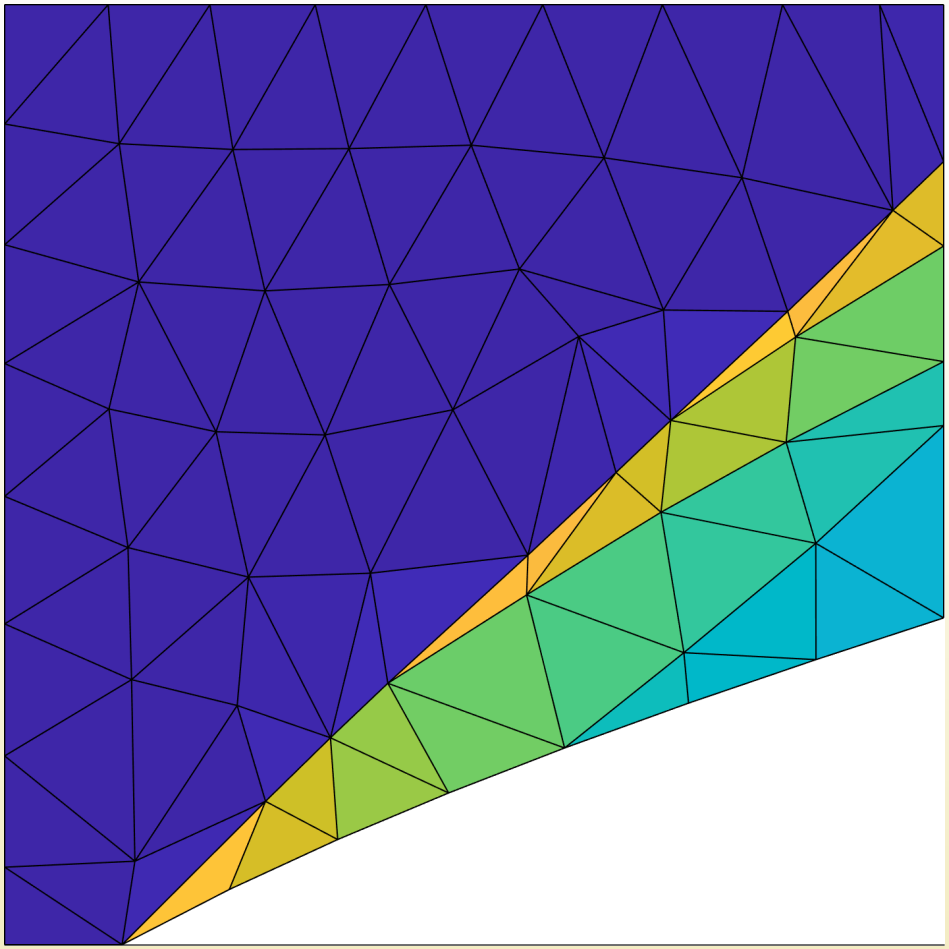
High-order implicit tracking: reacting inviscid flow

Original mesh: 100 triangular elements (unstructured)



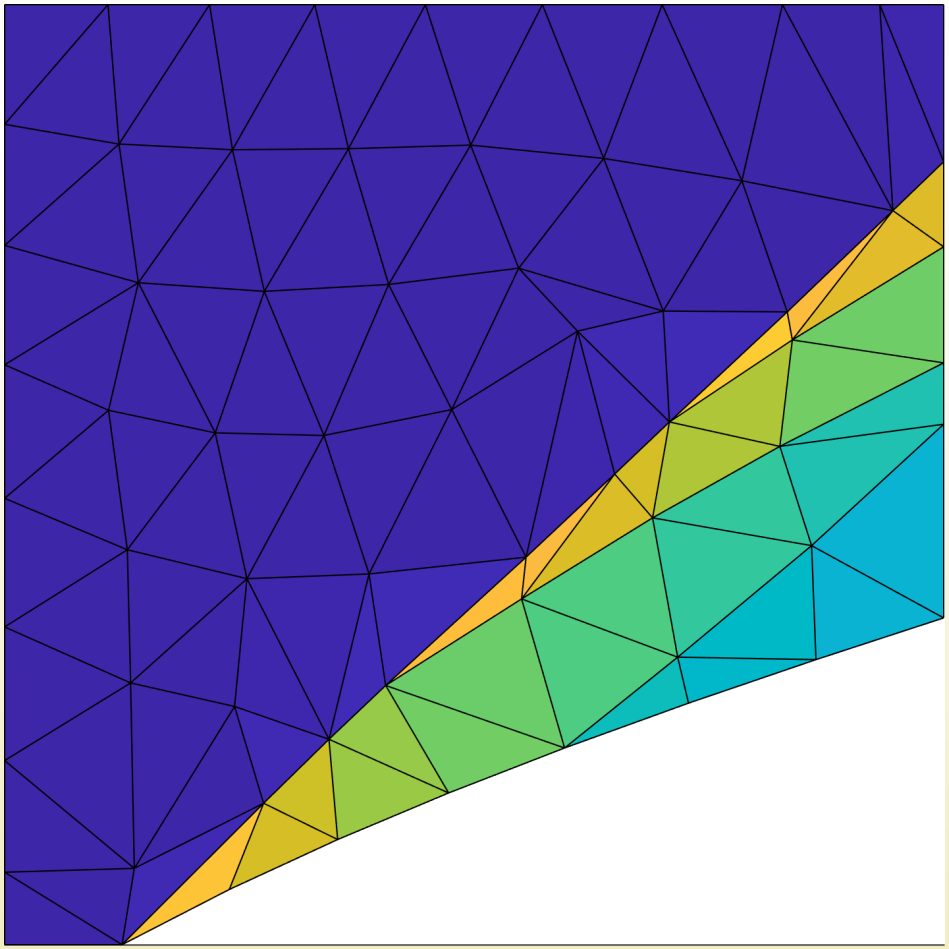
High-order implicit tracking: reacting inviscid flow

Original mesh: 100 triangular elements (unstructured)



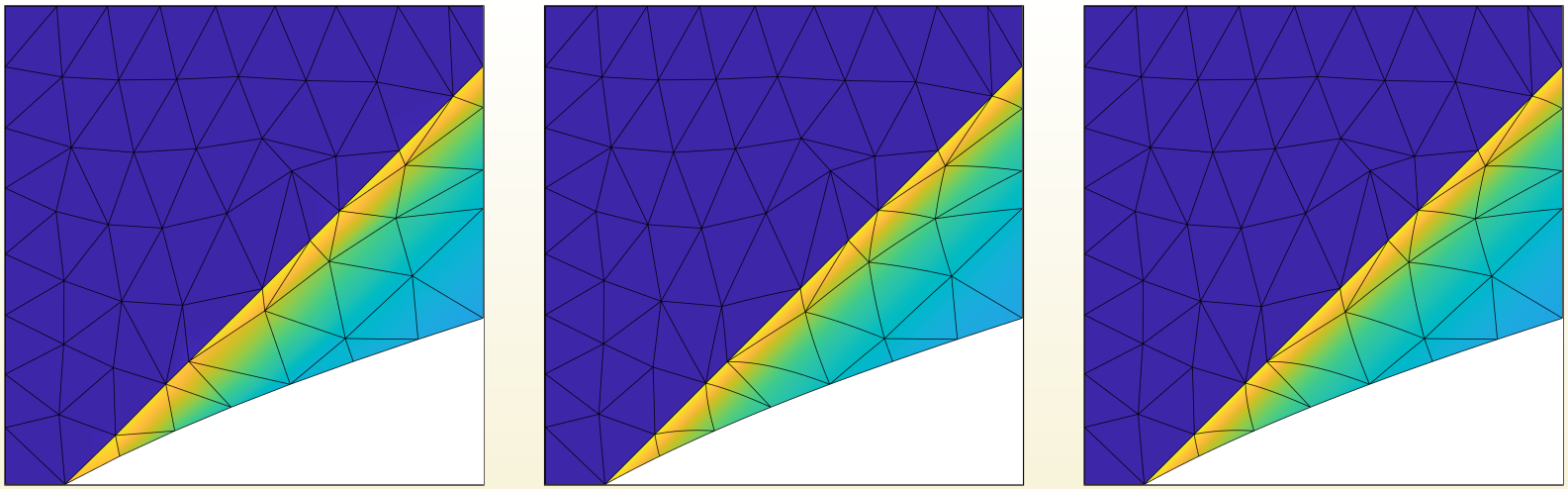
High-order implicit tracking: reacting inviscid flow

Original mesh: 100 triangular elements (unstructured)



Reacting inviscid flow: density (ρ)

$p = q = 1$ $p = q = 2$ $p = q = 3$



p: polynomial degree of solution
q: polynomial degree of mesh

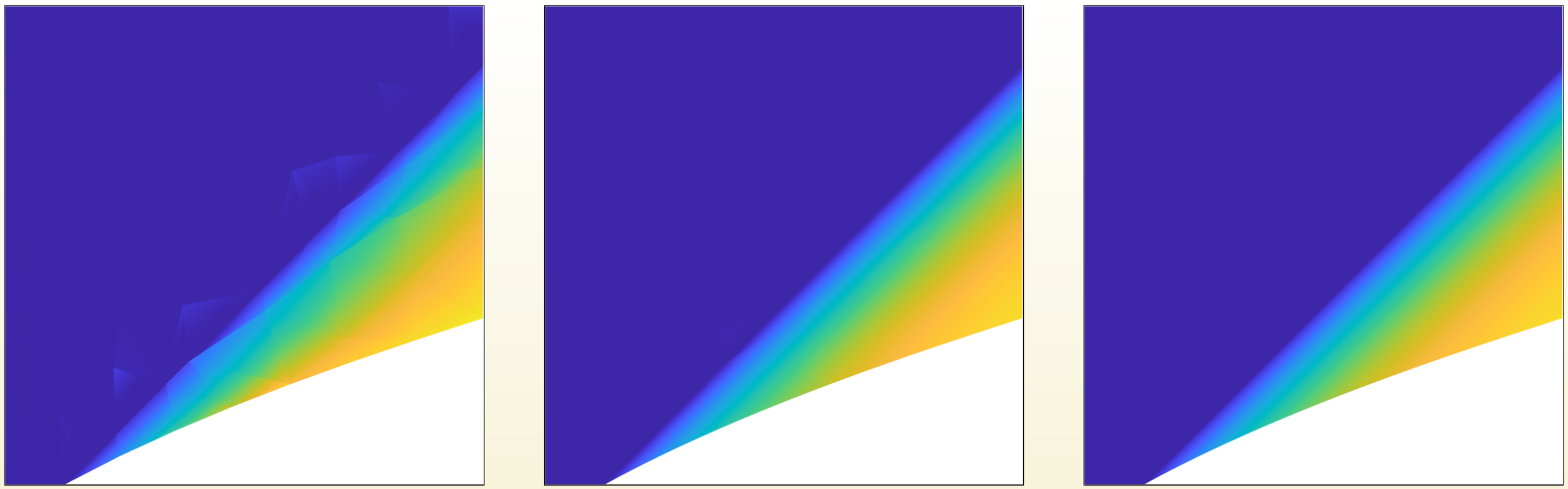
Reacting inviscid flow: density (ρ)

$p = q = 1$ $p = q = 2$ $p = q = 3$

p: polynomial degree of solution
q: polynomial degree of mesh

Reacting inviscid flow: reaction progress (λ)

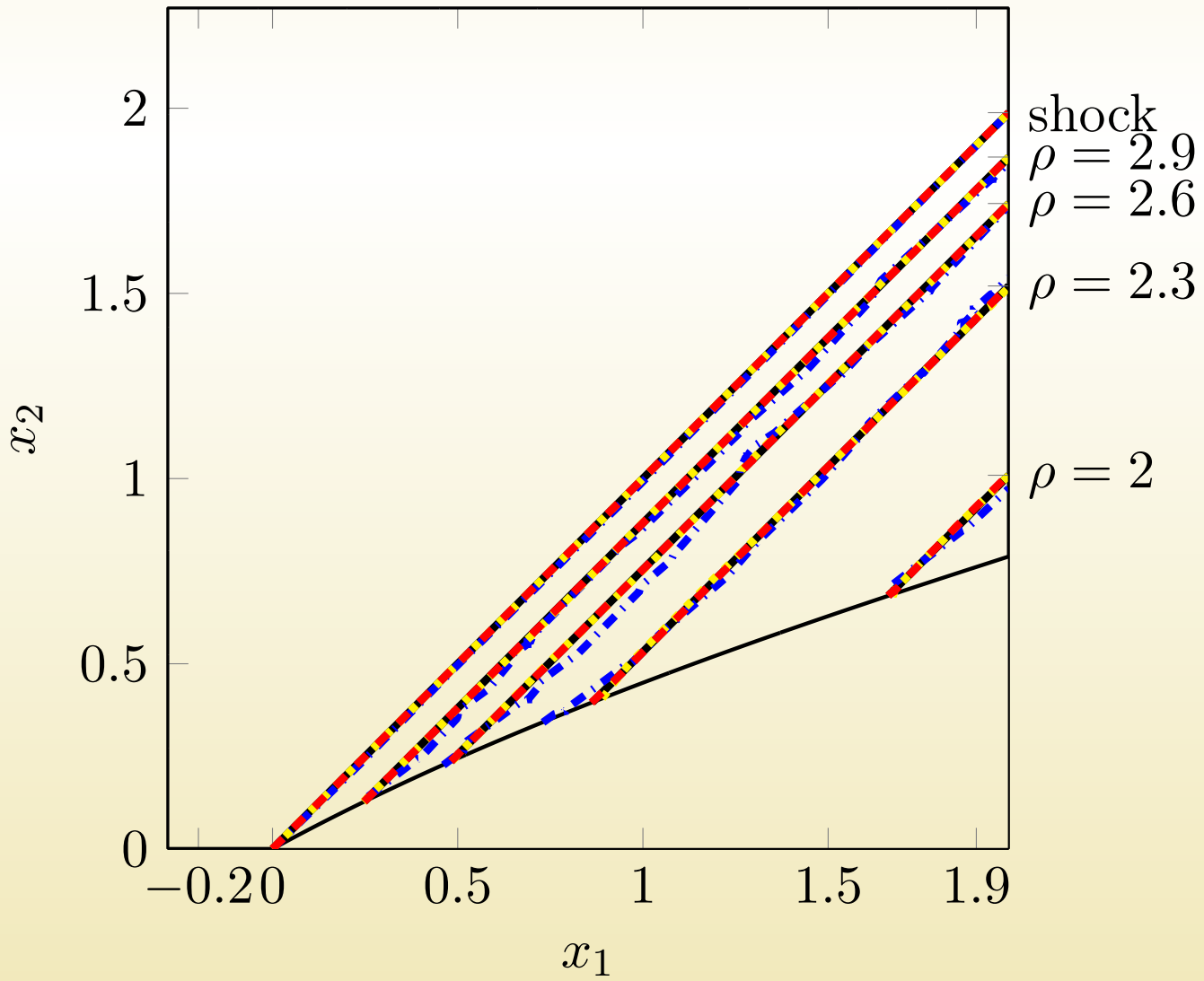
$p = q = 1$ $p = q = 2$ $p = q = 3$



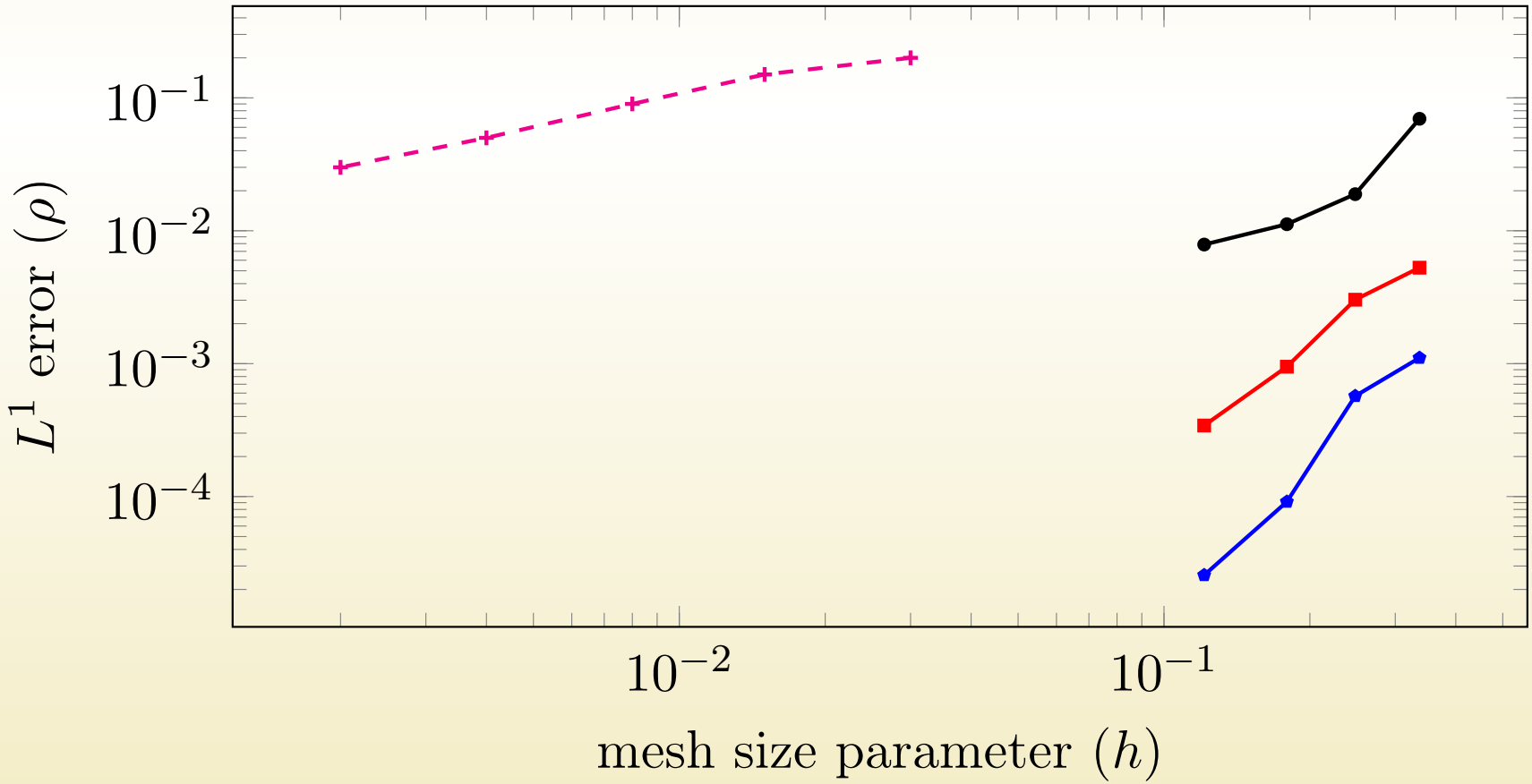
p: polynomial degree of solution
q: polynomial degree of mesh

Isolines of density indicate excellent agreement with analytical solution on coarse (100 element) mesh for $p > 1$

Analytical (—)
$p = 1$ (-·-·-)
$p = 2$ (·····)
$p = 3$ (- - -)



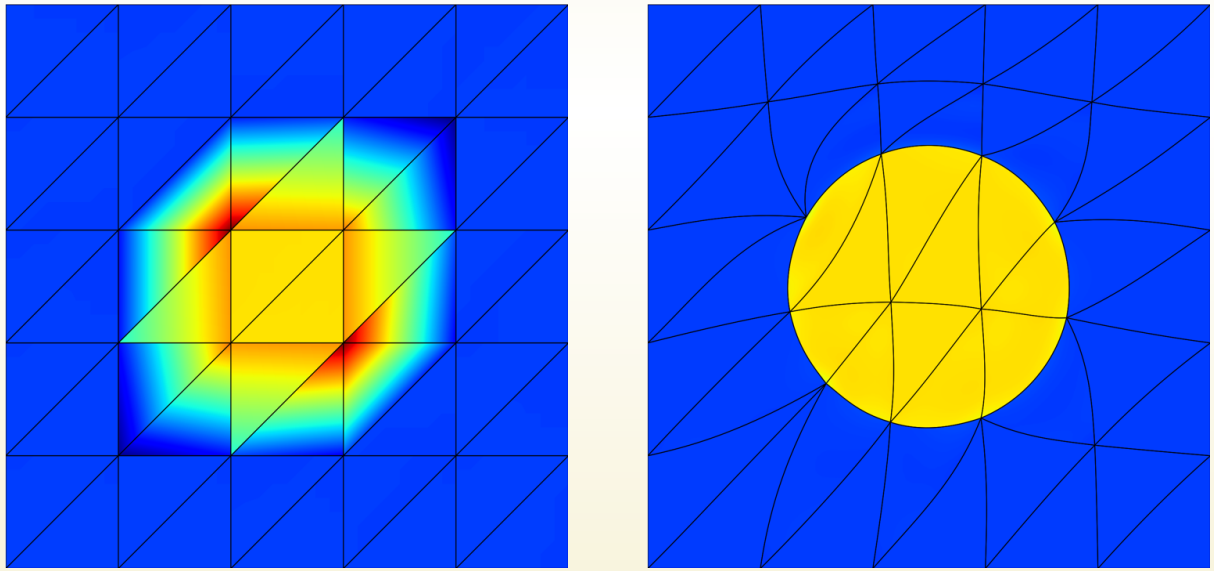
Implicit shock tracking recovers optimal $\mathcal{O}(h^{p+1})$ convergence rates; compares favorably to WENO



WENO (-+-), $p = 1$ (—●—), $p = 2$ (—■—), $p = 3$ (—●—)

Implicit shock tracking

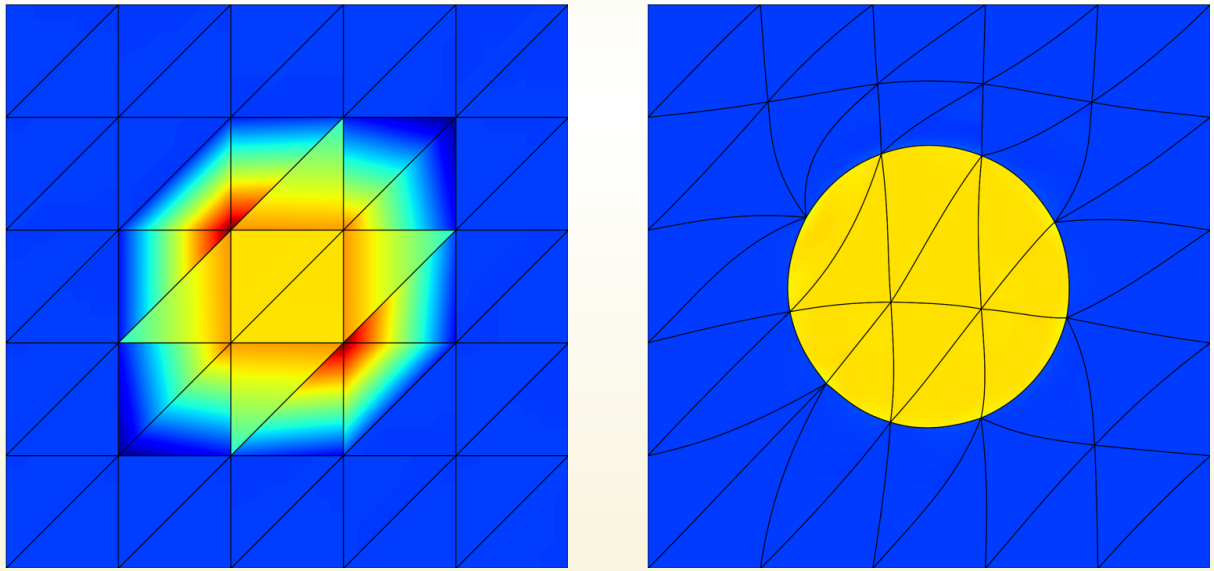
Goal: Align element faces with (unknown) discontinuities to perfectly capture them and approximate smooth regions to high-order



- DG discretization: inter-element jumps, high-order
- Discontinuity-aligned mesh is the solution of an optimization problem constrained by the discrete PDE $\implies$ **implicit tracking**
- Simultaneously converge solution and mesh to ensure solution of PDE never required on non-aligned mesh

Implicit shock tracking

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



- DG discretization: inter-element jumps, high-order
- Discontinuity-aligned mesh is the solution of an optimization problem constrained by the discrete PDE $\implies$ **implicit tracking**
- Simultaneously converge solution and mesh to ensure solution of PDE never required on non-aligned mesh

Discontinuous Galerkin discretization of conservation law

Inviscid conservation law:

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

Test $\mathcal{V}_{h,p'}$ and trial $\mathcal{V}_{h,p}$ spaces, where h is the mesh size and p/p' is the polynomial degree, to define the finite-dimensional DG residual:

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

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. The "standard" DG equations are: $\boldsymbol{r}(\boldsymbol{u}, \boldsymbol{x}) = \boldsymbol{0}$.

Discontinuous Galerkin discretization of conservation law

Inviscid conservation law:

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

Test $\mathcal{V}_{h,p'}$ and trial $\mathcal{V}_{h,p}$ spaces, where h is the mesh size and p/p' is the polynomial degree, to define the finite-dimensional DG residual:

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

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. The "standard" DG equations are: $\boldsymbol{r}(\boldsymbol{u}, \boldsymbol{x}) = \boldsymbol{0}$.

Discontinuous Galerkin discretization of conservation law

Inviscid conservation law:

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

Test $\mathcal{V}_{h,p'}$ and trial $\mathcal{V}_{h,p}$ spaces, where h is the mesh size and p/p' is the polynomial degree, to define the finite-dimensional DG residual:

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

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. The "standard" DG equations are: $\boldsymbol{r}(\boldsymbol{u}, \boldsymbol{x}) = \boldsymbol{0}$.

Implicit shock tracking: constrained optimization

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

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

We propose an objective function that *balances* the tracking objective with maintaining a high-quality mesh

$$f(\boldsymbol{u},\boldsymbol{x}) = \frac{1}{2}\boldsymbol{R}(\boldsymbol{u},\boldsymbol{x})^T\boldsymbol{R}(\boldsymbol{u},\boldsymbol{x}) + \frac{\kappa^2}{2}\boldsymbol{R}_{\text{msh}}(\boldsymbol{x})^T\boldsymbol{R}_{\text{msh}}(\boldsymbol{x}).$$

Implicit shock tracking: constrained optimization

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

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

We propose an objective function that *balances* the tracking objective with maintaining a high-quality mesh

$$f(\boldsymbol{u},\boldsymbol{x}) = \frac{1}{2}\boldsymbol{R}(\boldsymbol{u},\boldsymbol{x})^T\boldsymbol{R}(\boldsymbol{u},\boldsymbol{x}) + \frac{\kappa^2}{2}\boldsymbol{R}_{\text{msh}}(\boldsymbol{x})^T\boldsymbol{R}_{\text{msh}}(\boldsymbol{x}).$$

Implicit shock tracking: SQP 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 $\boldsymbol{z}_k$

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

where

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

$$\mathcal{L}(\boldsymbol{z}, \boldsymbol{\lambda}) = f(\boldsymbol{z}) - \boldsymbol{\lambda}^T \boldsymbol{r}(\boldsymbol{z}), \quad \hat{\boldsymbol{\lambda}}(\boldsymbol{z}) = \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}}(\boldsymbol{z})^{-T} \frac{\partial f}{\partial \boldsymbol{u}}(\boldsymbol{z})^T.$$

Implicit shock tracking: SQP 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 $\boldsymbol{z}_k$

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

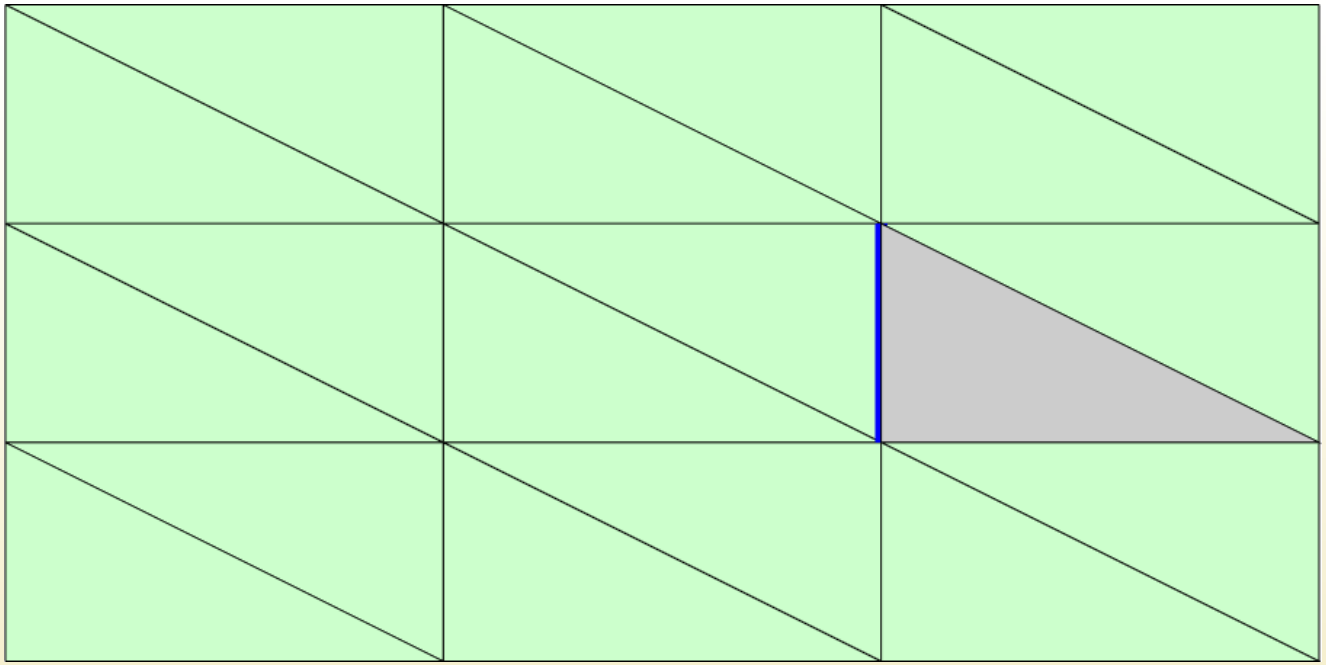
where

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

$$\mathcal{L}(\boldsymbol{z}, \boldsymbol{\lambda}) = f(\boldsymbol{z}) - \boldsymbol{\lambda}^T \boldsymbol{r}(\boldsymbol{z}), \quad \hat{\boldsymbol{\lambda}}(\boldsymbol{z}) = \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}}(\boldsymbol{z})^{-T} \frac{\partial f}{\partial \boldsymbol{u}}(\boldsymbol{z})^T.$$

Practical considerations: element collapse

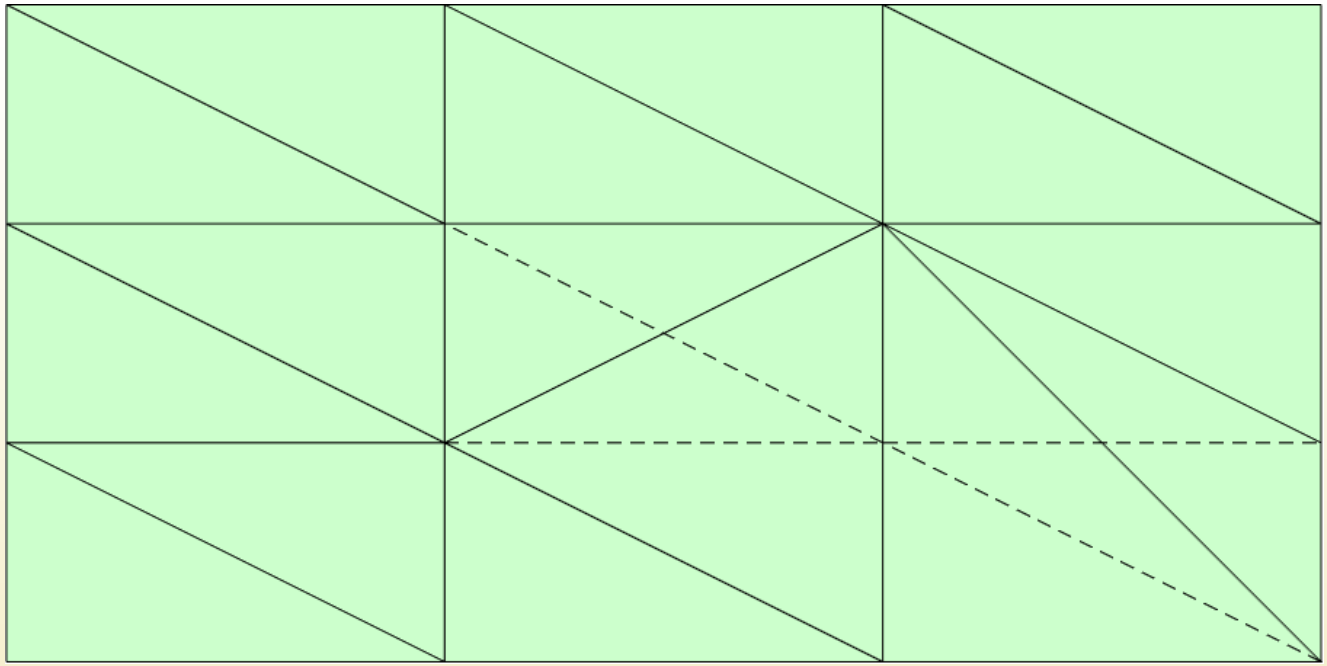
Despite measures to keep mesh well-conditioned, best option may be to *remove* element from the mesh: tag elements for removal based on volume, collapse shortest edge, trivial DG solution transfer



- Only meshing operation required by implicit shock tracking
- Simplices: arbitrary dimensions, arbitrary polynomial degree

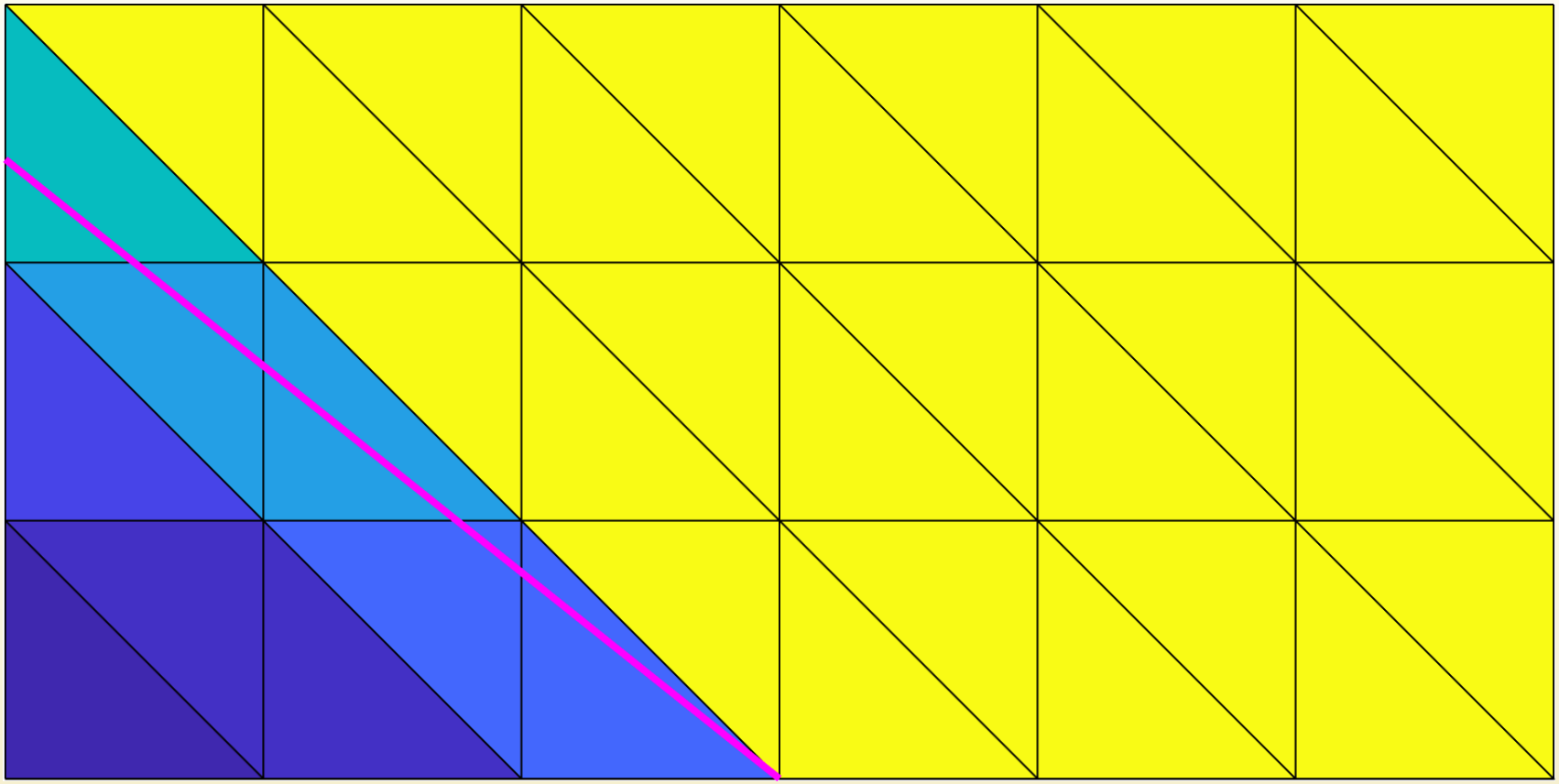
Practical considerations: element collapse

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



- Only meshing operation required by implicit shock tracking
- Simplices: arbitrary dimensions, arbitrary polynomial degree

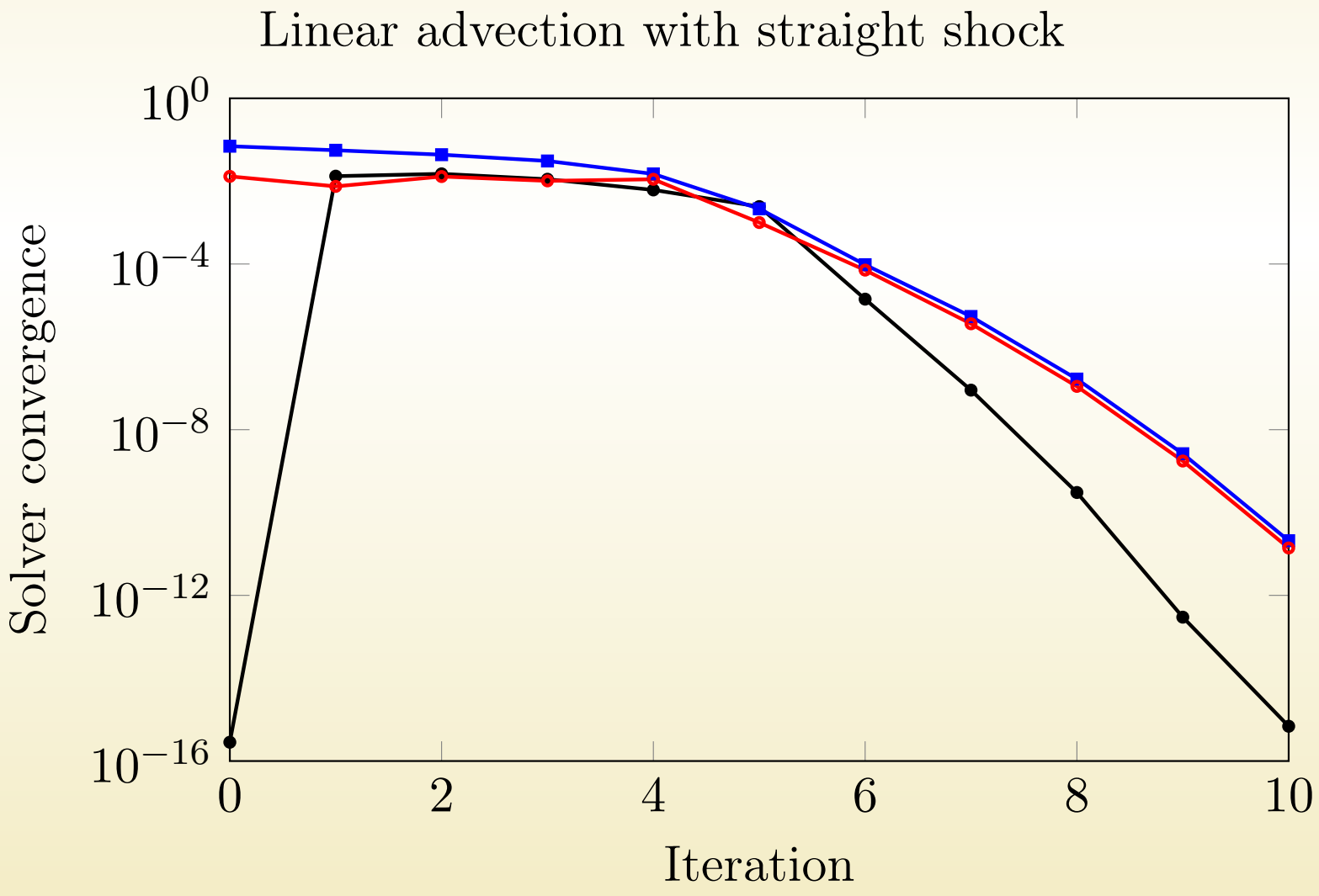
Linear advection, straight shock



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

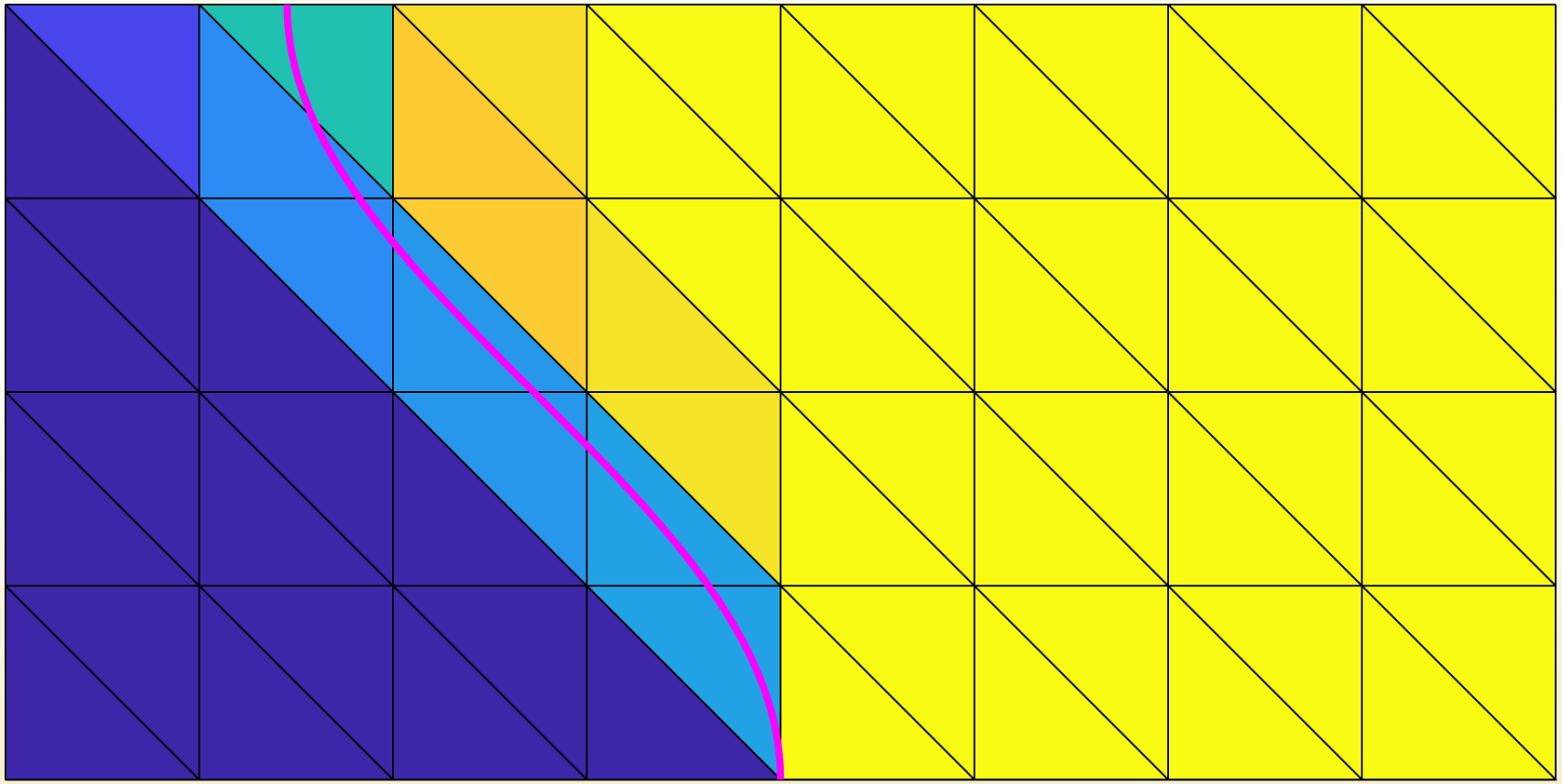
L_1 error $= 3.84 \times 10^{-11}$

Newton-like convergence when solution lies in DG space



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

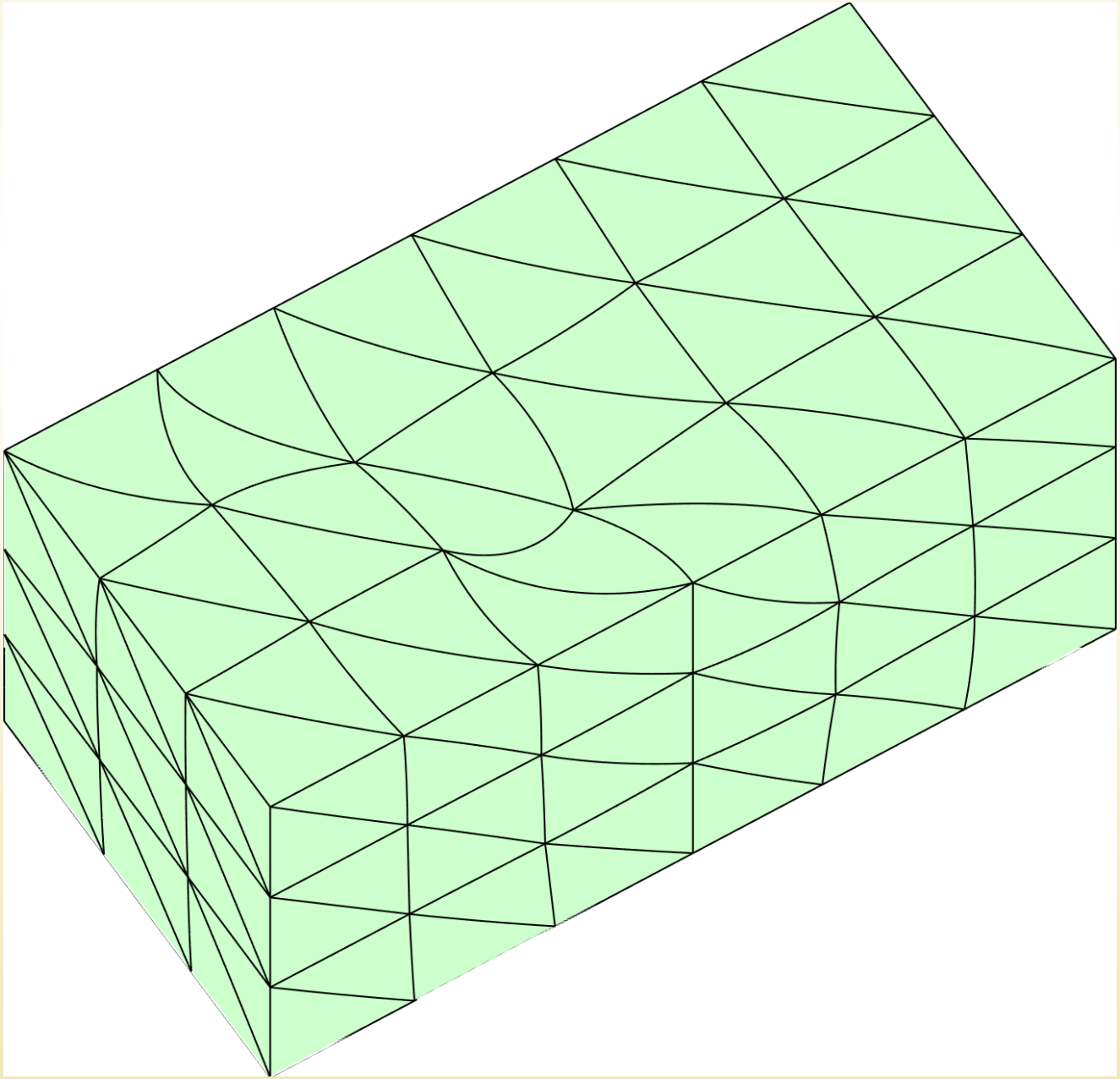
Linear advection, trigonometric shock



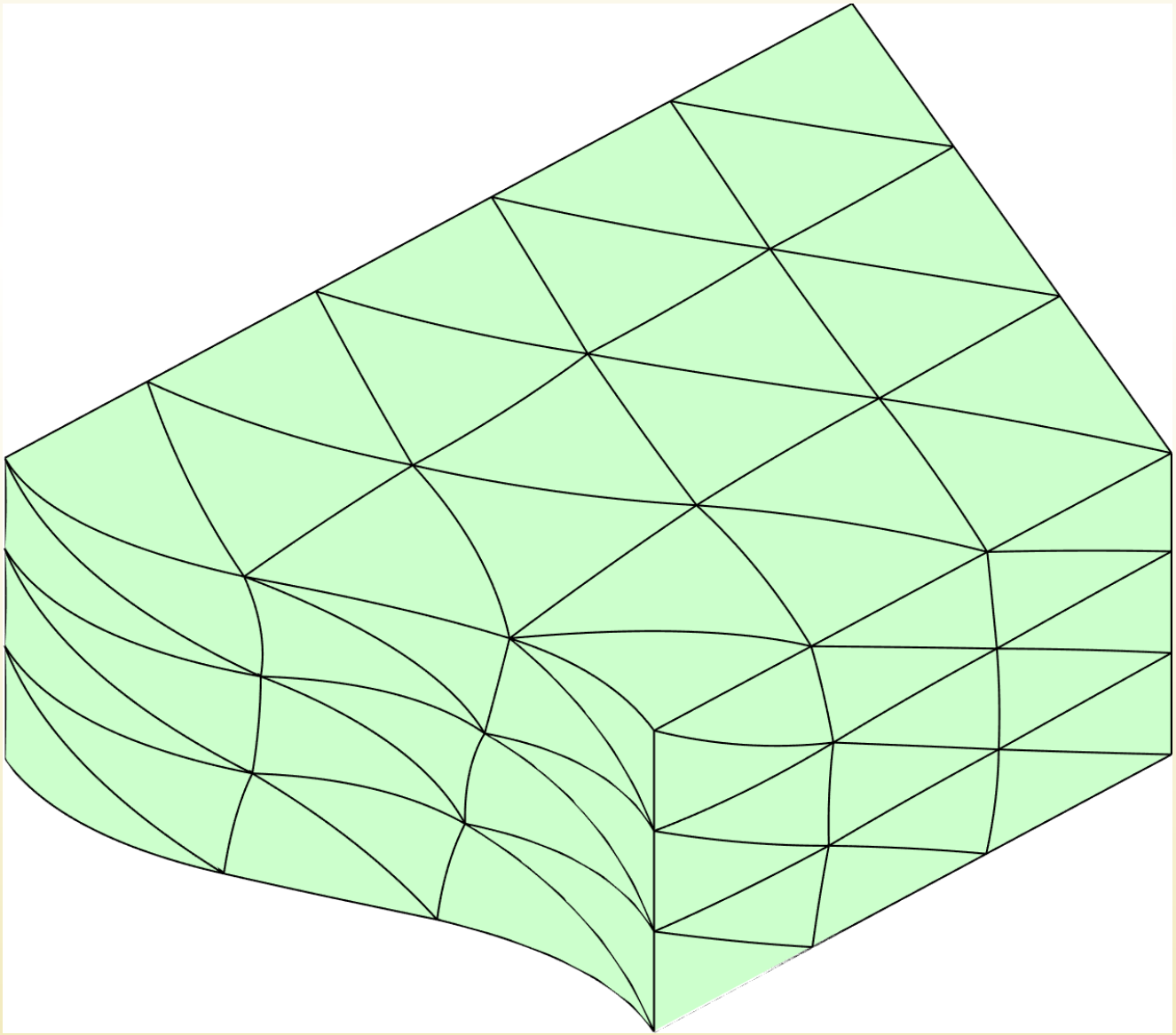
$p = 0$ space for solution, $q = 2$ space for mesh

L_1 error $= 1.15 \times 10^{-3}$

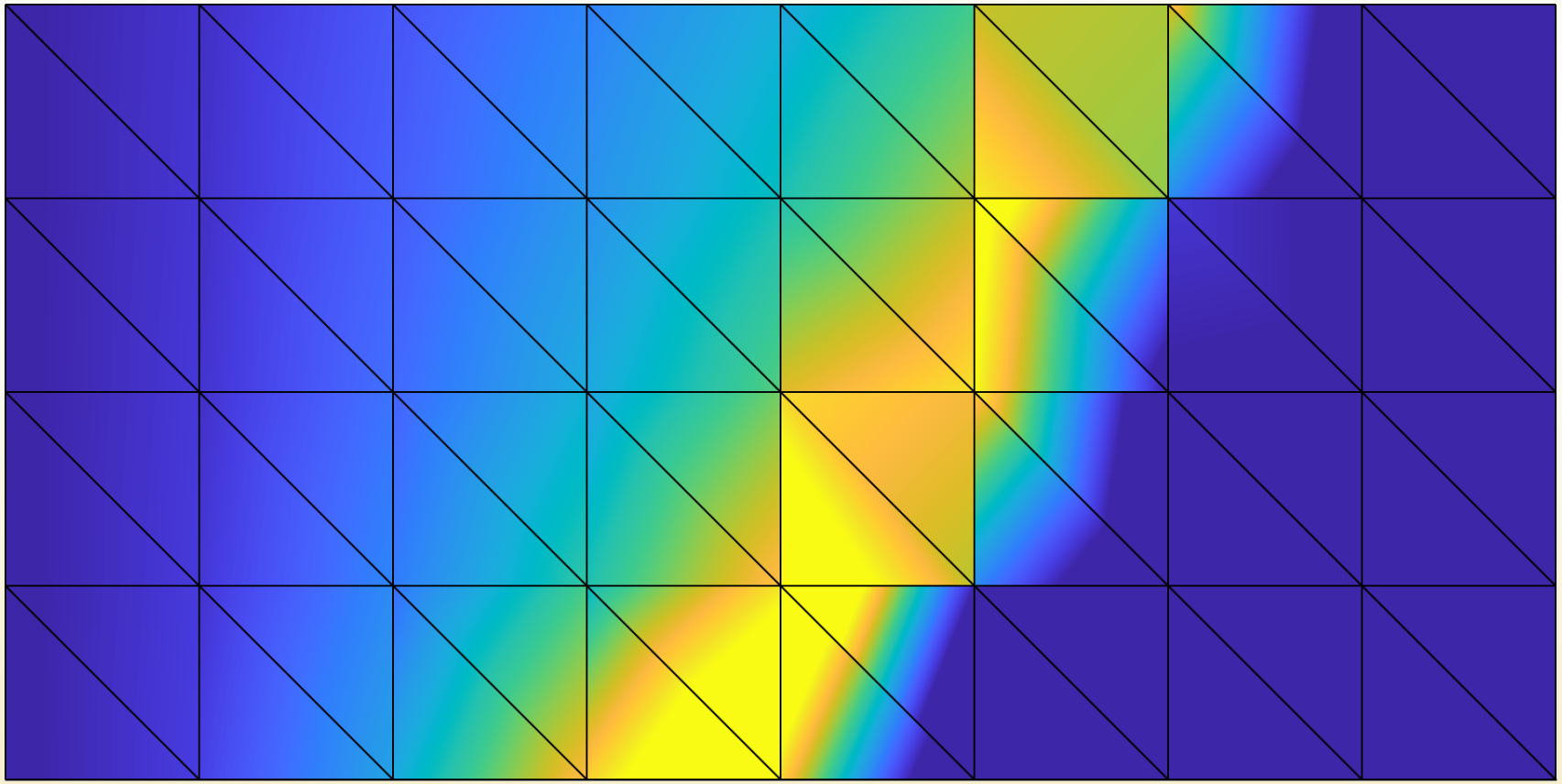
Linear advection, trigonometric shock, 3D



Linear advection, trigonometric shock, 3D

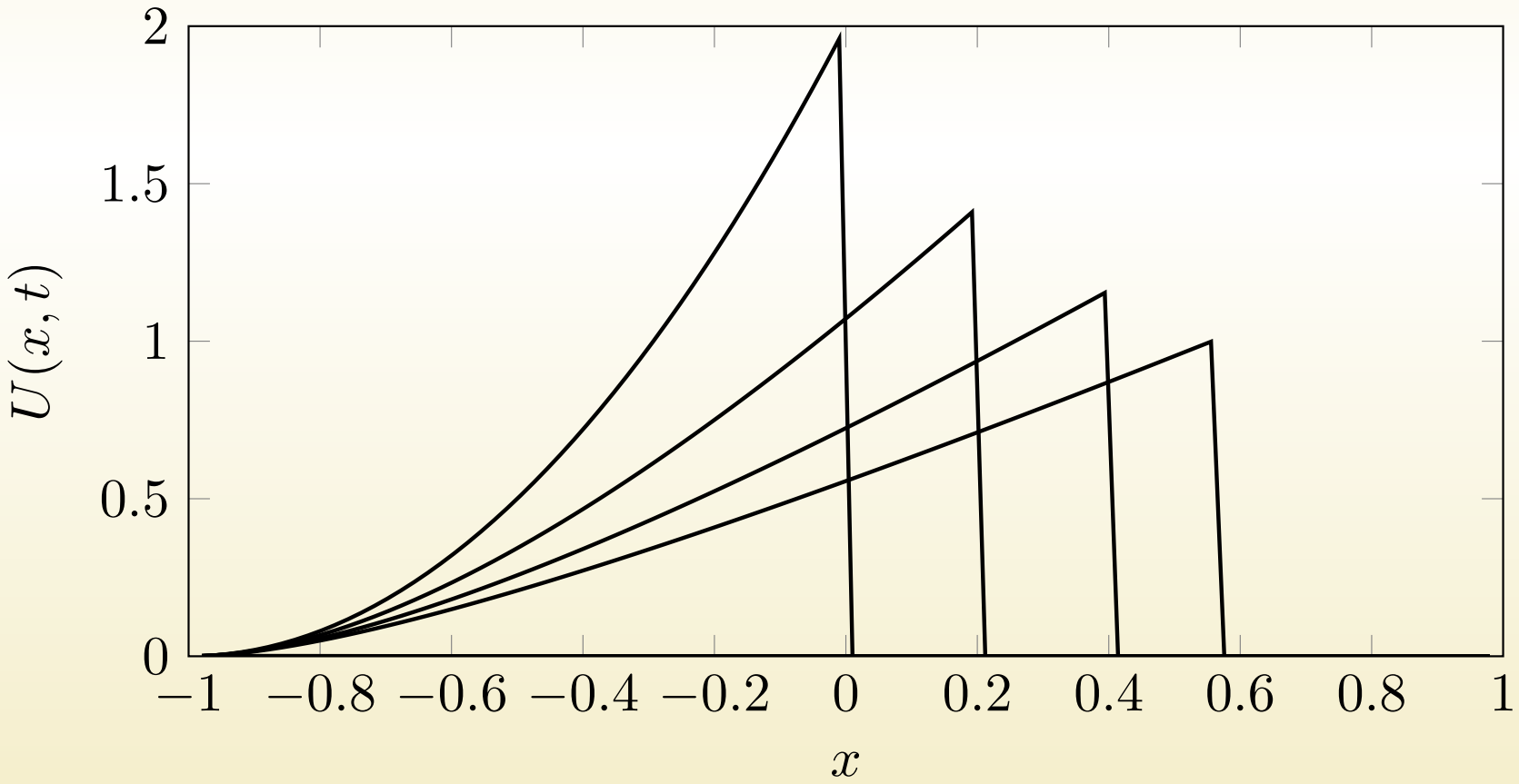


Inviscid Burgers' equation: space-time formulation



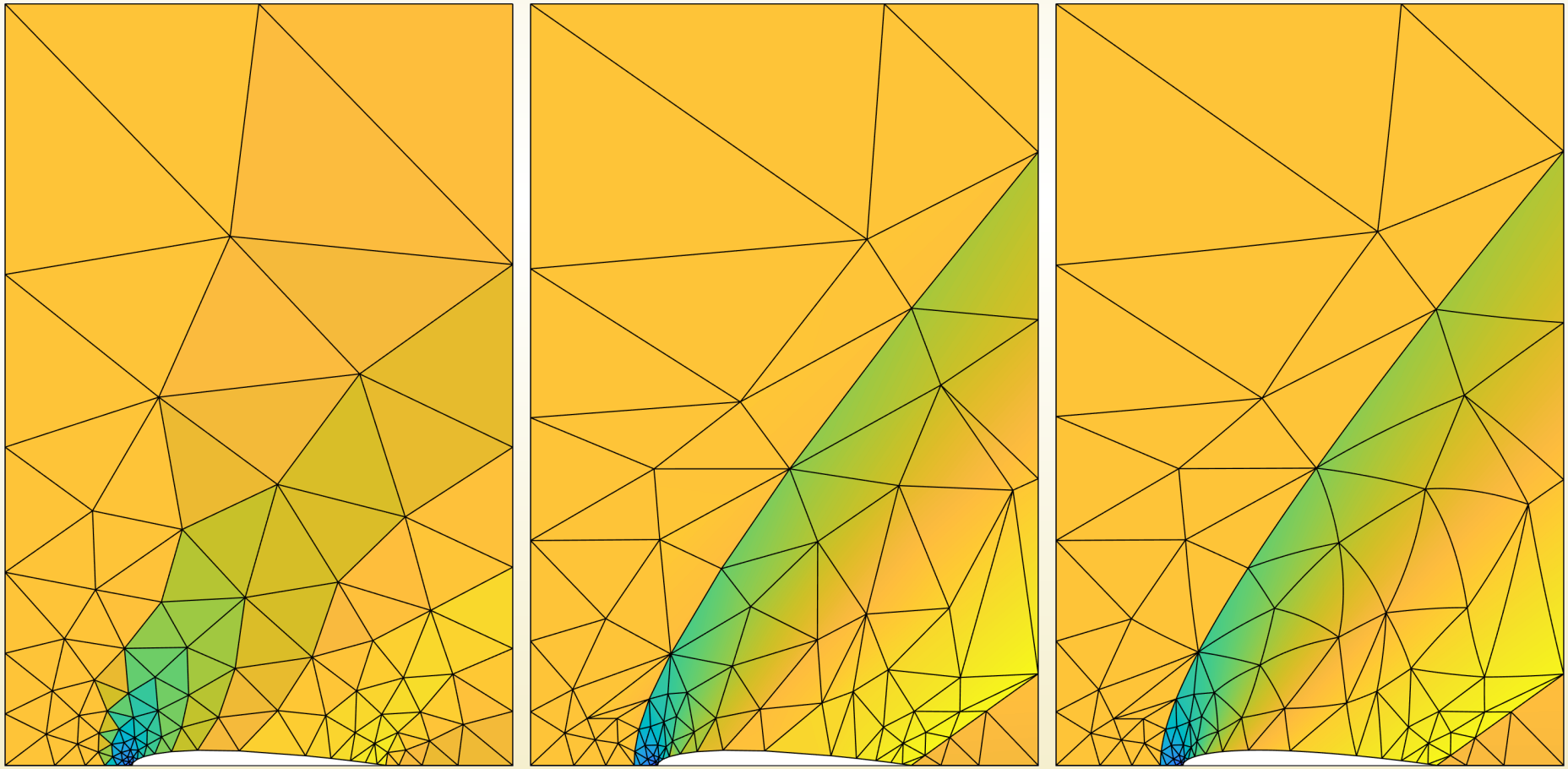
$p = 0$ space for solution, $q = 3$ space for mesh

Temporal snapshots show discontinuity perfectly represented, smooth solution well-approximated elsewhere



Temporal snapshots of inviscid Burgers' equation from $p = 4$ implicit tracking simulation

Supersonic flow past NACA0012 airfoil ($M = 1.5$)



Initialization

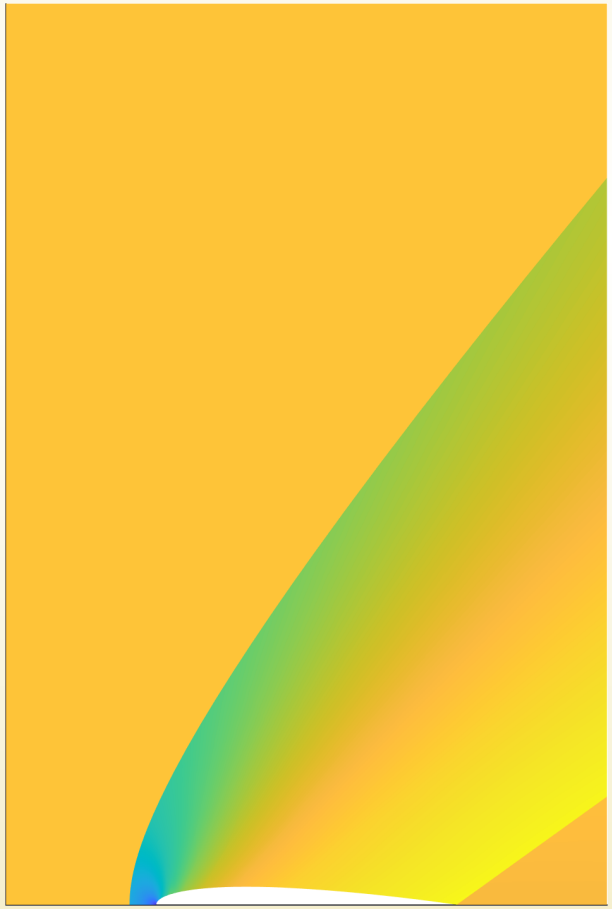
$p = 1$ tracking
$e_H = 1.30 \times 10^{-3}$

$p = 2$ tracking
$e_H = 6.73 \times 10^{-5}$

Supersonic flow past NACA0012 airfoil ($M = 1.5$)

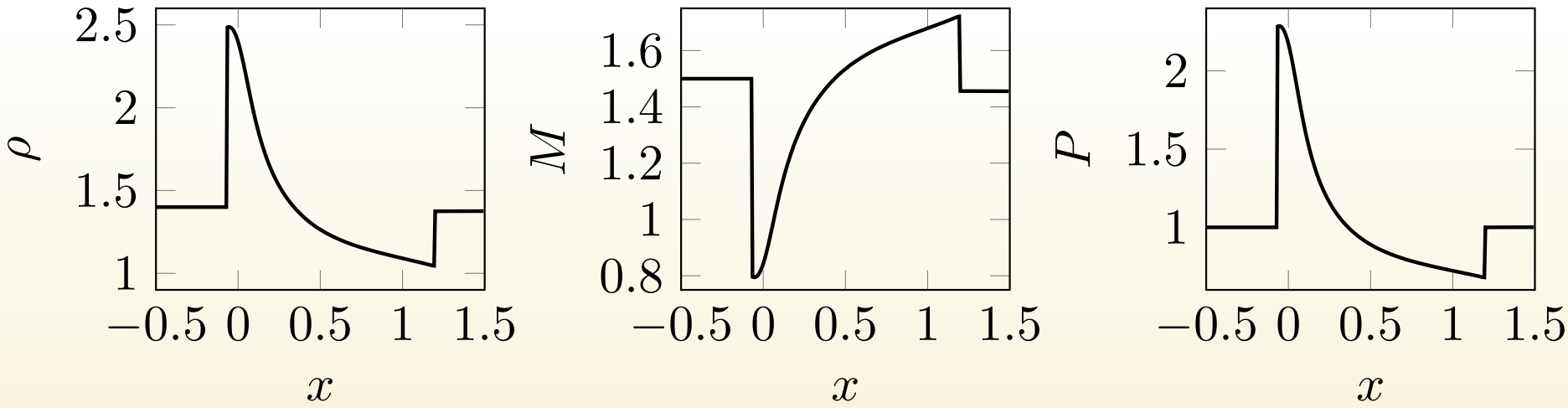
Initialization

$p = 1$ tracking
$e_H = 1.30 \times 10^{-3}$



$p = 2$ tracking
$e_H = 6.73 \times 10^{-5}$

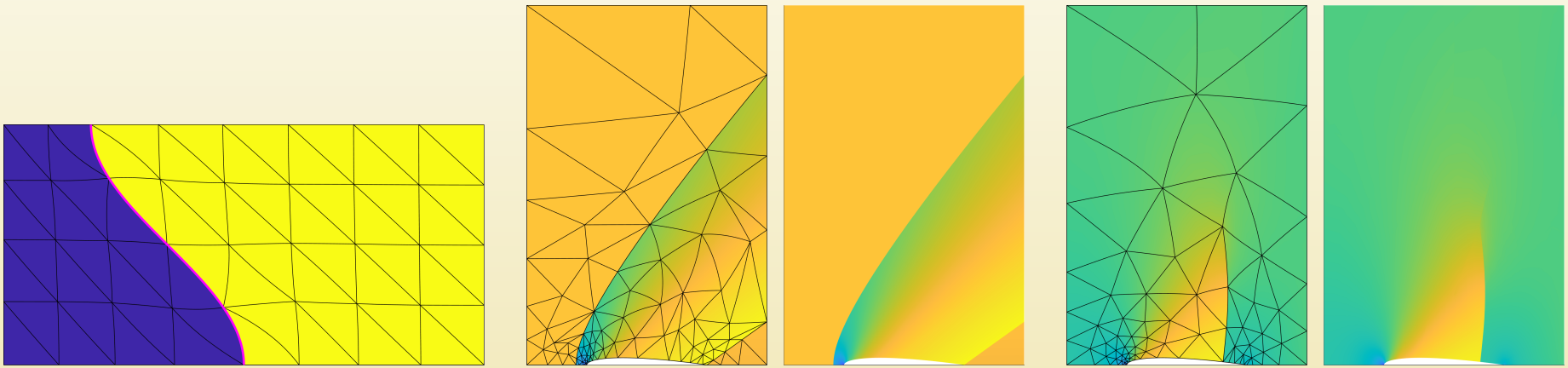
Spatial slices show both discontinuities are tracked and solution well-approximated elsewhere



Spatial slice of supersonic flow past NACA0012 airfoil ($M = 1.5$) from $p = 3$ implicit tracking simulation

High-order, implicit shock tracking

- **Primary benefit**: highly *accurate solutions* on coarse meshes, recover *optimal convergence* rates
- Traditional barrier to tracking (explicitly meshing unknown discontinuity surface) replaced with solving constrained optimization problem
- Future extensions: 3D, viscous, time-dependent, **model reduction**, **hypersonics**
- Zahr, Shi, Persson, *Journal of Computational Physics*, 2020.



Implicit shock tracking for hypersonics applications

Strong local features: shock waves and contact discontinuities

- Improved solver robustness (time-stepping, vanishing viscosity)

Viscous effects: thin boundary layer, critical to predict heating

- Extension to viscous problems straightforward
- Expect r-adaptivity near boundary layer to improve approximation

Interacting features: shock/shock and shock/boundary layer

- Handled by optimization formulation and element collapse

Unsteady: prediction of combustion stability

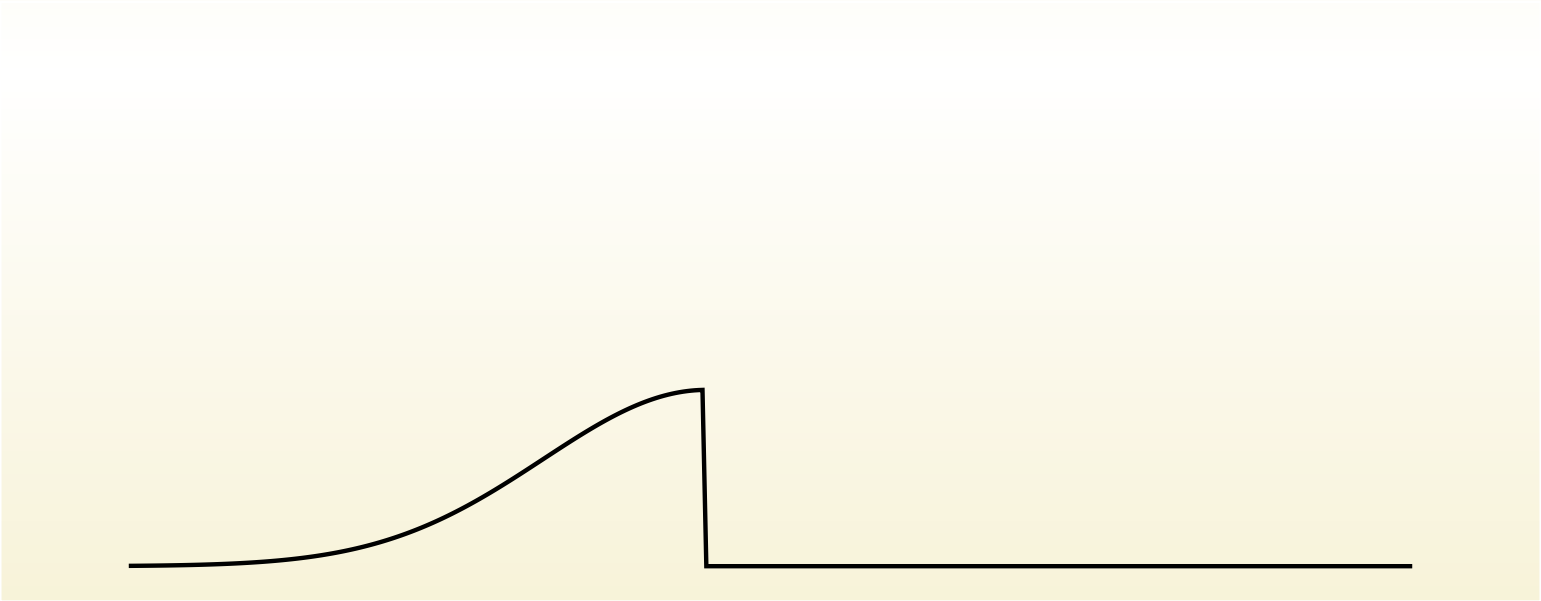
- Slab-based space-time formulation (moving features)
- Method-of-lines-based formulation (stationary features)

Multi-scale/physics: turbulence, ablation, conjugate heat transfer

- Clip objective function to avoid tracking all turbulent features
- Integrate in partitioned multiphysics setting

Reduction of parametrized discontinuities

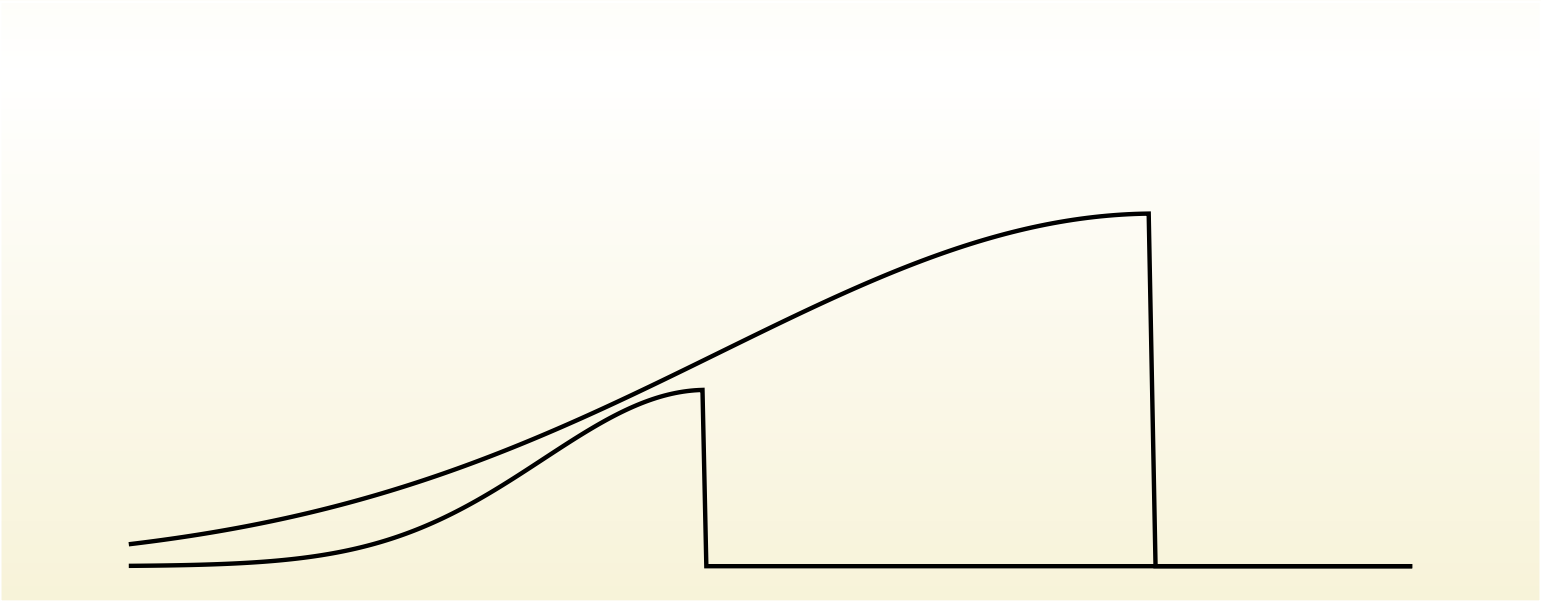
Fundamental issue: linear subspace approximation ill-suited for advection-dominated features (slowly decay Kolmogorov n-width)



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

Reduction of parametrized discontinuities

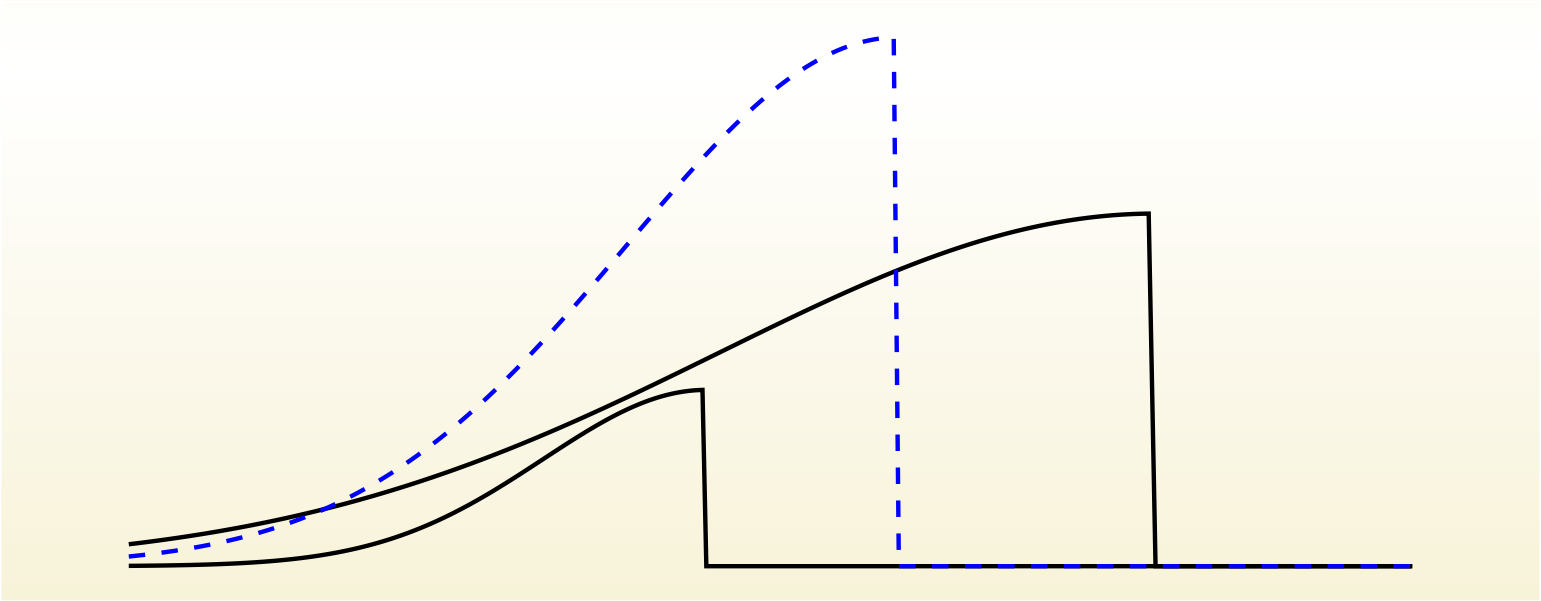
Fundamental issue: linear subspace approximation ill-suited for advection-dominated features (slowly decay Kolmogorov n-width)



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

Reduction of parametrized discontinuities

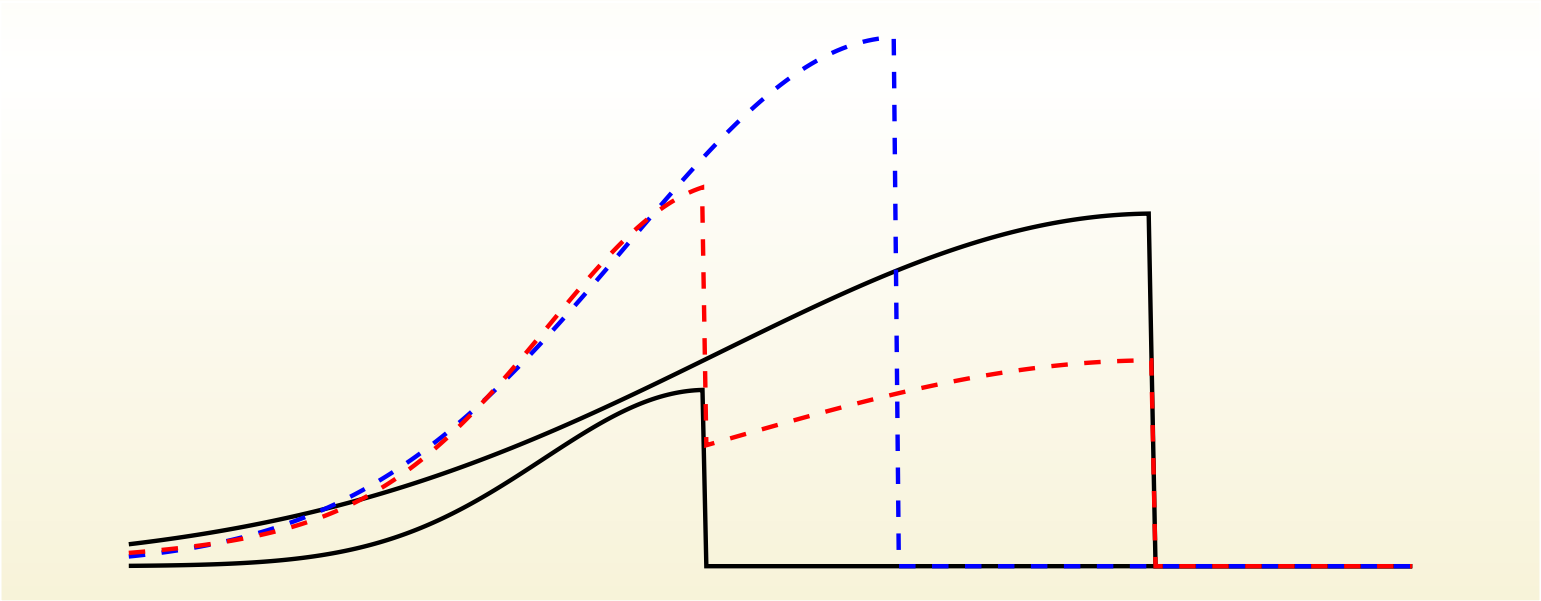
Fundamental issue: linear subspace approximation ill-suited for advection-dominated features (slowly decay Kolmogorov n-width)



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

Reduction of parametrized discontinuities

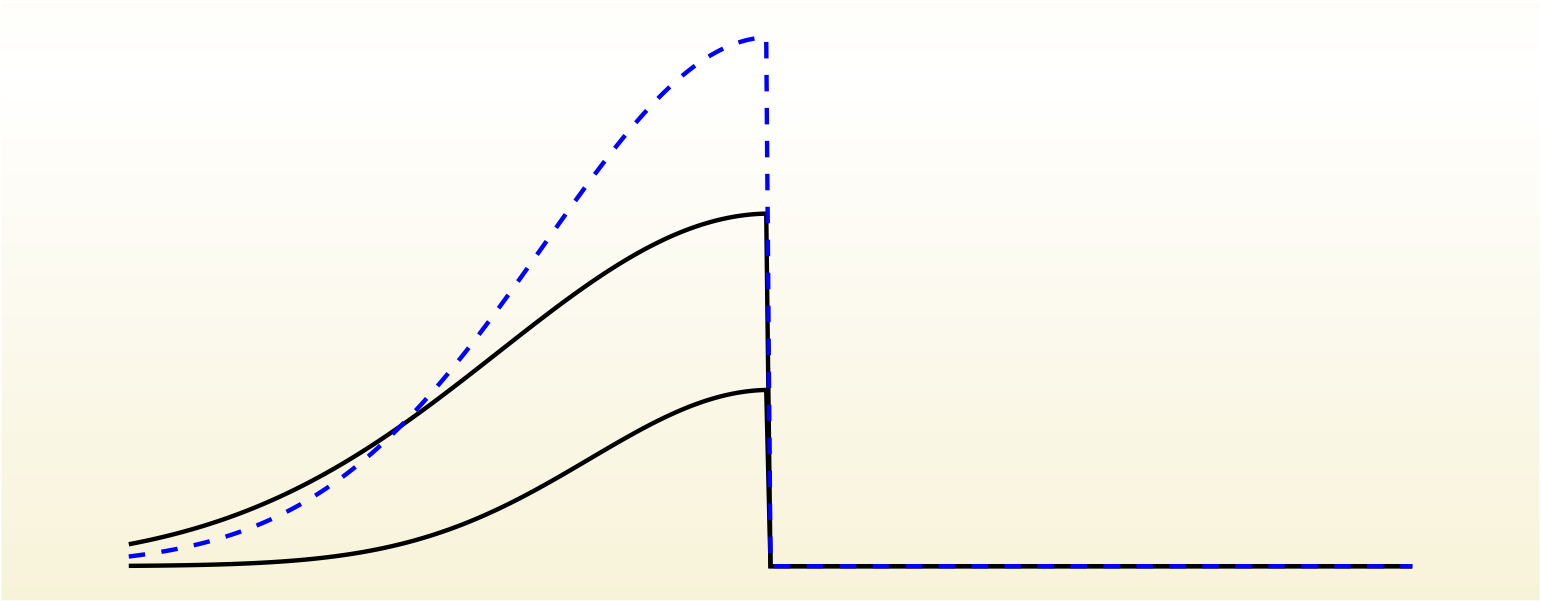
Fundamental issue: linear subspace approximation ill-suited for advection-dominated features (slowly decay Kolmogorov n-width)



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

Reduction of parametrized discontinuities

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

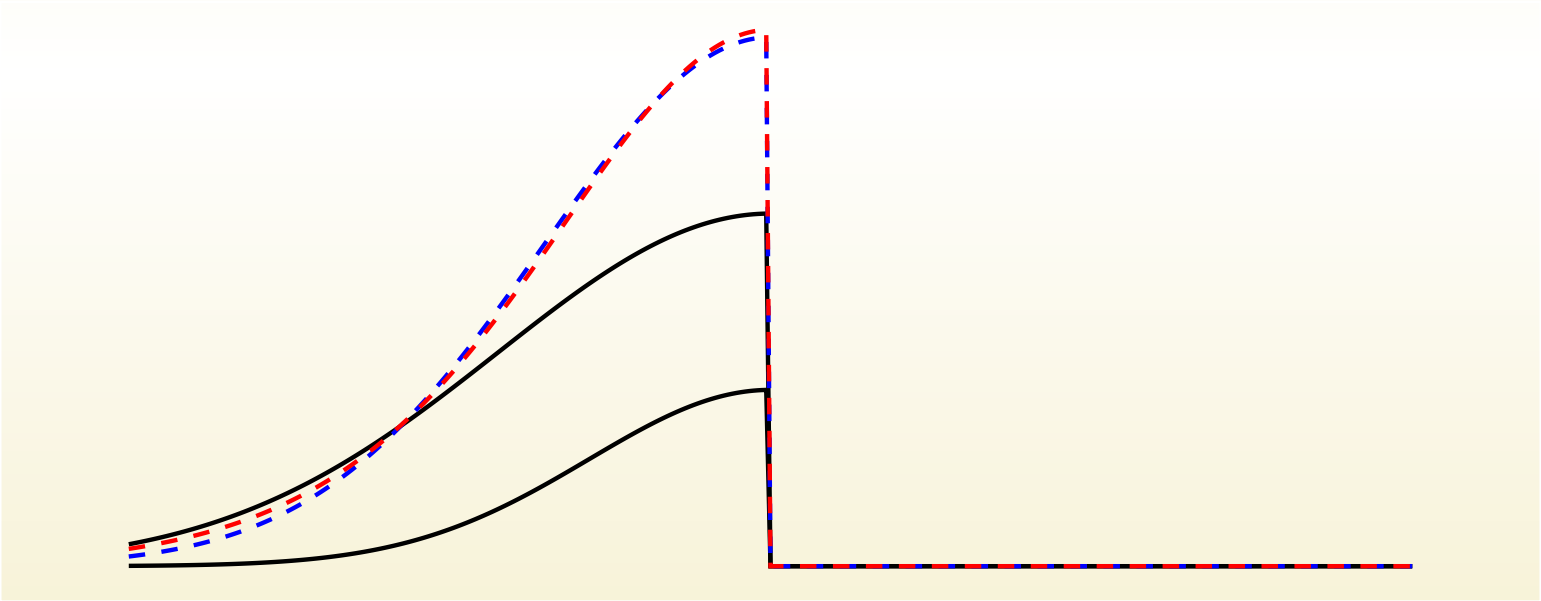


Proposed solution

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

Reduction of parametrized discontinuities

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

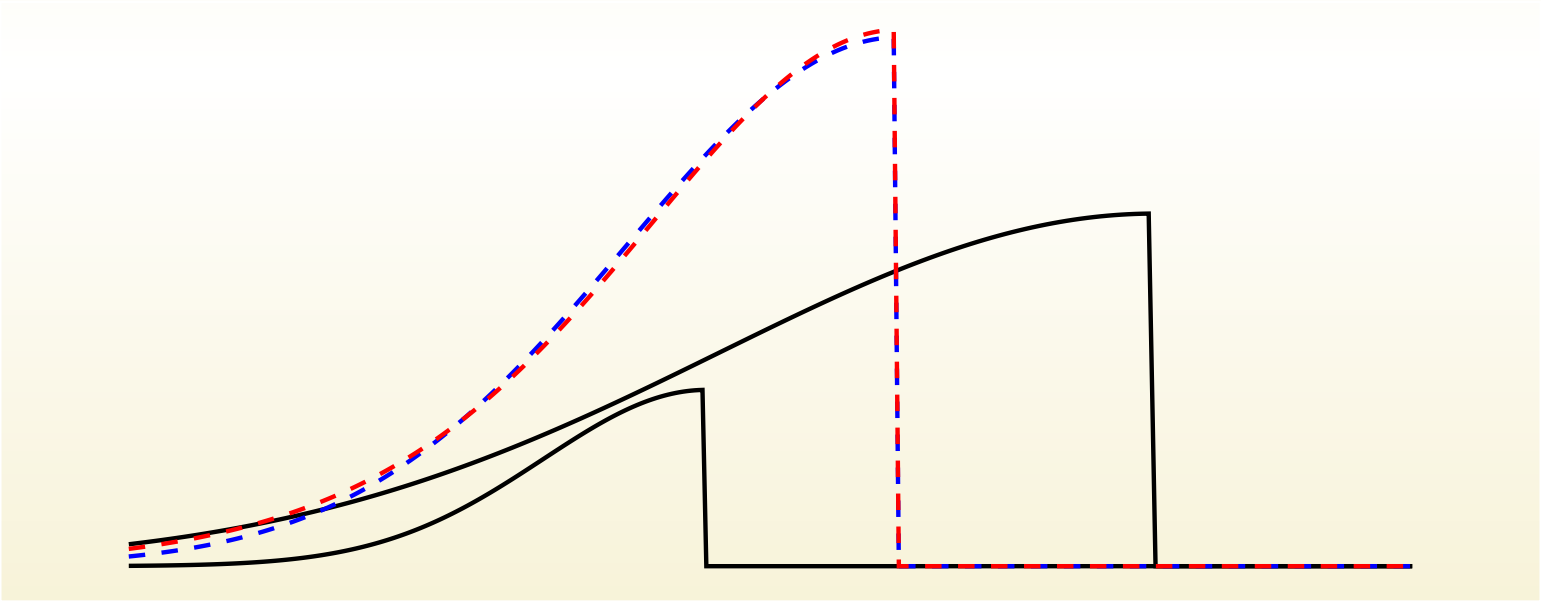


Proposed solution

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

Reduction of parametrized discontinuities

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



Proposed solution

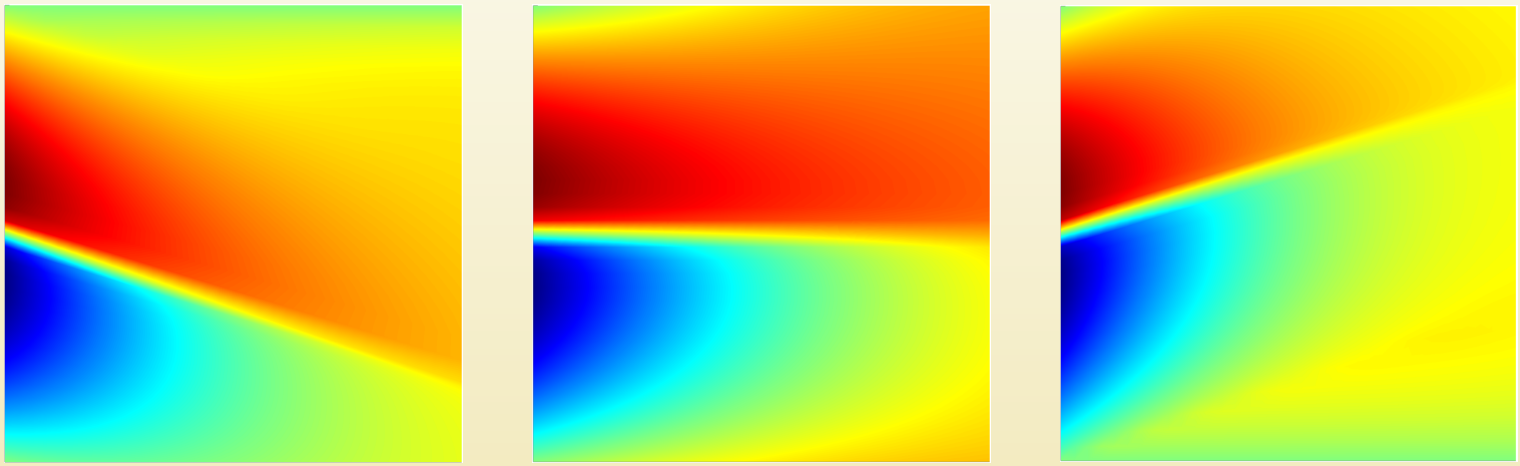
- apply parameter-dependent domain mapping to align features
- use linear subspace in reference domain to reduce dimension
- push forward to physical domain

Numerical experiment: parametrized linear advection

The governing parametrized conservation law is

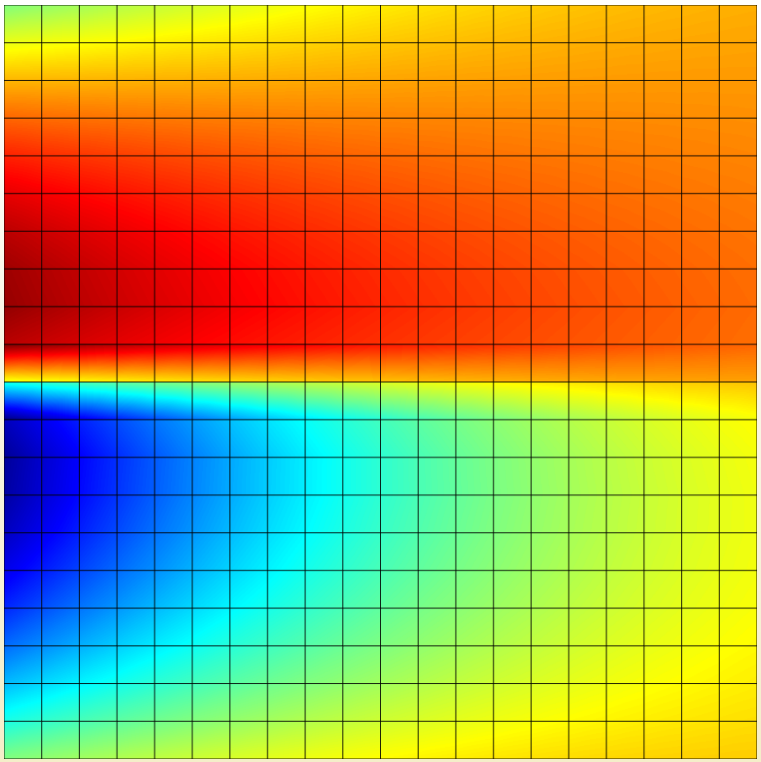
$$\begin{aligned} \nabla \cdot (\boldsymbol{\beta_\mu} U) + \sigma_{\boldsymbol{\mu}} U &= f_{\boldsymbol{\mu}} \quad \text{in } \Omega := (0,1)^2 \\ U &= \hat{U}_{\boldsymbol{\mu}} \quad \text{on } \Gamma_i := \{\boldsymbol{x} \in \partial\Omega \mid \boldsymbol{\beta_\mu} \cdot \boldsymbol{n} < 0\} \end{aligned}$$

μ_1 : shock angle ($\boldsymbol{\beta_\mu}$)
μ_2 : magnitude of source term ($\boldsymbol{\sigma_\mu}$)
μ_3 : magnitude of boundary condition ($\hat{U}_{\boldsymbol{\mu}}$)

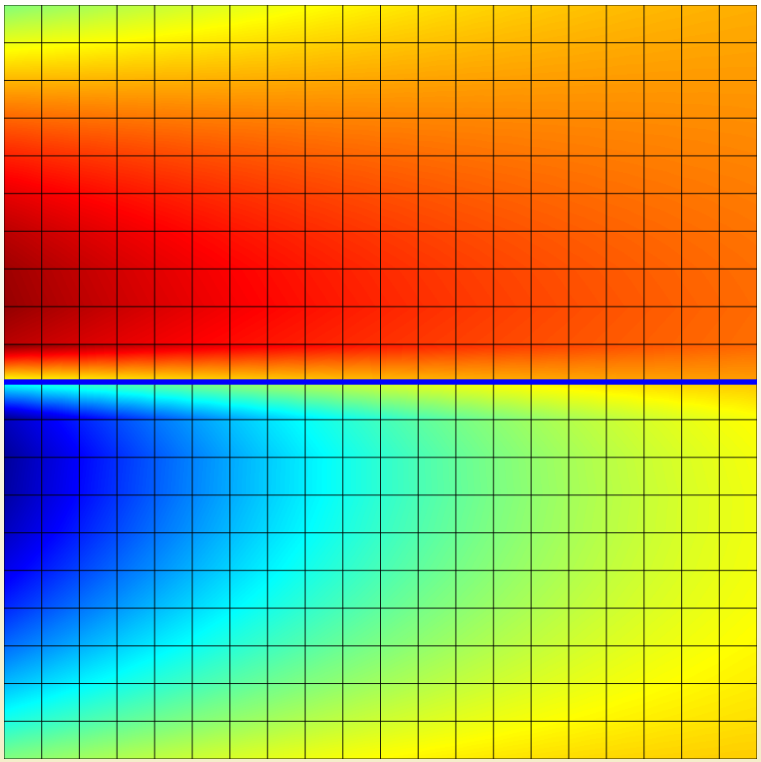


Optimization-based alignment of discontinuities in reference domain (similar to implicit shock tracking)

Snapshot 1



Reference domain

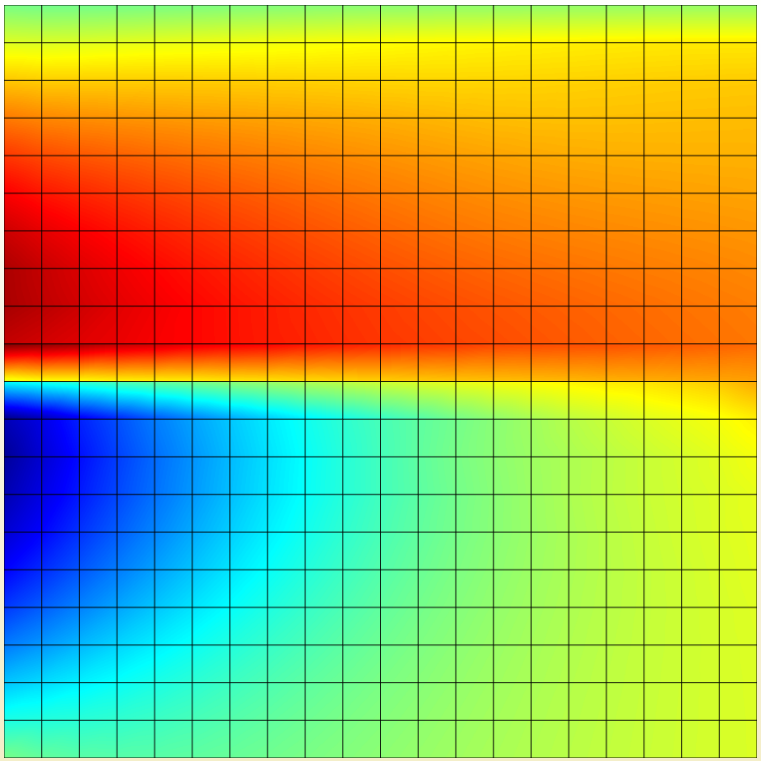


Physical domain

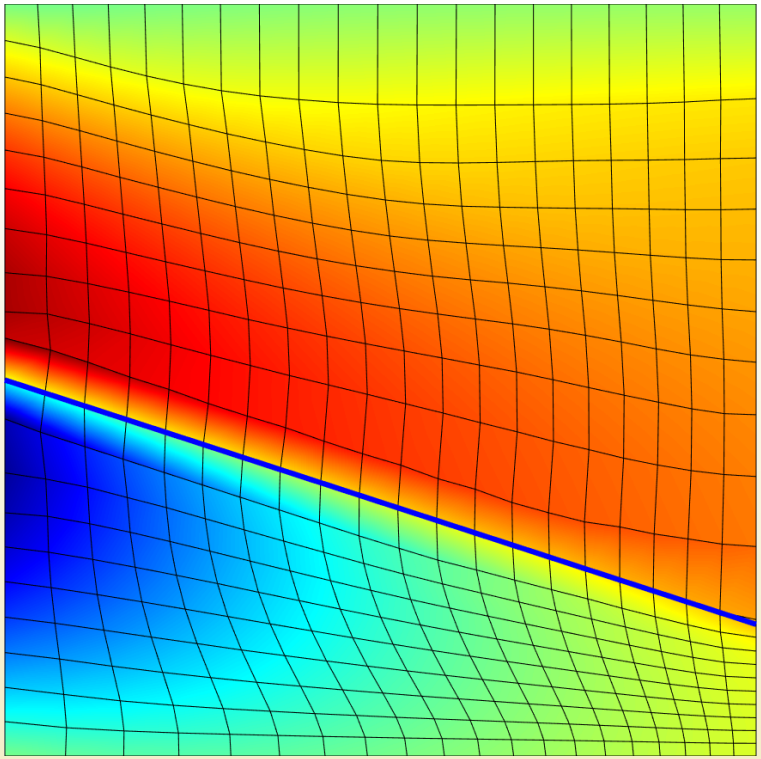
The blue line indicates the true orientation of the discontinuity in the physical domain

Optimization-based alignment of discontinuities in reference domain (similar to implicit shock tracking)

Snapshot 5



Reference domain

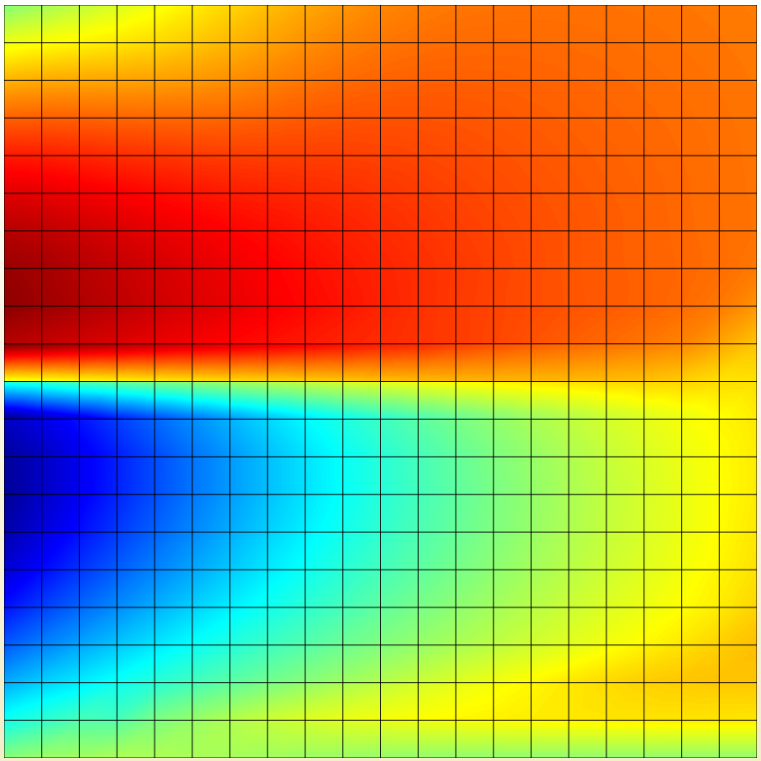


Physical domain

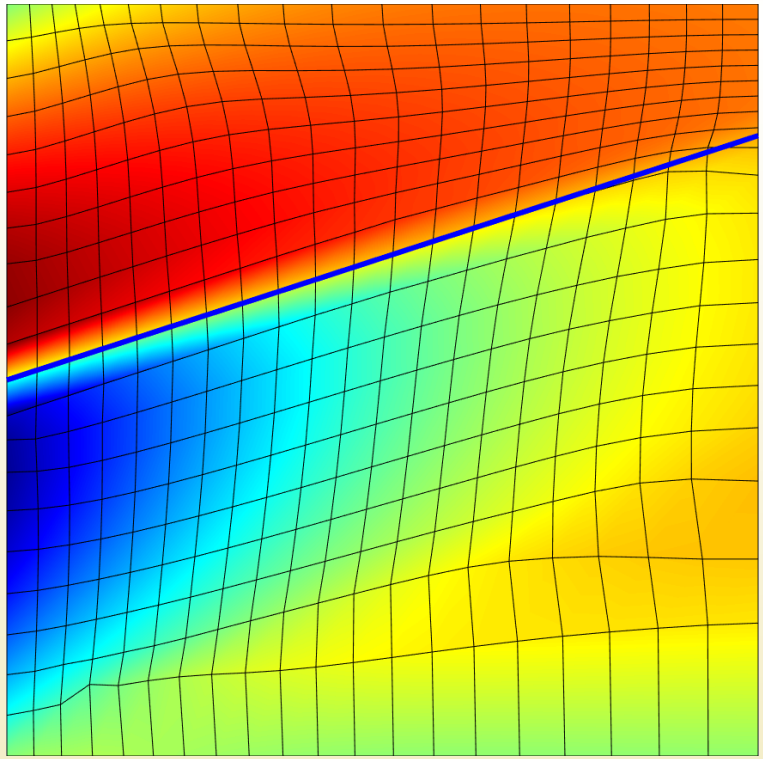
The blue line indicates the true orientation of the discontinuity in the physical domain

Optimization-based alignment of discontinuities in reference domain (similar to implicit shock tracking)

Snapshot 6



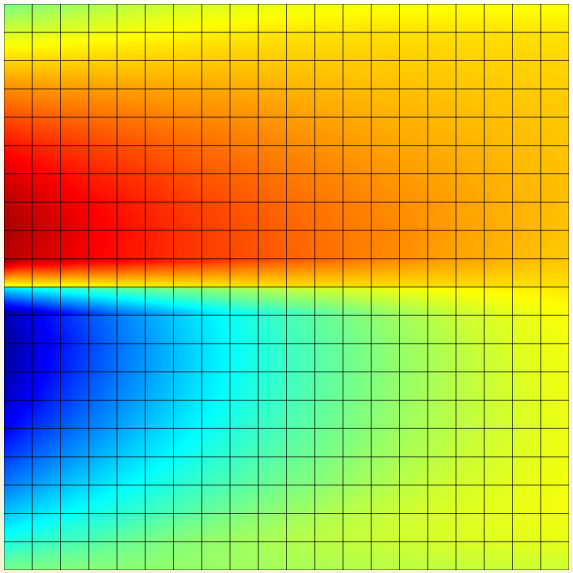
Reference domain



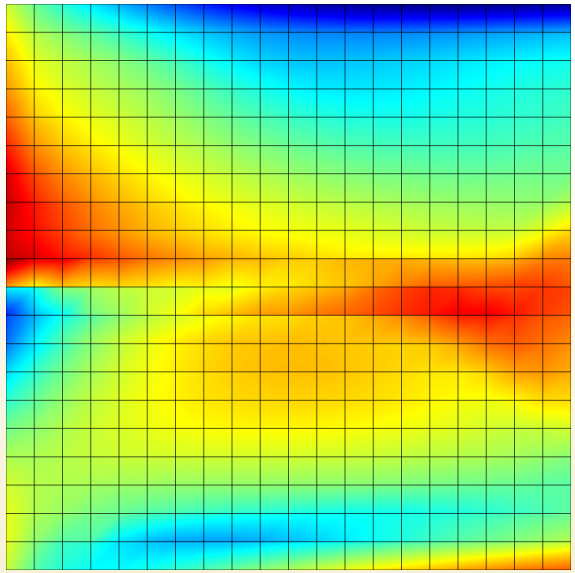
Physical domain

The blue line indicates the true orientation of the discontinuity in the physical domain

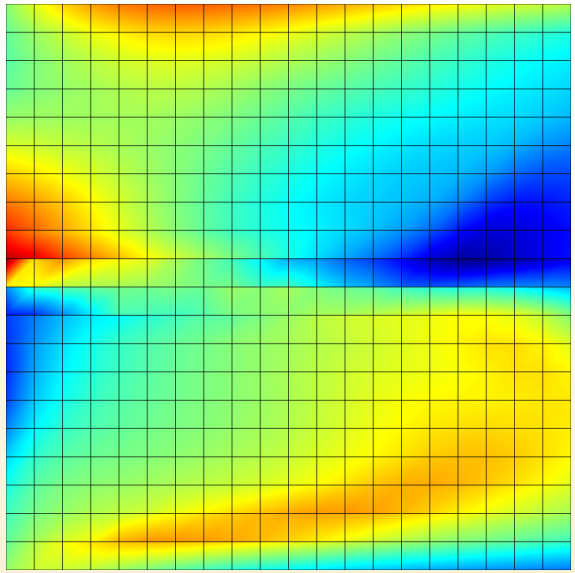
With feature alignment, POD modes do not have to resolve moving discontinuity since it is (mostly) fixed



Mode 1

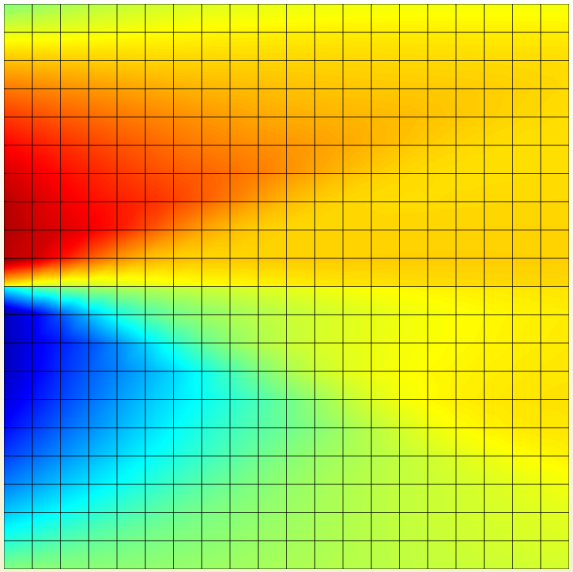


Mode 2

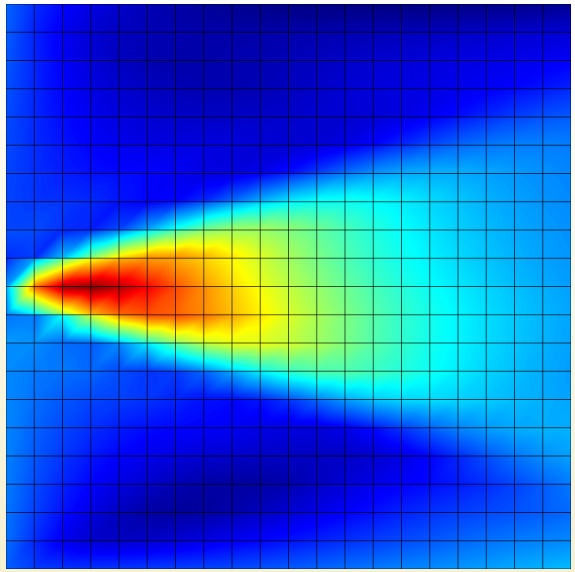


Mode 3

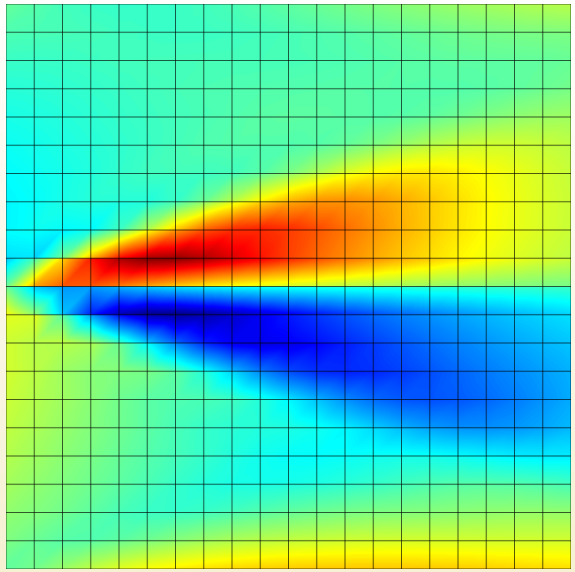
Without feature alignment, POD modes must resolve moving discontinuity, which leads to slowly decaying Kolmogorov n-width



Mode 1



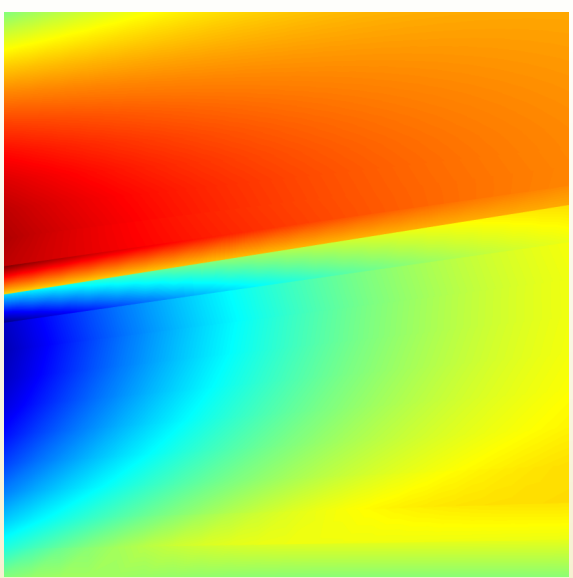
Mode 2



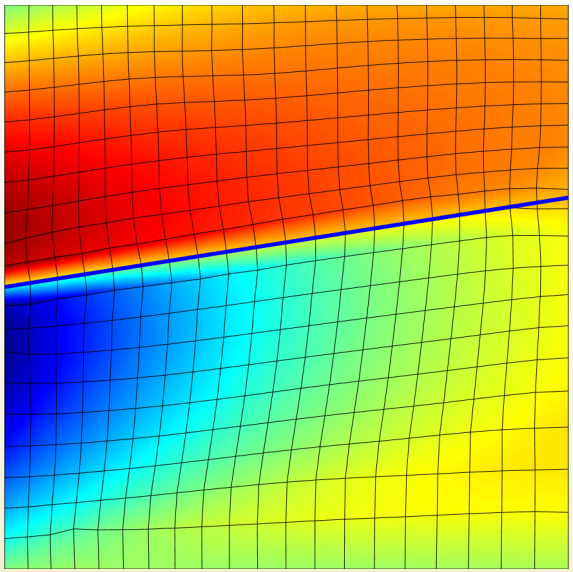
Mode 3

Alignment framework leads to reduction in error for discontinuities **not encountered** during training

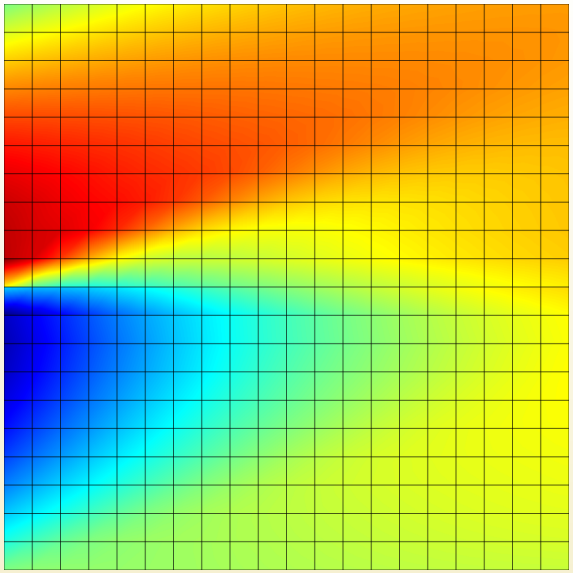
Test point 96 (discontinuity not in training)



HDM



ROM (aligned)



ROM (non-aligned)

The blue line indicates the true orientation of the discontinuity in the physical domain

Alignment framework leads to reduction in error for discontinuities **not encountered** during training

Testing set	Maximum error	Mean error
discontinuity in training	0.92%	0.45%
discontinuity not in training	26.5%	19.5%

Classical minimum-residual model reduction

Testing set	Maximum error	Mean error
discontinuity in training	2.61%	1.22%
discontinuity not in training	4.18%	3.31%

Minimum-residual model reduction with feature alignment

Parameters: 12 training, 125 testing

Appendix

Implicit shock tracking: SQP solver

The solution of the QP leads to the following linear system

$$\begin{bmatrix} \boldsymbol{B_{uu}}(\boldsymbol{z}_k, \hat{\boldsymbol{\lambda}}(\boldsymbol{z}_k)) & \boldsymbol{B_{ux}}(\boldsymbol{z}_k, \hat{\boldsymbol{\lambda}}(\boldsymbol{z}_k)) & \boldsymbol{J_u}(\boldsymbol{z}_k)^T \\ \boldsymbol{B_{ux}}(\boldsymbol{z}_k, \hat{\boldsymbol{\lambda}}(\boldsymbol{z}_k))^T & \boldsymbol{B_{xx}}(\boldsymbol{z}_k, \hat{\boldsymbol{\lambda}}(\boldsymbol{z}_k)) & \boldsymbol{J_x}(\boldsymbol{z}_k)^T \\ \boldsymbol{J_u}(\boldsymbol{z}_k) & \boldsymbol{J_x}(\boldsymbol{z}_k) & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \Delta \boldsymbol{u}_k \\ \Delta \boldsymbol{x}_k \\ \boldsymbol{\eta}_k \end{bmatrix} = - \begin{bmatrix} \boldsymbol{g_u}(\boldsymbol{z}_k) \\ \boldsymbol{g_x}(\boldsymbol{z}_k) \\ \boldsymbol{r}(\boldsymbol{z}_k) \end{bmatrix},$$

where

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

the approximate Hessian of the Lagrangian is partitioned as

$$\boldsymbol{B}_{\square\triangle}(\boldsymbol{z}, \boldsymbol{\lambda}) \approx \frac{\partial^2 \mathcal{L}}{\partial \square \partial \triangle}(\boldsymbol{z}, \boldsymbol{\lambda}),$$

and $\boldsymbol{\eta}_k$ are the Lagrange multipliers of the QP.

SQP solver: Levenberg-Marquardt Hessian approximation

The proposed objective function takes the form of a residual norm

$$f(\boldsymbol{z}) = \frac{1}{2}\left\|\boldsymbol{F}(\boldsymbol{z})\right\|_2^2, \qquad \boldsymbol{F}(\boldsymbol{z}) = \begin{bmatrix} \boldsymbol{R}(\boldsymbol{z}) \\ \kappa \boldsymbol{R}_{\text{msh}}(\boldsymbol{x}) \end{bmatrix}$$

and therefore the Hessian of the Lagrangian has the special structure

$$\boldsymbol{H}(\boldsymbol{z}, \boldsymbol{\lambda}) = \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{z}}(\boldsymbol{z})^T \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{z}}(\boldsymbol{z}) + \boldsymbol{F}_i(\boldsymbol{z}) \frac{\partial^2 \boldsymbol{F}_i}{\partial \boldsymbol{z} \partial \boldsymbol{z}}(\boldsymbol{z}) - \boldsymbol{\lambda}_i \frac{\partial^2 \boldsymbol{r}_i}{\partial \boldsymbol{z} \partial \boldsymbol{z}}(\boldsymbol{z})$$

The last two terms are dropped: they are difficult to compute and negligible if the $\|\boldsymbol{F}\|$ and $\|\boldsymbol{\lambda}\|$ are small (Gauss-Newton approximation)

$$\boldsymbol{H}(\boldsymbol{z}, \boldsymbol{\lambda}) \approx \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{z}}(\boldsymbol{z})^T \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{z}}(\boldsymbol{z})$$

To guard against ill-conditioning, a regularization matrix is added

$$\boldsymbol{B}_{\boldsymbol{uu}} = \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{u}}^T \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{u}}, \quad \boldsymbol{B}_{\boldsymbol{ux}} = \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{u}}^T \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{x}}, \quad \boldsymbol{B}_{\boldsymbol{xx}} = \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{x}}^T \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{x}} + \gamma \boldsymbol{D}.$$

SQP solver: Levenberg-Marquardt Hessian approximation

The proposed objective function takes the form of a residual norm

$$f(\boldsymbol{z}) = \frac{1}{2}\left\|\boldsymbol{F}(\boldsymbol{z})\right\|_2^2, \qquad \boldsymbol{F}(\boldsymbol{z}) = \begin{bmatrix} \boldsymbol{R}(\boldsymbol{z}) \\ \kappa \boldsymbol{R}_{\text{msh}}(\boldsymbol{x}) \end{bmatrix}$$

and therefore the Hessian of the Lagrangian has the special structure

$$\boldsymbol{H}(\boldsymbol{z}, \boldsymbol{\lambda}) = \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{z}}(\boldsymbol{z})^T \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{z}}(\boldsymbol{z}) + \boldsymbol{F}_i(\boldsymbol{z}) \frac{\partial^2 \boldsymbol{F}_i}{\partial \boldsymbol{z} \partial \boldsymbol{z}}(\boldsymbol{z}) - \boldsymbol{\lambda}_i \frac{\partial^2 \boldsymbol{r}_i}{\partial \boldsymbol{z} \partial \boldsymbol{z}}(\boldsymbol{z})$$

The last two terms are dropped: they are difficult to compute and negligible if the $\|\boldsymbol{F}\|$ and $\|\boldsymbol{\lambda}\|$ are small (Gauss-Newton approximation)

$$\boldsymbol{H}(\boldsymbol{z}, \boldsymbol{\lambda}) \approx \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{z}}(\boldsymbol{z})^T \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{z}}(\boldsymbol{z})$$

To guard against ill-conditioning, a regularization matrix is added

$$\boldsymbol{B}_{\boldsymbol{uu}} = \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{u}}^T \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{u}}, \quad \boldsymbol{B}_{\boldsymbol{ux}} = \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{u}}^T \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{x}}, \quad \boldsymbol{B}_{\boldsymbol{xx}} = \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{x}}^T \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{x}} + \gamma \boldsymbol{D}.$$

SQP solver: Levenberg-Marquardt Hessian approximation

The proposed objective function takes the form of a residual norm

$$f(\boldsymbol{z}) = \frac{1}{2}\left\|\boldsymbol{F}(\boldsymbol{z})\right\|_2^2, \qquad \boldsymbol{F}(\boldsymbol{z}) = \begin{bmatrix} \boldsymbol{R}(\boldsymbol{z}) \\ \kappa \boldsymbol{R}_{\text{msh}}(\boldsymbol{x}) \end{bmatrix}$$

and therefore the Hessian of the Lagrangian has the special structure

$$\boldsymbol{H}(\boldsymbol{z}, \boldsymbol{\lambda}) = \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{z}}(\boldsymbol{z})^T \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{z}}(\boldsymbol{z}) + \boldsymbol{F}_i(\boldsymbol{z}) \frac{\partial^2 \boldsymbol{F}_i}{\partial \boldsymbol{z} \partial \boldsymbol{z}}(\boldsymbol{z}) - \boldsymbol{\lambda}_i \frac{\partial^2 \boldsymbol{r}_i}{\partial \boldsymbol{z} \partial \boldsymbol{z}}(\boldsymbol{z})$$

The last two terms are dropped: they are difficult to compute and negligible if the $\|\boldsymbol{F}\|$ and $\|\boldsymbol{\lambda}\|$ are small (Gauss-Newton approximation)

$$\boldsymbol{H}(\boldsymbol{z}, \boldsymbol{\lambda}) \approx \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{z}}(\boldsymbol{z})^T \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{z}}(\boldsymbol{z})$$

To guard against ill-conditioning, a regularization matrix is added

$$\boldsymbol{B}_{\boldsymbol{uu}} = \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{u}}^T \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{u}}, \quad \boldsymbol{B}_{\boldsymbol{ux}} = \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{u}}^T \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{x}}, \quad \boldsymbol{B}_{\boldsymbol{xx}} = \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{x}}^T \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{x}} + \gamma \boldsymbol{D}.$$

Practical considerations: 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 $\mathcal{T}$: *critical* to maintain well-conditioned search directions for meshes where element size varies significantly.

Practical considerations: 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 $\mathcal{T}$: *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

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

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: 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.

Practical considerations: 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 $\hat{\boldsymbol{\lambda}}(\boldsymbol{z})$ ensures

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

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.

Practical considerations: 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 $\hat{\boldsymbol{\lambda}}(\boldsymbol{z})$ ensures

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

and therefore termination is based on the remaining KKT conditions

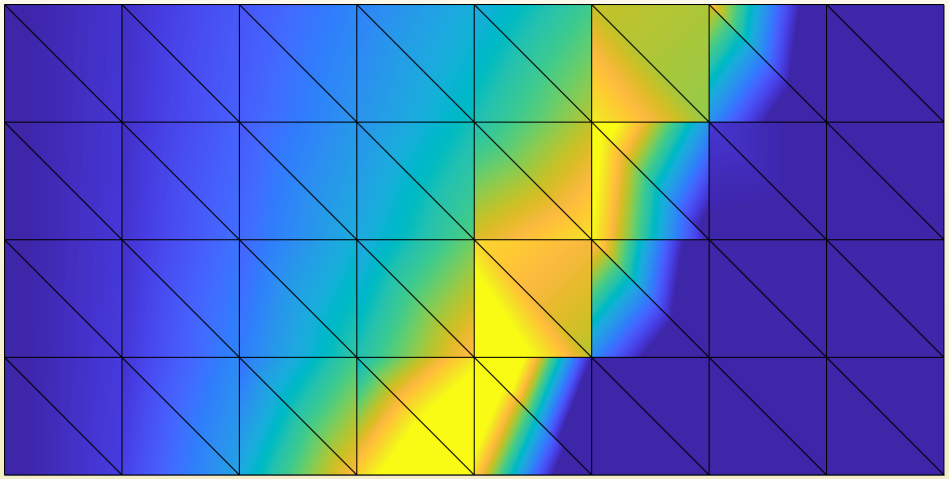
$$\left\|\nabla_{\boldsymbol{x}}\mathcal{L}(\boldsymbol{z},\hat{\boldsymbol{\lambda}}(\boldsymbol{z}))\right\|<\epsilon_1,\qquad \|\boldsymbol{r}(\boldsymbol{z})\|<\epsilon_2,$$

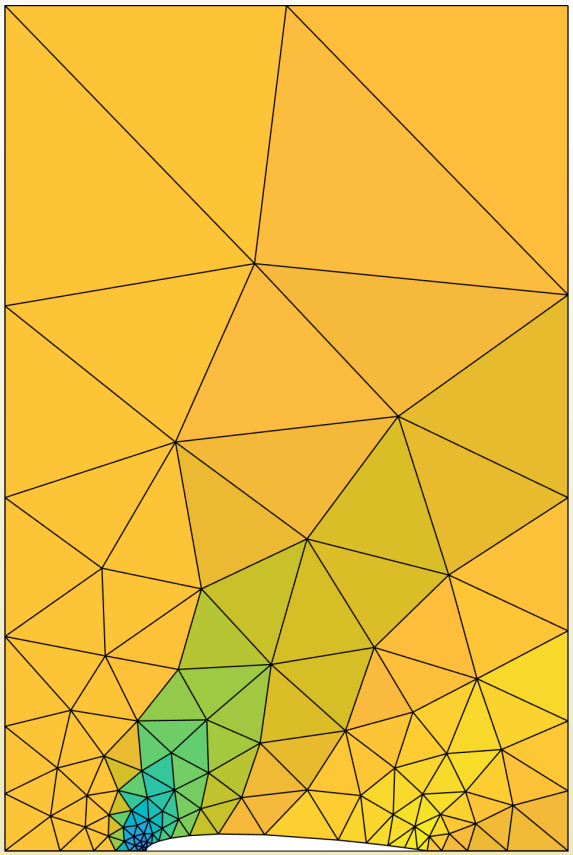
where $\epsilon_1,\epsilon_2>0$ are convergence tolerances.

Practical considerations: initialization

Since using gradient-based optimization, good initialization of $\boldsymbol{u}$ and $\boldsymbol{x}$ is crucial

- For $p = 1$ simulations, $\boldsymbol{x}_0$ comes from mesh generation without knowledge of the shock location and $\boldsymbol{u}_0$ is the DG($p = 0$) solution
- For $p > 1$ simulations, $\boldsymbol{x}_0$ and $\boldsymbol{u}_0$ are the $p = 1$ tracking solution

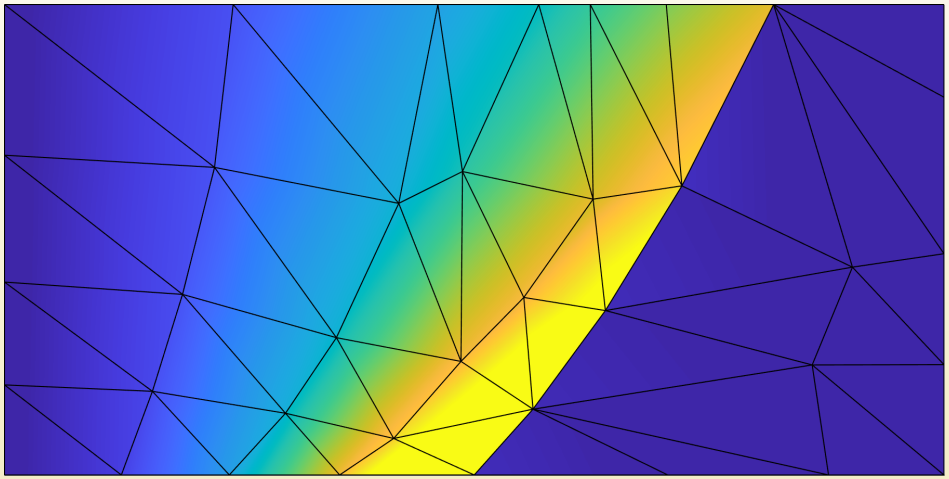


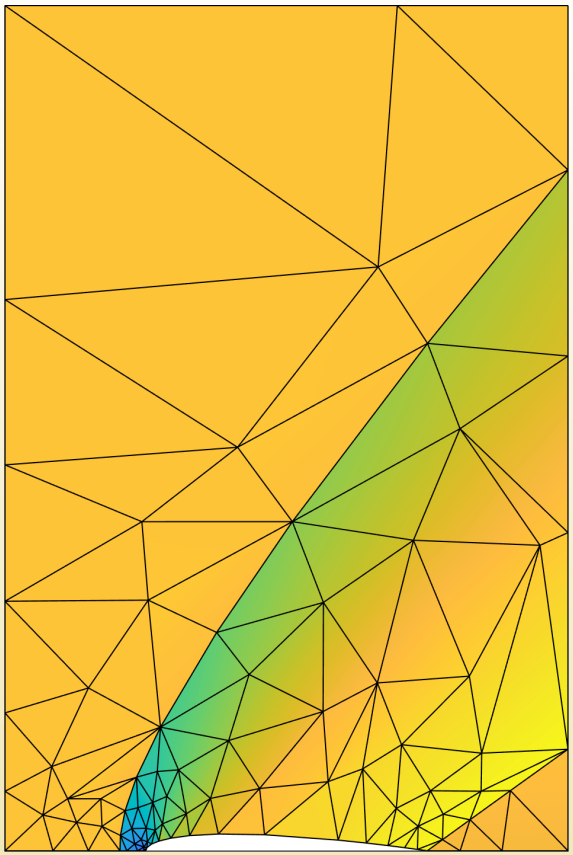


Practical considerations: initialization

Since using gradient-based optimization, good initialization of $\boldsymbol{u}$ and $\boldsymbol{x}$ is crucial

- For $p = 1$ simulations, $\boldsymbol{x}_0$ comes from mesh generation without knowledge of the shock location and $\boldsymbol{u}_0$ is the DG($p = 0$) solution
- For $p > 1$ simulations, $\boldsymbol{x}_0$ and $\boldsymbol{u}_0$ are the $p = 1$ tracking solution

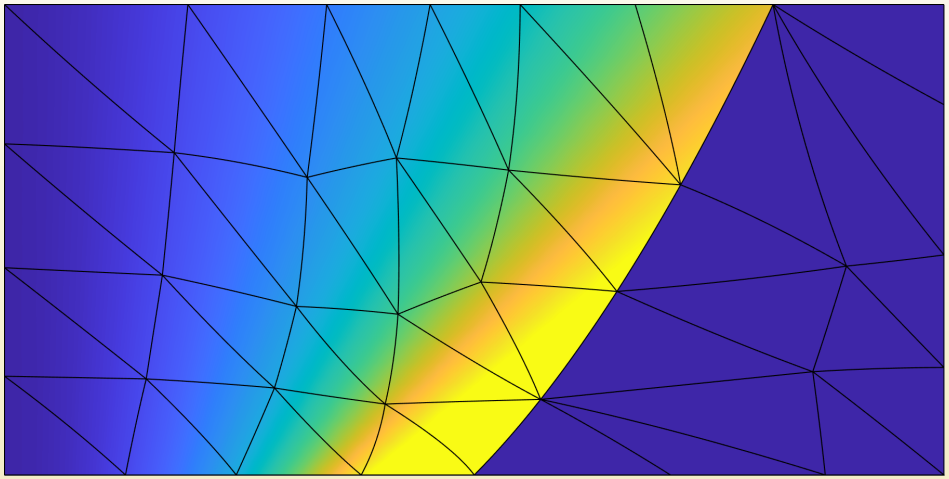


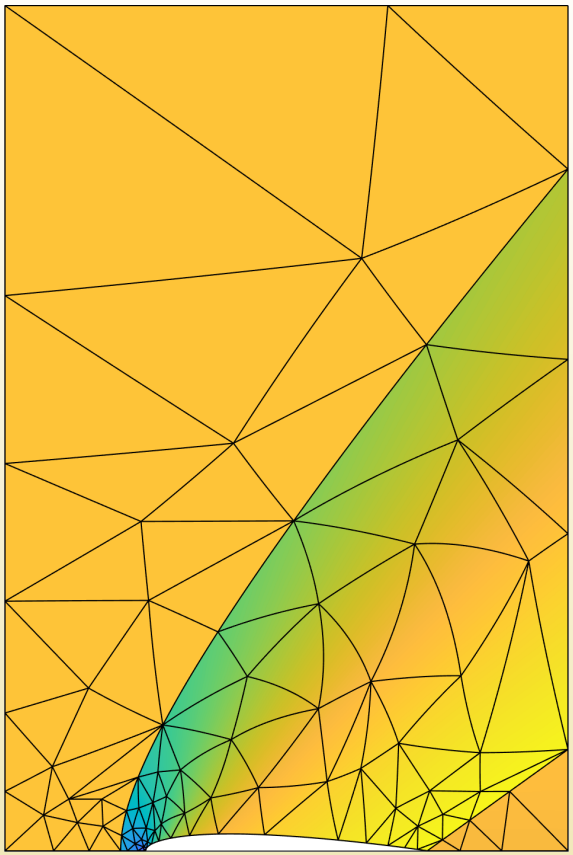


Practical considerations: initialization

Since using gradient-based optimization, good initialization of $\boldsymbol{u}$ and $\boldsymbol{x}$ is crucial

- For $p = 1$ simulations, $\boldsymbol{x}_0$ comes from mesh generation without knowledge of the shock location and $\boldsymbol{u}_0$ is the DG($p = 0$) solution
- For $p > 1$ simulations, $\boldsymbol{x}_0$ and $\boldsymbol{u}_0$ are the $p = 1$ tracking solution





Choice of numerical flux function for implicit tracking

Shock tracking places strict requirements on numerical flux function since inter-element jumps will not tend to zero on shock surface

i. consistent [DG stability]: $\mathcal{H}(U, U, n) = F(U)n$
ii. conservative [DG stability]: $\mathcal{H}(U, U', n) = -\mathcal{H}(U', U, -n)$
iii. preservation of Rankine-Hugoniot [conservation at discontinuities]

$$F(U^+)n = F(U^-)n \implies \mathcal{H}(U^+, U^-, n) = F(U^+)n = F(U^-)n$$

iv. smoothness w.r.t. n [gradient-based optimization]

Choice of numerical flux function for implicit tracking

Shock tracking places strict requirements on numerical flux function since inter-element jumps will not tend to zero on shock surface

i. consistent [DG stability]: $\mathcal{H}(U, U, n) = F(U)n$
ii. conservative [DG stability]: $\mathcal{H}(U, U', n) = -\mathcal{H}(U', U, -n)$
iii. preservation of Rankine-Hugoniot [conservation at discontinuities]

$$F(U^+)n = F(U^-)n \implies \mathcal{H}(U^+, U^-, n) = F(U^+)n = F(U^-)n$$

iv. smoothness w.r.t. n [gradient-based optimization]

Choice of numerical flux function for implicit tracking

Shock tracking places strict requirements on numerical flux function since inter-element jumps will not tend to zero on shock surface

i. consistent [DG stability]: $\mathcal{H}(U, U, n) = F(U)n$
ii. conservative [DG stability]: $\mathcal{H}(U, U', n) = -\mathcal{H}(U', U, -n)$
iii. preservation of Rankine-Hugoniot [conservation at discontinuities]

$$F(U^+)n = F(U^-)n \implies \mathcal{H}(U^+, U^-, n) = F(U^+)n = F(U^-)n$$

iv. smoothness w.r.t. n [gradient-based optimization]

Choice of numerical flux function for implicit tracking

Shock tracking places strict requirements on numerical flux function since inter-element jumps will not tend to zero on shock surface

i. consistent [DG stability]: $\mathcal{H}(U, U, n) = F(U)n$
ii. conservative [DG stability]: $\mathcal{H}(U, U', n) = -\mathcal{H}(U', U, -n)$
iii. preservation of Rankine-Hugoniot [conservation at discontinuities]

$$F(U^+)n = F(U^-)n \implies \mathcal{H}(U^+, U^-, n) = F(U^+)n = F(U^-)n$$

iv. smoothness w.r.t. n [gradient-based optimization]

Choice of numerical flux function for implicit tracking

Shock tracking places strict requirements on numerical flux function since inter-element jumps will not tend to zero on shock surface

i. consistent [DG stability]: $\mathcal{H}(U, U, n) = F(U)n$
ii. conservative [DG stability]: $\mathcal{H}(U, U', n) = -\mathcal{H}(U', U, -n)$
iii. preservation of Rankine-Hugoniot [conservation at discontinuities]

$$F(U^+)n = F(U^-)n \implies \mathcal{H}(U^+, U^-, n) = F(U^+)n = F(U^-)n$$

iv. smoothness w.r.t. n [gradient-based optimization]

Remark: Difficult to satisfy (i)-(iv); select standard numerical flux that satisfies (i)-(iii) and smooth any nonsmooth/discontinuous features

Example: Upwind numerical flux for linear advection

The upwind numerical flux for linear advection ($F_{\text{adv}}(U) = U\beta^T$) is:

$$\mathcal{H}_{\text{up}}(U^+, U^-, n) = (\beta \cdot n)\left[U^+ \cdot H(\beta \cdot n) + U^- \cdot (1 - H(\beta \cdot n))\right],$$

which satisfies (i)-(iii) but is not smooth where $\beta \cdot n = 0$ ((iv) fails).

To recover smoothness, we replace the Heaviside function $H(x)$ with the smoothed heaviside function $H_a(x) := \left(1 + e^{-2ax}\right)^{-1}$: $a = 5$ (——), $a = 10$ (——), $a = 30$ (——), $a = \infty$ (——).

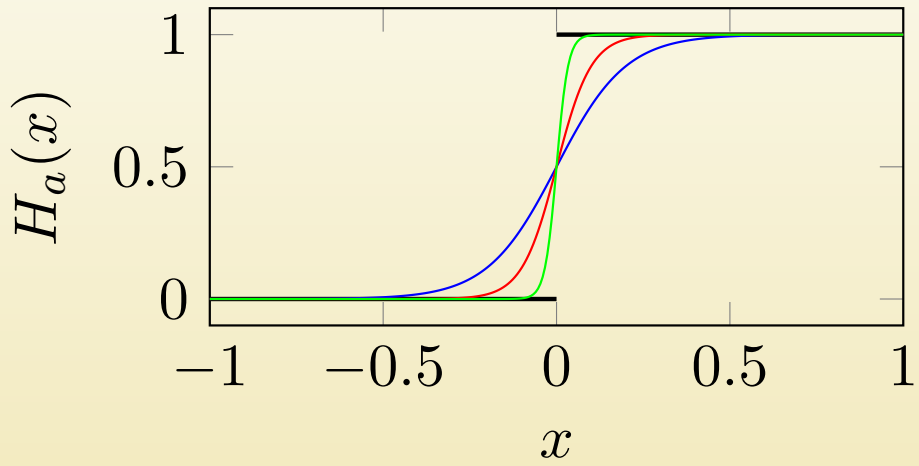
Example: Upwind numerical flux for linear advection

The upwind numerical flux for linear advection ($F_{\text{adv}}(U) = U\beta^T$) is:

$$\mathcal{H}_{\text{up}}(U^+, U^-, n) = (\beta \cdot n)\left[U^+ \cdot H(\beta \cdot n) + U^- \cdot (1 - H(\beta \cdot n))\right],$$

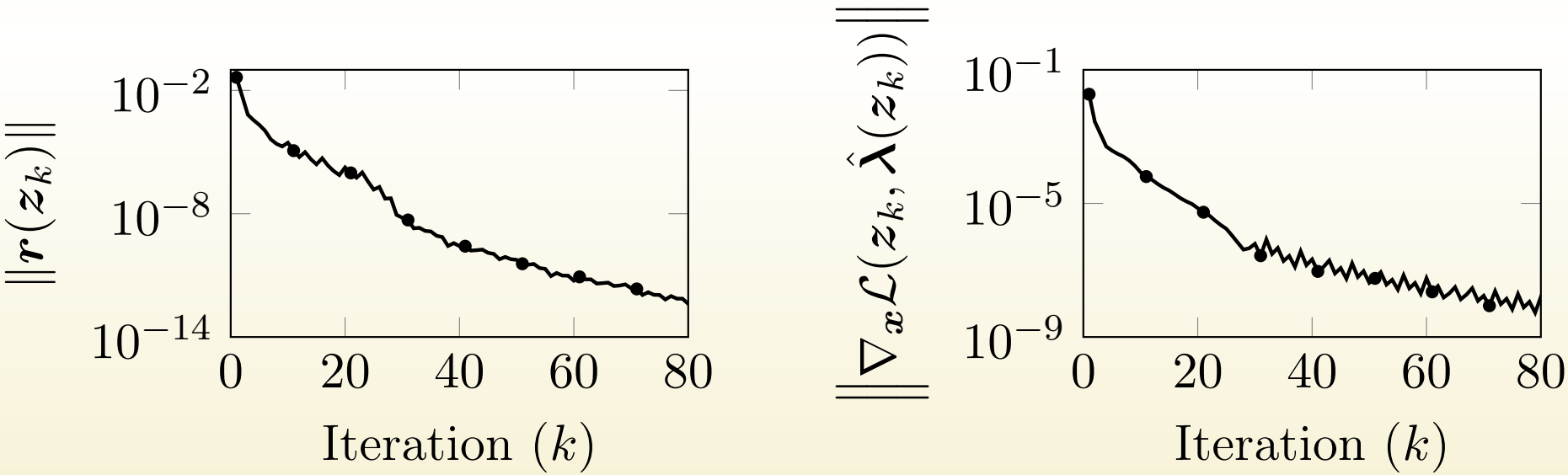
which satisfies (i)-(iii) but is not smooth where $\beta \cdot n = 0$ ((iv) fails).

To recover smoothness, we replace the Heaviside function $H(x)$ with the smoothed heaviside function $H_a(x) := \left(1 + e^{-2ax}\right)^{-1}$: $a = 5$ (——), $a = 10$ (——), $a = 30$ (——), $a = \infty$ (——).



Non-smooth numerical flux can cause SQP solver to fail

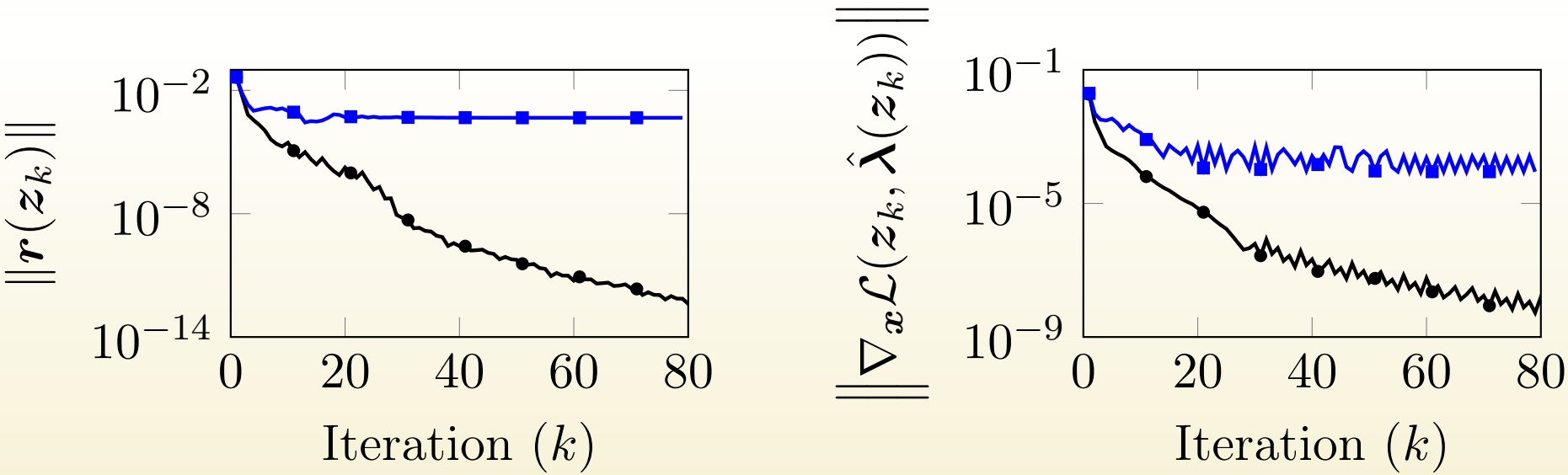
Linear advection with trigonometric shock ($p = 2$)



Convergence of solver with standard (non-smooth) upwind flux (—■—)
and smoothed upwind flux ($a = 10$) (—●—)

Non-smooth numerical flux can cause SQP solver to fail

Linear advection with trigonometric shock ($p = 2$)



Convergence of solver with standard (non-smooth) upwind flux (—■—)
and smoothed upwind flux ($a = 10$) (—●—)