




Continuous Artificial-Viscosity Shock Capturing for Hybrid Discontinuous Galerkin on Adapted Meshes

Yifan Bai* and Krzysztof J. Fidkowski†
University of Michigan, Ann Arbor, Michigan 48188

<https://doi.org/10.2514/1.J061783>

One technique for capturing shocks with high-order methods is through artificial viscosity. The key considerations of this approach are 1) deciding the amount of artificial viscosity to add; 2) maintaining stability and efficiency of the nonlinear solver; and 3) ensuring accuracy of the resulting solutions, particularly in the presence of strong shocks. To address consideration 1, we test a switch based on intraelement solution variation as well as one based on the difference between the solution and its low-order projection. To address consideration 2, we forego a complete linearization of the artificial-viscosity contribution to the residual in order to keep the residual Jacobian stencil compact. To address consideration 3, we introduce the viscosity in a piecewise-continuous fashion to avoid spurious entropy production. Furthermore, we use output-based error estimation and mesh optimization on the drag and the total enthalpy error, as well as with entropy variables, to minimize the output error. We test the shock capturing method coupled with mesh optimization on aerodynamic flow applications ranging from transonic to supersonic, which are discretized using the standard discontinuous Galerkin (DG) and hybridized DG methods. One of our findings is that the mesh optimization through error sampling and synthesis algorithm does not always generate the ideal mesh in the presence of strong shocks.

Nomenclature

C_e	=	elemental cost
c_D	=	drag coefficient
c_p	=	pressure coefficient
E	=	total energy
E_H	=	error in the total enthalpy
H	=	total enthalpy
\mathbf{H}	=	combined viscous/inviscid flux
J	=	output of interest
\mathbf{M}	=	mass matrix
\mathcal{M}	=	metric tensor
P	=	pressure
p	=	state approximation order
p_{epsilon}	=	artificial-viscosity approximation order
\mathbf{Q}	=	discrete state gradient
\mathbf{q}	=	state gradient
\mathbf{R}	=	discrete residual
\mathbf{S}	=	step matrix
S_e	=	smoothness indicator
\tilde{S}_e	=	transformed smoothness indicator
\mathbf{u}	=	conservative state vector
\mathbf{U}	=	discrete state vector
$\hat{\mathbf{u}}_h$	=	state on the faces of the mesh
\mathbf{v}	=	velocity
ϵ_e	=	error indicator
ϵ_{ij}	=	artificial-viscosity tensor field
ϵ_0	=	smooth artificial viscosity
ϵ_{0e}	=	baseline elemental artificial viscosity
$\hat{\epsilon}_e$	=	elemental artificial viscosity
Λ	=	discrete face state
ρ	=	density
Ψ	=	discrete adjoint

I. Introduction

HIGH-ORDER methods have enabled engineering computations at strict error tolerances for a variety of computational fluid dynamics applications requiring high accuracy. With increasing computational power and improvements in algorithms, these methods are becoming feasible for practical computations. The discontinuous Galerkin (DG) method is a popular high-order finite element method that builds on extensive work in Riemann solvers from the finite volume method. It enables rigorous high-order computations on general unstructured meshes, with a relatively simple implementation. However, DG remains expensive in degrees of freedom, and hence storage and CPU time. To tackle this, hybridized discontinuous Galerkin methods reduce the number of globally coupled unknowns in an implicit solution through a static condensation procedure [1–4]. Specifically, element-interior unknowns are locally eliminated in favor of face unknowns through a highly parallelizable step.

Computational cost is only one issue, however. Another important consideration for high-order methods is robustness. Although, at first, a discontinuous approximation space appears well suited for handling discontinuous features, in practice, taking advantage of these discontinuities is difficult for general cases. The discontinuous Galerkin method, like other high-order discretizations, suffers from oscillations near discontinuities and other under-resolved features. Much work exists in stabilizing DG, including Runge-Kutta discontinuous Galerkin (RKDG) [5,6], weighted essentially non-oscillatory (WENO) [7,8], and artificial viscosity [9–12]. This work is similar to more recent studies in smooth artificial viscosity [13,14] but with differences in the means of making the viscosity continuous, an extension to hybridized methods, and coupling with mesh adaptation.

II. Discretization

We consider conservation laws of the form

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot \mathbf{H}(\mathbf{u}, \nabla \mathbf{u}) = \mathbf{0} \quad (1)$$

where $\mathbf{u}(\mathbf{x}, t) \in \mathbb{R}^s$ is the conservative state vector, s is the state rank, and $\mathbf{H}(\mathbf{u}, \nabla \mathbf{u}) = \mathbf{F}(\mathbf{u}) + \mathbf{G}(\mathbf{u}, \nabla \mathbf{u})$ is the total, inviscid, and viscous flux.

We consider two discretizations: discontinuous Galerkin and hybridized discontinuous Galerkin (HDG). In each of these, the computational domain Ω is divided into N_e elements Ω_e in a non-overlapping tessellation T_h . Inside element Ω_e , the state is approximated by polynomials of order p , with no continuity constraints on the element boundary. Formally, we write $\mathbf{u}_h \in \mathcal{V}_h = [\mathcal{V}_h]^s$, where

Presented as Papers 2022-0585 at the 2022 AIAA SciTech Conference, San Diego, CA, January 3–7, 2022; received 25 February 2022; revision received 22 May 2022; accepted for publication 27 May 2022; published online 21 June 2022. Copyright © 2022 by Yifan Bai and Krzysztof J. Fidkowski. Published by the American Institute of Aeronautics and Astronautics, Inc., with permission. All requests for copying and permission to reprint should be submitted to CCC at www.copyright.com; employ the eISSN 1533-385X to initiate your request. See also AIAA Rights and Permissions www.aiaa.org/randp.

*Graduate Student, Department of Aerospace Engineering; yifanb@umich.edu. Student Member AIAA.

†Professor, Department of Aerospace Engineering; kfid@umich.edu. Associate Fellow AIAA.

$$\mathcal{V}_h = \{u \in L_2(\Omega) : u|_{\Omega_e} \in \mathcal{P}^p \quad \forall \Omega_e \in \mathcal{T}_h\}$$

and \mathcal{P}^p denotes polynomials of order p on the reference space of element Ω_e .

A. Discontinuous Galerkin

The DG weak form of Eq. (1) is obtained by multiplying the differential equation by test functions in the same approximation space, integrating by parts, and coupling elements via single-valued fluxes that are functions of the states on the two adjacent elements:

$$\int_{\Omega_e} \mathbf{w}_h^T \frac{\partial \mathbf{u}_h}{\partial t} d\Omega - \int_{\Omega_e} \nabla \mathbf{w}_h^T \cdot \mathbf{H}(\mathbf{u}_h, \nabla \mathbf{u}_h) d\Omega + \int_{\partial\Omega_e} \mathbf{w}_h^T \hat{\mathbf{H}} \cdot \mathbf{n} ds - \int_{\partial\Omega_e} \partial_i \mathbf{w}_h^{+T} \mathbf{K}_{ij}^+ (\mathbf{u}_h^+ - \hat{\mathbf{u}}_h) n_j ds = 0 \quad \forall \mathbf{w}_h \in \mathcal{V}_h \quad (2)$$

where $(\cdot)^T$ denotes transpose; \mathbf{K}_{ij}^+ is the diffusivity tensor; $\hat{\mathbf{u}}_h = (\mathbf{u}_h^+ + \mathbf{u}_h^-)/2$; on the element boundary $\partial\Omega_e$, $(\cdot)^+$ and $(\cdot)^-$ denote quantities taken from the element or its neighbor (or boundary condition), respectively; i, j index the spatial dimension; and summation is implied on repeated indices.

For the normal flux $\hat{\mathbf{H}} \cdot \mathbf{n}$, we use the Roe-approximate Riemann solver [15] unless specified otherwise and the second form of Bassi and Rebay (BR2) [16] for the viscous flux. Choosing a basis for the test and trial spaces yields the semidiscrete form

$$\mathbf{M} \frac{d\mathbf{U}}{dt} + \mathbf{R}(\mathbf{U}) = \mathbf{0}$$

where \mathbf{M} is the mass matrix; and \mathbf{U} and \mathbf{R} are the discrete state vector and the discrete residual, respectively.

For steady cases of $(\partial \mathbf{u} / \partial t) = 0$, we solve the discretized system of nonlinear equations $\mathbf{R}(\mathbf{U}) = \mathbf{0}$ using the Newton–Raphson method. The linearization of the left-hand side is the residual Jacobian matrix $\mathbf{A} \equiv \partial \mathbf{R} / \partial \mathbf{U}$, which is sparse and exhibits a nearest-neighbor block structure.

For unsteady cases, we integrate in time with a three-stage third-order diagonally implicit Runge–Kutta (DIRK) method [17]. The update from \mathbf{U}^n to \mathbf{U}^{n+1} proceeds through a solution of n_{stage} systems:

$$\mathbf{M}(\mathbf{U}_i - \mathbf{U}^n) + \Delta t a_{ii} \mathbf{R}(\mathbf{U}_i) + \Delta t \sum_{j=1}^{i-1} a_{ij} \mathbf{R}(\mathbf{U}_j) = \mathbf{0}, \quad i = 1, \dots, n_{\text{stage}} \quad (3)$$

$$\mathbf{U}^{n+1} = \mathbf{U}^n + \Delta t \sum_{j=1}^{n_{\text{stage}}} b_j \mathbf{R}(\mathbf{U}_j) \quad (4)$$

where

$$a_{ij} = \begin{bmatrix} \alpha & 0 & 0 \\ \frac{1-\alpha}{2} & \alpha & 0 \\ \frac{-(6\alpha^2 - 16\alpha + 1)}{4} & \frac{6\alpha^2 - 20\alpha + 5}{4} & \alpha \end{bmatrix},$$

$$b_i = \begin{bmatrix} \frac{-(6\alpha^2 - 16\alpha + 1)}{4} \\ \frac{6\alpha^2 - 20\alpha + 5}{4} \\ \alpha \end{bmatrix}$$

and $\alpha = 0.435866521508459$. Equation (3) can be solved with Newton–Raphson iterations at each stage in a similar way as in the steady case, with a slight modification to the Jacobian matrix that does not affect its sparsity.

B. Hybridized Discontinuous Galerkin

The starting point for the HDG discretization is the conversion of Eq. (1) to a system of first-order equations:

$$\mathbf{q} - \nabla \mathbf{u} = \mathbf{0} \quad (5)$$

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot \mathbf{H}(\mathbf{u}, \mathbf{q}) = \mathbf{0} \quad (6)$$

where $\mathbf{q}_h \in [\mathcal{V}_h]^{\text{dim}}$ approximates the state gradient. Multiplying these two equations by test functions $\mathbf{v}_h \in [\mathcal{V}_h]^{\text{dim}}$ and $\mathbf{w}_h \in \mathcal{V}_h$ and integrating by parts over an element Ω_e yields the weak forms:

$$\int_{\Omega_e} \mathbf{v}_h^T \cdot \mathbf{q}_h d\Omega + \int_{\Omega_e} \nabla \cdot \mathbf{v}_h^T \mathbf{u}_h d\Omega - \int_{\partial\Omega_e} \mathbf{v}_h^T \cdot \mathbf{n} \hat{\mathbf{u}}_h ds = 0 \quad \forall \mathbf{v}_h \in [\mathcal{V}_h]^{\text{dim}} \quad (7)$$

$$\int_{\Omega_e} \mathbf{w}_h^T \frac{\partial \mathbf{u}_h}{\partial t} d\Omega - \int_{\Omega_e} \nabla \mathbf{w}_h^T \cdot \mathbf{H} d\Omega + \int_{\partial\Omega_e} \mathbf{w}_h^T \hat{\mathbf{H}} \cdot \mathbf{n} ds = 0 \quad \forall \mathbf{w}_h \in \mathcal{V}_h \quad (8)$$

where $\hat{\mathbf{u}}_h$ is a new independent unknown: the state on faces of the mesh. The system is closed by a weak enforcement of flux continuity across faces

$$\int_{\sigma_f} \mu_h^T \{ \hat{\mathbf{H}} \cdot \mathbf{n}|_L + \hat{\mathbf{H}} \cdot \mathbf{n}|_R \} ds = 0 \quad \forall \mu_h \in \mathcal{W}_h \quad (9)$$

where \mathcal{W}_h denotes the order- p approximation space on the faces $\sigma_f \in F_h$ of the mesh: $\mathcal{W}_h = [\mathcal{W}_h]^s$, where

$$\mathcal{W}_h = \{u \in L_2(\sigma_f) : u|_{\sigma_f} \in \mathcal{P}^p \quad \forall \sigma_f \in F_h\}$$

and the subscripts L and R refer to the left and right sides of a face. In HDG, the face approximations are independent and generally discontinuous at nodes and edges in three dimensions. This increases the size of the global system relative to the embedded discontinuous Galerkin method [18], but it yields well-defined blocks in the Jacobian matrix that simplify preconditioning.

The fluxes in Eq. (8) are one-sided, meaning that they depend only on the state and gradient inside the element and the face state:

$$\hat{\mathbf{H}} \cdot \mathbf{n} = \mathbf{H}(\hat{\mathbf{u}}_h, \mathbf{q}_h) \cdot \mathbf{n} + \boldsymbol{\tau}(\hat{\mathbf{u}}_h, \mathbf{u}_h, \mathbf{n}),$$

$$\boldsymbol{\tau} = \left| \frac{\partial}{\partial \mathbf{u}} (\hat{\mathbf{F}} \cdot \mathbf{n}) \right|_{\mathbf{u}_h^*} (\mathbf{u}_h - \hat{\mathbf{u}}_h) + \eta \boldsymbol{\delta}_h \cdot \mathbf{n} \quad (10)$$

Note that $\boldsymbol{\tau}$ consists of a convective stabilization computed about the Roe-average state \mathbf{u}_h^* and a BR2 viscous stabilization [19], where η is set to a value that is at least the number of faces and $\boldsymbol{\delta}_h$ is the BR2 auxiliary variable driven by the state jump $\mathbf{u}_h - \hat{\mathbf{u}}_h$.

Choosing bases for the trial/test spaces in Eqs. (7–9) gives a nonlinear system of ordinary differential equations:

$$\mathbf{R}^Q = \mathbf{0},$$

$$\mathbf{M}^U \frac{d\mathbf{U}}{dt} + \mathbf{R}^U = \mathbf{0},$$

$$\mathbf{R}^\Lambda = \mathbf{0} \quad (11)$$

where \mathbf{M}^U is the mass matrix. For a steady case, the ordinary differential equations reduce to a nonlinear system of equations:

$$\mathbf{R}^Q = \mathbf{0}, \quad \mathbf{R}^U = \mathbf{0}, \quad \mathbf{R}^\Lambda = \mathbf{0} \quad (12)$$

with the Newton update system

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} \begin{bmatrix} \Delta \mathbf{Q} \\ \Delta \mathbf{U} \\ \Delta \Lambda \end{bmatrix} + \begin{bmatrix} \mathbf{R}^Q \\ \mathbf{R}^U \\ \mathbf{R}^\Lambda \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \quad (13)$$

where \mathbf{Q} , \mathbf{U} , and Λ are the discrete unknowns in the approximation of \mathbf{q} , \mathbf{u} , and $\hat{\mathbf{u}}$, respectively. $[\mathbf{A}, \mathbf{B}; \mathbf{C}, \mathbf{D}]$ is the primal Jacobian matrix partitioned into element-interior and interface unknown blocks. Note that \mathbf{A} , \mathbf{B} , and \mathbf{C} contain both \mathbf{Q} and \mathbf{U} components. In addition, \mathbf{A} is elementwise block diagonal, and hence easily invertible using element-local operations.

Statically condensing out the element-interior states gives a smaller system for the face degrees of freedom:

$$\underbrace{(\mathbf{D} - \mathbf{C}\mathbf{A}^{-1}\mathbf{B})}_{\kappa} \Delta \Lambda + (\mathbf{R}^\Lambda - \mathbf{C}\mathbf{A}^{-1}[\mathbf{R}^Q; \mathbf{R}^U]) = \mathbf{0} \quad (14)$$

Solving this set of equations constitutes the global solution of the problem. Following the global solution for $\Delta \Lambda$, an element-local backsubstitution yields the updates to \mathbf{Q} and \mathbf{U} .

If the unsteady term is present, we integrate in time with the same DIRK scheme used for the DG discretization:

$$\begin{aligned} \mathbf{R}^Q(\mathbf{Q}_i, \mathbf{U}_i, \Lambda_i) &= \mathbf{0}, \\ \mathbf{M}^U(\mathbf{U}_i - \mathbf{U}^n) + \Delta t a_{ii} \mathbf{R}^U(\mathbf{Q}_i, \mathbf{U}_i, \Lambda_i) + \Delta t \sum_{j=1}^{i-1} a_{ij} \mathbf{R}^U(\mathbf{Q}_j, \mathbf{U}_j, \Lambda_j) &= \mathbf{0}, \\ \mathbf{R}^\Lambda(\mathbf{Q}_i, \mathbf{U}_i, \Lambda_i) &= \mathbf{0}, \\ i &= 1, \dots, n_{\text{stage}} \end{aligned} \quad (15)$$

$$\mathbf{U}^{n+1} = \mathbf{U}^n + \Delta t \sum_{j=1}^{n_{\text{stage}}} b_j \mathbf{R}^U(\mathbf{Q}_j, \mathbf{U}_j, \Lambda_j) \quad (16)$$

For the Newton iterations used to solve Eq. (15) at each stage, the Jacobian matrix needs to be modified slightly from the steady case, but this does not affect the sparsity of \mathcal{K} .

III. Shock Capturing

A. Euler Equations

In this work, we restrict our attention to the Euler equations because physical viscosity is insufficient to stabilize shocks at high Reynolds numbers on typical computational meshes: even adapted ones. Given the state vector $\mathbf{u} = [\rho, \rho \mathbf{v}, \rho E]$ for the Euler equations, where ρ is the density, \mathbf{v} is the velocity, and E is the total energy, the pressure can be found as

$$P = \frac{1}{\gamma - 1} \left(\rho E - \frac{1}{2} \rho |\mathbf{v}|^2 \right)$$

where γ is the specific gas constant. The inviscid flux vector is

$$\mathbf{F}(\mathbf{u}) = \begin{bmatrix} \rho \mathbf{v} \\ \rho \mathbf{v} \otimes \mathbf{v} + P \mathbf{I} \\ \rho \mathbf{v} H \end{bmatrix} \quad (17)$$

where $H = E + (P/\rho)$ is the total enthalpy, and \mathbf{I} is the identity matrix of size $\text{dim} \times \text{dim}$.

B. Artificial Viscosity

This section outlines the shock capturing approach using artificial viscosity. The starting point is the general form of an unsteady convection–diffusion partial differential equation, which is written in index notation:

$$\partial_t u_k + \partial_i F_{ik} = \partial_i (K_{ijkl} \partial_j u_l) \quad (18)$$

where k and l index the state rank, i and j index the spatial dimension, F_{ik} is the convective flux, and K_{ijkl} is the diffusivity tensor. Both F and K generally depend on the state and could depend on the position. For shock capturing, we augment the physical diffusivity with an extra tensor field:

$$K_{ijkl}^{\text{stab}}(\mathbf{x}) = T_{kl} \epsilon_{ij}(\mathbf{x}) \quad (19)$$

where $T_{kl} = (\partial \tilde{u}_k / \partial u_l)$, $\tilde{\mathbf{u}}_k = [\rho, \rho \mathbf{u}, \rho H]$ is a modified state vector that makes the stabilization term preserve total enthalpy [12], and ϵ_{ij} is an artificial-viscosity tensor field. Numerical dissipation is added through both the convective and diffusive fluxes. The Roe flux function [15] does not preserve total enthalpy, and hence we also present test results for the van Leer–Hänel flux function [20], which is designed to preserve the total enthalpy.

The artificial-viscosity tensor field is found as

$$\epsilon_{ij}(\mathbf{x}) = C \frac{h_{ij}}{h} \epsilon_0(\mathbf{x}) \quad (20)$$

where C is an $\mathcal{O}(1)$ constant for adjusting the amount of stabilization, h_{ij}/h is a smoothly varying anisotropy field for which the calculation will be described in the following, and $\epsilon_0(\mathbf{x})$ is a smooth scalar that comes from averaging an element-based artificial viscosity to nodes.

The mesh-implied metric of a simplex element e (\mathcal{M}_e) defines a Euclidean vector space in which all edges of the element are unit length [21,22]; i.e., $\mathcal{M}_e \in \text{Sym}_d^+$ such that

$$\sqrt{v_i \mathcal{M}_{e,ij} v_j} = 1, \quad \forall v_i \in \text{Edges}(e) \quad (21)$$

The metric has units of inverse distance squared, and it yields a measure of the size of the element:

$$\bar{h}_e = [\det(\mathcal{M}_e)]^{-1/2 \text{dim}} \quad (22)$$

As the metric eigenvalues have units of inverse square distance, the smoothly varying anisotropy field of h_{ij}/h is obtained by dividing the inverse square root of the metric by \bar{h}_e , i.e., taking $(1/\bar{h}_e) \mathcal{M}_e^{-1/2}$, and averaging this quantity from elements to nodes. When needed at an arbitrary point in an element, the anisotropy tensor is interpolated from the linear nodes that make up the element.

The construction of the element-based artificial viscosity $\hat{\epsilon}_e$ starts from a baseline elemental artificial viscosity defined as

$$\epsilon_{0e} = \frac{\lambda_{\max} \bar{h}_e}{p} \bar{S}_e \quad (23)$$

where \bar{S}_e is a smoothness indicator computed from the states, which is either the resolution or the variation indicator in this paper; λ_{\max} is the maximum wave speed in element e ; and p is the approximation order.

The calculation of artificial viscosity is followed by a Laplace smoothing of ϵ_e , which simulates a diffusion effect on the elemental smoothness indicator. The formulation is similar to a Jacobi smoother:

$$\tilde{\epsilon}_e^{k+1} = (1 - c_s) \epsilon_{0e} + \frac{c_s}{n_e} \sum_{t \in \mathcal{N}(e)} \tilde{\epsilon}_t^k, \quad k = 0, \dots, n_{\text{smooth}} - 1 \quad (24)$$

$$\epsilon_e = \tilde{\epsilon}_e^{n_{\text{smooth}}} \quad (25)$$

where $\mathcal{N}(e)$ denotes the neighboring elements of element e (those with which it shares an edge), n_e is the number of the neighboring elements, and $0 < c_s \leq 1$ is a user-defined coefficient that adjusts the amount of diffusion. A larger c_s introduces more diffusion. This is only used in our two-dimensional experiments. Note that c_s is chosen to be 1.0 for

the transonic cases and 0.9 for the hypersonic cases. Also, $n_{\text{smooth}} = 10$ was found to be sufficient for all cases.

The final step in the calculation of $\hat{\epsilon}_e$ is a filter used for the hypersonic test cases presented in Sec. VI to clip away spurious small values generated by the Laplace smoothing. We adopt the smooth filter definition from Barter and Darmofal [12], except that we apply it before making the artificial viscosity continuous:

$$\hat{\epsilon}_e(\epsilon_e) = \begin{cases} 0, & \epsilon_e \leq \theta_L \\ \frac{1}{2}\theta_H \left(\sin \left[\pi \left(\frac{\epsilon_e - \theta_L}{\theta_H - \theta_L} - \frac{1}{2} \right) \right] + 1 \right), & \theta_L < \epsilon_e < \theta_H \\ \theta_H, & \epsilon_e \geq \theta_H \end{cases} \quad (26)$$

where θ_L and θ_H are the maximum and minimum values that $\hat{\epsilon}_e$ varies in between, and $\theta_H = \lambda_{\max} \tilde{h}_e / p$ and $\theta_L = 0.01\theta_H$ are used in this paper.

For unsteady cases, the artificial viscosity is calculated at each Newton iteration, and the updated viscosity values are used in the calculation of the residuals and the linearizations of the state variables. The artificial viscosity is treated as constant when calculating the linearizations. This freezing of the viscosity results precludes Newton convergence but, in practice, does not significantly increase the number of iterations required to obtain a solution because many iterations are expended before the Newton convergence ‘‘bucket.’’ The lack of an exact linearization, however, preserves the compact stencil, which is crucial for efficiency of the solver.

C. Smoothness Indicators

Both the resolution indicator and the variation indicator measure the smoothness of a selected quantity, which is chosen to be the density for our experiments so that more artificial viscosity is added where discontinuities appear in the solution. The one-dimensional results presented are generated with the resolution indicator; whereas for the two-dimensional cases, both indicators are used, and the type of indicator is specified in each case.

1. Resolution Indicator

The resolution indicator takes advantage of the fact that for a smooth solution, the coefficients of the Fourier series decay rapidly. It is defined as the difference between a p th-order quantity and its least-squares projection onto the space of $(p-1)$ th-order polynomials:

$$\bar{S}_e = \frac{f^2}{f+1}, \quad f = \frac{S_e}{S_0}, \quad S_e = \frac{\int_{\Omega_e} (u - \tilde{u})^2 d\Omega}{\int_{\Omega_e} \tilde{u}^2 d\Omega} \quad (27)$$

where u is the chosen scalar for measuring regularity, \tilde{u} is its $(p-1)$ th projection, $S_0 = 10^{-c_0 - c_p p}$ is an order-dependent variation scale, and c_0 and c_p are constants that adjust the amount of stabilization. The bigger both of the constants are, the more stabilization is added. When the resolution indicator is used, the amount of stabilization is controlled by both c_0 and c_p in a nonlinear mapping as well as the coefficient C in Eq. (20) in a linear way. We did not attempt to fine-tune all three at the same time for this paper. $C = 1$ is used, and c_0 and c_p are tuned for all one-dimensional examples because we tested for different polynomial orders for each case. For the two-dimensional examples, $c_0 = 0$ and $c_p = 1$ are used, and C is tuned whenever more stabilization is needed.

2. Variation Indicator

The variation indicator is based on the intraelement variation of a selected quantity:

$$\bar{S}_e = \begin{cases} 0, & S_e < S^* - \Delta S \\ \frac{S_e}{2} \left[1 + \sin \left(\frac{\pi}{2\Delta S} (S_e - S^*) \right) \right], & S^* - \Delta S \leq S_e \leq S^* + \Delta S \\ S_e, & S_e > S^* + \Delta S \end{cases} \quad (28)$$

$$S_e = \left[\frac{1}{|\Omega_e|} \int_{\Omega_e} \left(\frac{u}{\tilde{u}_e} - 1 \right)^2 d\Omega \right]^{1/2} \quad (29)$$

where u is the chosen scalar for measuring regularity, and

$$\tilde{u}_e = \frac{1}{|\Omega_e|} \int_{\Omega_e} u d\Omega$$

S^* and ΔS are user-defined parameters. The smooth scaling was presented by Persson and Peraire [10] and used in combination with the variation indicator by Ching et al. [14]. It preserves large values and clips down small ones. S^* is chosen to be 0.75 for our transonic cases and 1.25 for the hypersonic cases. ΔS is set to 0.25 throughout.

IV. Mesh Adaptation

A. Output Error Estimation

The mesh adaptation process used in this work is driven by the estimation of the output error. Let H denote the current, ‘‘coarse,’’ approximation space; and let h denote a fine space obtained by increasing the polynomial order by one on each element. An estimate of the error between the coarse and fine spaces for our output of interest J can be found using the adjoint-weighted residual [23,24]. We use discrete adjoint solutions for this purpose. For DG, the output error estimate reads

$$J_h(\mathbf{U}_h^H) - J_h(\mathbf{U}_h) \approx -\delta\Psi_h^T \mathbf{R}_h(\mathbf{U}_h^H) \quad (30)$$

where \mathbf{U}_h^H is the coarse state prolonged into the fine space, and $\delta\Psi_h$ is obtained by subtracting the coarse-space adjoint from the fine-space adjoint. For HDG, the output error estimate includes terms associated with the gradient and weak flux continuity equations:

$$J_h(\mathbf{U}_h^H) - J_h(\mathbf{U}_h) \approx -(\delta\Psi_h^Q)^T \mathbf{R}_h^Q - (\delta\Psi_h^U)^T \mathbf{R}_h^U - (\delta\Psi_h^\Lambda)^T \mathbf{R}_h^\Lambda \quad (31)$$

The localized error contributions on each element can be used as error indicators to drive the mesh adaptation process. For DG, the error indicator on element e is

$$\epsilon_e \equiv |\delta\Psi_{h,e}^T \mathbf{R}_{h,e}(\mathbf{U}_h^H)| \quad (32)$$

The HDG error indicators ϵ_e^Q and ϵ_e^U can be found in a similar way, but finding ϵ_e^Λ requires special considerations. A detailed discussion of the solution of the adjoint equations and the treatment of the error localization for HDG was presented in previous work by Fidkowski and Chen [18].

We also tested the entropy variables in place of the output adjoint for the Euler equations [25]. They are defined by $\mathbf{V} \equiv \partial U / \partial \mathbf{u}$, where U is the entropy function chosen as

$$U = -\frac{\rho(\ln P - \gamma \ln \rho)}{\gamma - 1} \quad (33)$$

More details and discussions on entropy-adjoint adaptation can be found in the referenced work.

B. Mesh Optimization Through Error Sampling and Synthesis

We outline the mesh optimization through error sampling and synthesis (MOESS) algorithm used in this paper in this section. The algorithm and a detailed discussion of its extension to HDG can be found in previous works by Fidkowski and Chen [18,22], which built on the earlier work of Yano [21].

To form a continuous optimization problem, the information about the size and stretching of elements in a mesh, which is discrete, is encoded using a continuous Riemannian metric $\mathcal{M}(\mathbf{x}) \in \mathbb{R}^{\dim \times \dim}$. The idea of the MOESS algorithm is to optimize the change in the current mesh-implied metric $\mathcal{M}_0(\mathbf{x})$ through a step matrix $\mathcal{S} \in \mathbb{R}^{\dim \times \dim}$ as

$$\mathcal{M} = \mathcal{M}_0^{1/2} \exp(\mathcal{S}) \mathcal{M}_0^{1/2} \quad (34)$$

given a target cost $\mathcal{C}_{\text{target}}$ so that the estimated output error is minimized.

The step matrix field $\mathcal{S}(\mathbf{x})$ is represented by values at the mesh vertices \mathcal{S}_v in the implementation of the MOESS. The optimization process in the MOESS is iterative. In each iteration, an error reduction to the cost introduction ratio is calculated at each vertex, the trace of the step matrices are modified so that the vertices with the largest values of this ratio are refined, the vertices with the smallest values of the ratio are coarsened, the trace-free part of the step matrices is updated to modify the element shape with the desired anisotropy, and finally the step matrices are manipulated to constrain the total cost. The process then repeats with the updated step matrices.

The rates of change of the total error and the cost with respect to the step matrices, $\partial \varepsilon / \partial \mathcal{S}_v$ and $\partial \mathcal{C} / \partial \mathcal{S}_v$, are calculated from element-based models that relate the error indicator ε_e and elemental cost \mathcal{C}_e to the step matrix on an element e . For DG, the error convergence model is

$$\varepsilon_e = \varepsilon_{e0} e^{\text{tr}(\mathcal{R}_e \mathcal{S}_e)} \quad (35)$$

where ε_{e0} is the current error on element e , and \mathcal{R}_e is an element-specific error rate tensor determined through an error sampling process. For HDG, $\varepsilon_e = \varepsilon_e^U + \varepsilon_e^Q + \varepsilon_e^\Lambda$, and

$$\varepsilon_e^U = \varepsilon_{e0}^U e^{\text{tr}(\mathcal{R}_e^U \mathcal{S}_e)}, \varepsilon_e^Q = \varepsilon_{e0}^Q e^{\text{tr}(\mathcal{R}_e^Q \mathcal{S}_e)}, \varepsilon_e^\Lambda = \varepsilon_{e0}^\Lambda e^{\text{tr}(\mathcal{R}_e^\Lambda \mathcal{S}_e)} \quad (36)$$

where \mathcal{R}_e^U , \mathcal{R}_e^Q , and \mathcal{R}_e^Λ are also found through error sampling. The cost model is related to the trace of the step matrix, which indicates the decrease of the area of an element. The local cost on element e is

$$\mathcal{C}_e = \mathcal{C}_{e0} e^{(1/2)\text{tr}(\mathcal{S}_e)} \quad (37)$$

where \mathcal{C}_{e0} is the current cost on element e , before refinement, measured by the number of degrees of freedom.

The updated Riemann metric field at the end of the optimization is used as the input to the bidimensional anisotropic mesh generator [26] to generate the updated mesh. In practice, several iterations of the mesh optimization and flow/adjoint solution are performed, and the convergence of the targeted output is monitored.

V. One-Dimensional Results

A. Steady Linear Advection

We start by showing an example of a steady case of the linear advection equation:

$$a \frac{du}{dx} = f \quad (38)$$

The source term f is chosen so that the exact solution is

$$u(x) = \frac{\tanh(256(x - 0.4)) + 3}{2} \quad (39)$$

which contains a sharp variation at $x = 0.4$.

The DG solutions of different polynomial orders on a uniform mesh are shown in Fig. 1. The boundary values are fixed. The upwind flux is used for advection, and the continuous artificial viscosity is tested for stabilization. The artificial viscosity is able to reduce the overshoots in the solutions, even at very high orders, up to $p = 14$ tested. The amount of the artificial viscosity can be tuned by changing

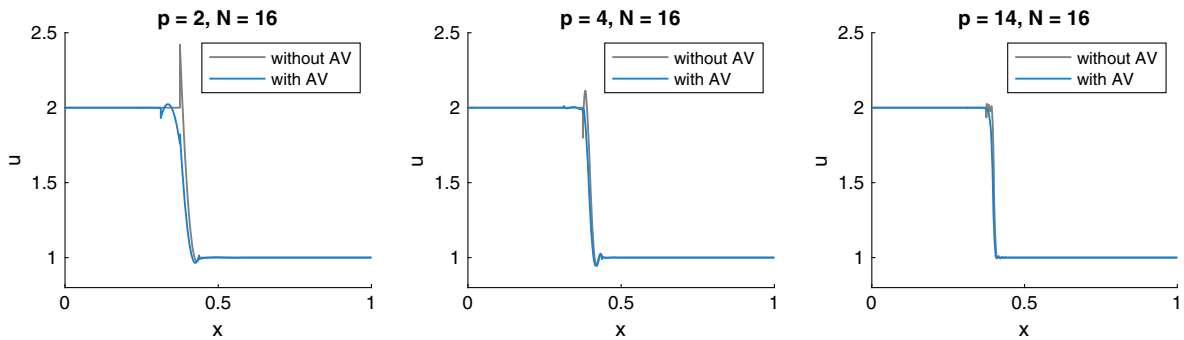


Fig. 1 DG solutions for steady advection: $a = 1.0$, $c_0 = 1.8$, and $c_p = 0.3$, with and without artificial viscosity (AV).

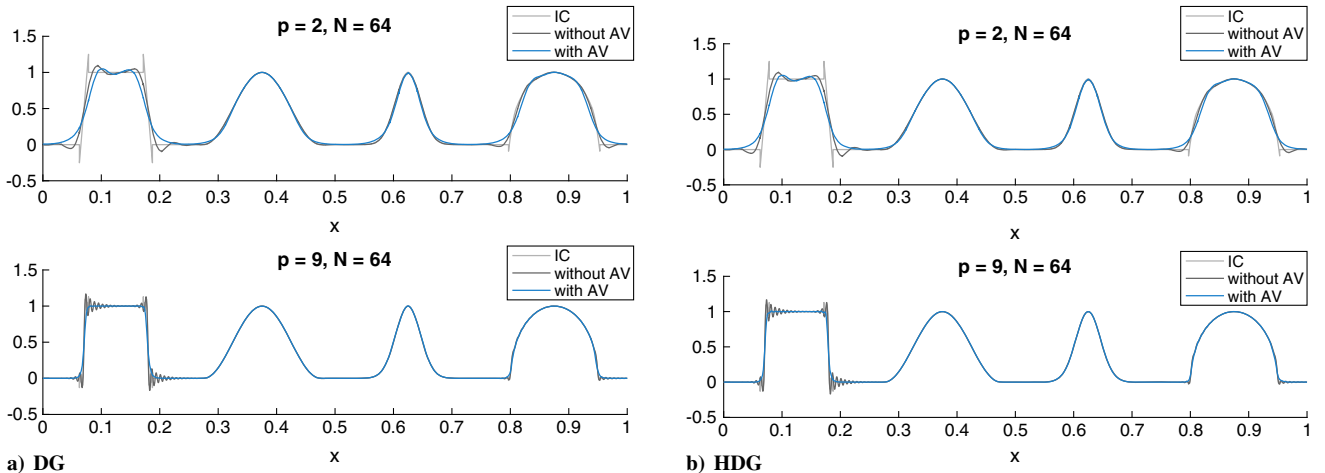


Fig. 2 DG and HDG solutions for unsteady advection: $t = 1.0$, $\Delta t = 0.001$, $c_0 = 1.0$, and $c_p = 0.4$.

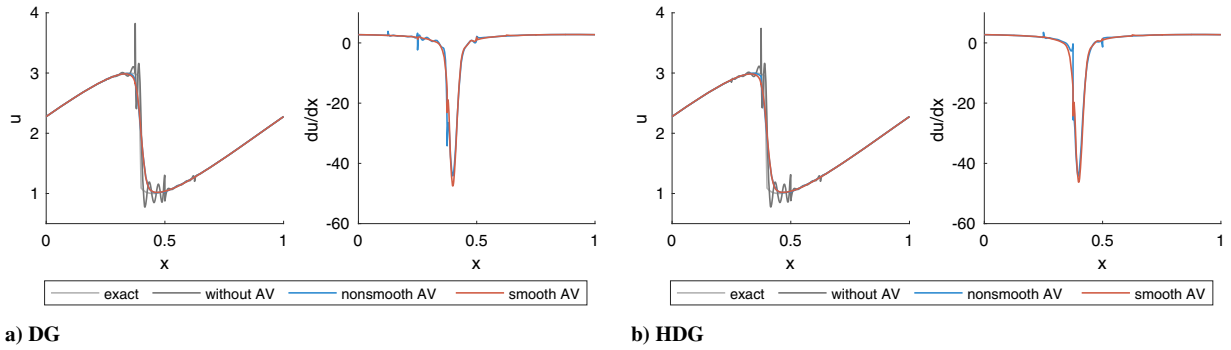
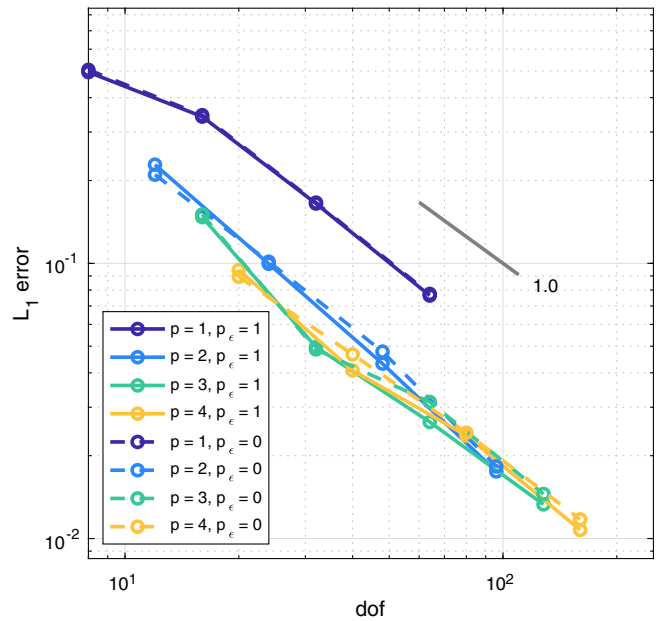
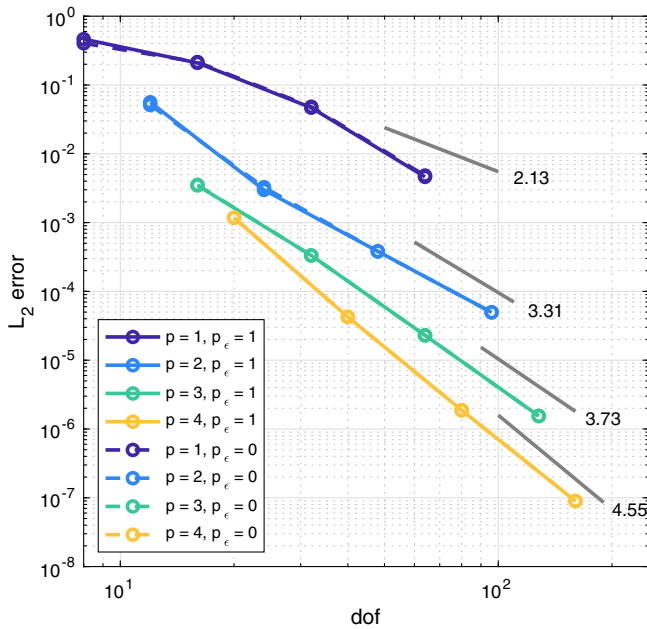
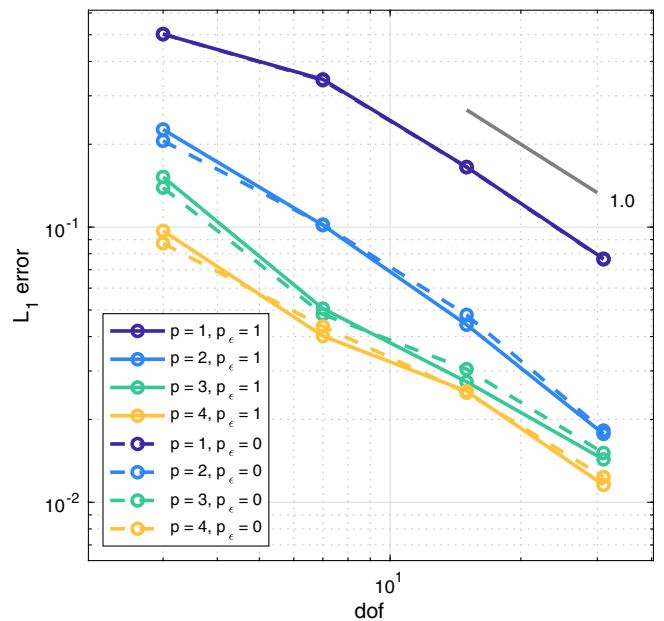
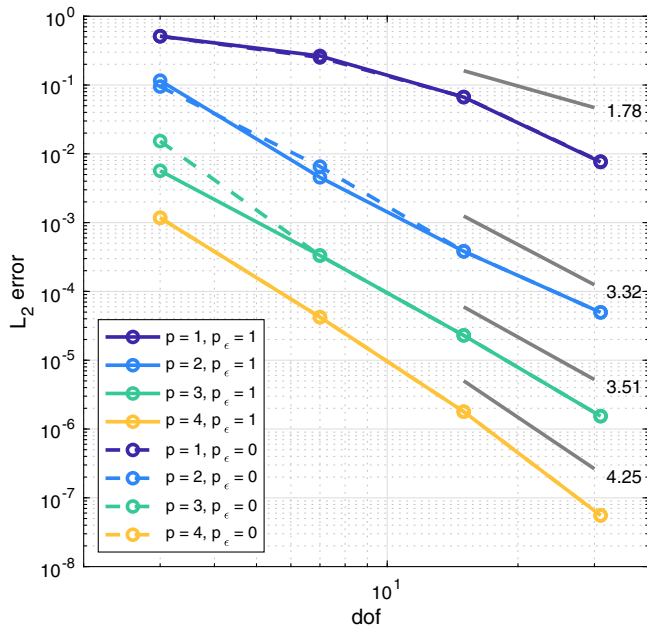


Fig. 3 DG and HDG solutions to the Burgers equation: $N = 8$, $p = 9$, $t = 0.2$, $\Delta t = 2 \times 10^{-4}$, $c_0 = 2.0$, and $c_p = 0.4$.



a) DG: $t = 0.05$ on the left and $t = 0.2$ on the right



b) HDG: $t = 0.05$ on the left and $t = 0.2$ on the right

Fig. 4 Error convergence for DG and HDG solutions to Burgers equation.

c_0 and c_p in Eq. (27) to further reduce the oscillations, but this comes at the cost of the sharpness of the discontinuity. A tradeoff needs to be made when deciding the amount of the artificial viscosity to add.

B. Unsteady Linear Advection

We demonstrate the ability of the continuous artificial viscosity to stabilize a transient solution with an example of the Zalesak “wave basket” [27] traveling with a constant speed of $a = 1$. The initial condition is imposed by a least-squares projection of the analytical solution to the order p approximation space. The solutions on a uniform mesh after one period of the wave traveling are shown in Fig. 2. The oscillations in the initial conditions, plotted as “IC,” are caused by the least-squares initialization. In this case, we test DG and HDG; and both methods preform similarly. The oscillations in the solutions are greatly reduced by the addition of the artificial viscosity.

C. Unsteady Inviscid Burgers

We compare the use of the continuous piecewise linear artificial viscosity to the discontinuous piecewise constant artificial viscosity through an example of the one-dimensional inviscid Burgers equation

$$\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} = 0 \tag{40}$$

with an initial sinusoidal wave

$$u_0(x) = \sin(2\pi(x - 0.5)) + 1 \tag{41}$$

and periodic boundary conditions. A shock wave will start to form at $t = (1/2\pi)$ as the wave propagates in space. We run the test case on a uniform mesh. The Godunov scheme [28] is used for the advection flux.

In Fig. 3, we show the solution values and first derivatives for the Burgers equation example. The derivatives of the solutions with the discontinuous artificial viscosity are more oscillatory for both the DG and HDG cases.

Given the implicit characteristics solution $u(x, t) = u(x - ut)$, the exact solution u_e can be found iteratively at each point in space. We define the L_1 error of the solution as

$$\int_0^1 |u - u_e| dx$$

and the L_2 error of the solution as

$$\sqrt{\int_0^1 (u - u_e)^2 dx}$$

The convergence of the L_1 and L_2 errors is shown in Fig. 4. At $t = 0.05$, the solution is smooth, and both DG and HDG show error convergence of the corresponding orders of the methods. At $t = 0.2$, a shock has formed, and the orders of error convergence for both DG and HDG drop to one. The errors for the piecewise constant artificial-viscosity cases at $t = 0.2$ are, in general, bigger than those of the continuous artificial-viscosity cases.

It is worth mentioning that the discontinuous artificial viscosity tends to reduce the stability of the Newton–Raphson iterations when the time step taken is relatively large. Although this instability can be alleviated by smaller time steps or under-relaxation of the Newton–Raphson iterations, it makes the solution with the discontinuous artificial viscosity more costly (about 20% as compared to the $p = 1$ artificial viscosity for this test case). This is a more serious issue when more artificial viscosity is added.

Moreover, if the amount of artificial viscosity is tuned down within the reasonable range so that the capturing of the shock is sharper but more oscillatory, the difference between the errors for the piecewise constant and the piecewise linear artificial viscosities increases further.

VI. Two-Dimensional Results

The two-dimensional test cases that we present are steady Euler cases with artificial viscosity, which are solved on unstructured triangular meshes with orders of $p_e = 1$ and $p = 2$. Laplace smoothing is used for all cases unless specified otherwise.

A. Transonic Airfoil

We demonstrate the solver’s shock capturing ability and the effect of Laplace smoothing with a transonic case at a freestream Mach number of $M = 0.8$ past a NACA 0012 airfoil at an angle of attack of $\alpha = 1.25$ deg. We run the case with both the resolution indicator ($C = 2$) and the variation indicator ($C = 0.5$), with and without Laplace smoothing, for more than 10 adaptive iterations based on the drag adjoint. The drag convergence over the adaptive iterations is shown in Fig. 5. The adapted mesh and solution for the resolution indicator with Laplace smoothing are shown in Fig. 6. One main shock appears above the airfoil, and a weak one appears below. Thin, anisotropic elements are placed along the shock interface by the mesh adaptation process.

Because the refined elements of the unstructured mesh are not perfectly aligned with the shock, the nonlinear smoothness indicators can cause oscillations in the artificial-viscosity values along the shock. Therefore, oscillations can form along the shock in the solutions. Laplace smoothing of the artificial viscosity has the potential to alleviate this effect. To compare the solutions, a line probe is taken of the entropy field behind the shock for the solutions on the final adapted meshes. Figure 7 shows the entropy measured along the line probe. Laplace smoothing is able to reduce the oscillations in the

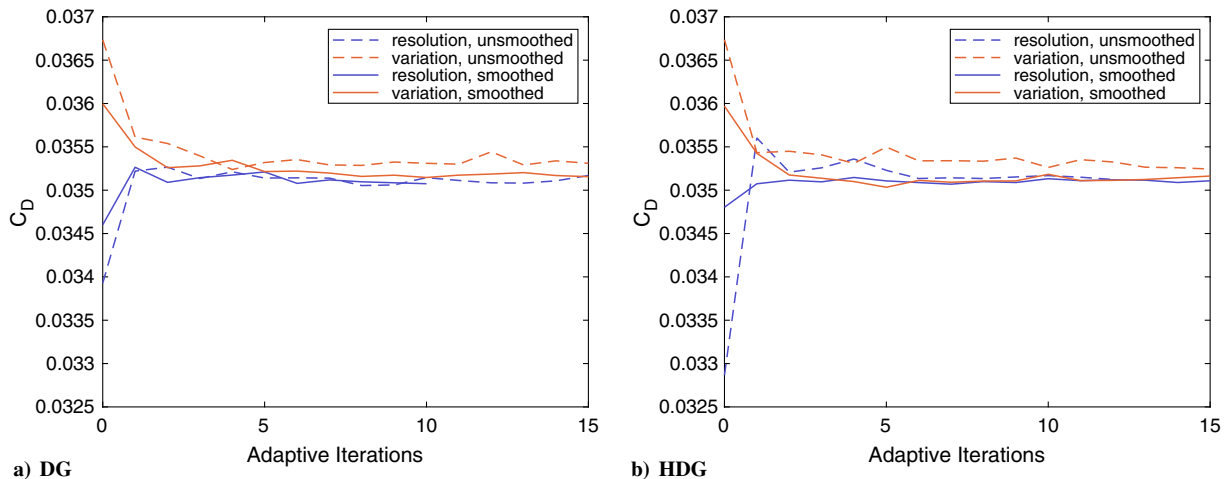


Fig. 5 Drag coefficient convergence for the transonic airfoil case.

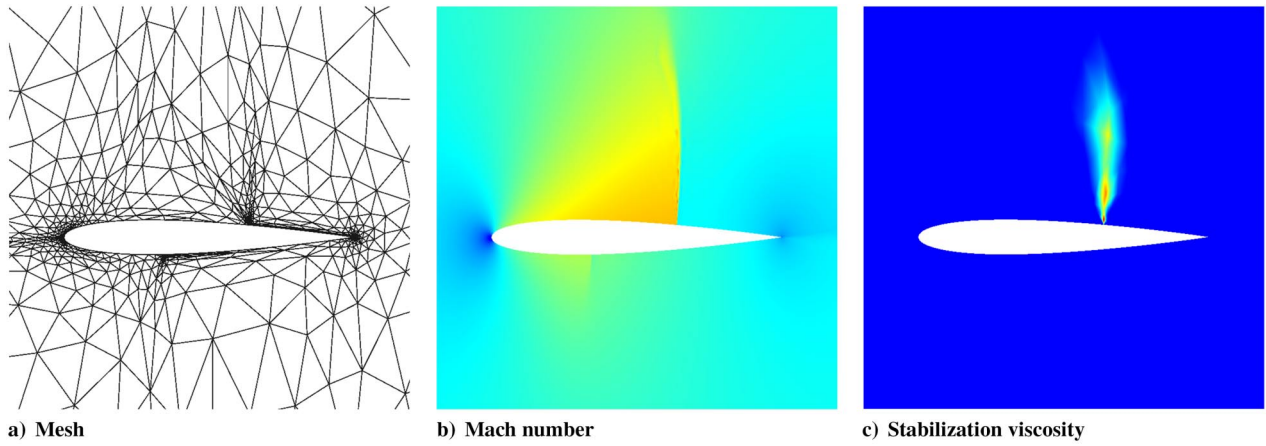


Fig. 6 Transonic airfoil DG solution with the resolution indicator and Laplace smoothing; DOFs = 1×10^4 .

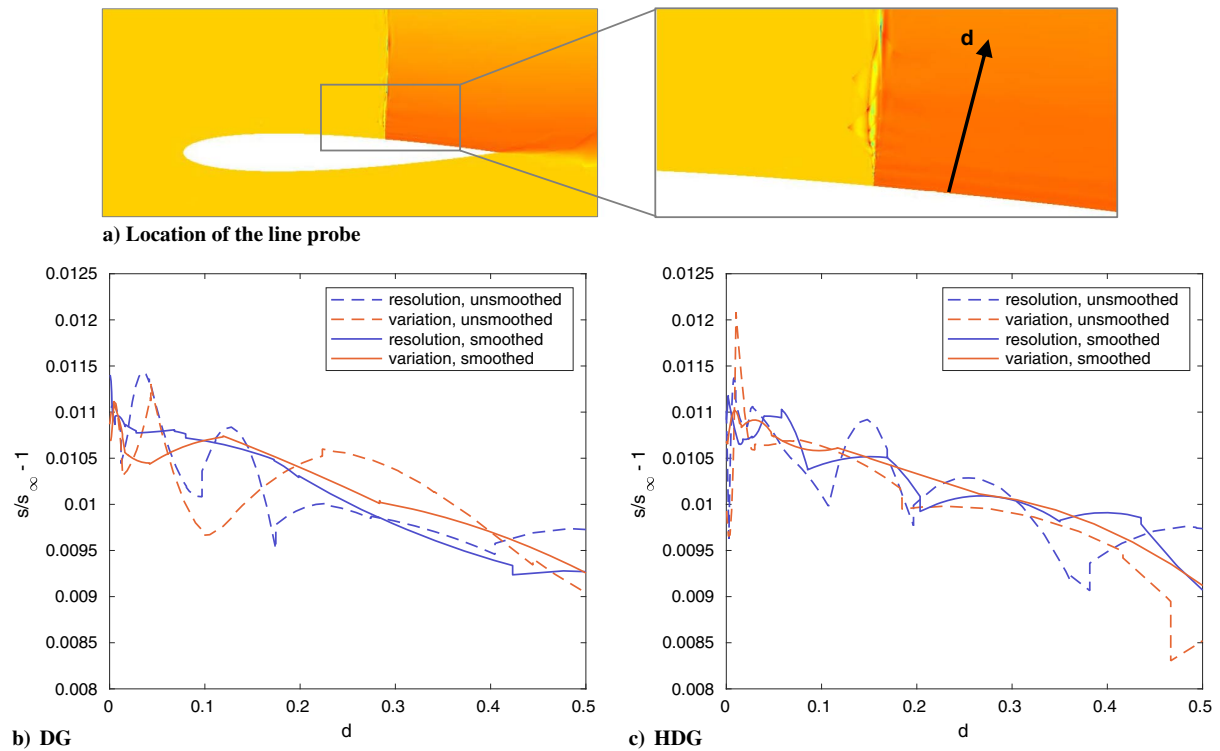


Fig. 7 Line probe of entropy behind the shock.

entropy for both the resolution and the variation indicators. Note that the Laplace smoothing is for the elemental artificial viscosity, and it happens during the process of setting up of the linear system for each Newton–Raphson iteration. The increase in computational time due to the smoothing iterations is negligible.

Figure 8 shows the pressure coefficient plots on the upper surface of the airfoil. The difference between the location of the shock determined with the resolution and the variation indicators is less than 0.1% of the chord length for both DG and HDG. The resolution of the shock improves as the mesh is refined.

B. Transonic Gaussian Bump

We investigate the errors in the total enthalpy generated by the numerical scheme with a test case of $M = 0.7$ channel flow with a smooth Gaussian bump geometry. The resolution indicator is used with $C = 4$ for all the results presented for this case. The channel is bounded in the region $[-1.5, 1.5] \times [0, 0.8]$. The bump on the bottom of the channel is defined by

$$y = 0.0625e^{-(x/0.2)^2} \quad (42)$$

The total enthalpy should be conserved across the shock for an exact inviscid solution. However, the added artificial viscosity as well as the inviscid flux function can serve as sources of total enthalpy. Figure 9 shows a convergence study of the L_2 error in the total enthalpy, which is defined as

$$E_H = \sqrt{\frac{\int_{\Omega} (H/H_{\infty} - 1)^2 d\Omega}{\int_{\Omega} d\Omega}} \quad (43)$$

on uniformly refined meshes. The modification of the state vector in the stabilization term [i.e., a nonidentity tensor T_{kl} in Eq. (19)] improves the total enthalpy solutions as well as the convergence rate. However, due to the nonlinearity of the shock capturing method, the convergence rate is still lower than one. The van Leer–Hänel flux function further reduces the total enthalpy generation. However, the use of the van Leer–Hänel flux function is found to impair the stability of the numerical scheme.

In the results in Fig. 10, we compare mesh adaptation with the drag adjoint and with the entropy variables. The drag coefficient is defined as

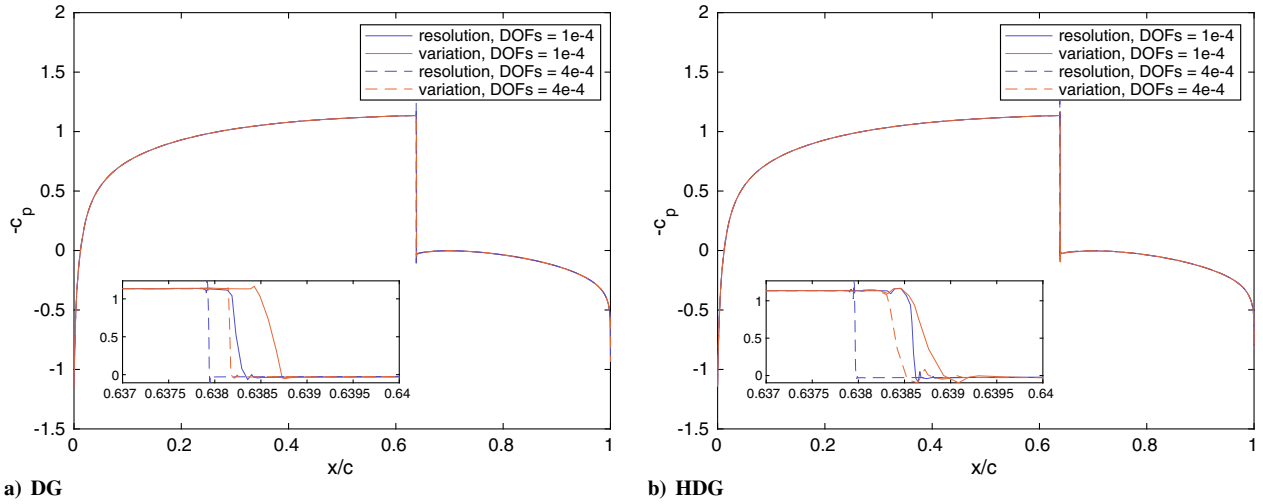


Fig. 8 Pressure coefficient distributions on the upper surface of the airfoil in transonic flow.

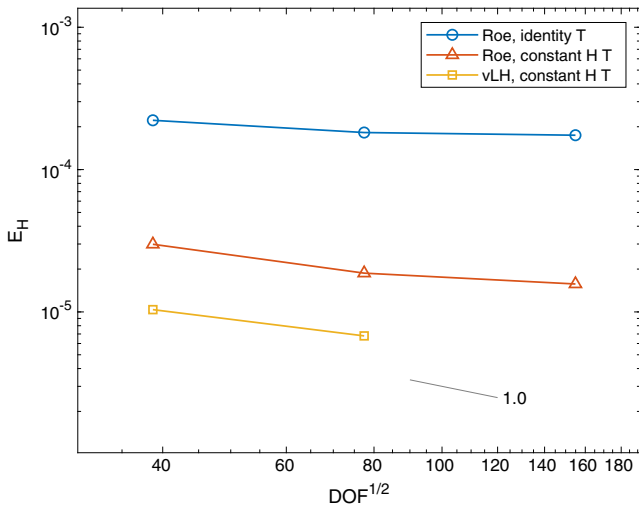


Fig. 9 Total enthalpy error convergence on uniformly refined meshes for the bump case using different diffusivity tensor transformations, as defined in Eq. (19) (T = Jacobian matrix in Eq. (19); H = total enthalpy; vLH = van Leer–Hänel flux).

$$C_D = \frac{\int_{\text{bottom}} (P - P_\infty) n_x dl}{(\gamma/2) P_\infty M_\infty^2 h} \quad (44)$$

where n_x is the horizontal component of the outward-pointing normal vector, and $h = 0.025$ is the height of the bump. The drag coefficient

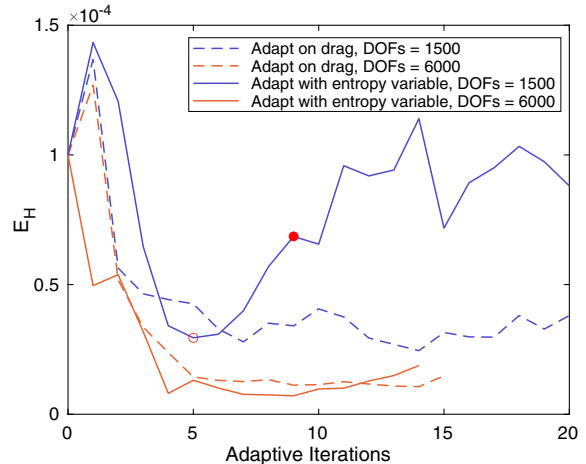
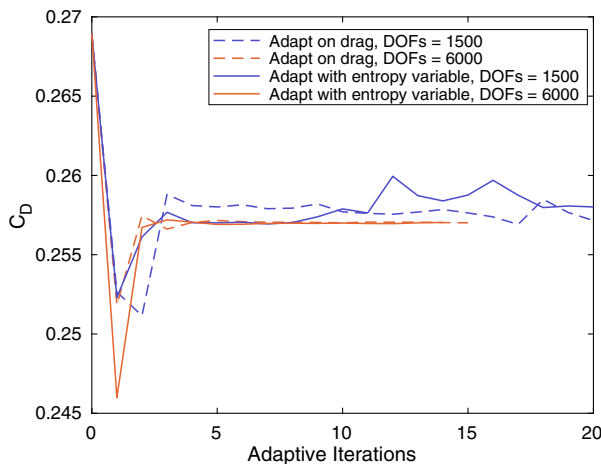


Fig. 10 Drag and total enthalpy convergence during adaptation for the bump case.

and the error in the total enthalpy both converge as the adaptation progresses when using the drag adjoint, and they eventually hover around the optimal values. However, when the entropy variables are used for adaptation, the drag coefficient and the error in the total enthalpy both eventually begin to increase after one point despite the effort of the adaptation to put elements along the shock. This seems to be relieved when the degrees of freedom increase, especially for the drag coefficient.

Figure 11 displays the meshes for the two adaptive iterations marked in Fig. 10. The mesh for iteration 9 contains more refined elements along the shock, whereas the mesh in the rest of the flow region is significantly coarsened. This iteration corresponds to the higher error in both drag and total enthalpy. This observation suggests that adaptation based on the entropy variables leads to meshes overly focused on the shock, causing insufficient resolution in the rest of the flow domain, and eventually results in inaccurate output values. Adaptation based on the drag output, on the other hand, is able to balance the resolution addition throughout the domain, producing more accurate output values and better total enthalpy preservation.

C. Hypersonic Flow past a Cylinder

The last test case that we present is hypersonic ($M = 5$) flow past a cylinder. A strong bow shock forms in front of the cylinder and stresses the high-order shock capturing. Part of the flow region behind the shock is subsonic, bounded within the $M = 1$ contour lines. Supersonic outflow boundary conditions are used at the outflow boundaries. The definition of the error in the total enthalpy varies slightly from the previous test case:

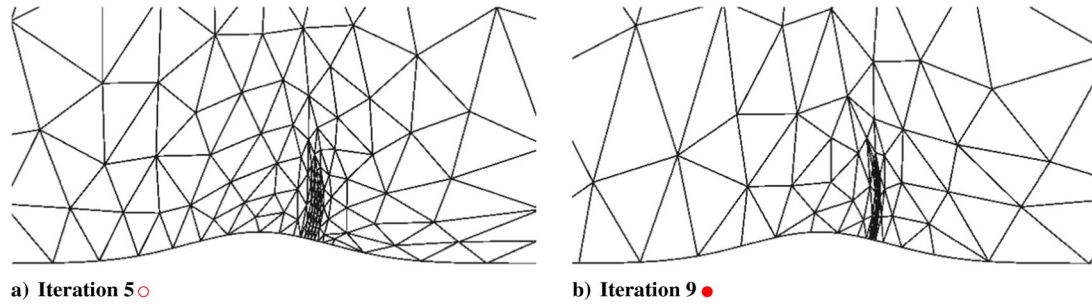


Fig. 11 Meshes adapted using the entropy variables for the bump case. The corresponding iterations are marked with the same circle marks in Fig. 10.

$$E_H = \frac{1}{D} \sqrt{\int_{\Omega} (H/H_{\infty} - 1)^2 d\Omega} \quad (45)$$

where $D = 2$ is the diameter of the cylinder. Because the primary source of enthalpy error is the region behind the shock, the length scale that is of our concern is the cylinder diameter instead of the area of the flow region.

We compare the DG and HDG performances by running both discretizations on the same mesh, starting from $M = 4.5$ solutions. The results are listed in Table 1. Compared to DG, HDG reduces the degrees of freedom (DOFs) and the number of nonlinear iterations required for convergence. However, the computational time does not improve for this case of $p = 2$. Because HDG requires static condensation and backsubstitution solutions before and after each solution of the linear system, the advantage of HDG over DG in terms of computational time is usually achieved with higher polynomial orders, when the reduction in degrees of freedom from DG to HDG is more drastic. In addition, HDG suffers from a weaker block-based preconditioner due to its smaller, more numerous blocks relative to DG; and this can increase the number of iterations taken by the linear solver.

We compare the results obtained by adapting on the total enthalpy output E_H with those obtained by adapting using the entropy variables. The Gaussian bump test case demonstrated that the entropy variables can intensively target the shock. The adaptive solutions were run for more than 15 iterations, starting from an initial structured mesh, until the outputs started oscillating about fixed values. The DG results with the resolution indicator are shown in Fig. 12. The same effect of the entropy variables is observed for the cylinder test case. The total enthalpy adjoint targets the shock and refines in the region behind the shock at the same time, whereas the entropy adjoint focuses extensively on the shock wave.

The errors in the drag coefficient and the total enthalpy are shown in Fig. 13. The drag coefficient is defined as

$$C_D = \frac{\int_{\text{cylinder}} (P - P_{\infty}) n_x dl}{(\gamma/2) P_{\infty} M_{\infty}^2 D} \quad (46)$$

where n_x is the horizontal component of the outward-pointing normal vector. The values for drag and E_H are found by averaging the output values of the last five adaptive iterations. The DOF counts plotted for both DG and HDG are the element-interior degrees of freedom because that is what the MOESS algorithm targets. The “exact” value of the drag was obtained with adaptation on E_H and DOFs = 38,400. The linear coefficient C in Eq. (20) has to be tuned to achieve convergence, as listed in Table 2. With a strong shock and the effect of the Laplace smoothing that smears

Table 1 Time and iterations required to converge to a $M = 5$ solution from a $M = 4.5$ one

	DOFs	Nonlinear iterations	CPU time, s
DG	2400	2803	3.9405×10^2
HDG	1875	2670	8.2952×10^2

the values to some extent, we resort to larger stabilization values than in the transonic cases. The variation indicator in general is more robust and requires less tuning. Moreover, the iterative convergence of HDG appears to be more sensitive to the amount of the stabilization, and so the coefficient value was lowered on the coarsest mesh.

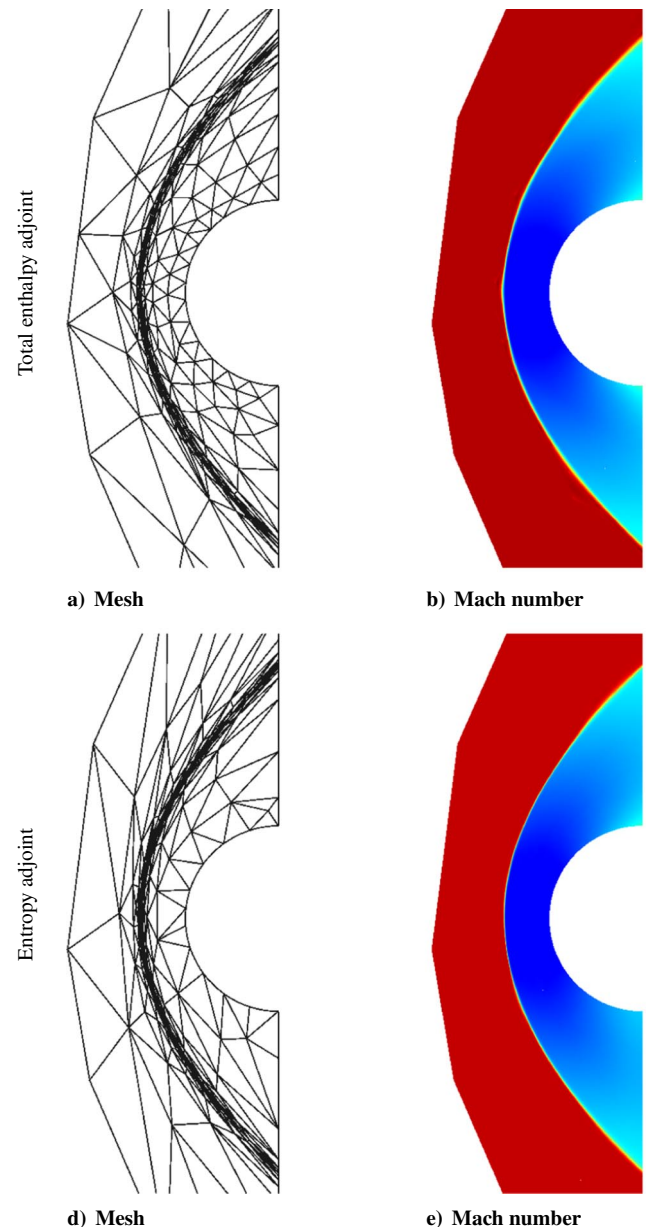


Fig. 12 Adaptive DG results for $M = 5$ inviscid flow past a cylinder. DOFs = 2400.

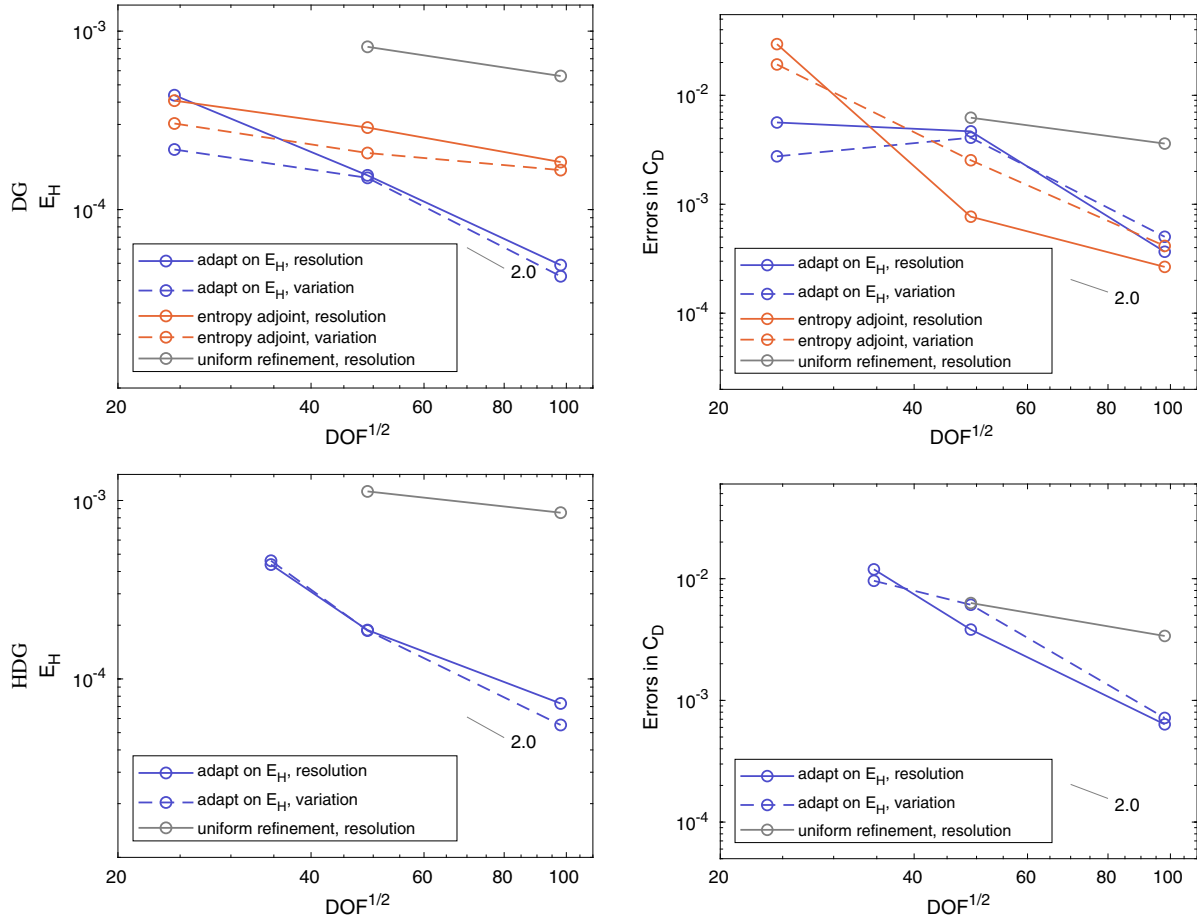
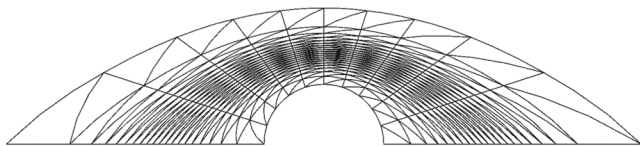


Fig. 13 Convergence plots for the cylinder case.

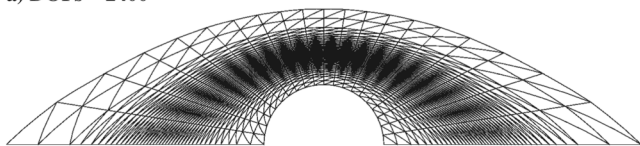
The errors for a series of structured shock-fitted meshes are also plotted in Fig. 13 for comparison. The shock-fitted meshes are shown in Fig. 14. To generate these meshes, the shock geometry is taken to be the $M = 3.5$ contour line. Uniform spacing is used in the tangent direction, and tanh spacing centered at the shock location is used to find the grid point locations along the radial direction. The tanh spacing is created with a transformation function that is applied to a linear spacing, which is defined as

$$x_{\tanh} = \frac{5}{8} \operatorname{arctanh}(x_{\text{linear}}) + 1 \quad (47)$$

where $x_{\text{linear}} \in [\tanh(-1.6), \tanh(2.4)]$, and $x_{\tanh} \in [0, 2.5]$. The radial locations of the mesh points are found as



a) DOFs = 2400



b) DOFs = 9600

Fig. 14 Shock-fitted meshes generated for comparison with mesh adaptation for the hypersonic cylinder case.

$$\frac{r}{R} = x_{\tanh} \frac{(r_s - R)}{R} + 1 \quad (48)$$

where r_s is the location of the shock for a particular tangential direction, and $R = D/2$ is the radius of the cylinder. After finding the grid points, the neighboring points are connected; and the elements are divided in half to form a triangular mesh. The flow solution is then found on the newly generated mesh, and the mesh generation and solution process are repeated several times. For the shock-fitted meshes, the elements are less skewed. $C = 32$ and $C = 16$ are used for DG and HDG, respectively, with the resolution indicator; and $C = 6$ is used with the variation indicator.

The mesh adaptation is evidently able to reduce the errors in both the total enthalpy and the drag, achieving convergence rates close to two for adaptation on E_H . The adaptation with the entropy variables shows worse convergence of E_H . However, the error in drag is reasonable. The drag does not seem to be affected too much by the lack of resolution behind the shock. Finally, the resolution and the variation indicator perform similarly for this case.

We also attempted mesh adaptation with the drag adjoint for this case. However, the adaptation focuses on the adjoint features and not as much on the shock, leading to an under-resolved shock and spurious solutions. To see the refinement along the shock more clearly, we show an example of a $M = 2$ case where the shock

 Table 2 C values in Eq. (20) for the hypersonic cylinder case

Indicator	DG	HDG
Resolution	256	128 (coarsest)/256 (otherwise)
Variation	6	6

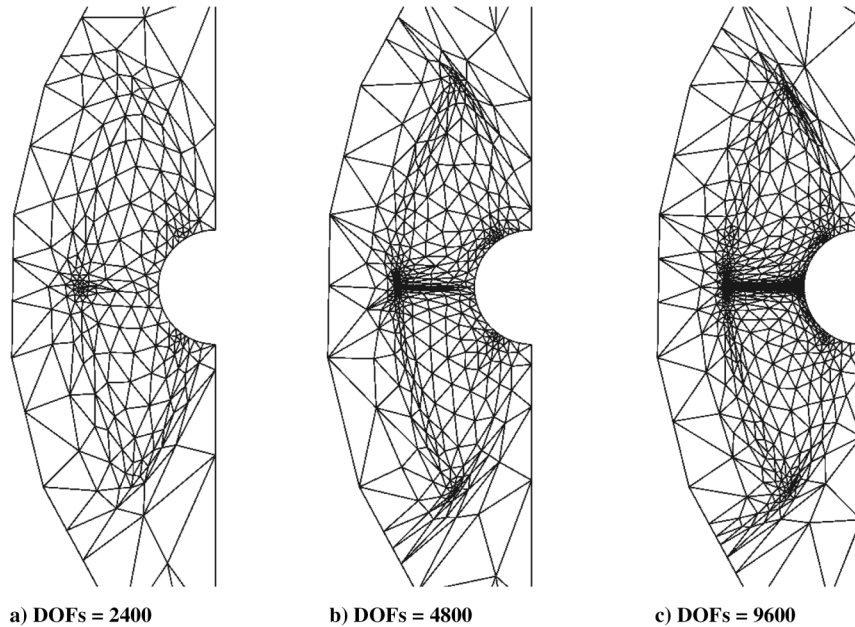


Fig. 15 Adapted meshes with MOESS for $M = 2$ flow past a cylinder using the drag adjoint.

is further away from the cylinder and the shock and the drag adjoint features are more distinguishable. The adapted meshes with $p = 2$ are shown in Fig. 15. From DOFs = 4800 to DOFs = 9600, the mesh is not refined along the shock but only on the adjoint features.

To determine whether this is a problem caused by the MOESS algorithm or the output-based mesh adaptation, we ran the same case with hanging-node adaptation [29,30]. The adaptation decides which elements to refine based on the error indicators presented in Sec. IV. Each adaptive iteration is governed by a fixed growth factor that is chosen to be 1.3. The adapted meshes are shown in Fig. 16. With hanging-node adaptation, the drag adjoint indicator leads to more refinement behind the shock and close to the cylinder boundary as compared to the total enthalpy adjoint indicator, but it does not overemphasize the drag adjoint features.

This suggests that the shock under-resolution observed with the MOESS may be due to a noisy error indicator that leads to a larger spread of error magnitudes, an inaccurate error convergence rate

calculation via MOESS sampling in certain parts of the flow, and/or an error indicator that does not respond well to refinement due to singular adjoint features. A comparison of the distributions of the adaptation error indicators between the drag adjoint and the total enthalpy adjoint is shown in Fig. 17. The large error indicator values for the drag adjoint case are localized to elements that are already small, suggesting that the MOESS is fixated on certain primal/adjoint features or their interaction. The adjoint features are specific to the drag output. The enthalpy adjoint appears smoother, and this likely makes the error indicator more spread out. As the MOESS optimizes the mesh to equidistribute the marginal error-to-cost ratio, large error indicators in certain consistent regions will lead the MOESS to gravitate mesh resolution to those regions, at the cost of lower resolution elsewhere. Hanging-node adaptation is not as sensitive to the relative error magnitudes because it only uses the magnitudes to select a fraction of elements to refine, and hence it does not suffer the same over-resolution as long as the fixed fraction is sufficiently large. From Fig. 17, the regions of overly high resolution for the MOESS appear to be the

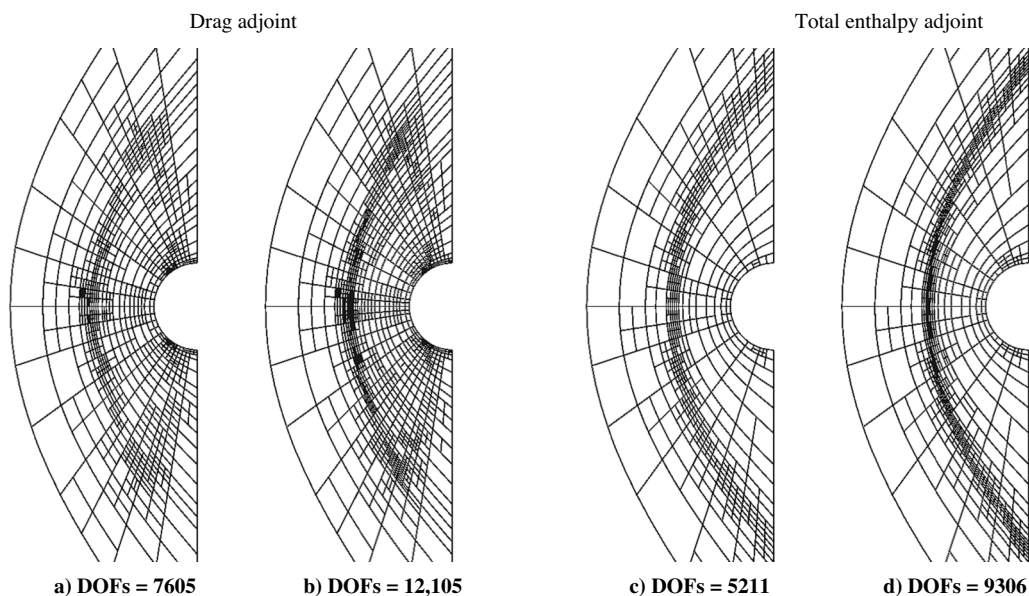


Fig. 16 Adapted meshes with hanging node adaptation for $M = 2$ flow past a cylinder.

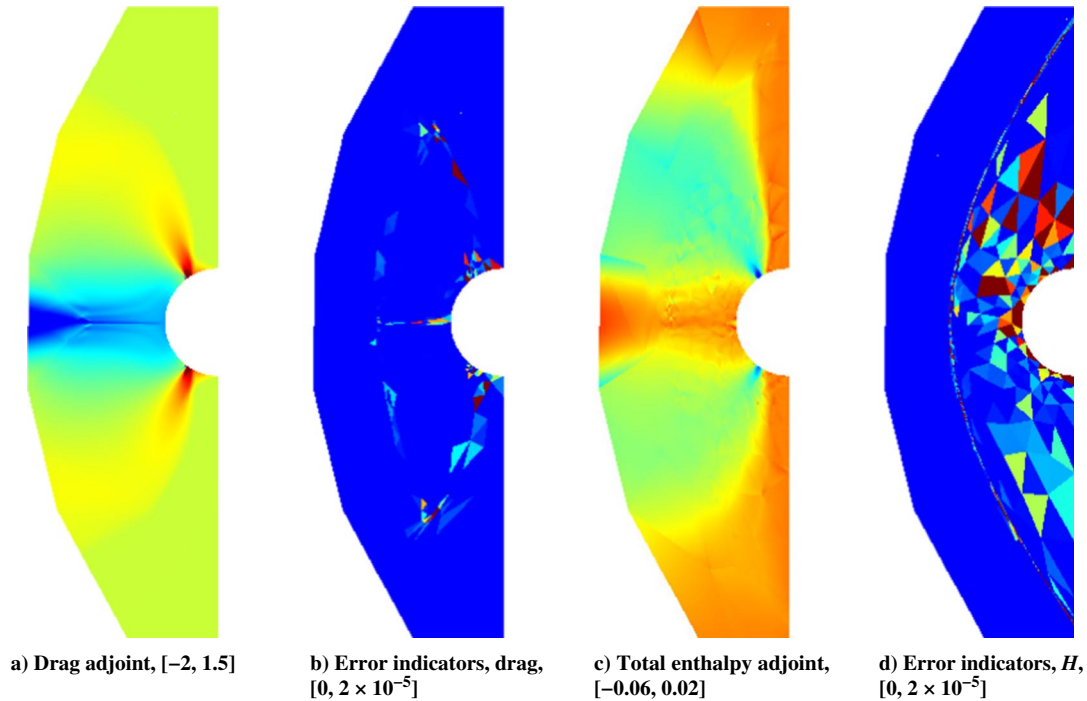


Fig. 17 Comparison of x -momentum adjoint and error indicators for $M = 2$ flow past a cylinder.

leading-edge stagnation streamline and the intersection of the bow shock and the adjoint “shock” (domain of influence boundary). Mitigating this output-based indicator disparity in the presence of strong shocks, through smoothing of the adjoint or adjustments to the adjoint-weighted residual calculation, is a topic for our future work.

VII. Conclusions

In this paper, the ability of the continuous artificial viscosity to capture shocks in an adaptive-mesh setting for both the DG and HDG discretizations was demonstrated through one-dimensional and two-dimensional simulations. Results were presented that support generally agreed upon ideas about the implementation of artificial-viscosity shock capturing. It was found that the continuous artificial viscosity works better than piecewise constant artificial viscosity in reducing oscillations near discontinuities. In the case of hypersonic flow, continuous artificial viscosity can potentially stabilize the cases that piecewise constant artificial viscosity cannot. Moreover, the use of the modified state vector in the added diffusion term is necessary to yield better total enthalpy preservation when dealing with the Euler equations.

It was discovered that the variation indicator can result in faster convergence most of the time because the resolution indicator relies on a low-order projection and is more nonlinear. However, the resolution indicator distinguishes the shock better from the other variations in the flowfield, e.g., expansion waves.

In the current experiments, the mesh adaptation with the entropy adjoint tends to focus too much on the shock, at the cost of reducing the degrees of freedom in other areas that are key to total enthalpy preservation and accurate output computation. Adaptation on the L_2 error of the total enthalpy or an integrated force can lead to better results in many cases.

Lastly, DG and HDG perform similarly in terms of the shock capturing quality for the experiments presented, with the exception that HDG is more sensitive to the amount of the artificial viscosity added. HDG has the potential to reduce the cost of the solution without compromising the solution quality.

Acknowledgments

This work used the Extreme Science and Engineering Discovery Environment (XSEDE) cluster, Expanse, at the San Diego Supercomputer Center through allocation TG-EVE210005.

References

- [1] Nguyen, N., Peraire, J., and Cockburn, B., “An Implicit High-Order Hybridizable Discontinuous Galerkin Method for Linear Convection-Diffusion Equations,” *Journal of Computational Physics*, Vol. 228, No. 9, 2009, pp. 3232–3254.
<https://doi.org/10.1016/j.jcp.2009.01.030>
- [2] Peraire, J., Nguyen, N. C., and Cockburn, B., “A Hybridizable Discontinuous Galerkin Method for the Compressible Euler and Navier-Stokes Equations,” AIAA Paper 2010-0363, 2010.
<https://doi.org/10.2514/6.2010-363>
- [3] Schütz, J., and May, G., “A Hybrid Mixed Method for the Compressible Navier–Stokes Equations,” *Journal of Computational Physics*, Vol. 240, May 2013, pp. 58–75.
<https://doi.org/10.1016/j.jcp.2013.01.019>
- [4] Wopen, M., Balan, A., May, G., and Schütz, J., “A Comparison of Hybridized and Standard DG Methods for Target-Based Hp-Adaptive Simulation of Compressible Flow,” *Computers and Fluids*, Vol. 98, July 2014, pp. 3–16.
<https://doi.org/10.1016/j.compfluid.2014.03.023>
- [5] Cockburn, B., and Shu, C.-W., “Runge-Kutta Discontinuous Galerkin Methods for Convection-Dominated Problems,” *Journal of Scientific Computing*, Vol. 16, No. 3, 2001, pp. 173–261.
<https://doi.org/10.1023/A:1012873910884>
- [6] Burbeau, A., Sagaut, P., and Bruneau, C., “A Problem-Independent Limiter for High-Order Runge-Kutta Discontinuous Galerkin Methods,” *Journal of Computational Physics*, Vol. 169, No. 1, 2001, pp. 111–150.
<https://doi.org/10.1006/jcph.2001.6718>
- [7] Qiu, J., and Shu, C., “Hermite WENO Schemes and Their Applications as Limiters for Runge-Kutta Discontinuous Galerkin Method: One-Dimensional Case,” *Journal of Computational Physics*, Vol. 193, No. 1, 2003, pp. 115–135.
<https://doi.org/10.1016/j.jcp.2003.07.026>
- [8] Luo, H., Baum, J. D., and Löhner, R., “A Hermite WENO-Based Limiter for Discontinuous Galerkin Method on Unstructured Grids,” *Journal of Computational Physics*, Vol. 225, No. 1, 2007, pp. 686–713.
<https://doi.org/10.1016/j.jcp.2006.12.017>
- [9] Neumann, J. V., and Richtmyer, R. D., “A Method for the Numerical Calculation of Hydrodynamic Shocks,” *Journal of Applied Physics*, Vol. 21, No. 3, 1950, pp. 232–237.
<https://doi.org/10.1063/1.1699639>
- [10] Persson, P.-O., and Peraire, J., “Sub-Cell Shock Capturing for Discontinuous Galerkin Methods,” AIAA Paper 2006-0112, 2006.
<https://doi.org/10.2514/6.2006-112>
- [11] Hartmann, R., “Adaptive Discontinuous Galerkin Methods with Shock-Capturing for the Compressible Navier-Stokes Equations,” *Internat-*

- tional Journal for Numerical Methods in Fluids*, Vol. 51, Nos. 9–10, 2006, pp. 1131–1156.
<https://doi.org/10.1002/flid.1134>
- [12] Barter, G. E., and Darmofal, D. L., “Shock Capturing with PDE-Based Artificial Viscosity for DGFEM: Part I, Formulation,” *Journal of Computational Physics*, Vol. 229, No. 5, 2010, pp. 1810–1827.
<https://doi.org/10.1016/j.jcp.2009.11.010>
- [13] Lv, Y., See, Y. C., and Ihme, M., “A General and Robust High-Order Numerical Framework for Shock-Capturing: Entropy-Bounding, Shock Detection and Artificial Viscosity,” AIAA Paper 2015-0572, 2015.
<https://doi.org/10.2514/6.2015-0572>
- [14] Ching, E. J., Lv, Y., Gnoffo, P., Barnhardt, M., and Ihme, M., “Shock Capturing for Discontinuous Galerkin Methods with Application to Predicting Heat Transfer in Hypersonic Flows,” *Journal of Computational Physics*, Vol. 376, Jan. 2019, pp. 54–75.
<https://doi.org/10.1016/j.jcp.2018.09.016>
- [15] Roe, P., “Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes,” *Journal of Computational Physics*, Vol. 43, No. 2, 1981, pp. 357–372.
[https://doi.org/10.1016/0021-9991\(81\)90128-5](https://doi.org/10.1016/0021-9991(81)90128-5)
- [16] Bassi, F., and Rebay, S., “GMRES Discontinuous Galerkin Solution of the Compressible Navier-Stokes Equations,” *Discontinuous Galerkin Methods: Theory, Computation and Applications*, edited by B. Cockburn, G. Karniadakis, and C.-W. Shu, Springer, Berlin, 2000, pp. 197–208.
- [17] Alexander, R., “Diagonally Implicit Runge–Kutta Methods for Stiff ODE’s,” *SIAM Journal on Numerical Analysis*, Vol. 14, No. 6, 1977, pp. 1006–1021.
<https://doi.org/10.1137/0714068>
- [18] Fidkowski, K. J., and Chen, G., “Output–Based Mesh Optimization for Hybridized and Embedded Discontinuous Galerkin Methods,” *International Journal for Numerical Methods in Engineering*, Vol. 121, No. 5, 2019, pp. 867–887.
<https://doi.org/10.1002/nme.6248>
- [19] Fidkowski, K. J., “A Hybridized Discontinuous Galerkin Method on Mapped Deforming Domains,” *Computers and Fluids*, Vol. 139, No. 5, 2016, pp. 80–91.
<https://doi.org/10.1016/j.compfluid.2016.04.004>
- [20] Hänel, D., Schwane, R., and Seider, G., “On the Accuracy of Upwind Schemes for the Solution of the Navier-Stokes Equations,” *8th Computational Fluid Dynamics Conference*, AIAA Paper 1987-1105, 1987.
<https://doi.org/10.2514/6.1987-1105>
- [21] Yano, M., “An Optimization Framework for Adaptive Higher-Order Discretizations of Partial Differential Equations on Anisotropic Simplex Meshes,” Ph.D. Thesis, Massachusetts Inst. of Technology, Cambridge, MA, 2012.
- [22] Fidkowski, K., “A Local Sampling Approach to Anisotropic Metric-Based Mesh Optimization,” *54th AIAA Aerospace Sciences Meeting*, AIAA Paper 2016-0835, 2016.
<https://doi.org/10.2514/6.2016-0835>
- [23] Becker, R., and Rannacher, R., “An Optimal Control Approach to A Posteriori Error Estimation in Finite Element Methods,” *Acta Numerica*, Vol. 10, May 2001, pp. 1–102.
<https://doi.org/10.1017/S0962492901000010>
- [24] Fidkowski, K. J., and Darmofal, D. L., “Review of Output-Based Error Estimation and Mesh Adaptation in Computational Fluid Dynamics,” *AIAA Journal*, Vol. 49, No. 4, 2011, pp. 673–694.
<https://doi.org/10.2514/1.J050073>
- [25] Fidkowski, K. J., and Roe, P. L., “An Entropy Adjoint Approach to Mesh Refinement,” *SIAM Journal on Scientific Computing*, Vol. 32, No. 3, 2010, pp. 1261–1287.
<https://doi.org/10.1137/090759057>
- [26] Borouchaki, H., George, P.-L., Hecht, F., Laug, P., and Saltel, E., “Mailleur Bidimensionnel de Delaunay Gouverné par une Carte de Métriques. Partie I: Algorithmes,” Ph.D. Thesis, National Inst. for Research in Computer Science and Automation, Rocquencourt, France, 1995.
- [27] Zalesak, S. T., “The Design of Flux-Corrected Transport (FCT) Algorithms for Structured Grids,” *Flux-Corrected Transport*, Springer, New York, 2005, pp. 29–78.
- [28] Godunov, S. K., “A Difference Scheme for Numerical Solution of Discontinuous Solution of Hydrodynamic Equations,” *Matematicheskii Sbornik*, Vol. 47, 1959, pp. 271–306.
- [29] Ceze, M. A., and Fidkowski, K. J., “Output-Driven Anisotropic Mesh Adaptation for Viscous Flows Using Discrete Choice Optimization,” AIAA Paper 2010-0170, 2010.
<https://doi.org/10.2514/6.2010-170>
- [30] Fidkowski, K. J., “Output Error Estimation Strategies for Discontinuous Galerkin Discretizations of Unsteady Convection-Dominated Flows,” *International Journal for Numerical Methods in Engineering*, Vol. 88, No. 12, 2011, pp. 1297–1322.
<https://doi.org/10.1002/nme.3224>

J. Larsson
 Associate Editor