Shock Capturing for High-Order Computational Fluid Dynamics

by

Yifan Bai

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Aerospace Engineering and Scientific Computing)
in The University of Michigan
2023

Doctoral Committee:

Professor Philip Roe, Chair
Professor Krzysztof Fidkowski
Professor Smadar Karni
Professor Feng Liu

Yifan Bai

yifanb@umich.edu

ORCID iD: 0000-0002-7877-2209

ACKNOWLEDGEMENTS

I need to first and foremost thank Prof. Roe for inviting me to Michigan and for his guidance throughout my PhD to make it a fruitful adventure. The knowledge and wisdom he shared are the most valuable things I have gained in the past few years. I want to thank Prof. Fidkowski for working with me and for being the best instructor in his CFD classes. His passion for research and teaching and his genuine care for his students have always been a great encouragement to me. Many thanks to Prof. Karni for being on my committee and for the suggestions she gave from her unique and valuable point of view. I am also thankful to Prof. Liu for being my committee member and for gently nudging me toward the direction of CFD at UC Irvine.

I have many thanks to say to the friends I have made during this journey. To Qiaoqian, Deying, Guangfei, Jie, Jianan, and Hengjie, my trusted friends from UC Irvine, for helping me through the nervousness of coming to the US for the first time until I could finally land on my feet. To Yuki, Devina, Guodong, Qingzhao, Aaditya, Fabian, and Nick, the first group of people I met in Michigan, for having made my life here so much more enjoyable. To my labmate Fanchen for offering helpful insights on research. To Abdul, Avin, Shiyao, and Tran for the fun potlucks and hangouts. To my current officemates, Miles and Alex, for countless lively conversations. To Hoang, for always being there for me, always believing in me, and sharing laughs, tears, and noodle soups with me.

I want to thank my mom for everything she has done for me. She grew up in a small town in China and earned her Master's degree in Public Health when I was in elementary school. She has already achieved more than I ever will.

The Aerospace department at the University of Michigan offered me GSI positions and fellowships to help me through PhD. Part of the computational resources for the research in the thesis is provided by Extreme Science and Engineering Discovery Environment (XSEDE).

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

ABSTRACT

This thesis contributes to shock-capturing and high-order computational fluid dynamics methods. We are concerned with convection-dominated equations, working towards Navier-Stokes solvers capable of handling large solution gradients. The two high-order methods that we focus on are the novel Active Flux method and the discontinuous Galerkin (DG) method with polynomial order 2 solution reconstruction, which has been gaining popularity in recent years.

We present a limiter for the Active Flux method based on the Flux-corrected transport approach. The limiter is designed for acoustics where no maximum principle can be determined and the solution can become unbounded due to wave focusing. It is capable of reducing and sometimes eliminating overshoots for linear and nonlinear systems. We also suggest ways for an efficient implementation of the evolution step of Active Flux. The Active Flux method produces better quality discontinuous solutions when compared to the DG method while maintaining similar efficiency.

The DG method, on the other hand, is more mature and suitable for challenging cases containing shocks, in particular for hypersonic flow. We showcase DG and hybrid DG (HDG) results from transonic to hypersonic flows using a simplified artificial viscosity shock-capturing approach coupled with output-based mesh refinement. Our discoveries include: Mesh Optimization through Error Sampling and Synthesis (MOESS) does not always generate satisfactory results in the presence of strong shocks due to error indicator disparity; an indicator based on the domain integrated total enthalpy balances the refinement on and behind the shock in many cases; HDG has the potential to reduce the cost of solution without compromising the solution quality.

CHAPTER I

Introduction

1.1 Overview of High-Order CFD methods

Since the beginning of its popularity several decades ago, computational fluid dynamics (CFD) has played a vital part in the aerodynamics community. Nevertheless, applications requiring high accuracies, such as those that involve unsteady vortices and turbulence, are out of reach for CFD methods with second-order accuracy or lower due to their formidable cost. High-order methods are favored in these scenarios for their potential to deliver high accuracy at a lower cost than the low-order methods. However, high-order methods often face a stricter time-step size requirement for explicit time-stepping. At the same time, the irregular pattern of the residual Jacobian matrix increases the cost of an implicit solution. Meanwhile, as pointed out by Wang et al. in a review paper in 2013 [1], high-order methods are usually more complicated and less robust than low-order methods. For these reasons, there is a need to develop high-order methods further.

There are a few popular categories of CFD methods, and each of them has either been extended or is native to high order. We go over some common categories in the following paragraphs.

The finite difference method becomes high order through an accurate approximation of the differential operators. This is achieved most of the time with a design of an enlarged stencil [2]. The finite difference method is usually introduced in non-conservative forms,

although conservative formulations also exist. For conservation laws, the schemes need to be carefully designed to maintain stability. In most cases, the finite difference method is limited to structured meshes, which reduces its flexibility.

The finite volume method is native to conservation laws. The stability of the scheme is achieved through upwind or stabilized fluxes. High-order finite volume methods require polynomial reconstructions of the solution and approximations of the spatial derivatives [3]. In general finite volume methods above second-order are less common, and each increment in order requires extra treatment of the solution derivatives.

In contrast, finite element methods are straightforward to bring to high orders. 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 was first introduced by Reed and Hill in 1973 for the neutron transport equation [4] and has started to receive more attention since the groundwork by Cockburn and Shu extending DG to non-linear hyperbolic problems in the 1990s [5, 6, 7]. The DG method enables high-order computations on general unstructured meshes by increasing the order of the basis functions, p. 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 [8, 9, 10, 11]. Specifically, element-interior unknowns are decoupled through the addition of the face unknowns, reducing the size of the global system to the number of face unknowns.

A few alternative high-order approaches exist. Spectral methods employ global trial functions. They offer rapid error convergence for smooth solutions but are difficult to extend to complex geometries [12]. Spectral element methods tackle this by partitioning the domain into multiple elements [13]. Spectral element methods are thus closely related to finite element methods, with the difference that boundaries in spectral element methods require special treatment. Another popular family of high-order methods in recent years is flux reconstruction methods proposed by Huynh in 2007[14]. Flux reconstruction methods work

with the differential form of the PDEs. The fluxes in the element interior are "corrected" by enforcing continuity of the reconstructions at the interfaces. The scheme reduces to the DG method with a particular choice of correction function.

The Active Flux method was first introduced in 2013 by Eymann and Roe as a multi-dimensional third-order method [15] for hyperbolic equations. Since then, the method has been extended to the Euler equations [16] and the Navier-Stokes equations [17, 18]. The Active Flux method is designed to have a continuous solution representation in space and to incorporate multidimensional physics. It allows a more relaxed time-step size compared to other high-order methods and has the potential to produce better quality and more efficient solutions, which we will discuss in this thesis.

Aside from the cost, another important consideration for high-order methods is robustness. High-order discretizations suffer from oscillations near discontinuities and other under-resolved features. In the case of the DG method, while a discontinuous approximation space appears well-suited for handling discontinuous features, in practice, taking advantage of these discontinuities is difficult for general cases [19]. In the next sections, we give an overview of some common treatments of shocks and discontinuities for high-order methods, particularly the Active Flux and the DG methods.

1.2 Resolving Discontinuities

Hyperbolic partial differential equations sustain discontinuities in their solutions. For non-linear equations, shock formation is observed in some scenarios. To capture shocks and discontinuities while preserving true high frequencies is a challenging task for high-order methods [20]. Even when a converged solution is achieved, a poorly resolved shock can pollute the solution downstream and lead to inaccurate output predictions. Several approaches exist to tackle this problem. The essential ideas include introducing numerical dissipation, reducing the solution order near the discontinuities, or leveraging the discontinuities in the solution representations. We cover a few of the popular methods here.

Artificial Viscosity

Von Neumann and Richtmyer pioneered the method of solution-based artificial viscosity in the 1950s [21]. A diffusion term is added to the governing equation where non-smoothness is present in the solution. This diffusion term usually does not directly mimic a physical viscosity but is usually taken for simplicity as a Laplacian smoothing with a variable coefficient known as the "artificial viscosity field". The amount of "viscosity" scales with the mesh size and disappears in smooth regions so that the consistency of the numerical method is maintained. Artificial viscosity has been notably adopted by Jameson et al. for their finite volume scheme [22]. It has been popular for the streamwise upwind Petrov-Galerkin finite element methods (SUPG) as a stabilization addition to the weak form of the equations since the work of Hughes and Mallet [23]. This approach was extended to the DG method by Hartmann and Houston [24, 25]. Persson and Peraire proposed a piecewise constant artificial viscosity field and the use of discontinuity sensors for DG [26]. They also argued that the artificial viscosity should scale with the DG resolution length, which is reconstruction order dependent. Up until this point, the smoothness of the artificial viscosity field has not been a concern of the research on this topic, and the formulation of the artificial viscosity is a Laplacian diffusion term. Later, Barter and Darmofal argued in their publication the importance of making the artificial viscosity field smooth and proposed a PDE-based method to find the smooth artificial viscosity [27]. Their PDE model for the artificial viscosity has the form of a heat conduction equation that has the piecewise constant artificial viscosity as the source term. The artificial viscosity field evolves together with the flow solution. Moreover, the artificial viscosity term is their implementation is total enthalpy preserving. To reduce the stiffness of the system, Ching et al. proposed a time-independent PDE as an alternative [28]. A simplified approach to make the viscosity continuous is presented and verified in Chap. 3 of this thesis.

Artificial viscosity is generally the most robust way of capturing shocks, especially for hypersonic flow. It has simple designs and is applicable to most CFD schemes and different

mesh types. However, the method relies on a smoothness indicator that is usually heuristic, and the amount of artificial viscosity often needs to be manually adjusted to achieve satisfactory solutions.

Shock Fitting

In contrast to the shock-capturing strategy of introducing stabilization, shock-fitting, or shock-tracking leverages solution discontinuities and placement of the element interfaces for high-quality shock resolution. Shock-fitting methods have the advantage of not requiring mesh refinement around the discontinuities. Bonfiglioli et al. designed a shocking fitting method that involves the remeshing of the areas around the shock interfaces [29]. It has been used to simulate shock/boundary layer interactions [30]. The development of implicit shock-tracking methods has attracted some attention in recent years [31, 32, 33, 34]. The solution that contains shocks, as well as the node locations for the node-aligned mesh, are solved at the same time. The lack of robustness is one concern for this type of methods, although some improvements have been demonstrated [32]. Moreover, these methods require major changes to the existing solver codes to incorporate, whereas artificial viscosity or limiters are much less intrusive.

Limiters

Introduced by Harten [35], the total variation diminishing (TVD) method is one of the most broadly adopted classes of limiters. For a scalar conservation law, the total variation of the solution u in the interval $x \in [0, L]$, defined as

$$\mathrm{TV}(u) = \int_0^L \left| \frac{\partial u}{\partial x} \right| dx, \tag{1.1}$$

should be non-increasing. This makes use of the fact that the time evolution does not generate new extrema or exaggerate existing extrema, and serves as the basis for designing

TVD-type limiters. Second-order TVD methods are interpreted through mathematical proof as a nonlinear combination of the first-order upwind, Lax-Wendroff, and Beam Warming methods. However, for higher dimensional systems, TVD carries over assumptions that are only true in 1D or for scalar problems. The TVD approach sometimes appears as slope limiters, e.g. MUSCL [36]. This interpretation has been taken to DG, which is well known as the RKDG scheme [7, 37]. The solution reconstruction reduces to first order where the limiter is active.

Instead of reducing the solution order near a shock, essential non-oscillatory (ENO) or weighted ENO (WENO) methods modify the solution reconstruction in troubled elements with information in neighboring elements. Created by Harten et al. [38], the idea of ENO is that the new solution should not have a greater number of extrema than the previous time step. WENO is an improvement to the ENO methods that chooses the most physically relevant stencil for the solution reconstruction [39]. WENO has been used to stabilize DG [40, 41]. As a reconstruction method, the extension of WENO to unstructured meshes is not straightforward and requires special treatment.

Flux-Corrected Transport

The flux-corrected transport method (FCT) was introduced by Boris and Book slightly preceding TVD [42, 43, 44] and was extended to multiple dimensions by Zalesak [45]. The idea has been applied to the finite element method by Lohner et al. [46]. It has been shown to generate quality solutions but has not been as popular as the TVD approach due to the lack of a mathematical proof. The essential idea of FCT is to construct an anti-diffusive flux that is the difference between the high-order flux and a low-order flux for a non-oscillatory scheme, and to limit this anti-diffusive flux before adding it to the low-order flux so that: (1) no new extrema can be created and (2) the existing extrema are not exaggerated in the low order solution. We are interested in the FCT method in particular because it does not make false assumptions for multiple dimensions. In Chap. 2 we will discuss the problem

posed by the acoustics equations: The solution is not bounded by the maximum principle and can become unbounded due to wave focusing. This makes FCT a strong candidate in our choice of limiters for the acoustics equations.

1.3 Contributions of the Thesis

- Designed a limiter for the acoustics part of the Active Flux method;
- Created test cases for linear and nonlinear equations to test the performance of Active Flux in the presence of discontinuities;
- Identified the first- and second-order components of the evolution operators for Active Flux and improved the efficiency of the code implementation;
- Investigated a continuous artificial viscosity that does not rely on the solution of an additional PDE;
- Tested the performance of the artificial viscosity for 2D transonic and hypersonic Euler cases for the DG and HDG methods;
- Investigated the use of output-based mesh adaptation with the MOESS algorithm in the presence of shocks.

1.4 Navier-Stokes Equations

We are concerned with different governing equations related to the Navier-Stokes equations or its nonviscous version, the Euler equations, throughout the thesis. Therefore they are listed here for reference.

Let $\mathbf{u}(\vec{x}, t) : \mathbb{R}^d \times \mathbb{R}^+ \to \mathbb{R}^m$ be a vector of m state variables in d spatial dimensions and time. In the physical domain Ω, a general conservation law is given by

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot \vec{\mathbf{H}}(\mathbf{u}, \nabla \mathbf{u}) = \mathbf{0}, \tag{1.2}$$

where $\vec{\mathbf{H}}(\mathbf{u}, \nabla\mathbf{u}) = \vec{\mathbf{F}}(\mathbf{u}) + \vec{\mathbf{G}}(\mathbf{u}, \nabla\mathbf{u})$, and $\vec{\mathbf{F}}(\mathbf{u}) : \mathbb{R}^m \to \mathbb{R}^{m\times d}$ is the inviscid flux, $\vec{\mathbf{G}}(\mathbf{u}, \nabla\mathbf{u}) : \mathbb{R}^m \times \mathbb{R}^{m\times d} \to \mathbb{R}^{m\times d}$ is the viscous flux.

Given the state vector $\mathbf{u} = [\rho, \rho\vec{v}, \rho E]$ for the Navier-Stokes equations, where ρ is the density, $\vec{v}$ is the velocity, and E is the total energy, the pressure can be found as $P = \frac{1}{\gamma-1}\left(\rho E - \frac{1}{2}\rho|\vec{v}|^2\right)$, where γ is the ratio of specific heats. We have $\gamma = 1.4$ for the Navier-Stokes and Euler cases throughout the thesis. The inviscid and viscous flux vectors are

$$\vec{\mathbf{F}}(\mathbf{u}) = \begin{bmatrix} \rho\vec{v} \\ \rho\vec{v}\otimes\vec{v} + P\mathbf{I} \\ \rho\vec{v}H \end{bmatrix}, \quad \vec{\mathbf{G}}(\mathbf{u}, \nabla\mathbf{u}) = \begin{bmatrix} 0 \\ -\boldsymbol{\tau} \\ -\boldsymbol{\tau}\cdot\vec{v} - k_T\nabla T \end{bmatrix}, \tag{1.3}$$

where $H = E + \frac{P}{\rho}$ is the total enthalpy, $\mathbf{I}$ is the identity matrix of size $d \times d$, T is the temperature, k_T is the conductivity coefficient, and $\boldsymbol{\tau}$ is the shear stress tensor. For a calorically-perfect gas, $P = \rho RT$, where R is the specific gas constant. The shear stress is $\boldsymbol{\tau} = \mu\left(\nabla\vec{v} + (\nabla\vec{v})^T\right) - \lambda\nabla\cdot\vec{v}\mathbf{I}$, where μ is the dynamics viscosity, and $\lambda = -\frac{2}{3}\mu$ is the bulk viscosity. The dynamic viscosity is assumed to adhere to Sutherland's law:

$$\mu = \mu_{\text{ref}}\left(\frac{T}{T_{\text{ref}}}\right)^{1.5}\left(\frac{T_{\text{ref}} + T_s}{T + T_s}\right), \tag{1.4}$$

where T_{ref} can be arbitrary and $T_s = 110\text{K}$. The thermal conductivity is related to the dynamic viscosity by the Prandtl number Pr:

$$k_T = \frac{\gamma\mu R}{(\gamma-1)Pr}. \tag{1.5}$$

We have $Pr = 0.71$ for our Navier-Stokes test cases.

The freestream viscosity is often given with the dimensionless Reynolds number $Re = \frac{\rho V_\infty L}{\mu_\infty}$, where V_∞ and L are the velocity and length scale of the flow. For the Euler equations that govern the inviscid flow behavior, μ and $\vec{\mathbf{G}}(\mathbf{u}, \nabla\mathbf{u})$ are zero.

CHAPTER II

Limiting for the Active Flux Method

2.1 Introduction to the Active Flux Method

Many of the most popular solvers in Computational Fluid Dynamics (CFD) rely on the projection of the states into one dimension depending on the mesh orientation, and the application of a Riemann solver to the discontinuous data to find the upwind flux. The result is an unsatisfactory representation of the physics and a loss of accuracy that could be potentially achieved [19]. The Active Flux method was introduced as an alternative approach to such projection. It is designed to incorporate multi-dimensional physics. The time evolution of the solution in Active Flux does not rely on the orientation of the mesh edges/faces. We are concerned with the Euler equations in Eqn. 1.2. A non-conservative form of the Euler equations, for primitive variables $\mathbf{q} = [\rho, \vec{v}, P]$ is

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \vec{v} \cdot \nabla \rho + \rho \nabla \cdot \vec{v} &= 0, \\ \frac{\partial \vec{v}}{\partial t} + \vec{v} \cdot \nabla \vec{v} + \frac{1}{\rho} \nabla P &= \vec{0}, \\ \frac{\partial P}{\partial t} + \vec{v} \cdot \nabla P + \gamma P \nabla \cdot \vec{v} &= 0. \end{aligned} \tag{2.1}$$

The second terms in these equations ($(\vec{v} \cdot \nabla)\mathbf{u}$) describe how some quantities are carried with the flow along the particle path. In the absence of other terms, the current value of a quantity can be found somewhere on the particle path at a previous time. This is termed

"advection" in Active Flux. All the other terms describe how disturbances propagate through the flow, e.g. acoustic waves. Acoustic waves propagate in all directions. The acoustics instead needs to be evaluated with an integral on a sphere. The foundation of Active Flux is the distinction between advection and acoustics. A thorough discussion of this can be found in Roe's publications [47, 19]. The distinction allows the treatment of advection and acoustics each with the correct data. The correct account of physics naturally leads to a stable scheme.

The separation of advection and acoustics also offers us the opportunity to apply limiters to each of them separately. For the advection equations, a complete set of the Riemann invariants can be found, and no new maxima or minima will be permitted. This allows the application of the limiting schemes based on the "maximum principles" that can be readily found. A survey of such schemes has been done by Zhang and Shu [48]. The acoustics equations, on the other hand, are not simultaneously diagonalizable and no maximum principles can be found. New maxima or minima can be generated even for linear acoustics due to wave focusing. Therefore developing a limiter for acoustics is the challenging part of the task for the limiting of the Active Flux method.

In the next few sections, we give a summary of a few components of the Active Flux method, before the limited scheme is presented.

2.1.1 Solution Reconstruction

The nodal values of the primitive variables are stored for Active Flux. The solution needs to be reconstructed for advection and acoustics operations. We restrict our attention to the third-order Active Flux method. Consider a computational domain Ω divided into N_e triangles. We represent a function u with discrete values U located on the element vertices and edge midpoints as in Figure 2.1. The continuity of the reconstructed solution is enforced by having a single value on each vertex and each edge. The number of discrete values (nodes) N_n is about $2N_e$. On triangle T, the third-order method requires a parabolic reconstruction

of the solution:

$$u^h|_T = \sum_{i=1}^{6} U_I \phi_{T,i}(\vec{x}), \tag{2.2}$$

where I is the global node index of the ith local node of T, and $\phi_{T,i}, i = 1, 2, \cdots, 6$ are parabolic Lagrange basis functions. To construct a low-order scheme for limiting purposes, the solution is reconstructed on a linear basis on the sub-triangles of T, see Fig. 2.1. On triangle T:

$$u^H|_T = \sum_{e=1}^{4} \sum_{i=1}^{3} U_J \varphi_{T,e,i}(\vec{x}), \tag{2.3}$$

where J is the global node index of the ith local node of sub-triangle T_e, and $\varphi_{T,e,i}, i = 1, 2, 3$ are linear Lagrange basis functions that are only non-zero inside sub-triangle T_e. The basis functions are found by mapping the elements from the physical space to the reference space, i.e., $\phi_{T,i} = \tilde{\phi}_i(\underline{J}_T^{-1}(\vec{x} - \vec{x}_0))$, $\varphi_{T,e,i} = \tilde{\varphi}_i(\underline{J}_{T,e}^{-1}(\vec{x} - \vec{x}_0))$, where $\underline{J} = \frac{\partial \vec{x}}{\partial \vec{\xi}}$ is the Jacobian of the mapping, $\tilde{\phi}_i(\vec{\xi})$ and $\tilde{\varphi}_i(\vec{\xi})$ are the Lagrange basis on a reference triangle, and $\vec{x}_0$ is the global coordinate of the node that is mapped to the origin in the reference space. To differentiate the solution with respect to the global coordinates, we use the chain rule:

$$\nabla u|_T = \nabla_{\vec{\xi}} u|_T \underline{J}_T^{-1}. \tag{2.4}$$

We sometimes consider the element average in the Active Flux method. The average value on a triangle T is defined as

$$\bar{u}|_T = \frac{1}{S_T} \int_T u \mathrm{d}S, \tag{2.5}$$

where S_T is the element area. The average for a parabolic reconstruction is the average of the nodal values on the three edge nodes.

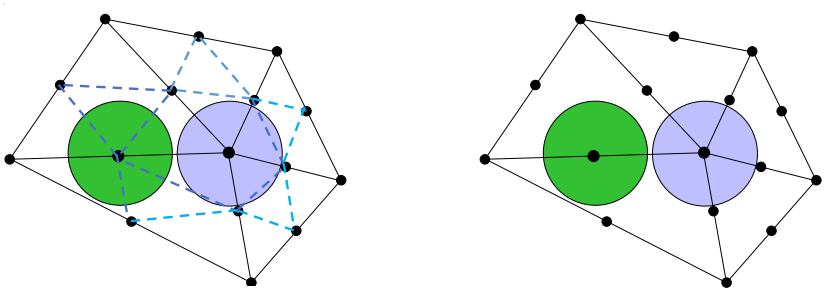


Figure 2.1: A cluster of elements. The solution is defined on the right by a quadratic interpolant and on the left by a piecewise linear interpolant on each subtriangle. The discs mark domains of dependence for edge and vertex nodes.

2.1.2 Spherical Mean Operators

The Active Flux method for the acoustics equations was first presented by Eymann and Roe [15], and refined by Fan and Roe [49]. Barsukow at el. studied the low Mach number limit of Active Flux on Cartesian grids and found the scheme to be vorticity preserving and match number compliant[50].

The linear acoustic system governs the propagation of small disturbances in a medium with constant properties and zero velocity. The flow is isentropic in this case, i.e. $\nabla P = c^2 \nabla \rho$, where c is the background speed of sound. We omit the derivation of the equations and the change of variables and present the equations as follows. For state variables $\mathbf{u} = [P, \vec{v}]$,

$$\begin{aligned} \frac{\partial P}{\partial t} + c_1 \nabla \cdot \vec{v} = 0, \\ \frac{\partial \vec{v}}{\partial t} + c_2 \nabla P = 0, \end{aligned} \tag{2.6}$$

where c_1 and c_2 are constants, and $c_1 c_2 = c^2$. Note that neither P nor $\vec{v}$ maps exactly to the variables for the Euler equations. These equations serve as a stepping stone for developing the full scheme for the Euler system. There is no maximum principle for this system. In fact, the linear case allows solutions to become unbounded due to wave focusing (see the discussion of the scalar wave equation that is satisfied here by the pressure in [51], p.215).

In the absence of rigorous bounds, we will adopt a version of the Flux-corrected Transport method (FCT) [52] for such systems.

The system in Eqn. 2.6 is equivalent to

$$\frac{\partial^2 P}{\partial t^2} = c^2 \nabla^2 P, \quad \frac{\partial^2 \vec{v}}{\partial t^2} = c^2 \nabla(\nabla \cdot \vec{v}). \tag{2.7}$$

We have an initial value problem with initial conditions: $P(\vec{x}, 0) = P_0(\vec{x})$, $\vec{v}(\vec{x}, 0) = \vec{v}_0(\vec{x})$. This problem has an exact solution in an infinite domain for $t \in [0, \infty)$, which is the classic Poisson's formula:

$$\begin{aligned} P(\vec{x}, t) &= P_0(\vec{x}) - c_1 t M\{\nabla \cdot \vec{v}_0\}(\vec{x}, ct) + \int_0^{ct} r M\{\nabla^2 P_0\}(\vec{x}, r)\, \mathrm{d}r, \\ \vec{v}(\vec{x}, t) &= \vec{v}_0(\vec{x}) - c_2 t M\{\nabla P_0\}(\vec{x}, ct) + \int_0^{ct} r M\{\nabla(\nabla \cdot \vec{v}_0)\}(\vec{x}, r)\, \mathrm{d}r, \end{aligned} \tag{2.8}$$

where $M\{f\}(\vec{x}, r) = \frac{1}{4\pi} \int_{\partial B(\vec{x}, r)} f(\vec{x} + r\vec{y}) \mathrm{d}\vec{y}$ is the mean operator on a sphere center at $\vec{x}$ with radius r. We define an operator:

$$\vec{I}\{\cdot\}(\cdot, r) := \frac{2}{r^2} \int_0^r r' M\{\nabla(\cdot)\}(\cdot, r')\, \mathrm{d}r'. \tag{2.9}$$

Eqn. 2.8 becomes

$$\begin{aligned} P(\vec{x}, t) &= P_0(\vec{x}) - c_1 t M\{\nabla \cdot \vec{v}_0\}(\vec{x}, ct) + \frac{(ct)^2}{2} \mathrm{tr}(\vec{I})\{\nabla P_0\}(\vec{x}, ct), \\ \vec{v}(\vec{x}, t) &= \vec{v}_0(\vec{x}) - c_2 t M\{\nabla P_0\}(\vec{x}, ct) + \frac{(ct)^2}{2} \vec{I}\{\nabla \cdot \vec{v}_0\}(\vec{x}, ct). \end{aligned} \tag{2.10}$$

To avoid evaluating second-order derivatives for the C^0 solution reconstruction, we use the relation:

$$M\{\nabla f\}(\vec{x}, r) = \frac{1}{r^2} \frac{\partial}{\partial r} \left(r^2 M\{f \vec{n}\}(\vec{x}, r) \right), \tag{2.11}$$

where $\vec{n}$ is the outward normal vector. This gives us:

$$\vec{I}\{f\}(\vec{x},r) = \frac{2}{r^2}\int_0^r \frac{1}{r'}\frac{\partial}{\partial r'}\left(r'^2 M\{f\vec{n}\}(\vec{x},r')\right)\,\mathrm{d}r'. \tag{2.12}$$

In two dimensions, the spherical operators degenerate to evaluations on circular disks,

$$\begin{aligned} M^{\mathrm{2D}}\{f\}(x,y,r) &= \frac{1}{2\pi r}\int_0^r\int_0^{2\pi} f(x+s\cos\theta, y+s\sin\theta)\mathrm{d}\theta\frac{s}{\sqrt{r^2-s^2}}\mathrm{d}s. \\ \vec{I}^{\mathrm{2D}}\{f\}(x,y,r) &= \\ &\frac{2}{r^2}\int_0^r \frac{1}{r'}\frac{\partial}{\partial r'}\left(\frac{r'}{2\pi}\int_0^{r'}\int_0^{2\pi} f(x+s\cos\theta, y+s\sin\theta)\begin{bmatrix}\cos\theta\\ \sin\theta\end{bmatrix}\mathrm{d}\theta\frac{s^2}{r'\sqrt{r'^2-s^2}}\mathrm{d}s\right)\mathrm{d}r'. \end{aligned} \tag{2.13}$$

Since our solution reconstructions are polynomials, we pay attention to the application of the operators to polynomials,

$$\begin{aligned} M^{\mathrm{2D}}\{x^p y^q\}(0,0,r) &= \frac{1}{2\pi}\int_0^{2\pi}\cos^p\theta\sin^q\theta\mathrm{d}\theta\ \underbrace{\frac{1}{r}\int_0^r\frac{s^{p+q+1}}{\sqrt{r^2-s^2}}\mathrm{d}s}_{\text{term } I}, \\ \vec{I}^{\mathrm{2D}}\{x^p y^q\}(0,0,r) &= \frac{1}{\pi}\int_0^{2\pi}\begin{bmatrix}\cos^{p+1}\theta\sin^q\theta\\ \cos^p\theta\sin^{q+1}\theta\end{bmatrix}\mathrm{d}\theta\ \underbrace{\frac{1}{r^2}\int_0^r\frac{1}{r'}\frac{\partial}{\partial r'}\left(r'\int_0^{r'}\frac{s^{p+q+2}}{r'\sqrt{r'^2-s^2}}\mathrm{d}s\right)\mathrm{d}r'}_{\text{term } II}. \end{aligned} \tag{2.14}$$

M and $\vec{I}$ are applied to first-order spatial derivatives of the solution. For a second-order polynomial reconstruction, the only terms present are $p+q \in \{0,1\}$. The two marked terms in Eqn. 2.14 evaluate to:

$$\begin{aligned} \text{term } I: &\quad \frac{1}{r}\int_0^r\frac{s}{\sqrt{r^2-s^2}}\mathrm{d}s = 1, \quad \frac{1}{r}\int_0^r\frac{s^2}{\sqrt{r^2-s^2}}\mathrm{d}s = \frac{\pi}{4}r, \\ \text{term } II: &\quad \frac{1}{r^2}\int_0^r\frac{1}{r'}\frac{\partial}{\partial r'}\left(r'\int_0^{r'}\frac{s^2}{r'\sqrt{r'^2-s^2}}\mathrm{d}s\right)\mathrm{d}r' = \frac{\pi}{2r}, \\ &\quad \frac{1}{r^2}\int_0^r\frac{1}{r'}\frac{\partial}{\partial r'}\left(r'\int_0^{r'}\frac{s^3}{r'\sqrt{r'^2-s^2}}\mathrm{d}s\right)\mathrm{d}r' = 1. \end{aligned} \tag{2.15}$$

Remarks:

1. To apply Eqn. 2.14 to a node N, the coordinates are shifted to center at $\vec{x}_N$ in the solution reconstruction in Eqn. 2.2 and Eqn. 2.3.

2. The spherical mean operator gets discretized when applied to discrete data. The solution reconstruction is a piecewise polynomial, therefore Eqn. 2.14 is evaluated partially inside each surrounding element of a node and summed together. For each solution node, the operator is a dot product of some coefficients that depend on the mesh geometry and the integral radius r, and the function values at the neighboring nodes. We denote the discrete operators for node N by $\mathcal{M}_r^N$ and $\mathcal{I}_r^N$. For a second-order polynomial reconstruction, the coefficients are further expanded into $\mathcal{M}_r^N = \mathcal{M}_0^N + \mathcal{M}_1^N r$ and $\mathcal{I}_r^N = \mathcal{I}_{-1}^N/r + \mathcal{I}_0^N$. In reality, the operators $\mathcal{M}_0\{\frac{\partial}{\partial x}(\cdot)\}$, $\mathcal{M}_1\{\frac{\partial}{\partial x}(\cdot)\}$, $\mathcal{M}_0\{\frac{\partial}{\partial y}(\cdot)\}$, $\mathcal{M}_1\{\frac{\partial}{\partial y}(\cdot)\}$, $\mathcal{I}_{x,-1}\{\frac{\partial}{\partial x}(\cdot)\}$, $\mathcal{I}_{x,0}\{\frac{\partial}{\partial x}(\cdot)\}$, $\mathcal{I}_{x,-1}\{\frac{\partial}{\partial y}(\cdot)\}$, $\mathcal{I}_{x,0}\{\frac{\partial}{\partial y}(\cdot)\}$, $\mathcal{I}_{y,-1}\{\frac{\partial}{\partial y}(\cdot)\}$, $\mathcal{I}_{y,0}\{\frac{\partial}{\partial y}(\cdot)\}$ are pre-computed and stored in memory before the time marching starts.

3. $\int_0^{2\pi} \cos^p\theta \sin^q\theta \mathrm{d}\theta = 0$ when $p+q$ is odd. Therefore if M and I are applied to a C^1 function, $\mathcal{M}_1$ and $\mathcal{I}_{-1}$ should evaluate to zero everywhere. This is consistent with the Taylor expansion of the spherical mean operators [53]. In a general case, when the solution is smooth (C^0) and r is proportional to the mesh size h, $\mathcal{M}_1 r^2$ and $\mathcal{I}_{-1} r$ in Eqn. 2.8 are of order $O(h^3)$.

4. $\mathcal{M}_0\{\frac{\partial}{\partial x}(\cdot)\}$, $\mathcal{M}_0\{\frac{\partial}{\partial y}(\cdot)\}$ provide estimations for the first order spatial derivatives and $\mathcal{I}_{x,0}\{\frac{\partial}{\partial x}(\cdot)\}$, $\mathcal{I}_{x,0}\{\frac{\partial}{\partial y}(\cdot)\}$, $\mathcal{I}_{y,0}\{\frac{\partial}{\partial y}(\cdot)\}$ provide estimations for the second order spatial derivatives. This observation is useful when we extend the scheme to nonlinear equations.

2.1.3 Advection Operator

Let us consider an advection example that is a high dimensional version of the inviscid Burger's equation:

$$\frac{\partial \vec{v}}{\partial t} + \vec{v} \cdot \nabla \vec{v} = 0. \tag{2.16}$$

For an initial condition $\vec{v}(\vec{x},0) = \vec{v}_0(\vec{x})$, the exact solution of these equations on an infinite domain is

$$\vec{v}(\vec{x},t) = \vec{v}_0(\vec{x} - \vec{v}(\vec{x},t)t). \tag{2.17}$$

For a third-order method, the advection speed only needs to be approximated to the second order since it is multiplied by t in the Eqn. 2.17. We do this with the approximation:

$$\vec{v}(\vec{x},t) \approx \vec{v}_0(\vec{x}) - \vec{v}_0(\vec{x}) \cdot \nabla\vec{v}_0(\vec{x})t, \tag{2.18}$$

where $\nabla\vec{v}_0$ is evaluated with the acoustics operators $\mathcal{M}_0\{\frac{\partial}{\partial x}(\cdot)\}$ and $\mathcal{M}_0\{\frac{\partial}{\partial y}(\cdot)\}$. The state is then interpolated at the upstream origin on the pathline,

$$\vec{x}^* = \vec{x} - (\vec{v}_0(\vec{x}) - \vec{v}_0(\vec{x}) \cdot \nabla\vec{v}_0(\vec{x})t)t. \tag{2.19}$$

This interpolation is done in two steps: search for the element in which $\vec{x}^*$ is located, and evaluate

$$U\{f,\vec{e}\}(\vec{x},d) = f(\vec{x} - d\vec{e}), \tag{2.20}$$

where $d = |\vec{x}^* - \vec{x}|$ and $\vec{e} = (\vec{x}^* - \vec{x})/d$ is a unit vector. The operator $U\{\cdot,\vec{e}\}$ is built during the search step when the element that contains $\vec{x}^*$ is determined. We denote the discretized advection operator for node U in direction $\vec{e}$ as $\mathcal{U}_e^N$.

2.1.4 Nonlinear Corrections

To give an example of how nonlinearity is tackled in Active Flux, we look at a set of nonlinear acoustics equations that are sometimes called the p-system. We present the equations in a simple form with variables that don't map exactly to the Euler equations.

For state variables $\mathbf{u} = [\rho, \vec{v}]$,

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \nabla \cdot \vec{v} &= 0, \\ \frac{\partial \vec{v}}{\partial t} + \nabla P &= 0, \end{aligned} \tag{2.21}$$

where $P = \frac{1}{\gamma}\rho^\gamma$ and γ is the specific heat ratio. Eqn. 2.21 gives rise to

$$\frac{\partial^2 \rho}{\partial t^2} = P'\nabla^2\rho + \underbrace{P''(\nabla\rho)\cdot(\nabla\rho)}_{\text{correction } I}, \quad \frac{\partial^2 \vec{v}}{\partial t^2} = P'\nabla(\nabla\cdot\vec{v}) + \underbrace{P''\nabla\cdot\vec{v}\nabla\rho}_{\text{correction } II}. \tag{2.22}$$

The wave speed in this case is $c = \sqrt{P'} = \rho^{(\gamma-1)/2}$. Comparing Eqn. 2.21 and Eqn. 2.22 with Eqn. 2.6 and Eqn. 2.7, we have two extra terms in the expansion. The derivatives in these terms are evaluated with the operators $\mathcal{M}_0\{\frac{\partial}{\partial x}(\cdot)\}$ and $\mathcal{M}_0\{\frac{\partial}{\partial y}(\cdot)\}$ introduced in the last section, and they are multiplied by $\frac{t^2}{2}$ to add to Eqn. 2.10 as nonlinear corrections.

We also list the expansion of the Euler equations here for reference:

$$\begin{aligned} \frac{\partial^2 \rho}{\partial t^2} =& -\frac{\partial}{\partial t}(\vec{v}\cdot\nabla\rho) - \frac{\partial}{\partial t}(\rho\nabla\cdot\vec{v}) \\ =& \underbrace{(\vec{v}\cdot\nabla\vec{v})\cdot\nabla\rho + \vec{v}\cdot\nabla(\vec{v}\cdot\nabla\rho)}_{\text{advection}} + \underbrace{\nabla^2 P}_{\text{acoustics}} \\ &+ \vec{v}\cdot\nabla(\rho\nabla\cdot\vec{v}) + \vec{v}\cdot\nabla\rho\nabla\cdot\vec{v} + \rho(\nabla\cdot\vec{v})^2 + \rho\nabla\cdot(\vec{v}\cdot\nabla\vec{v}) \\ \frac{\partial^2 \vec{v}}{\partial t^2} =& -\frac{\partial}{\partial t}(\vec{v}\cdot\nabla\vec{v}) - \frac{\partial}{\partial t}(\frac{1}{\rho}\nabla P) \\ =& \underbrace{(\vec{v}\cdot\nabla\vec{v})\cdot\nabla\vec{v} + \vec{v}\cdot\nabla(\vec{v}\cdot\nabla\vec{v})}_{\text{advection}} + \underbrace{\frac{\gamma P}{\rho}\nabla(\nabla\cdot\vec{v})}_{\text{acoustics}} + \frac{1}{\rho}\nabla P\cdot\nabla\vec{v} \\ &+ \vec{v}\cdot\nabla\left(\frac{1}{\rho}\nabla P\right) + \frac{1}{\rho^2}\vec{v}\cdot\nabla\rho\nabla P + \frac{\gamma-1}{\rho}\nabla\cdot\vec{v}\nabla P + \frac{1}{\rho}\nabla(\vec{v}\cdot\nabla P) \\ \frac{\partial^2 P}{\partial t^2} =& -\frac{\partial}{\partial t}(\vec{v}\cdot\nabla P) - \frac{\partial}{\partial t}(\gamma P\nabla\cdot\vec{v}) \\ =& \underbrace{(\vec{v}\cdot\nabla\vec{v})\cdot\nabla P + \vec{v}\cdot\nabla(\vec{v}\cdot\nabla P)}_{\text{advection}} + \underbrace{\frac{\gamma P}{\rho}\nabla^2 P}_{\text{acoustics}} + \underbrace{\frac{1}{\rho}\left(\nabla P - \frac{\gamma P}{\rho}\nabla\rho\right)\cdot\nabla P}_{=0 \text{ if isentropic}} \\ &+ \gamma\vec{v}\cdot\nabla(P\nabla\cdot\vec{v}) + \gamma\vec{v}\cdot\nabla P\nabla\cdot\vec{v} + \gamma^2 P(\nabla\cdot\vec{v})^2 + \gamma P\nabla\cdot(\vec{v}\cdot\nabla\vec{v}) \end{aligned} \tag{2.23}$$

To simplify the correction terms, the acoustics terms are calculated after the advection updates are added to the solution. We implicitly include second-order terms created by applying the first-order acoustics operator to the advection updates. Some of the terms in the corrections get eliminated, and we are left with the following,

$$\begin{aligned}
\frac{\partial^2 \rho}{\partial t^2}: &\quad \rho((\nabla \cdot \vec{v})^2 - \nabla \vec{v}^T : \nabla \vec{v}) \\
\frac{\partial^2 \vec{v}}{\partial t^2}: &\quad \frac{\gamma - 1}{\rho} \nabla \cdot \vec{v} \nabla P + \frac{1}{\rho} \nabla P \cdot (\nabla \vec{v} - \nabla \vec{v}^T) \\
\frac{\partial^2 P}{\partial t^2}: &\quad \frac{1}{\rho} \left(\nabla P - \frac{\gamma P}{\rho} \nabla \rho \right) \cdot \nabla P + \gamma P (\gamma (\nabla \cdot \vec{v})^2 - \nabla \vec{v}^T : \nabla \vec{v})
\end{aligned} \tag{2.24}$$

Only first-order derivatives remain in the correction terms. These terms are again evaluated with the operators $\mathcal{M}_0\{\frac{\partial}{\partial x}(\cdot)\}$ and $\mathcal{M}_0\{\frac{\partial}{\partial y}(\cdot)\}$.

2.1.5 Enforcing Conservation

The solution evolution with the advection or acoustics operators is not conservative. This is followed by a conservation step similar to the finite volume method, except that the fluxes are calculated with the evolution evaluations. The conservation step in Active Flux also serves as a "correction" step to the "prediction" of the evolution step to bring the scheme to third order.

We restrict our attention to inviscid flow. Integrating Eqn. 1.2 on a triangle T and from t^n to $t^n + \Delta t$ gives us the update rule for the element average:

$$S_T(\bar{\mathbf{u}}_T^{n+1} - \bar{\mathbf{u}}_T^n) = -\int_{t_n}^{t_n + \Delta t} \int_{\partial T} \vec{\mathbf{F}}(\mathbf{u}) \cdot \vec{n} \mathrm{d}l \mathrm{d}t, \tag{2.25}$$

where $\vec{n}$ is the outward normal vector. To evaluate the integral numerically on the right-hand side, we use a 9-point Simpson's rule, as illustrated in Fig. 2.2. For a function f on an edge

E with nodes m_1, m_2 and m_3, and m_2 being the midpoint node,

$$\begin{aligned}\int_{t_n}^{t_n+\Delta t}\int_E f dl dt &\approx \mathcal{F}_E^h \\ &= \frac{l_E \Delta t}{36}(F_{m_1}^n + 4F_{m_2}^n + F_{m_3}^n \\ &+ 4F_{m_1}^{n+1/2,*} + 16F_{m_2}^{n+1/2,*} + 4F_{m_3}^{n+1/2,*} \\ &+ F_{m_1}^{n+1,*} + 4F_{m_2}^{n+1,*} + F_{m_3}^{n+1,*}),\end{aligned} \tag{2.26}$$

where * denotes the solution from the evolution step. Note that this quadrature rule is exact for a parabolic function.

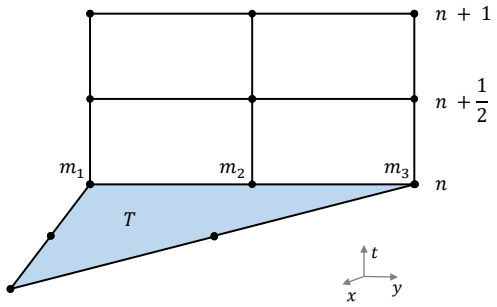


Figure 2.2: Illustration of Simpson's rule for calculating the time integral of the flux.

The discrepancies $\bar{\mathbf{u}}^{n+1,*} - \bar{\mathbf{u}}^{n+1}$ are distributed to the nodes [54]. The nodal values are updated as

$$\mathbf{u}^{n+1} = \mathbf{u}^{n+1,*} + \frac{\sum_{T\in E} S_T(\bar{\mathbf{u}}_T^{n+1} - \bar{\mathbf{u}}_T^{n+1,*})}{\sum_{T\in E} S_T}. \tag{2.27}$$

For equations that have different primitive variables and conservative variables, $\mathbf{q}^{n+1}$ needs to be found from $\mathbf{u}^{n+1}$ following this step.

2.2 Flux-Corrected-Transport Applied to the Acoustics Equations

2.2.1 Limited Active Flux

Algorithm 1

1: Initialize discrete nodal values $\mathbf{Q}^0$.
2: Calculate the time step size Δt from the geometry of the mesh and the CFL number.
3: $t = 0$, $n = 0$.
4: **while** $t < t_{end}$ **do**
5: Compute $\mathbf{Q}^{n+1,h}$, $\mathbf{U}^{n+\frac{1}{2},h}$ from the parabolic bases and $\mathbf{Q}^{n+1,l}$, $\mathbf{Q}^{n+\frac{1}{2},l}$ from the linear bases during the evolution step.
6: Compute the limited nodal values $\mathbf{Q}^{n+1,*}$, $\mathbf{Q}^{n+\frac{1}{2},*}$ and mark nodes that are limited.
7: Compute the limited temporal integral of the flux on each edge.
8: Calculate the cellular residuals from the fluxes and distribute the discrepancies onto $\mathbf{Q}^{n+1,*}$.
9: $t = t + \Delta t$.
10: $n = n + 1$.
11: **end while**

An algorithm of the limited Active Flux method is shown in Algorithm 1. At time step n, the updates to the discrete values $\mathbf{Q}^n = (P^n, U^n, V^n)$ are computed with Eqn. 2.8 on the solution reconstructions. The high order-solution $\mathbf{Q}^{n+1,h}$ computed from the parabolic reconstruction, and the low-order updates $\mathbf{Q}^{n+1,l}$ calculated from the linear reconstruction will be used to construct the limiter used in this chapter of the thesis.

To ensure stability, the time step size is constrained so that the spherical integration will not exceed the element bounds. This leads to the definition of the Courant-Friedrichs-Lewy (CFL) number:

$$\nu := \frac{c\Delta t}{\lambda}, \tag{2.28}$$

where $c\Delta t$ is the distance traveled by an acoustic wave and λ is the most restrictive distance for a particular mesh imposed by the CFL rule. This is one-half of the shortest altitude over all elements of the mesh, defined as

$$\lambda := \min_{T,E \in T} \frac{s_T}{l_{T,E}}, \tag{2.29}$$

where s_T is the area of triangle T and $l_{T,E}$ is its edge length of edge E. This definition is chosen so that the CFL limit will be given by $\nu = 1$. Note that due to the CFL constraint, the spherical integration on the linear bases will not exceed the sub-triangle bounds. See

Fig. 2.3 for an illustration.

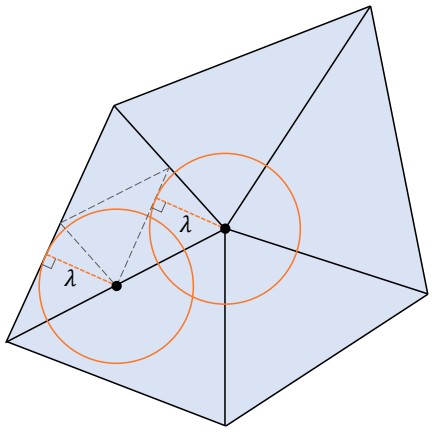


Figure 2.3: Illustration of the CFL constraint.

In our current implementation, the half-step solutions $\mathbf{Q}^{n+\frac{1}{2},h}$ and $\mathbf{Q}^{n+\frac{1}{2},l}$ will also need to be evaluated during the evolution step for the purpose of computing the temporal integration of the flux. However, it is possible to eliminate the half-time-step evaluations by manipulating the first and second-order updates, reducing memory usage, and improving the efficiency of the implementation. This is addressed in section 2.3.

2.2.1.1 FCT Limiting of the Evolution Step

The limiting criterion we employ comes from the FCT method [52]: *No value may be updated in a way that introduces a new extreme value into the solution, or that exaggerates an existing extremum.*

To implement this, for an unknown u and its discrete values U, the local maximum $U_k^{\max}$ and minimum $U_k^{\min}$ of $U^{n+1,l}$ are found on node k by a search among the neighboring nodes and itself. The neighboring nodes are sampled on the acoustics integral boundaries, as shown in Fig. 2.4. The high-order evolution update $U_k^{n+1,h}$ will then be compared to $U_k^{\max}$ and $U_k^{\min}$:

$$U_k^{n+1,*} = \begin{cases} U_k^{n+1,l}, & s_k < s_0 \textbf{ and } \left(U_k^{n+1,h} > U_k^{\max} \textbf{ or } U_k^{n+1,h} < U_k^{\min}\right) \\ U_k^{n+1,h}, & \text{else} \end{cases}, \quad k = 1, 2, \cdots, N_n \tag{2.30}$$

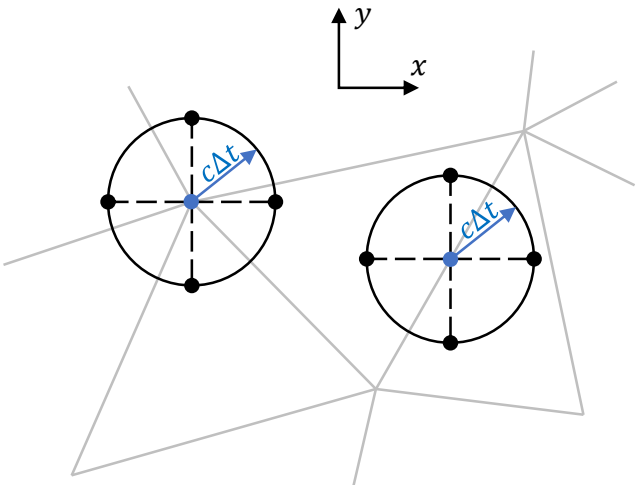


Figure 2.4: Neighboring nodes (marked by •) sampled on the acoustics boundary for a node at an element vertex or at an edge midpoint.

This is applied to each of the unknowns in $\mathbf{q} = (p, \mathbf{v})$. A smoothness indicator s_k is needed as an extra condition for turning on the limiter, which we will discuss in the next section. The difference between the high-order solution and the limited solution will also be stored as preparation for limiting the time integrals of the fluxes.

2.2.1.2 Smoothness Indicator

We construct a smoothness indicator on a node k as:

$$s_k = \frac{1}{1-\nu} \left| \frac{U_k^{n+1,h} - U_k^{n+1,l}}{U_k^{n+1,l} - U_k^n} \right|, \tag{2.31}$$

where ν is the CFL number defined in Eqn. 2.28. We decide that a function is smooth locally at the vertex or edge node k if $s_k < s_0$, where s_0 is an empirical parameter determining the limiter strength. This definition can be applied to any number of space dimensions. Based on the analysis below, we are actually dealing with smoothness in time. If s_0 is too big, too few points will get limited and if s_0 is too small, the smooth solutions will be contaminated.

Analysis:

To analyze the proposed smoothness indicator we turn to a simple linear advection problem in one dimension,

$$\partial_t u + a\partial_x u = 0, \tag{2.32}$$

where $a > 0$ is the wave speed. The solution propagates to the right: $u(x,t) = u(x - at)$. Consider a 1-dimensional domain divided uniformly into N cells of width Δx, $I_i = [x_{i-1/2}, x_{i+1/2}]$. The nodal values are stored at cell interfaces ($U_{i+1/2}$, $i = 1, 2, \cdots, N$) and also at cell centers (U_i, $i = 1, 2, \cdots, N$). We again have the parabolic reconstruction on a cell

$$u^h|_{[x_{i-1/2}, x_{i+1/2}]} = U_{i-1/2}\frac{(\xi-1)\xi}{2} + U_i(1-\xi^2) + U_{i+1/2}\frac{(\xi+1)\xi}{2}, \tag{2.33}$$

where $\xi = \frac{2(x-x_{i-1/2})}{x_{i+1/2}-x_{i-1/2}} - 1$, and the linear (low-order) reconstruction

$$\begin{aligned} u^H|_{[x_{i-1/2}, x_i]} &= U_{i-1/2}(-\xi) + U_i(1+\xi), \\ u^H|_{[x_i, x_{i+1/2}]} &= U_i(1-\xi) + U_{i+1/2}\xi. \end{aligned} \tag{2.34}$$

Consider a node at an interface $x_{i+1/2}$. Define the CFL number as $\nu = \frac{2a\Delta t}{\Delta x}$ ($0 < \nu \leq 1$), the evolution step produces

$$U^{n+1,h}_{i+1/2} = U^n_{i-1/2}\frac{-\nu(1-\nu)}{2} + U^n_i\nu(2-\nu) + U^n_{i+1/2}\frac{(2-\nu)(1-\nu)}{2}, \tag{2.35}$$

$$U^{n+1,l}_{i+1/2} = U^n_i\nu + U^n_{i+1/2}(1-\nu). \tag{2.36}$$

Using these relations, our definition of the smoothness indicator is related to the solution at time step n:

$$\begin{aligned} s_{i+1/2} &= \frac{1}{\nu}\left|\frac{U^{n+1,h}_{i+1/2} - U^{n+1,l}_{i+1/2}}{U^{n+1,l}_{i+1/2} - U^n_{i+1/2}}\right| \\ &= \frac{1}{\nu}\left|\frac{\frac{\nu^2-\nu}{2}(U^n_{i-1/2} - 2U_i + U_{i+1/2})}{(\nu-1)(U^n_i - U^n_{i+1/2})}\right| \\ &= \frac{1}{2}\left|1 - \frac{U^n_i - U^n_{i-1/2}}{U^n_{i+1/2} - U^n_i}\right| \end{aligned} \tag{2.37}$$

The analysis for nodes at cell centers produces the same results so it is omitted here. The smoothness indicator is designed using information in time, however in 1D it is shown to be a TVD-like formulation which uses curvature information to determine the smoothness at a nodal point.

2.2.1.3 Low Order Flux Integral

By using a subset of these 9 points in Fig. 2.2, we can easily design a lower order flux integral without extra information needed. We've found by numerical experiments that a first-order flux works well:

$$\int_{t_n}^{t_n+\Delta t}\int_E f dl dt \approx \mathcal{F}_E^l = \frac{l_E \Delta t}{6}\left(F_{m_1}^{n+1,*} + 4F_{m_2}^{n+1,*} + F_{m_3}^{n+1,*}\right). \tag{2.38}$$

The limited flux integral that we use for computing the residual is a linear combination of the low-order flux in Eqn. 2.38 and the high order flux in Eqn. 2.26,

$$\mathcal{F}_E = \epsilon_E \mathcal{F}_E^l + (1-\epsilon_E)\mathcal{F}_E^h, \tag{2.39}$$

where ϵ_E $(0 \leq \epsilon_E < 1)$ is calculated from the difference between the high-order updates and the limited updates in the evolution step. Note that this is a possible flux integral that we propose. Formulations other than a linear combination could also be viable. For a flux integral associated with unknown u, the definition of ϵ_E is

$$\epsilon_E(u) = \tanh\left(\alpha \frac{\sum_i^3 \frac{1}{3}\left|U_{m_i}^{n+1,h} - U_{m_i}^{n+1,*}\right|}{U^{n,\max} - U^{n,\min}}\right), \tag{2.40}$$

where $U^{n,\max}$ and $U^{n,\min}$ are the maximum and minimum nodal values in the solution at time step n, α is an empirical coefficient for adjusting the strength of the flux limiter. The bigger that α is, the sharper the tanh function becomes, the bigger ϵ_E is for the same solution values, and the more diffusive the flux integral becomes. We found that α has to be above

100 to have some effect. We tuned the value up to $\alpha = 500$ to get satisfactory limiting results for the numerical experiments in this thesis. Above this value, little change happens to the limited discontinuous cases and more numerical errors are added to the smooth cases.

2.2.1.4 Solution Initialization in the Presence of Discontinuities

We wish to obtain solutions that represent discontinuities in a "satisfactory" manner. This will depend on how discontinuous data is initialized, as well as on how it is subsequently propagated. There are several ways to initialize discontinuous data. We choose two initializations that work for smooth data, and then take a definition of discrete conservation that allows us to interpret the results for non-smooth data. Consider a simple pointwise initialization of a solution that contains a jump in values from u_1 to u_2 across some discontinuity not aligned with the local mesh. See Fig 2.5 for a triangle with nodal values that are assigned partially to u_1 and partially to u_2. The discontinuity in the solution reconstruction inside the triangle will probably be misaligned with the actual orientation of the discontinuity. This will result in different directions of wave propagation for different elements that contain the same discontinuity on an unstructured mesh. An example will be shown in the next section in which the interpolation initialization causes a certain level of artificial fluctuation in the solution.

Although we doubt that this problem can be eliminated, we attempt to alleviate it by opting for an L_2 initialization of the test cases that contain discontinuities in this paper. Inside a triangle T, where the exact initial data is $u_0(\vec{x})$, the quantity

$$\int_T \left(\tilde{U}_{T,i} \phi_{T,i}(\vec{x}) - u_0(\vec{x}) \right)^2 ds \tag{2.41}$$

is minimized to find $\tilde{U}_{T,i}, i = 1, 2, \cdots, 6$ locally for the nodes of T. The solution on a node

is then initialized to an average of the local nodal values from its neighboring elements:

$$U_I^0 = \frac{\sum_{I \in T} \tilde{U}_{T,i_T}}{N_I}, \tag{2.42}$$

where I is a global node number, T denotes the triangles that contain node i, i_T is the local node number of node I in triangle T, and N_I is the number of elements adjacent to node I.

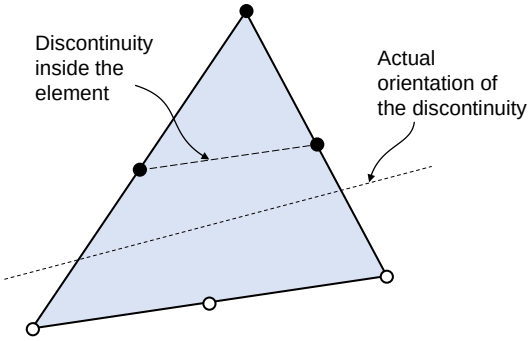


Figure 2.5: Discontinuity initialization inside a triangle. Filled nodes are assigned values u_1 and unfilled nodes are assigned u_2.

2.2.2 Test Cases: Linear Acoustics

The first set of test cases that we present is of linear acoustics in Eqn. 2.6. We choose the constant for the smoothness indicator to be $s_0 = 1.4$ for the examples in this section.

2.2.2.1 Case A: Periodic Sinusoidal Wave

We demonstrate the accuracy of the limited Active Flux scheme with a periodic sinusoidal wave on a computational domain $[-1, 1] \times [-1, 1]$ with periodic boundary conditions. This test case was first used by Lukácová-Medvidová et al. to test various second-order methods

[55]. The analytical solution is periodic in time with period $T = 1/c$:

$$\begin{aligned} P(\vec{x}, t) &= \frac{1}{c}\cos(2\pi ct)[\sin(2\pi x) + \sin(2\pi y)], \\ u(\vec{x}, t) &= -\frac{1}{c}\sin(2\pi ct)\cos(2\pi x), \\ v(\vec{x}, t) &= -\frac{1}{c}\sin(2\pi ct)\cos(2\pi y). \end{aligned} \tag{2.43}$$

We choose $c = 1$ ($T = 1$), $\nu = 0.7$. The L_2 error for a function denoted by g for time step n is defined as

$$E_g^n = \sqrt{\frac{1}{N_n}\sum_i^{N_n}(G_i^n - g^{\text{exact}}(\vec{x}_i, t_n))^2}. \tag{2.44}$$

This is measured on unstructured meshes and shown in Fig. 2.6 against the mesh size, defined as

$$h = \sqrt{\Omega/\text{DOF}}. \tag{2.45}$$

The solutions with limiters show a slight increase in errors but the third-order accuracy is preserved. The difference between the limited and unlimited solutions gets smaller as the degrees of freedom increase.

2.2.2.2 Case B: One-dimensional Discontinuities

The first discontinuous case we consider is a simple 1D discontinuity initialized on a 2D unstructured mesh on a domain $[-1, 1] \times [-1, 1]$.

$$P_0(x, y) = \begin{cases} 2, & x \leq 0 \\ 1, & x > 0 \end{cases}, \quad \vec{v}_0(x, t) = \vec{0}. \tag{2.46}$$

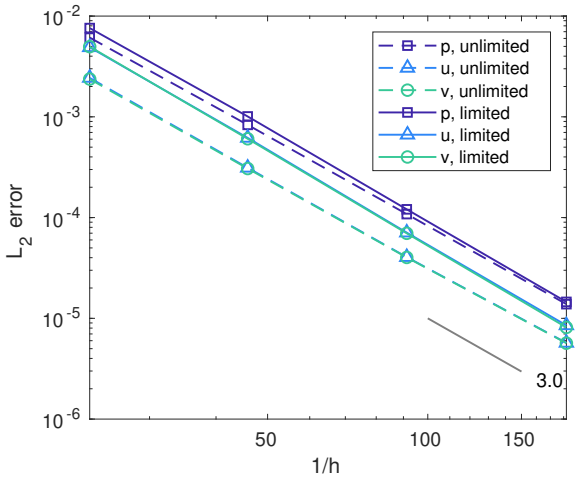


Figure 2.6: Error convergence of the double sinusoidal wave for $t = 1$.

The boundary values are fixed on the left and right sides and periodic boundary conditions are used for the upper and lower sides. The initial discontinuity in the pressure will create two discontinuities of the same strength propagating to the left and to the right.

To test our 2D solver, a comparison between the 1D and 2D unlimited solutions for this case is shown in Fig. 2.7. The 2D solver is used for the rest of the test cases presented in this section. The 2D solution is obtained on an unstructured mesh with L_2 initialization. The overshoots in both solutions are about the same magnitudes, as are the widths of the transitions. The main difference is that the 2D solution adds some scatter, especially in the central region. This is due to the initialization of the discontinuity on an unstructured mesh.

Next, we want to investigate the effect that the initialization has on the quality of the solution. The case was run with $\nu = 0.7$ without limiting until $t = 0.2$ and the unlimited solutions with different types of initialization are shown in Fig. 2.8. The solution initialized with point values clearly shows oscillations in the middle plateau created as the two discontinuities propagate in opposite directions. This can be explained by the discontinuity misalignment illustrated in Fig. 2.5. The L_2 initialization is able to alleviate this problem since it doesn't enforce a strict alignment between the orientation of the discontinuity and

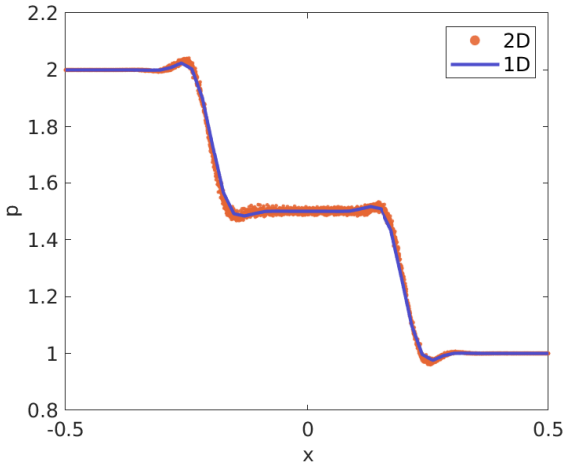


Figure 2.7: Comparison of 1D and 2D unlimited solutions at $t = 0.2$ for case B with similar mesh sizes.

the element edges inside each element. The L_2 error in the center region normalized by the jump magnitude is defined as

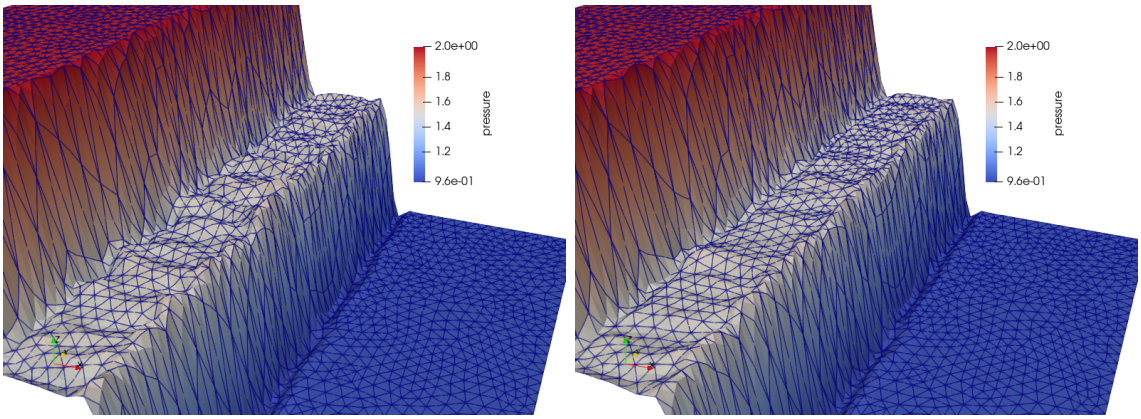
$$E^{\text{center}}(t) = \sqrt{\frac{\sum_{i,-\frac{ct}{2}<x_i<\frac{ct}{2}}^{N_n} (P_i - 1.5)^2}{\sum_{i,-\frac{ct}{2}<x_i<\frac{ct}{2}}^{N_n} 1}}. \tag{2.47}$$

These are plotted against time in Fig. 2.9 for different mesh sizes. The oscillations are shown to decrease in magnitude in proportion to the parameter $\frac{ct}{h}$, which is the number of cells crossed since initialization. The $L2$ initialization leads to lower- and faster-decaying errors in the center region.

The limited and unlimited solutions are shown in Fig. 2.10. The limiter is able to remove the major overshoots at the front of the discontinuities, and the sharpness of the discontinuities is well preserved. The oscillations in the central region also seem to have been slightly reduced.

2.2.2.3 Case C: Asymmetric Discontinuities in One Dimension

Since the same-strength wave propagating in case B could be a special example, we also show here a case with discontinuities of asymmetric strengths. This is created by having



(a) Interpolation (b) L_2

Figure 2.8: Comparison of solutions at $t = 0.2$ for two initialization strategies for test case B.

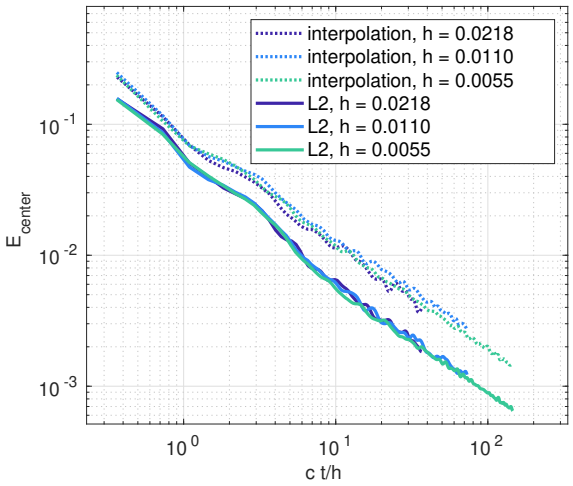
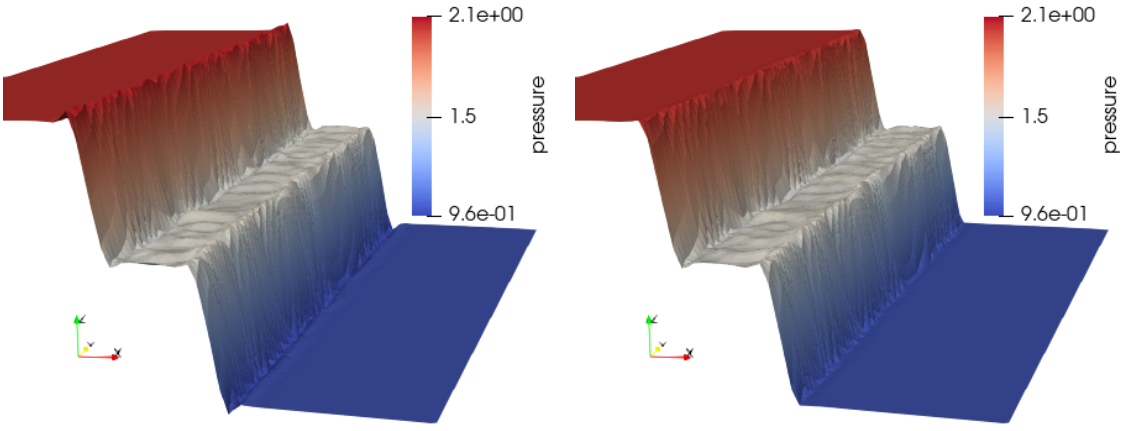


Figure 2.9: Error of the center region reduces in time for test case B.



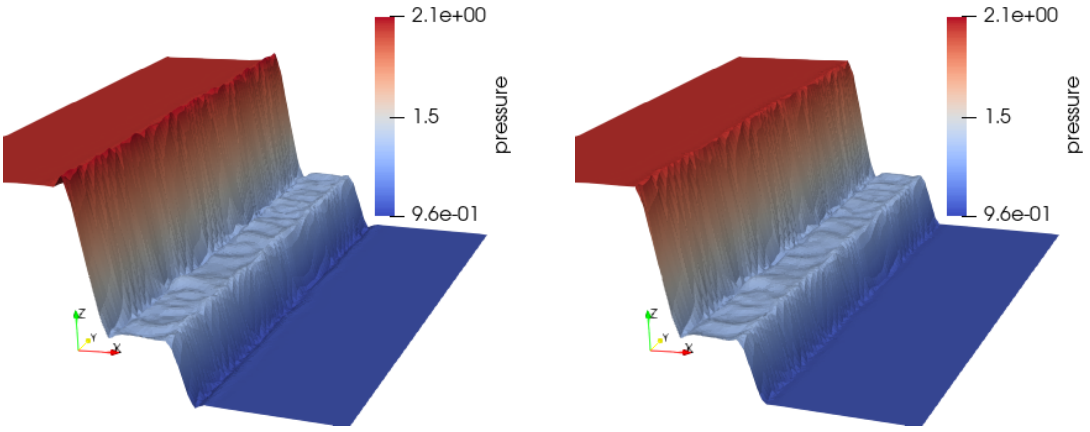
(a) Unlimited (b) Limited

Figure 2.10: Comparison of unlimited and limited solutions at $t = 0.2$ for test case B. $h = 0.0218$.

initial discontinuities in both the pressure and the velocity:

$$P_0(x,y) = \begin{cases} 2, & x \le 0 \\ 1, & x > 0 \end{cases}, \quad u_0(x,t) = \begin{cases} -0.2, & x \le 0 \\ 0.2, & x > 0 \end{cases}, \quad v_0(x,t) = 0. \tag{2.48}$$

The pressure solutions are shown in Fig. 2.11. The limiter continues to work well for this case, with the major overshoots on the two sides being eliminated.



(a) Unlimited (b) Limited

Figure 2.11: Comparison of unlimited and limited solutions at $t = 0.2$ for test case C. $h = 0.0218$.

2.2.2.4 Case D: Discontinuity in the Radial Direction

The next example is an initial discontinuity in the radial direction.

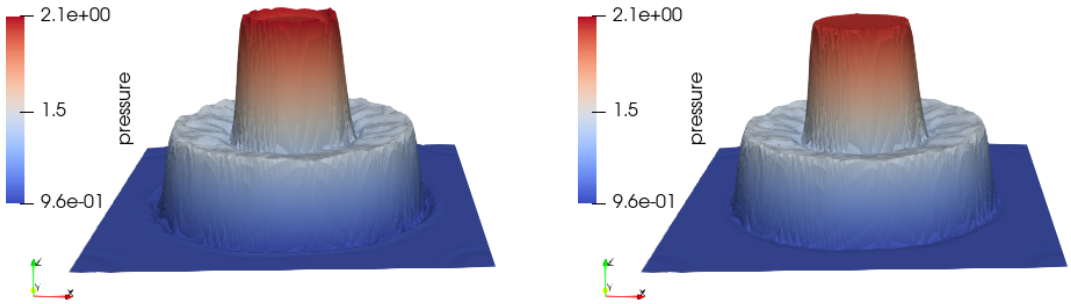
$$P_0(x,y) = \begin{cases} 2, & r \leq 0 \\ & \\ 1, & r > 0 \end{cases}, \quad \vec{v}_0(x,t) = \vec{0}, \tag{2.49}$$

where $r = \sqrt{x^2 + y^2}$. We again have a domain of $[-1,1] \times [-1,1]$ and periodic boundary conditions on all sides. Note that this doesn't reduce to the same case in the radial direction as case B. Assuming axial symmetry, the differential form of Eqn. 2.6 written in cylindrical coordinates and assuming axial-symmetry is

$$\begin{aligned} \frac{\partial P}{\partial t} + \frac{1}{r}\frac{\partial (r u_r)}{\partial r} &= 0, \\ \frac{\partial u_r}{\partial t} + \frac{\partial P}{\partial r} &= 0, \end{aligned} \tag{2.50}$$

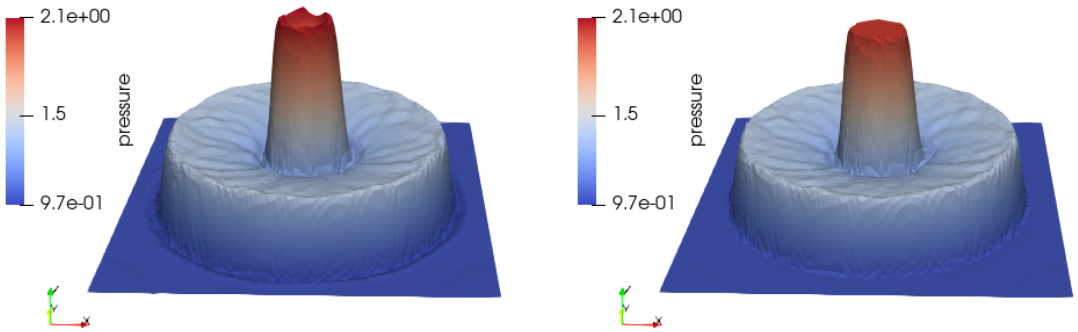
where u_r is the velocity in the radial direction. An extra source term is present in the equation for P.

The solutions for these cases with $\nu = 0.7$ are shown in Fig. 2.12 and Fig. 2.13. The major overshoots are removed and the sharpness of the discontinuity is kept. The oscillations in the major plateau have decreased over time.



(a) Unlimited (b) Limited

Figure 2.12: Comparison of unlimited and limited solutions at $t = 0.2$ for test case D. $h = 0.0218$.



(a) Unlimited (b) Limited

Figure 2.13: Comparison of unlimited and limited solutions at $t = 0.3$ for test case D. $h = 0.0218$.

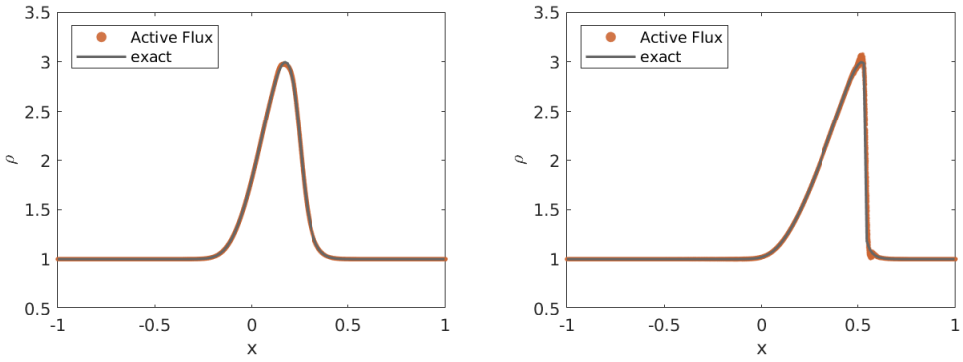
2.2.3 Test Cases: Converging-Diverging Shock

In this section, we focus on the nonlinear equations where shocks can form from smooth solutions and grow in strength. We are particularly interested in the cylindrical symmetric cases of an initial shock propagating inwards in the radial direction. This setup is similar to the Guderley problem for the Euler equations [56], which features an infinitely strong converging shock. Related problems are frequently found in laser fusion and astrophysics [57, 58, 59]. The Guderley problem has a similar solution involving a reflected diverging shock [60]. The amplitudes of the solution quantities blow up at the time of the collapse.

This problem is challenging for numerical solvers due to the growth of the shock strength and the amplitude blowup of the solution. We study here the performance of the Active Flux method for the converging-diverging problem governed by the p-system in Eqn. 2.21 and the Euler equations in Eqn. 2.1.

2.2.3.1 Validation Case for the p-System

A validation test case for our 2D p-system solver is a 1D shock formation from an initial Gaussian wave in a square domain of $[-1, 1] \times [-1, 1]$ with periodic boundaries on all sides.



(a) t = 0.1 (b) t = 0.3

Figure 2.14: p-system solution for an initial Gaussian wave. The solution is smooth at $t = 0.1$ and a shock has formed at $t = 0.3$.

The initial condition is:

$$\rho_0(x, y) = 1 + 2e^{-50x^2}, \; u_0 = \frac{2}{\gamma + 1}\rho_0^{\frac{\gamma+1}{2}}. \tag{2.51}$$

The initial condition is set so that one of the Riemann invariants is zero, and we have a simple wave governed by $\partial_t \rho + \rho^{\frac{\gamma-1}{2}} \partial_x \rho = 0$. We choose $\gamma = 2$ for the test cases presented in this thesis for the p-system. The CFL number is $\nu = 0.6$.

The exact solution of this simple wave is $\rho(x, t) = \rho_0(x - \sqrt{\rho}t)$. The initial Gaussian wave is expected to travel in the positive x-direction. A shock forms at $t = 0.24$. Fig. 2.14 shows the solutions before and after the shock formation. The exact solution is found iteratively with the Newton-Raphson method. The solver is able to capture the correct location of the shock with some slight oscillations. The L_2 errors of the density are shown in Fig. 2.15 against mesh size. The error convergence rate is close to third order when the solution is smooth and drops after the shock formation.

2.2.3.2 Converging-Diverging Shock for the p-System

To construct the imploding shock, the jumps in the variable states are calculated using the Rankine-Hugoniot condition. The initial shock is placed at $r_0 = 6$. The preshocked

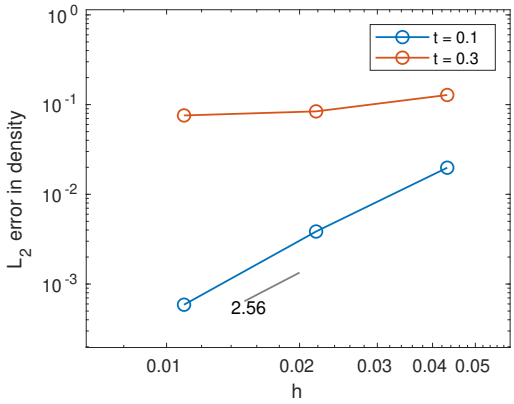


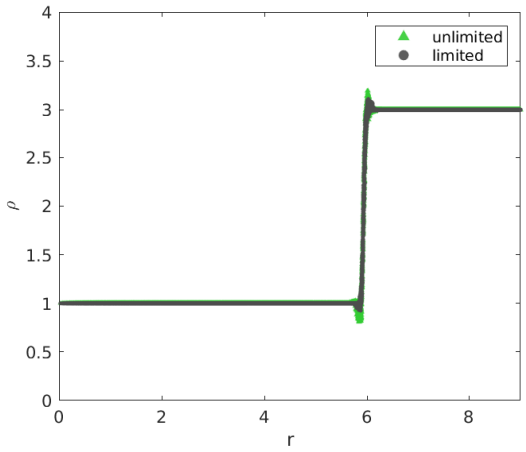
Figure 2.15: Error convergence plot for the Gaussian wave case.

region is stationary, and the flow outside is isentropic. The initial condition is given by

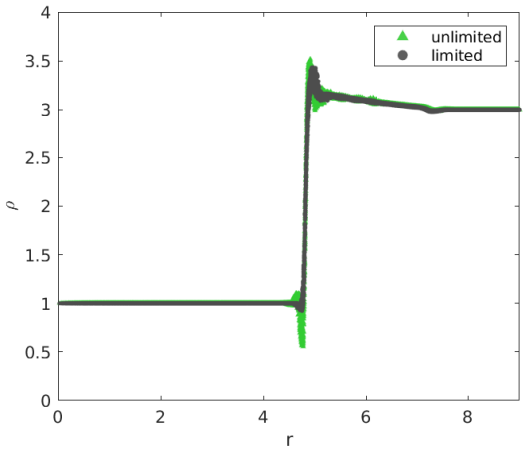
$$(\rho, v_r) = \begin{cases} (1, 0), & 0 \leq r < r_0 \\ (3, -\sqrt{8} r_0 / r), & r \geq r_0 \end{cases} \tag{2.52}$$

We run this case with a CFL number $\nu = 0.6$. The converging shock propagates inwards and grows in strength, as shown in Fig. 2.16. The limiter is able to reduce the overshoots in the solution and helps us run the simulation past the point of collapse of the shock. The moment of collapse in the solution is when the velocity at $r = 0$ changes from inwards to outwards. This time is $t = 3.74$ in our numerical solution. Theoretically, the amplitudes of the flow quantities blow up at the time of the collapse. Numerically, however, the maximum density that appears in the solution depends on the mesh size. The diverging shock shown in Fig. 2.17 propagates with diminishing strength. The contours of the density are shown in Figure 2.18 for before and after the collision. Although the shocks are sustained by only a few elements in these plots, their symmetries are well maintained.

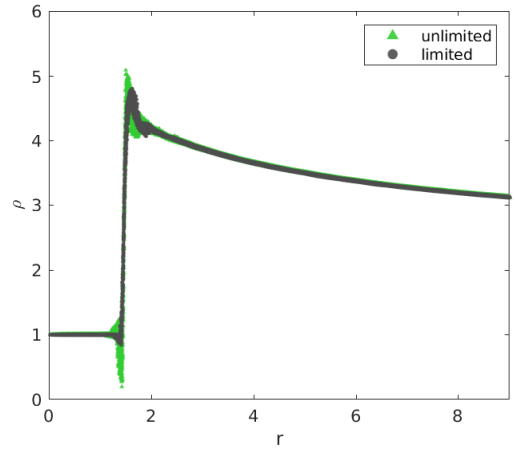
We believe that Active Flux's ability to properly resolve the shock relatively efficiently and produce an uncontaminated solution is due to its proper representation of multidimensional physics. This is showcased through a comparison with the DG method, which uses traditional Riemann solvers. We run the same test case with DG equipped with $p = 2$



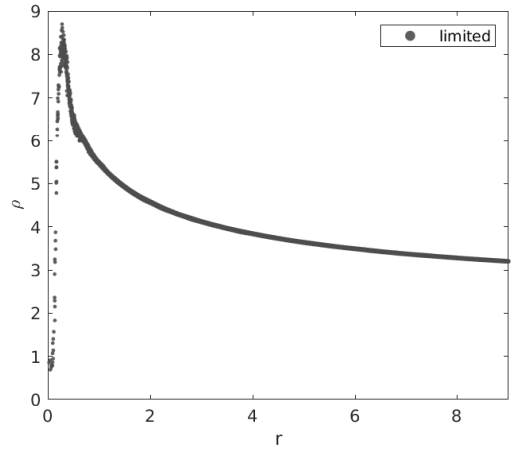
(a) $t = 0.027$



(b) $t = 0.806$

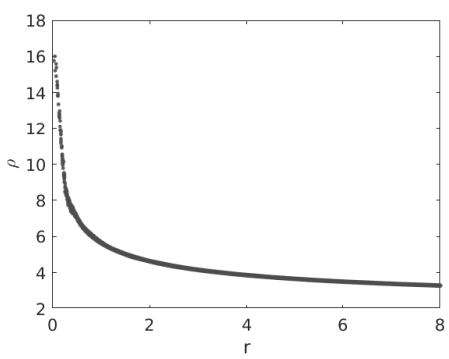


(c) $t = 3.0$

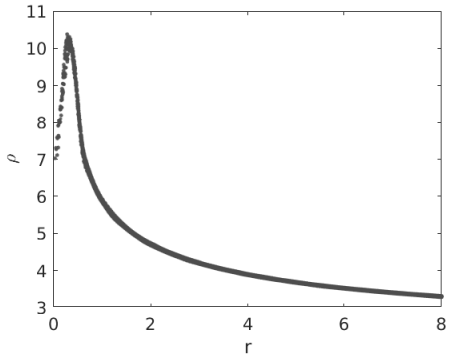


(d) $t = 3.74$

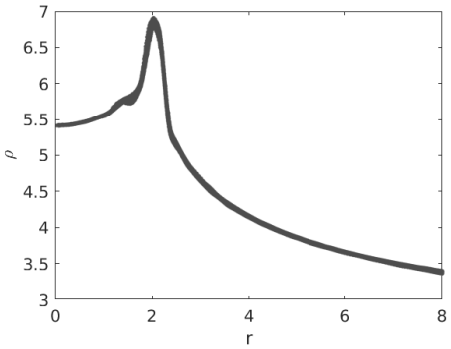
Figure 2.16: Converging shock for $h = 0.054$. Note the changes in the vertical scale.



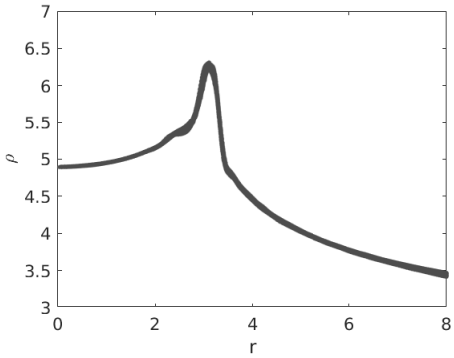
(a) $t = 3.83$



(b) $t = 3.92$

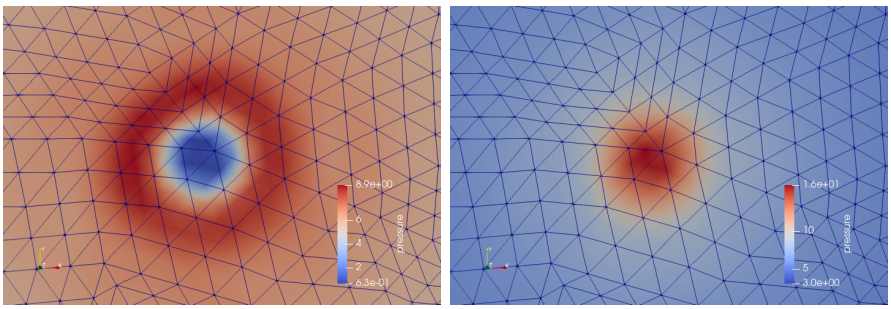


(c) $t = 4.56$



(d) $t = 5.0$

Figure 2.17: Diverging shock for $h = 0.054$. Note the changes in the vertical scale.



(a) $t = 3.74$ (b) $t = 3.83$

Figure 2.18: Density contours for the converging-diverging shock.

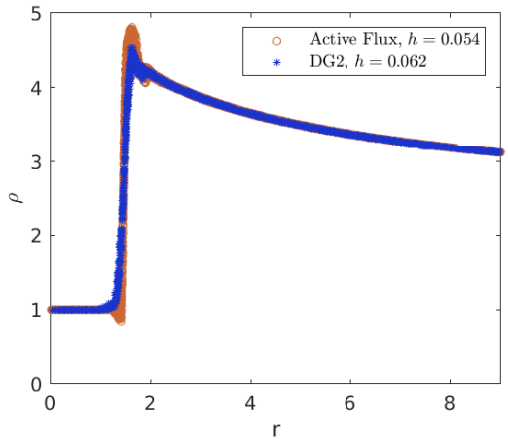


Figure 2.19: Comparison of the Active Flux and DG solutions for the strong converging shock at $t = 3.0$. DG has a CFL number close to 0.001.

solution representation, the Roe flux, the classic four-stage Runge-Kutta method, and an artificial viscosity shock-capturing approach presented in Chap. 3. The setup of this shock in this case is relatively strong, and DG exhibits instability. The maximum time step that we are able to use for our mesh size $h = 0.062$ is $\Delta t = 2 \times 10^{-4}$, which corresponds to a CFL number on the magnitude of 0.001. The solution of DG shown in Fig. 2.19 is of good quality, although it takes a significant amount of time to obtain.

If we instead set up a weaker shock with $(\rho_L, v_{r,L}) = (3, 0)$, $(\rho_R, v_{r,R}) = (5, -4)$, we are able to run DG with $\Delta t = 0.004$ for the same mesh size. However, the DG solution, in this case, is more oscillatory than Active Flux, as shown in Fig. 2.20. We believe this is because of the loss of accuracy due to the one-dimensional solution projection on the mesh edges.

2.2.3.3 Converging-Diverging Shock for the Euler Equations

The construction of an imploding shock for the Euler equations is not as obvious as the p-system. To satisfy the Rankine-Hugoniot conditions, the initial shock is designed to be

$$(\rho_L, v_{r,L}, P_L) = (1, 0, 0.685), \ (\rho_R, v_{r,R}, P_R) = (2, -0.774, 1.885). \tag{2.53}$$

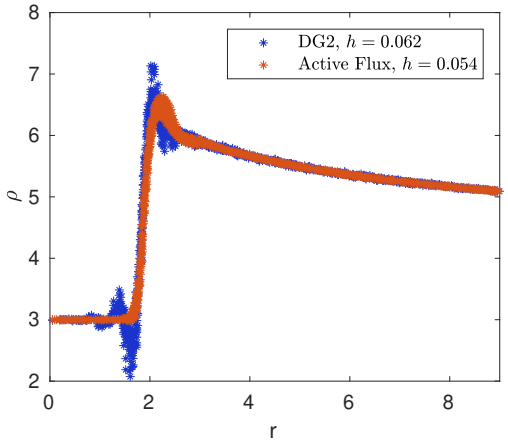


Figure 2.20: Comparison of the Active Flux and DG solutions for the weak converging shock at $t = 2.0$.

To construct the isentropic flow in the outer region, we start from

$$\frac{\partial M}{\partial r} = \frac{1}{c}\frac{\partial v_r}{\partial r} - \frac{v_r}{c^2}\frac{\partial c}{\partial r}, \tag{2.54}$$

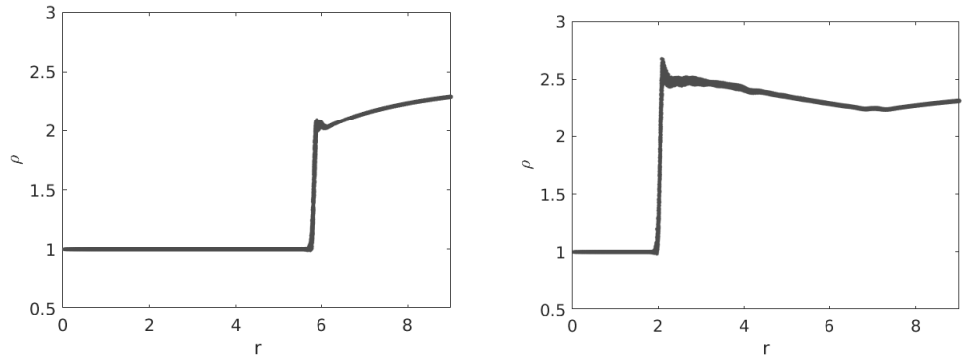
where $c = \sqrt{\frac{\gamma P}{\rho}}$ is the speed of sound, and $M = \frac{v_r}{c}$ is the Mach number. Using the Euler equations, we have

$$\begin{aligned}\frac{\partial M}{\partial r} &= -\frac{v_r}{c^2}\frac{1}{\gamma - 1}\frac{\partial c}{\partial r} - \frac{1}{c}\frac{v_r}{r} - \frac{v_r}{c^2}\frac{\partial c}{\partial r} \\ &= \frac{1}{2}\left(1 + \frac{\gamma - 1}{2}M^2\right)^{-1}\gamma M^2\frac{\partial M}{\partial r} - \frac{M}{r}\end{aligned} \tag{2.55}$$

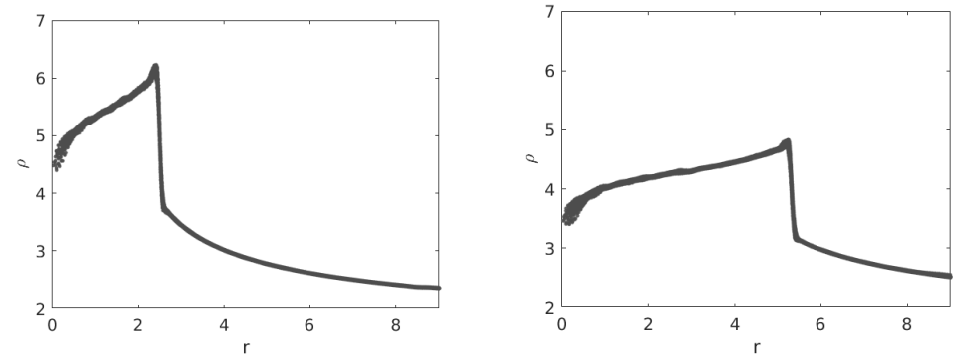
Organizing this gives us an equation to solve for the Mach number for the cylindrical symmetric isentropic flow:

$$\frac{dM}{dr} = -\frac{1 + \frac{\gamma-1}{2}M^2}{1 - \frac{M^2}{2}}\frac{M}{r}. \tag{2.56}$$

The initial condition is generated iteratively using Eqn. 2.56 starting from the shock location and the right-hand side of the condition in 2.53. The primitive variables are found with the



(a) $t = 0.1$ (b) $t = 2.39$



(c) $t = 5.0$ (d) $t = 7.0$

Figure 2.21: Density profiles of the Euler converging-diverging shock. Note the changes in the vertical scale.

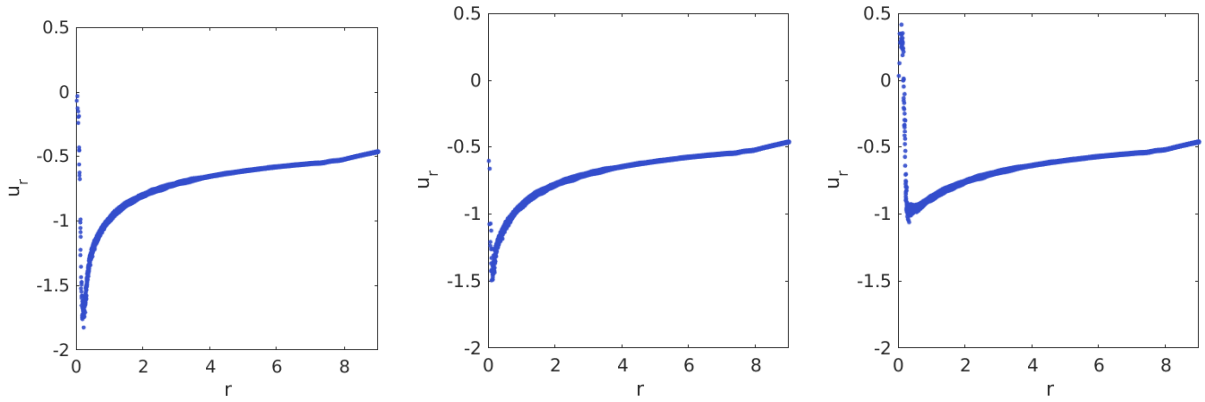
isentropic relations:

$$\rho = \rho_0 \left(1 + \frac{\gamma - 1}{2} M^2\right)^{-\frac{1}{\gamma - 1}}, \tag{2.57}$$

$$P = P_0 \left(1 + \frac{\gamma - 1}{2} M^2\right)^{-\frac{\gamma}{\gamma - 1}}, \tag{2.58}$$

where ρ_0 and P_0 are the stagnation density and pressure found from the postshock state.

The simulation is run with mesh size $h = 0.054$ and a CFL number $\nu = 0.7$. Density profiles of the shock converging ($t = 0.1$ and $t = 2.39$) and diverging ($t = 5$ and $t = 7$) are shown in Fig. 2.21. The limiter presented in section 2.2 is applied. The converging shock is stabilized and the diverging shock is captured with good quality even though the limiter is designed for acoustics systems.



(a) $t = 3.32$ (b) $t = 3.39$ (c) $t = 3.47$

Figure 2.22: Radial velocity profiles of the Euler shock reflection.

We take a close look at the shock reflection process in Fig. 2.22. The process from the onset of the reflection to the generation of the diverging shock is instant in theory and is simulated with about 20 time steps. We observe the origin ($r = 0$) becoming a sink at the onset of the reflection and a source a few time steps later. This reveals the mechanism of the shock reflection.

2.3 Efficient Implementation of Active Flux

We have introduced the construction of the third-order Active Flux method for unstructured meshes. When implementing the method into code, there are a few tricks to achieve the full potential of the efficiency that the method is designed to have. We introduce them here and present a comparison with the standard explicit Runge-Kutta DG method.

2.3.1 Half-Step Evaluation

The Simpson's quadrature in Eqn. 2.26 requires the solution for the half time step. An extra evaluation of the half-step aside from the full-step with evolution is costly and in fact unnecessary. If we write the evolution step in a general way, the solution is updated

according to

$$\mathbf{Q}^{n+1,*} = \mathbf{Q}^{n,*} + \mathcal{A}_1\mathbf{Q}^{n,*}\Delta t + \frac{1}{2}\mathcal{A}_2\mathbf{Q}^{n,*}\Delta t^2, \tag{2.59}$$

where $\mathcal{A}_1$ and $\mathcal{A}_2$ are the first and second order evolution operators, and $\mathcal{A}_1\mathbf{Q}^{n,*}$ and $\mathcal{A}_2\mathbf{Q}^{n,*}$ are estimations to $\frac{\partial \mathbf{u}}{\partial t}$ and $\frac{\partial^2 \mathbf{u}}{\partial t^2}$ at time step n. In fact, $\mathcal{A}_1\mathbf{Q}^{n,*}$ and $\mathcal{A}_2\mathbf{Q}^{n,*}$ are straightforward to identify in our implementation since the acoustics operators in Eqn. 2.14 and the advection operator in Eqn. 2.20 are evaluated as first- and second-order components. This means that the half time step is easily derived by multiplying the first-order components by $\frac{1}{2}$ and the second-order components by $\frac{1}{4}$.

2.3.2 Spherical Mean Evaluations

Consider the partially spherical integrals evaluated inside the circular sectors in an element. The acoustics operators are applied to the first-order spatial derivatives of the solution. For a parabolic solution reconstruction, the first-order spatial derivatives are linear.

Theorem:

For a linear function, the partial spherical integrals for the edge nodes are linear combinations of those for the vertex nodes.

Proof:

Consider the spherical integrals in Eqn. 2.14 applied to a linear function described by $f(x, y) = b_0 + b_x x + b_y y$, evaluated partially for angle $\sigma \in [\sigma_1, \sigma_2]$. For a location (x_0, y_0), we

have

$$
\begin{aligned}
M\{f\}_{(\sigma_1,\sigma_2)}(x_0,y_0,r) =& \frac{\sigma_2-\sigma_1}{2\pi} f(x_0,y_0) + \frac{r}{8}(\sin\sigma_2 - \sin\sigma_1)b_x - \frac{r}{8}(\cos\sigma_2 - \cos\sigma_1)b_y, \\
\vec{I}\{f\}_{(\sigma_1,\sigma_2)}(x_0,y_0,r) =& \frac{\sigma_2-\sigma_1}{2\pi}\begin{bmatrix} b_x \\ b_y \end{bmatrix} + \frac{f(x_0,y_0)}{2r}\begin{bmatrix} \sin\sigma_2 - \sin\sigma_1 \\ -\cos\sigma_2 + \cos\sigma_1 \end{bmatrix} + \\
& \begin{bmatrix} \frac{1}{4\pi}(\sin 2\sigma_2 - \sin 2\sigma_1)b_x - \frac{1}{4\pi}(\cos 2\sigma_2 - \cos 2\sigma_1)b_y \\ -\frac{1}{4\pi}(\cos 2\sigma_2 - \cos 2\sigma_1)b_x - \frac{1}{4\pi}(\sin 2\sigma_2 - \sin 2\sigma_1)b_y \end{bmatrix}
\end{aligned} \tag{2.60}
$$

Two important relationships that we can derive from the above are, for translation,

$$
M\{f\}_{(\sigma_1,\sigma_2)}(x_1,y_1,r) = M\{f\}_{(\sigma_1,\sigma_2)}(x_0,y_0,r) + (f(x_1,y_1) - f(x_0,y_0))\frac{\sigma_2-\sigma_1}{2\pi}, \tag{2.61}
$$

$$
\vec{I}\{f\}_{(\sigma_1,\sigma_2)}(x_1,y_1,r) = \vec{I}\{f\}_{(\sigma_1,\sigma_2)}(x_0,y_0,r) + \frac{(f(x_1,y_1) - f(x_0,y_0))}{2r}\begin{bmatrix} \sin\sigma_2 - \sin\sigma_1 \\ -\cos\sigma_2 + \cos\sigma_1 \end{bmatrix}, \tag{2.62}
$$

and for rotation,

$$
M\{f\}_{(\sigma_1+\pi,\sigma_2+\pi)}(x_0,y_0,r) = -M\{f\}_{(\sigma_1,\sigma_2)}(x_0,y_0,r) + f(x_0,y_0)\frac{\sigma_2-\sigma_1}{\pi}. \tag{2.63}
$$

$$
\vec{I}\{f\}_{(\sigma_1+\pi,\sigma_2+\pi)}(x_0,y_0,r) = \vec{I}\{f\}_{(\sigma_1,\sigma_2)}(x_0,y_0,r) - \frac{f(x_0,y_0)}{r}\begin{bmatrix} \sin\sigma_2 - \sin\sigma_1 \\ -\cos\sigma_2 + \cos\sigma_1 \end{bmatrix}. \tag{2.64}
$$

Consider an element illustrated in Fig. 2.23. We use M_n to denote the partial spherical integral for the circular sector at node n, where $1 \le n \le 6$, f_n is the function value for the node, and $\Delta\theta_n$ is the angle of the circular sector. We assume C^1 solution reconstruction in the plotted region for the purpose of the proof. The area denoted by n' is translated from the circular sector at node n. From Eqn. 2.61, we know

$$
M_n - f_n\frac{\Delta\theta_n}{2\pi} = M_{n'} - f_1\frac{\Delta\theta_n}{2\pi}, \tag{2.65}
$$

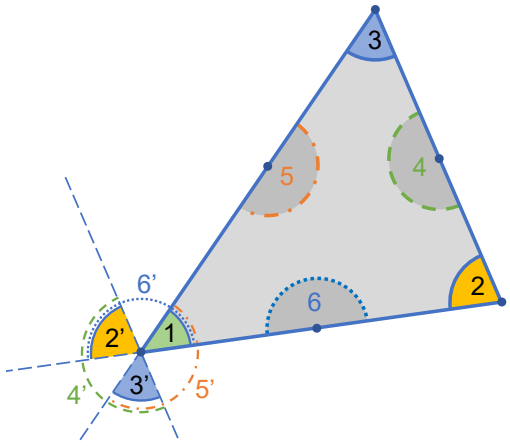


Figure 2.23: Partial spherical integrals inside an element.

and from Eqn. 2.63,

$$\begin{aligned} M_{4'} &= M_{2'} + M_{3'} + (f_1 \frac{\Delta\theta_1}{\pi} - M_1) \\ &= M_2 + M_3 - M_1 + f_1 \frac{\Delta\theta_1}{\pi} + (f_1 - f_2)\frac{\Delta\theta_2}{2\pi} + (f_1 - f_3)\frac{\Delta\theta_3}{2\pi} \end{aligned} \tag{2.66}$$

Meanwhile, M_4 and $M_{4'}$ are related by

$$M_4 = M_{4'} + \frac{1}{2}\left(\frac{f_2 + f_3}{2} - f_1\right). \tag{2.67}$$

Finally, we have

$$\begin{aligned} M_4 &= M_2 + M_3 - M_1 + f_2\left(\frac{1}{4} - \frac{\Delta\theta_2}{2\pi}\right) + f_3\left(\frac{1}{4} - \frac{\Delta\theta_3}{2\pi}\right) + f_1\frac{\Delta\theta_1}{2\pi} \\ &= M_2 + M_3 - M_1 + f_2\left(\frac{1}{4} - \frac{\Delta\theta_2}{2\pi}\right) + f_3\left(\frac{1}{4} - \frac{\Delta\theta_3}{2\pi}\right) + f_1\left(\frac{1}{2} - \frac{\Delta\theta_2}{2\pi} - \frac{\Delta\theta_3}{2\pi}\right) \end{aligned} \tag{2.68}$$

This equation can be adopted to find the other two edge integrals from the nodal integrals via mutations of the indices. We also omit the treatment of $\vec{I}$ since it is similar to the treatment of M. □

Because of this property, the partial spherical integrals inside each element are only evaluated for the vertex nodes, and the edge integrals are found with the nodal values

through cheap calculations.

2.3.3 Efficiency Comparison with DG

With these improvements to the naive implementation, we test the efficiency of our code with an Euler case of a moving vortex in the domain $[-10, 10] \times [-10, 10]$. The initial condition is given by

$$\begin{aligned} \vec{v}^0 &= \vec{v}_\infty + \vec{\tau}\frac{\epsilon}{2\pi}e^{\frac{(1-r^2)}{2}} \\ T^0 = T_\infty - \frac{(\gamma-1)\epsilon^2}{8\gamma\pi^2}e^{1-r^2}&, P^0 = (\rho^0)^\gamma \end{aligned} \tag{2.69}$$

where $r = \sqrt{x^2+y^2}$, $\tau = (-y, x)$, $\gamma = 1.4$, $\vec{v}_\infty = 1$, and $T_\infty = 1$. The L_2 error of the solution in Eqn. 2.44 is measured at $t = 20$ when the vortex has returned to its original location for the first time. The efficient implementation proposed is around 40% faster than the naive implementation without sacrificing the accuracy of the solution.

The performance of the Active Flux method is compared to DG with solution reconstruction order $p = 1$ and $p = 2$. The L_2 errors shown in Fig. 2.25 for DG and Active Flux are measured as

$$E_{AF} = \sqrt{\frac{\sum_{i=1}^{m}\sum_{j=1}^{N_n}(U_{i,j} - u_i^0(\vec{x}_j))^2}{N_n}}, \tag{2.70}$$

$$E_{DG} = \sqrt{\frac{\sum_{i=1}^{m}\int_\Omega (u_i - u_i^0)^2 \mathrm{d}\Omega}{\int_\Omega \mathrm{d}\Omega}}. \tag{2.71}$$

The DG method is advanced in time with a three-stage third-order Runge-Kutta scheme. The CFL number for both methods is $\nu = 0.7$. The time steps for Active Flux and DG are

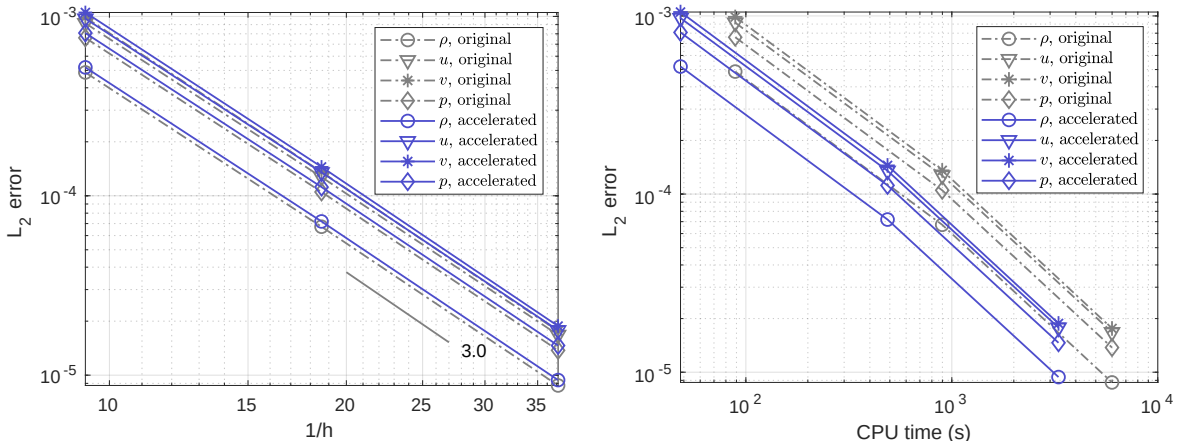


Figure 2.24: Error convergence of the moving vortex case.

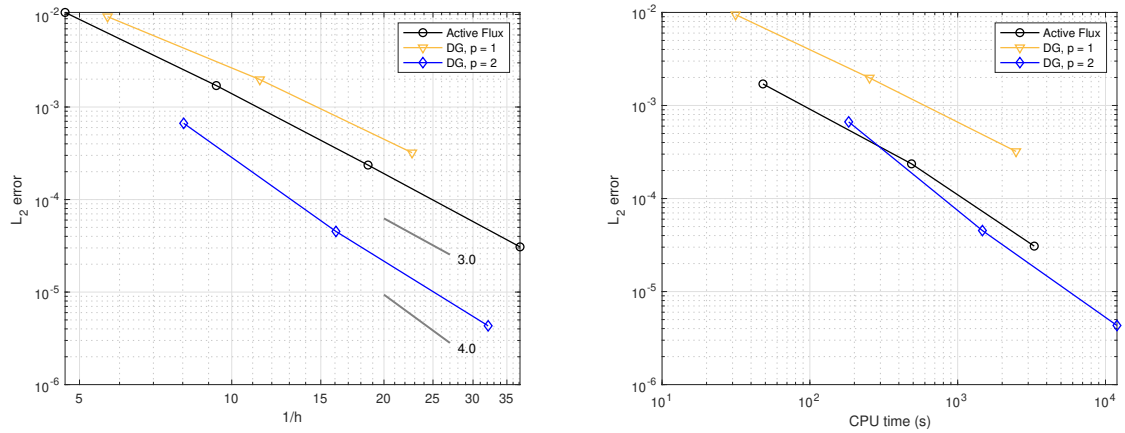


Figure 2.25: Comparison of the solution efficiency between Active Flux and DG.

calculated according to their time step restrictions as

$$\Delta t_{\mathrm{AF}} = \frac{\nu\lambda}{s^{\max}}, \tag{2.72}$$

$$\Delta t_{\mathrm{DG}} = \frac{\nu\lambda}{(2p+1)s^{\max}}, \tag{2.73}$$

where $s^{\max}$ is the maximum wave speed in the domain, discussed below, and λ is defined in Eqn. 2.28. The maximum wave speed is calculated as $|\vec{u}| + c$ for the Euler equations, where $c = \sqrt{\frac{\gamma P}{\rho}}$ is the speed of sound. Note that the maximum time step for DG is $\frac{1}{5}$ of the maximum time step for Active Flux.

The DG results exhibit superconvergence properties for this test case. For the same DOFs, Active Flux has slightly lower errors compared to the DG method with $p = 1$. However, if we compare the computation time spent for the same accuracy, Active Flux is comparable to DG with $p = 2$ for the same number of DOFs.

CHAPTER III

Artificial Viscosity Shock-capturing

3.1 Discontinuous Galerkin and Hybrid Discontinuous Galerkin

We consider two discretizations for the conservation law in Eqn. 1.2: discontinuous Galerkin (DG) 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: $\mathbf{u}_h \in \boldsymbol{\mathcal{V}}_h = [\mathcal{V}_h]^s$, where $\mathcal{V}_h = \{u \in L_2(\Omega) : u|_{\Omega_e} \in \mathcal{P}^p \ \forall \Omega_e \in T_h\}$, and $\mathcal{P}^p$ denotes polynomials of order p on the reference space of element Ω_e.

3.1.1 Discontinuous Galerkin (DG)

The DG weak form of Eqn. 1.2 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:

$$\begin{aligned}\int_{\Omega_e} \mathbf{w}_h^T \frac{\partial \mathbf{u}_h}{\partial t}\, d\Omega - \int_{\Omega_e} \nabla \mathbf{w}_h^T \cdot \vec{\mathbf{H}}(\mathbf{u}_h, \nabla \mathbf{u}_h)\, d\Omega + \int_{\partial\Omega_e} \mathbf{w}_h^T \widehat{\mathbf{H}} \cdot \vec{n}\, ds \\ - \int_{\partial\Omega_e} \partial_i \mathbf{w}_h^{+T} \mathbf{K}_{ij}^{+} \left(\mathbf{u}_h^{+} - \widehat{\mathbf{u}}_h\right) n_j\, ds \;\; = \;\; 0 \quad \forall \mathbf{w}_h \in \boldsymbol{\mathcal{V}}_h,\end{aligned} \tag{3.1}$$

where $(\cdot)^T$ denotes transpose, $\mathbf{K}_{ij}^+$ is the diffusivity tensor, $\widehat{\mathbf{u}}_h = (\mathbf{u}_h^+ + \mathbf{u}_h^-)/2$, and on the element boundary $\partial\Omega_e$, $(\cdot)^+, (\cdot)^-$ 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, $\widehat{\mathbf{H}} \cdot \vec{n}$, we use the Roe-approximate Riemann solver [61] unless specified otherwise and the second form of Bassi and Rebay (BR2) [62] for the viscous flux. Choosing a basis for the test and trial spaces yields the semi-discrete 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, $\frac{\partial \mathbf{u}}{\partial t} = 0$, we solve the discretized system of non-linear 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 [63]. The update from $\mathbf{U}^n$ to $\mathbf{U}^{n+1}$ proceeds through a solution of n_{stage} systems:

$$\mathbf{M}(\mathbf{U}_i - \mathbf{U}^n) + \Delta t a_{ii}\mathbf{R}(\mathbf{U}_i) + \Delta t \sum_{j=1}^{i-1} a_{ij}\mathbf{R}(\mathbf{U}_j) = \mathbf{0}, \quad i = 1, \cdots, n_{\text{stage}} \tag{3.2}$$

$$\mathbf{U}^{n+1} = \mathbf{U}^n + \Delta t \sum_{j=1}^{n_{\text{stage}}} b_j \mathbf{R}(\mathbf{U}_j), \tag{3.3}$$

where

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

and $\alpha = 0.435866521508459$. Eqn. 3.2 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.

3.1.2 Hybridized Discontinuous Galerkin (HDG)

The starting point for HDG discretization is the conversion of Eqn. 1.2 to a system of first-order equations,

$$\vec{\mathbf{q}} - \nabla \mathbf{u} = \vec{\mathbf{0}}, \tag{3.4}$$

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot \vec{\mathbf{H}}(\mathbf{u}, \vec{\mathbf{q}}) = \mathbf{0}, \tag{3.5}$$

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

$$\int_{\Omega_e} \vec{\mathbf{v}}_h^T \cdot \vec{\mathbf{q}}_h \, d\Omega + \int_{\Omega_e} \nabla \cdot \vec{\mathbf{v}}_h^T \mathbf{u}_h \, d\Omega - \int_{\partial\Omega_e} \vec{\mathbf{v}}_h^T \cdot \vec{n} \, \widehat{\mathbf{u}}_h \, ds = 0 \quad \forall \vec{\mathbf{v}}_h \in [\boldsymbol{\mathcal{V}}_h]^{\text{dim}}, \tag{3.6}$$

$$\int_{\Omega_e} \mathbf{w}_h^T \frac{\partial \mathbf{u}_h}{\partial t} \, d\Omega - \int_{\Omega_e} \nabla \mathbf{w}_h^T \cdot \vec{\mathbf{H}} \, d\Omega + \int_{\partial\Omega_e} \mathbf{w}_h^T \widehat{\mathbf{H}} \cdot \vec{n} \, ds = 0 \quad \forall \mathbf{w}_h \in \boldsymbol{\mathcal{V}}_h, \tag{3.7}$$

where $\widehat{\mathbf{u}}_h$ 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,

$$\int_{\sigma_f} \boldsymbol{\mu}_h^T \left\{ \widehat{\mathbf{H}} \cdot \vec{n} \big|_L + \widehat{\mathbf{H}} \cdot \vec{n} \big|_R \right\} ds = 0 \qquad \forall \boldsymbol{\mu}_h \in \boldsymbol{\mathcal{W}}_h, \tag{3.8}$$

where $\boldsymbol{\mathcal{W}}_h$ denotes the order-p approximation space on the faces $\sigma_f \in F_h$ of the mesh: $\boldsymbol{\mathcal{W}}_h = [\mathcal{W}_h]^s$, where $\mathcal{W}_h = \left\{ u \in L_2(\sigma_f) : u|_{\sigma_f} \in \mathcal{P}^p \; \forall \sigma_f \in F_h \right\}$, 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 [64], but it yields well-defined blocks in the Jacobian matrix that simplify preconditioning.

The fluxes in Eqn. 3.7 are one-sided, meaning that they depend only on the state and

gradient inside the element and the face state,

$$\widehat{\mathbf{H}} \cdot \vec{n} = \vec{\mathbf{H}}(\hat{\mathbf{u}}_h, \vec{\mathbf{q}}_h) \cdot \vec{n} + \boldsymbol{\tau}(\hat{\mathbf{u}}_h, \mathbf{u}_h, \vec{n}), \qquad \boldsymbol{\tau} = \left| \frac{\partial}{\partial \mathbf{u}} (\widehat{\mathbf{F}} \cdot \vec{n}) \right|_{\mathbf{u}_h^*} (\mathbf{u}_h - \hat{\mathbf{u}}_h) + \eta \vec{\boldsymbol{\delta}}_h \cdot \vec{n}. \tag{3.9}$$

Note that $\boldsymbol{\tau}$ consists of a convective stabilization computed about the Roe-average state, $\mathbf{u}_h^*$, and a BR2 viscous stabilization [65], where η is set to a value that is at least the number of faces and $\vec{\boldsymbol{\delta}}_h$ 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 Eqns. 3.6, 3.7, 3.8 gives a nonlinear system of ordinary differential equations,

$$\begin{aligned} \mathbf{R}^Q &= \mathbf{0}, \\ \mathbf{M}^U \frac{d\mathbf{U}}{dt} + \mathbf{R}^U &= \mathbf{0}, \\ \mathbf{R}^\Lambda &= \mathbf{0}, \end{aligned} \tag{3.10}$$

where $\mathbf{M}^U$ is the mass matrix. For a steady case, the ODEs reduce to a non-linear system of equations

$$\mathbf{R}^Q = \mathbf{0}, \quad \mathbf{R}^U = \mathbf{0}, \quad \mathbf{R}^\Lambda = \mathbf{0}, \tag{3.11}$$

with the Newton update system

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} \begin{bmatrix} \Delta\mathbf{Q} \\ \Delta\mathbf{U} \\ \Delta\boldsymbol{\Lambda} \end{bmatrix} + \begin{bmatrix} \mathbf{R}^Q \\ \mathbf{R}^U \\ \mathbf{R}^\Lambda \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \tag{3.12}$$

where $\mathbf{Q}$, $\mathbf{U}$, and $\boldsymbol{\Lambda}$ are the discrete unknowns in the approximation of $\vec{\mathbf{q}}$, $\mathbf{u}$, 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 element-wise 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,

$$\underbrace{\left(\mathbf{D}-\mathbf{C}\mathbf{A}^{-1}\mathbf{B}\right)}_{\boldsymbol{\mathcal{K}}}\Delta\boldsymbol{\Lambda}+\left(\mathbf{R}^{\Lambda}-\mathbf{C}\mathbf{A}^{-1}\left[\mathbf{R}^{Q};\mathbf{R}^{U}\right]\right)=\mathbf{0}. \tag{3.13}$$

Solving this set of equations constitutes the global solution of the problem. Following the global solution for $\Delta\boldsymbol{\Lambda}$, an element-local back-substitution 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:

$$\begin{aligned}
\mathbf{R}^{Q}(\mathbf{Q}_i,\mathbf{U}_i,\boldsymbol{\Lambda}_i)&=\mathbf{0},\\
\mathbf{M}^{U}(\mathbf{U}_i-\mathbf{U}^n)+\Delta t a_{ii}\mathbf{R}^{U}(\mathbf{Q}_i,\mathbf{U}_i,\boldsymbol{\Lambda}_i)+\Delta t\sum_{j=1}^{i-1}a_{ij}\mathbf{R}^{U}(\mathbf{Q}_j,\mathbf{U}_j,\boldsymbol{\Lambda}_j)&=\mathbf{0},\\
\mathbf{R}^{\Lambda}(\mathbf{Q}_i,\mathbf{U}_i,\boldsymbol{\Lambda}_i)&=\mathbf{0}, i=1,\cdots,n_{\text{stage}}\\
\mathbf{U}^{n+1}=\mathbf{U}^n+\Delta t\sum_{j=1}^{n_{\text{stage}}}b_j\mathbf{R}^{U}(\mathbf{Q}_j,\mathbf{U}_j,\boldsymbol{\Lambda}_j).&
\end{aligned} \tag{3.14}$$

For the Newton iterations used to solve Eqn. 3.14 at each stage, the Jacobian matrix needs to be modified slightly from the steady case, but this does not affect the sparsity of $\boldsymbol{\mathcal{K}}$.

3.2 Shock Capturing

3.2.1 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, written in index notation,

$$\partial_t u_k+\partial_i F_{ik}=\partial_i(K_{ijkl}\partial_j u_l), \tag{3.15}$$

where k, l index the state rank, i, 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,

$$K^{\text{stab}}_{ijkl}(\vec{x}) = T_{kl}\epsilon_{ij}(\vec{x}), \tag{3.16}$$

where $T_{kl} = \frac{\partial \tilde{u}_k}{\partial u_l}$ and $\tilde{u}_k = [\rho, \rho\vec{u}, \rho H]$ is a modified state vector that makes the stabilization term preserve total enthalpy [27], and ϵ_{ij} is an artificial viscosity tensor field. Numerical dissipation is added through both the convective and diffusive fluxes. The Roe flux function [61] does not preserve total enthalpy, and hence we also present test results for the van Leer–Hänel flux function [66], which is designed to preserve the total enthalpy.

The artificial viscosity tensor field is found as

$$\epsilon_{ij}(\vec{x}) = C\frac{h_{ij}}{\bar{h}}\epsilon_0(\vec{x}), \tag{3.17}$$

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

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 [67, 68], i.e., $\mathcal{M}_e \in Sym^+_d$ such that

$$\sqrt{v_i\mathcal{M}_{e,ij}v_j} = 1, \ \forall v_i \in \text{Edges}(e). \tag{3.18}$$

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

$$\bar{h}_e = [\det(\mathcal{M}_e)]^{-1/2\dim}. \tag{3.19}$$

As the metric eigenvalues have units of inverse square distance, the smoothly-varying anisotropy field, $h_{ij}/\bar{h}$, is obtained by dividing the inverse square root of the metric by $\bar{h}_e$, i.e., taking $\frac{1}{\bar{h}_e}\mathcal{M}_e^{-1/2}$, 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

$$\epsilon_{0e} = \frac{\lambda_{\max}\bar{h}_e}{p}\bar{S}_e, \tag{3.20}$$

where $\bar{S}_e$ is a smoothness indicator computed from the states, which is either the resolution or the variation indicator in this paper, $\lambda_{\max}$ 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:

$$\tilde{\epsilon}_e^{k+1} = (1-c_s)\epsilon_{0e} + \frac{c_s}{n_e}\sum_{t\in\mathcal{N}(e)}\tilde{\epsilon}_t^k, \quad k = 0, \cdots, n_{\text{smooth}} - 1 \tag{3.21}$$

$$\epsilon_e = \tilde{\epsilon}_e^{n_{\text{smooth}}} \tag{3.22}$$

where $\mathcal{N}(e)$ denotes the neighboring elements of an 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. c_s is chosen to be 1.0 for the transonic cases and 0.9 for the hypersonic cases. $n_{\text{smooth}} = 10$ 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 section 3.5 to clip away spurious small values generated by the Laplace smoothing. We

adopt the smooth filter definition from Barter and Darmofal [27], except that we apply it before making the artificial viscosity continuous:

$$\hat{\epsilon}_e(\epsilon_e) = \begin{cases} 0, & \epsilon_e \leq \theta_L \\ \frac{1}{2}\theta_H \left(\sin\left[\pi \left(\frac{\epsilon_e - \theta_L}{\theta_H - \theta_L} - \frac{1}{2} \right) \right] + 1 \right), & \theta_L < \epsilon_e < \theta_H \\ \theta_H & \epsilon_e \geq \theta_H \end{cases}, \tag{3.23}$$

where θ_L and θ_H are the maximum and minimum values that $\hat{\epsilon}_e$ varies in between, and $\theta_H = \lambda_{\max} \bar{h}_e / p$ 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, as many iterations are expended prior to the Newton convergence "bucket." The lack of an exact linearization, however, preserves the compact stencil, which is crucial for the efficiency of the solver.

3.2.2 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.

3.2.2.1 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 pth order quantity and its least-squares projection onto the space of $(p-1)$th order polynomials:

$$\bar{S}_e = \frac{f^2}{f+1}, \quad f = \frac{S_e}{S_0}, \quad S_e = \frac{\int_{\Omega_e}(u-\tilde{u})^2\, d\Omega}{\int_{\Omega_e} u^2\, d\Omega}, \tag{3.24}$$

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, c_p in a nonlinear mapping as well as the coefficient C in Eqn. (3.17) 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 since 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.

3.2.2.2 Variation Indicator

The variation indicator is based on the intra-element variation of a selected quantity:

$$\bar{S}_e = \begin{cases} 0, & S_e < S^* - \Delta S \\ \frac{S_e}{2}\left[1 + \sin\left(\frac{\pi}{2\Delta S}(S_e - S^*)\right)\right], & S^* - \Delta S \le S_e \le S^* + \Delta S\,, \\ S_e, & S_e > S^* + \Delta S \end{cases} \tag{3.25}$$

$$S_e = \left[\frac{1}{|\Omega_e|}\int_{\Omega_e}\left(\frac{u}{\bar{u}_e} - 1\right)^2 d\Omega\right]^{1/2} \tag{3.26}$$

where u is the chosen scalar for measuring regularity, and $\bar{u}_e = \frac{1}{|\Omega_e|}\int_{\Omega_e} u d\Omega$, S^* and ΔS are user-defined parameters. The smooth scaling was presented by Persson and Peraire [26] and

used in combination with the variation indicator by Ching et al. [28]. 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.

3.3 Mesh Adaptation

3.3.1 Adjoint Solution

The mesh adaptation process used in this thesis is driven by the estimation of the output error. We use discrete adjoint solutions for the error estimation. Let $\mathbf{U}_h \in \mathbb{R}^{N_h}$ and $\mathbf{R}_h \in \mathbb{R}^{N_h}$ be the vector of the unknowns and the residuals respectively. The output for this discretization is denoted as J_h. The adjoint vector, $\mathbf{\Psi}_h$ is the sensitivity of J_h with respect to the residual. An infinitesimal perturbation $\delta\mathbf{U}_h$ and the perturbation induced in the residuals $\delta\mathbf{R}_h$ satisfy

$$\frac{\partial \mathbf{R}_h}{\partial \mathbf{U}_h}\delta\mathbf{U}_h + \delta\mathbf{R}_h = 0. \tag{3.27}$$

The increment in the output is

$$\delta J_h = J_h(\mathbf{U}_h + \delta\mathbf{U}_h) - J_h(\mathbf{U}_h) = \mathbf{\Psi}_h^T \delta\mathbf{R}_h. \tag{3.28}$$

Assuming the output is differentiable, we have

$$\delta J_h = \frac{\partial J_h}{\partial \mathbf{U}_h}\delta\mathbf{U}_h = -\mathbf{\Psi}_h^T \frac{\partial \mathbf{R}_h}{\partial \mathbf{U}_h}\delta\mathbf{U}_h. \tag{3.29}$$

This gives us the equation for the discrete adjoint:

$$\left(\frac{\partial \mathbf{R}_h}{\partial \mathbf{U}_h}\right)^T \mathbf{\Psi}_h + \left(\frac{\partial J_h}{\partial \mathbf{U}_h}\right)^T = 0. \tag{3.30}$$

For HDG, we have three sets of unknowns. Eqn. 3.30 becomes

$$\begin{bmatrix} \mathbf{A}_h^T & \mathbf{C}_h^T \\ \mathbf{B}_h^T & \mathbf{D}_h^T \end{bmatrix} \begin{bmatrix} \boldsymbol{\Psi}_h^Q \\ \boldsymbol{\Psi}_h^U \\ \boldsymbol{\Psi}_h^\Lambda \end{bmatrix} + \begin{bmatrix} \frac{\partial J_h}{\partial \mathbf{Q}_h}^T \\ \frac{\partial J_h}{\partial \mathbf{U}_h}^T \\ \frac{\partial J_h}{\partial \boldsymbol{\Lambda}_h}^T \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \tag{3.31}$$

Similar to the forward solution for HDG, this is solved via static condensation. The face DOFs are solved with

$$\underbrace{\left(\mathbf{D}_h^T - \mathbf{B}_h^T \mathbf{A}_h^{-T} \mathbf{C}_h^T\right)}_{\boldsymbol{\kappa}_h^T} \boldsymbol{\Psi}_h^\Lambda + \left(\frac{\partial J_h}{\partial \boldsymbol{\Lambda}_h}^T - \mathbf{B}_h^T \mathbf{A}_h^{-T} \left[\frac{\partial J_h}{\partial \mathbf{Q}_h}^T ; \frac{\partial J_h}{\partial \mathbf{U}_h}^T \right] \right) = \mathbf{0}. \tag{3.32}$$

3.3.2 Output Error Estimation

Let H denote the current, "coarse", approximation space, and h a fine space obtained by increasing the polynomial order by 1 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 [69, 70]. With the adjoint solutions, the output error estimate for DG reads

$$J_h(\mathbf{U}_h^H) - J_h(\mathbf{U}_h) \approx -\delta \boldsymbol{\Psi}_h^T \mathbf{R}_h(\mathbf{U}_h^H), \tag{3.33}$$

where $\mathbf{U}_h^H$ is the coarse state prolongated into the fine space, and $\delta\boldsymbol{\Psi}_h$ is obtained by subtracting the coarse-space adjoint from the fine-space adjoint. The subtraction is done to remove the effect of the numerical errors on the coarse mesh that rise from the solution convergence and focus on the discretization errors. For HDG, the output error estimate includes terms associated with the gradient and weak flux continuity equations,

$$J_h(\mathbf{U}_h^H) - J_h(\mathbf{U}_h) \approx -(\delta \boldsymbol{\Psi}_h^Q)^T \mathbf{R}_h^Q - (\delta \boldsymbol{\Psi}_h^U)^T \mathbf{R}_h^U - (\delta \boldsymbol{\Psi}_h^\Lambda)^T \mathbf{R}_h^\Lambda. \tag{3.34}$$

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

$$\varepsilon_e \equiv \left| \delta \mathbf{\Psi}_{h,e}^T \mathbf{R}_{h,e}(\mathbf{U}_h^H) \right|. \tag{3.35}$$

The HDG error indicators ε_e^Q and ε_e^U can be found in a similar way, and the indicator associated be element face DOFs, ε_e^Λ, is found by distributing the face error estimate equally to the adjacent elements.

We also tested the entropy variables in place of the output adjoint for the Euler equations [71]. They are defined by $\mathbf{v} \equiv \partial U / \partial \mathbf{u}$, where U is the entropy function chosen as

$$U = -\frac{\rho(\ln P - \gamma \ln \rho)}{\gamma - 1}. \tag{3.36}$$

3.3.3 Mesh Optimization through Error Sampling and Synthesis (MOESS)

The MOESS algorithm used in this thesis was proposed by Fidkowski [68], which built on earlier work of Yano [67].

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 $\mathcal{M}(\vec{x}) \in \mathbb{R}^{\text{dim}\times\text{dim}}$. The idea of the MOESS algorithm is to optimize the change in the current, mesh-implied metric $\mathcal{M}_0(\vec{x})$, through a step matrix $\mathcal{S} \in \mathbb{R}^{\text{dim}\times\text{dim}}$ as,

$$\mathcal{M} = \mathcal{M}_0^{\frac{1}{2}} \exp(\mathcal{S}) \mathcal{M}_0^{\frac{1}{2}}, \tag{3.37}$$

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

The step matrix field $\mathcal{S}(\vec{x})$ is represented by values at the mesh vertices, $\mathcal{S}_v$, in the implementation of MOESS. The optimization process in MOESS is iterative. In each iteration, an error reduction to cost introduction ratio is calculated at each vertex, the trace of the step matrices is 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, $\frac{\partial \varepsilon}{\partial \mathcal{S}_v}$ and $\frac{\partial \mathcal{C}}{\partial \mathcal{S}_v}$, 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

$$\varepsilon_e = \varepsilon_{e0} e^{\mathrm{tr}(\mathcal{R}_e \mathcal{S}_e)}, \tag{3.38}$$

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

$$\varepsilon_e^U = \varepsilon_{e0}^U e^{\mathrm{tr}(\mathcal{R}_e^U \mathcal{S}_e)},\ \varepsilon_e^Q = \varepsilon_{e0}^Q e^{\mathrm{tr}(\mathcal{R}_e^Q \mathcal{S}_e)},\ \varepsilon_e^\Lambda = \varepsilon_{e0}^\Lambda e^{\mathrm{tr}(\mathcal{R}_e^\Lambda \mathcal{S}_e)}, \tag{3.39}$$

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

$$\mathcal{C}_e = \mathcal{C}_{e0} e^{\frac{1}{2}\mathrm{tr}(\mathcal{S}_e)}, \tag{3.40}$$

where $\mathcal{C}_{e0}$ is the current cost on element e, prior to 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 Bi-dimensional Anisotropic Mesh Generator (BAMG) [72] 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.

3.3.4 Error Sampling

The error rate tensors in Eqn. 3.38 and Eqn. 3.39 are estimated by sampling a few refinements for each element. These are the divisions of the original triangle into two or four equal-sized triangles. The error indicator after the refinement is

$$\varepsilon_{ei} = \varepsilon_{e0} - \Delta\varepsilon_{ei}. \tag{3.41}$$

The difference is estimated as

$$\Delta\varepsilon_{ei} = \left| \boldsymbol{\Psi}_{h,e}^{H_i\,T} \mathcal{R}_{h,e}(\mathbf{U}_h^H) \right|, \tag{3.42}$$

where $\boldsymbol{\Psi}_h^{H_i}$ is the adjoint on refined space, $\boldsymbol{\Psi}_{H_i}$, projected into the fine space of order $p+1$. In practice, $\boldsymbol{\Psi}_{H_i}$ is obtained by projecting the fine space adjoint to the refined space H_i. For HDG, the change in the error estimate for the DOFs on each face needs to be distributed equally to its neighboring elements.

The error rate tensor is then solved by minimizing

$$\sum_i \left[\log \frac{\varepsilon_{ei}}{\varepsilon_{e0}} - \mathrm{tr}(\mathcal{R}_e \mathcal{S}_{ei}) \right]^2, \tag{3.43}$$

where $\mathcal{S}_{ei}$ is the step matrix for refinement option i.

3.4 One-dimensional test cases

3.4.1 Steady Linear Advection

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

$$a\frac{du}{dx} = f. \tag{3.44}$$

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

$$u(x) = \frac{\tanh(256(x - 0.4)) + 3}{2}, \tag{3.45}$$

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

The DG solutions of different polynomial orders on a uniform mesh are shown in Fig. 3.1. 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 c_0 and c_p in Eqn. 3.24 to further reduce the oscillations, but this comes at the cost of the sharpness of the discontinuity. A trade-off needs to be made when deciding the amount of artificial viscosity to add.

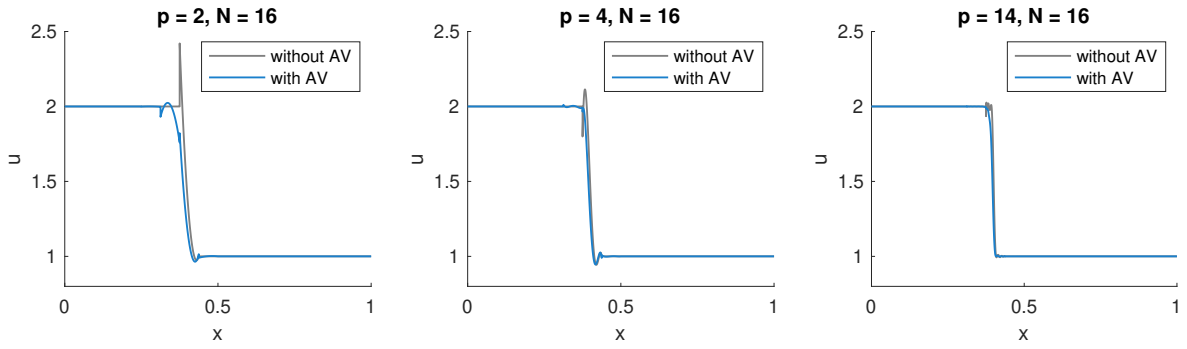
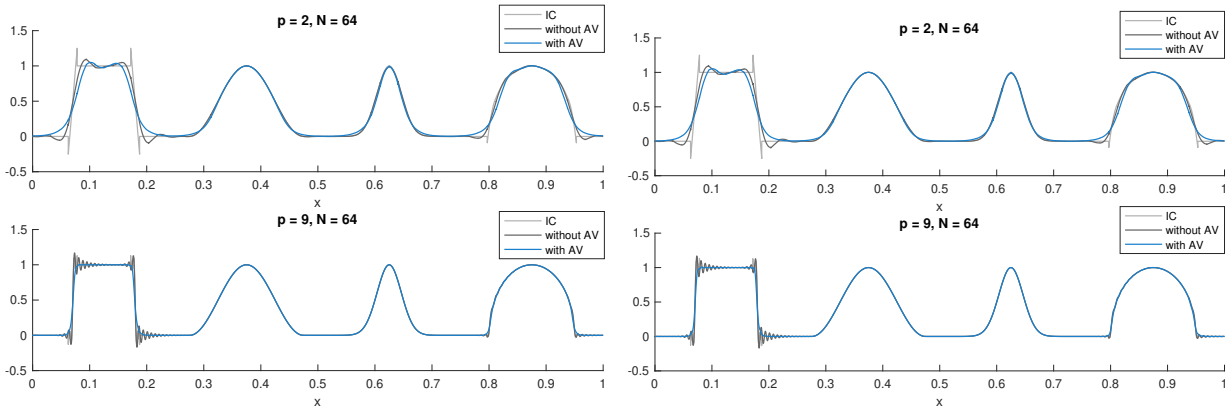


Figure 3.1: DG solutions for steady advection: $a = 1.0$, $c_0 = 1.8$, $c_p = 0.3$, with and without artificial viscosity (AV).

3.4.2 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" [73] traveling with constant speed $a = 1$. 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 wave traveling are shown in Fig. 3.2. 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 perform similarly. The oscillations in the solutions are greatly reduced by the addition of artificial viscosity.



(a) DG (b) HDG

Figure 3.2: DG and HDG solutions for unsteady advection: $t = 1.0$, $\Delta t = 0.001$, $c_0 = 1.0$, $c_p = 0.4$.

3.4.3 Unsteady Inviscid Burgers

We compare the use of the continuous piece-wise linear artificial viscosity to the discontinuous piece-wise constant artificial viscosity through an example of the 1D inviscid Burgers equation

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = 0, \tag{3.46}$$

with an initial sinusoidal wave

$$u_0(x) = \sin(2\pi(x - 0.5)) + 1, \tag{3.47}$$

and periodic boundary conditions. A shock wave will start to form at $t = \frac{1}{2\pi}$ as the wave propagates in space. We run the test case on a uniform mesh. The Godunov scheme [74] is used for the advection flux.

In Fig. 3.3, 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.

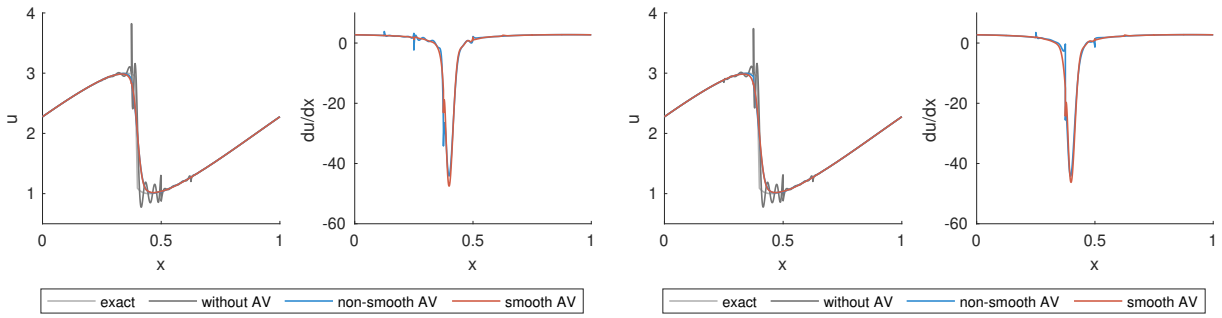
Given the implicit characteristics solution $u(x,t) = u(x-ut)$, the exact solution u_e can be found iteratively at each point in space. We define the L_1 error of the solution as $\int_0^1 |u-u_e|dx$ and the L_2 error of the solution as $\sqrt{\int_0^1 (u-u_e)^2 dx}$. The convergence of the L_1 and L_2 errors is shown in Fig. 3.4. 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 1. The errors for the piece-wise 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% 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 piece-wise constant and the piece-wise linear artificial viscosity increases further.

3.5 Two-dimensional test cases

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



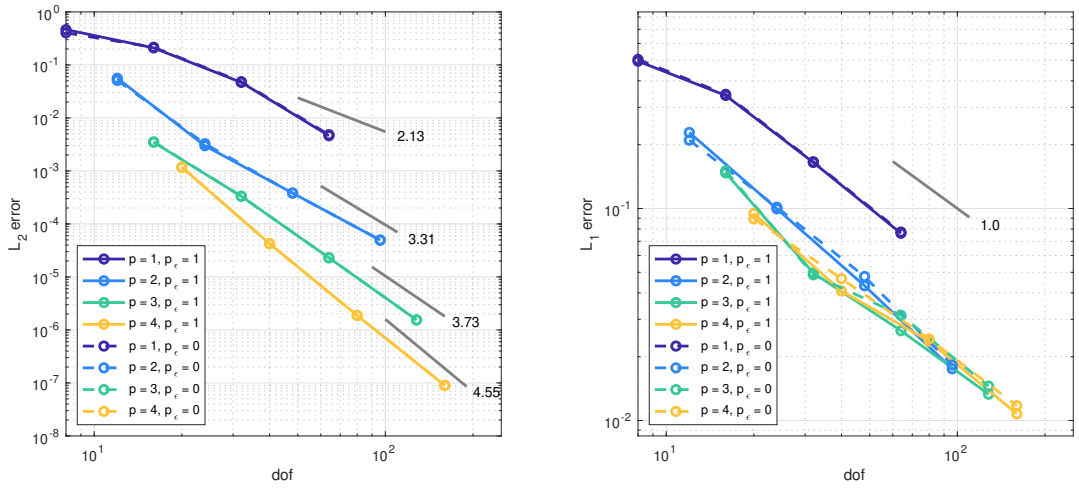
(a) DG (b) HDG

Figure 3.3: DG and HDG solutions to the Burgers equation: $N = 8$, $p = 9$, $t = 0.2.$, $\Delta t = 2 \times 10^{-4}$, $c_0 = 2.0$, $c_p = 0.4$.

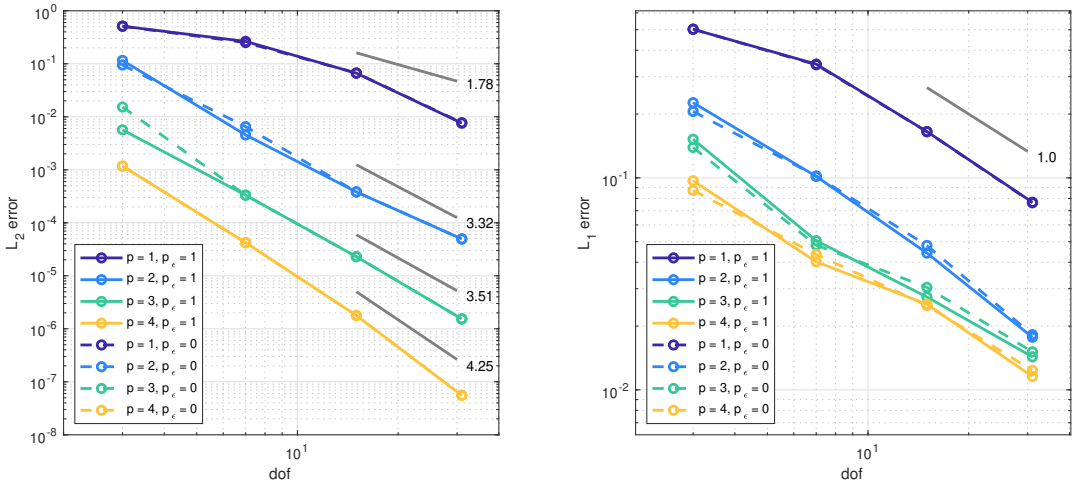
3.5.1 Transonic Airfoil

We demonstrate the solver's shock-capturing ability and the effect of Laplace smoothing with a transonic case at freestream Mach number $M = 0.8$ past a NACA 0012 airfoil at an angle of attack $\alpha = 1.25°$. 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. 3.5. The adapted mesh and solution for the resolution indicator with Laplace smoothing are shown in Fig. 3.6. 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.

Since the refined elements of the unstructured mesh are not perfectly aligned with the shock, the non-linear 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. Fig. 3.7 shows the entropy measured along the line probe. Laplace smoothing is able to reduce the oscillations in the entropy for both the resolution and the variation indicators. Note that the Laplace smoothing is for the elemental

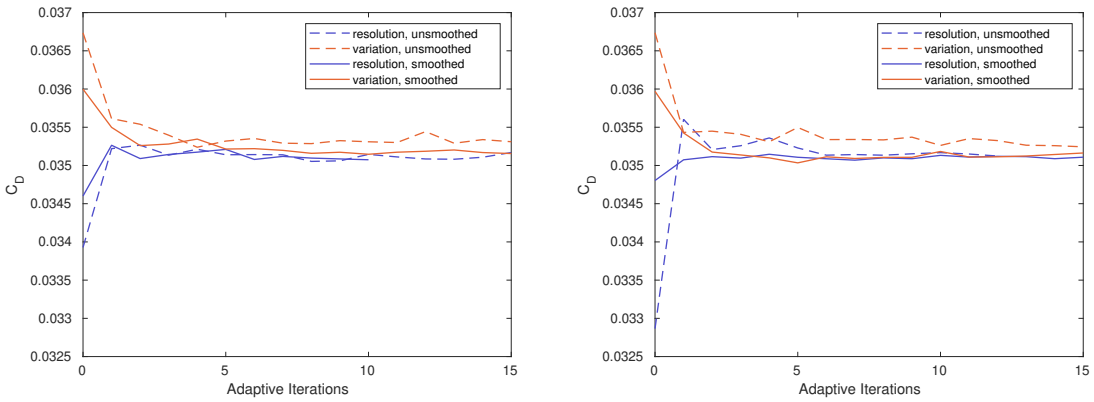


(a) DG, $t = 0.05$ on the left and $t = 0.2$ on the right.



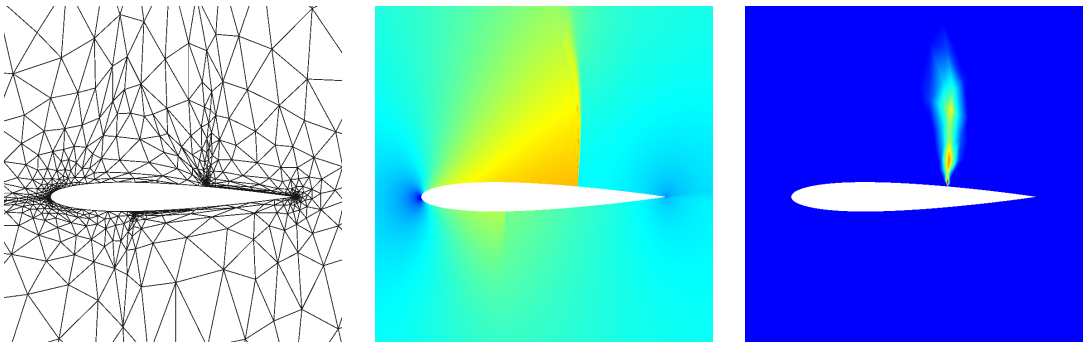
(b) HDG, $t = 0.05$ on the left and $t = 0.2$ on the right.

Figure 3.4: Error convergence for DG and HDG solutions to Burgers equation.



(a) DG (b) HDG

Figure 3.5: Drag coefficient convergence for the transonic airfoil case.

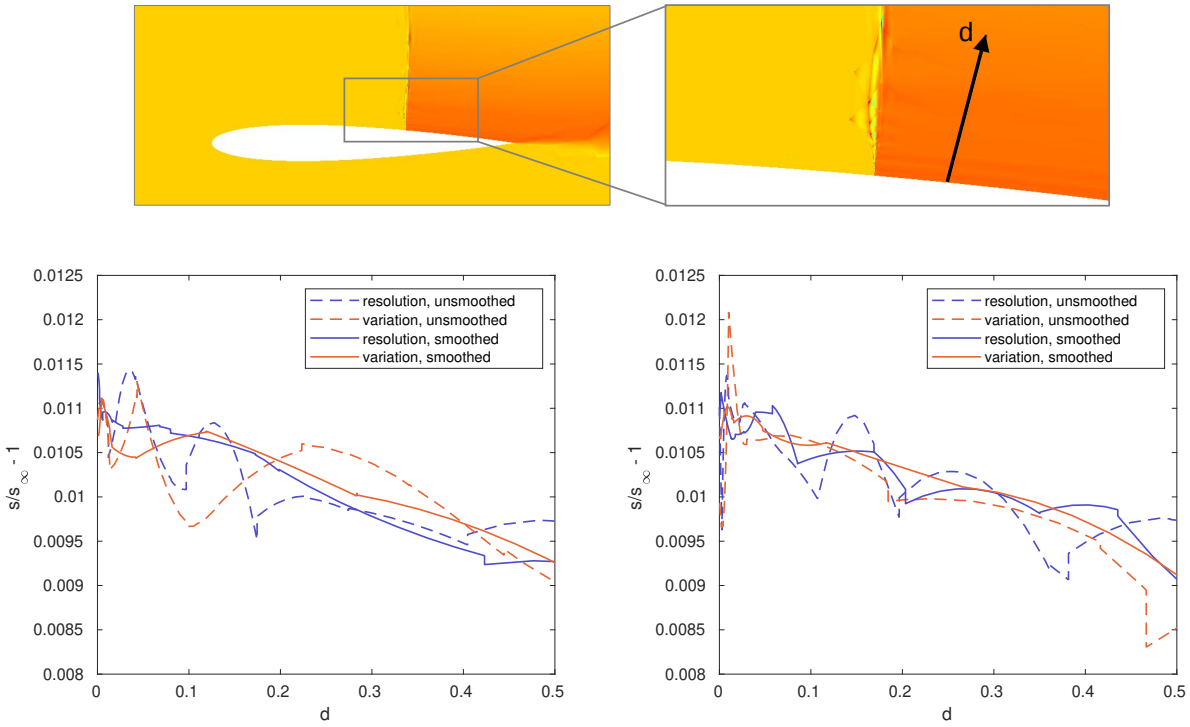


(a) Mesh (b) Mach number (c) Stabilization viscosity

Figure 3.6: Transonic airfoil DG solution with the resolution indicator and Laplace smoothing, $\texttt{dof} = 1 \times 10^4$.

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.

Fig. 3.8 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.

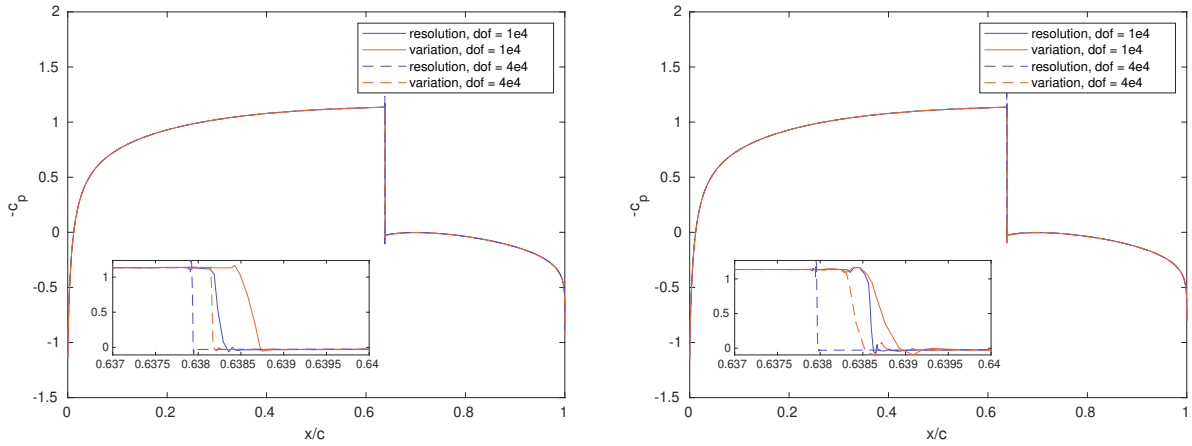


(b) DG (c) HDG

Figure 3.7: Line probe of entropy behind the shock.

3.5.2 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



(a) DG (b) HDG

Figure 3.8: Pressure coefficient distributions on the upper surface of the airfoil in transonic flow.

in the region $[-1.5, 1.5] \times [0, 0.8]$. The bump on the bottom of the channel is defined by

$$y = 0.0625e^{-(x/0.2)^2}. \tag{3.48}$$

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. Fig. 3.9 shows a convergence study of the L_2 error in the total enthalpy, defined as

$$E_H = \sqrt{\frac{\int_\Omega (H/H_\infty - 1)^2 d\Omega}{\int_\Omega d\Omega}}, \tag{3.49}$$

on uniformly refined meshes. The modification of the state vector in the stabilization term, i.e. a non-identity tensor T_{kl} in Eqn. 3.16, 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 1. 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. 3.10 we compare mesh adaptation with the drag adjoint and with the entropy variables. The drag coefficient is defined as

$$C_D = \frac{\int_{\text{bottom}} (P - P_\infty) n_x dl}{\frac{\gamma}{2} P_\infty M_\infty^2 h}, \tag{3.50}$$

where n_x is the horizontal component of the outward-pointing normal vector, $h = 0.025$ is the height of the bump. The drag coefficient and the error in the total enthalpy both converge as the adaptation progresses when using the drag adjoint and 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.

Fig. 3.11 displays the meshes for the two adaptive iterations marked in Fig. 3.10. The mesh for iteration 9 contains more refined elements along the shock while 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.

3.5.3 Inviscid Hypersonic Flow Past a Half-Cylinder

This test case is a hypersonic, $M = 5$, flow past a half-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

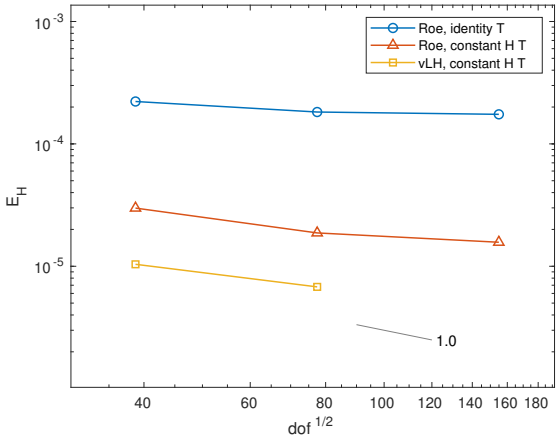


Figure 3.9: Total enthalpy error convergence on uniformly refined meshes for the bump case, using different diffusivity tensor transformations, as defined in Eqn. 3.16.

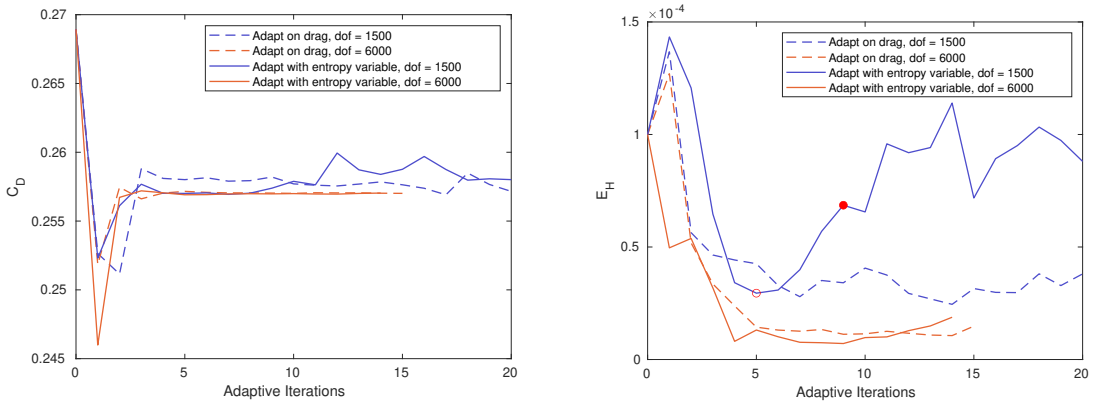
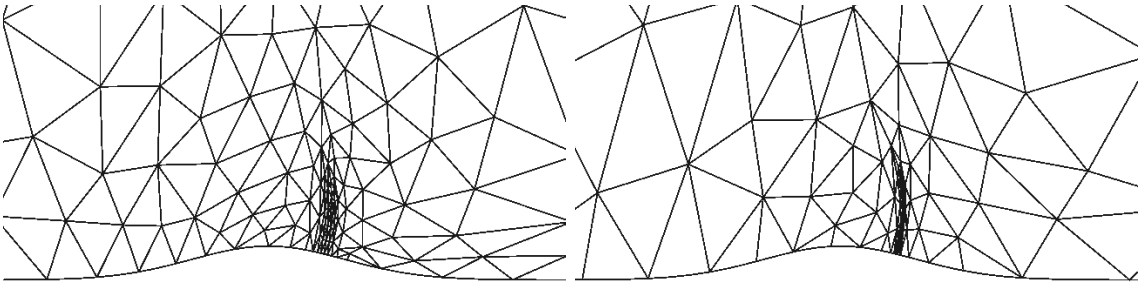


Figure 3.10: Drag and total enthalpy convergence during adaptation for the bump case.



(a) Iteration 5 ○ (b) Iteration 9 ●

Figure 3.11: Meshes adapted using the entropy variables for the bump case.

in the total enthalpy varies slightly from the previous test case:

$$E_H = \frac{1}{D}\sqrt{\int_\Omega (H/H_\infty - 1)^2 d\Omega}, \tag{3.51}$$

where $D = 2$ is the diameter of the cylinder. Since 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 3.1. Compared to DG, HDG reduces the degrees of freedom and the number of nonlinear iterations required for convergence. However, the computational time does not improve for this case of $p = 2$. Since HDG requires static condensation and back-substitution 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.

Table 3.1: Time and iterations required to converge to a $M = 5$ solution from a $M = 4.5$ one.

	`dof`	nonlinear iterations	CPU time (s)
DG	2400	2803	3.9405×10^2
HDG	1875	2670	8.2952×10^2

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. 3.13. 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, while the entropy adjoint focuses extensively on the shock wave.

The errors in the drag coefficient and the total enthalpy are shown in Fig. 3.14. The drag coefficient is defined as

$$C_D = \frac{\int_{\text{cylinder}} (P - P_\infty) n_x dl}{\frac{\gamma}{2} P_\infty M_\infty^2 D}, \tag{3.52}$$

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 5 adaptive iterations. The `dof` counts plotted for both DG and HDG are the element interior degrees of freedom, since that is what the MOESS algorithm targets. The "exact" value of the drag was obtained with adaptation on E_H and `dof` $= 38400$. The linear coefficient C in Eqn. (3.17) has to be tuned to achieve convergence, as listed in Table 3.2. With a strong shock and the effect of the Laplace smoothing that smears 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 stabilization, so the coefficient value was lowered on the coarsest mesh.

Table 3.2: C values in Eqn. (3.17) for the hypersonic cylinder case.

indicator	DG	HDG
resolution	256	128 (coarsest) / 256 (otherwise)
variation	6	6

The errors for a series of structured shock-fitted meshes are also plotted in Fig. 3.14 for comparison. The shock-fitted meshes are shown in Fig. 3.12. 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, defined as

$$x_{\text{tanh}} = \frac{5}{8}\text{arctanh}\left(x_{\text{linear}}\right) + 1, \tag{3.53}$$

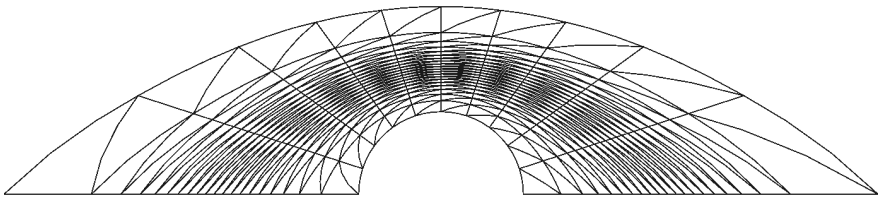
where $x_{\text{linear}} \in [\tanh(-1.6), \tanh(2.4)]$ and $x_{\text{tanh}} \in [0, 2.5]$. The radial locations of the mesh points are found as

$$\frac{r}{R} = x_{\text{tanh}}\frac{(r_s - R)}{R} + 1, \tag{3.54}$$

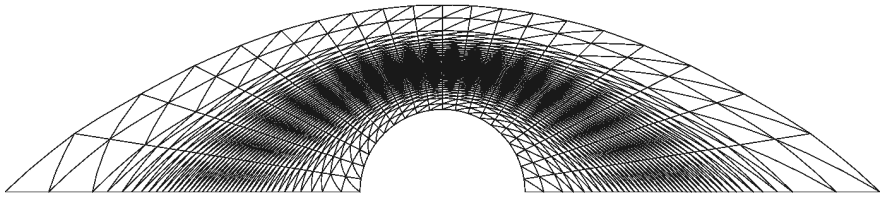
where r_s is the location of the shock for a particular tangential direction and $R = D/2$ is the radius of the cylinder. After finding the grid points, the neighboring points are connected, and the elements are divided in half to form a triangular mesh. The flow solution is then found on the newly generated mesh and the mesh generation and solution process are repeated several times. For the shock-fitted meshes, the elements are less skewed. $C = 32, 16$ 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 2 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 test 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 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. 3.15. From `dof` $= 4800$ to `dof` $= 9600$ the mesh is not



(a) `dof` = 2400



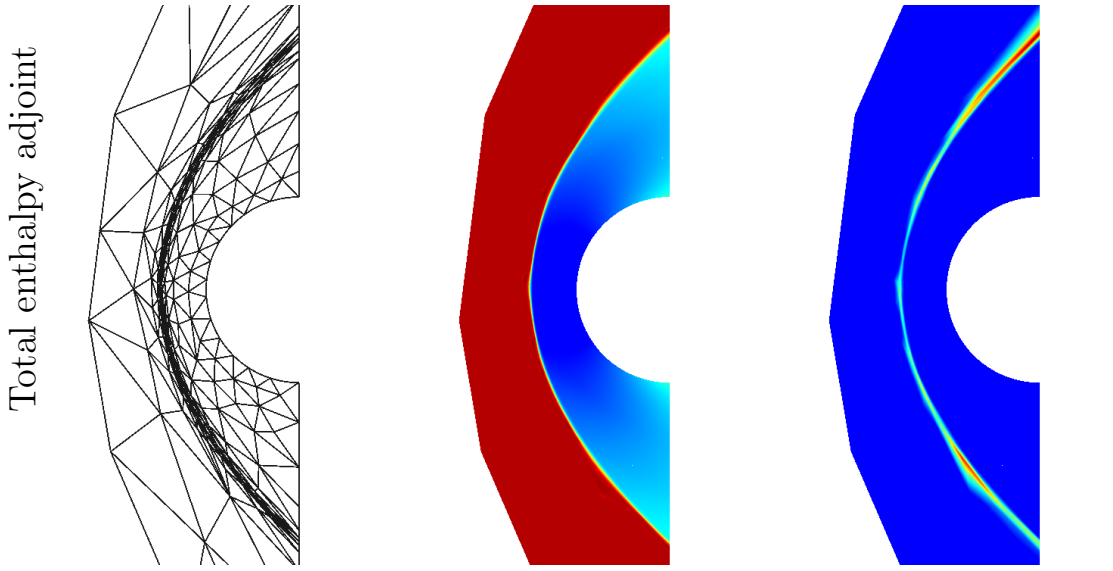
(b) `dof` = 9600

Figure 3.12: Shock-fitted meshes generated for comparison with mesh adaptation for the hypersonic cylinder case.

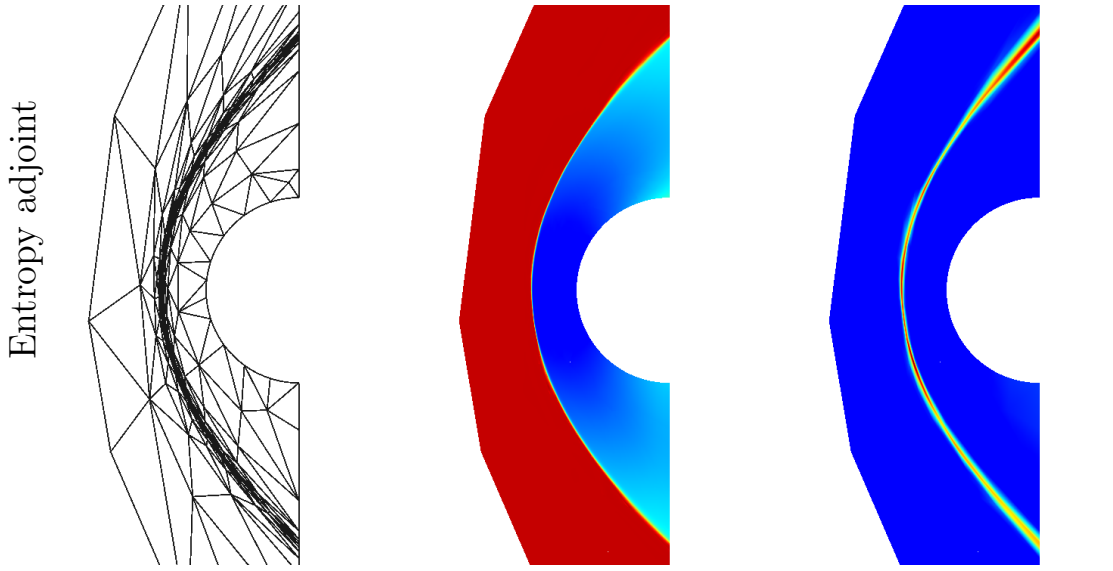
refined along the shock but only on the adjoint features.

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 [75, 76]. The adaptation decides which elements to refine based on the error indicators presented in section 3.3. Each adaptive iteration is governed by a fixed growth factor that is chosen to be 1.3. The adapted meshes are shown in Fig. 3.16. With hanging-node adaptation, the drag adjoint indicator leads to more refinement behind the shock and close to the cylinder boundary compared to the total enthalpy adjoint indicator, but it does not over-emphasize the drag adjoint features.

This suggests that the shock under-resolution observed with 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 ad-



(a) Mesh (b) Mach number (c) Stabilization viscosity



(d) Mesh (e) Mach number (f) Stabilization viscosity

Figure 3.13: Adaptive DG results for $M = 5$ inviscid flow past a cylinder. `dof` = 2400

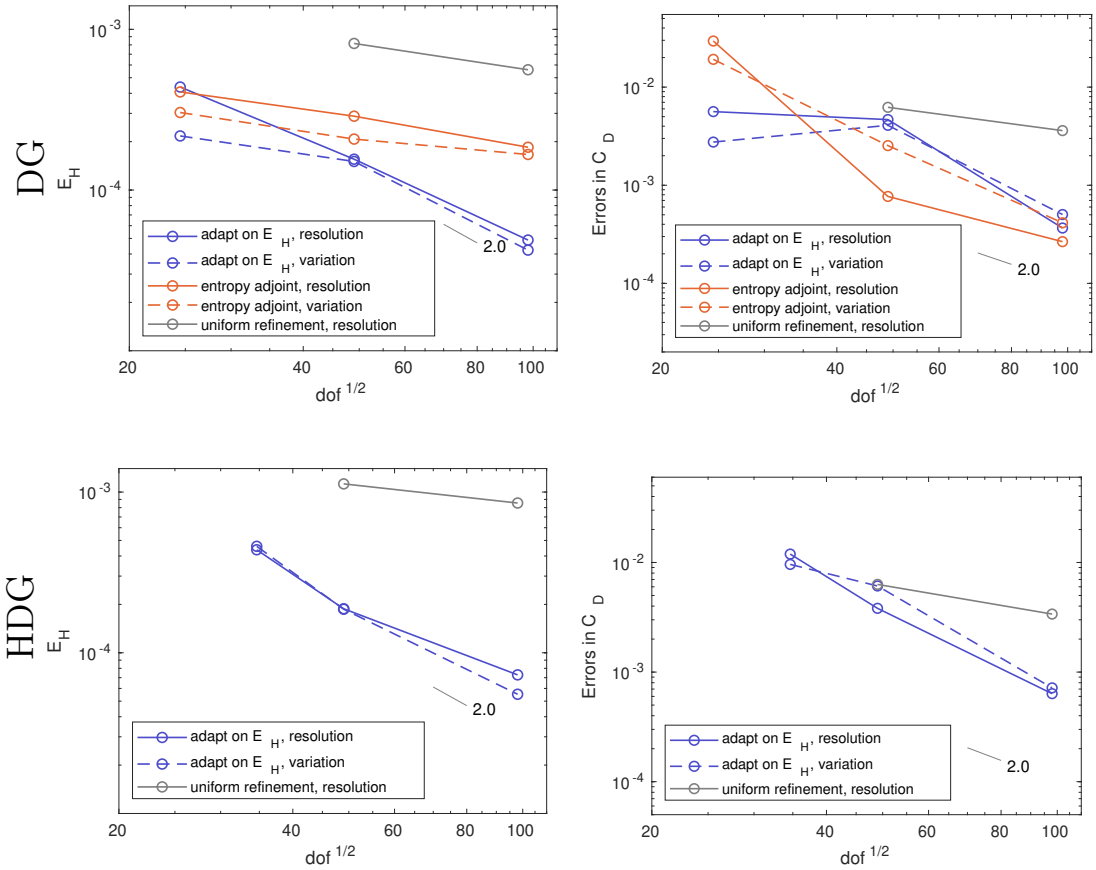
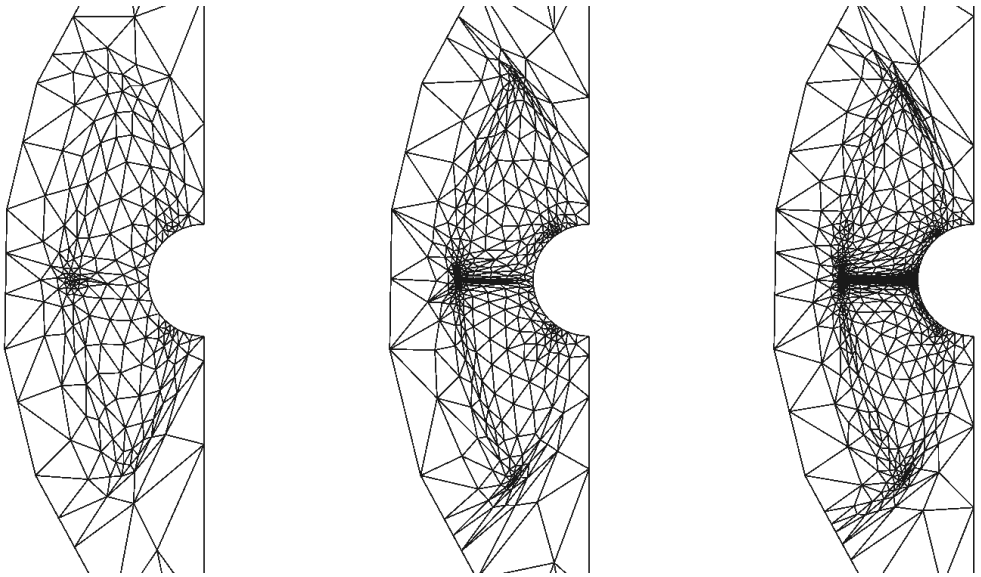
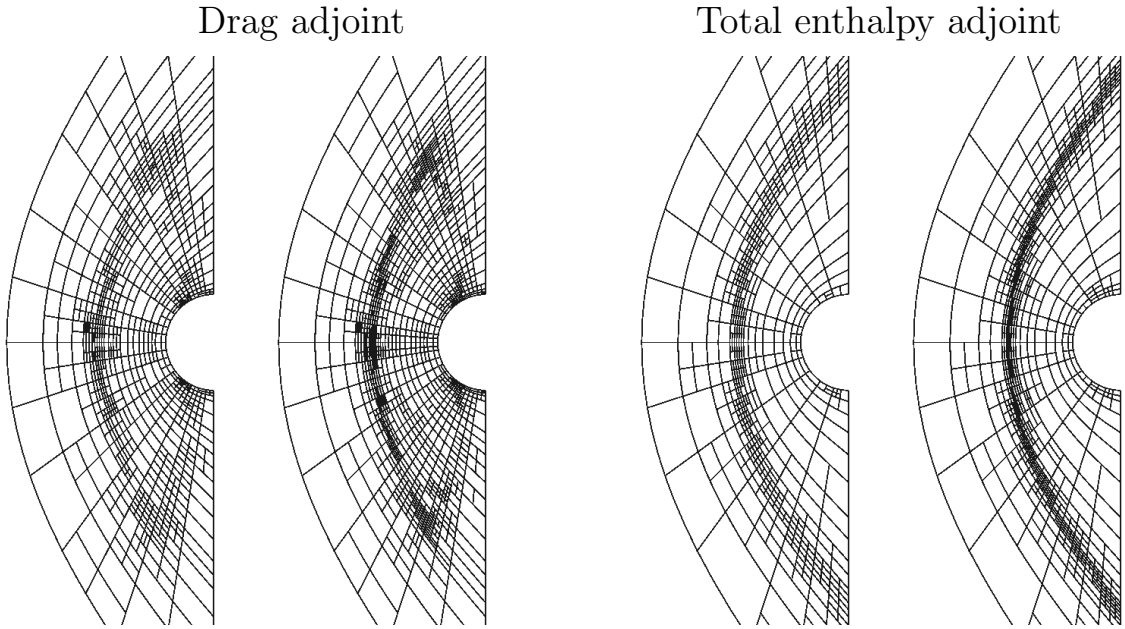


Figure 3.14: Convergence plots for the cylinder case.



(a) `dof` = 2400 (b) `dof` = 4800 (c) `dof` = 9600

Figure 3.15: Adapted meshes with MOESS for $M = 2$ flow past a cylinder using the drag adjoint.



(a) `dof` = 7605 (b) `dof` = 12105 (c) `dof` = 5211 (d) `dof` = 9306

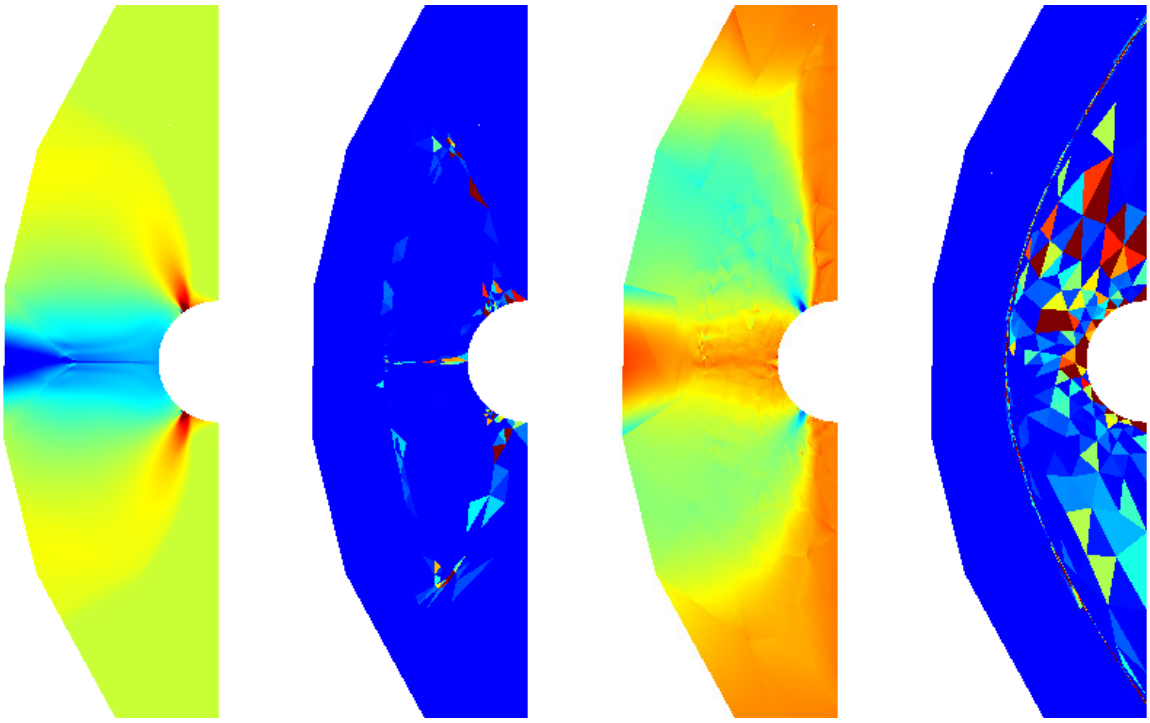
Figure 3.16: Adapted meshes with hanging node adaptation for $M = 2$ flow past a cylinder.

joint and the total enthalpy adjoint is shown in Fig. 3.17. The large error indicator values for the drag adjoint case are localized to elements that are already small, suggesting that 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 MOESS optimizes the mesh to equidistribute the marginal error-to-cost ratio, large error indicators in certain consistent regions will lead 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, as it only uses the magnitudes to select a fraction of elements to refine. Hence, it does not suffer the same over-resolution as long as the fixed fraction is sufficiently large. From Figure 3.17, the regions of overly high resolution for MOESS appear to be the 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 future work.

3.5.4 Viscous Hypersonic Flow Past a Half-Cylinder

One important goal for high-quality shock-capturing for hypersonic flow is to generate accurate predictions of wall heating. To test the heat transfer predictions of our solver, we use the canonical test case of $M = 17.605$ flow past a cylinder. The freestream Reynolds number is $Re = \rho_\infty R_c V_\infty / \mu_\infty = 376,930$, where $\rho_\infty, V_\infty, \mu_\infty$ are the freestream density, velocity magnitude, and viscosity, and R_c is the cylinder radius. The temperature for the freestream is $T_\infty = 200\text{K}$, and on the adiabatic wall we have $T_{\text{wall}}/T_\infty = 2.5$.

To validate our solutions, the DG solution for a fine mesh is shown in Fig. 3.18. Two quantities plotted along the wall boundary are the pressure coefficients and the Stanton



(a) drag adjoint, $[-2, 1.5]$ (b) error indicators, drag, $[0, 2 \times 10^{-5}]$ (c) total enthalpy adjoint, $[-0.06, 0.02]$ (d) error indicators, H, $[0, 2 \times 10^{-5}]$

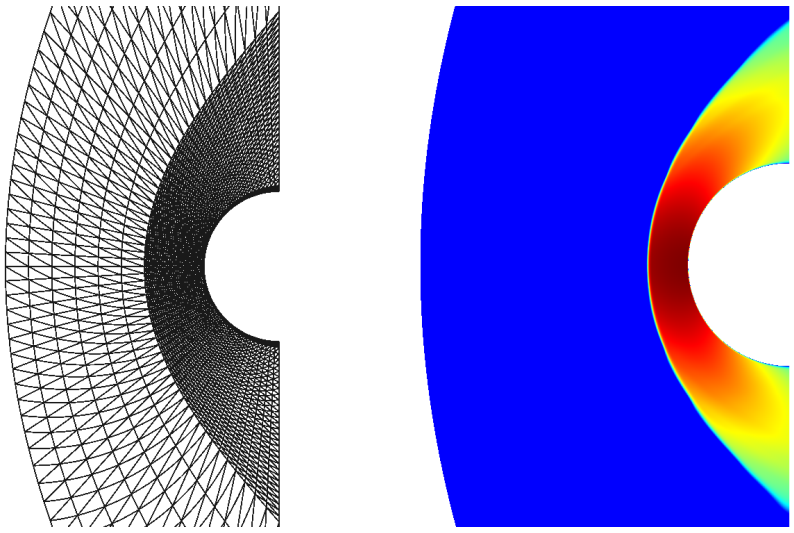
Figure 3.17: Comparison of x-momentum adjoint and error indicators for $M = 2$ flow past a cylinder.

number, defined as,

$$C_p = \frac{p - p_\infty}{\frac{1}{2}\rho V_\infty^2}, \quad C_H = \frac{q_{\text{wall}}}{c_p \rho_\infty V_\infty (T_{0,\infty} - T_{\text{wall}})}, \tag{3.55}$$

where q_{wall} is the heat transfer on the wall, found as the energy flux, $c_p = \frac{\gamma R}{\gamma - 1}$ is the constant pressure specific heat, and $T_{0,\infty}$ is the freestream stagnation temperature. Our DG and HDG solutions are overlapping for the pressure coefficient, and have some slight differences in the Stanton number, which could be caused by the different wall boundary treatments for DG and HDG. The Stanton number is compared to the DG solution of Ching at el. [28]. Their prediction with PDE-based artificial viscosity has good agreement with our DG result.

Our attempts for adaptation with MOESS and integrated heat flux on the wall as the targeted output were unsuccessful for this test case. The reason is when MOESS moves the boundary elements where variations are occurring inside the boundary layer, the solution projection on the adapted meshes either generates non-physical conditions or creates artificial discontinuities that become unstable in the next few time steps. This is likely alleviated if the adaptation is started from a relatively fine mesh. Instead, we opted for hanging-node adaptation. The adapted mesh and solution are shown in Fig. 3.19. In the heat adjoint plot, larger values appear along the stagnation streamline, indicating that the prediction of the wall heating is sensitive to changes in this area. The mesh is refined in the areas that affect the wall boundary, with the focus on the wall, the shock, the stagnation streamline, and the sonic lines behind the shock. With the Stanton number on an adapted fine mesh as the ground truth, the errors in the Stanton number are plotted in Fig. 3.20. The error converges faster with adaptation compared with uniform mesh refinement, showing the effectiveness of mesh adaptation in reducing the cost of an accurate solution.



(a) mesh (b) temperature, DG

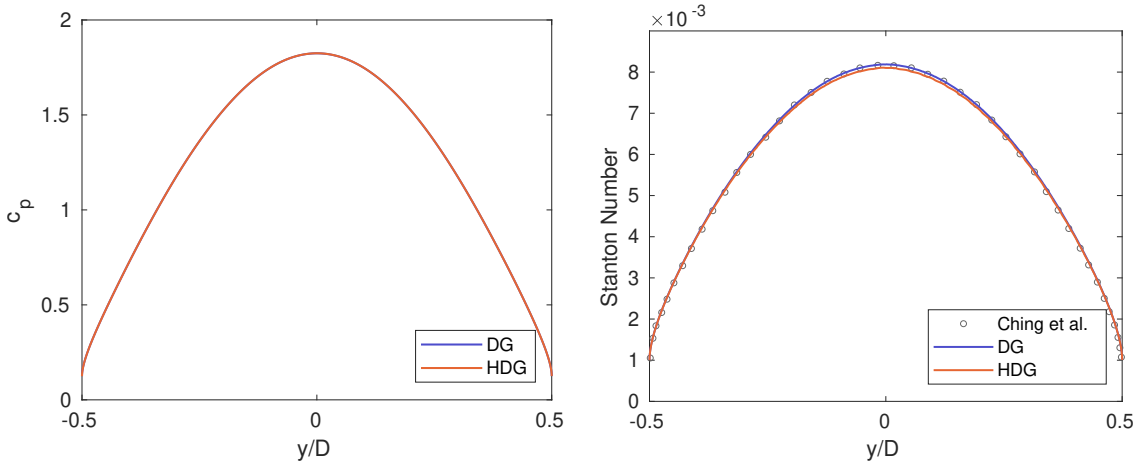
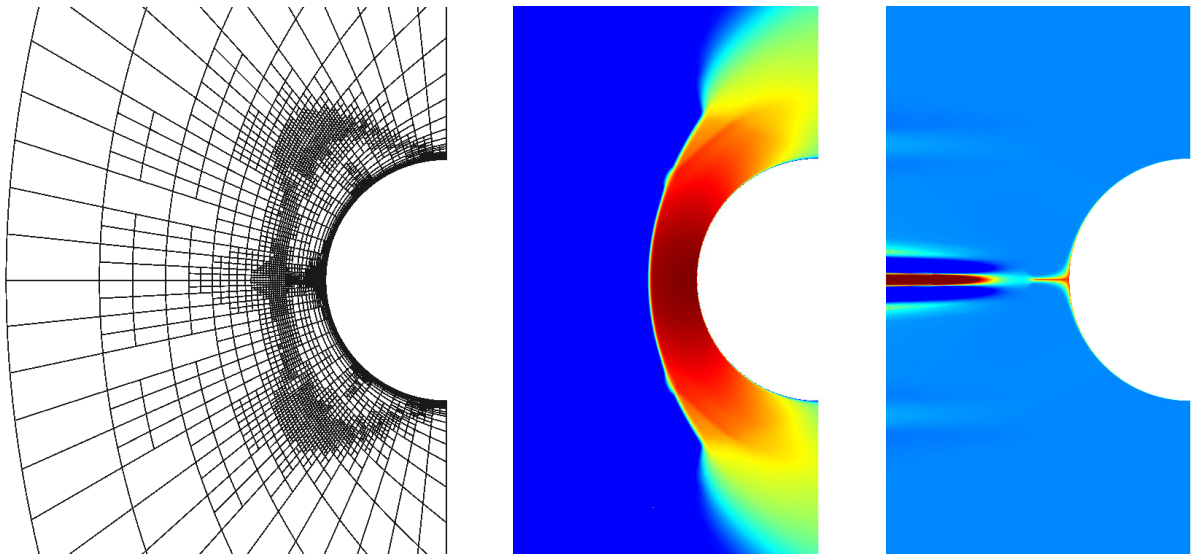


Figure 3.18: A fine mesh and the solution for viscous hypersonic flow past a half cylinder.



(a) mesh (b) temperature (c) adjoint

Figure 3.19: Mesh and solution adapted on drag for viscous hypersonic flow past a half cylinder.

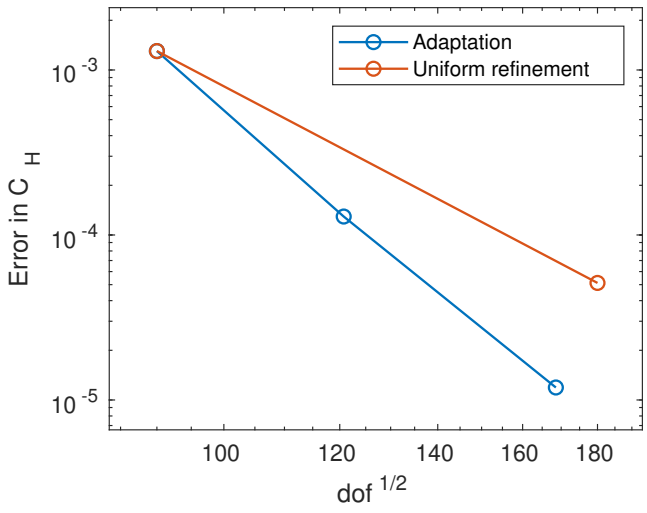


Figure 3.20: Error convergence for the Stanton number.

CHAPTER IV

Conclusion

Developing robust limiting and shock capturing approaches is critical for the application of high-order CFD methods. In this thesis, we focused on improving the efficiency and robustness of the Active Flux and DG methods in the presence of shocks.

The Active Flux method has the potential to produce solutions with better quality compared to the conventional CFD methods at the same cost. A physically based limiter for the Active Flux method is proposed [77]. To tackle the difficulty that no physical bounds can be determined for the acoustics systems, we employed the FCT limiting principles. The low-order method constructed as a baseline of the limiter uses linear basis functions on subtriangles. Test cases from linear to nonlinear equations have proven the effectiveness of the limiter. We paid particular attention to the problem of converging-diverging shocks that are shown to be challenging for the DG method, which falls into the category of conventional CFD methods. The success of Active Flux in simulating this problem is attributed to its multidimensional nature. Additionally, approaches to improve the efficiency of the implementation of Active Flux are discussed. The simplification of the half-step evaluation comes from the distinction of first- and second-order evolution terms. The improved Active Flux method has an efficiency comparable to that of the DG methods with Runge-Kutta time stepping.

The DG and HDG methods, on the other hand, are more mature and capable of handling

challenging applications. We demonstrated through one-dimensional and two-dimensional simulations the ability of the continuous artificial viscosity to capture shocks in an adaptive-mesh setting for both the DG and HDG discretizations [78]. We presented results that support generally agreed-upon ideas about the implementation of artificial-viscosity shock capturing. We found that the continuous artificial viscosity works better than piece-wise constant artificial viscosity in reducing oscillations near discontinuities. In the case of hypersonic flow, continuous artificial viscosity can potentially stabilize the cases that piece-wise constant artificial viscosity cannot. We discovered that the variation indicator results in faster convergence most of the time since 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 flow field, e.g. expansion waves. In our 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.

Some possible future work includes: a) addressing the limiting of the advection part for Active Flux separately from the acoustics part; b) tackling cases that involve shock and boundary layer interaction; c) improving the MOESS algorithm to mitigate the problem with error indicator disparity.

APPENDIX

APPENDIX A

Additional Test Case: Double Mach Reflection

A.1 Unsteady Adjoint

The space-time mesh adaptation used for this unsteady case is detailed in [79]. We give a summary of it here. For an unsteady output of the form

$$\bar{J} = \int_0^T J(\mathbf{U}(\mathbf{t}), t)\mathrm{d}t + J_T(\mathbf{U}(T)), \tag{A.1}$$

the continuous-in-time adjoint $\mathbf{\Psi}(t)$ is solved backwards in time from the differential equation:

$$-\mathbf{M}\frac{d\mathbf{\Psi}}{dt} + \frac{\partial \mathbf{R}}{\partial \mathbf{U}}^T \mathbf{\Psi} + \frac{\partial J}{\partial \mathbf{U}}^T = \mathbf{0}, \tag{A.2}$$

with boundary condition:

$$\mathbf{\Psi}(T) = -\mathbf{M}^{-1}\frac{dJ_T}{d\mathbf{U}}^T. \tag{A.3}$$

The output error is estimated as

$$\delta \bar{J} = \sum_{n=1}^{N_t} \sum_{e=1}^{N_e} \underbrace{\int_{t^{n-1}}^{t^n} -\mathbf{\Psi}_{h,e}^T \bar{\mathbf{R}}_{h,e}(\mathbf{U}_h^H) \mathrm{d}t}_{\varepsilon_e^n}, \tag{A.4}$$

where $\bar{\mathbf{R}} = \mathbf{M}\frac{d\mathbf{U}}{dt} + \mathbf{R}$ is the unsteady residual, and N_t is the number of time steps. The time contribution to the error is estimated using a spatially down-projected adjoint,

$$\delta \bar{J}^{\text{time}} = \sum_{n=1}^{N_t} \sum_{e=1}^{N_e} \underbrace{\int_{t^{n-1}}^{t^n} -(\mathbf{I}_H^h \mathbf{\Psi})_{h,e}^T \bar{\mathbf{R}}_{H,e}(\mathbf{U}_H) \mathrm{d}t}_{\varepsilon_e^{n,\text{time}}}, \tag{A.5}$$

where $\mathbf{I}_H^h$ is a least-square projection from order $p+1$ to p. The spatial error is then found as $\delta \bar{J}^{\text{space}} = \delta \bar{J} - \delta \bar{J}^{\text{time}}$. A growth factor for the total DOFs f^{tot} is prescribed by the user. The growth factors for the spatial and temporal DOFs f^{space} and f^{time} are calculated from $\delta \bar{J}^{\text{space}}$ and $\delta \bar{J}^{\text{time}}$ so that the marginal error to cost ratios for space and time are equal. The time dimension is then refined uniformly, and the mesh is adapted with MOESS and error indicators defined as

$$\epsilon_e^{\text{space}} = \left| \sum_{n=1}^{N_t} \varepsilon_e^{n,\text{space}} \right| = \left| \sum_{n=1}^{N_t} (\varepsilon_e^n - \varepsilon_e^{n,\text{time}}) \right| \tag{A.6}$$

for an element e.

A.2 Adaptation results

This inviscid test case is a $M = 10$ shock impinging on a 30° wedge, introduced by Woodward and Collela [80]. The computation domain is $[0, 4] \times [0, 1]$. The shock is given at

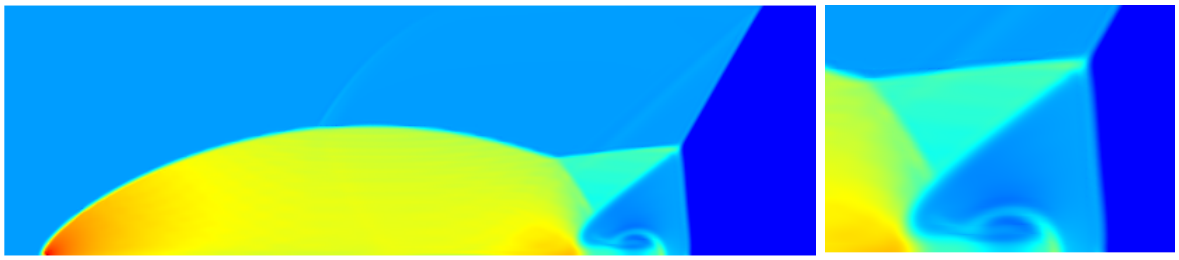


Figure A.1: Density solution on a fine mesh for double Mach reflection. The shear layer that involves the Kelvin-Helmholtz instability is zoomed in on the right.

time t as

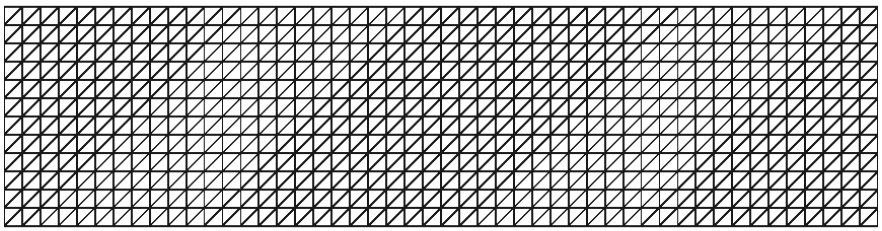
$$[\rho, u, v, P] = \begin{cases} [8, 8.25\cos(\frac{\pi}{6}), -8.25\sin(\frac{\pi}{6}), 116.5], \ x < x_0 + \frac{y}{\sqrt{3}} + \frac{2u_s t}{\sqrt{3}}, \\ [1.4, 0, 0, 1.0], \ x \geq x_0 + \frac{y}{\sqrt{3}} + \frac{2u_s t}{\sqrt{3}}, \end{cases} \tag{A.7}$$

where $x_0 = 1/6$ is the start of the wall on the domain bottom, and $u_s = 10$ is the shock speed. This relation is used to prescribe the initial condition and boundary condition on the left and top boundaries and between $[0, x_0]$ on the bottom boundary. Extrapolation is used at the right boundary. In practice, a tanh function is used to specify the shock to prevent non-physical conditions in the L_2 initialized flow field. Second-order backward difference and a third-order diagonal Runge-Kutta method are employed for the primal and the adjoint solutions, respectively. The coefficient for the amount of artificial viscosity is set to $C = 200$.

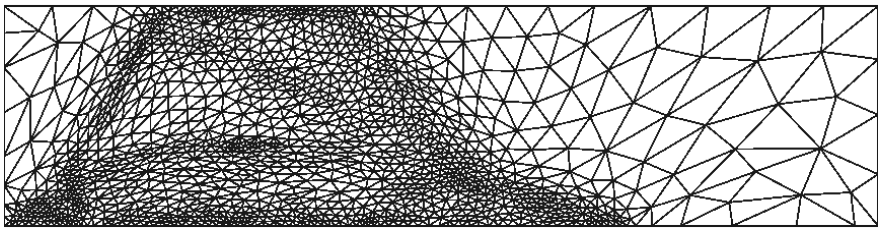
The DG solution at $t = 0.2$ on a fine 48×192 structured triangular mesh is shown in Fig. A.1. The triangles are created by splitting the quadrature elements in half. A self-similar structure forms upon impact over the ramp and grows in size as the shock propagates to the right. The flow consists of two triple points, each consisting of the intersection of three shocks and a shear layer. The shear layer originating from the triple point to the right exhibits Kelvin-Helmholtz instability. We are able to capture a jet stream adjacent to the bottom wall that emits from it.

We are interested in the areas that the mesh adaptation targets in this unsteady flow field with complex structures. The adaptation is started from a coarse $12 \times 48 \times 2$ mesh. Two outputs are used: lift on the bottom wall and the domain integral of the total enthalpy, I_{h0}, both at the final time step $t = 0.2$. The adapted meshes are shown in Fig. A.2 each after two adaptation iterations and a growth factor $f^{\text{tot}} = 1.5$. The adaptation with the total enthalpy adjoint targets the area swept by the shock structures. The anisotropy in the mesh follows the orientation of the impinging shock. On the other hand, the adaptation with the lift adjoint refines only in areas relevant to the lift prediction, including the initial shock location, the arched shock, and the wall boundary, leaving the final shock under-resolved.

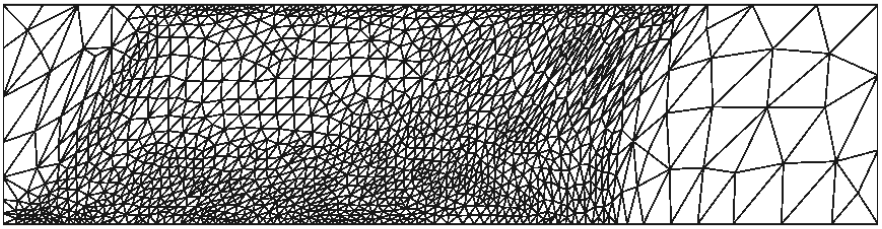
This study informs us in a similar way as the case of hypersonic flow past a half-cylinder about the relationship between the shock and the adaptation indicators. For the purpose of resolving the shock, the total enthalpy is a better indicator. However, the prediction of the surface integrals (e.g. lift and drag) does not necessarily require the resolution of the shock structures, particularly at all times in an unsteady simulation.



(a) Initial mesh.



(b) Adapted on lift.



(c) Adatped on I_{h0}.

Figure A.2: Adapted meshes for the double Mach reflection problem.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Zhijian J Wang, Krzysztof Fidkowski, Rémi Abgrall, Francesco Bassi, Doru Caraeni, Andrew Cary, Herman Deconinck, Ralf Hartmann, Koen Hillewaert, Hung T Huynh, et al. High-order CFD methods: current status and perspective. *International Journal for Numerical Methods in Fluids*, 72(8):811–845, 2013.

[2] Sanjiva K Lele. Compact finite difference schemes with spectral-like resolution. *Journal of computational physics*, 103(1):16–42, 1992.

[3] Chi-Wang Shu. High-order finite difference and finite volume WENO schemes and discontinuous Galerkin methods for CFD. *International Journal of Computational Fluid Dynamics*, 17(2):107–118, 2003.

[4] William H Reed and Thomas R Hill. Triangular mesh methods for the neutron transport equation. Technical report, Los Alamos Scientific Lab., N. Mex.(USA), 1973.

[5] Bernardo Cockburn and Chi-Wang Shu. The Runge–Kutta discontinuous Galerkin method for conservation laws V: multidimensional systems. *Journal of Computational Physics*, 141(2):199–224, 1998.

[6] Bernardo Cockburn and Chi-Wang Shu. The local discontinuous Galerkin method for time-dependent convection-diffusion systems. *SIAM journal on numerical analysis*, 35(6):2440–2463, 1998.

[7] Bernardo Cockburn and Chi-Wang Shu. Runge-Kutta discontinuous Galerkin methods for convection-dominated problems. *Journal of Scientific Computing*, 16(3):173–261, 2001.

[8] N.C. Nguyen, J. Peraire, and B. Cockburn. An implicit high-order hybridizable discontinuous Galerkin method for linear convection-diffusion equations. *Journal of Computational Physics*, 228(9):3232–3254, 2009.

[9] J. Peraire, N. C. Nguyen, and B. Cockburn. A hybridizable discontinuous Galerkin method for the compressible Euler and Navier-Stokes equations. AIAA Paper 2010-363, 2010.

[10] Jochen Schütz and George May. A hybrid mixed method for the compressible Navier–Stokes equations. *Journal of Computational Physics*, 240:58–75, May 2013.

[11] Michael Woopen, Aravind Balan, Georg May, and Jochen Schütz. A comparison of hybridized and standard DG methods for target-based hp-adaptive simulation of compressible flow. *Computers & Fluids*, 98:3–16, July 2014.

[12] M Yousuff Hussaini and Thomas A Zang. Spectral methods in fluid dynamics. *Annual review of fluid mechanics*, 19(1):339–367, 1987.

[13] Anthony T Patera. A spectral element method for fluid dynamics: laminar flow in a channel expansion. *Journal of computational Physics*, 54(3):468–488, 1984.

[14] Hung T Huynh. A flux reconstruction approach to high-order schemes including discontinuous Galerkin methods. In *18th AIAA computational fluid dynamics conference*, page 4079, 2007.

[15] Timothy A Eymann and Philip L Roe. Multidimensional active flux schemes. In *21st AIAA computational fluid dynamics conference*, page 2940, 2013.

[16] Philip L Roe, Jungyeoul Maeng, and Doreen Fan. Comparing Active Flux and discontinuous Galerkin methods for compressble flow. In *2018 AIAA Aerospace Sciences Meeting*, page 0836, 2018.

[17] Fanchen He and Philip L Roe. A novel numerical scheme based on Active Flux method for hyperbolic heat equations in multidimensional space. In *AIAA Aviation 2019 Forum*, page 3638, 2019.

[18] Fanchen He. *Towards a New-generation Numerical Scheme for the Compressible Navier-Stokes Equations With the Active Flux Method*. PhD thesis, 2021.

[19] Philip Roe. Is discontinuous reconstruction really a good idea? *Journal of Scientific Computing*, 73(2):1094–1114, 2017.

[20] Philip Roe. Designing CFD methods for bandwidth—a physical approach. *Computers & Fluids*, 214:104774, 2021.

[21] J. Von Neumann and R. D. Richtmyer. A method for the numerical calculation of hydrodynamic shocks. *Journal of Applied Physics*, 21(3):232–237, 1950.

[22] Antony Jameson, Wolfgang Schmidt, and Eli Turkel. Numerical solution of the euler equations by finite volume methods using runge kutta time stepping schemes. In *14th fluid and plasma dynamics conference*, page 1259, 1981.

[23] Thomas JR Hughes and Michel Mallet. A new finite element formulation for computational fluid dynamics: Iv. a discontinuity-capturing operator for multidimensional advective-diffusive systems. *Computer methods in applied mechanics and engineering*, 58(3):329–336, 1986.

[24] Ralf Hartmann and Paul Houston. Adaptive discontinuous galerkin finite element methods for the compressible euler equations. *Journal of Computational Physics*, 183(2):508–532, 2002.

[25] Ralf Hartmann. Adaptive discontinuous Galerkin methods with shock-capturing for the compressible Navier-Stokes equations. *International Journal for Numerical Methods in Fluids*, 51(9–10):1131–1156, 2006.

[26] P.-O. Persson and J. Peraire. Sub-cell shock capturing for discontinuous Galerkin methods. AIAA Paper 2006-112, 2006.

[27] Garrett E. Barter and David L. Darmofal. Shock capturing with PDE-based artificial viscosity for DGFEM: Part I, formulation. *Journal of Computational Physics*, 229(5):1810–1827, 2010.

[28] Eric J. Ching, Yu Lv, Peter Gnoffo, Michael Barnhardt, and Matthias Ihme. Shock capturing for discontinuous Galerkin methods with application to predicting heat transfer in hypersonic flows. *Journal of Computational Physics*, 376:54–75, January 2019.

[29] Aldo Bonfiglioli, Marco Grottadaurea, Renato Paciorri, and Filippo Sabetta. An unstructured, three-dimensional, shock-fitting solver for hypersonic flows. *Computers & Fluids*, 73:162–174, 2013.

[30] Alessia Assonitis, Renato Paciorri, and Aldo Bonfiglioli. Numerical simulation of shock/boundary-layer interaction using an unstructured shock-fitting technique. *Computers & Fluids*, 228:105058, 2021.

[31] Matthew J Zahr, Andrew Shi, and P-O Persson. Implicit shock tracking using an optimization-based high-order discontinuous Galerkin method. *Journal of Computational Physics*, 410:109385, 2020.

[32] Tianci Huang and Matthew J Zahr. A robust, high-order implicit shock tracking method for simulation of complex, high-speed flows. *Journal of Computational Physics*, 454:110981, 2022.

[33] Andrew Corrigan, Andrew D Kercher, and David A Kessler. A moving discontinuous Galerkin finite element method for flows with interfaces. *International Journal for Numerical Methods in Fluids*, 89(9):362–406, 2019.

[34] Andrew D Kercher and Andrew Corrigan. A least-squares formulation of the moving discontinuous Galerkin finite element method with interface condition enforcement. *Computers & Mathematics with Applications*, 95:143–171, 2021.

[35] Ami Harten. High resolution schemes for hyperbolic conservation laws. *Journal of computational physics*, 135(2):260–278, 1997.

[36] Bram Van Leer. Towards the ultimate conservative difference scheme. v. a second-order sequel to godunov's method. *Journal of computational Physics*, 32(1):101–136, 1979.

[37] A. Burbeau, P. Sagaut, and C.H. Bruneau. A problem-independent limiter for high–order Runge-Kutta discontinuous Galerkin methods. *Journal of Computational Physics*, 169(1):111–150, 2001.

[38] Ami Harten, Bjorn Engquist, Stanley Osher, and Sukumar R Chakravarthy. Uniformly high order accurate essentially non-oscillatory schemes, iii. In *Upwind and high-resolution schemes*, pages 218–290. Springer, 1987.

[39] Xu-Dong Liu, Stanley Osher, and Tony Chan. Weighted essentially non-oscillatory schemes. *Journal of computational physics*, 115(1):200–212, 1994.

[40] J. Qiu and C.W. Shu. Hermite WENO schemes and their applications as limiters for Runge-Kutta discontinuous Galerkin, method: One-dimensional case. *Journal of Computational Physics*, 193(1):115–135, 2003.

[41] Hong Luo, Joseph D Baum, and Rainald Löhner. A Hermite WENO-based limiter for discontinuous Galerkin method on unstructured grids. *Journal of Computational Physics*, 225(1):686–713, 2007.

[42] Jay P Boris and David L Book. Flux-corrected transport. i. shasta, a fluid transport algorithm that works. *Journal of computational physics*, 11(1):38–69, 1973.

[43] David L Book, Jay P Boris, and K Hain. Flux-corrected transport ii: Generalizations of the method. *Journal of Computational Physics*, 18(3):248–283, 1975.

[44] Jay P Boris and David L Book. Flux-corrected transport. iii. minimal-error fct algorithms. *Journal of Computational Physics*, 20(4):397–431, 1976.

[45] Steven T Zalesak. Fully multidimensional flux-corrected transport algorithms for fluids. *Journal of computational physics*, 31(3):335–362, 1979.

[46] Rainald Löhner, Ken Morgan, Jaime Peraire, and Mehdi Vahdati. Finite element flux-corrected transport (fem–fct) for the Euler and Navier–Stokes equations. *International Journal for Numerical Methods in Fluids*, 7(10):1093–1109, 1987.

[47] Philip Roe. Did numerical methods for hyperbolic problems take a wrong turning? In *XVI International Conference on Hyperbolic Problems: Theory, Numerics, Applications*, pages 517–534. Springer, 2016.

[48] Xiangxiong Zhang and Chi-Wang Shu. Maximum-principle-satisfying and positivity-preserving high-order schemes for conservation laws: survey and new developments. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 467(2134):2752–2776, 2011.

[49] Doreen Fan and Philip L Roe. Investigations of a new scheme for wave propagation. In *22nd AIAA Computational Fluid Dynamics Conference*, page 2449, 2015.

[50] Wasilij Barsukow, Jonathan Hohm, Christian Klingenberg, and Philip L Roe. The active flux scheme on Cartesian grids and its low mach number limit. *Journal of Scientific Computing*, 81(1):594–622, 2019.

[51] Gerald Beresford Whitham. *Linear and nonlinear waves*. John Wiley & Sons, 2011.

[52] Dmitri Kuzmin, Rainald Löhner, and Stefan Turek. *Flux-corrected transport: principles, algorithms, and applications*. Springer, 2012.

[53] Richard Courant and David Hilbert. *Methods of mathematical physics: partial differential equations*. John Wiley & Sons, 2008.

[54] Fanchen He and Philip L Roe. The treatment of conservation in the Active Flux method. In *AIAA AVIATION 2020 FORUM*, page 3032, 2020.

[55] Maria Lukacova-Medvidova, KW Morton, and Gerald Warnecke. Finite volume evolution Galerkin methods for hyperbolic systems. *SIAM Journal on Scientific Computing*, 26(1):1–30, 2004.

[56] KG Guderley. Starke kugelige und zylindrische verdichtungsstosse in der nahe des kugelmitterpunktes bnw. der zylinderachse. *Luftfahrtforschung*, 19:302, 1942.

[57] Hans Motz. The physics of laser fusion. *London and New York*, 1979.

[58] Stefano Atzeni and Jürgen Meyer-ter Vehn. *The physics of inertial fusion: beam plasma interaction, hydrodynamics, hot dense matter*, volume 125. OUP Oxford, 2004.

[59] Michael Fink, W Hillebrandt, and FK Röpke. Double-detonation supernovae of sub-chandrasekhar mass white dwarfs. *Astronomy & Astrophysics*, 476(3):1133–1143, 2007.

[60] Scott D Ramsey, James R Kamm, and John H Bolstad. The guderley problem revisited. *International Journal of Computational Fluid Dynamics*, 26(2):79–99, 2012.

[61] P.L. Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics*, 43(2):357–372, 1981.

[62] F. Bassi and S. Rebay. GMRES discontinuous Galerkin solution of the compressible Navier-Stokes equations. In Bernardo Cockburn, George Karniadakis, and Chi-Wang Shu, editors, *Discontinuous Galerkin Methods: Theory, Computation and Applications*, pages 197–208. Springer, Berlin, 2000.

[63] Roger Alexander. Diagonally implicit Runge–Kutta methods for stiff ode's. *SIAM Journal on Numerical Analysis*, 14(6):1006–1021, 1977.

[64] Krzysztof J. Fidkowski and Guodong Chen. Output-based mesh optimization for hybridized and embedded discontinuous Galerkin methods. *International Journal for Numerical Methods in Engineering*, 121(5):867–887, 2019.

[65] Krzysztof J. Fidkowski. A hybridized discontinuous Galerkin method on mapped deforming domains. *Computers and Fluids*, 139(5):80–91, November 2016.

[66] D Hänel, R Schwane, and G Seider. On the accuracy of upwind schemes for the solution of the Navier-Stokes equations. In *8th Computational Fluid Dynamics Conference*, page 1105, 1987.

[67] Masayuki Yano. *An optimization framework for adaptive higher-order discretizations of partial differential equations on anisotropic simplex meshes*. PhD thesis, Massachusetts Institute of Technology, 2012.

[68] Krzysztof Fidkowski. A local sampling approach to anisotropic metric-based mesh optimization. In *54th AIAA Aerospace Sciences Meeting*, page 0835, 2016.

[69] R. Becker and R. Rannacher. An optimal control approach to a posteriori error estimation in finite element methods. In A. Iserles, editor, *Acta Numerica*, pages 1–102. Cambridge University Press, May 2001.

[70] Krzysztof J. Fidkowski and David L. Darmofal. Review of output-based error estimation and mesh adaptation in computational fluid dynamics. *AIAA Journal*, 49(4):673–694, 2011.

[71] Krzysztof J Fidkowski and Philip L Roe. An entropy adjoint approach to mesh refinement. *SIAM Journal on Scientific Computing*, 32(3):1261–1287, 2010.

[72] Houman Borouchaki, Paul-Louis George, Frédéric Hecht, Patrick Laug, and Eric Saltel. *Mailleur bidimensionnel de Delaunay gouverné par une carte de métriques. Partie I: Algorithmes*. PhD thesis, INRIA, 1995.

[73] Steven T Zalesak. The design of Flux-Corrected Transport (FCT) algorithms for structured grids. In *Flux-Corrected Transport*, pages 29–78. Springer, 2005.

[74] Sergei Konstantinovich Godunov. A difference scheme for numerical solution of discontinuous solution of hydrodynamic equations. *Math. Sbornik*, 47:271–306, 1959.

[75] Marco A. Ceze and Krzysztof J. Fidkowski. Output-driven anisotropic mesh adaptation for viscous flows using discrete choice optimization. AIAA Paper 2010-0170, 2010.

[76] Krzysztof J. Fidkowski. Output error estimation strategies for discontinuous Galerkin discretizations of unsteady convection-dominated flows. *International Journal for Numerical Methods in Engineering*, 88(12):1297–1322, 2011.

[77] Yifan Bai and Philip L Roe. Toward physically-based limiting for the active flux scheme. In *AIAA AVIATION 2021 FORUM*, page 2744, 2021.

[78] Yifan Bai and Krzysztof J Fidkowski. Continuous artificial-viscosity shock capturing for hybrid discontinuous galerkin on adapted meshes. *AIAA Journal*, 60(10):5678–5691, 2022.

[79] Krzysztof J Fidkowski. Output-based space–time mesh optimization for unsteady flows using continuous-in-time adjoints. *Journal of Computational Physics*, 341:258–277, 2017.

[80] Paul Woodward and Phillip Colella. The numerical simulation of two-dimensional fluid flow with strong shocks. *Journal of computational physics*, 54(1):115–173, 1984.