Active Flux Schemes

by

Timothy Andrew Eymann

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
2013

Doctoral Committee:

Professor Philip Roe, Chair
Assistant Professor Krzysztof J. Fidkowski
Professor Smadar Karni
Scott Morton, DoD HPCMP CREATE Program

To the memory of my grandfather,

Frank Eymann

who inspired so many in my family to approach flight with a never-ending sense of wonder

ACKNOWLEDGEMENTS

As my friends and family are so fond of pointing out, I have been on this journey quite a long time. I have been extremely fortunate to have so much help along the way. First and foremost, I must thank my advisor, Prof. Phil Roe, for his support and all of the fruitful discussions. He taught me how to think about CFD and was always there with a dash of British *humour* and encouragement when I needed it. I cannot imagine having a better guide these past four and one-half years.

I am also grateful to my committee members Prof. Krzysztof Fidkowski, Prof. Smadar Karni, and Dr. Scott Morton for their insights and contributions to this work. Emeritus professor Bram van Leer deserves a special note of thanks not only for recruiting me to come to Michigan, but also for allowing me to take an idea of his, which was perhaps a little ahead of its time, and run with it. I also thank my masters advisor, Prof. Stephen Ruffin, for introducing me to the field of CFD and tossing me in the deep-end.

None of this would have been possible without funding from the DoD SMART program and the support of my supervisors at Eglin AFB. I am indebted to Sam Revill for hiring me at the Air Force SEEK EAGLE office and for all that he did to help me apply for a SMART fellowship and transition back to school. I thank Joe Keen and Steve Ellison who freed me from my day-to-day duties and allowed me to concentrate on this work. I am especially grateful to Scott Morton, who first planted the Ph.D. seed in my head and supported me along the way even though it meant having an undermanned team for the last few years. He was there for the beginning

of this process, and I am happy to have him on my committee at the end of it.

My time at Michigan would have been far more stressful without all of the great friends I have gained during my time in Ann Arbor. From the trivia nights, which often led to karaoke early-mornings, to the football games and everything in between, these people provided laughs and memories that I am sure will endure long after my Fortran code stops working. I thank all of my past and current office mates and members of our research group for all of the insightful discussions. I would also like to thank Steve Kast and Tyler Lung for providing some of the comparison data for this dissertation.

I am only here today because of the love from my great family. My parents Larry and Judy allowed me to cross the country in pursuit of my academic goals, without ever hesitating to give me their full support. My sisters Julie and Katie were always there to keep me smiling and keep me grounded. I thank my brother Don for blazing a trail through the aerospace industry and leaving behind enough breadcrumbs to get me started down my own path.

Finally, I thank my fiancée Molly for her unwaivering love and support. She, more than anyone, has had to sit through boring explanations of my research to friends and had to sacrifice plans because of an impending deadline or project. And yet, she was always there with a smile, a hug, and sometimes a fresh box of dry-erase markers. Throughout the past few years, she has had an uncanny ability to know exactly what I need to keep going and selflessly provided it. I am profoundly grateful and look forward to starting the next phase of our lives together.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

LIST OF ALGORITHMS

Algorithm

LIST OF APPENDICES

Appendix

LIST OF ABBREVIATIONS

AIAA	American Institute of Aeronautics and Astronautics
AF	active flux
CFD	computational fluid dynamics
DG	discontinuous Galerkin
DOF	degree(s) of freedom
ENO	essentially non-oscillatory
FCT	flux-corrected transport
FD	finite difference
FE	finite element
FV	finite volume
HLL	Harten, Lax, van Leer
PPM	piecewise parabolic method
RD	residual distribution
SD	spectral difference
SV	spectral volume
ULF	upwind leapfrog
WENO	weighted essentially non-oscillatory

ABSTRACT

Active Flux Schemes

by

Timothy A. Eymann

Chair: Philip Roe

This dissertation details the development of active flux schemes, a new class of methods for solving conservation laws. Active flux methods address three issues plaguing production-level computational fluid dynamics (CFD) codes: reliance on one-dimensional Riemann solvers, second-order accuracy, and computational stencils that do not easily parallelize. The key concept is that edge and vertex values are updated and evolved independently from the conserved cell-average quantities. Interface values are then used to calculate fluxes that conservatively update the cell-averages. Because the edge updates do not need to be conservative, any convenient method can be used and proper attention can be given to multidimensional physics. The scheme uses parabolic reconstructions, with a cubic bubble function to maintain conservation in two dimensions, making it third-order accurate by construction. All of the reconstructions and updates are local to a single element, giving AF schemes a very compact stencil suitable for parallelization. Additionally, the AF method is fully discrete, advancing from time-level n to $n+1$ in a single step.

The method is demonstrated on the linear advection, linear acoustics, and linearized Euler equations in one and two dimensions. The AF method has several ad-

vantages over more traditional schemes. For one, the extra degrees of freedom within the cell mean that frequencies up to 2π can be resolved, which is double the frequency range for comparable finite volume (FV) schemes. The AF scheme has superior dissipation and dispersion properties, especially as the Courant number approaches one. Its compact stencil makes the AF solution far less sensitive to irregular meshes than a third-order FV scheme. The AF scheme economically achieves third-order accuracy using two degree(s) of freedom (DOF) per element in one dimension and three DOF in two dimensions. This is comparable to the DOF in a discontinuous Galerkin scheme using linear reconstructions. The AF method achieves third-order accuracy for all of the equation sets using randomized, unstructured meshes. The multidimensional treatment of the acoustics system allows the AF method to preserve excellent symmetry properties on an irregular triangular mesh.

CHAPTER I

Introduction

1.1 Background

Computational methods have been vital tools for the design and analysis of numerous products for decades. Although embraced by a wide range of industrial disciplines, computational fluid dynamics (CFD) has been advanced and matured primarily by two fields: defense and aerospace. These fields are often closely linked and the two place similar demands on engineering requirements. During the design or analysis of defense systems, a premium is placed on the performance of the article. For planning and acquisition purposes, it is important to predict, with a very high degree of accuracy, the yield of a weapon, the flight characteristics of a vehicle [10, 29], or the complex interaction between the vehicle flow field and a store [20, 49]. In civilian aerospace applications, performance is measured by operating cost. One of the primary factors affecting the operating cost of an aircraft is its drag. This is the main motivation behind the American Institute of Aeronautics and Astronautics (AIAA) drag prediction workshops that have focused on developing and communicating CFD best-practices [55]. The pressure from both defense and civilian applications to increase the accuracy of these performance predictions, has led to the current generation of CFD solvers and techniques.

In the early days of CFD, the technical benchmark for a computational method

was its ability to calculate steady, transonic flow around a two-dimensional airfoil. Steadily, the problems grew in complexity to include viscous effects, time-dependent physics and three-dimensional phenomena, including turbulence. Consequently, the computational resources required to perform a CFD simulation grew. Algorithm development proceeded rapidly on two fronts: techniques to handle more complex physics, and methods to decrease the simulation run time. Eventually, efforts to improve simulation efficiency outpaced fundamental algorithm development as industry practitioners understandably focused on meeting production deadlines. At that point the structure of a CFD code became standardized with researchers only investigating changes to modules of the code that were essentially independent. The parallelization of CFD solvers further solidified the basic form of the code because major algorithm changes would require the solver developer to re-tune the application, which required an extensive investment in time. In a sense, it is as if one of the variables of the "CFD system", the code structure, had been held constant while the other, the code's speed, was optimized.

We now face a situation where it is becoming increasingly difficult to solve complex problems because the underpinnings of the code were designed and optimized individually rather than as a system. Today, the industry standard for CFD applications is a second-order accurate finite volume (FV) code, with a one-dimensional Riemann solver to calculate the flux at the cell interface. Despite numerous advances in time-integration algorithms, mesh refinement techniques, and turbulence modeling, just to name a few, most production CFD codes are built around a second-order approximation to one-dimensional physics. Furthermore, changing the core solution algorithm alone is not usually feasible because the time-marching and parallelization has been developed around a specific approach to solving the physics. A new paradigm for solving CFD problems is required to meet future computational challenges. This dissertation proposes a new CFD framework, the active flux (AF) method, as the

means of cutting the Gordian Knot formed by the conflict between the need to solve increasingly complex problems with methods optimized under a rigid code structure.

1.2 Shortcomings of contemporary solvers

The following sections describe some of the most glaring shortcomings of the current class of production-level CFD codes.

1.2.1 Multidimensional physics

Godunov [15] originated modern shock-capturing schemes by modeling the solution between two computational elements as the interaction between the discontinuous representations of the fluid within the cells. The discontinuity at the interface is resolved by solving a one-dimensional Riemann problem. Because the exact solution to the Riemann problem can be computationally expensive, many approximations have been developed. Roe successfully linearized the full Euler wave system [44], while others such as Rusanov [48] and later Harten, Lax, van Leer (HLL) [17] further approximated the system by reducing the number of waves considered. Einfeldt modified the wave speed treatment in the original HLL scheme, leading to the HLLE scheme [11]. All of these approximations, however, preserve the one-dimensional nature of the original method.

To obtain the flow solution for two-dimensional or three-dimensional elements, a one-dimensional Riemann problem is solved normal to each face. This approach works fine for smooth problems, but when there is a shock or other discontinuity in the flow, artificial numerical features can be generated if the shock is not aligned with the mesh. The carbuncle is the most documented and studied of these features [36, 39]. Another common manifestation of the problem is the oscillation of solution values behind an oblique shock that is not aligned with the mesh [52].

Several fixes have been proposed to eliminate carbuncles and other solution irreg-

ularities around shocks. Reducing the number of waves from three to two (HLL) or one (Rusanov) significantly reduces the occurrence of carbuncle-type behavior, but these solvers add excess dissipation that degrades the solution in regions such as shear and boundary layers. A common approach then is to use some combination of reduced-wave models and the full wave approximation. Park and Kwon [35] proposed a pressure switch that transitions between the HLLE Riemann solver and Roe's approximate Riemann solver, with Tremel [52] demonstrating the method using the structured CFD solver OVERFLOW. Another method of mitigating the effects of a purely one-dimensional strategy is the rotated Riemann approach [9, 24, 31, 42] where the velocity vector at the cell interface is decomposed and two Riemann problems are solved. These techniques are effective and can be computationally efficient, but they were designed under the constraint that they must work within the current framework adopted by most industrial solvers. They fix the symptoms of the problem, but do little to address the fact that the solvers are built around one-dimensional physics.

There have been previous attempts to create solvers that do not rely on one-dimensional Riemann solves. One obvious motivation for this is when the problem physics are omnidirectional, as with the acoustics equations. A common approach for two dimensions is to find the update at a point using bicharacteristics of the solution [5, 34, 41]. The main drawbacks to this approach are that they involve integrals that may be difficult to approximate and that they are, at best, second-order accurate. Chapter IV contains a more detailed discussion of bicharacteristic methods for the acoustic equations.

Fluctuation splitting methods are an important class of schemes closely related to this work. They offer the most complete framework for solving multidimensional fluid problems without relying on one-dimensional Riemann solves at interfaces. The concept was outlined for linear advection by Roe [45, 46] and later extended to hyperbolic/elliptic systems by Rad [40]. In this approach, the solution residual is eval-

uated over a triangular element, and then it is distributed to the nodes according to the upwind direction if solving an advection-type equation or distributed using a least-squares method if solving Cauchy-Riemann-type problems. These residual distribution (RD) schemes have been demonstrated to achieve third-order accuracy and nearly recover potential flow around elliptical bodies [32]. The RD schemes very nearly solve many of the problems associated with finite-volume schemes; however, they do lack some features required of production-level codes. Most, but not all of, the concepts translate easily to three-dimensional flows, severely limiting the application of the RD method. Also, RD schemes are formulated chiefly for solving steady-state problems, meaning some type of pseudo-time stepping is employed for unsteady cases [1, 43]. This increases the computational overhead of the scheme. Nonetheless, this class of schemes provided much of the inspiration for the framework proposed in this dissertation.

1.2.2 Accuracy

Second-order computational methods commonly employed in production codes, while sufficient for many applications, lack the accuracy required to perform certain simulations on affordable meshes, such as those that require high-fidelity vortex tracking and computational aeroacoustics. Clearly, to increase the accuracy of a scheme, one must address the error generated by the scheme. There are two primary means of reducing the error: minimize sources of error within the method and increase the order of accuracy of the scheme. The spatial order of accuracy determines the rate at which the scheme converges to the exact solution as the mesh is refined. As the order of a scheme increases, mesh refinement becomes a more effective tool to drive error out of the computed solution.

Increasing the order of accuracy of a scheme is the more traditional approach to improving the error properties of a scheme but one may also reduce the baseline

level of error produced by a computational method by carefully representing and interpolating the data on a space-time mesh. For example, it is possible to create dissipation-free, second-order schemes by including an additional time level in the stencil. These dissipation-free methods were motivated by the need to track flow features accurately over long distances and simulation times. Iserles [19] and Roe [47] investigated upwind leapfrog (ULF) schemes, which are two-level schemes that achieve zero dissipation due to their symmetric stencils in space and time. ULF methods were also thoroughly investigated by Thomas [50] who applied the idea to multiple dimensions and systems of equations. Using an additional time level in the scheme provides additional degree(s) of freedom (DOF) that can also be used to design optimum monotone schemes or minimize dispersion. Appendix A contains a discussion of how to develop multi-level finite difference (FD) schemes that achieve desired properties.

Two time level schemes can be effective at eliminating various sources of error, but they all require one initialization step before there is enough data to apply the two-level scheme. The CABARET scheme presented by Karabasov [21] collapses the two-level ULF to a single-step scheme by independently updating cell-averages and flux values. The scheme retains all the advantages of the ULF scheme while only using data from one time level. Independently updating the conserved cell-average values and flux values is a key concept that we utilize throughout this work. In fact, the CABARET scheme can be thought of as a second-order version of the third-order technique developed in this dissertation.

As stated earlier, the more common way of managing the error produced by a scheme is to increase the order of accuracy and refine the mesh. There is some debate about the optimum order of accuracy for a scheme. Methods with even orders of accuracy behave very differently at sharp features or discontinuities than methods with odd orders of accuracy. Fourier analysis allows us to represent the error as a

superposition of waves. From this type of analysis, we can categorize two types of error. Dissipation error, also known as diffusion error, is error associated with the amplitude of the waves and dispersion error indicates a misrepresentation of the wave speed. Modified equation analysis [18] for the advection equation reveals that the leading error term, which dominates and dictates the scheme's behavior, is dispersive for even-ordered schemes and dissipative for odd-ordered schemes. Even-order schemes tend to misrepresent the speed of waves while odd-order schemes misrepresent the wave amplitude. Because a sharp discontinuity is approximated by the scheme as the summation of several waves with varying frequency and amplitude, the differences between odd and even ordered schemes can be attributed to their leading error terms. The different leading error terms have the effect of leading to more oscillatory behavior for even-ordered schemes than odd-ordered schemes [4]. Practically, this means that there is a large advantage when increasing from second to third-order because there is less damping required to mitigate the effect of the oscillations. There is little incentive for increasing to a fourth-order scheme because that would bring back the oscillations. One could argue for skipping fourth-order and going straight for a fifth-order scheme, but the work required for a scheme increases dramatically as the order is increased [26]. For this reason, we focus on third-order accuracy.

One method for increasing the order of accuracy of a finite volume scheme is to expand the computational stencil, generating higher-order reconstructions from which the solution may be interpolated. An early third-order scheme using this idea was developed by Warming, Kutler, and Lomax [58]. Later, essentially non-oscillatory (ENO) [16] and weighted essentially non-oscillatory (WENO) [25] schemes were developed to obtain higher-order accuracy without generating spurious oscillations in the solution. WENO and ENO schemes perform the higher-order reconstructions by searching through a set of candidate stencils and either selecting the smoothest reconstruction or using a weighted average of the possible choices. This

approach is most straightforward to apply on a structured mesh, but it is also applicable to unstructured meshes [33, 14]. A major drawback of increasing accuracy by enlarging the computational stencil is that the solver attempts to build a high order reconstruction using data from regions of the flow that may not be physically relevant. Another is that synchronizing large stencils during parallel computations requires extremely careful implementation to minimize the communication overhead between processes. Because FV schemes only allow one DOF per element, increasing the accuracy using a compact stencil requires a different solution strategy.

1.2.3 Compact support

The combined need for higher accuracy and increased parallelization has driven massive interest in methods that achieve higher-order accuracy by increasing the internal degrees of freedom within a cell. In the late 1980s and 1990s, researchers began to employ finite element (FE) methods for solving fluids problems. Over time, these evolved into schemes such as the discontinuous Galerkin (DG) [6], spectral difference (SD) [57], and spectral volume (SV) method [56]. These schemes are well suited for parallel computations on unstructured meshes since they all are capable of generating higher-order reconstructions without extending the computational stencil beyond the neighboring cells. DG schemes are the most popular FE-type scheme within the research community. While a great deal of effort has been concentrated on DG schemes, there are two classes of problems preventing their widespread adoption for industrial applications. The first is a lack of robustness [2], which is closely related to the absence of sufficient limiters for nonlinear problems [22]. The second, and more fundamental issue, is their high memory requirement and computational expense. A DG scheme stores enough DOF within each cell to reconstruct basis functions of arbitrary order. Advancing the solution in time requires calculating the time-dependent coefficients for the basis functions, which involves inverting large matrices. A recent

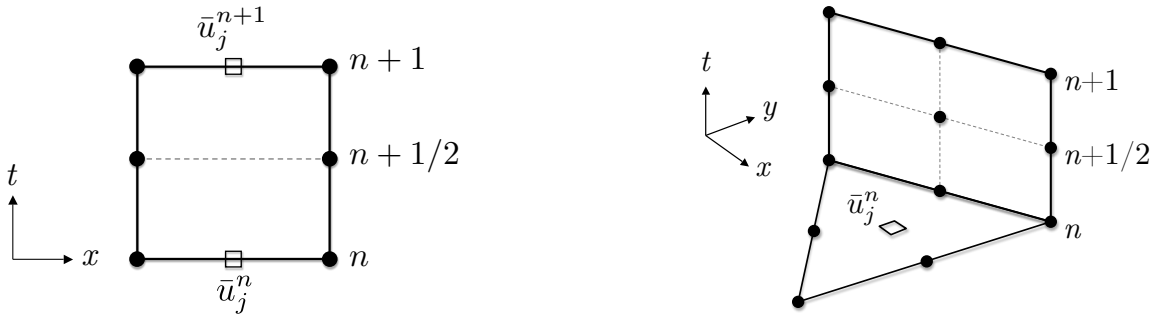
variant of the DG method, hybridizable discontinuous Galerkin [37] schemes, has attempted to reduce the memory overhead by sharing some DOFs between cells, but the formulation still leaves independent DOFs at the mesh vertices, precisely where there is most to gain from sharing between elements. Recognizing this deficiency, embedded discontinuous Galerkin methods [38] are formulated such that data are also shared at mesh vertices. These DG variants resolve the issue of repeating the DOF within elements, yet both hybirdizable and embedded discontinuous Galerkin schemes require implicit time marching, and thus incur a higher computational overhead than the single-step, explicit AF method.

Before FE approaches dominated the research involving compact stencils, the issue of generating higher-order solutions from local data was recognized and addressed by van Leer [53]. In his 1977 paper, he presented a series of linear and nonlinear advection schemes that evolved not only the cell-average value but also a solution gradient or edge value. He then used the extra degrees of freedom to increase the accuracy of the solution. A key aspect of the scheme was that the solution reconstruction he used was C^0 continuous at the cell interface, meaning DOF could be shared between elements. Furthermore, the continuous solution meant that finding the interface value did not require solving a Riemann problem. Despite these advantages, the prevailing thought at the time was that storing edge values in addition to conserved quantities and solving for each independently was simply too expensive for the computers of the era. The idea was largely abandoned although aspects of it later showed up in schemes like the piecewise parabolic method (PPM) [7]. Thirty-five years later, we can now look back and see that the method he simply referred to as Scheme V has many of the features that are lacking in modern production codes. The independent edge updates, third-order accuracy, and compact stencil provide a very attractive foundation around which to build a new computational method.

1.3 Active flux description

Active flux schemes provide a flexible and powerful framework for solving conservation laws. The name of the scheme is a direct reference to the fact that interface values are updated independently from conserved quantities. In a traditional scheme, the flux at an interface is determined by the solution to a Riemann problem using reconstructions or interpolations of conserved variables as the input. We refer to this type of update as a *passive flux* because the interface quantity is derived from conserved quantities. An *active flux* is computed directly from edge values in a way that depends both on previous cell values and previous edge values. Importantly, the interface update does not need to be conservative. The only requirement on fluxes is that they are consistent, so any convenient method can be used to generate a point update. As long as we consistently generate flux values from the interface data, we are able to obtain a conservative scheme. Therefore, the freedom to choose an edge update method allows us to build in the appropriate multidimensional physics rather than limit ourselves to solving a one-dimensional Riemann problem. This powerful idea opens a wide range of new possibilities for solving conservation laws.

The active flux scheme is third-order accurate by construction. The internal data are represented by parabolic functions, meaning the scheme will be exact for quadratic data, provided the edge fluxes are calculated to third-order accuracy. Consequently, AF schemes have less severe oscillations around flow discontinuities than typical second-order FV schemes. Additionally, AF schemes do not use data external to the element to update quantities within the cell, so they have a very compact stencil. Another advantage of active flux schemes is that they store the data required to define and update a given reconstruction very economically. Figure 1.1 illustrates the data storage for a one-dimensional and two-dimensional elements. Cell-averages are stored at the centroid of each cell with point values at face centroids and vertices. Because cells share data between faces and vertices, the scheme is able to achieve



(a) One dimension

(b) Two dimensions

Figure 1.1: Data storage locations for AF scheme

savings over other higher-order approaches. These savings become even more apparent in higher dimensions. In one dimension, active flux schemes require two DOF for third-order accuracy, and in either two or three dimensions, the AF method requires approximately three DOF per cell for the same level of accuracy. This is a large memory savings over other compact schemes such as DG.

Regardless of the conservation law, the AF method has the same basic procedure. Point values are updated at the interfaces, then average fluxes are constructed from the point-data and used to advance the conserved cell-average. Algorithm 1.1 lists how this update procedure requires one loop through the cells and one loop through the faces in the domain. The only difference between conservation laws is the multidimensional method used to update the interface values and how those interface values are used to calculate a flux. It is important to note that the AF method is a single-step, fully discrete scheme, meaning that the solution improves as the time step approaches its theoretical maximum. The AF scheme is also time-accurate and inherently capable of solving unsteady flows.

1.4 Thesis overview

The objective of this dissertation is to develop active flux schemes for linear model problems used to study the elemental behavior of fluids. It also includes a brief

Algorithm 1.1 Core active flux algorithm

for cell = 1, cells in domain **do**
  Calculate point-updates at flux quadrature points (defined in Chapter II)
end for
for face = 1, faces in domain **do**
  Calculate the average flux
  Add/subtract the proper residual value to the neighboring cells
end for
Store $n + 1$ values as n values for next time step

discussion of nonlinear equations and limiting. Throughout this work, the emphasis is placed on strictly implementing the theory rather than maximizing computational efficiency. I have indicated possible shortcuts/approximations wherever possible.

Chapter II introduces the AF discretization and implements the method for the familiar linear advection equation. Nonlinear extensions are discussed in Chapter III as well as a novel approach to solution limiting that maintains the compact stencil of the scheme. The truly multidimensional nature of the method is illustrated in Chapter IV, which details the AF method applied to the linear acoustics equation. That chapter also demonstrates how operator splitting can be used to combine different types of AF solves for more complex equations. Chapter V provides a summary of the work and details the major contributions to the field. The dissertation concludes with suggestions for future avenues of research.

CHAPTER II

Linear Advection

This chapter develops the active flux method for the scalar advection model equation given in Eq. (2.1).

$$\frac{\partial u}{\partial t} + \nabla \cdot (u\boldsymbol{\lambda}) = 0 \tag{2.1}$$

The two-dimensional conservation form is:

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} + \frac{\partial g(u)}{\partial y} = 0 \tag{2.2}$$

where $f(u) = au$ and $g(u) = bu$. In general, a and b, which represent the components of the wave speed $\boldsymbol{\lambda}$, may be a function of the spatial coordinates x and y. We'll begin with the one-dimensional forumulation where $b = 0$. This simplest case will be used to highlight the active flux scheme's error properties and grid independence. We then increase the complexity by developing the two-dimensional discretization and demonstrate the method for cases when the two-dimensional wave speed is constant throughout the domain and the cirucular advection case for which the wave speed is a function of the distance from the center of rotation.

2.1 One-dimensional discretization

2.1.1 Reference element

We seek a general representation of the solution within a given cell j, illustrated in Fig. 2.1, which has a width Δx. Working with nondimensional coordinates simplifies the definition of the solution within an element. The cell coordinates are normalized such that the left $(j-1/2)$ edge is at $\xi = 0$ and the right edge is $\xi = 1$. We then have the following mapping from physical coordinates to reference coordinates:

$$\begin{aligned} x &= x_{j-1/2} + \xi \Delta x \\ \xi &= \frac{x - x_{j-1/2}}{\Delta x} \end{aligned} \tag{2.3}$$

The solution to the hyperbolic partial differential equation (PDE) Eq. (2.1) is constant along special curves called characteristics. For one-dimensional linear advection, the characteristics are straight lines with a slope of $1/a$ on a x-t plot. Therefore, we can find the solution at any interface coordinate ξ_i and time t by tracing the characteristic back in time to a point when the solution is known. We define this point as the characteristic origin ξ_0. The conservation law tells us that the change in the solution between any two points is due to the flux through the end points [23]. The case for $a > 0$ is illustrated in Fig. 2.1. Here, the solution between ξ_0 and ξ_i changes as a result of the flux through ξ_i. The region is bounded by the solution characteristic originating at ξ_0 and terminating at (ξ_i, t). Note that this is a general illustration of the concept. In practice, the interface coordinate ξ_i will either be the left $(\xi = 0)$ or right $(\xi = 1)$ interface.

2.1.2 Internal reconstruction

The accuracy of the vertex update and average flux calculation is determined by the representation of the solution within the cell, i.e. the internal reconstruction. By

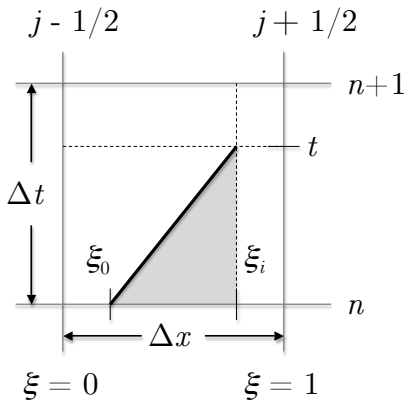


Figure 2.1: Conserved region for cell j

choosing parabolic representations, we guarantee the scheme is exact for quadratic data, and achieve third-order spatial accuracy. A quadratic is uniquely determined by three pieces of information. The cell averages represent the conserved quantity in each element, so this is a natural choice for one of the quadratic parameters. The left interface value and right interface value provide the other two degrees of freedom. The resulting quadratic can be written in several equivalent forms. We choose to follow the standard finite element convention and represent the solution as a summation of basis functions.

$$u(\xi) = \sum_{i=1}^{3} c_i \phi_i(\xi) \tag{2.4}$$

The indices are ordered left to right so that index 1 corresponds to the $j - 1/2$ interface, 2 the midpoint, and 3 the $j + 1/2$ interface. Standard one-dimensional, Lagrange basis functions are used, valid over the interval $\xi \in [0, 1]$. The coefficients and basis functions defining the reconstruction are listed in Table 2.1.

Table 2.1: Basis functions and coefficients for 1D reconstruction

Index	c_i	ϕ_i
1	$u_{j-1/2}$	$(2\xi - 1)(\xi - 1)$
2	$\frac{1}{4}\left(6\bar{u}_j^n - u_{j-1/2}^n - u_{j+1/2}^n\right)$	$4\xi(1 - \xi)$
3	$u_{j+1/2}$	$\xi(2\xi - 1)$

2.1.3 Updates

Recall that in an active flux scheme, the cell averages and edge point values are independently updated. The conserved quantities are evolved through flux functions constructed from the edge values at different time levels. A key advantage of the AF method is that the point updates do not need to be conservative, so any convenient method can be used. Our model problem is simple enough that there is a closed form for the origin of the solution characteristic passing through the interface ξ_i at time t. The linear nature of the characteristics for linear advection means the following relationship is valid:

$$at = (\xi_i - \xi_0)\,\Delta x \tag{2.5}$$

Rearranging to solve for the characteristic origin:

$$\xi_0 = \xi_i - \frac{at}{\Delta x} \tag{2.6}$$

The updated vertex value is simply $u(\xi_0)$.

There are multiple ways to calculate the flux. We need only choose the most convenient form that meets our requirements. One method is to find the exact flux expression by integrating the conservation law over the shaded, triangular region in Fig. 2.1.

$$\Delta x \int_{\xi_0}^{\xi_i} [u(\xi) - u(\xi_0)]\, d\xi = \int_0^{\Delta t} [f(t) - f(u(\xi_0))]\, dt \tag{2.7}$$

Rearranging Eq. (2.7), and dividing by the time interval leads to a general expression for the average flux, $\bar{f}$, at the interface ξ_i.

$$\bar{f} = \frac{1}{\Delta t} \int_0^{\Delta t} f(t) dt = \frac{\Delta x}{\Delta t} \left[\int_{\xi_0}^{\xi_i} u(\xi) d\xi - (\xi_i - \xi_0)\, u(\xi_0) \right] + f(u(\xi_0)) \tag{2.8}$$

Substitution of Eq. (2.6) into Eq. (2.8) results in the familiar result that the average flux in time is equal to the integration of the solution in space over the domain of influence.

$$\bar{f} = \frac{\Delta x}{\Delta t} \int_{\xi_0}^{\xi_i} u(\xi)\ d\xi \tag{2.9}$$

The average flux may also be determined by employing a sufficiently accurate numerical integration technique, such as Simpson's rule, which is exact for our quadratic representation of the solution. In this case, we need both the ξ_0^{n+1} value that intersects the interface at $t = \Delta t$ and an intermediate value $\xi_0^{n+1/2}$ value that crosses the interface at $\Delta t/2$. Once the two characteristic origins have been determined, the average flux is

$$\bar{f} = \frac{1}{6}\left[f(u(\xi_i)) + 4f(u(\xi_0^{n+1/2})) + f(u(\xi_0^{n+1}))\right] + \mathcal{O}(\Delta t^4) \tag{2.10}$$

In this case, both forms of the flux expression are mathematically equivalent, but Eq. (2.10) lends itself more easily to a general algorithm because a single function can compute the flux for various conservation equations if the basis coefficients and characteristic origin are available.

2.1.4 One-dimensional analysis and results

For one-dimensional linear advection, the characteristic origin is a function of the Courant number $\nu = a\Delta t/\Delta x$. The expressions below give the characteristic origin at the $n + 1/2$ and $n + 1$ time levels.

$$\xi_0^{n+1/2} = \xi_i - \frac{\nu}{2} \tag{2.11}$$

$$\xi_0^{n+1} = \xi_i - \nu \tag{2.12}$$

When the wave speed is positive, we only need to consider the right interface ($\xi_i = 1$) in every cell. As expected, we recover van Leer's original vertex update equation for Scheme V [53] by substituting Eq. (2.12) into the one-dimensional Lagrange basis functions in Table 2.1.

$$u_{j+1/2}^{n+1} = \nu\left(3\nu - 2\right) u_{j-1/2}^{n} + 6\nu\left(1 - \nu\right) \bar{u}_{j}^{n} + \left(1 - \nu\right)\left(1 - 3\nu\right) u_{j+1/2}^{n} \tag{2.13}$$

The flux function can be calculated from Eq. (2.10):

$$\bar{f}_{j+1/2} = a\left[\nu\left(\nu - 1\right) u_{j-1/2}^{n} + \nu\left(3 - 2\nu\right) \bar{u}_{j}^{n} + \left(1 - \nu\right)^{2} u_{j+1/2}^{n}\right] \tag{2.14}$$

The cell averages are updated conservatively from the flux functions:

$$\bar{u}_{j}^{n+1} = \bar{u}_{j}^{n} - \frac{\Delta t}{\Delta x}\left(\bar{f}_{j+1/2} - \bar{f}_{j-1/2}\right) \tag{2.15}$$

Substituting the flux functions for the left and right interface into Eq. (2.15) results in the full update for the conserved quantity.

$$\begin{aligned} \bar{u}_{j}^{n+1} = {} & \nu^{2}\left(\nu - 1\right) u_{j-3/2}^{n} + \nu^{2}\left(3 - 2\nu\right) \bar{u}_{j-1}^{n} + \nu\left(1 - \nu\right) u_{j-1/2}^{n} \\ & + \left(1 - \nu\right)^{2}\left(1 + 2\nu\right) \bar{u}_{j}^{n} - \nu\left(1 - \nu\right)^{2} u_{j+1/2}^{n} \end{aligned} \tag{2.16}$$

The update equations Eq. (2.13-2.16) reveal several notable properties of the scheme. First, it is obvious that the vertex update $u_{j+1/2}^{n+1}$ is exact for Courant numbers of $\nu = 1$ and $\nu = 0$. Second, the additional degrees of freedom within each cell double the frequencies that can be resolved by the scheme compared to standard finite-volume methods.

The independent updates for the flux values and conserved variable mean that the amplification factor, derived from a Fourier analysis of the method, is now a matrix

rather than a scalar.

$$\begin{pmatrix} \bar{u}_j \\ u_{j+1/2} \end{pmatrix}^{n+1} = \mathbf{G} \begin{pmatrix} \bar{u}_j \\ u_{j+1/2} \end{pmatrix}^{n} \tag{2.17}$$

We repeat the amplification matrix reported by van Leer for Scheme V[53]:

$$\begin{aligned} \mathbf{G} &= \begin{pmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{pmatrix} \\ g_{11} &= 1 + e^{-i\phi}\left(1 - e^{i\phi}\right)\nu^2(3 - 2\nu) \\ g_{12} &= e^{-2i\phi}\left(1 - e^{i\phi}\right)(1 - \nu)\nu\left[e^{i\phi}(1 - \nu) - \nu\right] \\ g_{21} &= 6\nu(1 - \nu) \\ g_{22} &= (1 - 2\nu) - \nu(2 - 3\nu)\left(1 + e^{-i\phi}\right) \end{aligned} \tag{2.18}$$

The two eigenvalues of this matrix correspond to the damping on the true solution and spurious mode of the scheme. Because the spurious mode is quickly damped, plots of the eigenvalue corresponding to the true solution reveal the diffusion and dispersion properties of the scheme.

We compare the AF scheme to a third-order FV scheme of Warming, Kutler, and Lomax [58] to highlight the advantages of the scheme over traditional finite volume methods. The Warming, Kutler, Lomax (WKL) scheme uses a larger stencil to achieve the same order of accuracy.

$$\begin{aligned} \bar{u}_j^{n+1} = \bar{u}_j^n - \frac{\nu}{2}(\bar{u}_{j+1}^n - \bar{u}_{j-1}^n) + \frac{\nu^2}{2}(\bar{u}_{j+1}^n - 2\bar{u}_j^n + \bar{u}_{j-1}^n) \\ + \frac{\nu(1 - \nu^2)}{6}(-\bar{u}_{j-2}^n + 3\bar{u}_{j-1}^n - 3\bar{u}_j^n + \bar{u}_{j+1}^n) \end{aligned} \tag{2.19}$$

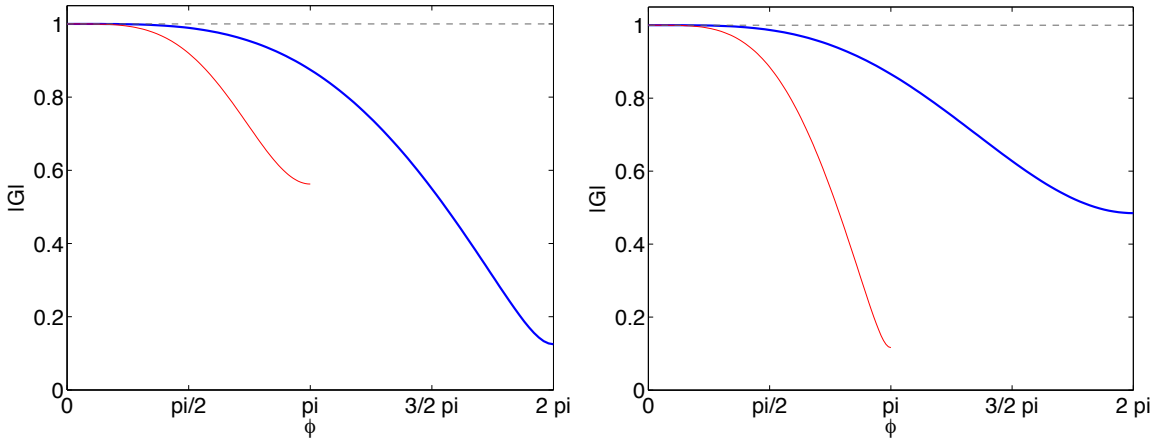
The amplification factor for the WKL scheme is:

$$\begin{aligned} g_{\mathrm{WKL}} = 1 &- \frac{1}{2}\nu - \nu^2 + \frac{1}{2}\nu^3 \\ &- \left[\frac{1}{6}\nu\left(\nu-1\right) + \frac{1}{2}\nu\left(\nu+1\right)\right]\left(\nu-2\right)\cos\phi \\ &+ \frac{1}{6}\nu\left(\nu-1\right)\left(\nu+1\right)\cos 2\phi \\ &- i\left\{\left[\frac{1}{6}\nu\left(\nu-1\right) - \frac{1}{2}\nu\left(\nu+1\right)\right]\left(\nu-2\right)\sin\phi \right. \\ &\qquad \left. + \frac{1}{6}\nu\left(\nu-1\right)\left(\nu+1\right)\sin 2\phi\right\} \end{aligned} \tag{2.20}$$

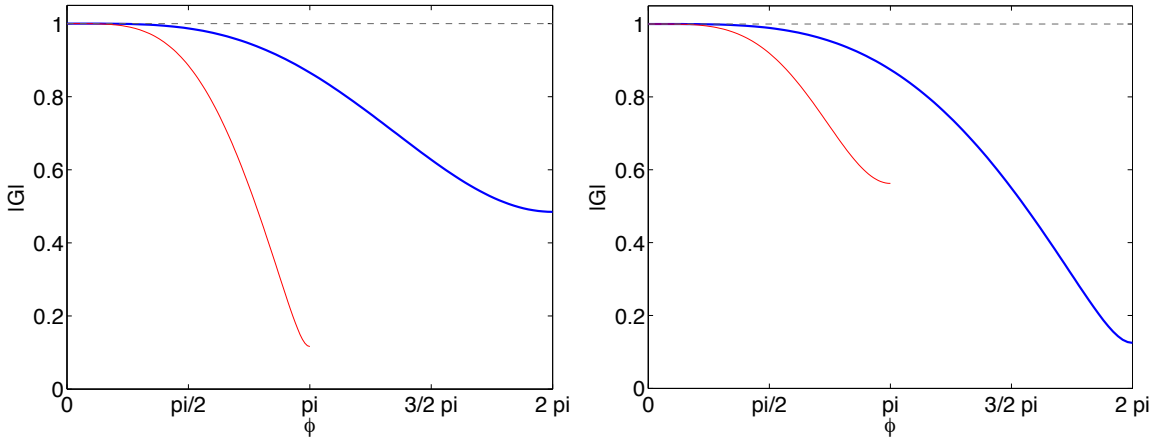
Figure 2.2 shows that the AF scheme has a much lower amount of diffusion than a standard third-order FV scheme at all frequencies. Note that the additional degrees of freedom within the cell allows the AF scheme to resolve frequencies up to $\phi = 2\pi$, compared to a maximum frequency of $\phi = \pi$ for the FV scheme. The AF scheme also represents the wave speeds more faithfully than the finite volume scheme, as illustrated in Fig. 2.3 where values close to zero indicate no error.

The amplification factor for a scheme, from which the diffusion and dispersion factors are derived, indicate how a scheme behaves from one iteration to the next, but a more practical method to evaluate a scheme is to examine the level of dissipation after a fixed amount of n iterations. The number of iterations is determined by calculating the time t required for the wave to move m times its wave length. We can further parameterize the problem by specifying the number of grid points per wave length as k. Thus the distance traveled by the wave is $l = mk\Delta x$. The required time to travel this distance is determined by the wave speed a, and dividing the distance expression by the speed gives us the time $t = l/a = mk\Delta x/a$. Finally, the number of iterations is found by dividing this time by the time step Δt. The final expression for the dissipation after our parameterized time is given below.

$$|g|^n = |g|^{\frac{mk\Delta x}{a\Delta t}} = |g|^{\frac{mk}{\nu}} \tag{2.21}$$



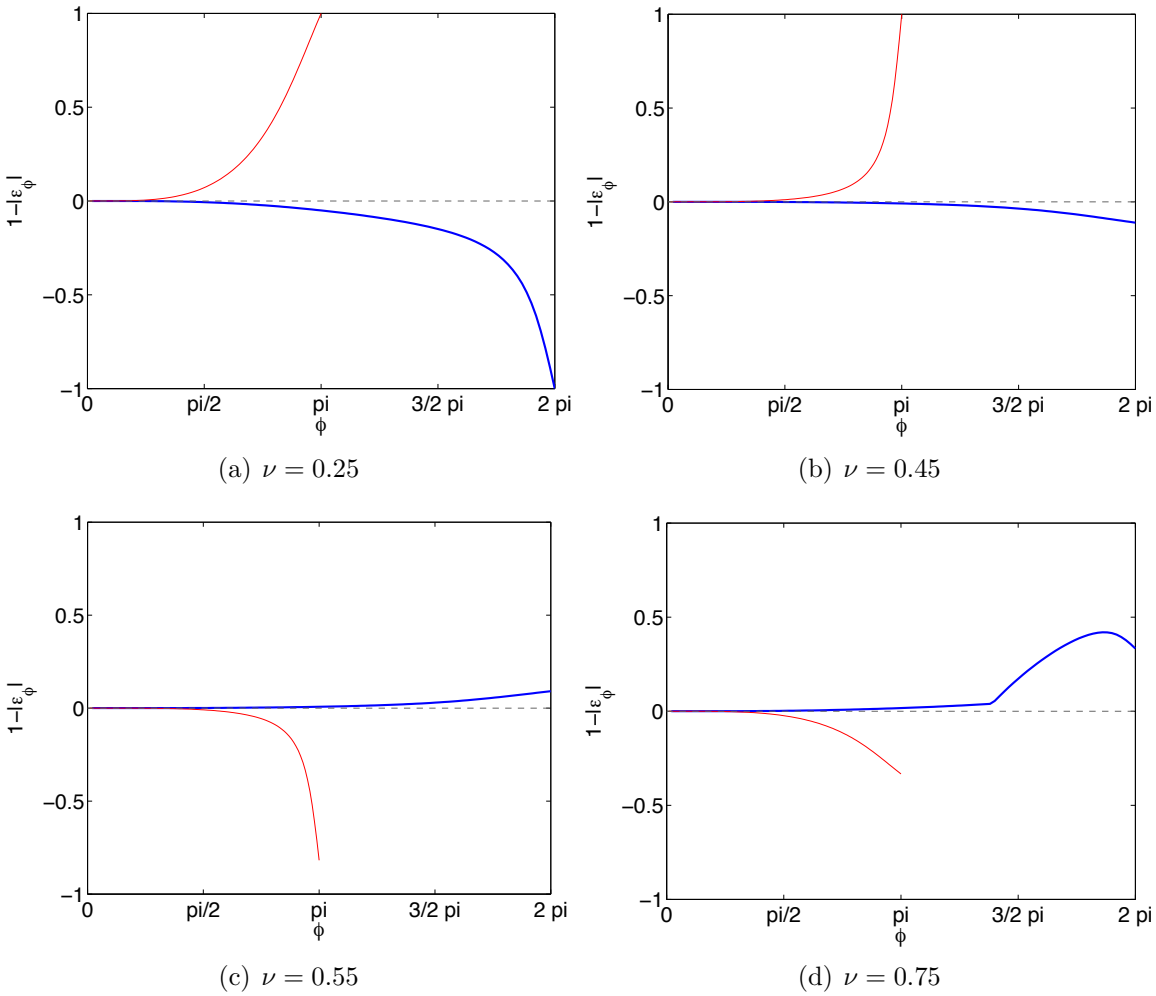
(a) $\nu = 0.25$ (b) $\nu = 0.45$



(c) $\nu = 0.55$ (d) $\nu = 0.75$

Figure 2.2: Diffusion comparison between AF and third-order FV scheme

We set our target distance for the wave to travel as ten times its own wave length. Twelve points per wavelength is generally accepted among industry practitioners of CFD to be the number of points required to resolve flow field features. Prior analysis by Thomas, who showed that second and third-order FV schemes required 11 to 18 points per wavelength to compute a solution with less than 1% dissipation at a Courant number of 3/4, is consistent with this observation [50]. A distance value of ten times the wavelength with a resolution of twelve points per wavelength makes the exponent on the amplification factor $120/\nu$. Figure 2.4 plots the AF dissipation for these parameters against the third-order FV scheme as well as the second-order



(a) $\nu = 0.25$ (b) $\nu = 0.45$

(c) $\nu = 0.55$ (d) $\nu = 0.75$

Figure 2.3: Dispersion comparison between AF and third-order FV scheme

Lax-Wendroff (LW) scheme. The figure shows the AF scheme has a much lower level of dissipation than the FV scheme for all Courant numbers and also outperforms the LW scheme over all but the lowest Courant numbers.

Another property of the AF scheme is that its compact stencil makes it more insensitive to mesh sizing irregularities than schemes that use more points in their update formula. This is clearly illustrated in Fig. 2.5, which compares the third-order FV and AF solutions for a Gaussian pulse after one cycle through the mesh. The nodal locations were randomly perturbed by $e_j h$, where e_j is a random variable, uniformly distributed from $[-d, d]$, and h is the baseline mesh spacing of 0.0125. As

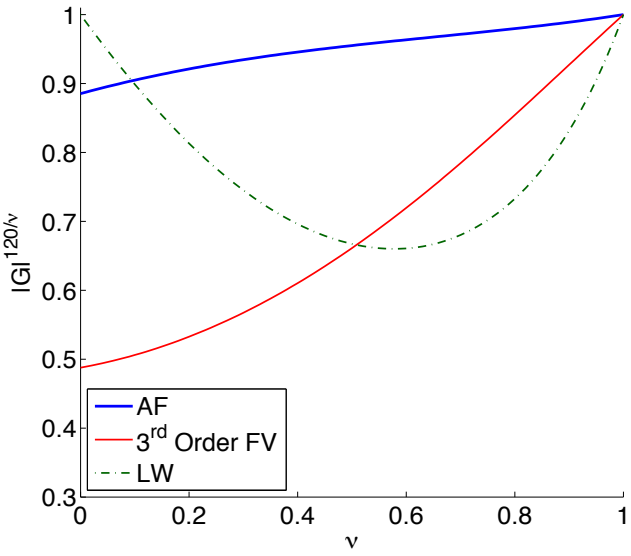


Figure 2.4: Dissipation of sine wave with frequency $\phi = \pi/6$, represented by twelve cells after traveling ten wave lengths

the perturbation increases, the ratio of minimum to maximum cell spacings increases. The figure shows that the AF method has much less dissipation on the random meshes than the FV scheme and that the maximum peak loss is only 0.82% for the AF scheme compared to 27% for the WKL scheme.

The AF scheme performs similarly well when the initial solution is discontinuous. Figure 2.6 shows the AF solutions on a uniform and randomized mesh, where Zalesak's waveform was used as the initial condition. The wave consists of a square wave, a raised cosine, a Gaussian, and a half-ellipse, all of which pose different challenges for an advection scheme. The figure shows that there are only minor differences between the uniform and random mesh, even for this extreme case where the deviation is $\pm 31/64h$.

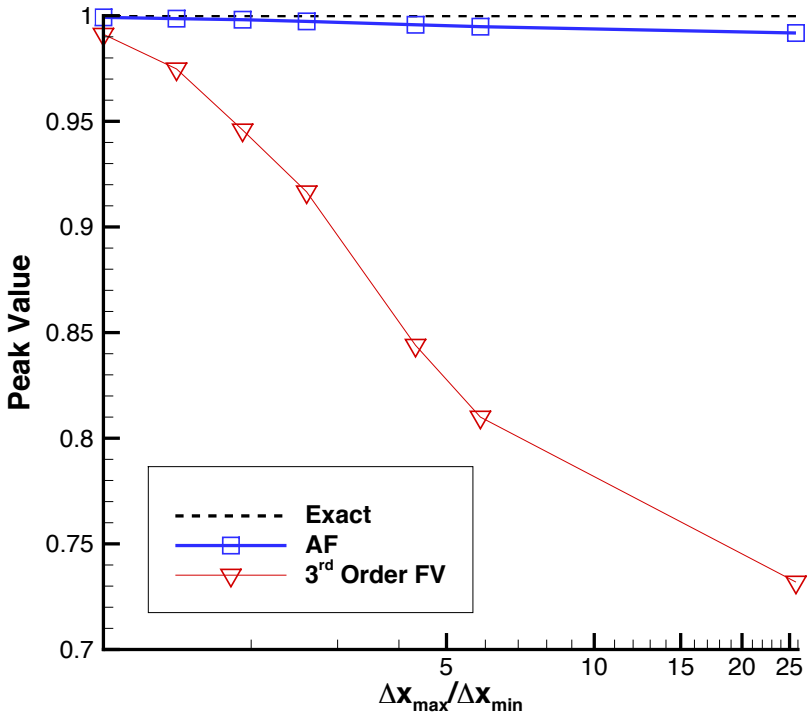
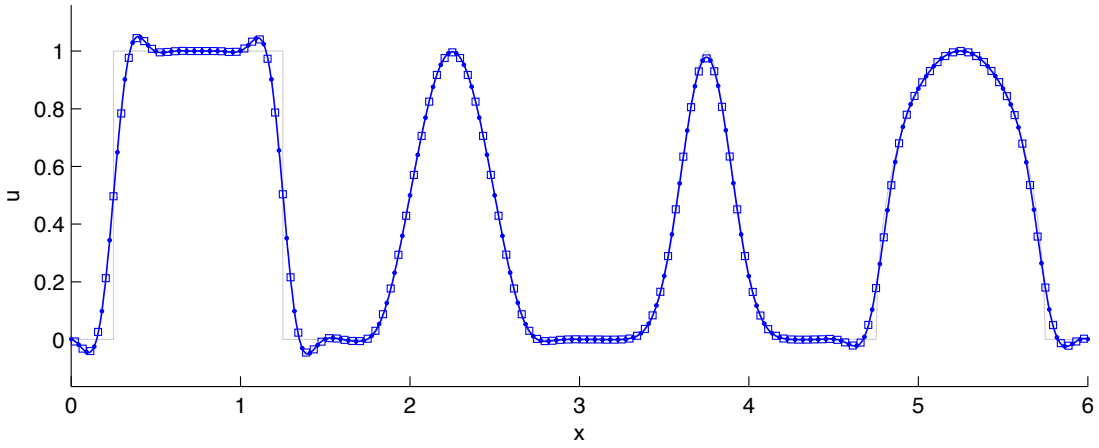
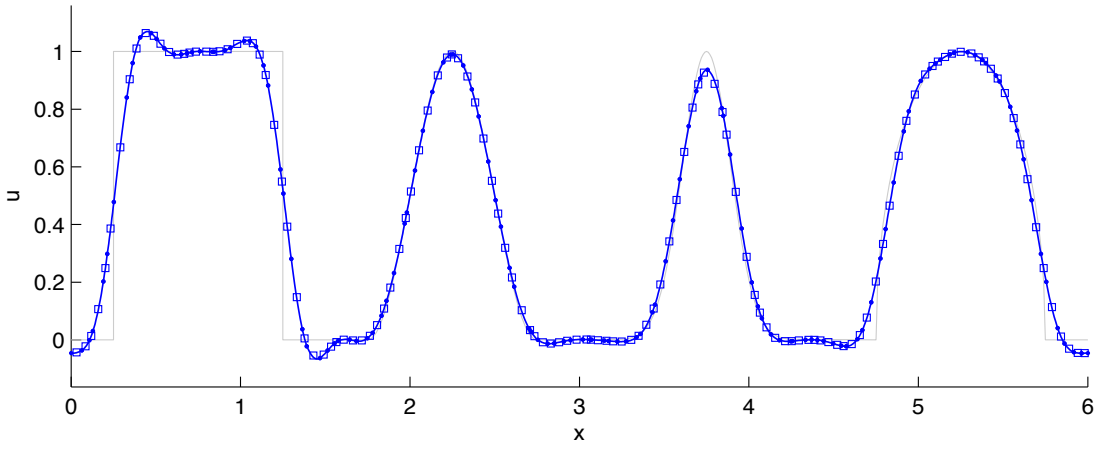


Figure 2.5: Gaussian peak value after one cycle through randomly-spaced mesh



(a) Uniform mesh



(b) Random mesh

Figure 2.6: Zalesak waveform after one cycle through mesh

2.2 Two-dimensional discretization

2.2.1 Reference element

In two dimensions, the mapping from physical space to reference space is slightly more complex. Any point $\mathbf{x}$ in the plane of the triangle can be expressed as a combination of the vectors forming the element edges. Assuming counterclockwise ordering of the nodes, we can write:

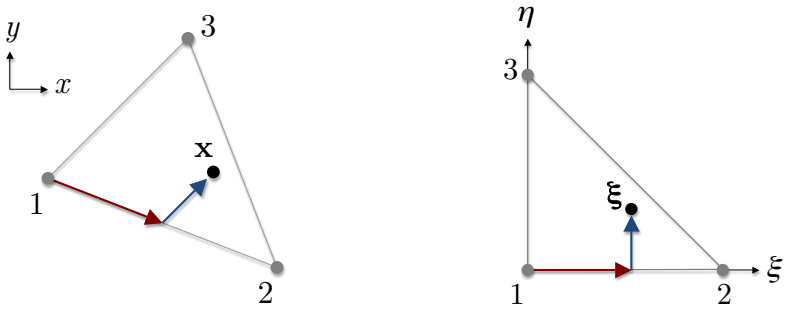
$$\mathbf{x} = \mathbf{x}_1 + \xi\left(\mathbf{x}_2 - \mathbf{x}_1\right) + \eta\left(\mathbf{x}_3 - \mathbf{x}_1\right) \tag{2.22}$$

Or rewriting as matrix:

$$\mathbf{x} = \mathbf{x}_1 + \mathbf{J}\boldsymbol{\xi} \tag{2.23}$$

where the Jacobian matrix is defined:

$$\mathbf{J} = \begin{pmatrix} x_2 - x_1 & x_3 - x_1 \\ y_2 - y_1 & y_3 - y_1 \end{pmatrix} \tag{2.24}$$



(a) Physical space (b) Reference space

Figure 2.7: Element mapping

2.2.2 Reconstruction

The six degrees of freedom provided by the triangle's vertices and edges are sufficient to define a quadratic surface, but the average of the resulting reconstruction is not guaranteed to match the average of the initial data over the element area. To ensure this property, which is required for a conservative scheme, we add a third-order bubble function to the six $p = 2$ Lagrange basis functions. We do not want the third-order function to influence the edge values, so the only non-zero $p = 3$ coefficient is the center value, labeled "7" in Fig. 2.8. While it is obvious that the coefficients

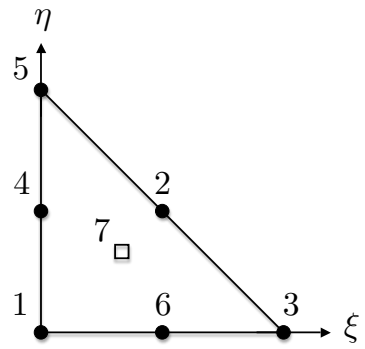


Figure 2.8: Node placement for AF basis functions

multiplying the quadratic basis functions should be the appropriate edge or vertex value, we have to carefully choose the bubble coefficient, c_7, to recover a cell average consistent with the given data. We start with the definition of the average for element j.

$$\begin{aligned}
\bar{u}_j &= \frac{1}{A} \iint_{\Omega_j} u(x,y)\ d\Omega \\
&= \frac{J}{A} \int_0^1 \int_0^{1-\xi} u(\xi(x,y), \eta(x,y))\ d\eta\ d\xi \\
&= 2 \int_0^1 \int_0^{1-\xi} \left[\left(\sum_{i=1}^{6} u_i \phi_i \right) + c_7 \phi_7 \right]\ d\eta\ d\xi
\end{aligned} \tag{2.25}$$

We can rearrange this equation to solve for the unknown bubble coefficient. The complete set of basis functions and coefficients is listed in Table 2.2.

Table 2.2: Basis functions and coefficients for 2D reconstruction

Index	$\boldsymbol{c_i}$	$\boldsymbol{\phi_i}$
1	u_1	$(1-\xi-\eta)(1-2\xi-2\eta)$
2	u_2	$4\xi\eta$
3	u_3	$\xi(2\xi-1)$
4	u_4	$4\eta(1-\xi-\eta)$
5	u_5	$\eta(2\eta-1)$
6	u_6	$4\xi(1-\xi-\eta)$
7 (bubble)	$\frac{20}{9}\left[\bar{u}_j - \frac{1}{3}(u_2+u_4+u_6)\right]$	$27\xi\eta(1-\xi-\eta)$

2.2.3 Updates

Once the reconstruction has been fully defined, we can develop expressions for the nodal updates, which are simply the function value at the characteristic origin. Similar to the one-dimensional case, we exploit the fact that the characteristics are straight lines.

$$t\boldsymbol{\lambda} = \mathbf{J}(\boldsymbol{\xi}_i - \boldsymbol{\xi}_0) \tag{2.26}$$

Solving for the origin:

$$\boldsymbol{\xi_0} = \boldsymbol{\xi_i} - t\mathbf{J}^{-1}\boldsymbol{\lambda} \tag{2.27}$$

Once the origin has been calculated, it can simply be used in the reconstruction function to find the solution at a given time.

$$u(\boldsymbol{\xi_i}, t) = u(\boldsymbol{\xi_0}) \tag{2.28}$$

A two-dimensional version of Simpson's numerical integration rule is used to estimate the average flux through an interface, where Eq. (2.28) is used to find the

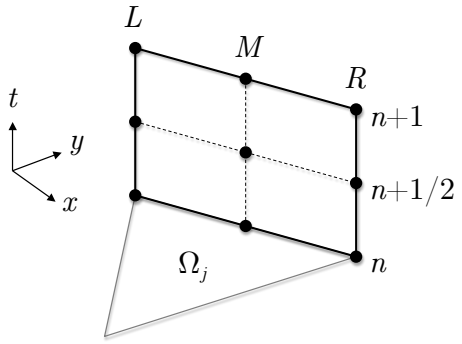


Figure 2.9: Nomenclature for 2D flux calculation

required point values.

$$\begin{aligned}\bar{\mathbf{f}} = \frac{1}{9}\Bigg[&\frac{1}{4}\left(\mathbf{f}_L^n + \mathbf{f}_R^n + \mathbf{f}_L^{n+1} + \mathbf{f}_R^{n+1}\right) \\ &+ \left(\mathbf{f}_M^n + \mathbf{f}_L^{n+1/2} + \mathbf{f}_R^{n+1/2} + \mathbf{f}_M^{n+1}\right) + 4\mathbf{f}_M^{n+1/2}\Bigg]\end{aligned} \tag{2.29}$$

The conserved variable in each cell is updated by integrating the flux around the boundary of the element. This is equivalent to dotting the flux with the face normal and multiplying by the face length for all three faces of the cell.

$$\bar{u}_j^{n+1} = \bar{u}_j^n - \frac{\Delta t}{\Omega_j}\sum_{m=1}^{3}\left(\bar{\mathbf{f}}_m \cdot \mathbf{n}_m\right)\ell_m \tag{2.30}$$

Note that each face makes the same contribution to the cells neighboring the interface, but with opposite signs due to the unique normal direction. Therefore, it is more efficient to perform the calculation within a loop over the mesh faces, changing the sign according to the normal direction. The point-update procedure used within the cell-loop is described by Algorithm 2.1.

2.2.4 Time step restriction

The AF point update formulas assume that characteristics drawn backward from a point do not cross more than one cell. This places a time step restriction on the

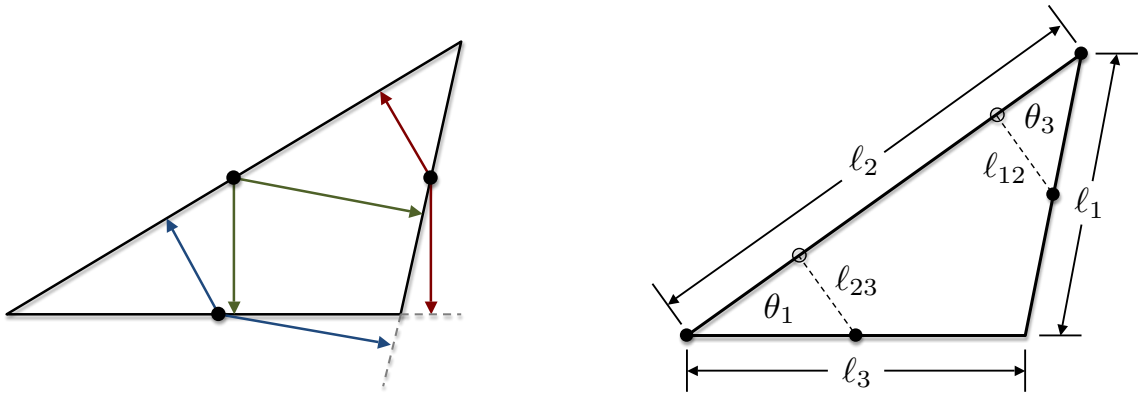
Algorithm 2.1 Point update procedure for scalar advection

```
for cell vertex = 1, 3 do
  if ξ_0^{n+1} lies within cell then
    Update the point values at n + 1/2 and n + 1; Eq. (2.28)
  end if
end for
if two-dimensional then
  for cell edges = 1, 3 do
    if ξ_0^{n+1} lies within cell then
      Update the point values at n + 1/2 and n + 1; Eq. (2.28)
    end if
  end for
end if
```

scheme and allows us to define a CFL condition.

$$\Delta t_{\max} = \frac{\ell_{\min}}{|\boldsymbol{\lambda}|} \tag{2.31}$$

This time step limitation has been implemented in a simplified form. Clearly, the characteristics stemming from the edge midpoints are most restrictive. In Eq. (2.31), $\ell_{\min}$ is the minimum distance from an edge midpoint to the other two faces and $|\boldsymbol{\lambda}|$ is the magnitude of the wave speed. We have imposed that these characteristics will not leave the cell even if they are directed normal to the opposite face. Figure 2.10(a) illustrates the various possibilities for $\ell_{\min}$. In practice, there is only one unique distance per cell edge. We can prove this is true by labeling the triangle as shown in Fig. 2.10(b). Using the fact that the sub-elements containing θ_1 and θ_3 are right-triangles we can write expressions for the lengths ℓ_{23} and ℓ_{12} connecting opposite edges.



(a) Minimum distances from edge centers

(b) Similar lengths ℓ_{12} and ℓ_{23}

Figure 2.10: Relevant distances for CFL calculation

$$\ell_{23} = \frac{\ell_3}{2} \sin\theta_1 \tag{2.32a}$$

$$\ell_{12} = \frac{\ell_1}{2} \sin\theta_3 \tag{2.32b}$$

Dividing Eq. (2.32a) by the length ℓ_1:

$$\frac{\ell_{23}}{\ell_1} = \frac{\ell_3}{2} \frac{\sin\theta_1}{\ell_1} \tag{2.33}$$

By the law of sines:

$$\frac{\sin\theta_1}{\ell_1} = \frac{\sin\theta_3}{\ell_3} \tag{2.34}$$

Combining Eq. (2.33) and Eq. (2.34) to eliminate $\sin\theta_1$, we arrive at an expression for the length ℓ_{23} that is identical to length ℓ_{12} from Eq. (2.32b).

$$\ell_{23} = \frac{\ell_1}{2} \sin\theta_3 \tag{2.35}$$

Therefore, we only need to check one distance per edge. The element area and one edge length can be used to efficiently calculate the minimum normal distance between an edge-midpoint and its opposing side. Let $\mathbf{x}_E$ represent the edge midpoint coordinate and $\mathbf{x}_I$ represent the intersection point, with ℓ defining the distance between

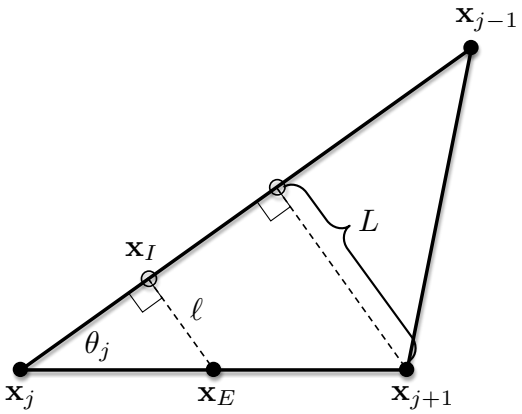


Figure 2.11: Nomenclature for minimum distance calculation

these two points. We can then use the well known formula for the area of a triangle to derive an expression for the length L between node $j+1$ and the opposite edge, where S is the element area.

$$\frac{L}{2} = \frac{S}{|\mathbf{x}_{j-1} - \mathbf{x}_j|} \tag{2.36}$$

The lengths ℓ and L are related because they are sides of similar triangles sharing angle θ_j.

$$\sin\theta_j = \frac{L}{|\mathbf{x}_{j+1} - \mathbf{x}_j|} = \frac{\ell}{\frac{1}{2}|\mathbf{x}_{j+1} - \mathbf{x}_j|} \tag{2.37}$$

The minimum normal distance can be calculated by combining Eq. (2.36) and Eq. (2.37) and looping over the three edge indices.

$$\ell_{\min} = \min_j \left(\frac{S}{|\mathbf{x}_{j-1} - \mathbf{x}_j|} \right) \tag{2.38}$$

This minimum length can then be used in Eq. (2.31) to (conservatively) determine the maximum allowable time step.

2.2.5 Two-dimensional linear advection

The simplest example using the two-dimensional AF algorithm is the case when the wave speed $\boldsymbol{\lambda} = (a, b)^{\mathrm{T}}$ is constant throughout the domain. With this definition,

the average flux at the interface becomes:

$$\begin{aligned}\bar{\mathbf{f}} =& \boldsymbol{\lambda}\frac{1}{9}\left[\frac{1}{4}\left(u_L^n + u_R^n + u_L^{n+1} + u_R^{n+1}\right)\right. \\ &\left. + \left(u_M^n + u_L^{n+1/2} + u_R^{n+1/2} + u_M^{n+1}\right) + 4u_M^{n+1/2}\right]\end{aligned} \tag{2.39}$$

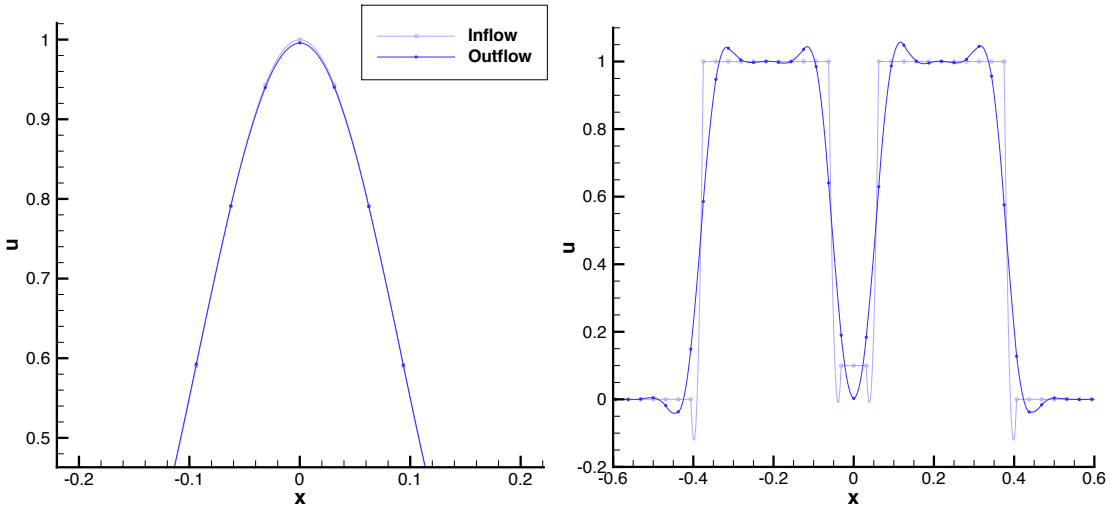
One method to verify the two-dimensional AF method is to compute a steady-state advection case. We prescribe a function at one boundary, advect the waveform through the domain, and compare the solution at the exit interface once the norm of the residual over the N cells, defined in Eq. (2.40), drops to machine zero.

$$\|R\|_2 = \sqrt{\frac{\sum_{j=1}^{N}\left[\sum_{m=1}^{3}\left(\bar{\mathbf{f}}_m \cdot \mathbf{n}_m\right)\ell_m\right]^2}{N}} \tag{2.40}$$

At steady state, the coordinate direction along the flow vector acts as a time-like variable. This reduces the dimensionality of the problem, and allows us to predict the solution at the exit interface, which should match the one-dimensional solution. If the advection speed only has one non-zero component and the domain is square, the solution profile at the outflow boundary should exactly match the inflow solution.

The steady-state case was run on a square mesh with dimensions $[-1, 1]$ in the x and y directions. The mesh consisted of 2398 randomly oriented elements. Figure 2.12 shows the steady-state solutions for a smooth Gaussian profile and the cross section of a slotted cylinder. Each case converged to machine zero within 600 iterations, and both show very good accuracy. The peak value for the Gaussian is only reduced 0.4%, while the cylinder exhibits typical third-order behavior with overshoots of 4%-6%.

Another quantitative measure that the scheme is behaving correctly is to calculate the error as the mesh is refined. The AF scheme, by construction, is third-order accurate. This means that, when computing smooth solutions, the error should decrease by a factor of eight (2^3) every time the relevant length scale is halved. Because the mesh is unstructured and the Courant number varies from cell to cell, we approximate



(a) Gaussian (b) Slotted cylinder

Figure 2.12: Steady advection solutions on randomized mesh

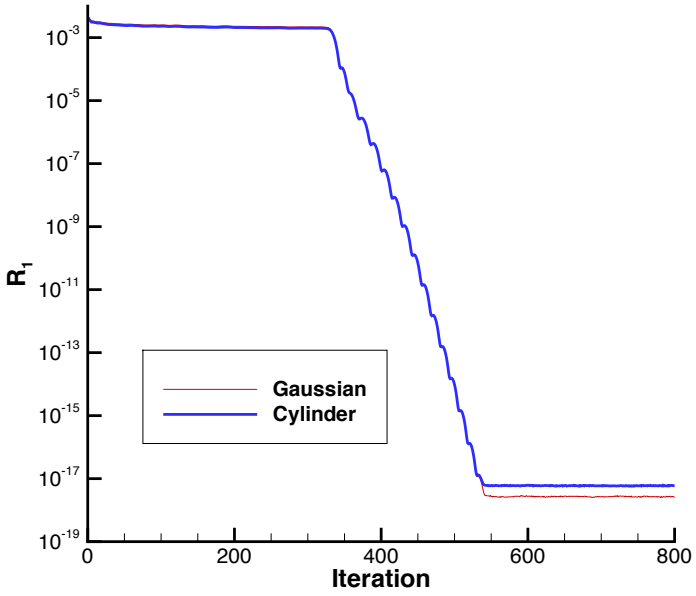


Figure 2.13: Convergence of steady advection cases

the average mesh spacing as $1/\sqrt{\text{DOF}}$. The error for any high-order scheme must be computed by integrating over each element. Equation (2.41) defines the error norm, where again N is the total number of cells, u is the computed solution after a set

number of iterations and U_0 is the projection of the exact solution.

$$\|\epsilon\|_p = \left[\frac{\sum_{j=1}^{N} \iint_{\Omega_j} |u - U_0|^p \ d\Omega_j}{\sum_{j=1}^{N} \Omega_j} \right]^{1/p} \tag{2.41}$$

In practice, the error integral is computed with an appropriate numerical integration method that samples the function at M locations within an element.

$$\|\epsilon\|_p = \left\{ \frac{\sum_{j=1}^{N} J_j \sum_{i=1}^{M} w_i \, |u(\boldsymbol{\xi}_i) - U_0(\boldsymbol{\xi}_i)|^p}{\sum_{j=1}^{N} \Omega_j} \right\}^{1/p} \tag{2.42}$$

For an AF scheme the total DOF are simply the sum of the nodes, faces, and cells in the mesh. Because nodes and faces are shared between elements, this works out to approximately three DOF per cell. A fair comparison is to compare the AF scheme to a DG scheme that also uses three DOF per cell. This means solving the problem using a DG1 scheme, which uses linear ($p = 1$) basis functions. Figure 2.14 compares the error with respect to the initial projection for the AF scheme and a DG1 solution computed with the XFlow solver [12], [13]. The AF scheme achieves third-order accuracy for the smooth Gaussian profile, and performs comparably to the DG1 code on the discontinuous slotted cylinder profile. It is important to note that the DG1 scheme achieves third-order accuracy through superconvergence. In general, three DOF per cell may not be sufficient to obtain third-order accuracy with the DG scheme; only second-order accuracy is guaranteed. By contrast, the AF scheme does not rely on superconvergence to realize third-order accuracy. Thus, if the DG scheme is not superconverging, and therefore using $p = 2$ basis functions to obtain third-order accuracy, the AF scheme will be much more efficient when computing solutions to the same error level, assuming the two methods have similar constants on the leading error term.

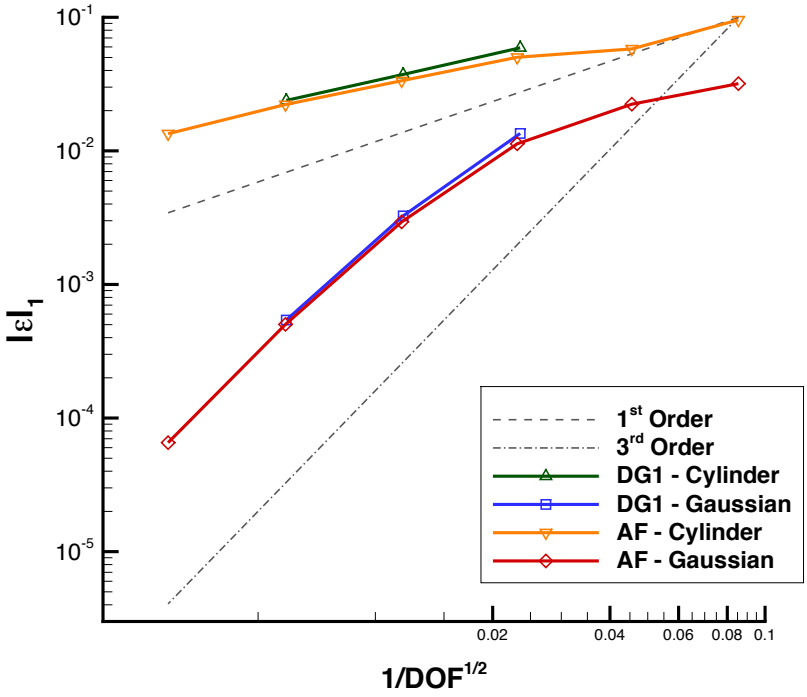


Figure 2.14: AF error compared to DG1 scheme

Table 2.3: Linear advection convergence of Gaussian using AF and DG methods

	$\mathbf{DOF}^{-1/2}$		$\|u\|_1$		$\mathcal{O}(L_1)$	
Ref. Level	**AF**	**DG**	**AF**	**DG**	**AF**	**DG**
1	8.5436×10^{-2}	-	3.1855×10^{-2}	-	-	-
2	4.5502×10^{-2}	-	2.2347×10^{-2}	-	0.5627	-
3	2.3119×10^{-2}	2.3531×10^{-2}	1.1293×10^{-2}	1.3503×10^{-2}	1.0080	-
4	1.1681×10^{-2}	1.1785×10^{-2}	2.9386×10^{-3}	3.2609×10^{-3}	1.9720	2.0549
5	5.8798×10^{-3}	5.9061×10^{-3}	5.0247×10^{-4}	5.4153×10^{-4}	2.5729	2.5988
6	2.9379×10^{-3}	-	6.5412×10^{-5}	-	2.9385	-

2.2.6 Two-dimensional circular advection

An intermediate step between linear advection, which has a constant wave speed, and nonlinear advection, in which the wave speed depends on the solution, is circular advection. In this case the wave speed is a function of the distance from the center

of rotation (x_0, y_0).

$$\boldsymbol{\lambda}(x,y) = \omega \begin{pmatrix} y_0 - y \\ x - x_0 \end{pmatrix} \tag{2.43}$$

The velocity function described by Eq. (2.43) represents a counter-clockwise rotation with angular velocity ω. The flexibility of the AF method means that there is very little difference between this case and the constant speed calculation. The non-constant speed can be handled by accounting for the velocity variation in the flux function.

$$\begin{aligned} \bar{\mathbf{f}} = \frac{1}{9} \Bigg\{ & \boldsymbol{\lambda}_L \left[\frac{1}{4} \left(u_L^n + u_L^{n+1} \right) + u_L^{n+1/2} \right] \\ & + \boldsymbol{\lambda}_M \left(u_M^n + 4u_M^{n+1/2} + u_M^{n+1} \right) \\ & + \boldsymbol{\lambda}_R \left[\frac{1}{4} \left(u_R^n + u_R^{n+1} \right) + u_R^{n+1/2} \right] \Bigg\} \end{aligned} \tag{2.44}$$

Figure 2.2.6 shows that the AF scheme maintains third-order accuracy for problems with varying wave speeds.

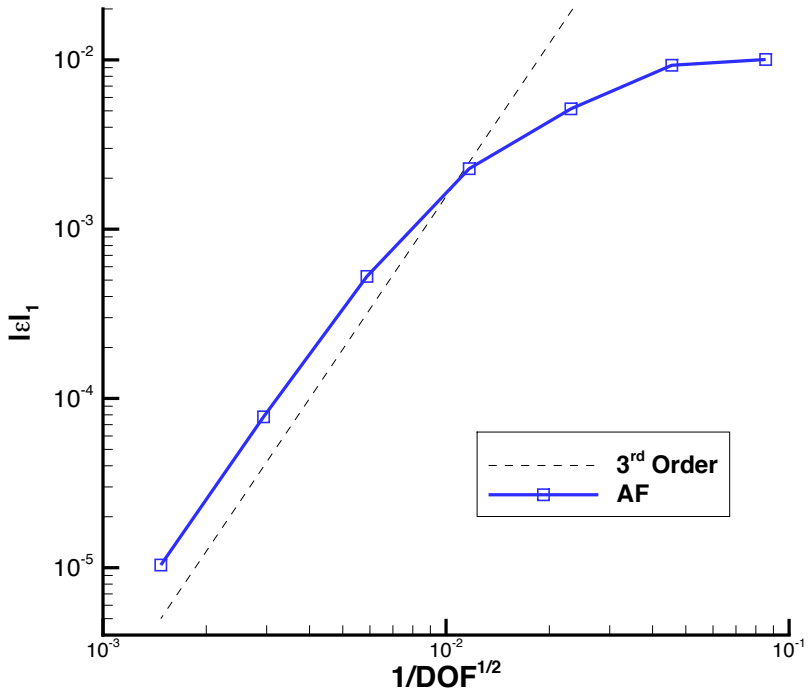


Figure 2.15: AF error for circular advection case

Table 2.4: Circular advection convergence

Ref. Level	$\mathbf{DOF}^{-1/2}$	$\|u\|_1$	$\mathcal{O}(L_1)$	$\|u\|_2$	$\mathcal{O}(L_2)$
1	8.5436×10^{-2}	1.0054×10^{-2}	-	2.3134×10^{-2}	-
2	4.5502×10^{-2}	9.2948×10^{-3}	0.1246	3.2855×10^{-2}	-0.5568
3	2.3119×10^{-2}	5.1388×10^{-3}	0.8752	2.9425×10^{-2}	0.1628
4	1.1681×10^{-2}	2.2767×10^{-3}	1.1925	1.6113×10^{-2}	0.8821
5	5.8798×10^{-3}	5.2578×10^{-4}	2.1351	4.4937×10^{-3}	1.8603
6	2.9379×10^{-3}	7.7843×10^{-5}	2.7531	7.4098×10^{-4}	2.5979
7	1.4741×10^{-3}	1.0370×10^{-5}	2.9228	1.0042×10^{-4}	2.8978

We have demonstrated many of the basic properties and advantages of the AF method in this chapter. The one-dimensional analysis of the AF scheme shows that it has superior error properties, especially compared to FV schemes. Its compact stencil invites comparisons with FE-based methods such as DG. The two-dimensional linear advection results demonstrate that the method performs similarly to DG1 scheme without relying on superconvergence for third-order accuracy. The AF scheme is able to achieve this accuracy efficiently by sharing edge and vertex values between elements. Finally, the circular advection results demonstrate the flexibility of the approach and show the good results are not limited to constant wave speeds. Together, the linear advection results show that the AF method is a capable scheme, and validate that the method has all that is required to tackle more complex cases.

CHAPTER III

Nonlinear Advection

When the advection speed is a function of the local velocity, the problem becomes nonlinear. In this case, the solution characteristics may intersect, creating shock waves, or diverge, creating expansions. The inviscid Burgers' equation is a simple, scalar equation that may be used to evaluate how a scheme treats these features.

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = 0 \tag{3.1}$$

Rewriting in conservation form:

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\left(\frac{1}{2}u^2\right) = 0 \tag{3.2}$$

The addition of flow discontinuities necessitates the use of solution limiters that control the oscillations created by these features. The flexibility of the AF method means that there are several possible techniques that can be employed to solve nonlinear problems. In this chapter, we investigate one possible method and detail an approach to limiting that we have found quite useful.

3.1 Burgers' equation

The edge updates for the linear advection equation are directionally biased. Each point is updated from only one of the two cells sharing that interface. When solving Burgers' equation, the cells on either side of the interface may contribute to the point update. It is important then for the update procedure to properly sum the individual contributions from every cell because the solution characteristics intersecting an update point can originate from multiple cells. The update procedure we follow in this chapter, which is one of several possibilities, is to write the element contributions to the interface as a correction to the vertex value at time n. The expression for the point-value updates is straightforward, and we use Eq. (2.8) to develop an expression for the flux signal.

$$\delta u = u(\xi_0) - u_0(\xi_i) \tag{3.3a}$$

$$\begin{aligned}\delta \bar{f} = \frac{\Delta x}{3\Delta t}\left(\xi_i - \xi_0\right)\left[u(\xi_i) - 2u(\xi_0) + \alpha\xi_i\xi_0 + \frac{\beta}{2}\left(\xi_i + \xi_0\right) + c_1\right] \\ + f(u(\xi_0)) - f(u_0(\xi_i))\end{aligned} \tag{3.3b}$$

where u_0 is the unlimited solution reconstruction at time n and the variables α and β are combinations of the one-dimensional basis coefficients.

$$\begin{aligned}\alpha &= 2\left(c_1 - 2c_2 + c_3\right) \\ \beta &= -\left(3c_1 - 4c_2 + c_3\right)\end{aligned} \tag{3.4}$$

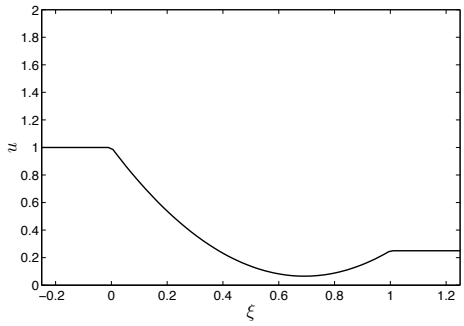
To find update equations for Burgers' equation, we follow the same procedure as the linear case with one notable exception. Although the solution characteristics are still straight, the wave speed λ is no longer constant. It is a function of the flow field, specifically $\lambda = u(\xi_0)$. Combining Eqs. (2.4-2.5) yields a quadratic equation in ξ_0

which has the solutions given below.

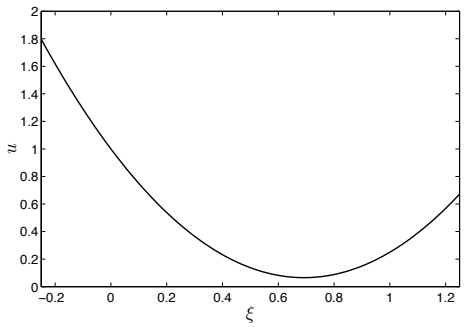
$$\xi_0 = \frac{2\left(\xi_i - c_1\frac{\Delta t}{\Delta x}\right)}{\left(\beta\frac{\Delta t}{\Delta x}+1\right) \pm \sqrt{\left(\beta\frac{\Delta t}{\Delta x}+1\right)^2 + 4\alpha\frac{\Delta t}{\Delta x}\left(\xi_i - c_1\frac{\Delta t}{\Delta x}\right)}} \tag{3.5}$$

The characteristic origin defined in Eq. (3.5) presents two primary challenges: selecting the proper solution branch and ensuring that the origin is real. Closely examining the solution characteristics for an example problem can help make sense of Eq. (3.5).

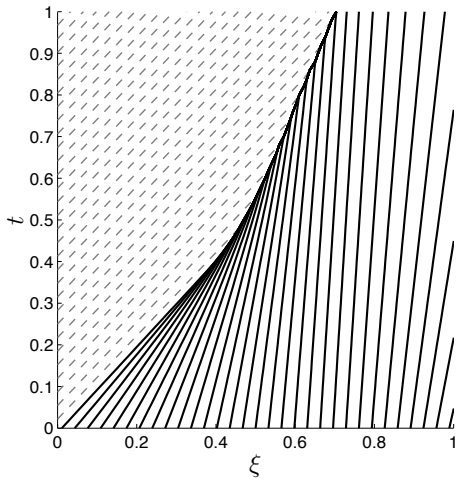
Take a moving shock problem for which the left state is $u_L = 1.0$ and the right state is $u_R = 0.25$. If we choose to look at a moment in time when the shock is positioned within the cell such that the cell average is $\bar{u}_j = 0.3$, we have the quadratic reconstruction shown in Fig. 3.1(a). The characteristic origin equation, however, does not account for neighboring information, so from the perspective of the mathematics, the quadratic reconstruction extends beyond the cell as illustrated in Fig. 3.1(b). The solution characteristics resulting from these two reconstructions are illustrated in Fig. 3.1(c) and Fig. 3.1(d), where the horizontal axis has been limited to the left and right cell interface. The dashed lines show characteristics originating outside of the cell, while the solid lines show the characteristics resulting from the parabolic reconstruction from $\xi = 0$ to $\xi = 1$. The non-physical picture shows that external characteristics and internal characteristics meet at the right interface. The two lines that intersect at every point in time at $\xi = 1$ correspond to the two solution branches for Eq. (3.5). The relevant solution for the origin ξ_0 is the characteristic that originates within the cell. In other words, we seek the solution branch that lies closest to the interface of interest. Whether we are finding the origin for the left interface, at which $\xi_i = 0$, or the right interface where $\xi_i = 1$, the difference between the two solution branches is solely determined by the denominator. Assuming the term beneath the square root is positive, it is clear from Eq. (3.5) that the positive solution branch maximizes the denominator and therefore minimizes the distance



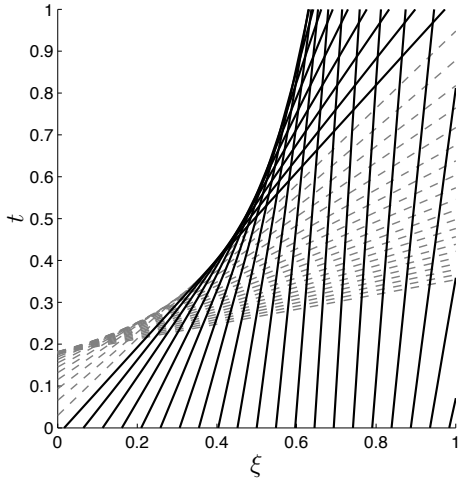
(a) Reconstruction for domain



(b) Reconstruction neglecting neighbors



(c) Physical characteristics for shock



(d) Non-physical characteristics

Figure 3.1: Characteristics for shock problem; $u_{j-1/2} = 1.0$, $\bar{u}_j = 0.3$, $u_{j+1/2} = 0.25$

between the interface and the characteristic origin. Therefore it is the positive branch that describes the characteristic origin closest to the test interface. Although we can deduce which is the proper solution branch, there is no guarantee that the radicand is positive and the origin is a real value. We enforce positivity of the term as part of a holistic limiting approach that addresses the major issues that arise from Burgers' equation.

3.2 Limiting

We seek a limiting strategy that eliminates overshoots in the conserved variables without being overly dissipative. We also want to develop a limiter that only uses

information local to a single cell, maintaining a compact stencil. The AF method gives us the freedom to modify the left and right edge values as long as we do not alter the cell average. Graphically representing the possible reconstructions for an element makes it easier to determine cases when the data may be suspect and identify methods to modify the data to produce a desired result. One general way of illustrating all the possible reconstructions for a given cell is to hold the cell average constant and plot the values at the right interface versus the values at the left interface. For example, the cell average for a linear reconstruction is defined as $\bar{u}_j = 1/2\left(u_{j+1/2} + u_{j-1/2}\right)$. This would be represented as the line $u_R = -u_L + 2u_M$ on the graph, where $u_L = u_{j-1/2}$, $u_M = \bar{u}_j$, and $u_R = u_{j+1/2}$.

At its heart, a limiting strategy comes down to deciding when to be skeptical of a solution and what to do when you don't trust the results. We choose to accept the general shape of a given reconstruction, measured by the ratio of the curvature to the gradient. This constrains the possible limited reconstructions to the straight line formed between the original data point (u_{L0}, u_{R0}) and the point (u_M, u_M), which is the first-order representation of the conserved data. We also need to define regions where the reconstruction is not acceptable. One such region is clear from Eq. (3.5), where negative values of the radicand will lead to complex values for the characteristic origin. A second region is the set of left and right values that produce an internal extremum that exceeds an acceptable value, for example the maximum (or minimum) of the left edge value, right edge value and cell average. We also want to avoid reconstructions that lead to intersecting characteristics, which produces a shock in the cell that invalidates the form of the characteristic origin equation given in Eq. (3.5). A final constraint is that the vertex updates from each cell do not give rise to an interface flux that creates a new local extremum.

Taken together, this set of constraints form the basis for a *shape-preserving* limiter. The main objective of the shape-preserving limiter is to select left and right edge

values such that the resulting reconstruction has the following properties:

1. Characteristics do not intersect (no shock within cell)
2. Characteristic origin is real
3. Reconstruction extremum is bounded by $\max(u_L, u_M, u_R)$ when curvature is positive and $\min(u_L, u_M, u_R)$ when curvature is negative
4. Vertex updates do not create new extremum

The first property defines a line where the left, right, and cell average values lead to a reconstruction that has characteristics that intersect at $t = \Delta t$. The second and third properties are represented by ellipses, illustrated in Fig. 3.2. We term the reconstruction defined by the unmodified data the *natural reconstruction*. The line between the natural reconstruction point and the first-order reconstruction may be parameterized with the equation below, where $\tau = 0$ is the original point (u_{L0}, u_{R0}) and $\tau = 1$ is the constant (flat) reconstruction point (u_M, u_M).

$$\tau = \frac{u_L - u_{L0}}{u_M - u_{L0}} = \frac{u_R - u_{R0}}{u_M - u_{R0}} \tag{3.6}$$

The intersections between the shape-preserving line and the features that define undesirable behavior lead to a set of finite choices for the limited reconstruction. Each intersection between an ellipse or shock formation line has an associated τ value. The methods for finding the intersections and their associated τ values are discussed in Appendix B. The most conservative choice is to choose the value closest to the first-order representation, but this may be overly dissipative. Figure 3.3 presents one possible algorithm that determines the minimum amount of limiting that eliminates overshoots. Within the flow chart, the intersection with the shock formation line is $\tau_{\text{s.f.}}$, the intersection with the bounded extremum ellipse is $\tau_{\text{b.e.}}$, the real ellipse

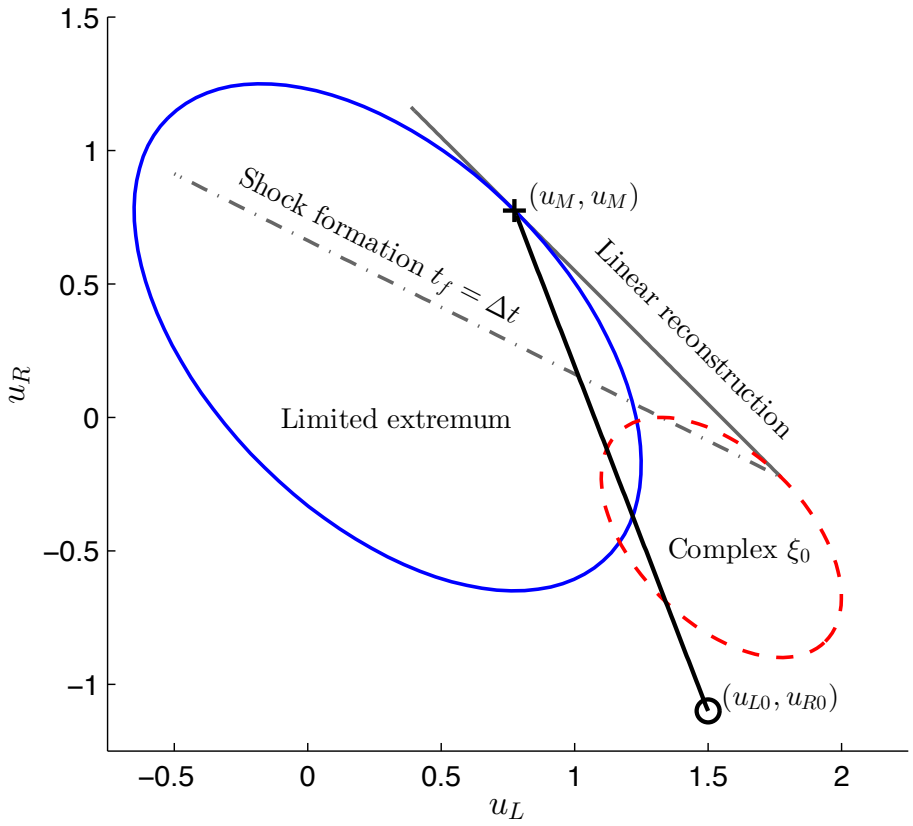


Figure 3.2: Limiting diagram for shock; $u_{L0} = 1.5, u_M = 0.775, u_{R0} = -1.1$

intersection point is τ_{real}, the first-order point (u_M, u_M) is τ_{flat}, and the point on the shape-preserving line that corresponds to the maximum signal value is τ_{sig}. Notably, for the simpler case of linear advection, the characteristics will always be real and will not intersect. This leaves only the left-hand of the flow chart pictured in Figure 3.3. Additionally, the only feature appearing in the limiting diagram, Figure 3.2, will be the limited extremum ellipse. When used with either linear advection or Burgers' cases, the limiter algorithm systematically determines when it is necessary to limit and if so, what the left and right states should be to produce the desired result.

3.3 Test cases

The simplest way to evaluate a limiter is to monitor the cell averages for new extrema. In addition to looking at the cell averages, an additional measure of the

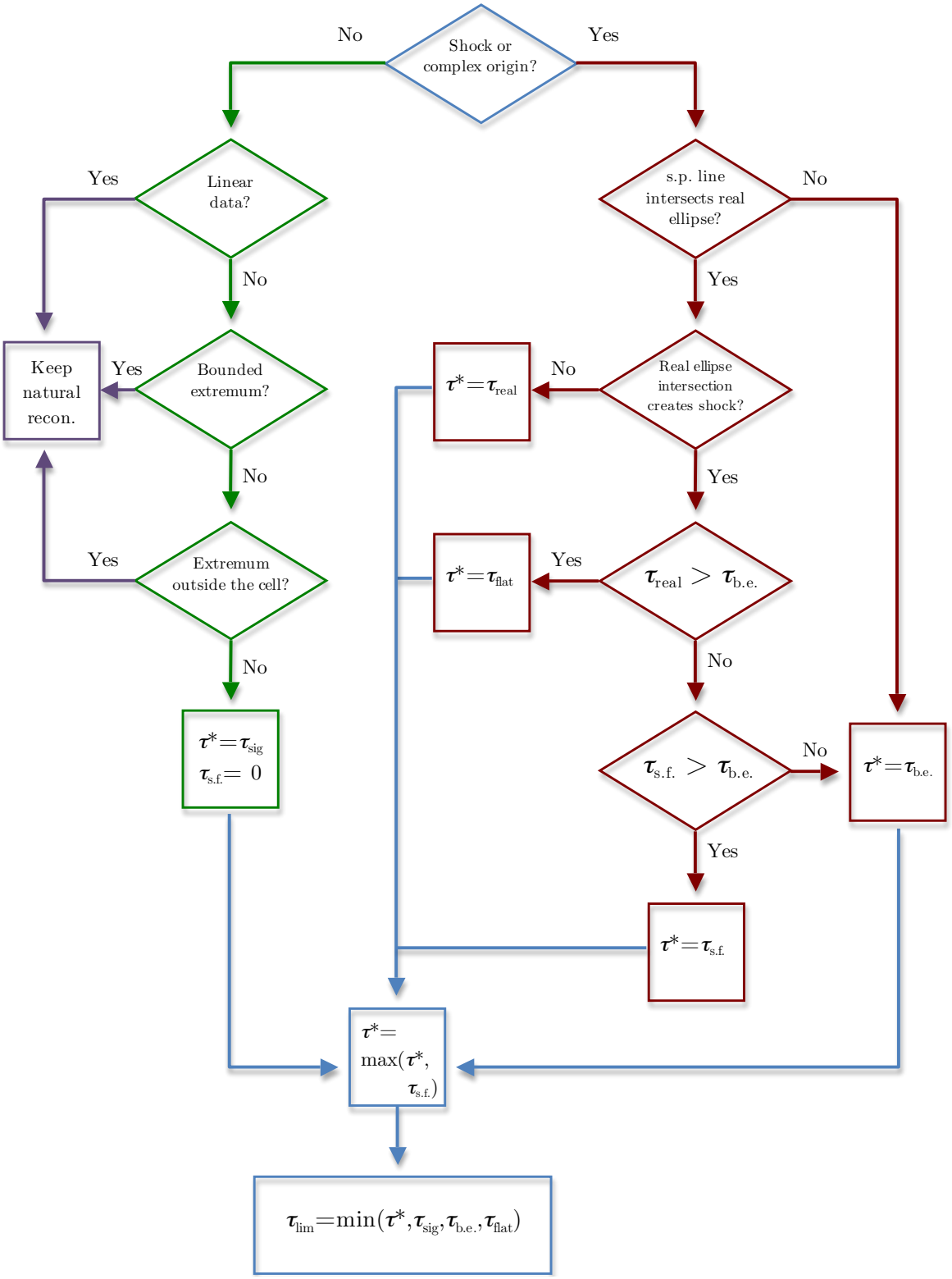


Figure 3.3: Limiter algorithm

limiter's effectiveness is to plot the time history of the fluxes. The exact solution for Burgers' equation can be used to find the cell averages at any point in time. With this information, we can calculate the flux difference across each cell in the one-dimensional domain, assuming forward Euler time integration.

$$F_{j-1/2} - F_{j+1/2} = \frac{\Delta x}{\Delta t}\left(\bar{u}_j^{n+1} - \bar{u}_j^n\right) \tag{3.7}$$

Equation (3.7) holds for each of the N cells in the domain and therefore defines a linear system that can be solved to find the expected flux at every edge. For simplicity, we assume equal spacing. In order to close the system, we set the flux at the right edge of the domain to the analytical flux calculated with the conserved value, which is valid as long as the right-most cell average does not change from the initial value.

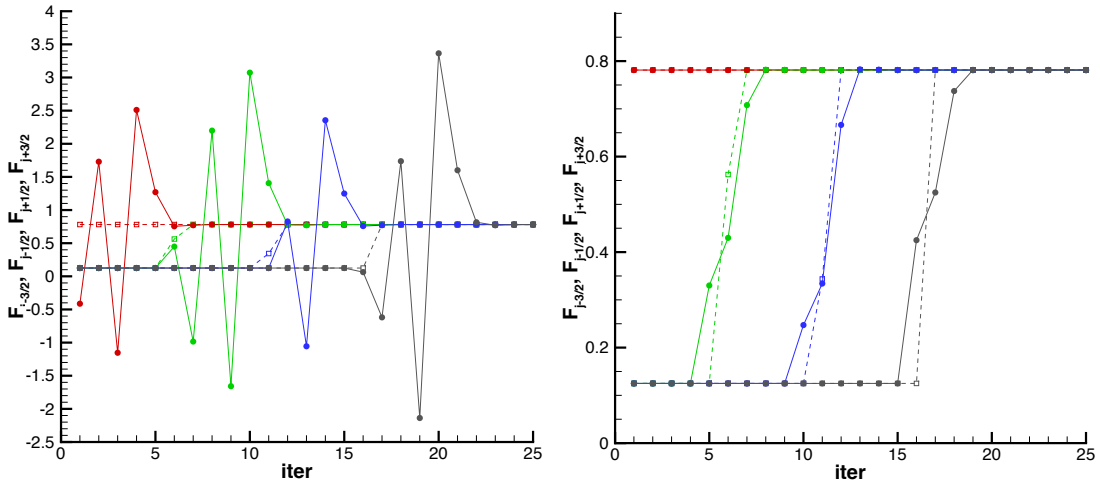
$$\begin{pmatrix} 1 & -1 & 0 & \cdots & 0 \\ 0 & 1 & -1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & 0 & \ddots & 1 & -1 \\ 0 & \cdots & \cdots & 0 & 1 \end{pmatrix} \begin{pmatrix} F_{1/2} \\ F_{3/2} \\ F_{5/2} \\ \vdots \\ F_{N+1/2} \end{pmatrix} = \begin{pmatrix} \frac{\Delta x}{\Delta t}\left(\bar{u}_1^{n+1} - \bar{u}_1^n\right) \\ \frac{\Delta x}{\Delta t}\left(\bar{u}_2^{n+1} - \bar{u}_2^n\right) \\ \vdots \\ \frac{\Delta x}{\Delta t}\left(\bar{u}_N^{n+1} - \bar{u}_N^n\right) \\ \frac{1}{2}\left(\bar{u}_N^n\right)^2 \end{pmatrix} \tag{3.8}$$

The limiting algorithm was tested using three test cases: a moving shock, an expansion, and a more complex wave that combines features of both.

3.3.1 Shock

The initial conditions for the shock case were $u_{\text{left}} = 1.25$ and $u_{\text{right}} = -0.5$ with the shock starting at a cell interface. The cell spacing and time step were set to $\Delta x = 0.1$ and $\Delta t = 5 \times 10^{-3}$, respectively. Figure 3.4 shows the shock fluxes for four cell edges with and without limiting, with the exact values shown with open symbols connected with dotted lines and the active flux solution with closed symbols

connected with solid lines. Without limiting it is possible to compute negative fluxes using the flux-signal expression Eq. (3.3b) due to incorrect contributions from the left and right cells. The figure clearly demonstrates that the limiter keeps the fluxes positive and controls overshoots in the flux values as the shock moves through the domain.



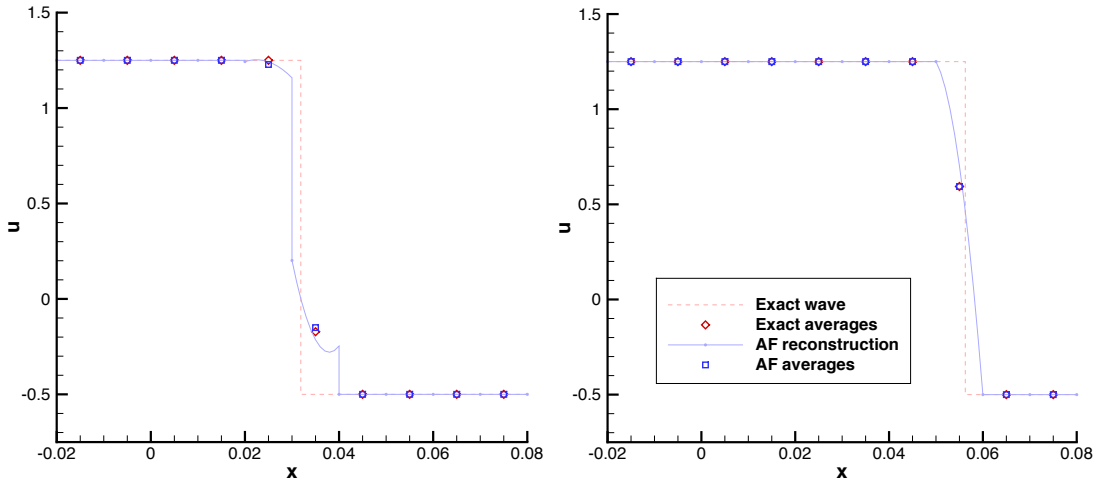
(a) No Limiting (b) Limited

Figure 3.4: Shock fluxes

The cell averages and internal reconstructions are shown in Fig. 3.5. The figure illustrates good agreement with the exact solution and shows that the scheme captures the shock within one or two cells.

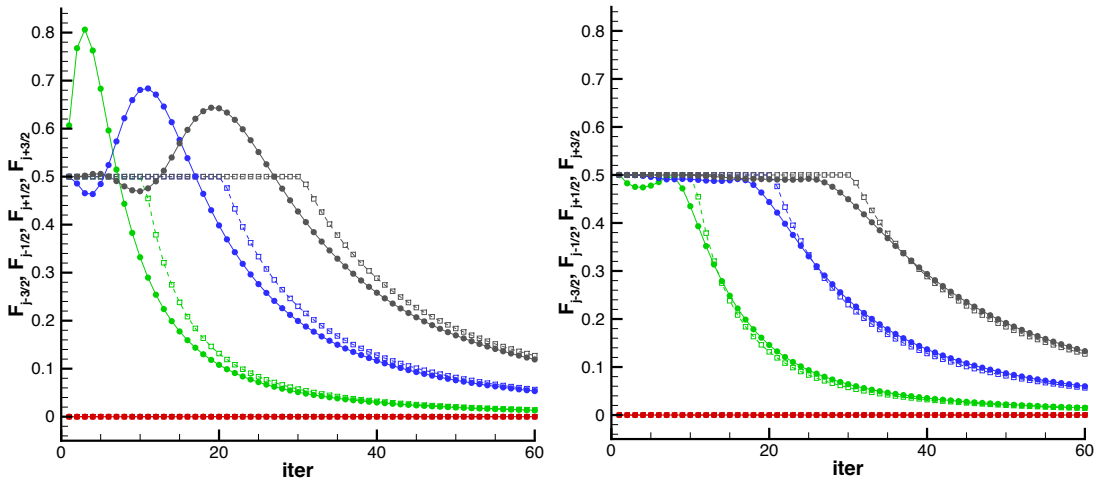
3.3.2 Expansion

To create an expansion, the left state was set to $u_{\text{left}} = -1.0$ and the right state $u_{\text{right}} = 1.0$ with the discontinuity again starting at a cell interface. The cell spacing and time step used were $\Delta x = 0.1$ and $\Delta t = 1 \times 10^{-3}$. Figure 3.6 compares the limited and unlimited fluxes for this case. Again, the limiter eliminates the overshoots in the flux values seen in the unlimited case. Figure 3.7 shows the cell averages and reconstructions for the expansion solution at $t = 0.1$. The AF scheme naturally breaks up the discontinuity without an entropy fix. The solution matches the exact



(a) $t = 0.085$ (iteration 17) (b) $t = 0.15$ (iteration 30)

Figure 3.5: Limited solution



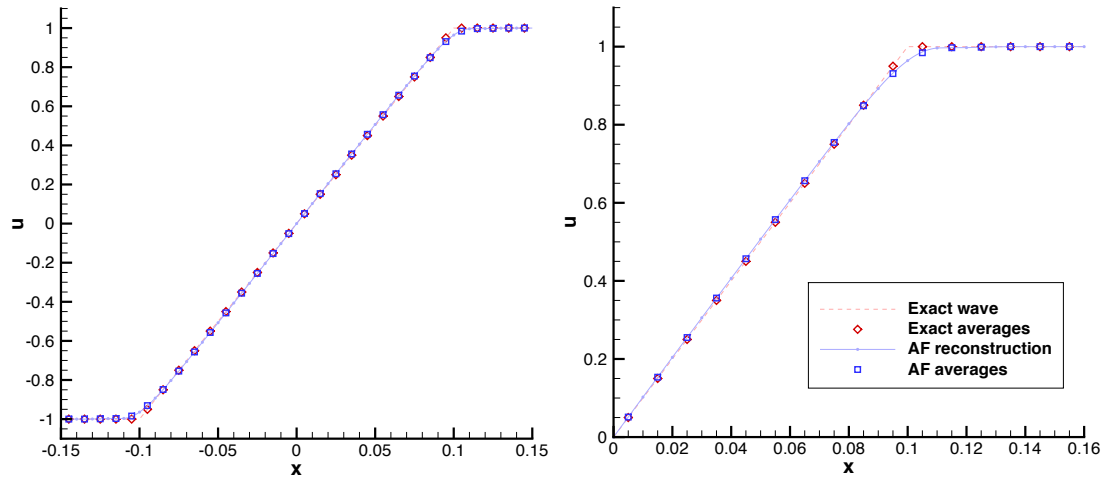
(a) No Limiting (b) Limited

Figure 3.6: Expansion fluxes

solution well and exhibits good symmetry.

3.3.3 E-C-E wave

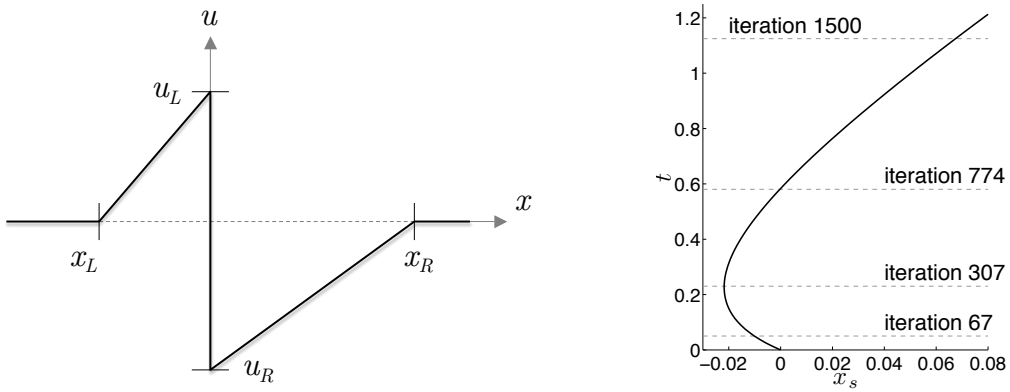
This case is a compression wave centered at $x = 0$, sandwiched between two expansions. Figure 3.8 shows the parameters that define the problem. Equation (3.9), taken from [54], defines the shock path and is valid for $x_s \in [x_L, x_R]$. Certain combinations of $x_{L,R}$ and $u_{L,R}$ can lead to a shock path that reverses direction over



(a) Full wave (b) Close-up of right wave front

Figure 3.7: Limited expansion solution at $t = 0.1$

time. We choose one such case, where $x_L = -1$, $x_R = 1/2$, $u_L = 3/2$, and $u_R = -2$. To ensure that the shock traverses several cells before reversing direction, the grid spacing was set at $\Delta x = 2 \times 10^{-3}$ and the time set to $\Delta t = 7.5 \times 10^{-4}$.



(a) Problem parameters (b) Shock path

Figure 3.8: E-C-E wave

$$x_s(t) = \frac{x_L x_R}{u_L x_R - u_R x_L}\left[u_L\left(1 - \frac{u_R\, t}{x_R}\right) - u_R\left(1 - \frac{u_L\, t}{x_L}\right) + (u_R - u_L)\sqrt{\left(1 - \frac{u_L\, t}{x_L}\right)\left(1 - \frac{u_R\, t}{x_R}\right)}\right] \tag{3.9}$$

Figure 3.9 shows the exact and AF solutions at four shock positions: half-way to the turnaround point (iteration 67), minimum x location (iteration 307), $x = 0$ (iteration 774), and the final shock location for the simulation (iteration 1500). There is good agreement between the exact and AF solutions and again the AF scheme captures the shock within one to two cells. There is a very small overshoot present in the solution at the left-most shock location, indicating that there may be some more tweaking necessary to ensure the limiter always prevents overshoots, but overall the shape-preserving limiter performs quite well.

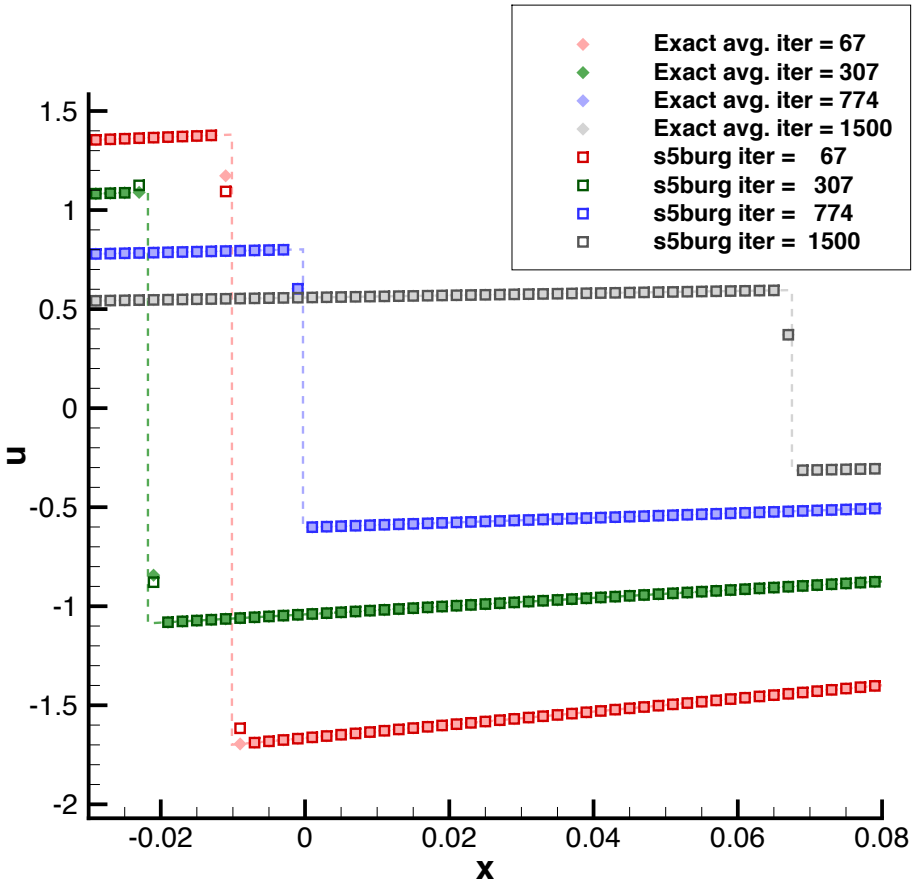


Figure 3.9: E-C-E wave solution at four times

3.4 Alternate approaches

Although the shape-preserving limiting algorithm presented in Fig. 3.3 is effective on a wide range of test cases, the flexibility of the AF method makes it possible to explore many other possibilities. In addition to opening multiple paths for limiting, the non-conservative interface updates of the AF scheme facilitate alternate methods of updating edge quantities. For instance, one could use a local linearization and assume the characteristic speed within each cell is constant. Here too, there are many ways to determine an average characteristic speed for the cell, but the most straightforward choice is to use the conserved value (cell average). This strategy eliminates the need to handle the complexities introduced by Eq. (3.5).

$$\xi_0 = \xi_i - \bar{u}_j \frac{\Delta t}{\Delta x} \tag{3.10}$$

Another way to simplify the characteristic origin calculation without sacrificing third-order accuracy is to use a series approximation for Eq. (3.5).

$$\begin{aligned} \xi_0 &= \xi_i - \left[c_1 + \xi_i \left(\beta + \alpha \xi_i\right)\right] \left(\frac{\Delta t}{\Delta x}\right) \left[1 - \left(\beta + 2\alpha\xi_i\right) \left(\frac{\Delta t}{\Delta x}\right)\right] + \mathcal{O}(\Delta t^3) \\ \alpha &= 2\left(c_1 - 2c_2 + c_3\right) \\ \beta &= -\left(3c_1 - 4c_2 + c_3\right) \end{aligned} \tag{3.11}$$

Both Eq. (3.10) and Eq. (3.11) are better suited to implementation in a multidimensional scheme than the exact origin expression of Eq. (3.5). Starting with two-dimensional cases, the exact characteristic origin equation becomes a nonlinear system, requiring the use of a root finder or some other comparable method, which greatly increases the cost. The local linearization makes it possible to use techniques from Chapter II. The alternate characteristic origin expressions also remove the square root from the equation, ensuring that the result is real.

Forcing the characteristic origin to be real simplifies the shape-preserving limiter diagram in Fig. 3.2, but it does not resolve the larger issue of how to apply shape-preserving ideas to multiple dimensions. In one dimension, there are only two free parameters, the left and right edge values. In two dimensions the number of parameters increases to six and in three dimensions it is ten. As the number of parameters increases, it is necessary to use more sophisticated methods to ensure that the limiter is effective without over-damping legitimate extrema. For example, one could define a set of objective functions and constraints and use optimization techniques to find edge values that do not produce new extrema. In short, the ideas presented in this chapter represent only one possible approach to nonlinear equations and limiting. The fact that the edge updates in the AF scheme do not need to be conservative makes the techniques for improving limiting and nonlinear solutions fertile ground for future research.

CHAPTER IV

Linear Acoustics and Linearized Euler

We can begin to study more complex flow behavior by examining the linear acoustic equations. The linear acoustics system is derived from the Euler equations by assuming that the density, pressure, and velocity peturbations around some background state are small.

$$\begin{aligned} \rho &= \rho_0 + \rho' \\ \mathbf{u} &= \mathbf{u}_0 + \mathbf{u}' \\ p &= p_0 + p' \end{aligned} \tag{4.1}$$

Because the peturbations are small, we can assume there is no change in entropy. Under the homentropic flow assumption, the pressure peturbations and velocity peturbations are related through the (constant) sound speed.

$$p' = a_0^2 \rho' \tag{4.2}$$

After substituting Eq. (4.1) and Eq. (4.2) into the Euler equations, we arrive at the linear acoustic equations.

$$\begin{aligned} \frac{\partial p'}{\partial t} + \rho_0 a_0^2 \left(\nabla \cdot \mathbf{u}'\right) = 0 \\ \frac{\partial \mathbf{u}'}{\partial t} + \frac{1}{\rho_0} \nabla p' = 0 \end{aligned} \tag{4.3}$$

We non-dimensionalize the pressure $p^* = p'/\rho_0 a_0^2$ and velocity $\mathbf{u}^* = (1/a_0)\,\mathbf{u}'$ and write as a first-order system:

$$\frac{\partial p^*}{\partial t} + a_0 \left(\frac{\partial u^*}{\partial x} + \frac{\partial v^*}{\partial y} + \frac{\partial w^*}{\partial z} \right) = 0 \tag{4.4a}$$

$$\frac{\partial u^*}{\partial t} + a_0 \frac{\partial p^*}{\partial x} = 0 \tag{4.4b}$$

$$\frac{\partial v^*}{\partial t} + a_0 \frac{\partial p^*}{\partial y} = 0 \tag{4.4c}$$

$$\frac{\partial w^*}{\partial t} + a_0 \frac{\partial p^*}{\partial x} = 0 \tag{4.4d}$$

Let the state vector $\mathbf{q} = (p^*, u^*, v^*, w^*)^{\mathrm{T}}$. The sound speed a_0 is constant, so we can easily write the system in conservation form:

$$\frac{\partial \mathbf{q}}{\partial t} + \frac{\partial \mathbf{f}}{\partial x} + \frac{\partial \mathbf{g}}{\partial y} + \frac{\partial \mathbf{h}}{\partial z} = 0 \tag{4.5}$$

where

$$\mathbf{f} = \begin{pmatrix} a_0 u^* \\ a_0 p^* \\ 0 \\ 0 \end{pmatrix} \qquad \mathbf{g} = \begin{pmatrix} a_0 v^* \\ 0 \\ a_0 p^* \\ 0 \end{pmatrix} \qquad \mathbf{h} = \begin{pmatrix} a_0 w^* \\ 0 \\ 0 \\ a_0 p^* \end{pmatrix} \tag{4.6}$$

Equation (4.6) defines the flux expressions that update the conserved variables,

but we also need a means of calculating the point values that appear. Rewriting the system defined by Eq. (4.4) makes it easier to see possible paths forward. When the time deriviative of Eq. (4.4a) and the gradient of Eqns. (4.4b-4.4d) are combined, it is clear that pressure obeys the wave equation.

$$\frac{\partial^2 p^*}{\partial t^2} - a_0^2 \, \nabla^2 p^* = 0 \tag{4.7}$$

Similarly, taking the time deriviative of the velocity terms from Eq. (4.4) and spatial derivatives of Eq. (4.4a) results in the following system for the velocity.

$$\frac{\partial^2 \mathbf{u}^*}{\partial t^2} - a_0^2 \, \nabla \left(\nabla \cdot \mathbf{u}^* \right) = 0 \tag{4.8}$$

We can rewrite the equation such that the left hand side has the form of the wave equation:

$$\frac{\partial^2 \mathbf{u}^*}{\partial t^2} - a_0^2 \, \nabla^2 \mathbf{u}^* = a_0^2 \left(\nabla \times \boldsymbol{\omega} \right) \tag{4.9}$$

where the vorticity $\boldsymbol{\omega} = \nabla \times \mathbf{u}^* = (\partial_y w^* - \partial_z v^*, \partial_z u^* - \partial_x w^*, \partial_x v^* - \partial_y u^*)^{\mathrm{T}}$. Equation (4.9) clearly shows that velocity obeys the wave equation in flows with constant (or zero) vorticity. For this subset of problems, the pressure and velocity at any point in time can be calclated using exact solutions to the wave equation. Our strategy for linear acoustics is exactly the same as for the linear advection case. The first step is to find the pressure and velocity at a specified set of quadrature points, which we then use to construct an average flux at the interface. We detail this process for one-dimensional and two-dimensional flows below.

4.1 One-dimensional linear acoustics

While the pressure and velocity satisfy the scalar wave equation in one dimension, the similarities between the one-dimensional acoustics and advection approaches are

easier to see when the system is left as a set of first-order PDEs. The one-dimensional acoustics system has the following form:

$$\frac{\partial \mathbf{q}}{\partial t} + \mathbf{A}\frac{\partial \mathbf{q}}{\partial x} = 0 \tag{4.10}$$

where the state vector $\mathbf{q} = (p^*, u^*)^{\mathrm{T}}$ and the Jacobian matrix $\mathbf{A}$ is

$$\mathbf{A} = \begin{pmatrix} 0 & a_0 \\ a_0 & 0 \end{pmatrix} \tag{4.11}$$

The acoustic system can be diagonalized by performing an eigenvalue decomposition of the Jacobian matrix.

$$\begin{aligned} \mathbf{A} &= \mathbf{R}\mathbf{\Lambda}\mathbf{L} \\ &= \begin{pmatrix} 1/2 & 1/2 \\ -1/2 & 1/2 \end{pmatrix} \begin{pmatrix} -a_0 & 0 \\ 0 & a_0 \end{pmatrix} \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix} \end{aligned} \tag{4.12}$$

Multiplying Eq. (4.10) by $\mathbf{L}$ results in a decoupled system of two linear advection equations for the characteristic variables $\mathbf{w} = \mathbf{L}\mathbf{q}$.

$$\frac{\partial \mathbf{w}}{\partial t} + \mathbf{\Lambda}\frac{\partial \mathbf{w}}{\partial x} = 0 \tag{4.13}$$

Once the system is diagonalized, each characteristic field can be updated using the linear advection techniques from Chapter II. The opposite signs for the characteristic speed means that each interface will be updated from both of the cells sharing the edge. Therefore the algorithm must update the interface point values by summing contributions from each cell. The contribution $\delta\mathbf{q}$ each cell makes to the interface

state can be calculated as a sum over the k characteristic fields:

$$\begin{aligned}\delta\mathbf{q}_i &= \sum_k w_k(\xi_{0_k})\mathbf{r}_k \\ &= \sum_k w_k(\xi_i - \nu_k)\mathbf{r}_k\end{aligned} \tag{4.14}$$

where, w_k is the value of the characteristic variable at the origin for that field ξ_{0_k}, and $\mathbf{r}_k$ is the k^{th} column of the right eigenvector matrix. The value of the Courant number ν will be different for every field because, in general, each field will be associated with a different characteristic speed. If the point ξ_0 lies outside the cell, w_k is set to zero and the cell does not modify the value of that variable.

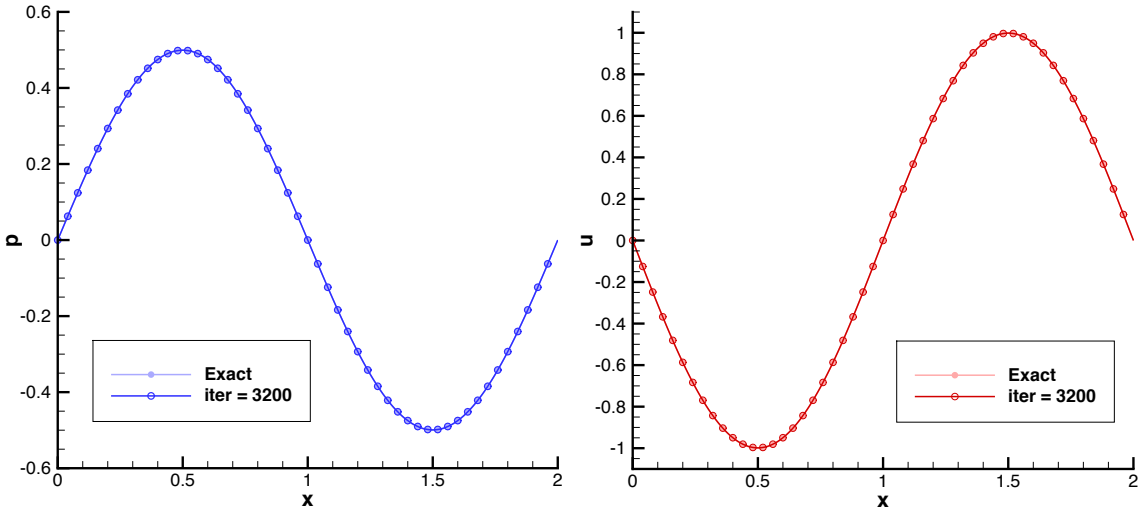
4.1.1 Simple wave

A simple wave is a region in which one of the characteristic variables, or Riemann invariants, is constant. The simplest method of initializing the flow field for a simple wave case is to choose the initial states such that one of the characteristic variables is zero. The initial conditions prescribed by Eq. (4.15) ensure that $w_2 = p^* + u^* = 0$ while keeping $w_1 = p^* - u^* \geq 0$.

$$\begin{aligned}p'(x,0) &= \frac{1}{2}\sin(\pi x) \\ u'(x,0) &= -\sin(\pi x) \\ a_0 &= 1 \\ \rho_0 &= 1/2\end{aligned} \tag{4.15}$$

Because there is only one non-zero characteristic variable, the one associated with λ_1, the exact solution for the simple wave problem is the initial data from Eq. (4.15) advected by a distance $\lambda_1 t = -a_0 t$. The one-dimensional acoustics implementation was tested on a uniform mesh with 50 cells and $x \in [0, 2]$. With the speed chosen as $a_0 = 1$, and a time step of $\Delta t = 0.03$ ($\nu = 0.75$), the wave completes three cycles

through the mesh every 200 iterations. Figure 4.1 shows the simulation results after 48 cycles through the mesh. The symbols denote the vertex values. The results at $t = 96$ are visually indistinguishable from the exact solution. The third-order method handles this smooth case exceptionally well, losing only 0.18% of the peak value in the pressure and velocity over the course of the simulation.



(a) Pressure (b) Velocity

Figure 4.1: Simple wave test after 48 cycles through mesh

4.1.2 Non-simple wave

When both characteristic variables vary over the domain, the solution is slightly more complex. The exact solution still has a simple form due to the linear nature of the problem.

$$\begin{aligned} w_1(x,t) &= w_1(x + a_0 t, 0) \\ w_2(x,t) &= w_2(x - a_0 t, 0) \end{aligned} \tag{4.16}$$

Re-writing in terms of the state variables:

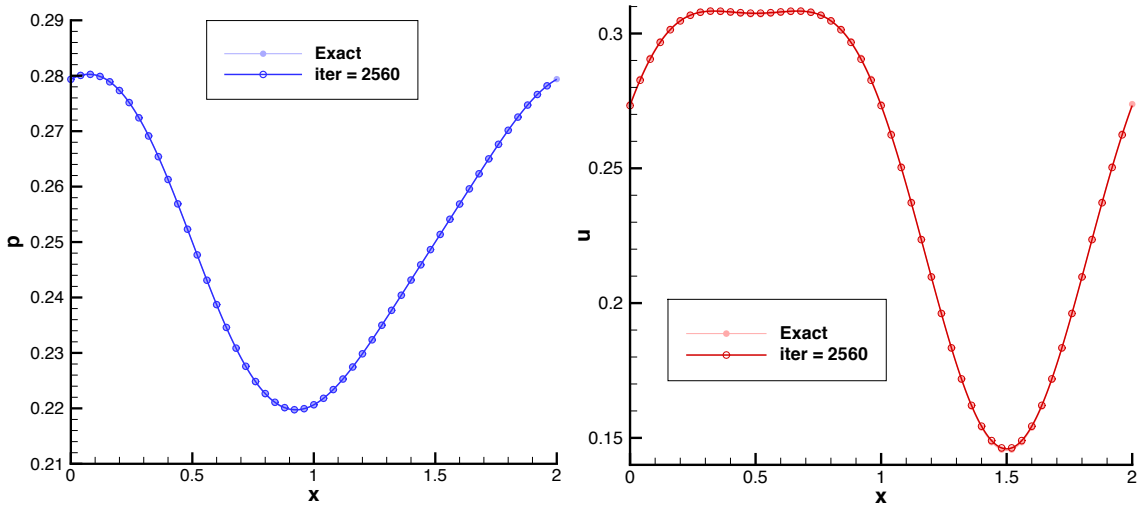
$$\begin{aligned} p^*(x,t) &= \frac{1}{2}\left[p^*(x-a_0t,0) + u^*(x-a_0t,0) + p^*(x+a_0t,0) - u^*(x+a_0t,0)\right] \\ u^*(x,t) &= \frac{1}{2}\left[p^*(x-a_0t,0) + u^*(x-a_0t,0) - p^*(x+a_0t,0) + u^*(x+a_0t,0)\right] \end{aligned} \tag{4.17}$$

The same domain and time step were used for the non-simple case. The initial conditions used are defined in Eq. (4.18).

$$\begin{aligned} p'(x,0) &= \frac{1}{4} + \frac{1}{80}\sin(2\pi x) \\ u'(x,0) &= \frac{1}{4} - \frac{1}{10}\sin(\pi x) \\ a_0 &= 1 \\ \rho_0 &= 1/2 \end{aligned} \tag{4.18}$$

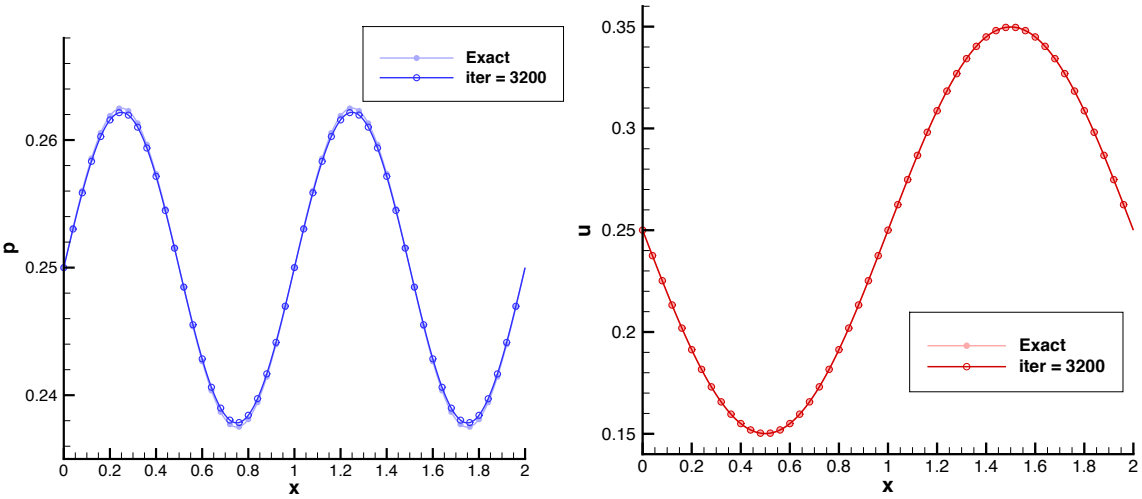
Figure 4.2 compares the exact and computed results far into the simulation at $t = 76.8$. Again, the AF solution nearly matches the exact solution. The deviation between the two solutions ranges from 0.04% to 0.13%. The difference between the exact and computed solution is slightly more apparent in Fig. 4.3. The higher frequency pressure wave leads to more dissipation than the simple wave case, but the peak value is reduced by just 0.13% after 48 cycles through the mesh. The computed velocity peak is 0.06% less than the exact value.

Both the simple and non-simple tests show that the AF scheme can accurately compute solutions for the one-dimensional wave equation, and by extension, any one-dimensional linear system. The key is computing the point values in such a way that each cell simply adds its individual contribution to the edge. As with linear advection, the algorithm has one loop through the cells to update the point values and one loop through the faces to calculate the average flux at each interface and update the conserved cell averages.



(a) Pressure (b) Velocity

Figure 4.2: Non-simple wave test; $t = 76.8$



(a) Pressure (b) Velocity

Figure 4.3: Non-simple wave test; $t = 96.0$

4.2 Two-dimensional linear acoustics

The behavior of the wave equation in even dimensions is significantly different from odd dimensions [8]. In one and three dimensions, the solution at any point in the flow depends only on the boundary of a region in the flow field; however, in two-dimensional flows, the domain of dependence for a point $\mathbf{x} = (x, y)$ at time t is the *interior* of a circle with radius $a_0 t$ centered at $\mathbf{x}$. There are two primary approaches

to numerically computing the solution while preserving the multidimensional nature of the physics. The first is to integrate around the bicharacteristic surface formed between the point of interest and the acoustic disc. The second approach is to directly integrate the domain of dependence.

4.2.1 Integral around circumference

Several computational methods for the scalar wave equation focus on approximating the integral around the bicharacteristic cone shown in Fig. 4.4. These methods include the evolution Galerkin method (EGM) of Lukáčová-Medvid'ová, Morton, and Warnecke [27], which is primarily based on work by Ostkamp [34], and methods by Butler [5] and Reddy [41]. As previously stated, the state at point F in Fig. 4.4 depends on a circle of radius $a_0\Delta t$ centered around point $\mathbf{x}$. The unit vector in

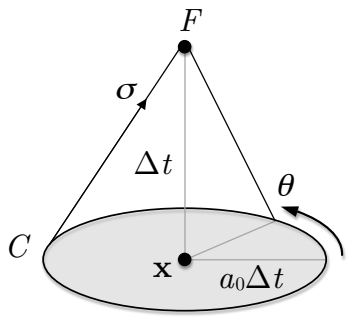


Figure 4.4: Two-dimensional characteristic cone

the characteristic direction σ can be found by subtracting the coordinates at point $F = (x, y, t + \Delta t)$ and $C = (x + a_0\Delta t \cos\theta, y + a_0\Delta t \sin\theta, t)$, which is on the characteristic manifold at time t.

$$\hat{\sigma} = \frac{1}{\sqrt{1 + a_0^2}} \begin{pmatrix} -a_0 \cos\theta \\ -a_0 \sin\theta \\ 1 \end{pmatrix} \tag{4.19}$$

The directional derivative is then:

$$\frac{\mathrm{d}}{\mathrm{d}\sigma} = -\frac{a_0 \cos\theta}{\sqrt{1+a_0^2}}\frac{\partial}{\partial x} - \frac{a_0 \sin\theta}{\sqrt{1+a_0^2}}\frac{\partial}{\partial y} + \frac{1}{\sqrt{1+a_0^2}}\frac{\partial}{\partial t} \tag{4.20}$$

The system in Eq. (4.4) can be multiplied by the proper weights and combined to obtain:

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}\sigma}\left(p - u\cos\theta - v\sin\theta\right) = & -a_0\left[\sin^2\theta\partial_x u - \cos\theta\sin\theta\left(\partial_y u + \partial_x v\right)\right. \\ & \left. + \cos^2\theta\partial_y v\right]\end{aligned} \tag{4.21}$$

Finally, integrating from point C to point F and then around the circumference of the characteristic footprint, we arrive at an expression for the pressure at point F.

$$\begin{aligned}p_F = & \frac{1}{2\pi}\int_0^{2\pi}\left(p_C - u_C\cos\theta - v_C\sin\theta\right)\mathrm{d}\theta \\ & - \frac{a_0}{2\pi}\int_t^{t+\Delta t}\int_0^{2\pi}\left[\sin^2\theta\partial_x u - \cos\theta\sin\theta\left(\partial_y u + \partial_x v\right) + \cos^2\theta\partial_y v\right]\mathrm{d}\theta\mathrm{d}\tau\end{aligned} \tag{4.22}$$

Expressions for the update of each velocity component are obtained by multiplying Eq. (4.3) first by $\cos\theta$ then by $\sin\theta$.

$$\begin{aligned}u_F = & \frac{1}{\pi}\int_0^{2\pi}\left(-p_C\cos\theta - u_C\cos^2\theta - v_C\cos\theta\sin\theta\right)\mathrm{d}\theta \\ & + \frac{a_0}{\pi}\int_t^{t+\Delta t}\int_0^{2\pi}\cos\theta\left[\sin^2\theta\partial_x u - \cos\theta\sin\theta\left(\partial_y u + \partial_x v\right) + \cos^2\theta\partial_y v\right]\mathrm{d}\theta\mathrm{d}\tau\end{aligned} \tag{4.23}$$

$$\begin{aligned}v_F = & \frac{1}{\pi}\int_0^{2\pi}\left(-p_C\sin\theta - u_C\cos\theta\sin\theta - v_C\sin^2\theta\right)\mathrm{d}\theta \\ & + \frac{a_0}{\pi}\int_t^{t+\Delta t}\int_0^{2\pi}\sin\theta\left[\sin^2\theta\partial_x u - \cos\theta\sin\theta\left(\partial_y u + \partial_x v\right) + \cos^2\theta\partial_y v\right]\mathrm{d}\theta\mathrm{d}\tau\end{aligned} \tag{4.24}$$

The difficulty in using these expressions is approximating the integrals appearing in the equations. The numerical methods based on this idea are distinguished by how they treat the spatial and temporal integrals. Most of the methods treat the mesh points as quadrature points for the spatial integrals. The time integral appearing in the last term of Eqns. (4.22-4.24) is particularly challenging. Lukáčová-Medvid'ová et al. use a trapezoidal rule to approximate the value, but this is only second-order accurate. One could use Simpson's rule to achieve third-order accuracy, but this would require solution data at $t + \Delta t/2$, which is not readily available.

4.2.2 Spherical means

Spherical means may be used as an alternate method of calculating the exact solution to the initial-value problem for the scalar wave equation at an arbitrary point in space and time [8]. Define the spherical mean of a function f over a sphere

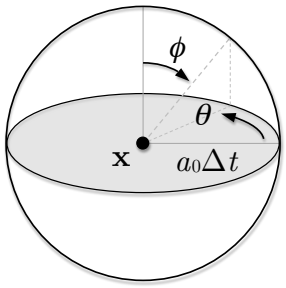


Figure 4.5: Sphere influenced by point $\mathbf{x}$; $R = a_0\Delta t$

of radius $R = a_0\Delta t$ to be its average over the surface:

$$M_R^{3D}\{f\}(x,y,z) = \frac{1}{4\pi R^2}\int_0^{2\pi}\int_0^{\pi} f\,(x + R\sin\phi\cos\theta, \\ y + R\sin\phi\sin\theta, z + R\cos\phi)\,R^2\sin\phi\;d\phi\;d\theta \tag{4.25}$$

We treat two-dimensional cases as a three-dimensional problem with no dependence on the z-coordinate. The method of descent allows us to convert the integral over the

spherical surface in Fig. 4.5 to an integral over the shaded disc.

$$M_R^{2D}\{f\}(x,y) = \frac{1}{2\pi R}\int_0^{2\pi}\int_0^{R} f(x + r\cos\theta, y + r\sin\theta)\frac{r}{\sqrt{R^2 - r^2}}dr\ d\theta \tag{4.26}$$

By using spherical means, we transform the original PDE into a form with a simple, and known, exact solution. Courant and Hilbert [8] provide the following solution for the wave equation $\partial_{tt}u = a_0^2\nabla^2 u$:

$$u(t) = tM_R\{\partial_t u\} + \partial_t\left[tM_R\{u\}\right] \tag{4.27}$$

We can expand the second term using the product rule to get:

$$u(t) = tM_R\{\partial_t u\} + M_R\{u\} + t\partial_t M_R\{u\} \tag{4.28}$$

The time derivative in the last term of Eq. (4.28) is cumbersome, so we seek alternate forms. We are able to replace the time derivative with a derivative with respect to R using the relationship $R = a_0 t$.

$$t\partial_t M_R\{u\} = t\partial_R M_R\{u\}\frac{\partial R}{\partial t} = R\partial_R M_R\{u\} \tag{4.29}$$

Substituting the expression Eq. (4.29) into Eq. (4.28) results in the final expression for $u(t)$:

$$u(t) = tM_R\{\partial_t u\} + M_R\{u\} + R\partial_R M_R\{u\} \tag{4.30}$$

All that remains is to apply the exact solution to the acoustic wave equation. We can use Eq. (4.4) to replace the time derivatives in Eq. (4.30) with spatial derivatives.

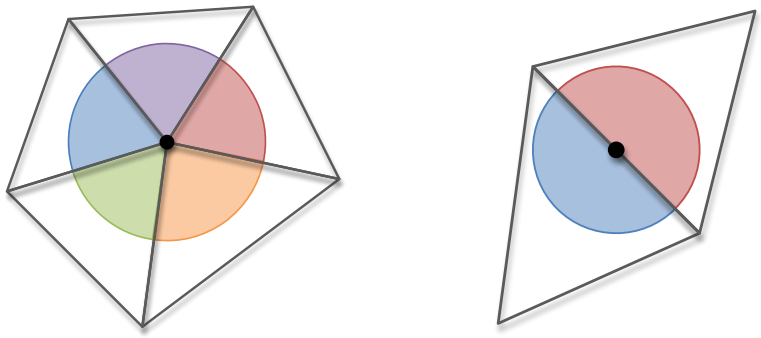
Assuming constant vorticity, we get the following update equations:

$$\begin{aligned} p^*(t) &= M_R\{p^*\} + R\left[\partial_R M_R\{p^*\} - M_R\{\text{div } \mathbf{u}^*\}\right] \\ \mathbf{u}^*(t) &= M_R\{\mathbf{u}^*\} + R\left[\partial_R M_R\{\mathbf{u}^*\} - M_R\{\nabla p^*\}\right] \end{aligned} \tag{4.31}$$

In flows with varying vorticity, the term on the right-hand side of Eq. (4.9) can be treated as a constant source term [8].

4.2.3 Integrals of the disc interior

We perform the integrals appearing in Eq. (4.31) over discs of radius $a_0 t$ centered at the solution nodes. Each cell that shares that node contributes to the integral over one sector, making the method upwind. In one-dimensional flow, the method of spherical means collapses to the method of characteristics [8] so that our method collapses to a regular upwind scheme in one dimension. Because we only seek updates



(a) Cells contributing to node update

(b) Cells contributing to edge update

Figure 4.6: Integrated areas for two-dimensional acoustics

at the vertices and edge-midpoints, the acoustic integral will either be centered at a node or edge, as pictured in Fig. 4.6. Within each element, the integral of the function is simply the sum of the integrals of each basis, multiplied by the appropriate weight.

Introducing a consistent nomenclature allows us to generalize the integral expressions by exploiting the fact that the elements are triangular. We number the nodes of

an element in a counterclockwise order, with the edges and their associated normals numbered to match the opposing node. The local element numbering is illustrated in Fig. 4.7.

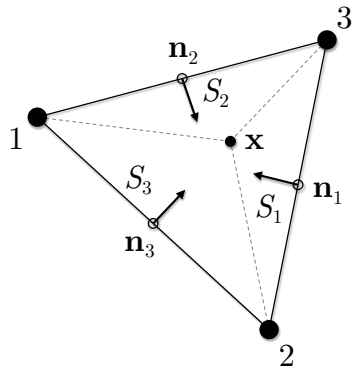
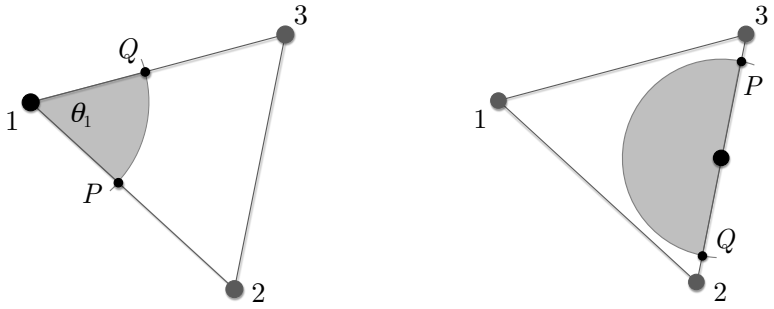


Figure 4.7: Element nomenclature

The intersection of the integral disc with the element edges sets the angular limits of integration. Point P and Q in Fig. 4.8 correspond to radial angles for a circle centered at the solution node. For example, the intersection angle for a disc centered at node 1 could be calculated as $\theta_Q = \tan^{-1}\left[(y_Q - y_1)\,/\,(x_Q - x_1)\right]$. Point P is always defined as the smaller angle. Following standard practice, angular values increase in the counterclockwise direction.



(a) Centered at node 1 (b) Centered at edge 1

Figure 4.8: Element integral (shaded area) for node and edge centered values

Any point $\mathbf{x}$ within the triangular element can be described by the area coordinates

S_1, S_2, and S_3, defined below. The total element area $S = S_1 + S_2 + S_3$.

$$\begin{aligned} S_1 &= \frac{1}{2}\left[(x_2 - x)(y_3 - y) - (y_2 - y)(x_3 - x)\right] \\ S_2 &= \frac{1}{2}\left[(x_3 - x)(y_1 - y) - (y_1 - y)(x_3 - x)\right] \\ S_3 &= \frac{1}{2}\left[(x_1 - x)(y_2 - y) - (y_2 - y)(x_1 - x)\right] \end{aligned} \tag{4.32}$$

We can then re-write the reference coordinates ξ and η in terms of the area coordinates.

$$\begin{aligned} \eta &= \frac{S_3}{S} \\ \xi &= \frac{S_2}{S} \\ 1 - \xi - \eta &= \frac{S_1}{S} \end{aligned} \tag{4.33}$$

Substituting these expressions into those in Table 2.2, we find that the edge basis functions and nodal basis functions have two basic forms. Equation (4.34) defines the basis functions in terms of area coordinates, where i represents a node index, j represents an edge index, and it is understood that the indices cycle (e.g. $3 + 1 = 1$ and $1 - 1 = 3$).

$$\begin{aligned} \phi_{2i-1} &= \frac{S_i(2S_i - S)}{S^2} \\ \phi_{2j} &= \frac{4S_{j-1}S_{j+1}}{S^2} \end{aligned} \tag{4.34}$$

The bubble function basis is the product of the area coordinates.

$$\phi_7 = \frac{27 S_1 S_2 S_3}{S^3} \tag{4.35}$$

The chain rule can be used to find expressions for the basis derivatives in terms of

the area coordinates.

$$\begin{aligned} \nabla\phi_{2i-1} &= \left(\frac{4S_i - S}{2S^2}\right)\mathbf{n}_i \\ \nabla\phi_{2j} &= \frac{2}{S^2}\left(S_{j+1}\mathbf{n}_{j-1} + S_{j-1}\mathbf{n}_{j+1}\right) \end{aligned} \tag{4.36}$$

The derivative of the bubble function is the sum of three terms:

$$\nabla\phi_7 = \frac{27}{2S^3}\sum_{j=1}^{3} S_{j-1}S_{j+1}\mathbf{n}_j \tag{4.37}$$

The normal vectors appearing in the derivative expressions Eq. (4.36) and Eq. (4.37) are not unit vectors. They have a length equal to the length of the side with which they are associated.

$$\mathbf{n}_k = \begin{pmatrix} y_{k+1} - y_{k-1} \\ x_{k-1} - x_{k+1} \end{pmatrix} \tag{4.38}$$

Fortunately, trigonometric identities can be used to simplify the resulting integral expressions to a form that is easily coded. The integrated basis functions and derivatives are listed in Appendix C. We have chosen to implement the closed form of the integrals; however, it may be more efficient to carry out the integration numerically. Algorithm 4.1 details the point-update procedure for linear acoustics.

4.2.4 Comparison of radial symmetry

One method of evaluating an acoustics method is to investigate its ability to maintain radial symmetry by plotting the solution at a specified time as a function of the distance from an initial disturbance. If the solution is perfectly symmetric, a scatter plot of the individual cell values plotted against the radial distance from the disturbance should collapse to a single line. Equation (4.39) defines the initial conditions used to evaluate the AF implementation, where the disturbance center $(x_0, y_0) = 0$ and $\mu = 50$. The sound speed was set at $a_0 = 1$ and the free stream

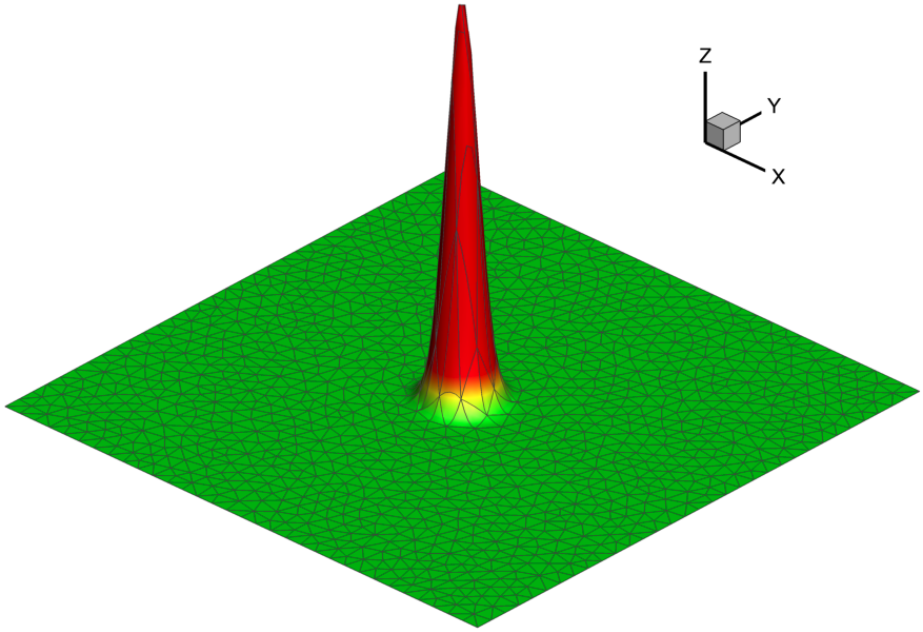
Algorithm 4.1 Point update procedure for linear acoustics

```
for cell nodes and edges (basis index = 1,6) do
  for τ = 1/2Δt and Δt do
    Integrate from θ ∈ [θ_P, θ_Q] and r = [0, a_0 τ] using Eq. (C.1), Eq. (C.19)
    Combine integrals to calculate partial updates; Eq. (4.31)
    if basis index is odd then
      Add contribution to vertex value
    else
      Add contribution to edge value
    end if
  end for
end for
```

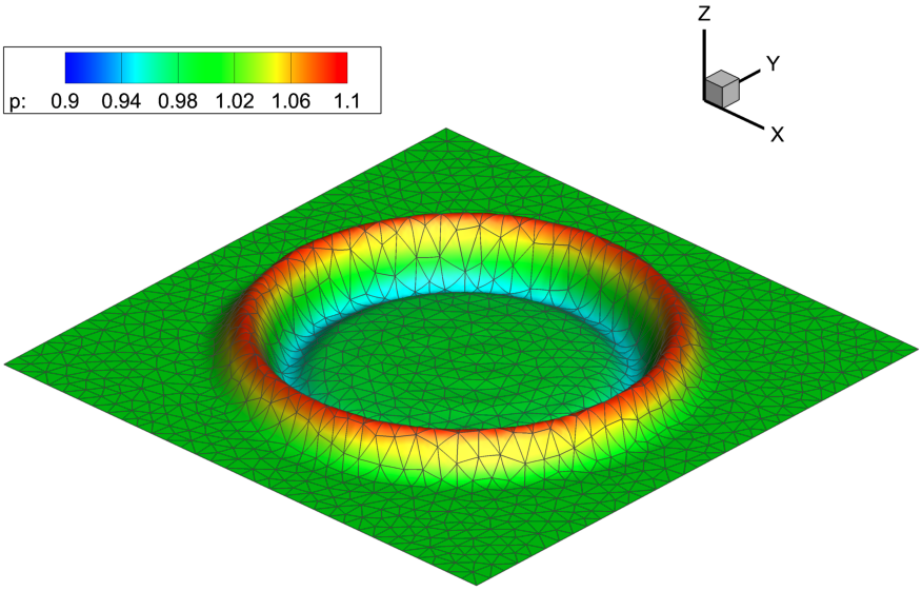
density was set to $\rho_0 = 1.4$.

$$\begin{aligned} p'(\mathbf{x}, 0) &= 1 + \exp\left\{-\mu\left[(x - x_0)^2 + (y - y_0)^2\right]\right\} \\ \mathbf{u}'(\mathbf{x}, 0) &= 0 \end{aligned} \tag{4.39}$$

The AF scheme was compared to two structured, second-order, schemes: a vorticity-preserving scheme of Morton and Roe [28], supplemented by a flux-corrected transport (FCT) limiter and a more traditional MUSCL-Hancock method [51]. Three mesh densities were tested in which the number of elements contained within the structured, quadrilateral mesh and unstructured, triangular mesh were approximately equal. A square domain was used with $x \in [-2, 2]$ and $y \in [-2, 2]$. Figure 4.9 shows the initial condition and AF solution at $t = 1.25$. All of the schemes used a time step of $\Delta t = 2.5 \times 10^{-2}$ to march to a final time of $t = 1.25$. Figures 4.10-4.12 show the AF result, the MUSCL-Hancock method (MUSCL), and vorticity-preserving (FCT) solutions for the coarse, medium and fine mesh densities. In all of the plots, the AF scheme shows superior symmetry properties in both pressure and velocity magnitude.

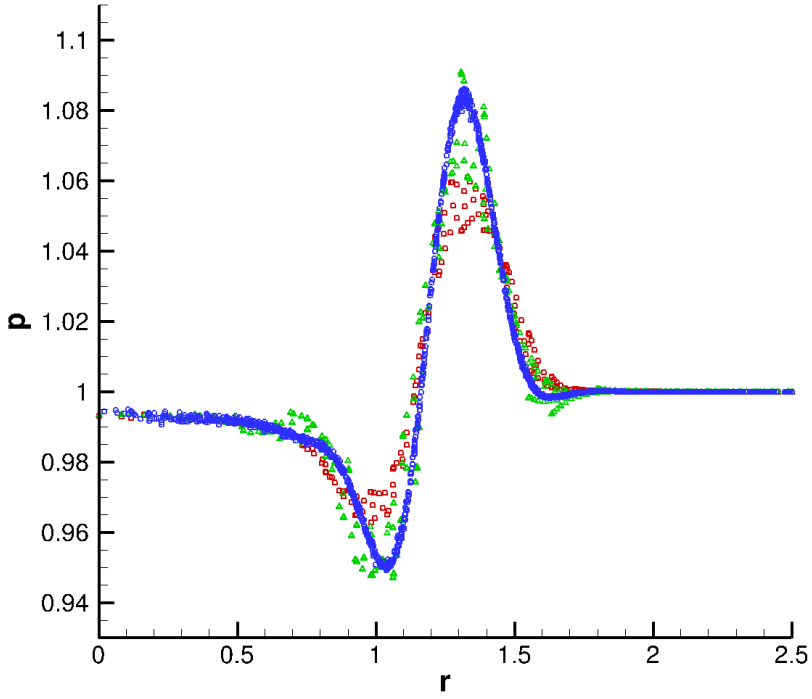


(a) $t = 0$

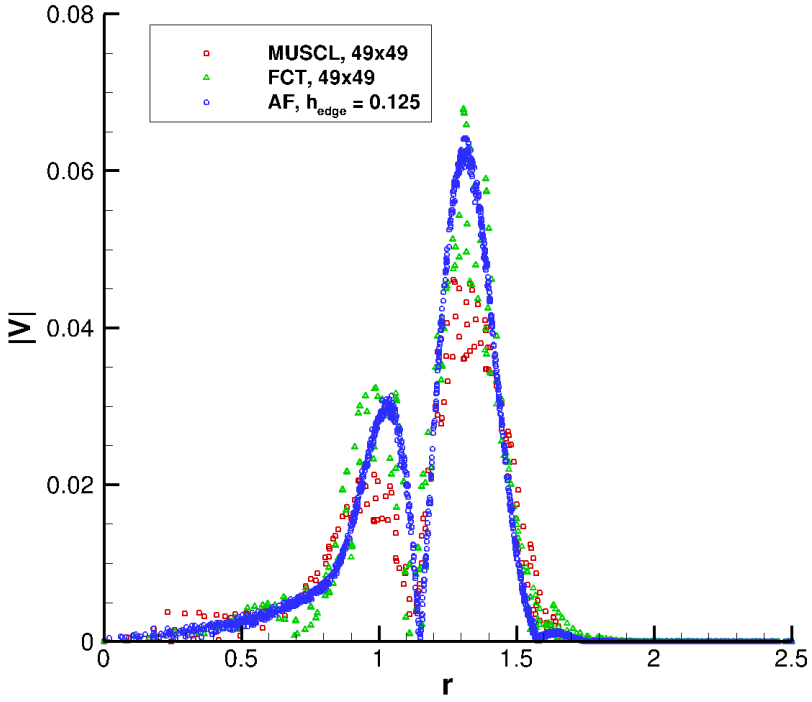


(b) $t = 1.25$

Figure 4.9: AF pressure solution for Gaussian pulse initial condition on the coarse mesh

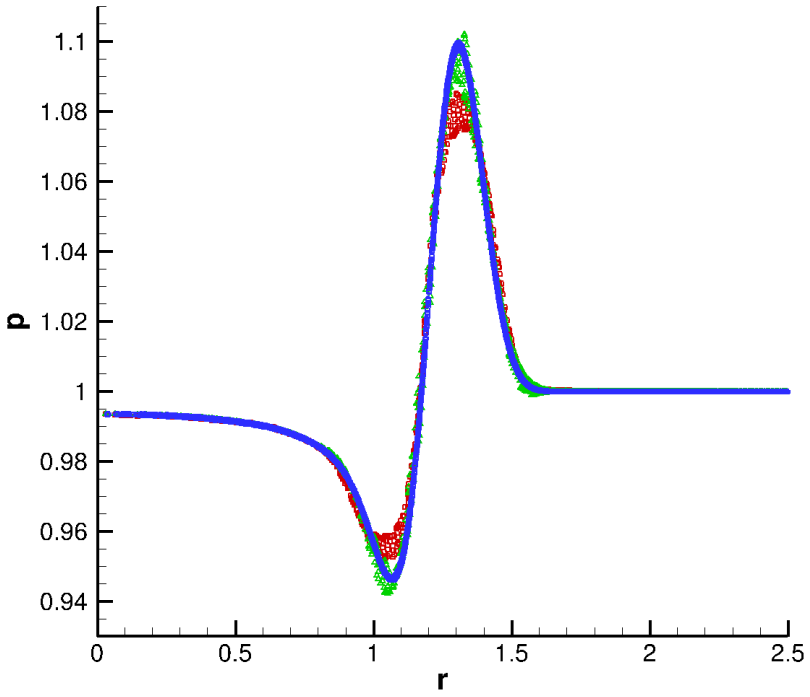


(a) Pressure

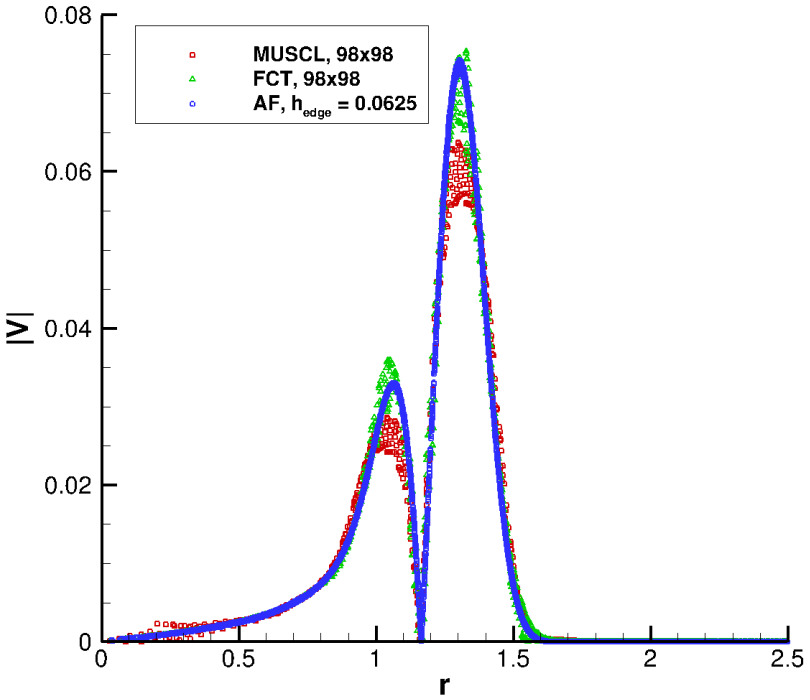


(b) Velocity magnitude

Figure 4.10: Coarse mesh

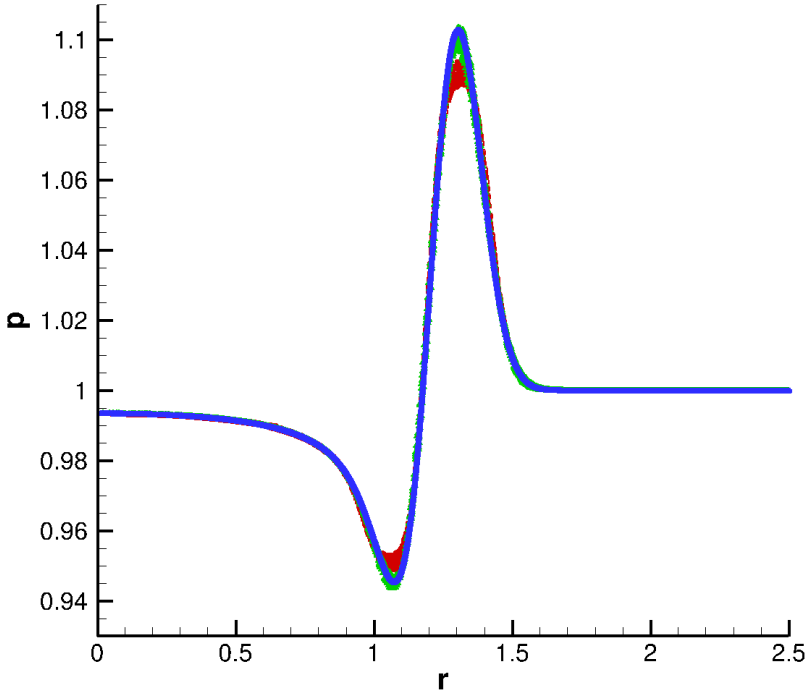


(a) Pressure

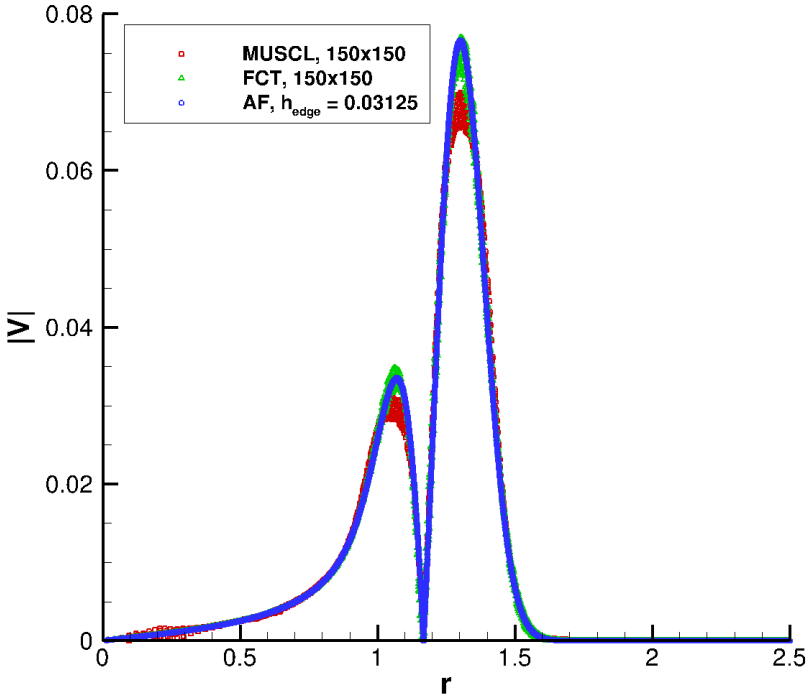


(b) Velocity magnitude

Figure 4.11: Medium mesh



(a) Pressure



(b) Velocity magnitude

Figure 4.12: Fine mesh

4.2.5 Comparison to exact solution

The order of accuracy of the acoustics method was confirmed by initializing the domain using the conditions proposed by Lukáčová-Medvid'ová, Morton, and Warnecke [27]:

$$\begin{aligned} p'(\mathbf{x},0) &= \frac{1}{a_0}\left[\sin(2\pi x)+\sin(2\pi y)\right] \\ u'(\mathbf{x},0) &= v'(\mathbf{x},0) = 0 \end{aligned} \tag{4.40}$$

They give the exact solution to Eq. (4.40) as:

$$\begin{aligned} p'(\mathbf{x},t) &= \frac{1}{a_0}\cos(2\pi a_0 t)\left[\sin(2\pi x)+\sin(2\pi y)\right] \\ \mathbf{u}'(\mathbf{x},t) &= \frac{1}{a_0}\sin(2\pi a_0 t)\cos(2\pi \mathbf{x}) \end{aligned} \tag{4.41}$$

It is clear from Eq. (4.41) that the exact solution matches the initial condition for integer values of $a_0 t$. To measure the order of accuracy, we set the sound speed to $a_0 = 1$ and evaluate the error at $t = 1$. Figure 4.13 and Table 4.2 both show that the AF scheme achieves third-order accuracy as the mesh is refined.

Table 4.1: Pressure convergence for linear acoustics

Ref. Level	$\mathbf{DOF}^{-1/2}$	$\|\boldsymbol{p}\|_{\mathbf{1}}$	$\boldsymbol{\mathcal{O}(L_1)}$
1	8.5436×10^{-2}	3.2661×10^{-1}	-
2	4.5502×10^{-2}	9.7451×10^{-2}	1.9197
3	2.3119×10^{-2}	1.3461×10^{-2}	2.9236
4	1.1681×10^{-2}	1.7868×10^{-3}	2.9580
5	5.8798×10^{-3}	2.2191×10^{-4}	3.0388
6	2.9379×10^{-3}	2.7876×10^{-5}	2.9899

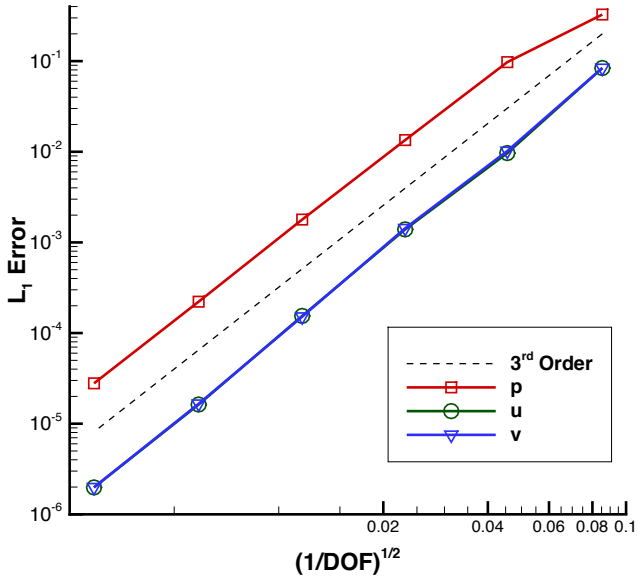


Figure 4.13: AF error convergence for acoustic test

Table 4.2: Velocity convergence for linear acoustics

Ref. Level	$\mathbf{DOF}^{-1/2}$	$\|u\|_1$	$\mathcal{O}(L_1)$	$\|v\|_1$	$\mathcal{O}(L_1)$
1	8.5436×10^{-2}	8.4059×10^{-2}	-	1.0430×10^{-1}	-
2	4.5502×10^{-2}	1.0200×10^{-2}	3.3477	1.2426×10^{-2}	3.3769
3	2.3119×10^{-2}	1.4328×10^{-3}	2.8988	1.6883×10^{-3}	2.9479
4	1.1681×10^{-2}	1.5168×10^{-4}	3.2894	1.9372×10^{-4}	3.1715
5	5.8798×10^{-3}	1.6585×10^{-5}	3.2244	2.1131×10^{-5}	3.2278
6	2.9379×10^{-3}	1.9890×10^{-6}	3.0567	2.6257×10^{-6}	3.0057

4.3 Combining advection and acoustics (linearized Euler)

Model equations such as the ones detailed in Chapter II, Chapter III and the linear acoustics system of this chapter are used to verify aspects of a numerical scheme in preparation for applying the scheme to more complex physical systems. The Euler equations accurately describe fluid behavior when viscous effects are negligible. In conservative form, the two-dimensional equations read:

$$\frac{\partial \mathbf{q}}{\partial t} + \frac{\partial \mathbf{f}}{\partial x} + \frac{\partial \mathbf{g}}{\partial y} = 0 \tag{4.42}$$

where,

$$\mathbf{q} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{pmatrix} \qquad \mathbf{f} = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uH \end{pmatrix} \qquad \mathbf{g} = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vH \end{pmatrix} \tag{4.43}$$

and,

$$\begin{aligned} E &= \frac{1}{\gamma - 1}\frac{p}{\rho} + \frac{1}{2}\left|\mathbf{u}\right|^2 \\ H &= E + \frac{p}{\rho} \end{aligned} \tag{4.44}$$

The conservative form obscures how the Euler equations contain nonlinear versions of the model problems studied thus far. Rewriting the equations using the primitive variables makes these terms clear:

$$\partial_t \rho + u\partial_x \rho + v\partial_y \rho + \rho\left(\partial_x u + \partial_y v\right) = 0 \tag{4.45a}$$

$$\partial_t u + u\partial_x u + v\partial_y u + \left(1/\rho\right)\partial_x p = 0 \tag{4.45b}$$

$$\partial_t v + u\partial_x v + v\partial_y v + \left(1/\rho\right)\partial_y p = 0 \tag{4.45c}$$

$$\partial_t p + u\partial_x x + v\partial_y p + \rho a^2\left(\partial_x u + \partial_y v\right) = 0 \tag{4.45d}$$

This form suggests three operators: advection, acoustics, and dilation.

$$\boxed{\begin{matrix}\partial_t \rho \\ \partial_t u \\ \partial_t v \\ \partial_t p\end{matrix}} + \boxed{\text{Advection}} + \boxed{\begin{matrix}\text{Dilation} \\ \hline \text{Acoustics}\end{matrix}} = 0$$

The idea behind operator splitting is to use known methods to solve each of the sub-problems in a way that reconstructs the solution to the full system after each time

step. To demonstrate this technique with the AF scheme, we develop an analogue for the nonlinear Euler system by adding advection terms to the linear acoustics equations, where the advection speed $\boldsymbol{\lambda} = (\alpha, \beta)^{\mathrm{T}}$.

$$\begin{aligned} \partial_t p^* + \alpha \partial_x p^* + \beta \partial_y p^* + a_0 \left(\partial_x u^* + \partial_y v^* \right) = 0 \\ \partial_t u^* + \alpha \partial_x u^* + \beta \partial_y u^* + a_0 \partial_x p^* = 0 \\ \partial_t v^* + \alpha \partial_x v^* + \beta \partial_y v^* + a_0 \partial_y p^* = 0 \end{aligned} \tag{4.46}$$

Rewriting in terms of linear coefficient matrices:

$$\frac{\partial \mathbf{q}}{\partial t} + \mathbf{A} \frac{\partial \mathbf{q}}{\partial x} + \mathbf{B} \frac{\partial \mathbf{q}}{\partial y} = 0 \tag{4.47}$$

where

$$\mathbf{q} = \begin{pmatrix} p^* \\ u^* \\ v^* \end{pmatrix} \qquad \mathbf{A} = \begin{pmatrix} \alpha & a_0 & 0 \\ a_0 & \alpha & 0 \\ 0 & 0 & \alpha \end{pmatrix} \qquad \mathbf{B} = \begin{pmatrix} \beta & 0 & a_0 \\ 0 & \beta & 0 \\ a_0 & 0 & \beta \end{pmatrix} \tag{4.48}$$

Both $\mathbf{A}$ and $\mathbf{B}$ have the form $\mathcal{L}_1 + \mathcal{L}_2$ where one operator represents the pure advection problem from Chapter II and the other represents the linear acoustics equation. Furthermore, it is easy to show that the operators commute ($\mathcal{L}_1 \mathcal{L}_2 = \mathcal{L}_2 \mathcal{L}_1$). We apply sequential operator splitting and solve each sub-problem separately. Because the operators commute, there is no error due to the splitting.

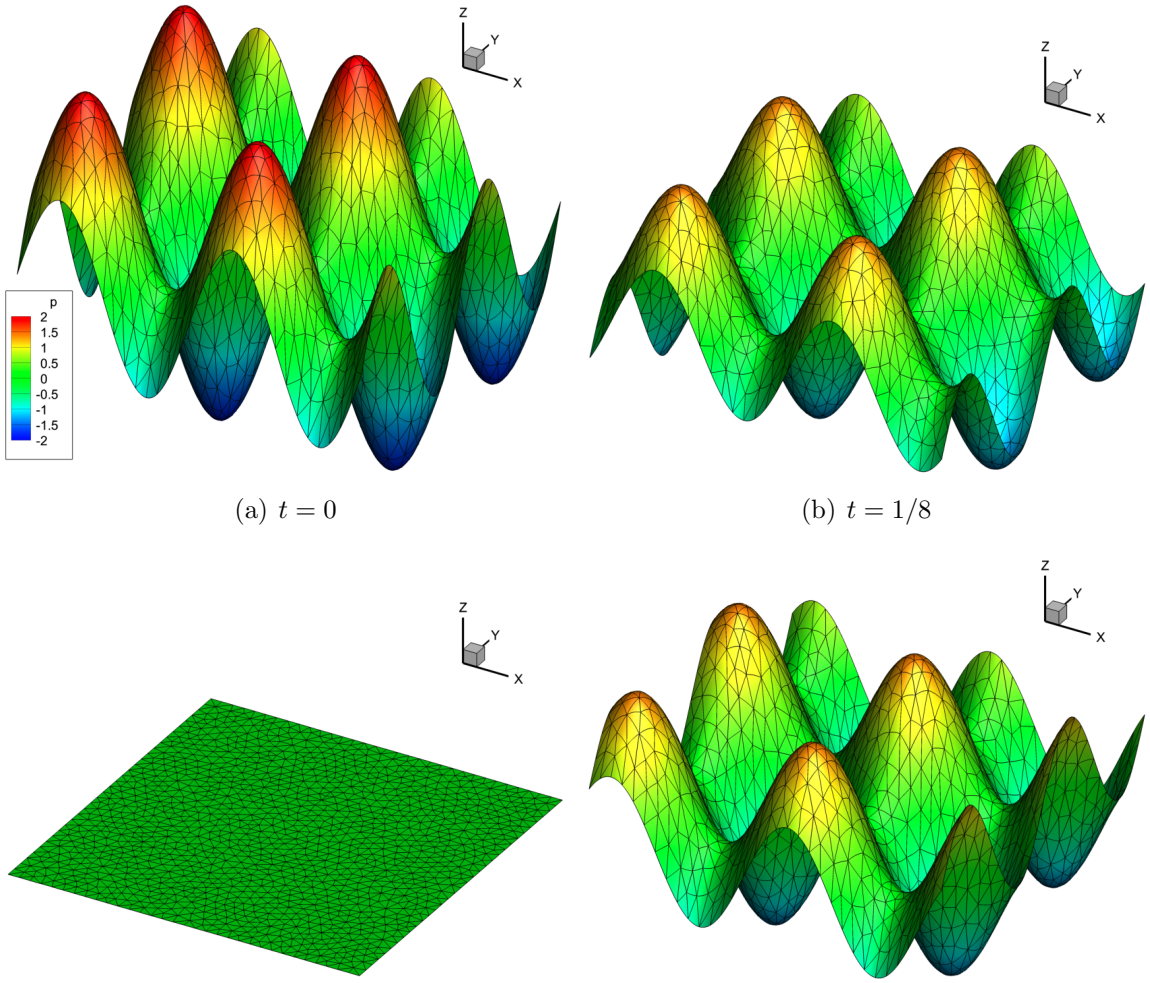
$$\begin{aligned} \tilde{\mathbf{q}} &= \mathbf{q}^n + \Delta t \mathcal{L}_1(\mathbf{q}) \\ \mathbf{q}^{n+1} &= \tilde{\mathbf{q}} + \Delta t \mathcal{L}_2(\mathbf{q}) \end{aligned} \tag{4.49}$$

Once methods for the advection and acoustics equations have been implemented, solving the combined case is trivial. Algorithmically, we find an intermediate solution $\tilde{\mathbf{q}}$ by computing the solution to the linear acoustics equation, directly followed by an

advection solve that uses the intermediate solution as the initial conditions. These steps may also be executed in the reverse order, that is, performing an advection solve followed by an acoustics solve.

To verify the implementation, we return to the pressure and velocity field specified in Eq. (4.40) and add a purely horizontal advection speed of $\alpha = 1$ ($\beta = 0$). The case was run with periodic boundary conditions. Setting the acoustic speed $a_0 = 1$, combined with an advection speed of $\boldsymbol{\lambda} = (1, 0)^{\mathrm{T}}$, yields an exact solution that returns to the initial state every two seconds. An order of accuracy suite was run on a square domain with $x \in [-1, 1]$ and $y \in [-1, 1]$. Figure 4.17 shows the error levels for the sequence of refined meshes and the convergence rates for pressure and velocity are listed in Table 4.3 and Table 4.4. The figure and tables verify that the AF scheme maintains third-order accuracy even with the operator splitting.

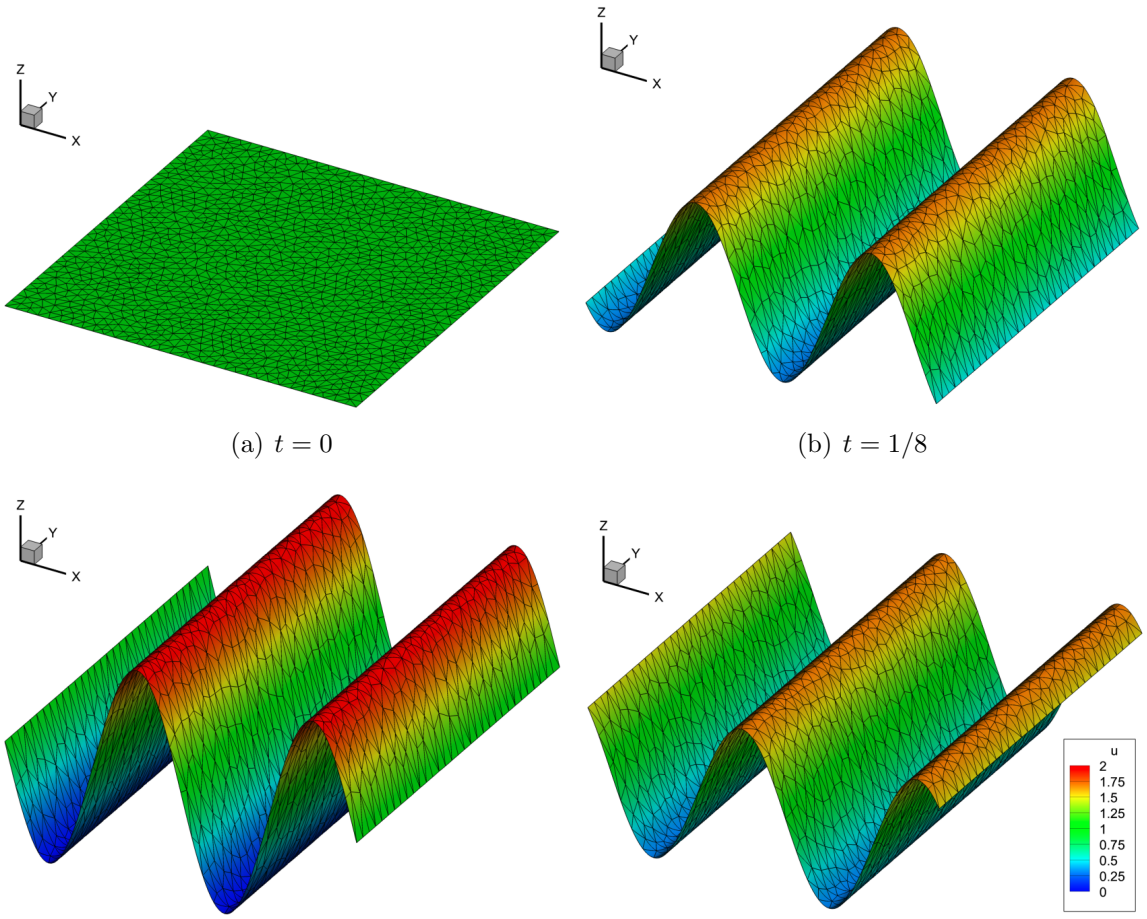
Demonstrating the AF method for the combined advection and acoustics case is an important step toward simulating more realistic flows. If a complex problem can be described by a series of sub-problems, we can leverage the flexibility of the AF scheme to obtain the solution of the simpler systems. All that is required is a means of updating variables at the interface and a flux function. The active flux acoustics tests demonstrate that the scheme can be combined with novel update approaches to systematically build accurate methods for complex flows.



(a) $t = 0$ (b) $t = 1/8$

(c) $t = 3/4$ (d) $t = 7/8$

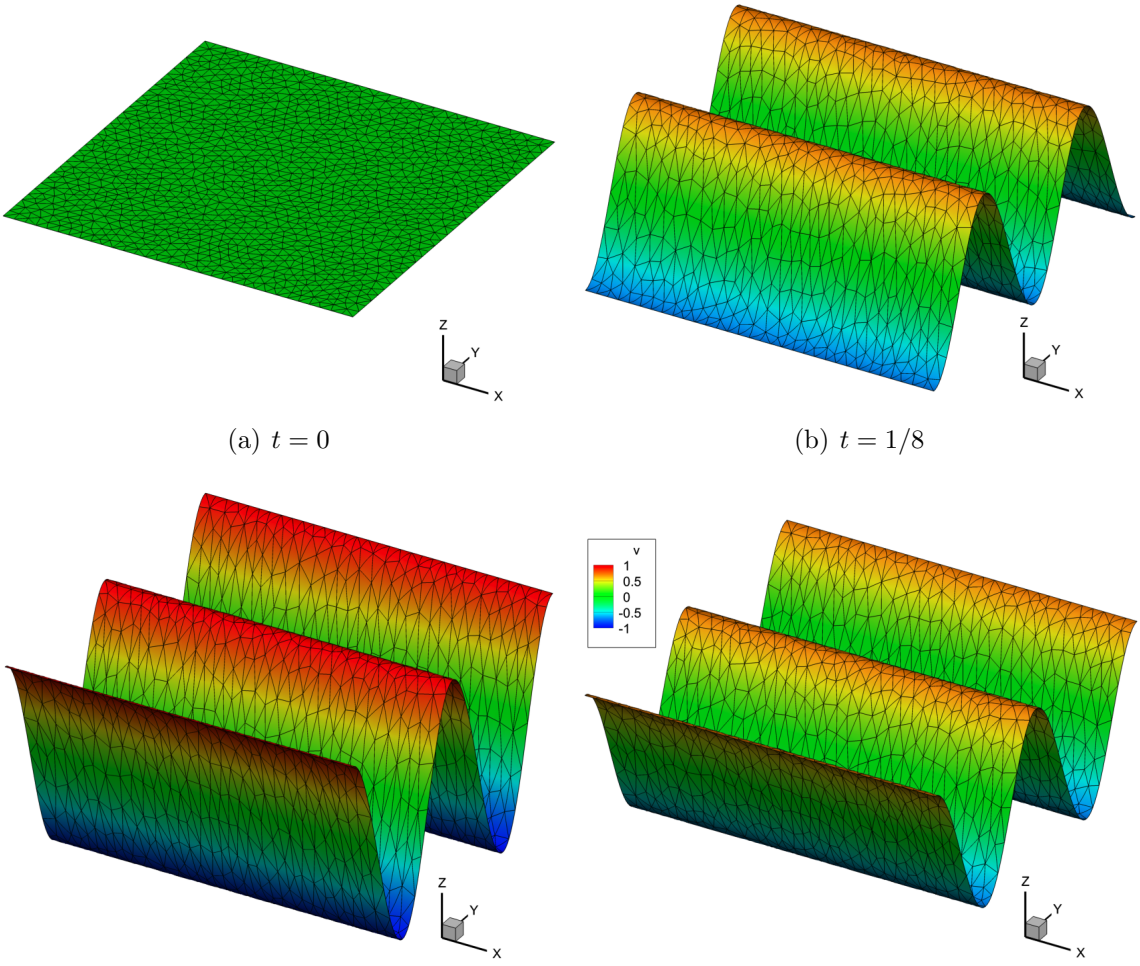
Figure 4.14: Pressure evolution for linearized Euler case



(a) $t = 0$ (b) $t = 1/8$

(c) $t = 3/4$ (d) $t = 7/8$

Figure 4.15: x-velocity evolution for linearized Euler case



(a) $t = 0$ (b) $t = 1/8$

(c) $t = 3/4$ (d) $t = 7/8$

Figure 4.16: y-velocity evolution for linearized Euler case

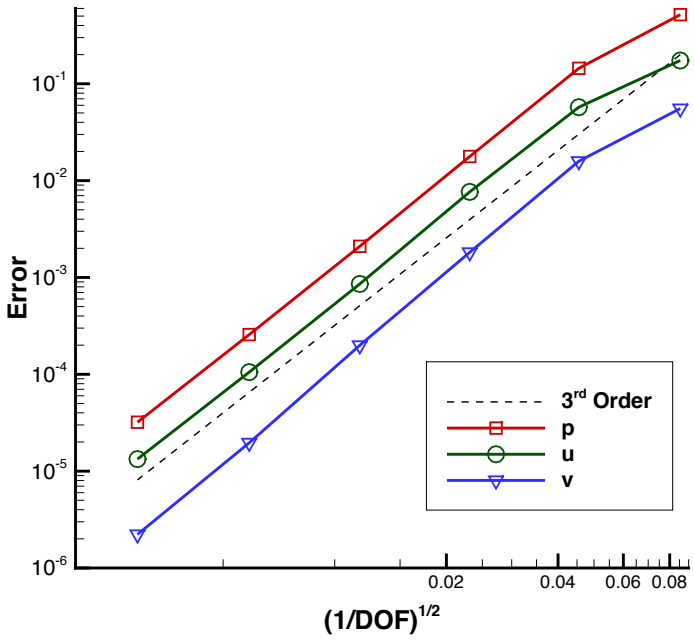


Figure 4.17: AF error convergence for linearized Euler equation

Table 4.3: Pressure convergence for linearized Euler equation

Ref. Level	**DOF**$^{-1/2}$	$\|p\|_1$	$\mathcal{O}(L_1)$
1	8.5436e-02	5.1728e-01	-
2	4.5502e-02	1.4452e-01	2.0240
3	2.3119e-02	1.7753e-02	3.0968
4	1.1681e-02	2.0975e-03	3.1286
5	5.8798e-03	2.5712e-04	3.0578
6	2.9379e-03	3.2014e-05	3.0027

Table 4.4: Velocity convergence for linearized Euler equation

Ref. Level	**DOF**$^{-1/2}$	$\|u\|_1$	$\mathcal{O}(L_1)$	$\|v\|_1$	$\mathcal{O}(L_1)$
1	8.5436×10^{-2}	5.1728×10^{-1}	-	1.9388×10^{-1}	-
2	4.5502×10^{-2}	1.4452×10^{-1}	1.9883	6.3559×10^{-2}	1.7702
3	2.3119×10^{-2}	1.7753×10^{-2}	3.1994	8.6957×10^{-3}	2.9377
4	1.1681×10^{-2}	2.0975×10^{-3}	3.2399	9.9351×10^{-4}	3.1777
5	5.8798×10^{-3}	2.5712×10^{-4}	3.3769	1.2217×10^{-4}	3.0533
6	2.9379×10^{-3}	3.2014×10^{-5}	3.1392	1.5747×10^{-5}	2.9528

CHAPTER V

Conclusions

5.1 Summary

This dissertation details the development of active flux schemes, a new class of methods for CFD. Active flux methods address three issues plaguing production-level CFD codes: reliance on one-dimensional Riemann solvers, second-order accuracy, and computational stencils that do not easily parallelize. The key concept is that edge and vertex values are updated and evolved independently from the conserved cell-average quantities. Interface values are then used to calculate fluxes that conservatively update the cell-averages. Because the edge updates do not need to be conservative, any convenient method can be used and proper attention can be given to multidimensional physics. The scheme uses parabolic reconstructions, with a cubic bubble function to maintain conservation in two dimensions, making it third-order accurate by construction. All of the reconstructions and updates are local to a single element, giving AF schemes a very compact stencil suitable for parallelization. Additionally, the AF method is fully discrete, advancing from time-level n to $n+1$ in a single step.

The method is demonstrated on the linear advection, linear acoustics, and linearized Euler equations in one and two dimensions, where the extension to three dimensions is straightforward. Each class of problem highlights different aspects of the AF method. The linear advection implementation covered in Chapter II shows

the basic mechanics of the update procedure and explicitly demonstrates how upwinding is incorporated. The linear acoustics equations are solved in Chapter IV using the fully multidimensional method of spherical means and the same chapter details how the freedom to use non-conservative edge updates allows radical departures from one-dimensional Riemann solutions. The end of Chapter IV also illustrates how the solution for the linearized Euler equations can be simply obtained by operator splitting. Because the scheme is third-order, oscillations develop in non-smooth regions of the flow. Chapter III discusses some one-dimensional limiter strategies for the AF method and demonstrates the scheme with the inviscid Burgers equation.

5.2 Contributions and conclusions

The major contributions of this dissertation are:

- Extension of van Leer's one-dimensional Scheme V into a general multidimensional framework applicable to linear conservation laws
- Development of a limiter for parabolic reconstructions that uses local information and is not overly dissipative
- Use of spherical means to obtain evolution equations for physical systems that obey the scalar wave equation

Chapter II shows that the AF method has several advantages over more traditional schemes. For one, the extra degrees of freedom within the cell mean that frequencies up to 2π can be resolved, which is double the frequency range for comparable FV schemes. The AF scheme has superior dissipation and dispersion properties, especially as the Courant number approaches one. The advantage of the compact stencil is also demonstrated by showing that the AF solution is far less sensitive to irregular meshes than a third-order FV scheme. The AF scheme achieves third-order accuracy using

two DOF per element in one dimension and three DOF in two dimensions. This is comparable with a DG1 scheme, which uses a linear reconstruction. If the DG scheme superconverges, the work needed to compute the solution for the AF and DG methods is approximately equal; however, superconvergence is not guaranteed. When the DG rate of convergence is only equal to $p+1$, the DG method requires six DOF per element for third-order accuracy, while the AF method needs approximately three. The increased number of DOF represents a significant workload increase for non-superconverging DG methods compared to the third-order AF solution.

Chapters III-IV clearly demonstrate the advantage of non-conservative edge updates. This feature makes the novel limiting approach possible and facilitates the use of spherical means to calculate edge and vertex updates. The flexibility of the approach is also highlighted by showing how trivial it is to extend the method to the linearized Euler equations. The AF method retains third-order accuracy even when these more exotic methods are used to update or modify interface values. The multidimensional treatment of the acoustics system allows the AF method to preserve excellent symmetry properties on the irregular triangular mesh. Both the acoustics and advection results demonstrate that the AF scheme is stable provided that the physical domain of dependence is contained within the nearest-neighbors of the point to be updated, giving the scheme the robustness and accuracy combination required of production-level codes.

5.3 Future work

This dissertation is meant to demonstrate the merit of the AF method and provide examples of its implementation. There are a wealth of research avenues that may proceed from this work.

- **Steady state:** The AF method is fully discrete, meaning that the solution is dependent on the time step. This means that it is possible to compute different steady-state solutions by using different time steps. Techniques can be developed to remove the time step dependence from the steady-state solution. One possible path forward is to use the first-order residuals of the sub-elements to produce time-independent flux values.

- **Computational efficiency:** There are numerous opportunities to improve the efficiency of the scheme. One obvious example is numerically integrating the spherical mean formulas rather than using the explicit form. Also, any term in the point-update formulas that is $\mathcal{O}(\Delta t^3)$ or higher can be neglected without losing third-order accuracy.

- **Three-dimensional flow:** All of the core ideas presented will extend to three-dimensions, but new expressions for the bubble function and acoustic integrals will need to be derived.

- **Alternate applications:** This dissertation illustrates the AF method as applied to the linear acoustics equation, but the derivation holds for any physical system that can be described by the wave equation, such as electromagnetics and linear elasticity. There are also integral solutions for the heat equation that are similar to those demonstrated for the wave equation. Additionally, the advection scheme from Chapter II may be of interest to the weather prediction and radiation hydrodynamics communities.

- **Multidimensional limiters:** It is difficult to imagine how the equations for the one-dimensional limiting technique in Chapter III directly extend to two and three dimensions. However, the concept of a shape-preserving limiter will hold and may provide useful insights toward limiting higher dimension flows.

- **Viscous terms:** The AF method must be able to solve the full Navier-Stokes equations in order to accurately compute near-body flows. One promising approach proposed by Nishikawa is to rewrite the Navier-Stokes equations as a first-order system of hyperbolic equations [30]. A drawback is that this formulation assumes a steady-state calculation, meaning subiterations would be needed within each time step for time-accurate computations.

- **Nonlinear equations:** The flexibility of the AF approach means that there are several possible ways to develop a solver capable of computing nonlinear Euler solutions. One possible way forward is to choose an appropriate linearization, which would define locally constant quantities, and apply the techniques from this dissertation. Initial investigations have been conducted for Burgers' equation but a much more detailed study will be required to find the best approach that extends to multidimensional systems of equations.

APPENDICES

APPENDIX A

Designing a compact scheme with finite differences

A.1 Compactness

Traditional methods of increasing the accuracy of a given scheme often involve extending the computational stencil in space. While the larger spatial stencil provides the degrees of freedom needed to achieve desired properties, it can make the resulting method difficult to parallelize because of increased communication between processes. Additionally, larger spatial stencils tend to sample the flow at points far from the region being updated where the physics may not be relevant. The compactness of a scheme can be preserved by extending the stencil in time. The temporally extended stencil reduces the parallel communication required when neighboring cells are on different processors and uses information from physically meaningful regions of the flow. Figure A.1 illustrates the stencil created by including one additional time-level in the first-order upwind scheme. This new stencil has four degrees of freedom, which allows for a maximum of third-order accuracy. A family of schemes may be defined by applying different weights to each of the points in the stencil.

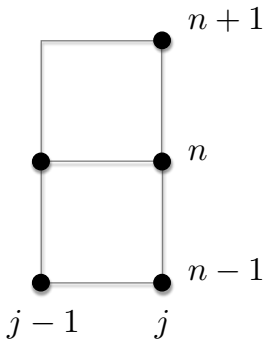


Figure A.1: Three-level upwind stencil, $a > 0$

A.1.1 Family definition

All of the schemes possible on this stencil may be written by leaving the coefficients on each point unknown. Consistency and accuracy arguments can then be used to define the coefficients and their corresponding schemes. Let c be the coefficients applied at the n time level and d be the coefficients at the $n-1$ level.

$$u_j^{n+1} = c_j u_j^n + c_{j-1} u_{j-1}^n + d_j u_j^{n-1} + d_{j-1} u_{j-1}^{n-1} \tag{A.1}$$

Substituting in the Taylor series expansions for each point and requiring consistency with the linear advection equation places the following constraints on the coefficients corresponding to the points at the n time level:

$$\begin{aligned} c_j &= (1 - d_j) - \nu\,(1 + d_j + d_{j-1}) \\ c_{j-1} &= \nu\,(1 + d_j + d_{j-1}) - d_{j-1} \end{aligned} \tag{A.2}$$

Inserting the result into Eq. (A.1) and rearranging leads to a three time level scheme with two parameters d_j and d_{j-1}.

$$\begin{aligned} u_j^{n+1} = u_j^n &- \nu(u_j^n - u_{j-1}^n) \\ &- d_j[(u_j^n - u_j^{n-1}) + \nu(u_j^n - u_{j-1}^n)] \\ &- d_{j-1}[(u_{j-1}^n - u_{j-1}^{n-1}) + \nu(u_j^n - u_{j-1}^n)] \end{aligned} \tag{A.3}$$

Equation (A.3) suggests that the family of schemes sharing this stencil are a linear combination of three residuals. A more natural choice for the residuals weighted by d_j and d_{j-1} are the upwind residuals between time levels n and $n-1$. The stencil with the associated weights α, β, and γ is illustrated in Fig. A.2, where $\alpha = 1$.

$$\begin{aligned} u_j^{n+1} = u_j^n &- \nu(u_j^n - u_{j-1}^n) \\ &- \beta[(u_{j-1}^n - u_{j-1}^{n-1}) + \nu(u_j^n - u_{j-1}^n)] \\ &- \gamma[(u_j^n - u_j^{n-1}) + \nu(u_j^{n-1} - u_{j-1}^{n-1})] \end{aligned} \tag{A.4}$$

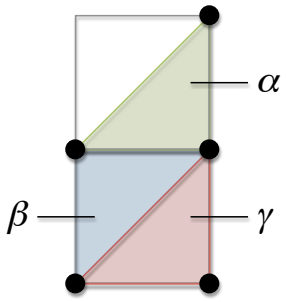


Figure A.2: Residuals contained within three-level definition

A.1.2 Positivity region

A series of first-order schemes may be derived by integrating over control volumes defined by the convex-hull surrounding u_j^{n+1} and any other two points in the stencil.

Positivity constraints on the coefficients of each point in Eq. (A.5) may also be used to define monotone, first-order schemes.

$$\begin{aligned} u_j^{n+1} = &\left[\nu - (1-\nu)\,\beta\right] u_{j-1}^n + \left[(1-\gamma) - \nu\,(1+\beta)\right] u_j^n \\ &+ (\beta + \nu\gamma)\, u_{j-1}^{n-1} + \gamma\,(1-\nu)\, u_j^{n-1} \end{aligned} \tag{A.5}$$

Solving for β and γ leads to the inequalities below.

$$\begin{aligned} 0 \leq \gamma &\leq 1 - \nu\,(\beta + 1) \\ -\nu\gamma \leq \beta &\leq \frac{\nu}{1-\nu} \end{aligned} \tag{A.6}$$

Note that setting $\gamma = 0$ leads to schemes that most closely surround characteristics originating from u_{j-1}^{n-1}. These schemes include a first-order monotone scheme with low truncation error and an optimally small stencil (OPTM) as well as the second-order upwind leapfrog (ULF) scheme [50]. Solving the system defined by Eq. (A.6) for the maximum value of β and γ leads to the three time-level version of the first-order upwind (FUP) scheme. Minimizing β and maximizing γ results in a first-order scheme with few desirable fewatures, which we've termed Scheme 1-2 because it was the second identified first-order scheme. Setting $\beta = \gamma = 0$ recovers the classical two-level FUP scheme. The two and three-level FUP schemes, OPTM scheme, and Scheme 1-2 define the positivity region for the family.

A.1.3 Truncation error and amplification factor

Performing a modified equation analysis on Eq. (A.4) leads to the values defined in Eqns. (A.7-A.8) for the leading diffusion and dispersion coefficients, respectively. A first-order low-phase-error (LPE) scheme may be defined by setting $\gamma = 0$ and

solving for the β that eliminates the leading dispersion term.

$$\frac{C_{\text{diffusion}}}{a\Delta x} = \frac{(1-\nu)(1-\beta+\gamma)}{2(1+\beta+\gamma)} \tag{A.7}$$

$$\begin{aligned}\frac{C_{\text{dispersion}}}{a\Delta x^2} = \frac{1}{6(1+\beta+\gamma)^2}\{(1-\nu)[4\beta(1+\gamma) - 2\nu\beta(4+\gamma) \\ - \beta^2(1-2\nu) - (1+\gamma)^2(1-2\nu)]\}\end{aligned} \tag{A.8}$$

The amplification factor g for the family of three-level schemes is given by Eq. (A.9). There are two solutions for the amplification factor because the scheme contains three time levels.

$$\begin{aligned}g = \frac{1}{2}\mathrm{e}^{-i\theta}\left\{\nu - \beta(1-\nu) - \mathrm{e}^{i\theta}[(\gamma-1) + \nu(1+\beta)]\right\} \\ \pm \frac{1}{2}\mathrm{e}^{-i\theta}\left\{4\mathrm{e}^{i\theta}\left[\beta + \gamma(\mathrm{e}^{i\theta}(1-\nu)+\nu)\right]\right. \\ \left. + \left\{\beta - \nu(1-\beta) + \mathrm{e}^{i\theta}[(\gamma-1) + \nu(\beta+1)]\right\}^2\right\}^{1/2}\end{aligned} \tag{A.9}$$

The diffusion and dispersion errors may be defined as:

$$\epsilon_D = |g| \tag{A.10}$$

$$\epsilon_\theta = \frac{1}{\nu\theta}\tan^{-1}\left(\frac{-\text{Im } g}{\text{Re } g}\right) \tag{A.11}$$

A.1.4 Family album

Table A.1 displays the coefficient values that define the various first-order schemes. Second-order schemes may be derived in the same manner as the first-order schemes. The most notable scheme in the second-order family is the upwind leapfrog scheme which has zero dissipation due to its stencil symmetry in space and time. Using the two parameters to eliminate both the leading diffusion and dispersion terms leads to the third-order chair scheme. Table A.2 gives the parameters defining these schemes.

Table A.1: First-order scheme definitions

Scheme	$\boldsymbol{\beta}$	$\boldsymbol{\gamma}$
FUP	0	0
OPTM, $\nu < \frac{1}{2}$	$\frac{\nu}{1-\nu}$	0
OPTM, $\nu > \frac{1}{2}$	$\frac{1-\nu}{\nu}$	0
LPE	$2-\sqrt{3}$	0
1-2	$\frac{-\nu}{1+\nu}$	$\frac{1}{1+\nu}$
3-Level FUP	$\frac{\nu}{1-\nu}$	$\frac{1-2\nu}{1-\nu}$

Table A.2: Notable second and third-order scheme definitions

Scheme	$\boldsymbol{\beta}$	$\boldsymbol{\gamma}$
ULF	1	0
Chair	$\frac{3\nu}{1+\nu}$	$\frac{2\nu-1}{1+\nu}$

A.1.5 Cousins: the q-schemes

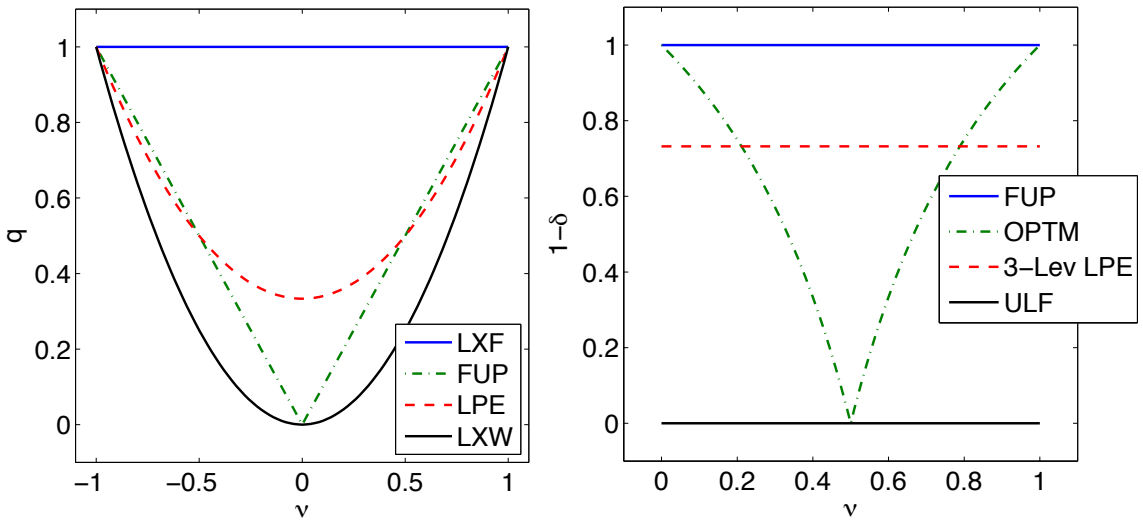
The three-level schemes with $\gamma = 0$ closely parallel the familiar q-schemes that are based on the stencil of points u^n_{j+1}, u^n_j, and u^n_{j-1}. The q-schemes are defined by Eq. (A.12).

$$u^{n+1}_j = u^n_j - \frac{\nu}{2}(u^n_{j+1} - u^n_{j-1}) + \frac{q}{2}(u^n_{j+1} - 2u^n_j + u^n_{j-1}) \tag{A.12}$$

Both the q-schemes and three-level families contain a scheme that minimizes the truncation error, a scheme that maximizes the truncation error, a low-phase-error scheme, and a uniquely defined second-order scheme. Table A.3 lists the q values that define these schemes and their corresponding three-level analogue. Figure A.3 visually compares the scheme definitions. Table A.3 shows that the FUP scheme,

Table A.3: q-Scheme comparison

	$\boldsymbol{q}$	$\boldsymbol{q}$**-Scheme**	**3-Level Scheme**
Min Error	$\lvert\nu\rvert$	FUP	OPTM
Max Error	1	Lax-Friedrichs (LXF)	FUP
Min Dispersion	$\frac{1+2\nu^2}{3}$	Low-Phase-Error (LPE)	3-Level LPE
Second-Order	ν^2	Lax-Wendroff (LXW)	ULF

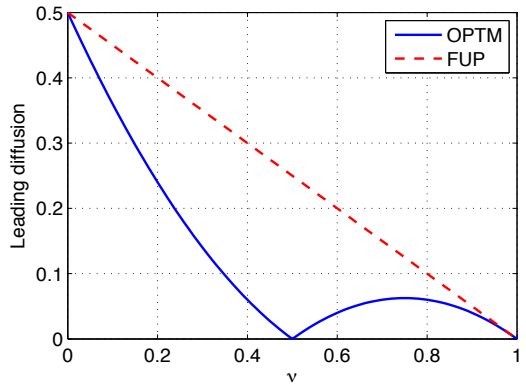


(a) Two-level definitions

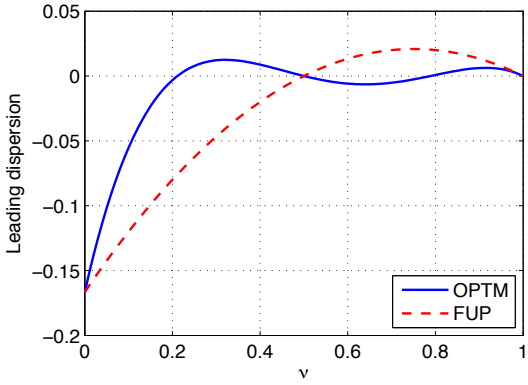
(b) Three-level definitions

Figure A.3: Family parameterizations

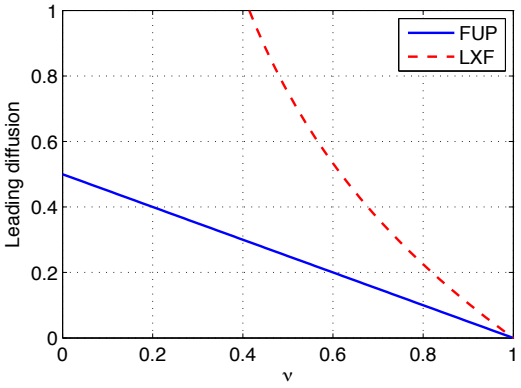
which has the lowest error of the two-level schemes, actually has the greatest error of the three-level schemes. In general, the three-level schemes have less error than their q-scheme counterparts. Figure A.4 compares the leading diffusion and dispersion errors of each scheme. Note that, for these definitions of the diffusion and dispersion coefficients, values of zero indicate no error.



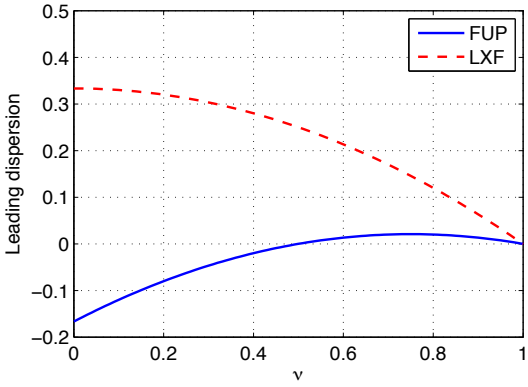
(a) Min error diffusion



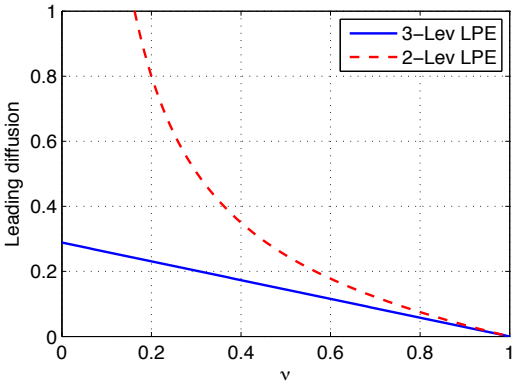
(b) Min error dispersion



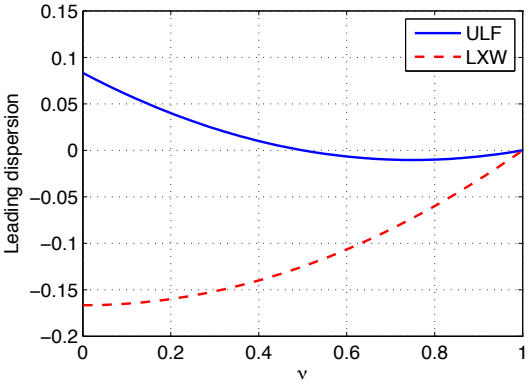
(c) Max error diffusion



(d) Max error dispersion



(e) Low phase error diffusion



(f) Second-order dispersion

Figure A.4: Leading diffusion and dispersion coefficient values

APPENDIX B

Shape-preserving limiter values

The shape-preserving limiter from §3.2 is based on calculating the intersection of the line representing the constant gradient/curvature ratio with various features that can negatively impact the solution. Recall that the line is parameterized by τ where $\tau = 0$ represents the original data and $\tau = 1$ represents a constant reconstruction equal to the cell average value (first-order solution). In (u_L, u_R) space, the shape-preserving line is decribed Eq. (B.1).

$$u_R^{\text{s.p.}} = -u_L \left(\frac{\delta_R}{\delta_L} \right) + u_M \left(1 + \frac{\delta_R}{\delta_L} \right) \tag{B.1}$$

where,

$$\begin{aligned} \delta_L &= \bar{u}_j^n - u_{j-1/2}^n = u_M - u_{L0} \\ \delta_R &= u_{j+1/2}^n - \bar{u}_j^n = u_{R0} - u_M \end{aligned} \tag{B.2}$$

B.1 Shock formation at Δt

A shock forms when two solution characteristics intersect. The shock formation time is given by the following expression [3]:

$$t_F = -\Delta x \left[\min\left(\frac{du}{d\xi}\right)\right]^{-1} \tag{B.3}$$

To prevent a shock from forming within a given time step, we set $t_F = \Delta t$. Equation (B.3) reveals that a necessary condition for a shock to form inside of a cell is that the minimum slope must be negative. The slope for our quadratic reconstruction is linear, meaning the slope will attain its minimum value at either the left or right interface.

$$\begin{aligned} \frac{du}{d\xi} &= 2\alpha\xi + \beta \\ \alpha &= 2\left(c_1 - 2c_2 + c_3\right) \\ \beta &= -\left(3c_1 - 4c_2 + c_3\right) \end{aligned} \tag{B.4}$$

The parameter α is proportional to the curvature. When $\alpha > 0$, the minimum negative slope will occur at the left edge and when $\alpha < 0$, occurs at the right edge. This leads to two possibilities for the shock formation line.

$$u_R^{\text{s.f.}} = \begin{cases} -2u_L + 3u_M + \frac{\Delta x}{2\Delta t} & \text{if } \alpha \geq 0 \\ -\frac{1}{2}\left(u_L - 3u_M + \frac{\Delta x}{2\Delta t}\right) & \text{if } \alpha < 0 \end{cases} \tag{B.5}$$

Finally, the u_L and u_R values corresponding to the intersection between the shape-preserving line and shock formation line can be found by combining Eq. (B.1) and Eq. (B.5).

$$\tau_{\text{s.f.}} = \begin{cases} 1 - \frac{\Delta x}{2\Delta t(\delta_R - 2\delta_L)} & \text{if } \alpha \geq 0 \\ 1 + \frac{\Delta x}{2\Delta t(2\delta_R - \delta_L)} & \text{if } \alpha < 0 \end{cases} \tag{B.6}$$

B.2 Real characteristic origin

The solution for the characteristic origin will be real if the radicand in Eq. (3.5) is positive. Setting the term to zero and solving for u_R leads to the ellipse shown in the limiting diagram. We can then set the result equal to Eq. (B.1) to find the u_L values at the intersection point.

$$u_L = \frac{2ru_M\delta_R^2 + \delta_L\left[\delta_L\left(\delta_L - \delta_R\right)\left(1 + 5ru_M - 3\xi_i\right) \pm \sqrt{\mu}\right]}{2r\left(\delta_L^2 - \delta_L\delta_R + \delta_R^2\right)} \tag{B.7}$$

where,

$$\begin{aligned} r &= \frac{\Delta t}{\Delta x} \\ \mu &= 3\delta_L + 3\left(ru_M - \xi_i\right)\left[\left(3\delta_L - \delta_R\right) + \left(\delta_L - \delta_R\right)\left(1 + 3ru_M - 3\xi_i\right)\right] \end{aligned} \tag{B.8}$$

If the shape-preserving line does not intersect the ellipse defining the complex region, then $\mu < 0$.

B.3 Bounded extremum

The extremum allowed in the cell is defined as

$$u_E = \begin{cases} \min\left(u_{L0}, u_M, u_{R0}\right) & \text{if } a_0 > 0 \\ \max\left(u_{L0}, u_M, u_{R0}\right) & \text{if } a_0 < 0 \\ \text{maxmod}\left(u_{L0}, u_{R0}\right) & \text{if } a_0 = 0 \end{cases} \tag{B.9}$$

The intersection of the ellipse with the shape-preserving line is:

$$\begin{aligned} u_L &= u_M - \frac{3\delta_L\left(\delta_R - \delta_L\right)\left(u_M - u_E\right)}{\delta_L^2 - \delta_L\delta_R + \delta_R^2} \\ u_R &= u_M + 3\left(u_M - u_E\right)\left(1 - \frac{\delta_L^2}{\delta_L^2 - \delta_L\delta_R + \delta_R^2}\right) \end{aligned} \tag{B.10}$$

B.4 Limited signal

Let σ be the maximum or minimum vertex signal that can be applied before the edge value is a new extremum. Again, define an intermediate variable, γ:

$$\gamma = \xi_i + r\left[u_M + \sigma + \delta_L\left(2 - 3\xi_i\right)\xi_i + \delta_R\left(1 - \xi_i\right)\left(1 - 3\xi_i\right)\right] \tag{B.11}$$

Here, the interface ξ_i affected by the signal is determined by estimating the average wave speed.

$$\xi_i = \begin{cases} 1 & \text{when } \frac{1}{2}\left(u_{L0} + u_{R0}\right) > 0 \\ \\ 0 & \text{when } \frac{1}{2}\left(u_{L0} + u_{R0}\right) < 0 \end{cases} \tag{B.12}$$

The point on the shape-preserving line with this specified vertex signal is:

$$\begin{aligned} u_L &= \frac{ru_M\delta_R\left(1-\gamma\right)\left(1-3\gamma\right) - \delta_L\left\{\gamma - ru_M\left[1+\gamma\left(2-3\gamma\right)\right] - \xi_i\right\}}{r\left[\delta_L\left(2-3\gamma\right)\gamma + \delta_R\left(1-\gamma\right)\left(1-3\gamma\right)\right]} \\ u_R &= \frac{\gamma r u_M\delta_L\left(2-3\gamma\right) + \left\{\gamma\left[1 - ru_M\left(4-3\gamma\right)\right] - \xi_i\right\}}{r\left[\delta_L\left(2-3\gamma\right)\gamma + \delta_R\left(1-\gamma\right)\left(1-3\gamma\right)\right]} \end{aligned} \tag{B.13}$$

Currently, the allowable signal is calculated from the maximum and minimum signals

$$\begin{aligned} u_{\max} &= \max\left(u_{L0}, u_M, u_{R0}\right) & u_{\min} &= \min\left(u_{L0}, u_M, u_{R0}\right) \\ \sigma_{\max} &= u_{\max} - u_0(\xi_i) & \sigma_{\min} &= u_{\min} - u_0(\xi_i) \end{aligned} \tag{B.14}$$

resulting in a final value for the allowable signal value:

$$\sigma = \max\left[\sigma_{\min}, \min\left(\sigma_{\max}, \sigma_0\right)\right] \tag{B.15}$$

Because the magnitude of the signal monotonically decreases along the shape-preserving line, these calculations can be avoided by checking if σ_0, which is based on the natural reconstruction, is within the acceptable bounds.

APPENDIX C

Calculating integrals for the 2D wave equation

We must compute integrals of the basis functions and derivatives in order to use spherical means to update edge and vertex values. The integrals have the form shown in Eq. (C.1).

$$M_R\{u\} = \frac{1}{2\pi R}\sum_{i=1}^{7} c_i \left(\int_{\theta_P}^{\theta_Q} \int_{0}^{R} \phi_i \frac{r}{\sqrt{R^2 - r^2}} dr d\theta \right) \tag{C.1}$$

The integral appearing in Eq. (C.1) is purely a function of the element geometry. Furthermore, the difference between the starting and ending integration angles with either be π if we are computing an edge update, or the vertex angle θ_i if we are computing a node update. The integrals for the basis functions and derivatives contain six common terms:

$$\begin{aligned}
g_1(k) &= n_{k_y}\cos\theta_P - n_{k_x}\sin\theta_P \\
g_2(k) &= n_{k_y}\cos\theta_Q - n_{k_x}\sin\theta_Q \\
g_3(k) &= g_1(k) - g_2(k) \\
g_4(k) &= n_{k_x}\cos\theta_P + n_{k_y}\sin\theta_P \\
g_5(k) &= n_{k_x}\cos\theta_Q + n_{k_y}\sin\theta_Q \\
g_6(k) &= \mathbf{n}_k \cdot \mathbf{n}_{k+1}
\end{aligned} \tag{C.2}$$

C.1 Simplified common functions

The common functions g_1 through g_6 appearing in the integral expressions may be further simplified using the knowledge that the elements are triangular.

C.1.1 Simplification of g_1

The function has two forms, depending on whether the acoustic disc is centered at a node or edge midpoint. Let subscript i represent the node at which the disc is centered, and subscript j be the edge over which the disc is centered. When the disc is centered on a node, the expression simplifies to:

$$g_1(k) = \begin{cases} \ell_k & \text{if } i = k+1 \\ \ell_k \cos(\theta_k + \theta_{i-1}) & \text{otherwise} \end{cases} \tag{C.3}$$

where ℓ_k is the length of side k. When the integral is centered at an edge midpoint, the function simplifies to:

$$g_1(k) = \begin{cases} \ell_k & \text{if } j = k \\ \ell_k \cos(\theta_k + \theta_j) & \text{otherwise} \end{cases} \tag{C.4}$$

C.1.2 Simplification of g_2

Again the simplified functions change whether the acoustic disc is centered at a node or edge midpoint. When the disc is centered on a node:

$$g_2(k) = \begin{cases} -\ell_k & \text{if } i = k-1 \\ -\ell_k \cos(\theta_k + \theta_{i+1}) & \text{otherwise} \end{cases} \tag{C.5}$$

When the integral is centered at an edge midpoint, the function simplifies to:

$$g_2(k) = \begin{cases} -\ell_k & \text{if } j = k \\ -\ell_k \cos(\theta_k + \theta_j) & \text{otherwise} \end{cases} \tag{C.6}$$

C.1.3 Simplification of g_4

The functions that contain sums of trigonemetric expressions, g_4 and g_5, only show up in the integral expressions for the nodal updates.

$$g_4(k) = \begin{cases} \ell_{i+1} \sin\theta_i & \text{if } i = k-1 \\ -\ell_{i+1} \sin\theta_i & \text{if } i = k \\ 0 & \text{if } i = k+1 \end{cases} \tag{C.7}$$

C.1.4 Simplification of g_5

$$g_5(k) = \begin{cases} 0 & \text{if } i = k-1 \\ -\ell_{i-1} \sin\theta_i & \text{if } i = k \\ \ell_{i-1} \sin\theta_i & \text{if } i = k+1 \end{cases} \tag{C.8}$$

C.1.5 Simplification of g_6

The dot product of two edge normals can be also be expressed as:

$$g_6(k) = -\ell_{k-1}\ell_{k+1} \cos\theta_k \tag{C.9}$$

C.2 Basis function integrals

As stated earlier, the basis functions have three basic forms: one for nodal bases, one for edge bases, and one for the bubble function. The form of the update equations

also fall into separate categories for the node and edge updates. Let the integral of a basis function be denoted by $I_{i/j,k}$, where the first index represents the basis index. Again, the subscript i is used for nodes and j used for edges. The second index represents the update index which is either associated with a node or an edge. With this notation, the spherical mean formula becomes:

$$M_R\{u\} = \frac{1}{2\pi R}\left[\sum_{i=1}^{3}\left(c_{2i-1}I_{i,k}\right) + \sum_{j=1}^{3}\left(c_{2j}I_{j,k}\right) + c_7 I_{7,k}\right] \tag{C.10}$$

C.2.1 Nodal basis index, nodal update

$$I_{i,k} = \begin{cases} \left[1 + \frac{1}{6}\left(\frac{l_i R}{S}\right)^2\right] R\theta_k + \frac{3\pi R^2}{8S} g_3(i) - \frac{R^3}{6S^2}\left[g_2(i)g_5(i) - g_1(i)g_4(i)\right] & \text{if } i = k \\ \\ \frac{1}{6}\left(\frac{l_i R}{S}\right)^2 R\theta_k - \frac{\pi R^2}{8S} g_3(i) - \frac{R^3}{6S^2}\left[g_2(i)g_5(i) - g_1(i)g_4(i)\right] & \text{otherwise} \end{cases} \tag{C.11}$$

C.2.2 Nodal basis index, edge update

$$I_{i,k} = \begin{cases} \frac{\pi R^2}{12S^2}\left[2Rl_i^2 - 3Sg_1(i)\right] & \text{if } i = k \\ \\ \frac{\pi R^2}{12S^2}\left[2Rl_i^2 + 3Sg_1(i)\right] & \text{otherwise} \end{cases} \tag{C.12}$$

C.2.3 Edge basis index, nodal update

$$I_{j,k} = \frac{R^3}{3S^2}\left\{g_6(j)\theta_k + \left[g_4(j+1)g_5(j-1) - g_1(j+1)g_2(j-1)\right]\sin\theta_k\right\} + \zeta(j,k) \tag{C.13}$$

where

$$\zeta(j,k) = \begin{cases} 0 & \text{if } j = k \\ g_3(j-1) & \text{if } j = k-1 \\ g_3(j+1) & \text{if } j = k+1 \end{cases} \tag{C.14}$$

C.2.4 Edge basis index, edge update

$$I_{j,k} = \frac{\pi R^3}{3S^2} g_6(j) + \zeta(j,k) \tag{C.15}$$

where

$$\zeta(j,k) = \begin{cases} \pi R \left[1 + \frac{R}{2S} g_1(k)\right] & \text{if } j = k \\ \\ \frac{\pi R^2}{2S} g_1(k) & \text{otherwise} \end{cases} \tag{C.16}$$

C.2.5 Bubble index, node update

$$\begin{aligned} I_{7,k} = {} & \frac{9R^3}{4S^2} \left\{ g_6(k)\theta_k + \left[g_4(k+1)g_5(k-1) + g_1(k+1)g_2(k-1) \right] \sin\theta_k \right\} \\ & + \frac{27\pi R^4}{128S^3} \Big\{ \left(n_{1_x} n_{2_x} n_{3_x} - n_{1_y} n_{2_y} n_{3_x} - n_{1_y} n_{2_x} n_{3_y} - n_{1_x} n_{2_y} n_{3_y} \right) \left(\sin^3\theta_P - \sin^3\theta_Q \right) \\ & \qquad + \left(n_{1_y} n_{2_x} n_{3_x} + n_{1_x} n_{2_y} n_{3_x} + n_{1_x} n_{2_x} n_{3_y} - n_{1_y} n_{2_y} n_{3_y} \right) \left(\cos^3\theta_P - \cos^3\theta_Q \right) \\ & \qquad + 3 \left[n_{1_y} n_{2_y} n_{3_y} \left(\cos\theta_P - \cos\theta_Q \right) - n_{1_x} n_{2_x} n_{3_x} \left(\sin\theta_P - \sin\theta_Q \right) \right] \Big\} \end{aligned} \tag{C.17}$$

C.2.6 Bubble index, edge update

$$
\begin{aligned}
I_{7,k} = \frac{9\pi R^2 l_k}{8S}\left(\frac{3}{2} - \frac{Rl_k}{S}\right) \\
+ \frac{27\pi R^4}{64S^3}\Big[\left(n_{1_x}n_{2_x}n_{3_x} - n_{1_y}n_{2_y}n_{3_x} - n_{1_y}n_{2_x}n_{3_y} - n_{1_x}n_{2_y}n_{3_y}\right)\sin^3\theta_P \\
+ \left(n_{1_y}n_{2_x}n_{3_x} + n_{1_x}n_{2_y}n_{3_x} + n_{1_x}n_{2_x}n_{3_y} - n_{1_y}n_{2_y}n_{3_y}\right)\cos^3\theta_P \\
+ 3\left(n_{1_y}n_{2_y}n_{3_y}\cos\theta_P - n_{1_x}n_{2_x}n_{3_x}\sin\theta_P\right)\Big]
\end{aligned}
\tag{C.18}
$$

C.3 Basis derivative integrals

The spherical mean formulas for the basis function derivatives have a similar form as the previous section and contain the same common functions listed in Eq. (C.2).

$$
\begin{aligned}
M_R\{\nabla u\} &= \frac{1}{2\pi R}\sum_{i=1}^{7} c_i \left(\int_{\theta_P}^{\theta_Q}\int_{0}^{R} \nabla\phi_i \frac{r}{\sqrt{R^2 - r^2}} dr d\theta\right) \\
&= \frac{1}{2\pi R}\left[\sum_{i=1}^{3}\left(c_{2i-1}\mathbf{J}_{i,k}\right) + \sum_{j=1}^{3}\left(c_{2j}\mathbf{J}_{j,k}\right) + c_7\mathbf{J}_{7,k}\right]
\end{aligned}
\tag{C.19}
$$

C.3.1 Nodal basis index, nodal update

$$
\mathbf{J}_{i,k} = \begin{cases} \frac{R}{4S^2}\left[\pi R g_3(i) + 6S\theta_k\right]\mathbf{n}_i & \text{if } i = k \\ \\ \frac{R}{4S^2}\left[\pi R g_3(i) - 2S\theta_k\right]\mathbf{n}_i & \text{otherwise} \end{cases}
\tag{C.20}
$$

C.3.2 Nodal basis index, edge update

$$\mathbf{J}_{i,k} = \begin{cases} \frac{\pi R}{2S^2}\left(Rg_1(i) - S\right)\mathbf{n}_i & \text{if } i = k \\ \\ \frac{\pi R}{2S^2}\left(Rg_1(i) + S\right)\mathbf{n}_i & \text{otherwise} \end{cases} \tag{C.21}$$

C.3.3 Edge basis index, nodal update

$$\mathbf{J}_{j,k} = \frac{\pi R^2}{4S^2}\left[\mathbf{n}_{j+1}g_3(j-1) + \mathbf{n}_{j-1}g_3(j+1)\right] + \zeta(j,k) \tag{C.22}$$

where,

$$\zeta(j,k) = \begin{cases} 0 & \text{if } j = k \\ \mathbf{n}_{j+1}\frac{2R\theta_k}{S} & \text{if } j = k+1 \\ \mathbf{n}_{j-1}\frac{2R\theta_k}{S} & \text{if } j = k-1 \end{cases} \tag{C.23}$$

C.3.4 Edge basis index, edge update

$$\mathbf{J}_{j,k} = \frac{\pi R^2}{2S^2}\left[\mathbf{n}_{j+1}g_1(j-1) + \mathbf{n}_{j-1}g_1(j+1) + \zeta(j,k)\right] \tag{C.24}$$

where,

$$\zeta(j,k) = \begin{cases} \frac{2S}{R}\left(\mathbf{n}_{j+1} + \mathbf{n}_{j-1}\right) & \text{if } j = k \\ \mathbf{n}_{j+1}\frac{2S}{R} & \text{if } j = k-1 \\ \mathbf{n}_{j-1}\frac{2S}{R} & \text{if } j = k+1 \end{cases} \tag{C.25}$$

C.3.5 Bubble index, node update

$$\begin{aligned}\mathbf{J}_{7,k} = \frac{27\pi R^2}{16S^2}\left[\mathbf{n}_{k-1}g_3(k+1) + \mathbf{n}_{k+1}g_3(k-1)\right] \\ + \frac{9R^3}{8S^2}\left\{\sum_{m=1}^{3}\mathbf{n}_m g_6(m+1)\right. \\ \left. + \sin\theta_k \sum_{m=1}^{3}\mathbf{n}_m\left[g_4(m+1)g_5(m-1) - g_1(m+1)g_2(m-1)\right]\right\}\end{aligned} \tag{C.26}$$

C.3.6 Bubble index, edge update

$$\mathbf{J}_{7,k} = \frac{27\pi R}{8S^2}\left[S - Rg_1(k)\right]\mathbf{n}_k + \frac{9\pi R^3}{8S^3}\sum_{m=1}^{3}\mathbf{n}_m g_6(m+1) \tag{C.27}$$

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Rémi Abgrall and Mohamed Mezine. Construction of second order accurate monotone and stable residual distribution shcmes for unsteady flow problems. *Journal of Computational Physics*, 188:16–55, 2003.

[2] Steven R. Allmaras, John E. Bussoletti, Craig L. Hilmes, Forrester T. Johnson, Robin G. Melvin, Edward N. Tinoco, Vinkat Venkatakrishnan, Laurence B. Wigdon, and David P. Young. Algorithm issues and challenges associated with the development of robust CFD codes. In *Variational Analysis and Aerospace Engineering*, pages 1–19. Springer New York, 2009.

[3] Peter Bakker and Bram van Leer. *AERO 520 Compressible Flow: Lecture Notes*. University of Michigan, 2008.

[4] Daniel Bouche, G. Bonnaud, and D. Ramos. Comparison of numerical schemes for solving the advection equation. *Applied Mathematics Letters*, 16:147–154, 2003.

[5] D. S. Butler. The numerical solution of hyperbolic systems of partial differential equations in three independent variables. In *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, volume 255, pages 232–252, 1960.

[6] Bernardo Cockburn, San-Yih Lin, and Chi-Wang Shu. TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws III: one-dimensional systems. *Journal of Computational Physics*, 84:90–113, 1989.

[7] Phillip Colella and Paul R. Woodward. The piecewise parabolic method (PPM) for gas-dynamical simulations. *Journal of Computational Physics*, 54:174–201, 1984.

[8] Richard Courant and David Hilbert. *Methods of Mathematical Physics*, volume 2. Wiley-VCH, 1962.

[9] S.F. Davis. A rotationally biased upwind difference scheme for the Euler equations. *Journal of Computational Physics*, 56:65–92, 1984.

[10] John P. Dean, Scott A. Morton, David R. McDaniel, James D. Clifton, and David J. Bodkin. Aircraft stability and control characteristics determined by

system identification of CFD simulations. In *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, 2008. AIAA Paper 2008-6378.

[11] B. Einfeldt. On Godunov-type methods for gas dynamics. *SIAM Journal on Numerical Analysis*, 25(2):294–318, 1988.

[12] Krzysztof J. Fidkowski and Y. Luo. Output-based space-time mesh adaptation for the compressible Navier-Stokes equations. *Journal of Computational Physics*, 230:5753–5773, 2011.

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

[14] Oliver Friedrich. Weighted essentially non-oscillatory schemes for the interpolation of mean values on unstructured grids. *Journal of Computational Physics*, 144:194–212, 1998.

[15] Sergei K. Godunov. A difference scheme for numerical computation of discontinuous solution of hydrodynamic equations. *Matematicheskii Sbornik*, 47:271–306, 1959.

[16] Ami Harten, Bjorn Engquist, Stanley Osher, and Sukumar R Charkravarthy. Uniformly high order accurate essentially non-oscillatory schemes, III. *Journal of Computational Physics*, 71:231–303, 1987.

[17] Amiram Harten, Peter D. Lax, and Bram van Leer. On upstream differencing and Godunov-type schemes for hyperbolic conservation laws. *SIAM Review*, 25(1):35–60, 1983.

[18] Charles Hirsch. *Numerical Computation of Internal and External Flows: Fundamentals of Fluid Dynamics*. Butterworth-Heinemann, 2nd edition, 2007.

[19] Arieh Iserles. Generalized leapfrog methods. *IMA Journal of Numerical Analysis*, 6:381–392, 1986.

[20] Rudy A. Johnson, Michael J. Stanek, and James E. Grove. Store separation trajectory deviations due to unsteady weapons bay aerodynamics. In *46th AIAA Aerospace Sciences Meeting*, 2008. AIAA Paper 2008-0188.

[21] S.A. Karabasov and V.M. Goloviznin. Compact Accurately Boundary-Adjusting high-REsolution Technique for fluid dynamics. *Journal of Computational Physics*, 228:7426–7451, 2009.

[22] Dmitri Kuzmin. A vertex-based hierarchical slope limiter for p-adaptive discontinuous Galerkin methods. *Journal of Computational and Applied Mathematics*, 233:3077–3085, 2010.

[23] Randall J. LeVeque. *Finite Volume Methods for Hyperbolic Problems*. Cambridge University Press, 2nd edition, 2004.

[24] D.W. Levy, Kenneth G. Powell, and Bram van Leer. Use of a rotated Riemann solver for the two-dimensional Euler equations. *Journal of Computational Physics*, 106:201–214, 1993.

[25] Xu-Dong Liu, Stanley Osher, and Tony Chan. Weighted essentially non-oscillatory schemes. *Journal of Compuatational Physics*, 115:200–212, 1994.

[26] Rainald Löhner. Error and work estimates for high order elements. In *49th AIAA Aerospace Sciences Meeting*, 2011. AIAA Paper 2011-0211.

[27] Maria Lukáčová-Medvid'ová, K.W. Morton, and Gerald Warnecke. Evolution Galerkin methods for hyperbolic systems in two space dimensions. *Mathematics of Computation*, 69(232):1355–1384, 2000.

[28] K.W. Morton and Philip L. Roe. Vorticity-preserving Lax-Wendroff-type schemes for the system wave equation. *SIAM Journal on Scientific Computing*, 23(1):170–192, 2001.

[29] Scott A. Morton, Russell M. Cummings, and Denis B. Kholodar. High resolution turbulence treatment of F/A-18 tail buffet. *Journal of Aircraft*, 44(6):1769–1775, 2007.

[30] Hiroaki Nishikawa. A first-order system approach for diffusion equation. II: unification of advection and diffusion. *Journal of Computational Physics*, 229:3989–4016, 2010.

[31] Hiroaki Nishikawa and Keiichi Kitamura. Very simple, carbuncle-free, boundary-layer-resolving, rotated-hybrid Riemann solvers. *Journal of Computational Physics*, 227:2560–2581, 2008.

[32] Hiroaki Nishikawa, Mani Rad, and Philip Roe. A third-order fluctuation splitting scheme that preserves potential flow. In *15th AIAA Computational Fluid Dynamics Conference*, 2001. AIAA Paper 2001-2595.

[33] Carl F. Ollivier-Gooch. Quasi-ENO schemes for unstructured meshes based on unlimited data-dependent least-squares reconstruction. *Journal of Computational Physics*, 133:6–17, 1997.

[34] S. Ostkamp. Multidimensional characteristic Galerkin mehods for hyperbolic systems. *Mathematical Methods in the Applied Sciences*, 20:1111–1125, 1997.

[35] S.H. Park and J. Kwon. An improved HLLE method for hypersonic viscous flows. In *19th AIAA Computational Fluid Dynamics Conference*, 2001. AIAA Paper 2001-2633.

[36] K.M. Peery and S.T. Imlay. Blunt-body flow simulations. In *AIAA/SAE/ASME/ASEE 24th Joint Propulsion Conference*, 1988. AIAA Paper 88-2904.

[37] Jaime Peraire, N.C. Nguyen, and Bernardo Cockburn. A hybridizable discontinuous Galerkin method for the compressible Euler and Navier-Stokes equations. In *48th AIAA Aerospace Sciences Meeting*, 2010. AIAA Paper 2010-0363.

[38] Jaime Peraire, N.C. Nguyen, and Bernardo Cockburn. An embedded discontinuous Galerkin method for the compressible Euler and Navier-Stokes equations. In *20th AIAA Computational Fluid Dynamics Conference*, 2011. AIAA Paper 2011-3228.

[39] James J. Quirk. A contribution to the great Riemann solver debate. *International Journal for Numerical Methods in Fluids*, 18:555–574, 1994.

[40] Mani Rad. *A Residual Distribution Approach to the Euler Equations that Preserves Potential Flow*. PhD thesis, University of Michigan, 2001.

[41] A. Sivasankara Reddy, V.G. Tikekar, and Phoolan Prasad. Numerical solution of hyperbolic equations by method of bicharacteristics. *Journal of Mathematical and Physical Sciences*, 16(6):575–603, 1982.

[42] Y.X. Ren. A robust shock-capturing scheme based on rotated Riemann solvers. *Computers and Fluids*, 32:1379–1403, 2003.

[43] Mario Ricchiuto. *Construction and analysis of compact residual discretizations for conservation laws on unstructred meshes*. PhD thesis, Université Libre de Bruxelles and von Karman Institute for Fluid Dynamics, 2005.

[44] Philip L. Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics*, 43:357–372, 1981.

[45] Philip L. Roe. Fluctuations and signals, a framework for numerical advection problems. In *Numerical Methods for Fluid Dynamics*, pages 219–257, 1982.

[46] Philip L. Roe. "Optimum" upwind advection on a triangular mesh. Technical report, ICASE, 1990. Technical Report 90-75.

[47] Philip L. Roe. Linear bicharacteristic schemes without dissipation. *SIAM Journal of Scientific Computing*, 19(5):1405–1427, 1998.

[48] V.V. Rusanov. The calculation of the interaction of non-stationary shock waves and obstacles. *USSR Computational Mathematics and Mathematical Physics*, 1(2):304–320, 1962.

[49] Michael J. Shea, Matthew M. Constantino, Christopher W. O'Brien, Murray R. Snyder, Scott A Simpson, and Alexis Cenko. Litening pod modification to improve MK-83 store trajectories. In *29th AIAA Applied Aerodynamics Conference*, 2011. AIAA Paper 2011-3158.

[50] Jeffrey Thomas. *An Investigation of the Upwind Leapfrog Method for Scalar Advection and Acoustic/Aeroacoustic Wave Propagation Problems*. PhD thesis, University of Michigan, 1996.

[51] Eleuterio F. Toro. *Riemann Solvers and Numerical Methods for Fluid Dynamics*. Springer, 3rd edition, 2009.

[52] Robert W. Tramel and Robert H. Nichols. Addition of improved shock-capturing schemes to OVERFLOW 2.1. In *19th AIAA Computational Fluid Dynamics Conference*, 2009. AIAA Paper 2009-3988.

[53] Bram van Leer. Towards the ultimate conservative difference scheme. IV. A new approach to numerical convection. *Journal of Computational Physics*, 23(3):276–299, 1977.

[54] Prasanna A. Varadarajan. *Noise Transmission Along Shock-Waves*. PhD thesis, University of Michigan, 2011.

[55] John C. Vassberg, Edward N. Tinoco, Mori Mani, Ben Rider, Tom Zickuhr, Devid W. Levy, Olaf P. Brodersen, Bernhard Eisfeld, Simone Crippa, Richard A. Wahls, Joseph H. Morrison, Dimitri J. Mavriplis, and Mitsuhiro Murayama. Summary of the fourth AIAA CFD drag prediction workshop. In *28th AIAA Applied Aerodynamics Conference*, 2010. AIAA Paper 2010-4547.

[56] Z. J. Wang. Spectral (finite) volume method for conservation laws on unstructured grids: Basic formulation. *Journal of Computational Physics*, 178:210–251, 2002.

[57] Z. J. Wang and Yen Liu. The spectral difference method for the 2D Euler equations on unstructured grids. In *17th AIAA Computational Fluid Dynamics Conference*, 2005. AIAA Paper 2005-5112.

[58] R.F. Warming, Paul Kutler, and Harvard Lomax. Second- and third-order noncentered differnce schemes for nonlinear hyperbolic equations. *AIAA Journal*, 11(2):189–196, 1973.