Multidimensional Active Flux Schemes

Timothy A. Eymann*

DoD HPCMP/CREATE Kestrel Team, Eglin AFB, FL 32542

Philip L. Roe†

Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI, 48109

We build on active flux (AF) schemes and extend the method to the two-dimensional linear advection, linear acoustics, and linearized Euler equations. The active flux method independently updates edge and centroid values. Because the interface fluxes are calculated from independently updated quantities, we say that they are actively updated, as opposed to a more traditional finite-volume scheme where the fluxes are updated passively from the conserved values. The one-dimensional active flux method is reformulated using Lagrange basis functions and the basic features of the method are reviewed. The two-dimensional formulation also uses standard basis functions, but includes a bubble function to maintain conservation. A novel approach for solving the linear wave system is presented that uses spherical means to compute the edge updates for the flux calculation. We demonstrate that the AF method is third-order accurate for advection, acoustics, and the linearized Euler equations.

I. Introduction

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,[1,2] or the complex interaction between the vehicle flow field and a store.[3,4] In civilian aerospace applications, a major performance metric is operating cost. One of the primary factors affecting the operating cost of an aircraft is its drag, therefore it is vital that companies have the ability to accurately predict outputs such as drag during the design process. This is the main motivation behind the AIAA drag prediction workshops that have focused on developing and communicating CFD best-practices.[5] 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.

Contemporary CFD simulations routinely include viscous effects, time-dependent physics and three-dimensional phenomena, including turbulence. Consequently, the computational resources required to perform a CFD simulation often push the jobs away from desktop machines toward specialized clusters. To achieve our modern simulation capability from the early CFD simulations of two-dimensional airfoils, 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, such as the inviscid flux calculation or turbulence model. The parallelization of CFD solvers further solidified the basic form of the code because major algorithm changes would compel the solver developer to re-tune the application, requiring 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 code with a one-dimensional Riemann solver to calculate

*Research Engineer, Member AIAA

†Professor, Fellow AIAA

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 paper expands on a new CFD framework, the active flux (AF) method. We present the AF method as a means of resolving the conflict that occurs when solving increasingly complex problems using techniques that were optimized under a code structure designed for simpler cases.

A. Shortcomings of contemporary solvers

The active flux method is designed to address three major issues plaguing contemporary, production-level CFD solvers:

1. Lack of true multidimensional physics
2. Second-order accuracy
3. Non-compact stencil

1. Multidimensional physics

Godunov[6] 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,[7] while others such as Rusanov[8] and later Harten, Lax, and van Leer[9] further approximated the system by reducing the number of waves considered. 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 typical technique is to solve a one-dimensional Riemann 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.[10,11] Another common manifestation of the problem is the oscillation of solution values behind an oblique shock that is not aligned with the mesh.[12] The independent interface updates in the AF method allow us to avoid solving a Riemann problem and instead facilitate the introduction of truly multidimensional physics, which in turn reduces the influence of the underlying mesh geometry on the flow solution.

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: minimizing sources of error within the method and increasing the order of accuracy of the scheme.

Increasing the order of accuracy of a scheme is the more common 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. Schemes such as the upwind leapfrog (ULF) method[13,14,15] use an additional time level in the stencil to create dissipation free methods. Using an additional time level in the scheme provides additional degrees of freedom (DOF) that can also be used to design optimum monotone schemes or minimize dispersion.[16]

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[17] collapses the two-level ULF to a single-step scheme by independently updating cell-averages and flux values. Independently updating the conserved cell-average values and flux values is a key concept that we utilize throughout this work. Karabasov's scheme retains all the advantages of the ULF scheme while only using data from one time level. In fact, the CABARET scheme can be thought of as a second-order version of the third-order active flux technique.

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. 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. The different leading error terms have the effect of exposing more oscillatory behavior for even-ordered schemes than odd-ordered schemes.[18] 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.[19] For this reason, we focus on third-order accuracy.

3. Compact stencil

The traditional method of 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.[20] Later, essentially non-oscillatory (ENO)[21] and weighted essentially non-oscillatory (WENO)[22] schemes were developed to obtain higher-order accuracy without generating spurious oscillations in the solution. This approach is most straightforward to apply on a structured mesh, but it is also applicable to unstructured meshes.[23,24] 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.

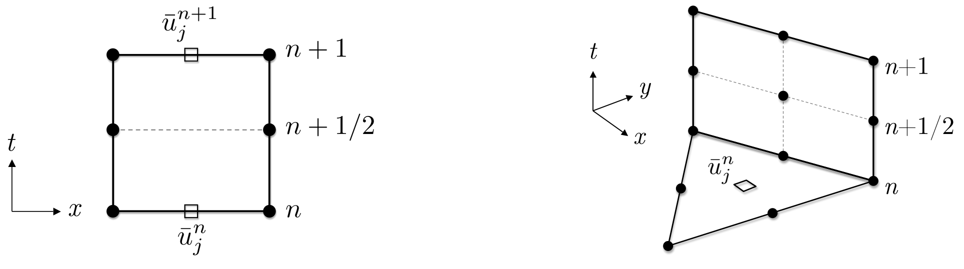
The combined need for higher accuracy and increased parallelization has driven massive interest in methods that achieve higher-order accuracy by increasing the internal degrees of freedom within a cell. In the late 1980s, researchers began to employ finite-element methods for solving fluids problems. Over time, these evolved into schemes such as the discontinuous-Galerkin (DG),[25] spectral difference (SD),[26] spectral volume (SV) method,[27] and streamline upwind Petrov-Galerkin (SU/PG) method.[28] 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,[29] which is closely related to the absence of sufficient limiters for nonlinear problems.[30] The second, and more fundamental issue, is their high memory requirement and computational expense. A recent variant of the DG method, the hybridizable discontinuous-Galerkin[31] scheme, reduces memory overhead by sharing DOF between cells, but the formulation still leaves independent DOF at the mesh vertices, precisely where there is most to gain from sharing between elements. Recognizing this deficiency, embedded discontinuous-Galerkin methods[32] 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.

B. Active flux schemes

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.[33] 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.[34] 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.

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 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



(a) One dimension (b) Two dimensions

Figure 1. Data storage locations for AF scheme

at face centroids and vertices. Because cells share data between faces and vertices, the scheme is able to achieve 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. 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.

C. Paper overview

The objective of this paper is to reintroduce active flux schemes and expand on the one-dimensional concepts presented previously by the authors.[35] Section II reviews the one-dimensional AF discretization and implements the method for the familiar linear advection equation, while Section III follows up with the two-dimensional extension. The truly multidimensional nature of the method is illustrated in Section IV, which details the AF method applied to the linear acoustics equation. Section V then demonstrates how operator splitting can be used to combine different types of active flux updates to solve more complex equations. Finally, the paper concludes with a summary of the work and suggestions for future avenues of research.

II. Active Flux Method Review

We will illustrate the basic mechanics of the AF method using the familiar linear advection equation. The governing equation for one-dimensional advection with a wave speed a is:

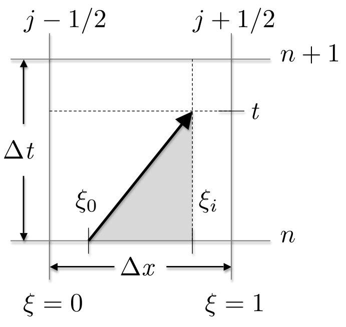
$$\partial_t u + \partial_x (au) = 0 \tag{1}$$

We take the wave speed to be constant in the flow, meaning the conservative flux function is $f(u) = au$. The well-known exact solution can be found from the method of characteristics, defined by Eq. (2).

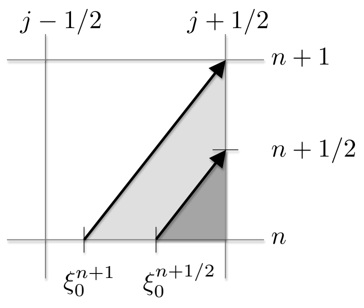
$$u(x,t) = u(x - at, 0) \tag{2}$$

We seek a general representation of the solution within a given cell j, illustrated in Fig. 2, which has a width Δx. Figure 2(a) shows a generalized update at an interface location ξ_i and time t; however, in practice, the interface location will either be at the left or right face, as illustrated in Fig. 2(b) for the case of positive wave speeds. 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:

$$\begin{aligned} x &= x_{j-1/2} + \xi \Delta x \\ \xi &= \left(x - x_{j-1/2}\right) / \Delta x \end{aligned} \tag{3}$$



(a) General update at (ξ_i, t)



(b) Update for right interface $\xi_i = 1$

Figure 2. Numerical domain of dependence for interface updates; $a > 0$

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 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. While the resulting quadratic has several equivalent forms, we choose to follow the standard finite element convention and represent the solution as a summation of basis functions.

$$u(\xi) = \sum_{i=1}^{3} c_i \phi_i(\xi) \tag{4}$$

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 given in Table 1.

For an active flux scheme, we need to know the updated solution value at the cell vertex as well as the average flux passing through the cell edge. A key advantage of the AF method is that these 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. For the 1D advection equation, the characteristic is a straight line traveling at speed $\lambda = a$. The linear nature of the characteristic means the following relationship is valid:

$$\lambda t = (\xi_i - \xi_0) \Delta x \tag{5}$$

Table 1. Basis functions and coefficients for 1D reconstruction

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

Rearranging to solve for the characteristic origin ξ_0:

$$\xi_0 = \xi_i - \frac{\lambda t}{\Delta x} \tag{6}$$

The updated vertex value is simply $u(\xi_0)$. The active flux scheme, as formulated, requires a Courant number $\nu \leq 1$. Physically, this means that only the cells directly surrounding the interface contribute to the face updates. If the characteristic origin ξ_0 is calculated to lie outside of the cell, we know that we have either violated the CFL condition or that the interface value is updated from the other cell sharing the interface.

The average flux may be determined by employing a sufficiently accurate numerical integration technique, such as Simpson's rule. 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 $\Delta t/2$. The case for $a > 0$ is illustrated in Fig. 2(b). Once the two characteristic origins have been determined, the average flux at an interface is

$$\bar{f} = \frac{1}{6}\left[f(u(\xi_i)) + 4f(u(\xi_0^{n+1/2})) + f(u(\xi_0^{n+1}))\right] + \mathcal{O}(\Delta t^4) \tag{7}$$

The cell averages are then updated conservatively using the average flux values.

$$\bar{u}_j^{n+1} = \bar{u}_j^n - \frac{\Delta t}{\Delta x}\left(\bar{f}_{j+1/2} - \bar{f}_{j-1/2}\right) \tag{8}$$

III. Two-dimensional advection

Two-dimensional advection is described by Eq. (9), where the wave speed $\boldsymbol{\lambda} = (a, b)^{\mathrm{T}}$ is again assumed constant.

$$\partial_t u + \partial_x (au) + \partial_y (bu) = 0 \tag{9}$$

The flux function is $\mathbf{f}(u) = (au, bu)^{\mathrm{T}}$. The multidimensional strategy is the same as the one-dimensional technique: choose an appropriate discretization, solve for the characteristic origin, and advance the cell averages using fluxes calculated from independently updated point quantities.

A. Reference element

In 2D, 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:

$$\mathbf{x} = \mathbf{x}_1 + \xi(\mathbf{x}_2 - \mathbf{x}_1) + \eta(\mathbf{x}_3 - \mathbf{x}_1) \tag{10}$$

Or rewriting as matrix:

$$\mathbf{x} = \mathbf{x}_1 + \mathbf{J}\boldsymbol{\xi} \tag{11}$$

where the Jacobian matrix is defined:

$$\mathbf{J} = \begin{pmatrix} x_2 - x_1 & x_3 - x_1 \\ y_2 - y_1 & y_3 - y_1 \end{pmatrix} \tag{12}$$

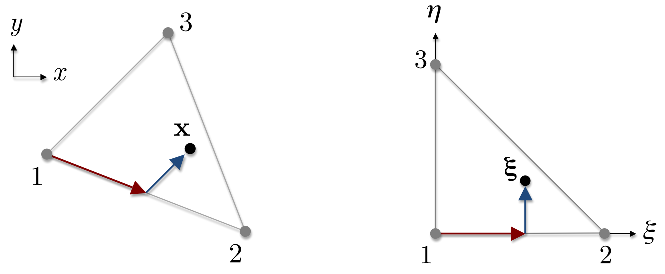


Figure 3. Element mapping

B. 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, shown in Fig. 4 as point 7. Because the index numbering is arbitrary, we choose to give the nodes odd-numbered values and label the edges with even-numbered values. While it is obvious

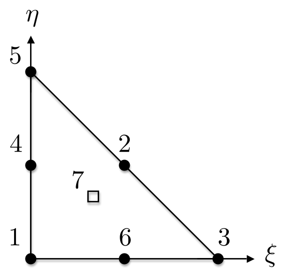


Figure 4. Node placement for AF basis functions

that the coefficients 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 the correct cell average. We start with the definition of the average for element j.

$$\begin{aligned}
\bar{u}_j &= \frac{1}{A} \iint_{\Omega_j} u(x, y)\, d\Omega \\
&= \frac{J}{A} \int_0^1 \int_0^{1-\xi} u(\xi(x, y), \eta(x, y))\, d\eta\, d\xi \\
&= 2 \int_0^1 \int_0^{1-\xi} \left[\left(\sum_{i=1}^{6} u_i \phi_i \right) + c_7 \phi_7 \right] d\eta\, d\xi
\end{aligned} \tag{13}$$

We can rearrange this equation to solve for the unknown bubble coefficient. Like the 1D counterpart, the two-dimensional discretization is entirely local and independent of the shape of the neighboring cells. Table 2 contains the complete set of basis functions and coefficients for the two-dimensional formulation.

C. Updates

Once the reconstruction has been well-defined, we can develop expressions for the nodal updates, which are simply the function value at the characteristic origin. Similar to the 1D case, we exploit the fact that the characteristics are straight lines.

$$t\boldsymbol{\lambda} = \mathbf{J}\left(\boldsymbol{\xi}_i - \boldsymbol{\xi}_0\right) \tag{14}$$

Table 2. Basis functions and coefficients for 2D reconstruction

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

Solving for the origin:

$$\boldsymbol{\xi_0} = \boldsymbol{\xi_i} - t\mathbf{J}^{-1}\boldsymbol{\lambda} \tag{15}$$

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

$$u(\boldsymbol{\xi_i}, t) = u(\boldsymbol{\xi_0}) \tag{16}$$

A 2D version of Simpson's numerical integration rule is used to estimate the average flux through an interface, where Eq. (16) is used to find the required point values.

$$\begin{aligned}\bar{\mathbf{f}} = \frac{1}{9}\Bigg[&\frac{1}{4}\left(\mathbf{f}_L^n + \mathbf{f}_R^n + \mathbf{f}_L^{n+1} + \mathbf{f}_R^{n+1}\right) \\ &+ \left(\mathbf{f}_M^n + \mathbf{f}_L^{n+1/2} + \mathbf{f}_R^{n+1/2} + \mathbf{f}_M^{n+1}\right) + 4\mathbf{f}_M^{n+1/2}\Bigg]\end{aligned} \tag{17}$$

The conserved variable in each cell is updated by integrating the flux around the boundary of the element. This is

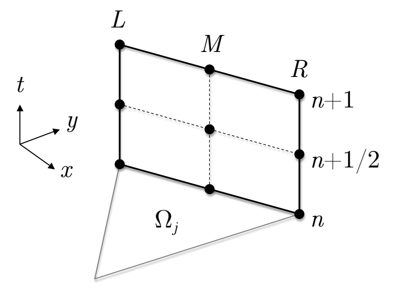


Figure 5. Nomenclature for 2D flux calculation

equivalent to dotting the flux with the face normal and multiplying by the face length for all three faces of the cell.

$$\bar{u}_j^{n+1} = \bar{u}_j^n - \frac{\Delta t}{\Omega_j}\sum_{m=1}^{3}\left(\bar{\mathbf{f}}_m \cdot \mathbf{n}_m\right) l_m \tag{18}$$

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 update the conserved cell-averages within a loop over the mesh faces, changing the sign according to the normal direction. We have eliminated the need to solve a Riemann problem by forcing a continuous representation, which removes jumps in the solution between cells. Additionally, upwinding is achieved because the method inherently analyzes the behavior of the solution within each element.

D. Linear advection results

The simplest example using the 2D 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:

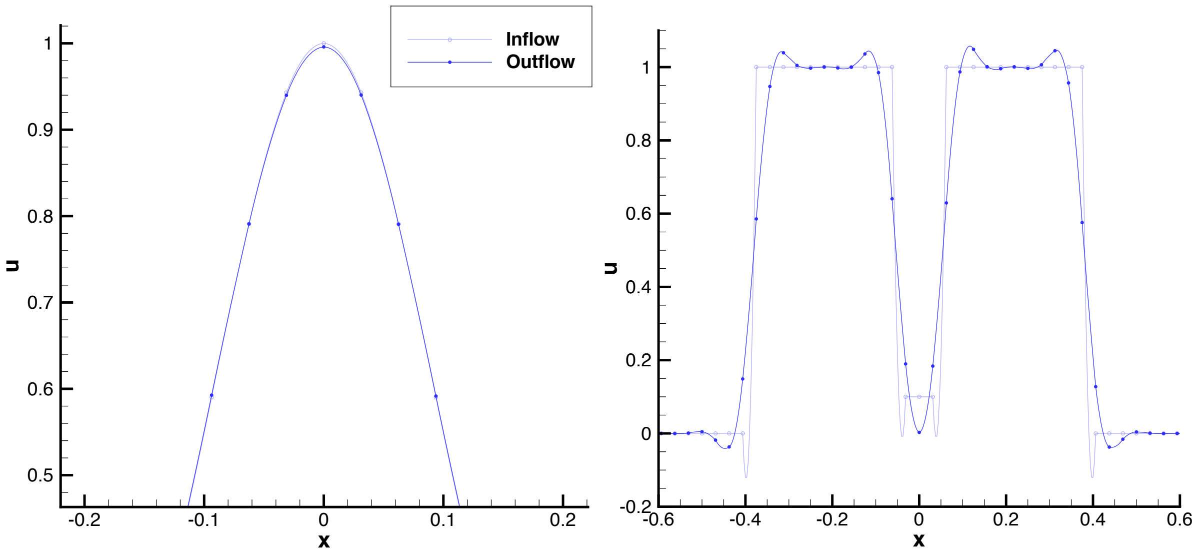
$$\bar{\mathbf{f}} = \boldsymbol{\lambda}\frac{1}{9}\left[\frac{1}{4}\left(u_L^n + u_R^n + u_L^{n+1} + u_R^{n+1}\right) + \left(u_M^n + u_L^{n+1/2} + u_R^{n+1/2} + u_M^{n+1}\right) + 4u_M^{n+1/2}\right] \tag{19}$$

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. (20), drops to machine zero.

$$\|R\|_2 = \sqrt{\frac{\sum_{j=1}^{N}\left[\sum_{m=1}^{3}\left(\bar{\mathbf{f}}_m \cdot \mathbf{n}_m\right)\ell_m\right]^2}{N}} \tag{20}$$

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 2400 randomly oriented elements. Figure 6 shows the steady state solutions for a smooth Gaussian profile and the cross section of a slotted cylinder. Each case converged to machine zero ($\approx 1 \times 10^{-16}$) 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%.



(a) Gaussian

(b) Slotted cylinder

Figure 6. Converged steady advection solutions on randomized mesh

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, meaning that with smooth solutions, the error should decrease by a factor of eight every time the relevant length scale is halved. Because the mesh is unstructured and the Courant number varies from cell to cell, we approximate the average mesh spacing as $1/\sqrt{\mathrm{DOF}}$. The error for any high-order scheme must be computed by integrating over each element. Equation (21) defines the error norm, where 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.

$$\|\epsilon\|_p = \left[\frac{\sum_{j=1}^{N}\iint_{\Omega_j}|u - U_0|^p \, d\Omega_j}{\sum_{j=1}^{N}\Omega_j}\right]^{1/p} \tag{21}$$

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

$$\|\epsilon\|_p = \left\{ \frac{\sum_{j=1}^{N} J_j \sum_{i=1}^{M} w_i \left| u(\boldsymbol{\xi}_i) - U_0(\boldsymbol{\xi}_i) \right|^p}{\sum_{j=1}^{N} \Omega_j} \right\}^{1/p} \tag{22}$$

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 7 compares the error with respect to the initial projection for the AF scheme and a DG1 solution computed with the XFlow[36,37] solver. 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 DG scheme achieves third-order accuracy through superconvergence, meaning that three DOF per cell may not be sufficient to obtain third-order accuracy for all cases. The AF scheme does not rely on superconvergence to realize third-order accuracy.

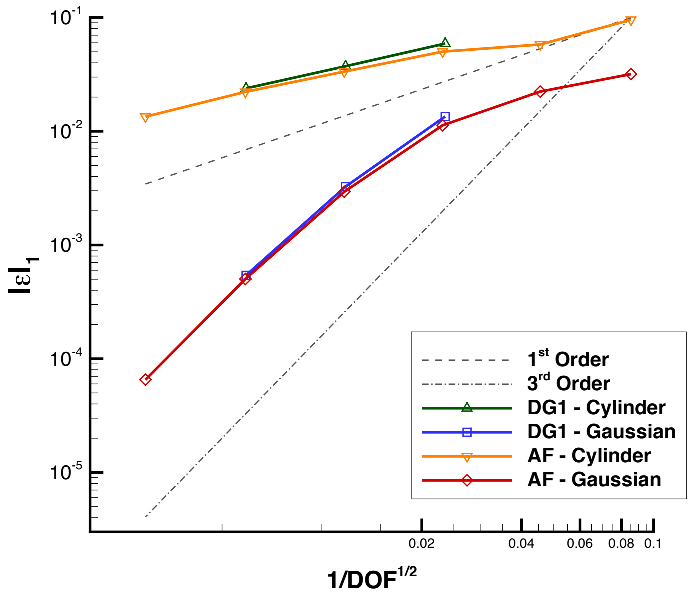


Figure 7. AF error compared to DG1 scheme

Table 3. Linear advection convergence of Gaussian using AF and DG methods

	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	-

IV. Acoustics

When the pressure disturbances in a flow are small, the Euler equations reduce to the linear acoustic equations in Eq. (23). This model system can be used to develop a scheme that properly models the multidimensional wave propagation in a fluid.

$$\begin{aligned} \partial_t p + \rho_0 a_0^2 \left(\nabla \cdot \mathbf{u}\right) &= 0 \\ \partial_t \mathbf{u} + \frac{1}{\rho_0}\nabla p &= 0 \end{aligned} \tag{23}$$

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, two-dimensional system:

$$\begin{aligned} \partial_t p^* + a_0 \left(\partial_x u^* + \partial_y v^*\right) &= 0 \\ \partial_t u^* + a_0 \partial_x p^* &= 0 \\ \partial_t v^* + a_0 \partial_y p^* &= 0 \end{aligned} \tag{24}$$

Combining the time derivative of the pressure equation and spatial derivatives of the velocity equations reveals that the pressure obeys the scalar wave equation.

$$\partial_{tt} p^* - a_0^2 \nabla^2 p^* = 0 \tag{25}$$

Similarly, we can write the velocity terms:

$$\partial_{tt} \mathbf{u}^* - a_0^2 \nabla^2 \mathbf{u}^* = a_0^2 \left(\nabla \times \omega\right) \tag{26}$$

where the vorticity $\omega = \partial_x v^* - \partial_y u^*$.

Equation (26) clearly shows that velocity also obeys the wave equation in flows with constant vorticity. For this subset of problems, the pressure and velocity at any point in time can be calculated using exact solutions to the wave equation. Our strategy for 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.

A. 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.[38] Define the spherical mean of a function f over a

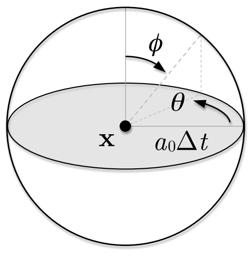


Figure 8. Sphere influenced by point x; $R = a_0\Delta t$

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

$$\begin{aligned} M_R^{3D}\{f\}(x,y,z) = \frac{1}{4\pi R^2}\int_0^{2\pi}\int_0^{\pi} f\,(x + R\sin\phi\cos\theta,& \\ y + R\sin\phi\sin\theta, z + R\cos\phi)&\, R^2 \sin\phi \; d\phi \; d\theta \end{aligned} \tag{27}$$

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. 8 to an integral over the shaded disc.

$$M_R^{2D}\{f\}(x,y) = \frac{1}{2\pi R}\int_0^{2\pi}\int_0^{R} f(x + r\cos\theta, y + r\sin\theta)\frac{r}{\sqrt{R^2 - r^2}} dr \; d\theta \tag{28}$$

By using spherical means, we transform the original PDE into a form with a simple, and known, exact solution. Courant and Hilbert[38] provide the following solution for the wave equation $\partial_{tt}u = a_0^2\nabla^2 u$:

$$u(t) = tM_R\left\{\partial_t u\right\} + \partial_t\left[tM_R\left\{u\right\}\right] \tag{29}$$

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

$$u(t) = tM_R\left\{\partial_t u\right\} + M_R\left\{u\right\} + t\partial_t M_R\left\{u\right\} \tag{30}$$

The time derivative in the last term of Eq. (30) 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 $R = a_0 t$.

$$t\partial_t M_R\left\{u\right\} = t\partial_R M_R\left\{u\right\}\frac{\partial R}{\partial t} = R\partial_R M_R\left\{u\right\} \tag{31}$$

Substituting the expression Eq. (31) into Eq. (30) results in the final expression for $u(t)$:

$$u(t) = tM_R\left\{\partial_t u\right\} + M_R\left\{u\right\} + R\partial_R M_R\left\{u\right\} \tag{32}$$

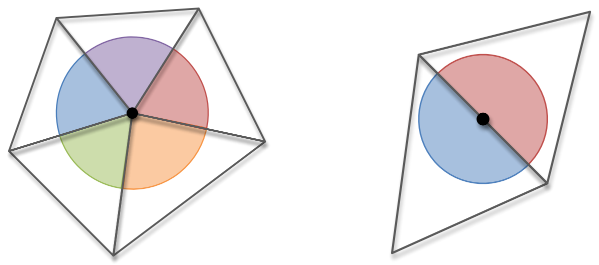
All that remains is to apply the exact solution to the acoustic wave equation. We can use Eq. (24) to replace the time derivatives in Eq. (32) with spatial derivatives. Assuming constant vorticity, we get the following update equations:

$$\begin{aligned} p^*(t) &= M_R\left\{p^*\right\} + R\left[\partial_R M_R\left\{p^*\right\} - M_R\left\{\text{div}\ \mathbf{u}^*\right\}\right] \\ \mathbf{u}^*(t) &= M_R\left\{\mathbf{u}^*\right\} + R\left[\partial_R M_R\left\{\mathbf{u}^*\right\} - M_R\left\{\nabla p^*\right\}\right] \end{aligned} \tag{33}$$

Incorporating vorticity requires only a single new term, but this has not yet been implemented.

B. Integrals

We perform the integrals appearing in Eq. (33) 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,[38] meaning our method collapses to a regular upwind scheme. Because we only seek updates at the vertices and edge-midpoints, the acoustic integral will either be



(a) Cells contributing to node update

(b) Cells contributing to edge update

Figure 9. Integrated areas for 2D acoustics

centered at a node or edge, as pictured in Fig. 9. Within each element, the integral of the function is simply the sum of the integrals of each basis, multiplied by the appropriate constant.

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 counter-clockwise order, with the edges and their associated normals numbered to match the opposing node. The local element numbering is illustrated in Fig. 10.

The intersection of the integral disc with the element edges sets the angular limits of integration. Point P and Q in Fig. 11 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 $\theta_Q = \tan^{-1}\left[\left(y_Q - y_1\right)/\left(x_Q - x_1\right)\right]$. Point P is always defined as the minimum angle, which increases in the counter-clockwise direction.

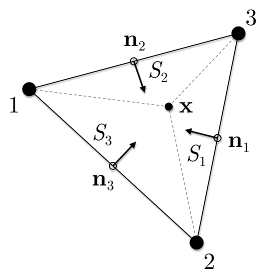
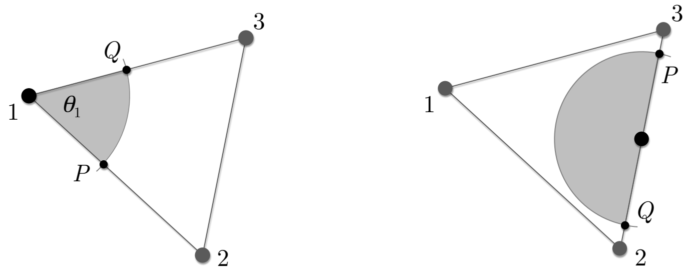


Figure 10. Element nomenclature



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

Figure 11. 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 $S = S_1 + S_2 + S_3$.

$$\begin{aligned} S_1 &= \frac{1}{2}\left[(x_2 - x)(y_3 - y) - (y_2 - y)(x_3 - x)\right] \\ S_2 &= \frac{1}{2}\left[(x_3 - x)(y_1 - y) - (y_1 - y)(x_3 - x)\right] \\ S_3 &= \frac{1}{2}\left[(x_1 - x)(y_2 - y) - (y_2 - y)(x_1 - x)\right] \end{aligned} \tag{34}$$

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

$$\begin{aligned} \eta &= \frac{S_3}{S} \\ \xi &= \frac{S_2}{S} \\ 1 - \xi - \eta &= \frac{S_1}{S} \end{aligned} \tag{35}$$

Substituting these expressions into those in Table 2, we find that all of the edge basis functions and nodal basis functions have two basic forms. Equation (36) 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 $1 - 1 = 3$).

$$\begin{aligned} \phi_{2i-1} &= \frac{S_i\,(2S_i - S)}{S^2} \\ \phi_{2j} &= \frac{4S_{j-1}S_{j+1}}{S^2} \end{aligned} \tag{36}$$

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

$$\phi_7 = 27S_1S_2S_3 \tag{37}$$

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

$$\begin{aligned}\nabla\phi_{2i-1} &= \left(\frac{4S_i - S}{2S^2}\right)\mathbf{n}_i \\ \nabla\phi_{2j} &= \frac{2}{S^2}\left(S_{j+1}\mathbf{n}_{j-1} + S_{j-1}\mathbf{n}_{j+1}\right)\end{aligned} \tag{38}$$

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

$$\nabla\phi_7 = \frac{27}{2S^3}\sum_{j=1}^{3} S_{j-1}S_{j+1}\mathbf{n}_j \tag{39}$$

The normal vectors are not unit vectors. They have a length equal to the length of the side with which they are associated.

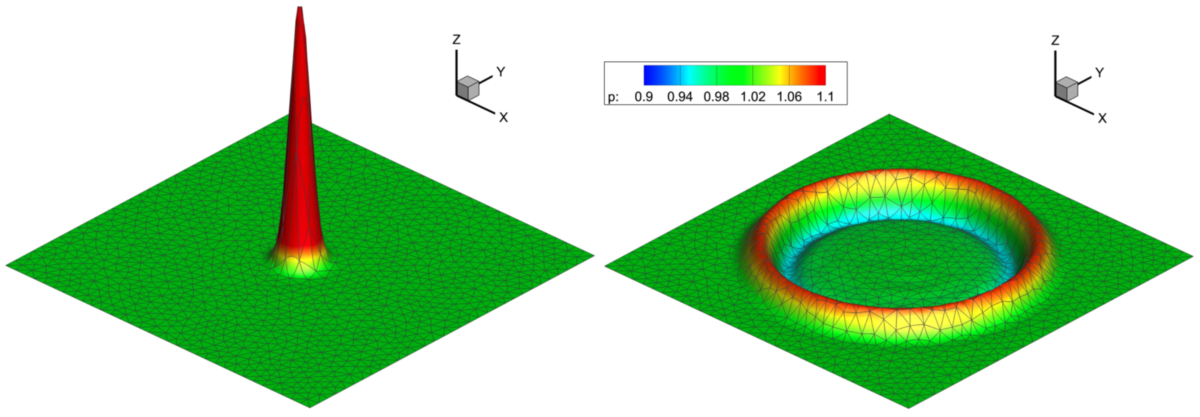
$$\mathbf{n}_k = \begin{pmatrix} y_{k+1} - y_{k-1} \\ x_{k-1} - x_{k+1} \end{pmatrix} \tag{40}$$

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 described in Appendix A. We have chosen to implement the closed form of the integrals; however, it may be more efficient to carry out the integration numerically. We plan to investigate the differences between the integration methods in the future.

C. Symmetry comparison with second-order schemes

One method of evaluating an acoustics method is to plot the solution at a specified time as a function of the radial distance from an initial disturbance. If the solution is perfectly symmetric, the solution should collapse to a single line. Equation (41) 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 $a_0 = 1.0$ and the free stream density was set to $\rho_0 = 1.4$. Figure 12 shows the initial condition and AF solution at $t = 1.25$.

$$\begin{aligned} p(\mathbf{x}, 0) &= 1 + \exp\left\{-\mu\left[(x - x_0)^2 + (y - y_0)^2\right]\right\} \\ \mathbf{u}(\mathbf{x}, 0) &= 0 \end{aligned} \tag{41}$$

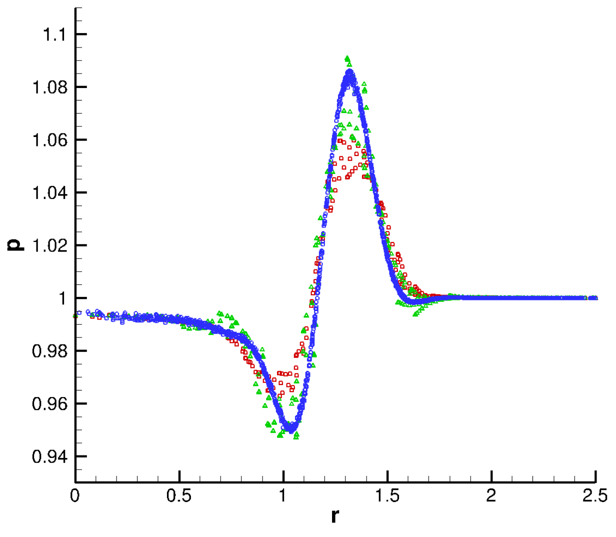


(a) $t = 0$ (b) $t = 1.25$

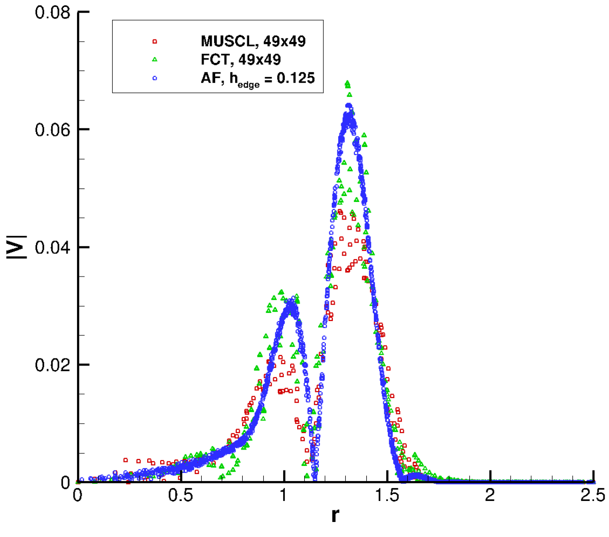
Figure 12. AF pressure solution for Gaussian pulse initial condition on the coarse mesh

The AF scheme was compared to two structured, second-order schemes: a vorticity-preserving scheme of Morton and Roe,[39] supplemented by a flux-corrected transport (FCT) limiter, and a more traditional MUSCL-Hancock method.[40] 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 $y \in [-2, 2]$. All of the schemes used a time step of $\Delta t = 2.5 \times 10^{-2}$ to march to a final time of $t = 1.25$. Figure

13 shows the AF result, MUSCL-Hancock method (MUSCL), and vorticity-preserving (FCT) second-order solutions for a coarse mesh while Fig. 14 shows the solutions for a finer mesh. In both cases, the AF shows greatly superior symmetry properties in both pressure and velocity magnitude.

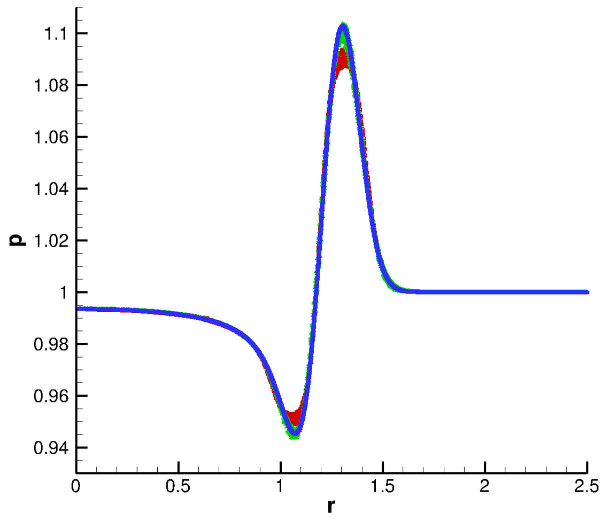


(a) Pressure

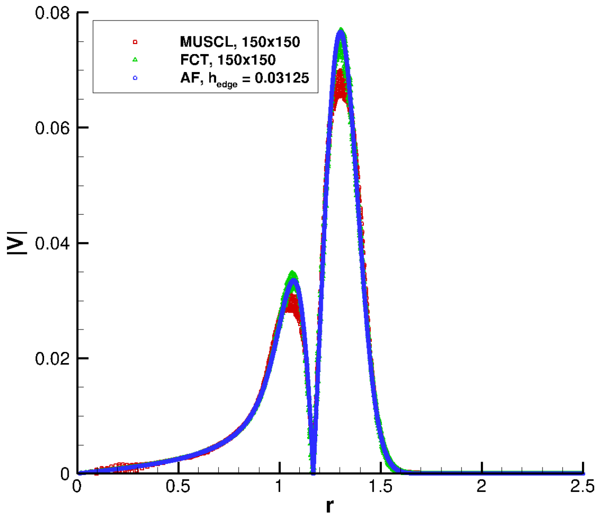


(b) Velocity magnitude

Figure 13. Acoustics solution as a function of radial distance from origin; coarse mesh



(a) Pressure



(b) Velocity magnitude

Figure 14. Acoustics solution as a function of radial distance from origin; fine mesh

D. 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:[41]

$$\begin{aligned} p(\mathbf{x},0) &= \frac{1}{a_0}\left[\sin(2\pi x)+\sin(2\pi y)\right] \\ u(\mathbf{x},0) &= v(\mathbf{x},0) = 0 \end{aligned} \tag{42}$$

They provide the exact solution to Eq. (42) as:

$$
\begin{aligned}
p(\mathbf{x},t) &= \frac{1}{a_0}\cos(2\pi a_0 t)\left[\sin(2\pi x)+\sin(2\pi y)\right] \\
\mathbf{u}(\mathbf{x},t) &= \frac{1}{a_0}\sin(2\pi a_0 t)\cos(2\pi \mathbf{x})
\end{aligned}
\tag{43}
$$

It is clear from Eq. (43) 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 15 and Tables 4-5 all show that the AF scheme achieves third-order accuracy as the mesh is refined.

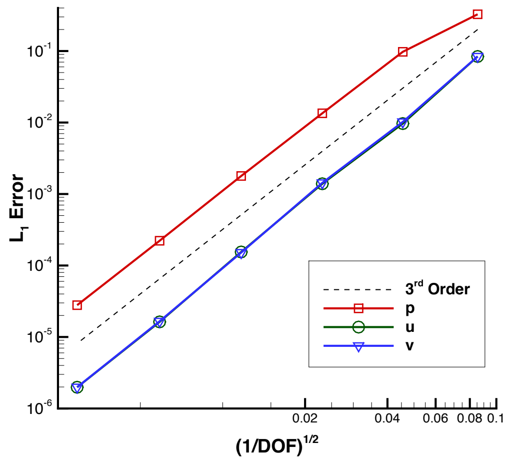


Figure 15. AF error convergence for acoustic test

Table 4. AF pressure convergence

Ref. Level	DOF$^{-1/2}$	$\|\|p\|\|_1$	$\mathcal{O}(L_1)$
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

Table 5. AF velocity convergence

Ref. Level	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

V. Linearized Euler

Model equations such as the ones detailed in Section II, Section III and Section IV 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:

$$\frac{\partial \mathbf{q}}{\partial t} + \frac{\partial \mathbf{f}}{\partial x} + \frac{\partial \mathbf{g}}{\partial y} = 0 \tag{44}$$

where,

$$\mathbf{q} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{pmatrix} \qquad \mathbf{f} = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho u v \\ \rho u H \end{pmatrix} \qquad \mathbf{g} = \begin{pmatrix} \rho v \\ \rho u v \\ \rho v^2 + p \\ \rho v H \end{pmatrix} \tag{45}$$

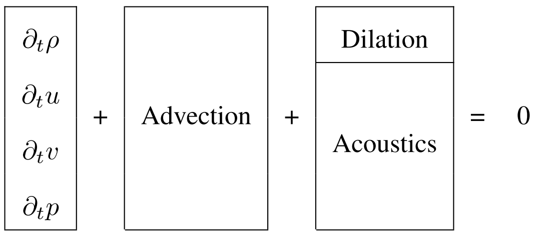
and,

$$\begin{aligned} E &= \frac{1}{\gamma - 1}\frac{p}{\rho} + \frac{1}{2}\left|\mathbf{u}\right|^2 \\ H &= E + \frac{p}{\rho} \end{aligned} \tag{46}$$

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:

$$\begin{aligned}
&\partial_t \rho &+\quad& u\partial_x \rho + v\partial_y \rho &+\quad& \rho\left(\partial_x u + \partial_y v\right) &=\quad& 0 &\quad&(47a)\\
&\partial_t u &+\quad& u\partial_x u + v\partial_y u &+\quad& \left(1/\rho\right)\partial_x p &=\quad& 0 &&(47b)\\
&\partial_t v &+\quad& u\partial_x v + v\partial_y v &+\quad& \left(1/\rho\right)\partial_y p &=\quad& 0 &&(47c)\\
&\partial_t p &+\quad& u\partial_x p + v\partial_y p &+\quad& \rho a^2\left(\partial_x u + \partial_y v\right) &=\quad& 0 &&(47d)
\end{aligned}$$

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



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 $\boldsymbol{\lambda} = (\alpha, \beta)^{\mathsf{T}}$.

$$\begin{aligned} \partial_t p^* + \alpha\partial_x p^* + \beta\partial_y p^* + a_0\left(\partial_x u^* + \partial_y v^*\right) = 0 \\ \partial_t u^* + \alpha\partial_x u^* + \beta\partial_y u^* + a_0\partial_x p^* = 0 \\ \partial_t v^* + \alpha\partial_x v^* + \beta\partial_y v^* + a_0\partial_y p^* = 0 \end{aligned} \tag{48}$$

Rewriting in terms of linear coefficient matrices:

$$\frac{\partial \mathbf{q}}{\partial t} + \mathbf{A}\frac{\partial \mathbf{q}}{\partial x} + \mathbf{B}\frac{\partial \mathbf{q}}{\partial y} = 0 \tag{49}$$

where

$$\mathbf{q} = \begin{pmatrix} p^* \\ u^* \\ v^* \end{pmatrix} \qquad \mathbf{A} = \begin{pmatrix} \alpha & a_0 & 0 \\ a_0 & \alpha & 0 \\ 0 & 0 & \alpha \end{pmatrix} \qquad \mathbf{B} = \begin{pmatrix} \beta & 0 & a_0 \\ 0 & \beta & 0 \\ a_0 & 0 & \beta \end{pmatrix} \tag{50}$$

Both $\mathbf{A}$ and $\mathbf{B}$ have the form $\mathcal{L}_1 + \mathcal{L}_2$ where one operator represents the pure advection problem from Section III 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.

$$\begin{aligned} \tilde{\mathbf{q}} &= \mathbf{q}^n + \Delta t \mathcal{L}_1(\mathbf{q}) \\ \mathbf{q}^{n+1} &= \tilde{\mathbf{q}} + \Delta t \mathcal{L}_2(\mathbf{q}) \end{aligned} \tag{51}$$

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. (42) 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 16 shows the error levels for the sequence of refined meshes and the convergence rates for pressure and velocity are detailed in Table 6 and Table 7. Comparision of the presssure results in Tables 4 and 6 shows that adding the convective terms increases the error by roughly 15%, but does not affect the order of accuracy. Similiarly, the velocity results in Tables 5 and 7 show that the linearized Euler solutions have a higher error level, but maintain third-order accuracy.

Demonstrating the AF method for the linearized Euler 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.

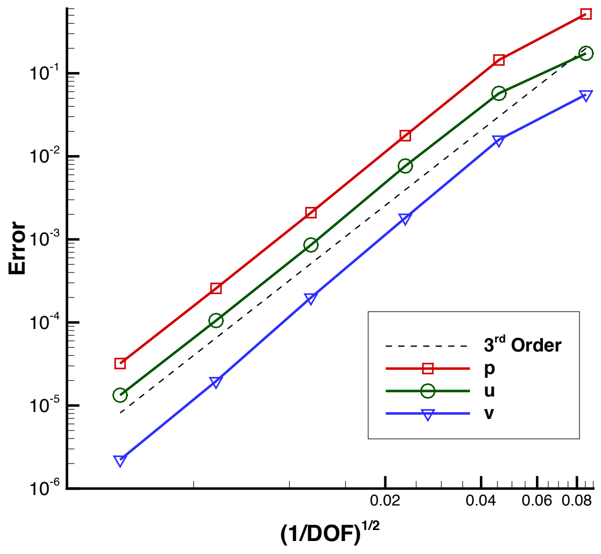


Figure 16. AF error convergence for linearized Euler equation

VI. Summary and Conclusions

This paper details the multidimensional extension 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,

Table 6. 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 7. 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

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 two dimensions, although the extension to three dimensions is straightforward. Each class of problem highlights different aspects of the AF method. The linear advection implementation covered in Sections II-III shows the basic mechanics of the update procedure and explicitly demonstrates how upwinding is incorporated. The linear acoustics equations are solved in Section IV using the fully multidimensional method of spherical means. Sections IV-V section detail how the freedom to use non-conservative edge updates allows radical departures from one-dimensional Riemann solutions. This feature facilitates the use of spherical means to calculate edge and vertex updates. The flexibility of the approach is also highlighted in Secion V by showing how trivial it is to extend the method to the linearized Euler equations.

As described, the AF method calculates exact fluxes, and therefore exact updates for arbitrary quadratic data, and this 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.

Future development of the method will be focused on the following areas: improving computational efficiency, proper treatment of full Euler equations, incorporation of viscous terms, addition of multidimensional limiters, and extension to three-dimensions. Most of these tasks are non-trivial; however, we are actively researching techniques for all of these topics and do not see any shortcomings with the method would prevent their full implementation.

A. Integrals of basis functions

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. (52).

$$M_R\{u\} = \frac{1}{2\pi R}\sum_{i=1}^{7} c_i \left(\int_{\theta_P}^{\theta_Q} \int_0^R \phi_i \frac{r}{\sqrt{R^2-r^2}} dr d\theta \right) \tag{52}$$

The integral appearing in Eq. (52) 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:

$$\begin{aligned} g_1(k) &= n_{k_y}\cos\theta_P - n_{k_x}\sin\theta_P \\ g_2(k) &= n_{k_y}\cos\theta_Q - n_{k_x}\sin\theta_Q \\ g_3(k) &= g_1(k) - g_2(k) \\ g_4(k) &= n_{k_x}\cos\theta_P + n_{k_y}\sin\theta_P \\ g_5(k) &= n_{k_x}\cos\theta_Q + n_{k_y}\sin\theta_Q \\ g_6(k) &= \mathbf{n}_k\cdot\mathbf{n}_{k+1} \end{aligned} \tag{53}$$

A. 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.

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:

$$g_1(k) = \begin{cases} \ell_k & \text{if } i = k+1 \\ \ell_k\cos(\theta_k + \theta_{i-1}) & \text{otherwise} \end{cases} \tag{54}$$

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

$$g_1(k) = \begin{cases} \ell_k & \text{if } j = k \\ \ell_k\cos(\theta_k + \theta_j) & \text{otherwise} \end{cases} \tag{55}$$

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:

$$g_2(k) = \begin{cases} -\ell_k & \text{if } i = k-1 \\ -\ell_k\cos(\theta_k + \theta_{i+1}) & \text{otherwise} \end{cases} \tag{56}$$

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

$$g_2(k) = \begin{cases} -\ell_k & \text{if } j = k \\ -\ell_k\cos(\theta_k + \theta_j) & \text{otherwise} \end{cases} \tag{57}$$

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.

$$g_4(k) = \begin{cases} \ell_{i+1}\sin\theta_i & \text{if } i = k-1 \\ -\ell_{i+1}\sin\theta_i & \text{if } i = k \\ 0 & \text{if } i = k+1 \end{cases} \tag{58}$$

4. *Simplification of* g_5

$$g_5(k) = \begin{cases} 0 & \text{if } i = k-1 \\ -\ell_{i-1}\sin\theta_i & \text{if } i = k \\ \ell_{i-1}\sin\theta_i & \text{if } i = k+1 \end{cases} \tag{59}$$

5. *Simplification of* g_6

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

$$g_6(k) = -\ell_{k-1}\ell_{k+1}\cos\theta_k \tag{60}$$

B. 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:

$$M_R\{u\} = \frac{1}{2\pi R}\left[\sum_{i=1}^{3}(c_{2i-1}I_{i,k}) + \sum_{j=1}^{3}(c_{2j}I_{j,k}) + c_7 I_{7,k}\right] \tag{61}$$

1. *Nodal basis index, nodal update*

$$I_{i,k} = \begin{cases} \left[1 + \frac{1}{6}\left(\frac{l_i R}{S}\right)^2\right] R\theta_k + \frac{3\pi R^2}{8S} g_3(i) - \frac{R^3}{6S^2}\left[g_2(i)g_5(i) - g_1(i)g_4(i)\right] & \text{if } i = k \\ \\ \frac{1}{6}\left(\frac{l_i R}{S}\right)^2 R\theta_k - \frac{\pi R^2}{8S} g_3(i) - \frac{R^3}{6S^2}\left[g_2(i)g_5(i) - g_1(i)g_4(i)\right] & \text{otherwise} \end{cases} \tag{62}$$

2. *Nodal basis index, edge update*

$$I_{i,k} = \begin{cases} \frac{\pi R^2}{12S^2}\left[2Rl_i^2 - 3Sg_1(i)\right] & \text{if } i = k \\ \\ \frac{\pi R^2}{12S^2}\left[2Rl_i^2 + 3Sg_1(i)\right] & \text{otherwise} \end{cases} \tag{63}$$

3. *Edge basis index, nodal update*

$$I_{j,k} = \frac{R^3}{3S^2}\left\{g_6(j)\theta_k + \left[g_4(j+1)g_5(j-1) - g_1(j+1)g_2(j-1)\right]\sin\theta_k\right\} + \zeta(j,k) \tag{64}$$

where

$$\zeta(j,k) = \begin{cases} 0 & \text{if } j = k \\ g_3(j-1) & \text{if } j = k-1 \\ g_3(j+1) & \text{if } j = k+1 \end{cases} \tag{65}$$

4. *Edge basis index, edge update*

$$I_{j,k} = \frac{\pi R^3}{3S^2} g_6(j) + \zeta(j,k) \tag{66}$$

where

$$\zeta(j,k) = \begin{cases} \pi R\left[1 + \frac{R}{2S} g_1(k)\right] & \text{if } j = k \\ \\ \frac{\pi R^2}{2S} g_1(k) & \text{otherwise} \end{cases} \tag{67}$$

5. Bubble index, node update

$$\begin{aligned} I_{7,k} = & \frac{9R^3}{4S^2}\left\{g_6(k)\theta_k + [g_4(k+1)g_5(k-1) + g_1(k+1)g_2(k-1)]\sin\theta_k\right\} \\ & + \frac{27\pi R^4}{128S^3}\Big\{\left(n_{1_x}n_{2_x}n_{3_x} - n_{1_y}n_{2_y}n_{3_x} - n_{1_y}n_{2_x}n_{3_y} - n_{1_x}n_{2_y}n_{3_y}\right)\left(\sin^3\theta_P - \sin^3\theta_Q\right) \\ & \qquad + \left(n_{1_y}n_{2_x}n_{3_x} + n_{1_x}n_{2_y}n_{3_x} + n_{1_x}n_{2_x}n_{3_y} - n_{1_y}n_{2_y}n_{3_y}\right)\left(\cos^3\theta_P - \cos^3\theta_Q\right) \\ & \qquad + 3\left[n_{1_y}n_{2_y}n_{3_y}\left(\cos\theta_P - \cos\theta_Q\right) - n_{1_x}n_{2_x}n_{3_x}\left(\sin\theta_P - \sin\theta_Q\right)\right]\Big\} \end{aligned} \tag{68}$$

6. Bubble index, edge update

$$\begin{aligned} I_{7,k} = & \frac{9\pi R^2 l_k}{8S}\left(\frac{3}{2} - \frac{Rl_k}{S}\right) \\ & + \frac{27\pi R^4}{64S^3}\Big[\left(n_{1_x}n_{2_x}n_{3_x} - n_{1_y}n_{2_y}n_{3_x} - n_{1_y}n_{2_x}n_{3_y} - n_{1_x}n_{2_y}n_{3_y}\right)\sin^3\theta_P \\ & \qquad + \left(n_{1_y}n_{2_x}n_{3_x} + n_{1_x}n_{2_y}n_{3_x} + n_{1_x}n_{2_x}n_{3_y} - n_{1_y}n_{2_y}n_{3_y}\right)\cos^3\theta_P \\ & \qquad + 3\left(n_{1_y}n_{2_y}n_{3_y}\cos\theta_P - n_{1_x}n_{2_x}n_{3_x}\sin\theta_P\right)\Big] \end{aligned} \tag{69}$$

C. 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 as Eq. (53).

$$\begin{aligned} M_R\{\nabla u\} &= \frac{1}{2\pi R}\sum_{i=1}^{7} c_i\left(\int_{\theta_P}^{\theta_Q}\int_0^R \nabla\phi_i \frac{r}{\sqrt{R^2-r^2}}drd\theta\right) \\ &= \frac{1}{2\pi R}\left[\sum_{i=1}^{3}\left(c_{2i-1}\mathbf{J}_{i,k}\right) + \sum_{j=1}^{3}\left(c_{2j}\mathbf{J}_{j,k}\right) + c_7\mathbf{J}_{7,k}\right] \end{aligned} \tag{70}$$

1. Nodal basis index, nodal update

$$\mathbf{J}_{i,k} = \begin{cases} \frac{R}{4S^2}\left[\pi R g_3(i) + 6S\theta_k\right]\mathbf{n}_i & \text{if } i = k \\ \\ \frac{R}{4S^2}\left[\pi R g_3(i) - 2S\theta_k\right]\mathbf{n}_i & \text{otherwise} \end{cases} \tag{71}$$

2. Nodal basis index, edge update

$$\mathbf{J}_{i,k} = \begin{cases} \frac{\pi R}{2S^2}\left(Rg_1(i) - S\right)\mathbf{n}_i & \text{if } i = k \\ \\ \frac{\pi R}{2S^2}\left(Rg_1(i) + S\right)\mathbf{n}_i & \text{otherwise} \end{cases} \tag{72}$$

3. Edge basis index, nodal update

$$\mathbf{J}_{j,k} = \frac{\pi R^2}{4S^2}\left[\mathbf{n}_{j+1}g_3(j-1) + \mathbf{n}_{j-1}g_3(j+1)\right] + \zeta(j,k) \tag{73}$$

where,

$$\zeta(j,k) = \begin{cases} 0 & \text{if } j = k \\ \mathbf{n}_{j+1}\frac{2R\theta_k}{S} & \text{if } j = k+1 \\ \mathbf{n}_{j-1}\frac{2R\theta_k}{S} & \text{if } j = k-1 \end{cases} \tag{74}$$

4. *Edge basis index, edge update*

$$\mathbf{J}_{j,k} = \frac{\pi R^2}{2S^2}\left[\mathbf{n}_{j+1} g_1(j-1) + \mathbf{n}_{j-1} g_1(j+1) + \zeta(j,k)\right] \tag{75}$$

where,

$$\zeta(j,k) = \begin{cases} \frac{2S}{R}\left(\mathbf{n}_{j+1} + \mathbf{n}_{j-1}\right) & \text{if } j = k \\ \mathbf{n}_{j+1}\frac{2S}{R} & \text{if } j = k-1 \\ \mathbf{n}_{j-1}\frac{2S}{R} & \text{if } j = k+1 \end{cases} \tag{76}$$

5. *Bubble index, node update*

$$\begin{aligned} \mathbf{J}_{7,k} = {} & \frac{27\pi R^2}{16S^2}\left[\mathbf{n}_{k-1} g_3(k+1) + \mathbf{n}_{k+1} g_3(k-1)\right] \\ & + \frac{9R^3}{8S^2}\left\{\sum_{m=1}^{3} \mathbf{n}_m g_6(m+1)\right. \\ & \left. + \sin\theta_k \sum_{m=1}^{3} \mathbf{n}_m \left[g_4(m+1) g_5(m-1) - g_1(m+1) g_2(m-1)\right]\right\} \end{aligned} \tag{77}$$

6. *Bubble index, edge update*

$$\mathbf{J}_{7,k} = \frac{27\pi R}{8S^2}\left[S - R g_1(k)\right]\mathbf{n}_k + \frac{9\pi R^3}{8S^3}\sum_{m=1}^{3} \mathbf{n}_m g_6(m+1) \tag{78}$$

References

[1]Dean, J. P., Morton, S. A., McDaniel, D. R., Clifton, J. D., and Bodkin, D. J., "Aircraft Stability and Control Characteristics Determined by System Identification of CFD Simulations," *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, 2008, AIAA Paper 2008-6378.

[2]Morton, S. A., Cummings, R. M., and Kholodar, D. B., "High Resolution Turbulence Treatment of F/A-18 Tail Buffet," *Journal of Aircraft*, Vol. 44, No. 6, 2007, pp. 1769–1775.

[3]Johnson, R. A., Stanek, M. J., and Grove, J. E., "Store Separation Trajectory Deviations Due to Unsteady Weapons Bay Aerodynamics," *46th AIAA Aerospace Sciences Meeting*, 2008, AIAA Paper 2008-0188.

[4]Shea, M. J., Constantino, M. M., O'Brien, C. W., Snyder, M. R., Simpson, S. A., and Cenko, A., "Litening Pod Modification to Improve MK-83 Store Trajectories," *29th AIAA Applied Aerodynamics Conference*, 2011, AIAA Paper 2011-3158.

[5]Vassberg, J. C., Tinoco, E. N., Mani, M., Rider, B., Zickuhr, T., Levy, D. W., Brodersen, O. P., Eisfeld, B., Crippa, S., Wahls, R. A., Morrison, J. H., Mavriplis, D. J., and Murayama, M., "Summary of the Fourth AIAA CFD Drag Prediction Workshop," *28th AIAA Applied Aerodynamics Conference*, 2010, AIAA Paper 2010-4547.

[6]Godunov, S. K., "A Difference Scheme for Numerical Computation of Discontinuous Solution of Hydrodynamic Equations," *Matematicheskii Sbornik*, Vol. 47, 1959, pp. 271–306.

[7]Roe, P. L., "Approximate Riemann solvers, parameter vectors, and difference schemes," *Journal of Computational Physics*, Vol. 43, 1981, pp. 357–372.

[8]Rusanov, V., "The Calculation of the Interaction of Non-stationary Shock waves and obstacles," *USSR Computational Mathematics and Mathematical Physics*, Vol. 1, No. 2, 1962, pp. 304–320.

[9]Harten, A., Lax, P. D., and van Leer, B., "On Upstream Differencing and Godunov-type Schemes for Hyperbolic Conservation Laws," *SIAM Review*, Vol. 25, No. 1, 1983, pp. 35–60.

[10]Peery, K. and Imlay, S., "Blunt-Body Flow Simulations," *AIAA/SAE/ASME/ASEE 24th Joint Propulsion Conference*, 1988, AIAA Paper 88-2904.

[11]Quirk, J. J., "A Contribution to the Great Riemann Solver Debate," *International Journal for Numerical Methods in Fluids*, Vol. 18, 1994, pp. 555–574.

[12]Tramel, R. W. and Nichols, R. H., "Addition of Improved Shock-Capturing Schemes to OVERFLOW 2.1," *19th AIAA Computational Fluid Dynamics Conference*, 2009, AIAA Paper 2009-3988.

[13]Iserles, A., "Generalized Leapfrog Methods," *IMA Journal of Numerical Analysis*, Vol. 6, 1986, pp. 381–392.

[14]Roe, P. L., "Linear Bicharacteristic Schemes Without Dissipation," *SIAM Journal of Scientific Computing*, Vol. 19, No. 5, 1998, pp. 1405–1427.

[15]Thomas, J., *An Investigation of the Upwind Leapfrog Method for Scalar Advection and Acoustic/Aeroacoustic Wave Propagation Problems*, Ph.D. thesis, University of Michigan, 1996.

[16]Eymann, T., *Active Flux Schemes*, Ph.D. thesis, University of Michigan, 2013.

[17]Karabasov, S. and Goloviznin, V., "Compact Accurately Boundary-Adjusting high-REsolution Technique for fluid dynamics," *Journal of Computational Physics*, Vol. 228, 2009, pp. 7426–7451.

[18]Bouche, D., Bonnaud, G., and Ramos, D., "Comparison of Numerical Schemes for Solving the Advection Equation," *Applied Mathematics Letters*, Vol. 16, 2003, pp. 147–154.

[19]Löhner, R., "Error and Work Estimates for High Order Elements," *49th AIAA Aerospace Sciences Meeting*, 2011, AIAA Paper 2011-0211.

[20]Warming, R., Kutler, P., and Lomax, H., "Second- and Third-Order Noncentered Differnce Schemes for Nonlinear Hyperbolic Equations," *AIAA Journal*, Vol. 11, No. 2, 1973, pp. 189–196.

[21]Harten, A., Engquist, B., Osher, S., and Charkravarthy, S. R., "Uniformly high order accurate essentially non-oscillatory schemes, III," *Journal of Computational Physics*, Vol. 71, 1987, pp. 231–303.

[22]Liu, X.-D., Osher, S., and Chan, T., "Weighted Essentially Non-oscillatory Schemes," *Journal of Compuatational Physics*, Vol. 115, 1994, pp. 200–212.

[23]Ollivier-Gooch, C. F., "Quasi-ENO Schemes for Unstructured Meshes Based on Unlimited Data-Dependent Least-Squares Reconstruction," *Journal of Computational Physics*, Vol. 133, 1997, pp. 6–17.

[24]Friedrich, O., "Weighted Essentially Non-Oscillatory Schemes for the Interpolation of Mean Values on Unstructured Grids," *Journal of Computational Physics*, Vol. 144, 1998, pp. 194–212.

[25]Cockburn, B., Lin, S.-Y., and Shu, C.-W., "TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws III: one-dimensional systems," *Journal of Computational Physics*, Vol. 84, 1989, pp. 90–113.

[26]Wang, Z. J. and Liu, Y., "The Spectral Difference Method for the 2D Euler Equations on Unstructured Grids," *17th AIAA Computational Fluid Dynamics Conference*, 2005, AIAA Paper 2005-5112.

[27]Wang, Z. J., "Spectral (Finite) Volume Method for Conservation Laws on Unstructured Grids: Basic Formulation," *Journal of Computational Physics*, Vol. 178, 2002, pp. 210–251.

[28]Brooks, A. N. and Hughes, T. J., "Streamline Upwind/Petrov-Galerkin Formulations for Convection Dominated Flows with Particular Emphasis on the Incompressible Navier-Stokes Equations," *Computer Methods in Applied Mechanics and Engineering*, Vol. 32, 1982, pp. 199–259.

[29]Allmaras, S. R., Bussoletti, J. E., Hilmes, C. L., Johnson, F. T., Melvin, R. G., Tinoco, E. N., Venkatakrishnan, V., Wigdon, L. B., and Young, D. P., "Algorithm Issues and Challenges Associated with the Development of Robust CFD Codes," *Variational Analysis and Aerospace Engineering*, Springer New York, 2009, pp. 1–19.

[30]Kuzmin, D., "A vertex-based hierarchical slope limiter for *p*-adaptive discontinuous Galerkin methods," *Journal of Computational and Applied Mathematics*, Vol. 233, 2010, pp. 3077–3085.

[31]Peraire, J., Nguyen, N., and Cockburn, B., "A Hybridizable Discontinuous Galerkin Method for the Compressible Euler and Navier-Stokes Equations," *48th AIAA Aerospace Sciences Meeting*, 2010, AIAA Paper 2010-0363.

[32]Peraire, J., Nguyen, N., and Cockburn, B., "An Embedded Discontinuous Galerkin Method for the Compressible Euler and Navier-Stokes Equations," *20th AIAA Computational Fluid Dynamics Conference*, 2011, AIAA Paper 2011-3228.

[33]van Leer, B., "Towards the Ultimate Conservative Difference Scheme. IV. A New Approach to Numerical Convection," *Journal of Computational Physics*, Vol. 23, No. 3, 1977, pp. 276–299.

[34]Colella, P. and Woodward, P. R., "The Piecewise Parabolic Method (PPM) for gas-dynamical Simulations," *Journal of Computational Physics*, Vol. 54, 1984, pp. 174–201.

[35]Eymann, T. A. and Roe, P. L., "Active Flux Schemes for Systems," *20th AIAA Computational Fluid Dynamics Conference*, 2011, AIAA Paper 2011-3840.

[36]Fidkowski, K. J. and Roe, P. L., "An Entropy Adjoint Approach to Mesh Refinement," *SIAM Journal on Scientific Computing*, Vol. 32, 2010, pp. 1261–1287.

[37]Fidkowski, K. J. and Luo, Y., "Output-based Space-Time Mesh Adaptation for the Compressible Navier-Stokes Equations," *Journal of Computational Physics*, Vol. 230, 2011, pp. 5753–5773.

[38]Courant, R. and Hilbert, D., *Methods of Mathematical Physics*, Vol. 2, Wiley-VCH, 1962.

[39]Morton, K. and Roe, P. L., "Vorticity-Preserving Lax-Wendroff-Type Schemes for the System Wave Equation," *SIAM Journal on Scientific Computing*, Vol. 23, No. 1, 2001, pp. 170–192.

[40]Toro, E. F., *Riemann Solvers and Numerical Methods for Fluid Dynamics*, Springer, 3rd ed., 2009.

[41]Lukáčová-Medvid'ová, M., Morton, K., and Warnecke, G., "Evolution Galerkin methods for hyperbolic systems in two space dimensions," *Mathematics of Computation*, Vol. 69, No. 232, 2000, pp. 1355–1384.