A Sweeping Positivity-Preserving Implicit Continuous Galerkin Scheme for Euler Equations

D. Chloe Griffin[1] · Chi-Wang Shu[1]

Abstract
We extend the positivity preserving sweeping algorithm in Griffin and Shu (2025) [8] for density and the nonlinear pressure function in the compressible Euler equations to a nodal, implicit, continuous finite element framework. The procedure corrects negative density and pressure values while maintaining conservation of mass, momentum, and energy. As in [8], the procedure is easily extended to other hyperbolic conservation law systems, and may be applied to any concave function of conserved variables. We provide numerical examples for both P_1 and P_2 elements in one and two dimensions on tough test cases including blast waves, interacting shocks, and near-vacuum states. Our emphasis in this paper is not to recommend the underlying spatial scheme, but to demonstrate the effectiveness of the sweeping limiter in handling large magnitude unphysical values. For smooth problems or numerical methods with sufficient oscillation control, we find only a few iterations are needed for nodal positivity treatment of the nonlinear pressure function. In oscillatory cases with large violations in magnitude or quantity, we enhance convergence of the iterative pressure solver using multi-grid techniques.

Keywords Positivity preserving · Compressible Euler equations · Hyperbolic conservation laws · Gas dynamics · Finite elements · Continuous Galerkin · Implicit Methods

Mathematics Subject Classification 65M60; 65M12; 76M10

1 Introduction

Positivity violations of density and pressure (and internal energy) in the Euler equations lead to ill-posedness, non-physical distortions, and numerical blow-ups. This is particularly due to imaginary sound speed computation which breaks hyperbolicity of the system. Density and pressure positivity are also relevant to Navier-Stokes and magnetohydrodynamics systems. Other physical quantities, including water height in shallow water equations, mass/volume fraction in multiphase flows, and the probability density function in Vlasov-Boltzmann transport equations, also have strict positivity requirements, leading to the need of robust and flexible positivity-preserving meth-

✉ Chi-Wang Shu
chi-wang_shu@brown.edu
D. Chloe Griffin
danielle_griffin@brown.edu

[1] Division of Applied Mathematics, Brown University, Providence, RI 02906, USA

ods. Applications of structure-preserving methods arise in computational fluid dynamics, computational astrophysics, plasma simulations, and other diverse fields [34]. Many numerical methods for hyperbolic conservation laws are not inherently bound-preserving.

Even popular high-order and robust methods such as the finite volume (FV) and finite difference (FD) essentially non-oscillatory (ENO) schemes [9, 13] and weighted ENO (WENO) schemes [13, 22, 28], and discontinuous Galerkin (DG) methods [7], are not inherently positivity preserving. For continuous Galerkin methods, the classical scheme requires stabilization for advection-dominated problems, e.g. streamlined-upwind Petrov Galerkin (SUPG) formulation given in [6]. We refer to a historical overview of the SUPG method for compressible flows in [11]. Even with stabilization, the method exhibits oscillations around shocks and discontinuities [11], requiring additional positivity treatment. Furthermore, popular shock-capturing techniques are not inherently bound-preserving. This is especially apparent in high Mach flows, near strong shocks, or in near-vacuum states [16]. Nevertheless, continuous finite element methods, and particularly the SUPG method, are popular in computational fluid dynamics and discretizations for compressible flows due to their flexibility on unstructured grids, popularity with a large range of partial differential equations (PDEs) in various applications, and availability of multiple computer softwares. One common approach to treat negative values is through flux corrected transport.

The flux-corrected transport method (FCT) was originally produced in [3–5]. The method ensures positivity by applying limited anti-diffusive corrections to bound-preserving low-order solutions. There are many applications of flux-corrected transport for continuous Galerkin formulations, coined FEM-FCT, including those for Euler and Navier-Stokes equations [14, 18, 19, 24]. Another class of methods which are similar to FCT are algebraic flux correction (AFC), which enforces an FCT like protocol through algebraic constraints. In the procedure, one modifies off-diagonal entries of the global stiffness matrix to create an $M-$matrix for which is used to produce a low-order, monotonic solution. The AFC procedure finds a limited solution using anti-diffusive fluxes as in FCT [17]. A more recent advance in the field is monolithic convex limiting, which uses convex combinations of nodal values to give invariant domain guarantees [15]. Modern applications of these methods are highly successful in explicit frameworks. However, extending to the implicit regime carries several computational and theoretical hurdles.

Implicit time-stepping procedures are of interest due to their computational efficiency over explicit time-steppers in situations where the CFL condition limits to very small time-step sizes or where complicated geometric restraints such as those in adaptive mesh refinement lead to time-step restrictions. Examples of such problems include stiff source terms in turbulence or chemical reaction models, as well as problems with high Reynolds number with boundary layer regions having large aspect ratio grids [10]. While the computational cost of a single iteration of an implicit time-stepping procedure is more expen-

sive than an explicit one, the total reduction in the number of time-steps often yields a more efficient scheme. Thus, it is of interest to provide a positivity-preserving method without additional restrictive conditions on the time-step or additional computational cost in the nonlinear Newton solve. Existing positivity preserving limiters for continuous finite elements, such as FEM-FCT, AFC, or monolithic flux limiting, are not easily extended to fully implicit schemes due to the non-differentiable updates to the residual and restrictive conditions on the time-step, which require approximate Jacobian and complex preconditioner for the Newton iteration, or specialized techniques such as pseudo-implicit time stepping [14, 16, 25, 26]. In this paper, we apply positivity treatment as a post-processing step as an extension of [8], therefore allowing the computation of an analytical Jacobian computation in the Newton solve. Nevertheless, Newton convergence remains a challenge in our simulations. Thus, we introduce adaptive time stepping within our sweeping framework.

The sweeping positivity preserving procedure in [8] extends the scalar sweeping procedure in [23] to treat positivity violations of density and the nonlinear pressure function for the Euler equations in a fifth-order finite difference WENO scheme. While the treatment of density follows directly from [23], the nonlinear pressure function follows a local protocol to iteratively adjust negative points with neighboring values using the Zhang-Shu scaling limiter in [33] as a projection technique. While the original scaling limiter preserves positivity by flattening a DG or finite volume reconstruction polynomial towards a positive cell-average, the sweeping limiter uses the scaling limiter to scale along a line between a point with negative pressure and its neighbor. If the neighboring point has positive pressure, the protocol gives a clear, corrected update for the negative point. In the case of a negative neighbor, the protocol obtains an estimated distance to the physically admissible set using a globally conserved quantity, but ensures the update will at most halve the distance between points. To ensure conservation, the neighbor used in the update is equally scaled in the opposite direction. The procedure is applicable to any nonlinear concave function of conserved variables, and may be easily extended to finite volume WENO and discontinuous Galerkin methods. However, the extension to continuous Galerkin methods is nontrivial due to the continuous nature of the CG polynomial. Furthermore, an implicit time-stepping procedure is an additional challenge in the extension, as well as the oscillatory nature of finite element method near strong shocks.

In this paper, we extend the simple sweeping procedure in [8] to a continuous Galerkin setting as a modular post-processing procedure that is incorporated in an implicit setting. To do this, we first incorporate weights into the procedure that were not necessary in the uniform finite difference WENO setting. While the incorporation of weights is a natural extension from the scalar sweeping limiter for density, the nodal adjustment procedure requires care in preserving the convexity of the neighboring updates. Furthermore, the choice of appropriate weights that maintains conservation while allowing nodal

positivity adjustments is not immediately obvious. To determine appropriate weights, we consider the locally conserved quantity defined in [27] which, when summed, recovers global conservation. With this quantity, we find that the weights are given as the normalized row sum of the consistent mass matrix, which we show are consistent with exact nodal quadrature rules. These findings allow a theoretical extension to many nodal finite element methods, and may scale to higher degree polynomials with positive weights.

For the implicit-time stepping procedure, we apply the sweep after each stage update. Thus, we apply a stabilization of choice, allow the Newton solver to converge with an analytical Jacobian (found through automatic differentiation) to potentially unphysical values, then apply the correction as a postprocessing procedure. For the stabilization, we use the SUPG method, and in most cases incorporate additional shock capturing or artificial viscosity terms. Our focus is on the sweeping procedure, not on the spatial procedure. Thus, we emphasize that the numerical results are preliminary and may not be optimal in terms of oscillation control or numerical dissipation. Instead, we highlight the quantity and magnitude of violations which the sweeping limiter successfully corrects. While the original method only needed one or two sweeps to converge for the pressure treatment in the finite difference WENO setting, we find that many iterations are needed in some numerical tests with large violations. Thus, we introduce a multigrid procedure to improve convergence. For the weighted sweeping procedure, we provide the same theoretical guarantees as in [8] for conservation, monotone improvement, and variance reduction. We also provide an accuracy proof for the density treatment in the CG setting. In our numerical tests, we provide results for P_1 and P_2 elements in one and two dimensions. We show that the procedure obtains optimal numerical convergence rates, maintains positivity, and prevents blow-ups for robust test cases.

The paper is organized as follows. In Section 2, we provide relevant description of the Euler equations, the spatial and temporal discretizations, and related shock capturing terms. In Section 3, we provide the core content of the paper, including the derivation of algorithmic weights, details of the sweeping procedure, theoretical results, optional multigrid acceleration, and extension to multiple dimensions. In Section 4, we provide numerical results. In Section **??**, we provide concluding remarks and describe future work.

2 Preliminaries

2.1 Euler Equations

We present a positivity preserving implicit continuous finite element method for the Euler equations for a perfect gas. In conservative form, the one-dimensional version is given by,

$$\mathbf{u}_t + \mathbf{f}(\mathbf{u})_x = 0, \quad t \geqslant 0, \quad x \in \mathbb{R}, \tag{1}$$

$$\mathbf{u} = \begin{pmatrix} \rho \\ m \\ E \end{pmatrix}, \quad \mathbf{f}(\mathbf{u}) = \begin{pmatrix} m \\ \rho v^2 + p, \\ (E+p)v \end{pmatrix}$$

and

$$m = \rho v, \quad E = \frac{1}{2}\rho v^2 + \rho e, \quad p = (\gamma - 1)\rho e,$$

where ρ is density, m is the momentum, v is the velocity, p is the pressure, E is total energy, and e is the internal energy. The constant $\gamma > 1$ is chosen as 1.4 for air. In conservative form, the nonlinear pressure function is written as

$$p(\mathbf{u}) = (\gamma - 1)(E - \frac{1}{2}\frac{m^2}{\rho}).$$

An important feature of the pressure function is its concavity for $\rho > 0$, which is easily verified through its negative semi-definite Hessian matrix. The concavity of the pressure function is useful in the design of the positivity preserving method for the pressure function in [8]. We extend the finite difference notation used there to consider weighted quantities which will be relevant for the finite element extension. We denote the number of degrees of freedom with n_{dof}. We extend the definition of the averaged conserved quantity as the global mean $\bar{\mathbf{u}} = \sum_{j=1}^{n_{\text{dof}}} w_j \mathbf{u}_j$, where $\sum_{j=1}^{n_{\text{dof}}} w_j = 1$, $w_j > 0$ for $j = 1, ..., n_{\text{dof}}$. Here, $\{\mathbf{u}_j\}_{j=1}^{n_{\text{dof}}}$ are the nodal degrees of freedom. The determination of appropriate weights is outlined in Section 3.1 for a generic nodal finite element method. Considering two nodal degrees of freedom $\mathbf{u}_1$ and $\mathbf{u}_2$ where $\rho_1, \rho_2 > 0$, we have

$$p(t\mathbf{u}_1 + (1-t)\mathbf{u}_2) \geqslant tp(\mathbf{u}_1) + (1-t)p(\mathbf{u}_2) \tag{2}$$

for all $t \in (0, 1)$. The extended sweeping algorithm is designed to ensure that all nodal degrees of freedom have positive density and pressure. We use $\epsilon = 10^{-12}$ as the robust positivity threshold. Thus, following the notation of [35] and [8], we aim to ensure all nodal degrees of freedom are in the convex admissible set

$$\mathcal{G} = \left\{ \mathbf{u} = \begin{pmatrix} \rho \\ m \\ E \end{pmatrix} \middle| \rho > \epsilon \quad \text{and} \quad p = (\gamma - 1)(E - \frac{1}{2}\frac{m^2}{\rho}) > \epsilon \right\}. \tag{3}$$

As in [8], we assume that the initial condition satisfies $\rho(\mathbf{u}_j^0) > \epsilon$ and $p(\mathbf{u}_j^0) > \epsilon$ for $j = 1, ..., n_{\text{dof}}$. For a conservative scheme with periodic or compact boundary condition, we have a natural assumption that

$$p(\bar{\mathbf{u}}^k) = p\left(\sum_{j=1}^{n_{\text{dof}}} w_j \mathbf{u}_j^k\right) = p\left(\sum_{j=1}^{n_{\text{dof}}} w_j \mathbf{u}_j^0\right) \geqslant \sum_{j=1}^{n_{\text{dof}}} w_j p(\mathbf{u}_j^0) > \epsilon \tag{4}$$

at arbitrary time level k. As in the finite difference pressure sweep, we obtain the global pivot point $\bar{\mathbf{u}}$ which is known to have positive pressure for all time

steps. Nevertheless, we may not assume that that $\overline{p(\mathbf{u})} = \sum_{j=1}^{n_{\text{dof}}} w_j p(\mathbf{u}_j) > \epsilon$ once time stepping begins without positivity treatment. For writing out the implicit continuous Galerkin scheme, it is useful to write the Euler equations in quasi-linear form. Denoting the flux Jacobian $\mathbf{A}(\mathbf{u}) = \frac{\partial \mathbf{f}}{\partial \mathbf{u}}$, we apply chain rule to obtain the quasi-linear system,

$$\mathbf{u}_t + A(\mathbf{u})\frac{\partial \mathbf{u}}{\partial x} = 0. \tag{5}$$

The quasi-linear form is useful in the description of the numerical discretization in Section 2.2.

2.2 Implicit Continuous Galerkin Scheme

For clarity, we define a 1D mesh with n_{dof} nodes x_j, mesh size $\Delta x = x_{j+1} - x_j$, and intervals $I_j = [x_j, x_{j+1}]$ for $j = 1, ..., n_{el} - 1$, where n_{el} denotes the number of elements. We also define half points $x_{j+\frac{1}{2}} = \frac{1}{2}(x_{j+1} + x_j)$, and dual intervals $I_j^* = [x_{j-\frac{1}{2}}, x_{j+\frac{1}{2}}]$ for $j = 1, ..., n_{el} - 1$. We denote the partitioning of the set as $\mathcal{I}_h = \{I_1, I_2, ..., I_{n_{el}-1}\}$. We define a finite-element space of continuous, piecewise polynomials,

$$\mathbf{V}_h = \{\mathbf{v} \in \mathbf{H}^1(\Omega) : \mathbf{v} \in \mathbf{P}_k(I), \forall I \in \mathcal{I}_h\},$$

where $\mathbf{H}^1(\Omega) = [H^1(\Omega)]^3$, $\mathbf{P}^k(\Omega) = [P^k(\Omega)]^3$ and $P_k(I)$ denotes the set of polynomials up to degree k defined on the cell I. The continuous Galerkin (CG) formulation for (5) becomes,

$$\begin{cases} \text{Find } \mathbf{u}_h(\cdot, t_n) \in \mathbf{V}_h^k \text{ such that for all test functions } \mathbf{v}_h \in \mathbf{V}_h^k, \\ 0 = \int_\Omega \mathbf{v}_h \cdot (\frac{\partial}{\partial t}\mathbf{u}_h + \mathbf{A}(\mathbf{u}_h)\frac{\partial \mathbf{u}_h}{\partial x}). \end{cases} \tag{6}$$

On its own, the Galerkin weak form is known to be unstable for advection dominated flows. Thus, we use a Streamlined-Upwind Petrov-Galerkin (SUPG) formulation as introduced by Brooks and Hughes in [6] and further detailed in [12] and [21]. We refer to [11] for a comparison of several stabilizations. We most closely follow the conservative formulation given in [21] with and without the shock capturing term depending on the severity of shocks in the numerical tests as detailed in Section 4. For the Petrov-Galerkin scheme, we define a separate finite-dimensional function space

$$\mathbf{W}_h = \{\mathbf{w} \in \mathbf{H}^1(\Omega) : \mathbf{w} \in \mathbf{P}_k(I), \forall I \in \mathcal{I}_h\},$$

where we use a weighted test function in place of $\mathbf{v}_h$ in equation (6). Thus, the updated form becomes,

$$\begin{cases} \text{Find } \mathbf{u}_h(\cdot, t_n) \in \mathbf{V}_h^k \text{ such that for all test functions } \mathbf{w}_h \in \mathbf{W}_h^k, \\ 0 = \int_\Omega \left(\mathbf{w}_h + \sum_{I \in \mathcal{I}_h} \mathbf{T}_I \frac{\partial \mathbf{w}_h}{\partial x}\right) \cdot (\frac{\partial}{\partial t}\mathbf{u}_h + \mathbf{A}(\mathbf{u}_h)\frac{\partial \mathbf{u}_h}{\partial x}), \end{cases} \tag{7}$$

where $\mathbf{T}_I = \tau \mathbf{A}^T$, and we use the a transient definition of τ which is given by

$$\tau = \left(\frac{1}{\Delta t^2} + \left(\frac{2\alpha_I}{h_I} \right)^2 \right)^{-1/2}, \tag{8}$$

where h_I is the element length, and α_I is the local maximum wave speed i.e. the spectral radius of the flux Jacobian given by

$$\alpha_I = \mathsf{v\_h} + c_h, \tag{9}$$

with local sound speed $c_h = \sqrt{\gamma p_h / \rho_h}$. The max α is computed locally over nodal degrees of freedom in the cell. The time-step $\Delta t = \text{cfl} \frac{h_{\min}}{\alpha}$ where $h_{\min}$ is the minimum cell diameter, and $\alpha = \max(\alpha_I)$ is the global maximum incorporating all degrees of freedom in the mesh. For our time discretizations, we use the third order, three stage, L-stable diagonally implicit Runge–Kutta (SDIRK3) scheme in [1]. Using the method of lines, we consider the spatial operator

$$\frac{\partial \mathbf{u}}{\partial t} = \mathcal{L}(u) = -A(\mathbf{u}) \frac{\partial \mathbf{u}}{\partial x}.$$

The update for stage (i) of the three-stage Runge-Kutta time integrator is written as

$$\begin{aligned} \mathbf{u}^{(i)} =& \mathbf{u}^n + \Delta t \sum_{j=1}^{i} \alpha_{ij} \mathcal{L}(\mathbf{u}^{(j)}) \\ =& \mathbf{u}^n + \Delta t \sum_{j=1}^{i} \alpha_{ij} \left(-\mathbf{A}(\mathbf{u}^{(j)}) \frac{\partial \mathbf{u}^{(j)}}{\partial x} \right). \end{aligned} \tag{10}$$

The coefficients α_{ij} are given in the Butcher tableau

$$\begin{array}{c|ccc} \gamma & \gamma & 0 & 0 \\ \frac{1+\gamma}{2} & \frac{1-\gamma}{2} & \gamma & 0 \\ 1 & b_1 & b_2 & \gamma \\ \hline & b_1 & b_2 & \gamma \end{array}$$

where γ is the middle root of the polynomial $x^3 - 3x^2 + 1.5x - \frac{1}{6} = 0$, $b_2 = \frac{1-4\gamma+2\gamma^2}{1-\gamma}$, and $b_1 = 1 - \gamma - b_2$. Using the SDIRK3 update, (10), we define

$$r^{(i)}_{\text{strong}} = \frac{\mathbf{u}^{(i)}_h - \mathbf{u}^n_h}{\Delta t} + \sum_{j=1}^{i} \alpha_{ij} \mathbf{A}(\mathbf{u}^{(j)}_h) \frac{\partial \mathbf{u}^{(j)}_h}{\partial x}.$$

Splitting the test function and applying integration by parts only to the flux term, we obtain the residual formula for the weak form,

$$\mathcal{R}_i(\mathbf{u}_h^{(i)}) = \int_\Omega \mathbf{w}_h \cdot \left(\frac{\mathbf{u}_h^{(i)} - \mathbf{u}_h^n}{\Delta t} \right) dx$$
$$- \sum_{j=1}^{i} \alpha_{ij} \int_\Omega \frac{\partial \mathbf{w}_h}{\partial x} \cdot \mathbf{F}(\mathbf{u}_h^{(j)})\, dx + \sum_{j=1}^{i} \alpha_{ij} \left[\mathbf{w}_h \cdot \mathbf{F}(\mathbf{u}_h^{(j)}) \right]_{x_L}^{x_R}$$
$$+ \int_\Omega \left(\sum_{I \in \mathcal{I}_h} \mathbf{T}_I \frac{\partial \mathbf{w}_h}{\partial x} \right) \cdot r_{\text{strong}}^{(i)}\, dx = 0.$$

Because $\mathcal{R}_i = 0$ is a highly non-linear system, we iterate toward the stage solution using a globalized Newton's method. Let k denote the Newton iteration index for a given stage i. The state is updated via

$$\mathbf{u}_h^{(i,k+1)} = \mathbf{u}_h^{(i,k)} + L\delta\mathbf{u}_h$$

where L is the step length. The search direction $\delta\mathbf{u}_h$ is found by solving the linear system

$$\mathcal{J}\delta\mathbf{u}_h = -\mathcal{R}_i(\mathbf{u}_h^{(i,k)}).$$

For the Newton solver, we implement Newton's method in three ways. First, a classic Newton's method assuming $\alpha = 1.0$. If this fails, we use a line-search algorithm. If this fails, we use a trust-region method. We also use an adaptive time-stepping feature, where if all three Newton solvers fail to converge after 12 iterations and to an absolute error tolerance of 10^{-8}, we halve Δt and restart that step. We apply a ramp factor to increase Δt back to the quantity set by the time-stepping condition. Results of the adaptive time-stepping protocol are documented in Section 4.

We note that there is a great deal of literature on the selection of τ. Our focus in this paper is the positivity preservation provided by the sweeping limiter. Thus, we do not claim that our selection of τ is optimal and we may see oscillations or overly smoothed regions in the numerical results. In fact, we document the large negative errors and quantity of unphysical points treated by the sweeping limiter in the numerical tests. For shock capturing, we consider two potential stabilization terms. The first is given in [21] and adds a shock capturing artificial viscosity term based on the entropy variable formulation. The second is the YZ-β shock capturing approach. Both of these implementations are described in Section 2.3. Different test cases warrant different applications, and the choices are outlined in Section 4.

2.3 Shock-Capturing and Artificial Viscosity

While the SUPG formulation in Section 2.2 provides optimal order of accuracy on smooth problems, the results become highly oscillatory in the presence of strong shocks. Many shock capturing terms and alternative formulations

exist in the literature. For this reason, we employ two strategies to mitigate these oscillations for improved results in numerical tests. The first is an artificial viscosity term included in [21] for the conservative variable formulation. The added term is derived from the entropy variables. The second is the YZ-β shock capturing term provided in [29] with $\beta = 1$. In both cases, the modified residual formula for the weak form is written,

$$\begin{aligned}\mathcal{R}_i(\mathbf{u}_h^{(i)}) = &\int_\Omega \mathbf{w}_h \cdot \left(\frac{\mathbf{u}_h^{(i)} - \mathbf{u}_h^n}{\Delta t} \right) dx \\ &- \sum_{j=1}^{i} \alpha_{ij} \int_\Omega \frac{\partial \mathbf{w}_h}{\partial x} \cdot \mathbf{F}(\mathbf{u}_h^{(j)})\, dx + \sum_{j=1}^{i} \alpha_{ij} \left[\mathbf{w}_h \cdot \mathbf{F}(\mathbf{u}_h^{(j)}) \right]_{x_L}^{x_R} \\ &+ \int_\Omega \left(\sum_{I \in \mathcal{I}_h} \mathbf{T}_I \frac{\partial \mathbf{w}_h}{\partial x} \right) \cdot r_{\text{strong}}^{(i)}\, dx \\ &+ \sum_{e=1}^{n_{el}} \int_{\Omega^e} \frac{\partial \mathbf{w}_h}{\partial x} \cdot \left(\nu \frac{\partial \mathbf{u}_h^{(i)}}{\partial x} \right) dx = 0. \end{aligned} \tag{11}$$

The ν parameter is chosen differently for each approach. For the artificial viscosity term, we set $\nu = \nu_d$. For the YZ-β formulation, we set $\nu = \nu_{YZ\beta}$.

We first describe the inclusion of an artificial viscosity term as described in [21]. To suppress spurious oscillations near shocks, we introduce a non-linear discontinuity-capturing operator based on the entropy variable formulation. The artificial viscosity ν_d acts as a scalar diffusion coefficient applied to the conservative variables. We define this parameter locally within each element as

$$\nu_d = \min \left(C_{DC} \left[\frac{\left(\mathbf{A} \frac{\partial \mathbf{u}_h^{(i)}}{\partial x} \right) \cdot \tilde{\mathbf{A}}_0^{-1} \left(\mathbf{A} \frac{\partial \mathbf{u}_h^{(i)}}{\partial x} \right)}{\left(\frac{\partial \xi}{\partial x} \frac{\partial \mathbf{u}_h^{(i)}}{\partial x} \right) \cdot \tilde{\mathbf{A}}_0^{-1} \left(\frac{\partial \xi}{\partial x} \frac{\partial \mathbf{u}_h^{(i)}}{\partial x} \right) + \epsilon} \right]^{1/2}, \frac{1}{2} h_I c_h \right),$$

where C_{DC} is a user-defined $\mathcal{O}(1)$ tuning constant, and ϵ is a small absolute regularization parameter to prevent division by zero in regions of uniform flow, and ξ is the reference (local) component. The bounding term $\frac{1}{2} h_I c_h$ ensures the artificial viscosity does not exceed the local characteristic transport limit, preventing over-smearing of the solution. The symmetrization matrix $\tilde{\mathbf{A}}_0$ is defined as the Jacobian mapping the entropy variables $\tilde{\mathbf{U}}$ to the conservative variables $\mathbf{U}$ such that $\tilde{\mathbf{A}}_0 = \frac{\partial \mathbf{U}}{\partial \tilde{\mathbf{U}}}$. By performing a transformation of variables $\tilde{\mathbf{U}} = \tilde{\mathbf{U}}(\mathbf{U})$, we can express the inverse matrix in terms of the variable mapping:

$$\tilde{\mathbf{A}}_0^{-1} = \frac{\partial \tilde{\mathbf{U}}}{\partial \mathbf{U}},$$

where the generalized entropy variables $\tilde{\mathbf{U}}$ (often denoted $\mathbf{V}$ in discrete implementations) for the Euler equations are:

$$\tilde{\mathbf{U}}(\mathbf{U}) = \frac{1}{\rho i}\begin{bmatrix} -U_4 + \rho i(\gamma + 1 - s + s_0) \\ U_2 \\ U_3 \\ -U_1 \end{bmatrix}.$$

In our numerical implementation, rather than explicitly assembling and inverting the 4×4 matrix $\tilde{\mathbf{A}}_0$, we define the analytic mapping from conservative to entropy variables and use automatic differentiation to evaluate $\tilde{\mathbf{A}}_0^{-1}$ directly. This provides a reduction in both memory and computation cost in comparison to inverting directly.

We now describe the YZ-β implementation as in Tezduyar and Senga [29], which suppresses most spurious oscillations near shocks without being as overly diffusive as the artificial viscosity term. The ν parameter is written as as

$$\nu_{\mathrm{YZ}\beta} = \min\left(C_{\mathrm{YZ}\beta}\, h_I \left(\frac{\|\tilde{\mathbf{r}}^{(i)}_{\mathrm{strong}}\|}{\|\tilde{\nabla}\tilde{\mathbf{u}}^{(i)}_h\|}\right)^{\beta}, \frac{1}{2} h_I c_h\right),$$

where $C_{\mathrm{YZ}\beta}$ is a user-defined $\mathcal{O}(1)$ tuning constant, and we select $\beta = 1$. As in the artificial viscosity stabilization, the bounding term $\frac{1}{2}h_I c_h$ prevents oversmearing. Because the components of the Euler equations have vastly different units and magnitudes, the residual and gradient norms are nondimensionalized. We define a diagonal scaling matrix $\mathbf{S}$ based on reference states:

$$\mathbf{S} = \mathrm{diag}\left(\frac{1}{\rho_{\mathrm{ref}}}, \frac{1}{\rho_{\mathrm{ref}} c_{\mathrm{ref}}}, \frac{1}{E_{\mathrm{ref}}}\right).$$

Using this scaling matrix, the non-dimensional residual and gradient norms used in the YZ-β stabilization are evaluated as:

$$\|\tilde{\mathbf{r}}^{(i)}_{\mathrm{strong}}\| = \sqrt{\left(\mathbf{S}\, r^{(i)}_{\mathrm{strong}}\right) \cdot \left(\mathbf{S}\, r^{(i)}_{\mathrm{strong}}\right) + \epsilon^2},$$

$$\|\tilde{\nabla}\tilde{\mathbf{u}}^{(i)}_h\| = \sqrt{\left(\mathbf{S}\frac{\partial \mathbf{u}^{(i)}_h}{\partial x}\right) \cdot \left(\mathbf{S}\frac{\partial \mathbf{u}^{(i)}_h}{\partial x}\right) + \epsilon^2},$$

where ϵ is a small absolute regularization parameter introduced to prevent division by zero in regions of uniform flow. Even with stabilization, the numerical scheme may produce large unphysical values for density and pressure near strong shocks or in near-vacuum states. Thus, we describe the positivity preserving sweeping procedure in the following section.

3 Sweeping Procedure

3.1 Density and Pressure Extension

To extend bound-preserving techniques from [8] to continuous FEM, we first obtain a globally conserved quantity that directly relates the nodal degrees of freedom $\{\mathbf{u}_i\}_{i=1}^{n_{\text{dof}}}$ with a set of weights, $\{w_i\}_{i=1}^{n_{\text{dof}}}$ where $\sum w_i = 1$ and $w_i > 0$ for $i = 1, ..., n_{\text{dof}}$. Towards this goal, we define a locally conserved quantity $\bar{\mathbf{u}}_j^n$ as in [27]. For continuous elements, this is given by

$$\bar{\mathbf{u}}_j^n = \frac{1}{\left|I_j^*\right|} \int_\Omega \mathbf{u}_h(x, t_n)\phi_j \, dx. \tag{12}$$

Applying the partition of unity property, we show that summing the locally conserved quantity recovers the exact global mass:

$$\begin{aligned}
\sum_{j=0}^{n_{\text{dof}}} \left|I_j^*\right| \bar{\mathbf{u}}_j^n &= \sum_{j=0}^{n_{\text{dof}}} \int_\Omega \mathbf{u}_h(x, t_n)\phi_j \, dx \\
&= \int_\Omega \mathbf{u}_h(x, t_n) \left(\sum_{j=0}^{n_{\text{dof}}} \phi_j \right) dx \\
&= \int_\Omega \mathbf{u}_h(x, t_n) \, dx.
\end{aligned}$$

Thus, using the locally conserved quantity, we rewrite the globally conserved quantity as

$$\begin{aligned}
\int_\Omega \mathbf{u}_h(x, t_n) dx = \sum_{j=0}^{n_{\text{dof}}} \left|I_j^*\right| \bar{\mathbf{u}}_j^n &= \sum_{j=0}^{n_{\text{dof}}} \left|I_j^*\right| \left(\frac{1}{|I_j|^*} \int_\Omega \mathbf{u}_h^{n_{\text{dof}}} \phi_j dx \right) \\
&= \sum_{i=0}^{n_{\text{dof}}} \mathbf{u}_i^n \sum_{j=0}^{n_{\text{dof}}} \int_\Omega \phi_i(x)\phi_j(x) dx.
\end{aligned}$$

Rewriting in this way, we obtain raw weights $W_i \equiv \sum_{j=0}^{n_{\text{dof}}} \int_\Omega \phi_i(x)\phi_j(x) dx$ that when multiplied and summed with the nodal finite element degrees of freedom recover global conservation. These raw weights are equivalent to the row sum of the mass matrix $\mathbf{M}$, where

$$W_i = \sum_j \mathbf{M}_{ij} = \sum_j \int_\Omega \phi_i(x)\phi_j(x) \, dx.$$

Furthermore, applying partition of unity and evaluating with a nodal quadrature rule (where points x_q match the nodes x_j) that is exact for polynomials of degree k, the Kronecker delta property ($\phi_i(x_j) = \delta_{ij}$) collapses the integral so that we find

$$W_i = \int_\Omega \phi_i(x) \left(\sum_j \phi_j(x) \right) dx = \int_\Omega \phi_i(x)\, dx = \sum_j \omega_j \phi_i(x_j) = \omega_i.$$

Thus, the raw weights and the row sums of the mass matrix are exactly the nodal weights in classic nodal quadrature rules as shown in Table 1.

Element	Geometry	Equivalent Quadrature Rule	Positivity
1D Simplicial (Lines)			
P_1 (Linear)	Line	Trapezoidal Rule	Yes ($W_i > 0$)
P_2 (Quadratic)	Line	Simpson's 1/3 Rule	Yes ($W_i > 0$)
P_3 (Cubic)	Line	Simpson's 3/8 Rule	Yes ($W_i > 0$)
2D Tensor-Product (Quadrilaterals)			
Q_1 (Bilinear)	Quad	2D Trapezoidal Rule	Yes ($W_i > 0$)
Q_2 (Biquadratic)	Quad	2D Simpson's 1/3 Rule	Yes ($W_i > 0$)
Q_3 (Bicubic)	Quad	2D Simpson's 3/8 Rule	Yes ($W_i > 0$)
2D Simplicial (Triangles)			
P_1 (Linear)	Triangle	3-Point Vertex Rule	Yes ($W_i > 0$)
P_2 (Quadratic)	Triangle	6-Point Nodal Rule	**No** ($\exists W_i = 0$)

Table 1 Table displaying positivity and equivalence of nodal quadrature rules and raw conservative weights W_i for $i = 1, ..., n_{\text{dof}}$ [36].

We observe that all of the quadrature rules listed in Table 1 provide positive raw weights, with the exception of P_2 in two dimensions, where weights at vertex nodes are zero. In two dimensions, we use a special treatment for P_2 vertex nodes as described in Section 3.3. The density and pressure sweep in [8] strictly requires dimensionless, positive weights that sum to one ($\sum w_i = 1$). Therefore, we obtain our final algorithmic weights, $\{w_i\}$ as

$$w_i = \frac{W_i}{\sum_{i=1}^{n_{\text{dof}}} W_i}, \text{for } i = 1, ..., n_{\text{dof}}.$$

The normalized weights, w_i, conserves global mean $\bar{\mathbf{u}} = \frac{1}{|\Omega|} \int_\Omega \mathbf{u}_h(x, t_n) dx$, which implies global mass conservation. Therefore, algorithm weights are easily obtained by the normalized quadrature weights, or normalized row sums of the mass mass matrix.

The extension of the pressure sweeping algorithm in [8] is given in Procedure 1. Given a set of nodal degrees of freedom $\{\mathbf{u}_i\}_{i=1}^{n_{\text{dof}}}$ where $\sum_{j=1}^{n_{\text{dof}}} w_j \rho_j > \epsilon$ and $p(\bar{\mathbf{u}}) = p(\sum_{j=1}^{n_{\text{dof}}} w_j \mathbf{u}_j) > \epsilon$, the weighted pressure sweeping procedure provides a method to iteratively adjust all nodal degrees of freedom towards the admissible set $\mathcal{G}$ in (3). The procedure continues until all density and pressure points $\rho_j \in \{\rho_j\}_{j=1}^{n_{\text{dof}}}$ and $p(\mathbf{u}_j) \in \{p(\mathbf{u}_j)\}_{j=1}^{n_{\text{dof}}}$ are positive. As in [8], the procedure ensures conservation of $\mathbf{u}$ so that the global average is unchanged. The procedure follows exactly that of [8] except for the incorporation of weights.

Weights are naturally incorporated into the density and pressure sweep as they are originally included in [23]. The incorporation of weights in the nodal adjustment is a notable extension. Here, we follow the same procedure as in [8], but must limit the adjustment according to the weighted contribution of the neighbor. Thus, we incorporate a cap t for the scaled adjustment which preserves convexity of the local adjustment and achieves the same monotonic improvement and variance reduction properties as in the original procedure. For the positive branch of the nodal adjustment, we note that the procedure still provides the optimal direction of adjustment, and is simply limited by weighted contribution of the positive neighbor. We note that the procedure is equivalent to [8] for uniform weights where $w_i = \frac{1}{n_{\text{dof}}}$ for $i = 1, ..., n_{\text{dof}}$ and provides clarity on the application for cell averages in a finite volume or discontinuous Galerkin context. We denote the discrete L^2 norm in $\mathbb{R}^3$ by $\|\cdot\|$, which is defined for any vector $\mathbf{x} = (x_1, x_2, x_3) \in \mathbb{R}^3$ as

$$\|\mathbf{x}\| = \sqrt{x_1^2 + x_2^2 + x_3^2}.$$

We provide the weighted sweeping algorithm here:

Procedure 1. (A conservative positivity-preserving sweeping procedure)
$\epsilon \leftarrow 10^{-12}$

1. Apply Algorithm 1 to $\{\rho_j\}_{j=1}^{n_{\text{dof}}}$:
   $\rho \leftarrow DensitySweep(\rho, w, \epsilon)$

2. Apply the positivity preserving sweep in Algorithm 2 using the nodal adjustment in Algorithm 3:
   while $\min(p(\mathbf{u})) < \epsilon$ **do**
   $\mathbf{u} \leftarrow PressureSweep(\mathbf{u}, w, \epsilon)$
   end while

Algorithm 1: $\rho^* = \text{DensitySweep}(\rho, w, \epsilon)$ (Algorithm 2.1 of [23])

for $j = 1 : n_{\text{dof}} - 1$ **do** ▷ Forward Sweep
  if $\rho_j < \epsilon$ **then**
    $\rho_j^* \leftarrow \epsilon$
    $d \leftarrow \rho_j - \epsilon$
    $\rho_{j+1}^* \leftarrow \rho_{j+1} + \frac{w_j}{w_{j+1}} d$
  end if
end for
for $j = n_{\text{dof}} : 2$ **do** ▷ Backward Sweep
  if $\rho_j < \epsilon$ **then**
    $\rho_j^* \leftarrow \epsilon$
    $d \leftarrow \rho_j - \epsilon$
    $\rho_{j-1}^* \leftarrow \rho_{j-1} + \frac{w_j}{w_{j-1}} d$
  end if
end for

Algorithm 2: $\mathbf{u}^* = \text{PressureSweep}(\mathbf{u}, w, \epsilon)$

for $j = 1 : n_{\text{dof}} - 1$ **do** ▷ Forward Sweep
  if $p(\mathbf{u}_j) < \epsilon$ **then**
    $\mathbf{u}_j^* \leftarrow \text{NodeAdjust}(\mathbf{u}_j, \mathbf{u}_{j+1}, \bar{\mathbf{u}}, w_j, w_{j+1}, \epsilon)$
    $\mathbf{d} \leftarrow \mathbf{u}_j - \mathbf{u}_j^*$
    $\mathbf{u}_{j+1}^* \leftarrow \mathbf{u}_{j+1} + \frac{w_j}{w_{j+1}} \mathbf{d}$
  end if
end for
for $j = n_{\text{dof}} : 2$ **do** ▷ Backward Sweep
  if $p(\mathbf{u}_j) < \epsilon$ **then**
    $\mathbf{u}_j^* \leftarrow \text{NodeAdjust}(\mathbf{u}_j, \mathbf{u}_{j-1}, \bar{\mathbf{u}}, w_j, w_{j-1}, \epsilon)$
    $\mathbf{d} \leftarrow \mathbf{u}_j - \mathbf{u}_j^*$
    $\mathbf{u}_{j-1}^* \leftarrow \mathbf{u}_{j-1} + \frac{w_j}{w_{j-1}} \mathbf{d}$
  end if
end for

Algorithm 3: $\mathbf{u}_1^* = \text{NodeAdjust}(\mathbf{u}_1, \mathbf{u}_2, \bar{\mathbf{u}}, w_1, w_2, \epsilon)$

if $p(\mathbf{u}_2) > \epsilon$ **then**
  $t \leftarrow \min\left(\frac{p(\mathbf{u}_1) - \epsilon}{p(\mathbf{u}_1) - p(\mathbf{u}_2)}, \frac{w_2}{w_1}\right)$
else
  $t_1 \leftarrow \frac{p(\mathbf{u}_1) - \epsilon}{(p(\mathbf{u}_1) - p(\bar{\mathbf{u}}))}$
  $t \leftarrow \min\left(t_1 \left(\frac{\|\mathbf{u}_1 - \bar{\mathbf{u}}\|}{\|\mathbf{u}_1 - \mathbf{u}_2\|}\right), 1/4, \frac{w_2}{w_1}\right)$
end if
$\mathbf{u}_1^* \leftarrow (1 - t)\mathbf{u}_1 + t\mathbf{u}_2$

We obtain the same useful properties of the algorithm as in [8].

Property 1 (Conservation)**.** *We show that none of the operations in Algorithm 2 change the globally conserved quantity* $\bar{\boldsymbol{u}}$*. We denote* $\boldsymbol{u}_n$ *as a node where* $p(\boldsymbol{u}_n) < \epsilon$ *and* $\boldsymbol{u}_p$ *as the neighboring node in the direction of sweep. The weighted update ensures*

$$w_n \boldsymbol{u}_n^* + w_p \boldsymbol{u}_p^* = w_n \boldsymbol{u}_n^* + w_p(\boldsymbol{u}_p + \frac{w_n}{w_p}(\boldsymbol{u}_n - \boldsymbol{u}_n^*)) = w_n \boldsymbol{u}_n + w_p \boldsymbol{u}_p.$$

Therefore, $\bar{\boldsymbol{u}}$ *is unchanged and* $p(\bar{\boldsymbol{u}}) > \epsilon$*.*

Property 2 (Monotonic Improvement)**.** *We show that the weighted operations still only decrease the distance between points, thereby increasing the average of pressure. We first rewrite*

$$\begin{aligned}
\boldsymbol{u}_p^* =& \boldsymbol{u}_p + \frac{w_n}{w_p}(\boldsymbol{u}_n - \boldsymbol{u}_n^*) \\
=& \boldsymbol{u}_p + \frac{w_n}{w_p}\boldsymbol{u}_n - \frac{w_n}{w_p}((1-t)\boldsymbol{u}_n + t\boldsymbol{u}_p)) \\
=& \frac{w_n}{w_p} t\boldsymbol{u}_n + (1 - \frac{w_n}{w_p} t)\boldsymbol{u}_p.
\end{aligned}$$

Using this definition and applying Cauchy-Schwarz the bound the adjusted distance as

$$\begin{aligned}
\left\|\boldsymbol{u}_p^* - \boldsymbol{u}_n^*\right\| =& \left\| \boldsymbol{u}_p + \frac{w_n}{w_p}(\boldsymbol{u}_n - \boldsymbol{u}_n^*) - \boldsymbol{u}_n^* \right\| \\
=& \left\| \boldsymbol{u}_p + \frac{w_n}{w_p}(\boldsymbol{u}_n - ((1-t)\boldsymbol{u}_n + t\boldsymbol{u}_p)) - ((1-t)\boldsymbol{u}_n + t\boldsymbol{u}_p) \right\| \\
=& \left\| (1 - t(1 + \frac{w_n}{w_p}))\boldsymbol{u}_p - (1 - t(1 + \frac{w_n}{w_p}))\boldsymbol{u}_n \right\| \\
=& \left\| (1 - t(1 + \frac{w_n}{w_p}))(\boldsymbol{u}_p - \boldsymbol{u}_n) \right\| \\
\leqslant& \left\| (1 - t(1 + \frac{w_n}{w_p})) \right\| \|(\boldsymbol{u}_p - \boldsymbol{u}_n)\|
\end{aligned}$$

From the nodal adjustment procedure, we know that $t \in (0, 1)$ *as in [8]. Furthermore, the minimum with* $\frac{w_p}{w_n}$ *ensures*

$$t \leqslant \frac{w_p}{w_n} \implies t\frac{w_n}{w_p} \leqslant 1.$$

Furthermore, $w_n, w_p \in (0, 1)$ *by definition. Thus,* $\frac{tw_n}{w_p} \in (0, 1)$ *and*

$$t(1+\frac{w_n}{w_p}) = t + \frac{tw_n}{w_p} \in (0,2),$$

which implies

$$\left\| 1 - t(1+\frac{w_n}{w_p}) \right\| \leqslant 1.$$

Therefore,

$$\left\| \boldsymbol{u}_p^* - \boldsymbol{u}_n^* \right\| \leqslant \| \boldsymbol{u}_p - \boldsymbol{u}_n \|.$$

As points move closer together, their combined pressure increases due to the concavity of the pressure function. Through Jensen's inequality, we find that

$$\begin{aligned} w_p p(\boldsymbol{u}_p^*) + w_n p(\boldsymbol{u}_n^*) =& w_p p(\frac{w_n}{w_p} t\boldsymbol{u}_n + (1 - \frac{w_n}{w_p} t)\boldsymbol{u}_p) + w_n p((1-t)\boldsymbol{u}_n + t\boldsymbol{u}_p) \\ &\geqslant w_n t p(\boldsymbol{u}_n) + w_p p(\boldsymbol{u}_p) - w_n t p(\boldsymbol{u}_p) \\ &+ w_n p(\boldsymbol{u}_n) - w_n t p(\boldsymbol{u}_n) + w_n t p(\boldsymbol{u}_p) \\ &= w_p p(\boldsymbol{u}_p) + w_n p(\boldsymbol{u}_n). \end{aligned}$$

Property 3 (Variance Reduction)**.** *As the points move closer together, the total variance decreases. We show that elements monotonically move toward the set $\mathcal{G}$ and the expected value, $\bar{\boldsymbol{u}}$ as sweeping is continually applied. We consider how the total weighted variance $\mathbb{V} = \sum_{j=1}^{n_{dof}} w_j \|\boldsymbol{u}_j - \bar{\boldsymbol{u}}\|^2$ changes upon each combined nodal adjustment. We define the local weighted variance for $\boldsymbol{u}_p$ and $\boldsymbol{u}_n$ as:*

$$\mathbb{V}_{n,p} := w_n \|\mathbb{V}_n\|^2 + w_p \|\mathbb{V}_p\|^2 := w_n \|\boldsymbol{u}_n - \bar{\boldsymbol{u}}\|^2 + w_p \|\boldsymbol{u}_p - \bar{\boldsymbol{u}}\|^2,$$

and we want to examine how it changes in relation to the adjusted local variance:

$$\mathbb{V}_{n,p}^* := w_n \|\mathbb{V}_n^*\|^2 + w_p \left\|\mathbb{V}_p^*\right\|^2 := w_n \|\boldsymbol{u}_n^* - \bar{\boldsymbol{u}}\|^2 + w_p \left\|\boldsymbol{u}_p^* - \bar{\boldsymbol{u}}\right\|^2.$$

Through a simple calculation, using the updated weighted points, we see that:

$$\mathbb{V}_n^* = \boldsymbol{u}_n^* - \bar{\boldsymbol{u}} = (1-t)\boldsymbol{u}_n + t\boldsymbol{u}_p - \bar{\boldsymbol{u}} = (1-t)(\boldsymbol{u}_n - \bar{\boldsymbol{u}}) + t(\boldsymbol{u}_p - \bar{\boldsymbol{u}}) = (1-t)\mathbb{V}_n + t\mathbb{V}_p$$

Similarly, using the weighted adjustment for $\boldsymbol{u}_p^*$, we find:

$$\mathbb{V}_p^* = \boldsymbol{u}_p^* - \bar{\boldsymbol{u}} = \frac{w_n}{w_p} t \mathbb{V}_n + \left(1 - \frac{w_n}{w_p} t\right) \mathbb{V}_p$$

We expand the updated local variance

$$\begin{aligned}
\mathbb{V}_{n,p}^* =& w_n \|V_n^*\|^2 + w_p \left\|V_p^*\right\|^2 \\
=& w_n \|(1-t)\mathbb{V}_n + t\mathbb{V}_p\|^2 + w_p \left\| \frac{w_n}{w_p} t \mathbb{V}_n + \left(1 - \frac{w_n}{w_p} t\right) \mathbb{V}_p \right\|^2 \\
=& w_n \left((1-t)^2 \|\mathbb{V}_n\|^2 + t^2 \|\mathbb{V}_p\|^2 + 2t(1-t)\mathbb{V}_n \cdot \mathbb{V}_p \right) \\
&+ w_p \left(\left(\frac{w_n}{w_p} t\right)^2 \|\mathbb{V}_n\|^2 + \left(1 - \frac{w_n}{w_p} t\right)^2 \|\mathbb{V}_p\|^2 + 2\frac{w_n}{w_p} t \left(1 - \frac{w_n}{w_p} t\right) \mathbb{V}_n \cdot \mathbb{V}_p \right)
\end{aligned}$$

Now, we collect the coefficients for $\|\mathbb{V}_n\|^2$, $\|\mathbb{V}_p\|^2$, *and* $2\mathbb{V}_n \cdot \mathbb{V}_p$. *For* $\|\mathbb{V}_n\|^2$*:*

$$w_n(1-t)^2 + w_p \left(\frac{w_n}{w_p} t\right)^2 = w_n(1 - 2t + t^2) + \frac{w_n^2}{w_p} t^2 = w_n - 2w_n t + w_n t^2 \left(1 + \frac{w_n}{w_p}\right)$$

For $\|\mathbb{V}_p\|^2$*:*

$$w_n t^2 + w_p \left(1 - \frac{w_n}{w_p} t\right)^2 = w_n t^2 + w_p \left(1 - 2\frac{w_n}{w_p} t + \frac{w_n^2}{w_p^2} t^2\right) = w_p - 2w_n t + w_n t^2 \left(1 + \frac{w_n}{w_p}\right)$$

For $2\mathbb{V}_n \cdot \mathbb{V}_p$*:*

$$w_n t(1-t) + w_p \left(\frac{w_n}{w_p} t\right) \left(1 - \frac{w_n}{w_p} t\right) = w_n t - w_n t^2 + w_n t - \frac{w_n^2}{w_p} t^2 = 2w_n t - w_n t^2 \left(1 + \frac{w_n}{w_p}\right)$$

Let $A = 2w_n t - w_n t^2 \left(1 + \frac{w_n}{w_p}\right)$. *We substitute the coefficients these back into the sum to find*

$$\begin{aligned}
\mathbb{V}_{n,p}^* =& (w_n - A)\|\mathbb{V}_n\|^2 + (w_p - A)\|\mathbb{V}_p\|^2 + 2A(\mathbb{V}_n \cdot \mathbb{V}_p) \\
=& w_n \|\mathbb{V}_n\|^2 + w_p \|\mathbb{V}_p\|^2 - A\left(\|\mathbb{V}_n\|^2 + \|\mathbb{V}_p\|^2 - 2\mathbb{V}_n \cdot \mathbb{V}_p\right).
\end{aligned}$$

Using the identity $\|\mathbb{V}_n - \mathbb{V}_p\|^2 = \|\mathbb{V}_n\|^2 + \|\mathbb{V}_p\|^2 - 2\mathbb{V}_n \cdot \mathbb{V}_p$ *and recognizing that* $w_n\|\mathbb{V}_n\|^2 + w_p\|\mathbb{V}_p\|^2 = \mathbb{V}_{n,p}$, *this simplifies to:*

$$\mathbb{V}_{n,p}^* = \mathbb{V}_{n,p} - A\|\mathbb{V}_n - \mathbb{V}_p\|^2;$$

$$\mathbb{V}_{n,p}^* = \mathbb{V}_{n,p} - A\|\boldsymbol{u}_n - \boldsymbol{u}_p\|^2.$$

Finally, we analyze the coefficient A:

$$A = 2w_n t - w_n t^2 \left(1 + \frac{w_n}{w_p}\right) = w_n t \left(2 - t\left(1 + \frac{w_n}{w_p}\right)\right)$$

From the earlier nodal adjustment procedure, we established that $t\left(1 + \frac{w_n}{w_p}\right) \in (0, 2)$. *Since* $t \in (0, 1)$ *and the weights are positive, it follows that* $A > 0$. *Thus, we obtain the same result as in [8]. The local variance* $\mathbb{V}^*_{p,n} < \mathbb{V}_{p,n}$ *as long as* $\boldsymbol{u}_p \neq \boldsymbol{u}_n$. *Because no other point values are affected, this guarantees that the total weighted variance* $\mathbb{V}$ *is monotonically decreasing.*

Remark 1 As in [8], we do not provide rigorous convergence results to the set $\mathcal{G}$. In [8], only one or two sweeps were needed even for robust test cases at any point when sweeping is triggered. We suspect this is due to the limited violations and smooth solution profiles in the finite difference WENO method. For oscillatory data with large magnitude or quantity of violations, we find that in some cases, $O(n_{\text{dof}})$ sweeps are needed in the numerical tests. Therefore, we note that if one views the sweeping procedure as an opinion transition system (much like in Degroot learning) we suspect that asymptotic convergence may be proven through the property of m-bounded fairness [2]. We may consider opinion dynamics theory for asymptotic consensus guarantees. Since $\bar{\mathbf{u}}$ lives in the admissible set, we may also obtain asymptotic convergence of the sweeping algorithm. Nevertheless, we save rigorous justification to future work, and provide an optional accelerated procedure to speed up convergence using multigrid strategies in Section 3.2.

We provide an accuracy proof of the scalar modification algorithm in [23] used for the density positivity preservation in Algorithm 1 specifically applied to continuous Galerkin approximations. The proof closely follows the proof of Theorem 3 provided in [20].

Theorem 1 *Let* $\boldsymbol{u}_h(x, t^n) \in V_h^k$ *satisfying Equation* (7) *be a* $k+1$ *order approximation to the exact solution* $\boldsymbol{f}(x, t_n)$ *at arbitrary time level* n, *such that* $\|\boldsymbol{u}_h - \boldsymbol{f}\|_{L^1(\Omega)} \leqslant O(h^{k+1})$. *Furthermore, we assume the exact solution* $\boldsymbol{f}$ *is sufficiently smooth such that its standard nodal interpolant,* $\mathcal{I}_h \boldsymbol{f}(x) = \sum_{i \in \mathcal{N}} \boldsymbol{f}_i \phi_i(x)$, *satisfies the standard interpolation error bound* $\|\mathcal{I}_h \boldsymbol{f} - \boldsymbol{f}\|_{L^1(\Omega)} \leqslant O(h^{k+1})$. *We evaluate* $\boldsymbol{u}_h(x, t_n) = \sum_{i \in \mathcal{N}} \boldsymbol{u}_i \phi_i(x)$. *For brevity, we drop the time-dependent notation for the degree of freedom vector* $\boldsymbol{u}$. *The density sweeping algorithm only affects the first component of* $\boldsymbol{u}$, *denoted as* ρ. *Thus we provide an accuracy estimate for* ρ. *Let the first component of the solution* $\boldsymbol{f}$ *be* f, *and assume* $f(x) > \epsilon$. *For simplicity, we let* $\epsilon = 0$. *Let* ρ^* *be the result of the density sweeping polynomial positivity preserving method in Algorithm 1 where* $\rho_h^*(x) = \sum_{i \in \mathcal{N}} \rho_i^* \phi_i(x)$. *We have the following error bound:*

$$\|\rho_h^* - f\|_{L_1} \leqslant O(h^{k+1}).$$

where $h = \max_{I \in \mathcal{I}_h} h_I$.

Proof We first recall the derivation of the weights w_i using the generalized conserved quantity as

$$\begin{aligned}\int_\Omega \mathbf{u}_h^*(x,t_n)dx &= \sum_{i\in\mathcal{N}} \mathbf{u}_i^* \sum_{j\in\mathcal{N}} \int_\Omega \phi_i(x)\phi_j(x) \\ &= \sum_{i\in\mathcal{N}} \mathbf{u}_i^* w_i.\end{aligned}$$

We may similarly write

$$\int_\Omega \mathbf{u}_h(x,t_n)dx = \sum_{i\in\mathcal{N}} \mathbf{u}_i w_i.$$

We examine the first solution component, the density ρ, from the algorithm adjustment, where f is the first component of the exact solution vector $\mathbf{f}$. Written in this form, the proof follows similarly to that in [20]. We write it out here for completeness. We denote the following subsets of nodal values where the original high order solution is negative or positive. We write

$$\mathcal{N}^- := \{i \in \mathcal{N} \mid \rho_i < 0\}$$

and

$$\mathcal{N}^+ := \{i \in \mathcal{N} \mid \rho_i \geqslant 0\}$$

We recall that conservation is maintained between the original and final solution, written as

$$\sum_{i\in\mathcal{N}} \rho_i^* w_i = \sum_{i\in\mathcal{N}} \rho_i w_i. \tag{13}$$

The updated value ρ_i^* for positive node ρ_i will not increase and is guaranteed to be corrected if needed. Therefore,

$$\rho_i^* \in [0, \rho_i], \forall i \in \mathcal{N}^+. \tag{14}$$

Furthermore, all negative values of the original solution are corrected to zero, ensuring

$$\rho_i^* = 0, \forall i \in \mathcal{N}^-. \tag{15}$$

By the triangle inequality, we have

$$\begin{aligned}\|\rho_h^* - f\|_{L^1(\Omega)} &\leqslant \|\rho_h^* - \rho_h\|_{L^1(\Omega)} + \|\rho_h - f\|_{L^1(\Omega)} \\ &\leqslant \|\rho_h^* - \rho_h\|_{L^1(\Omega)} + O(h^{k+1}).\end{aligned}$$

Thus, we need to calculate the error between the high order and updated solution. We recall that $w_i > 0, \forall i \in \mathcal{N}$. Therefore, we find

$$\begin{aligned}\|\rho_h^* - \rho_h\|_{L^1(\Omega)} &= \int_\Omega |\rho_h^*(x) - \rho_h(x)|dx \\ &= \sum_{i\in\mathcal{N}} w_i|\rho_i^* - \rho_i| \\ &= \sum_{i\in\mathcal{N}^+} w_i|\rho_i^* - \rho_i| + \sum_{i\in\mathcal{N}^-} w_i|\rho_i^* - \rho_i|.\end{aligned}$$

By (14), we know that $\rho_i^* \leqslant \rho_i$ for $i \in \mathcal{N}^+$, allowing us to remove the absolute values. Furthermore, by (15), the updated solution is exactly zero, while the high order is negative for $i \in \mathcal{N}^-$. Thus, we obtain

$$\|\rho_h^* - \rho_h\|_{L^1(\Omega)} = \sum_{i\in\mathcal{N}^+} w_i(\rho_i - \rho_i^*) - \sum_{i\in\mathcal{N}^-} w_i(\rho_i).$$

By conservation of the scheme as written in (13), we find

$$\begin{aligned}\sum_{i\in\mathcal{N}} w_i\rho_i &= \sum_{i\in\mathcal{N}} w_i\rho_i^* \\ \sum_{i\in\mathcal{N}^+} w_i\rho_i + \sum_{i\in\mathcal{N}^-} w_i\rho_i &= \sum_{i\in\mathcal{N}^+} w_i\rho_i^* \\ \implies \sum_{i\in\mathcal{N}^+} w_i(\rho_i - \rho_i^*) &= -\sum_{i\in\mathcal{N}^-} w_i(\rho_i).\end{aligned}$$

We plug in to find

$$\begin{aligned}\|\rho_h^* - \rho_h\|_{L^1(\Omega)} &= \sum_{i\in\mathcal{N}^+} w_i(\rho_i - \rho_i^*) - \sum_{i\in\mathcal{N}^-} w_i(\rho_i) \\ &= -2\sum_{i\in\mathcal{N}^-} w_i\rho_i.\end{aligned}$$

By norm equivalence on the finite-dimensional space V_h^k, there exists a constant $C > 0$ independent of h such that the discrete norm is bounded by the continuous L^1 norm of the polynomial. Thus, we use the assumption that $f(x) > 0$ to find

$$\begin{aligned}-2\sum_{i\in\mathcal{N}^-} w_i\rho_i &\leqslant 2\sum_{i\in\mathcal{N}^-} w_i(f_i - \rho_i) \leqslant 2\sum_{i\in\mathcal{N}} w_i|f_i - \rho_i| \\ &\leqslant 2C\|\mathcal{I}_h f - \rho_h\|_{L^1(\Omega)}.\end{aligned}$$

By the triangle inequality, we can split this into the interpolation error and the approximation error of the original method:

$$2C\|\mathcal{I}_h f - \rho_h\|_{L^1(\Omega)} \leqslant 2C\left(\|\mathcal{I}_h f - f\|_{L^1(\Omega)} + \|f - \rho_h\|_{L^1(\Omega)}\right).$$

Therefore, we arrive at the final bound

$$\|\rho_h^* - f\|_{L^1(\Omega)} \leqslant O(h^{k+1}). \tag{16}$$

While we do not show an accuracy proof for the nonlinear pressure function, we show optimal numerical convergence for P_1 elements in the numerical tests in Section 4. In the numerical tests, we also incorporate a symmetric sweep for the 1D Sedov blast and Woodward-Colella interacting shock problem, which divides the domain in two halves with the shared center point $c = \lfloor n_{\text{dof}}/2 \rfloor$. We alternate between an outward sweep and an inward sweep. During the outward sweep from the center c, we compute the right half from $j = c$ up to $n_{\text{dof}} - 1$ (updating each node j with respect to its rightward neighbor $j + 1$), and the left half from $j = c$ down to 2 (updating node j with respect to its leftward neighbor $j - 1$). Conversely, the inward sweep pulls updates from the boundaries back toward the center. This phase iterates the left half from $j = 1$ up to $c - 1$ (updating node j with respect to $j + 1$) and then iterates the right half from $j = n_{\text{dof}}$ down to $c+1$ (updating node j with respect to $j-1$). While this symmetric approach was not necessary in the finite difference WENO case in [8], it is helpful in preventing asymmetries from the large, initial density and pressure violations that occur in the continuous Galerkin framework. This symmetric approach is also used in the V-cycling procedure for the 1D Sedov blast and Woodward-Colella test cases. We describe the multigrid V-cycling procedure in the next section.

3.2 Multigrid V-cycling Procedure

We find that the iterative pressure-sweeping algorithm in Procedure 1 converges in only a few forward and backward sweeps on smooth solutions, or when the number of negative pressure degrees of freedom and the magnitude of the violation are small. This behavior is consistent with [8], where even in challenging test cases only one or two sweeps are needed with fifth-order finite-difference WENO discretizations. For our implicit continuous finite-element tests, we observe similarly rapid convergence in the shock diffraction test and the vortex evolution accuracy test in Section 4. However, for more difficult problems in which oscillations pollute the grid with many negative points or large violations, many sweeps may be required to restore positivity.

A representative example is the Sedov blast wave in one and two dimensions. Because the initial data concentrate most of the positive pressure in a few cells, the first few time steps produce many negative pressure points across the grid, and substantial exchange between distant nodes is needed before the violations are eliminated. For this reason, we incorporate multigrid techniques into the sweeping protocol to accelerate convergence of the positivity correction.

The motivation for a multigrid approach is that the limiter reduces local defects in a way that is analogous to classical relaxation methods such

as Gauss-Seidel. Consequently, defects with a large spatial profile typically require more iterations to eliminate than localized defects. Coarse grids are attractive because the same correction strategy can be carried out at substantially lower cost [30]. In this sense, the flat sweep is well suited to local, high-frequency error reduction, while low-frequency errors appear as higher-frequency components on the coarse grid [30].

Multi-grid methods were originally designed to increase the efficiency of numerical discretizations of elliptic and hyperbolic differential equations [31]. As in classical multi-grid, our accelerated sweeping procedure works by alternating corrections on coarse and fine grids. Following the notation in [31] and [30] and many classic multi-grid applications, we provide a restriction algorithm, a prolongation algorithm, and a V-cycling protocol which is incorporated into the accelerated sweeping procedure. We will later show that all operations ensure conservation. Since the pressure function is nonlinear, we follow the full approximation scheme (FAS) protocol for prolongation [30]. Due to the effectiveness of the flat passes in addressing local violations, we do not apply the sweeping procedure between coarsening events in the descent phase. Instead, we rely on the course-grid sweep to make more global corrections, and apply sweeping between each prolongation in the ascent phase of the V-cycle.

To track coarsening levels during the V-cycle procedure, we define the set of nodal indices at coarsening level ℓ with $n_{\text{dof}}^{(\ell)}$ nodal degrees of freedom as

$$\mathcal{N}^{(\ell)} = \{1, 2, \ldots, n_{\text{dof}}^{(\ell)}\}.$$

Here, the superscript (ℓ) should not be confused by a time-level or stage, as both Procedure 1 and Procedure 2 are fully isolated to whichever stage or time level in which a violation triggers the sweeping limiter. We refer to the "flat sweep" as the sweep applied to the finest collection of points i.e. with the full grid resolution $\mathbf{u}^{(0)} \equiv \mathbf{u} = \{\mathbf{u}\}_{j=1}^{n_{\text{dof}}}$ with associated weights $w^{(0)} \equiv w = \{w\}_{j=1}^{n_{\text{dof}}}$. To move from grid level (ℓ) to $(\ell+1)$ via coarsening in one dimension, we coarsen consecutive pairs. The simplest approach is to form indexing blocks for each coarse node, written $\pi_k^{(\ell+1)} = (2k-1, 2k)$ for $k = 1, ..., n_{\text{dof}}^{(\ell+1)}$, so that $\pi_1^{(\ell+1)} = (1, 2), \pi_2^{(\ell+1)} = (3, 4)$, and so on. To prevent biasing, particularly for odd grids where the final node will be a singleton, we alternative the setting of the first cluster at every V-cycling call by setting the first node as a singleton, and the remaining nodes as pairs formed by $\pi_k^1 = (2k-2, 2k-1)$, so that $\pi_1^{(1)} = (1), \pi_2^{(1)} = (2, 3)$, and so on. The restriction procedure for coarsening to obtain $\mathbf{u}^{(\ell+1)}$ and $w^{(\ell+1)}$ is described in Algorithm 5. The procedure produces coarse nodes using the weighted combination of fine nodes as to ensure conservation. The remaining V-cycling procedure in Algorithm 4 is described as follows.

The V-cycling procedure restricts down to a user-defined level, as decided by L_{max}. In our research, we compute a maximum coarsening level until the coarsest level contains less than or equal to eight nodes. Once the coarsest grid is obtained, we apply sweeping n_{coarse} times. After that, the adjustment is consecutively passed up to finer levels through the prolong procedure in Al-

gorithm 6, and sweeping is applied n_{fine} times after each ascent. We choose $n_{coarse} = 15$ and $n_{fine} = 3$, but these are both tuneable parameters. Procedure 2 describes the incorporation of V-cycling in an accelerated sweeping procedure. The density sweep remains unaffected. The pressure sweep is modified so that after every K_{max} flat sweeps, a V-cycle procedure is triggered. We choose $K_{max} = 10$ in our numerical tests. The description of the accelerated procedure and the incorporated algorithms are given below.

Procedure 2. (An accelerated, conservative positivity-preserving sweeping procedure)
$\epsilon \leftarrow 10^{-12}$

1. Apply Algorithm 1 to $\{\rho_j\}_{j=1}^{n_{\text{dof}}}$:
   $\rho \leftarrow DensitySweep(\rho, w, \epsilon)$

2. Apply the positivity preserving sweep in Algorithm 2 using the nodal adjustment in Algorithm 3:

```
while min(p(u)) < ε do
    for k = 1, ..., K_max do
        u ← PressureSweep(u, w, ε)
    end for
    if min(p(u)) < ε then
        u ← VCycle(u, w)
    end if
end while
```

Algorithm 4: $\mathbf{u}^{(0),*} = \text{VCycle}(\mathbf{u}^{(0)}, w^{(0)})$

```
for ℓ = 0, ..., L_max − 1 do                          ▷ Phase 1: Descent
    u^(ℓ+1), w^(ℓ+1) ← Restrict(u^(ℓ), w^(ℓ))
end for
u^(L_max),* ← u^(L_max)
for k = 1, ..., n_coarse do                           ▷ Phase 2: Coarsest Sweep
    u^(L_max),* ← PressureSweep(u^(L_max),*, w^(L_max), ε)
end for
for ℓ = L_max − 1 ... 0 do                            ▷ Phase 3: Ascent
    u^(ℓ) ← Prolong(u^(ℓ), u^(ℓ+1), u^(ℓ+1),*)
    u^(ℓ),* ← u^(ℓ)
    for k = 1, ..., n_fine do
        u^(ℓ),* ← PressureSweep(u^(ℓ),*, w^(ℓ), ε)
    end for
end for
```

Algorithm 5: $\mathbf{u}^{(\ell+1)}, w^{(\ell+1)} = \textsc{Restrict}(\mathbf{u}^{(\ell)}, w^{(\ell)})$

for $k = 1, ..., n_{\text{dof}}^{\ell+1}$ **do**
  $w_k^{(\ell+1)} \leftarrow \sum_{j \in \pi_k} w_j^{(\ell)}$
  $\mathbf{u}_k^{(\ell+1)} \leftarrow \frac{1}{w_k^{(\ell+1)}} \sum_{j \in \pi_k^{(\ell+1)}} w_j^{(\ell)} \mathbf{u}_j^{(\ell)}$
end for

Algorithm 6: $\mathbf{u}^{(\ell)} = \textsc{Prolong}(\mathbf{u}^{(\ell)}, \mathbf{u}^{(\ell+1)}, \mathbf{u}^{(\ell+1),*})$

for $k = 1, ..., n_{\text{dof}}^{\ell+1}$ **do**
  $\delta_k \leftarrow \mathbf{u}_k^{*(\ell+1)} - \mathbf{u}_k^{(\ell+1)}$
  for each $j \in \pi_k^{(\ell+1)}$ **do**
    $\tilde{\mathbf{u}}_j^{(\ell)} \leftarrow \mathbf{u}_j^{(\ell)} + \delta_k$
    $\tilde{\rho}_j^{(\ell)} \leftarrow \rho_j^{(\ell)} \left(\frac{\rho_k^{(\ell+1),*}}{\rho_k^{(\ell+1)}} \right)$
    $\mathbf{u}_j^{(\ell)} \leftarrow \tilde{\mathbf{u}}_j^{(\ell)}$
  end for
end for

We show that the V-cycling protocol maintains conservation of mass, momentum, and energy.

Property 4 (V-Cycle Conservation)**.** *We first show that the restriction in Algorithm 5 is globally conservative across coarsening levels. Since the set of all coarse node indexing arrays* $\{\pi^{\ell+1}\}$ *partitions* $\mathcal{N}^{\ell}$*, we find that*

$$\begin{aligned}
\sum_{k=1}^{n_{dof}^{(\ell+1)}} w_k^{(\ell+1)} \boldsymbol{u}_k^{(\ell+1)} &= \sum_{k=1}^{n_{dof}^{(\ell+1)}} w_k^{(\ell+1)} \frac{1}{w_k^{(\ell+1)}} \sum_{j \in \pi_k} w_j^{(\ell)} u_j^{(\ell)} \\
&= \sum_{k=1}^{n_{dof}^{(\ell+1)}} \sum_{j \in \pi_k} w_j^{(\ell)} u_j^{(\ell)} \\
&= \sum_{j=1}^{n_{dof}^{(\ell)}} w_j^{(\ell)} \boldsymbol{u}_j^{(\ell)}.
\end{aligned}$$

Thus, $\bar{\boldsymbol{u}}$ *remains the same at any level of coarsening. Next, we show that the additional sweeping and prolongation steps preserve the global mass. We first recall that at arbitrary level* $(\ell + 1)$*, conservation of sweeping ensures*

$$\sum_{k=1}^{n_{dof}^{(\ell+1)}} w_k^{(\ell+1)} \boldsymbol{u}_k^{(\ell+1)} = \sum_{k=1}^{n_{dof}^{(\ell+1)}} w_k^{(\ell+1)} \boldsymbol{u}_k^{(\ell+1),*},$$

which further implies

$$0 = \sum_{k=1}^{n_{dof}^{(\ell+1)}} w_k^{(\ell+1)}(\boldsymbol{u}_k^{(\ell+1)} - \boldsymbol{u}_k^{(\ell+1),*}) = \sum_{k=1}^{n_{dof}^{(\ell+1)}} w_k^{(\ell+1)}\delta_k.$$

Using this result, show that the additive prolongation step used for momentum and energy conserves each update. We find

$$\begin{aligned}
\sum_{j=1}^{n_{dof}^{(\ell)}} w_j^{(\ell)}\tilde{\boldsymbol{u}}_j^{(\ell)} &= \sum_{k=1}^{n_{dof}^{(\ell+1)}} \sum_{j\in\pi_k} w_j^{(\ell)}\tilde{\boldsymbol{u}}_j^{(\ell)} \\
&= \sum_{k=1}^{n_{dof}^{(\ell+1)}} \sum_{j\in\pi_k} w_j^{(\ell)}(\boldsymbol{u}_j^{(\ell)} + \delta_k) \\
&= \sum_{k=1}^{n_{dof}^{(\ell+1)}} \sum_{j\in\pi_k} w_j^{(\ell)}\boldsymbol{u}_j^{(\ell)} + \sum_{k=1}^{n_{dof}^{(\ell+1)}} \delta_k \sum_{j\in\pi_k} w_j^{(\ell)} \\
&= \sum_{k=1}^{n_{dof}^{(\ell+1)}} \sum_{j\in\pi_k} w_j^{(\ell)}\boldsymbol{u}_j^{(\ell)} + \sum_{k=1}^{n_{dof}^{(\ell+1)}} \delta_k w_k^{(\ell+1)} \\
&= \sum_{k=1}^{n_{dof}^{(\ell+1)}} \sum_{j\in\pi_k} w_j^{(\ell)}\boldsymbol{u}_j^{(\ell)} + 0 \\
&= \sum_{j=1}^{n_{dof}^{(\ell)}} w_j^{(\ell)}\boldsymbol{u}_j^{(\ell)}.
\end{aligned}$$

We show that the multiplicative update for ρ to ensure positivity is also conservative i.e. $\tilde{\rho}_j^{(\ell)} = \rho_j^{(\ell)}\left(\frac{\rho_k^{(\ell+1),*}}{\rho_k^{(\ell+1)}}\right)$. Noting from the restriction formula that $\sum_{j\in\pi_k} w_j^{(\ell)}\rho_j^{(\ell)} = w_k^{(\ell+1)}\rho_k^{(\ell+1)}$, the global density is conserved:

$$\begin{aligned}
\sum_{j=1}^{n_{dof}^{(\ell)}} w_j^{(\ell)} \tilde{\rho}_j^{(\ell)} &= \sum_{k=1}^{n_{dof}^{(\ell+1)}} \sum_{j \in \pi_k} w_j^{(\ell)} \rho_j^{(\ell)} \left(\frac{\rho_k^{(\ell+1),*}}{\rho_k^{(\ell+1)}} \right) \\
&= \sum_{k=1}^{n_{dof}^{(\ell+1)}} \left(\frac{\rho_k^{(\ell+1),*}}{\rho_k^{(\ell+1)}} \right) \sum_{j \in \pi_k} w_j^{(\ell)} \rho_j^{(\ell)} \\
&= \sum_{k=1}^{n_{dof}^{(\ell+1)}} \left(\frac{\rho_k^{(\ell+1),*}}{\rho_k^{(\ell+1)}} \right) w_k^{(\ell+1)} \rho_k^{(\ell+1)} \\
&= \sum_{k=1}^{n_{dof}^{(\ell+1)}} w_k^{(\ell+1)} \rho_k^{(\ell+1),*} \\
&= \sum_{j=1}^{n_{dof}^{(\ell)}} w_j^{(\ell)} \rho_j^{(\ell)}.
\end{aligned}$$

Therefore, $\bar{\boldsymbol{u}}$ is unchanged through each stage of the V-cycling procedure, and $p(\bar{\boldsymbol{u}}) > \epsilon$.

We note that beyond conservation, the V-cycling protocol does not inherent the properties as the flat sweep such as monotonic improvement and variance reduction due to the redistribution of violations in the grid. Nevertheless, we find that for particularly for blast problems, even though the V-cycling procedure may at some points decrease the average or minimum pressure, it ultimately improves the convergence in our numerical tests. Furthermore, we emphasize that the accelerated procedure is an optional, experimental, and highly tuneable feature of the sweeping framework. In well-behaved numerical cases or smooth problems that require few sweeps, the accelerated Procedure 2 performs exactly as Procedure 1 due to the initial flat sweeps which correct all violations before any V-cycling is performed.

3.3 Extension to multiple dimensions

In two spatial dimensions, the compressible Euler equations in conservative form is written as

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}(\mathbf{u})}{\partial x} + \frac{\partial \mathbf{g}(\mathbf{u})}{\partial y} = 0, (x, y) \in \mathbb{R}^2, t \geqslant 0, \tag{17}$$

$$\mathbf{u} = \begin{pmatrix} \rho \\ m \\ n \\ E \end{pmatrix}, \qquad \mathbf{f}(\mathbf{u}) = \begin{pmatrix} m \\ \frac{m^2}{\rho} + p \\ \frac{mn}{\rho} \\ \frac{m}{\rho}(E+p) \end{pmatrix}, \qquad \mathbf{g}(\mathbf{u}) = \begin{pmatrix} n \\ \frac{mn}{\rho} \\ \frac{n^2}{\rho} + p \\ \frac{n}{\rho}(E+p) \end{pmatrix}, \tag{18}$$

where ρ is the density, $m = \rho u$ and $n = \rho v$ are the momentum components in the x- and y-directions, respectively, E is the total energy, and pressure is determined from the ideal-gas law

$$p = (\gamma - 1)\left(E - \tfrac{1}{2}\frac{m^2 + n^2}{\rho}\right),$$

with $\gamma > 1$ describing the ratio of the specific heats. For $\rho > 0$, one can easily verify that p is a concave function as in the one-dimensional case. The convex, physically admissible set is

$$\mathcal{G} = \left\{ \mathbf{u} = \begin{pmatrix} \rho \\ m \\ n \\ E \end{pmatrix} \middle| \rho > \epsilon \quad \text{and} \quad p = (\gamma - 1)\left(E - \tfrac{1}{2}\frac{m^2 + n^2}{\rho}\right) > \epsilon \right\}. \tag{19}$$

For a two-dimensional grid, we will consider P_1 and P_2 triangular elements. The same procedure is easily extended to Q_1 and Q_2 quadrilateral elements. To extend the previously described framework to two-dimensional flows, we adopt a straightforward geometric discretization.

The 2D spatial domain $\Omega \subset \mathbb{R}^2$ is partitioned into a uniform Cartesian grid, where each quadrilateral cell is split along its diagonal to form a structured mesh of triangular elements, denoted $\mathcal{T}_h$. We note the uniform grid is not strictly necessary for any of the methods described so far, and all approaches can be extended to unstructured grids. The finite element spaces $\mathbf{V}_h$ and $\mathbf{W}_h$ are naturally extended to continuous, piecewise polynomials over these triangles. The primary difference in the two-dimensional formulation is the expansion of the flux and gradient operators. We obtain the following time-derivative and flux divergence form the multi-dimensional strong residual for stage i:

$$r^{(i)}_{\text{strong}} = \frac{\mathbf{u}_h^{(i)} - \mathbf{u}_h^n}{\Delta t} + \sum_{j=1}^{i} \alpha_{ij} \left(\mathbf{A}_x(\mathbf{u}_h^{(j)}) \frac{\partial \mathbf{u}_h^{(j)}}{\partial x} + \mathbf{A}_y(\mathbf{u}_h^{(j)}) \frac{\partial \mathbf{u}_h^{(j)}}{\partial y} \right).$$

Because the underlying time-stepping and stabilization principles are the same as in the the one-dimensional case described in Section 2.2, we write the comprehensive weak form including the continuous Galerkin terms, the SUPG stabilization, and the artificial viscosity in a single updated residual equation:

$$
\begin{aligned}
\mathcal{R}_i(\mathbf{u}_h^{(i)}) = &\int_\Omega \mathbf{w}_h \cdot \left(\frac{\mathbf{u}_h^{(i)} - \mathbf{u}_h^n}{\Delta t} \right) d\Omega \\
&- \sum_{j=1}^{i} \alpha_{ij} \int_\Omega \left(\frac{\partial \mathbf{w}_h}{\partial x} \cdot \mathbf{F}_x(\mathbf{u}_h^{(j)}) + \frac{\partial \mathbf{w}_h}{\partial y} \cdot \mathbf{F}_y(\mathbf{u}_h^{(j)}) \right) d\Omega \\
&+ \sum_{j=1}^{i} \alpha_{ij} \int_{\partial\Omega} \mathbf{w}_h \cdot \left(\mathbf{F}_x(\mathbf{u}_h^{(j)}) n_x + \mathbf{F}_y(\mathbf{u}_h^{(j)}) n_y \right) ds \\
&+ \int_\Omega \left[\tau \left(\mathbf{A}_x^T \frac{\partial \mathbf{w}_h}{\partial x} + \mathbf{A}_y^T \frac{\partial \mathbf{w}_h}{\partial y} \right) \right] \cdot r_{\text{strong}}^{(i)} \, d\Omega \\
&+ \int_\Omega \nu \left(\frac{\partial \mathbf{w}_h}{\partial x} \cdot \frac{\partial \mathbf{u}_h^{(i)}}{\partial x} + \frac{\partial \mathbf{w}_h}{\partial y} \cdot \frac{\partial \mathbf{u}_h^{(i)}}{\partial y} \right) d\Omega = 0,
\end{aligned}
$$

where $\hat{\mathbf{n}} = (n_x, n_y)$ is the outward facing normal vector. As in the one-dimensional formulation, the multi-dimensional SUPG perturbation is applied to the strong residual. The shock-capturing parameter ν acts as a scalar diffusion coefficient. The pure SUPG treatment (without shock capturing) is found by setting $\nu = 0$. Otherwise, ν is evaluated locally using either the conservative YZ-β sensor with $\nu_{YZ\beta}$ (computing the non-dimensional residual and gradient norms over the 2D elements) or the entropy-variable artificial viscosity formulation ν_d as derived in Section 2.3 with the multidimensional form given in [21].

The extension of the sweeping limiter to multiple dimensions is simple, and is similar to that of [8]. We generate indexing arrays as in [23], except we introduce additional deviations to the sweeping direction in order to reduce sweeping biases potentially introduced with the increased number of sweeps. We use the same one-dimensional sweeping algorithm as described in Procedure 1 to treat the 2D grid as described below.

Let the degrees of freedom be indexed by their logical coordinates (i, j) on a structured geometric grid, where $i = 1, \ldots, n_x$ corresponds to the x-direction and $j = 1, \ldots, n_y$ corresponds to the y-direction. We define four forward sweeping trajectories. For each forward path, a corresponding backward sweep is defined by reversing the sequence of nodes. This yields a total of eight directional sweeps:

- Sweep 1 (Row-Aligned Forward): For each row $j = 1, \ldots, n_y$, the horizontal index i is traversed as $i = 1, 2, \ldots, n_x$ if j is odd, and in reverse order $i = n_x, n_x - 1, \ldots, 1$ if j is even.
- Sweep 2 (Row-Aligned Backward): The reverse of Sweep 1.
- Sweep 3 (Column-Aligned Forward): For each column $i = 1, \ldots, n_x$, the vertical index j is traversed as $j = 1, 2, \ldots, n_y$ if i is odd, and in reverse order $j = n_y, n_y - 1, \ldots, 1$ if i is even.

- Sweep 4 (Column-Aligned Backward): The reverse of Sweep 3.
- Sweep 5 (Diagonal Forward): The grid is traversed along diagonals defined by the constant sum $k = i + j$. For $k = 2, \ldots, n_x + n_y$, the nodes within each diagonal group are traversed from bottom-right to top-left for odd k, and top-left to bottom-right for even k.
- Sweep 6 (Diagonal Backward): The reverse of Sweep 5.
- Sweep 7 (Anti-Diagonal Forward): The grid is traversed along anti-diagonals defined by the constant difference $k = i - j$. For $k = 1 - n_y, \ldots, n_x - 1$, the nodes within each anti-diagonal group are traversed alternatingly from bottom-left to top-right, and top-right to bottom-left, depending on the parity of k.
- Sweep 8 (Anti-Diagonal Backward): The reverse of Sweep 7.

As mentioned in 3.1, P_2 remains a special case due to the zero-weight vertex nodes. In this case, we apply the density and pressure treatment as described in Procedure 1 only to the midpoint nodes with nonzero weight. Then, at each vertex node, we simply clip the density to ϵ and use the scaling limiter to provide a guaranteed correction of pressure with a neighboring node in the direction of sweep. In other words, if $\mathbf{u}_1$ is a vertex node where $p(\mathbf{u}_1) < \epsilon$, we are guaranteed that its adjacent midpoint node $\mathbf{u}_2$ has $p(\mathbf{u}_2) > \epsilon$, and $\rho_1, \rho_2 > \epsilon$, because all of these quantities have been treated by the sweep. Therefore, we set

$$t = \left(\frac{p(\mathbf{u}_1) - \epsilon}{p(\mathbf{u}_1) - p(\mathbf{u}_2)} \right),$$

and update the updated vertex node

$$\mathbf{u}_1^* = (1 - t)\mathbf{u}_1 + t\mathbf{u}_2,$$

which guarantees $p(\mathbf{u}_1^*) > \epsilon$. After sweeping the midpoint nodes, we obtain a conservative update for all vertex nodes in a single pass for both density and pressure. We emphasize that this does not break conservation only because the weight associated to the vertex node in zero and does not contribute to the globally conserved quantity $\bar{\mathbf{u}}$.

Lastly, we describe the multi-dimensional extension to the accelerated sweeping algorithm in Procedure 2. In two dimensions, all nested algorithms and conservative guarantees immediately follow the 1D approach described in Section 3.2. To partition the 2D grid, we now group nodes into 2×2 spatial blocks instead of consecutive 1D pairs. We define the discrete sets of fine-grid indices belonging to the k-th coarse block at level $\ell+1$ as $\pi_k^{(\ell+1)}$. While the 1D approach prevents grid biasing by alternating the first node as a singleton, the 2D approach achieves this by cycling through a two-dimensional anchor offset $(d_x, d_y) \in \{(0,0), (1,0), (0,1), (1,1)\}$ at the initialization of each V-cycle.

4 Numerical results

In this section, we provide numerical results demonstrating the capacity of the sweeping limiter to conservatively maintain the positivity of density and pressure for many tough test cases. We emphasize that our focus is not on the underlying spatial scheme. Thus, our results may appear to be overly diffusive or oscillatory in comparison to higher order or more stable methods. In fact, more refinement in terms of parameter tuning for the YZ-β stabilization and SUPG τ parameter would likely yield less oscillatory results. Nevertheless, we show that even with massive unphysical values, the sweeping limiter successfully corrects the negative density and pressure and prevents numerical blow-up.

We provide results in both one- and two-dimensions. Our results mirror those in [8] and [35], with a few adjustments where negative values were not observed while other benchmark tests stressed the solver sufficiently. While implicit time stepping theoretically permits high CFL numbers, maintaining stability at those limits requires heavy artificial viscosity that excessively smears the solution. We deliberately use baseline YZ-β for most test cases at a lower CFL to preserve sharp gradients, allowing us to demonstrate that our sweeping positivity-preserving limiter can correct non-physical negative values without distorting the shock profile. In two dimensions, high-resolution results are omitted due to the computationally prohibitive cost of the Newton solve.

We implement the code using the open-source finite element framework FEniCS with the DOLFINx Python toolbox. LLM assistance (Anthropic's Claude and Google's Gemini) was used to expedite coding tasks, automate the various statistics generated in the numerical tests, and explore existing literature (including the introduction to multigrid approaches). We verified all code against the cited literature and manually checked for accuracy. For the Newton solver, we used the PETSc library. For the sweeping and V-cycling implementation, we used Numba @njit compiler to prevent slow array accessing in Python for-loops.

To emphasize the limiter's ability to make large corrections, we include a summary table with the magnitude and quantity of non-physical violations prior to correction in Tables 2 and 6. We record the extreme values of the minimum nodal pressure, $P_{\min}$, at the moment a sweep is initiated. We report the minimum, maximum, and average (min, max, and avg) of $P_{\min}$ in terms of how negative the value is before correction. We track the number of "bad" degrees of freedom, denoted as n_{bad}, which represents the discrete number of nodes (including midpoints for P_2) requiring correction during a given event. We provide the maximum and average n_{bad} to illustrate the spatial footprint of the non-physical states encountered over the course of the simulation. We also provide the total number of degrees of freedom (DOF) for scale comparison.

We then report the sweeping statistics in Tables 3 and 7. A sweep event is triggered whenever the algorithm detects non-physical states, such as negative pressures or densities, during a Runge-Kutta stage. We report the number

of sweep events and the total flat sweep passes executed across the entire time-evolution, as well as the minimum, maximum, and average number of passes required to resolve a single event. We track the total number of density sweeps, and remind the reader that only a single forward and backward sweep is needed for a triggered density correction. The V-cycle statistics document the performance of the solver during these corrections. We report the number of V-cycle events, the total V-cycles executed throughout the simulation, and the maximum and average V-cycles utilized per individual event. We also include general time-stepping metrics, detailing the total number of global time steps and Runge-Kutta stages evaluated.

To show the challenges of the Newton iteration and need for adaptive time-stepping, we include a time-stepping plot for each case in Figures 4 and 9. We show the fraction of the CFL used for the total time evolution, including reasons for time-step halving or increase. We include a set of reasons for the adjustment in time step factor. We use N to signify Newton divergence halving, S to signify a sweep not correcting all negative values within a set maximum number of sweeps, and R to signify the ramp recovery. We set the maximum number of sweeps as $2n_{el}$ in 1D, and $2(n_{el}^x \times n_{el}^y)$ in 2D, where n_{el}^x and n_{el}^y are the number of cells in the x and y direction respectively.

In every test, the scheme without the sweeping limiter crashes due to unphysical pressure and density values. We find that the sweeping limiter recovers the simulation and gives comparable results to literature for most test cases. For the Leblanc problem, we include a comparison with and without V-cycling. For that case, the large magnitude initial violations lead to drift in velocity when V-cycling is enabled, as shown in Figure 3. Thus, an alternative approach which improves results for that case is to disable the V-cycling and allow the adaptive time stepping to reduce Δt. The only other case with V-cycling artifacts is the Mach 2000 test case as shown in Figure 7. We note that these are the only two cases where the maximum number of sweeps is reached, and we conjecture that both the visible artifacts and the Δt halving are due to the large magnitude violations as shown in Tables 2 and 6. For the other tests, we follow the accelerated sweeping procedure (Procedure 2) and find that V-cycling improves robustness and reduces the number of sweeps without causing numerical artifacts.

4.1 1D Results

Example 1 (1D Sedov blast) We consider the one-dimensional Sedov blast wave which is an extreme low-density and low-pressure test case. The initial state consists of a uniform ambient gas at rest with a near-zero background energy and a point-source energy spike at the origin. To accommodate different finite-element polynomial spaces (e.g., P_1 vs. P_2), we scale the energy injection by the finite-element lumped-mass weight, w_0, of the center node. On the domain $[-2, 2]$, the initial condition is

$$\rho(x, 0) = 1, \quad u(x, 0) = 0, \text{ and}$$

$$E(x, 0) = \frac{10^{-9}}{\gamma - 1} + \begin{cases} \frac{3.2 \times 10^6}{w_0} & \text{if } x = 0 \\ 0 & \text{otherwise.} \end{cases}$$

We apply outflow boundary conditions at both ends of the domain, and run the simulation until a final time of $T = 0.001$ with $n_{el} = 800$ cells. We include P_1 and P_2 results with YZ-β stabilization and CFL $= 1.0$. As the results indicate, the numerical solution captures the blast profile of the exact solution while correcting the the large magnitude unphysical values in Table 2.

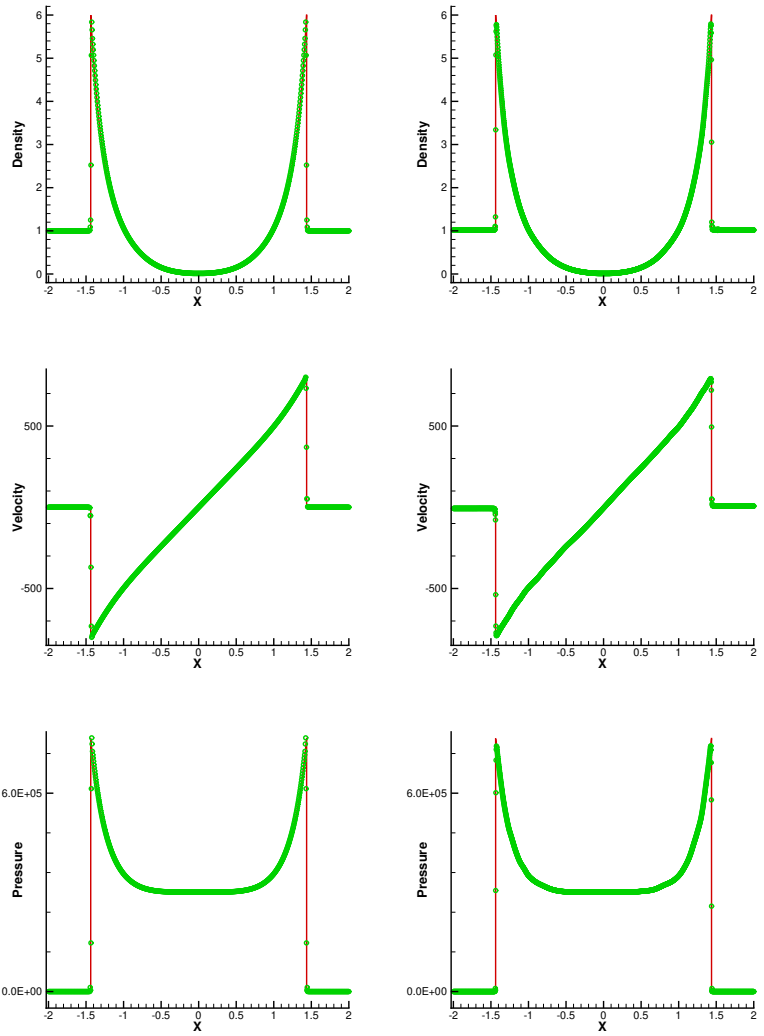


Fig. 1 Example 1: Numerical results 1D Sedov blast test case using with YZ-β shock capturing and sweeping limiter using P_1 (left) and P_2 (right) elements. Green symbols are approximate results. Red line is exact solution.

Example 2 (Woodward-Colella Interacting Shock Blasts) We simulate the classic Woodward-Colella interacting blast waves on the domain $[0, 1]$ as described in [32]. The gas is initially at rest with uniform unit density, but is divided into three pressure zones separated by sharp discontinuities, leading to strong interacting shocks. The initial state is defined as:

$$\rho(x, 0) = 1, \quad u(x, 0) = 0, \text{ and}$$
$$p(x, 0) = \begin{cases} 1000 & \text{if } 0 \leqslant x \leqslant 0.1 \\ 0.01 & \text{if } 0.1 < x < 0.9 \\ 100 & \text{if } 0.9 \leqslant x \leqslant 1. \end{cases}$$

The boundaries at $x = 0$ and $x = 1$ are reflective. We run to a final time of $T = 0.038$ with $n_{el} = 7010$ cells and CFL $= 1.0$. We include P_1 and P_2 results using YZ-β shock capturing for the stabilization. We see that both results are comparable to the reference solution in [32], although insufficient stabilization yields some spurious oscillations in the solution.

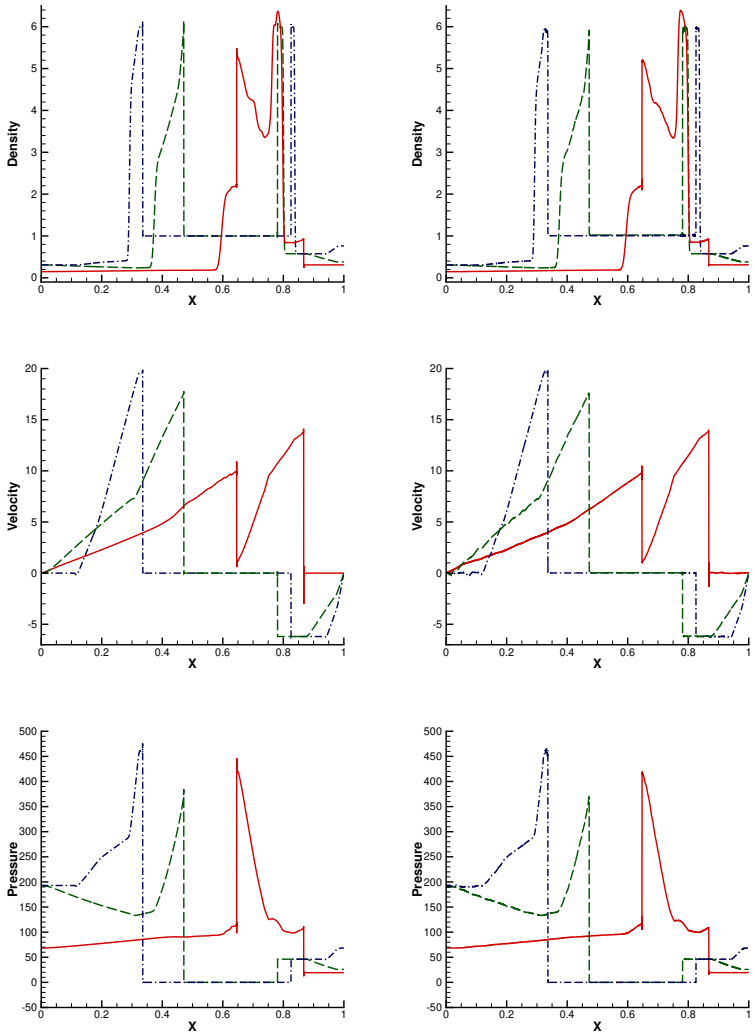


Fig. 2 Example 2: Numerical results for Woodward Colella interacting shock blast case with YZ-β shock capturing and sweeping limiter using P_1 (left) and P_2 (right) elements. Blue dash-dotted line is $T = 0.01$, green dashed line is $T = 0.016$, red solid line is $T = 0.038$.

Example 3 (Leblanc Problem) The Leblanc shock tube is a severe Riemann problem designed to test algorithmic robustness across extreme contact discontinuities and immense pressure ratios. The domain $[0, 20]$ is initialized with a large jump in both density and pressure at $x = 10$:

$$(\rho, u, p) = \begin{cases} (2.0, 0, 10^9) & \text{if } x \leqslant 10 \\ (0.001, 0, 1.0) & \text{if } x > 10. \end{cases}$$

The simulation is run to a final time of $T = 0.0001$ with $n_{el} = 1600$ for P_1 using YZ-β shock capturing and the sweeping limiter with and without V-cycling. We compare with and without V-cycling (flat sweep) to demonstrate that while the V-cycling procedure is beneficial in most cases and helps to correct large magnitude or quantity of errors, it can lead to numerical artifacts when large corrections are used on the highly coarsened grid. Here, we coarsen to the maximum level with less than eight nodes. We suspect that a more moderate coarsening choice could provide a balance between the two results, but save this exploration for future work.

We do not include a P_2 result due to issues with Newton convergence for $n_{el} = 1600$ at the first time step. We note that with $n_{el} = 1601$ points, we obtain a similar result to Figure 3 for P_2 using the V-cycling procedure. With the flat sweep alone for P_2 with $n_{el} = 1601$ points, the adaptive time stepping procedure fails with the maximum of 20 Δt halves, due to a combination of Newton convergence issues and the magnitude of errors being so large that the flat sweep alone does not correct within the set maximum number of sweeps. Thus, we suspect that the underlying stabilization is not sufficient for this problem due to the magnitude of unphysical values seen in Table 2.

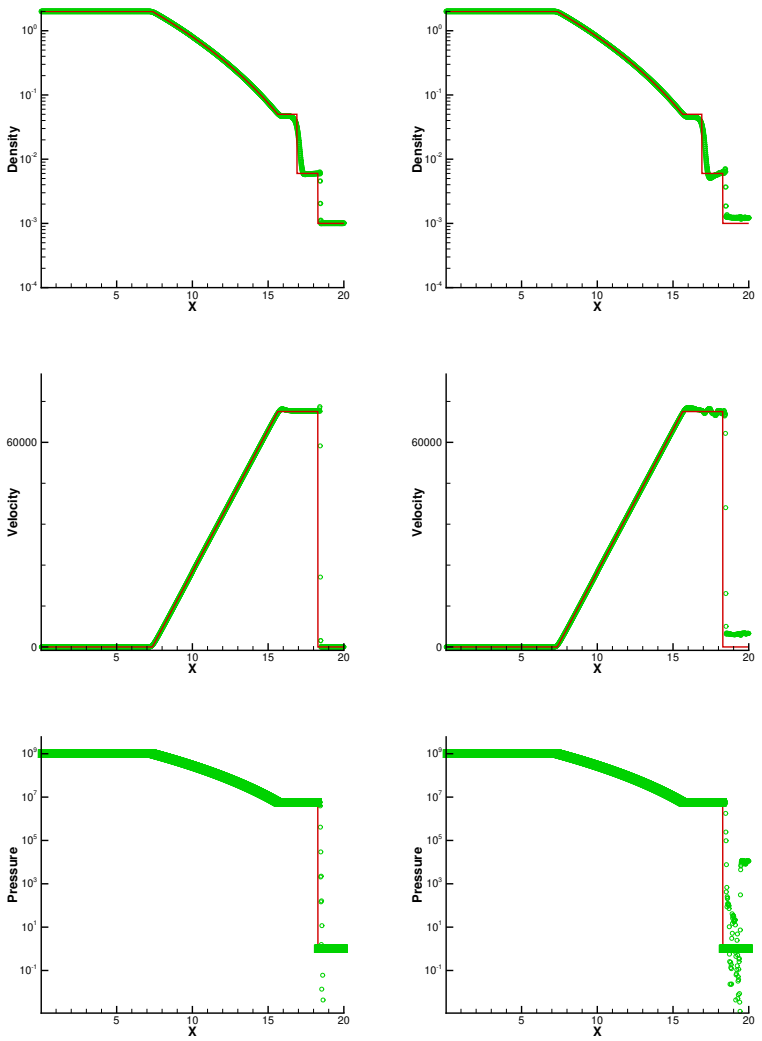


Fig. 3 Example 3: Numerical results for Leblanc problem using P_1 SUPG and YZ-β shock capturing with sweeping limiter with flat sweep (left) and V-cycling protocol (right). Green symbols are approximate results. Red line is exact solution.

Here, we provide a summary tables and plots for the 1D numerical runs.

Table 2 1D Correction Data

		1D Sedov P_1	1D Sedov P_2	Woodward P_1	Woodward P_2
Number	DOF	801	1,601	7,011	14,021
$P_{\min}$	Min	-1.70e-02	-2.26e-08	-7.90e-07	-2.39e-11
	Max	-2.08e+07	-1.39e+08	-9.33e+01	-1.97e+02
	Avg	-7.97e+04	-1.25e+05	-1.45e+01	-1.27e+01
n_{bad}	Max	30	671	837	1287
	Avg	25.1	88.0	50.3	149.9

		Leblanc V-Cycle	Leblanc Flat Sweep
Number	DOF	1,601	1,601
$P_{\min}$	Min	-8.89e-03	-4.28e-09
	Max	-4.10e+06	-4.10e+06
	Avg	-3.46e+05	-2.39e+05
n_{bad}	Max	390	361
	Avg	50.9	47.0

Table 3 1D Sweep Data

		1D Sedov P_1	1D Sedov P_2	Woodward P_1	Woodward P_2
Number	Timestep	1282	2632	11178	22393
	RK Stage	3846	7896	33534	67179
Sweep	Events	2682	7907	25725	52858
	Total	2682	144334	72253	1038622
	Min	1	1	1	1
	Max	1	151	43	101
	Avg	1.00	18.25	2.81	19.65
V-cycle	Events	0/2682	3448/7907	540/25725	29648/52858
	Total	0	12751	2250	95265
	Max	0	15	4	10
	Avg	0.00	1.61	0.09	1.80
Density	Total	2682	7907	25725	52858

		Leblanc V-Cycle	Leblanc Flat Sweep
Number	Timestep	891	971
	RK Stage	2673	2913
Sweep	Events	2700	3095
	Total	29044	72707
	Min	1	1
	Max	3200	3160
	Avg	10.76	23.49
V-cycle	Events	426/2700	0/3095
	Total	2302	0
	Max	319	0
	Avg	0.85	0.00
Density	Total	2700	3095

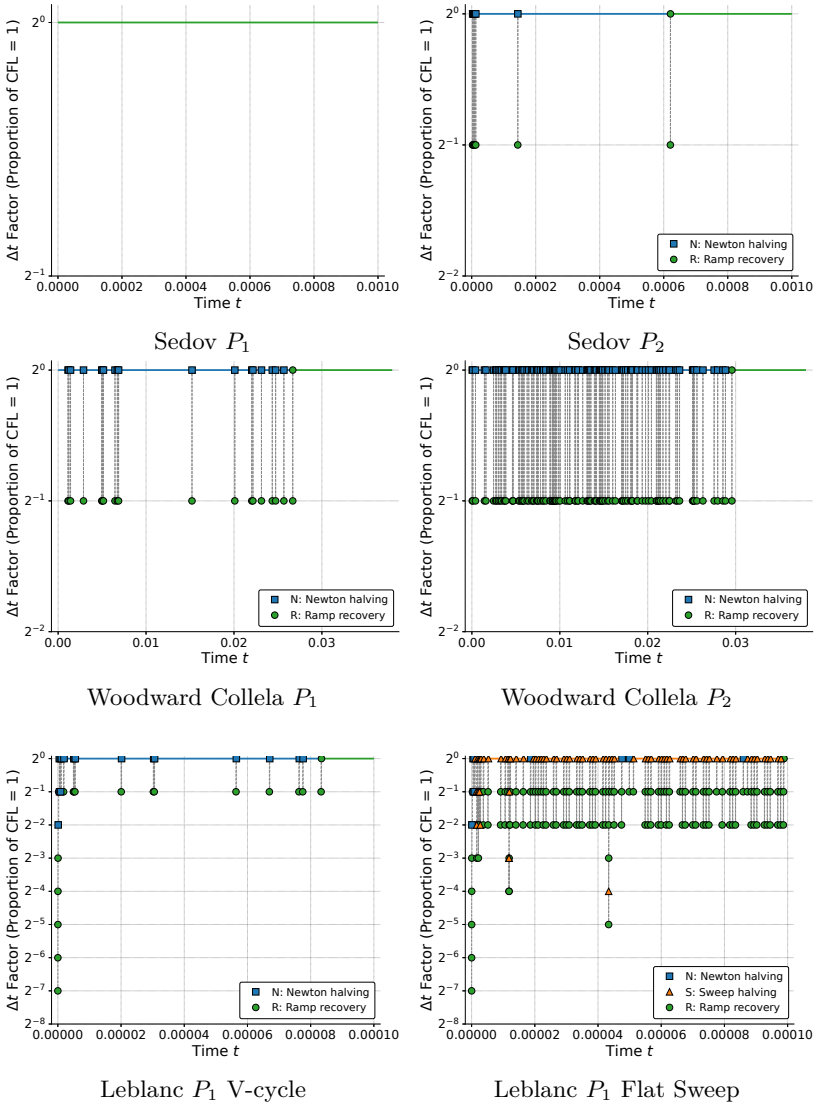


Fig. 4 Δt-halving plots for 1D Examples. Time on $x-$axis, Δt factor on y-axis.

4.2 2D Results

Example 4 (2D vortex evolution accuracy) To assess multi-dimensional spatial and temporal accuracy, we simulate the advection of an isentropic vortex by a uniform mean flow on a doubly periodic domain $\Omega = [0, 10] \times [0, 10]$. The mean flow $(\rho, u, v, p) = (1, 1, 1, 1)$ is perturbed by a vortex centered at $(x_0, y_0) = (5, 5)$. The initial condition is defined by:

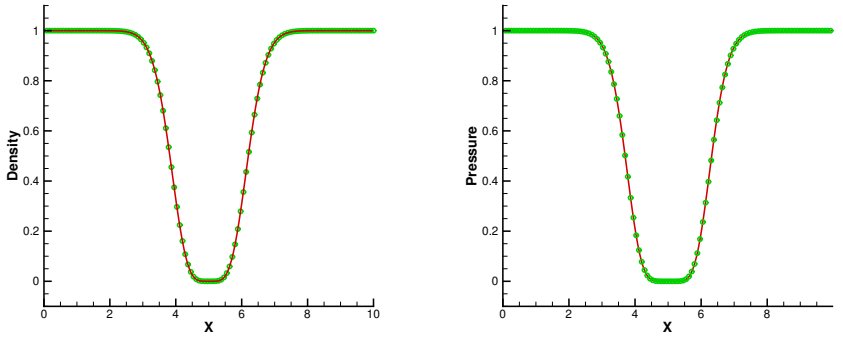
$$u(x, y, 0) = 1 - \frac{\Gamma}{2\pi} \exp\left(\frac{1 - r^2}{2}\right)(y - y_0),$$

$$v(x, y, 0) = 1 + \frac{\Gamma}{2\pi} \exp\left(\frac{1 - r^2}{2}\right)(x - x_0),$$

$$\rho(x, y, 0) = \left[1 - \frac{(\gamma - 1)\Gamma^2}{8\gamma\pi^2} \exp\big(1 - r^2\big)\right]^{\frac{1}{\gamma - 1}},$$

$$\text{and } p(x, y, 0) = \rho(x, y, 0)^{\gamma},$$

where $r^2 = (x - x_0)^2 + (y - y_0)^2$, and the vortex strength is $\Gamma = 10.0828$. The exact solution is the passive advection of the vortex by the mean flow, which we compute until a final time of $T = 0.01$. We show plots for P_1 with on a triangulation with 120×120 elements with pure SUPG (no shock capturing added). We include optimal numerical convergence results for P_1 in Tables 4 and 5 for both density and pressure, respectively. We omit results for P_2 for this case, due to the need for adaptive time-stepping. We show that the sweeping procedure does not destroy the underlying accuracy of the scheme.



Density Plot Pressure Plot

Fig. 5 Example 4: Numerical results for 2D vortex evolution test case using P_1 SUPG with sweeping limiter. Green symbols are approximate results. Red line is exact solution.

N	h	$L^1(\rho)$	rate	$L^2(\rho)$	rate	$L^\infty(\rho)$	rate
15×15	0.6667	5.57e-01	—	1.73e-01	—	1.17e-01	—
30×30	0.3333	1.43e-01	1.96	4.49e-02	1.95	4.45e-02	1.40
60×60	0.1667	3.64e-02	1.98	1.15e-02	1.97	1.16e-02	1.94
120×120	0.0833	9.11e-03	2.00	2.88e-03	1.99	2.96e-03	1.97

Table 4 Dense-quadrature convergence (2D) for Density using P_1 elements.

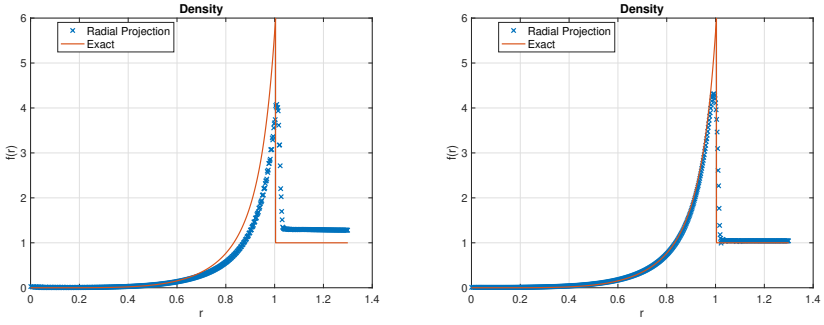
N	h	$L^1(p)$	rate	$L^2(p)$	rate	$L^\infty(p)$	rate
15×15	0.6667	6.04e-01	—	1.82e-01	—	1.49e-01	—
30×30	0.3333	1.63e-01	1.89	4.94e-02	1.88	4.30e-02	1.80
60×60	0.1667	4.14e-02	1.98	1.27e-02	1.96	1.29e-02	1.73
120×120	0.0833	1.04e-02	1.99	3.19e-03	1.99	3.27e-03	1.98

Table 5 Dense-quadrature convergence (2D) for Pressure using P_1 elements.

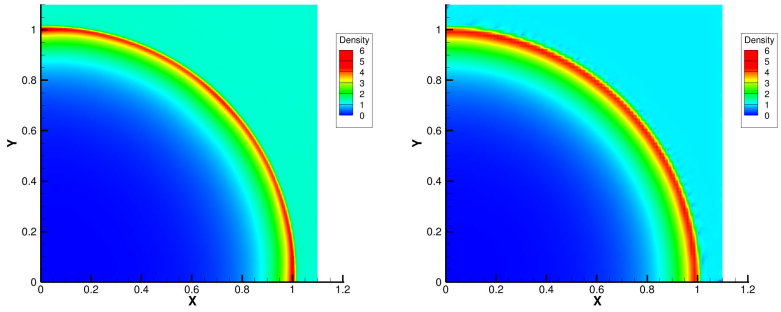
Example 5 (2D Sedov blast) We simulate the 2D Sedov point-blast on the quarter-domain $\Omega = [0, 1.1] \times [0, 1.1]$. The domain contains an ambient gas at rest with $\rho = 1$ and a near-vacuum background total energy of $E_{bg} = 10^{-12}$. A total blast energy of $E_0 = 0.244816$ is injected near the origin. To accommodate higher-order elements where the origin node may have a zero or undefined discrete weight, the discrete delta injection is replaced by a highly localized Gaussian distribution in proportion to the local element spacing Δx:

$$E(x, y, 0) = E_{bg} + \frac{E_0}{V_h} \exp\left(-\frac{r^2}{R_0^2}\right),$$

where $R_0 = 2\Delta x$, $r^2 = x^2 + y^2$, and $V_h = \sum_i w_i \exp\left(-r_i^2/R_0^2\right)$ is the discrete volume integrated over the nodal lumped-mass weights w_i. Reflective boundaries are applied at $x = 0$ and $y = 0$, with outflow boundaries on the top and right edges. The test is run until $T = 1.0$. We show a P_1 result with 160×160 elements and a P_2 result with 80×80 elements on a triangulation using YZ-β shock capturing. The P_1 scheme under-resolves the blast, whereas P_2 captures the wave front much better with the same total degrees of freedom, albeit with more oscillations.



Density Radial Projection



Density Contour

Fig. 6 Example 5: Numerical results for the 2D Sedov blast wave using P_1 (160×160 cells, left) and P_2 (80×80 cells, right) SUPG with YZ-β shock capturing and sweeping limiter.

Example 6 (Mach 2000 test) To demonstrate the ability of the V-cycling procedure to address both large magnitude and number of errors (but warn against insufficient stabilization), we run a Mach 2000 jet problem on the domain $\Omega = [0,1] \times [0,0.25]$. The initial condition is a uniform ambient gas at rest:

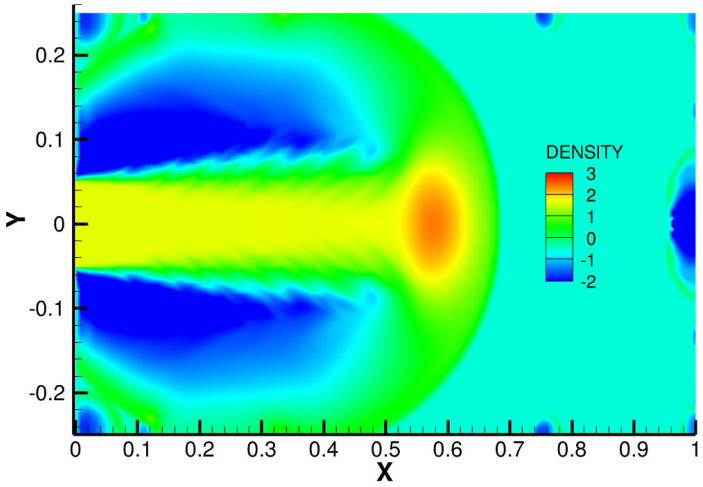
$$(\rho, u, v, p) = (0.5, 0, 0, 0.4127).$$

A high-speed jet with jet speed 800 is injected through a Dirichlet inflow boundary condition on the left edge ($x = 0$):

$$(\rho, u, v, p) = \begin{cases} (5, 800, 0, 0.4127) & \text{if } y \leqslant 0.05 \\ (0.5, 0, 0, 0.4127) & \text{if } y > 0.05. \end{cases}$$

The bottom boundary is reflective, while the top and right boundaries are set as outflow. The simulation is evaluated at $T = 0.001$. We show a P_1 result on a triangulation with 320×80 elements using artificial viscosity as shock capturing. This shock capturing method clearly smears the solution. Nevertheless, it allowed the Newton solver to pass the first step, whereas the solver failed immediately with YZ-β shock capturing.

We note the appearance of artifacts in the solution due to the large magnitude violations being treated on the coarse grid. Mass is transferred too aggressively, creating nonphysical distortions in the solution. We include this plot as a warning that while the sweeping limiter may be able to output a solution, it cannot save an inherently poor performing algorithm. We emphasize that the sweeping procedure performed much better on the same test case in the finite difference WENO test case in [8].



Density Plot

Fig. 7 Example 6: Color contour of results on a logarithmic scale for Mach 2000 test case using P_1 SUPG with artificial viscosity and sweeping limiter.

Example 7 (Shock diffraction) We consider the diffraction of a Mach 5.09 shock over a 90-degree corner. The computational domain is $\Omega = [0, 13] \times [0, 11]$ excluding the solid obstacle region $[0, 1] \times [0, 6]$. The test is initialized with a pure, right-moving shock located at $x = 0.5$. The ambient gas ahead of the shock ($x \geqslant 0.5$) is defined as:

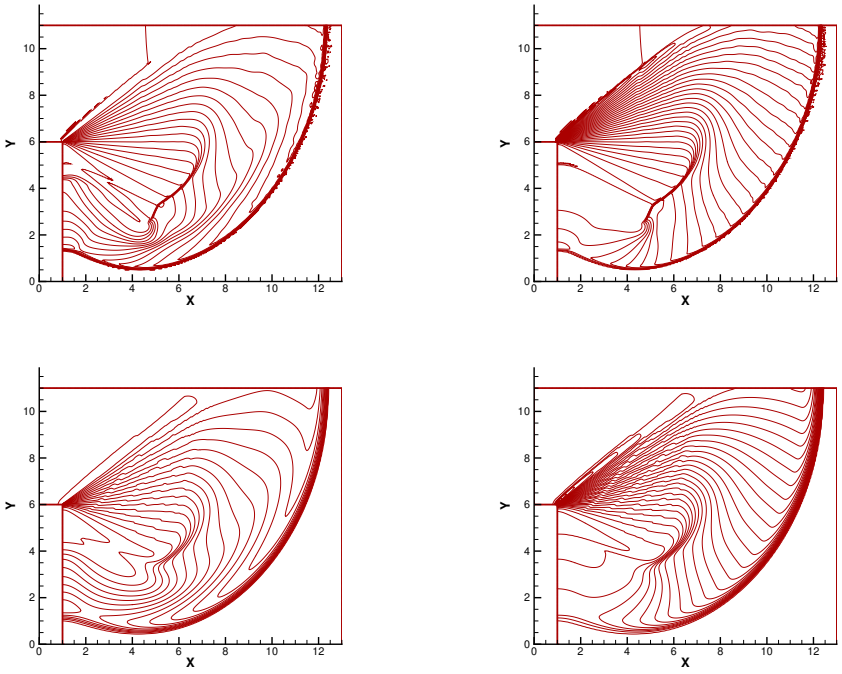
$$(\rho, u, v, p) = (\gamma, 0, 0, 1).$$

For $x < 0.5$, the post-shock state is derived from the Rankine-Hugoniot relations for a normal shock moving at Mach 5.09 into the ambient state. The left boundary maintains this post-shock inflow state to drive the shock forward, while we use solid reflective walls for the diffracting corner. The remaining boundaries are outflow. The results are obtained at a final time of $T = 2.3$. We show P_1 results on a triangulation with 260×260 elements. We show a comparison of the YZ-β shock capturing with artificial viscosity. We note that while the YZ-β shock capturing is less diffusive than artificial viscosity, the YZ-β shock capturing is still more diffusive than a discontinuous Galerkin approach at the same resolution. We obtain a similar P_2 result using artificial viscosity (omitted for brevity), but have Newton convergence issues using YZ-β shock capturing. The P_1 results demonstrates the difference in the number of sweeps needed depending on the level of numerical dissipation in the solution as shown in Table 7.

Table 6 2D Correction Data

		Vortex	2D Sedov P_1	2D Sedov P_2	Mach2000
Number	DOF	14,400	25,921	25,921	25,998
$P_{\min}$	Min	-1.8536e-08	-3.7597e-04	-1.7959e-03	0.0000e+00
	Max	-3.1075e-07	-1.2673e+01	-3.8667e+01	-5.6634e+05
	Avg	-1.2019e-07	-1.2047e-01	-4.2151e-01	-1.6936e+03
n_{bad}	Max	4	25657	25545	10436
	Avg	1.67	3166.39	5323.95	4890.61

		Shock Diffraction $YZ - \beta$	Shock Diffraction AV
Number	DOF	55,281	55,281
$P_{\min}$	Min	-1.5165e-01	-5.7297e-04
	Max	-8.0256e+00	-3.3822e+01
	Avg	-2.5931e+00	-3.4686e-01
n_{bad}	Max	204	102
	Avg	135.05	8.37



Density Plot Pressure Plot

Fig. 8 Example 7: Contour plot of results for shock diffraction test case using P_1 SUPG and sweeping limiter with YZ-β stabilization (top) and artificial viscosity (bottom). For density (left), we plot 20 equally spaced contour lines between 0.066227 and 7.0668. For pressure (right), we plot 40 contour lines between 0.091 and 42.

Table 7 2D Sweep Data

		Vortex	2D Sedov P_1	2D Sedov P_2	Mach2000
Number	Timestep	2	1373	544	1541
	RK Stage	6	4119	1632	4623
Sweep	Events	3	4120	1694	4719
	Total	4	1219610	434261	268021
	Min	1	7	10	1
	Max	2	10054	3907	491
	Avg	1.333	296.022	256.352	56.796
V-cycle	Events	0 / 3	1806 / 4120	1410 / 1694	4367 / 4719
	Total	0	119816	42498	24427
	Max	0	1005	390	49
	Avg	0.000	29.082	25.087	5.176
Density	Total	3	4120	1694	4719

		Shock Diffraction $YZ - \beta$	Shock Diffraction AV
Number	Timestep	564	528
	RK Stage	1692	1584
Sweep	Events	1739	544
	Total	3074	562
	Min	1	1
	Max	4	3
	Avg	1.768	1.033
V-cycle	Events	0 / 1739	0 / 544
	Total	0	0
	Max	0	0
	Avg	0.000	0.000
Density	Total	1739	544

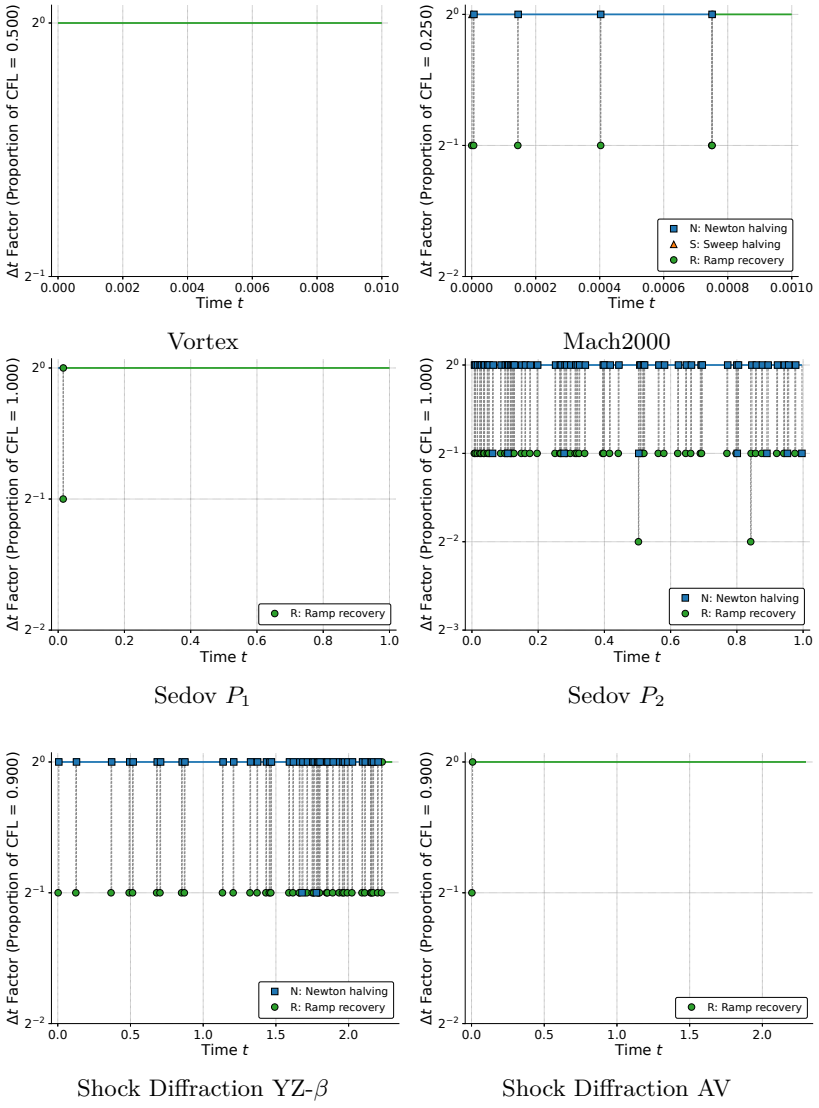


Fig. 9 Δt-halving plots for 2D Examples. Time on $x-$axis, Δt factor on y-axis.

5 Concluding Remarks

In this work, we provide a new sweeping framework to ensure positivity for density and the nonlinear pressure function for an implicit, continuous finite element discretization of the compressible Euler equations. As an extension of the finite difference WENO framework method in [8], we show the capacity of the sweeping limiter to treat large violations. To improve computational cost, we provide a multi-grid acceleration procedure that improves convergence for oscillatory data.

Ultimately, we find that the post-processing procedure performs best, particularly in terms of computational cost and accuracy, on smooth problems or those with small magnitude violations, as shown in the numerical tests. Thus, the best applications of the method are where the underlying scheme performs well, particularly in terms of accuracy and oscillation control, but is not inherently positivity preserving, as echoed in the results in [8].

The implicit approach enabled by the sweeping method is not limited to a continuous Galerkin setting, and can be applied to finite difference, finite volume, and discontinuous Galerkin schemes. Furthermore, the procedure extends beyond the nonlinear pressure function to any concave function of the conserved variables in hyperbolic systems. Therefore, our future work includes extending the approach to other hyperbolic systems including Navier-Stokes and magnetohydrodynamics. We are also interested in other implicit applications, such as nodal discontinuous Galerkin or finite volume methods.

End Matter

Acknowledgements This research was supported in part by an appointment to the first author at the Oak Ridge National Laboratory Research Student Internships Program or Technical and Professional Internships Program, sponsored by the U.S. Department of Energy and administered by the Oak Ridge Institute for Science and Education (RSI/TPI). The first author is also partially supported by the NSF Graduate Research Fellowship Program and GEM Fellowship program. The research of the second author is partially supported by NSF grant DMS-2309249 and DOE grant DE-NA0004265.

Data Availability Data will be made available on reasonable request.

Compliance with Ethical Standards

Conflict of Interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

Ethical Approval Not applicable.

References

[1] Alexander, R. Diagonally implicit Runge–Kutta methods for stiff O.D.E.'s. *SIAM Journal on Numerical Analysis*, 14(6), 1006–1021. (1977). https://doi.org/10.1137/0714068

[2] Aranda, J., Betancourt, S., Díaz, J.F., Paz, J., and Valencia, F. Fairness and consensus in an asynchronous opinion model for social networks. *Journal of Logical and Algebraic Methods in Programming*, 151, 101140. (2026). https://doi.org/10.1016/J.JLAMP.2026.101140

[3] Book, D.L., Boris, J.P., and Hain, K. Flux-corrected transport II: Generalizations of the method. *Journal of Computational Physics*, 18(3), 248–283. (1975). https://doi.org/10.1016/0021-9991(75)90002-9

[4] Boris, J.P. and Book, D.L. Flux-corrected transport. I. SHASTA, a fluid transport algorithm that works. *Journal of Computational Physics*, 11(1), 38–69. (1973). https://doi.org/10.1016/0021-9991(73)90147-2

[5] Boris, J.P. and Book, D.L. Flux-corrected transport. III. Minimal-error FCT algorithms. *Journal of Computational Physics*, 20(4), 397–431. (1976). https://doi.org/10.1016/0021-9991(76)90091-7

[6] Brooks, A.N. and Hughes, T.J.R. 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*, 32(1-3), 199–259. (1982). https://doi.org/10.1016/0045-7825(82)90071-8

[7] Cockburn, B. and Shu, C.-W. Runge–Kutta discontinuous Galerkin methods for convection-dominated problems. *Journal of Scientific Computing*, 16, 173–261. (2001). https://doi.org/10.1023/A:1012873910884

[8] Griffin, D. and Shu, C.-W. A sweeping positivity-preserving high order finite difference WENO scheme for Euler equations. *Research in the Mathematical Sciences*. 13(38). (2026). https://doi.org/10.1007/s40687-026-00621-2

[9] Harten, A., Engquist, B., Osher, S., and Chakravarthy, S.R. Uniformly high order accurate essentially non-oscillatory schemes, III. *Journal of Computational Physics*, 131(1), 3–47. (1997). https://doi.org/10.1006/JCPH.1996.5632

[10] Huang, Q.M., Zhou, H., Ren, Y.X., and Wang, Q. A general positivity-preserving algorithm for implicit high-order finite volume schemes solving the Euler and Navier-Stokes equations. *Journal of Computational Physics*, 508, 112999. (2024). https://doi.org/10.1016/J.JCP.2024.112999

[11] Hughes, T.J.R., Scovazzi, G., and Tezduyar, T.E. Stabilized methods for compressible flows. *Journal of Scientific Computing* 43, 343–368. (2010). https://doi.org/10.1007/s10915-008-9233-5.

[12] Hughes, T.J.R. and Tezduyar, T.E. Finite element methods for first-order hyperbolic systems with particular emphasis on the compressible Euler equations. *Computer Methods in Applied Mechanics and Engineering*, 45(1-3), 217–284. (1984). https://doi.org/10.1016/0045-7825(84)90157-9

[13] Jiang, G.S. and Shu, C.-W. Efficient implementation of weighted ENO schemes. *Journal of Computational Physics*, 126(1), 202–228. (1996). https://doi.org/10.1006/JCPH.1996.0130

[14] Kuzmin, D. Explicit and implicit FEM-FCT algorithms with flux linearization. *Journal of Computational Physics*, 228(7), 2517–2534. (2009). https://doi.org/10.1016/J.JCP.2008.12.011

[15] Kuzmin, D. Monolithic convex limiting for continuous finite element discretizations of hyperbolic conservation laws. *Computer Methods in Applied Mechanics and Engineering*, 361, 112804. (2020). https://doi.org/10.1016/J.CMA.2019.112804

[16] Kuzmin, D. and Möller, M. Algebraic flux correction II. Compressible Euler equations. (2005). In: Kuzmin, D., Löhner, R., Turek, S. (eds) Flux-Corrected Transport. Scientific Computation. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-27206-2_7

[17] Kuzmin, D. and Möller, M. (2005). Algebraic Flux Correction I. Scalar Conservation Laws. In: Kuzmin, D., Löhner, R., Turek, S. (eds) Flux-Corrected Transport. Scientific Computation. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-27206-2_6

[18] Kuzmin, D., Möller, M., and Turek, S. High-resolution FEM–FCT schemes for multidi-

mensional conservation laws. *Computer Methods in Applied Mechanics and Engineering*, 193(45-47), 4915–4946. (2004). https://doi.org/10.1016/J.CMA.2004.05.009

[19] Kuzmin, D. and Turek, S. Flux correction tools for finite elements. *Journal of Computational Physics*, 175(2), 525–558. (2002). https://doi.org/10.1006/JCPH.2001.6955

[20] Laiu, M.P. and Hauck, C.D. Positivity limiters for filtered spectral approximations of linear kinetic transport equations. *Journal of Scientific Computing*, 78, 918–950. (2019). https://doi.org/10.1007/S10915-018-0790-Y

[21] Le Beau, G.J., Ray, S.E., Aliabadi, S.K., and Tezduyar, T.E. SUPG finite element computation of compressible flows with the entropy and conservation variables formulations. *Computer Methods in Applied Mechanics and Engineering*, 104(3), 397–422. (1993). https://doi.org/10.1016/0045-7825(93)90033-T

[22] Liu, X.D., Osher S., and Chan T. Weighted essentially non-oscillatory schemes. *Journal of Computational Physics*, 115(1), 200–212. (1994). https://doi.org/10.1006/JCPH.1994.1187

[23] Liu, Y., Cheng, Y., and Shu, C.-W. A simple bound-preserving sweeping technique for conservative numerical approximations. *Journal of Scientific Computing*, 73, 1028–1071. (2017). https://doi.org/10.1007/s10915-017-0395-x

[24] Löhner, R., Morgan, K., Peraire, J., and Vahdati, M. Finite element flux-corrected transport (FEM–FCT) for the Euler and Navier–Stokes equations. *International Journal for Numerical Methods in Fluids*, 7(10), 1093–1109. (1987). https://doi.org/10.1002/FLD.1650071007

[25] Möller, M. On an efficient solution strategy of Newton type for implicit finite element schemes based on algebraic flux correction. *International Journal for Numerical Methods in Fluids*, 56(8), 1085–1091. (2008). https://doi.org/10.1002/FLD.1645

[26] Moujaes, P. and Kuzmin, D. Monolithic convex limiting and implicit pseudo-time stepping for calculating steady-state solutions of the Euler equations. *Journal of Computational Physics*, 523, 113687. (2025). https://doi.org/10.1016/J.JCP.2024.113687

[27] Shi, C. and Shu, C.-W. On local conservation of numerical methods for conservation laws. *Computers & Fluids*, 169, 3–9. (2018). https://doi.org/10.1016/J.COMPFLUID.2017.06.018

[28] Shu, C.-W. High order weighted essentially nonoscillatory schemes for convection dominated problems. *SIAM Review*, 51(1), 82–126. (2009). https://doi.org/10.1137/070679065

[29] Tezduyar, T.E. and Senga, M. SUPG finite element computation of inviscid supersonic flows with YZβ shock-capturing, 36(1), 147–159. (2007). https://doi.org/10.1016/j.compfluid.2005.07.009

[30] Trottenberg, U., Oosterlee, C.W., and Schuller, A. Multigrid, p. 631. Academic Press, London. (2001).

[31] Wesseling, P. Introduction to multigrid methods. Institute for Computer Applications in Science and Engineering, Hampton, VA. (1995).

[32] Woodward, P. and Colella, P. The numerical simulation of two-dimensional fluid flow with strong shocks. *Journal of Computational Physics*, 54(1), 115–173. (1984). https://doi.org/10.1016/0021-9991(84)90142-6

[33] Zhang, X. and Shu, C.-W. On maximum-principle-satisfying high order schemes for scalar conservation laws. *Journal of Computational Physics*, 229(9), 3091–3120. (2010). https://doi.org/10.1016/J.JCP.2009.12.030

[34] Zhang, X. and Shu, C.-W. Maximum-principle-satisfying and positivity-preserving high-order schemes for conservation laws: Survey and new developments. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 467(2134), 2752–2776. (2011). https://doi.org/10.1098/RSPA.2011.0153

[35] Zhang, X. and Shu, C.-W. Positivity-preserving high order finite difference WENO schemes for compressible Euler equations. *Journal of Computational Physics*, 231(5), 2245–2258. (2012). https://doi.org/10.1016/J.JCP.2011.11.020

[36] Zienkiewicz, O.C., Taylor, R.L., and Zhu, J.Z. The finite element method : its basis and fundamentals. Elsevier. (2013). https://doi.org/10.1016/C2009-0-24909-9