ORIGINAL PAPER

A Sub-element Adaptive Shock Capturing Approach for Discontinuous Galerkin Methods

Johannes Markert[1] · Gregor Gassner[2] · Stefanie Walch[3]

Abstract

In this paper, a new strategy for a sub-element-based shock capturing for discontinuous Galerkin (DG) approximations is presented. The idea is to interpret a DG element as a collection of data and construct a hierarchy of low-to-high-order discretizations on this set of data, including a first-order finite volume scheme up to the full-order DG scheme. The different DG discretizations are then blended according to sub-element troubled cell indicators, resulting in a final discretization that adaptively blends from low to high order within a single DG element. The goal is to retain as much high-order accuracy as possible, even in simulations with very strong shocks, as, e.g., presented in the Sedov test. The framework retains the locality of the standard DG scheme and is hence well suited for a combination with adaptive mesh refinement and parallel computing. The numerical tests demonstrate the sub-element adaptive behavior of the new shock capturing approach and its high accuracy.

Keywords High-order methods · Discontinuous Galerkin spectral element method · Finite volume method · Shock capturing · Astrophysics · Stellar physics

Mathematics Subject Classification 76M99

1 Introduction

For non-linear hyperbolic problems, such as the compressible Euler equations, there are two major sources of instabilities when applying discontinuous Galerkin (DG) methods as a high-order spatial discretization. (i) Aliasing is caused by under-resolution of, e.g., turbulent vortical structures and can lead to instabilities that may even crash the code. As a cure, de-aliasing mechanisms are introduced in the DG methodology based on, e.g., filtering

✉ Johannes Markert
  jmarkert@math.uni-koeln.de

1   Department for Mathematics and Computer Science, University of Cologne, Weyertal 86-90, 50931 Cologne, Germany

2   Department for Mathematics and Computer Science, Center for Data and Simulation Science, University of Cologne, Weyertal 86-90, 50931 Cologne, Germany

3   I. Physikalisches Institut, Center for Data and Simulation Science, University of Cologne, Zülpicher Str. 77, 50937 Cologne, Germany

[29, 30], polynomial de-aliasing [35, 36, 57] or analytical integration [23, 31], split forms of the non-linear terms that for instance preserve kinetic energy [17, 22, 26, 61], and entropy stability [3, 5–7, 19, 21, 25, 42–46, 60]. (ii) The second major source of instabilities is the Gibbs phenomenon, i.e., oscillations at discontinuities. It is well known that solutions of non-linear hyperbolic problems may develop discontinuities in finite time, so-called shocks. While aliasing issues can be mostly attributed to variational crimes, i.e., a bad design and implementation of the numerical discretization with simplifications that affects aliasing (e.g., collocation), oscillations at discontinuities have their root deeper in the innards of the high-order DG approach and are unfortunately an inherent property of the high-order polynomial approximation space. Oscillations are fatal when physical constraints and bounds on the solution exist that then break because of over- and undershoots, resulting in nonphysical solutions such as negative density or pressure.

In this work, we focus on a remedy for the second major stability issue, often referred to as shock capturing. There are many shock capturing approaches available in the DG literature, for instance based on slope limiters [10, 11, 40] or (H)WENO limiters [27, 50, 65], filtering [2], and artificial viscosity [9, 47]. Slope limiters or (H)WENO limiters are always applied in combination with troubled cell indicators that detect shocks and only flag the DG elements that are affected by oscillations. Only in these elements, the DG polynomial is replaced by an oscillation-free reconstruction using data from neighboring elements. Element-based slope limiting, e.g., with TVB limiters, is effective for low-order DG discretizations only, as its accuracy strongly degrades when increasing the polynomial order N, as in some sense, the size of the DG elements Δx gets larger and larger with higher polynomial degree N. The accuracy of the slope limiters is not based on the DG resolution $\sim \frac{\Delta x}{N}$, but only on the element size Δx. Considering that a finite volume (FV) discretization with high-order reconstructions and limiters typically resolve a shock within about 2–3 cells, the shock width for high-order DG schemes with large elements would be very wide when relying on element-based limited reconstructions. The same behavior is still an issue for high-order reconstruction-based limiters such as the (H)WENO methodology. While these formally use high-order reconstructions, its leading discretization parameter is still Δx and not $\frac{\Delta x}{N}$, and hence, the shock width still scales with Δx.

Sub-element resolution, i.e., a numerical shock width that scales proportional to $\frac{\Delta x}{N}$, can be achieved with, e.g., artificial viscosity. The idea is to widen the discontinuity into a sharp, but smooth profile, such that high-order polynomials can resolve it. Again, some form of troubled cell indicator is introduced to not only flag the element that contains the shock (or that oscillates), but also determine the amount of necessary viscosity, e.g., based on local entropy production [66]. It is interesting that less viscosity is needed, i.e., the sharper shock profiles are possible, the higher the polynomial degree of the DG discretization. Shocks can be captured in a single DG element if the polynomial order is high enough [47]. An issue of artificial dissipation is that a high amount of artificial viscosity is needed for very strong shocks, which makes the overall discretization very stiff with a very small explicit time-step restriction. Local time-stepping can be used to reduce this issue [24] or specialized many stage Runge-Kutta schemes with optimized coefficients for strongly dissipative operators [32].

An alternative idea to achieve sub-element resolution is to actually replace the elements by sub-elements (lower order DG on finer grid) or maybe even FV subcells. A straight forward, maybe even a natural idea in the DG methodology is to use the *hp*-adaptivity, where the polynomial degree is reduced, and simultaneously, the grid resolution is increased close to discontinuities. When switching to the lowest order DG, i.e., a first-order DG, which is nothing but a first-order FV method, it is assumed that the inherent artificial dissipation of the

first-order FV scheme is enough to clear all oscillations at even the strongest shocks. The low accuracy of the first-order FV method is compensated by a respective (very strong) increase in local grid resolution through the *h*-adaptivity. An obvious downside to the *hp*-adaptivity approach is the strong dynamic change of the approximation space during the simulation and the associated strong change of the underlying data structures.

Thus, as a simpler alternative, subcell-based FV discretizations with a first-order, second-order TVD, or high-order WENO reconstruction were introduced, e.g., in [13, 14, 56, 59, 62]. The idea is to completely switch the troubled DG element to a robust FV discretization on a fine subcell grid. This approach keeps the change of the data local to the affected DG element and helps to streamline the implementation. The robust and essential oscillation-free FV scheme on the finer subcell grid in the DG element is constructed based on the available DG data. This helps to keep the data structures feasible and reduces the shock capturing again to an element local technique instead of changing the global mesh topology and approximation space. Consequently, this again allows to combine the approach with the idea of a troubled cell mechanism to identify the oscillatory DG elements and replace the high-order DG update with the robust FV update. As noted, there are many different variants for the FV subcell scheme with the high-order WENO [13, 14, 56, 59, 62] variant being especially interesting, as it alleviates the "stress" on the troubled cell indicator. When accidentally switching the whole high-order DG element to the subcell FV scheme in smooth parts of the solution, accuracy may strongly degrade. A first-order FV scheme, even on the fine subcell grid, would wash away all fine structure details that the high-order DG approach simulated. Thus, from this point of view, a high-order subcell WENO FV scheme is highly desired in this context. A downside of a subcell WENO FV scheme is the non-locality of the data dependence due to the high-order reconstruction stencils. This is especially cumbersome close to the interface of the high-order DG elements, where the reconstruction stencils reach into the neighboring DG elements and thus fundamentally change the data dependency footprint of the resulting implementation, that consequently has a direct impact on the parallelization of the code.

In this paper, the goal is to augment a high-order DG method with sub-element shock capturing capabilities to allow the robust simulation of highly supersonic turbulent flows which feature strong shocks, as, e.g., in astrophysics, without changing the data dependency footprint. Instead of flagging an element and completely switching from the high-order DG scheme to the subcell FV scheme, we aim to smoothly combine different schemes in the flagged element. To achieve this, we first interpret the DG element as a collection of available mean value data. As depicted in Fig. 1, local reconstructions allow to define a hierarchy of approximation spaces, from pure piecewise constant approximations (subcell FV) up to a smooth global high-order polynomial (DG element), with all piecewise polynomial combinations (sub-element DG) in between. We then associate each possible approximation space with the corresponding FV or DG discretization to define a hierarchy of schemes that is available for the given collection of data.

Instead of switching between these different schemes, we aim to smoothly blend them to achieve the highest possible accuracy in every DG element. Assuming that the first-order subcell FV approximation has enough inherent dissipation to capture shocks in an oscillation-free manner, the goal is to give the FV discretization a high enough weight at a shock and then smoothly transition to higher order (sub-element) DG away from the shock. In contrast to the common approaches, where one indicator is used for the whole DG element, we aim to introduce sub-element indicators that are adaptively adjusted inside the DG element. A difficulty that arises when using sub-element indicators and an adaptive blending approach is to retain conservation of the final discretization for arbitrary combinations of blending weights.

Fig. 1 Sketch of the main idea in a 1D setting. Starting with eight mean values (red), we interpret them as a collection of available data. We construct four different approximation spaces (blue): piecewise constant on a subcell grid, piecewise linear and piecewise cubic on a sub-element grid, and the full seventh-degree polynomial on the element
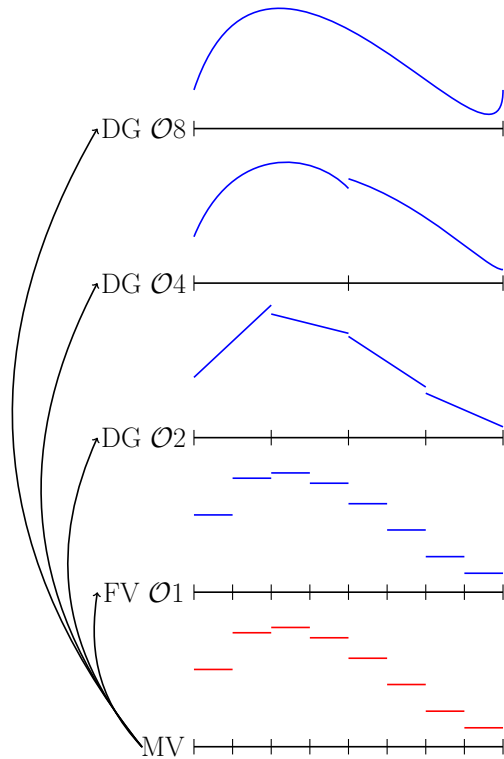
The final discretization is a blended scheme, where the weights of the different discretizations may vary inside the DG element. To demonstrate this idea, the resulting behavior of the new approach is depicted in Fig. 2 for the example of a Sedov blast wave in 2D. In this particular setup, we start with the eighth-order DG as our baseline high-order discretization. The gray grid lines indicate the eighth-order DG element borders. Re-interpreting the 2D eighth-order DG element as 8^2 mean value data, we can construct either a first-order FV scheme on 8^2 subcells, a fourth-order DG scheme on 2^2, or a second-order DG scheme on 4^2 sub-elements. The blending of the four available discretizations is visualized via a weighted blending factor (introduced in (57)) in Fig. 2 ranging from 1 (dark red, basically pure first-order FV) up to 8 (dark blue, basically pure eighth-order DG) and all combinations in between.

Clearly, the new approach adaptively changes on a sub-element level with a very localized low order near the shock front, while a substantial part of the high-order DG approximation is recovered away from the shock. We refer to the validation and application sections, Sects. 3 and 4, for a more detailed discussion.

The remainder of the paper is organized as follows: in Sect. 2, we introduce the numerical scheme with the blending of the different discretizations. Section 3 includes numerical validations and Sect. 4 shows an application of the presented approach with the conclusion given in the last section.
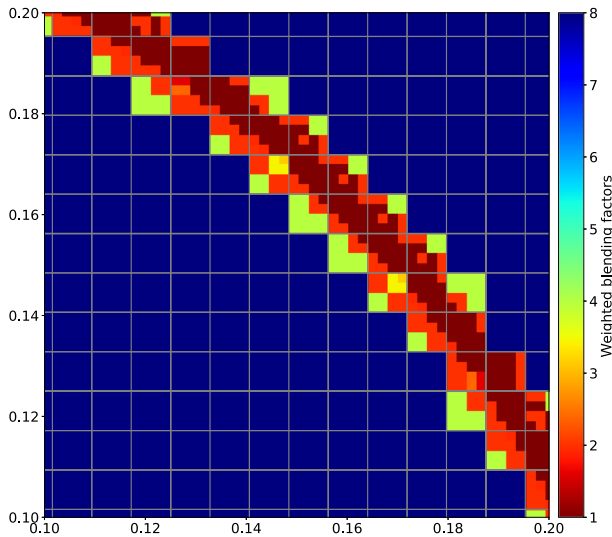
Fig. 2 Visualization of the sub-element adaptive blending approach for the 2D Sedov blast wave, see Sect. 3 for a detailed discussion. The gray grid lines represent the eighth-order DG elements. The color represents the blending process, where dark red (value 1) corresponds to pure first-order FV and dark blue (value 8) corresponds to the pure eighth-order DG scheme, with all combination in between. Note that the plotted value is only for illustration and does not correspond to an actual mixed mathematical order of the final discretization. However, the value nicely illustrates how even inside a DG element, the discretization is adaptively adjusted

2 The Sub-element Adaptive Blending Approach

The derivations in Sects. 2.1–2.6 are done in 1D, where the domain Ω is divided into Q non-overlapping elements. Each element q with midpoint x_q and size Δx_q is mapped to the reference space as

$$x(\chi) = x_q + \chi \, \Delta x_q \,, \quad \chi \in \left[-\frac{1}{2}, \frac{1}{2} \right]. \tag{1}$$

Each element holds a total number of degrees of freedom (DOF) N, where we require $N \in \{2^r, r \in \mathbb{N}\}$. For the FV method, the element is divided into N subcells of size $\frac{\Delta x_q}{N}$ leading to a uniform grid with midpoints μ_i and interfaces $\mu_{i \pm \frac{1}{2}}$ in the reference space at

$$\mu_i = -\frac{1}{2} + \frac{i}{N} - \frac{1}{2N} \quad \text{and} \quad \mu_{i \pm \frac{1}{2}} = \mu_i \pm \frac{1}{2N}, \quad i = 1, \cdots, N. \tag{2}$$

For the sub-element DG scheme of order $n = 2^r < N, r \in \mathbb{N}$, the element is divided into $\frac{N}{n}$ uniform sub-elements of size $\frac{n \Delta x_q}{N}$, and the mapping of each sub-element s to its (sub-)reference space reads as

$$\chi(\xi)_s^{\mathcal{O}(n)} = -\frac{1}{2} + \frac{s \, n}{N} - \frac{n}{2N} + \frac{n}{N} \xi, \quad s = 1, \cdots, \frac{N}{n}, \ \xi \in \left[-\frac{1}{2}, \frac{1}{2} \right]. \tag{3}$$

Figure A1 in Appendix A provides a visual representation of the reference element and its hierarchical decomposition into sub-elements and subcells.

2.1 Finite Volume Method

Consider the general 1D conservation law

$$\partial_t u + \partial_x f(u) = 0. \tag{4}$$

We approximate the exact solution $u(x)$ in element q with N mean values:

$$\bar{\boldsymbol{u}}_q = (\bar{u}_1, \cdots, \bar{u}_N)_q^{\mathrm{T}} \in \mathbb{R}^N, \tag{5}$$

residing on the uniform subcell grid (2). Then, the 1D FV semi-discretization on subcell i reads

$$\left(\partial_t \bar{u}_i\right)_q = -\frac{N}{\Delta x_q} \left(f^*\big((\bar{u}_i)_q, (\bar{u}_{i+1})_q\big) - f^*\big((\bar{u}_{i-1})_q, (\bar{u}_i)_q\big)\right), \quad i \in 1, \cdots, N, \tag{6}$$

where f^* denotes a consistent numerical two-point flux. The adjacent values from neighboring elements $q - 1$ and $q + 1$ are given by $(\bar{u}_0)_q := (\bar{u}_N)_{q-1}$ and $(\bar{u}_{N+1})_q := (\bar{u}_1)_{q+1}$. We define the FV difference matrix

$$\boldsymbol{\Delta} := \begin{pmatrix} -1 & 1 & 0 & \cdots & 0 \\ 0 & -1 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & -1 & 1 \end{pmatrix} \in \mathbb{R}^{N \times (N+1)}, \tag{7}$$

and rewrite (6) in compact matrix form

$$\partial_t \bar{\boldsymbol{u}}_q = -\frac{N}{\Delta x_q} \boldsymbol{\Delta} \, \boldsymbol{f}_q^* \tag{8}$$

with the numerical flux vector $\boldsymbol{f}_q^* \in \mathbb{R}^{N+1}$. For later use, we split $\boldsymbol{\Delta}$ into a volume and a surface operator:

$$\boldsymbol{\Delta} = \boldsymbol{\Delta}^{(\mathrm{V})} + \boldsymbol{\Delta}^{(\mathrm{S})} = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & -1 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & -1 & 0 \end{pmatrix} + \begin{pmatrix} -1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 0 & 1 \end{pmatrix}. \tag{9}$$

Remark 1 The FV volume operator vanishes when summed along columns, that is

$$\boldsymbol{1}^{\mathrm{T}} \boldsymbol{\Delta}^{(\mathrm{V})} = \boldsymbol{0}^{\mathrm{T}}, \tag{10}$$

where $\boldsymbol{1} = (1, \cdots, 1)^{\mathrm{T}} \in \mathbb{R}^N$ is the vector of ones.

2.2 DG Method

In this section, we briefly outline the discontinuous Galerkin spectral element method (DGSEM) [1, 34]. To derive the nth-order DG scheme within sub-element s ((3)), residing inside element q, we first rewrite the conservation law (4) into variational form with a test function ϕ and apply spatial integration-by-parts to the flux divergence:

$$\int_{\Omega_s} \partial_t u_q \, \phi \, \mathrm{d}\xi = \frac{N}{n \, \Delta x_q} \left(\int_{\Omega_s} f(u_q) \, \partial_\xi \phi \, \mathrm{d}\xi - f^*(u_q) \, \phi\big|_{\partial \Omega_s}\right). \tag{11}$$

We choose the test functions ϕ as Lagrange functions:

$$\ell_i(\xi) = \prod_{k=1, k \neq i}^{n} \frac{\xi - \xi_k}{\xi_i - \xi_k}, \quad i = 1, \cdots, n, \tag{12}$$

spanned with the collocation nodes $\xi_i \in \left[-\frac{1}{2}, \frac{1}{2}\right]$, and use the polynomial ansatz

$$u(t; \xi) \approx \sum_{i=1}^{n} \tilde{u}_i(t) \ell_i(\xi) = \tilde{\boldsymbol{u}}^{\mathrm{T}}(t) \boldsymbol{\ell}(\xi), \tag{13}$$

where

$$\tilde{\boldsymbol{u}}(t) = \big(\tilde{u}_1(t), \cdots, \tilde{u}_n(t)\big)^{\mathrm{T}} \quad \text{and} \quad \boldsymbol{\ell}(\xi) = \big(\ell_1(\xi), \cdots, \ell_n(\xi)\big)^{\mathrm{T}} \tag{14}$$

are the vectors of nodal coefficients and Lagrange polynomials. We use collocation of the non-linear flux functions with the same polynomial approximation. Numerical quadrature is collocated, as well. The nodes are the Gauss-Legendre quadrature nodes. These quadrature nodes and the interpolation polynomials are also illustrated in Fig. A1 in Appendix A.

Inserting everything into (11) gives the semi-discrete weak-form DG scheme

$$\mathcal{M} \, \partial_t \tilde{\boldsymbol{u}}_s = \frac{N}{n \, \Delta x_q} \left(\mathcal{D}^{\mathrm{T}} \mathcal{M} \, \tilde{\boldsymbol{f}}_s - \mathcal{B} \, \boldsymbol{f}_s^* \right). \tag{15}$$

The diagonal mass matrix is

$$(\mathcal{M})_{ij} = \delta_{ij} \, \omega_i \tag{16}$$

with associated numerical quadrature weights ω_i, and the differentiation matrix is

$$(\mathcal{D})_{ij} = \partial_\xi \ell_j(\xi)\big|_{\xi_i}. \tag{17}$$

The DG volume flux $\tilde{\boldsymbol{f}}_s$ is computed directly with the nodal values $\tilde{\boldsymbol{u}}_s$ of the polynomial ansatz (13)

$$\tilde{\boldsymbol{f}}_s = f\left(\tilde{\boldsymbol{u}}_s\right) := \Big(f\big((\tilde{u}_1)_s\big), \cdots, f\big((\tilde{u}_n)_s\big)\Big)^{\mathrm{T}} \in \mathbb{R}^n, \tag{18}$$

and the surface flux vector

$$\boldsymbol{f}_s^* = \Big(f^*(\underbrace{\tilde{u}_{s-1}^+, \tilde{u}_s^-}_{\substack{\text{left sub-element} \\ \text{interface}}}), 0, \cdots, 0, f^*(\underbrace{\tilde{u}_s^+, \tilde{u}_{s+1}^-}_{\substack{\text{right sub-element} \\ \text{interface}}})\Big)^{\mathrm{T}} \in \mathbb{R}^n \tag{19}$$

is constructed with the numerical flux evaluated at DG sub-element boundaries

$$\tilde{u}_s^+ = (\boldsymbol{b}^+)^{\mathrm{T}} \tilde{\boldsymbol{u}}_s \quad \text{and} \quad \tilde{u}_s^- = (\boldsymbol{b}^-)^{\mathrm{T}} \tilde{\boldsymbol{u}}_s, \tag{20}$$

where $\boldsymbol{b}^{\pm}$ are the boundary interpolation operators

$$\boldsymbol{b}^{\pm} = \Big(\ell_1\big(\pm \tfrac{1}{2}\big), \cdots, \ell_n\big(\pm \tfrac{1}{2}\big)\Big)^{\mathrm{T}} \in \mathbb{R}^n. \tag{21}$$

The boundary evaluation operator can be compactly written as

$$\mathcal{B} = \begin{pmatrix} -\ell_1(-\tfrac{1}{2}) & 0 & \cdots & 0 & \ell_1(\tfrac{1}{2}) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ -\ell_n(-\tfrac{1}{2}) & 0 & \cdots & 0 & \ell_n(\tfrac{1}{2}) \end{pmatrix} \in \mathbb{R}^{n \times n}. \tag{22}$$

We rewrite (15) and arrive at

$$\partial_t \tilde{\boldsymbol{u}}_s = \frac{N}{n\,\Delta x_q} \left(\boldsymbol{D}\,\tilde{\boldsymbol{f}}_s - \boldsymbol{B}\,\boldsymbol{f}_s^* \right) \tag{23}$$

with $\boldsymbol{D} := \mathcal{M}^{-1}\,\mathcal{D}^{\mathrm{T}}\,\mathcal{M}$ and $\boldsymbol{B} := \mathcal{M}^{-1}\mathcal{B}$.

Remark 2 The DG volume operator $\boldsymbol{D}$ vanishes when contracted with the vector of quadrature weights $\boldsymbol{\omega}^{\mathrm{T}} := (\omega_1, \cdots, \omega_n)^{\mathrm{T}} = \boldsymbol{1}^{\mathrm{T}}\,\mathcal{M}$,

$$\boldsymbol{\omega}^{\mathrm{T}}\,\boldsymbol{D} = \boldsymbol{0}^{\mathrm{T}}. \tag{24}$$

Proof

$$\boldsymbol{\omega}^{\mathrm{T}}\,\boldsymbol{D} = \boldsymbol{1}^{\mathrm{T}}\,\mathcal{M}\,\mathcal{M}^{-1}\,\mathcal{D}^{\mathrm{T}}\,\mathcal{M} = \boldsymbol{1}^{\mathrm{T}}\,\mathcal{D}^{\mathrm{T}}\,\mathcal{M} = (\mathcal{D}\,\boldsymbol{1})^{\mathrm{T}}\,\mathcal{M} = \boldsymbol{0}.$$

Here, we use the fact that $\mathcal{D}\,\boldsymbol{1} = \boldsymbol{0}$ is equivalent to taking the derivative of the constant function $u(x) = 1$, i.e., $\partial_x u(x) = \partial_x 1 = 0$.

2.3 Projection Operator

We consider a given polynomial of degree $n-1$ with its nodal data $\tilde{\boldsymbol{u}} \in \mathbb{R}^n$ in a (sub-)element, which we want to project to n_μ mean values on regular subcells. We define the projection operator $\boldsymbol{P}^{(n \to n_\mu)} \in \mathbb{R}^{n_\mu \times n}$ component-wise via the mean value of the polynomial in the interval of subcell μ_i:

$$\bar{u}_i = n_\mu \int_{\mu_{i-\frac{1}{2}}}^{\mu_{i+\frac{1}{2}}} \tilde{u}(\xi)\,\mathrm{d}\xi = \sum_{j=1}^{n} \tilde{u}_j \underbrace{n_\mu \int_{\mu_{i-\frac{1}{2}}}^{\mu_{i+\frac{1}{2}}} \ell_j(\xi)\,\mathrm{d}\xi}_{:=P_{ij}^{(n \to n_\mu)}}, \quad i = 1, \cdots, n_\mu. \tag{25}$$

The integration of the Lagrange polynomial is done exactly with an appropriate quadrature rule.

Remark 3 Contracting the projection operator $\boldsymbol{P}^{(n \to n_\mu)}$ with $\boldsymbol{1}_{n_\mu} := (1, \cdots, 1)^{\mathrm{T}} \in \mathbb{R}^{n_\mu}$ gives

$$\boldsymbol{1}_{n_\mu}^{\mathrm{T}}\,\boldsymbol{P}^{(n \to n_\mu)} = n_\mu\,\boldsymbol{\omega}^{\mathrm{T}}. \tag{26}$$

Proof

$$\sum_{i=1}^{n_\mu} P_{ij} = n_\mu \sum_{i=1}^{n_\mu} \int_{\mu_{i-\frac{1}{2}}}^{\mu_{i+\frac{1}{2}}} \ell_j(\xi)\,\mathrm{d}\xi = n_\mu \int_{\mu_{1-\frac{1}{2}}}^{\mu_{n_\mu+\frac{1}{2}}} \ell_j(\xi)\,\mathrm{d}\xi = n_\mu \int_{-\frac{1}{2}}^{\frac{1}{2}} \ell_j(\xi)\,\mathrm{d}\xi = n_\mu \omega_j. \tag{27}$$

Remark 4 When $n = n_\mu$, we succinctly write $\boldsymbol{P} := \boldsymbol{P}^{(n \to n)}$.

2.4 Reconstruction Operator

The inverse of the quadratic non-singular projection matrix $\boldsymbol{P}^{(n \to n)}$ can be interpreted as a reconstruction operator from n mean values to n nodal values of a polynomial with degree $n-1$:

$$\tilde{\boldsymbol{u}} = \boldsymbol{P}^{-1}\,\bar{\boldsymbol{u}} := \boldsymbol{R}\,\bar{\boldsymbol{u}}. \tag{28}$$

However, the naïve reconstruction (28) can lead to invalid data such as negative density or energy. Considering a convex set of permissible states for a given conservation law [64], we want to have a limiting procedure which recovers permissible states while still conserving the element mean value. Provided that the element average is a valid state

$$\langle \boldsymbol{u} \rangle := \frac{1}{n} \mathbf{1}^{\mathrm{T}} \bar{\boldsymbol{u}} \quad \in \{\text{permissible states}\}, \tag{29}$$

we expand the reconstruction process (28) by the following limiting procedure:

$$\boldsymbol{R}^{(\beta)} = \frac{1 - \beta}{n} \, \mathbf{1} \otimes \mathbf{1}^{\mathrm{T}} + \beta \, \boldsymbol{R}, \tag{30}$$

with the dyadic product $\mathbf{1} \otimes \mathbf{1}^{\mathrm{T}} \in \mathbb{R}^{n \times n}$ and the "squeezing" parameter $\beta \in [0, 1]$ chosen such that

$$\{ \underbrace{\tilde{\boldsymbol{u}}^{(\beta)}}_{\text{node values}}, \underbrace{(\boldsymbol{b}^{+})^{\mathrm{T}} \tilde{\boldsymbol{u}}^{(\beta)}}_{\text{left boundary}}, \underbrace{(\boldsymbol{b}^{-})^{\mathrm{T}} \tilde{\boldsymbol{u}}^{(\beta)}}_{\text{right boundary}} \} \subset \{\text{permissible states}\}, \tag{31}$$

where

$$\tilde{\boldsymbol{u}}^{(\beta)} = \boldsymbol{R}^{(\beta)} \, \bar{\boldsymbol{u}} = \left(\frac{1 - \beta}{n} \, \mathbf{1} \otimes \mathbf{1}^{\mathrm{T}} + \beta \, \boldsymbol{R} \right) \bar{\boldsymbol{u}} = (1 - \beta) \, \mathbf{1} \, \langle \boldsymbol{u} \rangle + \beta \, \tilde{\boldsymbol{u}}, \tag{32}$$

which shows that the extended reconstruction operator preserves the original element average, that is

$$\langle \tilde{\boldsymbol{u}}^{(\beta)} \rangle = \langle \tilde{\boldsymbol{u}} \rangle.$$

Remark 5 The expanded reconstruction operator fulfills the following relationships:

$$\boldsymbol{\omega}^{\mathrm{T}} \, \boldsymbol{R}^{(\beta)} = \boldsymbol{\omega}^{\mathrm{T}} \, \boldsymbol{R} = \frac{1}{n} \, \mathbf{1}^{\mathrm{T}}. \tag{33}$$

Proof We begin with the second relation. From Eq. (26), we know

$$\mathbf{1}^{\mathrm{T}} \, \boldsymbol{P} = n \, \boldsymbol{\omega}^{\mathrm{T}},$$
$$\mathbf{1}^{\mathrm{T}} \, \boldsymbol{P} \, \boldsymbol{P}^{-1} = n \, \boldsymbol{\omega}^{\mathrm{T}} \boldsymbol{P}^{-1} = n \, \boldsymbol{\omega}^{\mathrm{T}} \boldsymbol{R},$$
$$\frac{1}{n} \, \mathbf{1}^{\mathrm{T}} = \boldsymbol{\omega}^{\mathrm{T}} \boldsymbol{R}.$$

The first relation follows as

$$\boldsymbol{\omega}^{\mathrm{T}} \, \boldsymbol{R}^{(\beta)} = \frac{1 - \beta}{n} \, \underbrace{\boldsymbol{\omega}^{\mathrm{T}} \, \mathbf{1}}_{=1} \otimes \mathbf{1}^{\mathrm{T}} + \beta \, \underbrace{\boldsymbol{\omega}^{\mathrm{T}} \, \boldsymbol{R}}_{=\frac{1}{n} \mathbf{1}^{\mathrm{T}}} = \frac{1}{n} \, \mathbf{1}^{\mathrm{T}} = \boldsymbol{\omega}^{\mathrm{T}} \, \boldsymbol{R}.$$

The specific calculation of the parameter β depends on the permissibility constraints of the conservation law at hand. In Sect. 3.1, we solve the compressible Euler equations where the density ρ and pressure p are positive values. Based on [64], we calculate the squeezing parameter β as

$$\beta = \min \left(\frac{\langle \rho \rangle - \epsilon}{\langle \rho \rangle - \rho_{\min}}, \frac{\langle p \rangle - \epsilon}{\langle p \rangle - p_{\min}}, 1 \right), \tag{34}$$

where $\langle \rho \rangle$ and $\langle p \rangle$ are the element averages (29), and $\rho_{\min}$ and $p_{\min}$ are the minimum values of the unlimited polynomial given in (31) and $\epsilon = \min(10^{-20}, \langle \rho \rangle, \langle p \rangle)$.

Remark 6 We observed that the indispensable positivity limiting can have a potential negative impact on the robustness of the high-order DG scheme, especially for high polynomial degrees. It seems that huge artificial jumps and gradients can be generated at element interfaces. To alleviate this problem, it is advisable to only allow small squeezing margins between 5% and 10%. If, for example, the squeezing parameter β falls below 0.95, the reconstructed high-order polynomial is decided un-salvageable and the next lower order scheme is considered.

2.5 Single-Level Blending

We first focus on the case where only two different discretizations per element q are blended: a single-level blending of the low-order FV scheme and the high-order DG scheme,

$$\partial_t u_q = (1 - \alpha_q)\, \partial_t u_q^{(\text{low})} + \alpha_q\, \partial_t u_q^{(\text{high})}.$$

Note that we do not blend the solutions, but directly the discretizations themselves, i.e., the right-hand sides which we denote by $\partial_t u_q$. If both schemes operate with different data representations, appropriate transformations ensure the compatibility during the blending process. In our case, we use the projection operator $\boldsymbol{P}^{(N \to N)}$ introduced in Sect. 2.3 to transfer the nodal output of the DG operator to subcell mean values. The inputs for the DG scheme on the other hand, i.e., the nodal coefficients $\tilde{\boldsymbol{u}}_q \in \mathbb{R}^N$, are reconstructed from the given mean values $\bar{\boldsymbol{u}}_q \in \mathbb{R}^N$ as described in Sect. 2.4. The naïve single-level blended discretization reads as

$$\partial_t \bar{\boldsymbol{u}}_q = (1 - \alpha_q)\, \partial_t \bar{\boldsymbol{u}}_q^{(\text{FV})} + \alpha_q\, \boldsymbol{P}\, \partial_t \tilde{\boldsymbol{u}}_q^{(\text{DG})}.$$

Inserting (8) and (23) gives

$$\partial_t \bar{\boldsymbol{u}}_q = (1 - \alpha_q)\, \frac{-N}{\Delta x_q}\, \boldsymbol{\Delta}\, \boldsymbol{f}_q^{*(\text{FV})} + \alpha_q\, \frac{1}{\Delta x_q}\boldsymbol{P}\, \left(\boldsymbol{D}\, \tilde{\boldsymbol{f}}_q - \boldsymbol{B}\, \boldsymbol{f}_q^{*(\text{DG})} \right). \tag{35}$$

The fundamental issue of the naïve single-level blending discretization (35) is that it is not conservative if the blending parameter α_q varies from element to element. It turns out that the direct blending of the surface fluxes of both discretizations, namely $\boldsymbol{f}_q^{*(\text{FV})}$ and $\boldsymbol{f}_q^{*(\text{DG})}$, leads to a non-conservative balance of the net fluxes across element interfaces. To remedy this issue, the goal is to define unique surface fluxes common to both, the low-order and the high-order discretization, such that they can be directly blended. We thus compute a blending of the reconstructed interface values to evaluate a unique common surface flux $\boldsymbol{f}_q^*$. First, we define two interface vectors of length $N + 1$ as follows:

$$\boldsymbol{u}_q^+ := \big(\underbrace{u_{q-\frac{1}{2}}^+, (\bar{u}_1)_q, \cdots, (\bar{u}_{N-1})_q, u_{q+\frac{1}{2}}^+}_{\substack{\text{prolongated states at}\\\text{FV subcell interfaces}}} \big)^{\mathrm{T}} \text{ and } \boldsymbol{u}_q^- := \big(\underbrace{u_{q-\frac{1}{2}}^-, (\bar{u}_2)_q, \cdots, (\bar{u}_N)_q, u_{q+\frac{1}{2}}^-}_{\substack{\text{prolongated states at}\\\text{FV subcell interfaces}}} \big)^{\mathrm{T}},$$

$$\tag{36}$$

where the outermost interface values are given by

$$u_{q-\frac{1}{2}}^- = (1 - \alpha_q)\,(\bar{u}_1)_q + \alpha_q\, \tilde{u}_q^- \quad \text{and} \quad u_{q+\frac{1}{2}}^+ = (1 - \alpha_q)\,(\bar{u}_N)_q + \alpha_q\, \tilde{u}_q^+. \tag{37}$$

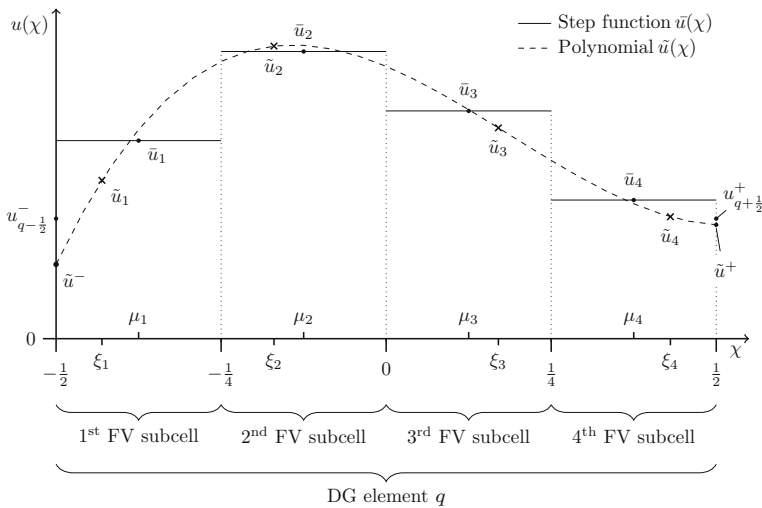A concrete example for the blended boundary values $u_{q\pm\frac{1}{2}}^\pm$ is shown in Fig. 3.

Commun. Appl. Math. Comput. (2023) 5:679–721

689

Fig. 3 Schematic of four ($N = 4$) mean values and the reconstructed polynomial on the reference element. The cell boundary values $u^-_{q-\frac{1}{2}}$ and $u^+_{q+\frac{1}{2}}$ were calculated with (37) where we assumed a blending factor of $\alpha = 0.75$

The common surface flux $\boldsymbol{f}^*_q$ is then evaluated with the interface vectors $\boldsymbol{u}^+_q$ and $\boldsymbol{u}^-_q$ as

$$\boldsymbol{f}^*_q = f^* \left(\boldsymbol{u}^+_q, \boldsymbol{u}^-_q \right) \in \mathbb{R}^{N+1}. \tag{38}$$

To make the DG surface operator $\boldsymbol{B}$ compatible with $\boldsymbol{f}^*_q$, we adapt the notation by inserting an additional column of zeros:

$$\boldsymbol{B}^{(0)} = \begin{pmatrix} B_{11} & 0 & \cdots & 0 & \cdots & 0 & B_{1N} \\ B_{21} & 0 & \cdots & 0 & \cdots & 0 & B_{2N} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\ B_{N1} & 0 & \cdots & 0 & \cdots & 0 & B_{NN} \end{pmatrix} \in \mathbb{R}^{N \times (N+1)}. \tag{39}$$

Corollary 1 *Contracting the DG surface operator* $\boldsymbol{B}^{(0)}$ *with the quadrature weights* $\boldsymbol{\omega}$ *is equivalent to the contraction of the FV surface operator, that is*

$$\boldsymbol{\omega}^{\mathrm{T}} \boldsymbol{B}^{(0)} = \mathbf{1}^{\mathrm{T}} \boldsymbol{\Delta}^{(\mathrm{S})}. \tag{40}$$

Proof We expand the left side of (40) and write

$$\boldsymbol{\omega}^{\mathrm{T}} \boldsymbol{B}^{(0)} = \left(-\sum_{i=1}^{N} \omega_i \frac{\ell_i(-\frac{1}{2})}{\omega_i}, 0, \cdots, 0, \sum_{i=1}^{N} \omega_i \frac{\ell_i(\frac{1}{2})}{\omega_i} \right)$$

$$= \left(-\sum_{i=1}^{N} \ell_i(-\tfrac{1}{2}), 0, \cdots, 0, \sum_{i=1}^{N} \ell_i(\tfrac{1}{2}) \right) = (-1, 0, \cdots, 0, 1) = \mathbf{1}^{\mathrm{T}} \boldsymbol{\Delta}^{(\mathrm{S})}.$$

We replace $\boldsymbol{f}^{*(\mathrm{FV})}_q$ and $\boldsymbol{f}^{*(\mathrm{DG})}_q$ in (35) with $\boldsymbol{f}^*_q$, and additionally, write the DG operators compactly, including the projection operator, as

$$\hat{\boldsymbol{D}} = \boldsymbol{P} \boldsymbol{D} \quad \text{and} \quad \hat{\boldsymbol{B}} = \boldsymbol{P} \boldsymbol{B}^{(0)}. \tag{41}$$

The final form of the single-level blended discretization is

$$\partial_t \bar{\boldsymbol{u}}_q = (1 - \alpha_q) \frac{-N}{\Delta x} \boldsymbol{\Delta} \, \boldsymbol{f}_q^* + \alpha_q \frac{1}{\Delta x} \left(\hat{\boldsymbol{D}} \, \tilde{\boldsymbol{f}}_q - \hat{\boldsymbol{B}} \, \boldsymbol{f}_q^* \right). \tag{42}$$

Remark 7 It is easy to see that with $\alpha_q = 0$ for all elements q, the pure subcell FV discretization is recovered, and with $\alpha_q = 1$ for all elements, the blending scheme recovers the pure high-order DG method.

Lemma 1 *Given arbitrary blending factors $\alpha_q \in \mathbb{R}$ for each element q, then the blending scheme* (42) *is conservative.*

Proof We discretely integrate the single-level blended discretization over all elements q with a total number of Q:

$$\sum_q^Q \Delta x_q \, \mathbf{1}^{\mathrm{T}} \partial_t \bar{\boldsymbol{u}}_q = \sum_q^Q \Delta x_q \, \mathbf{1}^{\mathrm{T}} \left[(1 - \alpha_q) \frac{-N}{\Delta x_q} \boldsymbol{\Delta} \, \boldsymbol{f}_q^* + \alpha_q \frac{1}{\Delta x_q} \left(\hat{\boldsymbol{D}} \, \tilde{\boldsymbol{f}}_q - \hat{\boldsymbol{B}} \, \boldsymbol{f}_q^* \right) \right].$$

With (9) and (41), we get

$$\sum_q^Q \Delta x_q \, \mathbf{1}^{\mathrm{T}} \partial_t \bar{\boldsymbol{u}}_q = \sum_q^Q \mathbf{1}^{\mathrm{T}} \left[-(1 - \alpha_q) \, N \left(\boldsymbol{\Delta}^{(\mathrm{V})} + \boldsymbol{\Delta}^{(\mathrm{S})} \right) \boldsymbol{f}_q^* + \alpha_q \left(\boldsymbol{P} \, \boldsymbol{D} \, \tilde{\boldsymbol{f}}_q - \boldsymbol{P} \, \boldsymbol{B}^{(0)} \, \boldsymbol{f}_q^* \right) \right].$$

With properties (10), (24), and (26) the volume terms vanish, that is

$$\mathbf{1}^{\mathrm{T}} \boldsymbol{\Delta}^{(\mathrm{V})} = \mathbf{0}^{\mathrm{T}} \quad \text{and} \quad \mathbf{1}^{\mathrm{T}} \boldsymbol{P} \, \boldsymbol{D} = N \, \boldsymbol{\omega}^{\mathrm{T}} \boldsymbol{D} = \mathbf{0}^{\mathrm{T}}.$$

We reformulate the DG surface term with (26) and (40) as

$$\mathbf{1}^{\mathrm{T}} \boldsymbol{P} \, \boldsymbol{B}^{(0)} = N \, \boldsymbol{\omega}^{\mathrm{T}} \boldsymbol{B}^{(0)} = N \, \mathbf{1}^{\mathrm{T}} \boldsymbol{\Delta}^{(\mathrm{S})}.$$

The expansion of the telescopic sum only leaves the outermost surface fluxes:

$$\sum_q^Q \left[-(1 - \alpha_q) \, N \, \mathbf{1}^{\mathrm{T}} \boldsymbol{\Delta}^{(\mathrm{S})} - \alpha_q \, N \, \mathbf{1}^{\mathrm{T}} \boldsymbol{\Delta}^{(\mathrm{S})} \right] \boldsymbol{f}_q^* = \sum_q^Q -N \, \mathbf{1}^{\mathrm{T}} \boldsymbol{\Delta}^{(\mathrm{S})} \, \boldsymbol{f}_q^*$$

$$= -N \left(\left(-f_1^* \right)_1 + \left(f_{N+1}^* \right)_Q \right) = 0,$$

which represent the change due to physical boundary conditions. For instance in case of periodic boundary conditions, these two fluxes would cancel to zero.

This concludes the description of the single-level blending scheme.

2.6 Multi-level Blending

As mentioned in the introduction, the general idea is the construction of a hierarchy of discretizations, with lower order DG schemes on sub-elements. In principle, it is thus possible to define a multi-level blending discretization, where the low-order FV scheme is blended with DG variants of different approximation orders. This multi-level extension is driven by the desire to retain as much "high order DG accuracy" as possible, especially on a sub-element level.

For the discussion, we consider a specific example setup with a DG element with polynomial degree $N_p = 7$. The data of this DG element are collected in form of $N = 8$ mean

values on a regular subcell grid. Our goal is to blend a first-order subcell FV scheme, a second- and fourth-order sub-element DG scheme, and the full eighth-order DG scheme. The construction of the individual schemes follows the discussion in Sects. 2.1, 2.2, and 2.5. In this section, the extension to a multi-level blending approach is presented.

We get the first data interpretation and its corresponding discretization by directly using the mean values of element q as a first-order approximation space with no reconstruction at all:

$$\bar{\boldsymbol{u}}_q^{\mathcal{O}(1)} := \bar{\boldsymbol{u}}_q = (\bar{u}_1, \bar{u}_2, \cdots, \bar{u}_8)_q^{\mathrm{T}}.$$

Next, we interpret the eight mean values as four second-order DG sub-elements. The reconstruction matrix $\boldsymbol{R}^{\mathcal{O}(2)} \in \mathbb{R}^{2 \times 2}$ transforms two adjacent mean values to two nodal values:

$$\tilde{\boldsymbol{u}}_q^{\mathcal{O}(2)} = \left(\underbrace{(\tilde{u}_1)_1^{\mathcal{O}(2)}, (\tilde{u}_2)_1^{\mathcal{O}(2)}}_{\mathcal{O}(2)-\text{sub-element}}, \cdots, \underbrace{(\tilde{u}_1)_4^{\mathcal{O}(2)}, (\tilde{u}_2)_4^{\mathcal{O}(2)}}_{\mathcal{O}(2)-\text{sub-element}} \right)_q^{\mathrm{T}} = \left[\mathbb{1}_4 \otimes \boldsymbol{R}^{\mathcal{O}(2)} \right] \bar{\boldsymbol{u}}_q. \quad (43)$$

The fourth- and eighth-order approximations are constructed analogously as

$$\tilde{\boldsymbol{u}}_q^{\mathcal{O}(4)} = \left(\underbrace{(\tilde{u}_1)_1^{\mathcal{O}(4)}, \cdots, (\tilde{u}_4)_1^{\mathcal{O}(4)}}_{\mathcal{O}(4)-\text{sub-element}}, \underbrace{(\tilde{u}_1)_2^{\mathcal{O}(4)}, \cdots, (\tilde{u}_4)_2^{\mathcal{O}(4)}}_{\mathcal{O}(4)-\text{sub-element}} \right)_q^{\mathrm{T}} = \left[\mathbb{1}_2 \otimes \boldsymbol{R}^{\mathcal{O}(4)} \right] \bar{\boldsymbol{u}}_q, \quad (44)$$

and

$$\tilde{\boldsymbol{u}}_q^{\mathcal{O}(8)} = \left(\underbrace{\tilde{u}_1^{\mathcal{O}(8)}, \cdots, \tilde{u}_8^{\mathcal{O}(8)}}_{\mathcal{O}(8)-\text{sub-element}} \right)_q^{\mathrm{T}} = \left[\mathbb{1}_1 \otimes \boldsymbol{R}^{\mathcal{O}(8)} \right] \bar{\boldsymbol{u}}_q \quad (45)$$

with $\boldsymbol{R}^{\mathcal{O}(4)} \in \mathbb{R}^{4 \times 4}$ and $\boldsymbol{R}^{\mathcal{O}(8)} \in \mathbb{R}^{8 \times 8}$. Here, we used the Kronecker product $\otimes$ in conjunction with the identity matrix $\mathbb{1}_n = \mathrm{diag}(1, \cdots, 1) \in \mathbb{R}^{n \times n}$ to generate appropriate block-diagonal matrices. Figure A1 in Appendix A illustrates the hierarchy of the approximation spaces for this example.

We correspond the approximation spaces with the respective discretizations:

$$\begin{cases} \partial_t \bar{\boldsymbol{u}}_q^{\mathcal{O}(1)} = -\frac{8}{\Delta x_q} \boldsymbol{\Delta} \boldsymbol{f}^*, \\ \partial_t \bar{\boldsymbol{u}}_q^{\mathcal{O}(2)} = \frac{4}{\Delta x_q} \left\{ \left[\mathbb{1}_4 \otimes \hat{\boldsymbol{D}}^{\mathcal{O}(2)} \right] \tilde{\boldsymbol{f}}_q^{\mathcal{O}(2)} - \left[\mathbb{1}_4 \otimes \hat{\boldsymbol{B}}^{\mathcal{O}(2)} \right]^{(0)} \boldsymbol{f}_q^* \right\}, \\ \partial_t \bar{\boldsymbol{u}}_q^{\mathcal{O}(4)} = \frac{2}{\Delta x_q} \left\{ \left[\mathbb{1}_2 \otimes \hat{\boldsymbol{D}}^{\mathcal{O}(4)} \right] \tilde{\boldsymbol{f}}_q^{\mathcal{O}(4)} - \left[\mathbb{1}_2 \otimes \hat{\boldsymbol{B}}^{\mathcal{O}(4)} \right]^{(0)} \boldsymbol{f}_q^* \right\}, \\ \partial_t \bar{\boldsymbol{u}}_q^{\mathcal{O}(8)} = \frac{1}{\Delta x_q} \left\{ \left[\mathbb{1}_1 \otimes \hat{\boldsymbol{D}}^{\mathcal{O}(8)} \right] \tilde{\boldsymbol{f}}_q^{\mathcal{O}(8)} - \left[\mathbb{1}_1 \otimes \hat{\boldsymbol{B}}^{\mathcal{O}(8)} \right]^{(0)} \boldsymbol{f}_q^* \right\}, \end{cases} \quad (46)$$

where the DG volume fluxes are computed with the respective polynomial (sub-)element reconstructions of orders $n = \{2, 4, 8\}$:

$$\tilde{\boldsymbol{f}}_q^{\mathcal{O}(n)} = f\left(\tilde{\boldsymbol{u}}_q^{\mathcal{O}(n)} \right).$$

Similar to (39), the DG surface operators $\left[\mathbb{1}_{N/n} \otimes \hat{\boldsymbol{B}}^{\mathcal{O}(n)} \right]$ have to be slightly adapted to be compatible with the common surface flux $\boldsymbol{f}^*$. We list the modified boundary evaluation matrices in Appendix A. Finally, all four candidate discretizations are blended starting from

the lowest order up to the highest order discretization:

$$
\begin{cases}
\partial_t \bar{\boldsymbol{u}}_q''' := \partial_t \bar{\boldsymbol{u}}_q^{\mathcal{O}(1)}, \\
\hookrightarrow \ \partial_t \bar{\boldsymbol{u}}_q'' = \left(\mathbf{1} - \boldsymbol{\alpha}_q^{\mathcal{O}(2)}\right) \odot \partial_t \bar{\boldsymbol{u}}_q''' + \boldsymbol{\alpha}_q^{\mathcal{O}(2)} \odot \partial_t \bar{\boldsymbol{u}}_q^{\mathcal{O}(2)}, \\
\hookrightarrow \ \partial_t \bar{\boldsymbol{u}}_q' = \left(\mathbf{1} - \boldsymbol{\alpha}_q^{\mathcal{O}(4)}\right) \odot \partial_t \bar{\boldsymbol{u}}_q'' + \boldsymbol{\alpha}_q^{\mathcal{O}(4)} \odot \partial_t \bar{\boldsymbol{u}}_q^{\mathcal{O}(4)}, \\
\hookrightarrow \ \partial_t \bar{\boldsymbol{u}}_q = \left(\mathbf{1} - \boldsymbol{\alpha}_q^{\mathcal{O}(8)}\right) \odot \partial_t \bar{\boldsymbol{u}}_q' + \boldsymbol{\alpha}_q^{\mathcal{O}(8)} \odot \partial_t \bar{\boldsymbol{u}}_q^{\mathcal{O}(8)},
\end{cases}
\tag{47}
$$

where $\odot$ denotes the component-wise multiplication with the vector of blending parameters. Each DG sub-element computes its own blending factor according to the local data, i.e., we get four blending factors for the $\mathcal{O}(2)$ variant, two blending factors for $\mathcal{O}(4)$, and one for $\mathcal{O}(8)$:

$$
\boldsymbol{\alpha}_q^{\mathcal{O}(2)} = \left(\alpha_1^{\mathcal{O}(2)}, \alpha_2^{\mathcal{O}(2)}, \alpha_3^{\mathcal{O}(2)}, \alpha_4^{\mathcal{O}(2)}\right)_q^{\mathrm{T}} \otimes (1, 1)^{\mathrm{T}},
$$

$$
\boldsymbol{\alpha}_q^{\mathcal{O}(4)} = \left(\alpha_1^{\mathcal{O}(4)}, \alpha_2^{\mathcal{O}(4)}\right)_q^{\mathrm{T}} \otimes (1, 1, 1, 1)^{\mathrm{T}},
$$

$$
\boldsymbol{\alpha}_q^{\mathcal{O}(8)} = \left(\alpha^{\mathcal{O}(8)}\right)_q^{\mathrm{T}} \otimes (1, 1, 1, 1, 1, 1, 1, 1)^{\mathrm{T}}.
$$

The actual strategy for the blending factors is described in Sect. 2.7.

Again, as discussed in the case of a single-blending approach in Sect. 2.5, a special care is needed to preserve the conservation of the final discretization. Expanding on the idea in the single-level case, we aim to compute unique surface fluxes for each discretization with a uniquely defined interface value at sub-element or subcell interfaces. We evaluate all high-order DG polynomials at the FV subcell interfaces $\boldsymbol{u}_q^{\pm \mathcal{O}(n)} \in \mathbb{R}^N$:

$$
\boldsymbol{u}_q^{\pm \mathcal{O}(1)} := \bar{\boldsymbol{u}}_q^{\mathcal{O}(1)}, \qquad\qquad \boldsymbol{u}_q^{\pm \mathcal{O}(2)} = \left[\mathbb{1}_4 \otimes \boldsymbol{I}^{\pm \mathcal{O}(2)}\right] \tilde{\boldsymbol{u}}_q^{\mathcal{O}(2)},
$$

$$
\boldsymbol{u}_q^{\pm \mathcal{O}(4)} = \left[\mathbb{1}_2 \otimes \boldsymbol{I}^{\pm \mathcal{O}(4)}\right] \tilde{\boldsymbol{u}}_q^{\mathcal{O}(4)}, \qquad \boldsymbol{u}_q^{\pm \mathcal{O}(8)} = \left[\mathbb{1}_1 \otimes \boldsymbol{I}^{\pm \mathcal{O}(8)}\right] \tilde{\boldsymbol{u}}_q^{\mathcal{O}(8)},
$$

where the interpolation operator $\boldsymbol{I}^{\pm \mathcal{O}(n)} \in \mathbb{R}^{n \times n}$ prolongates to the embedded FV subcell interfaces $\mu_{i\pm 1/2}^{\mathcal{O}(n)}$ of the respective DG sub-element, that is

$$
\left(\boldsymbol{I}^{\pm \mathcal{O}(n)}\right)_{ij} = \ell_j^{\mathcal{O}(n)}\left(\mu_{i\pm 1/2}^{\mathcal{O}(n)}\right), \quad i, j = 1, \cdots, n.
\tag{48}
$$

In Fig. A1 in Appendix A, two representatives of $\mu_{i\pm 1/2}^{\mathcal{O}(n)}$ are shown which are aligned at the dotted vertical lines, indicating the interfaces of the FV subcells.

We start with the first-order interpolation and stack on top of the next levels of orders via convex blending:

$$
\left(\boldsymbol{u}_q^{\pm}\right)'''' := \boldsymbol{u}_q^{\pm \mathcal{O}(1)},
$$

$$
\hookrightarrow \ \left(\boldsymbol{u}_q^{\pm}\right)''' = \left[\mathbf{1} - \boldsymbol{\alpha}_q^{\mathcal{O}(2)}\right] \odot \left(\boldsymbol{u}_q^{\pm}\right)'''' + \boldsymbol{\alpha}_q^{\mathcal{O}(2)} \odot \boldsymbol{u}_q^{\pm \mathcal{O}(2)},
$$

$$
\hookrightarrow \ \left(\boldsymbol{u}_q^{\pm}\right)'' = \left[\mathbf{1} - \boldsymbol{\alpha}_q^{\mathcal{O}(4)}\right] \odot \left(\boldsymbol{u}_q^{\pm}\right)''' + \boldsymbol{\alpha}_q^{\mathcal{O}(4)} \odot \boldsymbol{u}_q^{\pm \mathcal{O}(4)},
$$

$$
\hookrightarrow \ \left(\boldsymbol{u}_q^{\pm}\right)' = \left[\mathbf{1} - \boldsymbol{\alpha}_q^{\mathcal{O}(8)}\right] \odot \left(\boldsymbol{u}_q^{\pm}\right)'' + \boldsymbol{\alpha}_q^{\mathcal{O}(8)} \odot \boldsymbol{u}_q^{\pm \mathcal{O}(8)}.
$$

To arrive at a complete vector of interface values, we append the element interface values from the left and right neighbors $q - 1$ and $q + 1$:

$$\boldsymbol{u}_q^+ = \left(\overbrace{\left(u_N^+ \right)_{q-1}'}^{\text{left neighbor}}, \quad \overbrace{\left(u_1^+, \cdots, u_N^+ \right)_q'}^{\text{internal boundaries}} \right)^{\mathrm{T}},$$

$$\boldsymbol{u}_q^- = \left(\underbrace{\left(u_1^-, \cdots, u_N^- \right)_q'}_{\text{internal boundaries}}, \quad \underbrace{\left(u_1^- \right)_{q+1}'}_{\text{right neighbor}} \right)^{\mathrm{T}}.$$

These new common interface values allow us to evaluate a common surface flux (38) at each interface, which is again the key to preserve conservation in the final discretization. This concludes the description of the multi-level blending scheme.

2.7 Calculation of the Blending Factor α

A good shock indicator for high-order methods is supposed to recognize discontinuities, such as weak and strong shocks, in the solution early on and mark the affected elements for proper shock capturing. On the other hand, the indicator should avoid to flag non-shock-related fluid features such as shear layers and turbulent flows. There are numerous indicators available for DG schemes [2, 13, 14, 47, 59, 65].

In this work, we want to construct an a-priori indicator which relies on the readily available information within each element provided by the multi-level blending framework introduced in Sect. 2.6. The principal idea is to compare a measure of smoothness of the different order reconstructions with each other. Smooth, well-resolved flows are expected to yield rather similar solution profiles compared to data that contain strong variations.

The smoothness measure $\sigma_q^{\mathcal{O}(n)}$ within element q of the nth-order reconstruction $\tilde{u}_q^{\mathcal{O}(n)}$ is inferred from the L^1-norm of its first derivative. For example, if we want to calculate the blending factor $\alpha_q^{\mathcal{O}(8)}$ for the eighth order, we compute

$$\sigma_q^{\mathcal{O}(8)} = \int_{-\frac{1}{2}}^{\frac{1}{2}} \left| \partial_\xi \tilde{u}_q^{\mathcal{O}(8)}(\xi) \right| \mathrm{d}\xi \tag{49}$$

and

$$\sigma_q^{\mathcal{O}(4)} = \left(\sigma_1^{\mathcal{O}(4)} \right)_q + \left(\sigma_2^{\mathcal{O}(4)} \right)_q = \int_{-\frac{1}{2}}^{0} \left| \partial_\xi \left(\tilde{u}_1^{\mathcal{O}(4)} \right)_q (\xi) \right| \mathrm{d}\xi + \int_{0}^{\frac{1}{2}} \left| \partial_\xi \left(\tilde{u}_2^{\mathcal{O}(4)} \right)_q (\xi) \right| \mathrm{d}\xi$$

utilizing information from the top-level eighth-order element and its two fourth-order sub-elements. The integrals are evaluated with appropriate quadrature rules. Then, the blending factor is calculated by comparing the smoothness of the different orders as

$$\alpha_q^{\mathcal{O}(8)} = 1 - \mathrm{cutoff}\left(0, \tau_A \left(\frac{\left| \sigma_q^{\mathcal{O}(8)} - \sigma_q^{\mathcal{O}(4)} \right|}{\max \left(\sigma_q^{\mathcal{O}(4)}, \sigma_q^{\mathcal{O}(8)}, 1 \right)} - \tau_S \right), 1 \right), \tag{50}$$

where $\mathrm{cutoff}(x_{\mathrm{lower}}, x, x_{\mathrm{upper}}) = \min(\max(x_{\mathrm{lower}}, x), x_{\mathrm{upper}})$ and with two parameters $\tau_A, \tau_S > 0$. The design of (50) ensures that $\alpha_q^{\mathcal{O}(n)}$ is always within the unit interval, and that for very small $\sigma_q^{\mathcal{O}(n)}$, the indicator does not get hypersensitive due to floating point truncation. If not noted otherwise, we set the amplification parameter to $\tau_A = 100$ and the

sensitivity parameter $\tau_S = 0.05$. The latter parameter has the biggest influence on the behavior of the indicator and its value demarcates the lower bound where it does not interfere with the convergence tests conducted in Sect. 3. To capture all troubling flow features, we independently apply the indicator (50) on all primitive state variables and select the smallest of the resulting blending factors. The calculations of the two fourth-order blending factors $\left(\alpha_1^{\mathcal{O}(4)}\right)_q$ and $\left(\alpha_2^{\mathcal{O}(4)}\right)_q$ are done within each sub-element independently together with their respective second-order smoothness measures $\left(\sigma_s^{\mathcal{O}(2)}\right)_q$, $s = 1, \cdots, 4$.

For piecewise linear polynomials, the indicator (50) is not applicable. Instead, we directly use the squeezing parameter β obtained from the positivity limiter in (30), i.e., $\alpha_s^{\mathcal{O}(2)} := \beta_s^{\mathcal{O}(2)}$, $s = 1, \cdots, 4$.

Remark 8 The indicator (50) is also used for the single-level blending scheme.

2.8 Sketch of the Algorithm

In the last part of this section, a sketch of the algorithm for the blending framework is presented. We present a general outline of the necessary steps for the 1D single-blending scheme evolved with a multi-stage Runge-Kutta time-stepping method. We enter at the beginning of a Runge-Kutta cycle and do the following.

(I) Reconstruct the polynomial $\tilde{\boldsymbol{u}}_q$ from the given mean values $\bar{\boldsymbol{u}}_q$ for each element q as in (28).

(II) If the reconstructed polynomial $\tilde{\boldsymbol{u}}_q$ contains non-permissible states, see (31), then calculate the limited version $\tilde{\boldsymbol{u}}_q^{(\beta)}$ as in (30).

(III) If the squeezing parameter β_q is below β_L, then set $\alpha_q := 0$ else compute the blending factor α_q via (50) from the unlimited polynomial $\tilde{\boldsymbol{u}}_q$.

(IV) Compute the boundary values $u_q^{\pm}$ via (37) and exchange alongside zone boundaries in case of distributed computing.

(V) Determine the common surface flux $\boldsymbol{f}_q^*$ via (38).

(VI) Calculate the right-hand side $\partial_t \bar{\boldsymbol{u}}_q$:

- if the blending factor α_q above α_H, then compute $\partial_t \bar{\boldsymbol{u}}_q$ with the DG-only scheme;
- else if the blending factor α_q below α_L, then compute $\partial_t \bar{\boldsymbol{u}}_q$ with the FV-only scheme;
- else compute $\partial_t \bar{\boldsymbol{u}}_q$ with the single-level blending scheme (42).

(VII) Forward in time to the next Runge-Kutta stage and return to step (I).

The switching thresholds are set to $\alpha_H := 0.99$ and $\alpha_L := 0.01$ and the limiter threshold to $\beta_L := 0.95$. Note that the algorithm only applies the blending procedure where necessary to maintain the overall performance of the scheme.

This concludes the presentation of the 1D blending scheme. The description of the blending scheme on 3D Cartesian meshes as well as the algorithm outline for the multi-level blending scheme can be found in Appendixes A.3 and A.4.

3 Validation

For the computational investigations, the multi-level algorithm with the explicit SSP-RK(5, 4) time integrator [58] is implemented in a Fortran-2008 prototype code with a hybrid parallelization strategy based on MPI and OpenMP. Management of the AMR and load balancing

is provided by P4EST, a highly efficient Octree library [4]. The maximal time-step for dimensions $d = 1, 2, 3$ is estimated by the CFL condition:

$$\Delta t := \mathrm{CFL} \min_{q} \frac{\Delta x_q}{2^{d-1}(2\,N - 1)\,\bar{\lambda}_{q,\max}}, \tag{51}$$

where $\mathrm{CFL} := 0.8$, $N := 8$ is the number of mean values in each direction of the element q, and $\bar{\lambda}_{q,\max}$ is an estimate of the maximum eigenvalue given in (55). For the numerical interface fluxes f^*, we use the Harten-Lax-Leer (HLL) approximate Riemann solver [28] with Einfeldt signal speed estimates [15].

3.1 Governing Equations

We consider the compressible Euler equations:

$$\begin{cases} \partial_t \rho + \nabla \cdot (\rho\,\boldsymbol{v}) = 0, \\ \partial_t (\rho\,\boldsymbol{v}) + \nabla \cdot \left(\rho\,\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}} + p\,\mathbb{1}\right) = \boldsymbol{0}, \\ \partial_t \mathcal{E} + \nabla \cdot \left(\boldsymbol{v}\,(\mathcal{E} + p)\right) = 0, \end{cases} \tag{52}$$

with the vector of conserved quantities $\boldsymbol{u} = (\rho, \rho\,\boldsymbol{v}, \mathcal{E})^{\mathrm{T}}$, where ρ denotes the density, $\boldsymbol{v}$ the velocity, and $\mathcal{E}$ the total energy. We assume a perfect gas equation of state and compute the pressure as:

$$p(\boldsymbol{u}) = (\gamma - 1) \left[\mathcal{E} - \frac{\rho}{2}\,\boldsymbol{v}^{\mathrm{T}}\boldsymbol{v}\right]. \tag{53}$$

If not stated otherwise, we choose $\gamma = 1.4$. The set of permissible states is given by

$$\{\text{permissible states}\} = \{\forall\,\boldsymbol{u} \mid \rho > 0 \wedge p(\boldsymbol{u}) > 0\}. \tag{54}$$

For the CFL condition (51), the maximum eigenvalue is evaluated on all mean values $\bar{\boldsymbol{u}}_q$ of element q. Given dimension $d = \{1, 2, 3\}$, it reads as

$$\bar{\lambda}_{q,\max} = \max_{i,d} \left(\left|(\bar{v}_d)_i\right| + \sqrt{\gamma\,\frac{\bar{p}_i}{\bar{\rho}_i}}\right), \quad i = 1, \cdots, N^{d_{\max}}. \tag{55}$$

3.2 Convergence Test

We use the manufactured solution method [51] and validate the 3D multi-level blending scheme on a periodic cube of unit length ($L = 1$) where the resolution of the mesh is incrementally doubled. We define our manufactured solution in primitive state variables as

$$\begin{cases} \rho(t; x, y, z) = 1.0 + 0.35 \sin\left(\frac{2\,\pi\,(x-t)}{L}\right) + 0.24 \cos\left(\frac{2\,\pi\,(y-t)}{L}\right) + 0.1 \sin\left(\frac{2\,\pi\,(z-t)}{L}\right), \\ \boldsymbol{v}(t; x, y, z) = (0.1, 0.2, 0.3)^{\mathrm{T}}, \\ p(t; x, y, z) = 1.0 + 0.23 \cos\left(\frac{2\,\pi\,(x-t)}{L}\right) \\ \qquad\qquad + 0.19 \sin\left(\frac{2\,\pi\,(y-t)}{L}\right) + 0.2 \cos\left(\frac{2\,\pi\,(z-t)}{L}\right). \end{cases} \tag{56}$$

The final time of the simulation is $T = 2$ and the center of the domain is refined to introduce non-conforming interfaces in the computational domain. We determine the L^∞- and L^2-norms of the errors in the density and total energy. Tables 1, 2, 3, and 4 list the results for the first-, second-, fourth-, and eighth-order multi-level blending schemes. The initial conditions

Table 1 Experimental order of convergence of the first-order FV variant within the 3D multi-level blending framework

Resol. 2^r	Density $\|\epsilon\|_\infty$	$\|\epsilon\|_2$	Total energy $\|\epsilon\|_\infty$	$\|\epsilon\|_2$	Density EOC$_\infty$	EOC$_2$	Total energy EOC$_\infty$	EOC$_2$
16	5.98E−03	4.84E−03	1.34E−02	1.03E−02	n/a	n/a	n/a	n/a
32	3.51E−03	2.83E−03	7.91E−03	6.13E−03	0.77	0.77	0.76	0.75
64	2.21E−03	1.45E−03	4.30E−03	3.13E−03	0.67	0.97	0.88	0.97
128	1.42E−03	7.30E−04	2.26E−03	1.57E−03	0.63	0.99	0.93	0.99

Table 2 Experimental order of convergence of the second-order DG variant within the 3D multi-level blending framework

Resol. 2^r	Density $\|\epsilon\|_\infty$	$\|\epsilon\|_2$	Total energy $\|\epsilon\|_\infty$	$\|\epsilon\|_2$	Density EOC$_\infty$	EOC$_2$	Total energy EOC$_\infty$	EOC$_2$
16	1.46E−03	5.38E−04	2.21E−03	9.35E−04	n/a	n/a	n/a	n/a
32	4.59E−04	9.29E−05	6.63E−04	1.97E−04	1.67	2.53	1.74	2.25
64	1.11E−04	1.77E−05	1.48E−04	4.26E−05	2.04	2.39	2.17	2.21
128	2.63E−05	3.88E−06	3.09E−05	9.72E−06	2.08	2.19	2.25	2.13

Table 3 Experimental order of convergence of the fourth-order DG variant within the 3D multi-level blending framework

Resol. 2^r	Density $\|\epsilon\|_\infty$	$\|\epsilon\|_2$	Total energy $\|\epsilon\|_\infty$	$\|\epsilon\|_2$	Density EOC$_\infty$	EOC$_2$	Total energy EOC$_\infty$	EOC$_2$
16	1.10E−04	6.45E−05	3.20E−04	1.83E−04	n/a	n/a	n/a	n/a
32	7.86E−06	4.33E−06	2.19E−05	1.33E−05	3.80	3.90	3.87	3.79
64	5.18E−07	2.59E−07	1.31E−06	7.59E−07	3.92	4.07	4.07	4.13
128	3.45E−08	1.82E−08	7.75E−08	4.29E−08	3.91	3.83	4.07	4.15

and the source term of our manufactured solution (56) are in all cases evaluated and applied on the mean values via an appropriate quadrature rule to maintain high order. The results confirm that the discretizations behave as designed in this assessment.

3.3 Conservation Test

The goal in this assessment is to demonstrate that the multi-level-blending discretization is conservative for all choices of blending factors. To do so, we adapt the same setup as in Sect. 3.2, but deactivate the source term. The center of the domain is refined to introduce non-conforming interfaces in the computational domain. Additionally, as a stress test, the blending factors are randomly chosen and changed after each Runge-Kutta stage. As there are multiple blending factors at a given spatial location, we consider the following weighted blending factor:

$$\bar{\alpha} = \left\{ \left[\left(1 - \alpha^{\mathcal{O}(2)} \right) + 2\,\alpha^{\mathcal{O}(2)} \right] \left(1 - \alpha^{\mathcal{O}(4)} \right) + 4\,\alpha^{\mathcal{O}(4)} \right\} \left(1 - \alpha^{\mathcal{O}(8)} \right) + 8\,\alpha^{\mathcal{O}(8)}, \quad (57)$$

Table 4 Experimental order of convergence of the eighth-order DG variant within the 3D multi-level blending framework. Here, we set CFL := 0.1, as the time integration method is only fourth-order accurate

Resol. 2^r	Density		Total energy		Density		Total energy	
	$\|\|\epsilon\|\|_\infty$	$\|\|\epsilon\|\|_2$	$\|\|\epsilon\|\|_\infty$	$\|\|\epsilon\|\|_2$	EOC$_\infty$	EOC$_2$	EOC$_\infty$	EOC$_2$
16	2.77E−07	1.65E−07	7.77E−07	4.55E−07	n/a	n/a	n/a	n/a
32	2.77E−09	1.00E−09	6.72E−09	2.41E−09	6.65	7.36	6.85	7.56
64	1.73E−11	3.48E−12	4.15E−11	1.04E−11	7.32	8.17	7.34	7.86
128	8.06E−14	1.87E−14	1.85E−13	6.74E−14	7.74	7.54	7.81	7.27

to illustrate the distribution of the blending factors in Fig. 4. Note the limiting cases $\bar\alpha = 1$ for pure FV and $\bar\alpha = 8$ for the eighth-order DG scheme.

The simulation runs to $T = 300$ performing more than a quarter million timesteps. The result of the test is shown in Fig. 5 where we plot in log-scale the absolute value of the change of bulk $\partial_t \bar{\mathrm{u}}_{\text{total}}(t)$, integrated over the whole domain:

$$\partial_t \bar{\mathrm{u}}_{\text{total}}(t) = \sum_q^Q \frac{1}{|\Omega_q|} \int_{\Omega_q} \partial_t \bar{\mathrm{u}}(t)\, \mathrm{d}\Omega. \tag{58}$$

Q is the total number of elements and $|\Omega_q| = \Delta x_q \Delta y_q \Delta z_q$ is the volume per element. The results show that the conservation error lies within the range of 64 bit (double precision) floating point truncation and hence confirm that the multi-level blending discretization is fully conservative up to machine precision errors.

3.4 1D Shock Tube Problems

We validate the multi-level blending scheme with three well-established 1D problems, namely Sod, Lax, and Shu-Osher shock tubes. However, first, to gain insights into the individual contributions of the different schemes, we introduce a more intuitive reformulation of the blending factors $\alpha^{\mathcal{O}(n)}$. The multi-level blending operation (47) within element q can be interpreted as a linear superposition of four numerical schemes. For that, we collapse the following relations:

$$\partial_t \bar{\boldsymbol{u}}_q''' := \partial_t \bar{\boldsymbol{u}}_q^{\mathcal{O}(1)},$$
$$\hookrightarrow\ \partial_t \bar{\boldsymbol{u}}_q'' = \left(\boldsymbol{1} - \boldsymbol{\alpha}_q^{\mathcal{O}(2)}\right) \odot \partial_t \bar{\boldsymbol{u}}_q''' + \boldsymbol{\alpha}_q^{\mathcal{O}(2)} \odot \partial_t \bar{\boldsymbol{u}}_q^{\mathcal{O}(2)},$$
$$\hookrightarrow\ \partial_t \bar{\boldsymbol{u}}_q' = \left(\boldsymbol{1} - \boldsymbol{\alpha}_q^{\mathcal{O}(4)}\right) \odot \partial_t \bar{\boldsymbol{u}}_q'' + \boldsymbol{\alpha}_q^{\mathcal{O}(4)} \odot \partial_t \bar{\boldsymbol{u}}_q^{\mathcal{O}(4)},$$
$$\hookrightarrow\ \partial_t \bar{\boldsymbol{u}}_q = \left(\boldsymbol{1} - \boldsymbol{\alpha}_q^{\mathcal{O}(8)}\right) \odot \partial_t \bar{\boldsymbol{u}}_q' + \boldsymbol{\alpha}_q^{\mathcal{O}(8)} \odot \partial_t \bar{\boldsymbol{u}}_q^{\mathcal{O}(8)},$$

into one single term and define blending weights $\theta^{\mathcal{O}(n)}$, that is

$$\partial_t \bar{\boldsymbol{u}}_q = \boldsymbol{\theta}_q^{\mathcal{O}(1)} \odot \partial_t \bar{\boldsymbol{u}}_q^{\mathcal{O}(1)} + \boldsymbol{\theta}_q^{\mathcal{O}(2)} \odot \partial_t \bar{\boldsymbol{u}}_q^{\mathcal{O}(2)} + \boldsymbol{\theta}_q^{\mathcal{O}(4)} \odot \partial_t \bar{\boldsymbol{u}}_q^{\mathcal{O}(4)} + \boldsymbol{\theta}_q^{\mathcal{O}(8)} \odot \partial_t \bar{\boldsymbol{u}}_q^{\mathcal{O}(8)},$$
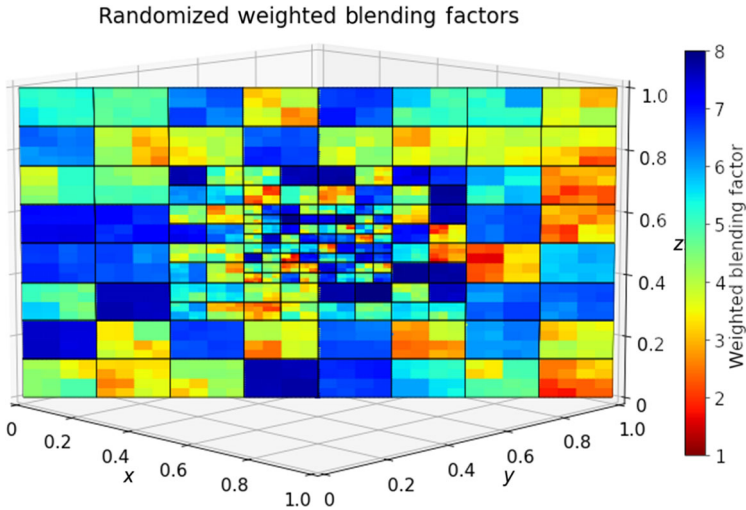
Fig. 4 Conservation test of the 3D multi-level blending scheme: slice through the computational domain showing weighted blending factor $\bar{\alpha}$ according to (57) on a Cartesian non-conforming mesh with refinement levels 3–5. The black lines depict the boundaries of the corresponding 8th order DG elements
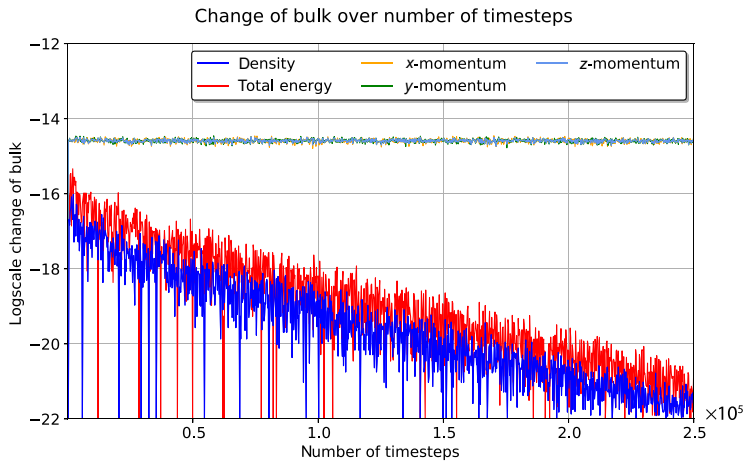

Fig. 5 Conservation test of the 3D multi-level blending scheme: evolution of the integrated change of bulk $\log_{10}(|\partial_t \tilde{u}_{\text{total}}|)$ (Eq. (58)) of each conservative state variable

where

$$
\begin{cases}
\theta_q^{\mathcal{O}(1)} := \left(1 - \alpha_q^{\mathcal{O}(8)}\right) \odot \left(1 - \alpha_q^{\mathcal{O}(4)}\right) \odot \left(1 - \alpha_q^{\mathcal{O}(2)}\right), \\
\theta_q^{\mathcal{O}(2)} := \left(1 - \alpha_q^{\mathcal{O}(8)}\right) \odot \left(1 - \alpha_q^{\mathcal{O}(4)}\right) \odot \alpha_q^{\mathcal{O}(2)}, \\
\theta_q^{\mathcal{O}(4)} := \left(1 - \alpha_q^{\mathcal{O}(8)}\right) \odot \alpha_q^{\mathcal{O}(4)}, \\
\theta_q^{\mathcal{O}(8)} := \alpha_q^{\mathcal{O}(8)}.
\end{cases}
\tag{59}
$$

By construction, the blending weights have the following property:

$$
\theta_q^{\mathcal{O}(1)} + \theta_q^{\mathcal{O}(2)} + \theta_q^{\mathcal{O}(4)} + \theta_q^{\mathcal{O}(8)} = 1,
\tag{60}
$$
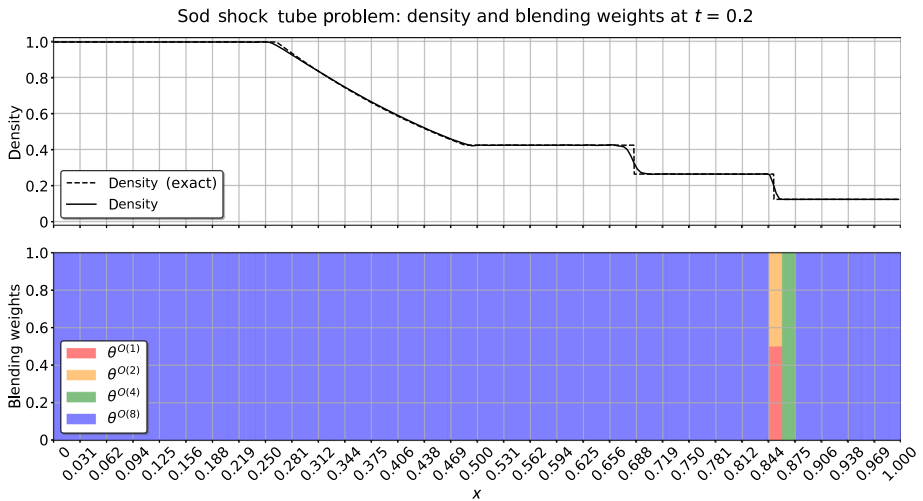
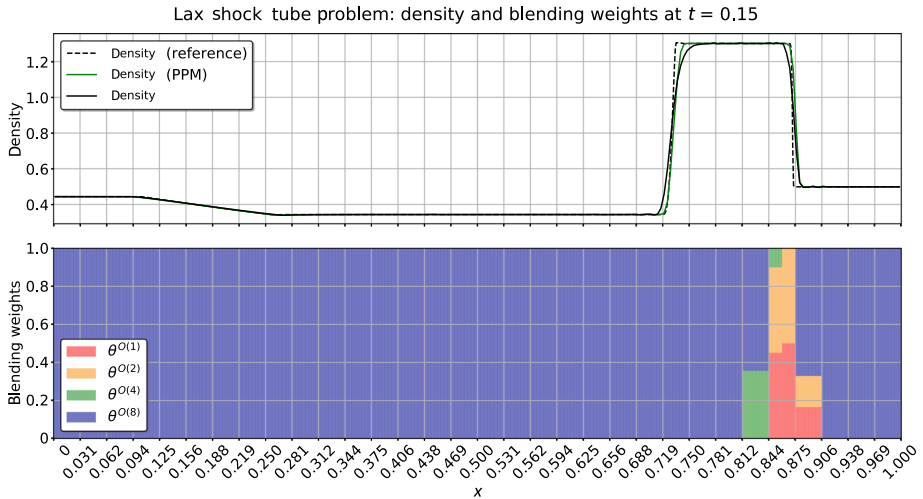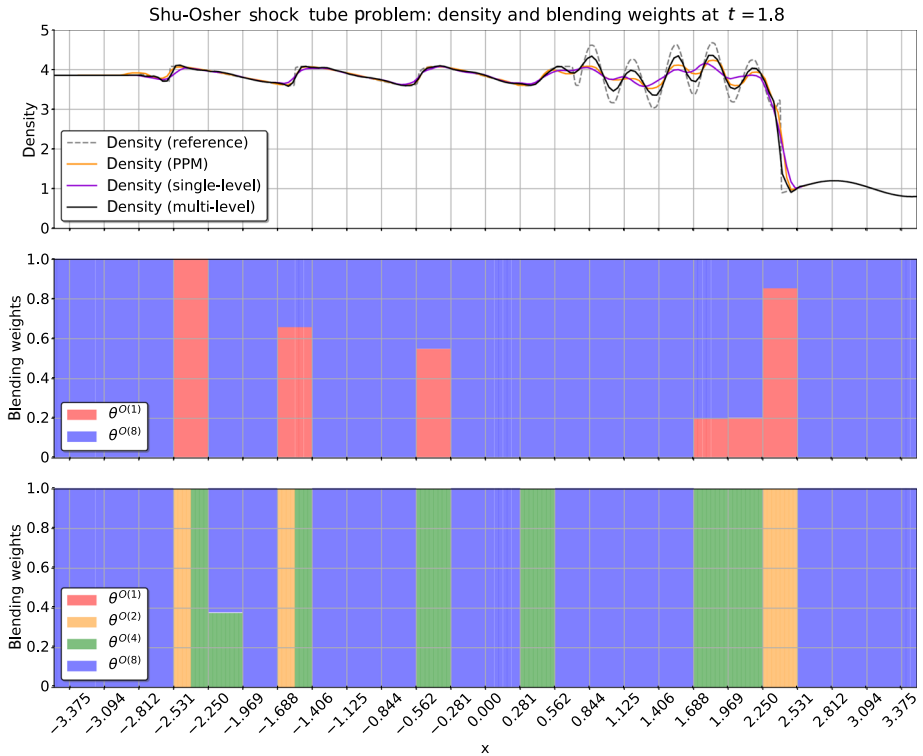**Fig. 6** Sod shock tube problem: numerical solution of the density profile (top row) with the DGFV8 multi-level blending scheme together with the exact solution at the final simulation time $T = 0.2$. The bottom row shows the blending weights $\theta^{\mathcal{O}(n)}$ (Eq. (59)) encoded by stacked bars in different colors. The light gray vertical grid lines depict the element boundaries

and thus give a proper fraction of each contribution.

The first shock tube problem is the Sod shock tube [55]. It is defined on the unit interval $\Omega = [0, 1]$ with a diaphragm located at $x_D = 0.5$. The initial condition in primitive state variables reads

$$\big(\rho_0(x), v_0(x), p_0(x)\big) = \begin{cases} (1, 0, 1), & x < x_D, \\ (0.125, 0, 0.1), & x \geqslant x_D. \end{cases}$$

The resolution is set to 32 elements of $N = 8$ mean values each. This amounts to 256 total DOF. The result for the density profile (top row) at the final simulation time $T = 0.2$ is presented in Fig. 6 together with the exact solution. It shows the correct approximation of the rarefaction wave, contact discontinuity, and the forward facing shock front. As designed, only at the shock front, the blending scheme gets triggered, which is visible in the bottom row of the plot. The stacked bar chart directly corresponds to the blending weights $\theta^{\mathcal{O}(n)}$ in (59). The vertical dimension of the stacked bars completely fill the unit interval mirroring property (60).

The second test case is the Lax shock tube [48] with initial data set to

$$\big(\rho_0(x), v_0(x), p_0(x)\big) = \begin{cases} (0.445, 0.689, 3.528), & x < x_D, \\ (0.5, 0, 0.571), & x \geqslant x_D. \end{cases}$$

All other simulation parameters as well as the resolution are the same as in the Sod test case. The result for the density profile (top row) at the final simulation time $T = 0.15$ is presented in Fig. 7 together with reference solution on a finer grid of 1 024 DOF with the third-order piecewise parabolic method (PPM) [49] implemented in the astrophysics code FLASH (version 4.6, March 2019), see, e.g., [20]. For comparison, we also include the solution of the PPM with the same grid resolution of 256 DOF. All flow features are resolved correctly

Fig. 7 Lax shock tube problem: numerical solution of the density profile (top row) with the DGFV8 (256 DOF) multi-level blending scheme together with the reference solution (PPM, 1 024 DOF) and PPM (256 DOF) at the final simulation time $T = 0.2$. The bottom row shows the blending weights $\theta^{\mathcal{O}(n)}$ (Eq. (59)) encoded by stacked bars in different colors. The vertical grid lines depict the element boundaries

and the blending scheme is only triggered in the region around the forward facing shock. Furthermore, this example clearly reveals the adaptive blending on the sub-element level.

The third shock tube, the Shu-Osher test [54], is a Mach three shock interacting with a sinusoidal density wave. It reveals the scheme's capability of capturing both, discontinuous and smooth parts of the flow. The computational domain in 1D is set to $\Omega = [-4.5, 4.5]$, the final simulation time is $T = 1.8$, and the primitive variables are initialized as

$$\big(\rho_0(x), v_0(x), p_0(x)\big) = \begin{cases} (3.856\ 714\ 3, 2.629\ 369, 10.333\ 33), & x < -4, \\ (1 + 0.2\ \sin(5\,x), 0, 1), & x \geqslant -4. \end{cases}$$

Here, we compare the results of the multi-level blending DGFV8 to PPM. To investigate the importance of the multi-level approach on the accuracy of the DGFV8 result, we additionally perform a simulation where we intentionally deactivate the sub-elements, i.e., $\boldsymbol{\alpha}^{\mathcal{O}(2)} := 0$ and $\boldsymbol{\alpha}^{\mathcal{O}(4)} := 0$. This is identical to the single-level blending approach presented in Sect. 2.5, where only the first-order FV scheme is blended with the eighth-order DG scheme. The resolution is as before 256 total DOF, whereas the reference solution is computed with PPM on a much finer grid of 2 048 DOF. The numerical experiments shown in Fig. 8 nicely demonstrate the benefit of using the multi-level approach. Whereas the shocks are about equally resolved, the multi-level blending variant gives the best results in the smooth parts of the solution. Since the resolution is not very high, non-shock related small scale flow features cannot always be resolved by the eighth-order DG scheme. This is especially visible in the range $x = [1.688, 2.25]$ where the lower order scheme has to completely or at least partially take over.

3.5 2D Riemann Problems

In this section, we present a selection of three 2D Riemann problems [39, 41, 52]. The domain for all simulations is set to $\Omega = [-0.5, 0.5]^2$ with a uniform grid resolution of 64^2 elements,

Fig. 8 Shu-Osher shock tube problem: numerical solution of the density profile (top row) with the DGFV8 multi-level blending scheme together with the reference solution (PPM on 2 048 DOF), PPM, and single-level blending scheme all using 256 DOF. The center row shows the blending weights $\theta^{\mathcal{O}(n)}$ Eq. (59) for the single-level blending, while the bottom row shows the multi-level blending. The vertical grid lines depict the element boundaries

respectively, and 512^2 DOF. The setup consists of the four quadrants each initialized with their own constant states. The exact parameters of each Riemann problem are given in [41] where they are addressed with a fixed configuration number. For brevity, we omit the setup parameters and refer to [41]. In this work, we show configurations 3, 4, and 6. We are interested in the change of the numerical solution when we incrementally add one level of blending order from one run to the next. Hence, we present three solutions for each 2D Riemann problem denoted by their respective multi-level schemes: DGFV2, DGFV4, and DGFV8. Note that the schemes are implemented in such a way that they operate on the same element size of $N^2 = 8^2$ mean values. The results are shown in Fig. 9, 10 and 11. The left column shows the density contour and the right column shows the weighted blending factors as in (57). The general observation is that with increasing order, there is more structure visible in the density plots and the blending patterns get more nuanced in tracing the flow structures.

3.6 2D Sedov Blast

The Sedov blast problem [12, 63, 64] describes the self-similar evolution of a radially symmetrical blast wave from an initial pressure point (delta distribution) at the center into the surrounding, homogeneous medium. The analytical solution is given in [38, 53]. In our setup,
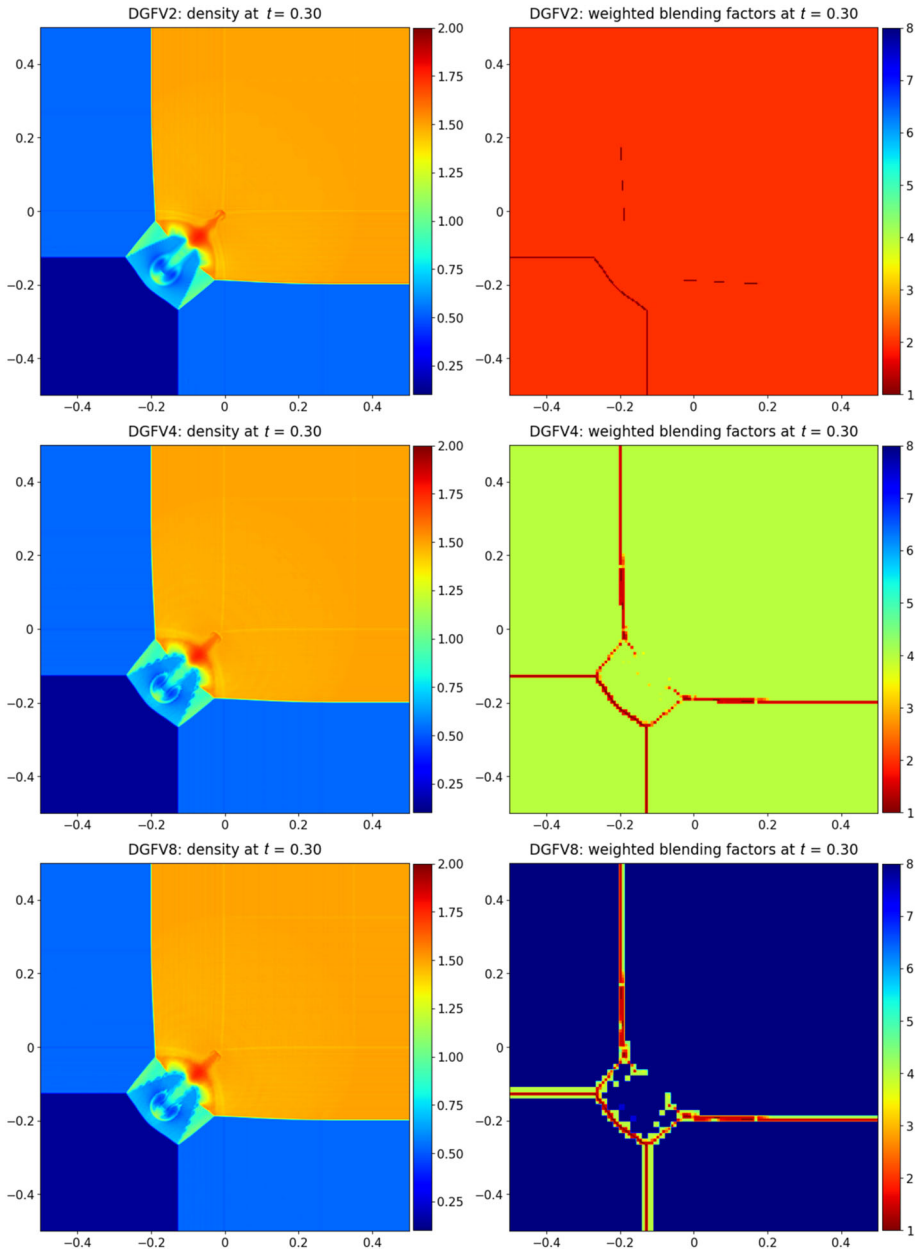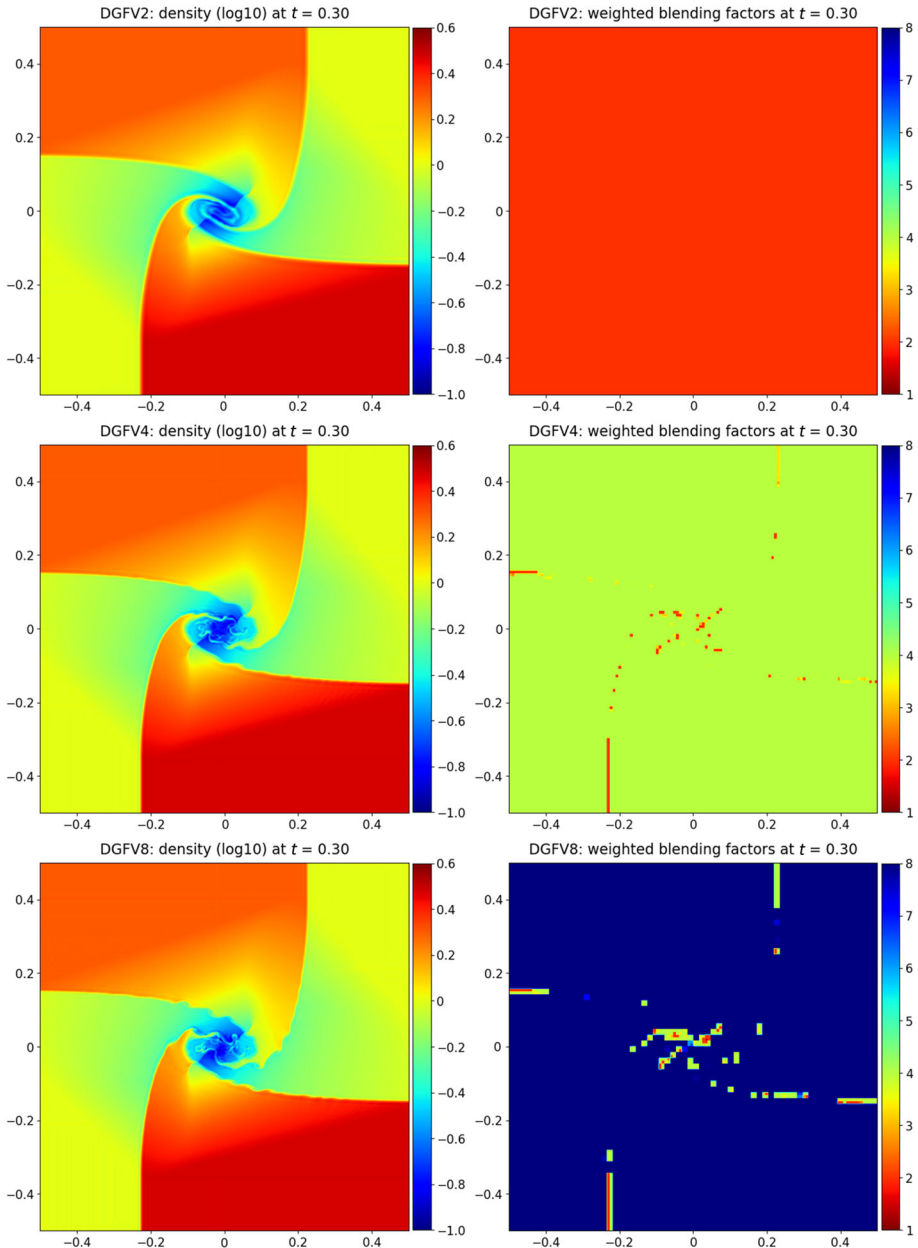
Fig. 9 2D Riemann problem (configuration 3, see [41]) computed with the (from top to bottom) DGFV2, DGFV4, and DGFV8 scheme. The resolution of 512^2 DOF is the same for all runs. Left: density contours at final time $T = 0.3$. Right: weighted blending factors as defined in (57). Dark blue represents the full eighth-order DG and dark red the full first-order FV scheme
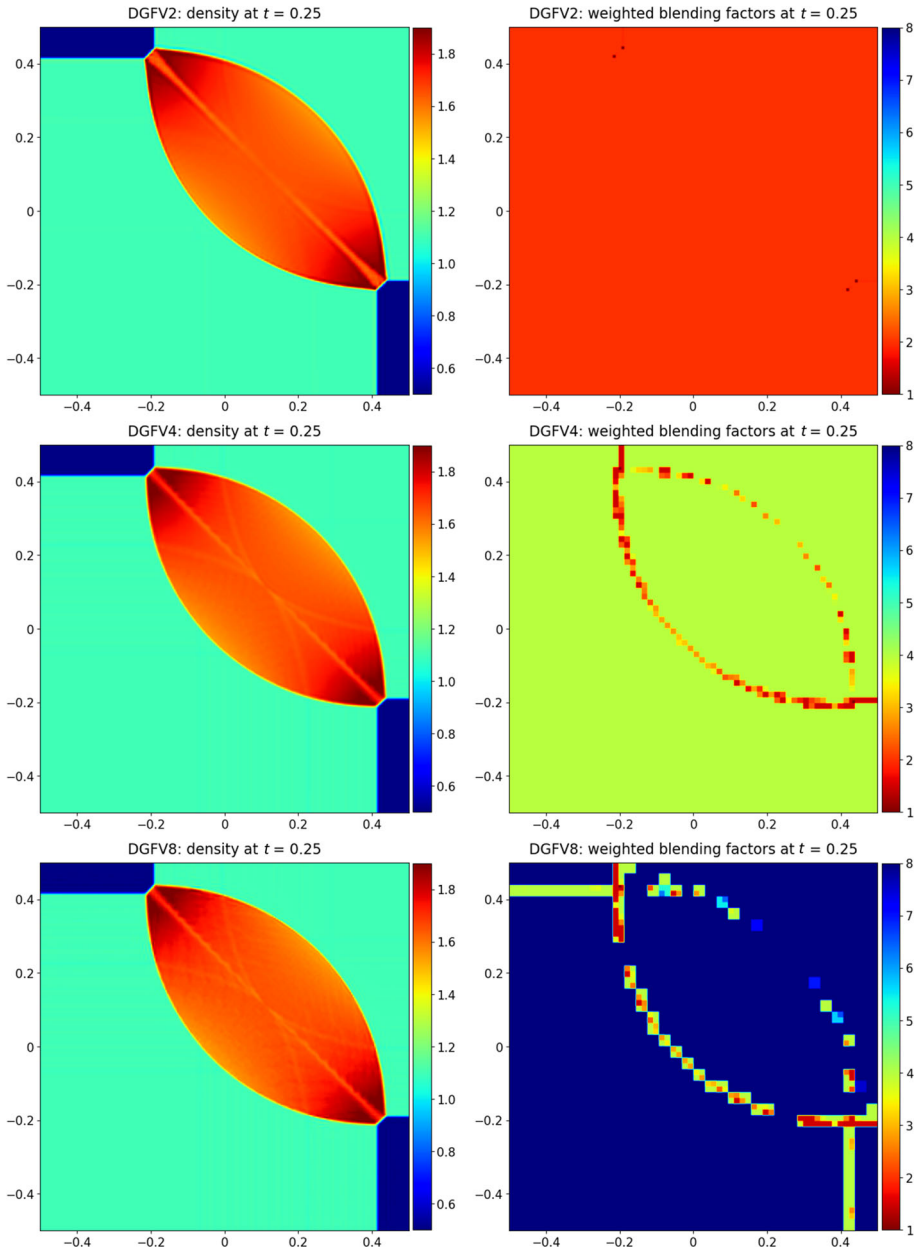
Fig. 10 2D Riemann problem (configuration 4, see [41]) computed with the (from top to bottom) DGFV2, DGFV4 and DGFV8 scheme. The resolution of 512^2 DOF is the same for all runs. Left: density contours at final time $T = 0.25$. Right: weighted blending factors as defined in (57). Dark blue represents the full eighth-order DG and dark red represents the full first-order FV scheme

we approximate the initial pressure point with a smooth Gaussian distribution:

$$\mathcal{E}_0(\boldsymbol{x}) = \frac{p_0}{\gamma - 1} + \frac{E}{(2\,\pi\,\sigma^2)^{d/2}}\,\exp\left(-\frac{1}{2}\,\frac{\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}}{\sigma^2}\right), \tag{61}$$

Fig. 11 2D Riemann problem (configuration 6, see [41]) computed with the (from top to bottom) DGFV2, DGFV4, and DGFV8 scheme. The resolution of 512^2 DOF is the same for all runs. Left: density contours in log-scale at final time $T = 0.3$. Right: weighted blending factors as defined in (57). Dark blue represents the full eighth-order DG and dark red the full first-order FV scheme

with the spatial dimension $d = 2$, the blast energy $E = 1$, and the width σ, such that the initial Gaussian is reasonably resolved. The surrounding medium is initialized with $\rho_0 = 1$ and $p_0 = 10^{-14}$. Dimensional analysis [53] reveals that the analytical solution of the density
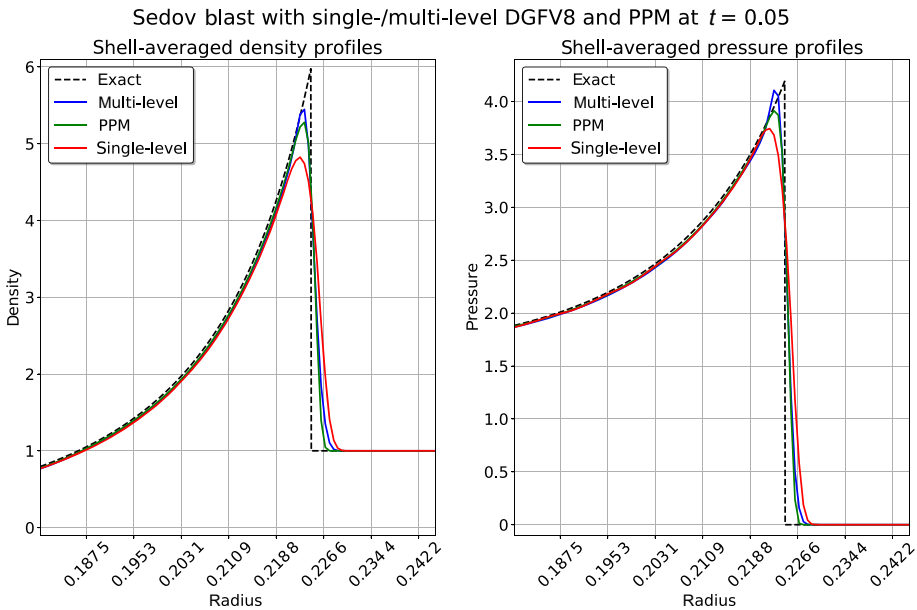
Fig. 12 2D Sedov blast: shell-averaged density and pressure profiles at final simulation time $T = 0.05$ computed with the single-/multi-level blending scheme (DGFV8) and PPM on an FV equivalent uniform grid resolution of 512^2 DOF. The vertical grid lines depict the element boundaries

right at the shock front is determined by

$$\rho_{\text{shock}} = \frac{\gamma + 1}{\gamma - 1} \rho_0.$$

With the adiabatic coefficient $\gamma = 1.4$, we investigate how close the numerical results match $\rho_{\text{shock}} = 6$. The Cartesian mesh has an FV equivalent uniform grid resolution of 512^2 DOF, i.e., when computing with the full eighth-order DG approximation space, the total number of DG elements is 64^2. The spatial domain is $\Omega \in [-0.25, 0.25]^2$ with the initial blast width $\sigma = 5 \times 10^{-3}$. We compare the accuracy of the results obtained with the single-level and the multi-level (DGFV8) blending discretizations as well as the PPM. Figure 12 shows the shell-averaged density and pressure profiles at final time $T = 0.05$. Figure 13 presents the numerical solution over the whole domain as computed with the multi-level DGFV8. To further illustrate the behavior of the multi-level and single-level blending approach, we also show in Fig. 14 the weighted blending factors along the x-axis. The shock front is much sharper for the multi-level blending compared to the single-level blending. It can be observed how the weighted blending factor is dominated by the first-order FV scheme ($\bar{\alpha} \approx 1$) directly at the shock, but transitions quickly to a blended discretization ($1 < \bar{\alpha} < 8$) up to the full eighth-order DG ($\bar{\alpha} \approx 8$) away from the shock, even within a single DG element. This behavior demonstrates the sub-element adaptivity of our novel approach. Again, similarly to the shock tube Sect. 3.4 and the 2D Riemann problem Sect. 3.5, the results for the 2D Sedov blast wave show the benefit of the multi-level approach, with the numerical profiles even slightly closer to $\rho_{\text{shock}} = 6$ compared to the PPM with equal resolution of 512^2 DOF.
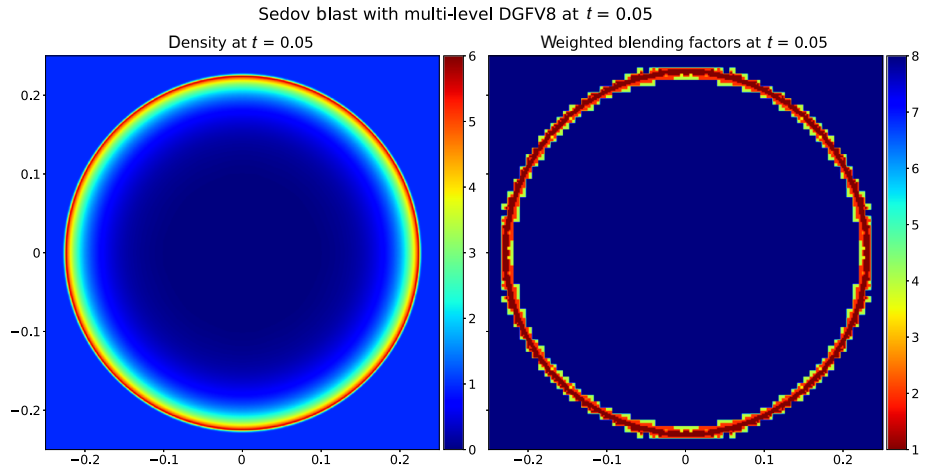
Fig. 13 2D Sedov blast: numerical solution computed with DGFV8 at final simulation time $T = 0.05$. Left: density contours. Right: weighted blending factors as defined in (57). Dark blue represents the full eighth-order DG and dark red represents the full first-order FV scheme
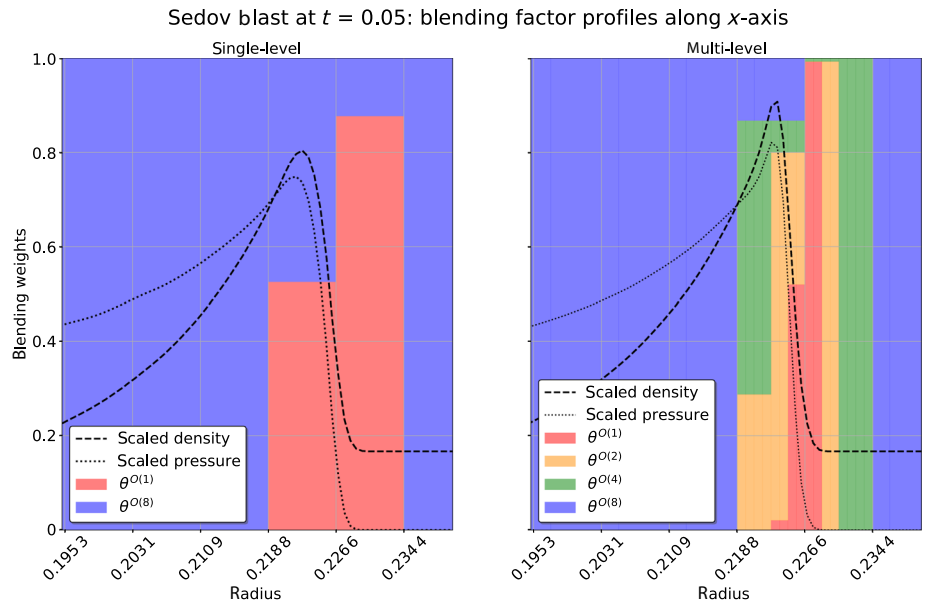

Fig. 14 2D Sedov blast: we compare the blending factor profiles along the x-axis of the single- and multi-level blending schemes at final time $T = 0.05$. The blending weights $\theta^{\mathcal{O}(n)}$ (Eq. (59)) are encoded with stacked bars in their respective colors. The vertical grid lines highlight the eighth-order DG element boundaries. For reference, the scaled density ($\rho/6$) and scaled pressure ($p/5$) are included

Table 5 Conversion from cgs units to simulation units

Quantity	Cgs units	Simulation units
Mass	g	$M_\odot = 1.989 \times 10^{33}$ g
Time	s	$yr = 3.154 \times 10^7$ s
Length	cm	$pc = 3.086 \times 10^{18}$ cm
Temperature	K	K

Table 6 Hydrodynamical parameters in cgs units and simulation units

Description	Cgs units	Simulation units
Domain size	$L = 1.543 \times 10^{19}$ cm	$L = 5\,\mathrm{pc}$
Blast energy	$E = 10^{51}\mathrm{erg}$	$E = 5.2516 \times 10^{-5}\,M_\odot\,\frac{pc^2}{yr^2}$
Ejecta mass	$M = 2.7846 \times 10^{33}$ g	$M = 1.4\,M_\odot$
Ambient density	$n_H = 0.13\,\mathrm{cm}^{-3}$	$\rho_a = 2.4539 \times 10^{-3}\,\frac{M_\odot}{pc^3}$
Ambient temperature	$T = 10^4$ K	$p_a = 2.1309 \times 10^{-13}\,\frac{M_\odot}{pc\,yr^2}$

4 Simulation of a Young Supernova Remnant

Supernova models have been analyzed and discussed for many decades and since they unite a broad range of features such as strong shocks, instabilities, and turbulence, they resemble a good test bed for our novel shock capturing approach in combination with AMR.

The general sequence of events of the presented supernova simulation is like this: we start with a constant distribution of very-low-density resembling interstellar media (ISM) that typically fills the space between stars. When a star explodes by turning into a supernova, it ejects its own mass at very high speeds into the ISM preceded by a strong shock front heating up the ISM. The ejected mass is rapidly decelerated by the swept-up ISM giving rise to a so-called reverse shock that travels backward to the center. The interface, or more precisely the contact discontinuity, between shocked ejecta and shocked ISM is unstable and leads to a layer of slowly growing Rayleigh-Taylor instabilities. This gradually expanding layer, called supernova remnant, is of special interest, since this is where astronomical observations reveal a lot of ongoing physics and chemistry, especially driven by mixing and turbulence.

We adapt the setup descriptions in [8, 16, 18] where we have the initial (internal) blast energy E and the ejecta mass M given in cgs (centimeter-gram-seconds) units. It is beneficial to convert the given units to convenient simulation units reflecting characteristic dimensions of the physical model at hand. Table 5 lists the conversion between cgs and simulation units, and Table 6 lists the initial parameters used in this simulation. The ambient density ρ_a is related to the mono-atomic particle (hydrogen) density n_H via $\rho_a = m_u\,n_H$, where m_u is $\frac{1}{12}$ of the mass of a carbon-12 atom. The ambient pressure p_a is calculated from the ideal gas law, i.e., $p_a = n_H\,k_B\,T$ with the Boltzmann constant k_B. There is some ambiguity regarding the ambient gas temperature T. The literature mentions a warm and neutral interstellar medium which is attributed to temperatures between 6×10^3 K and 10^4 K. The heat capacity ratio is fixed to $\gamma = 5/3$. The simulation time spans a period from $t_0 = 10$ years to $T = 500$ years. The expansion of the forward shock (64) is then expected to approximately reach $R_{FS} = 5\,\mathrm{pc}$, which determines the size of the computational domain $L := 5\,\mathrm{pc}$. Figure 15 depicts a
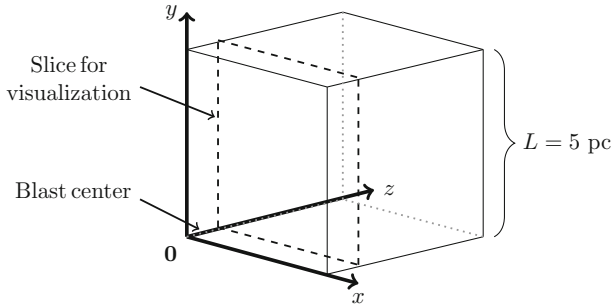
Fig. 15 Computational domain (cubic box) covering one octant of the supernova model. The faces at the coordinate axes are set to reflecting walls, while the opposite sides are set to outflow

schematic of the simulation setup. Due to the rotational symmetry of the setup, it is sufficient to simulate just one octant of the supernova. The following formulas have been derived in [8] and were adapted to the current setup, i.e., power law indices of $(s, n) = (0, 7)$. The self-similar solution at the initial time $t_0 = 10$ yr within the power law region and, respectively, the blast center, is given by

$$r(t) = t \sqrt{\frac{5}{3} \frac{E}{M}} \quad \text{and} \quad \rho(t) = \frac{25}{21\,\pi} \frac{E^2}{M} t^4 r(t)^{-7}. \tag{62}$$

We initialize the density as

$$\rho_0(\boldsymbol{x}) = \rho_a + \rho(t_0) \cdot \begin{cases} 1, & |\boldsymbol{x}| \leqslant r(t_0), \\ \left(\dfrac{|\boldsymbol{x}|}{r(t_0)}\right)^{-7}, & |\boldsymbol{x}| > r(t_0), \end{cases} \tag{63}$$

and the total energy with (61) where $p_0 = p_a, d = 3$, and $\sigma = \frac{3}{4} r(t_0)$. The initial momentum is $(\rho\,\boldsymbol{v})_0 = \boldsymbol{0}$. Since we are only interested in the evolution of the instability layer of the supernova, we apply the following rules for mesh refinement and coarsening. The expansion radius of the forward shock over time is given by

$$R_{\text{FS}}(t) = 1.06 \left(\frac{E^2}{M\,\rho_a}\right)^{1/7} t^{4/7}. \tag{64}$$

This allows us to assign an adaptive, high-resolution shell of maximal refined elements following the remnant as it expands into the computational domain. The inner and outer radii of the shell are estimated as

$$R_{\text{inner}}(t) = 0.7\, R_{\text{FS}}(t) \quad \text{and} \quad R_{\text{outer}}(t) = 1.15\, R_{\text{FS}}(t), \tag{65}$$

which have been found adequate via numerical experimentation. Moreover, up to $t = 200$ years, we enforce $R_{\text{inner}} = 0$, which ensures that the first phase of the explosion is well resolved in any case. The refinement levels range from 2 to 6, which translates to an FV equivalent resolution from $2^2 \cdot 8 = 32$ up to $2^6 \cdot 8 = 512$ cells in each spatial direction.

We perform three simulations with the 3D multi-level blending schemes DGFV2, DGFV4, and DGFV8, analog to the 2D Riemann problems discussed in Sect. 3.5. Figure 16 shows the density slice (left column) sketched in Fig. 15 at the final simulation time of $T = 500$ years, while Fig. 17 shows the corresponding 3D density contour rendering of the DGFV8 solution.
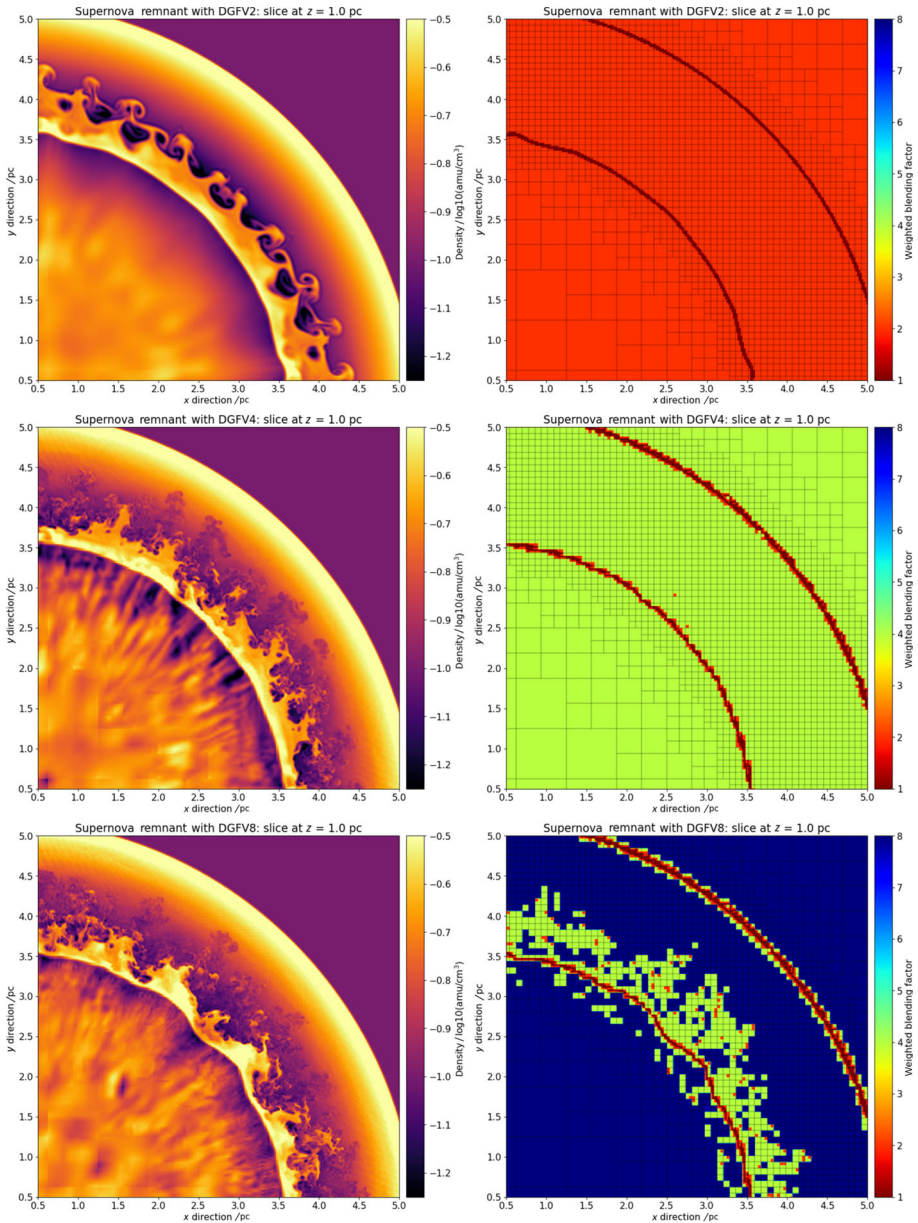
Fig. 16 Left column: 2D density slice (see Fig. 15) showing the instability region, respectively, remnant of the supernova at $T = 500$ years simulated with the (from top to bottom) DGFV2, DGFV4, and DGFV8 scheme. Right column: 2D slice of the weighted blending factors of the respective blending schemes at $T = 500$ years. The black lines correspond to the element boundaries of the Cartesian non-conforming mesh
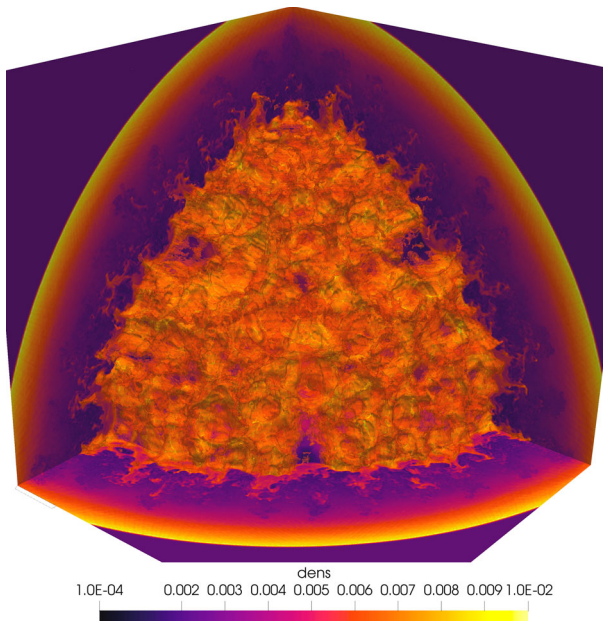
Fig. 17 3D density contour rendering showing the instability region, respectively, remnant of the supernova at the final simulation time $T = 500$ years simulated with the DGFV8 scheme

The shock front partially left the domain, which is not considered a problem since the region of interest, namely the instability layer, is still completely covered. The areas of no interest, i.e., the outer region as well as the center, are only coarsely resolved by the AMR scheme. Clearly, an increase in order leads to a much more detailed remnant structure emphasizing the advantage of higher order schemes in resolving small-scale turbulence driven by Rayleigh-Taylor instability, see [8]. The weighted blending factor is shown in the right column of Fig. 16. The band of highly refined elements is clearly visible following the remnant as intended. Two distinctive lines of blending activity trace the front and reverse shocks. The plots also show a strong qualitative difference of the amount of scales in the instability layer for DGFV2 and DGFV4. Clearly, the DGFV4 result features more scales and finer structures as the DGFV2 simulation with the same FV equivalent grid resolution. The difference in scales between DGFV4 and DGFV8 is less pronounced which is probably related to the extensive blending activity with O4 inside the instability layer of the DGFV8 simulation. In this turbulent part, we suspect that the standard collocation eighth-order DG scheme might face aliasing instability issues as described in the introduction. There are techniques to reduce aliasing issues available, such as filtering, consistent integration, and split forms. However, this aspect is not the main topic of the paper and it is interesting to see that the multi-level blending automatically adjusts to cope with these issues, as well.

5 Conclusion

In this work, we introduce an adaptive sub-element-based shock capturing approach for DG methods. We interpret an element first as a collection of piecewise constant data. This set of piecewise constant data can be interpreted and reconstructed in a variety of ways. We focus on piecewise polynomial approximation spaces, starting from a pure piecewise constant FV-type

interpretation (no reconstruction) up to the fully maximum order polynomial reconstruction and every combination in-between, e.g., piecewise linear and piecewise cubic reconstructions. In a second step, we link the data interpretation to a corresponding (high order) DG discretization. Thus, we get a hierarchy of discretizations acting on the same set of data.

The idea is then to adaptively blend this hierarchy of different discretization, where the low-order variants are chosen close to discontinuities and the high-order variants in smooth or turbulent parts of the simulation. Instead of having an element-based troubled cell indicator approach, we use the different data interpretations again to compute sub-element localized indicators, which allows for a sub-element adaptive blending of the discretizations. When the blending can change throughout the element, a special care is necessary to preserve exact conservation of the resulting multi-level blended discretization. We achieve exact conservation, by introducing unique, blended reconstruction states at subcell and sub-element interfaces.

In our prototype implementation, we further demonstrate that a natural combination of this sub-element adaptive approach is with adaptive mesh refinement. We base the AMR implementation on the P4EST Octree library, which allows for a straight forward parallelization of the whole computational framework.

We show standard numerical test cases to validate the new approach and a simplified model of a supernova remnant to highlight its high accuracy for challenging test cases with strong shocks and turbulence like structures.

Compliance with Ethical Standards

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

Appendix A

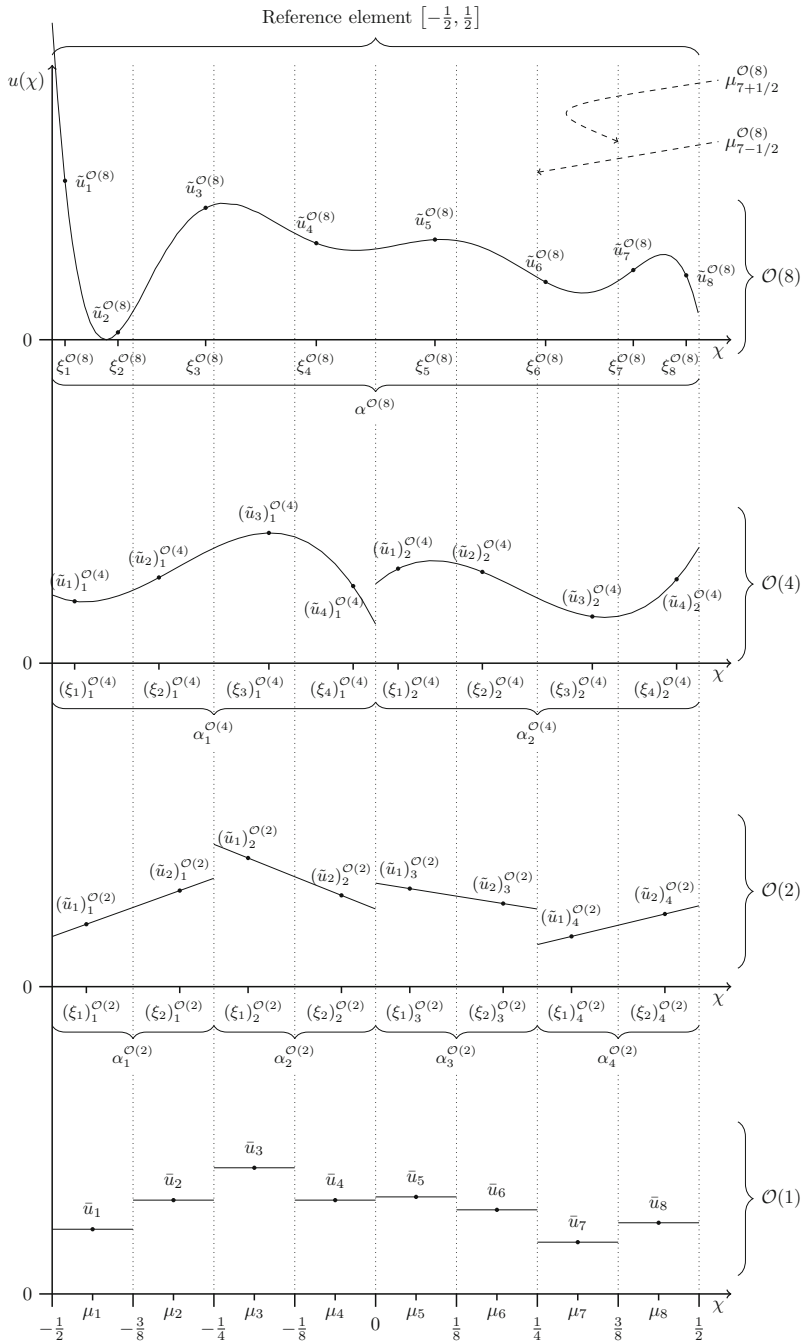## A.1 Four Levels of Data Interpretation of Eight Mean Values

Fig. A1 Schematic of eight ($N = 8$) mean values and its four levels of data interpretation. The values $\mu_{7\pm1/2}^{\mathcal{O}(8)}$ indicate the location of the interpolation nodes in (48) of the FV subcell interpolation operator $\boldsymbol{I}^{\pm\mathcal{O}(n)}$, $n = 2, 4, 8$

A.2 Boundary Evaluation Operators for the Multi-level Blending Scheme (46)

$$
\left[\mathbb{1}_4 \otimes \hat{\boldsymbol{B}}^{\mathcal{O}(2)}\right]^{(0)} =
\begin{pmatrix}
\hat{B}_{11}^{\mathcal{O}(2)} & 0 & \hat{B}_{13}^{\mathcal{O}(2)} & 0 & 0 & 0 & 0 & 0 & 0 \\
\hat{B}_{21}^{\mathcal{O}(2)} & 0 & \hat{B}_{23}^{\mathcal{O}(2)} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \hat{B}_{11}^{\mathcal{O}(2)} & 0 & \hat{B}_{13}^{\mathcal{O}(2)} & 0 & 0 & 0 & 0 \\
0 & 0 & \hat{B}_{21}^{\mathcal{O}(2)} & 0 & \hat{B}_{23}^{\mathcal{O}(2)} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \hat{B}_{11}^{\mathcal{O}(2)} & 0 & \hat{B}_{13}^{\mathcal{O}(2)} & 0 & 0 \\
0 & 0 & 0 & 0 & \hat{B}_{21}^{\mathcal{O}(2)} & 0 & \hat{B}_{23}^{\mathcal{O}(2)} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \hat{B}_{11}^{\mathcal{O}(2)} & 0 & \hat{B}_{13}^{\mathcal{O}(2} \\
0 & 0 & 0 & 0 & 0 & 0 & \hat{B}_{21}^{\mathcal{O}(2)} & 0 & \hat{B}_{23}^{\mathcal{O}(2)}
\end{pmatrix} \in \mathbb{R}^{8\times 9},
$$

$$
\left[\mathbb{1}_2 \otimes \hat{\boldsymbol{B}}^{\mathcal{O}(4)}\right]^{(0)} =
\begin{pmatrix}
\hat{B}_{11}^{\mathcal{O}(4)} & 0 & 0 & 0 & \hat{B}_{15}^{\mathcal{O}(4)} & 0 & 0 & 0 & 0 \\
\hat{B}_{21}^{\mathcal{O}(4)} & 0 & 0 & 0 & \hat{B}_{25}^{\mathcal{O}(4)} & 0 & 0 & 0 & 0 \\
\hat{B}_{31}^{\mathcal{O}(4)} & 0 & 0 & 0 & \hat{B}_{35}^{\mathcal{O}(4)} & 0 & 0 & 0 & 0 \\
\hat{B}_{41}^{\mathcal{O}(4)} & 0 & 0 & 0 & \hat{B}_{45}^{\mathcal{O}(4)} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \hat{B}_{11}^{\mathcal{O}(4)} & 0 & 0 & 0 & \hat{B}_{15}^{\mathcal{O}(4} \\
0 & 0 & 0 & 0 & \hat{B}_{21}^{\mathcal{O}(4)} & 0 & 0 & 0 & \hat{B}_{25}^{\mathcal{O}(4} \\
0 & 0 & 0 & 0 & \hat{B}_{31}^{\mathcal{O}(4)} & 0 & 0 & 0 & \hat{B}_{35}^{\mathcal{O}(4} \\
0 & 0 & 0 & 0 & \hat{B}_{41}^{\mathcal{O}(4)} & 0 & 0 & 0 & \hat{B}_{45}^{\mathcal{O}(4)}
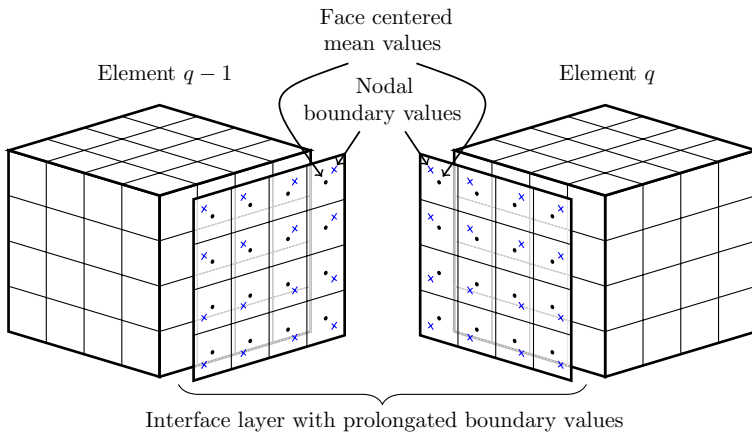\end{pmatrix} \in \mathbb{R}^{8\times 9},
$$

Fig. A2 Schematic of conforming interface $q - \frac{1}{2}$ with 1:1 adjacent elements $q-1$ and q for $N = 4$. The two kinds of boundary values occurring in (A6) are illustrated

$$\left[\mathbb{1}_1 \otimes \hat{\boldsymbol{B}}^{\mathcal{O}(8)}\right]^{(0)} = \begin{pmatrix} \hat{B}_{11}^{\mathcal{O}(8)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \hat{B}_{19}^{\mathcal{O}(8)} \\ \hat{B}_{21}^{\mathcal{O}(8)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \hat{B}_{29}^{\mathcal{O}(8)} \\ \hat{B}_{31}^{\mathcal{O}(8)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \hat{B}_{39}^{\mathcal{O}(8)} \\ \hat{B}_{41}^{\mathcal{O}(8)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \hat{B}_{49}^{\mathcal{O}(8)} \\ \hat{B}_{51}^{\mathcal{O}(8)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \hat{B}_{59}^{\mathcal{O}(8)} \\ \hat{B}_{61}^{\mathcal{O}(8)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \hat{B}_{69}^{\mathcal{O}(8)} \\ \hat{B}_{71}^{\mathcal{O}(8)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \hat{B}_{79}^{\mathcal{O}(8)} \\ \hat{B}_{81}^{\mathcal{O}(8)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \hat{B}_{89}^{\mathcal{O}(8)} \end{pmatrix} \in \mathbb{R}^{8\times 9}.$$

A.3 Extension to 3D Cartesian Grids

The extension to higher spatial dimensions with Cartesian conforming grids is relatively straight forward via a tensor-product strategy. In this section, we want to present the most important building blocks for the 3D single-level blending scheme. The steps of the 3D multi-level blending are analogous as in Sect. 2.6 and not detailed any further. Consider the general 3D conservation law:

$$\partial_t u + \partial_x f(u) + \partial_y g(u) + \partial_z h(u) = 0. \tag{A1}$$

We now have an element q with N^3 mean values as available data:

$$\bar{\mathbb{u}}_q \in \mathbb{R}^{N\times N\times N},$$

and the element sizes Δx_q, Δy_q, and Δz_q. With the tensor-product of Lagrange polynomials ℓ on Legendre-Gauss nodes ξ_i, $i = 1, \cdots, N$:

$$u(t; \xi, \eta, \zeta) \approx \sum_{i,j,k=1}^{N} \tilde{u}_{ijk}(t)\, \ell_i(\xi)\, \ell_j(\eta)\, \ell_k(\zeta), \tag{A2}$$

and the reconstruction operator $\boldsymbol{R}$, which we introduced in Sect. 2.4, we get the nodal coefficients as

$$\tilde{\mathbb{u}}_q = \bar{\mathbb{u}}_q \overset{z}{\underset{d=x}{\bigtimes}} \boldsymbol{R} = \bar{\mathbb{u}}_q \times_x \boldsymbol{R} \times_y \boldsymbol{R} \times_z \boldsymbol{R} \in \mathbb{R}^{N\times N\times N}. \tag{A3}$$

Here, we make use of the n-mode product notation [33]. A definition is given in Appendix A.5. In general, all vector and matrix operations presented so far can be directly expanded to 3D via the n-mode product. The blending scheme (42) in 3D reads as:

$$\begin{aligned}
_t\bar{\mathbb{u}}_q = &- (1 - \alpha_q)\frac{N}{\Delta x_q}\bar{\mathbb{f}}_q^* \times_x \boldsymbol{\Delta} + \alpha_q \frac{1}{\Delta x_q}\left\{\tilde{\mathbb{f}}_q \times_x \hat{\boldsymbol{D}} \times_y \boldsymbol{P} \times_z \boldsymbol{P} - \bar{\mathbb{f}}_q^* \times_x \hat{\boldsymbol{B}}\right\} \\
&- (1 - \alpha_q)\frac{N}{\Delta y_q}\bar{\mathbb{g}}_q^* \times_y \boldsymbol{\Delta} + \alpha_q \frac{1}{\Delta y_q}\left\{\tilde{\mathbb{g}}_q \times_y \hat{\boldsymbol{D}} \times_z \boldsymbol{P} \times_x \boldsymbol{P} - \bar{\mathbb{g}}_q^* \times_y \hat{\boldsymbol{B}}\right\} \\
&- (1 - \alpha_q)\frac{N}{\Delta z_q}\bar{\mathbb{h}}_q^* \times_z \boldsymbol{\Delta} + \alpha_q \frac{1}{\Delta z_q}\left\{\tilde{\mathbb{h}}_q \times_z \hat{\boldsymbol{D}} \times_x \boldsymbol{P} \times_y \boldsymbol{P} - \bar{\mathbb{h}}_q^* \times_z \hat{\boldsymbol{B}}\right\}. \tag{A4}
\end{aligned}$$

The DG volume fluxes are calculated from the node values as

$$\left(\tilde{\mathbb{f}}_q\right)_{ijk} = f\left(\left(\tilde{\mathbb{u}}_q\right)_{ijk}\right), \ \left(\tilde{\mathbb{g}}_q\right)_{ijk} = g\left(\left(\tilde{\mathbb{u}}_q\right)_{ijk}\right) \text{ and } \left(\tilde{\mathbb{h}}_q\right)_{ijk} = h\left(\left(\tilde{\mathbb{u}}_q\right)_{ijk}\right), \ i, j, k = 1, \cdots, N.$$

Remark A1 If necessary, the β-reconstruction (30) ensures permissible states for the reconstructed polynomial, that is

$$\tilde{\mathbb{u}}_q^{(\beta)} = \bar{\mathbb{u}}_q \overset{z}{\underset{d=x}{\times}} \boldsymbol{R}^{(\beta)}. \tag{A5}$$

Again, for $\bar{\mathbb{f}}_q^* \in \mathbb{R}^{(N+1)\times N \times N}$, $\bar{\mathbb{g}}_q^* \in \mathbb{R}^{N \times (N+1) \times N}$ and $\bar{\mathbb{h}}_q^* \in \mathbb{R}^{N \times N \times (N+1)}$, we need to compute common surface fluxes to preserve conservation. Therefore, we define two selection operators:

$$\boldsymbol{s}^- = (1, 0, \cdots, 0, 0)^{\mathrm{T}} \in \mathbb{R}^N \text{ and } \boldsymbol{s}^+ = (0, 0, \cdots, 0, 1)^{\mathrm{T}} \in \mathbb{R}^N,$$

mapping the outermost mean values of $\bar{\mathbb{u}}_q$ to the respective interface. Then, the 3D analog of the prolongation procedure (37) along direction d reads as

$$\bar{\mathbf{u}}_{q\pm\frac{1}{2}}^{\pm d} = (1 - \alpha_q) \underbrace{\bar{\mathbb{u}}_q \times_d \boldsymbol{s}^{\pm}}_{\substack{\text{face-centered} \\ \text{mean values}}} + \alpha_q \underbrace{\boldsymbol{P} \overbrace{\left(\tilde{\mathbb{u}}_q \times_d \boldsymbol{b}^{\pm}\right)}^{\substack{\text{nodal boundary} \\ \text{values}}} \boldsymbol{P}^{\mathrm{T}}}_{\substack{\text{face-centered} \\ \text{mean values}}} \in \mathbb{R}^{N \times N}. \tag{A6}$$

In Fig. A2, the two kinds of boundary values are illustrated.

Note that only the face-centered mean values $\bar{\mathbf{u}}_{q\pm\frac{1}{2}}^{\pm d}$ together with the blending factor α_q are supposed to be exchanged between processes in case of distributed computing.

Next, we reconstruct again a nodal representation from $\bar{\mathbf{u}}_{q\pm\frac{1}{2}}^{\pm d}$:

$$\tilde{\mathbf{u}}_{q\pm\frac{1}{2}}^{\pm d} = \underbrace{\boldsymbol{R} \, \bar{\mathbf{u}}_{q\pm\frac{1}{2}}^{\pm d} \, \boldsymbol{R}^{\mathrm{T}}}_{\substack{\text{nodal boundary} \\ \text{values}}} \in \mathbb{R}^{N \times N}. \tag{A7}$$

From the two representations of boundary values $\bar{\mathbf{u}}_{q\pm\frac{1}{2}}^{\pm x}$ and $\tilde{\mathbf{u}}_{q\pm\frac{1}{2}}^{\pm x}$, we calculate two candidate fluxes at the interface $q - \frac{1}{2}$. In x-direction, it reads as

$$\bar{\mathbf{f}}_{q-\frac{1}{2}}^{*'} = f^*\left(\bar{\mathbf{u}}_{q-\frac{1}{2}}^{+x}, \bar{\mathbf{u}}_{q-\frac{1}{2}}^{-x}\right) \in \mathbb{R}^{N \times N}, \tag{A8}$$

$$\tilde{\mathbf{f}}_{q-\frac{1}{2}}^{*} = f^*\left(\tilde{\mathbf{u}}_{q-\frac{1}{2}}^{+x}, \tilde{\mathbf{u}}_{q-\frac{1}{2}}^{-x}\right) \in \mathbb{R}^{N \times N}. \tag{A9}$$

Next, we determine a common surface flux $\bar{\mathbf{f}}_{q-\frac{1}{2}}^*$ by blending the two candidate interface fluxes:

$$\bar{\mathbf{f}}_{q-\frac{1}{2}}^* = (1 - \alpha_{q-\frac{1}{2}}) \, \bar{\mathbf{f}}_{q-\frac{1}{2}}^{*'} + \alpha_{q-\frac{1}{2}} \, \boldsymbol{P} \, \tilde{\mathbf{f}}_{q-\frac{1}{2}}^* \, \boldsymbol{P}^{\mathrm{T}} \in \mathbb{R}^{N \times N}, \tag{A10}$$

where $\alpha_{q-\frac{1}{2}} = \frac{\alpha_{q-1}+\alpha_q}{2}$. The final step is to calculate the inner interface fluxes analogous to (38):

$$\left(\bar{\mathbb{f}}_q^*\right)_{ijk} = f^*(\bar{u}_{i-1jk}, \bar{u}_{ijk}), \quad i = 2, \cdots, N, \ j, k = 1, \cdots, N$$
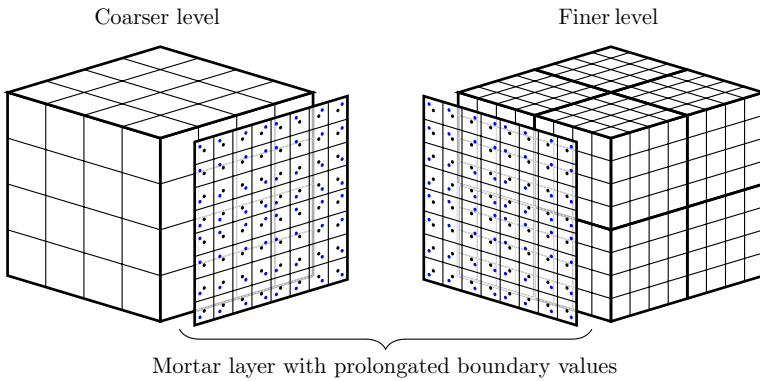
Fig. A3 Schematic of a non-conforming interface with 4:1 adjacent cells for $N = 4$. The two kinds of boundary values occurring in (A15) are illustrated. Compare with Fig. A2

and insert the common surface fluxes $\bar{f}^*_{q \pm \frac{1}{2}}$:

$$
\left(\bar{\mathbb{f}}^*_q \right)_{ijk} = \left(\bar{f}^*_{q - \frac{1}{2}} \right)_{jk}, \quad i = 1, \ j, k = 1, \cdots, N,
$$

$$
\left(\bar{\mathbb{f}}^*_q \right)_{ijk} = \left(\bar{f}^*_{q + \frac{1}{2}} \right)_{jk}, \quad i = N + 1, \ j, k = 1, \cdots, N.
$$

The y- and z-direction are treated analogously. The computation of the blending parameter α for 3D follows the same steps as in Sect. 2.7 where the integrals (49) are rewritten in 3D form. The presented parameters τ_a and τ_s stay the same. This concludes the 3D blending scheme with conforming interfaces.

We extend the 3D single-blending scheme to non-conforming grids where we assume 4:1 transitions only. See Fig. A3. The steps for the 3D multi-level blending scheme are analogous and not detailed any further. First, we define four matrix operators which allow us to construct refinement and coarsening procedures within the blending framework. We require that $N = 2^l$, $l \in \mathbb{N}$, and define the following expansion and compression operators:

$$
\boldsymbol{E}^{(N \to 2N)} := \mathbb{1}_N \otimes \mathbf{1}_2 \quad \in \mathbb{R}^{2N \times N} \tag{A11}
$$

and

$$
\boldsymbol{C}^{(N \to \frac{N}{2})} := \frac{1}{2} \mathbb{1}_N \otimes \mathbf{1}_2^{\mathrm{T}} \quad \in \mathbb{R}^{\frac{N}{2} \times N}. \tag{A12}
$$

Moreover, we need the projection operator $\boldsymbol{P}^{(N \to M)}$ introduced in Sect. 2.3 for mapping N node values to M mean values. We define

$$
\underbrace{\boldsymbol{E} := \boldsymbol{E}^{(N \to 2N)}, \quad \boldsymbol{C} := \boldsymbol{C}^{(N \to \frac{N}{2})}}_{\text{mean values to mean values}} \quad \text{and} \quad \underbrace{\boldsymbol{Z} := \boldsymbol{P}^{(N \to 2N)}, \quad \boldsymbol{Q} := \boldsymbol{P}^{(N \to \frac{N}{2})}}_{\text{node values to mean values}}.
$$

We refine the parent element q as

$$
\bar{\mathbb{r}}_q = (1 - \alpha_q)\, \bar{\mathbb{u}}_q \overset{z}{\underset{d=x}{\bigtimes}} \boldsymbol{E} + \alpha_q\, \tilde{\mathbb{u}}_q \overset{z}{\underset{d=x}{\bigtimes}} \boldsymbol{Z} \quad \in \mathbb{R}^{2N \times 2N \times 2N}. \tag{A13}
$$

The refined block $\bar{\bar{\mathtt{x}}}_q$ is then split into 8 child elements. The reverse operation, namely coarsening, is done by compressing each child element $\bar{\mathtt{u}}_r$, $r = 1, \cdots, 8$, separately:

$$\bar{\bar{\mathtt{c}}}_q = (1 - \alpha_q)\, \bar{\mathtt{u}}_q \overset{z}{\underset{d=x}{\times}} \boldsymbol{C} + \alpha_q\, \tilde{\mathtt{u}}_q \overset{z}{\underset{d=x}{\times}} \boldsymbol{Q} \quad \in \mathbb{R}^{\frac{N}{2} \times \frac{N}{2} \times \frac{N}{2}}. \tag{A14}$$

Afterward, the family of eight blocks $\bar{\bar{\mathtt{c}}}_q$ is glued together. These blending operations are especially useful for refining or coarsening of highly oscillatory data or at pronounced discontinuities.

For the treatment of non-conforming 4:1 interfaces, we need a procedure which maps the boundary values from the coarser element to a so-called mortar layer [37] that has the same resolution as the adjacent four smaller elements. See Fig. A3. To compute the mortar layer of the coarser element q, we formulate

$$\left(\bar{\mathtt{u}}^{\pm d}_{q \pm \frac{1}{2}}\right)_{\text{coarse}} = (1 - \alpha_q)\, \underbrace{\boldsymbol{E}\left(\bar{\mathtt{u}}_q \times_d \boldsymbol{s}^\pm\right) \boldsymbol{E}^{\mathrm{T}}}_{\substack{\text{face-centered} \\ \text{mean values}}} + \alpha_q\, \underbrace{\tilde{\boldsymbol{Z}}\overbrace{\left(\tilde{\mathtt{u}}_q \times_d \boldsymbol{b}^\pm\right)}^{\substack{\text{nodal boundary} \\ \text{values}}} \tilde{\boldsymbol{Z}}^{\mathrm{T}}}_{\substack{\text{face-centered} \\ \text{mean values}}} \quad \in \mathbb{R}^{2N \times 2N}. \tag{A15}$$

Note the similarity to the prolongation procedure for conforming interfaces (A6). The coarse side of the mortar layer $\left(\bar{\mathtt{u}}^{\pm d}_{q \pm \frac{1}{2}}\right)_{\text{coarse}}$ is split up to match the faces of the four finer elements $r = 1, \cdots, 4$. The boundary values of the smaller elements are constructed by applying (A6) individually. Only the mortar layers together with the associated blending factors α_r, $r = 1, \cdots, 4$, are supposed to be exchanged between processes in case of distributed computing. The separate computation of the interface fluxes $\bar{\mathtt{f}}^*_{\text{fine},r} \in \mathbb{R}^{N \times N}$, $r = 1, \cdots, 4$, follows the formulas (A7)–(A10). While the resulting four fluxes can be directly copied back to the smaller faces without further treatment, they have to be mapped to the coarser side via L^2 projection [37]:

$$\int_{-\frac{1}{2}}^{0} \int_{-\frac{1}{2}}^{0} \left(f^*_{\text{coarse}} - f^*_{\text{fine},1}\right) \phi \, \mathrm{d}\xi \, \mathrm{d}\eta + \int_{0}^{\frac{1}{2}} \int_{-\frac{1}{2}}^{0} \left(f^*_{\text{coarse}} - f^*_{\text{fine},2}\right) \phi \, \mathrm{d}\xi \, \mathrm{d}\eta$$

$$+ \int_{-\frac{1}{2}}^{0} \int_{0}^{\frac{1}{2}} \left(f^*_{\text{coarse}} - f^*_{\text{fine},3}\right) \phi \, \mathrm{d}\xi \, \mathrm{d}\eta + \int_{0}^{\frac{1}{2}} \int_{0}^{\frac{1}{2}} \left(f^*_{\text{coarse}} - f^*_{\text{fine},4}\right) \phi \, \mathrm{d}\xi \, \mathrm{d}\eta = 0.$$

The integrals are evaluated in the reference space of the coarse face. Applying exact quadrature rules gives

$$\left(\tilde{\mathtt{f}}^*_{\text{coarse}}\right)_{ij} = \sum_{kl}^{N} \left(\tilde{\mathtt{f}}^*_{\text{fine},1}\right)_{kl} \ell_i \left(\frac{1}{2}\xi_k - \frac{1}{4}\right) \frac{\omega_k}{2\,\omega_i} \ell_j \left(\frac{1}{2}\eta_l - \frac{1}{4}\right) \frac{\omega_l}{2\,\omega_j}$$

$$+ \sum_{kl}^{N} \left(\tilde{\mathtt{f}}^*_{\text{fine},2}\right)_{kl} \ell_i \left(\frac{1}{2}\xi_k + \frac{1}{4}\right) \frac{\omega_k}{2\,\omega_i} \ell_j \left(\frac{1}{2}\eta_l - \frac{1}{4}\right) \frac{\omega_l}{2\,\omega_j}$$

$$+ \sum_{kl}^{N} \left(\tilde{\mathtt{f}}^*_{\text{fine},3}\right)_{kl} \ell_i \left(\frac{1}{2}\xi_k - \frac{1}{4}\right) \frac{\omega_k}{2\,\omega_i} \ell_j \left(\frac{1}{2}\eta_l + \frac{1}{4}\right) \frac{\omega_l}{2\,\omega_j}$$

$$+ \sum_{kl}^{N} \left(\tilde{\mathtt{f}}^*_{\text{fine},4}\right)_{kl} \ell_i \left(\frac{1}{2}\xi_k + \frac{1}{4}\right) \frac{\omega_k}{2\,\omega_i} \ell_j \left(\frac{1}{2}\eta_l + \frac{1}{4}\right) \frac{\omega_l}{2\,\omega_j},$$

where $i, j = 1, \cdots, N$ and ξ_k, η_l are the collocation nodes of the tensor product (A2). We translate the above equation into matrix notation and get

$$\tilde{f}^*_{\text{coarse}} = \boldsymbol{L}_- \tilde{f}^*_{\text{fine},1} \boldsymbol{L}_-^{\mathrm{T}} + \boldsymbol{L}_+ \tilde{f}^*_{\text{fine},2} \boldsymbol{L}_-^{\mathrm{T}} + \boldsymbol{L}_- \tilde{f}^*_{\text{fine},3} \boldsymbol{L}_+^{\mathrm{T}} + \boldsymbol{L}_+ \tilde{f}^*_{\text{fine},4} \boldsymbol{L}_+^{\mathrm{T}} \in \mathbb{R}^{N \times N}, \quad \text{(A16)}$$

where $(\boldsymbol{L}_\pm)_{ij} = \ell_i \left(\frac{1}{2} \xi_j \pm \frac{1}{4} \right) \frac{\omega_j}{2\omega_i}, i, j = 1, \cdots, N$. Analogous to (A10), we determine a common surface flux for the coarse side:

$$\bar{f}^*_{q-\frac{1}{2}} = (1 - \alpha_{q-\frac{1}{2}}) \; \boldsymbol{C} \; \bar{f}^{*'}_{q-\frac{1}{2}} \; \boldsymbol{C}^{\mathrm{T}} \; + \; \alpha_{q-\frac{1}{2}} \; \boldsymbol{P} \; \left(\tilde{f}^*_{\text{coarse}} \right)_{q-\frac{1}{2}} \; \boldsymbol{P}^{\mathrm{T}} \quad \in \mathbb{R}^{N \times N}, \quad \text{(A17)}$$

where $\alpha_{q-\frac{1}{2}} = \frac{1}{2}\left(\alpha_{q-1} + \min\limits_{r=1,\cdots,4} (\alpha_r)_q\right)$. The face-centered fluxes $\bar{f}^{*'}_{q-\frac{1}{2}} \in \mathbb{R}^{2N \times 2N}$ are the result of the appropriate glue operation of the four first-order fluxes $\left(\bar{f}^{*'}_r\right)_{q-\frac{1}{2}} \in \mathbb{R}^{N \times N}$, $r = 1, \cdots, 4$. This concludes the description of the treatment of non-conforming 3D Cartesian grids within the blending framework.

A.4 Sketch of the Algorithm for the Multi-level Blending Scheme

The algorithm of the multi-level blending scheme is more involved, but still follows the same general sequence of steps as in the single-level scheme outlined in Sect. 2.8. Moreover, we use the notation for the 3D scheme (Appendix A.3). At each Runge-Kutta stage, we do as follows.

(I) Initialize two tracing variables: $n_{q,\min} = n_{q,\max} = 8$.

(II) Loop from highest to lowest order: $n_q = 8, 4, 2$.

- Set $n_{q,\min} = n_q$ of element q.
- For each sub-element s in element q at level n_q.
  - Reconstruct the polynomial $(\tilde{\mathbb{u}}_s)_q^{\mathcal{O}(n)}$ from given mean values $\bar{\mathbb{u}}_q$ via (43), (44) or (45).
  - If the reconstructed polynomial $(\tilde{\mathbb{u}}_s)_q^{\mathcal{O}(n)}$ contains non-permissible states, see (31), then calculate the limited version $\left((\tilde{\mathbb{u}}_s)_q^{\mathcal{O}(n)} \right)^{(\beta)}$ as in (30).
  - If the squeezing parameter $(\beta_s)_q^{\mathcal{O}(n)}$ is below β_L, then set $(\alpha_s)_q^{\mathcal{O}(n)} := 0$ else compute the blending factor $(\alpha_s)_q^{\mathcal{O}(n)}$ via (50) from the **unlimited** polynomial $(\tilde{\mathbb{u}}_s)_q^{\mathcal{O}(n)}$.
- If all blending factors $(\alpha_s)_q^{\mathcal{O}(n)}$ are above α_H, then break the loop.
- If all blending factors $(\alpha_s)_q^{\mathcal{O}(n)}$ are below α_L, then set $n_{q,\max} = n - 1$.

(III) Compute the multi-level blended boundary values $\bar{\mathbb{u}}^{\pm d}_{q\pm\frac{1}{2}}$ analog to (A6) within the minimum and maximum bounds given by $n_{q,\min}$ and $n_{q,\max}$. Exchange them together with all involved blending factors $\alpha_q^{\mathcal{O}(n)}$ alongside zone boundaries in case of distributed computing.

(IV) Determine the multi-level blended common surface flux $\bar{f}^*_{q-\frac{1}{2}}$ analog to (A10) within the minimum and maximum bounds given by $n_{q,\min}$ and $n_{q,\max}$.

(V) Compute the multi-level blended right-hand-side $\partial_t \bar{\mathbb{u}}_q$ as described in Sect. 2.6 but within the minimum and maximum bounds given by $n_{q,\min}$ and $n_{q,\max}$.

(VI) Forward in time to the next Runge-Kutta stage and return to step (I).

The switching thresholds are set to $\alpha_H := 0.99$ and $\alpha_L := 0.01$ and the limiter threshold to $\beta_L := 0.95$. Note that the algorithm only applies the blending procedure where necessary to maintain the overall performance of the scheme. The two bounds $n_{q,\min}$ and $n_{\max}$ even avoid redundant computation at levels sorted out by the shock indicator.

A.5 *N*-Mode Product

Given tensor $\mathbf{u} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_D}$ with a matrix $\boldsymbol{A} \in \mathbb{R}^{J \times I_d}$, then the n-mode product is defined as:

$$(\mathbf{u} \times_d \boldsymbol{A})_{i_1 \cdots i_{d-1} \, j \, i_{d+1} \cdots i_D} = \sum_{i_d}^{I_d} u_{i_1 \cdots i_d \cdots i_D} \, A_{j i_d}. \tag{A18}$$

The resulting tensor has following dimensions:

$$\mathbf{u} \times_d \boldsymbol{A} \in \mathbb{R}^{I_1 \times \cdots \times I_{d-1} \times J \times I_{d+1} \times \cdots \times I_D}.$$

Given the same tensor $\mathbf{u}$ and a vector $\boldsymbol{b} \in \mathbb{R}^{I_d}$, then the n-mode product acts as a contraction of $\mathbf{u}$ with $\boldsymbol{b}$ along dimension d. We write

$$(\mathbf{u} \times_d \boldsymbol{b})_{i_1 \cdots i_{d-1} \, i_{d+1} \cdots i_D} = \sum_{i_d}^{I_d} u_{i_1 \cdots i_d \cdots i_D} \, b_{i_d}. \tag{A19}$$

The resulting tensor has following dimensions:

$$\mathbf{u} \times_d \boldsymbol{b} \in \mathbb{R}^{I_1 \times \cdots \times I_{d-1} \times I_{d+1} \times \cdots \times I_D}.$$

References

1. Black, K.: A conservative spectral element method for the approximation of compressible fluid flow. Kybernetika **35**(1), 133–146 (1999)
2. Bohm, M., Schermeng, S., Winters, A.R., Gassner, G.J., Jacobs, G.B.: Multi-element SIAC filter for shock capturing applied to high-order discontinuous Galerkin spectral element methods. J. Sci. Comput. **81**(2), 820–844 (2019)
3. Bohm, M., Winters, A.R., Gassner, G.J., Derigs, D., Hindenlang, F., Saur, J.: An entropy stable nodal discontinuous Galerkin method for the resistive MHD equations. Part I: theory and numerical verification. J. Comput. Phys. **422**, 108076 (2020)
4. Burstedde, C., Wilcox, L.C., Ghattas, O.: p4est: Scalable algorithms for parallel adaptive mesh refinement on forests of octrees. SIAM J. Sci. Comput. **33**(3), 1103–1133 (2011)
5. Carpenter, M., Fisher, T., Nielsen, E., Frankel, S.: Entropy stable spectral collocation schemes for the Navier-Stokes equations: discontinuous interfaces. SIAM J. Sci. Comput. **36**(5), B835–B867 (2014)
6. Chan, J.: On discretely entropy conservative and entropy stable discontinuous Galerkin methods. J. Comput. Phys. **362**, 346–374 (2018)
7. Chen, T., Shu, C.-W.: Entropy stable high order discontinuous Galerkin methods with suitable quadrature rules for hyperbolic conservation laws. J. Comput. Phys. **345**, 427–461 (2017)
8. Chevalier, R.A.: Self-similar solutions for the interaction of stellar ejecta with an external medium. Astrophys. J. **258**, 790–797 (1982)
9. Ching, E.J., Lv, Yu., Gnoffo, P., Barnhardt, M., Ihme, M.: Shock capturing for discontinuous Galerkin methods with application to predicting heat transfer in hypersonic flows. J. Comput. Phys. **376**, 54–75 (2019)
10. Cockburn, B., Hou, S., Shu, C.-W.: The Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws. IV: the multidimensional case. Math. Comput. **54**(190), 545–581 (1990)
11. Cockburn, B., Shu, C.-W.: The Runge-Kutta discontinuous Galerkin method for conservation laws V: multidimensional systems. J. Comput. Phys. **141**(2), 199–224 (1998)
12. Dumbser, M., Fambri, F., Tavelli, M., Bader, M., Weinzierl, T.: Efficient implementation of ADER discontinuous Galerkin schemes for a scalable hyperbolic PDE engine. Axioms **7**(3), 63 (2018)

13. Dumbser, M., Loubere, R.: A simple robust and accurate a posteriori sub-cell finite volume limiter for the discontinuous Galerkin method on unstructured meshes. J. Comput. Phys. **319**, 163–199 (2016)

14. Dumbser, M., Zanotti, O., Loubere, R., Diot, S.: A posteriori subcell limiting of the discontinuous Galerkin finite element method for hyperbolic conservation laws. J. Comput. Phys. **278**, 47–75 (2014)

15. Einfeldt, B.: On Godunov-type methods for gas dynamics. SIAM J. Numer. Anal. **25**(2), 294–318 (1988)

16. Ferrand, G., Decourchelle, A., Safi-Harb, S.: Three-dimensional simulations of the thermal X-ray emission from young supernova remnants including efficient particle acceleration. Astrophys. J. **760**(1), 34 (2012)

17. Flad, D., Gassner, G.: On the use of kinetic energy preservaing DG-scheme for large eddy simulation. J. Comput. Phys. **350**, 782–795 (2017)

18. Fraschetti, F., Teyssier, R., Ballet, J., Decourchelle, A.: Simulation of the growth of the 3d Rayleigh-Taylor instability in supernova remnants using an expanding reference frame. Astron. Astrophys. **515**, A104 (2010)

19. Friedrich, L., Winters, A.R., Del Rey, D.C., Fernández, G.J., Gassner, M.P., Carpenter, M.H.: An entropy stable h/p non-conforming discontinuous Galerkin method with the summation-by-parts property. J. Sci. Comput. **77**(2), 689–725 (2018)

20. Fryxell, B., Olson, K., Ricker, P., Timmes, F.X., Zingale, M., Lamb, D.Q., MacNeice, P., Rosner, R., Truran, J.W., Tufo, H.: Flash: an adaptive mesh hydrodynamics code for modeling astrophysical thermonuclear flashes. Astrophys. J. Suppl. Ser. **131**(1), 273 (2000)

21. Gassner, G.: A skew-symmetric discontinuous Galerkin spectral element discretization and its relation to SBP-SAT finite difference methods. SIAM J. Sci. Comput. **35**(3), A1233–A1253 (2013)

22. Gassner, G.J.: A kinetic energy preserving nodal discontinuous Galerkin spectral element method. Int. J. Numer. Methods Fluids **76**(1), 28–50 (2014)

23. Gassner, G.J., Beck, A.D.: On the accuracy of high-order discretizations for underresolved turbulence simulations. Theor. Comput. Fluid Dyn. **27**(3/4), 221–237 (2013)

24. Gassner, G., Staudenmaier, M., Hindenlang, F., Atak, M., Munz, C.-D.: A space-time adaptive discontinuous Galerkin scheme. Comput. Fluids **117**, 247–261 (2015)

25. Gassner, G.J., Winters, A.R., Hindenlang, F.J., Kopriva, D.A.: The BR1 scheme is stable for the compressible Navier-Stokes equations. J. Sci. Comput. **77**, 154–200 (2018)

26. Gassner, G.J., Winters, A.R., Kopriva, D.A.: Split form nodal discontinuous Galerkin schemes with summation-by-parts property for the compressible Euler equations. J. Comput. Phys. **327**, 39–66 (2016)

27. Guo, W., Nair, R.D., Zhong, X.: An efficient WENO limiter for discontinuous Galerkin transport scheme on the cubed sphere. Int. J. Numer. Methods Fluids **81**(1), 3–21 (2016)

28. Harten, A., Lax, P.D., van Leer, B.: On upstream differencing and Godunov-type schemes for hyperbolic conservation laws. SIAM Rev. **25**(1), 35–61 (1983)

29. Hesthaven, J.S., Warburton, T.: Nodal Discontinuous Galerkin Methods. Springer, Berlin (2008)

30. Kenevsky, A.: High-order implicit-explicit Runge-Kutta time integration schemes and time-consistent filtering in spectral methods. Ph.D. dissertation, Brown University, USA (2006)

31. Kirby, R.M., Karniadakis, G.E.: De-aliasing on non-uniform grids: algorithms and applications. J. Comput. Phys. **191**, 249–264 (2003)

32. Klöckner, A., Warburton, T., Hesthaven, J.S.: Viscous shock capturing in a time-explicit discontinuous Galerkin method. Math. Model. Nat. Phenom. **6**(3), 57–83 (2011)

33. Kolda, T.G., Bader, B.W.: Tensor decompositions and applications. SIAM Rev. **51**(3), 455–500 (2009)

34. Kopriva, D.A.: Implementing Spectral Methods for Partial Differential Equations: Algorithms for Scientists and Engineers. Springer Science & Business Media, Berlin (2009)

35. Kopriva, D.A.: Stability of overintegration methods for nodal discontinuous Galerkin spectral element methods. J. Sci. Comput. **76**(1), 426–442 (2017)

36. Kopriva, D.A., Hindenlang, F.J., Bolemann, T., Gassner, G.J.: Free-stream preservation for curved geometrically non-conforming discontinuous Galerkin spectral elements. J. Sci. Comput. **79**(3), 1389–1408 (2019)

37. Kopriva, D.A., Woodruff, S.L., Hussaini, M.Y.: Computation of electromagnetic scattering with a non-conforming discontinuous spectral element method. Int. J. Numer. Methods Eng. **53**(1), 105–122 (2002)

38. Korobeinikov, V.P.: Problems of Point Blast Theory. Springer Science & Business Media, Berlin (1991)

39. Krais, N., Beck, A., Bolemann, T., Frank, H., Flad, D., Gassner, G., Hindenlang, F., Hoffmann, M., Kuhn, T., Sonntag, M., Munz, C.-D.: FLEXI: a high order discontinuous Galerkin framework for hyperbolic-parabolic conservation laws. Comput. Math. Appl. (2020)

40. Kuzmin, D.: Slope limiting for discontinuous Galerkin approximations with a possibly non-orthogonal Taylor basis. Int. J. Numer. Methods Fluids **71**(9), 1178–1190 (2012)

41. Lax, P.D., Liu, X.-D.: Solution of two-dimensional Riemann problems of gas dynamics by positive schemes. SIAM J. Sci. Comput. **19**(2), 319–340 (1998)

42. Liu, Y., Shu, C.-W., Zhang, M.: Entropy stable high order discontinuous Galerkin methods for ideal compressible MHD on structured meshes. J. Comput. Phys. **354**, 163–178 (2018)
43. Murman, S.M., Diosady, L., Garai, A., Ceze, M.: A space-time discontinuous-Galerkin approach for separated flows. In: 54th AIAA Aerospace Sciences Meeting, AIAA 2016-1059, AIAA Inc., Reston, VA (2016)
44. Parsani, M., Carpenter, M.H., Fisher, T.C., Nielsen, E.J.: Entropy stable staggered grid discontinuous spectral collocation methods of any order for the compressible Navier-Stokes equations. SIAM J. Sci. Comput. **38**(5), A3129–A3162 (2016)
45. Parsani, M., Carpenter, M.H., Nielsen, E.J.: Entropy stable discontinuous interfaces coupling for the three-dimensional compressible Navier-Stokes equations. J. Comput. Phys. **290**(C), 132–138 (2015)
46. Pazner, W., Persson, P.-O.: Analysis and entropy stability of the line-based discontinuous Galerkin method. J. Sci. Comput. **80**(1), 376–402 (2019)
47. Persson, P.-O., Peraire, J.: Sub-cell shock capturing for discontinuous Galerkin methods. In: 44th AIAA Aerospace Sciences Meeting and Exhibit, AIAA 2006-112, AIAA Inc., Reston, VA (2006)
48. Peter, D.L.: Weak solutions of nonlinear hyperbolic equations and their numerical computation. Commun. Pure Appl. Math. **7**(1), 159–193 (1954)
49. Phillip, C., Paul, R.W.: The piecewise parabolic method (ppm) for gas-dynamical simulations. J. Comput. Phys. **54**(1), 174–201 (1984)
50. Qiu, J., Zhu, J.: RKDG with WENO type limiters. In: Kroll, N., et al. (eds) ADIGMA - A European Initiative on the Development of Adaptive Higher-Order Variational Methods for Aerospace Applications. Notes on Numerical Fluid Mechanics and Multidisciplinary Design, vol. 113, pp. 67–80. Springer, Berlin (2010)
51. Roy, C.J., Nelson, C.C., Smith, T.M., Ober, C.C.: Verification of Euler/Navier-Stokes codes using the method of manufactured solutions. Int. J. Numer. Methods Fluids **44**(6), 599–620 (2004)
52. Schulz-Rinne, C.W.: Classification of the Riemann problem for two-dimensional gas dynamics. SIAM J. Math. Anal. **24**(1), 76–88 (1993)
53. Sedov, L.I.: Similarity and Dimensional Methods in Mechanics. Translation from 4th Russian edition, Academic Press, New York (1959)
54. Shu, C.-W., Osher, S.: Efficient implementation of essentially non-oscillatory shock-capturing schemes. J. Comput. Phys. **77**(2), 439–471 (1988)
55. Sod, G.A.: A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws. J. Comput. Phys. **27**(1), 1–31 (1978)
56. Sonntag, M., Munz, C.-D.: Shock capturing for discontinuous Galerkin methods using finite volume subcells. In: Fuhrmann, J., Ohlberger, M., Rohde, C. (eds) Finite Volumes for Complex Applications VII-Elliptic, Parabolic and Hyperbolic Problems. Springer Proceedings in Mathematics & Statistics, vol. 78, pp. 945–953. Springer, Cham (2014)
57. Spiegel, S.C., Huynh, H.T., DeBonis, J.R.: De-aliasing through over-integration applied to the flux reconstruction and discontinuous Galerkin methods. In: 22nd AIAA Computational Fluid Dynamics Conference, AIAA 2015-2744, AIAA Inc., Reston, VA (2015)
58. Spiteri, R.J., Ruuth, S.J.: A new class of optimal high-order strong-stability-preserving time discretization methods. SIAM J. Numer. Anal. **40**(2), 469–491 (2002)
59. Vilar, F.: A posteriori correction of high-order discontinuous Galerkin scheme through subcell finite volume formulation and flux reconstruction. J. Comput. Phys. **387**, 245–279 (2019)
60. Wintermeyer, N., Winters, A.R., Gassner, G.J., Warburton, T.: An entropy stable discontinuous Galerkin method for the shallow water equations on curvilinear meshes with wet/dry fronts accelerated by GPUs. J. Comput. Phys. **375**, 447–480 (2018)
61. Winters, A.R., Moura, R.C., Mengaldo, G., Gassner, G.J., Walch, S., Peiro, J., Sherwin, S.J.: A comparative study on polynomial dealiasing and split form discontinuous Galerkin schemes for under-resolved turbulence computations. J. Comput. Phys. **372**, 1–21 (2018)
62. Zanotti, O., Fambri, F., Dumbser, M., Hidalgo, A.: Space-time adaptive ADER discontinuous Galerkin finite element schemes with a posteriori sub-cell finite volume limiting. Comput. Fluids **118**, 204–224 (2015)
63. Zhang, X., Shu, C.-W.: On positivity-preserving high order discontinuous Galerkin schemes for compressible Euler equations on rectangular meshes. J. Comput. Phys. **229**(23), 8918–8934 (2010)
64. Zhang, X., Shu, C.-W.: Positivity-preserving high order finite difference WENO schemes for compressible Euler equations. J. Comput. Phys. **231**(5), 2245–2258 (2012)
65. Zhu, J., Qiu, J.: Hermite WENO schemes and their application as limiters for Runge-Kutta discontinuous Galerkin method, III: unstructured meshes. J. Sci. Comput. **39**(2), 293–321 (2009)
66. Zingan, V., Guermond, J.-L., Morel, J., Popov, B.: Implementation of the entropy viscosity method with the discontinuous Galerkin method. Comput. Methods Appl. Mech. Eng. **253**, 479–490 (2013)