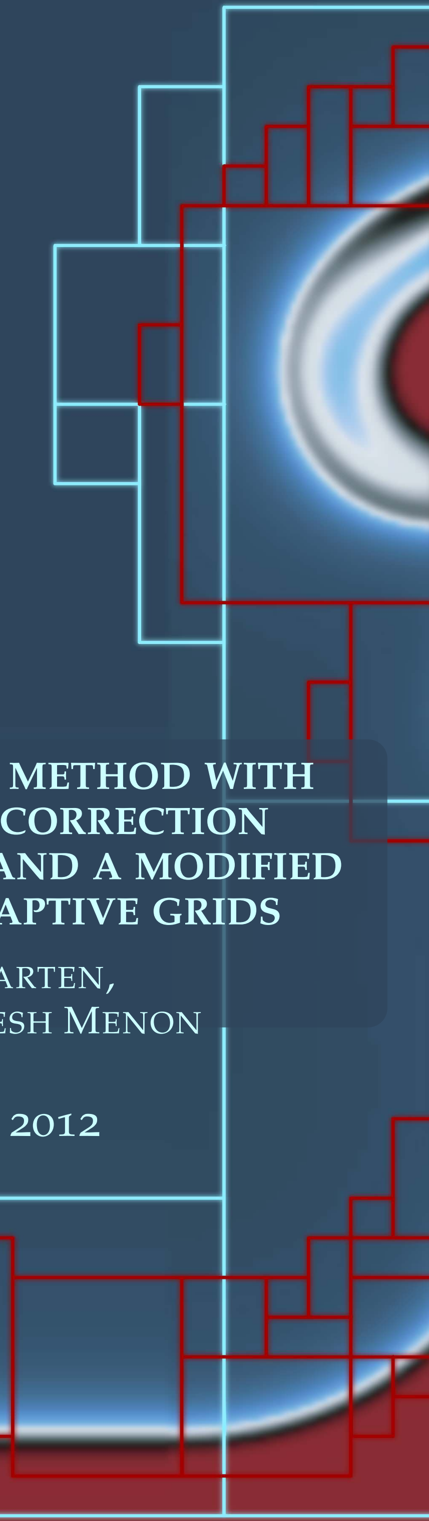
Communications in Applied Mathematics and Computational Science

DISCONTINUOUS GALERKIN METHOD WITH THE SPECTRAL DEFERRED CORRECTION TIME-INTEGRATION SCHEME AND A MODIFIED MOMENT LIMITER FOR ADAPTIVE GRIDS

LEANDRO D. GRYNGARTEN,
ANDREW SMITH AND SURESH MENON

vol. 7 no. 2 2012



msp

COMM. APP. MATH. AND COMP. SCI.
Vol. 7, No. 2, 2012
dx.doi.org/10.2140/camcos.2012.7.133

DISCONTINUOUS GALERKIN METHOD WITH THE SPECTRAL DEFERRED CORRECTION TIME-INTEGRATION SCHEME AND A MODIFIED MOMENT LIMITER FOR ADAPTIVE GRIDS

LEANDRO D. GRYNGARTEN, ANDREW SMITH AND SURESH MENON

The discontinuous Galerkin (DG) method is combined with the spectral deferred correction (SDC) time integration approach to solve the fluid dynamic equations. The moment limiter is generalized for nonuniform grids with hanging nodes that result from adaptive mesh refinement. The effect of characteristic, primitive, or conservative decomposition in the limiting stage is studied. In general, primitive variable decomposition is a better option, especially in two and three dimensions. The accuracy-preserving total variation diminishing (AP-TVD) marker for troubled-cell detection, which uses an averaged-derivative basis, is modified to use the Legendre polynomial basis. Given that the latest basis is generally used for DG, the new approach avoids transforming to the averaged-derivative basis, what results in a more efficient technique. Further, a new error estimator is proposed to determine where to refine or coarsen the grid. This estimator is compared against other estimator used in the literature and shows an improved performance. Canonical tests in one, two, and three dimensions are conducted to show the accuracy of the solver.

1. Introduction

The discontinuous Galerkin (DG) method belongs to the finite element (FE) family and uses a piecewise discontinuous space for the test function and the numerical solution [7]. The use of the same function space for the test function and solution defines all Galerkin methods. Usually, the basis to form the space is composed of Legendre [7] or Lagrange [10] polynomials, although other options have been studied in the literature [43]. The discontinuity is localized at the boundary of each element and the coupling between elements is done by computing fluxes as in finite volume (FV) schemes, e.g., using an approximate Riemann solver. This kind of coupling allows DG to formulate each element locally, making the implementation

MSC2010: 35L65, 35L67, 65L06, 65M50, 65M60.
Keywords: discontinuous Galerkin, moment limiter, high-order accuracy, adaptive mesh, troubled-cell detector, spectral deferred correction.

highly parallelizable, *h-p* adaptivity friendly, compatible with complex geometries, and is capable of achieving high-orders of accuracy even with unstructured grids and hanging nodes (e.g., see Remacle et al. [33]). Given that DG is a result of FE and FV, the terms element and cell are generally used indistinctly in this context.

The time integration scheme most widely used has been the ubiquitous 3rd-order TVD Runge–Kutta (RK) method, leading to what is known as the RKDG method [5; 4; 3; 6]. Given that DG has the ability to easily achieve high-order spatial accuracy, some effort to maintain comparable time accuracy has been reported [40]. Under some conditions, especially with higher order derivatives, the time step required for stability of the RKDG method can be very limiting. Recently, Xu and Shu [40] suggested that the spectral deferred correction (SDC) method, derived by Dutt et al. [8], may be an alternative time stepping scheme. It has been shown that SDC can be used in an explicit, semiimplicit, or fully implicit form, and it is easy to extend to high-order accuracy in time [27]. Xia et al. [38] studied a semiimplicit SDC method, in addition to other alternative techniques, to use with the local discontinuous Galerkin (LDG) method. SDC combined with DG (SDC-DG) has not yet been used extensively for practical applications. Grooss and Hesthaven [14] used a semiimplicit SDC to solve the incompressible Navier–Stokes with free-surface flows. Here, we report on new results that demonstrate the potential of SDC-DG with explicit integration and compare it against RKDG. Even though Gottlieb et al. [13] presented RK methods of order higher than 3, these schemes are very difficult to derive, while the extension of SDC to any order is straightforward. In addition, TVD-RK methods of 4th-order or greater require the governing equation to be invariant to time reversal [12; 13]. The Euler equations are invariant to this transformation, but the Navier–Stokes (NS) equations are not. Although we do not use the NS equations in this report, viscous fluxes will be included in future studies. Hence, TVD-RK schemes of 4th-order or higher are not considered here. The possibility of an SDC method with the strong stability preserving (SSP) property was studied by Gottlieb et al. [11] and more extensively by Liu et al. [26]. Note that TVD schemes are SSP schemes that were originally derived using the total variation norm [13], instead of a generic norm. Therefore, in practice the TVD and the SSP properties are equivalent, but TVD could be considered a particular case of SSP. Liu et al. [26] showed that SSP-SDC algorithms can be obtained, but the derivation gets very complicated as the order increases and the CFL coefficient is smaller than for the SSP-RK. Thus, in the current study we use SDC without the SSP property.

The current method also combines the SDC-DG approach with adaptive mesh refinement (AMR) to dynamically and locally refine or coarsen the grid based on an estimation of the numerical error. Issues with the implementation such as hanging nodes, particularly in quadrilateral or hexahedral grids are addressed. The

DG formulation works well with AMR because of its local nature [33] and its performance is demonstrated in this paper.

As with other numerical approaches, it is well known that DG methods may cause nonphysical oscillations close to discontinuities due to the Gibbs phenomenon, especially when higher order schemes are used because of lower numerical dissipation. Therefore, some approach to "limit" this effect is needed. One common technique consists of applying limiters inherited from FV techniques, several of which have been developed in the last two decades. Cockburn and Shu [5] demonstrated a modified *minmod* limiter for the DG method, but it has the disadvantages of dropping the order of accuracy when it is activated and relies on a user-defined parameter to make it total variation bounded (TVB) instead of total variation diminishing (TVD). Qiu and Shu [31] showed that the weighted essentially nonoscillatory (WENO) approach, borrowed from FV, can smooth the un-desired oscillations but increases the size of the stencil and loses the subcell information that DG provides. In a later study, Qiu and Shu [30] used a modified WENO scheme based on Hermite polynomials to reduce the stencil.

Other limiters, such as the moment limiter (ML), originally proposed by Biswas et al. [2] for uniform grids and further improved, e.g., by Krivodonova [22], has also been proposed for DG applications. The ML is generally applied to a Legendre polynomial basis limiting the conservative or the characteristic variables. Yang and Wang [42] modified the ML for unstructured grids for a spectral difference (SD) method, applying it to a polynomial basis based on the averaged derivatives along the cell, instead of estimating the derivatives at the cell center as in [22]. The hierarchical reconstruction (HR) method, introduced by Liu et al. [24], was applied to DG with a WENO-type reconstruction at each hierarchical level [41]. In this approach characteristic decomposition is not used, but rather small overshoots/undershoots appear especially as the order of accuracy is increased [25]. For DG schemes with very high order elements, artificial dissipation to smooth out discontinuities has also been proposed [16; 28; 1]. In this paper the ML as presented in [22] is modified for nonuniform grids with hanging nodes. The ML is usually applied to characteristic variables, which is only consistent in a one-dimensional sense. Therefore, we study the consequences of limiting the conservative, primitive, or characteristic variables to later apply it to multidimensional cases.

Even though good limiters tend to keep the original order of accuracy in smooth regions, they may increase the error slightly [29; 22]. Hence, the application of such limiters within the domain needs to be minimized. This task is carried out by what is usually called a troubled-cell detector, which identifies the cells that may be becoming oscillatory or unstable, and thus require a limiter. Moreover, if the detector is computationally faster than the limiter, the speed of the solver can be increased by reducing the number of cells where the limiter is applied. In the past, several limiters were adapted as detectors [29], and the ones with best success are the minmod-based TVB limiter [5], the shock-detector by Krivodonova et al. [21] (KXRCF), and the indicator based on Harten's subcell resolution [15]. In [42], the accuracy-preserving TVD (AP-TVD) detector is suggested in an SD frame and compared against the other detectors just mentioned above and was shown to produce better agreement. Therefore, we adapt the AP-TVD to the DG method with some additional modifications, as reported below.

AMR requires an indicator to determine where to refine or coarsen the grid based on an estimated numerical error. This numerical error depends on the scheme, thus error estimators used in FV or finite differences (FD) are not valid here. Considerable research has been invested in estimating the numerical error for the DG method for conservative hyperbolic equations [9], but usually these approaches are computationally expensive and therefore inefficient. Faster, though perhaps less accurate methods have also been derived for DG. Remacle et al. [33] used a simple error estimator based on the jump between elements, which is the same principle as used in the shock-detector KXRCF. Trouble-cell detectors have also been used as error estimators [34; 45]. Zhu et al. [45] compared a few of them and found that KXRCF provided very good results for typical one-dimensional shock problems. In addition, Leicht and Hartmann [23] used the jump between elements to determine the direction for anisotropic refinement. In this study we propose a new estimator which results from a combination of some of these detectors and has better efficiency.

The current paper is organized in the following way. Section 2 introduces the governing equation relevant to this study. Section 3 presents the numerical schemes and algorithm behind the solver. Section 4 includes the test cases that show the success of the method being proposed. Finally, Section 5 summarizes the observations and suggests areas where future work is necessary.

2. Governing equations

The governing equations are the conservation laws written in the general form

$$\begin{cases} \dfrac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot \boldsymbol{F}(\boldsymbol{u}) = 0 & \text{for } t > 0, \\ \boldsymbol{u} = \boldsymbol{u}_0 & \text{for } t = 0, \end{cases} \tag{1}$$

where $\boldsymbol{u}$ is the solution vector, $\boldsymbol{F}$ is the inviscid flux, and $\boldsymbol{u}_0$ is the initial value. Unless specified otherwise, they correspond to the Euler equation, so that

$$\boldsymbol{u} = (\rho, \rho v_1, \rho v_2, \rho v_3, \rho E_T)^T \tag{2}$$

and

$$\boldsymbol{F}_1 = \begin{pmatrix} \rho v_1 \\ \rho v_1^2 + p \\ \rho v_1 v_2 \\ \rho v_1 v_3 \\ (\rho E_T + p) v_1 \end{pmatrix} \quad \boldsymbol{F}_2 = \begin{pmatrix} \rho v_2 \\ \rho v_1 v_2 \\ \rho v_2^2 + p \\ \rho v_2 v_3 \\ (\rho E_T + p) v_2 \end{pmatrix} \quad \boldsymbol{F}_3 = \begin{pmatrix} \rho v_3 \\ \rho v_1 v_3 \\ \rho v_2 v_3 \\ \rho v_3^2 + p \\ (\rho E_T + p) v_3 \end{pmatrix},$$

where ρ is the density, v_j is the velocity component in the x_j direction, p is the pressure, E_T is the total energy defined as $E_T = e + \sum_{i=1}^{3} \frac{1}{2} v_i^2$ where e is the internal energy. The ideal gas equation of state is $\rho e = p/(\gamma - 1)$ where γ is the specific heat ratio and it is assumed constant. In addition, the speed of sound c is

$$c = \sqrt{\gamma \frac{p}{\rho}}. \tag{3}$$

The state vector $\boldsymbol{u}$ in (2) is given in conservative form. The state vector for the primitive form as used in this study is

$$\boldsymbol{u}_p = (\rho, v_1, v_2, v_3, p)^T . \tag{4}$$

Although in this paper we focus on the Euler equations to show the ability and accuracy of the proposed method, extension to full Navier–Stokes equations are also being evaluated and will be reported in the near future.

3. Numerical method

The DG method is applied to the conservation law described in (1). The domain, Ω, is divided into N nonoverlapping elements:

$$\Omega = \bigcup_{l=1}^{N} \Omega_l. \tag{5}$$

The solution vector $\boldsymbol{u}$ is approximated per element by $\boldsymbol{U}_l$, defined by the basis ϕ:

$$\boldsymbol{U}_l = \sum_{i=0}^{p} \phi_i \boldsymbol{c}_{i,l}, \tag{6}$$

where p defines the order of the finite element and $\boldsymbol{c}_{i,l}$ is the weight corresponding to each element of the basis ϕ. After multiplying by the test function, which is equal to the basis ϕ, and integrating by parts we arrive at the weak form of the DG method [7]:

$$\begin{cases} \displaystyle\int_{\Omega_l} \phi \frac{\partial \boldsymbol{U}_l}{\partial t} dV - \int_{\Omega_l} \nabla\phi \cdot \boldsymbol{F}(\boldsymbol{U}_l)\, dV + \int_{\partial\Omega_l} \phi \widehat{F}(\boldsymbol{U}^-, \boldsymbol{U}^+)\, dS = 0 & \text{for } t > 0, \\ \displaystyle\int_{\Omega_l} \phi \boldsymbol{U}_l\, dV = \int_{\Omega_l} \phi \boldsymbol{u}_0\, dV & \text{for } t = 0, \end{cases} \tag{7}$$

with appropriate boundary conditions. Here, $\widehat{F}$ is a numerical flux normal to the boundary of the element and needs to be properly defined given that it is computed at the face of the elements, which may be discontinuous. $\boldsymbol{U}^-$ is the value of $\boldsymbol{U}_l$ according to the current element l at the face and $\boldsymbol{U}^+$ is the value of $\boldsymbol{U}_m$ at the face based on the neighboring element m.

The numerical flux should be an exact or approximate Riemann solver. Here the local Lax–Friedrichs flux is used as it is known to provide good results and is simple to compute [5]. In this study, the spatial integration in (7) is done with a full quadrature rule using Lobatto points [20; 5; 4; 3; 6].

3.1. ***Time integration.*** Time integration is conducted explicitly using the Runge–Kutta (RK) method or the spectral deferred correction (SDC) method. Both approaches treat the governing equations as a system of ordinary differential equations (ODE):

$$\frac{d\boldsymbol{u}}{dt} = G(t, \boldsymbol{u}), \tag{8}$$

where $G(t, \boldsymbol{u}) = -\nabla \cdot \boldsymbol{F}(\boldsymbol{u})$. In this study, unless specified otherwise, for elements of polynomial order p a time integration of order $p+1$ is used. Unless specified otherwise, the time step is given by

$$\Delta t = \min_{l=1\ldots N}\left[\frac{C}{2p_l+1}\cdot \min_{i=1\ldots n_d}\left(\frac{\Delta x_{i,l}}{v_{i,l}+c_l}\right)\right], \tag{9}$$

where n_d is the number of dimensions and C is a constant. The flow velocity $v_{i,l}$ and the speed of sound c are considered at the centroid of element l. We use $C = 0.5$ unless specified otherwise. It is usually replaced by 1.0 (see [7]), however, in this study we choose 0.5 to be more conservative. Maximum stable time-step size for RKDG has been shown elsewhere [7]. Stability limits for SDC-DG have not been studied in the literature, at least to the author's knowledge. Equation (9) turns out to provide a stable condition for the tests presented in this study also for SDC-DG when the order in time is equal to $p+1$.

3.1.1. *The Runge–Kutta method.* The Runge–Kutta method is a well known family of schemes. The current study used the total variation diminishing RK (TVD-RK) of second and third orders [35], which can be summarized in three steps:

Step 1:

$$u^0 = u_n. \tag{10}$$

Step 2:

$$u^i = \sum_{l=0}^{i-1} \alpha_{il} w^{il}, \quad w^{il} = u^l + \frac{\beta_{il}}{\alpha_{il}} \Delta t G\left(u^l, t + \Delta t \cdot d_l\right), \quad \text{for } i = 1, \ldots, K. \tag{11}$$

Step 3:

$$u_{n+1} = u^K, \tag{12}$$

where the superindexes of u determine intermediate steps between u_n and u_{n+1}. The parameters are given in Table 1. A good property for these 2nd- and 3rd-order TVD schemes is that if for some seminorm $|\cdot|$, we have that $|w^{il}| \leq |u^l|$, then $|u_{n+1}| \leq |u_n|$.

Order	α_{il}			β_{il}			d_l
2	1			1			0
	1/2	1/2		0	1/2		1
3	1			1			0
	3/4	1/4		0	1/4		1
	1/3	0	2/3	0	0	2/3	1/2

Table 1. Parameters for TVD-RK of order 2 and 3.

3.1.2. *The spectral deferred correction method.* Although details of this method are given elsewhere [8; 27; 26; 40], we include the main algorithm for completeness. The scheme is based on first-order explicit integration of substeps and iterative correction [8]. For stiff problems, the scheme can be varied with a more implicit character, but only the explicit method is addressed here. Each time step $[t_n, t_{n+1}]$ is divided into J substeps: $t_n = t_{n,0} < t_{n,1} < \cdots < t_{n,m} < t_{n,m+1} < \cdots < t_{n,J} = t_{n+1}$. These points are chosen as quadrature points (Lobatto points in the current study). This approach makes the scheme more stable because it avoids a uniform distribution and leads to the spectral characteristic of the scheme [8]. This property is important to stabilize higher orders. Initially, the governing equations are integrated with a first-order explicit integration from t_n to t_{n+1} using $t_{n,m}$ points:

$$u^1_{n,m+1} = u^1_{n,m} + \Delta t_{n,m} G\left(t_{n,m}, u^1_{n,m}\right) \quad \text{for } m = 0, \ldots, J-1, \tag{13}$$

where $u^1_{n,0} = u_n$ and $\Delta t_{n,m} = t_{n,m+1} - t_{n,m}$.

Now K iterations are computed for $k = 1, \ldots, K$ and $m = 0, \ldots, J-1$ (m being the inner loop):

$$u^{k+1}_{n,m+1} = u^{k+1}_{n,m} + \theta \Delta t_{n,m} \left(G(t_{n,m}, u^{k+1}_{n,m}) - G(t_{n,m}, u^k_{n,m})\right) + I^{m+1}_m \left(G(t_{n,m}, u_{n,m})\right), \tag{14}$$

where $0 \leq \theta \leq 1$ and $I^{m+1}_m(G(t_{n,m}, u_{n,m}))$ is the integral of the interpolating polynomial along the quadrature points:

$$I^{m+1}_m(G(t_{n,m}, u_{n,m})) = \int_{t_{n,m}}^{t_{n,m+1}} G(\tau, u(\tau))\, d\tau. \tag{15}$$

Finally, $u_{n+1} = u_{n,J}^{K+1}$. Here, we use $\theta = 1$ as in the original study [8] and $K = J - 1$. For the cases studied in this report, we observed that neglecting the second term on the right-hand side, i.e., using $\theta = 0$, provides similar results but with greater numerical error.

3.2. ***The basis.*** Several options can be used to form the finite element space. Our basis ϕ is built on the Legendre polynomials P_i, which leads to an orthogonal, hierarchical, polynomial basis — an advantage in comparison with computationally more expensive functions (e.g., trigonometric or exponential). Another numerical advantage is an advantage in comparison with computationally more expensive functions the lower condition number of the Vandermonde matrix, which transforms from modal space to nodal space. The mass matrix is diagonal when the basis is orthogonal and the Jacobian is constant inside the element; indeed, if the basis is correctly normalized and the Jacobian is constant, the mass matrix is just the identity matrix times the Jacobian.

The normalized Legendre polynomials are given by

$$\phi_i = P_i \sqrt{\frac{2i+1}{2}} \quad \text{for } i = 0, \ldots, p, \tag{16}$$

and they are orthonormal: $\int_{\Omega_l} \phi_i \phi_j dV = \delta_{ij}$, where δ_{ij} is Dirac's delta function.

For quadrilateral and hexahedral elements the basis can easily be generated from the 1D basis by applying a tensor product. Thus, in 2D we have

$$\phi_{ij}(\xi, \eta) = \phi_i(\xi)\, \phi_j(\eta), \tag{17}$$

and in 3D

$$\phi_{ijk}(\xi, \eta, \zeta) = \phi_i(\xi)\, \phi_j(\eta)\, \phi_k(\zeta). \tag{18}$$

3.3. ***Adaptive mesh refinement.*** The solver relies on a tree to handle the hierarchical structure of the grid adaptations. The initial grid is composed of *root* cells, corresponding to the lowest level. Each cell can have children. A cell that does not have children is called a *leaf* cell. If a root cell does not have children it is also tagged as a leaf cell. The root cells correspond to level 1 and the maximum level is given by $\ell_{\max}$, which may depend on the problem. Each face of every cell has to be connected to a neighbor or to a boundary element. A cell can connect to a neighbor at the same level or at a lower level, but never at a higher level. In addition, there is a ghost tree to handle the ghost cells for interprocessor communications. Details about the tree structures are given in [19; 17].

When a cell is marked for refinement it is split into two, four, or eight, in dimension 1, 2, or 3. The variables from the parent are projected onto the children with an identity projection. On the other hand, when all the children of a cell are marked for coarsening, the variables from the children are projected onto the parent cell with a least-squares projection. Note that if the order p is kept constant, no data is lost when refinement is done; however, data is lost when coarsening is done.

Cells are marked for adaptation based on an error estimator. The error ϵ_l of cell l is then normalized by the maximum error $\epsilon_{\max}$ found in the whole domain. Then, a logarithmic scale is applied as in [9]. The current hierarchical level in the tree for cell l is $\ell(l)$. A target level ℓ_t is estimated as

$$\ell_t = \max\left(1, \ell_{\max} - \mathrm{INT}\left(\log(\epsilon_{\max}/\epsilon_j)/\log d\right)\right) \tag{19}$$

where d is a parameter that determines the sensibility of the refinement, the larger its value, the more refinement will be done. Even though the accuracy is expected to increase as d is raised, the computational cost will be higher too. The default value adopted here is $d = 10$ as in [9], which is a good balance between computational cost and accuracy. If ℓ_t is greater than $\ell(l)$ then cell l is marked for refinement. If ℓ_t is lower than $\ell(l)$ then cell l is merged for coarsening. Note that for coarsening to actually be feasible, all the children have to be marked for coarsening.

The level difference between neighboring cells is not allowed to be larger than 1. For example, suppose cells 1 and 2 are neighbors, with $\ell(1) = 3$ and $\ell(2) = 4$. If cell 2 is marked for refinement, then cell 1 will be marked for refinement also.

For the sake of simplicity and computational speed, a simple error estimator is used here. More accurate approaches are slower and may increase the overhead, making the adaptivity too costly.

Zhu et al. [45] compared a few different shock-detectors as estimators for refinement, and concluded that the most efficient based on their 1D discontinuous problems was the KXRCF [21]:

$$\epsilon_{A,l} = \frac{\left|\displaystyle\int_{\delta\Omega_l^-} (U^- - U^+)\, dS\right|}{h_l^{\frac{p+1}{2}} \displaystyle\int_{\delta\Omega_j^-} dS\, \|U_l\|}, \tag{20}$$

where U is some relevant variable, $\delta\Omega_l^-$ is the element boundary where the velocity is going into the element, h_l is the radius of the circle circumscribed to the element l, and the norm is based on an element average. In [33] the following error estimator was used for element l:

$$\epsilon_{B,l} = \int_{\Omega_l} |U^- - U^+|\, dS. \tag{21}$$

Here, we combine the best of (20) and (21) to obtain

$$\epsilon_{C,l} = \frac{\int_{\delta\Omega_l} |U^- - U^+| dS}{\int_{\delta\Omega_l} dS.} \tag{22}$$

In the current implementation the error is already normalized by the maximum error in the domain — see (19) — and so the additional normalization needed in (20) is not required. For the Euler equation two error estimators based on the density and the total energy are used. Below we refer to the estimators in (20)–(22) as KXRCF, JUMP1, and JUMP2, respectively. The difference between JUMP1 and JUMP2 can only be observed in 2 and 3 dimensions, so for the 1D cases JUMP1 is not used.

3.4. ***Moment-limiter for nonuniform grids.*** The limiting strategy of the ML is shown below for 1D. However, for completeness, the 2D and 3D extensions are discussed in the Appendix. In Section 3.4.2, we extend the original ML to nonuniform grids for 1D, but its extension to higher dimensions is trivial, except for when a neighbor is split due to refinement, in which case the average of the two is used, and when the neighbor is coarser, in which case a virtual refinement of the neighbor is done. This last step has no analytical complexity, but its implementation may not be trivial. Figure 1 shows the stencil used for limiting purposes when coarser, finer, or equal level neighbors are present. For the neighbor on the right-hand side, a virtual refinement was created, similar to the idea of partial neighboring cells in [41]. The neighbors on the top are virtually merged.

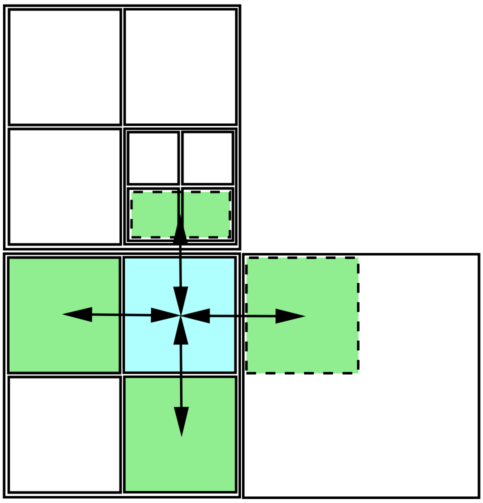


Figure 1. Example of a 2D stencil used for limiting when coarser, finer, or equal-level neighbors are present.

3.4.1. *The moment-limiter concept.* The idea is to limit the i-th derivative in x of U_l in the following way:

$$\frac{\partial^i \tilde{U}_l}{\partial x^i} = \operatorname{minmod}\left(\frac{\partial^i U_l}{\partial x^i}, \beta_i D_i^+, \beta_i D_i^-\right) \tag{23}$$

where

$$\operatorname{minmod}(a,b,c)=\begin{cases}\operatorname{sgn} a\ \min(|a|,|b|,|c|) & \text{if } \operatorname{sgn} a=\operatorname{sgn} b=\operatorname{sgn} c,\\ 0 & \text{otherwise,}\end{cases} \tag{24}$$

and $\tilde{U}_l$ is the solution U_l after the limiter is applied. $D_i^{+/-}$ is an estimation of the i-th derivative based on one-sided differences:

$$D_i^{+}=\frac{\dfrac{\partial^{i-1}U_{l+1}}{\partial x^{i-1}}-\dfrac{\partial^{i-1}U_{l}}{\partial x^{i-1}}}{\bar{x}_{l+1}-\bar{x}_l},\qquad D_i^{-}=\frac{\dfrac{\partial^{i-1}U_{l}}{\partial x^{i-1}}-\dfrac{\partial^{i-1}U_{l-1}}{\partial x^{i-1}}}{\bar{x}_{l}-\bar{x}_{l-1}}, \tag{25}$$

where $\bar{x}_l$ is the location of the centroid of element l, and β_i is a parameter to control the sensibility of the limiter. If there is a boundary condition against one of the faces of the element, then that side is neglected in (23).

In the literature (e.g., [22]) it is recommended to apply limiting to the characteristic variables when a system of equations is being solved. This means replacing (23) by

$$\left(\boldsymbol{L}\frac{\partial^i\tilde{\boldsymbol{U}}_l}{\partial x^i}\right)_k=\operatorname{minmod}\left(\left(\boldsymbol{L}\frac{\partial^i\boldsymbol{U}_l}{\partial x^i}\right)_k,\ \beta_i(\boldsymbol{L}\boldsymbol{D}_i^{+})_k,\ \beta_i(\boldsymbol{L}\boldsymbol{D}_i^{-})_k\right), \tag{26}$$

where $\boldsymbol{L}$ is a matrix composed by the left eigenvectors of the Jacobian $\partial\boldsymbol{F}/\partial\boldsymbol{u}$, and the subindex k refers the k-th characteristic variable. Each characteristic variable is limited individually and this means that if a variable in a given element is not limited the others can still be limited. If limiting is applied in an element the resulted characteristic variables have to be converted back to the conservative variables, multiplying the characteristic variables by the inverse of $\boldsymbol{L}$, which is composed by the right eigenvectors of $\partial\boldsymbol{F}/\partial\boldsymbol{u}$. In addition, if one wants to use primitive variables for this stage, $\boldsymbol{L}$ should be replaced by the Jacobian, $\partial\boldsymbol{u}_p/\partial\boldsymbol{u}$, where $\boldsymbol{u}_p$ is the state vector in primitive variables.

The algorithm to apply the limiter is the following:

(1) Apply (23) for $i=p$ to every element. If

$$\frac{\partial^i\tilde{U}_l}{\partial x^i}=\frac{\partial^i U_l}{\partial x^i}, \tag{27}$$

then mark the element as not needing limiting anymore.

(2) Apply (23) for $i=p-1$ to every element that still needs to be limited.

(3) Continue for $i=p-2,\dots,1$ or until no element requires limiting.

Note that only the derivatives are modified, not the mean value; thus the limiter does not violate the conservation property.

As in [22], we also add the following steps at the end of the limiting procedure for each element to avoid nonphysical values:

(1) If any integration point has a nonphysical state (e.g., negative pressure), make all the quadratic and higher-order moments equal to zero and go to step 2, otherwise the procedure is completed.

(2) If any integration point still has a nonphysical state (e.g., negative pressure), make all the linear moments equal to zero. This makes the solution piecewise constant.

Obviously, when these steps are applied the accuracy is forced to drop locally. Nonetheless, this is not needed often.

3.4.2. *The ML using a Legendre basis.* The $(i-1)$-th derivative with respect to x of $\boldsymbol{U}_l$, given in (6), can be expressed as

$$\frac{\partial^{i-1}\boldsymbol{U}_l}{\partial x^{i-1}} = \left(\frac{2}{\Delta x_l}\right)^{i-1}\left[\sqrt{\frac{2i-1}{2}}\,(2i-3)!!\,\boldsymbol{c}_{l,i-1} + \frac{\partial^{i-1}}{\partial \xi^{i-1}}\sum_{k=i}^{p}\boldsymbol{c}_{l,k}\phi_k(\xi)\right] \tag{28}$$

and the i-th derivative in x of (6) can be expressed as

$$\frac{\partial^{i}\boldsymbol{U}_l}{\partial x^{i}} = \left(\frac{2}{\Delta x_l}\right)^{i}\left[\sqrt{\frac{2i+1}{2}}\,(2i-1)!!\,\boldsymbol{c}_{l,i} + \frac{\partial^{i}}{\partial \xi^{i}}\sum_{k=i+1}^{p}\boldsymbol{c}_{l,k}\phi_k(\xi)\right], \tag{29}$$

where Δx_l is the length of element l.

At the same time, the i-th derivative could be estimated from the forward or backward differences of $\partial^{i-1}\boldsymbol{U}_l/\partial x^{i-1}$:

$$\frac{\partial^{i}\boldsymbol{U}_l}{\partial x^{i}} = \left(\frac{\partial^{i-1}U_{l+1}}{\partial x^{i-1}} - \frac{\partial^{i-1}U_{l}}{\partial x^{i-1}}\right)\frac{2}{\Delta x_{l+1}+\Delta x_l}, \tag{30}$$

$$\frac{\partial^{i}\boldsymbol{U}_l}{\partial x^{i}} = \left(\frac{\partial^{i-1}U_{l}}{\partial x^{i-1}} - \frac{\partial^{i-1}U_{l-1}}{\partial x^{i-1}}\right)\frac{2}{\Delta x_{l}+\Delta x_{l-1}}. \tag{31}$$

Therefore, ignoring higher order derivatives we obtain

$$\boldsymbol{c}_{l,i} = \frac{2\vartheta_+}{1+\vartheta_+}\sqrt{\frac{2i-1}{2i+1}}\,\frac{1}{2(2i-1)}\left(\vartheta_+^{i-1}\boldsymbol{c}_{l+1,i-1} - \boldsymbol{c}_{l,i-1}\right), \tag{32}$$

$$\boldsymbol{c}_{l,i} = \frac{2\vartheta_-}{1+\vartheta_-}\sqrt{\frac{2i-1}{2i+1}}\,\frac{1}{2(2i-1)}\left(\boldsymbol{c}_{l,i-1} - \vartheta_-^{i-1}\boldsymbol{c}_{l-1,i-1}\right), \tag{33}$$

where $\vartheta_- = \Delta x_l/\Delta x_{l-1}$ and $\vartheta_+ = \Delta x_l/\Delta x_{l+1}$. Note that if the grid is uniform $\vartheta_- = 1$, $\vartheta_+ = 1$, and the derived equations converge to the solution in [22]. Thus, the difference between the current derivation and the one in [22] starts in (30) and (31), where we do not assume a constant Δx.

One could apply the limiter as

$$\tilde{\boldsymbol{c}}_{l,i} = \operatorname{minmod}\left(\boldsymbol{c}_{l,i},\, \Delta_i^+,\, \Delta_i^-\right), \tag{34}$$

where Δ^+ and Δ^- are the right-hand sides of (32) and (33). However, to make the limiter less numerically diffusive, [22] uses an expression equivalent to

$$\tilde{\boldsymbol{c}}_{l,i} = \operatorname{minmod}\left(\boldsymbol{c}_{l,i}, 2\,(2i-1)\,\Delta_i^+, 2\,(2i-1)\,\Delta_i^-\right). \tag{35}$$

This equation should be the actual implementation of what (23) represents. The same procedure is easily extended to 2D and 3D. Note that this formulation is equivalent to what was presented in [22] except for the generalization for nonuniform grids and how to handle neighbors of different adaptive level.

3.5. ***Troubled-cell detector.*** The detector presented in this study is a modification of the AP-TVD detector presented in [42] for a spectral difference (SD) scheme. The adapted technique consists of two steps:

1. For each cell l compute

$$\bar{\boldsymbol{U}}_{\max,l} = \max\left(\bar{\boldsymbol{U}}_{l-1}, \bar{\boldsymbol{U}}_l, \bar{\boldsymbol{U}}_{l+1}\right), \tag{36}$$

$$\bar{\boldsymbol{U}}_{\min,l} = \min\left(\bar{\boldsymbol{U}}_{l-1}, \bar{\boldsymbol{U}}_l, \bar{\boldsymbol{U}}_{l+1}\right), \tag{37}$$

   where $\bar{\boldsymbol{U}}_l$ indicates the average of $\boldsymbol{U}$ in cell l. If for any node i in element l we have $\boldsymbol{U}_{i,l} > 1.001\,\bar{\boldsymbol{U}}_{\max,l}$ or $\bar{U}_{i,l} < 0.999\,\bar{\boldsymbol{U}}_{\min,l}$, then proceed to step 2; else the element is not marked.

2. For each dimension j the idea is to compute

$$\frac{\partial^2 \tilde{\boldsymbol{U}}_l}{\partial x_j^2} = \operatorname{minmod}\left(\frac{\partial^2 \boldsymbol{U}_l}{\partial x_j^2},\ \beta\frac{\frac{\partial \boldsymbol{U}_{l+1}}{\partial x_j} - \frac{\partial \boldsymbol{U}_l}{\partial x_j}}{x_{l+1}-x_j},\ \beta\frac{\frac{\partial \boldsymbol{U}_l}{\partial x_j} - \frac{\partial \boldsymbol{U}_{l-1}}{\partial x_j}}{x_l - x_{l-1}}\right). \tag{38}$$

   The derivatives are estimated from the Legendre polynomials as in 3.4.2, so the implementation of (38) is

$$\tilde{\boldsymbol{c}}_{l,2} = \operatorname{minmod}\left(\boldsymbol{c}_{l,2},\ \varrho\vartheta_+\frac{\vartheta_+\boldsymbol{c}_{l+1,1} - \boldsymbol{c}_{l,1}}{1+\vartheta_+},\ \varrho\vartheta_-\frac{\boldsymbol{c}_{l,1} - \boldsymbol{c}_{l-1,1}\vartheta_-}{1+\vartheta_-}\right) \tag{39}$$

   where $\varrho = 2\sqrt{3/5}$. If $\tilde{\boldsymbol{c}}_{2,l} \neq \boldsymbol{c}_{2,l}$ the cell is marked for limiting. According to [42]; β is a parameter between 1 and 2, the higher its value the less dissipative the scheme will be. We use $\beta = 2$ to make it consistent with the ML used here.

There are two main differences in the current limiter with respect to the AP-TVD developed earlier [42]. The first one is in step 1 where every node in the element is tested, while in [42] only the nodes at element boundaries are checked. The second difference is in the way the derivatives are estimated in step 2, averaged-derivatives were used in [42], while here we suggest using estimations of the derivatives, as done in the ML based on Legendre polynomials to avoid computing the averaged-derivatives. Thus, we call the current detector the moment-based AP-TVD or MB-AP-TVD.

Usually, the detection is done based on the conservative variables, instead of transforming to characteristic or primitive variables, in order to keep this stage as computationally cheap as possible.

4. Results and discussion

Various test cases used in past studies are used to establish the capability of this new numerical algorithm. In addition, we use some 2D and 3D cases to demonstrate the potential of the method for more complex problems. The details of the test cases and the rationals for them are summarized in Table 2.

Test case	Purpose
Order of convergence – linear	Order of accuracy in space with a smooth linear problem.
Order of convergence – nonlinear	Order of accuracy in space with a smooth nonlinear problem.
Accuracy in time	Order of accuracy in time with a smooth solution using RK and SDC.
Advection of mixed pulses	Order of accuracy with a nonsmooth solution with and without detector.
High order for a smooth and nonsmooth solution	Order of accuracy with a localized discontinuity.
Sod's problem	Limiting variables, with and without detector, and with and without AMR.
Lax's problem	Limiting variables, with and without detector, and with and without AMR.
Blast waves	Limiting variables, with and without detector, and with and without AMR.
Shock-entropy waves interaction	Limiting variables, with and without detector, and with and without AMR.
2D convection	Detector and AMR in 2D.
Double-Mach reflection	Example in 2D.
Vortex convection	Smooth example in 2D.
Shock-vortex interaction	Example in 2D.
Spherical shock test	Multidimensional symmetry (3D).

Table 2. Summary of test cases.

4.1. ***Order of convergence – linear.*** The order of convergence is studied with the one-dimensional convection equation because the exact solution is known:

$$\frac{\partial u(x,t)}{\partial t} + c\frac{\partial u(x,t)}{\partial x} = 0, \tag{40}$$

$$u(x,t=0) = \sin x,$$

where u is a passive scalar and c is the constant convection velocity equal to 1. The exact solution is $u(x,t) = \sin(x - ct)$. The domain has a length of 2π and periodic boundary conditions. The number of cells N and the polynomial order p are varied in this study.

This case is run without detector to show the effect of the limiters on smooth solutions. Moreover, given that the governing equation is not a system of equations, no characteristic decomposition is needed.

The time integration schemes used here are the SDC and the TVD-RK of 3rd order with a time step of 10^{-5}. This gives an error in time of the order of 10^{-15}, which is negligible with respect to the spatial error and of the order of the round-off error. The L_∞ error, e_{L_∞}, at $t = 2$ is computed at the centroid of the element and with respect to the exact solution, i.e.,

$$e_{L_\infty} = \max_{l=1,\dots,N} |U_l(\bar{x}_l, t) - u(\bar{x}_l, t)| \tag{41}$$

where $\bar{x}_l$ is the centroid of element l. The L_∞ error in the plots is normalized by the case with the largest error. As shown in Figure 2, elements of order p lead to an order of accuracy of $p+1$, as the literature predicts [7]. Although the solutions with limiter have the same order of accuracy, they have a greater error. Therefore, the limiter should not be used unless really needed. The curves in Figure 2 get flattened out for very low errors (around $O(10^{-11})$) due to accumulated round-off error.

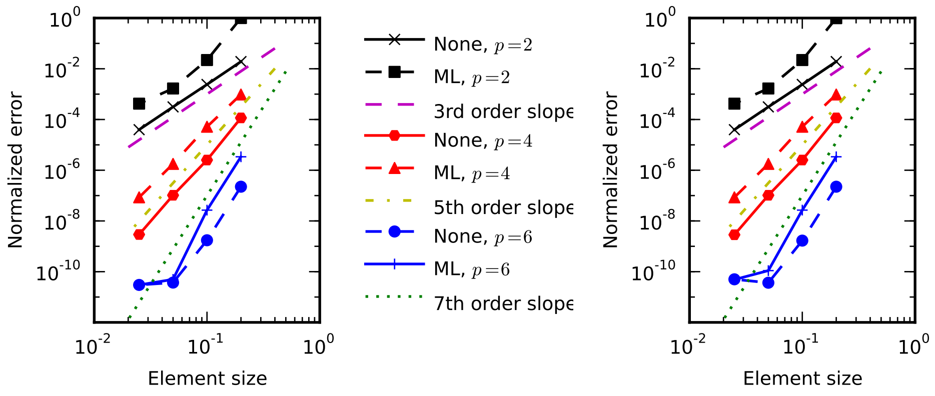


Figure 2. Grid convergence for different orders when a smooth solution is convected, for the 3rd-order TVD-RK (left) and the 3rd-order SDC (right).

4.2. ***Order of convergence – nonlinear.*** Now the order of convergence in space is studied with the one-dimensional burger equation as in [7]:

$$\frac{\partial u(x,t)}{\partial t} + \frac{\partial (u(x,t)^2/2)}{\partial x} = 0, \tag{42}$$

$$u(x, t=0) = \tfrac{1}{4} + \tfrac{1}{2}\sin(\pi(2x-1)),$$

where u is the velocity. The domain has a unit length and periodic boundary conditions. The number of cells N and the polynomial order p are varied in this study. The exact solution is estimated with $N = 2048$, $p = 2$, 3rd-order TVD-RK, and without limiter. The problem is solved until $t = 0.05$, when the solution is still smooth.

This case is run without detector to show the effect of the limiters on smooth solutions.

The time integration schemes used here are the SDC and the TVD-RK of 3rd order with a time step of 10^{-5}. This gives an error in time of the order of 10^{-15}, which is negligible with respect to the spatial error and of the order of the round-off error. The L_∞ error, e_{L_∞}, at $t = 0.05$ is computed at the centroid of the element and with respect to the estimated exact solution as in the previous case. As shown in Figure 3, the order of accuracy in space matches closely with what the theory predicted even for a nonlinear problem. Even though the solutions with limiter have the same order of accuracy, they have a greater error. Therefore, the limiter should not be used if it is not really needed. The curves in Figure 3 get flattened out for very low errors due to accumulated round-off error. In conclusion, the observations for the nonlinear case are very similar to the linear case above.

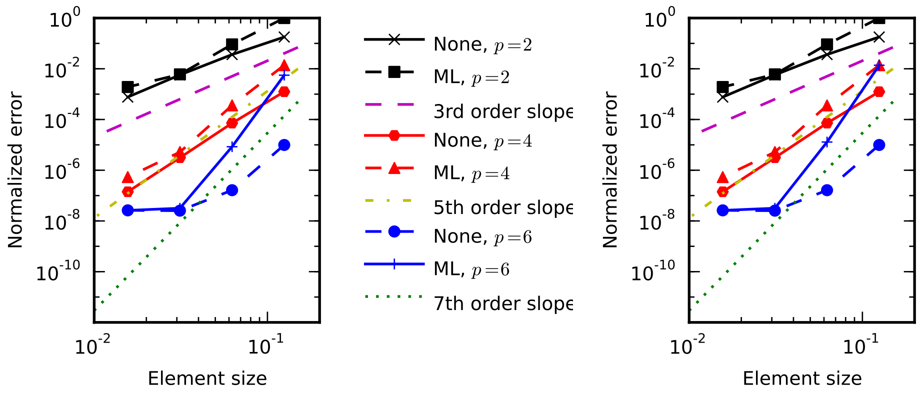


Figure 3. Grid convergence for different orders with a smooth nonlinear problem, for the 3rd-order TVD-RK (left) and the 3rd-order SDC (right).

4.3. ***Accuracy in time.*** The time integration is studied with the equation

$$\frac{\partial u(x,t)}{\partial t} + 0\frac{\partial u(x,t)}{\partial x} = u(x,t), \tag{43}$$

$$u(x,0) = 1,$$

which has the exact solution $u(x,t) = e^t$. The convection velocity is 0 so that the truncation error in space is zero and the truncation error in time can be studied by itself. A 1D domain of unit length, periodic boundaries and 100 elements is used. The time integration is performed with the TVD-RK and SDC methods for different order. The L_∞ error is computed at $t = 6.28$ for different number of time steps and shown in Table 3 along with the order of accuracy. The results are also shown in Figure 4 for a more clear appreciation. The order of accuracy for N_i elements (knowing that $N_i = 2N_{i-1}$) is computed as

$$\frac{\log(e_i/e_{i-1})}{\log(0.5)}. \tag{44}$$

The fact that the computed order approaches the order of the scheme verifies the proper implementation of the temporal integration. Also, note that at equal theoretical order, SDC results to be more accurate while they have very similar order of accuracy.

Number of	2nd order TVD-RK		3rd order TVD-RK		2nd order SDC	
time steps	e_{L_∞}	order	e_{L_∞}	order	e_{L_∞}	order
8	$1.6537 \cdot 10^2$	—	$3.5302 \cdot 10^1$	—	$1.6537 \cdot 10^2$	—
16	$6.0804 \cdot 10^1$	1.4435	$6.1493 \cdot 10^0$	2.5213	$6.0804 \cdot 10^1$	1.4435
32	$1.8276 \cdot 10^1$	1.7342	$9.0205 \cdot 10^{-1}$	2.7691	$1.8276 \cdot 10^1$	1.7342
64	$4.9757 \cdot 10^0$	1.8770	$1.2200 \cdot 10^{-1}$	2.8863	$4.9757 \cdot 10^0$	1.8770
128	$1.2948 \cdot 10^0$	1.9422	$1.5861 \cdot 10^{-2}$	2.9434	$1.2948 \cdot 10^0$	1.9422
256	$3.2999 \cdot 10^{-1}$	1.9722	$2.0219 \cdot 10^{-3}$	2.9717	$3.2999 \cdot 10^{-1}$	1.9722
Number of	3rd order SDC		4th order SDC		5th order SDC	
time steps	e_{L_∞}	order	e_{L_∞}	order	e_{L_∞}	order
8	$1.9510 \cdot 10^1$	—	$1.2840 \cdot 10^0$	—	$7.2084 \cdot 10^{-2}$	—
16	$2.8648 \cdot 10^0$	2.7677	$9.2132 \cdot 10^{-2}$	3.8007	$2.4229 \cdot 10^{-3}$	4.8949
32	$3.7984 \cdot 10^{-1}$	2.9150	$6.0812 \cdot 10^{-3}$	3.9213	$7.7303 \cdot 10^{-5}$	4.9700
64	$4.8588 \cdot 10^{-2}$	2.9667	$3.8890 \cdot 10^{-4}$	3.9669	$2.4288 \cdot 10^{-6}$	4.9922
128	$6.1332 \cdot 10^{-3}$	2.9859	$2.4556 \cdot 10^{-5}$	3.9852	$7.5987 \cdot 10^{-8}$	4.9984
256	$7.7005 \cdot 10^{-4}$	2.9936	$1.5422 \cdot 10^{-6}$	3.9931	$2.3588 \cdot 10^{-9}$	5.0096

Table 3. Error e_{L_∞} for TVD-RK (2nd and 3rd order) and for SDC (2nd to 5th order).

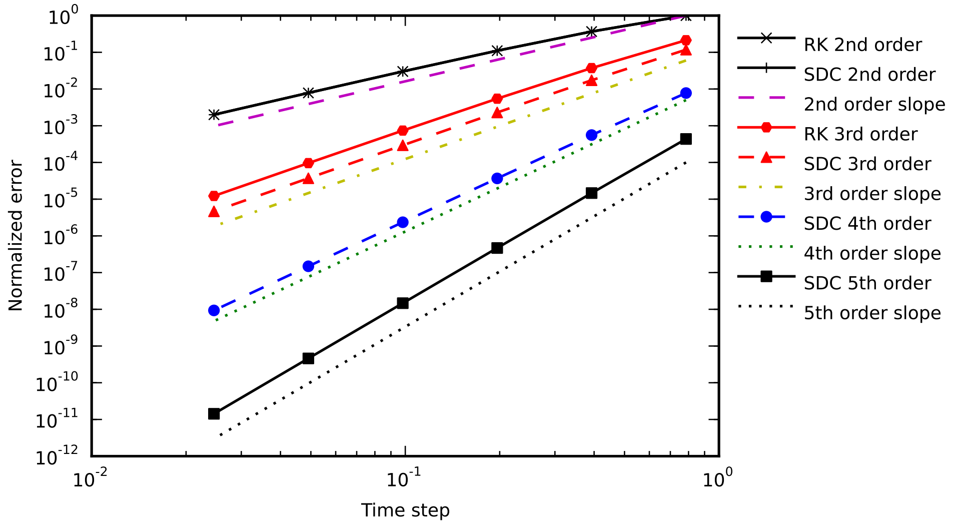


Figure 4. Normalized error for TVD-RK and SDC.

Figure 5 shows the CPU time against the e_{L_∞} obtained for different orders and schemes. The CPU time is normalized by the fastest case. The curves closer to the bottom left corner represent a more efficient scheme. For the same order, TVD-RK is more efficient than SDC. At the same time, the efficiency is increased with the order. For instance for this case 5th order SDC is more efficient than 3rd order TVD-RK.

In conclusion, the advantage of SDC with respect to RK is that the extension to higher orders is trivial. One could argue that SDC does not have the TVD property

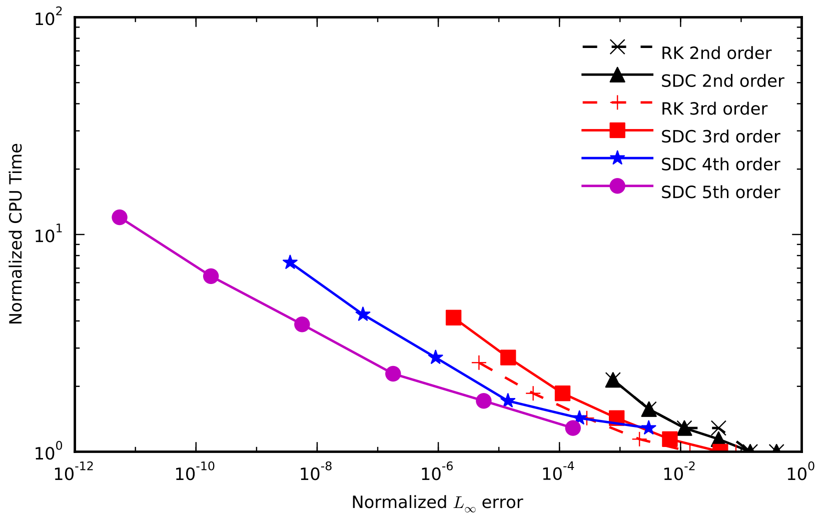


Figure 5. CPU time for different time integration schemes and orders.

of TVD-RK of 2nd and 3rd order, however, RK schemes of 4th order or greater are not TVD either. For certain problems where the error in time is important higher orders may be more suitable. Thus, explicit SDC seems to be a possible approach for high-order time integration of DG schemes.

The test cases below tend to have a dominant spatial error, thus very high orders in time are not required.

4.4. ***Advection of mixed pulses.*** The convection equation (40) is used with the initial value given by

$$u(x,0)=\begin{cases} \frac{1}{6}\left(G(x,\beta,z-\delta)+G(x,\beta,z+\delta)+4G(x,\beta,z)\right) & \text{if } -0.8\le x\le -0.6,\\ 1 & \text{if } -0.4\le x\le -0.2,\\ 1-|10(x-0.1)| & \text{if } \quad 0 \le x\le \quad 0.2,\\ \frac{1}{6}\left(F(x,\alpha,a-\delta)+F(x,\alpha,a+\delta)+4F(x,\alpha,z)\right) & \text{if } \quad 0.4\le x\le \quad 0.6,\\ 0 & \text{otherwise,}\end{cases}$$

for

$$G(x,\beta,z)=e^{-\beta(x-z)^2} \quad \text{and} \quad F(x,\alpha,a)=\sqrt{\max(1-\alpha^2(x-a)^2,0)},$$

with $a=0.5$, $z=-0.7$, $\delta=0.005$, $\alpha=10$, and $\beta=\log 2/(36\delta^2)$. The domain is a uniform grid from $x=-1$ to $x=1$ with periodic boundary conditions.

The SDC method of 3rd order is used. The ML is applied on every element or on the ones flagged by the MB-AP-TVD detector. The result at $t=8.0$ is shown in Figure 6 for $p=2,4$ and for 200 cells.

Figure 7 shows the L_1 error computed at the center of the elements for different number of cells and polynomial orders:

$$e_{L_1}=\sum_{l=1}^{N}\int_{\Omega_l}|U_l(x,t)-u(x,t)|dx \tag{45}$$

where U is the numerical result and u is the exact solution (or its estimation). In the previous test case L_∞ was used, which is an adequate parameter to analyze smooth solution, however, for discontinuous solutions L_1 is more appropriate.

A few observations can be made from this figure. The error is reduced as the number of elements n or the polynomial order p increases. Also, using the detector improves the accuracy. Note that the order of accuracy is approximately 1 because of the presence of discontinuities in the solution. Therefore, increasing the order p when discontinuities are present reduces the error, but not the order of accuracy.

The same case is run using the original AP-TVD detector, and the efficiency of the AP-TVD and MB-AP-TVD detectors are compared in Figure 8. Curves closer to the bottom left corner represent more accurate schemes.

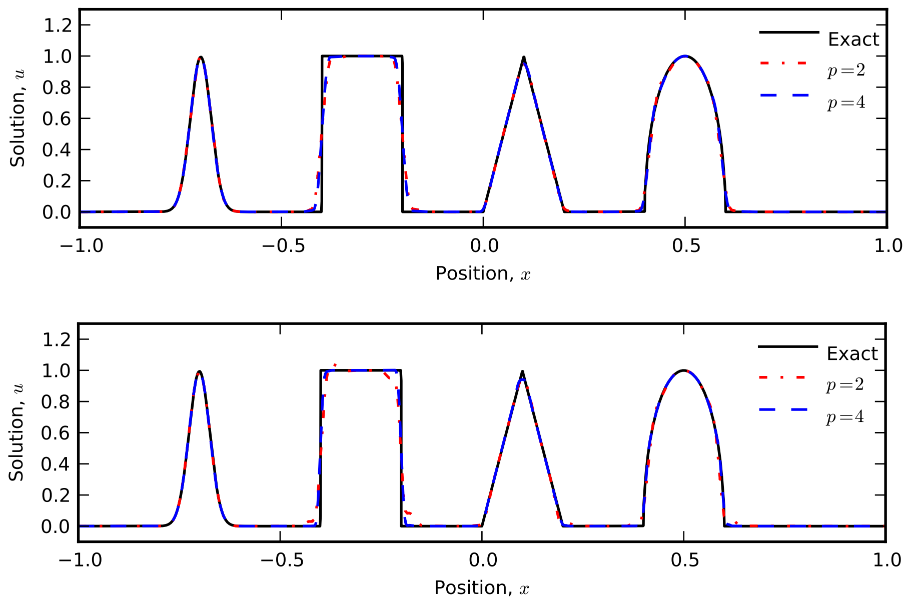


Figure 6. Convection of mixed pulses at $t = 8$, with 200 cells and $p = 2, 4$. Top: ML applied to all cells. Bottom: ML applied only to troubled cells flagged by the detector.

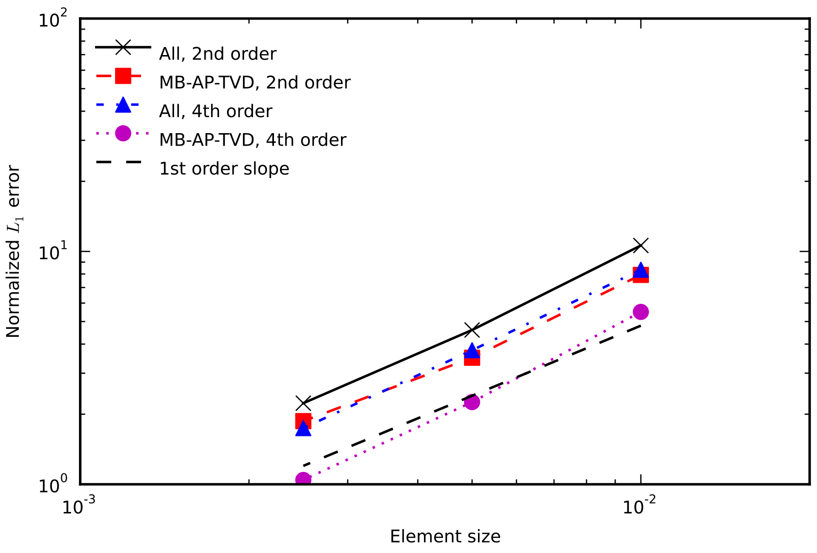


Figure 7. L_1 error for different orders and number of elements for the convection of mixed pulses.

Note that for the same number of elements AP-TVD tends to be slower, while the error is similar. Thus, MB-AP-TVD ends up being a better choice than AP-TVD when the default basis of the element is formed by Legendre polynomials. Given

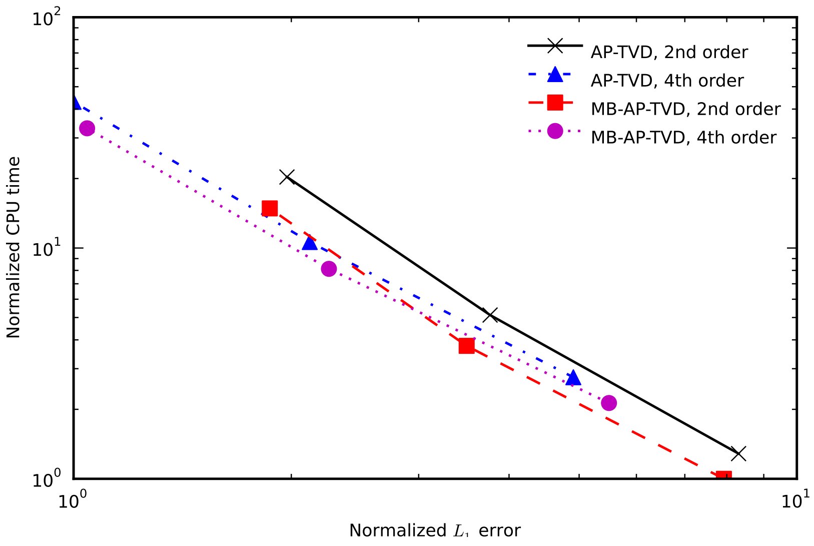


Figure 8. CPU time versus L_1 error for different orders for the convection of mixed pulses.

this, we only use the MB-AP-TVD for the cases below. In addition, Figure 8 shows that for this case, which contains discontinuities, increasing the order of the scheme does not improve its efficiency.

4.5. ***High order for a smooth and nonsmooth solution.*** The same case as in Section 4.2 is observed here for a longer period of time. At $t = 0.4$ a discontinuity is found at approximately $x = 0.1$, while the rest of the solution is smooth. The error is usually computed taking into account the whole domain. However, in order to analyze only the region with a smooth solution, it can be computed for part of the domain. For this purpose we define $\tilde{e}_{L_1}$:

$$\tilde{e}_{L_1} = \sum_{\substack{l=1 \\ 0.3 \le x \le 0.9}}^{N} \int_{\Omega_l} |U_l(x,t) - u(x,t)|dx \tag{46}$$

This is similar to (45), but the integration is done away from the discontinuity, i.e., for $0.3 \le x \le 0.9$. We estimate the exact solution with 512 elements with $p = 6$, and using the SDC of 7th order. The problem is studied using $p = 2, 4, 6$, $N = 10, 20, 30, 40, 80, 160$, SDC of order $p+1$, and limiting on all the elements or as flagged by the MB-AP-TVD detector. The errors e_{L_1} and $\tilde{e}_{L_1}$ are shown in Tables 4 and 5. Note that the order of accuracy based on $\tilde{e}_{L_1}$ is close to $p+1$, while for e_{L_1} it is close 1. At very low errors the order drops due to accumulated round-off error. For $p = 4$ and $p = 6$ using 20 elements the order is much greater then $p+1$ since the error for $N = 10$ is relatively large. This is due to the propagation

Number of time steps	Whole domain ($p=2$) e_{L_1}	order	Whole domain ($p=4$) e_{L_1}	order	Whole domain ($p=6$) e_{L_1}	order
10	$3.0333 \cdot 10^{-2}$	—	$2.6758 \cdot 10^{-2}$	—	$2.5755 \cdot 10^{-2}$	—
20	$1.2243 \cdot 10^{-2}$	1.3089	$1.1347 \cdot 10^{-2}$	1.2376	$1.1102 \cdot 10^{-2}$	1.2140
30	$7.6765 \cdot 10^{-3}$	1.1513	$7.2367 \cdot 10^{-3}$	1.1093	$7.1097 \cdot 10^{-3}$	1.0991
40	$5.6299 \cdot 10^{-3}$	1.0778	$5.3411 \cdot 10^{-3}$	1.0558	$5.2764 \cdot 10^{-3}$	1.0366
80	$2.7534 \cdot 10^{-3}$	1.0319	$2.6630 \cdot 10^{-3}$	1.0041	$2.0933 \cdot 10^{-3}$	1.3338
160	$1.3740 \cdot 10^{-3}$	1.0029	$9.1757 \cdot 10^{-4}$	1.5372	$1.1018 \cdot 10^{-3}$	0.9260
Number of time steps	Smooth region ($p=2$) $\tilde{e}_{L_1}$	order	Smooth region ($p=4$) $\tilde{e}_{L_1}$	order	Smooth region ($p=6$) $\tilde{e}_{L_1}$	order
10	$4.4679 \cdot 10^{-4}$	—	$5.0581 \cdot 10^{-5}$	—	$3.8820 \cdot 10^{-5}$	—
20	$2.8444 \cdot 10^{-5}$	3.9734	$1.3111 \cdot 10^{-8}$	11.914	$2.1739 \cdot 10^{-9}$	14.1242
30	$6.2605 \cdot 10^{-6}$	3.7332	$1.2282 \cdot 10^{-9}$	5.8399	$2.3454 \cdot 10^{-12}$	16.8493
40	$2.2165 \cdot 10^{-6}$	3.6093	$2.3009 \cdot 10^{-10}$	5.8220	$1.8132 \cdot 10^{-13}$	8.8985
80	$1.9190 \cdot 10^{-7}$	3.5298	$5.1326 \cdot 10^{-12}$	5.4864	$9.6648 \cdot 10^{-14}$	0.9077
160	$1.7365 \cdot 10^{-8}$	3.4661	$1.7473 \cdot 10^{-13}$	4.8765	$1.0902 \cdot 10^{-13}$	−0.1738

Table 4. The error e_{L_1} (top half) and $\tilde{e}_{L_1}$ (bottom half) with limiting on all the elements.

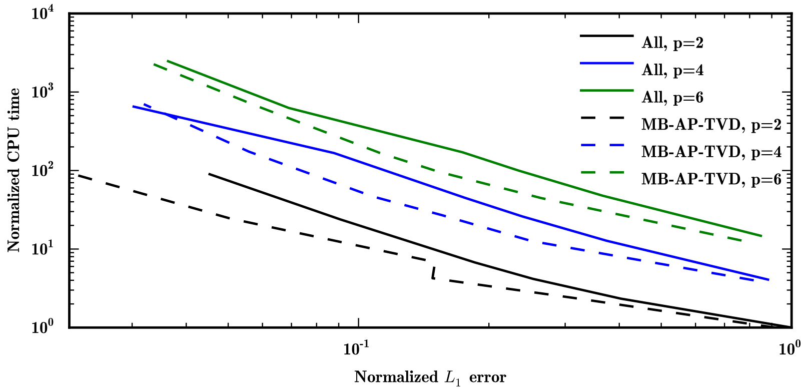
Number of time steps	Whole domain ($p=2$) e_{L_1}	order	Whole domain ($p=4$) e_{L_1}	order	Whole domain ($p=6$) e_{L_1}	order
10	$2.7784 \cdot 10^{-2}$	—	$2.4803 \cdot 10^{-2}$	—	$2.3243 \cdot 10^{-2}$	—
20	$9.7931 \cdot 10^{-3}$	1.5044	$7.6011 \cdot 10^{-3}$	1.7063	$8.0117 \cdot 10^{-3}$	1.5366
30	$4.4951 \cdot 10^{-3}$	1.9205	$4.7370 \cdot 10^{-3}$	1.1663	$4.6348 \cdot 10^{-3}$	1.3498
40	$4.5546 \cdot 10^{-3}$	0.0457	$3.2698 \cdot 10^{-3}$	1.2885	$3.3924 \cdot 10^{-3}$	1.0848
80	$1.5292 \cdot 10^{-3}$	1.5745	$1.6939 \cdot 10^{-3}$	0.9489	$1.7780 \cdot 10^{-3}$	0.9321
160	$6.5156 \cdot 10^{-4}$	1.2308	$9.6828 \cdot 10^{-4}$	0.8068	$9.5344 \cdot 10^{-4}$	0.8990
Number of time steps	Smooth region ($p=2$) $\tilde{e}_{L_1}$	order	Smooth region ($p=4$) $\tilde{e}_{L_1}$	order	Smooth region ($p=6$) $\tilde{e}_{L_1}$	order
10	$1.5107 \cdot 10^{-4}$	—	$4.2981 \cdot 10^{-5}$	—	$1.5876 \cdot 10^{-5}$	—
20	$9.2714 \cdot 10^{-6}$	4.0263	$6.3106 \cdot 10^{-9}$	12.7336	$9.5822 \cdot 10^{-12}$	20.6600
30	$2.2911 \cdot 10^{-6}$	3.4477	$6.1355 \cdot 10^{-10}$	5.7483	$4.4873 \cdot 10^{-13}$	7.5500
40	$8.4867 \cdot 10^{-7}$	3.4521	$1.2682 \cdot 10^{-10}$	5.4799	$1.2121 \cdot 10^{-13}$	4.5497
80	$8.4769 \cdot 10^{-8}$	3.3236	$3.3972 \cdot 10^{-12}$	5.2223	$9.6667 \cdot 10^{-14}$	0.3265
160	$9.0416 \cdot 10^{-9}$	3.2289	$1.4398 \cdot 10^{-13}$	4.5604	$1.0951 \cdot 10^{-13}$	−0.1799

Table 5. The error e_{L_1} (top half) and $\tilde{e}_{L_1}$ (bottom half) with limiting based on the MB-AP-TVD detector.

to smooth regions of instabilities generated at the discontinuity. For a large enough number of elements, $N \geq 20$, the instabilities do not affect the smooth area being considered in (46) for $\tilde{e}_{L_1}$.

Figure 9 shows the efficiency of the scheme for different orders, with and without the MB-AP-TVD detector. The error and CPU time are normalized based on the case with the largest error.

As observed for previous cases, the troubled-cell detector helps improve the accuracy and efficiency of the solver. For lower L_1 error high-order schemes become more efficient. The limiter reduces numerical oscillations at discontinuities, but with a minimal numerical diffusion, so small instabilities still exist. As the number of elements is increased the numerical error originated at the discontinuity is localized in a smaller region. Thus, probably, p-adaptivity could improve the efficiency by dropping the order at the discontinuity and keeping high order in the smooth region.



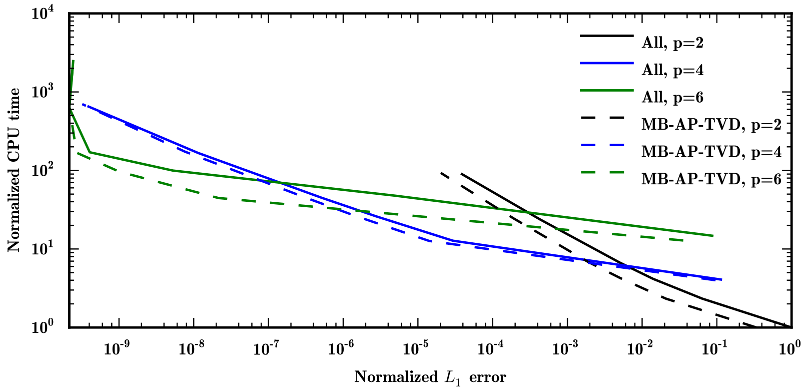


Figure 9. Efficiency of the scheme for different orders for a solution with one discontinuity. The limiter is applied to all the elements or based on the MB-AP-TVD detector. Top: e_{L_1} for the whole domain. Bottom: $\tilde{e}_{L_1}$ for the smooth region.

4.6. *Sod's problem.* The initial conditions are

$$(\rho, v, p) = \begin{cases} (1.0, 0.0, 1.0) & \text{if } x \leq 0.5, \\ (0.125, 0.0, 0.1) & \text{if } x > 0.5. \end{cases} \tag{47}$$

A 1D domain is used and it extends from $x = 0$ to $x = 1$. The case is run with different number of elements and the limiting is based on conservative, primitive, or characteristic variables. In addition, two options are tested, one applies the ML with the MB-AP-TVD detector, and the second option applies the ML to all cells. The grid is uniform with $p = 2$, and the time integration is performed using the 3rd-order SDC. The simulation is run until $t = 0.2$. The normalized CPU time versus the L_1 error of the final density is shown for the three cases in Figure 10(a).

Limiting with primitive or characteristic variables requires computing the respective Jacobians for each element, so it is computationally slightly more expensive than using conservative variables, but the error is lower. For primitive and characteristic variables, using the MB-AP-TVD detector to apply the ML to only troubled cells increases the efficiency and lowers the error since the solution is smooth in a large portion of the domain.

The same case is run enabling the adaptive mesh refinement for $\ell_{\max} = 1, 2, 3$. The CPU time versus the L_1 error of the density is shown in Figure 10(b) for limiting with primitive variables. Note that both variables are normalized by the fastest simulation. As $\ell_{\max}$ is increased the curves get slightly closer to the origin. This means that for this test case enabling the adaptivity produces some increase in the efficiency of the solver. Here, the MB-AP-TVD also shows to improve the efficiency.

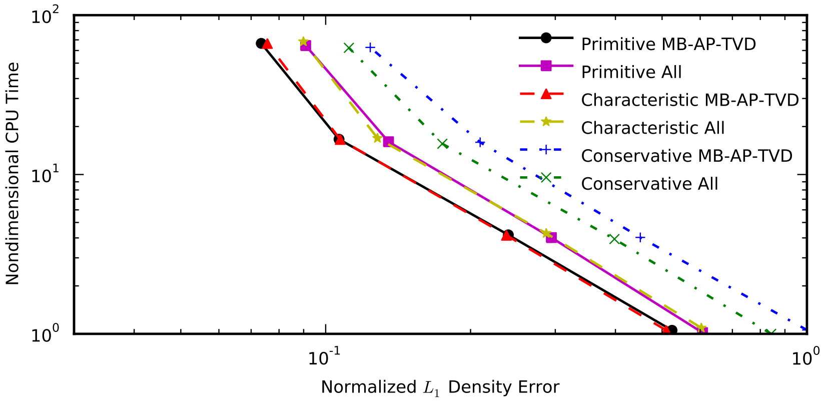
The estimators are compared using $\ell_{\max} = 3$, the MB-AP-TVD detector, and limiting with primitive variables. The efficiency is represented in Figure 10(c). Clearly, JUMP2 is more efficient than KXRCF for this case.

4.7. *Lax's problem.* The initial solution is:

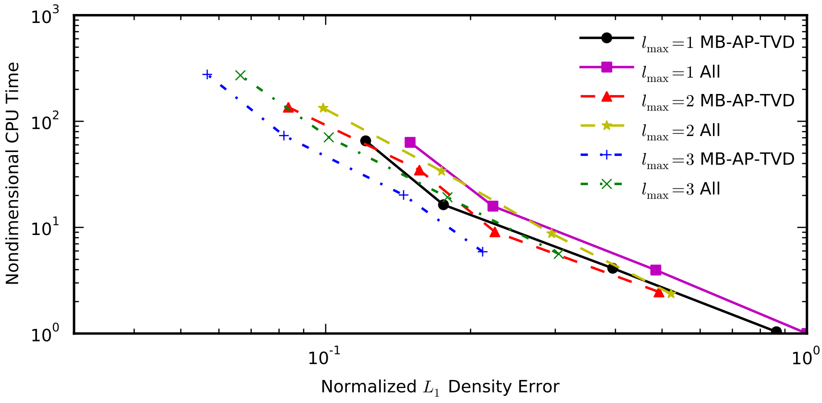
$$(\rho, v, p) = \begin{cases} (0.445, 0.698, 3.528) & \text{if } x \leq 0, \\ (0.5, 0, 0.571) & \text{if } x > 0. \end{cases} \tag{48}$$

The problem is solved in the 1D domain $[-0.5, 0.5]$ until $t = 0.13$.

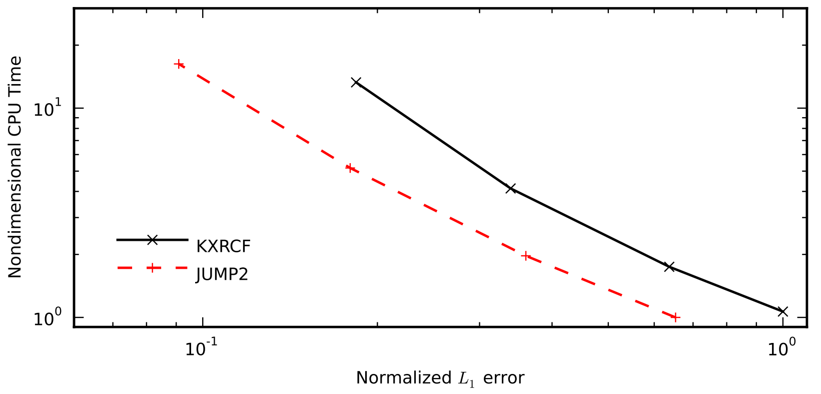
Initially, the effect of the variables used for limiting is studied. A uniform grid is used with $N = 64, 128, 256, 512$ and $p = 2$ with the limiter applied to either all cells or those flagged by the MB-AP-TVD detector. The integration in time is done with the 3rd-order SDC method. The CPU time versus the L_1 error of the density is shown in Figure 11(a). These results show that for this particular test problem the MB-AP-TVD detector increases the efficiency for conservative and characteristic variables, while for primitive variables it did not affect significantly. The CPU time is very similar independent of the set of variables used. Even though



(a) Comparison for different limiting variables with and without detector.

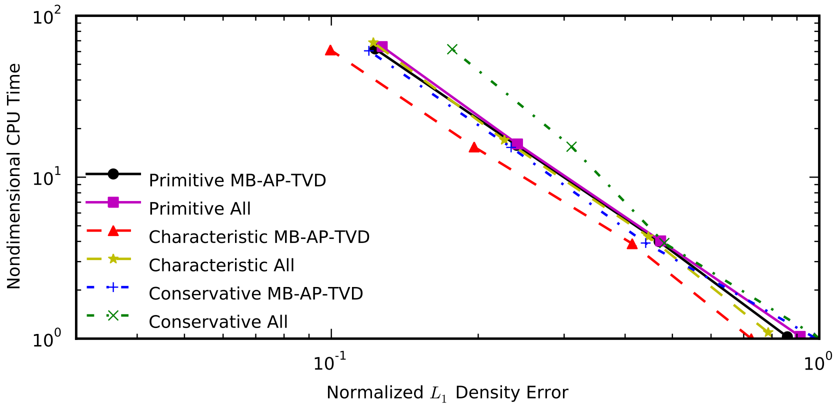


(b) AMR comparison; limiting using primitive variables.

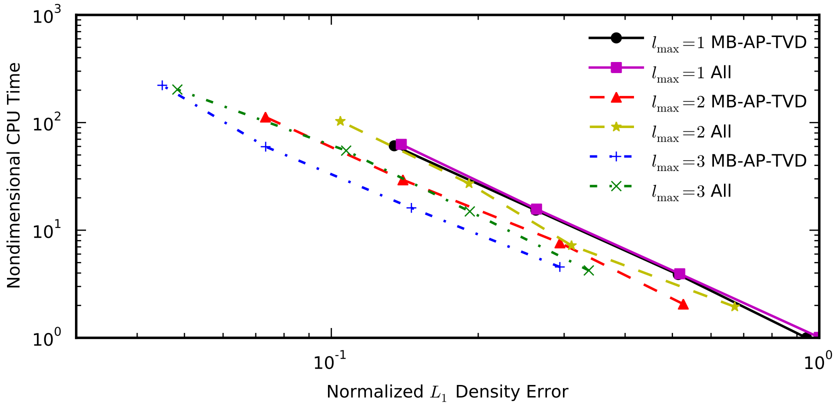


(c) Estimator comparison, limiting using primitive variables; the MB-AP-TVD detector and $\ell_{\max}=3$.

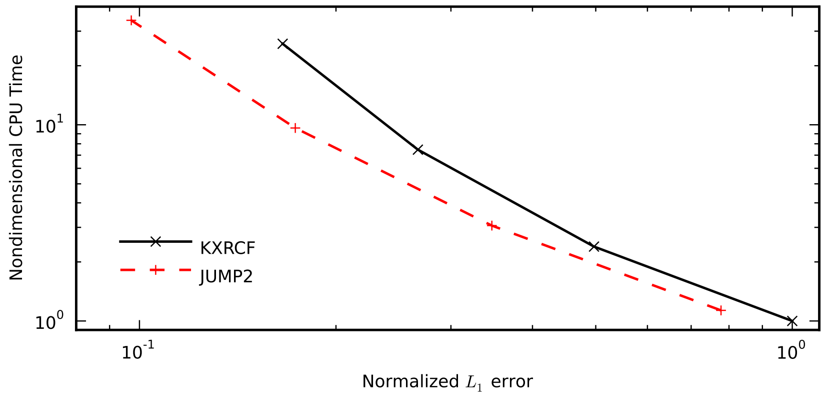
Figure 10. Sod's problem for different solver options. The curves closer to the bottom left corner represent a more efficient set of options.



(a) Limiting comparison.



(b) AMR comparison; limiting using primitive variables.



(c) Estimator comparison; limiting using primitive variables, the MB-AP-TVD detector and $\ell_{\max} = 3$.

Figure 11. Lax problem for different solver options.

conservative variables do not require to compute the Jacobian to transform between the variables, they may require more steps of the limiter.

Now the effect of the adaptation is studied, limiting with primitive variables. The same grid is used, but the adaptation is enable with $\ell_{\max} = 1, 2, 3$. The result is shown in Figure 11(b). When $\ell_{\max}$ is raised, the efficiency of the solver increases and it improves more using the MB-AP-TVD detector. The estimators are compared using $\ell_{\max} = 3$, the MB-AP-TVD detector, and limiting with primitive variables. The efficiency is represented in Figure 11(c). Clearly, JUMP2 is more efficient than KXRCF for this case.

4.8. ***Blast waves.*** Consider the initial data $\rho = 1.0$, $v = 0.0$, and

$$P = \begin{cases} 1000 & \text{if } 0 \le x < 0.1, \\ 0.01 & \text{if } 0.1 \le x < 0.9, \\ 100 & \text{if } 0.9 \le x \le 1.0. \end{cases} \tag{49}$$

This problem is a common test case first presented in [37]. Walls are located at $x = 0$ and $x = 1$.

The problem is run until $t = 0.038$ s for $p = 2$, $\ell_{\max} = 1, 2, 3$, different number of root cells and the 3rd-order SDC. This test case does not have an exact solution, so it is approximated using a uniform mesh with $N = 4096$, $p = 2$, $\ell_{\max} = 1$, the ML without detector and with characteristic decomposition, similar to [21].

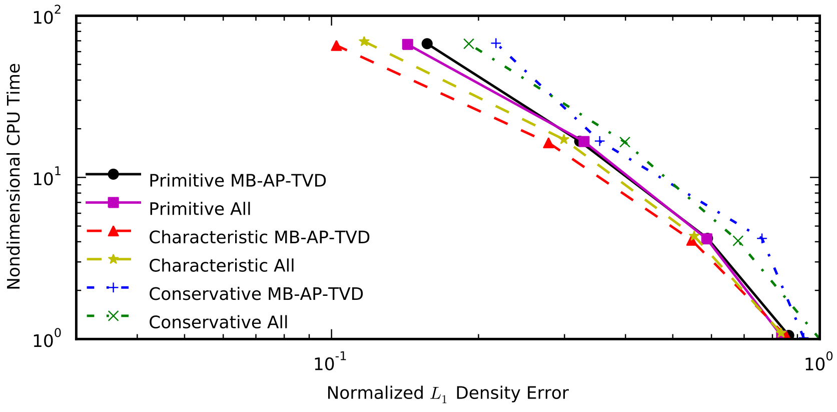
Figure 12 shows the CPU time versus the L_1 error in density, both variables are normalized by the fastest run. Part (a) shows the effect of the detector and the limiting variables. Part (b) represents the efficiency of the AMR approach using primitive variables. The efficiency of the solver clearly improves increasing $\ell_{\max}$. In this case the MB-AP-TVD detector does not produce any significant difference when studying the refinement aspects. The estimators are compared using $\ell_{\max} = 3$, the MB-AP-TVD detector, and limiting with primitive variables and the results are in Figure 12(c). JUMP2 tends to be more efficient than KXRCF for this case.

4.9. ***Shock-entropy wave interaction.*** Consider the Euler equation with the following initial values:

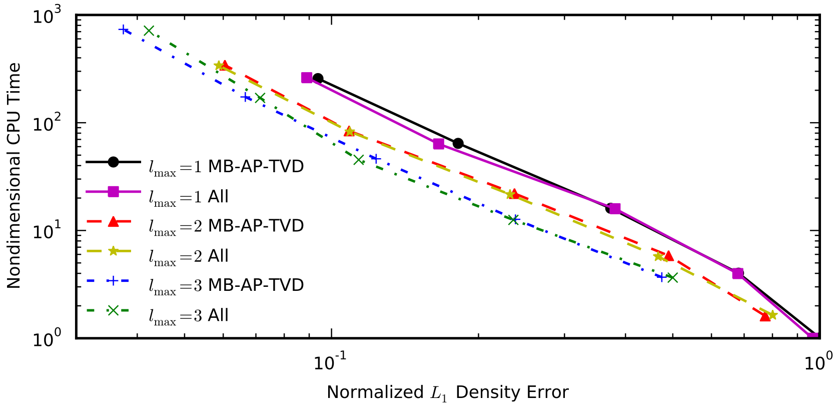
$$(\rho, v, p) = \begin{cases} (3.857143, 2.629369, 10.333333) & \text{if } x < -4, \\ (1.0 + 0.2\sin(5x), 0.0, 1.0) & \text{if } x \ge -4. \end{cases} \tag{50}$$

The problem is solved in the 1D domain $[-5, 5]$ until $t = 1.8$.

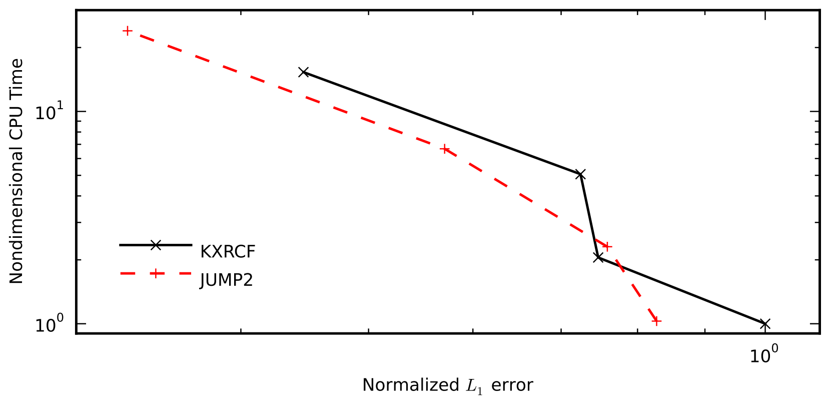
As before, the effect of the variables used for limiting are studied on a uniform grid with $N = 64, 128, 256, 512$ and $p = 2$. The integration in time is done with the 3rd-order SDC method. The CPU time versus the L_1 error of the density is shown in Figure 13. For this problem limiting using primitive variable is advantageous



(a) Limiting comparison.

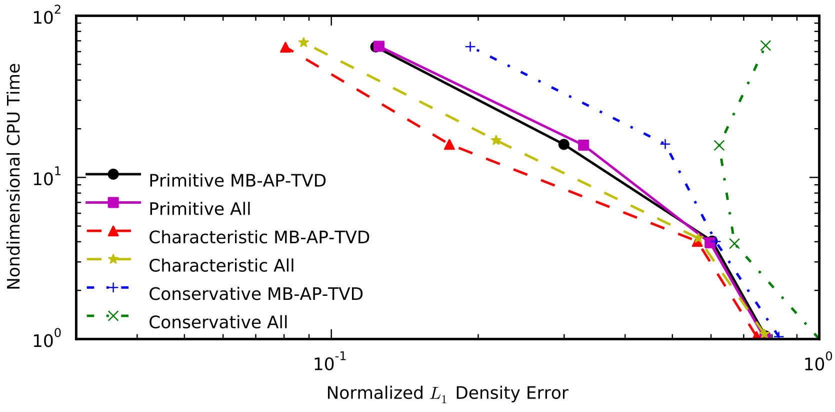


(b) AMR comparison; limiting using primitive variables.

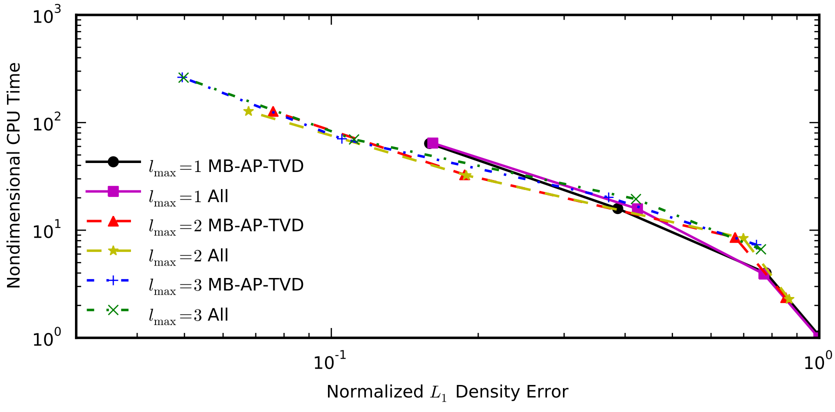


(c) Estimator comparison; limiting using primitive variables, the MB-AP-TVD detector and $\ell_{\max} = 3$.

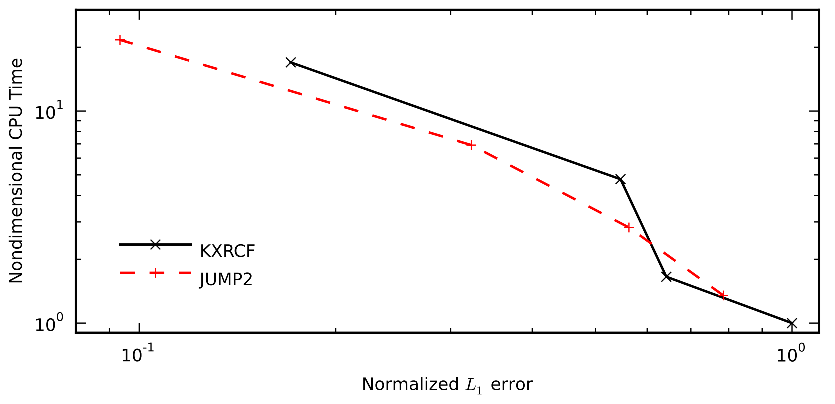
Figure 12. Interacting blast waves for different solver options.



(a) Limiting comparison.



(b) AMR comparison; limiting using the primitive variables.



(c) Estimator comparison; limiting using primitive variables, the MB-AP-TVD detector and $\ell_{\max} = 3$.

Figure 13. Shock-entropy wave interaction problem for different solver options.

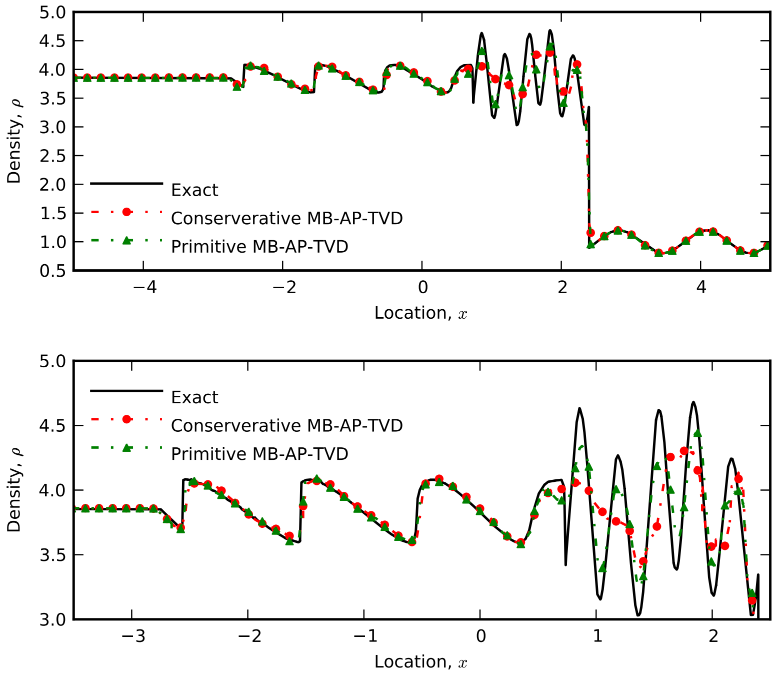


Figure 14. Shock-entropy wave interaction at $t = 1.8$ for $N = 256$, $p = 2$: density ρ as a function of location. The bottom pane shows a detail, to the left of the drop.

compared with conservative variables. Figure 14 shows the solution at $t = 1.8$; it clearly presents that limiting using primitive variables captures the smooth oscillations much more accurately than with conservative variables. Characteristic limiting provides a even more efficient solution than with primitive variables. Also, the MB-AP-TVD detector improves the efficiency. Using AMR for this test case gives no efficiency gains in the low element count (larger normalized error) regime, but AMR is more justified at lower errors where the number of elements increases.

The estimators are compared in Figure 13 using $\ell_{\max} = 3$, the MB-AP-TVD detector, and limiting with primitive variables. JUMP2 tends to be more efficient than KXRCF for this case.

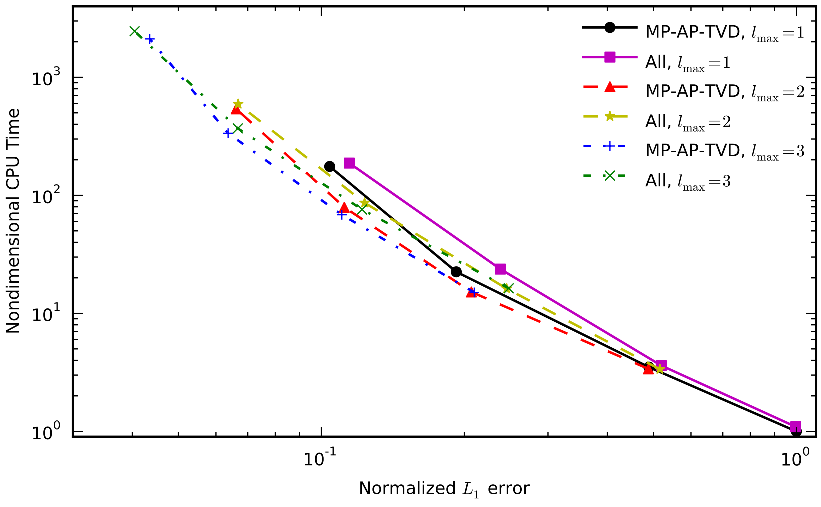
4.10. ***Convection in 2D.*** The limiter and adaptivity approach is studied in 2D using the two-dimensional convection equation

$$\frac{\partial u(\boldsymbol{x},t)}{\partial t} + c_1 \frac{\partial u(\boldsymbol{x},t)}{\partial x} + c_2 \frac{\partial u(\boldsymbol{x},t)}{\partial y} = 0 \tag{51}$$

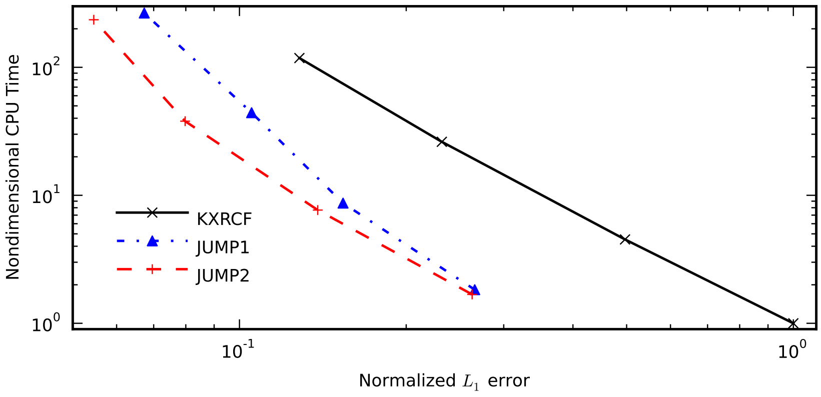
with initial condition

$$u(\boldsymbol{x}, t=0) = \begin{cases} 1 & \text{for } (x_1 - 0.5)^2 + (x_2 - 0.5)^2 \le 0.25^2, \\ 0 & \text{otherwise,} \end{cases}$$

where u is a passive scalar, and c_1 and c_2 are the constant convection velocities equal to 1. The domain is the unit square $[0, 1] \times [0, 1]$ with periodic boundary conditions. The time integration used is the 3rd-order SDC. Figure 15 shows the CPU time versus the L_1 error at $t = 1$ for different solver options varying the number of element. In part (a), $\ell_{\max}$ is varied together with the detector. Increasing the $\ell_{\max}$ improves the efficiency, and using the MB-AP-TVD helps too. In part (b), the error estimator for AMR is varied. For this 2D case JUMP1 and JUMP2 produce slightly different results, and JUMP2 is the most efficient of the three estimators.



(a) AMR comparison, with and without detector.



(b) Estimator comparison, using the MB-AP-TVD detector and $\ell_{\max} = 3$.

Figure 15. Two-dimensional convection for different solver options.

4.11. ***Double Mach reflection.*** This is a very common test case for the Euler equation first used by Woodward and Colella [37]. It was also solved by Krivodonova [22] using the ML with a uniform grid and without trouble-cell detector. This case consists of a strong shock impacting a wedge with a half-angle of 30°, thus it is usually simulated by a rectangular domain with a frame rotated 30° over the original horizontal axis.

The rectangular domain has a size of $[0, 4] \times [0, 1]$. A right-moving Mach 10 shock is initially located forming an angle of 60° with the x-axis passing by the coordinate $x = \frac{1}{6}$, $y = 0$. The undisturbed air on the right of the shock has a density of 1.4 and a pressure of 1. The specific heat ratio is $\gamma = 1.4$. A slip-wall boundary is located at the lower boundary from $x = \frac{1}{6}$ to $x = 4$. The right boundary is a supersonic outflow. The left boundary and bottom boundary for $x < \frac{1}{6}$ are supersonic inflow. The reason for applying supersonic inflow at the bottom boundary is to mimic the effect of the wedge. The top boundary mimics the exact motion of the moving shock.

The grid has 48×12 cells with $\ell_{\max} = 5$. The ML is used with the MB-AP-TVD detector. Second-order polynomial elements are used with the 3rd-order SDC method.

The results are shown for $t = 0.2$. Figure 16 shows 60 equally spaced density contours; the inset shows in black the cells flagged by the MB-AP-TVD detector as troubled cells. Note that the ML is not applied here where the flow is uniform, as intended. Figure 17 shows the level of refinement l. It can be noted that the

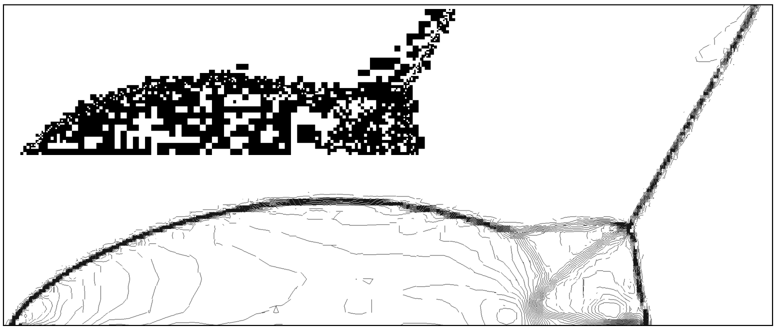


Figure 16. Double Mach reflection: density map at $t = 0.2$ and (inset) troubled cells.

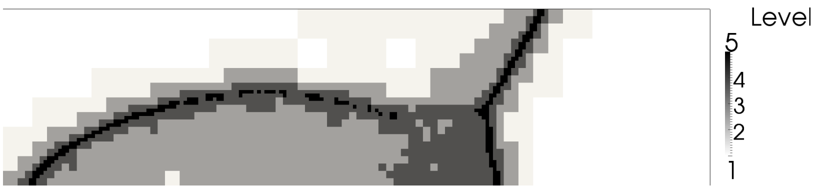


Figure 17. Double Mach reflection: refinement level.

level is increased where the features of the flow are smaller, as can be expected from Figure 16.

4.12. *Vortex convection.* As we have seen with previous tests, the global efficiency does not improve significantly for problems dominated by discontinuities when the order is increased. We studied simple smooth cases in 1D, but here we extend the study to a slightly more applicable case in 2D. An isentropic vortex is centered at the center of the domain $(x_c, y_c) = (0.5, 0.5)$. The flow is described by

$$v_1 = M\sqrt{\gamma} + \epsilon\ \tau\ e^{\alpha(1-\tau^2)}\sin\theta, \qquad v_2 = -\epsilon\ \tau\ e^{\alpha(1-\tau^2)}\cos\theta,$$

$$\rho = \left(1 - \frac{\gamma-1}{4\alpha\gamma}\epsilon^2 e^{2\alpha(1-\tau^2)}\right)^{\frac{1}{\gamma-1}}, \qquad p = \rho^\gamma,$$

where $M = 0.3$ and

$$\tau = \frac{r}{r_c}, \qquad r = \sqrt{(x-x_c)^2 + (y-y_c)^2}, \qquad \theta = \arctan\frac{y-y_c}{x-x_c}.$$

Three parameters describe the vortex: its the strength ϵ, its the decay rate α, and the critical radius r_c. For this test the following values are used: $\epsilon = 0.3$, $\alpha = 0.204$, and $r_c = 0.05$. The domain is a unit square with periodic boundaries in every direction. Different number of elements and spatial orders, p, are used. Even though this is a smooth problem, the ML limiter with the MB-AP-TVD detector are used. The range of length scales is very narrow, so AMR is not needed.

Figure 18(a) presents the CPU time versus the L_1 error after one period using the SDC of order $p+1$. The same pattern as for previous 1D cases is observed here. The efficiency increases with the order at in the high accuracy range since at equal CPU time the numerical error is smaller. However, in the low accuracy

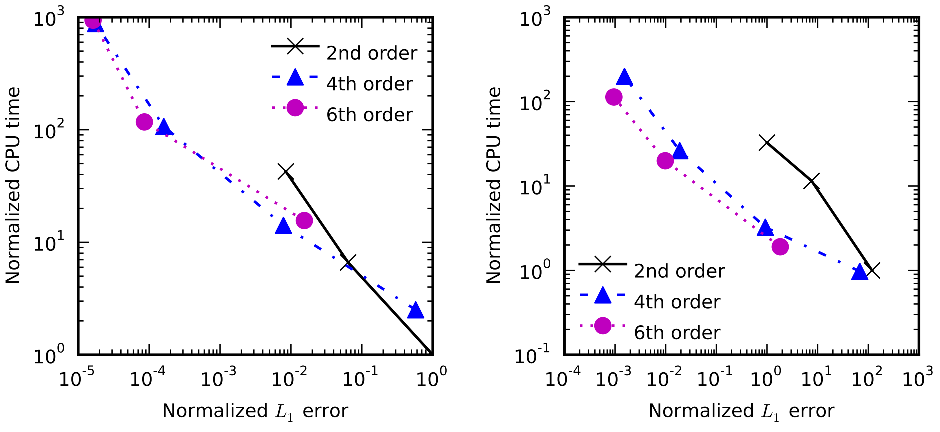


Figure 18. Efficiency for the convection of an isentropic vortex: SDC of order $p+1$ (left) and of order 3 (right).

range low order schemes perform more efficiently. Even though this problem is dominated by convection, the time-step size is limited by the acoustic time, so one could assume that we are over-resolving in time. Thus, we rerun the same cases with the 3rd-order SDC for every p. Note that in this case the CFL has to be adjusted for $p > 2$. We use $C = 0.5$, $C = 0.45$, and $C = 0.4$ for $p = 2$, $p = 4$, and $p = 6$, respectively. Now higher orders in space have a greater advantage. In cases where the error in time is more significant, increasing the order of the scheme in time would make improvements. In this case, however, higher orders in time only add more computational cost.

It can be concluded that the limiting procedure can be freely applied in the whole domain even where smooth features are present. This aspect is important for large-scale applied problems where several types of features can be present at the same time, so a generic and robust scheme is wanted. The shock-vortex interaction case shown below elaborates more on this.

4.13. ***Shock-vortex interaction.*** This problem consists of a vortex going through a shock and helps to test how the solver behaves when smooth features interact with discontinuities. For more information on this kind of problems see [32]. The initial conditions are the same as in [42; 18]. The size of the domain is $[0, 2] \times [0, 1]$. Reflective boundary conditions are used on top and bottom. The left boundary is a supersonic inflow, while the right boundary is an outflow. A stationary shock is located at $x = 0.5$, its preshock Mach number is $M_s = 1.1$, and the left side state is defined by $\rho = 1$, $u = M_s\sqrt{\gamma}$, $v = 0$ and $p = 1$. The right state can be determined from the left state by using the stationary shock relations. An isentropic vortex is centered at $(x_c, y_c) = (0.25, 0.5)$. Therefore, on the left-hand side of the shock the flow is described by

$$v_1 = M_s\sqrt{\gamma} + \epsilon\ \tau\ e^{\alpha(1-\tau^2)} \sin\theta, \qquad v_2 = -\epsilon\ \tau\ e^{\alpha(1-\tau^2)} \cos\theta,$$

$$\rho = \left(1 - \frac{(\gamma-1)\ \epsilon^2\ e^{2\alpha(1-\tau^2)}}{4\alpha\gamma}\right)^{\frac{1}{\gamma-1}}, \quad p = \rho^\gamma$$

where

$$\tau = \frac{r}{r_c} \quad, r = \sqrt{(x-x_c)^2 + (y-y_c)^2}, \quad \theta = \arctan\frac{y-y_c}{x-x_c}.$$

For this test the values used are $\epsilon = 0.3$, $\alpha = 0.204$, and $r_c = 0.05$.

A uniform grid with 32×16 cells and $\ell_{\max} = 4$ is used with $p = 2$. The time integration is done with the 3rd-order SDC method. The ML is used with the MB-AP-TVD detector. The pressure at $t = 0.8$ are shown in Figure 19 with 60 equally spaced contours. The two parts of Figure 20 indicate how the solver adapt to the solution to avoid instabilities and waste computational resources. The vortex successfully goes through the shock and features with different length scales are properly be resolved. Similar results were observed in [42; 18] using other numerical schemes.

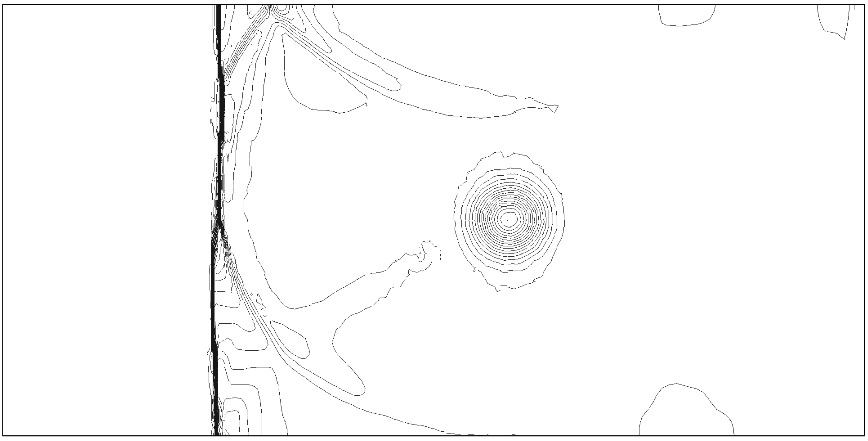


Figure 19. Pressure isocontours for a shock-vortex interaction.

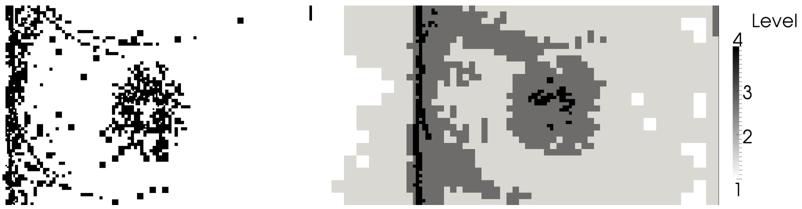


Figure 20. Shock-vortex interaction: troubled cells (left) and refinement level (right)

4.14. ***Spherical shock test.*** The final test is a spherical shock case in a cube defined in $[0, 1] \times [0, 1] \times [0, 1]$. The initial conditions are similar to the typical Sod's problem, but in this case spherical symmetry is used:

$$(\rho, v_1, v_2, v_3, p) = \begin{cases} (1.0, 0.0, 0.0, 0.0, 1.0) & \text{if } r \le 0.5, \\ (0.125, 0.0, 0.0, 0.0, 0.1) & \text{if } r > 0.5, \end{cases} \tag{52}$$

where r is the distance from $(0, 0, 0)$. The initial grid has 32^3 $p = 2$ elements, each allowed to refine to a level $\ell_{\max} = 3$. The integration in time is done with the 3rd-order SDC method.

An "exact" solution is estimated solving the Euler equation in spherical coordinates assuming spherical symmetry. Thus, the equation being solved in the domain $[0, 1]$ is

$$\frac{\partial \boldsymbol{u}}{\partial t} + \frac{\partial \boldsymbol{F}(\boldsymbol{u})}{\partial x} = \boldsymbol{S}, \tag{53}$$

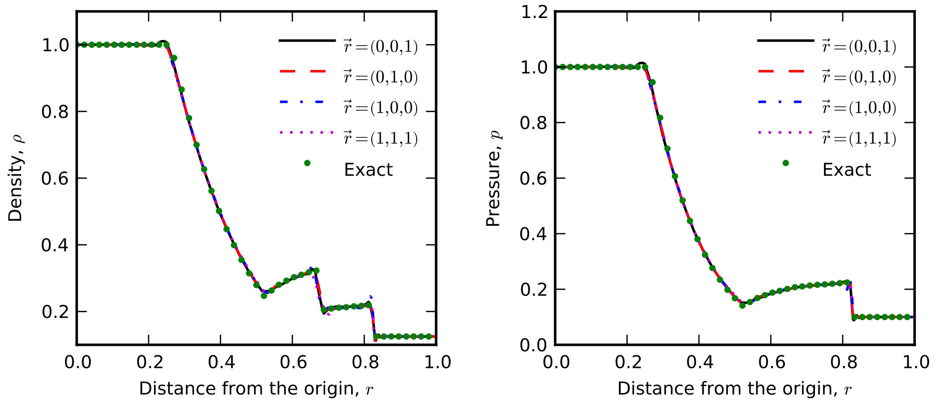


Figure 21. Spherical shock test at $t = 0.15$ over four different vectors.

where $\boldsymbol{S} = -2/x \left(\rho v_1, \rho v_1^2, (\rho E_T + p)v_1\right)^T$. This 1D problem is solved on a grid with 1024 cells with $p = 2$ and integrated in time with the 3rd-order SDC method.

A very similar test case to this one was studied in 2D in [39; 36].

The density and pressure at $t = 0.2$ over four different vectors are shown in Figure 21. Each of these four vectors are: (1, 0, 0), (0, 1, 0), (0, 0, 1), and (1, 1, 1). Given that the density on the different trajectories match, the scheme successfully respects the spherical symmetry of the problem. Note that the results shown do not match exactly the classical one-dimensional Sod shock-tube problem due to 3D effects. Figure 22 on the next page demonstrates the ability of AMR to track the shock and rarefaction waves as required.

5. Conclusions

The Euler equations are solved using the discontinuous Galerkin method with adaptive mesh refinement and high-order of accuracy in space and time.

It was shown using high-order schemes that problems with discontinuities can present high order of convergence in the smooth regions, while the global order of accuracy is close to 1 in the L_1 and L_∞ norm. Most of the cases studied include discontinuities. Therefore, in such cases the order in space and time of the scheme used is 3, since, as it was shown, increasing the order of the solver does not improve its efficiency significantly when discontinuities dominate. Given that the time step is limited by the acoustic time, convection-dominated problems end up being over-resolved in time, so in such cases increasing the order in time produces an unnecessary computational cost.

A simple and effective error estimator for adaptivity based on the interelement jump is suggested and it was shown to be more efficient than other estimator found in the literature. From a computational-resources point of view, the most

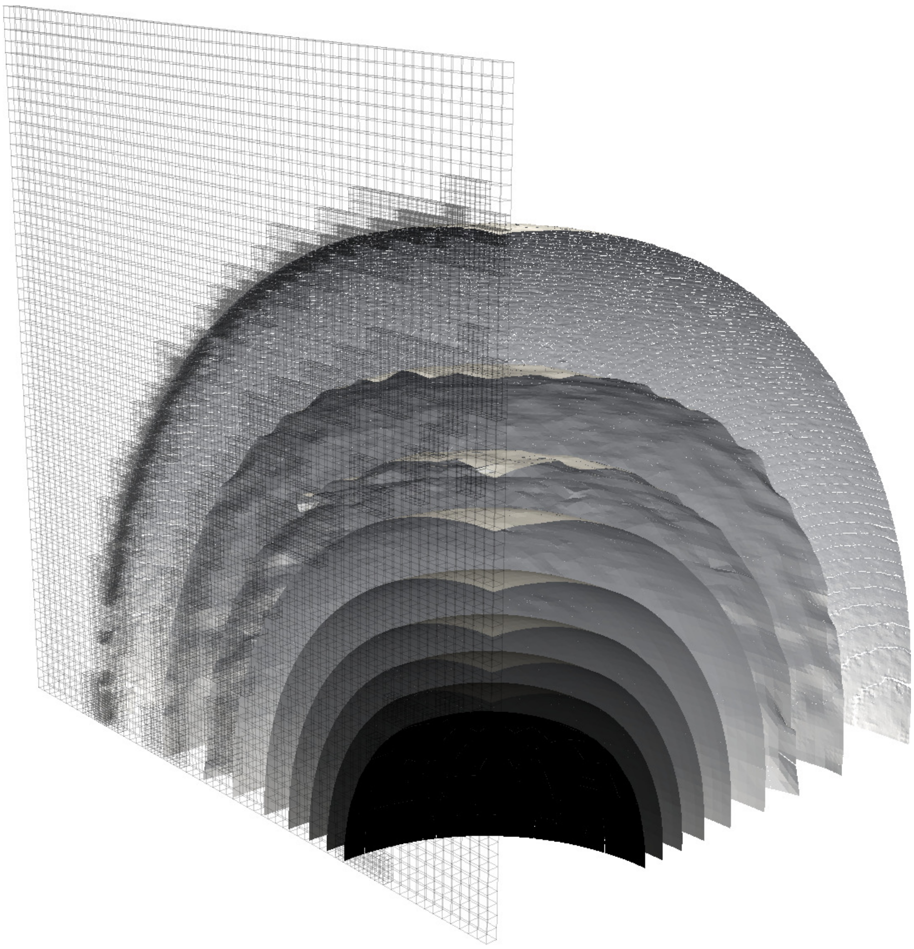


Figure 22. Spherical shock test: density contours and grid refinement.

efficient combination of maximum levels of refinement and initial number of cells is problem-dependent. In a few of the tested problems the overhead caused by the adaptation made it unnecessary. However, in no case with a wide range of scales AMR caused a significant loss of efficiency.

The AP-TVD detector in [42] was modified replacing the averaged-derivative basis that it originally required by the Legendre polynomial basis, which is commonly used in DG. Therefore, the current approach avoids the transformation and a better efficiency of the scheme is observed. We named it the moment-based AP-TVD (MB-AP-TVD) since it uses the default moments of the solution — like the moment limiter (ML) does. Yang and Wang [42] showed that the AP-TVD detector gives better results than the more common detectors used in [29], so the MB-AP-TVD should be even more efficient than those.

The troubled cells were treated with a ML modified for nonuniform meshes with hanging nodes. The limiting stage is done using primitive, characteristic, and conservative variables and then appropriately evaluated. The optimal choice of limiting variables and where to apply the limiter is case-specific, but based on the results of the one-dimensional tests limiting using primitive variables and the MB-AP-TVD detector is the recommended starting point, especially for multidimensional problems since the ML is inherently multidimensional and the characteristic decomposition, slightly better than primitive variables in 1D, cannot be applied in 2 or 3 dimensions. The computational cost due to the conversion from conservative to the other variables seems to be negligible. This Jacobian (and its inverse) is computed each time an element is being limited, but the CPU advantage of conservative variables seems to be lost since worse limiting requires more correction steps of the limiter.

In addition, most test cases were studied with SDC method, what shows that it is an adequate time-integration scheme that could be considered as an alternative to the Runge–Kutta methods for certain applications, especially as the order increases since it is easier to derive and implement. More research is still necessary to determine the numerical properties of SDC-DG, such as its maximum CFL number and its numerical dissipation at different frequencies. For cartesian, low-order cases DG may perform similarly to FD or FV [44]. However, it is important to note that when the conditions are more sophisticated (e.g., unstructured, noncartesian, high-order), where other schemes cannot even be applied, DG still performs well.

The scheme, including our new developments, are relatively simple to implement, robust, with great numerical properties. Thus, it presents a technique that should be exploited for more generic applications.

For steady-state problems the proposed approach may not be highly efficient. Common modifications to improve the convergence to a steady state include some type of filter in time for the limiter and other discrete operations, since they create oscillations that do not let the residual decrease enough. However, the goal of this study is to investigate methods needed for time dependent problems.

p-adaptivity could be useful especially for problems with discontinuities, which are better treated with low-order schemes. Application to the full Navier–Stokes equation will be reported in the near future.

Acknowledgements

The authors wish to thank Prof. Yingjie Liu of School of Mathematics, Georgia Institute of Technology for his helpful suggestions. This work is supported in part by General Electric Infrastructure – Aviation. Also, the authors are thankful for observations made by the reviewers.

Appendix: Moment limiter in two and three dimensions

The 1D momentum limiter presented in Section 3.4 is discussed here in two and three dimensions for completeness.

A.1. ***Two-dimensional moment limiter.*** In this case cross derivatives should be taken into account. Hence, for element l, m,

$$\frac{\partial^{i+j}\tilde{U}_{l,m}}{\partial x_1^i \partial x_2^j} = \operatorname{minmod}\left(\frac{\partial^{i+j}U_{l,m}}{\partial x_1^i \partial x_2^j}, \beta_{ij}D_{ij}^{x_1+}, \beta_{ij}D_{ij}^{x_1-}, \beta_{ij}D_{ij}^{x_2+}, \beta_{ij}D_{ij}^{x_2-}\right) \tag{54}$$

where the frame (x_1, x_2) is a rotation of (x, y) aligned to the computational coordinates (ξ, η) of the current element.

In this case, the limiting starts from orders (p, p), and continuous with the pair $(p, p-1)$ and $(p-1, p)$, then with the pair $(p, p-2)$ and $(p-2, p)$, and so on until $(p, 0)$ and $(0, p)$. Then the loop starts again from $(p-1, p-1)$, and continuous with $(p-1, p-2)$ and $(p-2, p-1)$, and so on. Whenever a pair is not changed the limiting procedure is stopped.

If a neighboring cell is split because of refinement, the average between the two neighboring children is used. If a neighboring cell is coarser because the current cell is more refined, the modes of the neighbor have to be computed as it were refined too. Note that the characteristic decomposition is only consistent in a 1D sense. Given that the ML can be multidimensional, the characteristic decomposition would have to be done in an arbitrary direction. Therefore, for multidimensional cases a primitive-variable decomposition may be more appropriate.

A.2. ***Three-dimensional moment limiter.*** In this case, for element l, m, n,

$$\frac{\partial^{i+j+k}\tilde{U}_{l,m,n}}{\partial x_1^i \partial x_2^j \partial x_3^k} = \operatorname{minmod}\left(\frac{\partial^{i+j+k}U_{l,m,n}}{\partial x_1^i \partial x_2^j \partial x_3^j}, \beta_{ijk}D_{ijk}^{x_1+}, \beta_{ijk}D_{ijk}^{x_1-}, \right.$$
$$\left. \beta_{ijk}D_{ijk}^{x_2+}, \beta_{ijk}D_{ijk}^{x_2-}, \beta_{ijk}D_{ijk}^{x_3+}, \beta_{ijk}D_{ijk}^{x_3-}\right), \tag{55}$$

where the frame (x_1, x_2, x_3) is a rotation of (x, y, z) aligned to the computational coordinates (ξ, η, ζ) of the current element.

In this case, the limiting starts from orders (p, p, p), and continuous for the triad $(p, p, p-1)$, $(p, p-1, p)$ and $(p-1, p, p)$, then for the triad $(p, p, p-2)$, $(p, p-2, p)$ and $(p-2, p, p)$, and so on until $(p, p, 0)$, $(p, 0, p)$ and $(0, p, p)$. Then the loop starts again from $(p-1, p-1, p-1)$, and continues for $(p-1, p-1, p-2)$, $(p-1, p-2, p-1)$ and $(p-2, p-1, p-1)$, and so on. Whenever a triad is not changed the limiting procedure is stopped.

If a neighboring cell is split because of refinement, the average between the four neighboring children is used. Like in the 2D case, a primitive-variable decomposition may be the optimal approach.

References

[1] G. E. Barter and D. L. Darmofal, *Shock capturing with higher-order, PDE-based artificial viscosity*, 18th AIAA Computational Fluid Dynamics Conference (2007-3823), AIAA, 2007.

[2] R. Biswas, K. D. Devine, and J. E. Flaherty, *Parallel, adaptive finite element methods for conservation laws*, Appl. Numer. Math. **14** (1994), no. 1-3, 255–283. MR 1273828 Zbl 0826.65084

[3] B. Cockburn, S. Hou, and C.-W. Shu, *The Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws, IV: The multidimensional case*, Math. Comp. **54** (1990), no. 190, 545–581. MR 90k:65162

[4] B. Cockburn, S. Y. Lin, and C.-W. Shu, *TVB Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws, III: One-dimensional systems*, J. Comput. Phys. **84** (1989), no. 1, 90–113. MR 90k:65161

[5] B. Cockburn and C.-W. Shu, *TVB Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws, II: General framework*, Math. Comp. **52** (1989), no. 186, 411–435. MR 90k:65160

[6] ———, *The Runge–Kutta discontinuous Galerkin method for conservation laws, V: Multidimensional systems*, J. Comput. Phys. **141** (1998), no. 2, 199–224. MR 99c:65181 Zbl 0920.65059

[7] ———, *Runge–Kutta discontinuous Galerkin methods for convection-dominated problems*, J. Sci. Comput. **16** (2001), no. 3, 173–261. MR 2002i:65099 Zbl 1065.76135

[8] A. Dutt, L. Greengard, and V. Rokhlin, *Spectral deferred correction methods for ordinary differential equations*, BIT **40** (2000), no. 2, 241–266. MR 2001e:65104 Zbl 0959.65084

[9] J. E. Flaherty, L. Krivodonova, J.-F. Remacle, and M. S. Shephard, *Aspects of discontinuous Galerkin methods for hyperbolic conservation laws*, Finite Elem. Anal. Des. **38** (2002), no. 10, 889–908. MR 2003e:65176 Zbl 0996.65106

[10] F. X. Giraldo, J. S. Hesthaven, and T. Warburton, *Nodal high-order discontinuous Galerkin methods for the spherical shallow water equations*, J. Comput. Phys. **181** (2002), no. 2, 499–525. MR 2003g:86004 Zbl 1178.76268

[11] S. Gottlieb, D. I. Ketcheson, and C.-W. Shu, *High order strong stability preserving time discretizations*, J. Sci. Comput. **38** (2009), no. 3, 251–289. MR 2010b:65161 Zbl 1203.65135

[12] S. Gottlieb and C.-W. Shu, *Total variation diminishing Runge–Kutta schemes*, Math. Comp. **67** (1998), no. 221, 73–85. MR 98c:65122 Zbl 0897.65058

[13] S. Gottlieb, C.-W. Shu, and E. Tadmor, *Strong stability-preserving high-order time discretization methods*, SIAM Rev. **43** (2001), no. 1, 89–112. MR 2002f:65132 Zbl 0967.65098

[14] J. Grooss and J. S. Hesthaven, *A level set discontinuous Galerkin method for free surface flows*, Comput. Methods Appl. Mech. Engrg. **195** (2006), no. 25-28, 3406–3429. MR 2007e:76204 Zbl 1121.76035

[15] A. Harten, *ENO schemes with subcell resolution*, J. Comput. Phys. **83** (1989), no. 1, 148–184. MR 90i:76010 Zbl 0696.65078

[16] J. Jaffré, C. Johnson, and A. Szepessy, *Convergence of the discontinuous Galerkin finite element method for hyperbolic conservation laws*, Math. Models Methods Appl. Sci. **5** (1995), no. 3, 367–386. MR 96c:65164 Zbl 0834.65089

[17] B. Jeon, J. D. Kress, L. A. Collins, and N. Grønbech-Jensen, *Parallel tree code for two-component ultracold plasma analysis*, Comput. Phys. Commun. **178** (2008), 272–279.

[18] G.-S. Jiang and C.-W. Shu, *Efficient implementation of weighted ENO schemes*, J. Comput. Phys. **126** (1996), no. 1, 202–228. MR 97e:65081 Zbl 0877.65065

[19] A. M. Khokhlov, *Fully threaded tree algorithms for adaptive refinement fluid dynamics simulations*, J. Comput. Phys. **143** (1998), no. 2, 519–543. MR 1631200

[20] R. M. Kirby and G. E. Karniadakis, *De-aliasing on non-uniform grids: algorithms and applications*, J. Comput. Phys. **191** (2003), 249–264. Zbl 1161.76534

[21] L. Krivodonova, J. Xin, J.-F. Remacle, N. Chevaugeon, and J. E. Flaherty, *Shock detection and limiting with discontinuous Galerkin methods for hyperbolic conservation laws*, Appl. Numer. Math. **48** (2004), no. 3-4, 323–338. MR 2056921 Zbl 1038.65096

[22] L. Krivodonova, *Limiters for high-order discontinuous Galerkin methods*, J. Comput. Phys. **226** (2007), no. 1, 879–896. MR 2008j:65162 Zbl 1125.65091

[23] T. Leicht and R. Hartmann, *Error estimation and anisotropic mesh refinement for 3d laminar aerodynamic flow simulations*, J. Comput. Phys. **229** (2010), no. 19, 7344–7360. MR 2011k:76054 Zbl 05785978

[24] Y. Liu, C.-W. Shu, E. Tadmor, and M. Zhang, *Central discontinuous Galerkin methods on overlapping cells with a nonoscillatory hierarchical reconstruction*, SIAM J. Numer. Anal. **45** (2007), no. 6, 2442–2467. MR 2009a:65256 Zbl 1157.65450

[25] Y. Liu, C.-W. Shu, and Z. Xu, *Hierarchical reconstruction with up to second degree remainder for solving nonlinear conservation laws*, Nonlinearity **22** (2009), no. 12, 2799–2812. MR 2011c:35338 Zbl 1184.65086

[26] Y. Liu, C.-W. Shu, and M. Zhang, *Strong stability preserving property of the deferred correction time discretization*, J. Comput. Math. **26** (2008), no. 5, 633–656. MR 2010f:65161 Zbl 1174.65036

[27] M. L. Minion, *Semi-implicit spectral deferred correction methods for ordinary differential equations*, Commun. Math. Sci. **1** (2003), no. 3, 471–500. MR 2005f:65085 Zbl 1088.65556

[28] P.-O. Persson and J. Peraire, *Sub-cell shock capturing for discontinuous Galerkin methods*, 44th AIAA Aerospace Sciences Meeting and Exhibit, AIAA, January 2006.

[29] J. Qiu and C.-W. Shu, *A comparison of troubled-cell indicators for Runge–Kutta discontinuous Galerkin methods using weighted essentially nonoscillatory limiters*, SIAM J. Sci. Comput. **27** (2005), no. 3, 995–1013. MR 2006j:65293 Zbl 1092.65084

[30] ———, *Hermite WENO schemes and their application as limiters for Runge–Kutta discontinuous Galerkin method, II: Two dimensional case*, Comput. & Fluids **34** (2005), no. 6, 642–663. MR 2005j:65086

[31] ———, *Runge–Kutta discontinuous Galerkin method using WENO limiters*, SIAM J. Sci. Comput. **26** (2005), no. 3, 907–929. MR 2005j:65088 Zbl 1077.65109

[32] A. Rault, G. Chiavassa, and R. Donat, *Shock-vortex interactions at high Mach numbers*, J. Sci. Comput. **19** (2003), no. 1-3, 347–371. MR 2004i:76145 Zbl 1039.76047

[33] J.-F. Remacle, J. E. Flaherty, and M. S. Shephard, *An adaptive discontinuous Galerkin technique with an orthogonal basis applied to compressible flow problems*, SIAM Rev. **45** (2003), no. 1, 53–72. MR 2004k:65176 Zbl 1127.65323

[34] J.-F. Remacle, X. Li, M. S. Shephard, and J. E. Flaherty, *Anisotropic adaptive simulation of transient flows using discontinuous Galerkin methods*, Internat. J. Numer. Methods Engrg. **62** (2005), no. 7, 899–923. MR 2005h:76057 Zbl 1078.76042

[35] C.-W. Shu and S. Osher, *Efficient implementation of essentially nonoscillatory shock-capturing schemes. II*, J. Comput. Phys. **83** (1989), no. 1, 32–78. MR 90i:65167 Zbl 0674.65061

[36] E. F. Toro, *Riemann solvers and numerical methods for fluid dynamics: A practical introduction*, 2nd ed., Springer, Berlin, 1999. MR 2000f:76091 Zbl 0923.76004

[37] P. Woodward and P. Colella, *The numerical simulation of two-dimensional fluid flow with strong shocks*, J. Comput. Phys. **54** (1984), no. 1, 115–173. MR 85e:76004 Zbl 0573.76057

[38] Y. Xia, Y. Xu, and C.-W. Shu, *Efficient time discretization for local discontinuous Galerkin methods*, Discrete Contin. Dyn. Syst. Ser. B **8** (2007), no. 3, 677–693. MR 2008e:65307 Zbl 1141.65076

[39] J. Xin and J. E. Flaherty, *Viscous stabilization of discontinuous Galerkin solutions of hyperbolic conservation laws*, Appl. Numer. Math. **56** (2006), no. 3-4, 444–458. MR 2006j:65296 Zbl 1089.65101

[40] Y. Xu and C.-W. Shu, *Local discontinuous Galerkin methods for high-order time-dependent partial differential equations*, Commun. Comput. Phys. **7** (2010), no. 1, 1–46. MR 2011g:65204

[41] Z. Xu, Y. Liu, and C.-W. Shu, *Hierarchical reconstruction for discontinuous Galerkin methods on unstructured grids with a WENO-type linear reconstruction and partial neighboring cells*, J. Comput. Phys. **228** (2009), no. 6, 2194–2212. MR 2010b:65213 Zbl 1165.65392

[42] M. Yang and Z. J. Wang, *A parameter-free generalized moment limiter for high-order methods on unstructured grids*, Adv. Appl. Math. Mech. **1** (2009), no. 4, 451–480. MR 2010h:65182

[43] L. Yuan and C.-W. Shu, *Discontinuous Galerkin method based on non-polynomial approximation spaces*, J. Comput. Phys. **218** (2006), no. 1, 295–323. MR 2008c:65267 Zbl 1104.65094

[44] T. Zhou, Y. Li, and C.-W. Shu, *Numerical comparison of WENO finite volume and Runge–Kutta discontinuous Galerkin methods*, J. Sci. Comput. **16** (2001), no. 2, 145–171. MR 2002k:65133 Zbl 0991.65083

[45] H. Zhu and J. Qiu, *Adaptive Runge–Kutta discontinuous Galerkin methods using different indicators: one-dimensional case*, J. Comput. Phys. **228** (2009), no. 18, 6957–6976. MR 2011a:65339 Zbl 1173.65339

Received December 15, 2010. Revised April 8, 2012.

LEANDRO D. GRYNGARTEN: leandro@gatech.edu
Aerospace Engineering, Georgia Institute of Techonology, 270 Ferst Drive, Atlanta, GA 30332, United States

ANDREW SMITH: andrew.g.smith@gatech.edu
Aerospace Engineering, Georgia Institute of Techonology, 270 Ferst Drive, Atlanta, GA 30332, United States

SURESH MENON: suresh.menon@ae.gatech.edu
Aerospace Engineering, Georgia Institute of Techonology, 270 Ferst Drive, Atlanta, GA 30332, United States