SIAM J. SCI. COMPUT.
Vol. 42, No. 6, pp. A3677–A3703

A STABILIZED DG CUT CELL METHOD FOR DISCRETIZING THE LINEAR TRANSPORT EQUATION*

CHRISTIAN ENGWER†, SANDRA MAY‡, ANDREAS NÜßING†, AND FLORIAN STREITBÜRGER‡

Abstract. We present new stabilization terms for solving the linear transport equation on a cut cell mesh using the discontinuous Galerkin (DG) method in two dimensions with piecewise linear polynomials. The goal is to allow for explicit time stepping schemes despite the presence of cut cells. Using a method of lines approach, we start with a standard upwind DG discretization for the background mesh and add penalty terms that stabilize the solution on small cut cells in a conservative way. Then one can use explicit time stepping, even on cut cells, with a time step length that is appropriate for the background mesh. In one dimension, we show monotonicity of the proposed scheme with a constant basis and total variation diminishing in the means stability for piecewise linear polynomials. We also present numerical results in one and two dimensions that support our theoretical findings.

Key words. cut cell, unfitted finite elements, discontinuous Galerkin method, stabilization, small cell problem, hyperbolic conservation law

AMS subject classifications. 65M60, 65M12, 65M20, 35L02, 35L65

DOI. 10.1137/19M1268318

1. Introduction.

Finite element and more recently discontinuous Galerkin (DG) schemes have successfully been used on a huge variety of equations and are overall well understood. Therefore, research has advanced from solving model problems on simple geometries to trying to solve real-world problems. As a result, grid generation has become a major issue. Simulating flow around an airplane, flow in blood vessels, or phase transitions requires meshing very complicated geometries, which are often implicitly given or as computer-aided design models. The generation of corresponding body-fitted meshes is a very involved process.

In recent years, the usage of embedded boundary or cut cell meshes has become increasingly popular. The details of these approaches vary. In this work, we will focus on the approach of cutting a given geometry out of a given background mesh, resulting in so-called *cut cells* along the boundary of the embedded object. These cells can have various shapes and may in particular become arbitrarily small. Special schemes must be developed to guarantee stability on these cells. There already exists a significant amount of literature for stabilizing problems of elliptic and parabolic type on cut cell meshes; see, e.g., [1, 2, 6, 12, 14, 19, 25, 30, 39] and the references therein. For small cells, stability of higher derivatives is lost, and different stabilization techniques have been proposed. A successful approach is the ghost penalty stabilization [13], sometimes referred to as the cutFEM method [14].

For hyperbolic conservation laws on cut cell meshes, different problems arise compared to solving elliptic and parabolic PDEs. Probably the biggest issue is the *small*

*Submitted to the journal's Methods and Algorithms for Scientific Computing section June 13, 2019; accepted for publication (in revised form) July 20, 2020; published electronically December 1, 2020.

https://doi.org/10.1137/19M1268318

†Applied Mathematics Münster, Institute for Analysis and Numerics, University of Münster, Münster, Germany (christian.engwer@uni-muenster.de, andreas.nuessing@uni-muenster.de).

‡Department of Mathematics, TU Dortmund University, Dortmund, Germany (Sandra.May@math.tu-dortmund.de, florian.streitbuerger@math.tu-dortmund.de).

cell problem—that standard explicit schemes are not stable on the arbitrarily small cut cells when the time step is chosen according to the cell size of the background mesh. An additional complication is the fact that there is typically no concept of coercivity that could serve as a guideline for constructing stabilization terms. Furthermore, the numerical scheme should satisfy properties such as monotonicity and total variation diminishing (TVD) stability in order to avoid overshoots or oscillations in the presence of discontinuities, which could result in unphysical solutions.

In the context of *finite volume (FV) schemes*, which have traditionally been used for the solution of hyperbolic conservation laws, already several solution approaches exist that solve the small cell problem while dealing with arbitrarily small cut cells, e.g., the *flux redistribution* method [16, 18, 37], the *h-box method* [9, 11, 26], and the *mixed explicit implicit scheme* [34]. For the solution of hyperbolic conservation laws on cut cell meshes using *DG schemes*, very little work has been done. As one of the first such works, Bastian et al. [7] use an implicit Euler method to overcome the small cell problem for a linear transport problem, but this approach is bound to first-order accuracy. For explicit time stepping schemes, some work relies on so-called *cell merging* or *cell agglomeration* [29, 36]. In this approach, cut cells that are too small are merged with neighboring cells. As this approach puts the complexity back into the mesh generation, we do not want to consider it here. Recently, Gürkan and Massing [24] suggested a new scheme for solving the *steady* advection-reaction problem on a cut cell mesh with potentially arbitrarily small cut cells that uses a ghost penalty approach. However, the authors consider only the steady case. Also, Sticko and Kreiss [41] have worked on using penalty terms to stabilize the solution of the wave equation. In both cases, the penalty term employed has great similarity to the ghost penalty stabilization used for elliptic problems [13].

To the best of our knowledge, the scheme suggested in this work is the first DG scheme for overcoming the small cell problem for time-dependent scalar conservation laws by dealing with the potentially arbitrarily small cut cells while ensuring monotonicity and TVD stability. Therefore, we will focus on the linear advection equation as the standard model problem in the following. Many problems that occur for solving more complex hyperbolic equations already show up for this simple model.

Our approach is based on stabilizing the spatial discretization. One is free in the choice of the time stepping scheme. In particular, our stabilized spatial discretization allows for using standard *explicit* time stepping schemes everywhere, even on cut cells. We obtain stability on cut cells by adding penalty terms. In that sense, the suggested scheme is similar to the ghost penalty approach [13]. However, the details of the terms differ significantly. In particular, the terms are fundamentally different from the ones used in [24, 41]. Conceptually, the terms are designed to restore the proper domains of dependence on small cut cells and their neighbors, similar to the idea of the h-box method but without an explicit geometry reconstruction. We therefore refer to our method as *Domain-of-Dependence* (DoD) stabilization. In this work, we consider piecewise *linear* polynomials in one and two space dimensions.

This paper is structured as follows. In section 2.1, we provide background information, such as mesh and geometry information as well as the standard scheme that we plan to stabilize. Section 3 contains the core of this work, the formulation of the stabilization terms in two dimensions (2D). In section 4, we focus on the 1D case for a better understanding of the proposed stabilization and provide theoretical results for the case of piecewise constant and piecewise linear polynomials. We discuss implementational aspects of the proposed scheme in section 5. And in section 6, we

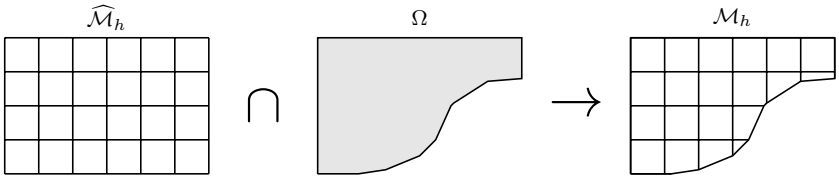


FIG. 1. *Construction of a cut cell mesh $\mathcal{M}_h$: The background mesh $\widehat{\mathcal{M}}_h$ of the larger domain $\widehat{\Omega}$ is intersected with the computational domain Ω, leading to cut cells $E = \widehat{E} \cap \Omega$, where $\widehat{E} \in \widehat{\mathcal{M}}_h$ is an element of the background mesh.*

provide numerical results in both 1D and 2D that support our theoretical findings. We conclude with a short summary and an outlook in section 7.

2. Discretization.

2.1. Preliminaries.

In this work, we focus on the linear advection equation as the classic model problem for hyperbolic scalar conservation laws, which is given by

$$u_t + \langle \beta, \nabla u \rangle = 0 \quad \text{in } \Omega \times (0,T), \tag{2.1a}$$

$$u = g \quad \text{on } \partial\Omega^{\text{in}} \times (0,T), \tag{2.1b}$$

$$u = u_0 \quad \text{on } \Omega \times \{t = 0\}. \tag{2.1c}$$

Here, $\Omega \subset \mathbb{R}^d, d \in \{1,2\}$, is an open, connected, polygonal domain, and $\partial\Omega$ denotes its boundary, $n \in \mathbb{R}^d$ its outer unit normal vector on $\partial\Omega$, and $\partial\Omega^{\text{in}} := \{x \in \partial\Omega : \langle \beta(x), n(x) \rangle < 0\}$ its inflow boundary. The velocity field $\beta \in \mathbb{R}^d$ is assumed to satisfy $\nabla \cdot \beta = 0$, and $\langle \cdot, \cdot \rangle$ denotes the canonical scalar product in $\mathbb{R}^d$.

To construct our discrete approximation, we first consider a larger polygonal domain $\widehat{\Omega} \supset \Omega$, which is easy to mesh; see Figure 1.

DEFINITION 2.1 (cut cell mesh). *Given a background mesh $\widehat{\mathcal{M}}_h$ of $\widehat{\Omega}$, we introduce a cut cell mesh $\mathcal{M}_h$. Let $\widehat{\mathcal{M}}_h$ be a nonoverlapping set of (open) shape-regular elements $\widehat{E}$ such that $\bigcup_{\widehat{E} \in \widehat{\mathcal{M}}_h} \overline{\widehat{E}} = \overline{\widehat{\Omega}}$. We assume that $\widehat{\mathcal{M}}_h$ corresponds to a Cartesian background mesh with square cells (i.e., $h_x = h_y = h$). Intersecting Ω and the background mesh induces the cut cell mesh*

$$\mathcal{M}_h := \left\{ E := \widehat{E} \cap \Omega \;\middle|\; \widehat{E} \in \widehat{\mathcal{M}}_h \right\}.$$

Note that $\mathcal{M}_h$ is a partition of Ω consisting of structured (Cartesian) cells and cut cells. The internal and external skeletons of the partitioning are given by

$$\Gamma_{int} = \{ e_{E_1,E_2} = \partial E_1 \cap \partial E_2 \mid E_1, E_2 \in \mathcal{M}_h \text{ and } E_1 \neq E_2 \text{ and } |e_{E_1,E_2}| > 0 \}, \tag{2.2}$$

$$\Gamma_{ext} = \{ e_E = \partial E \cap \partial\Omega \mid E \in \mathcal{M}_h \text{ and } |e_E| > 0 \}, \tag{2.3}$$

with $|e|$ denoting the length of an edge e.

DEFINITION 2.2 (discrete function space). *Following the usual DG formulation, we define the discrete space $V_h^k(\mathcal{M}_h) \subset L^2(\Omega)$ by*

$$V_h^k(\mathcal{M}_h) = \left\{ v_h \in L^2(\Omega) \,|\, \forall E \in \mathcal{M}_h, v_h|_E \in P^k(E) \right\}, \tag{2.4}$$

where P^k denotes the polynomial space of degree k.

In this paper, we will only consider the cases $k = 0$ and $k = 1$, i.e., piecewise constant and piecewise linear polynomials.

On the skeleton Γ_{int}, functions $u_h \in V_h^k$ are not well defined. Therefore, denoting by n_E the outer unit normal vector of a cell E, we make the following definitions.

DEFINITION 2.3 (jump and average). *The* jump *in normal direction over a face* $e_{E_1,E_2} = \partial E_1 \cap \partial E_2$ *between two elements* E_1 *and* E_2 *is defined as*

$$[\![u_h]\!] := u_h|_{E_1} n_{E_1} + u_h|_{E_2} n_{E_2}. \tag{2.5}$$

Note that the jump of a scalar quantity is vector-valued, i.e., $[\![u_h]\!] \in \mathbb{R}^2$. *The (scalar-valued)* average *on a face is defined as*

$$\{\!\{u_h\}\!\} := \frac{1}{2}(u_h|_{E_1} + u_h|_{E_2}).$$

Note that for $V_h^1(\mathcal{M}_h)$, the jump and average vary linearly over the face.

2.2. Unstabilized DG formulation. We now introduce the DG scheme as used on structured background cells. Additional stabilization terms will be necessary on small cut cells; see section 3. We will use a method of lines approach and first discretize (2.1) in space and then in time.

The spatial discretization of (2.1) corresponds to the standard DG discretization using upwind fluxes [20]: Find $u_h \in V_h^k(\mathcal{M}_h)$ such that

$$(d_t u_h(t), w_h)_{L^2} + a_h^{\text{upw}}(u_h(t), w_h) + l_h(w_h) = 0 \quad \forall w_h \in V_h^k(\mathcal{M}_h), \tag{2.6}$$

with

$$\begin{aligned} a_h^{\text{upw}}(u_h, w_h) := &- \sum_{E\in\mathcal{M}_h} \int_E u_h \langle \beta, \nabla_h w_h \rangle \,\mathrm{d}x + \sum_{e\in\Gamma_{\text{ext}}} \int_e \langle \beta, n\rangle^{\oplus} u_h w_h \,\mathrm{d}s \\ &+ \sum_{e\in\Gamma_{\text{int}}} \int_e \left(\{\!\{u_h\}\!\} \langle \beta, [\![w_h]\!] \rangle + \frac{1}{2} |\langle \beta, n_e\rangle| \langle [\![u_h]\!], [\![w_h]\!] \rangle \right) \mathrm{d}s\,, \end{aligned} \tag{2.7}$$

$$l_h(w_h) := - \sum_{e\in\Gamma_{\text{ext}}} \int_e \langle \beta, n\rangle^{\ominus} g\, w_h \,\mathrm{d}s\,, \tag{2.8}$$

where $n_e \in \mathbb{R}^2$ denotes a unit normal on a face e (of arbitrary but fixed orientation). Further, $(\cdot,\cdot)_{L^2}$ denotes the scalar product in $L^2(\Omega)$. The negative and positive components of a quantity $x \in \mathbb{R}$ are defined as $x^{\ominus} := \frac{|x|-x}{2}$ and $x^{\oplus} := \frac{|x|+x}{2}$. Note that $x^{\ominus}, x^{\oplus} \geq 0$.

The new method does not rely on a particular *time stepping scheme*. Nevertheless, it is desirable that the scheme be explicit in order to allow for fast operator evaluation and to ease the use of limiters. Furthermore, it should be of the same order of accuracy as the spatial discretization and result in a discretization that is TVD.

While for piecewise constant polynomials in space the explicit Euler scheme suffices, it will diminish the convergence order for $V_h^1(\mathcal{M}_h)$. We thus decided to employ the explicit second-order strong stability preserving (SSP) Runge–Kutta (RK) scheme [23], that is given for the ODE $y_t = F(y)$ by

$$\begin{aligned} y^{(1)} &= y^n + \Delta t F(y^n), \\ y^{n+1} &= \frac{1}{2} y^n + \frac{1}{2} y^{(1)} + \frac{1}{2} \Delta t F(y^{(1)}). \end{aligned} \tag{2.9}$$

A *limiter* is necessary to avoid unphysical oscillations close to discontinuities when using piecewise linear polynomials. We have chosen a limiter of Barth–Jespersen type [3] that has been extended to the DG setting by exploiting the structure of the local Taylor basis [31]. This limits the gradient in such a way that the local solution of a cell E evaluated at each neighboring centroid does not overshoot/undershoot the maximum/minimum taken over the cell's E average value and the average values of all of cell's E face neighbors. The limiter is applied as a postprocessing step to both the intermediate solution $u^{(1)}$ and the solution u^{n+1}.

3. DoD stabilization. There are two problems that need to be dealt with in order to ensure stability on cut cells and to avoid overshoots and oscillations. First, the average mass in each cell must be controlled so that a piecewise constant approximation does not produce oscillations. Second, the gradients must be controlled to avoid oscillations within a single cell and the creation of new extrema at cell faces. Both goals must be reached without violating mass conservation. For this purpose, we introduce two stabilization terms, J_h^0 and J_h^1: The first one will control average values and the second one gradients.

One way to think of the stabilization terms is that they ensure that only a certain fraction of the inflow of a small cut cell stays in the small cut cell. The remaining part of the cut cell's inflow is directly transported through the small cut cell into its outflow neighbors.

DEFINITION 3.1 (inflow and outflow faces). *For each cell E, we denote the set of inflow faces and outflow faces as*

$$\mathcal{F}_i(E) := \{e \in \partial E \mid \langle \beta, n_E \rangle < 0 \text{ on } e\},$$
$$\mathcal{F}_o(E) := \{e \in \partial E \mid \langle \beta, n_E \rangle \geq 0 \text{ on } e\}.$$

We further define

$$|\mathcal{F}_i(E)| = \sum_{e \in \mathcal{F}_i(E)} |e| \quad \text{and} \quad |\mathcal{F}_o(E)| = \sum_{e \in \mathcal{F}_o(E)} |e|.$$

DEFINITION 3.2 (neighbors of $E \in \mathcal{M}_h$). *A neighbor $\mathcal{N}_{E,e}$ of a cell $E \in \mathcal{M}_h$ over a given face $e \in \partial E$ is uniquely defined by the relation*

$$\overline{\mathcal{N}}_{E,e} \cap \overline{E} = e.$$

The sets of inflow and outflow neighbors $\mathcal{N}_i(E)$ and $\mathcal{N}_o(E)$ of a cell E are defined by

$$\mathcal{N}_i(E) := \{\mathcal{N}_{E,e} \,|\, e \in \mathcal{F}_i(E)\}\,,$$
$$\mathcal{N}_o(E) := \{\mathcal{N}_{E,e} \,|\, e \in \mathcal{F}_o(E)\}\,.$$

DEFINITION 3.3 (capacity). *We define the capacity of a cut cell E as*

$$\alpha_{E,\omega} := \min\left(\omega \frac{|E|}{\Delta t \int_{\partial E} \langle \beta, n_E \rangle^{\ominus} \, \mathrm{d}s}, 1\right), \quad \omega \in (0,1]. \tag{3.1}$$

For $\omega = 1$, the capacity measures the fraction of the inflow that is allowed to flow into the cut cell E without producing overshoot.

We further denote by $\mathcal{I}$ the set of cut cells on which the stabilization should be applied. The idea is that only small cut cells need stabilization,

$$\mathcal{I} = \{E \in \mathcal{M}_h \mid |E| < \lambda h \min(|\mathcal{F}_i(E)|, |\mathcal{F}_o(E)|)\}\,, \tag{3.2}$$

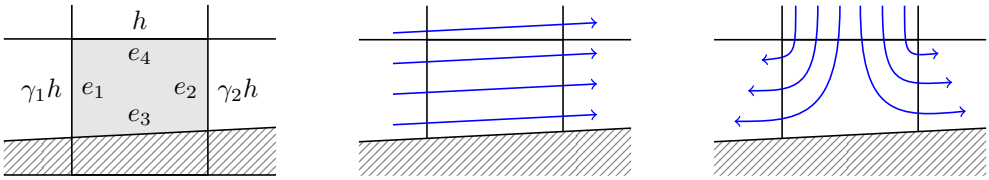


FIG. 2. *4-sided cut cell and the two basic flow configurations:* $\gamma_1, \gamma_2 \in (0, 1]$ *denote the fraction of the edge length of the cut cell compared to* h.

with a CFL constant $\lambda \in (0, \frac{1}{2}]$. Additional stability constraints on λ might apply, e.g., depending on the background mesh or the polynomial degree. This choice of $\mathcal{I}$ is motivated by the goal to ensure the following lemma.

LEMMA 3.4. *If for the chosen time step the constraint* $\Delta t \leq \lambda \frac{h}{\|\beta(x)\|}$ *holds for all* $x \in \Omega$*, the capacity for any cell* $E \notin \mathcal{I}$ *is* $\alpha_{E,1} = 1$*; i.e., these cells do not require stabilization.*

Proof. We first consider the denominator of $\alpha_{E,1}$ and estimate (using the abbreviations $\mathcal{F}_i = \mathcal{F}_i(E)$ and $\mathcal{F}_o = \mathcal{F}_o(E)$)

$$\Delta t \int_{\partial E} \langle \beta, n_E \rangle^{\ominus} \, \mathrm{d}s = \Delta t \sum_{e \in \mathcal{F}_i} \int_e - \langle \beta, n_E \rangle \, \mathrm{d}s \leq \sum_{e \in \mathcal{F}_i} \int_e \Delta t \|\beta\| \, \mathrm{d}s \leq \lambda h |\mathcal{F}_i|.$$

Due to incompressibility, there holds $\sum_{e \in \mathcal{F}_i} \int_e \langle \beta, n_E \rangle \, \mathrm{d}s = - \sum_{e \in \mathcal{F}_o} \int_e \langle \beta, n_E \rangle \, \mathrm{d}s$. Therefore,

$$\Delta t \int_{\partial E} \langle \beta, n_E \rangle^{\ominus} \, \mathrm{d}s = \Delta t \sum_{e \in \mathcal{F}_o} \int_e \langle \beta, n_E \rangle \, \mathrm{d}s \leq \lambda h |\mathcal{F}_o|.$$

We combine the results to get

$$\Delta t \int_{\partial E} \langle \beta, n_E \rangle^{\ominus} \, \mathrm{d}s \leq \lambda h \min(|\mathcal{F}_i|, |\mathcal{F}_o|).$$

As for any cell $E \notin \mathcal{I}$ it holds that $|E| \geq \lambda h \min(|\mathcal{F}_i|, |\mathcal{F}_o|)$, it follows that

$$\frac{|E|}{\Delta t \int_{\partial E} \langle \beta, n_e \rangle^{\ominus} \, \mathrm{d}s} \geq \frac{|E|}{\lambda h \min(|\mathcal{F}_i|, |\mathcal{F}_o|)} \geq 1,$$

and thus $\alpha_{E,1} = 1$ for any cell not in $\mathcal{I}$. ∎

In 2D for straight cuts, cut cells are either 3-sided, 4-sided, or 5-sided. For the above definition of $\mathcal{I}$, we can give an explicit statement about which cells are stabilized.

LEMMA 3.5. *Consider the Cartesian background mesh in 2D with square cells and an elementwise straight embedded boundary on which* $\langle \beta, n \rangle = 0$ *holds. Then the set* $\mathcal{I}$ *as defined in* (3.2) *only contains 3-sided cut cells if* $\lambda \leq \frac{1}{2}$.

Proof. To show that neither 4- nor 5-sided cut cells are in $\mathcal{I}$, we will bound $|E|$ from below by an upper bound of $\min(|\mathcal{F}_i|, |\mathcal{F}_o|)$.

For 4-sided cells (see Figure 2), two different flow conditions must be considered. In case 1, the flow is parallel to the cut (picture in the middle). In case 2, we see an element with a stagnation point (picture to the right). Here the inflow is roughly orthogonal to the cut, and both cut faces are outflow faces. The same holds true if we change the sign of the velocity.

We define

$$\gamma^- = \min(\gamma_1, \gamma_2), \quad \gamma^+ = \max(\gamma_1, \gamma_2).$$

The volume $|E|$ is given by

$$|E| = \tfrac{1}{2}(\gamma^+ + \gamma^-)h^2.$$

If $|E| > \frac{h^2}{2}$, it is easy to conclude that $E \notin \mathcal{I}$. If $|E| \leq \frac{h^2}{2}$, it holds that $(\gamma^+ + \gamma^-) \leq 1$. Regarding the face area, we can bound $\min(|\mathcal{F}_i|, |\mathcal{F}_o|)$ in the two cases by

$$\text{Case 1:} \quad \min(|\mathcal{F}_i|, |\mathcal{F}_o|) \leq \gamma^+ h \quad \Rightarrow \quad \frac{|E|}{\min(|\mathcal{F}_i|, |\mathcal{F}_o|)} \geq \tfrac{h}{2}\frac{\gamma^+ + \gamma^-}{\gamma^+} \geq \tfrac{h}{2},$$

$$\text{Case 2:} \quad \min(|\mathcal{F}_i|, |\mathcal{F}_o|) = (\gamma^- + \gamma^+)h \quad \Rightarrow \quad \frac{|E|}{\min(|\mathcal{F}_i|, |\mathcal{F}_o|)} \geq \tfrac{h}{2}\frac{\gamma^+ + \gamma^-}{\gamma^+ + \gamma^-} \geq \tfrac{h}{2}.$$

For 5-sided cut cells, one can show the same lower bound. Let γ^- and γ^+ again denote the relative size of the smaller and longer cut edge. The volume is $|E| = (1 - \frac{(1-\gamma^-)(1-\gamma^+)}{2})h^2$. The face area can be bounded from above by $\min(|\mathcal{F}_i|, |\mathcal{F}_o|) \leq (1 + \gamma^-)h$ so that we again obtain the lower bound $\frac{|E|}{\min(|\mathcal{F}_i|, |\mathcal{F}_o|)} \geq \frac{h}{2}$. ∎

Remark 3.1. The condition in (3.2) implies that a cut cell is only stabilized if it is small in the flow direction. Consider the situation of a Cartesian mesh that is cut by a horizontal line, resulting in many cut cells of the shape of slim rectangles along the cut boundary. The flow direction is parallel to the cut boundary. Therefore, even though the cut cells might have a small volume, they actually have regular length in the flow direction. These cells are not stabilized.

Remark 3.2. For $\lambda = \frac{1}{2}$, the set $\mathcal{I}$ contains almost all 3-sided cut cells. First, there is no need to stabilize triangular cells, for which the length of the cell in the flow direction (corresponding to the length of the hypotenuse) is even larger than h. Second, often in 2D, λ is chosen smaller than $\frac{1}{2}$, making the set $\mathcal{I}$ of cells that need stabilization smaller. In our numerical tests, we only stabilize a subset of 3-sided cut cells.

In the following presentation, we assume that for every pair of neighboring cut cells, at most one of the two cut cells is an element of $\mathcal{I}$. While it is possible to extend the proposed scheme to several neighboring cut cells, the implementation would be more difficult, as it introduces long-range couplings. As the set $\mathcal{I}$ is relatively small, this is often not necessary. We design the stabilization such that the CFL condition for explicit time stepping schemes essentially only depends on the resolution of the background mesh and not on the size of small cut cells from the set $\mathcal{I}$.

The stabilization is designed to restore the proper domains of dependence; i.e., it ensures that the outflow neighbors of a small cut cell get information from the inflow neighbors of that cut cell. The stabilization term at a given point on an *outflow* face thus depends on the solution in the inflow cell.

In order to incorporate information from upstream cells, we introduce the following definitions.

DEFINITION 3.6 (extension operator and extended jump). *We first introduce an extension operator $\mathcal{L}^{\text{ext}}_{E'}$ that extends a function $u_h \in V_h^k$ from a cell $E' \in \mathcal{M}_h$ to the whole domain Ω:*

$$\mathcal{L}^{\text{ext}}_{E'} : V_h^k(\mathcal{M}_h)|_{E'} \to P^k(\Omega) \qquad \text{s.t. } \mathcal{L}^{\text{ext}}_{E'}(u_h) \in P^k(\Omega) \text{ and } \mathcal{L}^{\text{ext}}_{E'}(u_h)|_{E'} = u_h|_{E'}.$$

This extension is trivial, as $u_h|_{E'} \in P^k(E')$ *and polynomials can simply be evaluated outside of their original support.*

Using $\mathcal{L}^{\text{ext}}_{E'}$*, we introduce an extended jump operator* $[\cdot]^{\text{ext}}_E$ *on a cell* E *as*

$$[u_h]^{\text{ext}}_E(x) := \left[\sum_{E' \in \mathcal{N}_i(E)} \omega_{E,E'}(x)\mathcal{L}^{\text{ext}}_{E'}(u_h)(x)\right] - u_h(x)$$

with a weighting function $\omega_{E,E'}$ *and* $x \in \overline{E}$*. For the considered Cartesian background mesh, the set* $\mathcal{I}$ *of stabilized cut cells only contains* 3*-sided cells. Due to the incompressibility condition and the no-flux boundary condition on the cut boundary, this implies that we have exactly one inflow neighbor per stabilized cut cell, and thus* $\omega_{E,E'} \equiv 1$*.*

The penalty term J^0_h is now constructed in such a way that any inflow that exceeds the capacity of a small cut cell is moved to its downwind neighbors and is given by

$$\begin{aligned} J^0_h(u_h, w_h) &= \sum_{E \in \mathcal{I}} J^{0,E}_h(u_h, w_h), \qquad \text{with} \\ J^{0,E}_h(u_h, w_h) &= \eta_E \sum_{e \in \mathcal{F}_o(E)} \int_e [u_h]^{\text{ext}}_E \langle \beta, [\![w_h]\!] \rangle \, \mathrm{d}s\,, \end{aligned} \tag{3.3}$$

and with $\eta_E = 1 - \alpha_{E,\frac{1}{2}}$.

This stabilization term is applied on the outflow faces and transports the fraction of the flow $\eta_E = 1 - \alpha_{E,\frac{1}{2}}$, which should *not* remain in the cut cell, into its downwind neighbors. We only allow the fraction $\alpha_{E,\frac{1}{2}}$ instead of $\alpha_{E,1}$ to stay within the cut cell, as the latter one would result in too restrictive slope limiting, leading to close to zero gradients on the cut cells. Note that the extended jump $[u_h]^{\text{ext}}_E$, which contains information from the inflow neighbors of E, is evaluated on the outflow edge of E.

The main task of the second stabilization term J^1_h is to restore control over gradients. It describes a volume transport within the cut cell, driven by a flux $[u_h]^{\text{ext}}_E \beta$. This additional flux properly redistributes the quantity $\eta_E = 1 - \alpha_{E,\frac{1}{2}}$ within the cut cell and is given by

$$\begin{aligned} J^1_h(u_h, w_h) &= \sum_{E \in \mathcal{I}} J^{1,E}_h(u_h, w_h), \qquad \text{with} \\ J^{1,E}_h(u_h, w_h) &= -\eta_E \int_E [u_h]^{\text{ext}}_E \langle \beta, \nabla w_h \rangle \, \mathrm{d}x\,. \end{aligned} \tag{3.4}$$

The DoD stabilized spatial discretization is then given by the following: Find $u_h \in V^k_h(\mathcal{M}_h)$ such that $\forall\, w_h \in V^k_h(\mathcal{M}_h)$

$$(d_t u_h(t), w_h)_{L^2} + a^{\text{upw}}_h(u_h(t), w_h) + J^0_h(u_h, w_h) + J^1_h(u_h, w_h) + l_h(w_h) = 0. \tag{3.5}$$

Next, we will examine the mass conservation properties of the stabilization terms.

DEFINITION 3.7 (indicator function). *The indicator function of a cell or cell patch is defined as*

$$\chi_E(x) = \begin{cases} 1 & \text{for } x \in E, \\ 0 & \text{otherwise.} \end{cases} \tag{3.6}$$

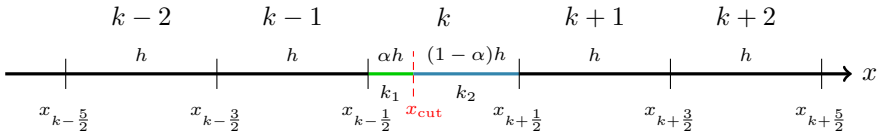


FIG. 3. *Model problem MP: equidistant mesh with cell k split into two cells of lengths αh and $(1-\alpha)h$ with $\alpha \in (0, \frac{1}{2}]$.*

LEMMA 3.8. *Let $E \in \mathcal{I}$. Then*

$$J_h^{0,E}(u_h, \chi_{E\cup\mathcal{N}_o(E)}) + J_h^{1,E}(u_h, \chi_{E\cup\mathcal{N}_o(E)}) = 0. \tag{3.7}$$

Proof. For the considered test function, $J_h^{1,E}(u_h, \chi_{E\cup\mathcal{N}_o(E)}) = 0$. Due to the linearity of $J_h^{0,E}$ in w_h, there holds

$$\begin{aligned} &J_h^{0,E}(u_h, \chi_{E\cup\mathcal{N}_o(E)}) = J_h^{0,E}(u_h, \chi_E) + J_h^{0,E}(u_h, \chi_{\mathcal{N}_o(E)}) \\ &= \sum_{e\in\mathcal{F}_o(E)} \int_e \eta_E \, [u_h]_E^{\text{ext}} \, \langle \beta, \chi_E n_E \rangle \, \mathrm{d}s + \sum_{e\in\mathcal{F}_o(E)} \int_e \eta_E \, [u_h]_E^{\text{ext}} \, \langle \beta, -\chi_{\mathcal{N}_o(E)} n_E \rangle \, \mathrm{d}s \\ &= 0. \end{aligned}$$

∎

This result (together with the local mass conservation properties of the standard DG discretization (2.7)) guarantees *local mass conservation* in the extended control volume $E \cup \mathcal{N}_o(E)$ of a small cut cell $E \in \mathcal{I}$: The same amount of mass that is *not* allowed to stay in E is redistributed to E's outflow neighbors. This slightly extended setting of local mass conservation is a natural consequence of our stabilization.

4. The 1D case. For a better understanding of the proposed scheme and for the validation of some theoretical properties, we focus on the 1D case in this section. Without loss of generality, we consider the interval $I = (0, 1)$ and assume $\beta > 0$ to be constant. Our PDE is given by

$$u_t(x,t) + \beta u_x(x,t) = 0 \text{ in } I \times (0,T), \quad u(0,t) = g(t) \text{ for } t \in (0,T), \tag{4.1}$$

with initial data $u(x,0) = u_0(x)$. For the analysis in this section, we will focus on solving the model problem MP shown in Figure 3: We discretize the interval I in N cells $E_j = (x_{j-\frac{1}{2}}, x_{j+\frac{1}{2}})$ of equidistant length h. Then we split one cell, the cell k, in two cells of lengths αh (referred to as cell k_1) and $(1-\alpha)h$ (referred to as cell k_2) with $\alpha \in (0, \frac{1}{2}]$.

Remark 4.1 (notation). Different from 2D, there is a natural order of cells in 1D. In 1D, we will therefore use the notation indicated in Figure 3: We will refer to "cell j" and denote faces by $x_{j\pm1/2}$ instead of using "cell E", "cell E_j", or "face e."

Using the notation $x_{j+\frac{1}{2}}^{\pm} = \lim_{\varepsilon\to 0} x_{j+\frac{1}{2}} \pm \varepsilon$, the bilinear form (2.7) simplifies to

$$\begin{aligned} a_h^{\text{upw}}(u_h, w_h) = &- \sum_{j=1, j\neq k}^{N} \int_j \beta u_h \partial_x w_h \, \mathrm{d}x - \int_{k_1} \beta u_h \partial_x w_h \, \mathrm{d}x - \int_{k_2} \beta u_h \partial_x w_h \, \mathrm{d}x \\ &+ \beta u_h(x_{N+\frac{1}{2}}^-) w_h(x_{N+\frac{1}{2}}^-) + \sum_{j=1}^{N-1} \beta u_h(x_{j+\frac{1}{2}}^-) \, [\![w_h]\!]_{j+\frac{1}{2}} + \beta u_h(x_{\text{cut}}^-) \, [\![w_h]\!]_{\text{cut}} . \end{aligned} \tag{4.2}$$

The definitions of the jump and of l_h, given by (2.5) and (2.8) in 2D, reduce in 1D to

$$[\![w_h]\!]_{j+\frac{1}{2}} = w_h(x^-_{j+\frac{1}{2}}) - w_h(x^+_{j+\frac{1}{2}}), \quad l_h(w_h) := -\beta g(0) w_h(x^+_{\frac{1}{2}}). \tag{4.3}$$

Further, we define the CFL number

$$\lambda = \frac{\beta \Delta t}{h}. \tag{4.4}$$

Note that λ is chosen only with respect to the background mesh width h, but the volume fraction $\alpha \in \left(0, \frac{1}{2}\right]$ in MP is allowed to become arbitrarily close to 0.

For the considered model problem MP, the set of cells that need to be stabilized consists of $\mathcal{I} = \{k_1\}$, and therefore J_h^0 and J_h^1 coincide with J_h^{0,k_1} and J_h^{1,k_1}, respectively. The stabilization terms J_h^{0,k_1} and J_h^{1,k_1}, given by (3.3) and (3.4) in 2D, simplify significantly for the considered setup in 1D: For cell k_1, the inflow face is $k - \frac{1}{2}$, and the outflow face is k_{cut}. Therefore, the extended jump operator from definition 3.6 simplifies to

$$[u_h]^{\text{ext}}_{k_1}(x) = u_{h,k-1}(x) - u_{h,k_1}(x). \tag{4.5}$$

Here, $u_{h,j}$ denotes the (extended) polynomial solution from cell j. The DoD stabilization term is then given by

$$J_h(u_h, w_h) = \beta\, \eta_{k_1}\, [u_h]^{\text{ext}}_{k_1}(x_{\text{cut}})\, [\![w_h]\!]_{\text{cut}} - \beta\, \eta_{k_1} \int_{k_1} [u_h]^{\text{ext}}_{k_1}(x) \partial_x w_h \, \mathrm{d}x \tag{4.6}$$

with

$$\eta_{k_1} = 1 - \alpha_{k_1,\frac{1}{2}}, \quad \alpha_{k_1,\frac{1}{2}} = \min\left(\frac{\alpha}{2\lambda}, 1\right). \tag{4.7}$$

Note that $\alpha_{k_1,\omega}$ is exactly the 1D equivalent of the capacity $\alpha_{E,\omega}$ defined in (3.1) as using $\lambda = \frac{\beta \Delta t}{h}$; we can reformulate

$$\alpha_{E,\omega} = \min\left(\omega \frac{\alpha h}{\beta \Delta t}, 1\right) = \min\left(\omega \frac{\alpha}{\lambda}, 1\right).$$

The resulting DoD stabilized scheme is then given by the following: Find $u_h \in V_h^k(\mathcal{M}_h)$ such that

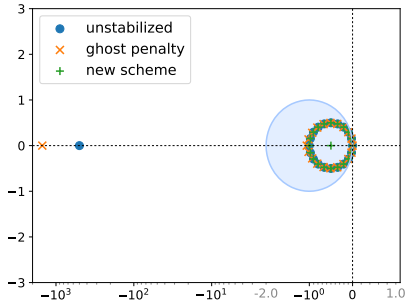
$$(d_t u_h(t), w_h)_{L^2} + a_h^{\text{upw}}(u_h(t), w_h) + J_h(u_h(t), w_h) + l_h(w_h) = 0 \quad \forall w_h \in V_h^k(\mathcal{M}_h). \tag{4.8}$$

Definition 4.1 (MP0 and MP1). *Both MP*0 *and MP*1 *refer to*

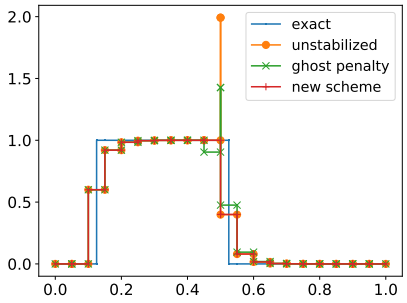
- *solving the advection equation* (4.1) *using periodic boundary conditions for the model problem MP shown in Figure* 3*;*
- *using the stabilized scheme* (4.8) *with CFL* $\lambda = \frac{\beta \Delta t}{h}$.

*By MP*0*, we refer to the case of piecewise constant polynomials and by MP*1 *to the case of piecewise linear polynomials, respectively.*

4.1. Behavior of the space discretization *without* stabilization term J_h. Before we discuss the properties of our stabilized scheme, we shortly present numerical results that illuminate the different kind of instabilities that occur if one does *not* use the stabilization term J_h.



(a) Distribution of eigenvalues for the three schemes. In blue, we depict the stability region of the explicit Euler.



(b) Solution after one time step using implicit Trapezoidal rule in time.

FIG. 4. *Numerical results for Test* 1, *comparing unstabilized, ghost penalty, and the DoD stabilized scheme.*

4.1.1. Test 1: 1D single small cut cell. We consider the model problem MP and place the cell k such that $x_{k-1/2} = 0.5$. We use discontinuous initial data

$$u_0(x) = \begin{cases} 1 & 0.1 \leq x \leq 0.5, \\ 0 & \text{otherwise.} \end{cases} \tag{4.9}$$

We set $\beta = 1$, $\alpha = 0.001$, $\lambda = 0.5$, and $h = 0.05$ and use $V_h^0(\mathcal{M}_h)$ as well as periodic boundary conditions.

We compare three scenarios: (1) We do not use stabilization, i.e., we apply (4.8) *without* the term J_h; (2) we apply the DoD stabilization J_h; (3) instead of the J_h suggested in this work, we use a straightforward adaption of the ghost penalty stabilization [13] to stabilize the problem. The ghost penalty stabilization has been used very successfully to stabilize the solution of *elliptic* problems on cut cell meshes. A first attempt to transfer the stabilization to the situation considered here would result in using a stabilization term of the form

$$\rho_1 \left[\!\left[u_h\right]\!\right]_{k-\frac{1}{2}} \left[\!\left[w_h\right]\!\right]_{k-\frac{1}{2}} + \rho_2 \left[\!\left[u_h\right]\!\right]_{\text{cut}} \left[\!\left[w_h\right]\!\right]_{\text{cut}} \tag{4.10}$$

(instead of the J_h that we suggest). We use $\rho_1 = \rho_2 = \eta_{k_1}$ (as for J_h).

We rewrite the described spatial discretization as a system of ODEs of the form $\underline{y}_t = L\underline{y}$ with a suitable matrix L. In Figure 4(a), we present the eigenvalue distribution of $\Delta t L$, where $\Delta t = \lambda h$. Most values lie in the stability region of the explicit Euler scheme. But if we do not stabilize or use the ghost penalty stabilization (4.10), there is one outlier eigenvalue (corresponding to the small cut cell), leading to stability problems. With our stabilization, all values are in the stability region of explicit Euler.

Next, we consider the usage of an implicit time stepping scheme. The result after one time step for using the implicit Trapezoidal scheme is shown in Figure 4(b). Despite the time stepping scheme being stable in an ODE stability sense, we observe a strong overshoot: Instead of staying between 0 and 1, the value on the cut cell jumps up to 2 if we do not stabilize. When using the very simple ghost penalty stabilization (4.10), the overshoot is smaller but still significant. The reason for this behavior is that the implicit Trapezoidal scheme is *not* unconditionally TVD. In fact, no second- or higher-order scheme has this property [28, 40]. As a result, if the cell fraction α

becomes too small, the scheme develops oscillations and/or overshoots unless one uses a stabilization that prohibits that behavior. Our stabilization has been designed to achieve this.

In the following, we will support these numerical results by mathematical facts: We will prove that the DoD stabilization term J_h (a) makes explicit time stepping stable again and (b) guarantees that no oscillations and/or overshoots occur.

4.2. Piecewise constant polynomials. In the special case of piecewise constant polynomials, the discrete solution u_h on cell j at time t^n, which we will denote by u_j^n, is an approximation to the average of u over cell j, and the method is equivalent to a first-order FV scheme. For MP0, the stabilization term (4.6) reduces to (with η_{k_1} given by (4.7))

$$J_h(u_h, w_h) = \beta\, \eta_{k_1}\, [u_h]_{k_1}^{\text{ext}}(x_{\text{cut}})\, [\![w_h]\!]_{\text{cut}} = \beta\, \eta_{k_1}\, [\![u_h]\!]_{k-\frac{1}{2}}\, [\![w_h]\!]_{\text{cut}}\,. \tag{4.11}$$

Here, we exploited that the solution u_h is piecewise constant.

4.2.1. Monotonicity. One very desirable property of a first-order scheme for hyperbolic conservation laws is to be monotone. This property guarantees, for example, that an overshoot as seen in Figure 4(b) cannot occur. We recall the definition of monotonicity according to [43].

DEFINITION 4.2. *A method* $u_j^{n+1} = H(u_{j-i_L}^n, u_{j-i_L+1}^n, \dots, u_{j+i_R}^n)$ *is called* monotone *if* $\forall j$ *there holds for every* l *with* $-i_L \le l \le i_R$

$$\frac{\partial H}{\partial u_{j+l}}(u_{j-i_L}, \dots, u_{j+i_R}) \ge 0. \tag{4.12}$$

For a linear scheme, this implies that all coefficients need to be positive. We will verify this property for our stabilized scheme using the concept of M-matrices.

DEFINITION 4.3 (see [38]). *Let* $B \in \mathbb{R}^{n\times n}$ *be a Z-Matrix; i.e., for* $B = (b_{ij})$, *there holds* $b_{ij} \le 0$ *for every* $i \ne j$. *If* $b_{ii} > 0\ \forall i$ *and there exists a positive diagonal matrix* D *such that* BD *is strictly diagonal dominant, we call* B *an* M-matrix.

LEMMA 4.4 (see [38]). *Let* $B \in \mathbb{R}^{n\times n}$ *be an M-matrix. Then* B^{-1} *exists, and all matrix entries of* B^{-1} *are nonnegative.*

LEMMA 4.5. *Consider MP*0. *Discretize the time using the theta scheme. Then the method is monotone under the CFL condition* $\lambda < \frac{1}{2(1-\Theta)}$.

Proof. The theta scheme for the ODE $y_t = F(y)$ is defined as

$$y^{n+1} = y^n + \Delta t \left[\Theta F(y^{n+1}) + (1-\Theta) F(y^n)\right], \tag{4.13}$$

with $\Theta \in [0,1]$ ($\Theta = 0$: explicit Euler; $\Theta = 1$: implicit Euler; $\Theta = \frac{1}{2}$: implicit Trapezoidal). Let us first assume $\alpha < 2\lambda$, which is the interesting case. Then $\alpha_{k_1,\frac{1}{2}} = \frac{\alpha}{2\lambda}$. Rewriting one time step of our stabilized discretization in matrix-vector formulation results in

$$\underbrace{(M + \Delta t\Theta(A+J))}_{=:B} u^{n+1} = \underbrace{(M - \Delta t(1-\Theta)(A+J))}_{=:C} u^n. \tag{4.14}$$

Here M denotes the mass matrix, A the stiffness matrix, and J the additional stabilization operator. Defining $\tau_1 := \Delta t\Theta\beta \ge 0$ and $\tau_2 := \Delta t\,(1-\Theta)\,\beta \ge 0$, the two

matrices are of the form (entries on the main diagonal are boxed)

$$B = \begin{pmatrix} \boxed{h+\tau_1} & 0 & \cdots & & & \cdots & 0 & -\tau_1 \\ -\tau_1 & \boxed{h+\tau_1} & \ddots & & & & & 0 \\ 0 & \ddots & \ddots & & & & & \vdots \\ \vdots & & -\tau_1\frac{\alpha}{2\lambda} & \boxed{\alpha h+\tau_1\frac{\alpha}{2\lambda}} & & & & \\ & & -\tau_1\left(1-\frac{\alpha}{2\lambda}\right) & -\tau_1\frac{\alpha}{2\lambda} & \boxed{(1-\alpha)h+\tau_1} & & & \vdots \\ \vdots & & & & \ddots & \ddots & & 0 \\ 0 & \cdots & & & \cdots & 0 & -\tau_1 & \boxed{h+\tau_1} \end{pmatrix}$$

and

$$C = \begin{pmatrix} \boxed{h-\tau_2} & 0 & \cdots & & & \cdots & 0 & \tau_2 \\ \tau_2 & \boxed{h-\tau_2} & 0 & & & & & 0 \\ 0 & \ddots & \ddots & & & & & \vdots \\ \vdots & & \tau_2\frac{\alpha}{2\lambda} & \boxed{\alpha h-\tau_2\frac{\alpha}{2\lambda}} & & & & \\ & & \tau_2\left(1-\frac{\alpha}{2\lambda}\right) & \tau_2\frac{\alpha}{2\lambda} & \boxed{(1-\alpha)h-\tau_2} & & & \vdots \\ \vdots & & & & \ddots & \ddots & & 0 \\ 0 & \cdots & & & \cdots & 0 & \tau_2 & \boxed{h-\tau_2} \end{pmatrix}.$$

According to (4.12), we need to show that $B^{-1}C$ is positive. Following Lemma 4.4, we will verify that B is an M-matrix and C is positive.

Most cells use a standard first-order upwind scheme in space and the theta scheme with standard CFL condition in time and obviously satisfy these conditions. We will focus on the rows corresponding to cells k_1 and k_2, which differ from the remaining rows due to the stabilization term J_h. We start with matrix B and the row corresponding to cell k_2. The diagonal entry $(1-\alpha)h+\tau_1$ is positive, while the remaining entries are negative since $\alpha < 2\lambda$. To prove the diagonal dominance, we compute

$$|b_{k_2,k_2}| - \sum_{j\neq k_2} |b_{k_2,j}| = (1-\alpha)\,h + \tau_1 - \left(\tau_1 - \frac{\alpha\tau_1}{2\lambda}\right) - \frac{\alpha\tau_1}{2\lambda} = (1-\alpha)h > 0.$$

Similar calculations for the row corresponding to k_1 imply that B is an M-matrix. Next we will show positivity of C, considering the CFL condition $\lambda < \frac{1}{2(1-\Theta)}$. For a standard, equidistant cell, the positivity of the entries is guaranteed already by the standard CFL condition and thus also by our slightly stricter condition. Straightforward calculations further reveal that all entries in rows k_1 and k_2 are positive. We note that in particular, the positivity of entry c_{k_2,k_2} is guaranteed by the CFL condition and the fact that a lower bound for the size of k_2 is $\frac{h}{2}$.

Finally, for $\alpha \geq 2\lambda$, there holds $\alpha_{k_1,\frac{1}{2}} = 1$, and therefore the factor $\eta_{k_1} = 1-\alpha_{k_1,\frac{1}{2}}$ in the definition of J_h evaluates to 0; i.e., we would not stabilize. But as in this case $c_{k_1,k_1} = \alpha h - \tau_2 \geq \lambda h\Theta \geq 0$, all required properties of B and C would hold true without stabilization. This concludes the proof. ∎

Remark 4.2 (ghost penalty stabilization). Using the stabilization (4.10) instead of the J_h given in (4.11) results in additional 2×2 blocks in the matrices B and C at positions $((k-1):k_1,(k-1):k_1)$, and $(k_1:k_2,k_1:k_2)$, respectively. We note that it is *not* possible to find factors ρ_1 and ρ_2 that guarantee that all entries of C

are positive for $\alpha \to 0$. This implies that for explicit Euler time stepping, it is not possible to create a monotone scheme using this stabilization (see also [42] for more detail). In contrast, our stabilizing 2×2 block has been shifted to $(k_1 : k_2, (k-1) : k_1)$. This shifted stabilization reflects the fact that for the hyperbolic equations, there is a designated direction of information transport, whereas for elliptic problems, there is not. This shifted stabilization is also a major difference to the stabilization terms used by other authors [24, 41].

4.2.2. L^1 stability and TVD stability. Next, we will show that MP0 with *explicit Euler* in time is L^1 stable as well as TVD stable under the standard CFL condition, independent of the size of α.

On standard cells away from the cut cells k_1 and k_2, the scheme given by (4.8) in combination with explicit Euler in time corresponds to the standard upwind scheme [32]. We will assume for the rest of this subsection that $\alpha < 2\lambda$. If this were not the case, we would still have a nonuniform mesh but would no longer violate the CFL condition on cell k_1 needed for L^1 and TVD stability, and the results in the following will remain valid. For $\alpha < 2\lambda$, we get the following formulae in the neighborhood of the small cut cell k_1 after a minor reordering and simplification of the terms:

$$u_{k-1}^{n+1} = u_{k-1}^n - \lambda\left(u_{k-1}^n - u_{k-2}^n\right), \tag{4.15a}$$

$$u_{k_1}^{n+1} = \frac{1}{2}u_{k_1}^n + \frac{1}{2}u_{k-1}^n, \tag{4.15b}$$

$$u_{k_2}^{n+1} = \left(1 - \frac{\lambda}{1-\alpha}\right)u_{k_2}^n + \frac{\alpha}{2(1-\alpha)}u_{k_1}^n + \frac{\lambda - \frac{\alpha}{2}}{1-\alpha}u_{k-1}^n, \tag{4.15c}$$

$$u_{k+1}^{n+1} = u_{k+1}^n - \lambda\left(u_{k+1}^n - u_{k_2}^n\right). \tag{4.15d}$$

LEMMA 4.6. *Consider MP0 with explicit Euler in time. Then, for $\lambda < \frac{1}{2}$, there holds*

$$\left\|u^{n+1}\right\|_{L^1} \le \|u^n\|_{L^1} \quad \forall n \ge 0.$$

We note that the required CFL condition is *independent* of the size of α (but takes into account that the bigger cut cell k_2 is not stabilized).

Proof. We define

$$\sum_j |u_j^{n+1}|h_j = \underbrace{\sum_{j\le k-1} |u_j^{n+1}|h}_{T_1} + \underbrace{|u_{k_1}^{n+1}|\alpha h}_{T_2} + \underbrace{|u_{k_2}^{n+1}|(1-\alpha)h}_{T_3} + \underbrace{\sum_{j\ge k+1} |u_j^{n+1}|h}_{T_4}.$$

On all cells except for cells k_1 and k_2, the standard upwind scheme is used (compare also (4.15a) and (4.15d)). Plugging in the corresponding formulae and using $\lambda > 0$ as well as $1 - \lambda > 0$ gives

$$T_1 \le \sum_{j\le k-2} |u_j^n|h + (1-\lambda)|u_{k-1}^n|h, \qquad T_4 \le \sum_{j\ge k+1} |u_j^n|h + \lambda h|u_{k_2}^n|.$$

For cells k_1 and k_2, we directly obtain from (4.15b) and (4.15c) (using $1 - \alpha - \lambda \ge 0$ and $2\lambda - \alpha \ge 0$)

$$T_2 \le \frac{\alpha h}{2}\left(|u_{k_1}^n| + |u_{k-1}^n|\right), \quad T_3 \le (1-\alpha-\lambda)\,h|u_{k_2}^n| + \frac{\alpha h}{2}|u_{k_1}^n| + \left(\lambda - \frac{\alpha}{2}\right)h|u_{k-1}^n|.$$

Summing up the estimates for $T_1, \ldots, T_4$ implies the claim. ∎

DEFINITION 4.7 (see [17]). *A DG scheme is called* TVD in the means *(TVDM) if for all* $n \geq 0$

$$TV(\overline{u}^{n+1}) \leq TV(\overline{u}^n) \quad \text{with } TV(\overline{u}^n) = \sum_j |\overline{u}^n_{j+1} - \overline{u}^n_j|. \tag{4.16}$$

Here, $\overline{u}^n_j$ *denotes the mean of* u *on cell* j *at time* t_n.

For piecewise constant polynomials, the means $\overline{u}^n_j$ correspond to the unknowns u^n_j, and TVDM coincides with the TVD [32] property. (For later use for piecewise *linear* polynomials, we provide the more general definition here.)

LEMMA 4.8. *Consider MP*0 *with explicit Euler in time. Then the scheme is TVD stable for* $\lambda < \frac{1}{2}$.

Proof. We decompose

$$\sum_j |u^{n+1}_j - u^{n+1}_{j-1}| = \underbrace{\sum_{j\leq k-1} |u^{n+1}_j - u^{n+1}_{j-1}|}_{T_1} + \underbrace{|u^{n+1}_{k_1} - u^{n+1}_{k-1}|}_{T_2} + \underbrace{|u^{n+1}_{k_2} - u^{n+1}_{k_1}|}_{T_3}$$
$$+ \underbrace{|u^{n+1}_{k+1} - u^{n+1}_{k_2}|}_{T_4} + \underbrace{\sum_{j\geq k+2} |u^{n+1}_j - u^{n+1}_{j-1}|}_{T_5}.$$

For the standard parts of the scheme, we get

$$T_1 \leq \sum_{j\leq k-2} |u^n_j - u^n_{j-1}| + (1-\lambda)|u^n_{k-1} - u^n_{k-2}|, \quad T_5 \leq \sum_{j\geq k+2} |u^n_j - u^n_{j-1}| + \lambda|u^n_{k+1} - u^n_{k_2}|.$$

Using (4.15a) and (4.15b), a direct substitution of the formulae results in

$$T_2 \leq \frac{1}{2}|u^n_{k_1} - u^n_{k-1}| + \lambda|u^n_{k-1} - u^n_{k-2}|.$$

For T_3 and T_4, we reorder the terms resulting from the definitions (4.15b)–(4.15d) to get

$$T_3 \leq \left(1 - \frac{\lambda}{1-\alpha}\right)|u^n_{k_2} - u^n_{k_1}| + \left(\frac{1}{2} - \frac{2\lambda-\alpha}{2(1-\alpha)}\right)|u^n_{k_1} - u^n_{k-1}|,$$
$$T_4 \leq (1-\lambda)\,|u^n_{k+1} - u^n_{k_2}| + \frac{\lambda}{1-\alpha}|u^n_{k_2} - u^n_{k_1}| + \frac{2\lambda-\alpha}{2(1-\alpha)}|u^n_{k_1} - u^n_{k-1}|.$$

We emphasize that all factors outside of the absolute values are positive due to the made assumptions. Summing up the estimates for $T_1, \ldots, T_5$ implies the claim. ∎

Remark 4.3. On an *equidistant* mesh, a standard FV scheme that is monotone is automatically TVD [27]. As the standard proof for this result does not transfer to our situation, we provide here the proof for TVD stability as well.

4.2.3. Comparison to the *h*-box method. The idea behind the proposed stabilization is similar to that of the h-box method [9, 10, 26]. There, one creates boxes of length h at the cut cell edges to reconstruct the appropriate domains of dependence for small cut cells and their outflow neighbors. These boxes are then used for the flux computation on cut cell edges.

We compare the FV-based h-box method to our DG stabilization for piecewise constant polynomials and explicit Euler in time. We consider a 1D mesh (which is

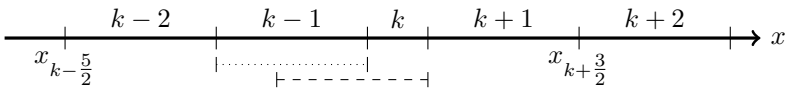


FIG. 5. *1D FV model mesh and boxes used for the h-box stabilization.*

often used as cut model for FV schemes) of width h with one small cell of length αh, $\alpha \in (0, 1]$, referred to as cell k, in the middle (see Figure 5). We solve the advection equation (4.1) with $\beta = 1$ using upwind fluxes. Then, for both methods, the update formulae on cells $i \leq k-1$ and $i \geq k+2$ correspond to the standard upwind scheme.

Using the h-box method on the cut cell k results in the general update formula

$$u_k^{n+1} = u_k^n - \frac{\Delta t}{\alpha h}\left(u_{k+\frac{1}{2}}^{n,L} - u_{k-\frac{1}{2}}^{n,L}\right) \tag{4.17}$$

with upwind fluxes

$$u_{k-\frac{1}{2}}^{n,L} = u_{k-1}^n \qquad \text{and} \qquad u_{k+\frac{1}{2}}^{n,L} = \hat{\eta} u_k^n + (1-\hat{\eta}) u_{k-1}^n.$$

Here, $u_{k+\frac{1}{2}}^{n,L}$ is constructed using linear interpolation with a parameter $\hat{\eta}$. Therefore,

$$u_k^{n+1} = u_k^n - \frac{\Delta t \hat{\eta}}{\alpha h}\left(u_k^n - u_{k-1}^n\right). \tag{4.18}$$

The DoD stabilization (4.11) for this setting is given by

$$u_k^{n+1} = u_k^n - \frac{\Delta t (1-\eta)}{\alpha h}\left(u_k^n - u_{k-1}^n\right). \tag{4.19}$$

Substituting $\hat{\eta} = 1 - \eta$, (4.18) and (4.19) are identical. The same can be shown for the update of the cell $k+1$. In other words, the two methods coincide up to the choice of the parameter.

In [10], two values for η are proposed: $\eta_1 = 1 - \alpha$ and $\eta_2 = 1 - \frac{2\alpha}{1+\alpha}$. The choice η_1 is based on cell averaging and the choice η_2 on optimizing the one step error: It is the only choice to lead to a second-order one-step error (compared to first order otherwise). We propose $\eta_4 = 1 - \frac{\alpha}{\lambda}$ and $\eta_3 = 1 - \frac{\alpha}{2\lambda}$. The choice η_4 results in exact advection of a piecewise constant solution in the situation above. The choice η_3 is motivated by its significantly better performance for piecewise linear polynomials compared to η_4.

All four choices yield similar monotonicity and TVD results. Also, numerical results are comparable, and for smooth solutions, the errors at a fixed time T are almost identical. The apparent benefit of η_2 regarding the one-step error does not impact the accuracy at time T. For η_1, [10] shows global first-order convergence, and a variant of the proof can be shown for our suggestion $\eta_3 = 1 - \frac{\alpha}{2\lambda}$.

PROPOSITION 4.9. *We consider the situation described above (mesh shown in Figure 5; upwind scheme on all cells stabilized by the DoD stabilization on cell k). Then, using $\eta = \eta_3$, the scheme is of first order for sufficiently smooth solutions.*

Proof. The proof is based on the ideas of Wendroff and White [44] and follows step-by-step the proof of [10, Proposition 1] using the grid function

$$w_k^n = u_k^n + \left(2\lambda - \frac{1}{2}(1+\alpha)\right) h u_x(x_k, t^n) \tag{4.20}$$

on the small cut cell k. On all other cells, one uses $w_i^n = u_i^n$. ∎

Remark 4.4. The result of Proposition 4.9 can also be shown for the model mesh shown in Figure 3, using the additional modification of the grid function on cell k_2:

$$w_{k_2}^n = u_{k_2}^n - \frac{\alpha}{2} h u_x(x_{k_2}, t^n).$$

Using the four different values of η in the DoD stabilization for piecewise linear polynomials, we observe numerically that η_3 does best, even though the results for η_2 are very close.

Despite the h-box method and the DoD stabilization being almost identical for this simple model setup, there are substantial differences for the extension of the method to 2D and to second order. One major difference is, for example, the integral term (3.4) showing up in the DoD stabilization.

4.3. Piecewise linear polynomials. In this subsection, we examine MP1 more closely. In this case, the stabilization term J_h is given by (4.6). For simplicity, we will assume in this subsection that we use a centered, rescaled moment basis; see also section 5.1. In time, we will consider the second-order explicit SSP-RK scheme given by (2.9).

4.3.1. Eigenvalue analysis. We now want to motivate the stability of our stabilized scheme in combination with the second-order explicit RK scheme (2.9) by means of an eigenvalue analysis using symbolic computations with `sympy` [35]. We consider the model problem MP for the special case of $N = 5$; i.e., we start with four equidistant cells of length h and split the third cell into two cells of length αh and length $(1-\alpha)h$. We use Dirichlet boundary conditions, and without loss of generality, $\beta = 1$.

We consider the stabilized scheme (4.8) with the stabilization term J_h given by (4.6). For the construction of the stabilization term J_h, we have made several design choices, e.g., the choices of ω and η_{k_1} and the general structure of the terms. We rewrite the variational formulation (4.8) as a system of ODEs with M being the mass matrix, A being the stiffness matrix incorporating $a_h^{\mathrm{upw}}(\cdot,\cdot)$, and J incorporating the corresponding parts of $J_h(\cdot,\cdot)$. We then symbolically set up the operator

$$R = M^{-1}(M - \Delta t(A + J)) - \mathrm{Id}, \tag{4.21}$$

which is the stability function used in the ODE stability analysis. As $\alpha \leq \frac{1}{2}$, a lower bound for the size of the unstabilized cut cell k_2 is $\frac{h}{2}$. The limiting CFL number λ for piecewise linear polynomials is then $\lambda = \frac{1}{6}$. We therefore choose $\Delta t = \frac{h}{6}$ to compute the eigenvalues of R. We find three pairs of complex eigenvalues:

$$\lambda_{1,2} = -\frac{2 \pm \sqrt{2}i}{6},$$
$$\lambda_{3,4} = \frac{2 \pm \sqrt{2}i}{6\,(\alpha - 1)},$$
$$\lambda_{5,6} = -\frac{2 \pm \sqrt{2}i}{2}.$$

All eigenvalues are stable: $\lambda_{1,2}$ and $\lambda_{5,6}$ are independent of α, and $\lambda_{3,4}$ converge to $\lambda_{1,2}$ for $\alpha \to 0$.

For a sequence of decreasing α, we visualize the eigenvalues of R and observe that for $\Delta t = \frac{h}{6}$, all eigenvalues stay within the stability region of the explicit second-order SSP-RK scheme (2.9) (Figure 6), indicating the stability of the explicit time stepping scheme.

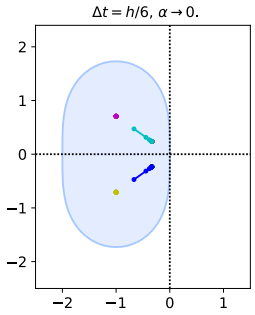


FIG. 6. *Eigenvalues of* (4.21)*: The stability region of the explicit RK scheme is shown in blue. The colors refer to the different eigenvalues and show their evolution for decreasing* α *(*$\alpha = 2^{-i}$*,* $i \in [1, \dots, 10]$*).*

4.3.2. TVDM stability. We can only expect to obtain a TVDM stability result for piecewise linear polynomials if we apply a limiter. Different than 2D, in 1D we only reconstruct to cell faces instead of to neighboring centroids, which results in an MC-like limiter [32]. (In 2D, depending on the mesh, this could result in a limiter that is not linearity preserving.) By assumption, our solution u_j^* on cell j resulting from solving (4.8) with the time stepping scheme (2.9), where $(\cdot)^*$ stands for $(\cdot)^{(1)}$ or $(\cdot)^{n+1}$, uses the centered moment basis

$$\bar{u}_j^* + \nabla u_j^*(x - x_j). \tag{4.22}$$

Then we enforce

$$\nabla u_j^* = \text{minmod}\left(\nabla u_j^*, \frac{\bar{u}_{j+1}^* - \bar{u}_j^*}{x_{j+1/2} - x_j}, \frac{\bar{u}_j^* - \bar{u}_{j-1}^*}{x_j - x_{j-1/2}}\right) \tag{4.23}$$

with

$$\text{minmod}\,(a_1, \dots, a_m) = \begin{cases} s \min_{1 \le i \le m} |a_i| & \text{if } |s| = 1, s = \frac{1}{m}\sum_{i=1}^m \text{sgn}\,(a_i), \\ 0 & \text{otherwise.} \end{cases}$$

LEMMA 4.10. *Consider MP*1 *with explicit Euler in time. Assume that the moment basis* (4.22) *and limiter* (4.23) *is used, which has been modified on cell* $k-1$ *to additionally enforce*

$$\min\left(\bar{u}_{k-1}^n, \bar{u}_{k_1}^n\right) \le u_{h,k-1}(x_{cut}) \le \max\left(\bar{u}_{k-1}^n, \bar{u}_{k_1}^n\right). \tag{4.24}$$

Then, for $\lambda < \frac{1}{4}$ *the scheme is TVDM stable.*

Proof. The proof follows that of lemma 4.8. Details are given in appendix A. ∎

COROLLARY 4.11. *Consider MP*1 *with the second-order SSP-RK scheme* (2.9) *in time. Assume that the moment basis* (4.22) *and limiter* (4.23) *is used, which has been modified on cell* $k-1$ *to additionally enforce* (4.24)*. Then, for* $\lambda < \frac{1}{4}$*, the scheme is TVDM stable.*

Proof. The result follows directly from the fact that SSP-RK schemes are constructed to be convex combinations of explicit Euler steps [23]. ∎

5. Implementation. The implementation is based on the DUNE [4, 5] framework, a feature-rich C++ finite element library. For the cut cell discretization, we rely on the `dune-udg` package [21] and its integration with `dune-pdelab` [8]. The `dune-udg` module was originally developed for the unfitted DG method [6]. We modified it to realize the extended stencil of the DoD stabilization terms J_h^0 and J_h^1.

The domain is represented as a discrete level set function that uses vertex values, i.e., as a bilinear finite element function. Given this representation, cut cells and their corresponding quadrature rules are constructed using the TPMC library [22]. We note that this setup results in a restriction of the geometry that can be realized: It only allows for a polygonal representation of the domain, and for the numerical tests, we cannot resolve smooth geometries exactly.

5.1. Construction of local shape functions. Although in principle the choice of the local basis should not have an impact on the method itself, there are practical aspects to consider. In particular the implementation of the limiter becomes significantly simpler if one can separate the average value in the cell from the local fluctuations. Thus, we choose to use a moment basis. In 1D, we have chosen the local basis Φ_j in cell j (in global coordinates) as

$$\{1, (x - x_j)/h_j\}, \tag{5.1}$$

with h_j denoting the cell length of cell j. We also use this representation on cut cells. This allows modifying the gradient without changing the average mass in the cell.

In 2D, we do not give an explicit formula, but we again construct the basis via moments in such a way; i.e., we split into a constant function and linear functions with average zero. The linear functions are centered around the center of mass of the cut cell, but in contrast to the 1D case, we do not do the additional rescaling, so that the gradients are the same as on the background mesh.

6. Numerical results. In this section, we support our theoretical findings by numerical experiments in 1D and 2D. For most cases, we will use piecewise linear polynomials. We remind the reader that V_h^0 is the function space of piecewise constant polynomials and that V_h^1 is the function space of piecewise linear polynomials.

6.1. Numerical results for the 1D case. For the numerical results in 1D, we solve (4.8) with the stabilization term J_h defined in (4.6) using $V_h^1(\mathcal{M}_h)$.

6.1.1. Test 2: 1D smooth initial data. We consider a variant of MP1 by changing the model problem MP slightly: We split all cells between $x = 0.1$ and $x = 0.9$ in cut cell pairs of length $\alpha_k h$ and $(1 - \alpha_k)h$ and compare three different scenarios:

1. **S1:** the fraction α_k is the same for all cells, i.e., $\alpha_k = \alpha$ ("$\alpha = 10^{-\square}$");
2. **S2:** the cell fraction α_k varies and is computed as $\alpha_k = 0.1X_k$ with X_k being a uniformly distributed random number in $(0, 1)$ ("random α");
3. **S3:** reference solution without any cut cells ("equidistant").

We use smooth initial data $u_0(x) = \sin(2\pi x)$, set $\beta = 1$, $\lambda = \frac{1}{6}$, and use the time stepping scheme (2.9).

In Figure 7, the error at time $T = 1$, measured in the L^1 and L^∞ norm, are shown. We observe second-order convergence for all tested scenarios. In particular, the errors for pairs with $\alpha = 0.1$ and $\alpha = 10^{-4}$ are almost the same and fairly comparable to the case of not cutting the cells.

Figure 8 (left) shows the solution at time $T = 1$ for S2 without and with using the limiter described in Lemma 4.10 for a coarse mesh with $h = 0.05$. Without limiter,

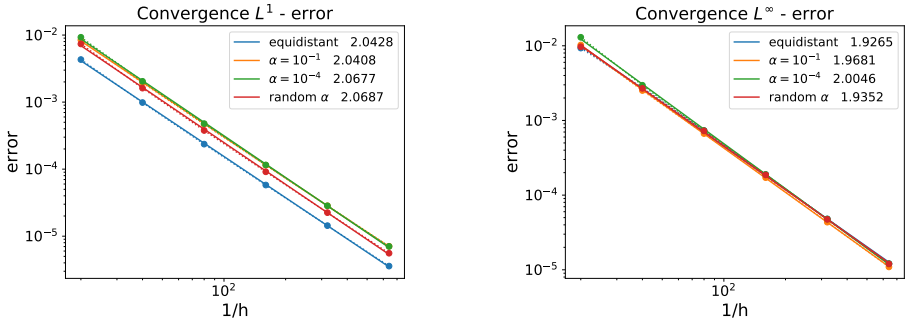


FIG. 7. *Test 2 (1D, smooth initial data): optimal (second order) convergence rates for the error at end time* $T = 1$.

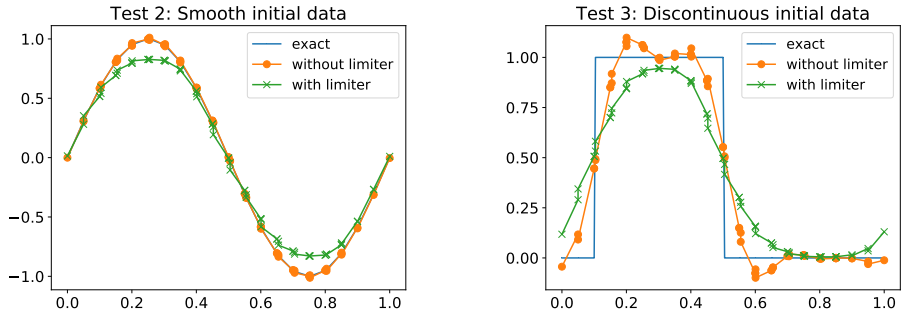


FIG. 8. *Solutions of Test 2 and 3 at* $T = 1$ *for* $h = 0.05$ *with and without limiter.*

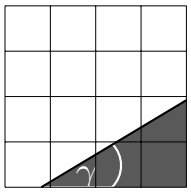


FIG. 9. *Ramp geometry.*

the solution matches very well with the exact solution. With limiter, we observe the expected peak clipping, but there is no form of staircasing or other unwanted interaction between the limiter and the gradient stabilization.

6.1.2. Test 3: 1D discontinuous initial data. We use the setup of Test 2 with the scenario S2 but use discontinuous initial data u_0 given by (4.9). In Figure 8 (right) we show the solution at time $T = 1$ for $h = 0.05$ with and without limiter. As expected, the solution does not show overshoots when the limiter is applied.

Additionally, we measure $\mathrm{TV}(\overline{u}^n)$, defined in (4.16), and observe that it is non-increasing during the simulation if the limiter is used. We note that this is not the case if we do not apply the additional condition (4.24) for limiting on the inflow neighbor of a small cut cell of type $k_1 \in \mathcal{I}$.

6.2. Numerical results for the 2D case. For the numerical experiments in 2D we focus on the geometry of a ramp with angle γ, see Figure 9: we use $\widehat{\Omega} = (0,1)^2$ and discretize it with $N \times N$ Cartesian cells, resulting in the mesh width $h = \frac{1}{N}$.

Afterwards, we cut out a ramp at $x = 0.2001$ with angle γ with a straight intersection, resulting in the computational domain Ω. Our initial data u_0 are defined with respect to a standard Cartesian coordinate system (x, y) and then transformed to our rotated and shifted coordinate system $(\hat{x}, \hat{y})$ by using the transformation

$$\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = \begin{pmatrix} \cos\gamma & \sin\gamma \\ -\sin\gamma & \cos\gamma \end{pmatrix} \cdot \begin{pmatrix} x - 0.2001 \\ y \end{pmatrix}. \tag{6.1}$$

The cut cells are located along the ramp and have various shapes and sizes. In our experiments, the smallest volume fractions of cut cells E, computed as $\frac{|E|}{h^2}$, varied typically between 10^{-5} and 10^{-8}. In this particular setup, we used a slightly simplified definition of $\mathcal{I} = \{E \in \mathcal{M}_h \mid \frac{|E|}{h^2} < 0.1\}$, which corresponds to a superset of the set $\mathcal{I}$ defined in (3.2). Using this definition of $\mathcal{I}$ for $5° \leq \gamma \leq 45°$, the shortest hypotenuse of a triangle cut cell that is *not* stabilized corresponds to roughly $0.6h$.

When choosing the time step size Δt, we need to take into account that the step size needs to be appropriate for $V_h^k(\mathcal{M}_h)$, $k \in \{0, 1\}$, for both Cartesian cells of size h^2 and cut cells E that are not stabilized, i.e., for cut cells $E \notin \mathcal{I}$. Finding a tight bound for the latter category is nontrivial and a project in itself. We have chosen the constraint

$$\Delta t \leq 0.6 \frac{1}{2k+1} \frac{0.5\, h}{\max_{ij} \|\beta_{ij}\|}, \tag{6.2}$$

which has worked well in our numerical experiments and which is in good agreement with a recently suggested CFL condition for DG schemes on triangular meshes [15].

For β, we choose

$$\beta^V(x, y) = \frac{1}{2\sqrt{1 + \tan^2(\gamma)}} \begin{pmatrix} 1 \\ \tan(\gamma) \end{pmatrix} (2 + (x - 0.2001)\sin\gamma + y\cos(\gamma + \pi)). \tag{6.3}$$

This incompressible velocity field transports the mass parallel to the ramp with decreasing speed for increasing distance to the ramp. For comparison, we also use the constant velocity field

$$\beta^C(x, y) = \frac{2}{\sqrt{1 + \tan^2(\gamma)}} \begin{pmatrix} 1 \\ \tan(\gamma) \end{pmatrix}. \tag{6.4}$$

6.2.1. Test 4: 2D smooth initial data. We use the velocity field β^V. We choose smooth initial data

$$u_0(\hat{x}, \hat{y}) = \sin\left(\frac{\sqrt{2}\pi\hat{x}}{1 - 0.2001} \right),$$

and compute the solution at time T = 0.5 using $V_h^1(\mathcal{M}_h)$.

Figure 10 shows the error at time T in the L^1 and L^∞ norm. We observe second-order convergence in the L^1 norm. Due to the irregularity of the cut cells, the L^∞ error is *not smooth*, indicated by the zigzag behavior of the plotted results. We therefore use a least squares fit to compute the convergence rates. The rates vary, depending on the angle, and lie between 1.52 and 1.63. This slightly reduced order of convergence will need to be examined in more detail in the future. In general, it is challenging to achieve full second-order accuracy on cut cell meshes in 2D. The one step error on cut

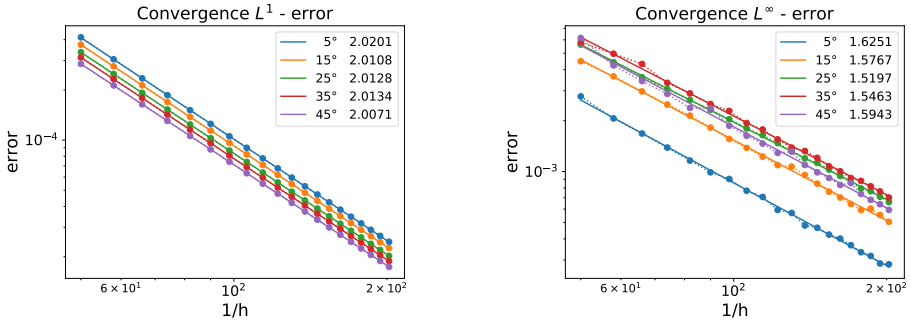


FIG. 10. *Test 4 (2D, smooth initial data): L^1 and L^∞ convergence rates for the error at end time T for various angles γ. Reduced L^∞ order, compared to the 1D tests.*

TABLE 1
Convergence rates (L^1 and L^∞ norm) for Test 4 for constant (β^C) and varying (β^V) velocity field.

Angle γ:		5°	10°	15°	20°	25°	30°	35°	40°	45°
β^C	L^1	2.02	2.01	2.01	2.00	2.00	2.01	2.01	2.01	2.02
	L^∞	1.88	1.68	1.63	1.60	1.60	1.56	1.54	1.53	1.58
β^V	L^1	2.02	2.01	2.01	2.01	2.01	2.01	2.01	2.01	2.00
	L^∞	1.62	1.60	1.57	1.55	1.51	1.55	1.54	1.53	1.59

cell meshes is typically one order lower than on structured meshes. In 1D, it is often observed that these additional error sources do not accumulate; compare also section 4.2.3. In 2D, error accumulation is barely understood.

For comparison, we show in Table 1 the convergence rates for additional angles γ as well as for using the constant velocity field β^C. The results for the constant velocity β^C are very similar to the ones for β^V, except for 5°. We note that it is reasonable that the results for a 5° ramp are better, as most cut cells have full length in the flow direction; i.e., such a ramp contains significantly fewer "problematic" cut cells than a ramp with a higher angle, and the sizes of neighboring cut cells do not differ as strongly.

Finally, in Figure 11 (left column), we show the solution for β^V and $\gamma = 30°$ on a coarse mesh of $N = 30$. The 1D profile along the cut boundary shows the expected sine curve. For the contour plot, we observe that the contour lines are straight lines all the way to the boundary. We note that for the initial data u_0, the contour lines are perpendicular to the ramp. Due to β^V having decreasing speed for increasing distance to the ramp, the lines have been rotated during the simulation.

6.2.2. Test 5: 2D discontinuous initial data. We use the velocity field β^V and compute until time $T = 0.4$ for discontinuous initial data

$$u_0(\hat{x}, \hat{y}) = \begin{cases} 1 & \text{for } \hat{x} < \frac{4}{15}, \\ 0 & \text{otherwise.} \end{cases}$$

Figure 11 (columns 2–4) shows the discrete solution for $N = 30$ for $\gamma = 30°$ for both $V_h^0(\mathcal{M}_h)$ and $V_h^1(\mathcal{M}_h)$, the latter without and with limiter. For both $V_h^0(\mathcal{M}_h)$ and $V_h^1(\mathcal{M}_h)$ with limiter, the discrete solution stays between 0 and 1, and we do not observe overshoots. This is also indicated by the 1D line plots along the cut

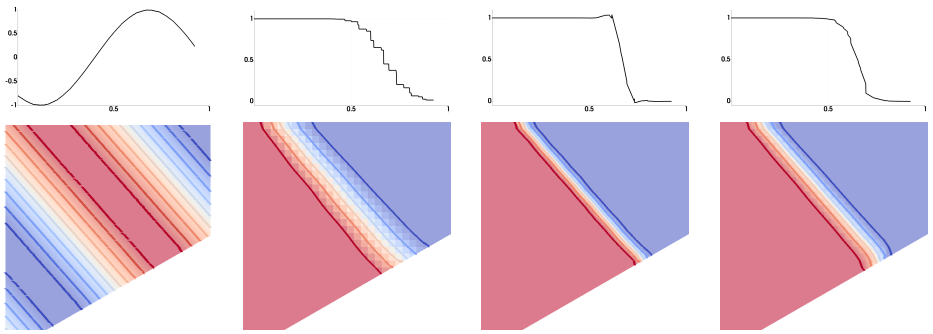


FIG. 11. *Solutions of Test* 4 *(column* 1*) and Test* 5 *(columns* 2–4*) at time* $T = 0.4$*, with* $N = 30$ *for four different scenarios. Straight contour lines in columns* 2 *and* 3 *indicate that our stabilization does not add extra diffusion on the cut cells. The upper row plots show* u *along the cut boundary.*

boundary. Further, we verified numerically for $V_h^0(\mathcal{M}_h)$ that all matrix indices in the matrix $B^{-1}C$ (compare (4.14) for the 1D version) are positive.

Examining the contour lines more closely, we find that for $V_h^0(\mathcal{M}_h)$ and $V_h^1(\mathcal{M}_h)$ without limiter, the contour lines are straight lines all the way to the boundary, indicating that our stabilization does not add extra diffusion on the cut cells. For the case of $V_h^1(\mathcal{M}_h)$ with limiter, we observe slightly more diffusion along the boundary. We attribute this to the limiter: On a Cartesian cell, we only limit in x- and y-directions. On cut cells, however, we also limit roughly in advection direction when reconstructing to neighboring centroids. We expect this behavior to improve if a more accurate limiter, e.g., the LP limiter [33], is used, but this is not the focus of this work.

7. Conclusions and outlook. We have presented a new stabilization for DG schemes for linear scalar conservation laws on cut cell meshes that solves the small cell problem and makes explicit time stepping stable again. Our stabilization is designed to only let a certain portion of the inflow of a small cut cell stay in that small cut cell and to transport the remaining portion directly into the cut cell's outflow neighbors. As a by-product, we reconstruct the proper domain of dependence of the small cut cell's outflow neighbors. In that sense, our stabilization relies on similar ideas as the h-box method [9, 26] but without an explicit geometry reconstruction.

The approach for realizing these ideas in a DG setting was inspired by the ghost penalty method [13], but significant changes were necessary to adjust the terms that were developed for elliptic problems to the setting of hyperbolic equations. In this contribution, we have focused on using two standard explicit time stepping schemes (the explicit Euler scheme and the standard second-order SSP-RK scheme) and the standard Barth–Jespersen limiter, but there are no explicit obstacles for working with other choices of time stepping schemes and limiters.

Our stabilization ensures conservation (in a slightly extended meaning). In one dimension, we have shown that the stabilized scheme is monotone for piecewise constant polynomials and TVD in the means for piecewise linear polynomials in combination with explicit time stepping schemes. In numerical tests, we observed optimal convergence rates in the L^1 and L^∞ norm for 1D. In 2D, the numerical results in the L^1 norm showed again full second-order convergence, but the convergence rate in the L^∞ norm was slightly reduced. This will be examined in more detail in future work. In

this contribution, we focused on 1D and 2D. We believe that the proposed scheme can be extended to 3D in a fairly straightforward way. The corresponding implementation will surely be more involved, though. We also plan to extend the stabilization to higher-order polynomial degrees and to nonlinear conservation laws.

Appendix A. Proof of TVDM property.

Proof of Lemma 4.10. The structure follows that of the proof of Lemma 4.8. We will show that

$$\textstyle\sum_j |\bar{u}_j^{n+1} - \bar{u}_{j-1}^{n+1}| \le \sum_j |\bar{u}_j^n - \bar{u}_{j-1}^n| \qquad \forall n \ge 0 \text{ and } \forall j.$$

We exploit the fact that a moment basis is used. Testing with $w_h = \chi_j$ yields the update for $\bar{u}_j$. For the update of the gradient, it is sufficient to consider the information that the postprocessing by applying the limiter provides. Again it is sufficient to consider the case $\alpha < 2\lambda$.

In the following, $u_j(x)$ denotes the full (linear) solution of cell j evaluated at x, which can be written as $u_j(x) = \bar{u}_j + (x - x_j)\nabla u_j$. We first provide the update formulae in the neighborhood of the cut cell k_1 induced by the stabilization (4.8):

$$\bar{u}_{k-1}^{n+1} = \bar{u}_{k-1}^n - \lambda \left(u_{k-1}^n(x_{k-\frac{1}{2}}) - u_{k-2}^n(x_{k-\frac{3}{2}}) \right), \tag{A.1a}$$

$$\bar{u}_{k_1}^{n+1} = \bar{u}_{k_1}^n - \tfrac{1}{2} \left(u_{k_1}^n(x_{\text{cut}}) - u_{k-1}^n(x_{\text{cut}}) \right) - \lambda h \nabla u_{k-1}^n, \tag{A.1b}$$

$$\bar{u}_{k_2}^{n+1} = \bar{u}_{k_2}^n - \frac{\lambda}{1-\alpha} \left(u_{k_2}^n(x_{k+\frac{1}{2}}) - \frac{\alpha}{2\lambda} u_{k_1}^n(x_{\text{cut}}) - \left(1 - \frac{\alpha}{2\lambda}\right) u_{k-1}^n(x_{\text{cut}}) \right). \tag{A.1c}$$

We decompose the sum of the TV in the means at time t^{n+1} in terms T_1–T_5:

$$\begin{aligned} \textstyle\sum_j |\bar{u}_j^{n+1} - \bar{u}_{j-1}^{n+1}| = & \underbrace{\textstyle\sum_{j \le k-1} |\bar{u}_j^{n+1} - \bar{u}_{j-1}^{n+1}|}_{T_1} + \underbrace{|\bar{u}_{k_1}^{n+1} - \bar{u}_{k-1}^{n+1}|}_{T_2} + \underbrace{|\bar{u}_{k_2}^{n+1} - \bar{u}_{k_1}^{n+1}|}_{T_3} \\ & + \underbrace{|\bar{u}_{k+1}^{n+1} - \bar{u}_{k_2}^{n+1}|}_{T_4} + \underbrace{\textstyle\sum_{j \ge k+2} |\bar{u}_j^{n+1} - \bar{u}_{j-1}^{n+1}|}_{T_5}. \end{aligned}$$

In the following, we will estimate each of the terms T_1–T_5 separately. We will use that the MC limiter guarantees

$$0 \le \frac{\nabla u_j}{\frac{2}{h_j}(\bar{u}_{j+1}^n - \bar{u}_j^n)} \le 1, \quad 0 \le \frac{\nabla u_j}{\frac{2}{h_j}(\bar{u}_j^n - \bar{u}_{j-1}^n)} \le 1, \tag{A.2}$$

which implies

$$\frac{u_j^n(x_{j+\frac{1}{2}}) - u_{j-1}^n(x_{j-\frac{1}{2}})}{\bar{u}_j^n - \bar{u}_{j-1}^n} = 1 + \frac{\frac{h_j}{2}\nabla u_j}{\bar{u}_j^n - \bar{u}_{j-1}^n} - \frac{\frac{h_{j-1}}{2}\nabla u_{j-1}}{\bar{u}_j^n - \bar{u}_{j-1}^n} \quad \in [0, 2]. \tag{A.3}$$

Using this property and $\lambda < \frac{1}{2}$ (to guarantee the positivity of the prefactors), there holds for two equidistant cells away from cells k_1 and k_2

$$\begin{aligned} \left|\bar{u}_j^{n+1} - \bar{u}_{j-1}^{n+1}\right| \le & \left(1 - \lambda \frac{u_j^n(x_{j+\frac{1}{2}}) - u_{j-1}^n(x_{j-\frac{1}{2}})}{\bar{u}_j^n - \bar{u}_{j-1}^n}\right) \left|\bar{u}_j^n - \bar{u}_{j-1}^n\right| \\ & + \lambda \frac{u_{j-1}^n(x_{j-\frac{1}{2}}) - u_{j-2}^n(x_{j-\frac{3}{2}})}{\bar{u}_{j-1}^n - \bar{u}_{j-2}^n} \left|\bar{u}_{j-1}^n - \bar{u}_{j-2}^n\right|. \end{aligned}$$

This yields bounds for T_1 and T_5:

$$T_1 \leq \sum_{j\leq k-2} \left|\bar{u}_j^n - \bar{u}_{j-1}^n\right| + \left(1 - \lambda \frac{u_{k-1}^n(x_{k-\frac{1}{2}}) - u_{k-2}^n(x_{k-\frac{3}{2}})}{\bar{u}_{k-1}^n - \bar{u}_{k-2}^n}\right) \left|\bar{u}_{k-1}^n - \bar{u}_{k-2}^n\right|,$$

$$T_5 \leq \sum_{j\geq k+2} \left|\bar{u}_j^n - \bar{u}_{j-1}^n\right| + \lambda \frac{u_{k+1}^n(x_{k+\frac{3}{2}}) - u_{k_2}(x_{k+\frac{1}{2}}))}{\bar{u}_{k+1}^n - \bar{u}_{k_2}^n} \left|\bar{u}_{k+1}^n - \bar{u}_{k_2}^n\right|.$$

For T_2, we can show using (A.1a) and (A.1b)

$$T_2 \leq \left(1 - \frac{1}{2}\frac{u_{k_1}^n(x_{\text{cut}}) - u_{k-1}^n(x_{\text{cut}})}{\bar{u}_{k_1}^n - \bar{u}_{k-1}^n} - \frac{\lambda h \nabla u_{k-1}}{\bar{u}_{k_1}^n - \bar{u}_{k-1}^n}\right) \left|\bar{u}_{k_1}^n - \bar{u}_{k-1}^n\right| + \lambda \frac{u_{k-1}^n(x_{k-\frac{1}{2}}) - u_{k-2}^n(x_{k-\frac{3}{2}})}{\bar{u}_{k-1}^n - \bar{u}_{k-2}^n} \left|\bar{u}_{k-1}^n - \bar{u}_{k-2}^n\right|.$$

For this, we need to verify that the two prefactors are positive and that it is therefore allowed to pull them out of the absolute value. The prefactor of the second term is positive due to (A.3). For the first prefactor, we obtain using (4.24)

$$\frac{u_{k_1}^n(x_{\text{cut}}) - u_{k-1}^n(x_{\text{cut}})}{\bar{u}_{k_1}^n - \bar{u}_{k-1}^n} = 1 + \frac{\frac{\alpha h}{2}\nabla u_{k_1}}{\bar{u}_{k_1}^n - \bar{u}_{k-1}^n} - \frac{\left(\frac{h}{2} + \alpha h\right)\nabla u_{k-1}}{\bar{u}_{k_1}^n - \bar{u}_{k-1}^n} \quad \in [0,2].$$

This implies with (A.2), $\lambda < \frac{1}{4}$, and (4.24)

$$1 - \frac{1}{2}\frac{u_{k_1}^n(x_{\text{cut}}) - u_{k-1}^n(x_{\text{cut}})}{\bar{u}_{k_1}^n - \bar{u}_{k-1}^n} - \frac{\lambda h \nabla u_{k-1}}{\bar{u}_{k_1}^n - \bar{u}_{k-1}^n} = \frac{1}{2} - \frac{1}{2}\frac{\frac{\alpha h}{2}\nabla u_{k_1}}{\bar{u}_{k_1}^n - \bar{u}_{k-1}^n} + \frac{1}{2}\frac{\left(\left(\frac{h}{2} + \alpha h\right) - 2\lambda h\right)\nabla u_{k-1}}{\bar{u}_{k_1}^n - \bar{u}_{k-1}^n} \quad \in [0,1].$$

For T_3, we get from (A.1b) and (A.1c)

$$T_3 \leq \left(1 - \frac{\lambda}{1-\alpha}\frac{u_{k_2}^n(x_{k+\frac{1}{2}}) - u_{k_1}^n(x_{\text{cut}})}{\bar{u}_{k_2}^n - \bar{u}_{k_1}^n}\right) \left|\bar{u}_{k_2}^n - \bar{u}_{k_1}^n\right| + \left(\left(\frac{1}{2} + \frac{\alpha - 2\lambda}{2(1-\alpha)}\right)\frac{u_{k_1}^n(x_{\text{cut}}) - u_{k-1}^n(x_{\text{cut}})}{\bar{u}_{k_1}^n - \bar{u}_{k-1}^n} + \frac{\lambda h \nabla u_{k-1}}{\bar{u}_{k_1}^n - \bar{u}_{k-1}^n}\right) \left|\bar{u}_{k_1}^n - \bar{u}_{k-1}^n\right|.$$

Again we check the prefactors. The first prefactor is obviously positive because of (A.3) and $\frac{\lambda}{1-\alpha} \leq \frac{1}{2}$. Further, $\frac{1}{2} + \frac{\alpha - 2\lambda}{2(1-\alpha)} \geq 0$ for $\lambda \leq \frac{1}{2}$. Therefore, using (A.2), the second prefactor is positive as well. Finally, for T_4, we estimate

$$T_4 \leq \left(1 - \lambda \frac{u_{k+1}^n(x_{k+\frac{3}{2}}) - u_{k_2}(x_{k+\frac{1}{2}})}{\bar{u}_{k+1}^n - \bar{u}_{k_2}^n}\right) \left|\bar{u}_{k+1}^n - \bar{u}_{k_2}^n\right| + \frac{\lambda}{1-\alpha}\frac{u_{k_2}^n(x_{k+\frac{1}{2}}) - u_{k_1}(x_{\text{cut}})}{\bar{u}_{k_2}^n - \bar{u}_{k_1}^n} \left|\bar{u}_{k_2}^n - \bar{u}_{k_1}^n\right| + \left(\frac{\lambda}{1-\alpha} - \frac{\alpha}{2(1-\alpha)}\right)\frac{u_{k_1}^n(x_{\text{cut}}) - u_{k-1}(x_{\text{cut}})}{\bar{u}_{k_1}^n - \bar{u}_{k-1}^n} \left|\bar{u}_{k_1}^n - \bar{u}_{k-1}^n\right|.$$

For the last term, there holds $\frac{\lambda}{1-\alpha} - \frac{\alpha}{2(1-\alpha)} \geq 0$ due to $2\lambda \geq \alpha$. Together with other previously shown estimates, this implies that all three prefactors are positive. Finally, summing up the estimates for $T_1, \ldots, T_5$ implies the claim. ∎

REFERENCES

[1] S. Badia, F. Verdugo, and A. Martín, *The aggregated unfitted finite element method for elliptic problems*, Comput. Methods Appl. Mech. Engrg., 336 (2018), pp. 533–553.

[2] J. W. Barrett and C. M. Elliott, *Fitted and unfitted finite-element methods for elliptic equations with smooth interfaces*, IMA J. Numer. Anal., 7 (1987), pp. 283–300.

[3] T. J. Barth and D. Jespersen, *The design and application of upwind schemes on unstructured meshes*, Technical report AIAA-89-0366, AIAA 27th Aerospace Sciences Meeting, Reno, NV, 1989.

[4] P. Bastian, M. Blatt, A. Dedner, C. Engwer, R. Klöfkorn, M. Ohlberger, and O. Sander, *A generic grid interface for parallel and adaptive scientific computing. Part* I: *Abstract framework*, Computing, 82 (2008), pp. 103–119, https://doi.org/10.1007/s00607-008-0003-x.

[5] P. Bastian, M. Blatt, A. Dedner, C. Engwer, R. Klöfkorn, R. Kornhuber, M. Ohlberger, and O. Sander, *A generic grid interface for parallel and adaptive scientific computing. Part* II: *Implementation and tests in DUNE*, Computing, 82 (2008), pp. 121–138, https://doi.org/10.1007/s00607-008-0004-9.

[6] P. Bastian and C. Engwer, *An unfitted finite element method using discontinuous Galerkin*, Internat. J. Numer. Methods Engrg., 79 (2009), pp. 1557–1576.

[7] P. Bastian, C. Engwer, J. Fahlke, and O. Ippisch, *An unfitted discontinuous Galerkin method for pore-scale simulations of solute transport*, Math. Comput. Simulation, 81 (2011), pp. 2051–2061.

[8] P. Bastian, F. Heimann, and S. Marnach, *Generic implementation of finite element methods in the distributed and unified numerics environment (dune)*, Kybernetika, 46 (2010), pp. 294–315.

[9] M. Berger and C. Helzel, *A simplified h-box method for embedded boundary grids*, SIAM J. Sci. Comput., 34 (2012), pp. A861–A888.

[10] M. J. Berger, C. Helzel, and R. LeVeque, *H-box method for the approximation of hyperbolic conservation laws on irregular grids*, SIAM J. Numer. Anal., 41 (2003), pp. 893–918.

[11] M. J. Berger and R. LeVeque, *A rotated difference scheme for Cartesian grids in complex geometries*, AIAA Paper CP-91-1602, AIAA, Reston, VA, 1991.

[12] S. Bordas, E. Burman, M. Larson, and M. Olshanskii, *Geometrically Unfitted Finite Element Methods and Applications: Proceedings of the UCL Workshop 2016*, Lect. Notes Comput. Sci. Eng., Springer, New York, 2018.

[13] E. Burman, *Ghost penalty*, C. R. Math., 348 (2010), pp. 1217–1220, https://doi.org/10.1016/j.crma.2010.10.006.

[14] E. Burman, S. Claus, P. Hansbo, M. G. Larson, and A. Massing, *CutFEM: Discretizing geometry and partial differential equations*, Internat. J. Numer. Methods Engrg., 104 (2015), pp. 472–501.

[15] N. Chalmers and L. Krivodonova, *A robust CFL condition for the discontinuous Galerkin method on triangular meshes*, J. Comput. Phys., 403 (2020), p. 109095.

[16] I.-L. Chern and P. Colella, *A Conservative Front Tracking Method for Hyperbolic Conservation Laws*, Technical report, Lawrence Livermore National Laboratory, Livermore, CA, 1987, preprint UCRL-97200.

[17] B. Cockburn and C.-W. Shu, *Runge–Kutta discontinuous Galerkin methods for convection-dominated problems*, J. Sci. Comput., 16 (2001), pp. 173–261.

[18] P. Colella, D. T. Graves, B. J. Keen, and D. Modiano, *A Cartesian grid embedded boundary method for hyperbolic conservation laws*, J. Comput. Phys., 211 (2006), pp. 347–366.

[19] F. de Prenter, C. Lehrenfeld, and A. Massing, *A note on the penalty parameter in Nitsche's method for unfitted boundary value problems*, Comput. Math. Appl., 75 (2018), pp. 4322–4336.

[20] D. Di Pietro and A. Ern, *Mathematical Aspects of Discontinuous Galerkin Methods*, Springer New York, 2012.

[21] C. Engwer and F. Heimann, *Dune-udg: A cut-cell framework for unfitted discontinuous Galerkin methods*, in Advances in DUNE, Springer, New York, 2012, pp. 89–100.

[22] C. Engwer and A. Nüssing, *Geometric reconstruction of implicitly defined surfaces and domains with topological guarantees*, ACM Trans. Math. Software, 44 (2017), p. 14.

[23] S. Gottlieb and C.-W. Shu, *Total-variation-diminishing Runge-Kutta schemes*, Math. Comp., 67 (1998), pp. 73–85.
[24] C. Gürkan and A. Massing, *A Stabilized Cut Discontinuous Galerkin Framework:* II. *Hyperbolic Problems*, preprint, arXiv:1807.05634[math.NA], 2020.
[25] A. Hansbo and P. Hansbo, *An unfitted finite element method, based on Nitsche's method, for elliptic interface problems*, Comput. Methods Appl. Mech. Engrg., 191 (2002), pp. 5537–5552.
[26] C. Helzel, M. J. Berger, and R. LeVeque, *A high-resolution rotated grid method for conservation laws with embedded geometries*, SIAM J. Sci. Comput., 26 (2005), pp. 785–809.
[27] H. Holden and N. H. Risebro, *Front Tracking for Hyperbolic Conservation Laws*, Springer, New York, 2015.
[28] D. I. Ketcheson, C. B. Macdonald, and S. Gottlieb, *Optimal implicit strong stability preserving Runge-Kutta methods*, Appl. Numer. Math., 59 (2009), pp. 373–392.
[29] L. Krivodonova and R. Qin, *A discontinuous Galerkin method for solutions of the Euler equations on Cartesian grids with embedded geometries*, J. Comput. Sci., 4 (2013), pp. 24–35.
[30] F. Kummer, *Extended discontinuous Galerkin methods for two-phase flows: The spatial discretization*, Internat. J. Numer. Methods Engrg., 109 (2017), pp. 259–289.
[31] D. Kuzmin, *A vertex-based hierarchical slope limiter for p-adaptive discontinuous Galerkin methods*, J. Comput. Appl. Math., 233 (2010), pp. 3077–3085.
[32] R. J. LeVeque, *Numerical Methods for Conservation Laws*, Lectures in Mathematics ETH Zürich, Department of Mathematics Research Institute of Mathematics, Springer, New York, 1992, https://books.google.de/books?id=3WhqLPcMdPsC.
[33] S. May and M. J. Berger, *Two-dimensional slope limiters for finite volume schemes on non-coordinate-aligned meshes*, SIAM J. Sci. Comput., 35 (2013), pp. A2163–A2187.
[34] S. May and M. J. Berger, *An explicit implicit scheme for cut cells in embedded boundary meshes*, J. Sci. Comput., 71 (2017), pp. 919–943.
[35] A. Meurer et. al., *Sympy: Symbolic computing in python*, PeerJ Computer Science, 3 (2017), e103, https://doi.org/10.7717/peerj-cs.103.
[36] B. Müller, S. Krämer-Eis, F. Kummer, and M. Oberlack, *A high-order discontinuous Galerkin method for compressible flows with immersed boundaries*, Internat. J. Numer. Methods Engrg., (2016).
[37] R. Pember, J. B. Bell, P. Colella, W. Crutchfield, and M. L. Welcome, *An adaptive Cartesian grid method for unsteady compressible flow in irregular regions*, J. Comput. Phys., 120 (1995), pp. 278–304.
[38] R. J. Plemmons, *M-matrix characterizations.* I—*Nonsingular M-matrices*, Linear Algebra Appl., 18 (1977), pp. 175–188, https://doi.org/10.1016/0024-3795(77)90073-8.
[39] R. Saye, *Implicit mesh discontinuous Galerkin methods and interfacial gauge methods for high-order accurate interface dynamics, with applications to surface tension dynamics, rigid body fluid-structure interaction, and free surface flow: Part* I, J. Comput. Phys., 344 (2017), pp. 647–682.
[40] M. N. Spijker, *Contractivity in the numerical solution of initial value problems*, Numer. Math., 42 (1983), pp. 271–290.
[41] S. Sticko and G. Kreiss, *Higher order cut finite elements for the wave equation*, J. Sci. Comput., 80 (2019), pp. 1867–1887.
[42] F. Streitbürger, C. Engwer, S. May, and A. Nüssing, *Monotonicity Considerations for Stabilized DG Cut Cell Schemes for the Unsteady Advection Equation*, preprint, arXiv:1912.11933[math.NA], 2019.
[43] E. F. Toro, *Riemann Solvers and Numerical Methods for Fluid Dynamics*, 3rd ed., Springer, New York, 2009.
[44] B. Wendroff and A. B. White, *A supraconvergent scheme for nonlinear hyperbolic systems*, Comput. Math Appl., 18 (1989), pp. 761–767, https://doi.org/10.1016/0898-1221(89)90232-0.