ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Seminar for Applied Mathematics SAM

An Explicit Implicit Scheme for Cut Cells in Embedded Boundary Meshes

S. May and M. Berger

Research Report No. 2015-44
December 2015

Seminar für Angewandte Mathematik
Eidgenössische Technische Hochschule
CH-8092 Zürich
Switzerland

Funding: DE-FG02-88ER25053, FA9550-13-1-0052

An Explicit Implicit Scheme for Cut Cells in Embedded Boundary Meshes*

Sandra May† Marsha Berger‡

We present a new mixed explicit implicit time stepping scheme for solving the linear advection equation on a Cartesian cut cell mesh. Our scheme uses a standard second-order explicit scheme on Cartesian cells away from the embedded boundary. On cut cells, a second-order implicit scheme is used. This approach overcomes the small cell problem – that standard schemes are not stable on the arbitrarily small cut cells – while keeping the cost fairly low. We examine several approaches for coupling the schemes. For one of them, which we call *flux bounding*, we can show a TVD result. We also discuss the solution of the resulting implicit systems. All components of the scheme have been kept simple enough to afford the extension of the scheme to three dimensions. Numerical results in one, two, and three dimensions indicate that the resulting scheme is second-order accurate in L^1 and between first- and second-order accurate along the embedded boundary.

1. Introduction

The popularity of Cartesian cut cell methods continues to grow, due to the ease of grid generation for arbitrarily complicated domains and the efficiency of using Cartesian grids over most of the domain. The drawbacks continue as well - the difficulty of obtaining accurate solutions at the irregularly shaped cut cells where the geometry intersects the Cartesian grid. In addition, for problems where the integration method of choice is an explicit scheme, the so-called *small cell* problem remains an issue: explicit schemes are not stable on the arbitrarily small cut cell volumes without doing something special.

Several methods have been proposed to deal with the small cell problem. Cell merging [6, 36] is the most obvious idea but is much more difficult to do well than one would think. To our knowledge it has not yet been implemented in a three dimensional code that can automatically and robustly handle complicated geometry. Flux redistribution [15, 34, 18] has been implemented in three dimensions, but is only first order accurate at the cut cell boundaries. H-box methods [12, 11, 22] are second-order accurate, but are very complicated

*This work was supported in part by the DOE office of Advanced Scientific Computing Research under grant DE-FG02-88ER25053 and by AFOSR grant FA9550-13-1-0052

†Seminar for Applied Mathematics, ETH Zurich, Rämistrasse 101, 8092 Zurich, Switzerland

‡Courant Institute of Mathematical Sciences, New York University, 251 Mercer Street, New York, NY 10012, USA

and have not yet been implemented in three dimensions. So the final word on cut cell methods has not yet been written.

The goal of this research is to explore another obvious idea that for the most part seems to have been overlooked - treat the cut cells implicitly, even if the rest of the domain is updated with an explicit method. This should guarantee stability, allowing us to look directly at the accuracy issues. Since the implicit scheme is only used at the boundary, the cost won't be high. It is also a very intuitive approach for enabling an existing code that uses an explicit time stepping scheme on Cartesian grids to handle more complicated geometry. The main challenge is how to connect the explicit and the implicit scheme. We note that the approach examined in this work does *not* correspond to a classical IMEX approach: instead of using the explicit and implicit scheme on different operators of the equation, we use an implicit scheme on cells close to the embedded boundary and an explicit scheme elsewhere.

The idea of using a mixed explicit implicit scheme has been considered before. An early work [19] based on this approach uses either an implicit or explicit scheme based on the CFL number of each characteristic field, a related but more complicated approach. Jebens et al. [23] develop a partially implicit scheme in the context of cut cells, and provide an ODE-stability analysis for the time stepping scheme. However they do not discuss the spatial discretization, or the stability and conservation properties of their finite volume scheme, nor do they present accuracy studies. These aspects are a major focus of our paper. Some elements of our work are similar to those in [31], where fluxes are computed using an implicit (for inflow) and explicit (for outflow) scheme.

We develop the combined scheme in the context of the linear advection equation for an incompressible velocity field. Overcoming the small cell problem for the linear advection equation is an essential ingredient of our longer-term goal of extending an existing projection algorithm [4, 2, 8, 9] for solving the incompressible Euler and Navier-Stokes equations from Cartesian grids to cut cell grids. The projection method is based on 2 steps: *Step 1*, the prediction of the velocity field at the new time step when ignoring the incompressibility condition; and *Step 2*, the projection. In *Step 1*, a predictor-corrector approach based on a second-order MUSCL scheme is used. This explicit time stepping scheme leads to the small cell problem. To overcome this issue, we use our new explicit implicit scheme. We note that once the divergence-free velocities on edges/faces have been computed in the predictor step, the corrector step essentially treats the linear advection step we study here.

Our implementation is based on `BoxLib` [7], a library for massively parallel AMR applications. For the Cartesian cells away from the boundary we use `VarDen`, an implementation of the projection method referenced above. For the generation of the cut cells, we use `patchCubes`, a variant of the `cubes` mesh generator that is part of `Cart3D` package [14, 1]. The combination of these packages sets the framework for a parallel, three dimensional, adaptively refined flow solver for complex geometries once all the parts are done.

Our approach is to use MUSCL for the explicit scheme and trapezoidal time stepping (combined with a second-order finite volume scheme in space) for the implicit scheme. To couple them, we use what we refer to as *flux bounding*. Under suitable conditions we can show a total variation diminishing (TVD) result using flux bounding. Our numerical results in two and three dimensions show that our new algorithm is second-order accurate measured in the L^1 norm over the whole domain and between first- and second-order accurate along the embedded boundary.

The paper is organized as follows. In section 2 we discuss how to connect the explicit and implicit schemes in one dimension. We prove a theoretical result showing that among the

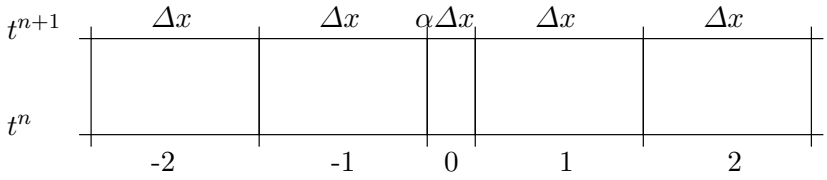


Figure 1: 1d model problem: Equidistant grid with one small cell of length $\alpha\Delta x$ labeled as cell 0.

two obvious choices for connecting the schemes, only flux bounding leads to a TVD result. In section 3 we discuss the extension of the mixed scheme to multiple dimensions. Section 4 focuses on how to solve the resulting sparse linear system. Since it only couples the cells close to the embedded boundary it is a much easier task than solving implicitly over the entire domain. Section 5 shows convergence studies for the linear advection equation for various test problems, as well as computational results for the incompressible Euler equations. This is just to show that these parts of the algorithm would carry over, although we do not discuss the projection algorithm further in this paper. We end with future directions for this research.

2. The scheme in one dimension

We consider the linear advection equation

$$s_t + us_x = 0, \quad u > 0 \,\text{constant}, \tag{1}$$

with CFL number $\lambda = \frac{u\Delta t}{\Delta x}$. Our mixed explicit implicit scheme uses (i) explicit time stepping on the Cartesian cells away from the embedded boundary, and (ii) implicit time stepping on cut cells. This way we avoid the small cell problem, and the overall cost is kept relatively low since only cells close to the boundary are treated implicitly. The challenge is to find a suitable way of transitioning between the schemes that ensures mass conservation and avoids unnecessary loss of accuracy.

We use MUSCL (Monotonic Upwind Scheme for Conservation Laws) [40, 16] for the explicit scheme. This scheme is well-established and has been implemented in many scientific codes. In addition, the projection algorithm in `Varden` uses that scheme in *Step 1* of the algorithm. In one dimension, the MUSCL scheme for the linear advection equation (1) on an equidistant grid is given by

$$s_i^{n+1} = s_i^n - \frac{\Delta t}{\Delta x}\left(F_{i+1/2}^{n+1/2} - F_{i-1/2}^{n+1/2}\right), \quad \text{with} \quad F_{i+1/2}^{n+1/2} = u\left(s_i^n + (1-\lambda)s_{x,i}^n \frac{\Delta x}{2}\right) \tag{2}$$

and $s_{x,i}^n \approx \partial_x s(x_i, t^n)$, computed using a standard slope limiter [26].

We use a model problem to study the behavior of the transition from explicit to implicit in the neighborhood of a cut cell. Consider the mesh shown in Figure 1: we use an equidistant grid with mesh width Δx except for one small cell with mesh width $\alpha\Delta x$, labeled as cell 0. Here, $\alpha \in (0, 1]$ denotes the *volume fraction* – the ratio of the small cell to full cell volume.

On the small cell 0, we use an implicit scheme for stability. If we used the explicit MUSCL scheme, we would need to adjust the time step to satisfy $\Delta t \leq \alpha\Delta x/u$ which is not feasible since α can be very small. For example, in our computations in two dimensions, the smallest

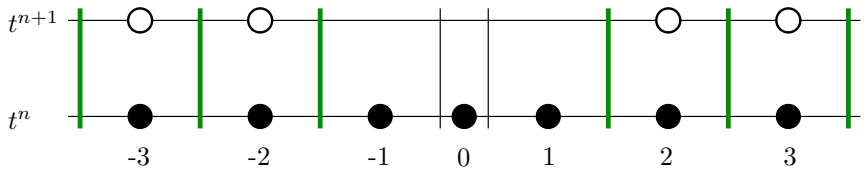
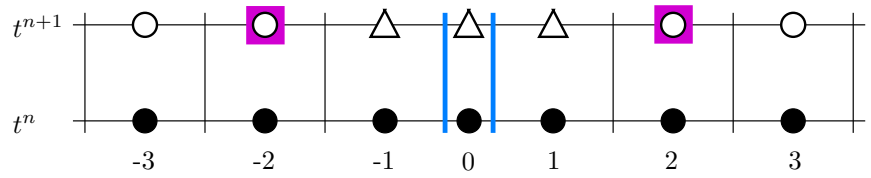
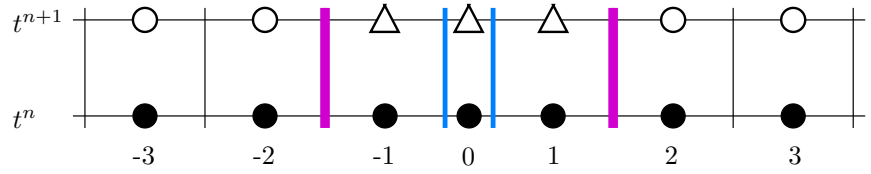


Figure 2: Cells with indices less or equal to -2 and greater or equal to 2 have been updated (empty circles) using fluxes computed by the explicit scheme (green lines); the neighborhood of the small cell has not yet been done.



(a) Cell bounding: values s_{-2}^{n+1} and s_{2}^{n+1} are input for implicit scheme.



(b) Flux bounding: explicit fluxes $F_{-3/2}^{n+1/2}$ and $F_{3/2}^{n+1/2}$ are input for implicit scheme.

Figure 3: Two approaches for switching the schemes: cell bounding and flux bounding. In both cases, the small cell uses implicit fluxes (blue line). The options differ in how the explicit and implicit schemes are connected (indicated by the pink color).

volume fraction α is between 10^{-3} and 10^{-6}. Our goal is to choose Δt only based on the size of the Cartesian cells. This is the *small cell problem*.

The region of implicitly treated cells should be as small as possible. If only cell 0 was treated implicitly, the explicit update of cell 1 would not have the right domain of dependence. If both cell 0 and 1 were treated implicitly (but not cell -1), then our algorithm would not carry over to systems of equations with a characteristic in the other direction. We are thus led to considering all 3 cells, -1, 0 and 1, to be in the implicit region.

In the example in Figure 2, cells with indices less than or equal to -2 or greater than or equal to 2 are updated with the explicit MUSCL scheme. Note that we can use the standard MUSCL scheme (2) for this step; however we do need to modify the computation of the slope on cell 1, which uses the value of the small cell 0. The cell 1 gradient is needed to compute the incoming flux for cell 2. We will discuss slope reconstruction in the presence of irregular cells later in this section.

We consider two different approaches for switching between the explicit and implicit scheme which we call *cell bounding* and *flux bounding*. In cell bounding, we use the *values* of the explicitly updated cells at time t^{n+1} as input for the implicit scheme. This is shown in Figure 3(a). In flux bounding, we use the *fluxes* that have been used to update the explicitly treated cells as input for the implicit scheme as shown in Figure 3(b). We will show that flux bounding

has superior stability and monotonicity properties.

We examine these two ways of switching more closely using a combination of explicit and implicit Euler time stepping, both using upwind differencing (without slope reconstruction) in space. Explicit Euler time stepping, or the upwind scheme, corresponds to the MUSCL scheme with gradients set to zero. Implicit Euler time stepping is known to be the most stable way of time stepping. If this way of transitioning shows instabilities, it will also cause problems when used in combination with a second-order implicit time stepping scheme with slope reconstruction (which is what we use for the computational results).

2.1. Cell bounding

Using the upwind scheme on cells -2 and 2 results in

$$\begin{aligned} s_{-2}^{n+1} &= s_{-2}^n - \lambda \left(s_{-2}^n - s_{-3}^n \right), \\ s_2^{n+1} &= s_2^n - \lambda \left(s_2^n - s_1^n \right). \end{aligned}$$

Implicit Euler time stepping applied on cells -1, 0, and 1 gives

$$\begin{aligned} s_{-1}^{n+1} &= s_{-1}^n - \lambda \left(s_{-1}^{n+1} - s_{-2}^{n+1} \right), \\ s_0^{n+1} &= s_0^n - \frac{\lambda}{\alpha} \left(s_0^{n+1} - s_{-1}^{n+1} \right), \\ s_1^{n+1} &= s_1^n - \lambda \left(s_1^{n+1} - s_0^{n+1} \right). \end{aligned}$$

Note that cell -1 uses us_{-2}^{n+1} as incoming flux. Since the value s_{-2}^{n+1} has already been updated explicitly, we can compute the values s_{-1}^{n+1}, s_0^{n+1}, and s_1^{n+1} in turn.

Unfortunately, this formulation is *not* conservative. The update on cell -2 uses us_{-2}^n as outgoing flux whereas the update on cell -1 uses us_{-2}^{n+1} as incoming flux. In general these two values will not match resulting in a loss or gain of mass. A similar problem occurs for the flux between cells 1 and 2.

To ensure mass conservation, we need a postprocessing step to fix the mass conservation. There are two possible options: we can either add the mass difference to the explicitly treated cell (cell -2) or to the implicitly treated cell (cell -1). Both of these options might lead to a loss of monotonicity as shown in the following.

Lemma 2.1. *Consider the following scheme: (i) compute s_{-2}^{n+1} using the explicit scheme, (ii) compute preliminary state $\tilde{s}_{-1}^{n+1}$ using the implicit scheme, (iii) adjust $\tilde{s}_{-1}^{n+1}$ accounting for the difference in mass resulting in s_{-1}^{n+1}, i.e.,*

$$(i): \quad s_{-2}^{n+1} = s_{-2}^n - \lambda \left(s_{-2}^n - s_{-3}^n \right), \tag{3a}$$

$$(ii): \quad \tilde{s}_{-1}^{n+1} = s_{-1}^n - \lambda \left(\tilde{s}_{-1}^{n+1} - s_{-2}^{n+1} \right), \tag{3b}$$

$$(iii): \quad s_{-1}^{n+1} = \tilde{s}_{-1}^{n+1} + \lambda s_{-2}^n - \lambda s_{-2}^{n+1}. \tag{3c}$$

This scheme can create overshoots and is not monotonicity preserving.

Proof. Suppose $s_i^n = 0$ for $i \leq -3$ and $s_i^n = 1$ for $i \geq -2$. A short computation shows

$$s_{-1}^{n+1} = 1 + \frac{\lambda^3}{1+\lambda} > 1,$$

for $\lambda > 0$. This is outside of the range of values of s_i^n and therefore is an overshoot. Also, monotonicity will not be preserved, since $s_i^{n+1} = 1$ for $i \geq 3$. □

The other option is also not monotonicity preserving.

Lemma 2.2. *Consider the following scheme: (i) compute a preliminary state $\hat{s}_{-2}^{n+1}$ using the explicit scheme, (ii) compute s_{-1}^{n+1} using the implicit scheme, (ii) adjust $\hat{s}_{-2}^{n+1}$ accounting for the difference in mass resulting in s_{-2}^{n+1}, i.e,*

$$(i): \quad \hat{s}_{-2}^{n+1} = s_{-2}^{n} - \lambda \left(s_{-2}^{n} - s_{-3}^{n} \right), \tag{4a}$$

$$(ii): \quad s_{-1}^{n+1} = s_{-1}^{n} - \lambda \left(s_{-1}^{n+1} - \hat{s}_{-2}^{n+1} \right), \tag{4b}$$

$$(iii): \quad s_{-2}^{n+1} = \hat{s}_{-2}^{n+1} + \lambda s_{-2}^{n} - \lambda \hat{s}_{-2}^{n+1}. \tag{4c}$$

This scheme is not monotonicity preserving for $\lambda \in \left(\frac{\sqrt{5}-1}{2}, 1 \right]$.

Proof. Consider again $s_i^n = 0$ for $i \leq -3$ and $s_i^n = 1$ for $i \geq -2$. Simple calculations show

$$s_{-2}^{n+1} = 1 - \lambda + \lambda^2 \quad \text{and} \quad s_{-1}^{n+1} = \frac{1 + \lambda - \lambda^2}{1 + \lambda}.$$

Therefore,

$$s_{-2}^{n+1} \leq s_{-1}^{n+1} \quad \Leftrightarrow \quad \lambda^2 + \lambda - 1 \leq 0.$$

This is true for $\lambda \geq \frac{\sqrt{5}+1}{2}$ and $\lambda \leq \frac{\sqrt{5}-1}{2} \approx 0.62$. □

Remark 2.1. We remark that without the conservation fixup step, there is no monotonicity problem with the combined explicit/implicit scheme. Simple calculations show that the transition between explicit to implicit and back, as well as the cells surrounding the small cell where the mesh width changes, all preserve monotonicity.

2.2. Flux bounding

Next we examine flux bounding more closely. As with cell bounding, we first update the explicitly treated cells -2 and 2,

$$s_{-2}^{n+1} = s_{-2}^{n} - \lambda \left(s_{-2}^{n} - s_{-3}^{n} \right),$$
$$s_{2}^{n+1} = s_{2}^{n} - \lambda \left(s_{2}^{n} - s_{1}^{n} \right).$$

For flux bounding, we use the fluxes $F_{-3/2} = u s_{-2}^n$ and $F_{3/2} = u s_1^n$, computed with the explicit upwind scheme, as input for the implicitly treated cells. Therefore, it only remains to choose the fluxes $F_{-1/2}$ and $F_{1/2}$. Since we want to treat the small cell fully implicitly, we choose implicit Euler time stepping for these fluxes. This results in the update formulae

$$s_{-1}^{n+1} = s_{-1}^{n} - \lambda \left(s_{-1}^{n+1} - s_{-2}^{n} \right), \tag{5a}$$

$$s_{0}^{n+1} = s_{0}^{n} - \frac{\lambda}{\alpha} \left(s_{0}^{n+1} - s_{-1}^{n+1} \right), \tag{5b}$$

$$s_{1}^{n+1} = s_{1}^{n} - \lambda \left(s_{1}^{n} - s_{0}^{n+1} \right). \tag{5c}$$

This approach of connecting the schemes ensures mass conservation by construction.

Remark 2.2. We refer to cells -1 and 1 as transition cells.

Unlike cell bounding, we can show a positive result for flux bounding.

Lemma 2.3. *The scheme described in equations* (5a)-(5c) *is monotonicity preserving for* $0 \leq \lambda \leq 1$.

Proof. We focus on cells -2 to 2. Suppose WLOG that $s_i^n \leq s_{i+1}^n$. We need to show that $s_i^{n+1} \leq s_{i+1}^{n+1}$. Using the notation $\mathcal{C}(a_1, \ldots, a_n)$ to denote a convex combination of elements $a_1, \ldots, a_n$, there holds

$$s_{-1}^{n+1} = \frac{1}{1+\lambda} s_{-1}^n + \frac{\lambda}{1+\lambda} s_{-2}^n = \mathcal{C}(s_{-1}^n, s_{-2}^n) \geq s_{-2}^{n+1}$$

as $s_{-2}^{n+1} = \mathcal{C}(s_{-2}^n, s_{-3}^n)$. Further,

$$s_0^{n+1} = \frac{1}{1+\frac{\lambda}{\alpha}} s_0^n + \frac{\frac{\lambda}{\alpha}}{1+\frac{\lambda}{\alpha}} s_{-1}^{n+1} \geq \frac{1}{1+\frac{\lambda}{\alpha}} s_{-1}^{n+1} + \frac{\frac{\lambda}{\alpha}}{1+\frac{\lambda}{\alpha}} s_{-1}^{n+1} = s_{-1}^{n+1}$$

as $s_{-1}^{n+1} = \mathcal{C}(s_{-1}^n, s_{-2}^n) \leq s_0^n$. Finally, using $s_0^{n+1} = \mathcal{C}(s_0^n, s_{-1}^n, s_{-2}^n)$ as well as

$$s_1^{n+1} = (1-\lambda)\, s_1^n + \lambda s_0^{n+1} = \mathcal{C}(s_1^n, s_0^n, s_{-1}^n, s_{-2}^n),$$

and the monotonicity of data at t^n, there holds

$$\begin{aligned} s_1^{n+1} &= (1-\lambda)\, s_1^n + \lambda s_0^{n+1} \geq s_0^{n+1}, \\ s_2^{n+1} &= (1-\lambda) s_2^n + \lambda s_1^n \geq s_1^{n+1}, \end{aligned}$$

which implies the claim. □

Next, we consider the mixed scheme consisting of the second-order MUSCL scheme and the first-order implicit Euler time stepping scheme with piecewise constant data. To be more precise, the scheme is

$$\begin{aligned} s_{-3}^{n+1} &= s_{-3}^n - \frac{\Delta t}{\Delta x}\left[F_{-5/2}^{n+1/2} - F_{-7/2}^{n+1/2}\right], \quad \text{(MUSCL)}, \\ s_{-2}^{n+1} &= s_{-2}^n - \frac{\Delta t}{\Delta x}\left[u s_{-2}^n - F_{-5/2}^{n+1/2}\right], \\ \\ s_{-1}^{n+1} &= s_{-1}^n - \lambda\left[s_{-1}^{n+1} - s_{-2}^n\right], \\ s_0^{n+1} &= s_0^n - \frac{\lambda}{\alpha}\left[s_0^{n+1} - s_{-1}^{n+1}\right], \\ s_1^{n+1} &= s_1^n - \lambda\left[s_1^n - s_0^{n+1}\right], \\ \\ s_2^{n+1} &= s_2^n - \frac{\Delta t}{\Delta x}\left[F_{5/2}^{n+1/2} - u s_1^n\right], \\ s_3^{n+1} &= s_3^n - \frac{\Delta t}{\Delta x}\left[F_{7/2}^{n+1/2} - F_{5/2}^{n+1/2}\right], \quad \text{(MUSCL)}, \end{aligned} \tag{6}$$

where the slopes $s_{x,i}^n$ have been computed using the minmod limiter and the slopes for fluxes $F_{-3/2}$ and $F_{3/2}$ have been set to zero.

Theorem 2.1. *The scheme* (6) is TVD *for the linear advection equation* (1) *for* $0 \leq \lambda \leq 1$, *if the exact solution has compact support.*

The proof is somewhat lengthy and is given in Appendix A. Note that this also implies that the mixed scheme (6) is monotonicity preserving, and therefore this would be an alternative proof of Lemma 2.3. However we presented the proof of Lemma 2.3 since it shows the main properties of flux bounding in a simpler setting.

Based on these positive results for flux bounding, and on the negative results for cell bounding, we use *flux bounding* to connect the explicit and the implicit scheme in one, two, and three dimensions.

2.3. Choice of the implicit scheme

So far we have chosen an explicit scheme and have decided on how to connect the explicit and the implicit scheme. It remains to choose an implicit scheme. We need a second-order implicit time stepping scheme. In addition, we would like to take into account that our cut cells can be arbitrarily small. Therefore, we would like to use a scheme that is unconditionally TVD/SSP (strong stability preserving). Unfortunately, there is no second-order implicit scheme with this property [24, 38].

Currently, we use the implicit trapezoidal rule, which is given by

$$y^{n+1} = y^n + \frac{\Delta t}{2}\left(f(y^n) + f(y^{n+1})\right)$$

for the ODE $\frac{d}{dt}y(t) = f(y(t))$. One major reason for this decision was the following. The MUSCL scheme evaluates the flux at time $t^{n+1/2}$. For the mixed scheme to be exact for linear functions, we need an implicit scheme that also (on average) evaluates the flux at $t^{n+1/2}$. The trapezoidal rule is the simplest method that has this property. We also looked at other schemes as TR-BDF2 or some versions of L-stable SDIRK scheme that on average evaluate the flux at $t^{n+1/2}$. These schemes have two implicit solves, which makes them more expensive, while having similar issues in terms of monotonicity as the trapezoidal rule.

For the space reconstruction on the non-equally spaced cells -1, 0, and 1, we use a standard least squares slope reconstruction – which extends in a straight-forward way to higher dimensions [5]. We compute the unlimited least squares slope $s_{x,i}$ as solution to the problem $\min_{s_{x,i}} \|\mathbf{r}\|_2$ with

$$\mathbf{r} = \begin{bmatrix} x_{i+1} - x_i \\ x_{i-1} - x_i \end{bmatrix} s_{x,i} - \begin{bmatrix} s_{i+1} - s_i \\ s_{i-1} - s_i \end{bmatrix}$$

and x_i denoting the centroid of cell i. We compute limited slopes by applying the `minmod` slope limiter [26].

The full scheme then has the form

$$\begin{aligned}
s_{-2}^{n+1} &= s_{-2}^n - \frac{\Delta t}{\Delta x}\left[F_{-3/2}^{n+1/2} - F_{-5/2}^{n+1/2}\right], \quad \text{(MUSCL)},\\
s_{-1}^{n+1} &= s_{-1}^n - \frac{\Delta t}{\Delta x}\left[F_{-1/2}^{T} - F_{-3/2}^{n+1/2}\right],\\
s_{0}^{n+1} &= s_{0}^n - \frac{\Delta t}{\alpha\Delta x}\left[F_{1/2}^{T} - F_{-1/2}^{T}\right],\\
s_{1}^{n+1} &= s_{1}^n - \frac{\Delta t}{\Delta x}\left[F_{3/2}^{n+1/2} - F_{1/2}^{T}\right],\\
s_{2}^{n+1} &= s_{2}^n - \frac{\Delta t}{\Delta x}\left[F_{5/2}^{n+1/2} - F_{3/2}^{n+1/2}\right], \quad \text{(MUSCL)},
\end{aligned} \tag{7}$$

Table 1: Error for mixed scheme for taking one time step and computing up to time T.

Final time	Δx	L^1 error	order	L^∞ error	order
1 step error	1/80	2.39e-05	–	2.28e-03	–
	1/160	2.95e-06	3.02	5.52e-04	2.04
	1/320	3.65e-07	3.01	1.36e-04	2.02
1 period error	1/80	2.10e-04	–	1.58e-03	–
	1/160	5.68e-05	1.89	3.51e-04	2.17
	1/320	1.48e-05	1.94	7.41e-05	2.24

with

$$F^T_{i+1/2} = \frac{u}{2}\left(s^n_i + s^n_{x,i}(x_{i+1/2} - x_i) + s^{n+1}_i + s^{n+1}_{x,i}(x_{i+1/2} - x_i)\right).$$

We apply this mixed scheme to a *linear* test function and take one time step. It is easily verified numerically that the mixed scheme is exact for this test, and does not cause errors on the transition cells -1 and 1 or in the small cell 0. When applying the scheme to a general smooth test function using unlimited slope reconstruction, we observe second-order one-step errors on the transition cells due to switching schemes and on the small cell. However, as often observed with embedded boundary methods, these errors do not accumulate in the usual way, as our numerical results confirm. This is discussed further below.

Use of the trapezoidal rule unfortunately leads to a combined scheme (see (7)) that is not TVD, so is not guaranteed to preserve the monotonicity we were seeking earlier. Our plan is to use an FCT (flux corrected transport) approach [13, 42] combining trapezoidal rule with slope reconstruction with implicit Euler time stepping with piecewise constant data, which, based on Theorem 2.1, would result in a TVD scheme.

2.4. Numerical results in one dimension

We conclude the description of the one-dimensional method with numerical results concerning the accuracy of the scheme. We compute on the grid shown in Figure 1 with $\alpha = 10^{-4}$. We use smooth initial data given by $\sin\left(\frac{2\pi x}{1+\alpha\Delta x}\right)$ with periodic boundary conditions, and solve $s_t + 2s_x = 0$ with $\lambda = 0.8$ on the interval $[0, 1 + \alpha\Delta x]$. The L^1 error has been normalized to account for the changing domain length.

We examine the results for taking one time step, as well as for running until time $T = 0.5 \cdot (1 + \alpha\Delta x)$, when the test function is back to its original position. Both results are shown in Table 1. The second-order convergence of the error in the L^∞ norm for taking one time step confirms the theoretical considerations about reduced accuracy on the transition and cut cells. However, the second-order convergence of the error in both norms for computing up to time T also confirms that the one step error does not accumulate in the usual way and that the mixed scheme is indeed fully second-order accurate for this test. This is a previously known and interesting phenomena, see, e.g. [32, 41], also [20].

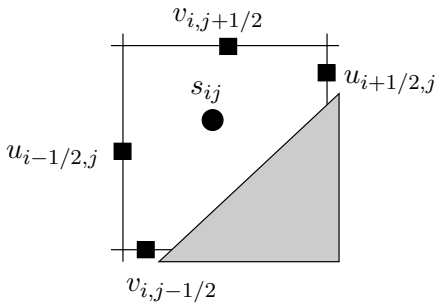


Figure 4: Notation used to denote the cell centered and edge centered variables.

3. The scheme in multiple dimensions

In this section, we describe the extension of the mixed scheme to multiple dimensions. We will focus on the extension to two dimensions as the extension to three dimensions follows the same ideas but is technically more complicated. We will remark on complications in three dimensions at the end of this section.

In two dimensions, we solve the linear advection equation

$$s_t + (us)_x + (vs)_y = 0. \tag{8}$$

Here, $s(x, y, t)$ denotes a scalar field located at the cell centroid, as shown in Figure 4 in the case of a cut cell. For a second-order scheme, we can identify the unknown s_{ij}^n, which represents the cell average of cell (i, j) at time t^n, with the value at the cell centroid.

Let $\mathbf{u} = (u, v)$ denote an edge centered divergence-free velocity field that satisfies (up to machine precision)

$$\frac{u_{i+1/2,j}^{n+1/2} - u_{i-1/2,j}^{n+1/2}}{\Delta x} + \frac{v_{i,j+1/2}^{n+1/2} - v_{i,j-1/2}^{n+1/2}}{\Delta y} = 0$$

for Cartesian cells and

$$\frac{u_{i+1/2,j}^{n+1/2}\beta_{i+1/2,j} - u_{i-1/2,j}^{n+1/2}\beta_{i-1/2,j}}{\Delta x} + \frac{v_{i,j+1/2}^{n+1/2}\beta_{i,j+1/2} - v_{i,j-1/2}^{n+1/2}\beta_{i,j-1/2}}{\Delta y} = 0$$

for cut cells (see Figure 4 for velocity centering). Here, $\beta_{i+1/2,j}, \beta_{i,j+1/2} \in [0, 1]$ represent the area fractions of cut cell edges $(i + 1/2, j)$ and $(i, j + 1/2)$. We assume no flow boundary conditions on the embedded boundary Γ given by $\mathbf{u} \cdot \mathbf{n} = 0$ on Γ, with $\mathbf{n}$ denoting the outer normal vector of the embedded object.

As explicit scheme we use an unsplit version of the MUSCL scheme. Unsplit schemes in two dimensions were first introduced by Colella [17]. The scheme is significantly more complicated in two dimensions than in one dimension since it has transverse derivatives for corner-coupling. We will not give a detailed description here and instead refer to Almgren et al [4].

We use flux bounding to switch between the explicit and the implicit scheme. As in one dimension, we introduce transition cells. These are full Cartesian cells that share an edge with a cut cell as shown in Figure 5. The update formula for cut cells uses implicit fluxes for all existing edges. The update formula for transition cells uses implicit fluxes on edges shared with cut cells and explicit fluxes on edges shared with Cartesian cells.

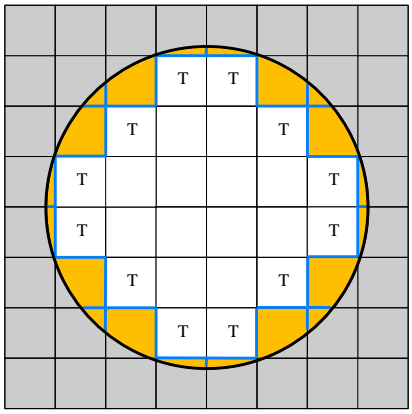


Figure 5: Switching between schemes in two dimensions: The domain used in the simulation corresponds to the interior of the circle. 'Transition' cells (marked with 'T') are full Cartesian cells that share an edge with a cut cell. Cut cells are marked with yellow color, implicit fluxes by a bold, blue line. The remaining fluxes are explicit.

In particular, the flux between two transition cells is computed using the MUSCL scheme. Due to corner-coupling (essentially the use of a transverse derivative in computing the normal edge state), this could involve a cut cell for the evaluation of a transverse derivative. Therefore, some minor adjustments of the MUSCL scheme due to the presence of cut cells are necessary but straightforward, involving only the cut cell gradient computation (described next) and prediction of cut cell edge states. For details see [28].

Analogous to one dimension, our mixed algorithm follows the following structure: given s_{ij}^n,

(i) compute all explicit fluxes using the MUSCL scheme and update all fully explicitly treated cells to s_{ij}^{n+1};

(ii) compute all implicit fluxes and update cut cells and transition cells to s_{ij}^{n+1}.

We again use trapezoidal rule time stepping combined with linear reconstruction in space. At both cut and transition cells, we use a least squares formulation to compute the unlimited gradient. For limiting we use the LP limiter developed specifically for cut cells [29]: this limiter limits the x- and y-slope separately, reducing numerical diffusion compared to a scalar limiter, while satisfying a positivity result. We also apply the LP limiter on transition cells which need special treatment since the transition cell centroid is not coordinate-aligned with the cut cell centroid.

For a cut cell, the update is given by

$$s_{ij}^{n+1} = s_{ij}^n - \frac{\Delta t}{\alpha_{ij}\Delta x \Delta y}\left[F_{i+1/2,j}^T - F_{i-1/2,j}^T + G_{i,j+1/2}^T - G_{i,j-1/2}^T\right] \tag{9}$$

with $\alpha_{ij} \in (0,1)$ representing the volume fraction,

$$F_{i+1/2,j}^T = \frac{1}{2} u_{i+1/2,j}^{n+1/2} \beta_{i+1/2,j} \Delta y (s_{i+1/2,j}^n + s_{i+1/2,j}^{n+1}),$$

and

$$s^n_{i+1/2,j} = \begin{cases} s^n_{ij} + (x_{i+1/2,j} - x_{ij}) s^n_{x,ij} + (y_{i+1/2,j} - y_{ij}) s^n_{y,ij} & \text{if } u^{n+1/2}_{i+1/2,j} > 0, \\ s^n_{i+1,j} + (x_{i+1/2,j} - x_{i+1,j}) s^n_{x,i+1,j} + (y_{i+1/2,j} - y_{i+1,j}) s^n_{y,i+1,j} & \text{if } u^{n+1/2}_{i+1/2,j} < 0. \end{cases}$$

Here $(x_{i+1/2,j}, y_{i+1/2,j})$ denotes the location of the edge midpoint of face $(i+1/2, j)$, (x_{ij}, y_{ij}) denotes the centroid of cut cell (i, j), and $s_{x,ij}$ and $s_{y,ij}$ refer to the reconstructed x- and y-slopes respectively in cell (i, j). The fluxes $G^T_{i,j+1/2}$ are defined analogously.

The update of a transition cell uses both explicit and implicit fluxes. Consider for example a transition cell (i, j) whose cell neighbors $(i+1, j)$ and $(i, j+1)$ are cut cells and whose cell neighbors $(i-1, j)$ and $(i, j-1)$ are Cartesian cells. Then, the update is given by

$$s^{n+1}_{ij} = s^n_{ij} - \frac{\Delta t}{\Delta x \Delta y} \left[F^T_{i+1/2,j} - F^{n+1/2}_{i-1/2,j} + G^T_{i,j+1/2} - G^{n+1/2}_{i,j-1/2} \right], \tag{10}$$

with $F^{n+1/2}_{i-1/2,j}$ and $G^{n+1/2}_{i,j-1/2}$ representing the MUSCL fluxes.

Remark 3.1. As in two dimensions, we also use a fully unsplit version of the explicit MUSCL scheme in three dimensions based on the ideas of Saltzmann [37]. (See also [33] for more information about unsplit schemes in three dimensions.) The scheme includes face neighbors, edge neighbors, and node neighbors in the computation of the fluxes. As a consequence, the scheme is stable for a CFL number of 1.0.

The extension of the mixed explicit implicit scheme to three dimensions follows the same ideas as the extension to two dimensions. We define transition cells in three dimensions as both the face and edge neighbors of cut cells. As in two dimensions, the flux between a cut cell and a transition cell uses implicit time stepping, whereas the flux between Cartesian transition and flow cells is computed with the explicit scheme. On transition cells we have disabled the use of node neighbors in the flux computation of the MUSCL scheme and only use face and edge neighbors. If this version was used in the whole domain, this would lead to a reduced CFL number of 0.8 (compared to 1.0) [2]. However, one can show [10] that in many cases, one can take a larger time step at the boundary on a cut cell and still maintain stability. In particular, in the case of a first order scheme with a cut cell volume that is at least half the regular cell size, a full CFL can still be used. In our numerical results in three dimensions we have not observed instabilities using a CFL number of up to 0.95.

4. Solving the implicit systems

An implicit system coupling the cut cells and transitions cells needs to be solved at each time step. Note that the number of unknowns is one dimension lower than the number of overall grid cells. In two dimensions on a Cartesian grid with N cells in each direction, we expect $O(N)$ unknowns in the implicit system, so the overall increase in cost is reasonable.

We currently use an inexact damped Newton method to solve the implicit systems. Let $Z \in \mathbb{R}^m$ be the vector of unknown values in cut and transition cells. Let the function $F(Z)$ incorporate the corresponding updates of the form (9) and (10), respectively, so that we can write the system to be solved in the form $F(Z) = 0$.

For an exact Newton method, we would need to build the Jacobian $\left(\frac{\partial F(Z)}{\partial Z}\right)_{i,j=1}^{m}$ based on the second-order implicit scheme described above, which in particular includes (limited) slope reconstruction. Instead, we use an approximate Jacobian based on using trapezoidal time stepping with piecewise constant data. This has the advantages that the Jacobian is very sparse and strictly diagonal dominant as shown in the following lemma. In particular, this implies that the linear systems that need to be solved in each Newton iteration are well-posed. Approximate Jacobians are a commonly used tool in computational fluid dynamics calculations [35, 27].

Lemma 4.1. *We consider the Jacobian J corresponding to the mixed explicit implicit scheme with piecewise constant data for the implicit scheme. Let the velocity field be divergence-free and let the CFL condition*

$$\max\left(\max_{i,j}\frac{|u_{i+1/2,j}|\Delta t}{\Delta x},\max_{i,j}\frac{|v_{i,j+1/2}|\Delta t}{\Delta y}\right)<1$$

be satisfied for Cartesian cells in two dimensions, and

$$\max\left(\max_{i,j,k}\frac{|u_{i+1/2,j,k}|\Delta t}{\Delta x},\max_{i,j,k}\frac{|v_{i,j+1/2,k}|\Delta t}{\Delta y},\max_{i,j,k}\frac{|w_{i,j,k+1/2}|\Delta t}{\Delta z}\right)<\frac{2}{3}$$

in three dimensions. We assume that the geometry is sufficiently resolved and that the velocity field satisfies $\mathbf{u}\cdot\mathbf{n}=0$ on the embedded boundary. Then, the matrix J

1. *has at most $2d-1$ non-zero, non-diagonal entries in a row (d denoting the dimension),*
2. *has real entries with the entries on the diagonal being positive,*
3. *is strictly diagonally dominant.*

Remark 4.1. Note that the implicit scheme is only used at embedded boundaries. Therefore, other boundaries can be ignored.

Proof. In the proof we focus on the two-dimensional case. The three-dimensional proof is completely analogous unless otherwise indicated. We use the shorthand notation

$$u_{\pm}=u^{n+1/2}_{i\pm1/2,j},\quad v_{\pm}=v^{n+1/2}_{i,j\pm1/2}.$$

The proof relies on the incompressibility condition given by

$$\frac{u_{+}\beta_{i+1/2,j}-u_{-}\beta_{i-1/2,j}}{\Delta x}+\frac{v_{+}\beta_{i,j+1/2}-v_{-}\beta_{i,j-1/2}}{\Delta y}=0.$$

Here, $\beta_{(\cdot)}\in[0,1]$ corresponds to the area fraction on each face, i.e., $\beta_{(\cdot)}=1$ for a full Cartesian face, and $\beta_{(\cdot)}=0$ for a non-existing face. This condition implies in particular that every cell has inflow and outflow faces, which in turn implies claim 1.

To show claim 2, that the entries on the main diagonal are positive, we define

$$U_{\pm}=\pm\frac{\Delta t}{2V_{ij}}\Delta y\cdot u_{\pm}\cdot\beta_{i\pm1/2,j},\quad V_{\pm}=\pm\frac{\Delta t}{2V_{ij}}\Delta x\cdot v_{\pm}\cdot\beta_{i,j\pm1/2}.$$

Here, V_{ij} denotes the area of cell (i,j). The entries on the main diagonal are given by

$$a_{ii} = 1 + \underbrace{U_+ \cdot \mathbb{I}_{\{u_+>0\}}}_{>0} + \underbrace{U_- \cdot \mathbb{I}_{\{u_-<0\}}}_{>0} + \underbrace{V_+ \cdot \mathbb{I}_{\{v_+>0\}}}_{>0} + \underbrace{V_- \cdot \mathbb{I}_{\{v_-<0\}}}_{>0} > 1,$$

with $\mathbb{I}_{\{u_+>0\}} = 1$ if $u_+ > 0$ and $\mathbb{I}_{\{u_+>0\}} = 0$ otherwise. This concludes the proof of claim 2.

For claim 3, let us first consider the situation of a cut cell. The sum of the absolute values of the off-diagonal entries is given by

$$\sum_{j\neq i} |a_{ij}| = -U_+ \cdot \mathbb{I}_{\{u_+<0\}} - U_- \cdot \mathbb{I}_{\{u_->0\}} - V_+ \cdot \mathbb{I}_{\{v_+<0\}} - V_- \cdot \mathbb{I}_{\{v_->0\}}.$$

This implies

$$\begin{aligned} |a_{ii}| - \sum_{j\neq i} |a_{ij}| &= 1 + U_+ \cdot \mathbb{I}_{\{u_+>0\}} + U_- \cdot \mathbb{I}_{\{u_-<0\}} + V_+ \cdot \mathbb{I}_{\{v_+>0\}} + V_- \cdot \mathbb{I}_{\{v_-<0\}} \\ &\quad + U_+ \cdot \mathbb{I}_{\{u_+<0\}} + U_- \cdot \mathbb{I}_{\{u_->0\}} + V_+ \cdot \mathbb{I}_{\{v_+<0\}} + V_- \cdot \mathbb{I}_{\{v_->0\}} \\ &= 1 + U_+ + U_- + V_+ + V_-. \end{aligned}$$

Due to the incompressibility condition, there holds $U_+ + U_- + V_+ + V_- = 0$, which implies $|a_{ii}| - \sum_{j\neq i} |a_{ij}| = 1$.

We now examine the case of a transition cell for which there are differences between two and three dimensions. Based on the condition that the geometry is sufficiently resolved, we deduce that a transition cell has at most d cut cell face neighbors. (Otherwise split each Cartesian cell in 2^d cells; then every transition cell has at most d cut cell neighbors.) Using the same argument as used for a cut cell, there holds $|a_{ii}| \geq 1$. As explicit fluxes contribute to the right hand side of the linear system, the row of J corresponding to this cell has at most d non-zero, non-diagonal entries that are given by either $U_+, U_-, \ldots$ or W_-. Based on the CFL condition, there holds

$$\max\left(|U_+|, |U_-|, |V_+|, |V_-|, |W_+|, |W_-|\right) < \nu(d)$$

with $\nu(d) = \frac{1}{2}$ for $d = 2$ and $\nu(d) = \frac{1}{3}$ for $d = 3$. Therefore,

$$|a_{ii}| - \sum_{j\neq i} |a_{ij}| > 1 - d \cdot \frac{1}{d} = 0,$$

which implies the claim. □

In our inexact Newton method, we use $\|J^{-1}F(Z^{(k)})\|_\infty / \|J^{-1}F(Z^{(0)})\|_\infty$ as stopping criterion. We measure the error in the maximum norm (instead of, e.g., the L^1 or L^2 norm) to avoid significant errors for solving the implicit system on the potentially very small cut cells, which would result in computing unstable numerical fluxes for cut cells. Depending on the accuracy needed, we usually ask for a reduction of the residual by 4-7 orders of magnitude.

There is an issue of convergence stalling when the implicit scheme is used with limited slopes. In each step of the Newton method, we use a line search based on backtracing to determine the optimal step length of the Newton update. If the step length algorithm results in the smallest possible step length 2^{-12}, we freeze the slopes. In most cases this results in convergence of the method in the next iteration. In some cases the convergence stalling continues; we then accept the latest iterate as the solution. We will examine this issue more closely in the future when

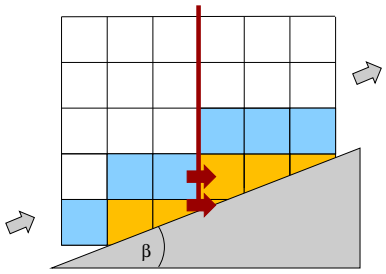
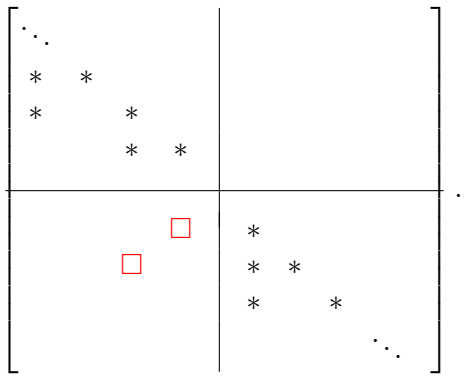


Figure 6: Grid setup for the ramp test in two dimensions: the values of both cut cells (marked in yellow) and transition cells (marked in light blue) are determined by an implicit system. The red line indicates the splitting in patches.

we will develop an FCT approach for coupling our scheme with implicit Euler time stepping in order to guarantee TVD stability.

For practical purposes, we need to be able to solve the implicit system in the context of several patches. This is essential for mesh refinement and parallel computing. We currently use a block-Jacobian approach. Consider the example of the ramp test shown in Figure 6. We split the grid in two patches following the red line. Let the corresponding Jacobian J of the first-order space discretization be given by



In our block-Jacobian approach we ignore contributions connecting the two patches, in this case the two contributions marked by a red square in the matrix representation. These correspond to the fluxes marked by the red arrows in Figure 6. Instead, we invert the blocks individually and treat the fluxes marked with the red arrows as fixed Dirichlet inflow data (evaluated using the previous iterate). The communication between patches takes place via the update of ghost cells after each Newton iteration. This approach worked very well for our parallel tests using unlimited slopes and up to 12 cores. More extensive studies that solve the implicit system using a higher level of parallelism will be future work.

5. Numerical results

This section presents numerical results in two and three dimensions. All tests were run on a 12 core Dell precision desktop machine. Parallel performance scaling studies will be future work.

5.1. Numerical results in two dimensions

In the following experiments we will examine the error e measured both in the L^1 norm and in the L^∞ norm over the whole domain given by

$$L^1(e) = \sum_{i,j} |e_{ij}| \, V_{ij} \quad \text{and} \quad L^\infty(e) = \max_{i,j} |e_{ij}|,$$

with e_{ij} denoting the pointwise error at the cell centroid and V_{ij} denoting the area of cell (i, j). Given that the MUSCL scheme is a second-order scheme and that the boundary has lower dimensionality, we expect $L^1(e)$ to converge with second order provided that the scheme close to the boundary is at least first-order accurate. To get a better idea of the error close to the boundary, we also examine $L^\infty(e)$.

In the following tests, we compute the time step based on the CFL number λ as

$$\Delta t = \lambda \min \left(\frac{\Delta x}{\max_{i,j} |u_{i+1/2,j}|}, \frac{\Delta y}{\max_{i,j} |v_{i,j+1/2}|} \right).$$

(Note that this formula does not take the size of cut cells into account.) We use $\lambda = 0.9$ to satisfy the time step restriction for the MUSCL scheme. We use $\Delta x = \Delta y$ but this is not a requirement.

5.1.1. Advection along a ramp

We start with advection along a ramp: we intersect a ramp with angle β with a Cartesian grid. This is sketched in Figure 6. We note that one can easily construct a Cartesian grid without cut cells for this simple geometry (as is the case for most of the tests presented here). However, these simple tests are very suitable for testing the properties of our mixed explicit implicit scheme.

To test the accuracy of our scheme we use unlimited gradients both for our explicit and implicit scheme. Our test function is a one-dimensional quadratic function with respect to the line that is perpendicular to the ramp. Since the MUSCL scheme is exact for quadratics, we expect the error in the interior of the domain to be close to machine precision, and all error should be caused by switching schemes in the transition cells and by the irregularity of the cut cells. The velocity field is given by $(u, v)^T = (2, 2\tan(\beta))^T$ and the final time is $T = 0.1$.

The results are shown in Table 2. We observe second-order convergence of $L^1(e)$. The error measured in the L^∞ norm varies with the grid angle. For all angles though the convergence rates are better than first order but less than second order. It is clearly seen in the table that the maximum error is not a smooth function of the angle or the mesh width, as there is no smoothness in the cut cells.

Table 2: Ramp test: Error and convergence rate for mixed scheme for different ramp angles.

Angle β	Δx	$L^1(e)$	order	$L^\infty(e)$	order
5°	1/32	1.425e-06	–	2.992e-04	–
	1/64	2.112e-07	2.75	9.806e-05	1.61
	1/128	5.287e-08	2.00	2.616e-05	1.91
	1/256	1.362e-08	1.96	8.284e-06	1.66
	1/512	3.656e-09	1.90	2.998e-06	1.47
20°	1/32	4.014e-06	–	4.139e-04	–
	1/64	1.106e-06	1.86	1.532e-04	1.43
	1/128	2.902e-07	1.93	5.226e-05	1.55
	1/256	7.837e-08	1.89	2.220e-05	1.24
	1/512	2.082e-08	1.91	8.742e-06	1.34
30°	1/32	7.347e-06	–	4.974e-04	–
	1/64	1.997e-06	1.88	1.996e-04	1.32
	1/128	5.384e-07	1.89	7.509e-05	1.41
	1/256	1.447e-07	1.90	2.935e-05	1.35
	1/512	3.884e-08	1.90	1.160e-05	1.34
40°	1/32	1.071e-05	–	4.482e-04	–
	1/64	2.657e-06	2.01	1.738e-04	1.37
	1/128	6.702e-07	1.99	6.100e-05	1.51
	1/256	1.699e-07	1.98	2.186e-05	1.48
	1/512	4.305e-08	1.98	8.439e-06	1.37

5.1.2. Advection in interior of circle

The next test is advection with a variable velocity field in the interior of a circle. The circle has radius 1.0. The velocity field is given by $(u, v)^T = (-2y, 2x)^T$. The initial condition is chosen as

$$s(x, y, 0) = 1 + \exp\left(-60 \cdot \left((x - 0.85)^2 + y^2\right)\right)$$

and the final time is $T = \pi$, when the test function is back to its original position. We again use unlimited gradients.

Figure 7 shows the solution at the final time for a run with $\Delta x = 1.43653/128$. The contour lines indicate that the shape of the test function has been very well preserved, including along the cut cells.

For this test, we expect the maximum error to be located at the peak of the test function, which has a distance of 0.85 from the circle midpoint and therefore lies close to the boundary but in the interior of the domain. To get a fair assessment of our method, we measure the L^1 and L^∞ error (i) over all cells, (ii) over cut cells only, and (iii) over transition cells only. The results for these errors measured in the L^∞ norm are shown in Figure 8. The lines correspond to the slopes of least squares fits. The precise numbers of the slopes are given in Table 3. For the least squares fit we include samples with grid sizes smaller than 10^{-2}. The results are similar to the ramp test: $L^1(e)$ converges with second order, while the maximum error measured over cut cells and transition cells converges with a rate that lies between 1 and 2.

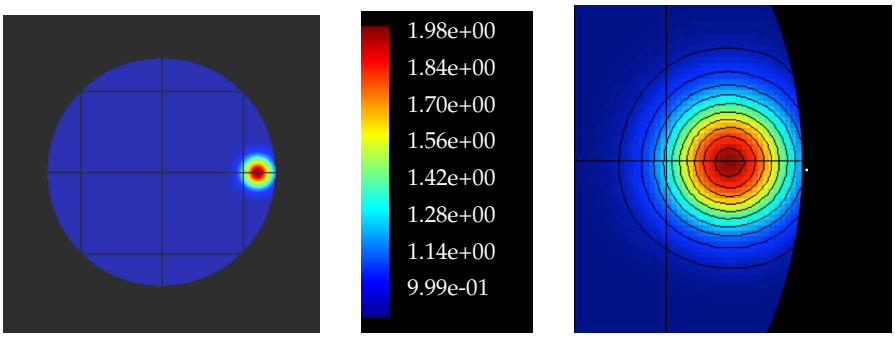


Figure 7: Circle test: solution at final time for mesh width $\Delta x = 1.43653/128$. *Left:* Full solution; *right:* zoom.

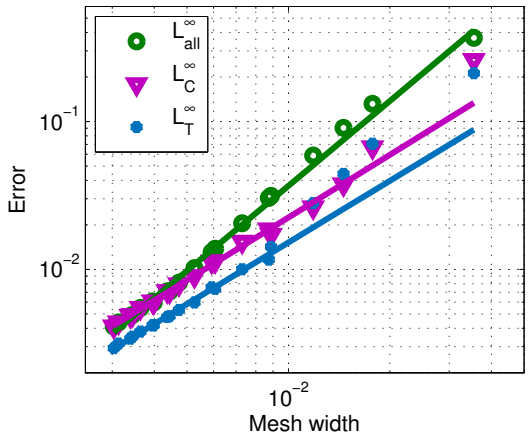


Figure 8: Circle test: L^∞ error measured over different sets of cells: measured over all cells (L^∞_{all}), measured over cut cells only (L^∞_C), and measured over transition cells only (L^∞_T).

Table 3: Circle test: convergence rates computed by a least squares fit involving all tests with mesh width $\Delta x \leq 10^{-2}$.

Error computed over:	L^1 error	L^∞ error
all cells	2.03	1.91
cut cells only	1.49	1.41
transition cells only	1.76	1.39

Table 4: Ramp test using discontinuous initial data: Maximum values measured over cut cells and transition cells only (C/T) and over fully explicitly treated Cartesian cells only (flow), respectively.

Δx	$\beta = 5°$		$\beta = 20°$		$\beta = 30°$		$\beta = 40°$	
	flow	C/T	flow	C/T	flow	C/T	flow	C/T
1/32	1.004	1.000	1.017	1.000	1.053	1.000	1.076	1.000
1/64	1.004	1.000	1.017	1.000	1.030	1.000	1.033	1.000
1/128	1.004	1.000	1.024	1.000	1.030	1.000	1.033	1.000
1/256	1.004	1.000	1.017	1.000	1.030	1.014	1.033	1.000
1/512	1.004	1.000	1.031	1.007	1.030	1.000	1.033	1.000

5.1.3. Discontinuity advected along ramp

In this section we compare the one-dimensional theory about the hybrid scheme's monotonicity from section 2.2 with two-dimensional numerical experiments of advection of a discontinuity along a ramp. Again we use the ramp indicated in Figure 6. The initial data is discontinuous with respect to the line perpendicular to the ramp: cells with centroids to the left of the line have the value 1 and to the right have value 0. We take 1 time step.

For this test we use MUSCL with MC limiter as explicit scheme and implicit Euler time stepping without slope reconstruction as implicit scheme, combined using flux bounding. This approximately imitates the setup of the TVD result reported in Theorem 2.1. It is well-known that advection in diagonal direction can lead to significant overshoot for second-order schemes if one-dimensional, limited slope reconstructions are used, see e.g. the discussion in [30]. Therefore, we do expect to see some overshoot. The goal of this test is to compare the lack of monotonicity over the cut/transition cells and the explicitly treated flow cells, by measuring their minimum and maximum values.

For the flow cells, we observe undershoots between 10^{-6} and 10^{-15}. The overshoots for various scenarios are reported in Table 4. The amount of overshoot depends on the angle. For a ramp angle of 5 degrees, when the flow is still fairly well aligned with the Cartesian coordinate directions, there is very little overshoot. For almost diagonal advection, there is significant overshoot even though we use the MC limiter.

More interesting to us is the behavior on cut and transition cells. We always observe a minimum value of zero up to machine precision. For the maximum value, the table shows a slight overshoot in only two cases. In both cases, the overshoot is caused by a transition cell. The maximum overshoot on only cut cells is 10^{-10} in all cases tested. These results indicate that the one-dimensional flux bounding theory carries over to two dimensions.

Next, we examine numerically another theoretical consideration from one dimension concerning the monotonicity (or lack thereof) of the trapezoidal time stepping (without slope reconstruction): we use 0-1 discontinuous data and place the jump directly in front of a very small cell. In the limit as $\alpha \to 0$ we would expect a maximum value of 2 in the small cell in the next time step [28]. To imitate that test in two dimensions, we use $\beta = 30°$ and $\Delta x = 1/512$ and place the discontinuity in front of a cell with $\alpha_{ij} = 5.0 \cdot 10^{-7}$. Taking one time step results in a value of 1.997 on that small cell, which is consistent with our one-dimensional theoretical considerations. For comparison, we repeat the test using implicit Euler time stepping (without slope reconstruction) as implicit scheme. For this setup, the overshoot on this tiny cell is 10^{-11}.

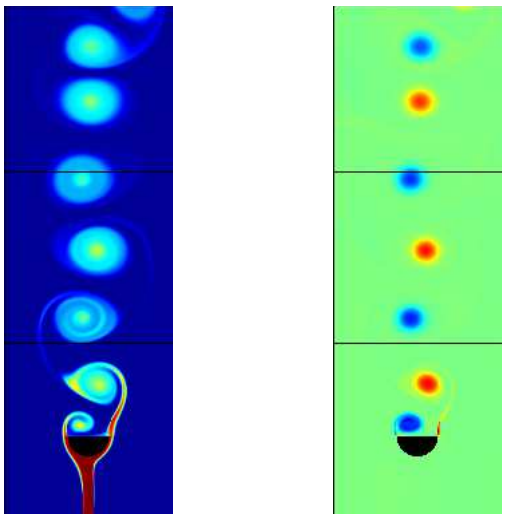


Figure 9: Passively advected scalar (left) and vorticity (right) for flow past a half-cylinder.

Overall, the results for this test indicate that for problems with discontinuous initial data and/or shocks, some additional procedure will be needed. One possibility is an FCT approach that combines trapezoidal time stepping with implicit Euler time stepping. Another possibility is to use a discontinuity detector [25] and only apply the more diffusive scheme at these isolated 'problem' cells.

5.1.4. Incompressible Euler equations

We conclude the section on two dimensional numerical results with an example involving the full incompressible Euler equations

$$\begin{aligned} \mathbf{u}_t + (\mathbf{u} \cdot \nabla)\mathbf{u} + \nabla p &= 0, \\ \nabla \cdot \mathbf{u} &= 0. \end{aligned}$$

The main incentive for developing the mixed explicit implicit time stepping scheme was to extend an existing solver for the incompressible Euler equations from Cartesian grids to Cartesian embedded boundary grids. In this test, we solve the full nonlinear equations, and use the mixed explicit implicit scheme for both update of the velocity field and the passively advected scalar.

In *Step 1* of the projection method [4, 2, 8, 9], a velocity field at cell edges is predicted at time $t^{n+1/2}$, and then a MAC-projection is applied to make the velocity field divergence-free (up to machine precision), resulting in U^M. Next, an equation of the form

$$\frac{U^* - U^n}{\Delta t} + \left[(U^M \cdot \nabla)U\right]^{n+1/2} + \nabla p^{n-1/2} = 0 \tag{11}$$

is solved for U^*, where the lagged pressure is treated as a source term. Then, in *Step 2*, U^{n+1} is derived from U^* by applying a projection.

Our extension of this algorithm to embedded boundary grids follows the same outline. When solving equation (11) on a cut cell grid, we face the small cell problem. We overcome this problem by using our new mixed explicit implicit scheme. Note that equation (11) is very

similar to the linear advection equation (8). Then, in *Step 2*, we apply a projection that has been adjusted to the presence of cut cells. More details can be found in [28].

We compute flow around a half-cylinder roughly following the test setup in [3]. The test domain is given by $[0, 8] \times [0, 24]$ and we cut out a half-cylinder with diameter $D = 2.0$. We use inflow and outflow boundary conditions at the low and high y boundary, respectively, and slip wall boundary conditions on both x boundaries. The inflow is given by $v = 1$. We use a projection step to compute the initial velocity field, and add a small perturbation of size 0.01 close to the inflow boundary to break the symmetry. In addition to the velocity field, we compute the behavior of a passively advected scalar which is advected in from the low y edge. We stop the simulation at time $T \approx 51.5$ when the flow is periodic.

Figure 9 shows the passively advected scalar variable on the left and the vorticity field on the right, using 128 cells in the x direction. For both the update of the velocity field and the advection of the scalar, we use the mixed explicit implicit time stepping scheme with limited slopes. To be more precise, we use MUSCL with MC limiter as explicit scheme and trapezoidal time stepping with LP limiter as implicit scheme. We did not observe stability issues.

5.2. Numerical results in three dimensions

This example is included to show that the ideas and results previously presented carry over to three dimensions. We compute the L^1 and L^∞ norm analogously to two dimensions. For this example we take the CFL number to be $\lambda = 0.9$, where the time step is computed using

$$\Delta t = \lambda \min \left(\frac{\Delta x}{\max_{i,j,k} |u_{i+1/2,j,k}|}, \frac{\Delta y}{\max_{i,j,k} |v_{i,j+1/2,k}|}, \frac{\Delta z}{\max_{i,j,k} |w_{i,j,k+1/2}|} \right).$$

We take $\Delta x = \Delta y = \Delta z$.

5.2.1. Advection along a ramp

Consider advection along a three-dimensional ramp. The velocity field is given by

$$(u, v, w) = (1, \tan(20^\circ), \tan(35^\circ)).$$

The geometry setup is explained in Figure 10: a plane is cut out of the bounding box $[0, 1]^3$. The plane is determined by the two vectors $\vec{a}$ and $\vec{b}$. We choose $\vec{a} = (u, v, w)$ to guarantee that the flow is parallel to the ramp. The vector $\vec{b}$ is arbitrary. We use $\vec{b} = (0, 1, \tan(15^\circ))$.

The exact solution is a quadratic moving tangent to the plane; the value of the quadratic depends only on the distance of a point (x, y, z) to the plane that passes through $(0, 0, 0)$ and has (u, v, w) as normal. This closely mimics the two dimensional ramp test, where all the error is at the boundary cells, since a pure MUSCL scheme is exact for quadratics.

Figure 11 shows the error measured both in the L^1 and the L^∞ norm over the whole domain. The slopes of the solid lines correspond to convergence orders computed by a least squares fit and are given by 1.92 and 1.33, respectively. This confirms that our new scheme is second-order accurate in L^1 and between first- and second-order accurate in L^∞.

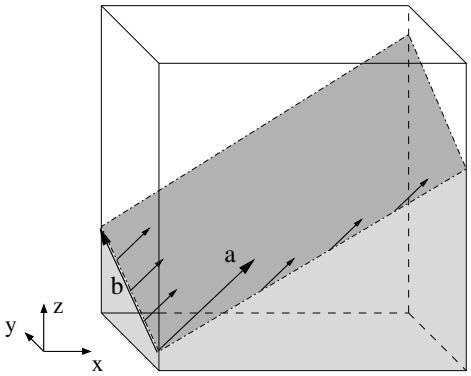


Figure 10: Setup of the ramp test in 3d: the (darker) grey plane is cut out of the bounding box $[0,1]^3$. The flow domain corresponds to area above the plane. The small arrows indicate the flow direction. Both the low x face and low y face of the box are inflow boundaries.

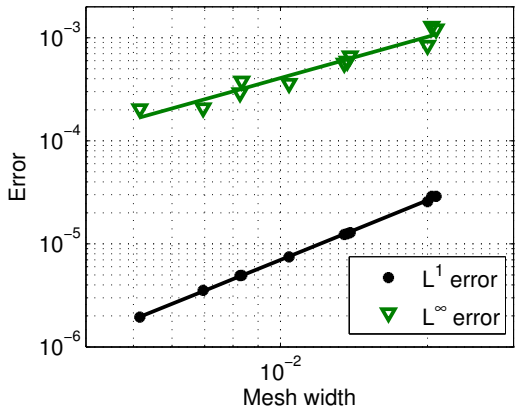


Figure 11: Ramp test: Error plots for the L^1 and the L^∞ error over the whole domain. The slopes of the solid lines are computed by a least squares fit and are given by 1.92 and 1.33, respectively.

6. Conclusions and Future Plans

We have presented a mixed explicit implicit time stepping scheme for overcoming the small cell problem for the linear advection equation. The scheme uses an implicit scheme, the trapezoidal rule with slope reconstruction, on cut cells and the explicit MUSCL scheme on cells away from boundary. The change in scheme happens in transitions cells – Cartesian cells that share at least one edge/face with a cut cell. Transition cells use implicit fluxes for edges / faces shared with cut cells and explicit fluxes otherwise, a technique that we call *flux bounding* and that is conservative by construction. Numerical results in two and three dimensions show that the resulting scheme is second-order accurate in L^1 and between first- and second-order accurate along the embedded boundary, and quite stable. We plan to investigate improvements in the discretization at the cut cells, where by using a higher order reconstruction, and possibly a higher order integration rule for the fluxes, a fully second order scheme will result. We also have ideas for how to improve the convergence rate on transition cells.

In one dimension we can show a TVD result for combining MUSCL with implicit Euler time stepping by means of flux bounding. There is no such result using trapezoidal time stepping with slope reconstruction. Theoretical considerations in one dimension and numerical tests in two dimensions confirm that for computations involving discontinuities and/or shocks, an additional procedure such as an FCT-type approach that combines implicit Euler time stepping with the second-order trapezoidal rule will be necessary. We plan to pursue this direction in the near future as it would be an essential step towards extending this method from the linear advection equation to the compressible Euler equations. We will also continue extending the incompressible flow solver from Cartesian grids to cut cell meshes. The most important step is extending the projection step in three dimensions to cut cells.

Acknowledgments

The authors would like to thank Ann Almgren, John Bell, and Andy Nonaka from Lawrence Berkeley National Laboratory for providing and helping the authors with the software packages BoxLib and VarDen, as well as for helpful discussions. This work was supported in part by the DOE office of Advanced Scientific Computing under grant DE-FG02-88ER25053 and by AFOSR grant FA9550-13-1-0052. S. M. was also supported by ERC STG. N 306279, SPARCCLE.

A. Proof of Theorem 2.1

Theorem A.1 (Restatement of Theorem 2.1)**.** *The scheme* (6) is TVD *for the linear advection equation* (1) *for* $0 \leq \lambda \leq 1$*, if the exact solution has compact support.*

Proof. We cannot directly apply Harten's theorem [21, 39] to show that the scheme is TVD, but we imitate its proof in order to show

$$\sum_i |s_{i+1}^{n+1} - s_i^{n+1}| \leq \sum_i |s_{i+1}^n - s_i^n|. \tag{12}$$

We split the sum on the left hand side in the following way

$$\sum_i |s_{i+1}^{n+1} - s_i^{n+1}| = \sum_{i\le -3} |s_{i+1}^{n+1} - s_i^{n+1}| + \sum_{i\ge 2} |s_{i+1}^{n+1} - s_i^{n+1}| \\ + |s_{-1}^{n+1} - s_{-2}^{n+1}| + |s_0^{n+1} - s_{-1}^{n+1}| + |s_1^{n+1} - s_0^{n+1}| + |s_2^{n+1} - s_1^{n+1}|. \quad (13)$$

We estimate the sums $\sum_{i\le -3}|s_{i+1}^{n+1} - s_i^{n+1}|$ and $\sum_{i\ge 2}|s_{i+1}^{n+1} - s_i^{n+1}|$ in the first line (whose behavior is dominated by the explicit scheme) and the four terms in the second line (whose behavior is dominated by the implicit scheme) separately.
Terms involving the explicit scheme: The MUSCL scheme can be written as

$$s_i^{n+1} = s_i^n - C_{i-1/2}^n \left(s_i^n - s_{i-1}^n\right)$$

with

$$C_{i-1/2}^n = \lambda \frac{s_i^n + (1-\lambda)s_{x,i}^n \frac{\Delta x}{2} - s_{i-1}^n - (1-\lambda)s_{x,i-1}^n \frac{\Delta x}{2}}{s_i^n - s_{i-1}^n},$$

which satisfy $0 \le C_{i-1/2}^n \le 1$ for the chosen limiter. This relation holds on all cells with indices $i \le -2$ or $i \ge 2$ (with $s_{x,-2}^n = s_{x,1}^n = 0$). This implies for $i \le -3$ and $i \ge 2$ the relation

$$|s_{i+1}^{n+1} - s_i^{n+1}| \le (1 - C_{i+1/2}^n)|s_{i+1}^n - s_i^n| + C_{i-1/2}^n |s_i^n - s_{i-1}^n|.$$

Taking the compact support into account, there holds

$$\sum_{i\le -3} |s_{i+1}^{n+1} - s_i^{n+1}| \le \sum_{i\le -3} |s_{i+1}^n - s_i^n| - C_{-5/2}^n |s_{-2}^n - s_{-3}^n| \quad (14)$$

and

$$\sum_{i\ge 2} |s_{i+1}^{n+1} - s_i^{n+1}| \le \sum_{i\ge 2} |s_{i+1}^n - s_i^n| + C_{3/2}^n |s_2^n - s_1^n|. \quad (15)$$

Terms involving the implicit scheme: To estimate the remaining four terms in the second line of equation (13) in terms of data at time t^n we exploit the convexity of the implicit scheme. We write the updates for s_{-1}^{n+1} and s_0^{n+1} as

$$s_{-1}^{n+1} = \lambda_{-1} s_{-1}^n + (1-\lambda_{-1}) s_{-2}^n \quad \text{and} \quad s_0^{n+1} = \lambda_0 s_0^n + (1-\lambda_0) s_{-1}^{n+1}$$

with $\lambda_{-1} = 1/(1+\lambda)$ and $\lambda_0 = 1/(1+\lambda/\alpha)$, $\lambda_{-1}, \lambda_0 \in (0,1]$. Then we have the following estimates: for the difference $|s_2^{n+1} - s_1^{n+1}|$, we get

$$|s_2^{n+1} - s_1^{n+1}| = |(1 - C_{3/2}^n)s_2^n + C_{3/2}^n s_1^n - (1-\lambda)s_1^n - \lambda s_0^{n+1}| \\ \le (1 - C_{3/2}^n)|s_2^n - s_1^n| + \lambda |s_1^n - s_0^{n+1}|.$$

For the difference $|s_1^{n+1} - s_0^{n+1}|$, there holds

$$|s_1^{n+1} - s_0^{n+1}| = |(1-\lambda)s_1^n + \lambda s_0^{n+1} - s_0^{n+1}| = (1-\lambda)|s_1^n - s_0^{n+1}|$$

with

$$|s_1^n - s_0^{n+1}| = |s_1^n - \lambda_0 s_0^n - (1-\lambda_0)s_{-1}^{n+1}| \le |s_1^n - s_0^n| + (1-\lambda_0)|s_0^n - s_{-1}^{n+1}|.$$

For the difference $|s_0^{n+1} - s_{-1}^{n+1}|$, we get

$$|s_0^{n+1} - s_{-1}^{n+1}| = |\lambda_0 s_0^n + (1-\lambda_0)s_{-1}^{n+1} - s_{-1}^{n+1}| = \lambda_0 |s_0^n - s_{-1}^{n+1}|.$$

This implies

$$\begin{aligned} &|s_2^{n+1} - s_1^{n+1}| + |s_1^{n+1} - s_0^{n+1}| + |s_0^{n+1} - s_{-1}^{n+1}| \\ &\qquad \leq (1 - C_{3/2}^n)|s_2^n - s_1^n| + |s_1^n - s_0^{n+1}| + \lambda_0 |s_0^n - s_{-1}^{n+1}| \\ &\qquad \leq (1 - C_{3/2}^n)|s_2^n - s_1^n| + |s_1^n - s_0^n| + |s_0^n - s_{-1}^{n+1}|. \end{aligned}$$

Finally, there holds

$$|s_0^n - s_{-1}^{n+1}| = |s_0^n - \lambda_{-1} s_{-1}^n - (1-\lambda_{-1})s_{-2}^n| \leq |s_0^n - s_{-1}^n| + (1-\lambda_{-1})|s_{-1}^n - s_{-2}^n|$$

as well as

$$\begin{aligned} |s_{-1}^{n+1} - s_{-2}^{n+1}| &= |\lambda_{-1} s_{-1}^n + (1-\lambda_{-1})s_{-2}^n - (1 - C_{-5/2}^n)s_{-2}^n - C_{-5/2}^n s_{-3}^n| \\ &\leq \lambda_{-1}|s_{-1}^n - s_{-2}^n| + C_{-5/2}^n |s_{-2}^n - s_{-3}^n|. \end{aligned}$$

To summarize, we get

$$\begin{aligned} &|s_2^{n+1} - s_1^{n+1}| + |s_1^{n+1} - s_0^{n+1}| + |s_0^{n+1} - s_{-1}^{n+1}| + |s_{-1}^{n+1} - s_{-2}^{n+1}| \\ &\qquad \leq (1 - C_{3/2}^n)|s_2^n - s_1^n| + |s_1^n - s_0^n| + |s_0^n - s_{-1}^n| + |s_{-1}^n - s_{-2}^n| + C_{-5/2}^n |s_{-2}^n - s_{-3}^n|. \end{aligned}$$

Therefore, we have estimated all terms in (13). Putting the results in equations (14) and (15) as well as our results for the implicit terms together implies the claim (12). □

References

[1] M. J. Aftosmis, M. J. Berger, and J. E. Melton. Robust and efficient Cartesian mesh generation for component-based geometry. *AIAA Journal*, 36:952–960, 1998.

[2] A. S. Almgren, J. B. Bell, P. Colella, L. H. Howell, and M. L. Welcome. A conservative adaptive projection method for the variable density incompressible Navier-Stokes equations. *J. Comput. Phys.*, 142:1–46, 1998.

[3] A. S. Almgren, J. B. Bell, P. Colella, and T. Marthaler. A Cartesian grid projection method for the incompressible Euler equations in complex geometries. *SIAM J. Sci. Comput*, 18:1289–1309, 1997.

[4] A. S. Almgren, J. B. Bell, and W. G. Szymczak. A numerical method for the incompressible Navier-Stokes equations based on an approximate projection. *SIAM J. Sci. Comput.*, 17:358–369, 1996.

[5] T.J. Barth. A 3-d least-squares upwind Euler solver for unstructured meshes. In M. Napolitano and F. Sabetta, editors, *Thirteenth International Conference on Numerical Methods in Fluid Dynamics*, volume 414 of *Lecture Notes in Physics*, pages 240–244. Springer, 1993.

[6] S. A. Bayyuk, K. G. Powell, and B. van Leer. A simulation technique for 2-d unsteady inviscid flows around arbitrarily moving and deforming bodies of arbitrary geometry. *AIAA Paper 93-2991-CP*, 1993.

[7] J. B. Bell, A. Almgren, V. Beckner, M. Day, M. Lijewski, A. Nonaka, and W. Zhang. BoxLib User's Guide. Technical report, CCSE, Lawrence Berkeley National Laboratory, 2012. `https://ccse.lbl.gov/BoxLib/BoxLibUsersGuide.pdf`.

[8] J. B. Bell, P. Colella, and H. M. Glaz. A second order projection method for the incompressible Navier-Stokes equations. *J. Comput. Phys.*, 85:257–283, 1989.

[9] J. B. Bell, P. Colella, and L. H. Howell. An efficient second-order projection method for viscous incompressible flow. In *Proceedings of the Tenth AIAA Computational Fluid Dynamics Conference*, pages 360–367. AIAA, 1991.

[10] M. J. Berger. A note on the stability of cut cells and cell merging. *Appl. Numer. Math*, 96:180–186, 2015.

[11] M. J. Berger, C. Helzel, and R. LeVeque. H-Box method for the approximation of hyperbolic conservation laws on irregular grids. Technical report, Courant Mathematics and Computing Laboratory, New York University, 2002. Technical Report 02-001.

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

[13] J. P. Boris and D. L. Book. Flux corrected transport. I. SHASTA, a fluid transport algorithm that works. *J. Comput. Phys.*, 11:38–69, 1973.

[14] `http://people.nas.nasa.gov/~aftosmis/cart3d/`.

[15] 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.

[16] P. Colella. A direct Eulerian MUSCL scheme for gas dynamics. *SIAM J. Sci. Stat. Comput.*, 6:104–117, 1985.

[17] P. Colella. Multidimensional upwind methods for hyperbolic conservation laws. *J. Comput. Phys.*, 87:171–200, 1990.

[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.*, 211:347–366, 2006.

[19] J.P. Collins, P. Colella, and H.M. Glaz. An implicit-explicit Eulerian Godunov scheme for compressible flow. *J. Comput. Phys.*, 116:195–211, 1995.

[20] B. Gustafsson. The convergence rate for difference approximations to mixed initial boundary values problems. *Math. Comp.*, 29:396–406, 1975.

[21] A. Harten. High resolution schemes for hyperbolic conservation laws. *J. Comput. Phys.*, 49:357–393, 1983.

[22] C. Helzel, M. J. Berger, and R. LeVeque. A high-resolution rotated grid method for conservation laws with embedded geometries. *Siam J. Sci. Comput.*, 26:785–809, 2005.

[23] S. Jebens, O. Knoth, and R. Weiner. Partially implicit peer methods for the compressible Euler equations. *J. Comput. Phys.*, 230:4955–4974, 2011.

[24] D. I. Ketcheson, C. B. Macdonald, and S. Gottlieb. Optimal implicit strong stability preserving Runge-Kutta methods. *Appl. Numer. Math.*, 59:373–392, 2009.

[25] L. Krivodonova, J. Xin, J.-F. Remacle, and J.E. Flaherty. Shock detection and limiting with discontinuous Galerkin methods for hyperbolic conservation laws. *Appl. Numer. Math*, 48:323–338, 2004.

[26] R. J. LeVeque. *Finite Volume Methods for Hyperbolic Problems*. Cambridge University Press, 2002.

[27] D. Mavriplis. Unstructured grid techniques. *Annu. Rev. Fluid. Mech.*, 29:473–514, 1997.

[28] S. May. *Embedded Boundary Methods for Flow in Complex Geometries*. PhD thesis, Courant Institute of Mathematical Sciences, New York University, 2013.

[29] S. May and M. J. Berger. Two-dimensional slope limiters for finite volume schemes on non-coordinate-aligned meshes. *SIAM J. Sci. Comput.*, 35:A2163–A2187, 2013.

[30] S. May, A. J. Nonaka, A. S. Almgren, and J. B. Bell. An unsplit, higher order Godunov method using quadratic reconstruction for advection in two dimensions. *Comm. App. Math. and Comp. Sci.*, 6:27–61, 2011.

[31] Karol Mikula, Mario Ohlberger, and Jozef Urbán. Inflow-implicit/outflow-explicit finite volume methods for solving advection equations. *Appl. Numer. Math*, 85:16–37, 2014.

[32] K.W. Morton. On the analysis of finite volume methods for evolutionary problems. *SIAM J. Numer. Anal.*, 35:2195–2222, 1998.

[33] A. J. Nonaka, S. May, A. S. Almgren, and J. B. Bell. A three-dimensional, unsplit Godunov method for scalar conservation laws. *SIAM J. Sci. Comput.*, 33:2039–2062, 2011.

[34] 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.*, 120:278–304, 1995.

[35] Tom Pulliam. Solution methods in computational fluid dynamics. Based on Lecture Notes for the von Karman Institute for Fluid DYnamics, 1985.

[36] J. J. Quirk. An alternative to unstructured grids for computing gas dynamic flows around arbitrarily complex two-dimensional bodies. *Comput. & Fluids*, 23:125–142, 1994.

[37] J. Saltzman. An unsplit 3d upwind method for hyperbolic conservation laws. *J. Comput. Phys.*, 115:153–168, 1994.

[38] M.N. Spijker. Contractivity in the numerical solution of initial value problems. *Numer. Math.*, 42:271–290, 1983.

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

[40] B. van Leer. Towards the ultimate conservative difference scheme. V. A second order sequel to Godunov's methods. *J. Comput. Phys.*, 32:101–136, 1979.

[41] B. Wendroff and A. B. White. A supraconvergent scheme for nonlinear hyperbolic systems. *Computers Math. Applic*, 18:761–767, 1989.

[42] S. T. Zalesak. Fully multidimsional flux-corrected transport algorithms for fluids. *J. Comput. Phys.*, 31:335–362, 1979.

Recent Research Reports

Nr.	Authors/Title

2015-34 H. Ammari and M. Ruiz and S. Yu and H. Zhang
Mathematical analysis of plasmonic resonances for nanoparticles: the full Maxwell equations

2015-35 H. Ammari and J.K. Seo and T. Zhang
Mathematical framework for multi-frequency identification of thin insulating and small conductive inhomogeneities

2015-36 H. Ammari and Y.T. Chow and J. Zou
Phased and phaseless domain reconstruction in inverse scattering problem via scattering coefficients

2015-37 A. Kunoth and Ch. Schwab
Sparse adaptive tensor Galerkin approximations of stochastic PDE-constrained control problems

2015-38 P. Grohs and R. Hiptmair and S. Pintarelli
Tensor-product discretization for the spatially inhomogeneous and transient Boltzmann equation in 2D

2015-39 T. Zimmermann and S. Mishra and B. Doran and D. Gordon and A. Landsman
Tunneling time and weak measurement in strong field ionisation

2015-40 R. Kaeppeli and S. Mishra
A well-balanced finite volume scheme for the Euler equations with gravitation

2015-41 A. Perego and R. Cabezón and R. Käppeli
An advanced leakage scheme for neutrino treatment in astrophysical simulations