Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Doctoral Thesis

Parallel Numerical Methods for Conservation Laws

Jayesh Badwaik
Department of Mathematics
Chair of Mathematics VI (Mathematical Fluid Mechanics)

Prof. Dr. Christian Klingenberg
Advisor

Parallel Numerical Methods for Conservation Laws

- Doctoral Thesis -

Submitted by

Jayesh Badwaik

Würzburg, May 21, 2024

Julius-Maximilians-Universität Würzburg

Faculty of Mathematics and Computer Science

It takes a village to raise a child.

This thesis would not have been possible without all the people who got up one day

and for some reason decided to help out this starry-eyed child

To my family, friends, teachers, colleagues, and mentors...

I'm forever grateful

Acknowledgements

I would like to deeply thank my advisor Prof. Dr. Christian Klingenberg for his support and trust in me, from accepting me as his student to guiding me through the very end of my thesis, especially in the initial and last years of the thesis. I appreciate the freedom he gave me to explore my ideas and the discussions we had on various topics related to the thesis.

Next, I would like to thank my other mentors, Prof. Praveen Chandrashekar, for his extensive support with my research and introducing me to the field of discontinuous Galerkin method on moving meshes. He always had an open door for even the simplest of my questions and showed great care about his students and the work they were doing.

I would like to thank Prof. Nils Henrik Risebro for introducing me to the problem of numerical methods for porous media flows and a lot of help that he provided me during my time in Oslo. In Oslo, I also had an extremely enjoyable and fruitful collaboration with Dr. Adrian M. Ruf on the topic of porous media flows. Without him, this thesis would not have been in the shape it is today.

I would like to thank Prof. Phillip Helluy for introducing me to an extremely interesting topic of Palindromic discontinuous Galerkin methods for kinetic methods and for the collaboration on the topic.

Finally I would like to thank the following people: Andrea Thomann, Gero Schnücke, Juan Pablo Gallego Valencia, Markus Zenk, Farah Kanbar and Jens Klotzky for being the best group to start my PhD with. I could not have asked for a more supportive and fun group to start my PhD with. I would also like to thank the Jonas Berberich, Sandra Warnecke, Simon Markfelder and Marlies Pirner for the great support during the PhD. My time in Oslo would not have been as good as it was without my former roommate and also a classmate at times, Neelabja Chatterjee.

Contents

List of Publications

This thesis is based on the following papers:

1. Jayesh Badwaik and Adrian M Ruf. “Convergence rates of monotone schemes for conservation laws with discontinuous flux.” In: **SIAM Journal on Numerical Analysis** 58.1 (2020), pp. 607–629

   This paper (linked here) is introduced in Chapter 2

2. Jayesh Badwaik, Christian Klingenberg, Nils Henrik Risebro, and Adrian M Ruf. “Multilevel Monte Carlo finite volume methods for random conservation laws with discontinuous flux.” In: **ESAIM: Mathematical Modelling and Numerical Analysis** 55.3 (2021), pp. 1039–1065

   This paper (linked here) is introduced in Chapter 3

3. Jayesh Badwaik, Praveen Chandrashekar, and Christian Klingenberg. “Single-step arbitrary Lagrangian--Eulerian discontinuous Galerkin method for 1-d Euler equations.” In: **Communications on Applied Mathematics and Computation** 2 (2020), pp. 541–579

   This paper (linked here) is introduced in Chapter 5 and Chapter 6

4. Jayesh Badwaik and Praveen Chandrashekar. “Arbitrary Lagrangian--Eulerian Discontinuous Galerkin Method for 1D Euler Equations.” In: **Theory, Numerics and Applications of Hyperbolic Problems I: Aachen, Germany, August 2016**. Springer. 2018, pp. 323–334

   This paper (linked here) is introduced in Chapter 5 and Chapter 6

5. Jayesh Badwaik, Matthieu Boileau, David Coulette, Emmanuel Franck, Philippe Helluy, Christian Klingenberg, Laura Mendoza, and Herbert Oberlin. “Task-based parallelization of an implicit kinetic scheme.” In: **ESAIM: Proceedings and Surveys** 63 (2018), pp. 60–77

   This paper (linked here) is introduced in Chapter 7

Part I.

Porous Media Flows

Chapter 1 Introduction

In this part, we consider the conservation laws with discontinuous fluxes described in eq. (1.1). Here, $u\colon \mathbb{R}\times[0,\infty)\to\mathbb{R}$ is the unknown scalar quantity, $f(k,x)\colon \mathbb{R}^2\to\mathbb{R}$ is the flux function. The function $k\colon \mathbb{R}\to\mathbb{R}$ is a spatially varying co-efficient which can exhibit discontinuity in the spatial domain x. $u_0\colon \mathbb{R}\to\mathbb{R}$ is the initial datum for the problem at time $t=0$.

$$\frac{\partial u}{\partial t}+\frac{\partial f(k(x),u)}{\partial x}=0,\quad x\in\mathbb{R} \text{ and } \quad t>0 \tag{1.1a}$$

$$u(x,0)=u_0(x),\quad x\in\mathbb{R} \tag{1.1b}$$

Conservation laws with discontinuous fluxes as described in (1.1) are of great interest in several areas of physics and engineering. In particular, they arise in modeling traffic flow on highways with changing road conditions [LW55], polymer flooding in oil recovery [She17], two-phase flow through heterogeneous porous media [GR92; GR93; RT91] and sedimentation processes [Bür+03; Die96].

Even in the absence of flux discontinuities, solutions of (1.1) develop discontinuities in finite time and for this reason, weak solutions are sought. Weak solutions to (1.1) are not unique, hence, we use additional entropy conditions. For a smooth $x\mapsto f(k(x),u)$, the uniqueness follows from the classical Kruzkov entropy conditions.

In the presence of spatial flux discontinuities, Kruzkov entropy conditions are not enough. This difficulty is usually resolved by requiring that the Kruzkov entropy conditions hold away from the spatial flux discontinuities and imposing additional jump conditions along the spatial interfaces [AKR11; GR91; KR95] or by adapting the Kruzkov entropy conditions in a suitable way [AP05; BJ97; PT18; Ruf24; Tow20].

In the last two decades, there has been a large interest in the numerical approximation of entropy solutions of (1.1) under various assumptions on the flux function f and the co-efficient k. We refer to [Tow00] for a partial list of references regarding finite volume methods and the front tracking methods respectively. Specifically, in the adapted entropy framework, we want to highlight the results of [BR20; GJT20; Tow20] and [BJ97; PT18; Ruf24] regarding finite volume methods and front tracking methods.

The classical paradigm for designing efficient numerical schemes assumes that data for (1.1) are known exactly. In particular, the assumptions is that the values of u_0, the flux f and the spatial dependency co-efficient k are known exactly. However, in many situations of interest, there is an inherent uncertainty in the data due to uncertainty in experimental data. For example, in the case of two-phase flow through heterogeneous porous medium, the position of the interface between two rock types is typically not known exactly. Often, these parameters are only known up to certain statistical quantities like the mean, the variance and the higher moments.

The primary objective of the work done in this part of the thesis was to develop a fast and robust numerical algorithm for quantifying the uncertainty in the solutions of (1.1) due to random data. First, we need to find a useful setting in which the solution to the underlying deterministic problem satisfies the requirements for an uncertainty quantification framework to exist. In particular, we need to ensure that the solution exists, is unique, and is stable with respect to the modeling parameters. We also need to ensure that that there exists numerical method used to solve the deterministic problem has a provable convergence rate.

Subsequently, we need to develop a mathematical framework for quantifying the uncertainty in the for quantifying the uncertainty in the solutions. Finally, we need to develop the numerical algorithm for approximating the mean of the random entropy solution, prove that it is convergent and quantify the computational cost of the algorithm.

The setting that we have chosen for the deterministic problem is the conservation law (1.1) where the flux function $f\colon \mathbb{R}^2 \to \mathbb{R}$ is assumed to be a smooth function, $f(k,u) \in \mathcal{C}^2(\mathbb{R}^2;\mathbb{R})$ with the additional condition that $\mathcal{D}_u f > \alpha > 0$ for all u. The co-efficient $k\colon \mathbb{R} \to \mathbb{R}$ is assumed to be a piecewise constant function with finitely many discontinuities in the spatial domain and the initial datum $u_0(x) \in (\mathrm{L}^\infty \cap \mathrm{BV})(\mathbb{R})$.

We want to emphasize that, to our knowledge, the framework of adapted entropy solutions and more specifically the setting of (1.1) is currently the only setting for conservation laws with discontinuous flux where we simultaneously have existence[Tow20], uniqueness[AP05], stability [Ruf24] and provable convergence rates for the numerical methods [BR20].

The rest of the part is organized as follows. In chapter 2, we lay out the setting for the deterministic problem (1.1) and prove that all monotone finite volume methods with the upwind property which obey the discrete Rankine-Hugoniot condition across the discontinuities of k converge at a rate of $\sqrt{\Delta x}$ to the unique entropy solution of the conservation law (1.1).

In chapter 3, we develop the mathematical framework for quantifying the uncertainty in the solutions of conservation laws with discontinuous flux with random discontinuous spatial dependency. And, then we develop a multilevel combination of Monte Carlo (MC) sampling and a pathwise Finite Volume Method (FVM) method to approximate the mean of the random entropy solution. We prove the convergence rates for the Monte Carlo finite volume method (MC FVM) and multi-level Monte Carlo finite volume method (MLMC FVM) methods and optimize the MLMC FVM method.

Chapter 2 Error Estimates for Monotone Schemes for Discontinuous Flux

In this chapter, we sketch the arguments of [BR20] (linked here) and prove that all monotone finite volume methods for scalar conservation laws with discontinuous flux, which have the upwind property and obey the discrete Rankine-Hugoniot condition across the discontinuities, converge at a rate of $\sqrt{\Delta x}$ to the unique entropy solution of the conservation law.

In Section 2.1, we lay out the setting in which we want to solve the problem. Then, we describe the numerical scheme that we use to solve the problem in Section 2.2. In Section 2.3, we use the Rankine-Hugoniot condition at the discontinuities to show that solving the problem is equivalent to solving a decomposition of the problem into finitely many initial-boundary value problems.

In Section 2.4, Section 2.5 and Section 2.6, we prove convergence rates for each of the three kinds of initial-boundary value problems that we find in the decomposition. Here, the novel aspect of our proof is the strong bound on temporal total variation of the finite volume approximation. The strong bound allows us to estimate the boundary terms at the discontinuities that appear when applying the classical Kuznetsov theory to the problem. Finally, we use the translation invariance of the problem to combine the results for all the initial-boundary value problems to prove the main result in Section 2.7. We validate the main result by conducting numerical experiments in Section 2.8.

2.1. The Problem Statement

We consider the conservation law (2.1)

$$\frac{\partial u}{\partial t} + \frac{\partial f(k(x), u)}{\partial x} = 0, \quad x \in \mathbb{R} \text{ and } \quad t > 0 \tag{2.1a}$$

$$u(x, 0) = u_0(x), \quad x \in \mathbb{R} \tag{2.1b}$$

where the flux function $f\colon \mathbb{R}^2 \to \mathbb{R}$ is smooth, that is , $f(k, u) \in \mathcal{C}^2(\mathbb{R}^2; \mathbb{R})$ with the additional condition that it is strictly monotone $\mathcal{D}_u f > \alpha > 0$ for all u. The co-efficient $k\colon \mathbb{R} \to \mathbb{R}$ is a piecewise constant function with finitely many discontinuities in the spatial domain. The initial datum is integrable, bounded and of finite total variation, that is, $u_0(x) \in (\mathrm{L}^\infty \cap \mathrm{BV})(\mathbb{R})$.

We assume that there are N discontinuities of k denoted by $\xi_0 < \xi_1 < \dots < \xi_N - 1$. The interval between two adjacent discontinuities is denoted by $D_i = (\xi_i, \xi_{i+1})$. We fix $\xi_0 = -\infty$ and $\xi_{N-1} = \infty$. k_i is the constant value of k on the interval D_i. Due to the piecewise constant nature of k, we can define the set of functions $f^{(i)}$ as in (2.2). Furthermore, due to the strict monotonicity of f, we can define the inverse function $(f^i)^{-1}$ for all $u \in \mathbb{R}$.

$$f^{(i)}(u) = f(k_i, u) \quad \text{for } u \in \mathbb{R} \text{ and } i = 0, 1, \dots, N \tag{2.2}$$

We define the function $H_i(u)$ over the domain D_i as in (2.3). The function $H_i(u)$ is motivated from what an adapted entropy function would be for a conservation law where the discontinuities are at the boundary of the domain D_i.

$$
\begin{aligned}
H_i(u,c_i) = \quad & \int_0^T \int_{D_i} \Big[|u-c_i|\,\varphi_t + \operatorname{sgn}(u-c_i)\left(f^i(u) - f^i(c_i)\right)\varphi_x \Big]\,\mathrm{d}x\mathrm{d}t \\
& - \int_{D_i} |u(x,T)-c_i|\,\varphi(x,T)\mathrm{d}x + \int_{D_i} |u_0(x)-c_i|\,\varphi(x,i)\mathrm{d}x \\
& - \int_0^T \operatorname{sgn}\Big(u\left(\xi_{i+1}^-,t\right)-c_i\Big)\Big[f^i\left(u\left(\xi_{i+1}^-,t\right)\right) - f^i(c_i)\Big]\,\varphi\left(\xi_{i+1},t\right)\mathrm{d}t \\
& + \int_0^T \operatorname{sgn}\Big(u\left(\xi_i^+,t\right)-c_i\Big)\Big[f^i\left(u\left(\xi_i^+,t\right)\right) - f^i(c_i)\Big]\,\varphi\left(\xi_i,t\right)\mathrm{d}t
\end{aligned}
\tag{2.3}
$$

Definition 2.1 *(Entropy Solution).* We say that $u \in C^0([0,\infty);\mathrm{L}^\infty(\mathbb{R}))$ is an entropy solution of (2.1), if for all $c \in \mathbb{R}$ and for all non-negative $\varphi \in C^\infty(\mathbb{R}\times[0,T])$, the entropy solution satisfies (2.4).

$$
\sum_{i=0}^{N} H_i(u,c_i) \geq 0 \tag{2.4}
$$

$$
c_0 = c \qquad\qquad c_{i+1} = \left(f^{(i+1)}\right)^{-1}\left(f^i(c_i)\right) \quad \text{for } i = 0,1,\dots,N-1 \tag{2.5}
$$

Theorem 2.1 *(Existence and Uniqueness of Entropy Solution).* Due to [BJ97] and [AP05], there exists a unique entropy solution $u \in \mathcal{C}([0,\infty);\mathrm{L}^1(\mathbb{R}))$ of (2.1) in the framework of adapted entropies.

Theorem 2.2 *(Rankine-Hugoniot Condition).* Due to [AKR11, Remark 2.3], we can show that there exists strong traces $u(\xi_i^\pm)$ of the entropy solution u at the discontinuities ξ_i and that they satisfy the Rankine-Hugoniot condition across ξ_i which gives us (2.6). We will later use the this result in Section 2.3 to decompose the problem into finitely many initial-boundary value problems.

$$
f^{i-1}\left(u(\xi_i^-,t)\right) = f^i\left(u(\xi_i^+,t)\right) \quad i = 0,1,\dots,N \tag{2.6}
$$

2.2. The Numerical Scheme

We discretize the domain $\mathbb{R}\times[0,T]$ using the spatial and temporal grid discretization parameters Δx and Δt respectively. The resulting grid cells then are $C_j = [x_{j-1/2}, x_{j+1/2}]$ and $C^n = [t_{n-1}, t_n)$ respectively for points $x_{j+\frac{1}{2}}$ such that $x_{j+\frac{1}{2}} - x_{j-\frac{1}{2}} = \Delta x$ for all $j \in \mathbb{Z}$ and $t_n - t_{n-1} = \Delta t$ for $n = 0,1,\dots,M+1$. The final time of simulation T is then given by $T = (M+1)\Delta t$. Further, we denote the spatio-temporal domain $C_j \times C^n$ by C_j^n.

Without loss of generality, we can assume that the grid is uniform on the whole real line. Furthermore, we can assume that grid is aligned in such a way that the discontinuities of k lie on the cell interfaces. That is $\xi_i = P_{i-\frac{1}{2}}$ for some integers P_i, $i = 1,2,\dots,N$.

We will consider two point numerical fluxes $F(u,v)$ that have the upwind property such that if $D_u f \geq 0$, then $F(u,v) = f(u)$. The upwind flux, the Godunov flux and the Engquist-Osher flux are some of the example of such fluxes. We are now ready to define our numerical scheme.

Definition 2.2 *(Numerical Scheme)***.** Given a discretization as mentioned above, we define the numerical scheme as in (2.7) for all $n \geq 0$ and $0 \leq i \leq N-1$

$$u_j^{n+1} = u_j^n - \lambda \left[f^{(i)} \left(u_j^n \right) - f^{(i)} \left(u_{j-1}^n \right) \right] \qquad P_i < j < P_{i+1} \tag{2.7a}$$

$$u_j^0 = \frac{1}{\Delta x} \int_{C_j} u_0(x) \mathrm{d}x \qquad j \in \mathbb{Z} \tag{2.7b}$$

$$u_{P_i}^{n+1} = \left(f^{(i)} \right)^{-1} \left(f^{(i-1)} \left(u_{P_i-1}^{n+1} \right) \right) \tag{2.7c}$$

where $\lambda = \frac{\Delta t}{\Delta x}$. We assume that the time step satisfies the CFL condition

$$\max_i \max_u \left(\left(f^{(i)} \right)' (u) \right) \lambda \leq 1 \tag{2.8}$$

Note that the (2.7c) represents a discrete version of the Rankine-Hugoniot condition in (2.6). Here, we use the ghost cells C_{P_i}, $i = 1, 2, \dots, N$ to explicitly impose the Rankine-Hugoniot condition. While this makes the numerical scheme (2.7) non-conservative, the convergence result in the paper, coupled with the fact that the limit is conservative, shows that the contribution of the non-conservative part of the scheme vanishes in the limit.

2.3. Decomposition into Simpler Problems

As mentioned earlier, one of the main ideas of the proof is to decompose the problem into finitely many initial-boundary value problems. In particular, the entropy solution u of (2.1) can be decomposed as

$$u = \sum_{i=1}^{N} u^{(i)} \qquad \text{where} \quad u^{(i)} \stackrel{\text{def}}{=} u \mathbb{1}_{D_i \times [0,T]} \tag{2.9}$$

such that $u^{(0)}$ solves the initial-value problem

$$\frac{\partial u^{(0)}}{\partial t} + \frac{\partial f^{(0)} \left(u^{(0)} \right)}{\partial x} = 0 \qquad (x,t) \in D_0 \times (0,T) \tag{2.10a}$$

$$u^{(0)}(x,0) = u_0(x) \qquad x \in D_0 \tag{2.10b}$$

and $u^{(i)}$ solves the initial-boundary value problem

$$\frac{\partial u^{(i)}}{\partial t} + \frac{\partial f^{(i)} \left(u^{(i)} \right)}{\partial x} = 0 \qquad (x,t) \in D_i \times (0,T) \tag{2.11a}$$

$$u^{(i)}(x,0) = u_0(x) \qquad x \in D_i \tag{2.11b}$$

$$u^{(i)}(\xi_i^+, t) = \left(f^{(i)} \right)^{-1} \left(f^{(i-1)} \left(u^{(i-1)} \left(\xi_i^-, t \right) \right) \right) \qquad t \in (0,T) \tag{2.11c}$$

Conversely, we can show that if $u^{(0)}$ and $u^{(i)}$, $i = 1, 2, \dots, N$ are the entropy solutions of (2.10) and (2.11) respectively, then u as defined in (2.9) is the entropy solution of (2.1). Furthermore, the numerical scheme defined in Section 2.2 when restricted to the subdomain D_i is will converge to the entropy solution $u^{(i)}$ on

D_i. The discrete Rankine-Hugoniot condition will then allows us to break down the problem of finding the convergence rate on the whole real line to finding convergence rates on each of the subdomains D_i.

If we look at the subdomains D_i, we see that there are three types of subdomains: D_0, D_N and D_i for $1 \leq i \leq N-1$. Without loss of generality, we can assume $D_0 = \mathbb{R}^-$ with discontinuity at $\xi = 0$, $D_N = \mathbb{R}^+$ with the discontinuity at $xi = 0$, and $D_i = [0, L]$ with discontinuity at $xi = 0$. This is because we can always translate and scale the problem to fit this setting. In the next sections, we prove the convergence rate of the numerical scheme (2.7) on each of the subdomains $\mathbb{R}^-$, $\mathbb{R}^+$ and $[0, L]$.

2.4. Case 1: Convergence on $\mathbb{R}^-$

For the case of $\mathbb{R}^-$, we consider the initial value problem Equation (2.12). Here, the function flux function $f \in C^2$ and is strictly monotone, that is $f'(u) > \alpha > 0$ for all $u \in \mathbb{R}$.

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0 \qquad (x,t) \in \mathbb{R}^- \times (0,T) \tag{2.12a}$$

$$u(x,0) = u_0(x) \qquad x \in \mathbb{R}^- \tag{2.12b}$$

The numerical scheme that we use to approximate the solution of (2.12) is given by

$$u_j^{n+1} = u_j^n - \lambda\left[g\left(u_j^n\right) - g\left(u_{j-1}^n\right)\right] \qquad j < 0, n \geq 0 \tag{2.13a}$$

$$u_j^0 = \frac{1}{\Delta x}\int\limits_{C_j} u_0(x)\mathrm{d}x \qquad j < 0 \tag{2.13b}$$

Definition 2.3 *(Entropy Solutions on $\mathbb{R}^-$).* We say that $u \in \mathcal{C}([0,T]; L^1(\mathbb{R}^-)) \cap L^\infty((0,T), \mathbb{R}^-)$ is an entropy solution of (2.12) if for all $c \in \mathbb{R}$, u satisfies (2.14) for all non-negative $\varphi \in \mathcal{C}^\infty(\mathbb{R}^- \times [0,T))$.

$$\begin{aligned}\int\limits_0^T\int\limits_{\mathbb{R}^-}\left[|u-c|\frac{\partial\varphi}{\partial t} + |f(u)-f(c)|\frac{\partial\varphi}{\partial x}\right]\mathrm{d}x\mathrm{d}t - \int\limits_{\mathbb{R}^-}|u(x,T)-c|\,\varphi(x,T)\mathrm{d}x \\ + \int\limits_{\mathbb{R}^-}|u_0(x)-c|\,\varphi(x,0)\mathrm{d}x - \int\limits_0^T\Big|g\left(u\left(0^-,t\right)\right) - g\left(c\right)\Big|\,\phi(0,t)\mathrm{d}t \geq 0\end{aligned} \tag{2.14}$$

Based on the definition of entropy solutions, and following classical Crandall-Majda arguments in [CM80, Proposition 4.1], we can show that the numerical scheme satisfies a discrete entropy equality (2.15a) away from the spatial boundaries.

$$D_+^t\eta_j^n + D_-q_j^n \leq 0 \qquad n \geq 1, j < 0 \tag{2.15a}$$

$$\eta_j^n = \left|u_j^n - c\right| \qquad\qquad q_j^n = \left|f(u_j^n) - f(c)\right| \tag{2.15b}$$

$$D_+^t a_j^n = \frac{a_j^{n+1} - a_j^n}{\Delta t} \qquad\qquad D_- a_j^n = \frac{a_j^n - a_j^{n-1}}{\Delta t} \tag{2.15c}$$

Then, we can prove a Kuznetsov-type lemma Theorem 2.3 for the solution u

Theorem 2.3 *(Kuznetsov-type Lemma).* Let u be the entropy solution of (2.12). Then, for any function $v\colon [0,T] \to (L^1 \cap \mathrm{BV})(\mathbb{R}^-)$ such that the one-sided limits $v(t^\pm)$ exist in L^1, we have (2.16) for some constant C independent of ϵ and ϵ_0.

$$\begin{aligned}
&\|u(\cdot,T) - v(\cdot,T)\|_{L^1(\mathbb{R}^-)} \\
\leq \quad &-\int_0^T \int_{\mathbb{R}^-} \int_0^T \Big[q\left(u\left(0^-,t\right), v(y,s)\right) + q\left(v\left(0^-,t\right), u(y,s)\right)\Big] \varphi(0,t,y,s) \mathrm{d}t\mathrm{d}y\mathrm{d}s \\
&+ \|u_0 - v(\cdot,0)\|_{L^1(\mathbb{R}^-)} - \Lambda_{\epsilon,\epsilon_0}(v,u) \\
&+ C\Big[\epsilon + \epsilon_0 + \nu_T(v,\epsilon_0) + \nu_0(v,\epsilon_0) + \mu\left(v(\cdot,T),\epsilon\right) + \mu\left(v(\cdot,0),\epsilon\right)\Big]
\end{aligned} \tag{2.16}$$

$$\varphi(x,t,y,s) = \omega_\epsilon(x-y)\omega_{\epsilon_0}(t-s) \tag{2.17a}$$

$$\Lambda_{\epsilon,\epsilon_0}(v,u) = \int_0^T \int_{\mathbb{R}^-} L(u, v(y,s), \varphi(\cdot,\cdot,y,s)) \mathrm{d}y\mathrm{d}s \tag{2.17b}$$

$$\nu_t(w,\epsilon_0) = \sup_{|\sigma|\leq\epsilon_0} \|w(\cdot,t+\sigma) - w(\cdot,t)\|_{L^1(\mathbb{R}^-)} \tag{2.17c}$$

$$\mu(w(\cdot,t),\epsilon) = \sup_{|z|\leq\epsilon} \|w(\cdot+z,t) - w(\cdot,t)\|_{L^1(\mathbb{R}^-)} \tag{2.17d}$$

The term $\Lambda_{\epsilon,\epsilon_0}(v,u)$ in (2.16) can be estimated by the inequality (2.18) where C is a constant which is independent of Δx, Δt, ϵ, and ϵ_0.

$$\Lambda_{\epsilon,\epsilon_0}(v,u) \leq C\left(\Delta x + \frac{\Delta x}{\epsilon} + \frac{\Delta t}{\epsilon_0}\right) \tag{2.18}$$

Combining the results, we can state our convergence result

Theorem 2.4 *(Convergence Rate on $\mathbb{R}^-$).* Let u be the entropy solution of (2.12) and $u_{\Delta t}$ be the solution of the numerical scheme given by (2.13) where λ is a constant. Then the error estimate for the numerical approximation is given by (2.19) where C is a constant independent of Δx.

$$\|u(\cdot,T) - u_{\Delta t}(\cdot,T)\|_{L^1(\mathbb{R}^-)} \leq C(\Delta x)^{\frac{1}{2}} \tag{2.19}$$

2.5. Case 2: Convergence on $\mathbb{R}^+$

For the case of convergence on $\mathbb{R}^+$, we consider the following initial-boundary value problem

$$\frac{\partial u}{\partial t} + \frac{\partial g(u)}{\partial x} = 0 \quad (x,t) \in \mathbb{R}^+ \times (0,T) \tag{2.20}$$

$$u(x,0) = u_0(x) \qquad x \in \mathbb{R}^+ \tag{2.21}$$

$$u(0,t) = f^{-1}\left(g\left(u\left(0^+,t\right)\right)\right) \quad t \in (0,T) \tag{2.22}$$

and the numerical scheme

$$u_j^{n+1} = u_j^n - \lambda \left[f\left(u_j^n\right) - f\left(u_{j-1}^n\right) \right] \qquad j \geq 1, n \geq 0 \tag{2.23a}$$

$$u_j^0 = \frac{1}{\Delta x} \int\limits_{C_j} u_0(x) \mathrm{d}x \qquad j \geq 0 \tag{2.23b}$$

$$u_0^n = f^{-1}\left(g\left(u_{-1}^n\right)\right) \qquad n \geq 1 \tag{2.23c}$$

where the boundary data is given in terms of $u(0, -, t)$ and u_{-1}^n respectively. Not that again we have a discrete entropy inequality of the form

$$D_+^t \eta_j^n + D_- q_j^n \leq 0 \qquad j \geq 1, n \geq 1 \tag{2.24}$$

Definition 2.4 *(Entropy Solution on $\mathbb{R}^+$).* We say $u \in \mathcal{C}\left([0,T]; L^1(\mathbb{R}^+)\right) \cap L^\infty(\mathbb{R}^+ \times (0,T))$ is an entropy solution of (2.20) if for all $c \in \mathbb{R}$, u satisfies (2.25) for all $\varphi \in \mathcal{C}^\infty\left([0,\infty) \times [0,T]\right)$.

$$\begin{aligned} \int\limits_0^T \int\limits_{\mathbb{R}^-} \left[|u - c| \frac{\partial \varphi}{\partial t} + |f(u) - f(c)| \frac{\partial \varphi}{\partial x} \right] \mathrm{d}x \mathrm{d}t - \int\limits_{\mathbb{R}^-} |u(x,T) - c| \, \varphi(x,T) \mathrm{d}x \\ + \int\limits_{\mathbb{R}^-} |u_0(x) - c| \, \varphi(x,0) \mathrm{d}x - \int\limits_0^T \left| g\left(u\left(0^+, t\right)\right) - g\left(c\right) \right| \phi(0,t) \mathrm{d}t \geq 0 \end{aligned} \tag{2.25a}$$

$$f\left(u\left(0^+, t\right)\right) = g\left(u\left(0^-, t\right)\right) \qquad \text{for almost every } t \in (0,T) \tag{2.25b}$$

Before we can prove the convergence rates, we need two auxiliary lemmas that are the consequences of the monotonicity of the flux.

Lemma 2.1 *(Temporal Variation Bound).* If the numerical scheme (2.23) satisfies the CFL condition (2.8), then the temporal variation of the numerical solution is bounded, specifically, for every $j \in \mathbb{Z}$, we have

$$\sum_{n=0}^{M} \left| u_j^{n+1} - u_j^n \right| \leq C \mathrm{TV}(u_0) \tag{2.26}$$

where $\mathrm{TV}(u_0)$ refers to the total variation of u_0 on the whole real line.

Lemma 2.2 *(Lipschitz Continuity).* Given an entropy solution u of (2.20), $f(u)$ is Lipschitz continuous in space, in the sense that

$$\int\limits_0^T \left| f\left(u(x,t)\right) - f\left(u(y,t)\right) \right| \mathrm{d}t \leq C \left| x - y \right| \quad \text{for all } x, y \in \mathbb{R}^+ \tag{2.27}$$

Theorem 2.5 *(Kuznetsov-type Lemma).* Let u be the entropy solution of (2.12). Then, for any function $v \colon [0,T] \to (L^1 \cap \mathrm{BV})(\mathbb{R}^+)$ such that the one-sided limits $v(t^\pm)$ exist in L^1, we have (2.16) for some constant C independent of ϵ and ϵ_0.

$$\begin{aligned}
&\|u(\cdot,T)-v(\cdot,T)\|_{L^1(\mathbb{R}^+)} \\
\leq\quad & +\int_0^T\int_{\mathbb{R}^+}\int_0^T\left[q\left(u\left(0^+,t\right),v(y,s)\right)+q\left(v\left(0^+,t\right),u(y,s)\right)\right]\varphi(0,t,y,s)\mathrm{d}t\mathrm{d}y\mathrm{d}s \\
&+\|u_0-v(\cdot,0)\|_{L^1(\mathbb{R}^+)}-\Lambda_{\epsilon,\epsilon_0}(v,u) \\
&+C\left[\epsilon+\epsilon_0+\nu_T(v,\epsilon_0)+\nu_0(v,\epsilon_0)+\mu\left(v(\cdot,T),\epsilon\right)+\mu\left(v(\cdot,0),\epsilon\right)\right]
\end{aligned} \tag{2.28}$$

$$\varphi(x,t,y,s)=\omega_\epsilon(x-y)\omega_{\epsilon_0}(t-s) \tag{2.29a}$$

$$\Lambda_{\epsilon,\epsilon_0}(v,u)=\int_0^T\int_{\mathbb{R}^-}L(u,v(y,s),\varphi(\cdot,\cdot,y,s))\mathrm{d}y\mathrm{d}s \tag{2.29b}$$

$$\nu_t(w,\epsilon_0)=\sup_{|\sigma|\leq\epsilon_0}\|w(\cdot,t+\sigma)-w(\cdot,t)\|_{L^1(\mathbb{R}^-)} \tag{2.29c}$$

$$\mu(w(\cdot,t),\epsilon)=\sup_{|z|\leq\epsilon}\|w(\cdot+z,t)-w(\cdot,t)\|_{L^1(\mathbb{R}^-)} \tag{2.29d}$$

Lemma 2.3. Following the same method as for Section 2.4, we can prove the estimate (2.30) holds for some constant C independent of Δx, Δt, ϵ and ϵ_0.

$$-\Lambda_{\epsilon,\epsilon_0}(u_{\Delta t},u)\leq C\left(\Delta x+\frac{\Delta x}{\epsilon}+\frac{\Delta x}{\epsilon_0}+\frac{\Delta t}{\epsilon_0}\right) \tag{2.30}$$

Theorem 2.6 *(Convergence Rate on $\mathbb{R}^+$).* Let u be the entropy solution of (2.20) and $u_{\Delta t}$ be the solution of the numerical scheme given by (2.23) where λ is a constant. Then the error estimate for the numerical approximation is given by (2.31) where C is a constant independent of Δx.

$$\|u(\cdot,T)-u_{\Delta t}(\cdot,T)\|_{L^1(\mathbb{R}^+)}\leq C(\Delta x)^{\frac{1}{2}} \tag{2.31}$$

2.6. Case 3: Convergence on $[0, L]$

Due to the monotonicity of the fluxes, by restricting the solution u and the numerical approximation $u_{\Delta t}$ to the interval $[0, L]$, we can apply the results of Section 2.5 to obtain the following to yield the convergence rate on $(0, L)$.

Theorem 2.7 *(Convergence Rate on $(0, L)$).* Let u be the entropy solution of (2.20) on the bounded interval $[0, L]$ and $u_{\Delta t}$ be the solution of the numerical scheme given by (2.23) where λ is a constant. Then the error estimate for the numerical approximation is given by (2.32) where C is a constant independent of Δx.

$$\|u(\cdot,T)-u_{\Delta t}(\cdot,T)\|_{L^1(0,L)}\leq C(\Delta x)^{\frac{1}{2}} \tag{2.32}$$

2.7. The Main Result

Combining all the results on the three difference cases of $\mathbb{R}^-$, $\mathbb{R}^+$ and $[0, L]$, and using the invariances of the conservation laws, we can prove the main result of the chapter.

Theorem 2.8 *(Convergence Rate on $\mathbb{R}$).* Let u be the entropy solution of (2.1) and $u_{\Delta t}$ be the solution of the numerical scheme given by (2.7) where λ is a constant. Then the error estimate for the numerical approximation is given by (2.33) where C is a constant independent of Δx.

$$\|u(\cdot,T) - u_{\Delta t}(\cdot,T)\|_{L^1(\mathbb{R})} \leq C(\Delta x)^{\frac{1}{2}} \tag{2.33}$$

2.8. Numerical Experiments

To illustrate our results, we now present experiments considering the two flux case:

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\Big[H(x)f(u) + (1 - H(x))g(u)\Big] = 0 \qquad (x,t) \in \mathbb{R} \times (0,T) \tag{2.34}$$

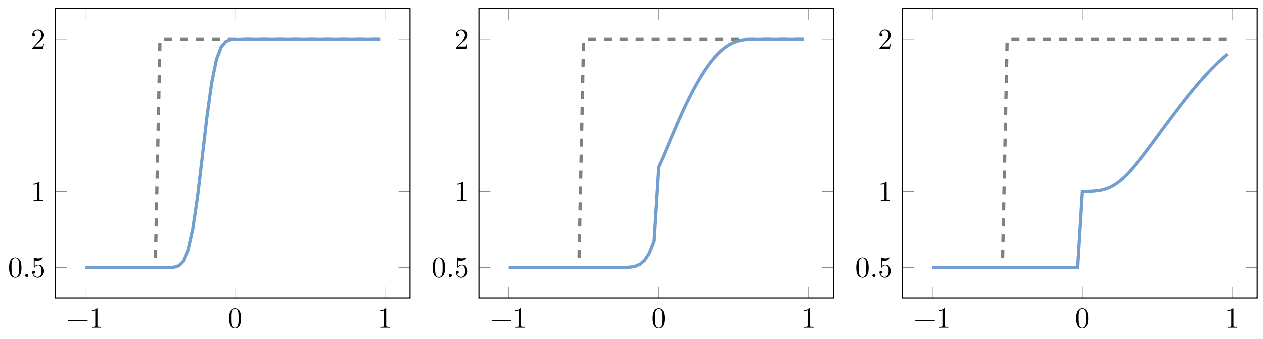
$$u(x,0) = u_0(x) \qquad x \in \mathbb{R} \tag{2.35}$$

2.8.1. Numerical Experiment 1

In our first numerical experiment, we choose $g(u) = u$ and $f(u) = u^2/2$, such that we switch from the transport equation to the Burgers equation across $x = 0$. The initial datum we consider for the experiment is

$$u_0(x) = \begin{cases} 0.5 & \text{if } x < -\frac{1}{2} \\ 2 & \text{if } x > -\frac{1}{2} \end{cases} \tag{2.36}$$

which is chosen such that the Rankine-Hugoniot condition at $x = 0$ gives $u(0^-,t) = u(0^+,t)$ before the jump at $x = -0.5$ interacts with the interface. Figure 2.1 shows the numerical solution calculated with the scheme with open boundaries in blue and the initial datum in gray (dashed line) at various times (before, during, and after interaction with the interface). We used $Deltax = \frac{2}{n}$ with $n = 64$, end time $T = 0.9$, and $\lambda = 0.5$. We clearly recognize the characteristic features of the transport equation and the Burgers equation here as the upward jump in the initial datum is transported to the right as a shock until it crosses the interface at $x = 0$ where the shock, as it enters the Burgers regime, subsequently becomes a rarefaction wave.



(a) $t = 0.3$. (b) $t = 0.6$. (c) $t = 0.9$.

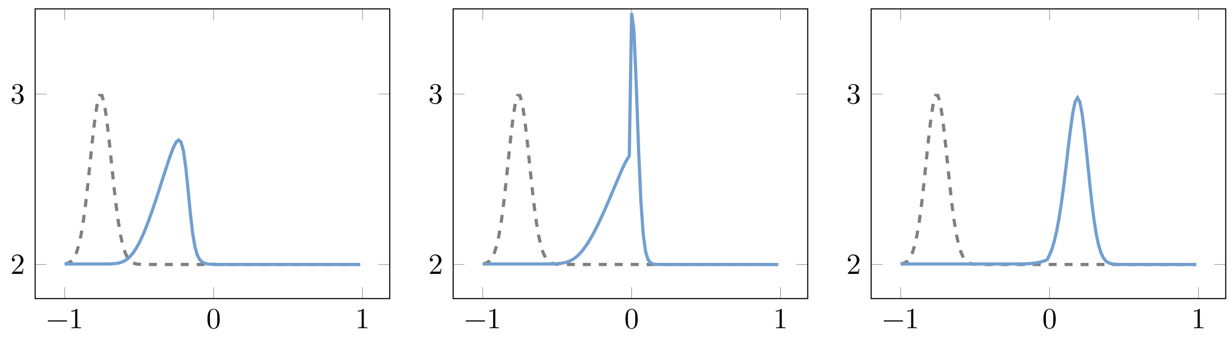
Figure 2.1.: Numerical solution of Experiment 1 with $\Delta x = \frac{2}{64}$ at various times.

2.8.2. Numerical Experiment 2

In our second numerical experiment we choose $g(u) = \frac{u^2}{2}$ and $f(u) = u$ such that we switch from the Burgers equation to the transport equation across $x = 0$. The initial datum is

$$u_0(x) = 2 + \exp(-100(x + 0.75)^2).$$

Again, the offset of the initial datum is chosen in a way such that the Rankine--Hugoniot condition at $x = 0$ gives $u(0-, t) = u(0+, t)$ before the non-constant part of u_0 interacts with the interface. Figure 2.2 shows the numerical solution calculated with the scheme (2.7) with open boundaries in blue and the initial datum in gray (dashed line) at various times (immediately before, during, and after interaction with the interface). We used $\Delta x = \frac{2}{n}$ with $n = 128$, end time $T = 0.5$, and $\lambda = 0.2$. We clearly recognize the shock formation due to the Burgers regime to the left of the interface (see Figure 2.2 (a)). Note that -- although difficult to see in Figure 2.2 (c) because of numerical diffusion -- the shock is preserved over the interface (only with a different profile).



(a) $t = 0.2$.   (b) $t = 0.3$.   (c) $t = 0.5$.

Figure 2.2.: Numerical solution of Experiment 2 with $\Delta x = \frac{2}{128}$ at various times.

Table 2.1 shows the observed convergence rates of the solution at time $T = 0.9$ for Experiment 1 and at time $T = 0.5$ for Experiment 2 for various values of Δx. As a reference solution, we used a numerical solution on a very fine grid ($n = 2048$) in both cases. As expected from experience in the case of spatially independent flux we observe convergence rates strictly between $\frac{1}{2}$ and 1 (cf. e.g. [LeV02, clawpack software]).

n	L^1 error	L^1 OOC
16	1.751×10^{-1}	--
32	1.256×10^{-1}	0.48
64	8.865×10^{-2}	0.50
128	5.918×10^{-2}	0.58
256	3.637×10^{-2}	0.70
512	1.978×10^{-2}	0.88
1024	8.145×10^{-3}	1.28

n	L^1 error	L^1 OOC
16	2.771×10^{-1}	--
32	1.823×10^{-1}	0.60
64	1.261×10^{-1}	0.53
128	8.390×10^{-2}	0.59
256	5.125×10^{-2}	0.71
512	2.780×10^{-2}	0.88
1024	1.132×10^{-2}	1.30

Table 2.1.: Convergence rates of Experiment 1 and 2.

2.9. Conclusion and Future Work

Scalar conservation laws with discontinuous flux frequently occur in physical applications and several numerical schemes have been considered in the literature. In contrast to the case where the nonlinear flux does not have a spatial dependency, however, convergence rate results for monotone finite volume schemes have not been available until now.

In this paper, we have established a convergence rate for upwind-type finite volume methods for the case where f is strictly monotone in u and the spatial dependency k is piecewise constant with finitely many discontinuities. The central idea of this paper is to split the problem into finitely many conservation laws between two neighboring discontinuities of k and get a convergence rate as a consequence of convergence rates on bounded domains. Here, the novel feature of this paper is the strong bound on the temporal total variation of the finite volume approximation which allows us to estimate the boundary terms in space at the discontinuities of k that appear when applying the classical Kuznetsov theory to (2.1)

As an outlook we name four possible directions of future research. One can extend the convergence rate result of this paper to the cases where k is not piecewise constant and f is not monotone. Second, it might be interesting to investigate convergence rates of monotone schemes in the Wasserstein distance. A third direction of future research might be to see whether the results of this paper can be extended to monotone schemes in conservation form. Lastly, convergence rates of the front tracking method for conservation laws with discontinuous flux are highly desirable as well.

Chapter 3 Multi-level Monte Carlo Methods for Discontinuous Flux

In this chapter, we sketch the arguments of [Bad+21] (linked here)and propose and analyse a single- and multi-level Monte Carlo method for random conservation laws with discontinuous fluxes. In Section 3.1, we introduce the concept of random entropy solutions and prove the existence and uniqueness of the random entropy solution. In Section 3.2, we introduce the Monte Carlo method for the random conservation law. In Section 3.3, we introduce the multi-level Monte Carlo method for the random conservation law. In Section 3.4, we estimate the work required to achieve a given accuracy. In Section 3.5, we optimise the sample numbers required to achieve a given accuracy. In Section 3.6, we present numerical experiments to validate the theory.

3.1. Well-Posedness of the Random Problem

$$\frac{\partial u(\omega;x,t)}{\partial t}+\frac{\partial f\left(\omega;k(\omega;x),u(\omega;x,t)\right)}{\partial x}=0 \qquad \omega\in\Omega, x\in\mathbb{R}, t\in\mathbb{R}^+ \tag{3.1a}$$

$$u(\omega;x,0)=u_0(\omega;x) \qquad \omega\in\Omega, x\in\mathbb{R} \tag{3.1b}$$

We consider the random version of the deterministic conservation laws (2.1) where the initial datum, the flux function and the discontinuous spatial dependency are subject to randomness. The random quantities of the initial datum, the flux function and the discontinuous spatial dependency are collectively referred to as the **random data** for the random conservation law (3.1) and are defined by (3.1).

Definition 3.1 *(Random Data).* Given constants $C_{\text{TV}}, F_f \in \mathbb{R}, \alpha in(0,\infty), N_k \in \mathbb{Z}$ and given a rectangle $R_1\times R_2\subset\mathbb{R}^2$, let $\mathbb{D}$ be the Banach space

$$\mathbb{D}=(\text{BV}\cap L^\infty)(\mathbb{R})\times L^\infty(\mathbb{R})\times \mathcal{C}^2(R;\mathbb{R}) \tag{3.2}$$

endowed with the norm

$$\|(u_0,k,f)\|_{\mathbb{D}}=\|u_0\|_{L^1(\mathbb{R})}+\text{TV}(u_0)+\|u_0\|_{L^\infty(\mathbb{R})}+\|k\|_{L^\infty(\mathbb{R})}+\|f\|_{\mathcal{C}^2(R;\mathbb{R})} \tag{3.3}$$

We say that a strongly measurable map $(u_0,k,f)\mapsto d, (u_0,k,f)\colon(\Omega,\mathcal{F})\to(\mathbb{D},\mathcal{B}(\mathbb{D}))$ is a random data for the random conservation law (3.1) if for $\mathbb{P}-a.e.\omega$,

$$u_0(\omega;x)\in R_1 \qquad k(\omega;x)\in R_2 \qquad \text{for a.e. } x\in\mathbb{R} \tag{3.4a}$$

$$u_0(\omega), f(\omega), k(\omega) \text{ satisfy the assumptions in Section 2.1} \tag{3.4b}$$

$$\mathrm{TV}(u_0(\omega)) \leq C_{\mathrm{TV}} < \infty \qquad \text{where } C_{\mathrm{TV}} \text{ is a constant independent of } \omega \tag{3.4c}$$

$$\|f(\omega;\cdot,\cdot)\|_{\mathcal{C}^2(R;\mathbb{R})} \leq C_f < \infty \qquad C_f \text{ is a constant independent of } \omega \tag{3.4d}$$

$$k(\omega;\cdot) \text{ has at most } N_k \text{ discontinuities} \tag{3.4e}$$

$$D_u f(\omega;k,u) \geq \alpha > 0 \qquad \text{and } f(\omega;k,0) = 0 \qquad \text{for all } (k,u) \in R \tag{3.4f}$$

Having defined the random data, we now define the random entropy solution of the random conservation law (3.1).

Definition 3.2 *(Random Entropy Solution).* Given random data $d\colon \Omega \to \mathbb{D}$, we say that a random variable $u\colon \Omega \to \mathcal{C}([0,T];L^1(\mathbb{R}))$ is a random entropy solution of the random conservation law (3.1) if for all $p \in \mathbb{R}$ and $\mathbb{P} - a.e\omega \in \Omega$, $u(\omega)$ satisfies (3.5) for all non-negative $\varphi \in C_c^\infty(\mathbb{R}\times[0,T])$.

$$\begin{aligned}\int_0^T\int_{\mathbb{R}} &\left[\left|u(\omega;x,t) - c_p(\omega;x)\right|\frac{\partial\varphi}{\partial t} + \mathcal{Q}\left(\omega;u(\omega;x,t)\right)\right]\mathrm{d}x\mathrm{d}t \\ &- \int_{\mathbb{R}}\left|u(\omega;x,T) - c_p(\omega;x)\right|\varphi(x,T)\mathrm{d}x + \int_{\mathbb{R}}\left|u_0(\omega;x) - c_p(\omega;x)\right|\varphi(x,0)\mathrm{d}x \geq 0\end{aligned} \tag{3.5}$$

Here, we have used the notation

$$\mathcal{Q}(\omega;u_0,k,f) = \mathrm{sgn}(u - c_p(\omega;x))\left[f(\omega;k(\omega;x),u(\omega;x,t)) - f(\omega;k(\omega;x),c_p(\omega;x))\right] \tag{3.6}$$

There are multiple notions of uniqueness for solutions in probability spaces. For this particular scenario, we are interested in pathwise uniqueness of the random entropy solutions, which we define in Definition 3.3.

Definition 3.3 *(Pathwise Uniqueness of Random Entropy Solutions).* If d_1 and d_2 are two random data and u_1 and u_2 are the corresponding random entropy solutions to the random conservation law (3.1), then pathwise-uniqness means that the implication (3.7) holds.

$$\mathbb{P}(d_1 = d_2) = 1 \implies \mathbb{P}(u_1 = u_2) = 1 \tag{3.7}$$

Given the definitions, we can now prove the existence and pathwise-uniqueness of random entropy solutions to the random conservation law (3.1). The existence proof is based on the fact composition of a Lipschitz function with a strongly measurable function is strongly measurable. The pathwise-uniqueness proof also uses the Lipschitz-continuity property of solution.

Theorem 3.1 *(Existence and Pathwise Uniqueness of Random Entropy Solutions).* Let $d\colon \Omega \to \mathbb{D}$ be an instance of random data. Then there exists a unique random entropy solution $u\colon \Omega \to \mathcal{C}([0,T];L^1(\mathbb{R}))$ to the random conservation law (3.1) which is pathwise unique.

Lemma 3.1 *(Uniform Bound on L^r Norms of Random Entropy Solutions).* Let $(u_0,k,f$ be random data and $D \subset \mathbb{R}$ be a bounded interval. Let $furtheru_0 \in L^r(\Omega;L^\infty(D))$, for some $1 \leq r \leq \infty$. Then the random entropy solution u of (3.1) is in $L^r(\Omega;C([0,T];L^r(D)))$ for all $1 \leq p \leq \infty$ and satisfies the uniform bound

(3.8) for all $0 \leq t \leq T$

$$\|u\|_{L^r(\Omega;L^p(D)} \leq C \|u_0\|_{L^r(\Omega;L^\infty(D))} \tag{3.8}$$

3.2. The Monte-Carlo Method

We now describe the Monte-Carlo method for approximating the expected value of the random entropy solution of the random conservation law (3.1).

Definition 3.4 *(The Monte-Carlo Finite Volume Method).* Given $M \in \mathbb{N}$, we generate M independent and identically distributed sample $\hat{f}^i, \hat{k}^i, \hat{u}_0^i$ for $i = 1, 2, \dots, M$ of the random data f, k, u_0. Let now $u_{\Delta x}^{\hat{i}}$, $i = 1, 2, \dots, M$ denote the numerical solutions generated the finite volume method Equation (2.7) at time t. Then, the M-sample MC FVM approximation to $\mathbb{E}[u(\cdot, t)]$ is given by

$$E_M\left[u_{\Delta x}(\cdot, t)\right] = \frac{1}{M} \sum_{i=1}^{M} u_{\Delta x}^{\hat{i}}(\cdot, t) \tag{3.9}$$

By using a combination of triangle inequality and Holder's inequality, we can show that the Monte-Carlo approximation (3.9) satisfies the error estimate (3.11).

Theorem 3.2 *(MC FVM Error Estimate).* Let (u_0, k, f) be random data and u the corresponding random entropy solution of (3.1). Assume that u_0 satisfies the r-th moment condition

$$\|u_0\|_{L^r(\Omega;L^\infty(D))} < \infty \tag{3.10}$$

for some $1 \leq r \leq \infty$. Then, for each $1 \leq p \leq \infty$, $0 \leq t \leq T$ and for $q = \min(2, r) > 1$, the MC FVM approximation satisfies the error estimate (3.11)

$$\left\|\mathbb{E}\left[u(\cdot, t)\right] - E_M\left[u_{\Delta x}(\cdot, t)\right]\right\|_{L^p(\Omega;L^p(D))} \leq C\left[M^{\frac{1-q}{q}} \|u_0\|_{L^r(\Omega;L^\infty(D))} + \|u_0\|_{L^r(\Omega;L^\infty(D))}^{1-\frac{1}{p}} \Delta x^{\frac{1}{2p}}\right] \tag{3.11}$$

In particular, the MC FVM approximation converges to the expected value $\mathbb{E}[u(\cdot, t)]$ in $L^q(\Omega; L^p(D))$ as $M \to \infty$ and $\Delta x \to 0$.

3.3. The Multi-Level Monte-Carlo Method

Instead of just considering Monte-Carlo samples of a single fixed resolution of the finite volume method, we now detail the corresponding multi-level variant, the MLMC FVM method. The key ingredient of the MLMC FVM method is the simultaneous MC sampling on different levels of resolution of the finite volume method with level-dependent numbers M_l of MC samples.

Definition 3.5 *(The multi-level Monte Carlo (MLMC) Method).* Given a sequence of meshes with mesh sizes Δx_l, $l = 0, 1, \dots, L$, we define an L-level MLMC FVM approximation to $\mathbb{E}[u(\cdot, t)]$ by (3.12).

$$E^L\left[u(\cdot, t)\right] = \sum_{l=0}^{L} E_{M_l}\left[u_{\Delta x_l}(\cdot, t) - u_{\Delta x_{l-1}}(\cdot, t)\right] \tag{3.12}$$

E_{M_l} denotes the MC approximation (3.9) with M_l samples on the mesh with size Δx_l.

Using a similar, but slightly more involved, argument to the one in the proof of Theorem 3.2, we can show that the MLMC FVM approximation (3.12) satisfies the error estimate (3.14).

Theorem 3.3 *(MLMC FVM Error Estimates).* Let $L > 0$, (u_0, k, f) be random data and u the corresponding random entropy of (3.1). Assume that u_0 satisfies the r-th moment condition

$$\|u_0\|_{L^r(\Omega;L^\infty(D))} < \infty \tag{3.13}$$

for some $1 \le r \le \infty$. Then for each $0 \le t \le T$, for any sequence $(M_l)_{l=0}^{L}$ of sample sizes at mesh level l, the MLMC FVM approximation satisfies the error estimate (3.14) for $q = \min(2, r) > 1$

$$\begin{aligned} &\left\|\mathbb{E}\left[u(\cdot,t) - E^L\left[U(\cdot,t)\right]\right]\right\|_{L^q(\Omega;L^p(D))} \\ &\le C\left[\|u_0\|_{L^1(\Omega;L^\infty(D))}^{1-\frac{1}{\tilde{p}}} \Delta x_L^{\frac{1}{2p}} + \|u_0\|_{L^q(\Omega;L^\infty(D))} M_0^{\frac{1-q}{q}} + \|u_0\|_{L^q(\Omega;L^\infty(D))}^{1-\frac{1}{\tilde{p}}} \sum_{l=0}^{L} M_l^{\frac{1-q}{q}} \Delta x_l^{\frac{1}{2p}}\right] \end{aligned} \tag{3.14}$$

Here $\tilde{p} = \max(p, q)$. In particular, for fixed L, the MLMC FVM approximation $E^L\left[u(\cdot,t)\right]$ converges to the expected value $\mathbb{E}[u(\cdot,t)]$ in $L^q(\Omega; L^p(D))$ as $\Delta x_L \to 0$ and $M_l \to \infty$.

3.4. Work Estimates

In order to analyze the efficiency of the MC and MLMC methods, we need to estimate the computational work required to achieve a given error tolerance and how it scales with the mesh size, the number of samples and the number of levels. Here, by computational work, we mean the number of floating-point operations performed when executing the algorithm. Before proceeding with the work estimates for the MC and MLMC methods, we need to establish the computational work required to solve the finite volume method for a deterministic problem.

3.4.1. Work Estimate for the Deterministic Problem

For a finite volume method, we assume that the computational work required to solve the deterministic problem is a constant C for each grid cell. Assume that the grid cell has the size Δx. Then the total number of grid cells and the total number of time steps scale as Δx^{-d}. Therefore, computational work required to solve the finite volume method on a mesh with cells of size Δx is then given by (3.15).

$$W^{\text{FVM}}(\Delta x) = C\Delta x^{-2} \tag{3.15}$$

For the sake of generality, we assume that the computational work instead scales as Δx^{-w} for some $w > 0$. As we have seen in eq. (2.33), we have the L^p convergence rate estimate given by (3.16) for $s = \frac{1}{2}$.

$$\|u - u_{\Delta x}\|_{L^p(D)} \le C\Delta x^{\frac{s}{p}} \tag{3.16}$$

Combing the two results, we can estimate the error in terms of computational work by (3.17).

$$\|u - u_{\Delta x}\|_{L^p(D)} \le C\left(W^{\text{FVM}}\right)^{-\frac{s}{wp}} \tag{3.17}$$

In particular, for $p = 1, w = 2$, and $s = \frac{1}{2}$, we have

$$\|u - u_{\Delta x}\|_{L^1(D)} \leq C \left(W^{\text{FVM}}\right)^{-\frac{1}{4}} \tag{3.18}$$

3.4.2. Work Estimate for the Monte Carlo Method

Since M deterministic solutions are required to compute the MC approximation, the computational work required to compute the MC approximation is given by (3.19).

$$W^{\text{MC}}(M) = CM\Delta x^{-w} \tag{3.19}$$

In order to obtain an error estimate for the MC approximation, we equilibrate the terms $M^{\frac{1-q}{q}}$ and $\Delta x^{\frac{s}{p}}$ in (3.11) by choosing $M = C\Delta x^{\frac{sq}{p(1-q)}}$. Inserting this into (3.19), we get the work estimate in terms of error by (3.20).

$$\|\mathbb{E}\left[u(\cdot,t)\right] - E_M\left[u_{\Delta x}(\cdot,t)\right]\|_{L^q(\Omega;L^p(D))} \leq C\left(W_M^{\text{MC}}\right)^{-\frac{s}{wp+s\frac{q}{q-1}}} \tag{3.20}$$

In particular, for $p = 1, r \geq 2, w = 2$ and $s = \frac{1}{2}$, we get the error estimate in (3.21).

$$\|\mathbb{E}\left[u(\cdot,t)\right] - E_M\left[u_{\Delta x}(\cdot,t)\right]\|_{L^2(\Omega;L^1(D))} \leq C\left(W_M^{\text{MC}}\right)^{-\frac{1}{6}} \tag{3.21}$$

3.5. Sample Number Optimization

Instead of computing a generic error estimate for the MLMC method, we can instead use the optimization result from [Kol+13] and directly compute the optimal number of samples for the MLMC method, and therefore the optimal computational work estimates.

Theorem 3.4 *(Optimal Number of Samples for the MLMC Method [Kol+13])***.** Assume that the work of a MLMC FVM method with L discretization levels scales asymptotically as

$$W^{\text{MLMC}}(L) = C\sum_{l=0}^{L} M_l \Delta x_l^{-w} \tag{3.22}$$

for some $w > 0$ and that the approximation error scales as

$$\mathcal{E}_L = C\left[\Delta x_L^{\frac{sq}{p}} + M_0^{1-q} + \sum l = 0^L M_l^{1-q}\Delta x_l^{\frac{sq}{\tilde{p}}}\right] \tag{3.23}$$

where $\tilde{p} = \max(p, q)$. Then, given an error tolerance $\epsilon > 0$, the optimal sample numbers M_l minimizing the computational work, given the error tolerance ϵ, are given by (3.24).

$$M_0 \simeq \left[\frac{1 + \Delta x_0^{\frac{s}{q}} \sum_{l=1}^{L} 2^{l\left(w\frac{q-1}{q} - \frac{s}{p}\right)}}{\epsilon - \Delta x_L^{\frac{sq}{p}}}\right]^{\frac{1}{q-1}} \tag{3.24a}$$

$$M_l \simeq M_0 \Delta x_0^{\frac{s}{p}} 2^{-l\left(\frac{s}{q}+\frac{w}{q}\right)} \tag{3.24b}$$

where $\simeq$ indicates this is the number of samples up to a constant which is independent of l and L. The minimal amount of work then is

$$W_L^{\text{MLMC}} \simeq \Delta x_0^{-w} \left[1 + \Delta x_0^{\frac{s}{q}} \sum_{l=1}^{L} 2^{l\left(w\frac{q-1}{q} - \frac{s}{p}\right)}\right] \left[\frac{1 + \Delta x_0^{\frac{s}{p}} \sum_{l=1}^{L} 2^{l\left(w\frac{q-1}{q} - \frac{s}{p}\right)}}{\epsilon - \Delta x_L^{\frac{sq}{p}} 2^{-L\frac{qs}{p}}}\right]^{\frac{1}{q-1}} \tag{3.25}$$

Corollary 3.1. In addition to assumptions of theorem 3.4, if we assume that $w\frac{q-1}{q} - \frac{s}{p} > 0$ and that L and Δx_0 are large enough such that

$$\Delta x_L^{\frac{s}{p}\frac{q}{q-1}-w} > \Delta x_0^{-w} \tag{3.26}$$

. Then for each $0 \leq t \leq T$ and for $q = \min(2, r)$, the $L^q(\Omega; L^p(D))$-approximation error of the MLMC FVM method scales with respect to the computational work

$$\left\|\mathbb{E}[u(\cdot,t)] - E^L\left[U(\cdot,t)\right]\right\|_{L^q(\Omega;L^p(D))} \leq C\left(W_L^{\text{MLMC}}\right)^{-\frac{s}{wp+s\frac{\tilde{p}-p}{\tilde{p}}\frac{q}{q-1}}} \tag{3.27}$$

3.6. Numerical Experiments

In this section, we present numerical experiments motivated by two-phase flow in a heterogeneous porous medium. The time evolution of the oil saturation $u \in [0,1]$ can be modeled by (2.1) where the flux is given by

$$f(k(x), u) = \frac{\lambda_{\text{o}}(u)}{\lambda_{\text{o}}(u) + \lambda_{\text{w}}(u)}(1 - k(x)\lambda_{\text{w}}(u)), \tag{3.28}$$

see [HR15, Ex. 8.2] Here, the functions λ_{o} and λ_{w} denote the phase mobilities/relative permeabilities of the oil and the water phase, respectively. Typically, one uses the simple expressions

$$\lambda_{\text{o}}(u) = u^2, \qquad \lambda_{\text{w}}(u) = (1-u)^2$$

which we will also do in the subsequent experiments. The coefficient k in (3.28) corresponds to the absolute permeability of the medium. Since the medium is usually layered to some extent throughout the reservoir and even continuously varying geology is typically mapped onto some grid, the coefficient k is often modeled as a piecewise constant function [GR93].

Since numerical experiments for conservation laws where the initial datum or the flux is uncertain have been reported in other works (albeit without spatially discontinuous flux), we will here focus on numerical experiments where the discontinuous coefficient k is subject to randomness. We consider the initial datum

$$u_0(x) = \begin{cases} 0.8, & -0.9 < x < -0.2, \\ 0.4, & \text{otherwise} \end{cases} \tag{3.29}$$

on the spatial domain $D = [-1, 1]$ with periodic boundary conditions. Figure 3.1 shows two examples of fluxes of the form (3.28) and indicates the relevant domain determined by the initial datum (3.29). In all

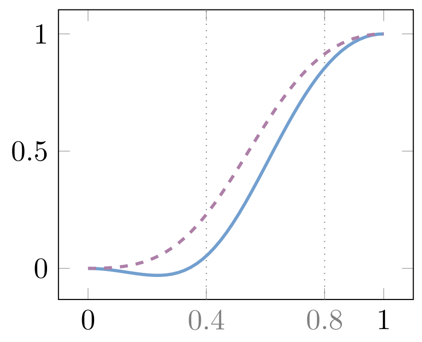


Figure 3.1.: Two possible fluxes of the form (3.28) for $k(x) = 0.7$ (dashed line) and $k(x) = 2.3$ (straight line)

experiments we use $\lambda = \frac{\Delta t}{\Delta x} = 0.2$ in the finite volume approximation (2.7).

When choosing the number of samples for the MLMC estimator we use the formulae (3.24) with " $=$ " replacing " $\simeq$ " and rounding to the next biggest integer. Here we use $p = 1, r = q = 2, w = 2, s = \frac{1}{2}$, and $\varepsilon = 2\Delta x_L^{2s}$ For example, for $L = 7$ and $\Delta x_0 = 2^{-4}$ we use $(M_l)_{l=0}^L = (95646, 20107, 8454, 3555, 1495, 629, 265, 112)$ samples.

In order to compute an estimate of the approximation error

$$\left\|\mathbb{E}[u(\cdot,T)] - E^L[U(\cdot,T)]\right\|_{\mathrm{L}^2(\Omega;\mathrm{L}^1(D))} = \left(\mathbb{E}\left[\left\|\mathbb{E}[u(\cdot,T)] - E^L[U(\cdot,T)]\right\|^2_{\mathrm{L}^1(D)}\right]\right)^{\frac{1}{2}}$$

we use the root mean square estimator introduced in [MS12]: We denote by $U_{\text{ref}}(\cdot,T)$ a reference solution and by $(U_i(\cdot,T))_{i=1}^K$ a sequence of independent approximate solutions $E^L[U(\cdot,T)]$ obtained by running the MLMCFVM estimator with L levels K times. Then, we estimate the relative error by

$$\mathcal{RMS} = \left(\frac{1}{K}\sum_{i=1}^K (\mathcal{RMS}_i)^2\right)^{\frac{1}{2}}$$

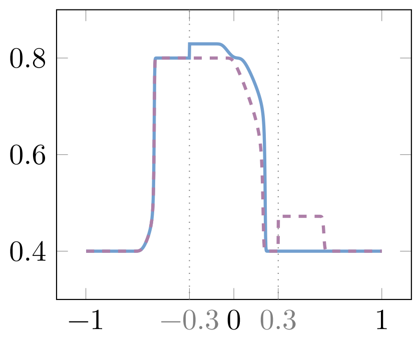
where

$$\mathcal{RMS}_i = 100 \times \frac{\|U_{\text{ref}}(\cdot,T) - U_i(\cdot,T)\|_{\mathrm{L}^1(D)}}{\|U_{\text{ref}}(\cdot,T)\|_{\mathrm{L}^1(D)}}.$$

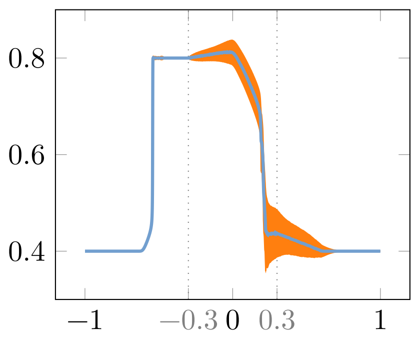
Here, as suggested in [MS12], we use $K = 30$ which was shown to be sufficient for most problems. In order to compute the reference approximation $U_{\text{ref}}(\cdot,T)$ of $\mathbb{E}[u(\cdot,T)]$ we take a large number of uniformly-spaced points $(\omega_i)_{i=1}^N$ in Ω (which in our examples are a closed interval and a rectangle) and compute corresponding finite volume approximations $u_{\Delta x^*}(\omega_i;\cdot,T)$ for a very small discretization parameter Δx^* and then determine $U_{\text{ref}}(\cdot,T)$ by applying the trapezoidal rule to approximate the integral $\int_\Omega u(\omega;\cdot,T)\,\mathrm{d}\mathbb{P}(\omega)$ using the points $(u_{\Delta x^*}(\omega_i;\cdot,T))_{i=1}^N$.

In our experiments we also indicate the approximated standard deviation. To that end, we approximate the variance by

$$V_L = \sum_{l=0}^L E_{M_l}\left[(u_{\Delta x_l}(\cdot,T) - u_{\Delta x_{l-1}}(\cdot,T) - E_{M_l}[u_{\Delta x_l}(\cdot,T) - u_{\Delta x_{l-1}}(\cdot,T)])^2\right].$$



(a) Two samples of the random entropy solution ($\xi = -0.3$ (straight line), $\xi = 0.3$ (dashed line), $\Delta x = 2^{-9}$).



(b) MLMCFVM approximation ($\Delta x_0 = 2^{-4}$, $L = 7$).

Figure 3.2.: Two samples and a MLMCFVM approximation of the (mean of the) random entropy solution for Experiment 1 with $T = 0.2$ and $\lambda = 0.2$. The orange area indicates the area between the mean $\pm$ standard deviation. For each sample the discontinuity of k is located in the interval between the dotted lines.

3.6.1. Uncertain position of rock layer interface

For our first numerical experiment we will model the absolute permeability parameter as

$$k(x) = \begin{cases} 1, & x < \xi(\omega), \\ 2, & x > \xi(\omega) \end{cases}$$

corresponding to an uncertain position of the interface between two rock types in the reservoir. Here, the random variable ξ is uniformly distributed in $[-0.3, 0.3]$. Figure 3.2a shows two samples of the approximate random entropy solution (with $\xi = -0.3$ and $\xi = 0.3$ respectively) calculated using 2^{10} grid points at time $T = 0.2$ and Figure 3.2b shows an estimate of the expectation $\mathbb{E}[u(\cdot, T)]$ computed by the MLMCFVM with $\Delta x_0 = 2^{-4}$ and $L = 7$.

Table 3.1 and Figure 3.3 show the estimated $\mathcal{RMS}$ error as a function of the number of levels. In particular, Table 3.1a shows the observed order of convergence (OOC) with respect to Δx_L while Table 3.1b shows the observed order of convergence with respect to the computational work calculated based on a best linear fit under the assumptions that $\mathcal{RMS} \sim (\Delta x_L)^{r_1}$ and $\mathcal{RMS} \sim (\text{work})^{r_2}$. Here, we use the runtime as a surrogate for the computational work. We observe that in Experiment 1 both rates are better than the rates guaranteed by our convergence analysis.

To compute the reference solution in Experiment 1, we approximated the expectation with respect to the uniform probability distribution on the interval $[-0.3, 0.3]$ using the trapezoidal rule with $N = 200$ equidistant points and choosing $\Delta x^* = 2^{-11}$ for the finite volume approximations.

3.6.2. Uncertain absolute permeabilities

For our second numerical experiment we will model the absolute permeability parameter as

$$k(x) = \begin{cases} 1 + \xi_1(\omega), & x < 0, \\ 2 + \xi_2(\omega), & x > 0 \end{cases}$$

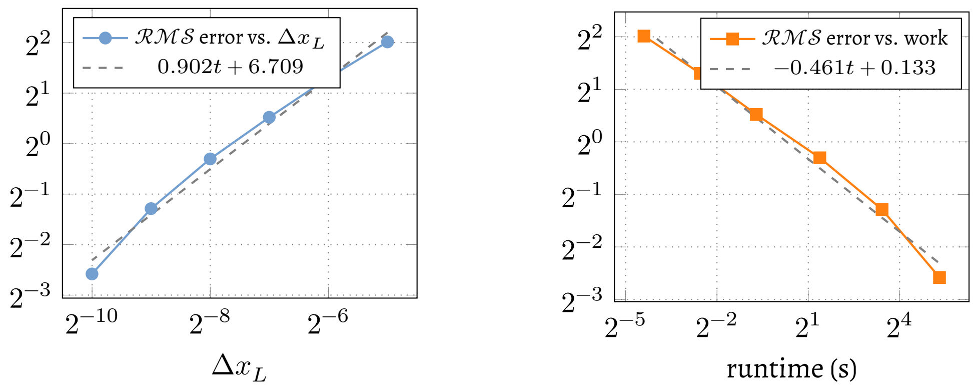
L	Δx_L	$\mathcal{RMS}$	OOC
1	2^{-5}	4.04	
2	2^{-6}	2.47	
3	2^{-7}	1.44	
4	2^{-8}	0.81	
5	2^{-9}	0.41	
6	2^{-10}	0.17	0.90

(a) $\mathcal{RMS}$ error versus Δx_L.

L	runtime	$\mathcal{RMS}$	OOC
1	0.05	4.04	
2	0.17	2.47	
3	0.61	1.44	
4	2.60	0.81	
5	10.72	0.41	
6	39.64	0.17	−0.46

(b) $\mathcal{RMS}$ error versus work.

Table 3.1.: $\mathcal{RMS}$ error in Experiment 1 as a function of the finest grid resolution Δx_L and as a function of the work (here measured by the runtime in s) for various values of L and for $\Delta x_0 = 2^{-4}$.

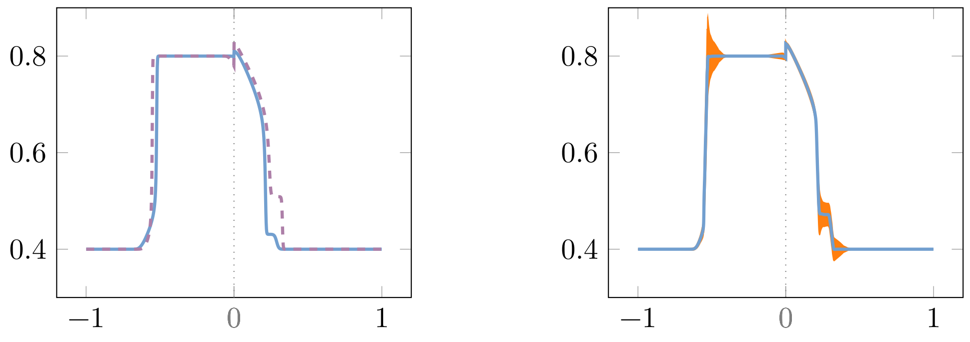


(a) $\mathcal{RMS}$ error versus Δx_L.

(b) $\mathcal{RMS}$ error versus work.

Figure 3.3.: $\mathcal{RMS}$ error in Experiment 1 as a function of the finest grid resolution Δx_L and as a function of the work (here measured by the runtime in s) corresponding to the values in Table 3.1. The dashed lines indicate the observed order of convergence based on a best linear fit.

corresponding to uncertain absolute permeabilities of two rock layers. Here, the random variables ξ_1 and ξ_2 are both uniformly distributed in $[-0.3, 0.3]$.



(a) Two samples of the random entropy solution $((\xi_1, \xi_2) = (0.3, -0.3)$ (straight line), $(\xi_1, \xi_2) = (-0.3, 0.3)$ (dashed line), $\Delta x = 2^{-9}$).

(b) MLMCFVM approximation ($\Delta x_0 = 2^{-4}, L = 7$).

Figure 3.4.: Two samples and a MLMCFVM approximation of the (mean of the) random entropy solution for Experiment 2 with $T = 0.2$ and $\lambda = 0.2$. The orange area indicates the area between the mean $\pm$ standard deviation and the dotted line marks the (fixed) position of the discontinuity of k.

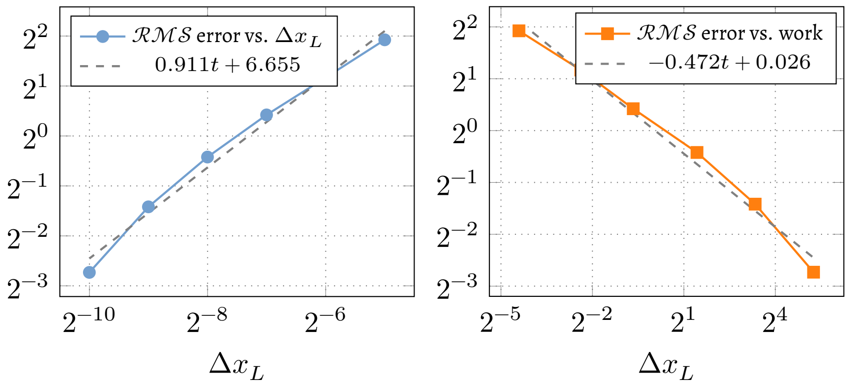
L	Δx_L	$\mathcal{RMS}$	OOC
1	2^{-5}	3.80	
2	2^{-6}	2.25	
3	2^{-7}	1.34	
4	2^{-8}	0.75	
5	2^{-9}	0.37	
6	2^{-10}	0.15	0.91

(a) $\mathcal{RMS}$ versus Δx_L.

L	runtime (s)	$\mathcal{RMS}$	OOC
1	0.05	3.80	
2	0.19	2.25	
3	0.63	1.34	
4	2.70	0.75	
5	10.12	0.37	
6	38.14	0.15	−0.47

(b) $\mathcal{RMS}$ versus work.

Table 3.2.: $\mathcal{RMS}$ error in Experiment 2 as a function of the finest grid resolution Δx_L and as a function of the work (here measured by the runtime in s) for various values of L and for $\Delta x_0 = 2^{-4}$.



(a) $\mathcal{RMS}$ error versus Δx_L. (b) $\mathcal{RMS}$ error versus work.

Figure 3.5.: $\mathcal{RMS}$ error in Experiment 2 as a function of the finest grid solution Δx_L and as a function of the work (here measured by the runtime in s) corresponding to the values in Table 3.2. The dotted lines indicate the observed order of convergence based on a best linear fit.

Figure 3.4a shows two samples of the approximate random entropy solution (with $(\xi_1, \xi_2) = (0.3, -0.3)$ and $(\xi_1, \xi_2) = (-0.3, 0.3)$ respectively) calculated using 2^{10} grid points at time $T = 0.2$ and Figure 3.4b shows an estimate of the expectation $\mathbb{E}[u(\cdot, T)]$ computed by the MLMCFVM with $\Delta x_0 = 2^{-4}$ and $L = 7$.

Table 3.2 and Figure 3.5 again show the root mean square error estimate and the observed order of convergence with respect to Δx_L and with respect to the computational work. As before, we observe that the observed convergence rates are better than the theoretical bounds.

In order to compute a reference solution for Experiment 2, we used a tensorized trapezoidal rule with 60×60 points in the stochastic domain $[-0.3, 0.3]^2$ and $\Delta x^* = 2^{-11}$ for the finite volume approximations.

3.7. Conclusion

In this chapter, we have considered conservation laws with discontinuous flux where the model parameters, i.e., the initial datum, the flux function, and the discontinuous spatial dependency coefficient, are uncertain. Based on adapted entropy solutions for the deterministic case, we have introduced a notion of random entropy solutions and have proved well-posedness.

To numerically approximate the mean of a random entropy solution, we have proposed Monte Carlo methods coupled with a class of finite volume methods suited for conservation laws with discontinuous flux. Our convergence analysis includes convergence rate estimates for the Monte Carlo and multilevel Monte Carlo finite volume method. Further, we have provided error versus work rates which show that the multilevel Monte Carlo finite volume method is much faster than the (single-level) Monte Carlo finite volume method.

We have presented numerical experiments motivated by two-phase flow in heterogeneous porous media, e.g., oil reservoirs with different rock layers. The numerical experiments verify our theoretical results concerning convergence rates of the multilevel Monte Carlo finite volume method.

As a possible direction of future research, we want to mention that -- from a practical standpoint -- it would be desirable to design multilevel Monte Carlo finite volume methods based on finite volume methods that require no processing of the flux discontinuities. Such numerical methods have been considered in [Tow20], however, there are currently no convergence rate results available for these methods.

Part II.

Arbitrary Lagrangian-Eulerian Methods

Chapter 4 Introduction

The ability to predict various physical phenomena to a reasonable degree of accuracy is one of the cornerstones of modern sciences and engineering. Hyperbolic conservation laws model a large number of such physical systems either exactly or approximately. Consequently, the study of hyperbolic conservation laws has been instrumental in our ability to successfully construct such systems and is a highly research topic.

Hyperbolic conservation laws are partial differential equations of the form

$$\frac{\partial \mathbf{u}(\mathbf{x},t)}{\partial t} + \nabla_{\mathbf{x}} \cdot \mathcal{F}(\mathbf{x},t,\mathbf{u}) = 0 \qquad \mathbf{u} \in \mathbb{R}^n,\ (\mathbf{x},t) \in \mathbb{R}^d \times \mathbb{R}^+ \tag{4.1}$$

where the flux Jacobian $D_{\mathbf{u}}\mathcal{F}$ is such that for all $\mathbf{u} \in \mathbb{R}^n$ and all $\boldsymbol{\nu} \in \mathbb{R}^d$, the matrix $\langle D_{\mathbf{u}}\mathcal{F}, \boldsymbol{\nu}\rangle$ has n real eigenvalues and n linearly independent eigenvectors. If, in addition, these eigenvalues are distinct, then the system is said to be **strictly hyperbolic**. The hyperbolic nature indicates to the finite speed of propagation of information in the system.

Despite the considerable success of using hyperbolic conservation laws in modeling physical phenomena, we note that the models are more often than not only an imperfect description of the underlying phenomena. It is generally impossible to model accurately all aspects of physics in a single model, and hence, we often limit ourselves to set of aspects that are enough to capture the phenomena of interest. Unsurprisingly then, hyperbolic conservation laws are often only an approximation of the underlying physical phenomena where some of the physics is neglected or simplified. For example, the Euler equations result from the Navier-Stokes equations by neglecting the viscous terms, while the Navier-Stokes equations themselves are an approximation of the Boltzmann equation by neglecting the molecular scale physics.

Therefore, it should not come as a surprise when such models exhibit unphysical solutions. In the case of conservation laws, one of the ways this phenomenon is manifests itself is in infinitely many weak solutions to the conservation laws. At such times, we must use additional information from physics to attempt to single out the physically relevant solutions. For scalar conservation laws, we now have a complete theory which uniquely identifies the physically relevant solutions using entropy conditions. However, this is not always true, as is evinced by the discussion of wild solutions in the contemporary literature.

Designing numerical methods for these models introduces further complications in the process. Numerical methods are an approximation to the often continuous model and not all approximations lead to the physically relevant solution as singled out by the analysis. Designing methods which converge to the physically relevant solutions is a priori non-trivial. In some cases like systems of hyperbolic conservation laws, the design of such methods is still an open problem. In such cases, one can not list sufficient conditions to establish the correctness of the numerical solutions. However, it is often possible to propose a set of necessary conditions which are enough for the use cases generally found in most applications.

Most of the equations are non-linear partial differential equation (PDE) due to which it is often impossible

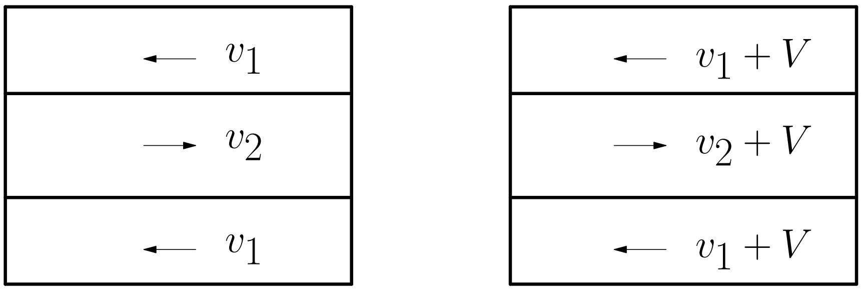


Figure 4.1.: Problem Setup to Study Kelvin-Helmholtz Instabilities

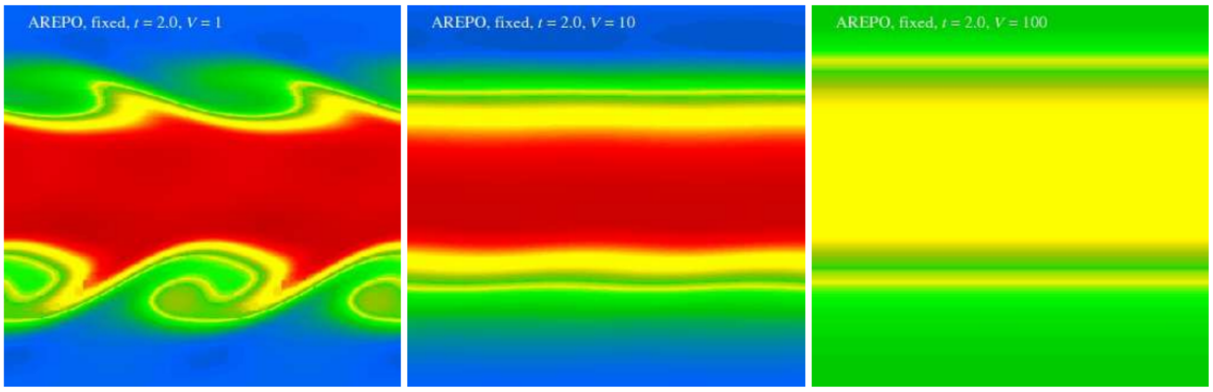


Figure 4.2.: Kelvin-Helmholtz Instabilities in Fluid Flow

to obtain the solutions of these equations in closed form. Numerical methods then become an important tool to study such equations. Numerical methods have developed into an especially important methods of study of physical systems due to their reliability and relatively inexpensive nature. In contrast to the considerably well-developed theory of linear partial differential equations, each non-linear equations presents its own set of difficulties.

In the past few decades, mesh based methods like finite volume and discontinuous Galerkin finite element methods have been exceedingly successful in the task of modelling compressible fluid flows. Due to the upwind nature of the finite volume schemes, dissipation is implicitly built in the schemes. This allows the schemes to be able to compute discontinuous solutions in a stable manner. Discontinuous Galerkin methods can be considered to be higher order generalizations of the finite volume methods which also make use of the upwinding technology but instead of cell averages, evolve a polynomial inside each cell.

In the most basic form of these methods, the mesh used for simulations is kept fixed over the course of simulations. This can sometimes be an issue for unresolved solutions, where the truncation error from the simulation can dominate the behavior of fluids. A striking example of such a situation is the growth of the Kelvin-Helmholtz instabilities between two fluids of different densities. The problem setup consists three scenarios each having an unresolved discontinuity (a slip-surface) between two fluids moving in opposite directions, but in each scenario, there is a boost fig. 4.1.

Due to the principle of Galilean invariance, the fluid should behave identically irrespective of the value of V_0. However, using AREPO code [Spr10] for example, one can see in fig. 4.2 that the instabilities disappear in the moving frames. The disappearance of the instabilities is due to dampening of the unstable modes in the fluid by the numerical dissipation of method.

The problem of numerical dissipation is not limited to the moving frames. The effect is observable even in rest frames where the instabilities disappear at lower resolutions. So, even though the methods themselves are Galilean invariant, the corresponding truncation error is not Galilean invariant. Apart from suppression of instabilities, the violation of Galilean invariance of truncation error is a major contributor to smearing out of shocks and contact discontinuities, especially in higher-order methods. Even for smooth solutions, the accuracy of the solution can be reduced due to the effects of truncation error.

One of the ways to deal with these challenges is to move the mesh along with the fluid, thereby reducing the truncation error in the simulations. A natural choice is to move the mesh with the velocity of the fluid and thereby exploit the Lagrangian formulation for fluid mechanics. The resulting method is called the Lagrangian method and was first developed in 1950s and 60s [VR50; Wil63]. A general review of these methods can be found in [Ben92].

In Lagrangian simulations, a given region is always approximated by the same number of mesh points. This ensures that the accuracy of the simulation is in general maintained to the accuracy of the initial approximation. Furthermore, compared to the fixed mesh methods, the number of points required to get a similar accuracy in Lagrangian simulations is often surprisingly small. This makes Lagrangian methods quite attractive for fluid simulations.

Unfortunately, the very features of the Lagrangian method which make it so effective are also the ones which make it totally unsatisfactory for simulations involving large shear flows. The property of mesh points to exactly follow the fluid flow causes the underlying mesh to become highly distorted, if not degenerate. This results in the reduction of accuracy of approximation and consequently of the simulation, with the simulation not being able to continue till completion in some cases.

In case of small deformations, one can add corrections to the mesh velocity in order to maintain the quality of the mesh for a longer time and often till the completion of the simulation. At this stage however, the mesh velocity is not equal the fluid velocity, and Lagrangian formulation is no longer an appropriate setting for our computations. We must instead use the Arbitrary Lagrangian-Eulerian (ALE) formulation where the vertices of the computational mesh may be moved in an arbitrary fashion. Methods using ALE formulation were originally developed in [FL64; HAC74; Noh63; Tru66] in the context of finite difference and finite volume methods. The method was subsequently adopted in the finite element context and early applications can found in [BK78; DGH82; HLZ81; KBS79].

In theory, the mesh velocities in the ALE formulation can be any arbitrary value but in practice, it is desirable to keep the mesh velocities as close to fluid velocity as possible. Methods which satisfy this property are called **almost-Lagrangian** methods. The methods are practical in cases of small deformations of the meshes, but for large deformations, it becomes impractical to maintain a good quality mesh with mesh velocities close to fluid velocities and it becomes necessary to make changes to the mesh topology.

An approach to handle the problem of the mesh quality is to move the mesh for as long as possible with the fluid velocity, and remesh when the mesh quality becomes unacceptable [BLL94; Has+07]. For each remeshing, the simulation has to be stopped, a new mesh must be generated and the solution must be transferred to the new mesh. This approach can be efficient if the number of remeshing required are very low, since the simulation is fully ALE and free from interpolation errors.

The approach described above is sometimes also known as remap ALE methods or indirect ALE methods. Remap ALE schemes have been used extensively in some problems and some of the recent work on the topic can be found in [Ber+11; BRS13; Bre+13; KS12; Yan+13]. For large deformations, the number of required remeshing increase and this can be both costly and result in poor accuracy due to the solution transfer step.

Another approach to get around this problem was presented in [Spr10], where the connectivity of the moving mesh is dynamically regenerated via a moving unstructured but conforming Voronoi tessellation of the domain. Building on the idea, a higher order ALE Discontinuous Galerkin (DG) method was designed by [Gab+19]. In these methods, a new mesh is generated at every time step, a correspondence is generated between the elements in the two time steps, and finally a ALE DG method is solved on the mesh, where the space-time information is gleaned from the correspondence information.

In the above method, there is a possibility of non-regular (``crazy'') finite elements appearing in the mesh which is taken care of using the techniques in [Gab+19]. Due to no inherent need to remap the solutions in these methods, these methods are often referred to as *direct* ALE DG methods. Since there is no remapping of solutions needed, the methods do not suffer from the interpolation error, however, it requires a mesh to be generated at every time step and corresponding connection information to be determined which can be costly.

There is a third approach which was first outlined in [Com+10; DF08], where one only performs a local remeshing operations, such as vertex insertion, vertex collapse, connectivity changes and vertex displacements, to preserve a good mesh quality throughout the simulation. The advantage of this method is that it maintains an acceptable mesh quality without needing to stop, remesh and resume the simulation. Another important aspect is that for all the cells which do not need to be modified, there is no interpolation error during the local cell remeshing. This helps in maintaining a high accuracy of solutions through the simulation. [Ala14; BA19] further showed that for ALE DG simulations with a elasticity-based mesh velocities, only mesh velocity corrections and connectivity changes (face swapping) is enough to maintain a good quality mesh.

This is the idea we use in this part of the thesis for our ALE DG method. The objective was to design an almost-Lagrangian ALE DG method which can be run efficiently on parallel architectures. To achieve this, we propose an ALE DG method with the following properties:

1. Single step method with only one communication between the cells in a time step.
2. Local remeshing algorithms.
3. Local mesh velocity algorithms.

Chapter

5

ALE-DG Methods for One Dimensional Euler Equations

In this chapter, we sketch the contents of the paper [BCK20] (linked here) and [BC18] (linked here) which introduces the ALE-DG method for the one-dimensional Euler equations. As mentioned in chapter 4, there are multiple problems that need to be solved in order to develop a robust and efficient multi-dimensional ALE-DG method for compressible flows. In the spirit of tackling the problems in a gradual manner, we begin with a single-dimensional problem. The details of the methods have already been published in [BCK20]. In this chapter, we present an overview of the ideas, the methods and showcase a selection of numerical results.

5.1. The Mesh

Consider a partition of the domain $D = [x_{\text{min}}, x_{\text{max}}]$ into disjoint cells with the j-th cell being denoted by $C_j(t) = [x_{j-\frac{1}{2}}(t), x_{j+\frac{1}{2}}(t)]$. The time levels are denoted by t_n with the time step $\Delta t_n = t_{n+1} - t_n$. Inside a single time interval (t_n, t_{n+1}), the boundaries of the cells move with a constant velocity denoted by (5.1).

$$w_{j+\frac{1}{2}} = w^n_{j+\frac{1}{2}} \qquad t_n < t < t_{n+1} \tag{5.1}$$

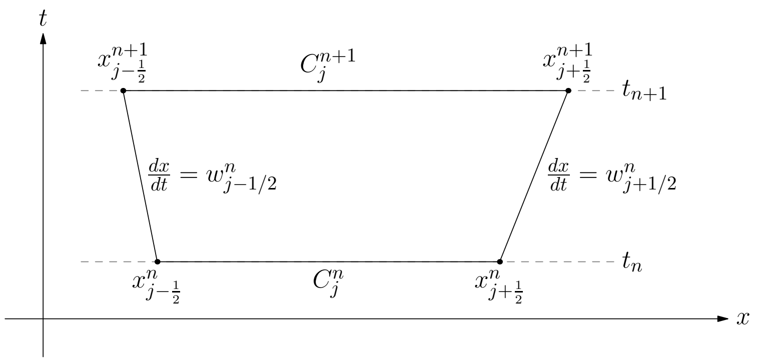

t
$x^{n+1}_{j-\frac{1}{2}}$
C^{n+1}_j
$x^{n+1}_{j+\frac{1}{2}}$
t_{n+1}
$\frac{dx}{dt} = w^n_{j-1/2}$
$\frac{dx}{dt} = w^n_{j+1/2}$
t_n
$x^n_{j-\frac{1}{2}}$
C^n_j
$x^n_{j+\frac{1}{2}}$
x

Figure 5.1.: Example of a space-time cell in the time interval (t_n, t_{n+1}).

We can use (5.1) to define a cell in space-time as shown in Figure 5.1. The location of the cell boundaries, the cell center and the cell length are given by (5.2).

$$x_{j+\frac{1}{2}}(t) = x^n_{j+\frac{1}{2}} + w_{j+\frac{1}{2}}(t - t_n) \qquad t_n < t < t_{n+1} \tag{5.2a}$$

$$x_j(t) = \frac{1}{2}\left[x_{j-\frac{1}{2}}(t) + x_{j+\frac{1}{2}}(t)\right] \qquad h_j(t) = x_{j+\frac{1}{2}}(t) - x_{j-\frac{1}{2}}(t) \tag{5.2b}$$

The mesh velocity inside the cell is given by the linear interpolation of the mesh velocities at the vertex as shown in (5.3).

$$w_j(t) = \frac{x_{j+\frac{1}{2}}(t) - x}{h_j(t)} w_{j-\frac{1}{2}} + \frac{x - x_{j-\frac{1}{2}}(t)}{h_j(t)} w_{j+\frac{1}{2}} \tag{5.3}$$

Sometimes, it is convenient to represent the geometry of the cell in terms of the reference coordinates $\xi \in [-1, 1]$ where ξ is given by (5.4).

$$\tau = t \qquad \xi = 2\frac{x - x_j}{h_j} \tag{5.4}$$

5.2. The ALE-DG Method

In the DG method, we approximate the solution of the Euler equations by piecewise polynomials which are allowed to be discontinuous at the cell boundaries as shown in Figure 5.2. We evolve the modes of the polynomial in the space-time cells of the ALE mesh. Finally, we use a standard TVB/TVD limiter to control the oscillations of the solution.

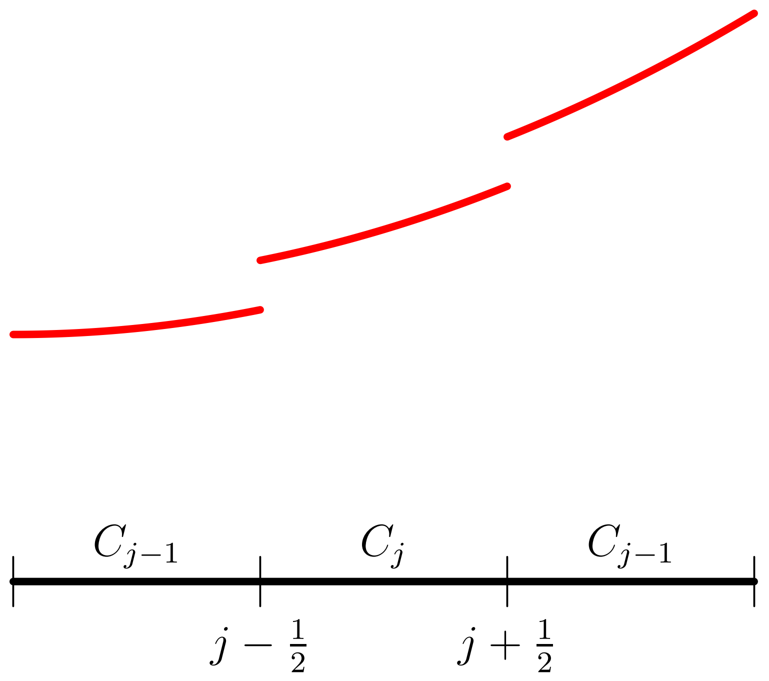


Figure 5.2.: Discontinuous Galerkin approximation of the solution.

For a given degree $k \geq 0$, the solution in the j-th cell is approximated by a polynomial of degree k as shown in (5.5) where $\{\mathbf{u}_{j,m} \in \mathbb{R}^3, 0 \leq m \leq k\}$ are the are the degrees of freedom associated with the j-th cell. The basis functions $\{\phi_m\}$ can be any set of polynomials that are orthogonal with respect to the cell geometry. In this work, we use functions derived from the Legendre polynomials P_m, $m \in \mathbb{N}$ as shown in (5.6).

$$\mathbf{u}_h(x,t) = \sum_{m=0}^{k} u_{j,m}(t)\phi_m(x,t) \tag{5.5}$$

$$\phi_m(x,t) = \hat{\phi}_m(\xi) = \sqrt{2m+1}P_m(\xi) \tag{5.6}$$

We can now calculate the rate of change of the of the l-th moment of the solution, which gives us the semi-discrete form of the ALE-DG scheme (5.7).

$$\begin{aligned} h_j^{n+1}\mathbf{u}_{j,l}^{n+1} &= h_j^n\mathbf{u}_{j,l}^n + \int_{t_n}^{t_{n+1}}\int_{x_{j-\frac{1}{2}}}^{x_{j+\frac{1}{2}}} \mathbf{g}(\mathbf{u}_h, w)\frac{\partial}{\partial x}\phi_l(x,t)\,\mathrm{d}x\,\mathrm{d}t \\ &+ \int_{t_n}^{t_{n+1}} \left[\hat{\mathbf{g}_{j-\frac{1}{2}}}(t)\phi_l\left(x_{j-\frac{1}{2}}(t),t\right) - \hat{\mathbf{g}_{j+\frac{1}{2}}}(t)\phi_l\left(x_{j+\frac{1}{2}}(t),t\right)\right]\,\mathrm{d}t \end{aligned} \tag{5.7}$$

Here, $\mathbf{g}$ is the ALE flux function defined by (5.8) and $\hat{\mathbf{g}}_{j-\frac{1}{2}}$ and $\hat{\mathbf{g}}_{j+\frac{1}{2}}$ are the numerical fluxes at the cell boundaries defined by (5.9).

$$\mathbf{g}(\mathbf{u}_h, w) = \mathbf{f}(\mathbf{u}) - w\mathbf{u} \tag{5.8}$$

$$\hat{\mathbf{g}_{j-\frac{1}{2}}}(t) = \mathbf{g}\left(\mathbf{u}^-_{j-\frac{1}{2}}(t), \mathbf{u}^+_{j+\frac{1}{2}}(t), w_{j-\frac{1}{2}}\right) \tag{5.9}$$

The above scheme, as it stands, is implicit in time. In order to make it explicit, we assume that we have available with us a predicted solution $\mathbf{U}_h$ in the time-step. We can use this predicted solution to compute the integrals on the right hand side of (5.7) and obtain the following explicit scheme (5.10). Here θ_r are the time quadrature weights and η_q are the spatial quadrature weights.

$$\begin{aligned} h_j^{n+1}\mathbf{u}_{j,l}^{n+1} &= h_j^n\mathbf{u}_{j,l}^n \\ &+ \Delta t_n \sum_r \theta_r h_j(\tau_r) \sum_q \eta_q \mathbf{g}(\mathbf{U}_h(x_q,\tau_r), w(x_q,\tau_r))\frac{\partial}{\partial x}\phi_l(x_q,\tau_r) \\ &+ \Delta t_n \sum_r \theta_r \left[\hat{\mathbf{g}}_{j-\frac{1}{2}}\left(\mathbf{U}_h(\tau_r)\right)\phi_l(x_{j-\frac{1}{2}}(\tau_r)) - \hat{\mathbf{g}}_{j+\frac{1}{2}}\left(\mathbf{U}_h(\tau_r)\right)\phi_l(x_{j+\frac{1}{2}}(\tau_r))\right] \end{aligned} \tag{5.10}$$

5.2.1. Mesh Velocity

In principle, we can choose any arbitrary mesh velocity for the ALE-DG scheme. However, in practice, we need to choose a mesh velocity close local fluid velocity in order to preserve the Lagrangian nature of the scheme. Since the solution is discontinuous, there is no unique fluid velocities at the mesh boundaries. Therefore, we need to choose a mesh velocity that is a good approximation to the multiple local fluid velocities at the boundary. In our work, we make two different choices of mesh velocities

1. Average Of Face Velocities (denoted by ADG)

$$w_{j-\frac{1}{2}} = \frac{1}{2}\left(v^-_{j-\frac{1}{2}} + v^+_{j+\frac{1}{2}}\right) \tag{5.11}$$

2. Velocity obtained by solving a linearized Riemann problem (denoted by RDG)

$$w_{j-\frac{1}{2}} = \frac{1}{2}\left(v^-_{j-\frac{1}{2}} + v^+_{j+\frac{1}{2}}\right) + \frac{1}{2}\left(\mathbf{u}^+_{j+\frac{1}{2}} - \mathbf{u}^-_{j-\frac{1}{2}}\right) \tag{5.12}$$

5.3. ALE-Aware Numerical Flux

The ALE scheme requires a numerical flux $\hat{\mathbf{g}}$ which is usually based on some approximate Riemann solver. The numerical flux is assumed to be consistent in the sense that it reduces to the exact flux in the case of a smooth solution as in (5.13). Here, we list some of the ALE numerical fluxes that we use in our work.

$$\hat{\mathbf{g}}(\mathbf{u}, \mathbf{u}, w) = \mathbf{g}(\mathbf{u}, w) \qquad \forall \mathbf{u} \in \mathbb{R}^3, w \in \mathbb{R} \tag{5.13}$$

Rusanov Flux

The Rusanov flux is a variant of the Lax-Friedrich flux and is given by

$$\hat{\mathbf{g}}(\mathbf{u}_l, \mathbf{u}_r, w) = \frac{1}{2}\left[\mathbf{g}(\mathbf{u}_l, w) + \mathbf{g}(\mathbf{u}_r, w)\right] - \frac{1}{2}\lambda_{lr}(\mathbf{u}_r - \mathbf{u}_l) \tag{5.14}$$

where

$$\lambda(\mathbf{u}_l, \mathbf{u}_r, w) = \max\left\{|v_l - w| + c_l, |v_r - w| + c_r\right\} \tag{5.15}$$

which is an estimate of the largest wave speed in the Riemann problem. Since the mesh velocity is close to the fluid velocity, the value of λ is close to the local sound speed. Thus the numerical dissipation is independent of the velocity scale.

Roe Flux

The Roe scheme [Roe81] is based on a local linearization of the conservation law and then exactly solving the Riemann problem for the linear approximation. The flux can be written as

$$\hat{\mathbf{g}}(\mathbf{u}_l, \mathbf{u}_r, w) = \frac{1}{2}\left[\mathbf{g}(\mathbf{u}_l, w) + \mathbf{g}(\mathbf{u}_r, w)\right] - \frac{1}{2}\left|A_w\right|(\mathbf{u}_r - \mathbf{u}_l) \tag{5.16}$$

where the Roe average matrix $A_w = A_w(\mathbf{u}_l, \mathbf{u}_r)$ satisfies

$$\mathbf{g}(\mathbf{u}_r, w) - \mathbf{g}(\mathbf{u}_l, w) = A_w(\mathbf{u}_r - \mathbf{u}_l) \tag{5.17}$$

where we define $|A_w| = R\,|\Lambda - wI|\,R^{-1}$. This matrix is evaluated at the average state $\mathbf{u}(\overline{\mathbf{q}})$, where $\overline{\mathbf{q}} = \frac{1}{2}(\mathbf{q}_l + \mathbf{q}_r)$ where $\mathbf{q} = \sqrt{\rho}[1, v, H]^T$ is the parameter vector introduced by Roe.

5.4. Analysis of the Scheme

$$h_j^{n+1}\mathbf{u}_j^{n+1} = h_j^n\mathbf{u}_j^n - \Delta t_n\left[\hat{\mathbf{g}}_{j+\frac{1}{2}} - \hat{\mathbf{g}}^n_{j-\frac{1}{2}}\right] \tag{5.18}$$

By considering the first-order version of the ALE-DG scheme, (5.18) with Rusanov flux (5.14) , we can show that the scheme preserves the positivity property of the solution if the time-step is chosen as shown in (5.19) for any $\beta \in [0, 1]$. Furthermore, we can show that the scheme can preserve constant states for any mesh motion.

$$\Delta t_n \leq \min_j \left\{ \frac{\left(1 - \frac{\beta}{2}\right) h_j^n}{\frac{1}{2}\left(\lambda_{j-\frac{1}{2}}^n + \lambda_{j+\frac{1}{2}}^n\right)}, \frac{\beta h_j^n}{\left|w_{j+\frac{1}{2}}^n - w_{j-\frac{1}{2}}^n\right|} \right\} \tag{5.19}$$

5.5. Mesh Adaptation

The size of the cells can change considerably during the time evolution process due to the near Lagrangian movement of the cell boundaries. Near shocks, the cells will be compressed to smaller sizes which will reduce the allowable time step since a CFL condition has to be satisfied. In some regions, e.g., inside expansion fans, the cell size can increase considerably which may lead to loss of accuracy. In order to avoid too small or too large cells from occurring in the grid, we implement cell merging and refinement into our scheme. If a cell becomes smaller than some specified size h_{min}, then it is merged with one of its neighbouring cells and the solution is transferred from the two cells to the new cell by performing an L^2 projection. If a cell size becomes larger than some specified size h_{max}, then this cell is refined into two cells by division and the solution is again transferred by L^2 projection. The use of L^2 projection for solution transfer ensures the conservation of mass, momentum and energy and preserves the accuracy in smooth regions. We also ensure that the cell sizes do not change drastically between neighbouring cells. To keep a track of refinement of cells, each cell is assigned an initial level equal to 0. The daughter cells created during refinement are assigned a level incremented from the parent cell, while the coarsened cells are assigned a level decremented from the parent cell.

The algorithm for refinement and coarsening is carried out in three sweeps over all the active cells. In the first sweep, we mark the cells for refinement or coarsening based on their size and the level of neighboring cells. Cells are marked for coarsening if the size is less than a pre-specified minimum size. They are marked for refinement if either the size of the cell is larger than the maximum size or if the level of the cell is less than the level of the neighboring cells. If none of the conditions are satisfied, the cells are marked for no change. In the second sweep, a cell is marked for refinement if both the neighboring cells are marked for refinement. A cell is also marked for refinement if the size of the cell is larger than twice the size of either of the neighboring cells, and is also larger than twice the minimum size. The last condition is inserted in order to prevent a cell being alternately marked for refinement and coarsening in consecutive adaptation cycles. In the third and final sweep, we again mark cells for refinement if both the neighboring cells are marked for refinement. Further, we ensure that a cell marked for refinement does not have a neighboring cell marked for a coarsening, since this can lead to an inconsistent mesh.

5.6. Numerical Results

5.6.1. Order of Accuracy Test Case

We study the convergence rate of the schemes by applying them to a problem with a known smooth solution. The initial condition is taken as

N	$k=1$		$k=2$	
	Error	Rate	Error	Rate
100	2.053E-02	-	2.277E-03	-
200	4.312E-03	2.251	3.425E-04	2.732
400	1.031E-03	2.064	4.565E-05	2.907
800	2.550E-04	2.015	5.812E-06	2.973
1600	6.356E-05	2.004	7.315E-07	2.990

Table 5.1.: Order of accuracy study on moving mesh using Rusanov flux using Higher Order Limiter [ZS13]

N	$k=1$		$k=2$		$k=3$	
	Error	Rate	Error	Rate	Error	Rate
100	4.370E-02	-	3.498E-03	-	3.883E-04	-
200	6.611E-03	2.725	4.766E-04	2.876	1.620E-05	4.583
400	1.332E-03	2.518	6.415E-05	2.885	9.376E-07	4.347
800	3.151E-04	2.372	8.246E-06	2.910	5.763E-08	4.239
1600	7.846E-05	2.280	1.031E-06	2.932	3.595E-09	4.180

Table 5.2.: Order of accuracy study on static mesh using Rusanov flux

$$\rho(x,0) = 1 + \exp(-10x^2) \tag{5.20}$$

$$u(x,0) = 1 \tag{5.21}$$

$$p(x,0) = 1 \tag{5.22}$$

whose exact solution is $\rho(x,t) = \rho(x-t,0)$, $u(x,t) = 1$, $p(x,t) = 1$. The initial domain is $[-5,+5]$ and the final time is $t = 1$ units. The results are presented using Rusanov and HLLC numerical fluxes. The L^2 norm of the error in density are shown in table (5.2), (5.3) for the static mesh and in table (5.4), (5.5) for the moving mesh. In each case, we see that the error behaves as $O(h^{k+1})$ which is the optimal rate we can expect for smooth solutions. In table (5.1), we show that the ALE DG methods preserves its higher order in presence of a limiter.

5.6.2. Smooth Test Case with Non-Constant Velocity

We also test the accuracy of our schemes on a isentropic problem with smooth solutions. The test case The initial conditions are given by

$$\rho(x,0) = 1 + 0.9999995\sin(\pi x) \qquad u(x,0) = 0 \qquad p(x,0) = \rho^\gamma(x,0) \tag{5.23}$$

with $\gamma = 3$ and periodic boundary conditions. For this kind of special isentropic problem, the Euler equations are equivalent to the two Burgers equations in terms of their two Riemann invariants which can then be used to derive the analytical solution. The errors are then computed with respect to the given analytical

N	$k = 1$		$k = 2$		$k = 3$	
	Error	Rate	Error	Rate	Error	Rate
100	4.582E-02		3.952E-03		3.464E-04	
200	9.611E-03	2.253	4.048E-04	3.287	2.058E-05	4.073
400	2.052E-03	2.240	4.640E-05	3.206	1.287E-06	4.036
800	4.803E-04	2.192	5.623E-06	3.152	8.061E-08	4.023
1600	1.184E-04	2.149	6.929E-07	3.119	5.050E-09	4.016

Table 5.3.: Order of accuracy study on static mesh using HLLC flux

N	$k = 1$		$k = 2$		$k = 3$	
	Error	Rate	Error	Rate	Error	Rate
100	2.331E-02	-	3.979E-03	-	8.633E-04	-
200	6.139E-03	1.9250	4.058E-04	3.294	1.185E-05	6.186
400	1.406E-03	2.0258	5.250E-05	3.122	7.079E-07	5.126
800	3.375E-04	2.0366	6.626E-06	3.077	4.340E-08	4.760
1600	8.278E-05	2.0344	8.304E-07	3.057	2.689E-09	4.573

Table 5.4.: Order of accuracy study on moving mesh using Rusanov flux

N	$k = 1$		$k = 2$		$k = 3$	
	Error	Rate	Error	Rate	Error	Rate
100	1.590E-02		1.626E-03		1.962E-04	
200	4.042E-03	1.977	2.072E-04	2.972	1.269E-05	3.950
400	1.014E-03	1.985	2.605E-05	2.982	7.983E-07	3.971
800	2.538E-04	1.990	3.261E-06	2.988	4.997E-08	3.980
1600	6.349E-05	1.992	4.077E-07	2.991	3.124E-09	3.985

Table 5.5.: Order of accuracy study on moving mesh using HLLC flux

N	$k = 1$		$k = 2$		$k = 3$	
	Error	Rate	Error	Rate	Error	Rate
100	1.735E-02	-	1.798E-03	-	2.351E-04	-
200	4.179E-03	2.051	2.848E-04	2.676	1.416E-05	4.069
400	1.054E-03	2.035	4.301E-05	2.703	8.578E-07	4.041
800	2.615E-04	1.943	6.012E-06	2.838	5.476E-08	3.958
1600	7.279E-05	1.852	8.000E-07	2.909	3.505E-09	3.966

Table 5.6.: Order of accuracy study on moving mesh using HLLC flux with randomly perturbed mesh velocity

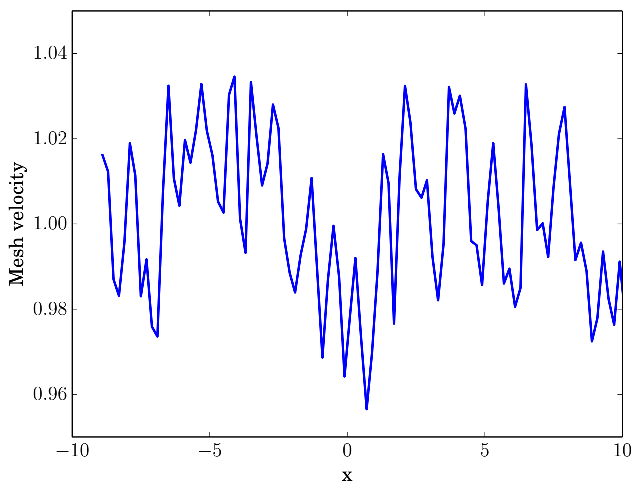


Figure 5.3.: Example of randomized velocity distribution for smooth test case

N	$k = 1$		$k = 2$	
	Error	Rate	Error	Rate
100	8.535E-03	-	1.033E-03	-
200	1.958E-03	2.124	1.221E-04	3.08
400	4.721E-04	2.052	1.581E-05	2.95
800	1.238E-04	1.931	2.14E-06	2.89
1600	3.563E-05	1.796	2.63E-07	3.02

Table 5.7.: Order of accuracy study on fixed mesh using Roe flux with Non-Constant Velocity Smooth Test Case

solution. In contrast to the previous test case, the velocity and pressure are not constant which makes this a more challenging test case. We run the simulation with a WENO-type limiter from [Zhu+13] and positivity limiter enabled. As we can see from Tables 5.7, 5.8, the rate of convergence is maintained for the moving mesh method with the moving mesh methods exhibiting much lower errors.

5.6.3. Single contact wave

In this example, we choose a Riemann problem which gives rise to a single contact wave in the solution that propagates with a constant speed. The initial condition is given by

$$(\rho, v, p) = \begin{cases} (2.0, 1.0, 1.0) & \text{if } x < 0.5 \\ (1.0, 1.0, 1.0) & \text{if } x > 0.5 \end{cases}$$

and the contact wave moves with a constant speed of 1.0. The solution on static and moving meshes are shown in figure (5.4) at time $t = 0.5$ using Roe flux. The moving mesh is able to exactly resolve the contact wave while the static mesh scheme adds considerable numerical dissipation that smears the discontinuity over many cells. The accurate resolution of contact waves is a key advantage of such moving mesh methods, which are capable of giving very good resolution of the contact discontinuity even on coarse meshes.

N	$k = 1$		$k = 2$	
	Error	Rate	Error	Rate
100	4.235E-03	-	2.238E-04	-
200	1.058E-03	2.001	3.255E-05	2.87
400	2.586E-04	2.035	4.301E-05	3.133
800	5.804E-05	2.155	5.762E-06	2.901
1600	1.271E-05	2.192	7.401E-07	2.96

Table 5.8.: Order of accuracy study on moving mesh using Roe flux with Non-Constant Velocity Smooth Test Case

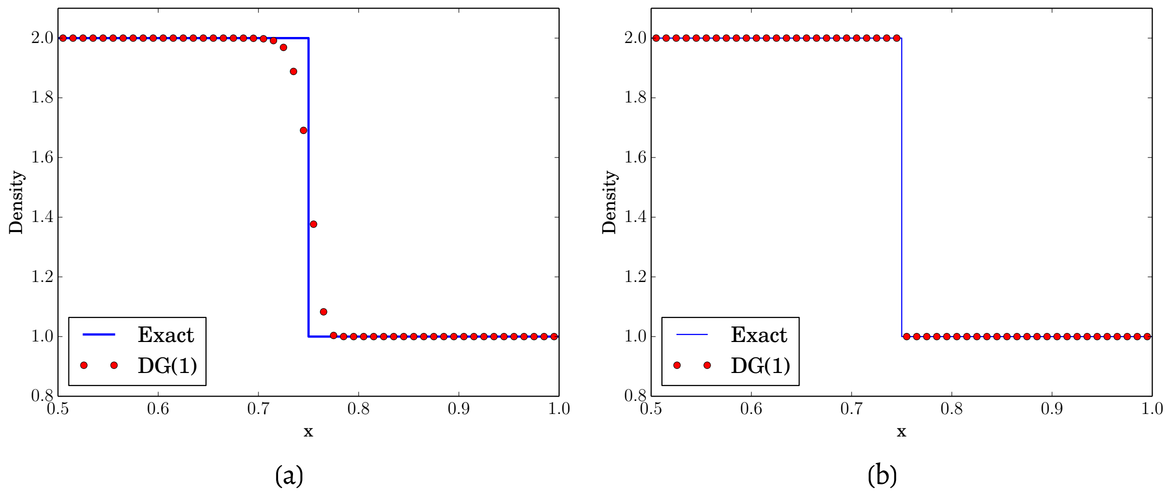


Figure 5.4.: Single contact wave using Roe flux and 100 cells: (a) static mesh, (b) moving mesh

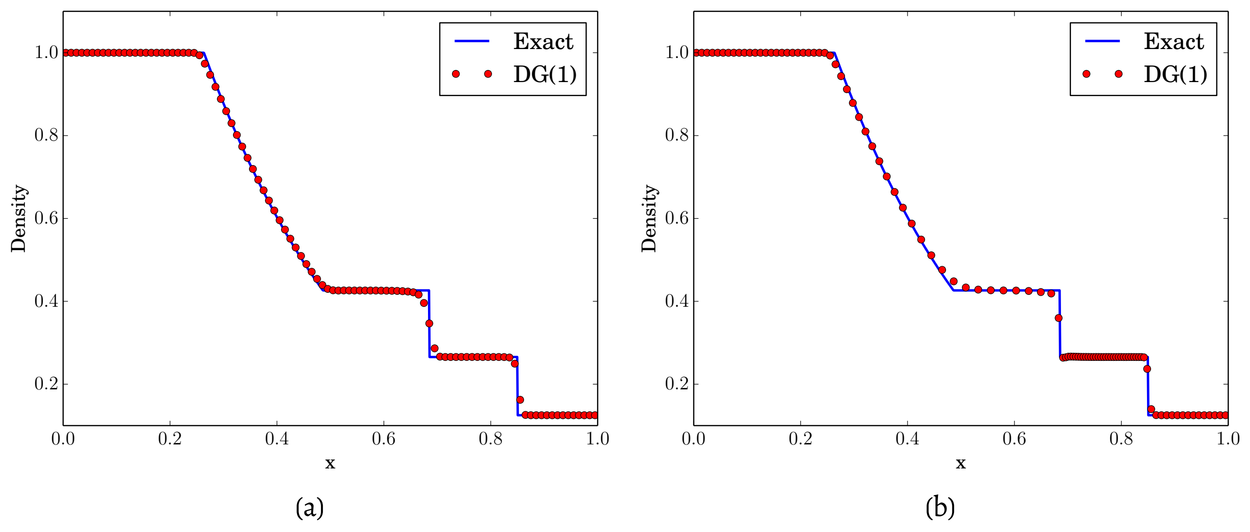


Figure 5.5.: Sod problem using Roe flux, 100 cells and TVD limiter: (a) static mesh (b) moving mesh

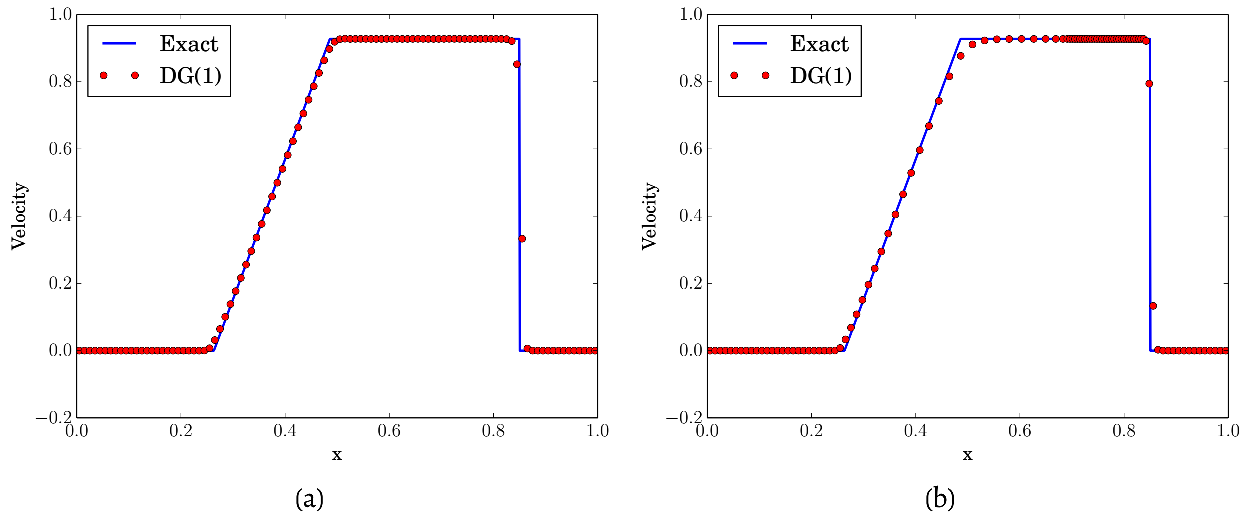


Figure 5.6.: Sod problem using Roe flux, 100 cells and TVD limiter: (a) static mesh (b) moving mesh

5.6.4. Sod problem

The initial condition for the Sod test case is given by [Sod78]

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

and the solution is computed upto a final time of $T = 0.2$ with the computational domain being $[0, 1]$. Since the fluid velocity is zero at the boundary, the computational domain does not change with time for the chosen final time. The exact solution consists of a rarefaction fan, a contact wave and a shock wave. In figure (5.5), we show the results obtained using Roe flux with 100 cells and TVD limiter on static and moving mesh. The contact wave is considerably well resolved on the moving mesh as compared to the static mesh due to reduced numerical dissipation on moving meshes.

To study the Galilean invariance or the dependence of the solution on the choice of coordinate frame, we

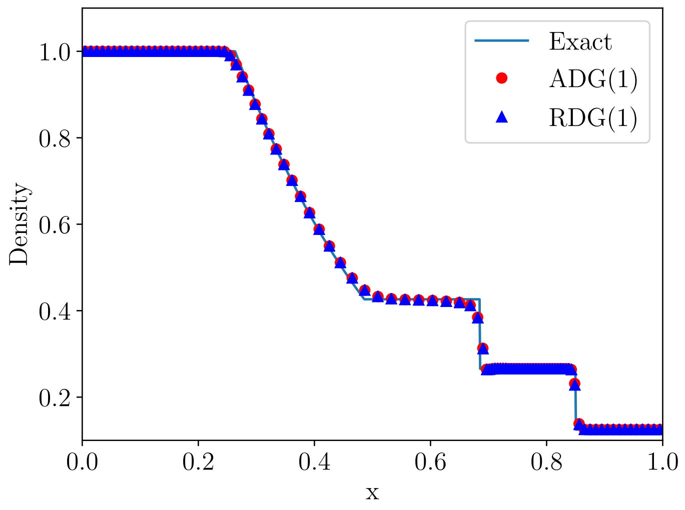


Figure 5.7.: Sod problem using Roe flux, 100 cells and TVD limiter. ADG : Average Velocity, RDG : Linearized Riemann Velocity

V	0	10	100
static mesh	144	810	6807
moving mesh	176	176	176

Table 5.9.: Number of iterations required to reach time $t = 0.2$ for Sod test for different boost velocity of the coordinate frame

add a boost velocity of $V = 10$ or $V = 100$ to the coordinate frame, while implies the initial fluid velocity is $v(x, 0) = V$and the other quantities remain as before. Figure (5.8a) shows that the accuracy of the static mesh results degrades with increase in velocity of the coordinate frame, particularly the contact discontinuity is highly smeared. The results given in figure (5.8b) clearly show the independence of the results on the moving mesh with respect to the coordinate frame velocity. The allowed time step from CFL condition decreases with increase in coordinate frame speed for the static mesh case, while in case of moving mesh, it remains invariant. This means that in case of static mesh, we have to perform more time steps to reach the same final time as shown in table (5.9), which increases the computational time. Thus the moving mesh scheme has the additional advantage of allowing a larger time step compared to the fixed mesh scheme.

Finally, we compute the solutions using quadratic and cubic polynomials and the results are shown in figure (5.9). The solutions look similar to the case of linear polynomials and have the same sharp resolution of discontinuities.

5.6.5. Lax problem

The initial condition is given by

$$(\rho, v, p) = \begin{cases} (0.445, 0.698, 3.528) & \text{if } x < 0 \\ (0.5, 0.0, 0.571) & \text{if } x > 0 \end{cases}$$

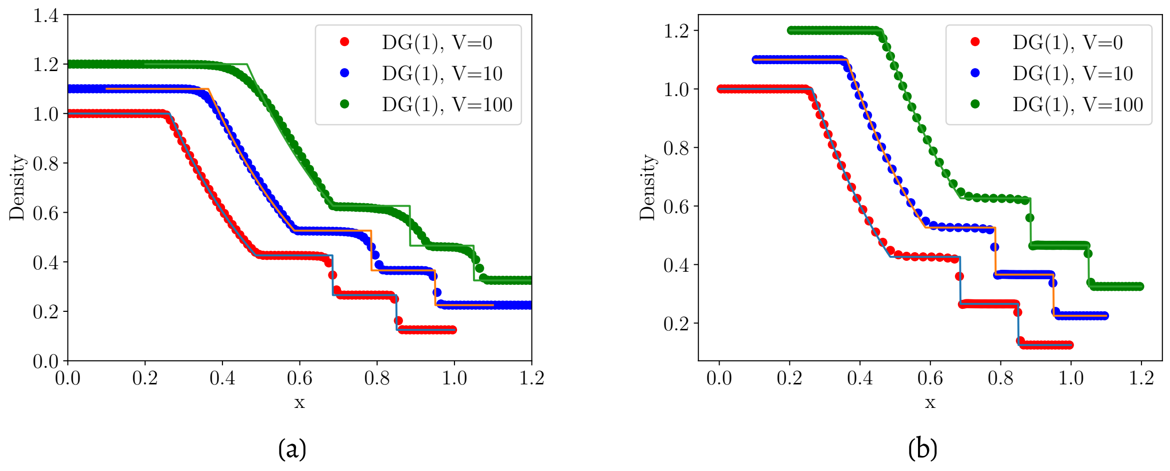


Figure 5.8.: Effect of coordinate frame motion on Sod problem using Roe flux, 100 cells and TVD limiter: (a) static mesh (b) moving mesh

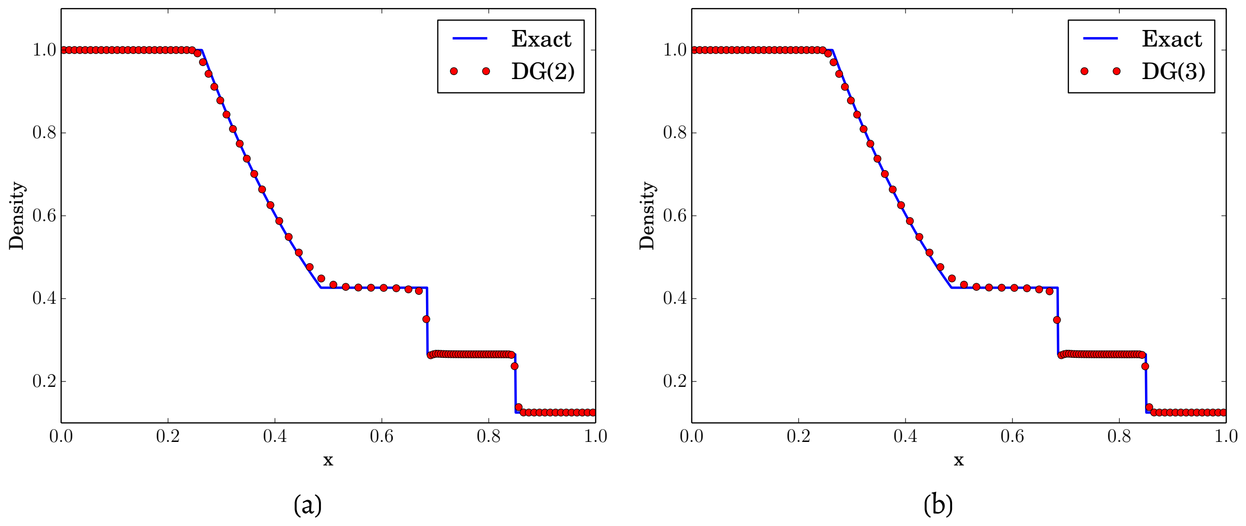


Figure 5.9.: Sod problem on moving mesh using Roe flux, 100 cells and TVD limiter: (a) Degree = 2 (b) Degree = 3

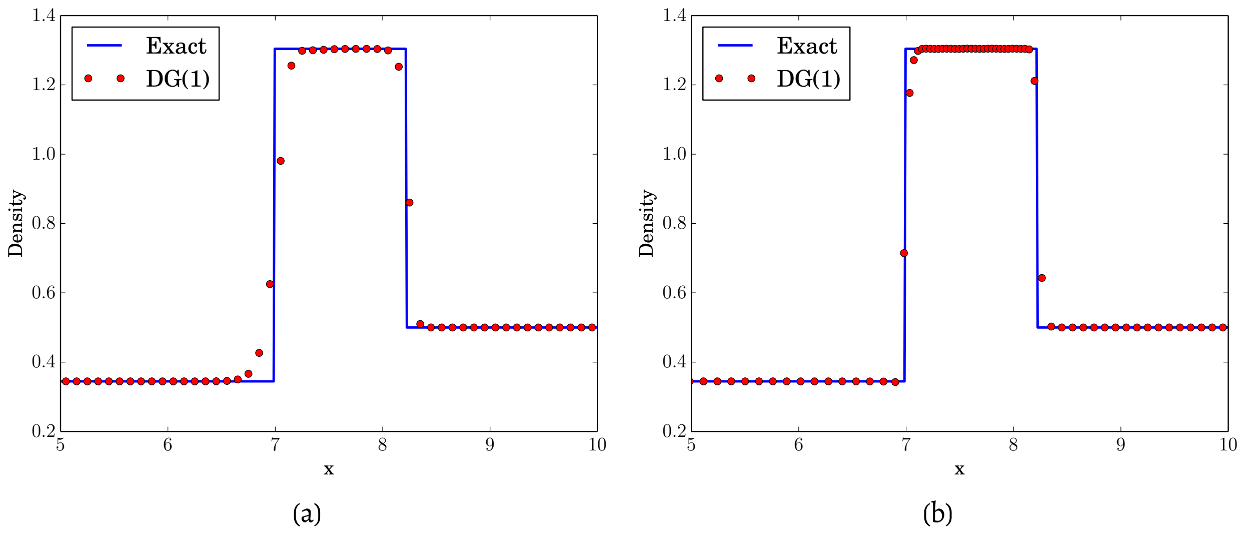


Figure 5.10.: Lax problem using HLLC flux, 100 cells and TVD limiter: (a) static mesh (b) moving mesh

The computational domain is $[-10, +10]$ and we compute the solution up to a final time of $T = 1.3$. This problem has a strong shock and a contact wave that is difficult to resolve accurately. The zoomed view of density is shown at the final time in figure (5.10), and we observe the moving mesh results are more accurate for the contact wave, which is the first discontinuity in the figure. The second discontinuity is a shock which is equally well resolved in both cases. We can observe that the grid is automatically clustered in the region between the contact and shock wave, but no explicit grid adaptation was used in this simulation.

5.6.6. Shu-Osher problem

The initial condition is given by [SO88]

$$(\rho, v, p) = \begin{cases} (3.857143, 2.629369, 10.333333) & \text{if } x < -4 \\ (1 + 0.2\sin(5x), 0.0, 1.0) & \text{if } x > -4 \end{cases}$$

which involves a smooth sinusoidal density wave which interacts with a shock. The domain is $[-5, +5]$ and the solution is computed up to a final time of $T = 1.8$. The solutions are shown in figure (5.12a)-(5.12b) on static and moving meshes using 200 cells and TVD limiter. The moving mesh scheme is considerably more accurate in resolving the sinusoidal wave structure that arises after interaction with the shock. In figure (5.12c) we compute the solution on static mesh with TVB limiter and the parameter $M = 100$. In this case the solutions on static mesh are more accurate compared to the case of TVD limiter but still not as good as the moving mesh results. The moving mesh result has more than 200 cells in the interval $[-5, +5]$ at the final time since cells enter the domain from the left side. Hence in figure (5.12d), we show the static mesh results with 300 cells and using TVB limiter. The results are further improved for the static mesh case but still not as accurate as the moving mesh case. The choice of parameters in the TVB limiter is very critical but we do not have a rigorous algorithm to choose a good value for this. Hence it is still advantageous to use the moving mesh scheme which gives improved solutions even with TVD limiter.

The above results show the ALE method is very accurate in terms of the cell averages. In figure (5.13), we show a zoomed view of density and pressure, where we also plot the linear polynomial solution. The slope of the solution is not accurately predicted with the Roe scheme and there are spurious contact discontinuities

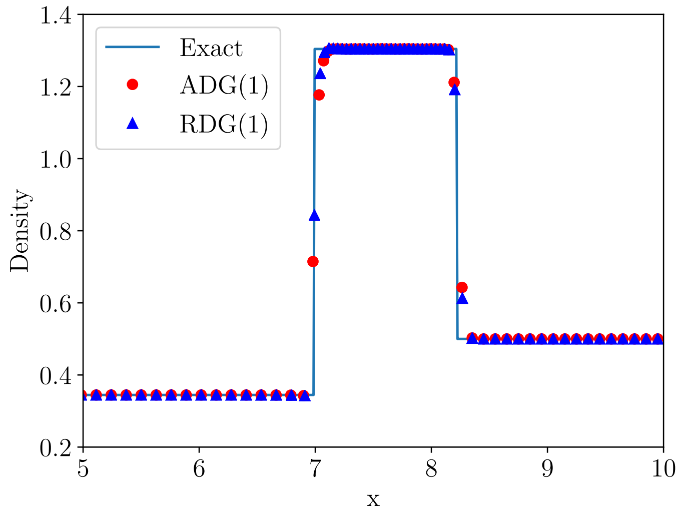


Figure 5.11.: Lax problem using HLLC flux, 100 cells and TVD limiter. ADG : Average Velocity, RDG : Linearized Riemann Velocity

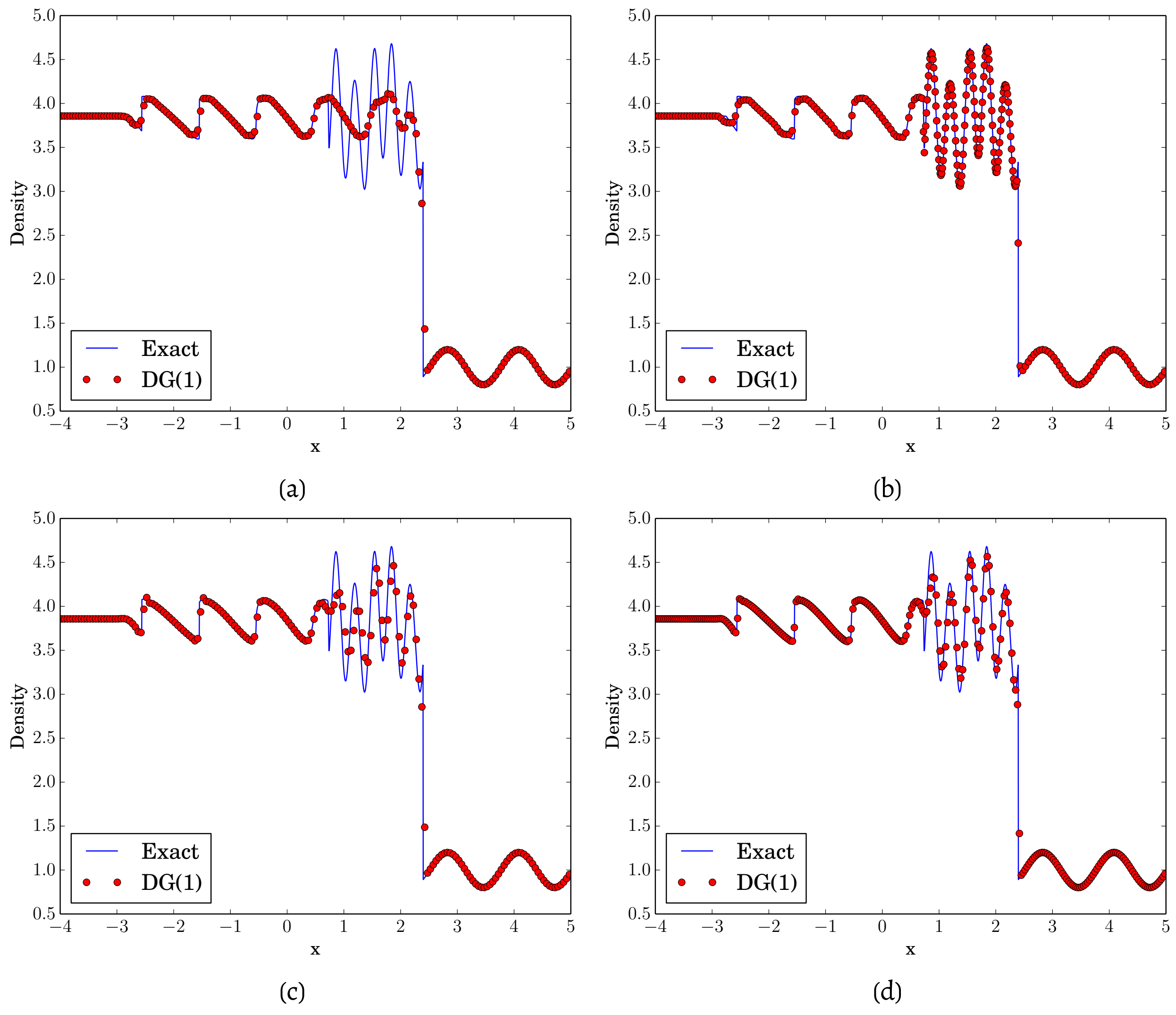


Figure 5.12.: Shu-Osher problem using Roe flux: (a) static mesh, 200 cells, $M = 0$ (b) moving mesh, 200 cells, $M = 0$ (c) static mesh, 200 cells, $M = 100$ (d) static mesh, 300 cells, $M = 100$

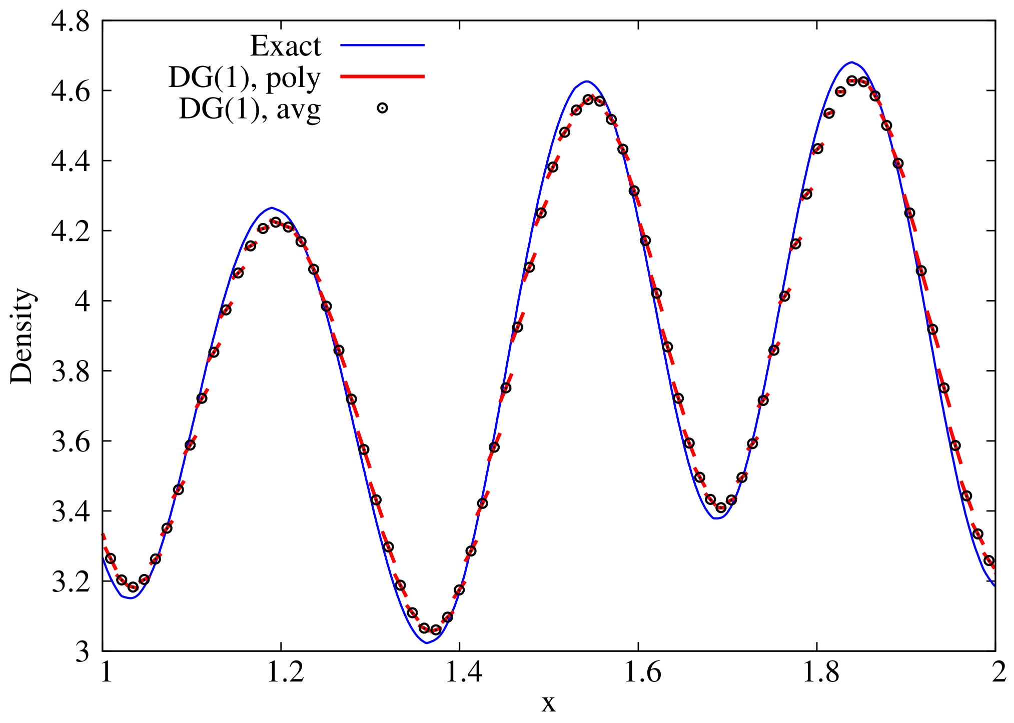


Figure 5.13.: Shu-Osher problem using Roe flux on moving mesh

as the pressure and velocity are nearly continuous. This behaviour is observed with all contact preserving fluxes like Roe, HLLC and HLL-CPS but not with the Rusanov flux. Due to the almost Lagrangian character of the scheme, the eigenvalue corresponding to the contact wave, $\lambda_2 = v - w$, is nearly zero, which leads to loss of dissipation in the corresponding characteristic field. If a spurious contact wave is generated during the violent dynamics, then this wave will be preserved by the scheme leading to wrong solutions. We modify the Roe scheme by preventing this eigenvalue from becoming too small or zero, which is similar to the approach used for the entropy fix. The eigenvalue $|\lambda_2|$ used in the dissipative part of the Roe flux is determined from

$$|\lambda_2| = \begin{cases} |v - w| & \text{if } |v - w| > \delta = \alpha c \\ \frac{1}{2}(\delta + |v - w|^2/\delta) & \text{otherwise} \end{cases}$$

With this modification and using $\alpha = 0.1$, the solution on moving mesh is shown in figure (5.14) and we do not observe the spurious contact discontinuities which arise with the standard Roe flux, while at the same time, the solution accuracy compares favourably with the previous results that did not use the eigenvalue fix.

We next compute the solutions using quadratic polynomials. Figure (5.15) shows the results obtained with the TVD limiter which shows the dramatically better accuracy that is achieved on moving mesh compared to static mesh. In figure (5.16) we perform the same computation with a WENO limiter taken from [ZS13]. The static mesh results are now improved over the case of TVD limiter but still not as good as the moving mesh results in terms of capturing the extrema. In figure (5.17) we show a zoomed view of the results on moving mesh with TVD and WENO limiter. We see that the TVD limiter is also able to capture all the features and is almost comparable to the WENO limiter.

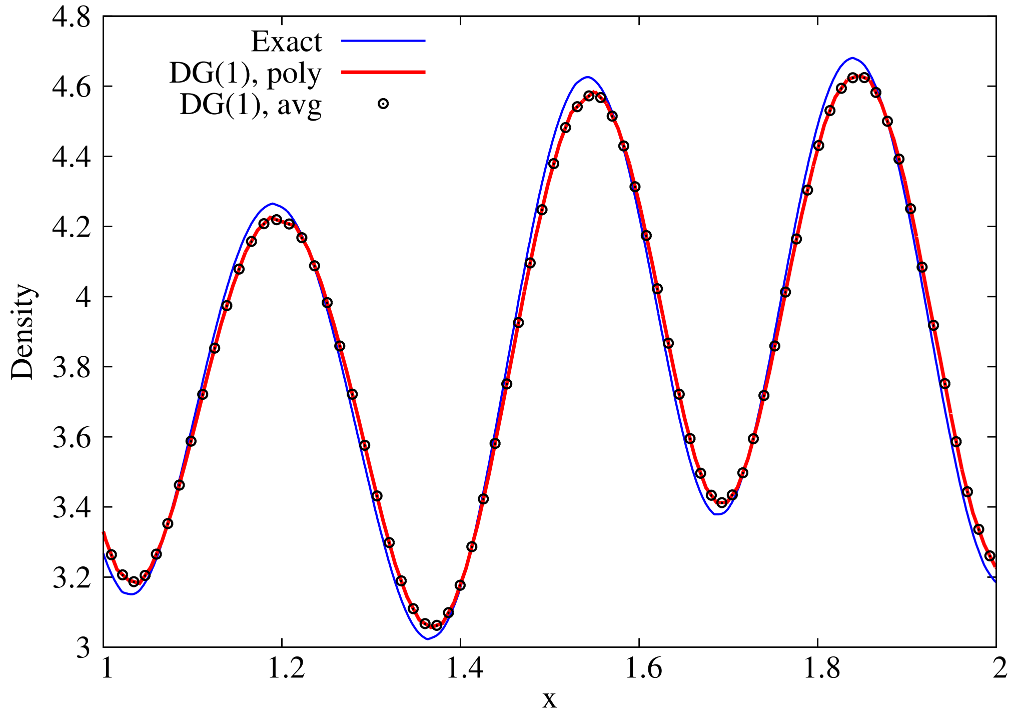


Figure 5.14.: Shu-Osher problem using modified Roe flux on moving mesh

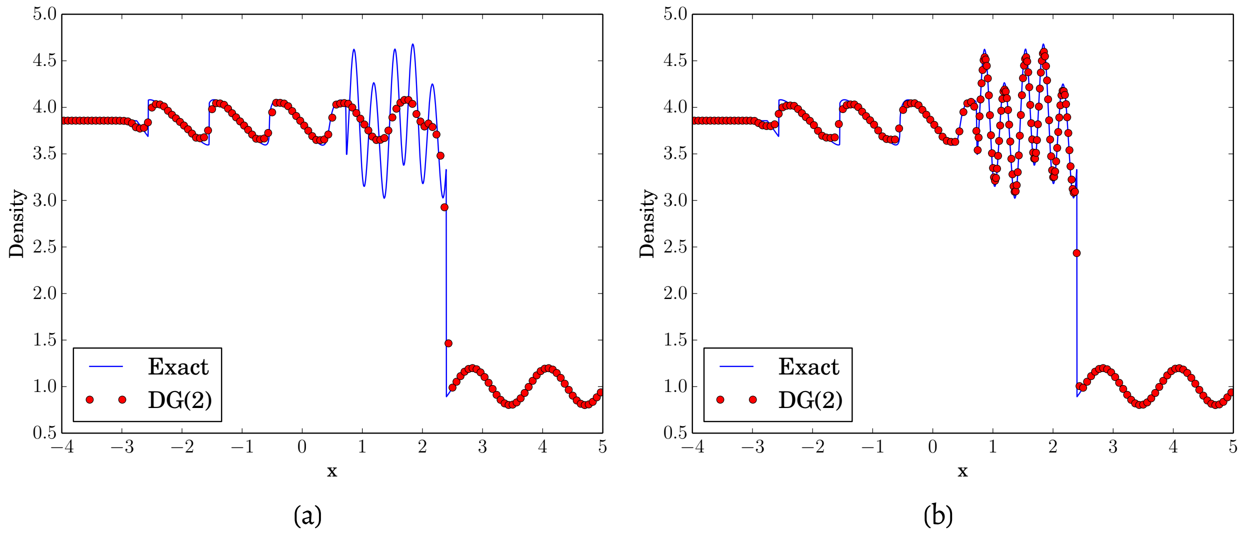


Figure 5.15.: Shu-Osher problem using modified Roe flux, TVD limiter, quadratic polynomials and 150 cells. (a) static mesh, (b) moving mesh

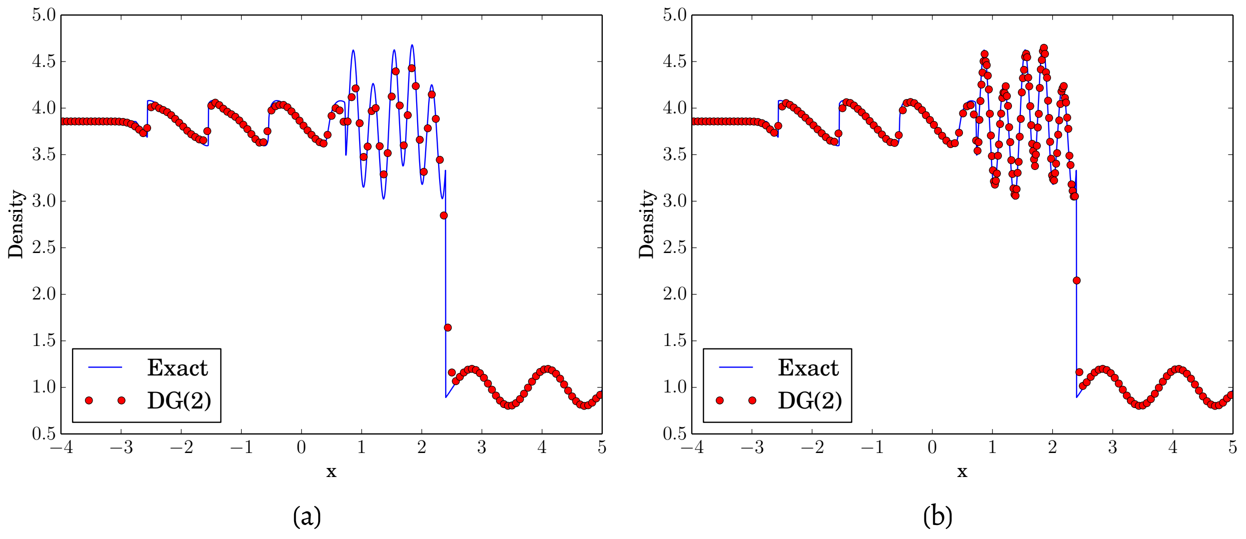


Figure 5.16.: Shu-Osher problem using modified Roe flux, WENO limiter, quadratic polynomials and 150 cells. (a) static mesh, (b) moving mesh

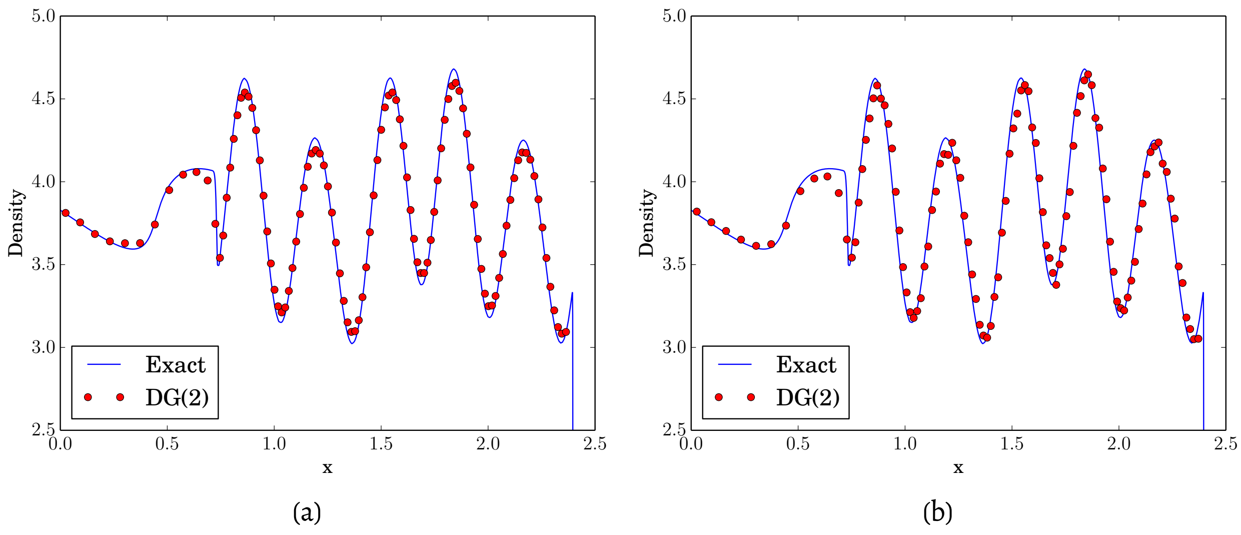


Figure 5.17.: Shu-Osher problem using modified Roe flux, moving mesh, quadratic polynomials and 150 cells. (a) TVD limiter, (b) WENO limiter

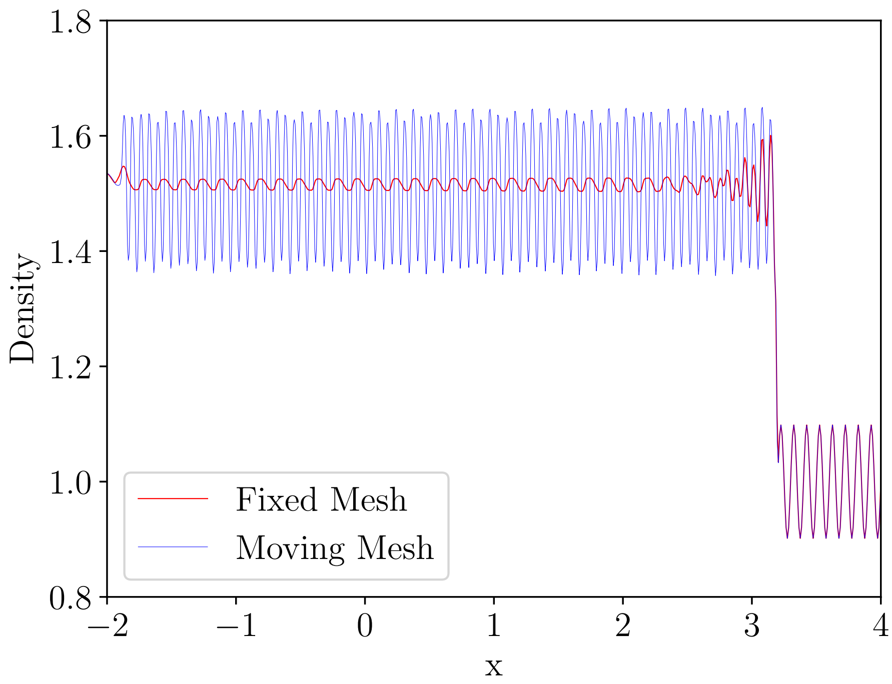


Figure 5.18.: Titarev Problem with HLLC flux, 1000 cells and TVD limiter

5.6.7. Titarev-Toro problem

Titarev-Toro problem is an extension of the Shu-Osher problem [TT04] to test a severely oscillatory wave interacting with a shock wave. It aims to test the ability of higher-order methods to capture the extremely high frequency waves. The initial condition is given by

$$(\rho, v, p) = \begin{cases} (1.515695, 0.523346, 1.805), & -5 < x \leq -4.5 \\ (1 + 0.1\sin(20\pi x), 0, 1), & -4.5 < x \leq 5 \end{cases} \tag{5.24}$$

The computation is carried out on a mesh of 1000 cells with the final time $T = 5$ and the density at this final time is shown in Figures (5.18), (5.19). The fixed mesh is not able to resolve the high frequency oscillations due to dissipation in the fluxes and the TVD limiter, but the ALE scheme gives an excellent resolution of these high frequency oscillations. Note that the ALE scheme also uses the same TVD limiter but it is still able to resolve the solution to a very degree of accuracy. This result again demonstrates the superior accuracy that can be achieved by using a nearly Lagrangian ALE scheme in problems involving interaction of shocks and smooth flow structures.

5.6.8. 123 problem

The initial condition is given by [Tor09]

$$(\rho, v, p) = \begin{cases} (1.0, -2.0, 0.4) & x < 0.5 \\ (1.0, +2.0, 0.4) & x > 0.5 \end{cases}$$

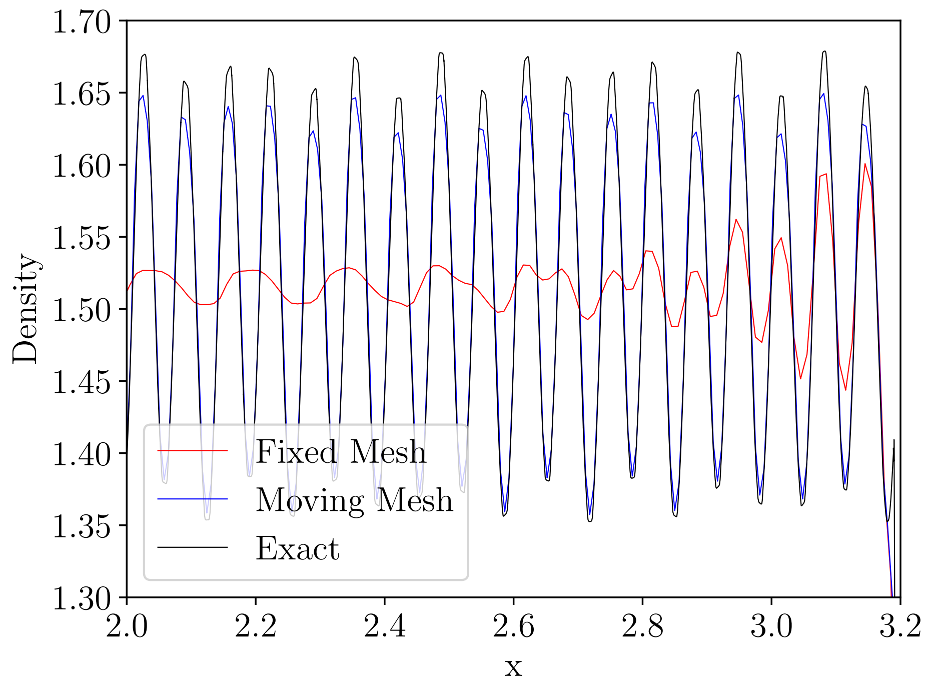


Figure 5.19.: Titarev Problem with HLLC flux, 1000 cells and TVD limiter. (Zoomed Version)

The computational domain is $[0,1]$ and the final time is $T = 0.15$. The density using 100 cells is shown in figure (5.20) with static and moving meshes. The mesh motion does not significantly improve the solution compared to the static mesh case since the solution is smooth. On the contrary, the mesh becomes rather coarse in the expansion region, though the solution is still well resolved. However, severe expansion may lead to very coarse meshes which may be undesirable. To prevent very coarse cells, we switch on the mesh refinement algorithm as described before and use the upper bound on the mesh size as $h_{\max} = 0.05$. The resulting solution is shown in figure (5.21) where the number of cells has increased to 108 at the time shown. The central expansion region is now resolved by more uniformly sized cells compared to the case of no grid refinement.

5.6.9. Blast problem

The initial condition is given by

$$(\rho, v, p) = \begin{cases} (1.0, 0.0, 1000.0) & x < 0.1 \\ (1.0, 0.0, 0.01) & 0.1 < x < 0.9 \\ (1.0, 0.0, 100.0) & x > 0.9 \end{cases}$$

with a domain of $[0,1]$ and the final time is $T = 0.038$. A reflective boundary condition is used at $x = 0$ and $x = 1$. A mesh of 400 cells is used for this simulation and in case of moving mesh, we perform grid adaptation with $h_{min} = 0.001$ since some cells become very small during the collision of the two shocks. The positivity preserving limiter of [ZS10] is applied together with TVD limiter and HLLC flux. The static mesh results shown in figure (5.23a) indicate too much numerical viscosity in the contact wave around $x = 0.6$. This wave is more accurately resolved in the moving mesh scheme as seen in figure (5.23b) which is an advantage due

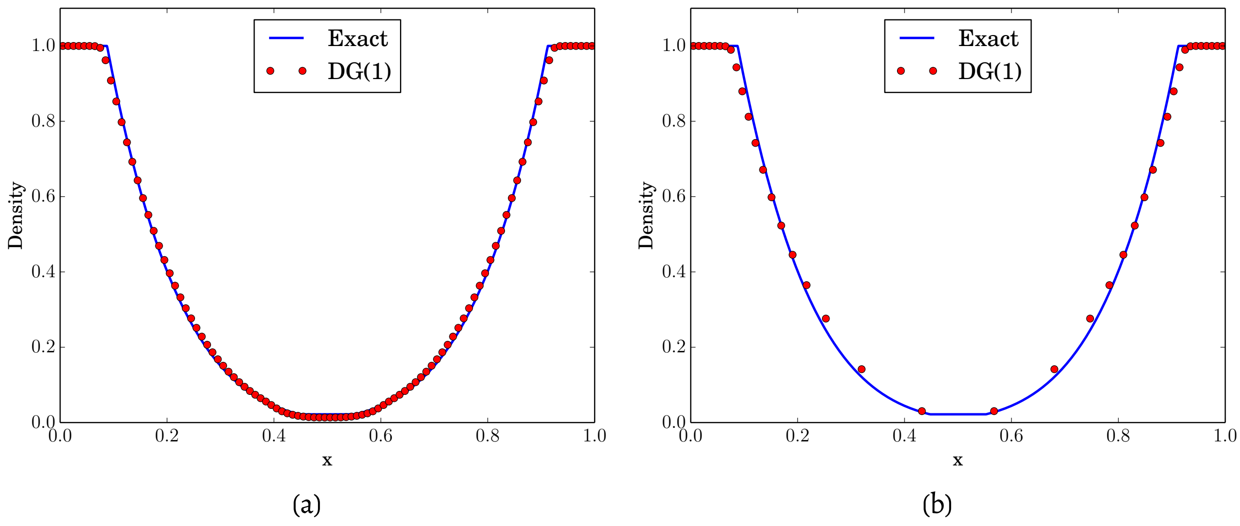


Figure 5.20.: 123 problem using HLLC flux and 100 cells: (a) static mesh, (b) moving mesh

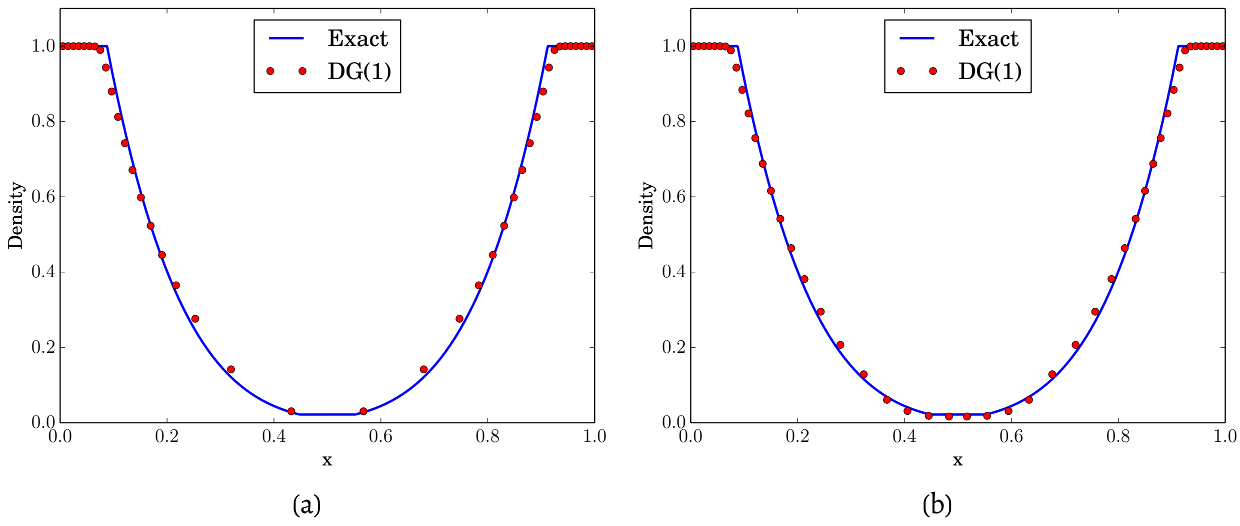


Figure 5.21.: 123 problem using HLLC flux and grid refinement: (a) static mesh, (b) moving mesh with mesh adaptation ($h_{\max} = 0.05$) leading to 108 cells at final time

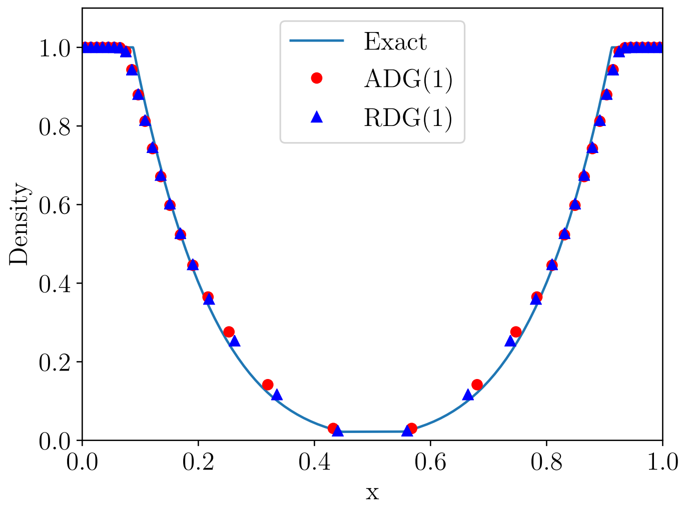


Figure 5.22.: 123 problem using HLLC flux, 100 cells and TVD limiter. ADG : Average Velocity, RDG : Linearized Riemann Velocity

to the ALE scheme and is a very good indicator of the scheme accuracy as this is a very challenging feature to compute accurately. We next compute the same problem using quadratic polynomials with all other parameters being as before. The solutions are shown in figure (5.24) and indicate that the Lagrangian moving mesh scheme is more accurate in resolving the contact discontinuity. The higher polynomial degree does not show any major improvement in the solution compared to the linear case, which could be a consequence of the strong shock interactions present in this problem, see figure (4.11-4.12) in [ZS13] and figure (3.7) in [ZQ16] in comparison to current results.

5.6.10. Le Blanc shock tube test case

The Le Blanc shock tube test case is an extreme shock tube problem where the initial discontinuity separates a region of high energy and density from one of low energy and density. This is a much more severe test than the Sod problem and hence more challenging for numerical schemes. The computational domain is $0 \leq x \leq 9$ and is filled with an ideal gas with $\gamma = 5/3$. The gas is initially at rest and we perform the simulation up to a time of $T = 6$ units. The initial discontinuity is at $x = 3$ and the initial condition is given by

$$(\rho, v, p) = \begin{cases} (1.0, 0.0, 0.1) & \text{if } x < 3 \\ (0.001, 0.0, 10^{-7}) & \text{if } x > 3 \end{cases} \tag{5.25}$$

Note that both the density and pressure have a very large jump in the initial condition. The solution that develops from this initial condition consists of a rarefaction wave moving to the left and a contact discontinuity and a strong shock moving to the right. In Figure (5.26), we show the comparison of the internal energy profile at final time between a fixed mesh solution and moving mesh solutions with two different mesh velocities as described before. Most methods tend to generate a very large spike in the internal energy in the contact region, e.g., compare with Figure (11) in [Lou05], while the present ALE method here is able to give a better profile. We plot the pressure profile in Figure (5.27) which shows that the ALE scheme is able to better represent the region around the contact wave as compared to fixed mesh method.

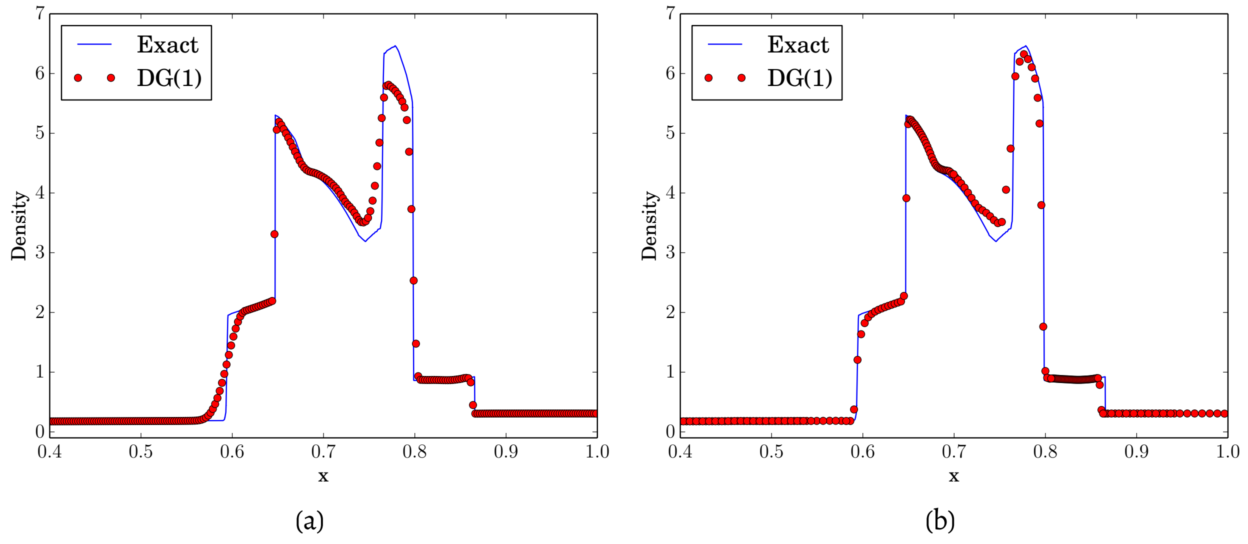


Figure 5.23.: Blast problem using HLLC flux and 400 cells. (a) static mesh, (b) moving mesh with adaptation ($h_{min} = 0.001$) leading to 303 cells at final time.

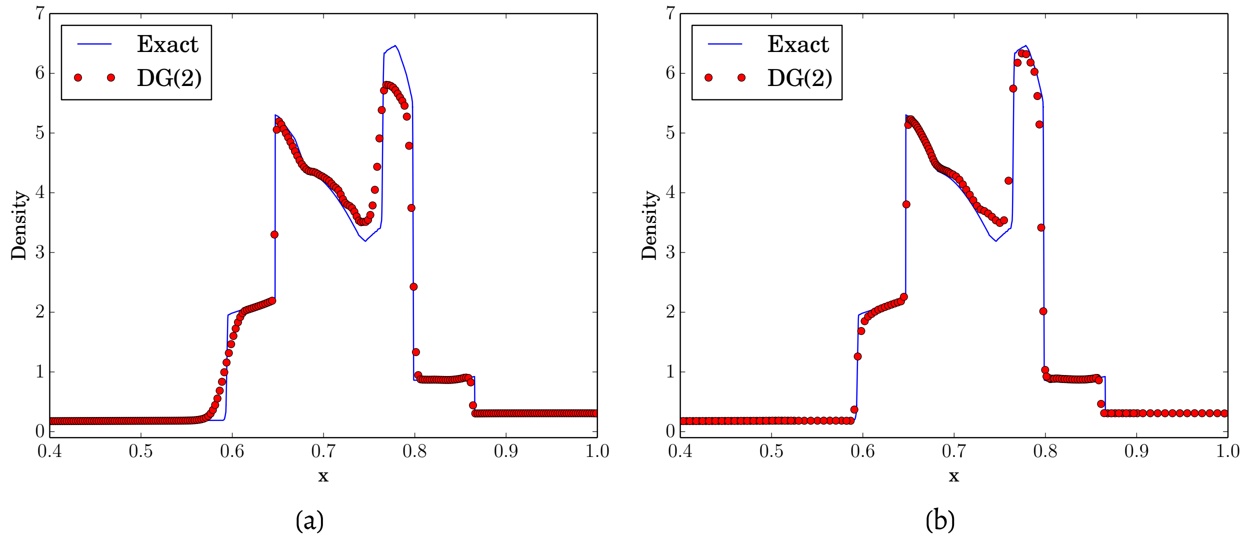


Figure 5.24.: Blast problem using HLLC flux, quadratic polynomials and 400 cells. (a) static mesh, (b) moving mesh with adaptation ($h_{min} = 0.001$) leading to 293 cells at final time.

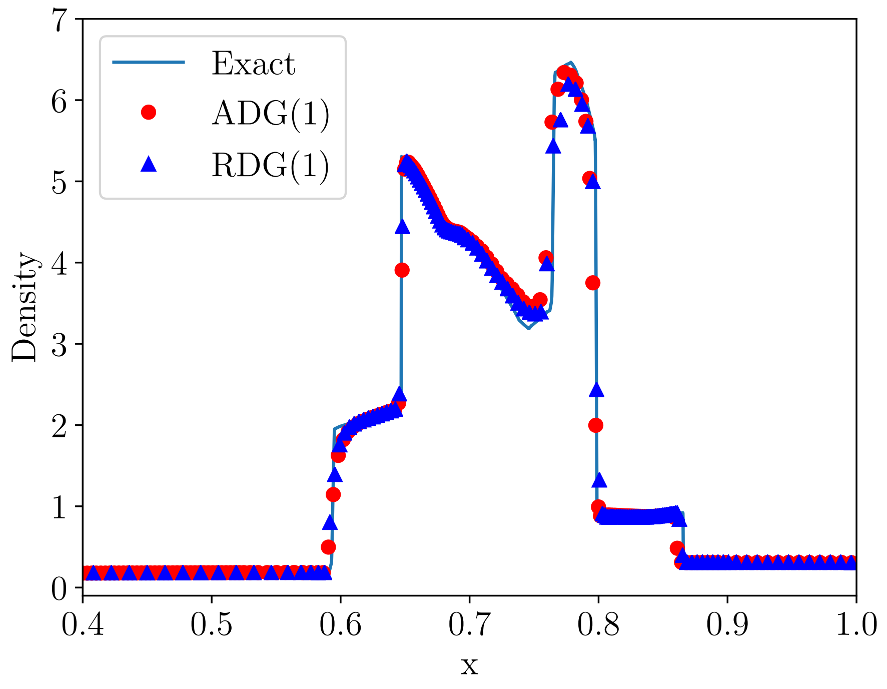


Figure 5.25.: Blast problem using HLLC flux, 100 cells and TVD limiter. ADG : Average Velocity, RDG : Linearized Riemann Velocity

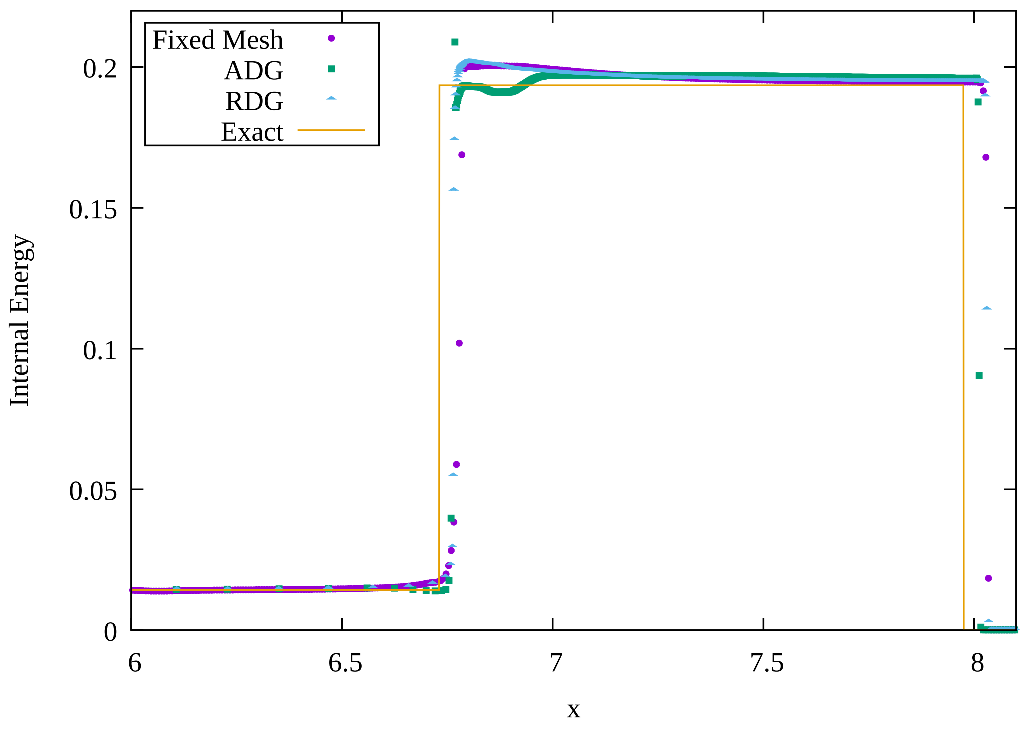


Figure 5.26.: Internal energy for Le Blanc Shock Tube with Rusanov flux, 1400 cells and TVD limiter, ADG : Average Velocity, RDG : Linearized Riemann Velocity

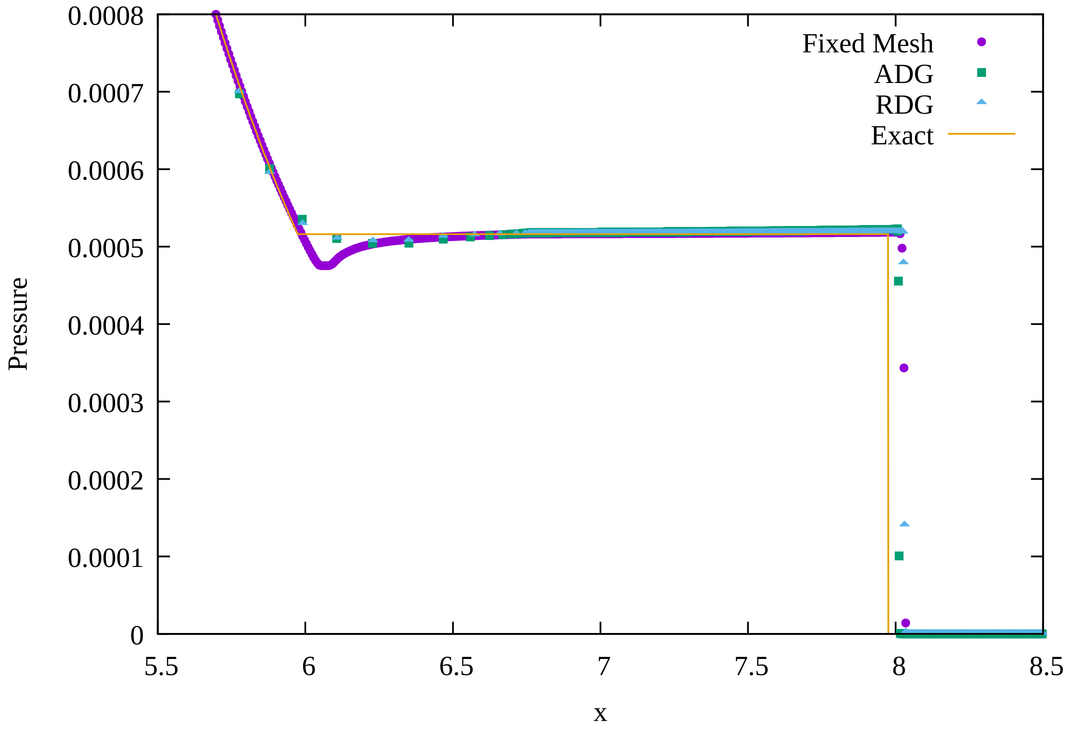


Figure 5.27.: Pressure for Le Blanc Shock Tube with Rusanov flux, 1400 cells and TVD limiter, ADG : Average Velocity, RDG : Linearized Riemann Velocity

5.7. Conclusions

We have developed an explicit DG scheme on moving meshes using ALE framework and space-time expansion of the solutions within each cell. The near Lagrangian nature of the mesh motion dramatically reduces the numerical dissipation especially for contact waves. Even moving contact waves can be exactly computed with a numerical flux that is exact for stationary contact waves. The scheme is shown to yield superior results even in the presence of large boost velocity of the coordinate system indicating its Galilean invariance property. The standard Roe flux does not suffer from entropy violation when applied in the current nearly Lagrangian framework. However, in some problems with strong shocks, spurious contact waves can appear and we propose to fix the dissipation in Roe-type schemes that eliminates this issue. The method yields accurate solutions even in combination with standard TVD limiters, where fixed grid methods perform poorly. The mesh motion provides automatic grid adaptation near shocks but may lead to very coarse cells inside expansion waves. A grid adaptation strategy is developed to handle the problem of very small or very large cells. The presence of the DG polynomials makes it easy to transfer the solution during grid adaptation without loss of accuracy. The proposed methodology is general enough to be applicable to other systems of conservation laws modelling fluid flows. The basic idea can be extended to multi-dimensions but additional considerations are required to maintain good mesh quality under fluid deformations.

Chapter 6 ALE-DG Methods in Two-Dimensions

In this chapter, we extend the scheme introduced in Chapter 5 to two-dimensional conservation laws. We first describe a formulation of the conservation laws on a moving domains in Section 6.1. Then we introduce the concepts of simulation domain, the mesh and the discrete setting for the solution in Section 6.2 and Section 6.3 respectively. Next, we present a broad overview of the ALE DG algorithm in Section 6.4. Then we elaborate the details of the components in further sections by presenting the evolution of mesh in a single time-step in Section 6.5, the numerical scheme inside a single time-step in Section 6.6, and the mesh adaptation algorithms in Section 6.7. Finally, we present the results of using the ALE DG method to solve the two-dimensional Euler equations for isentropic vortex in Section 6.8. We describe our conclusions and future work in Section 6.9.

6.1. The Formulation on Moving Domains

The classical formulation of the conservation laws is based on the assumption that the domain of interest is fixed in time. The corresponding frame of reference is called the fixed or Eulerian frame of reference. By contrast, in the moving mesh methods, the domain of interest is allowed to move in time. The classical fixed-mesh formulation of the conservation laws is not directly applicable to such problems and one needs to come up with a formulation that is valid on moving domains.

In coming up with such a formulation, one either needs to account for the motion of the domain in the formulation of the conservation laws in the fixed frame or one can state the problem itself in the moving frame of reference. A moving frame of reference where the frame moves exactly with the velocity of the fluid is called the Lagrangian frame. The ALE frame of reference is a moving frame where the velocity of the frame is arbitrary with respect to the velocity of quantities of interest. In practice, for fluid dynamics problems, the ALE frame is often chosen to move with velocity close to fluid velocity (almost-Lagrangian).

In this section, we first state the conservation laws on fixed domains in the Eulerian frame, then we define the ALE frame and state the Reynolds transport theorem. Finally, we derive the conservation laws on moving domains in the ALE frame.

6.1.1. Conservation Laws on Fixed Domains

Consider an open bounded set $\Omega \subset \mathbb{R}^d$ with a Lipschitz boundary which is constant in time. Let $\boldsymbol{\nu}$ denote the outward unit normal along Ω. Let $\mathbf{u}(\mathbf{x}, t)$, $\mathbf{u}\colon \mathbb{R}^d \times \mathbb{R}^+ \to \mathbb{R}^n$, be a function describing the density of various conserved quantities, let $\mathcal{F}(\mathbf{x}, t, \mathbf{u})$, $\mathcal{F}: \mathbb{R}^d \times \mathbb{R}^+ \times \mathbb{R}^n \to \mathbb{R}^{d\times n}$, $\mathcal{F} = (\mathbf{F}_1, \mathbf{F}_2, \dots, \mathbf{F}_d)^\top$, govern the rate of change of $\mathbf{u}(\mathbf{x}, t)$ within Ω. One can then assume that the behavior of the quantity $\mathbf{u}$ in Ω can be described by the integral form of the conservation laws (6.1)

$$\frac{\mathrm{d}}{\mathrm{d}t}\int\limits_{\Omega}\mathbf{u}(\mathbf{x},t)\,\mathrm{d}\mathbf{x}+\int\limits_{\partial\Omega}\mathcal{F}(\mathbf{x},t,\mathbf{u})\cdot\boldsymbol{\nu}\,\mathrm{d}\mathbf{s}=0 \tag{6.1}$$

We can apply the Gauss divergence theorem to the second integral term to get

$$\frac{\mathrm{d}}{\mathrm{d}t}\int\limits_{\Omega}\mathbf{u}(\mathbf{x},t)\,\mathrm{d}\mathbf{x}+\int\limits_{\Omega}\nabla_{\mathbf{x}}\boldsymbol{\cdot}\mathcal{F}(\mathbf{x},t,\mathbf{u})\,\mathrm{d}\mathbf{x}=0 \tag{6.2}$$

Next, by noting that the domain is independent of time, we take the time derivative under the first integral and then substitute it with a partial derivative instead as in (6.3).

$$\int\limits_{\Omega}\frac{\partial}{\partial x}\mathbf{u}(\mathbf{x},t)\,\mathrm{d}\mathbf{x}+\int\limits_{\Omega}\nabla_{\mathbf{x}}\boldsymbol{\cdot}\mathcal{F}(\mathbf{x},t,\mathbf{u})\,\mathrm{d}\mathbf{x}=0 \tag{6.3}$$

Finally, since the choice of initial domain Ω was arbitrary, we can convert the integral form to a differential form and we obtain the conservation law in a PDE form eq. (6.4).

$$\frac{\partial}{\partial t}\mathbf{u}(\mathbf{x},t)+\nabla\boldsymbol{\cdot}\mathcal{F}(\mathbf{x},t,\mathbf{u})=0 \tag{6.4}$$

6.1.2. The ALE Frame and Reynolds Transport Theorem

We define the ALE frame by the coordinate system $(\boldsymbol{\chi},t)$ where the mapping between the ALE and the Eulerian coordinates is given by mapping $\boldsymbol{\Phi}$ where $\boldsymbol{\Phi}$ is a homeomorphism and differentiable in t almost everywhere over the interval $[T_0,T_f)$. The mapping describes the motion of the ALE frame with respect to the Eulerian frame.

$$\boldsymbol{\Phi}\colon \Omega_{\boldsymbol{\chi}}\times[T_0,T_f)\to\Omega_{\mathbf{x}}\times[T_0,T_f) \tag{6.5}$$

$$(\boldsymbol{\chi},t)\mapsto\boldsymbol{\Phi}(\mathbf{x},t)=(\mathbf{x},t) \tag{6.6}$$

The velocity of the ALE frame is denoted by $\mathbf{w}(\boldsymbol{\chi},t)$ and can be defined by (6.7).

$$\mathbf{w}(\boldsymbol{\chi},t)=\left.\frac{\partial\mathbf{x}}{\partial t}\right|_{\boldsymbol{\chi}} \tag{6.7}$$

The determinant of the time-independent mapping of the domain $\Omega_{\mathbf{x},t}$ to $\Omega_{\boldsymbol{\chi},t}$ is often useful in computing the change of variables in the integrals. We denote the mapping by $\mathcal{J}$ and it is given by (6.8).

$$\mathcal{J}(\boldsymbol{\chi},t)=\frac{\partial\mathbf{x}}{\partial\boldsymbol{\chi}} \tag{6.8}$$

The gradient for the mapping $\boldsymbol{\Phi}$ can be written as (6.9).

$$\frac{\partial\boldsymbol{\Phi}}{\partial(\boldsymbol{\chi},t)}=\begin{pmatrix}\mathcal{J}(\boldsymbol{\chi},t) & \mathbf{w}(\boldsymbol{\chi},t)\\ \mathbf{0}^{\mathsf{T}} & 1\end{pmatrix} \tag{6.9}$$

The mapping $\boldsymbol{\Phi}$ is bijective and orientation-preserving if and only if the determinant of the Jacobian of the mapping $\mathbf{\Phi}$ is strictly positive as in (6.10).

$$\det \mathcal{J}(t) > 0 \qquad \forall t \in [0, T) \text{ and } \mathbf{x} \in \Omega_{\mathbf{x}}(t) \tag{6.10}$$

As we will see later, in the context of numerical methods, the maximum time-step size is directly proportion to $\det \mathcal{J}(t)$. Therefore, to ensure that the there is minimal time-step size, it is important to ensure that the $\det \mathcal{J}(t)$ also has a lower bound as shown in (6.11).

$$\det \mathcal{J}(t) \geq \alpha > 0 \qquad \forall t \in [0, T) \text{ and } \mathbf{x} \in \Omega_{\mathbf{x}}(t) \tag{6.11}$$

Let $\mathbf{u} \in \mathbb{R}^n$ be a physical quantity described by $\mathbf{u}(\mathbf{x}, t)$ and $\mathbf{u}(\chi, t)$ in the Eulerian frame and ALE frame respectively. Assume that the mapping $\mathbf{\Phi}$ is bijective and orientation-preserving. Then, the derivatives of $\mathbf{u}(\mathbf{x}, t)$ in the Eulerian frame and the ALE frame are related by (6.12).

$$\frac{\partial \mathbf{u}(\chi, t)}{\partial \chi} = [\nabla_{\mathbf{x}} \mathbf{u}(\mathbf{x}, t)] \, \frac{\partial \mathbf{x}}{\partial \chi} \qquad \frac{\partial \mathbf{u}(\chi, t)}{\partial t} = [\nabla_{\mathbf{x}} \mathbf{u}(\mathbf{x}, t)] \mathbf{w} + \frac{\partial \mathbf{u}(\mathbf{x}, t)}{\partial t} \tag{6.12}$$

Let $\Omega(t)$ be a physical domain and let $\Omega_{\mathbf{x}}(t)$ be its description in Eulerian frame. Let $\mathbf{u} \in \mathbb{R}^n$ be a physical quantity described by $\mathbf{u}(\mathbf{x}, t)$ in the Eulerian frame. Then, using the equations of change of variables, we have the Reynolds transport theorem given by (6.13).

$$\frac{\mathrm{d}}{\mathrm{d}t} \int\limits_{\Omega_{\mathbf{x}}(t)} \mathbf{u}(\mathbf{x}, t) \, \mathrm{d}\mathbf{x} = \int\limits_{\Omega_{\mathbf{x}}(t)} \left[\frac{\partial}{\partial t} \mathbf{u}(\mathbf{x}, t) + \nabla_{\mathbf{x}} \cdot [\mathbf{u} \otimes \mathbf{w}] \right] \mathrm{d}\mathbf{x} \tag{6.13}$$

Here, the term $\mathbf{u} \otimes \mathbf{w}$ is the outer product of $\mathbf{u}$ and $\mathbf{w}$ and satisfies (6.14).

$$\nabla_{\mathbf{x}} \cdot [\mathbf{u} \otimes \mathbf{w}] = (\nabla \cdot \mathbf{u}) \, \mathbf{w} + \mathbf{u} \cdot \nabla \mathbf{w} \tag{6.14}$$

6.1.3. Conservation Laws on Moving Domains

We can now use the Reynolds transport theorem (6.13) to derive the conservation laws on moving domains. We note that the right hand side of the Reynolds transport theorem is in the Eulerian frame. Therefore, we can use the PDE form of the conservation laws (6.4) to rewrite the right hand side of the Reynolds transport theorem in terms of the space derivatives as shown in (6.15).

$$\frac{\mathrm{d}}{\mathrm{d}t} \int\limits_{\Omega_{\mathbf{x}}(t)} \mathbf{u}(\mathbf{x}, t) \, \mathrm{d}\mathbf{x} = \int\limits_{\Omega_{\mathbf{x}}(t)} \left[- \nabla_{\mathbf{x}} \cdot \mathcal{F}(\mathbf{x}, t, \mathbf{u}) + \nabla_{\mathbf{x}} \cdot [\mathbf{u} \otimes \mathbf{w}] \right] \mathrm{d}\mathbf{x} \tag{6.15}$$

Moving all the terms to the left hand side, we obtain the integral form conservation laws on moving domains as shown in (6.16).

$$\frac{\mathrm{d}}{\mathrm{d}t} \int\limits_{\Omega_{\mathbf{x}}(t)} \mathbf{u}(\mathbf{x}, t) \, \mathrm{d}\mathbf{x} + \int\limits_{\Omega_{\mathbf{x}}(t)} \nabla_{\mathbf{x}} \cdot \left[\mathcal{F}(\mathbf{x}, t, \mathbf{u}) - [\mathbf{u} \otimes \mathbf{w}] \right] \mathrm{d}\mathbf{x} = 0 \tag{6.16}$$

6.2. The Mesh Description

Unlike in the case of fixed mesh methods, the mapping between the physical domain, the simulation domain and the mesh can be quite complex in the case of moving mesh methods. In particular, the behavior of the three can diverge significantly at the boundaries. In order to keep the treatment simple, in this thesis, we assume that the boundaries of the physical domain, the simulation domain and the mesh coincide. With this context, we now describe the spatial simplicial mesh and the spatio-temporal simplicial mesh that we use in the ALE DG method.

6.2.1. Spatial Simplicial Mesh

In this section, we introduce the concept of a spatial simplicial mesh for a fixed time t. We first define the concept of a simplex and a simplicial complex, and then define a simplicial mesh.

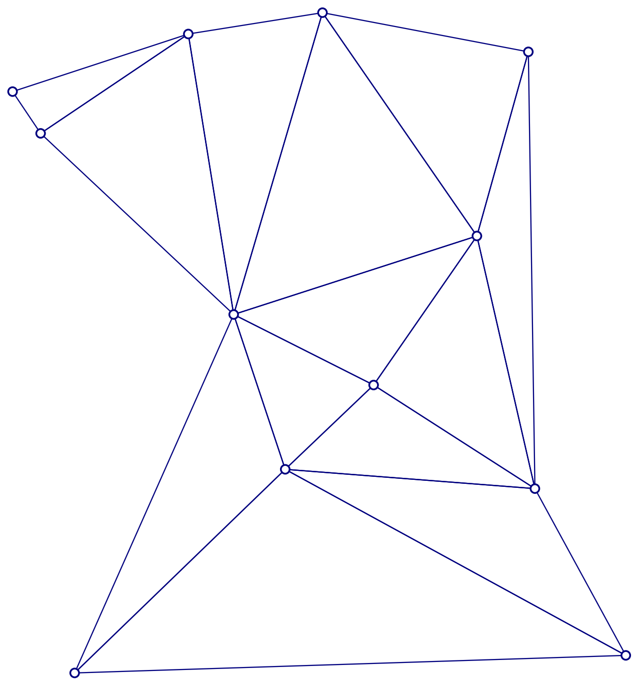


Figure 6.1.: A Simplex Mesh in Two Dimensions

Definition 6.1 *(Simplexes, Faces, Edges and Vertexes).* Let X be a collection of $d+1$ affinely independent points in $\mathbb{R}^d$. Then, a d-dimensional simplex ξ is the convex hull of a set X of $d+1$ affinely independent points. In particular, 0-simplex is a point, 1-simplex is an edge, 2-simplex is a triangle, and 3-simplex is a tetrahedron. Furthermore,

1. a simplex is an element of ξ if it is a convex hull of a non-empty subset of X.
2. a vertex of ξ is an element of ξ of dimension 0.
3. an edge of ξ is an element ξ of dimension 1.
4. a face of ξ is an element of ξ of dimension $d-1$

We denote the i-th cell in the mesh by K_i and the set of all cells in the mesh by $\mathscr{K}$. Similarly, the i-th face of the mesh is denoted by F_i and the set of all faces in the mesh is denoted by $\mathscr{F}$. Finally, the i-th vertex of the mesh is denoted by v_i and the set of all vertexes in the mesh is denoted by $\mathscr{V}$.

Definition 6.2 *(A Simplicial Complex).* A set Ξ of finitely many simplexes ξ is a simplicial complex if

- Ξ contains every element of every simplex in ξ.
- For any two simplex $\xi_1, \xi_2 \in \Xi$, $\xi_1 \cap \xi_2$ is either empty or an element of both ξ_1 and ξ_2.

Definition 6.3 *(Orientation of a Cell).* Let K_i be a cell in a simplicial complex which is the convex hull of the subset $\mathscr{V}_{K_i}$ of vertices $\mathscr{V}$. Associate a permutation $\sigma = (v_0, v_1, \dots, v_d), v_i \in V_{K_i}$ of the vertices and call the resultant **oriented cell** $K_{i,\sigma}$. Then, one can define an orientation of the simplex $K_{i,\sigma}$ by the sign of the determinant of the matrix $A_\sigma(K_{i,\sigma})$ defined as

$$A(K_{i,\sigma}) = \begin{bmatrix} v_1 - v_0 & v_2 - v_0 & \cdots & v_d - v_0 \end{bmatrix} \tag{6.17}$$

The orientation of the cell $K_{i,\sigma}$ is denoted by $\operatorname{sgn}(K_{i,\sigma})$ and can be defined as

$$\operatorname{sgn}(K_{i,\sigma}) = \operatorname{sgn}(\det(A_\sigma(K_{i,\sigma}))) \tag{6.18}$$

By definition, the orientation can either be positive or negative for a non-degenerate cell.

Definition 6.4 *(Simplicial Mesh).* Let $\mathscr{V}$ be a set of points in $\mathbb{R}^d$. A d-dimensional simplicial mesh of $\mathscr{V}$ is a simplicial complex Ξ such that

- $\mathscr{V}$ is the set of vertices in Ξ
- The union of all the simplexes in Ξ is the convex hull of $\mathscr{V}$
- Orientation of all oriented cells in $\mathscr{K} \in \Xi$ is positive.

Since all cells have positive orientation, we can drop the orientation subscript and denote all oriented cells $K_{i,\sigma}$ by their unoriented notation K_i.

A significant aspect of computing meaningful error estimates is existence of classic inverse, trace and interpolation inequalities on the family of meshes as we improve the resolution of the mesh. An error estimates that approaches zero as the mesh is refined is a good indicator of the convergence of the numerical scheme.

The existence of such inequalities is not available in general for an arbitrary simplicial mesh as specified in Definition 6.4. Instead, one must constrain the mesh further using additional conditions on its geometry. One such set of conditions that guarantees the existence of such inequalities is the regularity of a family of simplicial meshes [Cia02]. In order to define the condition of regularity, we first need to define the diameter and inradius of a mesh.

Definition 6.5 *(Diameter and Inradius of a Mesh).* Given a simplicial mesh Ξ, for any element $K \in \Xi$, the diameter of K, $h(K)$ is the circumradius of the element K, while $\rho(K)$ denotes the inradius of K. The diameter of the mesh is defined as the maximum of the diameters over its elements while the inradius is defined as the minimum of all the inradii of the elements of the mesh.

$$h(\Xi) = \max_{K \in \Xi} h(K) \qquad \rho(\Xi) = \min_{K \in \Xi} \rho(K) \tag{6.19}$$

Definition 6.6 *(Regular Simplicial Mesh).* A family of simplicial meshes $\mathbf{\Xi}_h$ is called regular if there are constants $\sigma > 0$ and $\tau > 0$ such that for each $\Xi \in \mathbf{\Xi}_h$

$$\frac{h(\Xi)}{\rho(\Xi)} \leq \sigma \qquad\qquad \frac{\rho(K)}{h} \leq \tau \tag{6.20}$$

6.2.2. A Spatio-Temporal Simplicial Mesh

Having defined the spatial simplicial mesh, we now define a spatio-temporal simplicial mesh that evolves with the simulation. As noted before, the goal is to have a mesh that moves with velocities very close to the fluid velocities and the mesh velocity is constant inside a single time step. Furthermore, the mesh might undergo topological changes between the time steps but doesn't change inside a single time step. Finally, we want to describe the behavior of the mesh at the boundaries of the simulation domain.

Without any loss of generalization, we can assume that the simulation is performed in the time interval $[T_0, T_f]$ with N time steps $t_0, t_1, \dots, t_N$. We denote the time interval of the n-th time step as $I_n = [t_n, t_{n+1}]$, $i = 0, 1, \dots, N-1$ with $t_N = T$. There is a possible topological change in the mesh at a time step t_n, which also requires adaptation of the finite element space and the discontinuous Galerkin approximation. For a given time interval I_n, we denote the spatial domain at a time $t \in I_n$ as $\Omega_n(t)$, and the spatial simplicial mesh at a time $t \in I_n$ as $\Xi_n(t)$.

ATTENTION!. Note that the spatial domain Ω_n and the spatial simplicial mesh Ξ_n are defined on the time interval I_n and not at the time-steps t_n as is typical in the literature. In the rest of the thesis, the subscript n will denote the interval of the time-step I_n and not the time step t_n themselves.

Given the set of vertices $\mathscr{V}_n$ of the spatial simplicial mesh Ξ_n, the velocity of the mesh is characterized by the velocity of the vertices. We denote the velocity of the mesh at the vertex $v_{n,i} \in \mathscr{V}_n$ as $\mathbf{w}_{n,i}$. We assume that the velocity of the mesh in the interior of a cell K_k is a barycentric interpolation of the velocities of the vertices of the cell. Therefore, the velocity of an arbitrary point $\mathbf{x} \in \Omega_n(t)$ is given by (6.21) where $\mathbf{x}$ is contained in cell K_k ($\mathbf{x} \in K_k$) and $(\alpha_i)_{i=0}^{d}$ are the barycentric coordinates of $\mathbf{x}$ with respect to the cell K_k.

$$\mathbf{w}_n(\mathbf{x}) = \sum_{j=0}^{d} \alpha_j \mathbf{w}_{n,k_j} \tag{6.21}$$

In Section 6.2.1, we assumed that the orientation of the cells in the simplicial mesh is always positive. In the following theorem, we outline the conditions under which the orientation of the cells is preserved during the evolution of the mesh inside an interval I_n.

Theorem 6.1 *(Preservation of Cell Orientation).* Consider a mesh Ξ_n in the time interval I_n, such that the orientation of all elements of the mesh is positive at time t_n, that is $\operatorname{sgn}(K(t_n)) > 0$ for all $K \in \Xi_n$. Then mesh preserves its for all $t \in I_n$, that is $\operatorname{sgn}(K(t)) > 0$ for all $K \in \Xi_n$ and all $t \in I_n$, if for all $t \in I_n$, we have

$$\|A_K^{-1}(t_n)A_K(t_{n+1})\|_{\mathcal{L}(\mathbb{R}^d,\mathbb{R}^d)} < \epsilon(t) \qquad \text{or} \, \|A_K^{-1}(t_{n+1})A_K(t_n)\|_{\mathcal{L}(\mathbb{R}^d,\mathbb{R}^d)} < \epsilon(t)^{-1} \tag{6.22}$$

where

$$\epsilon(t) = \frac{t_{n+1} - t}{t - t_n} \tag{6.23}$$

6.3. The Discrete Setting for the ALE DG Method

As described in Section 6.2.2, the mesh is defined in a spatio-temporal sense in a given time interval. Therefore, the finite cell space defined over the mesh also needs to be defined in a spatio-temporal sense. For the ALE DG method, we use a spatio-temporal broken polynomial space to define a finite cell space over the mesh. We first define a reference space-time cell, and a polynomial space over it. We then define the spatio-temporal finite cell space over the mesh as a broken polynomial space corresponding to the polynomial space over the reference space-time cell.

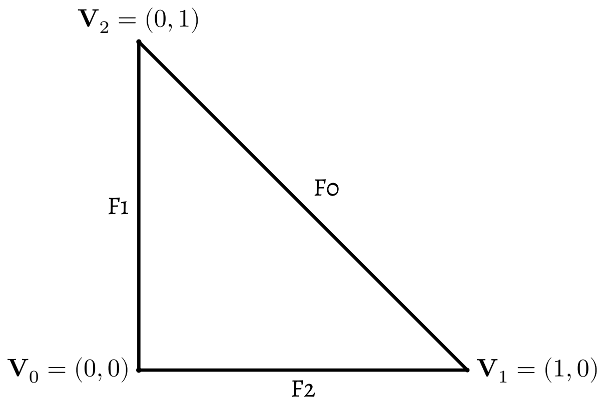


Figure 6.2.: The reference space-time cell in two-dimensions.

The reference cell is a space time prism with a triangular base given by (6.24) where $\mathfrak{K}$ is the triangle with vertices $(0,0)$, $(1,0)$, and $(0,1)$ in the x-y plane. We note that for any given space-time cell $K(t) \in \Xi(t)$, the mapping $\mathcal{T}(\widehat{K}, K)$ between the cell and the reference cell is an affine mapping.

$$\widehat{K}(\mathbf{x},t) = \mathfrak{K}\Big\{(0,0),(1,0),(0,1)\Big\} \times [0,1] \tag{6.24}$$

Given a reference space time cell $\widehat{K}$, let $\mathcal{P}^k(K)$ denote the space of polynomials in $\mathcal{K}$ of degree at most k. We then define the finite element space over the mesh as a broken polynomial space corresponding to the polynomial space over the reference space-time cell as shown in (6.25).

$$\mathcal{V}_\Omega(t) = \Big\{v \in L^2(\Omega(t)) : v|_{K(t)} \in \mathcal{P}^k(K(t)), \forall K(t) \in \Xi(t)\Big\} \tag{6.25}$$

6.4. A Sketch of the ALE-DG Algorithm

Having defined the mesh and setting of the problem in the previous sections, we now describe the outline of the ALE DG algorithm for simulation. Individual steps of the algorithms are described in more detail in the following sections.

We would like to evolve the solution of the conservation laws (6.4) from time $t = T_0$ to $t = T_f$. We assume that we have the initial simulation domain $\Omega(0)$, a conforming mesh for the domain $\Xi(0)$ and the initial condition for the fluid velocity $\mathbf{u}(\mathbf{x},0)$ defined on the simulation domain. We further assume that the strategy for imposing the boundary conditions for both the simulation domain and the physics is known for any future time $t < T_f$.

The simulation follows a time-marching algorithm over multiple time steps till the simulation time reaches the final time. In each time step, there are three different steps:

1. We first compute the mesh velocity and the time step.
2. Once the mesh velocities have been computed, we can evolve the solution of the equation according to the ALE DG scheme.
3. Concurrently, the mesh can also be evolved over the given time step, checked for quality and adaptations are carried out if needed.
4. Finally, the solution is adapted to the new mesh.

We would like to note that the steps 2 and 3 of the above algorithm can be carried out concurrently. A short form of the ALE DG algorithm for the simulation of the conservation laws (6.4) is given by Algorithm 1.

Algorithm 1 The `dgale` Algorithm

```
1:  procedure DGALE($\Omega(0), \Xi(0), \mathbf{u}(0), L$)
2:      $n \leftarrow 0, t_n \leftarrow T_f$
3:      $\Xi_n(T_0) \leftarrow \Xi(T_0)$
4:      while $t_n < T_f$ do
5:          Compute the time step $\Delta t_n$ and mesh velocity $\mathbf{w}_n$ for $\Xi_n$
6:          Evolve the solution $\mathbf{u}_n(t)$ on the time interval $I_n$
7:          Evolve and adapt the mesh from $\Xi_n(t_n)$ to $\Xi_{n+1}(t_n)$
8:          Adapt the solution to the new mesh.
9:          $n \leftarrow n + 1$
10:         $t_n \leftarrow t_{n-1} + \Delta t_n$
11:     end while
12:     return $\Xi_N$ and $\mathbf{u}(T_f)$
13: end procedure
```

6.5. The Time Step and Mesh Velocity

In this section, we describe the algorithm to compute the mesh velocity and the time step for a given time interval I_n. The mesh velocity also then characterizes the evolution of the mesh in the time interval.

In a moving mesh methods, there are two different constraints that can affect the computation of time step at any given time in the simulation. The eventual time step used for the time interval is a minimum of the two time-steps.

1. Geometric Time Step

   The geometric constraint comes from the regularity condition which imposes conditions on the time step that would ensure the regularity of the mesh. We call the maximum time step that allows for the regularity to be constrained as the **geometric time step**.

2. Physical Time Step

   The physical constraint comes from the CFL-like conditions to ensure the fidelity of the simulations. As we have seen in Chapter 5, making the CFL-like conditions aware of the moving mesh allows one to make the time-step computations larger than for the fixed mesh simulations. The time step computed from such considerations is called the **physical time step**.

As we can see here, the computation of the time steps depends on the mesh velocity. As we will show further, the computation of the mesh velocity, in turn, depends on the time step themselves. To break this cyclic dependency, we generate a conservative approximation for the fluid velocity at the vertex which we use as the initial guess for our mesh velocity. We use that initial guess to compute the time step, and then we finally use the resultant time step to compute the eventual mesh velocity. A conservative approximation is the one that ensures that the computed time step is guaranteed to be less than a time step that would have been computed taking the eventual mesh velocities in to account.

6.5.1. Approximation to Fluid Velocity at the Vertexes

The DG solution is discontinuous at at the interfaces of the cells, and therefore, at the vertices. Hence, there is no unique fluid velocity available from our scheme at the vertices. Instead, we can only compute an approximation for the fluid velocity. The mesh velocity must be close to the local fluid velocity in order to have a Lagrangian character to the scheme.

Some researchers, especially in the context of Lagrangian methods, solve a Riemann problem at the cell face to determine the face velocity. Since we use an ALE formulation, we do not require the exact fluid velocity. Furthermore, following the exact trajectory of the fluid would also lead to curved trajectories for the grid point, which would contradict one of our core assumptions of the velocity being constant in a single time interval. In our work, we use the maximum of velocities over the neighboring cells.

$$\mathbf{w}_v = \frac{1}{N_v} \sum_{i=1}^{N_v} \mathbf{w}_{N_v} \tag{6.26}$$

6.5.1.1. Computation of Velocities at the Boundary

The above mentioned methods to compute mesh velocity algorithms are appropriate for the interior vertices. On the boundary vertices, however, we need to ensure that the mesh velocities accurately reflect the fluid motion without negatively affecting the mesh motion. In this work, we only deal with periodic boundary conditions where

For periodic boundaries, the vertex on one boundary is a clone of the vertex on the periodic boundary. This requires that the velocity computed for the paired vertices should be equal. In order to achieve that, we generate the list of cells which are neighbors of either of the vertices and then use the average mesh velocity formula. Let NK_{n_k} be the neighbors of the vertex K_k and let NK_{n_l} be the neighbors of the vertex K_{k_p} where K_{k_p} is the paired vertex. Let $NK = NK_{n_k} \cup NK_{n_l}$, then we have

$$\mathbf{w} = \frac{1}{|NK|} \sum_{K \in NK} \mathbf{w}_K \tag{6.27}$$

6.5.2. Computing the Time Step

As already mentioned, computing the time step for the for the moving mesh methods has two constraints, namely the geometric and the physical time step. The eventual time step is the minimum of the time steps computed with the two constraints. In this section, we describe the algorithms for computing the two kinds of time step.

6.5.2.1. The Geometric Time Step

The geometric time step is an upper bound on time step which ensures that the regularity of the mesh is maintained. Additionally, for purpose of numerical accuracy, it is generally also desirable to ensure that the size of the mesh cell doesn't change by more than $\frac{1}{2}$ times its original size. Finally, we want to allow for large translations of the cells.

All of these factors can be achieved by making sure that the maximum difference between the distance traveled by the centroid and the distance traveled by any of the edges is less than $1/4$-th of the smallest side of the triangle. The velocity of the centroid if given by

$$\mathbf{w}_c = \frac{1}{3}\sum_{i=0}^{2} \mathbf{w}_i \tag{6.28}$$

Let s_0 be the smallest side of the triangle, then we have the following condition on the geometric time step

$$\Delta t \leq \min_{i=0,1,2} \frac{s_0}{4\left\|\mathbf{w}_i - \mathbf{w}_c\right\|} \tag{6.29}$$

6.5.2.2. The Physical Time Step

The physical time step is an upper bound on the time step that comes from the CFL-like conditions. For a cell K, the CFL-like condition can be written down as

$$\Delta t \leq \frac{r}{\max\limits_{\mathbf{x}\in K}\left\|\mathbf{v}(\mathbf{x}) - \mathbf{w}(\mathbf{x})\right\| + \max\limits_{\mathbf{x}\in K} c(\mathbf{x})} \tag{6.30}$$

where r is the inradius of the triangle, $\mathbf{v}(\mathbf{x})$ is the fluid velocity, and c is the sound speed.

6.5.3. The Mesh Velocity

Having computed the time step, we can now again compute a mesh velocity that is suitable for ALE simulations. We would want such a velocity to be as close to fluid velocity as possible. However, at the same time, the variability in the velocity should be low enough to prevent mesh distortion as much as possible.

One of the basic techniques to deal with mesh distortion is smoothing of mesh velocities. The technique prevents mesh quality from degrading severely in a single time step and hence, allowing mesh adaptation techniques to repair the mesh before the mesh quality degrades unacceptably. In our work, we use a special type of mesh velocity smoothing known as Laplacian smoothing

The objective of Laplacian smoothing is to minimize the distance between the position of a vertex and the position of the centroid of the polygon formed by neighboring vertices.

Let $\{\mathbf{w}_i^n\}_{i=1}^N$ be the velocity of the vertices of the mesh at time t_n and $\mathbf{x}_i^{n+1}$ be the resultant position of the vertex i obtained from. Let $\mathbf{X}_i^{n+1}$ be the average position of the vertexes connected to the i-th vertex. We would like the $\mathbf{x}_i^{n+1}$ to be close to $\mathbf{X}_i^{n+1}$. Hence, we can choose the mesh velocity to be

$$\mathbf{w}_i^n = \alpha\mathbf{w}_i^n + (1-\alpha)\frac{\mathbf{X}_i^{n+1} - \mathbf{x}_i^{n+1}}{\Delta t} \qquad \alpha \in [0,1] \tag{6.31}$$

The α factor moves $\mathbf{x}_i^{n+1}$ closer to $\mathbf{X}_i^{n+1}$ while the number of steps decrease the distance. We can apply a few iterations of the above smoothing in order to get a better velocity.

6.6. The ALE DG Scheme for a Single Time Step

In this section, we introduce the numerical scheme for a single time step I_n. The numerical scheme iterates over every cell in the mesh and computes the solution at the end of the time step. We first introduce the notation and terminology to be used in this section.

Notation and Terminology In order to keep the notation manageable, we also introduce some simplifications along the way. In this section, we restrict ourselves to a single time interval $I_n = [t_n, t_{n+1}]$ and a single cell $K_k \in \Xi_n$. We then drop the relevant markers from the notation and assume that the variables are local to the time interval I_n and the cell K_k. In particular, we have the following simplifications in the notation.

- The time interval I_n is denoted by I and the cell K_k is denoted by K.
- The solution in time interval I_n is denoted by $\mathbf{u}(t)$ instead of $\mathbf{u}_n(t)$.
- The moments of the solution $\mathbf{u}$ in the cell K are denoted as $\mathbf{u}_m(t)$ instead of $\mathbf{u}_{k,m}(t)$.
- The faces and vertices of the cell K are denoted by F_i and V_i instead of $F_{k,i}$ and $V_{k,i}$.
- The mesh velocity is constant in the time interval I and is denoted by as $\mathbf{w}(\mathbf{x})$. The velocity at the vertex V_i is denoted by $\mathbf{w}_i$.
- The basis function for the cell K is denoted by $\phi_m(\mathbf{x}, t)$ instead of $\phi_{k,m}(\mathbf{x}, t)$.
- The Jacobian of the transformation from the reference cell $\hat{K}$ to the cell K is denoted by $\mathcal{J}_K(t)$ instead of $\mathcal{J}_{K_k}(t)$.

6.6.1. The Semi-Discrete Scheme

We now present the semi-discrete scheme for the ALE DG method. The central idea of the ALE DG method is to compute the evolution of the moments of the solution in each cell. The moments of the solution in the cell $K_k(t)$ are defined as

$$\left|\mathcal{J}_{K(t)}\right| \mathbf{u}_m(t) = \int_{K(t)} \mathbf{u}(\mathbf{x}, t)\phi_m(\mathbf{x}, t)\mathrm{d}\mathbf{x} = \int_{\hat{K}} \mathbf{u}(\chi, t)\phi_m(\chi) \left|\mathcal{J}_K(t)\right| \mathrm{d}\chi \tag{6.32}$$

We can compute the evolution of the moment by computing the derivative of moment with respect to time and using the Reynolds transport theorem as well as the PDE equation to get

$$\left|\mathcal{J}_{K(t)}\right| \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{u}_m(t) = -\int_{K(t)} \nabla_{\mathbf{x}} \cdot \left[\mathcal{F}(\mathbf{x}, t, \mathbf{u}) - \mathbf{u} \otimes \mathbf{w}\right] \phi_m(\mathbf{x}, t)\mathrm{d}\mathbf{x} \tag{6.33}$$

Applying integration by parts, and using the numerical flux to represent the solution at the boundaries, we get

$$\frac{\mathrm{d}}{\mathrm{d}t}\left|\mathcal{J}_{K(t)}\right|\mathbf{u}_m(t) = \int\limits_{K(t)} \left[\mathcal{F}(\mathbf{x},t,\mathbf{u}) - \mathbf{u}\otimes\mathbf{w}\right]\boldsymbol{\cdot}\nabla_{\mathbf{x}}\phi_m(\mathbf{x},t)\mathrm{d}\mathbf{x} - \int\limits_{\partial K(t)} \mathcal{H}(\mathbf{x},t,\mathbf{u}^-,\mathbf{u}^+,\mathbf{w};\hat{\boldsymbol{\nu}})\boldsymbol{\cdot}\hat{\boldsymbol{\nu}}\phi_m(\mathbf{s},t)\mathrm{d}\mathbf{s} \tag{6.34}$$

We now integrate this over time to get

$$\left|\mathcal{J}_{K(t_{n+1})}\right|\mathbf{u}_m(t_{n+1}) - \left|\mathcal{J}_{K(t_n)}\right|\mathbf{u}_m(t_n) = \int\limits_{t_n}^{t_{n+1}}\int\limits_{K(t)} \left[\mathcal{F}(\mathbf{x},t,\mathbf{u}) - \mathbf{u}\otimes\mathbf{w}\right]\boldsymbol{\cdot}\nabla_{\mathbf{x}}\phi_m(\mathbf{x},t)\mathrm{d}\mathbf{x} - \int\limits_{t_n}^{t_{n+1}}\int\limits_{\partial K(t)} \mathcal{H}(\mathbf{x},t,\mathbf{u}^-,\mathbf{u}^+,\mathbf{w};\hat{\boldsymbol{\nu}})\boldsymbol{\cdot}\hat{\boldsymbol{\nu}}\phi_m(\mathbf{s},t)\mathrm{d}\mathbf{s}$$

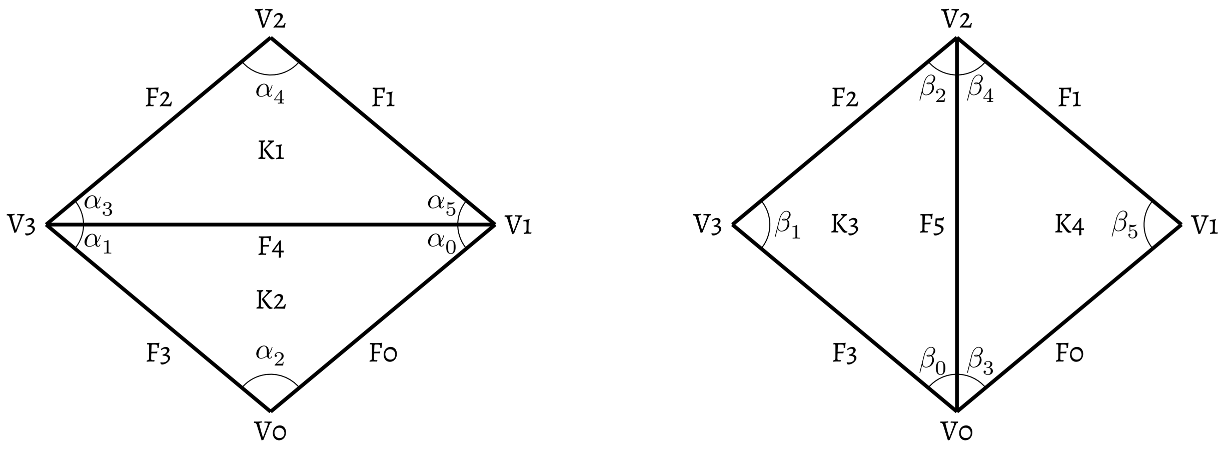
This is now the semi-discrete form of the scheme. In order to obtain the fully discrete form, we replace the integrals with quadrature to get

$$\begin{aligned}\mathbf{u}_m(t_{n+1}) = {}& \frac{|\mathcal{J}_{K^n}|}{|\mathcal{J}_{K^{n+1}}|}\mathbf{u}_m(t_n) \\ &+ \sum_{g=1}^{n_g}\sum_{c=1}^{n_c}\frac{|\mathcal{J}_{\Delta t}|}{|\mathcal{J}_{K^{n+1}}|}\left[w_g w_c \mathcal{G}(\mathbf{x}_c^g, t^g, \mathbf{u}_c^g, \mathbf{w}_c)\cdot\left[\nabla_{\boldsymbol{\xi}_c}\phi_m(\boldsymbol{\xi}_c)\cdot\widehat{\mathcal{J}}_{K^g}^{-1}\right]\right] \\ &- \sum_{g=1}^{n_g}\sum_{F^g}\sum_{f=1}^{n_f}\frac{|\mathcal{J}_{\Delta t}|\,|\mathcal{J}_{F^g}|}{|\mathcal{J}_{K^{n+1}}|}\left[w_g\, w_f\, \phi_m(\boldsymbol{\sigma})\mathcal{H}(\mathbf{x}^g, t^g, \mathbf{u}_f^{-,g}, \mathbf{u}_f^{+,g}; \mathbf{w}_f, \hat{\boldsymbol{\nu}})\right]\end{aligned}$$

We can either solve this equation implicitly or use the predictor approach in a manner similar to the one-dimensional case.

6.7. Local Mesh Adaption Techniques

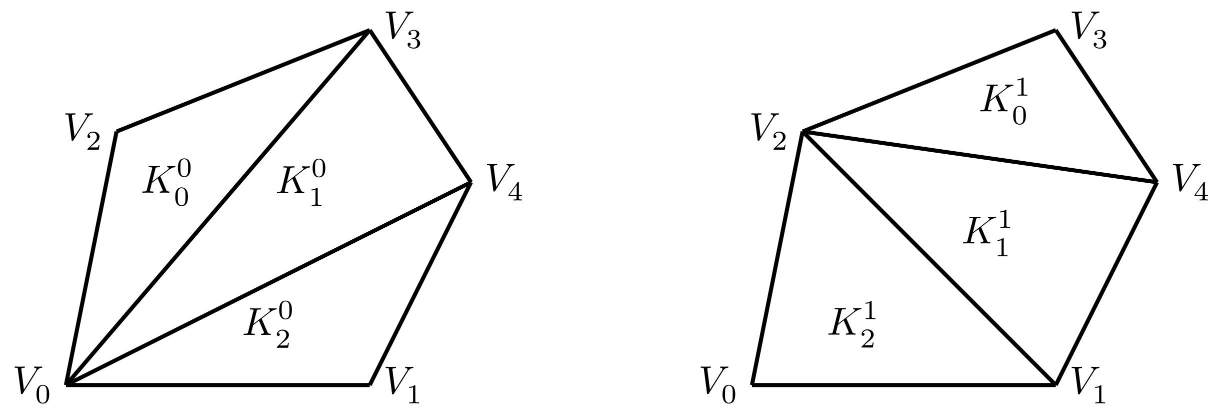
Smoothing of velocity can maintain the mesh quality, however, it can destroy the almost-Lagrangian nature of the mesh motion. In order to maintain almost-Lagrangian, it is necessary to locally change the mesh topology. One of the methods to do so is the edge swapping methods.



(a) Before (b) After

Figure 6.3.: Face Swap

In an edge swap method, we change the connectivity of an edge in order to improve the quality of the mesh. As can be seen in fig. 6.3, the mesh quality can improve in certain cases due to edge swaps. In order to figure out the situations in which the edge swaps are favorable, we use various mesh quality indicators. However, direct usage the mesh quality indicator for swapping can often be suboptimal. This is because, often, a naive face swap algorithm is not commutative.



(a) Before (b) After

Figure 6.4.: Local Remesh

Consider that we have a section Ω_K of the mesh which is locally remeshed as shown in fig. 6.4. Let K_i^0 be the list of triangles before remeshing and let K_i^1 be the list of triangles obtained after remeshing. An triangle $R_i \subset \Omega_K$ is an interpolation region if $R_i \subseteq K_j^0$ and $R_i \subseteq K_k^1$, for some j, k. Given a remeshing as shown above, one can decompose the domain into disjoint interpolation regions $\{R_i\}_{i=1}^{M}$ such that $R_i \cap R_j = \emptyset$ for $i \neq j$ and $\cup R_i = \Omega_K$. In general, one can always obtain such a region using a constrained Delaunay triangulation. However, for some special cases, like face swapping and face deletion, it is trivial to obtain such a region as will be shown later. For the case shown in fig. 6.4, we show one such decomposition in fig. 6.5.

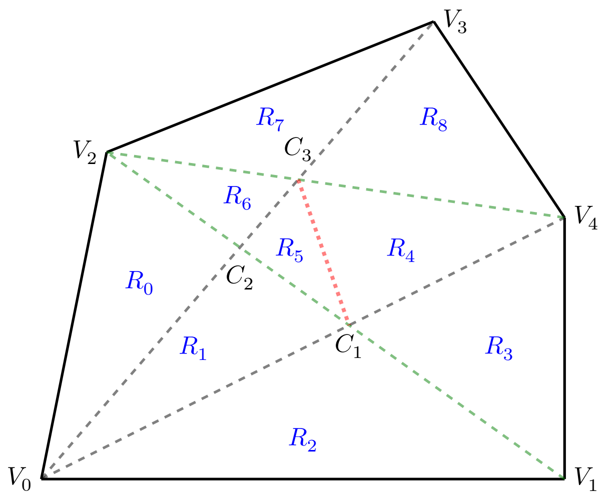


Figure 6.5.: Interpolation Regions

Having obtained such a decomposition, the solution transfer can be done as follows. Let $\mathbf{u}$ be the solution over the domain Ω_K. $\mathbf{u}_i^0$ be the solution in triangle K_i^0 for all i. We want to obtain the solution in triangle $\mathbf{u}_i^1$. We have

$$\mathbf{u}_{i,m}^1 = \int\limits_{K_i^1} \mathbf{u}\phi_{i,m}^1 \mathbf{dx} \tag{6.35}$$

Now, we know that $K_i^1 = \cup R_j$ and in each of R_j, $\mathbf{u}$ is a polynomial, and hence we can write

$$\mathbf{u}_{i,m}^1 = \sum_{R_j} \int\limits_{R_j} \mathbf{u}\phi_{i,m}^1 \mathbf{dx} \tag{6.36}$$

The integrals can now be evaluated exactly using quadrature rules. The remeshing algorithm now needs to provide only a description of how the regions are contained the triangles.

6.8. Experiments on Isentropic Vortex

The test case we consider involves an isentropic vortex that is advecting with constant velocity and is a smooth solution for which error norms can be calculated. The test is carried out on a square domain $[-10, 10] \times [-10, 10]$ with periodic boundary conditions. The initial conditions is an isentropic vortex

$$T = 1 - \frac{(\gamma-1)\beta^2}{8\gamma\pi^2} e^{1-r^2} \tag{6.37}$$

$$\rho = T^{\frac{1}{\gamma-1}} \tag{6.38}$$

$$u = u_\infty - \frac{\beta}{2\pi} y e^{\frac{1-r^2}{2}} \tag{6.39}$$

$$v = v_\infty - \frac{\beta}{2\pi} y e^{\frac{1-r^2}{2}} \tag{6.40}$$

$$p = \rho^\gamma \tag{6.41}$$

with $u_\infty = 1, v_\infty = 0, \gamma = 1.4, \beta = 10$. As the solution evolves in time, the mesh becomes quite deformed because the vortex is continually shearing the mesh, which can lead to degenerate meshes, as shown in figure 6.6. With the ALE DG method, we are able to maintain a good mesh quality even after the vortex has rotated 20 times around its center as shown in figures 6.7. As the vortex is translating, we plot the solution in a window centered at the vortex center. We can see that the method maintains its high order of accuracy from the convergence rates of the error shown in table 6.1; using linear basis functions yields second order convergence while quadratic basis functions lead to third order convergence. We can also see that the error in the moving mesh method evolves considerably slower than the error in the fixed mesh mesh.

N	Fixed Mesh		Moving Mesh	
	Error	Rate	Error	Rate
50x50	2.230e-03		1.225e-03	
100x100	5.987E-04	1.945	3.122e-04	1.972
200x200	1.498E-04	1.998	8.628e-05	1.855
400x400	3.786E-05	1.984	1.894e-05	2.187
800x800	9.617E-06	1.977	5.451e-06	1.796

Table 6.1.: Isentropic Vortex in 2D: Order of accuracy study on two dimensional mesh

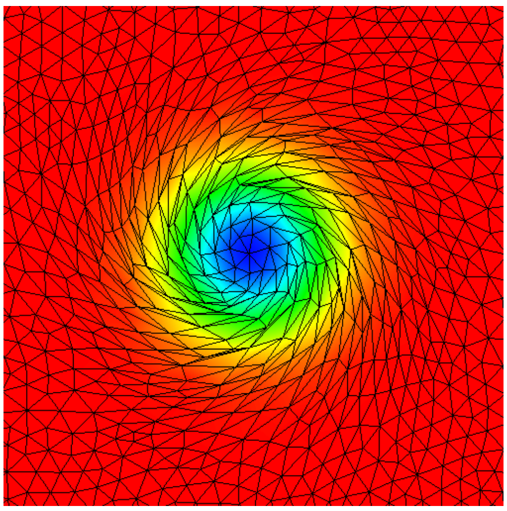


Figure 6.6.: Isentropic Vortex in 2-D: Skewed Mesh without Remeshing $t = 2.660534$

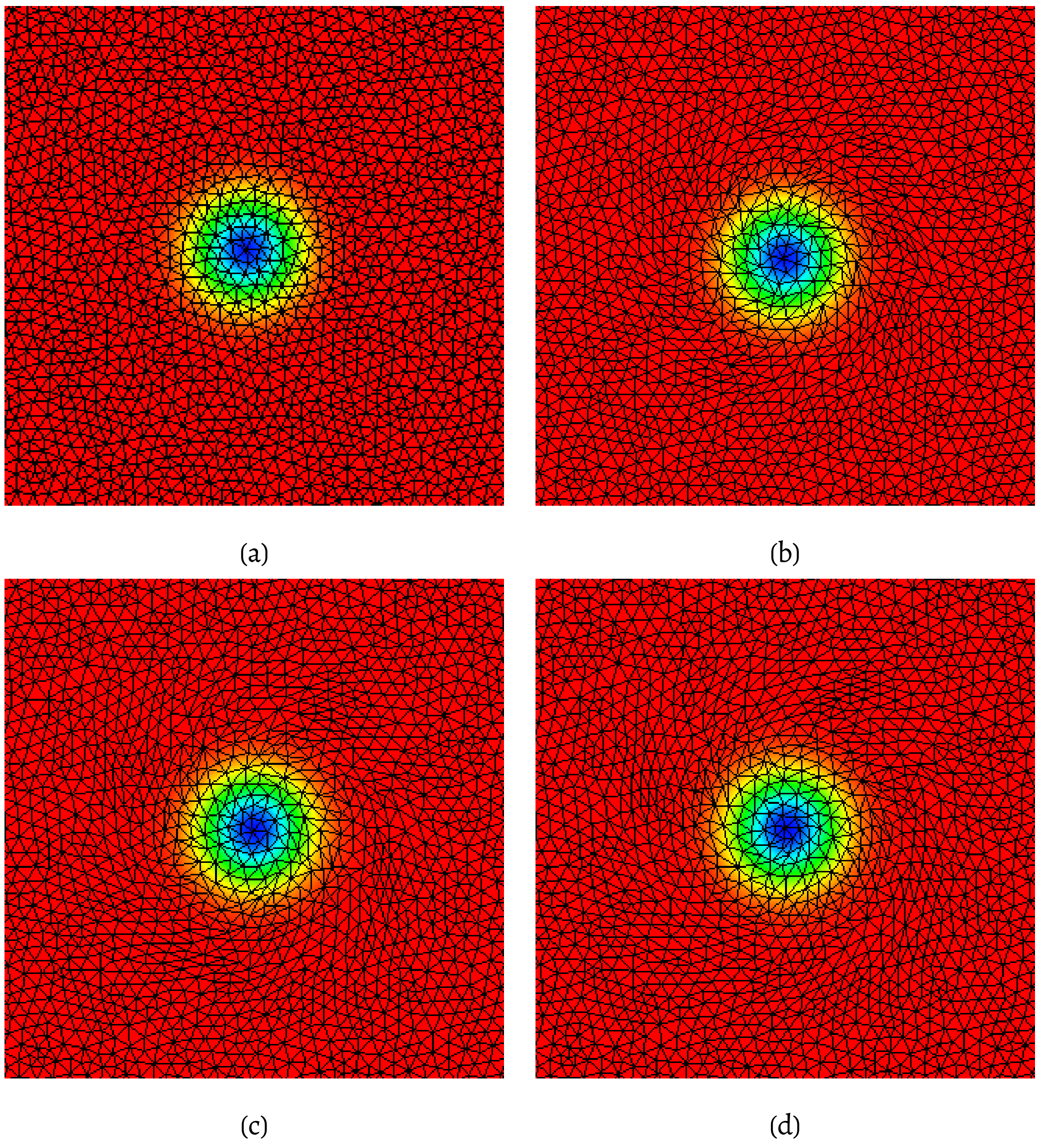


Figure 6.7.: Mesh and at various times (a) $t = 0$ (b) $t = 25$ (c) $t = 50$ (d) $t = 100$

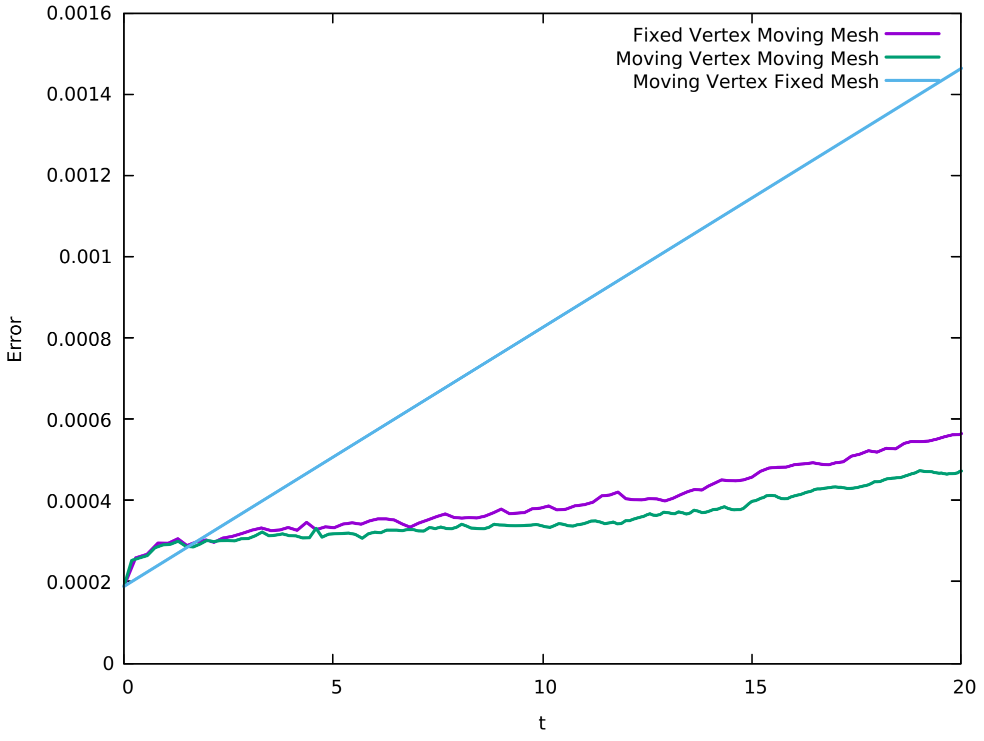


Figure 6.8.: Evolution of L^2 error for Isentropic Vortex in 2D

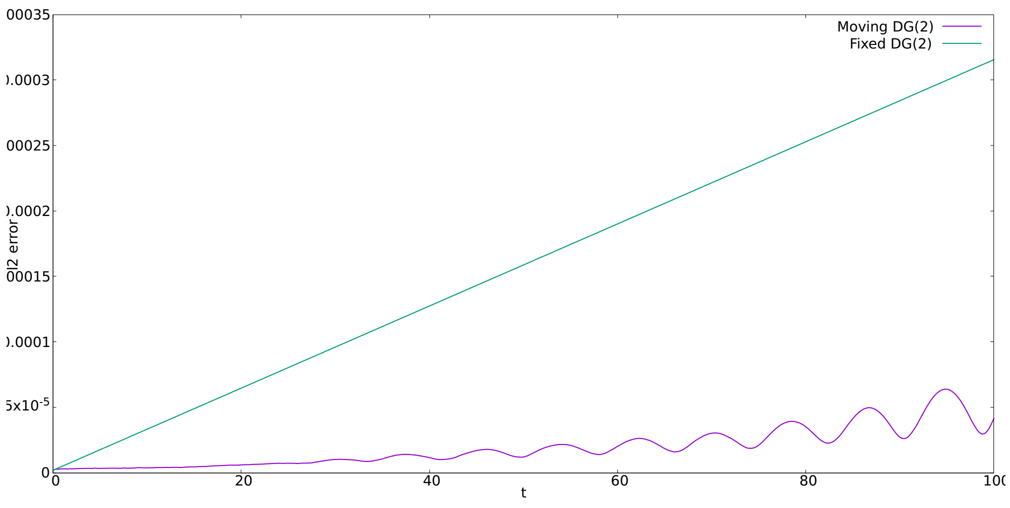


Figure 6.9.: Evolution of L^2 error for Isentropic Vortex in 2D

6.8.1. Isentropic Vortex with Non-Uniform Mesh

Due to the self-adaption nature of the moving mesh method, it is possible to use a non-uniform mesh to simulate a translating isentropic vortex. We start with a mesh which is fine near the vortex and coarse elsewhere. The vortex itself moves over the course of the simulation and we show that the mesh tracks the vortex. This allows us to concentrate the mesh in places with more features and thus reducing the computational cost of the code.

In comparison with uniform meshes, the non-uniform mesh only requires 1000 cells in comparison to uniform mesh's 10,000 cells to achieve a similar accuracy of error of $1.6e-4$. This represents a significant reduction in computational costs.

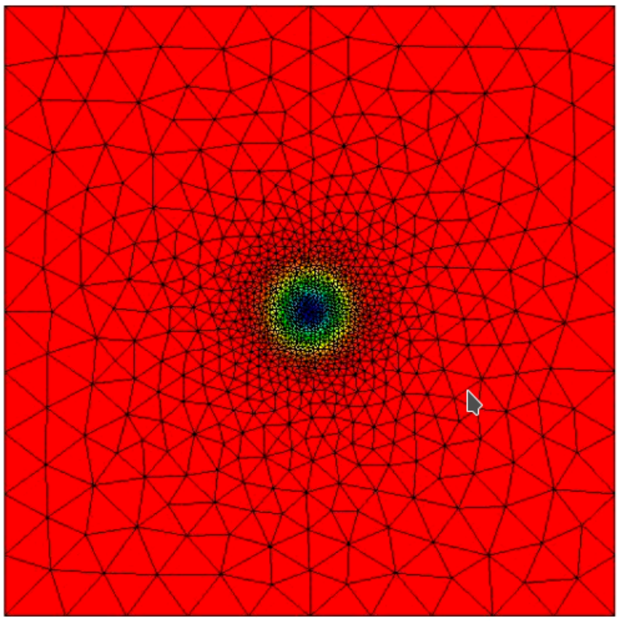


Figure 6.10.: Isentropic Vortex with Non-Uniform Mesh, $T = 0$

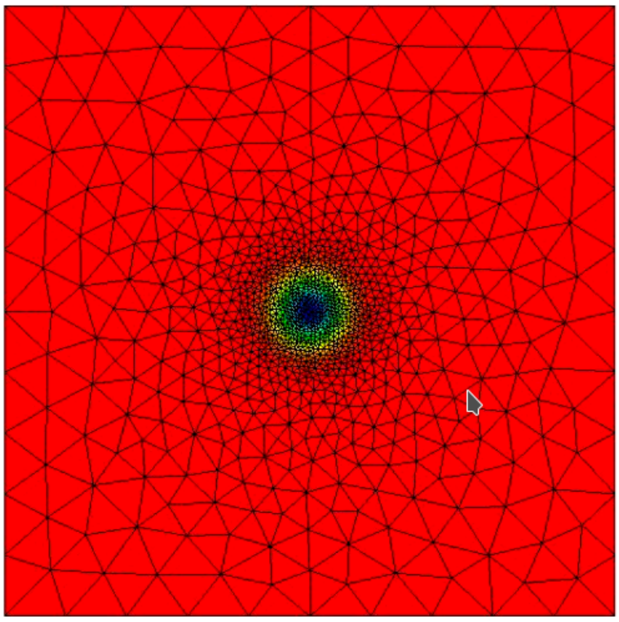


Figure 6.11.: Isentropic Vortex with Non-Uniform Mesh, $T = 100$

6.8.2. Comparison of Computational Costs between Methods

The moving of the mesh and grid adaptation introduces some computational overhead. In case of smooth solutions, due to the absence of sharp discontinuities, the quality of solution itself does not significantly change. Therefore, it is useful to look at the improvement in error against the increased computational cost for the moving mesh method. For that purpose, we have analysed the time spent by the scheme in different sections of the code.

We observe that the velocity smoothing algorithm is the most significant cost for the moving mesh algorithm and reduces the effectiveness of moving mesh methods for smooth solutions. However, the problem can be resolved by using non-uniform meshes as shown in Section 6.8.1.

6.9. Conclusions and Future Work

In this part of the thesis, we have developed an explicit single step Discontinuous Galerkin scheme on moving mesh methods using ALE Framework and space-time expansion of the solutions within each cell. The near Lagrangian nature of the mesh motion dramatically reduces the numerical dissipation especially for contact waves. Even moving contact waves can be exactly computed with a numerical flux that is exact for stationary contact waves in one dimensional simulations.. The scheme is shown to yield superior results even in the presence of large boost velocity of the coordinate system indicating its Galilean invariance property. The standard Roe flux does not suffer from entropy violation when applied in the current nearly Lagrangian framework. The mesh motion provides automatic grid adaptation near shocks but may lead to very coarse cells inside expansion waves. Furthermore, in two dimensions, the shear flow makes the mesh distorted and sometimes even degenerate. We fix these problems by using a combination of mesh velocity correction algorithms and local mesh adaptation algorithms. Due to the local nature of mesh adaptation algorithms as well as the scheme, the method is embarrassingly parallel. The method itself does not depend on any physical model, and we expect the method to work for most other physical models. Due to the nature of algorithms, in the worst case scenario, the algorithms falls back to static mesh method, hence, it is at least as accurate as a fixed mesh simulation. We can glean the following conclusions from the exploration

- Completely local and explicit higher order ALE DG Methods are feasible for fluid simulations and provide much better results than fixed mesh methods.
- It is feasible to use only local remeshing methods to simulate fluid behavior in face of shocks and shear flows. This is possible while maintaining higher order accuracy and good mesh quality.
- The method behaves extremely well even in situations involving non-uniform grids. This indicates promising behavior for large simulations.
- The complete decoupling of mesh adaptation from solution evolution in the algorithm is promising, in that, it separates the physics of the model from the behavior of mesh. This indicates that future methods tackling different arbitrary physical models won't have to devise new methods for mesh adaptation, and instead can build on it.

6.9.1. Future Work

There are quite a few avenues to extend for the moving mesh method presented in the thesis along with some unanswered questions:

- One of the major avenues for exploration is to extend the method to other physical models. The main challenge is to find the correct mesh velocity algorithms which can work with a variety of different physical model as well as work with mixed physical models. There is no single answer to this problem, however, we believe that a combination of better mesh velocity algorithms and mesh adaptation algorithms, among other measures will be good enough to solve these problems.

- Even though, there is no apparent coupling of the limiting step with the step of moving in the mesh in this algorithm, the effect of one on another is quite apparent, and often manifest itself in the higher modes of the solution. One of the factors which we believe would improve the situation is to have more accurate predictors for the solution, which will reduce the need to limit the solution. There are different ways in which this can be achieved, one can use more accurate solutions inside a cell instead of a predictor to predict the solutions with some minimal input from neighboring cells. Another possibility is to use machine learning algorithms to predict the solution inside the cell.
- A big area which has been left unexplored in the thesis is the area of mesh adaptation algorithms. In effect, we have used extremely rudimentary mesh adaptation algorithms, and those have been quite successful in the simulations. The advantage of using such rudimentary algorithms is that they are extremely cheap to use. However, a big disadvantage of these algorithms is the lack of control when meshes are highly distorted. In such scenarios, it makes sense to fall back to more powerful local remeshing techniques like constrained Delaunay triangulations. These algorithms will ensure that the simulation will never fail, and will maintain the accuracy at least as good as a fixed mesh algorithm.
- The local nature of the algorithm allows for a favorable GPGPU based implementation.

Part III.

Kinetic Methods

Chapter

7 Introduction

$$\partial_t \mathbf{w} + \sum_{k=1}^{D} \partial_k \mathbf{q}^k(\mathbf{w}) = \mathbf{s} \tag{7.1}$$

Various models of compressible fluid that appear in gas dynamics, biology, astrophysics, or plasma physics for tokamaks can be unified as (7.1). Here, $\mathbf{w}\colon \mathbb{R}^D \times [0, T_{\text{max}}] \to \mathbb{R}^m$ is the vector of conserved variable, $\mathbf{q}^k(\mathbf{w})\colon \mathbb{R}^m \times \mathbb{R}^m$ is the flux and $\mathbf{s}\colon \mathbb{R}^D \times \mathbb{R} \times \mathbb{R}^m \to \mathbb{R}^m$ is a source term. D represents the physical space dimension and m the number of unknowns. In domains like MHD flows, low Mach Euler equations, and shallow-water with sedimentation, the model presents several time scales associated to the propagation of different waves. When the time scale of fast phenomena, which constrains the explicit CFL condition, is very small compared to the time scale of the most relevant phenomena, it becomes necessary to switch to implicit schemes. However classical implicit schemes are very costly in 2D or 3D because they require the resolution of linear or non-linear systems at each time step. In addition, the matrices associated with the hyperbolic systems are generally ill-conditioned.

In this work, we propose to follow another approach for avoiding the solution of complicated linear systems. Instead of solving the full fluid model in (7.1) directly, we use the idea from [Cou+16], where we replace the model with a simpler kinetic interpretation made of of a set of transport equations coupled through a stiff relaxation terms[AN00; BGP00; Gra14]. The resultant kinetic system is then solved by a splitting method where the transport and the relaxation stages are treated separately.

The one-dimensional version of the work was already presented in [Cou+16]. In this work, we present a massively parallel implementation of the method in higher dimensions. We particularly focus our presentation of the massive parallelization of the method with the StarPU runtime system

Chapter

8

Task-based Parallelization of an Implicit Kinetic Scheme

In this chapter, we present a task-based parallelization of an implicit kinetic scheme presented in the paper [Bad+18] (linked here). As mentioned in Chapter 7, the main focus of this work is to present a massively parallel implementation of the kinetic scheme from [Cou+16] in higher dimensions. In this chapter, we first show that there exists a general kinetic interpretation of any system of conservation laws. We then detail an approximation based on discontinuous Galerkin method with an upwind numerical flux and Gauss-Lobatto quadrature points. We then layout a scheme to parallelize the kinetic ethods effectively using the task-parallelization techniques supported in the StarPU library. Finally, we present some experimental results verifying the correctness of the scheme and showing the scaling efficiency of the implementation.

8.1. The Kinetic Interpretation

We consider the following kinetic equation

$$\partial_t \mathbf{f} + \sum_{k=1}^{D} \mathbf{V}^k \partial_k \mathbf{f} = \frac{1}{\tau}(\mathbf{f}^{eq}(\mathbf{f}) - \mathbf{f}) + \mathbf{g}. \tag{8.1}$$

The unknown is a vectorial distribution function $\mathbf{f}(\mathbf{x}, t) \in \mathbb{R}^{n_v}$ depending on the space variable $\mathbf{x} = (x^1 \dots x^D) \in \mathbb{R}^D$ and time $t \in \mathbb{R}$. $\mathbf{g}(\mathbf{x}, t, \mathbf{f})$ is a vectorial source term, possibly depending on space, time and $\mathbf{f}$. The partial derivatives are noted

$$\partial_t = \frac{\partial}{\partial t}, \quad \partial_k = \frac{\partial}{\partial x^k}.$$

The relaxation time τ is a small positive constant. The constant matrices $\mathbf{V}^k$, $1 \le k \le D$ are diagonal

$$\mathbf{V}^k = \begin{pmatrix} v_1^k & & & \\ & v_2^k & & \\ & & \ddots & \\ & & & v_{n_v}^k \end{pmatrix}$$

In other words, (8.1) is a set of n_v transport equations at constant velocities $\mathbf{v}_i = (v_i^1, \dots, v_i^D)$, coupled through a stiff BGK relaxation, and with an optional additional source term. We denote by $\mathbf{V} \cdot \partial = \sum_{k=1}^{D} \mathbf{V}^k \partial_k$ the transport operator, and by $\mathbf{Nf} = (\mathbf{f}^{eq}(\mathbf{f}) - \mathbf{f})/\tau$ the BGK relaxation term (also called the ``collision'' term).

Generally, this kinetic model represents an underlying hyperbolic system of conservation laws. The macroscopic conservative variables $\mathbf{w}(\mathbf{x},t) \in \mathbb{R}^m$ are obtained through a linear transformation

$$\mathbf{w} = \mathbf{P}\mathbf{f}, \tag{8.2}$$

where $\mathbf{P}$ is a $m \times n_v$ matrix. Generally the number of conservative variables is smaller than the number of kinetic data: $m < n_v$. The equilibrium (or ``Maxwellian'') distribution $\mathbf{f}^{eq}(\mathbf{f})$ is such that

$$\mathbf{P}\mathbf{f} = \mathbf{P}\mathbf{f}^{eq}(\mathbf{f}),$$

and

$$\mathbf{w} = \mathbf{P}\mathbf{f}_1 = \mathbf{P}\mathbf{f}_2 \Rightarrow \mathbf{f}^{eq}(\mathbf{f}_1) = \mathbf{f}^{eq}(\mathbf{f}_2), \tag{8.3}$$

which states that the equilibrium actually depends only on the macroscopic data $\mathbf{w}$. We could have used the notation $\mathbf{f}^{eq} = \mathbf{f}^{eq}(\mathbf{w}) = \mathbf{f}^{eq}(\mathbf{P}\mathbf{f})$, but we have decided to respect a well-established tradition.

When $\tau \to 0$, the kinetic equations provide an approximation of the system of conservation laws

$$\partial_t \mathbf{w} + \sum_{k=1}^{D} \partial_k \mathbf{q}^k(\mathbf{w}) = \mathbf{s}, \tag{8.4}$$

where the flux is given by

$$\mathbf{q}^k(\mathbf{w}) = \mathbf{P}\mathbf{V}^k\mathbf{f}^{eq}(\mathbf{f}).$$

The flux is indeed a function of $\mathbf{w}$ only because of (8.3).

Similarly the source term is given by

$$\mathbf{s}(\mathbf{x},t,\mathbf{w}) = \mathbf{P}\mathbf{g}(\mathbf{x},t,\mathbf{f}^{eq}) \tag{8.5}$$

System (8.1) has to be supplemented with conditions at the boundary $\partial\Omega$ of the computational domain Ω. We denote by $\mathbf{n} = (n_1 \ldots n_D)$ the outward normal vector on $\partial\Omega$. For simplicity, we shall only consider very simple imposed and time-independent boundary conditions $\mathbf{f}^b$. We note

$$\mathbf{V}\cdot\mathbf{n} = \sum_{k=1}^{D} \mathbf{V}^k n_k, \quad \mathbf{V}\cdot\mathbf{n}^+ = \max(\mathbf{V}\cdot\mathbf{n}, 0), \quad \mathbf{V}\cdot\mathbf{n}^- = \min(\mathbf{V}\cdot\mathbf{n}, 0).$$

A natural boundary condition, which is compatible with the transport operator $\mathbf{V}\cdot\partial$, is

$$\mathbf{V}\cdot\mathbf{n}^-\mathbf{f}(\mathbf{x},t) = \mathbf{V}\cdot\mathbf{n}^-\mathbf{f}^b(\mathbf{x}), \quad \mathbf{x} \in \partial\Omega. \tag{8.6}$$

It states that for a given velocity $\mathbf{v}_i$, the corresponding boundary data f_i^b is used only at the inflow part of the boundary.

Let us point out that the programming optimization that we propose in this paper rely in an essential way on the nature of the boundary condition (8.6). For other boundary conditions, such as periodic or wall conditions, additional investigations are still needed.

8.2. Numerical method

8.2.1. Discontinuous Galerkin approximation

For solving (8.1) we shall treat the transport operator $\mathbf{V} \cdot \partial$ and the collision operator $\mathbf{N}$ efficiently, thanks to a splitting approach. This allows to achieve a better parallelism. Let us start with the description of the transport solver.

For a simple exposition, we only consider one single scalar transport equation for $f(\mathbf{x}, t) \in \mathbb{R}$ at constant velocity $\mathbf{v}$

$$\partial_t f + \mathbf{v} \cdot \nabla f = 0. \tag{8.7}$$

The general vectorial case is easily deduced.

We consider a mesh $\mathcal{M}$ of Ω made of open sets, called ``cells", $\mathcal{M} = \{L_i,\ i = 1 \ldots N_c\}$. In the most general setting, the cells satisfy

1. $L_i \cap L_j = \emptyset$, if $i \neq j$;
2. $\overline{\cup_i L_i} = \overline{\Omega}$.

In each cell $L \in \mathcal{M}$ we consider a basis of functions $(\varphi_{L,i}(\mathbf{x}))_{i=0\ldots N_d-1}$ constructed from polynomials of order d. We denote by h the maximal diameter of the cells. With an abuse of notation we still denote by f the approximation of f, defined by

$$f(\mathbf{x}, t) = \sum_{j=0}^{N_d-1} f_{L,j}(t)\varphi_{L,j}(\mathbf{x}), \quad \mathbf{x} \in L.$$

The DG formulation then reads: find the $f_{L,j}$'s such that for all cell L and all test function $\varphi_{L,i}$

$$\int_L \partial_t f \varphi_{L,i} - \int_L f\mathbf{v} \cdot \nabla \varphi_{L,i} + \int_{\partial L} (\mathbf{v} \cdot \mathbf{n}^+ f_L + \mathbf{v} \cdot \mathbf{n}^- f_R)\varphi_{L,i} = 0. \tag{8.8}$$

In this formula (see Figure 8.20):

- R denotes the neighboring cell to L along its boundary $\partial L \cap \partial R$, or the exterior of Ω on $\partial L \cap \partial \Omega$.
- $\mathbf{n} = \mathbf{n}_{LR}$ is the unit normal vector on ∂L oriented from L to R.
- f_R denotes the value of f in the neighboring cell R on $\partial L \cap \partial R$.
- If L is a boundary cell, one may have to use the boundary values instead: $f_R = f^b$ on $\partial L \cap \partial \Omega$.
- $\mathbf{v} \cdot \mathbf{n}^+ f_L + \mathbf{v} \cdot \mathbf{n}^- f_R$ is the standard upwind numerical flux encountered in many finite volume or DG methods.

In our application, we consider hexahedral cells. We have a reference cell

$$\hat{L} =]-1, 1[^D$$

and a smooth transformation $\mathbf{x} = \tau_L(\hat{\mathbf{x}})$, $\hat{\mathbf{x}} \in \hat{L}$, that maps $\hat{L}$ on L

$$\tau_L(\hat{L}) = L.$$

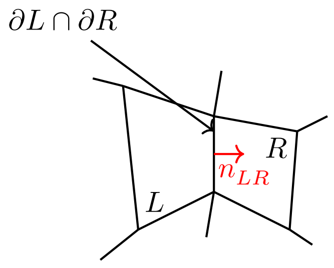


Figure 8.1.: Convention for the L and R cells orientation.

We assume that τ_L is invertible and we denote by τ_L' its (invertible) Jacobian matrix. We also assume that τ_L is a direct transformation

$$\det \tau_L' > 0.$$

In our implementation τ_L is a quadratic map based on hexahedral curved ``H20" finite elements with 20 nodes. The mesh of H20 finite elements is generated by `gmsh` [GR09].

On the reference cell, we consider the Gauss-Lobatto points $(\hat{\mathbf{x}}_i)_{i=0\dots N_d-1}$, $N_d = (d+1)^D$ and associated weights $(\omega_i)_{i=0\dots N_d-1}$. They are obtained by tensor products of the $(d+1)$ one-dimensional Gauss-Lobatto (GL) points on $]-1,1[$. The reference GL points and weights are then mapped to the physical GL points of cell L by

$$\mathbf{x}_{L,i} = \tau_L(\hat{\mathbf{x}}_i), \quad \omega_{L,i} = \omega_i \det \tau_L'(\hat{\mathbf{x}}_i) > 0. \tag{8.9}$$

In addition, the six faces of the reference hexahedral cell are denoted by F_ϵ, $\epsilon = 1 \dots 6$ and the corresponding outward normal vectors are denoted by $\hat{\mathbf{n}}_\epsilon$. A big advantage of choosing the GL points is that the volume and the faces share the same quadrature points. A special attention is necessary for defining the face quadrature weights. If a GL point $\hat{\mathbf{x}}_i \in F_\epsilon$, we denote by μ_i^ϵ the corresponding quadrature weight on face F_ϵ. We also use the convention that $\mu_i^\epsilon = 0$ if $\hat{\mathbf{x}}_i$ does not belong to face F_ϵ. A given GL point $\hat{\mathbf{x}}_i$ can belong to several faces when it is on an edge or in a corner of $\hat{L}$. Because of symmetry, we observe that if $\mu_i^\epsilon \neq 0$, then the weight μ_i^ϵ does not depend on ϵ.

We then consider basis functions $\hat{\varphi}_i$ on the reference cell: they are the Lagrange polynomials associated to the Gauss-Lobatto point and thus satisfy the interpolation property

$$\hat{\varphi}_i(\hat{\mathbf{x}}_j) = \delta_{ij}.$$

The basis functions on cell L are then defined according to the formula

$$\varphi_{L,i}(\mathbf{x}) = \hat{\varphi}_i(\tau_L^{-1}(\mathbf{x})).$$

In this way, they also satisfy the interpolation property

$$\varphi_{L,i}(\mathbf{x}_{L,j}) = \delta_{ij}. \tag{8.10}$$

In this paper, we only consider conformal meshes: the GL points on cell L are supposed to match the GL points of cell R on their common face. Dealing with non-matching cells is the object of a forthcoming work.

Let L and R be two neighboring cells. Let $\mathbf{x}_{L,j}$ be a GL point in cell L that is also on the common face between L and R. In the case of conformal meshes, it is possible to define the index j' such that

$$\mathbf{x}_{L,j} = \mathbf{x}_{R,j'}.$$

Applying a numerical integration to (8.8), using (8.9) and the interpolation property (8.10), we finally obtain

$$\partial_t f_{L,i}\omega_{L,i} - \sum_{j=0}^{N_d-1} \mathbf{v}\cdot\nabla\varphi_{L,i}(\mathbf{x}_{L,j})f_{L,j}\omega_{L,j} + \sum_{\epsilon=1}^{6}\mu_i^\epsilon\left(\mathbf{v}\cdot\mathbf{n}_\epsilon(\mathbf{x}_{L,i})^+ f_{L,i} + \mathbf{v}\cdot\mathbf{n}_\epsilon(\mathbf{x}_{L,i})^- f_{R,i'}\right) = 0. \quad (8.11)$$

We have to detail how the gradients and normal vectors are computed in the above formula. Let $\mathbf{A}$ be a square matrix. We recall that the cofactor matrix of $\mathbf{A}$ is defined by

$$\mathrm{co}(\mathbf{A}) = \det(\mathbf{A})\left(\mathbf{A}^{-1}\right)^T. \quad (8.12)$$

The gradient of the basis function is computed from the gradients on the reference cell using (8.12)

$$\nabla\varphi_{L,i}(\mathbf{x}_{L,j}) = \frac{1}{\det\tau_L'(\hat{\mathbf{x}}_i)}\mathrm{co}(\tau_L'(\hat{\mathbf{x}}_j))\hat{\nabla}\hat{\varphi}_i(\hat{\mathbf{x}}_j).$$

In the same way, the scaled normal vectors $\mathbf{n}_\epsilon$ on the faces are computed by the formula

$$\mathbf{n}_\epsilon(\mathbf{x}_{L,i}) = \mathrm{co}(\tau_L'(\hat{\mathbf{x}}_i))\hat{\mathbf{n}}_\epsilon.$$

We introduce the following notation for the cofactor matrix

$$\mathbf{c}_{L,i} = \mathrm{co}(\tau_L'(\hat{\mathbf{x}}_i)).$$

The DG scheme then reads

$$\partial_t f_{L,i} - \frac{1}{\omega_{L,i}}\sum_{j=0}^{N_d-1}\mathbf{v}\cdot\mathbf{c}_{L,j}\hat{\nabla}\hat{\varphi}_i(\hat{\mathbf{x}}_j)f_{L,j}\omega_j + \frac{1}{\omega_{L,i}}\sum_{\epsilon=1}^{6}\mu_i^\epsilon\left(\mathbf{v}\cdot\mathbf{c}_{L,i}\hat{\mathbf{n}}_\epsilon{}^+ f_{L,i} + \mathbf{v}\cdot\mathbf{c}_{L,i}\hat{\mathbf{n}}_\epsilon{}^- f_{R,i'}\right) = 0. \quad (8.13)$$

On boundary GL points, the value of $f_{R,i'}$ is given by the boundary condition

$$f_{R,i'} = f^b(\mathbf{x}_{L,i}), \quad \mathbf{x}_{L,i} = \mathbf{x}_{R,i'}.$$

For practical reasons, it is interesting to also consider $f_{R,i'}$ as an artificial unknown in the fictitious cell. The fictitious unknown is then a solution of the differential equation

$$\partial_t f_{R,i'} = 0. \quad (8.14)$$

In the end, if we put all the unknowns in a large vector $\mathbf{F}(t)$, (8.13), (8.14) read as a large system of coupled differential equations

$$\partial_t \mathbf{F} = \mathbf{L}_h \mathbf{F}. \tag{8.15}$$

In the following, we call $\mathbf{L}_h$ the transport matrix. The transport matrix satisfies the following properties:

- $\mathbf{L}_h \mathbf{F} = 0$ if the components of $\mathbf{F}$ are all the same.
- Let $\mathbf{F}$ be such that the components corresponding to the boundary term vanish. Then $\mathbf{F}^T \mathbf{L}_h \mathbf{F} \leq 0$. This dissipation property is a consequence of the choice of an upwind numerical flux.
- In many cases, and with a good numbering of the unknowns in $\mathbf{F}$, $\mathbf{L}_h$ has a triangular structure. This aspect is discussed in Subsection 8.3.1.

As stated above, we actually have to apply a transport solver for each constant velocity $\mathbf{v}_i$.

Let L be a cell of the mesh $\mathcal{M}$ and $\mathbf{x}_i$ a GL point in L. As in the scalar case, we denote by $\mathbf{f}_{L,i}$ the approximation of $\mathbf{f}$ in L at GL point i. In the sequel, with an abuse of notation and according to the context, we may continue to note $\mathbf{F}(t)$ the big vector made of all the vectorial values $\mathbf{f}_{L,j}$ at all the GL points j in all the (real or fictitious) cells L.

We may also continue to denote by $\mathbf{L}_h$ the matrix made of the assembly of all the transport operators for all velocities $\mathbf{v}_i$. With a good numbering of the unknowns it is possible in many cases to suppose that $\mathbf{L}_h$ is block-triangular. More precisely, because in the transport step the equations are uncoupled, we see that $\mathbf{L}_h$ can be made block-diagonal, each diagonal block being itself block-triangular. See Section 8.3.1.

8.2.2. Palindromic time integration

We can also define an approximation $\mathbf{N}_h$ of the collision operator $\mathbf{N}$. We define by $\mathbf{F}^{eq}(\mathbf{F})$ the big vector made of all the $\mathbf{f}^{eq}(\mathbf{f}_{L,i})$, $L \in \mathcal{M}$, $i = 0 \ldots N_d - 1$.

We set

$$\mathbf{N}_h \mathbf{F} = \frac{1}{\tau}(\mathbf{F}^{eq}(\mathbf{F}) - \mathbf{F}). \tag{8.16}$$

Similarly we note $\mathbf{G}_h$ the discrete approximation of the kinetic source term $\mathbf{g}$.

The DG approximation of (8.1) finally reads

$$\partial_t \mathbf{F} = \mathbf{L}_h \mathbf{F} + \mathbf{N}_h \mathbf{F} + \mathbf{G}_h \mathbf{F}.$$

We use the following Crank-Nicolson second order time integrator for the transport equation:

$$\exp(\Delta t \mathbf{L}_h) \simeq T_2(\Delta t) := (\mathbf{I} + \frac{\Delta t}{2}\mathbf{L}_h)(\mathbf{I} - \frac{\Delta t}{2}\mathbf{L}_h)^{-1}. \tag{8.17}$$

Similarly, for the collision integrator, we use

$$\exp(\Delta t \mathbf{N}_h) \simeq C_2(\Delta t) := (\mathbf{I} + \frac{\Delta t}{2}\mathbf{N}_h)(\mathbf{I} - \frac{\Delta t}{2}\mathbf{N}_h)^{-1}.$$

Because during the collision step, the conservative variables $\mathbf{w} = \mathbf{P}\mathbf{f}$ do not change, the collision integrator is only apparently implicit. We have the explicit formula:

$$C_2(\Delta t)\mathbf{F} = \frac{(2\tau - \Delta t)\mathbf{F}}{2\tau + \Delta t} + \frac{2\Delta t \mathbf{F}^{eq}(\mathbf{F})}{2\tau + \Delta t}. \tag{8.18}$$

The source operator is also approximated by a Crank-Nicolson integrator

$$\exp(\Delta t \mathbf{G}_h) \simeq S_2(\Delta t) := (\mathbf{I} + \frac{\Delta t}{2}\mathbf{G}_h)(\mathbf{I} - \frac{\Delta t}{2}\mathbf{G}_h)^{-1},$$

requiring to solve a nonlinear local equation whenever **g** depends on **f**.

If $\tau > 0$, we observe that the operators T_2 and C_2 are **time-symmetric**: if we set $O_2 = T_2$, $O_2 = C_2$, or $O_2 = S_2$, then O_2 satisfies

$$O_2(-\Delta t) = O_2(\Delta t)^{-1}, \quad O_2(0) = Id. \tag{8.19}$$

This property implies that O_2 is necessarily a second order approximation of the exact integrator [Hai+06; MQ02] When $\tau = 0$, we also remark that

$$C_2(\Delta t)\mathbf{F} = 2\mathbf{F}^{eq}(\mathbf{F}) - \mathbf{F} \neq \mathbf{F}$$

and then C_2 does not satisfy (8.19) anymore.

For $\tau > 0$, the Strang formula permits us to construct a five steps second order time-symmetric approximation

$$M_2^s(\Delta t) = T_2(\frac{\Delta t}{2})S_2(\frac{\Delta t}{2})C_2(\Delta t)S_2(\frac{\Delta t}{2})T_2(\frac{\Delta t}{2}) = \exp\left(\Delta t\left(\mathbf{L}_h + \mathbf{N}_h + \mathbf{S}_h\right)\right) + O(\Delta t^3),$$

and a three step one

$$M_2(\Delta t) = T_2(\frac{\Delta t}{2})C_2(\Delta t)T_2(\frac{\Delta t}{2}) = \exp\left(\Delta t\left(\mathbf{L}_h + \mathbf{N}_h\right)\right) + O(\Delta t^3),$$

in the source-less case.

However this formula is no more a second order approximation of (8.1) when $\tau \to 0$. Indeed, when $\tau = 0$

$$M_2(0)\mathbf{F} = 2\mathbf{F}^{eq}(\mathbf{F}) - \mathbf{F}.$$

As explained in [Cou+16] it is better to consider the following method, which remains second order accurate even for infinitely fast relaxation:

$$M_2^{kin}(\Delta t) = T_2(\frac{\Delta t}{4})C_2(\frac{\Delta t}{2})T_2(\frac{\Delta t}{2})C_2(\frac{\Delta t}{2})T_2(\frac{\Delta t}{4}).$$

By palindromic compositions of the second order method M_2^{kin} it is then very easy to achieve any even order of accuracy (see [Cou+16]). However, in this paper, we concentrate on the parallel optimization of the method and we shall only present numerical results at second order for the limit system 8.4. To that end it is sufficient to use the method M_2, as $PM_2(0)$ properly converges towards identity on the macroscopic variable space when $\tau \to 0$.

8.3. Optimization of the kinetic solver

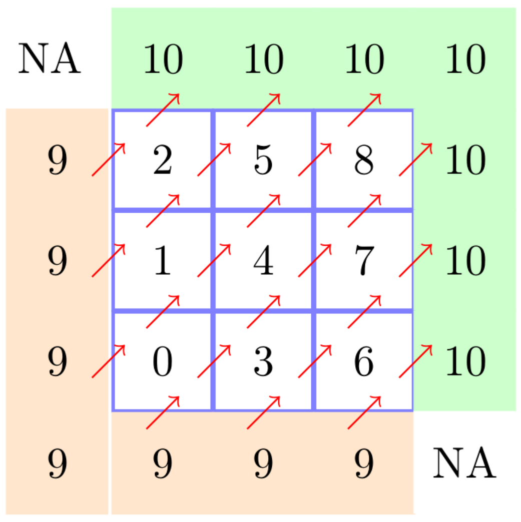
In this section, we describe the optimizations that can be applied in the implementation of the previous numerical method.

8.3.1. Triangular structure of the transport matrix

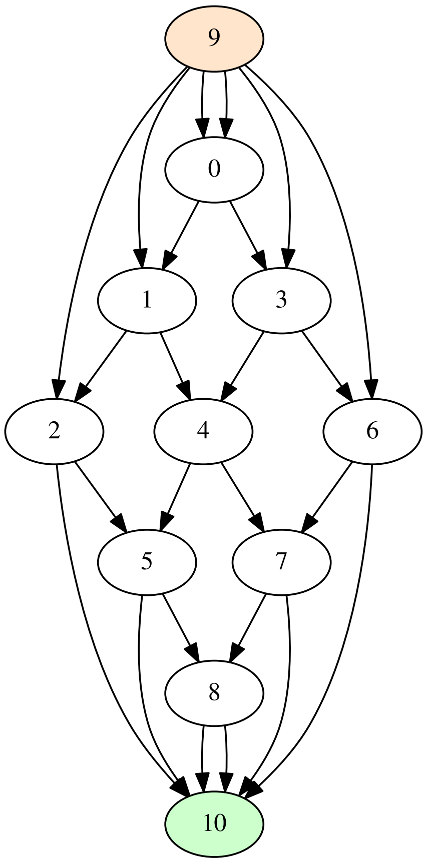
Because of the upwind structure of the numerical flux, it appears that the transport matrix is often block-triangular. This is very interesting because this allows to apply implicit schemes to (8.15) without the costly inversion of linear systems [Mou+15]. We can provide the formal structure of $\mathbf{L}_h$ through the construction of a directed graph $\mathcal{G}$ with a set of vertices $\mathcal{V}$ and a set of edges $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$. The vertices of the graph are associated to the (real or fictitious) cells of $\mathcal{M}$. Consider now two cells L and R with a common face F_{LR}. We denote by $\mathbf{n}_{LR}$ the normal vector on F_{LR} oriented from L to R. If there is at least one GL point $\mathbf{x}$ on F_{LR} such that

$$\mathbf{n}_{LR}(\mathbf{x}) \cdot \mathbf{v} > 0,$$

then the edge from L to R belongs to the graph (see Figure 8.2):



(a) Example of mesh (it is structured here but it is not necessary)



(b) The corresponding dependency graph $\mathcal{G}$

Figure 8.2.: Construction of the dependency graph. Left: example of mesh (it is structured here but it is not necessary) with 9 interior cells. The velocity field v is indicated by red arrows. We add two fictitious cells: one for the upwind boundary condition (cell 9) and one for the outflow part of $\partial\Omega$ (cell 10). Right: the corresponding dependency graph $\mathcal{G}$. By examining the dependency graph, we observe that the values of $\mathbf{F}^{n+1}$ in cell 0 have to be computed first, using the boundary conditions. Then cells 1 and 3 can be computed in parallel, then cells 2, 4, and 6 can be computed in parallel, then **etc**.

In (8.13) we can distinguish between several kinds of terms. We write

$$\partial_t f_L + \Gamma_{L \leftarrow L} f_L + \sum_{(R,L)\in\mathcal{E}} \Gamma_{L \leftarrow R} f_R,$$

with

$$\Gamma_{L\leftarrow L} f_L = -\frac{1}{\omega_{L,i}} \sum_{j=0}^{N_d-1} \mathbf{v}\cdot \mathbf{c}_{L,j} \hat{\nabla}\hat{\varphi}_i(\hat{\mathbf{x}}_j) f_{L,j}\omega_j +$$
$$\frac{1}{\omega_{L,i}} \sum_{\epsilon=1}^{6} \mu_i^\epsilon \mathbf{v}\cdot \mathbf{c}_{L,i} \hat{\mathbf{n}}_\epsilon{}^+ f_{L,i},$$

and, if $(R,L)\in\mathcal{E}$,

$$\Gamma_{L\leftarrow R} f_R = \frac{1}{\omega_{L,i}} \mu_i^\epsilon \mathbf{v}\cdot \mathbf{c}_{L,i} \hat{\mathbf{n}}_\epsilon{}^- f_{R,i'}.$$

We can use the following convention

$$(R,L)\notin\mathcal{E} \Rightarrow \Gamma_{L\leftarrow R} = 0. \tag{8.20}$$

$\Gamma_{L\leftarrow L}$ contains the terms that couple the values of f inside the cell L. They correspond to diagonal blocks of size $(d+1)^D \times (d+1)^D$ in the transport matrix $\mathbf{L}_h$. $\Gamma_{L\leftarrow R}$ contains the terms that couple the values inside cell L with the values in the neighboring upwind cell R. If R is a downwind cell relatively to L then $\mu_i^\epsilon \mathbf{v}\cdot C_{L,i}\hat{\mathbf{n}}_\epsilon{}^- = 0$ and $\Gamma_{L\leftarrow R} = 0$ is indeed compatible with the above convention (8.20).

Once the graph $\mathcal{G}$ is constructed, we can analyze it with standard tools. If it contains no cycle, then it is called a Directed Acyclic Graph (DAG). Any DAG admits a topological ordering of its nodes. A topological ordering is a numbering of the cells $i \mapsto L_i$ such that if there is a path from L_i to L_j in $\mathcal{G}$ then $j > i$. In practice, it is useful to remove the fictitious cells from the topological ordering. In our implementation they are put at the end of the list.

Once the new ordering of the graph vertices is constructed, we can construct a numbering of the components of **F** by first numbering the unknowns in L_0 then the unknowns in L_1, **etc**. More precisely, we set

$$F_{kN_d+i} = f_{L_k,i}.$$

Then, with this ordering, the matrix $\mathbf{L}_h$ is lower block-triangular with diagonal blocks of size $(d+1)^D \times (d+1)^D$. It means that we can apply implicit schemes to (8.15) without inverting large linear systems.

As stated above, we actually have to apply a transport solver for each constant velocity $\mathbf{v}_i$. In the sequel, with another abuse of notation and according to the context, we continue to note **F** the big vector made of all the vectorial values $\mathbf{f}_{L,j}$ at all the GL points j in all the (real or fictitious) cells L.

We may also continue to denote by $\mathbf{L}_h$ the matrix made of the assembly of all the transport operators for all velocities $\mathbf{v}_i$. With a good numbering of the unknown it is still possible to suppose that $\mathbf{L}_h$ is block-triangular. More precisely, as in the transport step the equations are uncoupled, we see that $\mathbf{L}_h$ can be made a block-diagonal matrix, each diagonal block being itself block-triangular.

8.3.2. Parallelization of the implicit solver

In this section, we explain how it is possible to parallelize the transport solver. Here again we consider the single transport equation (8.7) and the associated differential equation (8.15). We apply a second order Crank-Nicolson implicit scheme. We have explained in Section 8.2.2 how to increase the order of the scheme. We compute an approximation $\mathbf{F}^n$ of $\mathbf{F}(n\Delta t)$. The implicit scheme reads

$$(\mathbf{I} - \Delta t\mathbf{L}_h)\mathbf{F}^{n+1} = (\mathbf{I} + \Delta t\mathbf{L}_h)\mathbf{F}^n. \tag{8.21}$$

As explained above, the matrices $(\mathbf{I} - \Delta t \mathbf{L}_h)$ and $(\mathbf{I} + \Delta t \mathbf{L}_h)$ are lower triangular. It is thus possible to solve the linear system explicitly cell after cell, assuming that the cells are numbered in a topological order.

It is possible to perform further optimization by harnessing the parallelism exhibited by the dependency graph. Indeed, once the values of f in the first cell are computed, it is generally possible to compute in parallel the values of f in neighboring downwind cells. For example, as can be seen on Figure 8.2, once the values in cells 0, 1 and 2 are known, we can compute independently, and in parallel, the values in cells 2, 4 and 6.

We observe that at the beginning and at the end of the time step, the computations are ``less parallel'' than in the middle of the time step, where the parallelism is maximal.

Implementing this algorithm with OpenMP or using pthread is not very difficult. However, it requires to compute the data dependencies between the computational tasks carefully, and to set adequate synchronization points in order to get correct results. In addition, a rough implementation will probably not exhibit optimized memory access. Therefore, we have decided to rely on a more sophisticated tool called StarPU[1] for submitting the parallel tasks to the available computational resources.

StarPU is a runtime system library developed at Inria Bordeaux [Aug+12]. It relies on the data-based parallelism paradigm.

The user has first to split its whole problem into elementary computational tasks. The elementary tasks are then implemented into *codelets*, which are simple C functions. The same task can be implemented differently into several codelets. This allows the user to harness special acceleration devices, such as vectorial CPU cores, GPUs or Intel KNL devices, for example. In the StarPU terminology these devices are called *workers*.

For each task, the user has also to describe precisely what are the input data, in *read* mode, and the output data, in *write* or *read-write* mode. The user then submits the task in a sequential way to the StarPU system. StarPU is able to construct at runtime a task graph from the data dependencies. The task graph is analyzed and the tasks are scheduled automatically to the available workers (CPU cores, GPUs, **etc**.). If possible, they are executed in parallel into concurrent threads. The data transfer tasks between the threads are automatically generated and managed by StarPU, which greatly simplifies the programming.

When a StarPU program is executed, it is possible to choose among several schedulers. The simplest *eager* scheduler adopts a very simple strategy, where the tasks are executed in the order of submission by the free workers, without optimization. More sophisticated schedulers, such as the *dmda* scheduler, are able to measure the efficiency of the different codelets and the data transfer times, in order to apply a more efficient allocation of tasks.

Recently a new data access mode has been added to StarPU: the *commute* mode. In a task, a buffer of data can now be accessed in commute mode, in addition to the write or read-write modes. A commute access tells to StarPU that the execution of the corresponding task may be executed before or after other tasks containing commutative access. This allows StarPU to perform additional optimizations.

There exists also a MPI version of StarPU. In the MPI version, the user has to decide an initial distribution of data among the MPI nodes. Then the tasks are submitted as usual (using the function starpu_mpi_insert_-task instead of starpu_insert_task). Required MPI communications are automatically generated by StarPU. For the moment, this approach does not guarantee a good load balancing. It is the responsibility of the user to migrate data from one MPI node to another for improving the load balancing, if necessary.

[1]http://starpu.gforge.inria.fr

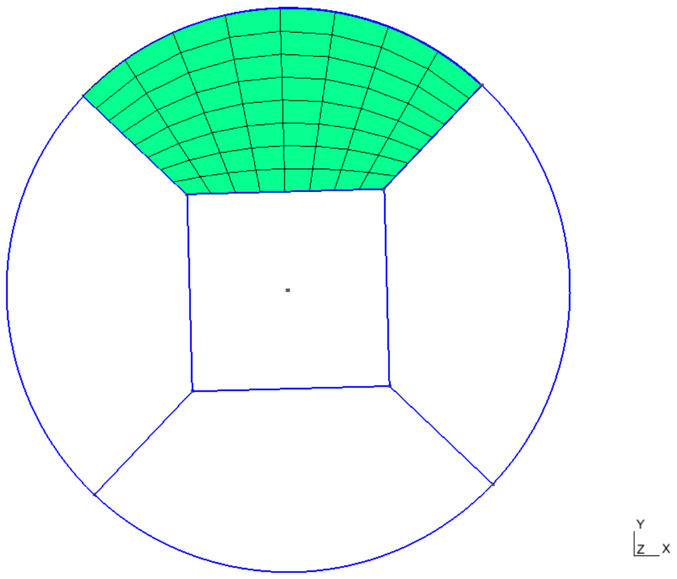


Figure 8.3.: Macrocell approach: an example of a mesh made of five macrocells. Each macrocell is then split into several subcells. Only the subcells of the top macrocell are represented here (in green).

8.3.3. Macrocell approach

StarPU is quite efficient, but there is an unavoidable overhead due to the task submissions and to the on-the-fly construction and analysis of the task graph. Therefore it is important to ensure that the computational tasks are not too small, in which case the overhead is not amortized, or not too big, in which case some workers are idle.

For achieving the right balance, we have decided not to apply directly the above task submission algorithm to the cells but to groups of cells instead.

The implementation of the whole kinetic scheme has been made into the `schnaps` software. `schnaps` is a C99 software dedicated to the numerical simulation of conservation laws.

In `schnaps` we construct first a *macromesh* of the computational domain. Then each *macrocell* of the macromesh is split into subcells. See Figure 8.3. We also arrange the subcells into a regular sub-mesh of the macrocells. In this way, it is possible to apply additional optimizations. For instance, the subcells L of a same macrocell $\mathcal{L}$ can now share the same geometrical transformation τ_L, which saves memory.

In `schnaps` we have also defined an *interface* structure in order to manage data communications between the macrocells. An interface contains the faces that are common to two neighboring macrocells. We do not proceed exactly as in Section 8.3.1 where the vertices of graph $\mathcal{G}$ were associated to cells and the edges to faces. Instead, we construct an upwind graph whose vertices are associated to macrocells, and edges to interfaces. This graph is then sorted, and the macrocells are numbered in a topological order.

For solving one time step of one transport equation (8.21), we split the computations into several elementary operations: for all macrocell $\mathcal{L}$ taken in a topological order, we perform the following tasks:

1. Volume residual assembly: this task computes in the macrocell $\mathcal{L}$ the part of the right hand side of (8.21) that comes from the values of f inside $\mathcal{L}$;

2. Interface residual assembly: this task computes, in the macrocell $\mathcal{L}$, the part of the right hand side of (8.21) that comes from upwind interface values;

3. Boundary residual assembly: this task computes, in the macrocell $\mathcal{L}$, the part of the right hand side of (8.21) that comes from upwind boundaries values.

4. Volume solve: this task solves the local transport linear system in the macrocell.

5. Extraction: this task copies the boundary data of $\mathcal{L}$ to the neighbor downwind interfaces.

Let us point out that in step 4 above, the macrocell local transport solver is reassembled and refactorized at each time step: we have decided not to store a sparse linear system in the macrocell for each velocity $\mathbf{v}_i$, in order to save memory. The local sparse linear system is solved thanks to the KLU library [DP10]. This library is able to detect efficiently sparse triangular matrix structures, which makes the resolution quite efficient. In practice, the factorization and resolution time of the KLU solver is of the same order as the residual assembly time.

In schnaps, we use the MPI version of StarPU. The macromesh is initially split into several subdomains and the subdomains are distributed to the MPI nodes. Then the above tasks are launched asynchronously with the starpu_mpi_insert_task function. MPI communications are managed automatically by StarPU.

It is clear that if we were solving a single transport equation our strategy would be very inefficient. Indeed, the downwind subdomains would have to wait for the end of the computations of the upwind subdomains. We are saved by the fact that we have to solve many transport equations in different directions. This helps the MPI nodes to be equally occupied. Our approach is more efficient if we avoid a domain decomposition with internal subdomains, because these subdomains have to wait the results coming from the boundaries.

In our approach it is also essential to launch the tasks in a completely asynchronous fashion. In this way, if a MPI node is waiting for results of other subdomains for a given velocity $\mathbf{v}_i$ it is not prevented from starting the computation for another velocity $\mathbf{v}_j$.

8.3.4. Collisions

In this section we explain how is computed the collision step (8.18). The computations are purely local to each GL point, which makes the collision step embarrassingly parallel. However it is not so obvious to attain high efficiency because of memory access. If the values of $\mathbf{F}$ are well arranged in memory in the transport stage, it means that the values of $\mathbf{f}$ attached to a given velocity $\mathbf{v}_i$ are close in memory, for a better data locality. On the contrary, in the collision step at a given GL point, a better locality is achieved if the values of $\mathbf{f}$ corresponding to different velocities are close in memory. Additional investigations and tests are needed in order to evaluate the importance of data locality in our algorithm.

In our implementation, we have adopted the following strategy. We have first identified the following task:

1. Reduction task for a velocity $\mathbf{v}_i$: this task is associated with one macrocell $\mathcal{L}$. It computes the contribution to $\mathbf{w}$ of the components of $\mathbf{f}$ that have been transported at velocity $\mathbf{v}_i$ with formula (8.2). The StarPU access to the buffer containing $\mathbf{w}$ is performed in read-write and commute modes. In this way the contribution from each velocity can be added to $\mathbf{w}$ as soon as it is available.

2. Relaxation task for a velocity $\mathbf{v}_i$: this task is associated to one macrocell $\mathcal{L}$. Once $\mathbf{w}$ is known, it computes the components of $\mathbf{f}^{eq}$ corresponding to velocity $\mathbf{v}_i$. Then it computes the relaxation step (8.18) for the associated component of $\mathbf{f}$.

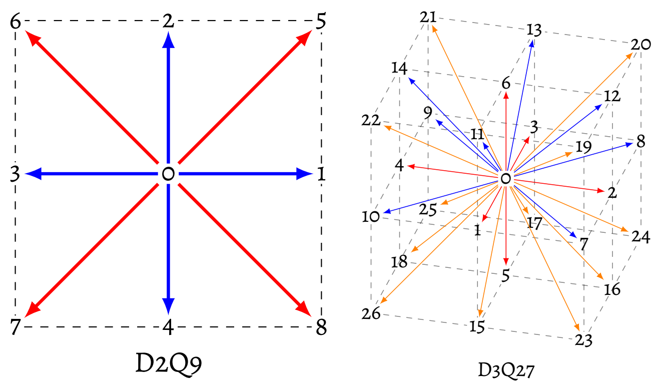


Figure 8.4.: D2Q9 and D3Q27 velocity grids.

In step 2 we can separate the computations for each velocity because the collision term (8.16) is diagonal. Some Lattice Boltzmann Methods rely on non-diagonal relaxations. It can be useful for representing more general viscous terms for instance. For non-diagonal relaxation we would have to change a little the algorithm.

We can now make a few comments about the storage cost of the method. In the end, we have to store at each GL point $\mathbf{x}_i$ and each cell L the values of $\mathbf{f}_{L,i}$ and $\mathbf{w}_{L,i}$. We do not have to keep the values of the previous time-step, $\mathbf{f}_{L,i}$ and $\mathbf{w}_{L,i}$ can be replaced by the new values as soon as they are available. In this sense, our method is ``low-storage". As explained in [Cou+16] it is also possible to increase the time order of the method, without increasing the storage.

8.3.5. Scaling test

For all performance tests presented in this section, we used standard models from the family Lattice-Boltzmann-Method (LBM) kinetic models devised for the simulation of Euler/Navier Stokes systems. We will not give a detailed description of their properties from the modeling point of view: we simply take them as good representative of the typical workload of kinetic relaxation schemes. The most relevant feature impacting the performance of our algorithm is the discrete velocity set of the kinetic model, which determines the task graph structure of the transport step when combined with a particular mesh topology. In standard LBM models, velocity sets are usually built-up from a sequence of pairs of opposite velocities with an additional zero velocity node. On Figure 8.4 we show the two representative velocity sets of the $2DQ9$ and $3DQ27$ LBM models.

8.3.5.1. Multithread performance (D2Q9, D3Q*)

We first tested the multithread performance of our implementation for the full (transport + relaxation) scheme for the standard D2Q9 model. All tests were performed on a single node of the IRMA-ATLAS cluster, with 24 available cores. We considered several square meshes build-up from 1 to 64 macrocells. The number of geometric degrees of freedom per element has been kept constant with a value of 3375 points per macrocell, so that the workload per macrocell did not change. For each mesh, we allowed StarPu to use from 1 to the full 24 cores of the node and measured the total wall time. The results for this first batch of performance measurements are given in Fig. 8.5. First we verify that for 1 unique macrocell, parallel performance saturates when the number of cores roughly equals the number of velocities in the model. This is to be expected, as no topological parallelism can be exploited in that case. Increasing the number of macro-element allows to take advantage of topological parallelism. For that workload, parallel efficiency saturates at about 80,

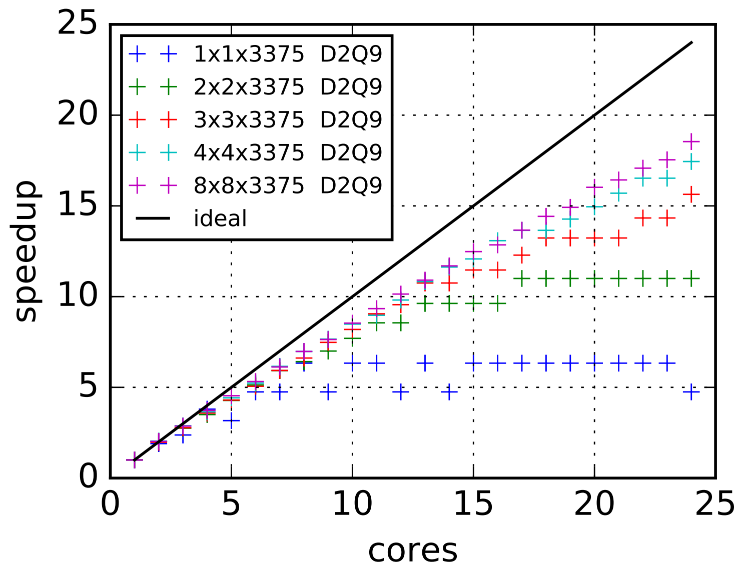


Figure 8.5.: Multithread scaling for the D2Q9 model and a collection of square meshes from 1 to 64 macro-elements.

which is quite good. On Fig. 8.6, we considered on the same cubic mesh three different models differing by the number of velocity values. Those cases exhibit a large amount of potential parallelism, due to the large number of velocities combined with the macrocell decomposition. On an ideal machine, they could in theory scale perfectly up to 24 cores. The observed saturation, still around 80 efficiency, is still quite good and comes from the unavoidable concurrency in memory access between the various cores and the scheduling overhead. It is important to note when considering those results that the bulk of the computational cost occurs in the transport step of the algorithm: though the collision step forces synchronization between all the fields corresponding to individual velocities for the computation of the macroscopic fields, its actual cost is negligible in regard of the transport step.

8.3.5.2. MPI scaling: D3Q15 in a torus

Having verified the good multithread performance of our code on a single node, we now check whether for larger problem sizes the workload can be distributed among several computing nodes. To that end, accounting for the fact that we aim notably at performing simulations for Tokamak physics, we considered a toroidal mesh subdivided into 720 macrocells. The workload distribution across nodes was made using a standard domain decomposition approach: the mesh was partitioned statically into as many sub-domains as computing nodes, ranging from 1to 4 for our experiment on the IRMA-ATLAS cluster. From an implementation point of view, the transition from a multithreaded code to a hybrid MPI/multithread one is made fairly easy by StarPU. When declaring data to StarPU, one simply has to specify the MPI process owning the data. At runtime, each MPI process hosts a local scheduler instance which acts only on data relevant to the local execution graph. All MPI communications are handled transparently by the local scheduler when inter-node data transfers are necessary. In table 8.1 we show the wall time for a hundred iterations of the full scheme for the D3Q15 model. The number of available threads per node was set to 14, matching the number of velocities actually participating in transport (there is one null velocity in the model). We observe a super-linear scaling when the load is spread from 1to 4 node. This is not surprising for such an experiment with fixed total problem size as both the memory load and size of the local task graph for each decrease when

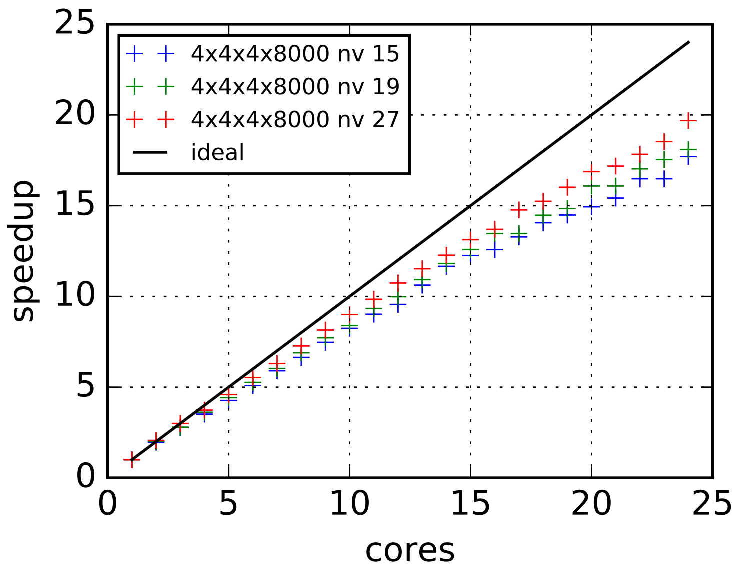


Figure 8.6.: Multithread scaling on a 4x4x4 macro-element mesh for models D3Q15, D3Q19 and D3Q27

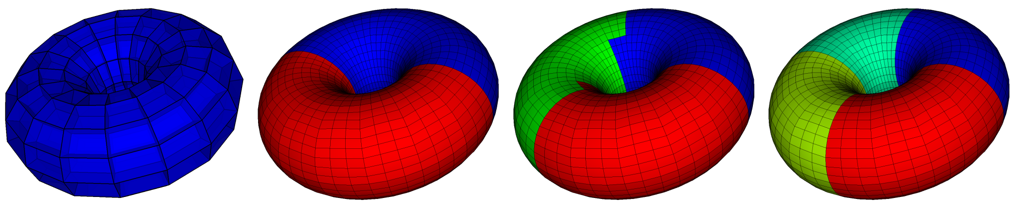


Figure 8.7.: Toroidal macromesh (720 macrocells) - Mesh partitions used in the MPI scaling tests.

the number of sub-domains increases.

8.4. Numerical results

8.4.1. Euler with gravity

For this test case we consider the isothermal Euler equations in two dimensions with a constant gravity source term

$$\partial_t \rho + \partial_k (\rho \mathbf{u}^{\mathbf{k}}) = 0, \tag{8.22}$$

$$\partial_t (\rho \mathbf{u}^k) + \partial_j \Pi^{kj} = \rho \mathbf{g}, \tag{8.23}$$

with $\Pi = \begin{bmatrix} \rho c^2 + \rho u_x^2 & \rho u_x u_y \\ \rho u_x u_y & \rho c^2 + u_y^2 \end{bmatrix}$ and $\mathbf{g} = -g\mathbf{e}_y$.

Nthreads/Nmpi	1	2	3	4
14	6862	2772	1491	1014

Table 8.1.: Wall time (in seconds) for the D3Q15 model for 1 to 4 mpi processes with 14 threads per process.

The conservative variables vector is thus $\mathbf{w} = [\rho, \rho u_x, \rho u_y]^t$. The kinetic model is the standard $D2Q9$ one with nine velocities

$$\mathbf{V} = \lambda \text{diag}\,[(0,0),(1,0),(0,1),(-1,0),(0,-1),(1,1),(-1,1),(-1,-1),(1,-1)] \tag{8.24}$$

and the (3×9) projection matrix Preads

$$P = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & \lambda & 0 & -\lambda & 0 & \lambda & -\lambda & -\lambda & \lambda \\ 0 & 0 & \lambda & 0 & -\lambda & \lambda & \lambda & -\lambda & -\lambda \end{bmatrix}, \tag{8.25}$$

i.e $\rho = \sum_i f_i$, $\rho\mathbf{u} = \sum_i f_i \mathbf{v}_i$.

The equilibrium distribution function is given by

$$f_i = w_i \rho \left(1 + \frac{(\mathbf{u}\cdot\mathbf{v}_i)}{c^2} + \frac{(\mathbf{u}\cdot\mathbf{v}_i)^2}{2c^4} - \frac{\mathbf{u}\cdot\mathbf{u}}{2c^2} \right) \tag{8.26}$$

with $c = \lambda/\sqrt{3}$, and the weights $w_0 = \frac{4}{9}$, $w_i = \frac{1}{9}$ for $i = 1, \dots, 4$, $w_i = \frac{1}{36}$ for $i = 5, \dots, 8$.

The stationary solution for a fluid at rest in the gravity field is

$$\rho = \rho_0 \exp(-gy/c^2), \qquad \mathbf{u} = 0 \tag{8.27}$$

For this test case, the numerical scheme is made up of three stages: a transport step (T), a source step (S) where the source is applied on the equilibrium part of the distribution function, and the collision step (C). Due to the absence of explicit time dependency and the linearity in $\mathbf{w}$ of the source, the local nonlinear resolution of the source operator converges in one Picard iteration. All steps are implemented as weighted implicit schemes, parametrized by a weight θ and a time stepΔt.

We compared several 1^{st} and 2^{nd} order splitting schemes built up from either fully implicit ($\theta = 1$) first order or Crank-Nicolson ($\theta = \frac{1}{2}$) steps:

- Lie first order splitting scheme $M_1^s = T_1(\Delta t)S_1(\Delta t)C_1(\Delta t)$ with first order building blocks.
- Lie first order splitting scheme $M_{1,2}^s = T_2(\Delta t)S_2(\Delta t)C_2(\Delta t)$ with second order building blocks, for which the order loss comes from the splitting error.
- a palindromic second order Strang scheme $M_2^s = T_2(\frac{\Delta t}{2})S_2(\frac{\Delta t}{2})C_2(\Delta t)S_2(\frac{\Delta t}{2})T_2(\frac{\Delta t}{2})$.
- a collapsed version of the second order Strang scheme $M_2^s$for which the last transport substep of each global step and the first transport substep of the next one are fused in a single $T_2(\Delta t)$ substep except obviously for the first and last time steps of the simulation.

We performed time order convergence tests on a $2D$ square mesh partitioned into 4×4 macrocells with 1024 points per macrocell. As shown by Fig. 8.8, we obtain the expected convergence orders for each of the splitting schemes used.

8.4.2. 2D Flow around a cylinder using a penalization method

In this test case we considered the flow of a fluid in a rectangular duct with a cylindrical solid obstacle. The simulation domain is the rectangle $[-1,1]\times[-5,5]$

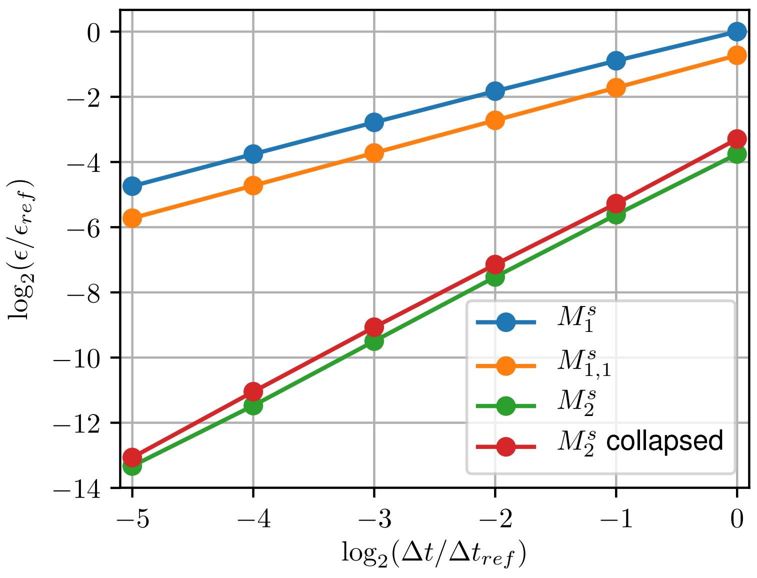


Figure 8.8.: Time order convergence for the $2D$ Euler gravity test case with $D2Q9$ model. Convergence is estimated using the relative L^2 error ϵ on macroscopic variables with respect to the analytical solution at $t_{max} \approx 0.12$. The reference values $(\Delta t_{ref}, \epsilon_{ref})$ of the logarithmic scale are $\Delta t_{ref} = 0.024$, $\epsilon_{ref} = 0.0143$.

The effect of the obstacle on the flow is modeled using a volumic source term of the form

$$\mathbf{s} = K(\mathbf{x})(\mathbf{w} - \mathbf{w}_s), \tag{8.28}$$

with $\mathbf{w}_s = [1.0, 0, 0]^t$ the target fluid state in the ``solid'' part of the domain and the relaxation frequency $K(\mathbf{x})$ is given by

$$K(\mathbf{x}) = K_s \exp(-\kappa(\mathbf{x} - \mathbf{x}_c)^2), \tag{8.29}$$

with $K_s = 300$, $\mathbf{x}_c = [-4, 0]^t$and $\kappa = 40$. The net effect is a very stiff relaxation towards a flow with zero velocity and the reference density near the center $\mathbf{x}_c$ of the frequency mask. The effective diameter of the cylinder for this simulation is about 0.5. The initial state, which is also applied at the duct boundaries for the whole simulation is $\rho = 1$, $u_x = 0.03$, $u_y = 0$. Accounting for the fact that for this model the sound speed is $1/\sqrt{3}$, the Mach number of the unperturbed flow is approximately $0, 017$. The simulation was performed on a macromesh with 16×16 macrocells stretched with a $1 : 5$ aspect ratio to match the domain dimension; each macrocell contains 12×60 integration points. On figure 8.9 we show the vorticity norm at $t = 83$, when turbulence is well developed in the wake of the obstacle.

8.5. Conclusion

In this paper, we have presented an optimized implementation of the Palindromic Discontinuous Galerkin Method for solving kinetic equations with stiff relaxation. The method presents the following interesting features:

- It can be used for solving any hyperbolic system of conservation laws.
- It is asymptotic-preserving with respect to the stiff relaxation.

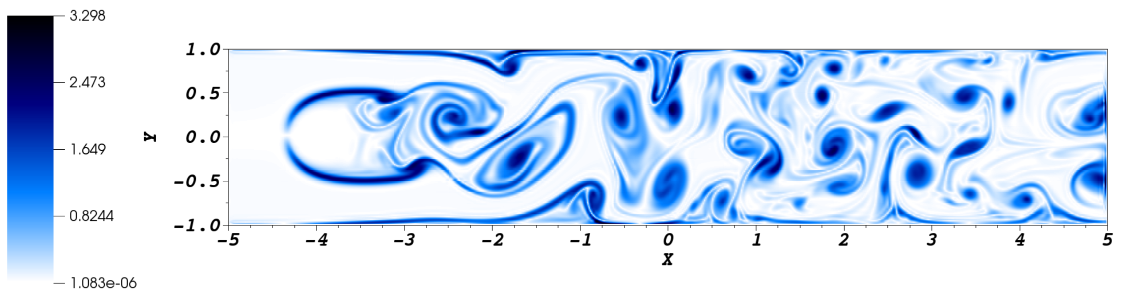


Figure 8.9.: . Flow around cylindrical obstacle. Vorticity norm $|\nabla \times u|$ at $t = 83$, showing the turbulent field behind the obstacle.

- It is implicit and thus is not limited by CFL conditions.
- Despite being formally implicit, it requires only explicit computations.
- It is easy to increase the time order with a composition method.
- It presents many opportunities for parallelization and optimization: in this paper we have presented the parallelization of the method with the aid of the MPI version of the StarPU runtime system. In this way we address both shared memory and distributed memory MIMD parallelism.

Our perspectives are now to apply the method for computing MHD instabilities in tokamaks. We will also try to extend the method to more general boundary conditions.

Bibliography

[Ala14] Frédéric Alauzet. “A changing-topology moving mesh technique for large displacements.” In: **Engineering with Computers** 30.2 (2014), pp. 175–200.

[AKR11] Boris Andreianov, Kenneth Hvistendahl Karlsen, and Nils Henrik Risebro. “A theory of L 1-dissipative solvers for scalar conservation laws with discontinuous flux.” In: **Archive for rational mechanics and analysis** 201 (2011), pp. 27–86.

[AN00] Denise Aregba-Driollet and Roberto Natalini. “Discrete kinetic schemes for multidimensional systems of conservation laws.” In: **SIAM Journal on Numerical Analysis** 37.6 (2000), pp. 1973–2004.

[AP05] Emmanuel Audusse and Benoît Perthame. “Uniqueness for scalar conservation laws with discontinuous flux via adapted entropies.” In: **Proceedings of the Royal Society of Edinburgh Section A: Mathematics** 135.2 (2005), pp. 253–265.

[Aug+12] Cédric Augonnet, Olivier Aumage, Nathalie Furmento, Raymond Namyst, and Samuel Thibault. “StarPU-MPI: Task programming over clusters of machines enhanced with accelerators.” In: **Recent Advances in the Message Passing Interface: 19th European MPI Users’ Group Meeting, EuroMPI 2012, Vienna, Austria, September 23-26, 2012. Proceedings 19**. Springer. 2012, pp. 298–299.

[Bad+18] Jayesh Badwaik, Matthieu Boileau, David Coulette, Emmanuel Franck, Philippe Helluy, Christian Klingenberg, Laura Mendoza, and Herbert Oberlin. “Task-based parallelization of an implicit kinetic scheme.” In: **ESAIM: Proceedings and Surveys** 63 (2018), pp. 60–77.

[BC18] Jayesh Badwaik and Praveen Chandrashekar. “Arbitrary Lagrangian--Eulerian Discontinuous Galerkin Method for 1D Euler Equations.” In: **Theory, Numerics and Applications of Hyperbolic Problems I: Aachen, Germany, August 2016**. Springer. 2018, pp. 323–334.

[BCK20] Jayesh Badwaik, Praveen Chandrashekar, and Christian Klingenberg. “Single-step arbitrary Lagrangian--Eulerian discontinuous Galerkin method for 1-d Euler equations.” In: **Communications on Applied Mathematics and Computation** 2 (2020), pp. 541–579.

[Bad+21] Jayesh Badwaik, Christian Klingenberg, Nils Henrik Risebro, and Adrian M Ruf. “Multilevel Monte Carlo finite volume methods for random conservation laws with discontinuous flux.” In: **ESAIM: Mathematical Modelling and Numerical Analysis** 55.3 (2021), pp. 1039–1065.

[BR20] Jayesh Badwaik and Adrian M Ruf. “Convergence rates of monotone schemes for conservation laws with discontinuous flux.” In: **SIAM Journal on Numerical Analysis** 58.1 (2020), pp. 607–629.

[BJ97] Paolo Baiti and Helge Kristian Jenssen. “Well-posedness for a class of 2×2 conservation laws with L^∞ data.” In: **Journal of Differential Equations** 140.1 (1997), pp. 161–185. ISSN: 0022-0396. DOI: 10.1006/jdeq.1997.3308.

[BA19] Nicolas Barral and Frédéric Alauzet. "Three-dimensional CFD simulations with large displacement of the geometries using a connectivity-change moving mesh approach." In: **Engineering with Computers** 35.2 (2019), pp. 397–422.

[BLL94] Joseph Baum, Hong Luo, and Rainald Loehner. "A new ALE adaptive unstructured methodology for the simulation of moving bodies." In: **32nd Aerospace Sciences Meeting and Exhibit**. 1994, p. 414.

[BK78] TB Belytschko and James M Kennedy. "Computer models for subassembly simulation." In: **Nuclear Engineering and Design** 49.1-2 (1978), pp. 17–38.

[Ben92] David J. Benson. "Computational methods in Lagrangian and Eulerian hydrocodes." In: **Computer Methods in Applied Mechanics and Engineering** 99.2-3 (1992), pp. 235–394. ISSN: 0045-7825. DOI: 10.1016/0045-7825(92)90042-I.

[Ber+11] Markus Berndt, JéRôMe Breil, StéPhane Galera, Milan Kucharik, Pierre-Henri Maire, and Mikhail Shashkov. "Two-step hybrid conservative remapping for multimaterial arbitrary Lagrangian--Eulerian methods." In: **Journal of Computational Physics** 230.17 (2011), pp. 6664–6687.

[BRS13] Pavel Bochev, Denis Ridzal, and Mikhail Shashkov. "Fast optimization-based conservative remap of scalar fields through aggregate mass transfer." In: **Journal of Computational Physics** 246 (2013), pp. 37–57.

[BGP00] François Bouchut, François Golse, and Mario Pulvirenti. **Kinetic equations and asymptotic theory**. Elsevier, 2000.

[Bre+13] Jérôme Breil, Thibault Harribey, Pierre-Henri Maire, and Mikhail Shashkov. "A multi-material ReALE method with MOF interface reconstruction." In: **Computers & Fluids** 83 (2013), pp. 115–125.

[Bür+03] R Bürger, KH Karlsen, C Klingenberg, and NH Risebro. "A front tracking approach to a model of continuous sedimentation in ideal clarifier--thickener units." In: **Nonlinear Analysis: Real World Applications** 4.3 (2003), pp. 457–481.

[Cia02] Philippe G Ciarlet. **The finite element method for elliptic problems**. SIAM, 2002.

[Com+10] Gaëtan Compere, Jean-François Remacle, Johan Jansson, and Johan Hoffman. "A mesh adaptation framework for dealing with large deforming meshes." In: **International Journal for Numerical Methods in Engineering** 82.7 (2010), pp. 843–867.

[Cou+16] David Coulette, Emmanuel Franck, Philippe Helluy, Michel Mehrenberger, and Laurent Navoret. "Palindromic discontinuous Galerkin method for kinetic equations with stiff relaxation." In: **arXiv preprint arXiv:1612.09422** (2016).

[CM80] Michael G Crandall and Andrew Majda. "Monotone difference approximations for scalar conservation laws." In: **Mathematics of Computation** 34.149 (1980), pp. 1–21.

[DP10] Timothy A Davis and Ekanathan Palamadai Natarajan. "Algorithm 907: KLU, a direct sparse solver for circuit simulation problems." In: **ACM Transactions on Mathematical Software (TOMS)** 37.3 (2010), pp. 1–17.

[Die96] Stefan Diehl. "A conservation law with point source and discontinuous flux function modelling continuous sedimentation." In: **SIAM Journal on Applied Mathematics** 56.2 (1996), pp. 388–419.

[DF08] Cécile Dobrzynski and Pascal Frey. “Anisotropic Delaunay mesh adaptation for unsteady simulations.” In: **Proceedings of the 17th international Meshing Roundtable**. Springer, 2008, pp. 177–194.

[DGH82] J. Donea, S. Giuliani, and J.P. Halleux. “An arbitrary lagrangian-eulerian finite element method for transient dynamic fluid-structure interactions.” In: **Computer Methods in Applied Mechanics and Engineering** 33.1 (1982), pp. 689–723. ISSN: 0045-7825. DOI: 10.1016/0045-7825(82)90128-1.

[FL64] R. M. Frank and R. B. Lazarus. “Mixed Lagrangian-Eulerian Method.” In: **Methods in Computational Physics. Advances in Research and Applications** (1964).

[Gab+19] Elena Gaburro, Walter Boscheri, Simone Chiocchetti, Christian Klingenberg, Volker Springel, and Michael Dumbser. “High order direct Arbitrary-Lagrangian-Eulerian schemes on moving Voronoi meshes with topology changes.” In: (May 2, 2019). arXiv: 1905.00967v1 [math.NA].

[GR09] Christophe Geuzaine and Jean-François Remacle. “Gmsh: A 3-D finite element mesh generator with built-in pre-and post-processing facilities.” In: **International Journal for Numerical Methods in Engineering** 79.11 (2009), pp. 1309–1331.

[GJT20] Shyam Sundar Ghoshal, Animesh Jana, and John D Towers. “Convergence of a Godunov scheme to an Audusse--Perthame adapted entropy solution for conservation laws with BV spatial flux.” In: **Numerische Mathematik** 146 (2020), pp. 629–659.

[GR91] Tore Gimse and Nils Henrik Risebro. “Riemann problems with a discontinuous flux function.” In: **Proceedings of Third International Conference on Hyperbolic Problems**. Vol. 1. 1991, pp. 488–502.

[GR92] Tore Gimse and Nils Henrik Risebro. “Solution of the Cauchy problem for a conservation law with a discontinuous flux function.” In: **SIAM Journal on Mathematical Analysis** 23.3 (1992), pp. 635–648.

[GR93] Tore Gimse and Nils Henrik Risebro. “A note on reservoir simulation for heterogeneous porous media.” In: **Transport in porous media** 10 (1993), pp. 257–270.

[Gra14] Benjamin Graille. “Approximation of mono-dimensional hyperbolic systems: A lattice Boltzmann scheme as a relaxation method.” In: **Journal of Computational Physics** 266 (2014), pp. 74–88.

[Hai+06] Ernst Hairer, Marlis Hochbruck, Arieh Iserles, and Christian Lubich. “Geometric numerical integration.” In: **Oberwolfach Reports** 3.1 (2006), pp. 805–882.

[Has+07] O Hassan, KA Sørensen, K Morgan, and NP Weatherill. “A method for time accurate turbulent compressible fluid flow simulation with moving boundary components employing local remeshing.” In: **International Journal for Numerical Methods in Fluids** 53.8 (2007), pp. 1243–1266.

[HAC74] C. W. Hirt, A. A. Amsden, and J. L. Cook. “An arbitrary Lagrangian-Eulerian computing method for all flow speeds.” In: **Journal of Computational Physics** (1974). ISSN: 10902716. DOI: 10.1016/0021-9991(74)90051-5.

[HR15] Helge Holden and Nils Henrik Risebro. **Front tracking for hyperbolic conservation laws**. Vol. 152. Springer, 2015.

[HLZ81] Thomas JR Hughes, Wing Kam Liu, and Thomas K Zimmermann. "Lagrangian-Eulerian finite element formulation for incompressible viscous flows." In: **Computer methods in applied mechanics and engineering** 29.3 (1981), pp. 329–349.

[KBS79] J.M. Kennedy, T. Belytschko, and D.F. Schoeberle. **Quasi-Eulerian formulation for fluid-structure interaction**. 1979.

[KR95] Christian Klingenberg and Nils Henrik Risebro. "Convex conservation laws with discontinuous coefficients. Existence, uniqueness and asymptotic behavior." In: **Communications in Partial Differential Equations** 20.11-12 (1995), pp. 1959–1990.

[Kol+13] U. Koley, N. H. Risebro, Ch. Schwab, and F. Weber. "Multilevel Monte Carlo for Random Degenerate Scalar Convection Diffusion Equation." In: (Nov. 7, 2013). arXiv: 1311.1752v1 [math.AP].

[KS12] Milan Kucharik and Mikhail Shashkov. "One-step hybrid remapping algorithm for multi-material arbitrary Lagrangian--Eulerian methods." In: **Journal of Computational Physics** 231.7 (2012), pp. 2851–2864.

[LeV02] Randall J LeVeque. **Finite volume methods for hyperbolic problems**. Vol. 31. Cambridge university press, 2002.

[LW55] Michael J Lighthill and Gerald Beresford Whitham. "On kinematic waves. II. A theory of traffic flow on long crowded roads." In: **Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences**. Vol. 229. 1178. The Royal Society. 1955, pp. 317–345.

[Lou05] Raphaël Loubére. "Validation test case suite for compressible hydrodynamics computation." In: **Unpublished notes, Los Alamos National Laboratory** (2005).

[MQ02] Robert I McLachlan and G Reinout W Quispel. "Splitting methods." In: **Acta Numerica** 11 (2002), pp. 341–434.

[MS12] Siddhartha Mishra and Ch Schwab. "Sparse tensor multi-level Monte Carlo finite volume methods for hyperbolic conservation laws with random initial data." In: **Mathematics of Computation** 81.280 (2012), pp. 1979–2018.

[Mou+15] Salli Moustafa, Ivan Dutka-Malen, Laurent Plagne, Angélique Ponçot, and Pierre Ramet. "Shared memory parallelism for 3D Cartesian discrete ordinates solver." In: **Annals of Nuclear Energy** 82 (2015), pp. 179–187.

[Noh63] W.F. Noh. "CEL: A Time-Dependent, Two-Space-Dimensional, Coupled Eulerian-Lagrange Code." In: **Meth. Comp. Phys.** 3 (Aug. 1963).

[PT18] Benedetto Piccoli and Magali Tournus. "A general BV existence result for conservation laws with spatial heterogeneities." In: **SIAM Journal on Mathematical Analysis** 50.3 (2018), pp. 2901–2927.

[RT91] Nils Henrik Risebro and Aslak Tveito. "Front tracking applied to a nonstrictly hyperbolic system of conservation laws." In: **SIAM Journal on Scientific and Statistical Computing** 12.6 (1991), pp. 1401–1419.

[Roe81] P. L. Roe. "Approximate Riemann solvers, parameter vectors, and difference schemes." In: **Journal of Computational Physics** 43.2 (1981), pp. 357–372. DOI: 10.1016/0021-9991(81)90128-5.

[Ruf24] Adrian M Ruf. "Flux-stability for conservation laws with discontinuous flux and convergence rates of the front tracking method." In: **IMA Journal of Numerical Analysis** 42.2 (2024), pp. 1116–1142.

[She17] Wen Shen. “On the uniqueness of vanishing viscosity solutions for Riemann problems for polymer flooding.” In: **Nonlinear Differential Equations and Applications NoDEA** 24 (2017), pp. 1–25.

[SO88] Chi-Wang Shu and Stanley Osher. “Efficient implementation of essentially non-oscillatory shock-capturing schemes.” In: **Journal of Computational Physics** 77.2 (1988), pp. 439–471. ISSN: 0021-9991. DOI: 10.1016/0021-9991(88)90177-5.

[Sod78] Gary A Sod. “A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws.” In: **Journal of computational physics** 27.1 (1978), pp. 1–31.

[Spr10] Volker Springel. “E pur si muove: Galilean-invariant cosmological hydrodynamical simulations on a moving mesh.” In: **Monthly Notices of the Royal Astronomical Society** 401.2 (2010), pp. 791–851.

[TT04] Vladimir A Titarev and Eleuterio F Toro. “Finite-volume WENO schemes for three-dimensional conservation laws.” In: **Journal of Computational Physics** 201.1 (2004), pp. 238–260.

[Tor09] E. F. Toro. **Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction**. Berlin Heidelberg: Springer, 2009. ISBN: 9783540498346.

[Tow00] John D Towers. “Convergence of a difference scheme for conservation laws with a discontinuous flux.” In: **SIAM Journal on Numerical Analysis** 38.2 (2000), pp. 681–698.

[Tow20] John D Towers. “An existence result for conservation laws having BV spatial flux heterogeneities-without concavity.” In: **Journal of Differential Equations** 269.7 (2020), pp. 5754–5764.

[Tru66] John G Trulio. **Theory and Structure of the AFTON Codes**. Tech. rep. NORTRONICS NEWBURY PARK CA, 1966.

[VR50] John VonNeumann and Robert D Richtmyer. “A method for the numerical calculation of hydrodynamic shocks.” In: **Journal of applied physics** 21.3 (1950), pp. 232–237.

[Wil63] Mark L Wilkins. **Calculation of elastic-plastic flow**. Tech. rep. California Univ Livermore Radiation Lab, 1963.

[Yan+13] Yury V Yanilkin, Evgeny A Goncharov, Vadim Yu Kolobyanin, Vitaly V Sadchikov, James R Kamm, Mikhail J Shashkov, and William J Rider. “Multi-material pressure relaxation methods for Lagrangian hydrodynamics.” In: **Computers & Fluids** 83 (2013), pp. 137–143.

[ZS10] Xiangxiong Zhang and Chi-Wang Shu. “On maximum-principle-satisfying high order schemes for scalar conservation laws.” In: **Journal of Computational Physics** 229.9 (2010), pp. 3091–3120.

[ZS13] Xinghui Zhong and Chi-Wang Shu. “A simple weighted essentially nonoscillatory limiter for Runge--Kutta discontinuous Galerkin methods.” In: **Journal of Computational Physics** 232.1 (2013), pp. 397–415.

[ZQ16] Jun Zhu and Jianxian Qiu. “A new fifth order finite difference WENO scheme for solving hyperbolic conservation laws.” In: **Journal of Computational Physics** 318 (2016), pp. 110–121.

[Zhu+13] Jun Zhu, Xinghui Zhong, Chi-Wang Shu, and Jianxian Qiu. “Runge--Kutta discontinuous Galerkin method using a new type of WENO limiters on unstructured meshes.” In: **Journal of Computational Physics** 248 (2013), pp. 200–220.

Part IV.

Publications

Convergence Rates for Conservation Laws with Discontinuous Flux

SIAM J. NUMER. ANAL.
Vol. 58, No. 1, pp. 607–629

CONVERGENCE RATES OF MONOTONE SCHEMES FOR CONSERVATION LAWS WITH DISCONTINUOUS FLUX*

JAYESH BADWAIK† AND ADRIAN M. RUF‡

Abstract. We prove that a class of monotone finite volume schemes for scalar conservation laws with discontinuous flux converge at a rate of $\sqrt{\Delta x}$ in L^1, whenever the flux is strictly monotone in u and the spatial dependency of the flux is piecewise constant with finitely many discontinuities. We also present numerical experiments to illustrate the main result. To the best of our knowledge, this is the first proof of any type of convergence rate for numerical methods for conservation laws with discontinuous, nonlinear flux. Our proof relies on convergence rates for conservation laws with initial and boundary value data. Since those are not readily available in the literature we establish convergence rates in that case en passant in the appendix.

Key words. hyperbolic conservation laws, discontinuous flux, numerical methods, convergence rate

AMS subject classifications. 35L65, 65M08, 65M12, 35R05

DOI. 10.1137/19M1283276

1. Introduction.

We prove a convergence rate for a class of monotone, upwind-type finite volume schemes for scalar conservation laws with *discontinuous* flux of the form

$$\begin{aligned} u_t + f(k(x),u)_x &= 0, \quad (x,t) \in \mathbb{R} \times (0,T), \\ u(x,0) &= u_0(x), \quad x \in \mathbb{R}. \end{aligned} \tag{1.1}$$

Here, we assume that the flux f is *strictly monotone* in u and has a *discontinuous* spatial dependency through the coefficient k which is *piecewise constant* with finitely many discontinuities.

MAIN THEOREM. *Let f be strictly monotone in u in the sense that $f_u > 0$, k piecewise constant with finitely many discontinuities, and $u_0 \in (\mathrm{L}^1 \cap \mathrm{BV})(\mathbb{R})$. Then all monotone finite volume methods with the upwind property which obey the discrete Rankine–Hugoniot condition across the discontinuities of k converge at a rate of $\sqrt{\Delta x}$ to the unique entropy solution of the conservation law* (1.1).

The full theorem is stated in section 5. Our proof uses the Rankine–Hugoniot condition at the discontinuities of k to break down the problem into finitely many initial-boundary value problems for each of which we will prove a convergence rate using the classical "doubling of variables" technique.

1.1. Background on conservation laws with discontinuous fluxes.

Problem (1.1) is of great practical interest in several areas of physics and engineering. In

*Received by the editors August 26, 2019; accepted for publication (in revised form) December 9, 2019; published electronically February 6, 2020.
https://doi.org/10.1137/19M1283276

Funding: The work of the authors was supported by the European Union's Framework Programme for Research and Innovation Horizon 2020 (2014-2020) under the Marie Skłodowska-Curie grant 642768. The work of the first author was also supported by the Priority Programme 1648 of the Deutsche Forschungsgemeinschaft (DFG). The work of the second author was also supported by the NFR-DAAD grant 281268.

†Department of Mathematics, University of Würzburg, Germany (badwaik.jayesh@gmail.com).
‡Seminar for Applied Mathematics, ETH Zürich, Switzerland (adrian.ruf@sam.math.ethz.ch).

particular, it arises in modeling traffic flow on highways with changing road conditions (see [25]), in the modeling of two-phase flow in a porous medium (see [13, 32]), and in modeling sedimentation processes (see [11, 7]).

The flux in (1.1) depends on the space variable through a coefficient k which may be discontinuous. The dependence can, for example, be of the additive type, i.e., $f(k(x), u) = f(u) - k(x)$ (see [14]), or of the multiplicative type, i.e., $f(k(x), u) = k(x)f(u)$ which is more common (see, e.g., [37]). However, for the sake of generality we do not assume any particular algebraic structure of the flux $f(k(x), u)$ here. The case we consider in this paper where k is piecewise constant with finitely many discontinuities corresponds to switching from one u-dependent flux function, $f^{(i-1)}$, to another, $f^{(i)}$, across a discontinuity ξ_i of k. The case when k has just one discontinuity—the so-called two flux case—given by

$$u_t + (H(x)f(u) + (1 - H(x))g(u))_x = 0 \tag{1.2}$$

where H is the Heaviside function was studied in a series of papers by Adimurthi, Mishra, and Gowda (see [27, 2, 1] and references therein). Most notably, in [2], the authors showed existence of infinitely many L^1-stable semigroups of solutions to (1.2). We remark that, because of the assumption that k is piecewise constant, the convergence rate of monotone schemes for (1.2) will be the building block for the general case of (1.1).

Equations of type (1.1) have been dealt with extensively in the literature from a purely academic point of view as well as with a specific application in mind. In [12, 13], Gimse and Risebro calculated solutions for the Riemann problem assuming convexity of the flux in u and used the solutions to show existence of a weak solution for the general Cauchy problem with a front tracking algorithm. Other results based on the front tracking algorithm were obtained in [20, 19, 6, 21, 7], and in [9] with a time-dependent discontinuous coefficient. Out of the aforementioned results, we want to highlight [6] from Baiti and Jenssen who proved existence and uniqueness of entropy solutions in the case that the flux is strictly monotone in u which is the case we consider in this paper as well.

The first results for finite volume schemes for (1.1) (assuming a multiplicative spatial dependency) were obtained by Towers in [37, 38]. Specifically, in [37], the author developed staggered versions of the Godunov and Engquist–Osher schemes for the case where f is convex in u and k is strictly positive. In [38] similar results were proved for the case of non-convex fluxes. In [16], Karlsen, Risebro, and Towers studied (1.1) with an added degenerate parabolic term using an Engquist–Osher-type scheme and in [17] the authors proved existence of the vanishing viscosity limit using compensated compactness. In [18], Karlsen and Towers showed convergence of the Lax–Friedrichs scheme for (1.1) (with a time-dependent discontinuous coefficient). They were able to handle very general fluxes and sign-changing coefficients by using compensated compactness.

A general framework for well-posedness of (1.2) was proposed by Andreianov, Karlsen, and Risebro in [3].

Lastly, we want to point out that the monotonicity assumption, $f_u > 0$, we use in this paper implies that the equivalent system

$$\begin{aligned} u_t + f(k, u)_x &= 0, \\ k_t &= 0 \end{aligned}$$

is hyperbolic and not resonant; see [37, 38, 19, 20, 21].

1.2. Background on convergence rates. When dealing with numerical methods for (1.1), where an approximate solution $u_{\Delta t}$ depends on a grid discretization parameter Δx, having a provable bound of the type

$$\|u(T) - u_{\Delta t}(T)\|_{\mathrm{L}^1(\mathbb{R})} \leq C\Delta x^r$$

—specifying how fast the numerical scheme converges—is highly desirable. Specifically, convergence rates can be used for a posteriori error based mesh adaptation [39] and optimal design of multilevel Monte Carlo methods [5]. To this date the only result concerning convergence rates of finite volume schemes for conservation laws with discontinuous flux is due to Wen and Jin and pertains the most basic case of the linear advection equation with piecewise constant wave speed that changes across a single discontinuity [40]. So far, in the nonlinear case, convergence rates of finite volume schemes for (1.1) are only available in the absence of a spatial dependency, i.e., k being constant. The main difficulty in obtaining convergence rates when the flux has a discontinuous spatial dependency is that in this case the classical "doubling of variables" technique (see [22]) involves both terms with $k(x)$ and terms with $k(y)$.

In the case of a spatially independent flux the seminal paper by Kuznetsov [23] shows that monotone schemes converge towards the entropy solution of (1.1) without spatial dependency at a rate of $\mathcal{O}(\sqrt{\Delta x})$ in L^1. This rate was proved for initial data in $(\mathrm{L}^1 \cup \mathrm{BV})(\mathbb{R})$, and in this generality the rate $\mathcal{O}(\sqrt{\Delta x})$ is in fact optimal, as was shown by Şabac in [34]. There are certain classes of initial data for which higher orders of convergence for monotone schemes have been shown; e.g., Teng and Zhang [36] showed a convergence rate of $\mathcal{O}(\Delta x)$ for the case of piecewise constant initial data. See also [33] for a more comprehensive overview of convergence rate results for (1.1) without spatial dependency.

An alternative approach to convergence rates in the case where the flux only depends on u was initiated by Nessyahu, Tadmor, and Tassa [28, 29]. The authors used the Wasserstein distance instead of the L^1 norm and were able to show that a large class of monotone schemes converge at a rate of $\mathcal{O}(\Delta x)$ in the Wasserstein distance for Lip^+-bounded, compactly supported initial data. This rate was recently proved to be optimal by Ruf, Sande, and Solem [33].

Since the proof of our main theorem makes use of convergence rates for conservation laws on bounded domains, it is worth mentioning that Ohlberger and Vovelle claimed a convergence rate of $\mathcal{O}(\Delta x^{1/3})$ for conservation laws with initial and boundary data in one dimension [30, page 135]. In our specific case of a strictly monotone flux, however, we are able to prove a better rate of $\mathcal{O}(\sqrt{\Delta x})$.

1.3. Outline of the paper. We have organized the paper in the following way. In section 2, we will define entropy solutions of (1.1) and show that—when restricted to a subdomain between two neighboring discontinuities of k—they are entropy solutions of a certain initial boundary value problem with spatially independent flux. Here the respective boundary datum is given through the Rankine–Hugoniot condition across a discontinuity of k. In section 3, we describe our finite volume scheme and show that we can establish a convergence rate of our numerical method for (1.1) by proving a convergence rate for each of those initial-boundary value problems. In section 4, we start by considering just one discontinuity of k, i.e., (1.2), and proving a convergence rate on $\mathbb{R}^-$ and $\mathbb{R}^+$ separately. Section 5 contains the statement and proof of the main result where we use the translation invariance of conservation laws and the results of the previous section in our main proof. Section 6 describes numerical experiments that illustrate our convergence rate result as well as the class of fluxes that is covered by our theory. In section 7, we summarize the findings of this paper and provide an outlook. Lastly, in Appendix A, we show that—with minimal changes—our results can be applied to general initial-boundary value problems where the prescribed boundary datum is arbitrary.

2. Preliminaries.

Throughout this paper, we will assume that the initial datum u_0 is integrable, bounded, and of finite total variation, i.e., $u_0 \in (\mathrm{L}^1 \cap \mathrm{BV})(\mathbb{R})$, and that f is strictly monotone in u, i.e., $f_u \geq \alpha > 0$. Further, we will denote the discontinuities of k as $\xi_1, \ldots, \xi_N$ and the interval between two adjacent discontinuities as $D_i = (\xi_i, \xi_{i+1})$, $i = 0, \ldots, N$. Here, we used the notation $\xi_0 = -\infty$ and $\xi_{N+1} = \infty$. Then we can write

$$f(k(x), \cdot) =: f^{(i)}(\cdot) \qquad \text{for } x \in D_i.$$

We will consider entropy solutions of (1.1) in the following sense.

Definition 2.1 (entropy solution). *We say $u \in \mathcal{C}([0,T]; \mathrm{L}^1(\mathbb{R})) \cap \mathrm{L}^\infty((0,T) \times \mathbb{R})$ is an entropy solution of (1.1) if for all $c \in \mathbb{R}$*

$$\begin{aligned}\sum_{i=0}^{N} \bigg(\int_0^T \int_{D_i} & \Big(|u - c_i| \varphi_t + \operatorname{sign}(u - c_i) \left(f^{(i)}(u) - f^{(i)}(c_i) \right) \varphi_x \Big) \,\mathrm{d}x \,\mathrm{d}t \\ & - \int_{D_i} |u(x,T) - c_i| \varphi(x,T) \,\mathrm{d}x + \int_{D_i} |u_0(x) - c_i| \varphi(x,0)) \,\mathrm{d}x \\ & - \int_0^T \operatorname{sign}(u(\xi_{i+1}-,t) - c_i) \left(f^{(i)}(u(\xi_{i+1}-,t)) - f^{(i)}(c_i) \right) \varphi(\xi_{i+1},t) \,\mathrm{d}t \\ & + \int_0^T \operatorname{sign}(u(\xi_i+,t) - c_i) \left(f^{(i)}(u(\xi_i+,t)) - f^{(i)}(c_i) \right) \varphi(\xi_i,t) \,\mathrm{d}t \bigg) \geq 0\end{aligned}$$

for all nonnegative $\varphi \in \mathcal{C}^\infty(\mathbb{R} \times [0,T])$. Here, the c_i are given by $c_0 := c$ and

$$c_{i+1} = \left(f^{(i+1)} \right)^{-1} \left(f^{(i)}(c_i) \right) \qquad \text{for } i = 1, \ldots, N. \tag{2.1}$$

Remark 2.2. Note that, due to the monotonicity of the fluxes $f^{(i)}$, the inverse of $f^{(i)}$ used in (2.1) and throughout this paper exists.

Remark 2.3. Note that existence and uniqueness of entropy solutions of (1.1) are guaranteed by the theory developed by Baiti and Jenssen in [6] using adapted entropies in the sense above (cf. also [4] where adapted entropies are used as well). In particular, the traces in Definition 2.1 are well defined (cf. [3, Remark 2.3])

Remark 2.4. Like for conservation laws without (discontinuous) spatial dependency of the flux, a Rankine–Hugoniot-type argument shows that weak solutions of (1.1) necessarily satisfy the Rankine–Hugoniot condition across all discontinuities ξ_i, i.e.,

$$f^{(i-1)}(u(\xi_i-,t)) = f^{(i)}(u(\xi_i+,t)). \tag{2.2}$$

The following observation is at the heart of the proof of the main result. The entropy solution u of (1.1) can be decomposed as $u = \sum_{i=0}^{N} u^{(i)}$, where $u^{(i)} := u\mathbb{1}_{D_i \times [0,T]}$ such that $u^{(0)}$ solves

$$\begin{aligned} u_t^{(0)} + f^{(0)}\left(u^{(0)}\right)_x &= 0, \quad (x,t) \in D_0 \times (0,T), \\ u^{(0)}(x,0) &= u_0(x), \quad x \in D_0 \end{aligned} \tag{2.3}$$

and $u^{(i)}$ solves

$$\begin{aligned} u_t^{(i)} + f^{(i)}\left(u^{(i)}\right)_x &= 0, \quad (x,t) \in D_i \times (0,T), \\ u^{(i)}(x,0) &= u_0(x), \quad x \in D_i, \\ u^{(i)}(\xi_i+,t) = \left(f^{(i)}\right)^{-1}\left(f^{(i-1)}(u^{(i-1)}(\xi_i-,t))\right), &\quad t \in (0,T) \end{aligned} \tag{2.4}$$

for $i = 1, \dots, N$ (cf. Definitions 4.1 and 4.5 below). Note that the boundary condition on the domain D_i, $i = 1, \dots, N$, given by the last line of (2.4) reflects the Rankine–Hugoniot condition (2.2).

Conversely, if $u^{(0)}$ is the entropy solution of (2.3) on D_0 and $u^{(i)}$ is the entropy solution of (2.4) on D_i for $i = 1, \dots, N$, then the composite function $u := \sum_{i=0}^N u^{(i)}$ is the entropy solution of (1.1) in the sense of Definition 2.1. This can be seen by adding the entropy inequalities of $u^{(i)}$ and choosing the respective constant in each entropy inequality in accordance with (2.1).

In the remainder of the paper, we will construct a numerical scheme which satisfies the Rankine–Hugoniot condition (2.2) (or, equivalently, the last line of (2.4)) across the discontinuities of k on the discrete level. This scheme, when restricted to the subdomain D_i, will converge towards the entropy solution on D_i. The discrete Rankine–Hugoniot condition will then allow us to break down the problem of finding a convergence rate on the whole real line to finding convergence rates on each of the subdomains D_i.

3. The numerical scheme.

We discretize the domain $\mathbb{R} \times [0,T]$ using the spatial and temporal grid discretization parameters Δx and Δt. The resulting grid cells then are $\mathcal{C}_j = (x_{j-1/2}, x_{j+1/2})$ and $\mathcal{C}^n = (t^n, t^{n+1})$ for points $x_{j+1/2}$ such that $x_{j+1/2} - x_{j-1/2} = \Delta x$, $j \in \mathbb{Z}$, and $t^n = n\Delta t$ for $n = 0, \dots, M+1$. Note that $T = (M+1)\Delta t$. Further we write $\mathcal{C}_j^n$ to denote the rectangle $\mathcal{C}_j \times \mathcal{C}^n$.

In the following we will assume that the grid is aligned in such a way that all discontinuities of k lie on cell interfaces, i.e., $\xi_i = x_{P_i - 1/2}$ for some integers P_i, $i = 1, \dots, N$. In general this can easily be achieved by considering a globally nonuniform grid that is uniform on each D_i and then taking $\Delta x = \max_{i=0,\dots,N} \Delta x_i$ where Δx_i is the grid discretization parameter in D_i. For simplicity, however, here we will assume that the grid is uniform on the whole real line.

Further, we will consider two-point numerical fluxes $F(u,v)$ that have the upwind property such that if $f' \geq 0$, then $F(u,v) = f(v)$. Such fluxes include the upwind flux, the Godunov flux, and the Engquist–Osher flux. Thus, the numerical scheme we will analyze is the following:

$$\begin{aligned} u_j^{n+1} &= u_j^n - \lambda\left(f^{(i)}(u_j^n) - f^{(i)}\left(u_{j-1}^n\right)\right), \quad n \geq 0,\ P_i < j < P_{i+1},\ 0 \leq i \leq N, \\ u_j^0 &= \frac{1}{\Delta x}\int_{\mathcal{C}_j} u_0(x)\,\mathrm{d}x, \quad j \in \mathbb{Z}, \\ u_{P_i}^{n+1} &= \left(f^{(i)}\right)^{-1}\left(f^{(i-1)}\left(u_{P_i-1}^{n+1}\right)\right), \quad n \geq 0,\ 0 < i \leq N, \end{aligned} \tag{3.1}$$

where $\lambda = \Delta t/\Delta x$. We assume that the grid discretization parameters satisfy the CFL condition

$$\max_i \max_u \left(f^{(i)}\right)'(u)\lambda \leq 1. \tag{3.2}$$

Note that the last line of (3.1) represents a discrete version of the Rankine–Hugoniot condition (2.2). Here, we use the ghost cells $\mathcal{C}_{P_i}$, $i = 1, \ldots, N$ to explicitly enforce the Rankine–Hugoniot condition on the discrete level. While this makes the numerical scheme (3.1) nonconservative, the convergence result in this paper, coupled with the fact that the limit is conservative, shows that the contribution of the nonconservative part of the scheme vanishes in the limit.

To get a convergence rate of the numerical scheme (3.1) we decompose the entropy solution u as $u = \sum_{i=0}^{N} u^{(i)}$ where $u^{(i)}$, $i = 0, \ldots, N$, are the respective entropy solutions on D_i and the numerical solution $u_{\Delta t}$ as $\sum_{i=0}^{N} u_{\Delta t}^{(i)}$ where

$$u_{\Delta t}^{(i)}(x,t) = \begin{cases} u_j^n & \text{if } (x,t) \in \mathcal{C}_j^n \subset D_i \times \mathcal{C}^n, \\ 0 & \text{otherwise.} \end{cases}$$

Then we have

$$\|u(T) - u_{\Delta t}(T)\|_{\mathrm{L}^1(\mathbb{R})} = \sum_{i=0}^{N} \left\| u^{(i)}(T) - u_{\Delta t}^{(i)}(T) \right\|_{\mathrm{L}^1(D_i)},$$

and the problem of finding a convergence rate for $u_{\Delta t}$ can be broken down to finding convergence rates for $u_{\Delta t}^{(i)}$ on each of the subdomains D_i. In the following sections, we will show that

$$\left\| u^{(i)}(T) - u_{\Delta t}^{(i)}(T) \right\|_{\mathrm{L}^1(D_i)} \le C\sqrt{\Delta x}.$$

Note that convergence of the numerical scheme (3.1) towards the entropy solution of (1.1) follows from our convergence rate estimate. At this point we want to point out that instead of assuming $f_u > 0$ our proof can readily be adapted for the case $f_u < 0$.

4. Convergence rates for fluxes with one discontinuity.

We will first consider the case where k has just two constant values separated by a discontinuity ξ_1, and for ease of notation we will assume that $\xi_1 = 0$. Further, we will denote the flux left of ξ_1 as g and right of ξ_1 as f. In order to get a convergence rate for problem (1.1) we will derive convergence rates on $D_0 = \mathbb{R}^-$, on $D_1 = \mathbb{R}^+$, and on $(0, L)$ for $L > 0$.

4.1. Convergence rate estimates on $\mathbb{R}^-$.

As a first step we consider the initial value problem

$$\begin{aligned} u_t + g(u)_x &= 0, \quad (x,t) \in \mathbb{R}^- \times (0,T), \\ u(x,0) &= u_0(x), \quad x \in \mathbb{R}^- \end{aligned} \tag{4.1}$$

on $\mathbb{R}^-$ with the flux g being strictly monotone and consider entropy solutions in the following sense.

Definition 4.1 (entropy solution on $\mathbb{R}^-$). *We say $u \in \mathcal{C}([0,T]; \mathrm{L}^1(\mathbb{R}^-)) \cap \mathrm{L}^\infty((0,T) \times \mathbb{R}^-)$ is an entropy solution of* (4.1) *if for all $c \in \mathbb{R}$,*

$$\int_0^T \int_{\mathbb{R}^-} (|u - c|\varphi_t + |g(u) - g(c)|\varphi_x) \,\mathrm{d}x\,\mathrm{d}t - \int_{\mathbb{R}^-} |u(x,T) - c|\varphi(x,T)\,\mathrm{d}x$$
$$+ \int_{\mathbb{R}^-} |u_0(x) - c|\varphi(x,0))\,\mathrm{d}x - \int_0^T |g(u(0-,t)) - g(c)|\varphi(0,t)\,\mathrm{d}t \ge 0$$

for all nonnegative $\varphi \in \mathcal{C}^\infty((-\infty, 0] \times [0,T])$.

Note that here $u(0-,t)$ denotes the limit of $u(x,t)$ as $x \to 0$ from the left.

As before, we will write $\mathcal{C}_j = (x_{j-1/2}, x_{j+1/2})$, $j \in \mathbb{Z}$, where now $x_{-1/2} := 0$. Our numerical scheme then reads

$$\begin{aligned} u_j^{n+1} &= u_j^n - \lambda\left(g(u_j^n) - g\left(u_{j-1}^n\right)\right), \quad j<0,\ n\geq 0,\\ u_j^0 &= \frac{1}{\Delta x}\int_{\mathcal{C}_j} u_0(x)\,\mathrm{d}x, \quad j<0, \end{aligned} \tag{4.2}$$

where $\lambda = \Delta t/\Delta x$ satisfies the CFL condition (3.2).

We note that the numerical scheme satisfies a discrete entropy inequality away from the spatial boundary

$$D_+^t \eta_j^n + D_- q_j^n \leq 0, \qquad n \geq 1,\ j<0 \tag{4.3}$$

which can be seen by adopting the classical Crandall–Majda arguments in [10, Proposition 4.1] for $j<0$. Here, $\eta_j^n = \eta(u_j^n, c) = |u_j^n - c|$, $q_j^n = q(u_j^n, c) = \operatorname{sign}(u_j^n - c)$ $(g(u_j^n) - g(c)) = |g(u_j^n) - g(c)|$, and

$$D_+^t a^n = \frac{a^{n+1} - a^n}{\Delta t} \qquad \text{and} \qquad D_- a_j = \frac{a_j - a_{j-1}}{\Delta x}$$

denote standard difference operators.

In order to derive convergence rates we will develop a Kuznetsov-type lemma in the following. For any function $u \in \mathcal{C}([0,T];\mathrm{L}^1(\mathbb{R}^-))$ we define

$$\begin{aligned} L(u,c,\varphi) = &\int_0^T\int_{\mathbb{R}^-} \left(|u-c|\varphi_t + q(u,c)\varphi_x\right)\mathrm{d}x\,\mathrm{d}t - \int_{\mathbb{R}^-} |u(x,T) - c|\varphi(x,T)\,\mathrm{d}x\\ &+ \int_{\mathbb{R}^-} |u_0(x) - c|\varphi(x,0)\,\mathrm{d}x - \int_0^T q(u(0-,t),c)\varphi(0,t)\,\mathrm{d}t, \end{aligned}$$

where $q(u,c) = |g(u) - g(c)|$ is the Kružkov entropy flux. Note that if u is an entropy solution of (4.1), then $L(u,c,\varphi) \geq 0$ for all $c \in \mathbb{R}$ and test functions $\varphi \geq 0$. We now take $c = v(y,s)$ and the test function

$$\varphi(x,t,y,s) = \omega_\varepsilon(x-y)\omega_{\varepsilon_0}(t-s),$$

where $\omega_\varepsilon, \omega_{\varepsilon_0}$ are standard symmetric mollifiers for $\varepsilon, \varepsilon_0 > 0$. Note that $\varphi_t = -\varphi_s$, $\varphi_x = -\varphi_y$, and

$$\varphi(x,t,y,s) = \varphi(y,t,x,s) = \varphi(y,s,x,t) = \varphi(x,s,y,t) \tag{4.4}$$

as well as

$$\begin{aligned} &\int_{\mathbb{R}} \omega_\varepsilon(x-y)\,\mathrm{d}y \leq 1, &&\int_{\mathbb{R}} |\omega_\varepsilon'(x-y)|\,\mathrm{d}y \leq \frac{C}{\varepsilon},\\ &\int_0^T \omega_{\varepsilon_0}(t-s)\,\mathrm{d}s \leq 1, &&\int_0^T |\omega_{\varepsilon_0}'(t-s)|\,\mathrm{d}s \leq \frac{C}{\varepsilon_0} \end{aligned} \tag{4.5}$$

for all $x \in \mathbb{R}$, $t \in [0,T]$. Let now

$$\Lambda_{\varepsilon,\varepsilon_0}(u,v) = \int_0^T\int_{\mathbb{R}^-} L(u, v(y,s), \varphi(\cdot,\cdot,y,s))\,\mathrm{d}y\,\mathrm{d}s. \tag{4.6}$$

For functions $w \in \mathcal{C}([0,T];\mathrm{L}^1(\mathbb{R}^-))$, we further define the moduli of continuity

$$\nu_t(w,\varepsilon_0) = \sup_{|\sigma|\le\varepsilon_0} \|w(\cdot,t+\sigma) - w(\cdot,t)\|_{\mathrm{L}^1(\mathbb{R}^-)},$$
$$\mu(w(\cdot,t),\varepsilon) = \sup_{|z|\le\varepsilon} \|w(\cdot+z,t) - w(\cdot,t)\|_{\mathrm{L}^1(\mathbb{R}^-)}.$$

LEMMA 4.2 (Kuznetsov-type lemma). *Let u be the entropy solution of* (4.1). *Then, for any function $v : [0,T] \to (\mathrm{L}^1 \cap \mathrm{BV})(\mathbb{R}^-)$ such that the one-sided limits $v(t\pm)$ exist in L^1, we have*

$$\begin{aligned}
&\|u(\cdot,T) - v(\cdot,T)\|_{\mathrm{L}^1(\mathbb{R}^-)} \\
&\quad + \int_0^T \int_{\mathbb{R}^-} \int_0^T \left(q(u(0-,t),v(y,s)) + q(v(0-,t),u(y,s))\right) \varphi(0,t,y,s)\,\mathrm{d}t\,\mathrm{d}y\,\mathrm{d}s \\
&\le \|u_0 - v(\cdot,0)\|_{\mathrm{L}^1(\mathbb{R}^-)} - \Lambda_{\varepsilon,\varepsilon_0}(v,u) \\
&\quad + C\Big(\varepsilon + \varepsilon_0 + \nu_T(v,\varepsilon_0) + \nu_0(v,\varepsilon_0) + \mu(v(\cdot,T),\varepsilon)) + \mu(v(\cdot,0),\varepsilon))\Big)
\end{aligned}$$

for some constant C independent of ε and ε_0.

Proof. Using that $\varphi_t = -\varphi_s$, $\varphi_x = -\varphi_y$ and the symmetry relations (4.4) we get

$$\begin{aligned}
&\Lambda_{\varepsilon,\varepsilon_0}(u,v) \\
&\quad = -\Lambda_{\varepsilon,\varepsilon_0}(v,u) \\
&\qquad \underbrace{- \int_0^T \int_{\mathbb{R}^-} \int_{\mathbb{R}^-} \left(|u(x,T) - v(y,s)| + |v(x,T) - u(y,s)|\right) \varphi(x,T,y,s)\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}s}_{=:\mathbf{A}} \\
&\qquad \underbrace{+ \int_0^T \int_{\mathbb{R}^-} \int_{\mathbb{R}^-} \left(|u_0(x,t) - v(y,s)| + |v(x,0) - u(y,s)|\right) \varphi(x,0,y,s)\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}s}_{=:\mathbf{B}} \\
&\qquad \underbrace{- \int_0^T \int_{\mathbb{R}^-} \int_0^T \left(q(u(0-,t),v(y,s)) + q(v(0-,t),u(y,s))\right) \varphi(0,t,y,s)\,\mathrm{d}t\,\mathrm{d}y\,\mathrm{d}s}_{=:\mathbf{C}}.
\end{aligned}$$

Since u is an entropy solution we find

$$0 \le \Lambda_{\varepsilon,\varepsilon_0}(u,v) = -\Lambda_{\varepsilon,\varepsilon_0}(v,u) - \mathbf{A} + \mathbf{B} - \mathbf{C}$$

and thus

$$\mathbf{A} + \mathbf{C} \le -\Lambda_{\varepsilon,\varepsilon_0}(v,u) + \mathbf{B}.$$

The terms $\mathbf{A}$ and $\mathbf{B}$ also appear in the case of an unbounded spatial domain and can be estimated by

$$\begin{aligned}
\mathbf{A} \ge{}& \|u(\cdot,T) - v(\cdot,T)\|_{\mathrm{L}^1(\mathbb{R}^-)} \\
&- \frac{1}{2}\left(\nu_T(u,\varepsilon_0) + \mu(u(\cdot,T),\varepsilon) + \nu_T(v,\varepsilon_0) + \mu(v(\cdot,T),\varepsilon)\right)
\end{aligned}$$

and

$$\mathbf{B} \le \|u_0 - v(\cdot,0)\|_{\mathrm{L}^1(\mathbb{R}^-)} + \frac{1}{2}\left(\nu_0(u,\varepsilon_0) + \mu(u_0,\varepsilon) + \nu_0(v,\varepsilon_0) + \mu(v(\cdot,0),\varepsilon)\right);$$

see [8] or [15] for details. Lastly, due to the Lipschitz continuity in time and the TVD property (see [15, Theorem 2.15] and [15, Lemma A.1]) the entropy solution of (4.1) satisfies

$$\nu_0(u,\varepsilon_0),\ \nu_T(u,\varepsilon_0) \le C\,\mathrm{TV}(u_0)\varepsilon_0$$
$$\text{and} \qquad \mu(u_0,\varepsilon),\ \mu(u(\cdot,T),\varepsilon) \le \mathrm{TV}(u_0)\varepsilon$$

which completes the proof. ∎

In order to derive a convergence rate the next step is to estimate the term $\Lambda_{\varepsilon,\varepsilon_0}(u_{\Delta t},u)$.

LEMMA 4.3. *The estimate*

$$-\Lambda_{\varepsilon,\varepsilon_0}(u_{\Delta t},u) \le C\left(\Delta x + \frac{\Delta x}{\varepsilon} + \frac{\Delta t}{\varepsilon_0}\right)$$

holds for some constant C independent of $\Delta x, \Delta t, \varepsilon$, and ε_0.

Proof. The proof of Lemma 4.3 for conservation laws on the real line can be found, e.g., in [15]. Here, we only need to replace any sum of the form $\sum_{j=-\infty}^{\infty}$ by $\sum_{-\infty}^{-1}$ and note that the boundary term in space cancels after integration by parts. See also the proof of Lemma 4.9 in section 4.2 for details. ∎

THEOREM 4.4 (convergence rate on $\mathbb{R}^-$). *Let u be the entropy solution of the initial-boundary value problem* (4.1) *and $u_{\Delta t}$ the numerical approximation given by* (4.2) *where we take λ constant. Then we have the following convergence rate estimate:*

$$\|u(\cdot,T) - u_{\Delta t}(\cdot,T)\|_{\mathrm{L}^1(\mathbb{R}^-)} \le C\sqrt{\Delta x}$$

for some constant C independent of Δx.

Proof. The numerical solution $u_{\Delta t}$ is Lipschitz continuous in time and TVD and therefore satisfies

$$\nu_0(u_{\Delta t},\varepsilon_0),\ \nu_T(u_{\Delta t},\varepsilon_0) \le C\,\mathrm{TV}(u_0)(\varepsilon_0 + \Delta t)$$
$$\text{and} \qquad \mu(u_{\Delta t}(\cdot,0),\varepsilon),\ \mu(u_{\Delta t}(\cdot,T),\varepsilon) \le \mathrm{TV}(u_0)\varepsilon.$$

Thus, taking into consideration Lemmas 4.2 and 4.3, we have

$$\begin{aligned}&\|u(\cdot,T) - u_{\Delta t}(\cdot,T)\|_{\mathrm{L}^1(\mathbb{R}^-)}\\ &\quad + \int_0^T\int_{\mathbb{R}^-}\int_0^T \left(q(u(0-,t),u_{\Delta t}(y,s)) + q(u_{\Delta t}(0-,t),u(y,s))\right)\varphi(0,t,y,s)\,\mathrm{d}t\,\mathrm{d}y\,\mathrm{d}s\\ &\quad \le \|u_0 - u_{\Delta t}(\cdot,0)\|_{\mathrm{L}^1(\mathbb{R}^-)} + C\left(\Delta x + \Delta t + \varepsilon + \varepsilon_0 + \frac{\Delta x}{\varepsilon} + \frac{\Delta x}{\varepsilon_0} + \frac{\Delta t}{\varepsilon_0}\right).\end{aligned}$$

Because of our choice of discretizing the initial datum as $u_j^0 = \frac{1}{\Delta x}\int_{\mathcal{C}_j} u_0(x)\,\mathrm{d}x$ we have $\|u_{\Delta t}(\cdot,0) - u_0\|_{\mathrm{L}^1(\mathbb{R}^-)} \le C\,\mathrm{TV}(u_0)\Delta x$. Now, in order to get a convergence rate, we take $\lambda = \frac{\Delta t}{\Delta x}$ constant and minimize the right-hand side of the above estimate for ε and ε_0. This yields $\varepsilon = \varepsilon_0 = \sqrt{\Delta x}$, and hence

(4.7)
$$\begin{aligned}&\|u(\cdot,T) - u_{\Delta t}(\cdot,T)\|_{\mathrm{L}^1(\mathbb{R}^-)}\\ &\quad + \int_0^T\int_{\mathbb{R}^-}\int_0^T \left(q(u(0-,t),u_{\Delta t}(y,s)) + q(u_{\Delta t}(0-,t),u(y,s))\right)\varphi(0,t,y,s)\,\mathrm{d}t\,\mathrm{d}y\,\mathrm{d}s\\ &\quad \le C\sqrt{\Delta x}.\end{aligned}$$

Using the monotonicity of g we find

$$q(u,v) = |g(u) - g(v)| \ge 0,$$

and thus the integral term in (4.7) is nonnegative which concludes the proof. ∎

4.2. Convergence rate estimates on $\mathbb{R}^+$. As a second step we now consider the initial-boundary value problem

$$\begin{aligned} u_t + f(u)_x &= 0, \quad (x,t) \in \mathbb{R}^+ \times (0,T), \\ u(x,0) &= u_0(x), \quad x \in \mathbb{R}^+, \\ u(0,t) &= f^{-1}\left(g(u(0-,t))\right), \quad t \in (0,T) \end{aligned} \tag{4.8}$$

and the numerical scheme

$$\begin{aligned} u_j^{n+1} &= u_j^n - \lambda\left(f(u_j^n) - f(u_{j-1}^n)\right), \quad j \geq 1,\ n \geq 0, \\ u_j^0 &= \frac{1}{\Delta x}\int_{\mathcal{C}_j} u_0(x)\,\mathrm{d}x, \quad j \geq 0, \\ u_0^n &= f^{-1}\left(g(u_{-1}^n)\right), \quad n \geq 1, \end{aligned} \tag{4.9}$$

where the boundary data is given in terms of $u(0,-,t)$ and u_{-1}^n, respectively, and those are known from the previous section. Note that again we have a discrete entropy inequality of the form

$$D_+^t \eta_j^n + D_- q_j^n \leq 0, \qquad n \geq 1,\ j \geq 1. \tag{4.10}$$

Definition 4.5 (entropy solution on $\mathbb{R}^+$). *We say $u \in \mathcal{C}([0,T];\mathrm{L}^1(\mathbb{R}^+)) \cap \mathrm{L}^\infty(\mathbb{R}^+ \times (0,T))$ is an entropy solution of* (4.8) *if for all $c \in \mathbb{R}$,*

$$\begin{aligned} &\int_0^T \int_{\mathbb{R}^+} (|u-c|\varphi_t + |f(u)-f(c)|\varphi_x)\,\mathrm{d}x\,\mathrm{d}t - \int_{\mathbb{R}^+} |u(x,T)-c|\varphi(x,T)\,\mathrm{d}x \\ &\quad + \int_{\mathbb{R}^+} |u_0(x)-c|\varphi(x,0))\,\mathrm{d}x + \int_0^T |f(u(0+,t)) - f(c)|\varphi(0,t)\,\mathrm{d}t \geq 0 \end{aligned}$$

for all nonnegative $\varphi \in \mathcal{C}^\infty([0,\infty) \times [0,T])$ and

$$f(u(0+,t)) = g(u(0-,t))$$

holds for almost every $t \in (0,T)$.

Before we calculate convergence rates on $\mathbb{R}^+$ we need two auxiliary lemmas that are consequences of the monotonicity of the flux.

Lemma 4.6 (bound on the temporal total variation). *If the numerical scheme* (4.9) *satisfies the CFL condition* (3.2) *the temporal variation of the numerical solution is bounded; specifically, for every $j \in \mathbb{Z}$ we have*

$$\sum_{n=0}^{M} \left|u_j^{n+1} - u_j^n\right| \leq C\,\mathrm{TV}(u_0),$$

where $\mathrm{TV}(u_0)$ refers to the total variation of u_0 on the whole real line.

Proof. Let first $j \geq 1$. Using the CFL condition (3.2) and the monotonicity of the flux, i.e., $f' > 0$, we find that

$$\begin{aligned}|u_j^n - u_{j-1}^n - \lambda(f(u_j^n) - f(u_{j-1}^n))| &= |u_j^n - u_{j-1}^n - \lambda f'(u^*)(u_j^n - u_{j-1}^n)| \\ &= (1 - \lambda f'(u^*))|u_j^n - u_{j-1}^n| \\ &= |u_j^n - u_{j-1}^n| - \lambda f'(u^*)|u_j^n - u_{j-1}^n| \\ &= |u_j^n - u_{j-1}^n| - \lambda|f(u_j^n) - f(u_{j-1}^n)|\end{aligned}$$

and hence

$$\begin{aligned}|u_j^{n+1} - u_{j-1}^{n+1}| &= |u_j^n - u_{j-1}^n - \lambda(f(u_j^n) - f(u_{j-1}^n)) + \lambda(f(u_{j-1}^n) - f(u_{j-2}^n))| \\ &\leq |u_j^n - u_{j-1}^n - \lambda(f(u_j^n) - f(u_{j-1}^n))| + \lambda|f(u_{j-1}^n) - f(u_{j-2}^n)| \\ &= |u_j^n - u_{j-1}^n| - \lambda|f(u_j^n) - f(u_{j-1}^n)| + \lambda|f(u_{j-1}^n) - f(u_{j-2}^n)| \\ &= |u_j^n - u_{j-1}^n| - |u_j^{n+1} - u_j^n| + |u_{j-1}^{n+1} - u_{j-1}^n|,\end{aligned}$$

where we have used the definition of the numerical scheme (4.9) in the last step. Taking the sum over $n = 0, \dots, M-1$ yields

$$\sum_{n=0}^{M-1} \left|u_j^{n+1} - u_{j-1}^{n+1}\right| \leq \sum_{n=0}^{M-1} |u_j^n - u_{j-1}^n| - \sum_{n=0}^{M-1} |u_j^{n+1} - u_j^n| + \sum_{n=0}^{M-1} |u_{j-1}^{n+1} - u_{j-1}^n|,$$

where we can cancel equal terms to get

$$\left|u_j^M - u_{j-1}^M\right| \leq \left|u_j^0 - u_{j-1}^0\right| - \sum_{n=0}^{M-1} \left|u_j^{n+1} - u_j^n\right| + \sum_{n=0}^{M-1} \left|u_{j-1}^{n+1} - u_{j-1}^n\right|. \tag{4.11}$$

Because of the CFL condition (3.2) we have

$$\left|u_j^{M+1} - u_j^M\right| = \lambda \left|f(u_j^M) - f(u_{j-1}^M)\right| = \lambda f'(u^*) \left|u_j^M - u_{j-1}^M\right| \leq \left|u_j^M - u_{j-1}^M\right|$$

which together with (4.11) yields

$$\left|u_j^{M+1} - u_j^M\right| \leq |u_j^0 - u_{j-1}^0| - \sum_{n=0}^{M-1} \left|u_j^{n+1} - u_j^n\right| + \sum_{n=0}^{M-1} \left|u_{j-1}^{n+1} - u_{j-1}^n\right|$$

and thus

$$\sum_{n=0}^{M} \left|u_j^{n+1} - u_j^n\right| \leq \left|u_j^0 - u_{j-1}^0\right| + \sum_{n=0}^{M-1} \left|u_{j-1}^{n+1} - u_{j-1}^n\right|. \tag{4.12}$$

By substituting f with g in the above calculations, the estimate (4.12) also holds for $j < 0$. The estimate (4.12) now allows us to bound the temporal variation of the numerical scheme by the total variation of the initial datum in the following way. If $j > M$ or $j < 0$ we can use the estimate (4.12) iteratively to get

$$\sum_{n=0}^{M} \left|u_j^{n+1} - u_j^n\right| \leq \sum_{i=j-M+1}^{j} \left|u_i^0 - u_{i-1}^0\right| + \left|u_{j-M}^1 - u_{j-M}^0\right|.$$

Using the definition of the scheme (4.9), we get

$$\left|u_{j-M}^1 - u_{j-M}^0\right| = \lambda \left|f(u_{j-M}^0) - f(u_{j-M-1}^0)\right| \leq C\lambda \left|u_{j-M}^0 - u_{j-M-1}^0\right|$$

such that we have

$$\sum_{n=0}^{M} \left|u_j^{n+1} - u_j^n\right| \leq C \sum_{i=j-M}^{j} \left|u_i^0 - u_{i-1}^0\right| \leq C \,\mathrm{TV}(u_0).$$

If on the other hand $0 \leq j \leq M$ we get

$$\sum_{n=0}^{M} \left|u_j^{n+1} - u_j^n\right| \leq \sum_{i=1}^{j} \left|u_i^0 - u_{i-1}^0\right| + \sum_{n=0}^{M-j} \left|u_0^{n+1} - u_0^n\right|.$$

Using the definition of u_0^n in (4.9) and applying (4.12) iteratively again, we get

$$\begin{aligned}\sum_{n=0}^{M-j} \left|u_0^{n+1} - u_0^n\right| &= \sum_{n=0}^{M-j} \left|f^{-1}\left(g(u_{-1}^{n+1})\right) - f^{-1}\left(g(u_{-1}^n)\right)\right| \\ &\leq \frac{C}{\alpha} \sum_{n=0}^{M-j} |u_{-1}^{n+1} - u_{-1}^n| \\ &\leq C \sum_{i=-1-(M-j)}^{-1} |u_i^0 - u_{i-1}^0|\end{aligned}$$

such that we have

$$\sum_{n=0}^{M} \left|u_j^{n+1} - u_j^n\right| \leq C \sum_{i=-1-(M-j)}^{j} |u_i^0 - u_{i-1}^0| \leq C \,\mathrm{TV}(u_0)$$

which concludes the proof. ∎

LEMMA 4.7. *Let u be the entropy solution of (4.8), and assume $f' > 0$. Then $f(u)$ is Lipschitz continuous in space, in the sense that*

$$\int_0^T |f(u(x,t)) - f(u(y,t))| \,\mathrm{d}t \leq C|x-y| \qquad \text{for all } x, y \in \mathbb{R}^+.$$

Proof. Since u is bounded, we can assume that $f' \geq \alpha > 0$. Thus the flux is invertible with Lipschitz continuous inverse. By setting $w = f(u)$ and $h = f^{-1}$ we find that w satisfies

$$w_x + h(w)_t = 0, \qquad (t,x) \in (0,T) \times \mathbb{R}^+.$$

By the standard theory for conservation laws (with the roles of x and t reversed) adapted to the bounded domain $[0, T]$ we see that w is Lipschitz continuous in x with values in $\mathrm{L}^1(0, \mathrm{T})$, i.e.,

$$\int_0^T |f(u(x,t)) - f(u(y,t))| \, \mathrm{d}t = \int_0^T |w(x,t) - w(y,t)| \, \mathrm{d}t \le C|x-y|;$$

cf. [15, Theorem 2.15] or [31, Lemma 4]. Note that an application of [31, Lemma 4] requires, in particular, a temporal total variation bound of $u(0+, t)$ which follows from Lemma 4.6 on a discrete level and carries over in the limit. ∎

We will now describe how to modify the steps in section 4.1 in order to get a convergence rate on $\mathbb{R}^+$. We start by defining

$$\begin{aligned} L(u, c, \varphi) = &\int_0^T \int_{\mathbb{R}^+} (|u-c|\varphi_t + q(u,c)\varphi_x) \, \mathrm{d}x \, \mathrm{d}t - \int_{\mathbb{R}^+} |u(x,T) - c|\varphi(x,T) \, \mathrm{d}x \\ &+ \int_{\mathbb{R}^+} |u_0(x) - c|\varphi(x,0) \, \mathrm{d}x + \int_0^T q(u(0+,t), c)\varphi(0,t) \, \mathrm{d}t \end{aligned}$$

and

$$\Lambda_{\varepsilon,\varepsilon_0}(u, v) = \int_0^T \int_{\mathbb{R}^+} L(u, v(y,s), \varphi(\cdot,\cdot,y,s)) \, \mathrm{d}y \, \mathrm{d}s,$$

where again $\varphi = \omega_\varepsilon(x-y)\omega_{\varepsilon_0}(t-s)$.

Lemma 4.8 (Kuznetsov-type lemma). *Let u be the entropy solution of* (4.8). *Then, for any function $v : [0, T] \to (\mathrm{L}^1 \cap \mathrm{BV})(\mathbb{R}^+)$ such that the one-sided limits $v(t\pm)$ exist in L^1, we have*

$$\begin{aligned} \|u(\cdot,T) - v(\cdot,T)\|_{\mathrm{L}^1(\mathbb{R}^+)} \le &\|u_0 - v(\cdot,0)\|_{\mathrm{L}^1(\mathbb{R}^+)} - \Lambda_{\varepsilon,\varepsilon_0}(v, u) \\ &+ C\Big(\varepsilon + \varepsilon_0 + \nu_T(v, \varepsilon_0) + \nu_0(v, \varepsilon_0) + \mu(v(\cdot,T), \varepsilon)) + \mu(v(\cdot,0), \varepsilon))\Big) \\ &+ \int_0^T \int_{\mathbb{R}^+} \int_0^T (q(u(0+,t), v(y,s)) + q(v(0+,t), u(y,s)))\, \varphi(0,t,y,s) \, \mathrm{d}t \, \mathrm{d}y \, \mathrm{d}s \end{aligned}$$

for some constant C independent of ε and ε_0.

Note that this time the term involving q is on the right-hand side of the inequality.

Proof. The proof follows the same steps, mutatis mutandis, as the proof of the Kuznetsov-type lemma 4.2 on $\mathbb{R}^-$. ∎

Lemma 4.9. *The estimate*

$$-\Lambda_{\varepsilon,\varepsilon_0}(u_{\Delta t}, u) \le C\left(\Delta x + \frac{\Delta x}{\varepsilon} + \frac{\Delta x}{\varepsilon_0} + \frac{\Delta t}{\varepsilon_0}\right)$$

holds for some constant C independent of $\Delta x, \Delta t, \varepsilon$, and ε_0.

Note that the right-hand side of the inequality contains the term $\frac{\Delta x}{\varepsilon_0}$ which was not present in Lemma 4.3 but will not change the overall convergence rate.

Proof. Using summation by parts and the discrete entropy inequality (4.10), $D_+^t\eta_j^n + D_- q_j^n \le 0$ for $j \ge 1$, we find

$$\begin{aligned}
&-L(u_{\Delta t}, u, \varphi)\\
&\quad= -\sum_{n=0}^{M}\sum_{j=0}^{\infty}\left(\eta_j^n \iint_{\mathcal{C}_j^n}\varphi_t\,\mathrm{d}x\,\mathrm{d}t + q_j^n\iint_{\mathcal{C}_j^n}\varphi_x\,\mathrm{d}x\,\mathrm{d}t\right)\\
&\qquad -\sum_{j=0}^{\infty}\eta_j^0\int_{\mathcal{C}_j}\varphi^0\,\mathrm{d}x + \sum_{j=0}^{\infty}\eta_j^{M+1}\int_{\mathcal{C}_j}\varphi^{M+1}\,\mathrm{d}x - \sum_{n=0}^{M} q_0^n\int_{\mathcal{C}^n}\varphi_{-1/2}\,\mathrm{d}t\\
&\quad= \sum_{n=0}^{M}\sum_{j=0}^{\infty}\left(D_+^t\eta_j^n\iint_{\mathcal{C}_j^n}\varphi^{n+1}\,\mathrm{d}x\,\mathrm{d}t + D_- q_{j+1}^n\iint_{\mathcal{C}_j^n}\varphi_{j+\frac{1}{2}}\,\mathrm{d}x\,\mathrm{d}t\right)\\
&\quad\le \sum_{n=0}^{M}\left(D_+^t\eta_0^n\iint_{\mathcal{C}_0^n}\varphi^{n+1}\,\mathrm{d}x\,\mathrm{d}t + \sum_{j=0}^{\infty}D_- q_{j+1}^n\iint_{\mathcal{C}_j^n}\varphi_{j+1/2}\,\mathrm{d}x\,\mathrm{d}t\right.\\
&\qquad\qquad \left. - \sum_{j=1}^{\infty}D_- q_j^n\iint_{\mathcal{C}_j^n}\varphi^{n+1}\,\mathrm{d}x\,\mathrm{d}t\right)\\
&\quad= \sum_{n=0}^{M}\underbrace{D_+^t\eta_0^n\iint_{\mathcal{C}_0^n}\varphi^{n+1}\,\mathrm{d}x\,\mathrm{d}t}_{=:\mathbf{D^n}} + \sum_{n=0}^{M}\sum_{j=1}^{\infty}\underbrace{D_- q_j^n\iint_{\mathcal{C}_j^n}(\varphi_{j-1/2} - \varphi^{n+1})\,\mathrm{d}x\,\mathrm{d}t}_{=:\mathbf{E_j^n}},
\end{aligned}$$

where we have used the notation $\varphi^n = \varphi(x, t^n, y, s)$ and $\varphi_{j+1/2} = \varphi(x_{j+1/2}, t, y, s)$. Concerning the term $\mathbf{D^n}$, using summation by parts again, we find

$$\begin{aligned}
&\sum_{n=0}^{M}\int_0^T\int_{\mathbb{R}^+}\mathbf{D^n}\,\mathrm{d}y\,\mathrm{d}s\\
&\quad= \int_0^T\int_{\mathbb{R}^+}\left(\eta_0^{M+1}\int_{\mathcal{C}_0}\varphi^{M+1}\,\mathrm{d}x - \eta_0^0\int_{\mathcal{C}_0}\varphi^0\,\mathrm{d}x - \sum_{n=0}^{M}\eta_0^n\iint_{\mathcal{C}_0^n}D_+^t\varphi^n\,\mathrm{d}x\,\mathrm{d}t\right)\mathrm{d}y\,\mathrm{d}s.
\end{aligned}$$

Here, using the boundedness of η' and the properties of the mollifiers (4.5), the boundary terms can be estimated as follows:

$$\int_0^T\int_{\mathbb{R}^+}\underbrace{\eta_0^{M+1}}_{\le C\|u_0\|_\infty}\int_{\mathcal{C}_0}\varphi^{M+1}\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}s \le C\Delta x$$

and similarly

$$\int_0^T\int_{\mathbb{R}^+}\eta_0^0\int_{\mathcal{C}_0}\varphi^0\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}s \le C\Delta x.$$

For the remaining term, we can proceed in the following way:

$$\begin{aligned}
&\sum_{n=0}^{M}\int_0^T\int_{\mathbb{R}^+}\underbrace{\eta_0^n}_{\le C\|u_0\|_\infty}\iint_{\mathcal{C}_0^n}D_+^t\varphi^n\,\mathrm{d}x\,\mathrm{d}t\,\mathrm{d}y\,\mathrm{d}s\\
&\qquad \le C\sum_{n=0}^{M}\int_0^T\int_{\mathbb{R}^+}\frac{1}{\Delta t}\iint_{\mathcal{C}_0^n}\int_{\mathcal{C}^n}|\omega'_{\varepsilon_0}(\tau - s)|\,\mathrm{d}\tau\omega_\varepsilon(x-y)\,\mathrm{d}x\,\mathrm{d}t\,\mathrm{d}y\,\mathrm{d}s
\end{aligned}$$

$$\leq C \sum_{n=0}^{M} \frac{\Delta x \Delta t^2}{\Delta t \varepsilon_0}$$
$$\leq C \frac{\Delta x}{\varepsilon_0}.$$

We split the term involving $\mathbf{E_j^n}$ as follows:

$$\sum_{n=0}^{M}\sum_{j=1}^{\infty} \mathbf{E_j^n} \leq \sum_{n=0}^{M}\sum_{j=1}^{\infty} \underbrace{|D_- q_j^n| \iint_{\mathcal{C}_j^n} \int_{x_{j+1/2}}^{x} |\varphi_x(z,t)| \,\mathrm{d}z\,\mathrm{d}x\,\mathrm{d}t}_{=:\mathbf{F_j^n}}$$
$$+ \sum_{n=0}^{M}\sum_{j=1}^{\infty} \underbrace{|D_- q_j^n| \iint_{\mathcal{C}_j^n} \int_{t}^{t^{n+1}} |\varphi_t(x,\tau)| \,\mathrm{d}\tau\,\mathrm{d}x\,\mathrm{d}t}_{=:\mathbf{G_j^n}}.$$

For the first term, using the properties of the mollifiers (4.5) and the Lipschitz continuity of f, we find

$$\int_0^T \int_{\mathbb{R}^+} \sum_{n=0}^{M}\sum_{j=1}^{\infty} \mathbf{F_j^n} \,\mathrm{d}y\,\mathrm{d}s$$
$$= \sum_{n=0}^{M}\sum_{j=1}^{\infty} |D_- q_j^n| \int_0^T \int_{\mathbb{R}^+} \iint_{\mathcal{C}_j^n} \int_{x_{j+1/2}}^{x} |\omega_\varepsilon'(z-y)| \,\mathrm{d}z \omega_{\varepsilon_0}(t-s)\,\mathrm{d}x\,\mathrm{d}t\,\mathrm{d}y\,\mathrm{d}s$$
$$\leq \sum_{n=0}^{M}\sum_{j=1}^{\infty} \frac{C}{\Delta x} |u_j^n - u_{j-1}^n| \frac{\Delta x^2 \Delta t}{\varepsilon}$$
$$\leq C\,\mathrm{TV}(u_0) \frac{\Delta x}{\varepsilon}$$

and similarly

$$\int_0^T \int_{\mathbb{R}^+} \sum_{n=0}^{M}\sum_{j=1}^{\infty} \mathbf{G_j^n} \,\mathrm{d}y\,\mathrm{d}s \leq C\,\mathrm{TV}(u_0) \frac{\Delta t}{\varepsilon_0}.$$

Thus, we have

$$-\Lambda_{\varepsilon,\varepsilon_0}(u_{\Delta t}, u) \leq C \left(\Delta x + \frac{\Delta x}{\varepsilon} + \frac{\Delta x}{\varepsilon_0} + \frac{\Delta t}{\varepsilon_0} \right)$$

which concludes the proof. ◻

THEOREM 4.10 (convergence rate on $\mathbb{R}^+$). *Let u be the entropy solution of the initial-boundary value problem* (4.8) *and $u_{\Delta t}$ the numerical approximation given by* (4.9). *Then we have the following convergence rate estimate:*

$$\|u(\cdot,T) - u_{\Delta t}(\cdot,T)\|_{\mathrm{L}^1(\mathbb{R}^+)} \leq C\sqrt{\Delta x}$$

for some constant C independent of Δx.

Proof. The numerical solution $u_{\Delta t}$ is Lipschitz continuous in time and TVD (for the TVD property of conservation laws on bounded domains see [31, Lemma 2]) and therefore satisfies

$$\begin{aligned} &\nu_0(u_{\Delta t},\varepsilon_0),\ \nu_T(u_{\Delta t},\varepsilon_0) \le C\,\mathrm{TV}(u_0)(\varepsilon_0+\Delta t)\\ \text{and}\qquad &\mu(u_{\Delta t}(\cdot,0),\varepsilon),\ \mu(u_{\Delta t}(\cdot,T),\varepsilon) \le \mathrm{TV}(u_0)\varepsilon. \end{aligned}$$

Thus, taking into consideration Lemmas 4.2 and 4.3, we have

$$\begin{aligned} \|u(\cdot,T)-u_{\Delta t}(\cdot,T)\|_{\mathrm{L}^1(\mathbb{R}^+)} &\le \|u_0-u_{\Delta t}(\cdot,0)\|_{\mathrm{L}^1(\mathbb{R}^+)}\\ &\quad + C\left(\Delta x+\Delta t+\varepsilon+\varepsilon_0+\frac{\Delta x}{\varepsilon}+\frac{\Delta x}{\varepsilon_0}+\frac{\Delta t}{\varepsilon_0}\right)+\mathbf{C}, \end{aligned}$$

where

$$\mathbf{C}=\int_0^T\int_{\mathbb{R}^+}\int_0^T\Big(q(u(0+,t),u_{\Delta t}(y,s))+q(u_{\Delta t}(0+,t),u(y,s))\Big)\varphi(0,t,y,s)\,\mathrm{d}t\,\mathrm{d}y\,\mathrm{d}s.$$

Because of our choice of discretizing the initial datum as $u_j^0=\frac{1}{\Delta x}\int_{\mathcal{C}_j}u_0(x)\,\mathrm{d}x$ we have $\|u_{\Delta t}(\cdot,0)-u_0\|_{\mathrm{L}^1(\mathbb{R}^+)}\le C\,\mathrm{TV}(u_0)\Delta x$, and thus it remains to estimate the term

$$\begin{aligned} \mathbf{C}=\int_0^T\int_{\mathbb{R}^+}\int_0^T\Bigg(&\underbrace{|f(u(0+,t))-f(u_{\Delta t}(y,s))|}_{=:\mathbf{H}}\\ &+\underbrace{|f(u_{\Delta t}(0+,t))-f(u(y,s))|}_{=:\mathbf{J}}\Bigg)\varphi(0,t,y,s)\,\mathrm{d}t\,\mathrm{d}y\,\mathrm{d}s. \end{aligned}$$

Here, we split

$$\mathbf{H}\le\underbrace{|f(u(0+,t))-f(u_{\Delta t}(0+,s))|}_{\mathbf{H_1}}+\underbrace{|f(u_{\Delta t}(0+,s))-f(u_{\Delta t}(y,s))|}_{\mathbf{H_2}}.$$

Using the Rankine–Hugoniot condition, the term $\mathbf{H_1}$ can be estimated as follows:

$$\begin{aligned} \mathbf{H_1}&=|f(u(0+,t))-f(u_{\Delta t}(0+,s))|\\ &=|g(u(0-,t))-g(u_{\Delta t}(0-,s))|\\ &\le|g(u(0-,t))-g(u_{\Delta t}(y,s))|+|g(u_{\Delta t}(y,s))-g(u_{\Delta t}(0-,s))|. \end{aligned}$$

Because $\mathbf{H_1}$ does not depend on y we can use the symmetry of φ with respect to y and the estimate (4.7) to get

$$\begin{aligned} &\int_0^T\int_{\mathbb{R}^+}\int_0^T\mathbf{H_1}\varphi(0,t,y,s)\,\mathrm{d}t\,\mathrm{d}y\,\mathrm{d}s\\ &\quad=\int_0^T\int_{\mathbb{R}^-}\int_0^T\mathbf{H_1}\varphi(0,t,y,s)\,\mathrm{d}t\,\mathrm{d}y\,\mathrm{d}s\\ &\quad\le\int_0^T\int_{\mathbb{R}^-}\int_0^T|g(u(0-,t))-g(u_{\Delta t}(y,s))|\varphi(0,t,y,s)\,\mathrm{d}t\,\mathrm{d}y\,\mathrm{d}s\\ &\qquad+\int_0^T\int_{\mathbb{R}^-}\int_0^T|g(u_{\Delta t}(y,s))-g(u_{\Delta t}(0-,s))|\varphi(0,t,y,s)\,\mathrm{d}t\,\mathrm{d}y\,\mathrm{d}s\\ &\quad\le C\sqrt{\Delta x}+\int_0^T\int_{\mathbb{R}^-}\int_0^T|g(u_{\Delta t}(y,s))-g(u_{\Delta t}(0-,s))|\varphi(0,t,y,s)\,\mathrm{d}t\,\mathrm{d}y\,\mathrm{d}s. \end{aligned}$$

Using the identity

$$|g(u_{i+1}^n) - g(u_i^n)| = \frac{1}{\lambda}|u_{i+1}^{n+1} - u_{i+1}^n|$$

and setting $N = \lceil \frac{\varepsilon}{\Delta x} \rceil$, we can employ Lemma 4.6 to estimate the integral term in the foregoing estimate as follows:

$$\begin{aligned}
&\int_0^T \int_{\mathbb{R}^-} \int_0^T |g(u_{\Delta t}(y,s)) - g(u_{\Delta t}(0-,s))|\varphi(0,t,y,s)\,\mathrm{d}t\,\mathrm{d}y\,\mathrm{d}s \\
&\quad = \sum_{n=0}^{M} \sum_{j=-N}^{-1} |g(u_j^n) - g(u_{-1}^n)| \int_0^T \iint_{\mathcal{C}_j^n} \omega_{\varepsilon_0}(t-s)\omega_\varepsilon(y)\,\mathrm{d}t\,\mathrm{d}y\,\mathrm{d}s \\
&\quad \le C\frac{\Delta t \Delta x}{\varepsilon} \sum_{n=0}^{M} \sum_{j=-N}^{-1} \sum_{i=j}^{-2} \underbrace{|g(u_{i+1}^n) - g(u_i^n)|}_{=\frac{1}{\lambda}|u_{i+1}^{n+1} - u_{i+1}^n|} \\
&\quad \le C\frac{\Delta t \Delta x}{\varepsilon} \sum_{j=-N}^{-1} \sum_{i=j}^{-2} \sum_{n=0}^{M} |u_{i+1}^{n+1} - u_{i+1}^n| \\
&\quad \le C\frac{\Delta t \Delta x}{\varepsilon} \sum_{j=-N}^{-1} (-j) \\
&\quad \le C\frac{\Delta t \Delta x}{\varepsilon} \frac{N(N+1)}{2} \\
&\quad \le C\Delta t \left(\frac{\varepsilon}{\Delta x} + 1\right) \\
&\quad \le C(\varepsilon + \Delta t).
\end{aligned}$$

The term involving $\mathbf{H_2}$ can be estimated analogously. Then it remains to treat the integral involving $\mathbf{J}$. We split $\mathbf{J}$ as follows:

$$\mathbf{J} \le \underbrace{|f(u_{\Delta t}(0+,t)) - f(u(0+,s))|}_{\mathbf{J_1}} + \underbrace{|f(u(0+,s)) - f(u(y,s))|}_{\mathbf{J_2}}$$

and note that the $\mathbf{J_1}$ is the same as $\mathbf{H_1}$. Lastly, with the help of Lemma 4.7 we find

$$\begin{aligned}
&\int_0^T \int_{\mathbb{R}^+} \int_0^T \mathbf{J_2}\varphi(0,t,y,s)\,\mathrm{d}t\,\mathrm{d}y\,\mathrm{d}s \\
&\quad = \int_0^T \int_{\mathbb{R}^+} \int_0^T |f(u(0+,s)) - f(u(y,s))|\varphi(0,t,y,s)\,\mathrm{d}t\,\mathrm{d}y\,\mathrm{d}s \\
&\quad \le \frac{1}{\varepsilon}\int_0^\varepsilon \underbrace{\int_0^T |f(u(0+,s)) - f(u(y,s))|\,\mathrm{d}s}_{\le C|y|}\,\mathrm{d}y \\
&\quad \le \frac{C}{\varepsilon}\int_0^\varepsilon |y|\,\mathrm{d}y \\
&\quad \le C\varepsilon.
\end{aligned}$$

Finally, we have

$$\|u(\cdot,T) - u_{\Delta t}(\cdot,T)\|_{\mathrm{L}^1(\mathbb{R}^+)} \le C\left(\Delta x + \Delta t + \varepsilon + \varepsilon_0 + \frac{\Delta x}{\varepsilon} + \frac{\Delta x}{\varepsilon_0} + \frac{\Delta t}{\varepsilon_0}\right).$$

In order to get a convergence rate, again we take $\lambda = \frac{\Delta t}{\Delta x}$ constant and minimize the right-hand side of the above estimate for ε and ε_0. This yields $\varepsilon = \varepsilon_0 = \sqrt{\Delta x}$ which concludes the proof. ∎

4.3. Convergence rate estimates on $(0,L)$. By restricting the solution u and the numerical approximation $u_{\Delta t}$ to a bounded interval $(0, L)$, Theorem 4.10 and the estimate (4.7) yield a convergence rate on $(0, L)$. Note that this is only possible since f is strictly monotone.

Corollary 4.11 (convergence rate on $(0, L)$). *Let u be the entropy solution of the initial-boundary value problem* (4.8) *on the bounded interval* $[0, L]$ *and $u_{\Delta t}$ the numerical approximation given by* (4.9). *Then we have the following convergence rate estimate:*

$$\|u(\cdot, T) - u_{\Delta t}(\cdot, T)\|_{\mathrm{L}^1(0,L)} \le C\sqrt{\Delta x}$$

for some constant C independent of Δx.

Proof. Without repeating all calculations of sections 4.1 and 4.2 we will highlight the adjustments to the respective proofs that need to be done. If we consider solutions on $(0, L)$ instead of $\mathbb{R}^+$ the definition of $\Lambda_{\varepsilon,\varepsilon_0}(u, v)$ in 4.6 needs to be adjusted so that $\Lambda_{\varepsilon,\varepsilon_0}(u, v)$ contains the term

$$-\int_0^T\int_0^L\int_0^T q(u(L-, t), v(y, s))\varphi(L, t, y, s)\,\mathrm{d}t\,\mathrm{d}y\,\mathrm{d}s,$$

and all instances of $\mathbb{R}^+$ need to be changed to $(0, L)$. Following the proofs of Theorems 4.4 and 4.10 in the same way finally yields

$$\begin{aligned}
&\|u(\cdot, T) - u_{\Delta t}(\cdot, T)\|_{\mathrm{L}^1(0,L)}\\
&\quad + \int_0^T\int_0^L\int_0^T \left(q(u(L, t), u_{\Delta t}(y, s)) + q(u_{\Delta t}(L, t), u(y, s))\right)\varphi(L, t, y, s)\,\mathrm{d}t\,\mathrm{d}y\,\mathrm{d}s\\
&\le C\sqrt{\Delta x}.
\end{aligned} \tag{4.13}$$

Using the monotonicity of f we find

$$q(u, v) = |f(u) - f(v)| \ge 0,$$

and thus the integral term in (4.13) is nonnegative which concludes the proof. ∎

5. Statement and proof of the main theorem. Our main result now reads as follows.

Theorem 5.1 (convergence rate for conservation laws with discontinuous flux). *Let u be the entropy solution of* (1.1) *and $u_{\Delta t}$ the numerical solution given by* (3.1). *Then we have the following convergence rate:*

$$\|u(\cdot, T) - u_{\Delta t}(\cdot, T)\|_{\mathrm{L}^1(\mathbb{R})} \le C\sqrt{\Delta x}$$

for some constant C independent of Δx.

Proof. As before, we decompose the entropy solution u as $u = \sum_{i=0}^N u^{(i)}$ where $u^{(i)}$, $i = 0, \dots, N$, are the respective entropy solutions on D_i, i.e., solutions of (2.3) and (2.4), respectively. Further, we decompose the numerical solution $u_{\Delta t}$ as $\sum_{i=0}^N u_{\Delta t}^{(i)}$ where

$$u_{\Delta t}^{(i)}(x,t) = \begin{cases} u_j^n & \text{if } (x,t) \in \mathcal{C}_j^n \subset D_i \times \mathcal{C}^n, \\ 0 & \text{otherwise} \end{cases}$$

and u_j^n is given by (3.1). Then we have

$$\|u(T) - u_{\Delta t}(T)\|_{\mathrm{L}^1(\mathbb{R})} = \sum_{i=0}^{N} \left\| u^{(i)}(T) - u_{\Delta t}^{(i)}(T) \right\|_{\mathrm{L}^1(D_i)}.$$

Using Theorem 4.4 for D_0, Theorem 4.10 for D_N, and Corollary 4.11 for each D_i, $i = 1, \ldots, N-1$, shows that

$$\left\| u^{(i)}(T) - u_{\Delta t}^{(i)}(T) \right\|_{\mathrm{L}^1(D_i)} \leq C\sqrt{\Delta x}$$

for $i = 0, \ldots, N$ which concludes the proof. □

Remark 5.2. Note that the rate of Theorem 5.1 is optimal in the sense that it cannot be improved without further assumptions on the initial datum. This can easily be shown in the same way as in the absence of spatial dependency since the specific initial datum u_0 constructed by Şabac in [34] can be chosen in a way such that u_0 is supported away from the last discontinuity.

6. Numerical experiments.

To illustrate our results we now present two numerical experiments. We consider the "two flux" case

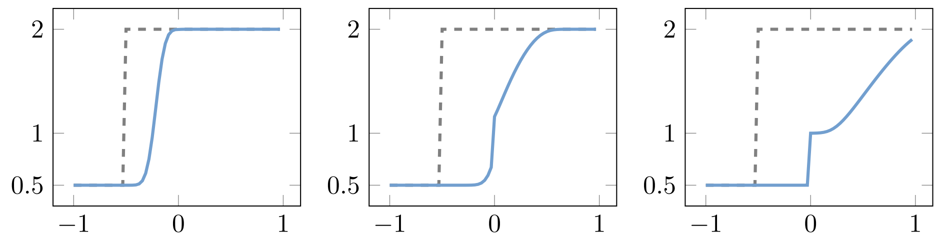
$$\begin{aligned} u_t + (H(x)f(u) + (1 - H(x))g(u))_x &= 0, \quad (x,t) \in \mathbb{R} \times (0,T), \\ u(x,0) &= u_0(x), \quad x \in \mathbb{R}, \end{aligned}$$

where H is the Heaviside function. This corresponds to switching from one u-dependent flux, g, to another, f.

Experiment 1. In our first numerical experiment we choose $g(u) = u$ and $f(u) = {}^{u^2}\!/_2$ such that we switch from the transport equation to the Burgers equation across $x = 0$. The initial datum we consider for Experiment 1 is

$$u_0(x) = \begin{cases} 0.5 & \text{if } x < -0.5, \\ 2 & \text{if } x > -0.5 \end{cases}$$

which is chosen such that the Rankine–Hugoniot condition at $x = 0$ gives $u(0-,t) = u(0+,t)$ before the jump at $x = -0.5$ interacts with the interface. Figure 1 shows the numerical solution calculated with the scheme (3.1) with open boundaries in blue



(a) $t = 0.3$. (b) $t = 0.6$. (c) $t = 0.9$.

FIG. 1. *Numerical solution of Experiment* 1 *with* $\Delta x = {}^{2}\!/_{64}$ *at various times.*

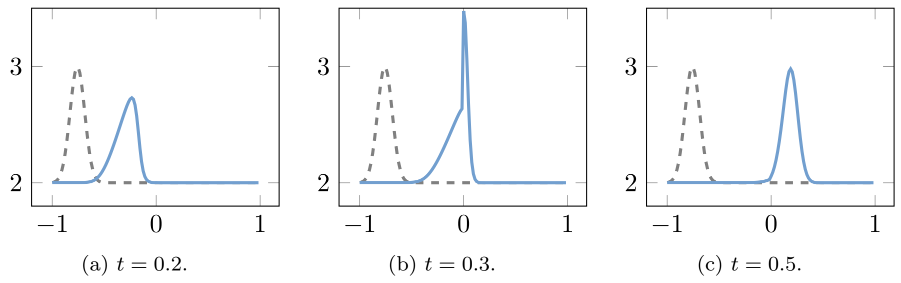
and the initial datum in gray (dashed line) at various times (before, during, and after interaction with the interface). We used $\Delta x = {}^2/_n$ with $n = 64$, end time $T = 0.9$, and $\lambda = 0.5$. We clearly recognize the characteristic features of the transport equation and the Burgers equation here as the upward jump in the initial datum is transported to the right as a shock until it crosses the interface at $x = 0$ where the shock, as it enters the Burgers regime, subsequently becomes a rarefaction wave.

Experiment 2. In our second numerical experiment we choose $g(u) = {}^{u^2}/_2$ and $f(u) = u$ such that we switch from the Burgers equation to the transport equation across $x = 0$. The initial datum we consider is

$$u_0(x) = 2 + \exp(-100(x + 0.75)^2).$$

Again, the offset of the initial datum is chosen in a way such that the Rankine–Hugoniot condition at $x = 0$ gives $u(0-, t) = u(0+, t)$ before the nonconstant part of u_0 interacts with the interface. Figure 2 shows the numerical solution calculated with the scheme (3.1) with open boundaries in blue and the initial datum in gray (dashed line) at various times (immediately before, during, and after interaction with the interface). We used $\Delta x = {}^2/_n$ with $n = 128$, end time $T = 0.5$, and $\lambda = 0.2$. We clearly recognize the shock formation due to the Burgers regime to the left of the interface (see Figure 2(a)). Note that—although difficult to see in Figure 2(c) because of numerical diffusion—the shock is preserved over the interface (only with a different profile).

Table 1 shows the observed convergence rates of the solution at time $T = 0.9$ for Experiment 1 and at time $T = 0.5$ for Experiment 2 for various values of Δx. As a reference solution, we used a numerical solution on a very fine grid ($n = 2048$) in both cases. As expected from experience in the case of spatially independent flux we observe convergence rates strictly between ${}^1/_2$ and 1 (cf., e.g., [24, clawpack software]).



(a) $t = 0.2$. (b) $t = 0.3$. (c) $t = 0.5$.

FIG. 2. *Numerical solution of Experiment* 2 *with* $\Delta x = {}^2/_{128}$ *at various times.*

TABLE 1
L^1 *error and observed order of convergence for Experiments* 1 *and* 2.

n	L^1 error	L^1 OOC
16	1.751×10^{-1}	–
32	1.256×10^{-1}	0.48
64	8.865×10^{-2}	0.50
128	5.918×10^{-2}	0.58
256	3.637×10^{-2}	0.70
512	1.978×10^{-2}	0.88
1024	8.145×10^{-3}	1.28

(a) Experiment 1.

n	L^1 error	L^1 OOC
16	2.771×10^{-1}	–
32	1.823×10^{-1}	0.60
64	1.261×10^{-1}	0.53
128	8.390×10^{-2}	0.59
256	5.125×10^{-2}	0.71
512	2.780×10^{-2}	0.88
1024	1.132×10^{-2}	1.30

(b) Experiment 2.

7. Conclusion.

Scalar conservation laws with discontinuous flux frequently occur in physical applications, and several numerical schemes have been considered in the literature. In contrast to the case where the nonlinear flux does not have a spatial dependency, however, convergence rate results for monotone finite volume schemes have not been available until now.

In this paper, we have established a convergence rate for upwind-type finite volume methods for the case where f is strictly monotone in u and the spatial dependency k is piecewise constant with finitely many discontinuities. The central idea of this paper is to split the problem into finitely many conservation laws between two neighboring discontinuities of k and thus get a convergence rate as a consequence of convergence rates on bounded domains. Here, the novel feature of this paper is the strong bound on the temporal total variation of the finite volume approximation which allows us to estimate the boundary terms in space at the discontinuities of k that appear when applying the classical Kuznetsov theory to problem (1.1).

As an outlook we name four possible directions of future research. A first direction would be to extend the convergence rate result of this paper to the cases where k is not piecewise constant and f is not monotone. Second, it might be interesting to investigate convergence rates of monotone schemes in the Wasserstein distance. In the case of spatially independent fluxes, convergence rates in the Wasserstein distance are well-understood due to Nessyahu, Tadmor, and Tassa [28, 29]. A third direction of future research might be to see whether the results of this paper can be extended to monotone schemes in conservation form, i.e., where the definition of $u_{P_i}^{n+1}$ in (3.1) is replaced by $u_{P_i}^{n+1} = u_{P_i}^n - \lambda(f^{(i)}(u_{P_i}^n) - f^{(i-1)}(u_{P_i-1}^n))$. Lastly, convergence rates of the front tracking method for conservation laws with discontinuous flux are highly desirable as well. In the case of spatially independent fluxes, convergence rates of the front tracking method are known in L^1 due to Lucier [26] and in the Wasserstein distances due to Solem [35].

Appendix A. Convergence rate estimates for general initial-boundary value problems.

With the techniques developed in this paper, we can also derive a convergence rate for the initial-boundary value problem

$$\begin{aligned} u_t + f(u)_x &= 0, \quad (x,t) \in (0,L) \times (0,T),\\ u(x,0) &= u_0(x), \quad x \in (0,L),\\ u(0,t) &= a(t), \quad t \in (0,T) \end{aligned} \tag{A.1}$$

and the numerical scheme

$$\begin{aligned} u_j^{n+1} &= u_j^n - \lambda\left(f(u_j^n) - f(u_{j-1}^n)\right), \quad j \geq 1,\ n \geq 0,\\ u_j^0 &= \frac{1}{\Delta x}\int_{\mathcal{C}_j} u_0(x)\,\mathrm{d}x, \quad j \geq 0,\\ u_0^n &= \frac{1}{\Delta t}\int_{\mathcal{C}^n} a(s)\,\mathrm{d}s, \quad n \geq 1. \end{aligned}$$

Here we need to assume that $a \in (\mathrm{L}^1 \cap \mathrm{BV})(0,T)$ which allows us to use the total variation of a directly instead of crossing the discontinuity in Lemma 4.6. The assertion of Lemma 4.6 should then read

$$\sum_{n=0}^{M} |u_j^{n+1} - u_j^n| \leq C(\mathrm{TV}(u_0) + \mathrm{TV}(a))$$

which can be used at the same place Lemma 4.6 is used in Theorem 4.10. Hence, Corollary 4.11 gives the convergence rate $\mathcal{O}(\sqrt{\Delta x})$ for the general initial-boundary value problem (A.1). Note that this is a higher rate than the $\mathcal{O}(\Delta x^{1/3})$ rate mentioned in [30].

Acknowledgments. We thank Nils Henrik Risebro for several useful discussions and Ulrik Skre Fjordholm for his careful reading of the manuscript. We would also like to thank the referees for their constructive and insightful comments.

REFERENCES

[1] Adimurthi, S. Mishra, and G. V. Gowda, *Conservation law with the flux function discontinuous in the space variable—II: Convex–concave type fluxes and generalized entropy solutions*, J. Comput. Appl. Math., 203 (2007), pp. 310–344.

[2] Adimurthi, S. Misra, and G. V. Gowda, *Optimal entropy solutions for conservation laws with discontinuous flux-functions*, J. Hyperbolic Differ. Equ., 2 (2005), pp. 783–837.

[3] B. Andreianov, K. H. Karlsen, and N. H. Risebro, *A theory of L^1-dissipative solvers for scalar conservation laws with discontinuous flux*, Arch. Ration. Mech. Anal., 201 (2011), pp. 27–86.

[4] E. Audusse and B. Perthame, *Uniqueness for scalar conservation laws with discontinuous flux via adapted entropies*, Proc. Roy. Soc. Edinburgh Sect. A, 135 (2005), pp. 253–265.

[5] J. Badwaik, N. H. Risebro, and C. Klingenberg, *Multilevel Monte Carlo Finite Volume Methods for Random Conservation Laws with Discontinuous Flux*, preprint, arXiv:1906.08991, 2019.

[6] P. Baiti and H. K. Jenssen, *Well-posedness for a class of 2×2 conservation laws with L^∞ data*, J. Differential Equations, 140 (1997), pp. 161–185.

[7] R. Bürger, K. Karlsen, C. Klingenberg, and N. Risebro, *A front tracking approach to a model of continuous sedimentation in ideal clarifier–thickener units*, Nonlinear Anal. Real World Appl., 4 (2003), pp. 457–481.

[8] G. Coclite, J. Ridder, and N. Risebro, *A convergent finite difference scheme for the Ostrovsky-Hunter equation on a bounded domain*, BIT, 57 (2017), pp. 93–122.

[9] G. M. Coclite and N. H. Risebro, *Conservation laws with time dependent discontinuous coefficients*, SIAM J. Math. Anal., 36 (2005), pp. 1293–1309.

[10] M. G. Crandall and A. Majda, *Monotone difference approximations for scalar conservation laws*, Math. Comp., 34 (1980), pp. 1–21.

[11] S. Diehl, *A conservation law with point source and discontinuous flux function modelling continuous sedimentation*, SIAM J. Appl. Math., 56 (1996), pp. 388–419.

[12] T. Gimse and N. H. Risebro, *Riemann problems with a discontinuous flux function*, in Proceedings of the Third International Conference on Hyperbolic Problems, vol. 1, 1991, pp. 488–502.

[13] T. Gimse and N. H. Risebro, *Solution of the cauchy problem for a conservation law with a discontinuous flux function*, SIAM J. Math. Anal., 23 (1992), pp. 635–648.

[14] J. Greenberg, A. Leroux, R. Baraille, and A. Noussair, *Analysis and approximation of conservation laws with source terms*, SIAM J. Numer. Anal., 34 (1997), pp. 1980–2007.

[15] H. Holden and N. H. Risebro, *Front Tracking for Hyperbolic Conservation Laws*, Springer-Verlag, Berlin, 2015.

[16] K. Karlsen, N. Risebro, and J. Towers, *Upwind difference approximations for degenerate parabolic convection–diffusion equations with a discontinuous coefficient*, IMA J. Numer. Anal., 22 (2002), pp. 623–664.

[17] K. H. Karlsen, N. H. Risebro, and J. D. Towers, *On a nonlinear degenerate parabolic transport-diffusion equation with a discontinuous coefficient*, Electron. J. Differential Equations, 2002 (2002), 93.

[18] K. H. Karlsen and J. D. Towers, *Convergence of the Lax-Friedrichs scheme and stability for conservation laws with a discontinuous space-time dependent flux*, Chin. Ann. Math., 25 (2004), pp. 287–318.

[19] R. A. Klausen and N. H. Risebro, *Stability of conservation laws with discontinuous coefficients*, J. Differential Equations, 157 (1999), pp. 41–60.

[20] C. Klingenberg and N. H. Risebro, *Convex conservation laws with discontinuous coefficients. Existence, uniqueness and asymptotic behavior*, Comm. Partial Differential Equations, 20 (1995), pp. 1959–1990.

[21] C. Klingenberg and N. H. Risebro, *Stability of a resonant system of conservation laws modeling polymer flow with gravitation*, J. Differential Equations, 170 (2001), pp. 344–380.

[22] S. N. Kružkov, *First order quasilinear equations in several independent variables*, Sb. Math., 10 (1970), pp. 217–243.

[23] N. Kuznetsov, *Accuracy of some approximate methods for computing the weak solutions of a first-order quasi-linear equation*, USSR Comput. Math. Math. Phys., 16 (1976), pp. 105–119.

[24] R. J. LeVeque, *Finite Volume Methods for Hyperbolic Problems*, Cambridge Texts Appl. Math., Cambridge University Press, Cambridge, UK, 2002.

[25] M. J. Lighthill and G. B. Whitham, *On kinematic waves* II. *A theory of traffic flow on long crowded roads*, Proc. A, 229 (1955), pp. 317–345.

[26] B. J. Lucier, *A moving mesh numerical method for hyperbolic conservation laws*, Math. Comp., 46 (1986), pp. 59–69.

[27] S. Mishra, *Convergence of upwind finite difference schemes for a scalar conservation law with indefinite discontinuities in the flux function*, SIAM J. Numer. Anal., 43 (2005), pp. 559–577.

[28] H. Nessyahu and E. Tadmor, *The convergence rate of approximate solutions for nonlinear scalar conservation laws*, SIAM J. Numer. Anal., 29 (1992), pp. 1505–1519.

[29] H. Nessyahu, E. Tadmor, and T. Tassa, *The convergence rate of Godunov type schemes*, SIAM J. Numer. Anal., 31 (1994), pp. 1–16.

[30] M. Ohlberger and J. Vovelle, *Error estimate for the approximation of nonlinear conservation laws on bounded domains by the finite volume method*, Math. Comp., 75 (2006), pp. 113–150.

[31] J. Ridder and A. M. Ruf, *A convergent finite difference scheme for the Ostrovsky–Hunter equation with Dirichlet boundary conditions*, BIT, 2019.

[32] N. H. Risebro and A. Tveito, *Front tracking applied to a nonstrictly hyperbolic system of conservation laws*, SIAM J. Sci. Statist. Comput., 12 (1991), pp. 1401–1419.

[33] A. M. Ruf, E. Sande, and S. Solem, *The optimal convergence rate of monotone schemes for conservation laws in the Wasserstein distance*, J. Sci. Comput., 2019.

[34] F. Şabac, *The optimal convergence rate of monotone finite difference methods for hyperbolic conservation laws*, SIAM J. Numer. Anal., 34 (1997), pp. 2306–2318.

[35] S. Solem, *Convergence rates of the front tracking method for conservation laws in the Wasserstein distances*, SIAM J. Numer. Anal., 56 (2018), pp. 3648–3666.

[36] Z.-H. Teng and P. Zhang, *Optimal L^1-rate of convergence for the viscosity method and monotone scheme to piecewise constant solutions with shocks*, SIAM J. Numer. Anal., 34 (1997), pp. 959–978.

[37] J. Towers, *Convergence of a difference scheme for conservation laws with a discontinuous flux*, SIAM J. Numer. Anal., 38 (2000), pp. 681–698.

[38] J. Towers, *A difference scheme for conservation laws with a discontinuous flux: The nonconvex case*, SIAM J. Numer. Anal., 39 (2001), pp. 1197–1218.

[39] D. A. Venditti and D. L. Darmofal, *Adjoint error estimation and grid adaptation for functional outputs: Application to quasi-one-dimensional flow*, J. Comput. Phys., 164 (2000), pp. 204–227.

[40] X. Wen and S. Jin, *Convergence of an immersed interface upwind scheme for linear advection equations with piecewise constant coefficients* I*: L^1-error estimates*, J. Comput. Math., 26 (2008), pp. 1–22.

MLMC Methods for Random Conservation Laws with Discontinuous Flux

ESAIM: M2AN 55 (2021) 1039–1065
https://doi.org/10.1051/m2an/2021011

ESAIM: Mathematical Modelling and Numerical Analysis
www.esaim-m2an.org

MULTILEVEL MONTE CARLO FINITE VOLUME METHODS FOR RANDOM CONSERVATION LAWS WITH DISCONTINUOUS FLUX

Jayesh Badwaik[1], Christian Klingenberg[1], Nils Henrik Risebro[2] and Adrian M. Ruf[3,*]

Abstract. We consider conservation laws with discontinuous flux where the initial datum, the flux function, and the discontinuous spatial dependency coefficient are subject to randomness. We establish a notion of random adapted entropy solutions to these equations and prove well-posedness provided that the spatial dependency coefficient is piecewise constant with finitely many discontinuities. In particular, the setting under consideration allows the flux to change across finitely many points in space whose positions are uncertain. We propose a single- and multilevel Monte Carlo method based on a finite volume approximation for each sample. Our analysis includes convergence rate estimates of the resulting Monte Carlo and multilevel Monte Carlo finite volume methods as well as error *versus* work rates showing that the multilevel variant outperforms the single-level method in terms of efficiency. We present numerical experiments motivated by two-phase reservoir simulations for reservoirs with varying geological properties.

Mathematics Subject Classification. 35L65, 35R05, 65C05, 65M12.

Received September 25, 2020. Accepted March 9, 2021.

1. Introduction

This paper concerns uncertainty quantification for conservation laws with *discontinuous flux* of the form

$$\begin{aligned} u_t + f(k(x),u)_x &= 0, \quad x\in\mathbb{R},\ t>0,\\ u(x,0) &= u_0(x), \quad x\in\mathbb{R}. \end{aligned} \tag{1.1}$$

Here, $u\colon \mathbb{R}\times[0,\infty)\to\mathbb{R}$ is the unknown and $f\in\mathcal{C}^2(\mathbb{R}^2;\mathbb{R})$ is the flux function having a possibly *discontinuous* spatial dependency through the coefficient k. In particular, we will assume that the initial datum u_0 is in $(\mathrm{L}^\infty\cap\mathrm{BV})(\mathbb{R})$, the flux f is strictly increasing in u, and the coefficient k is piecewise constant with finitely many discontinuities. Going back to (1.1), this amounts to switching from one u-dependent flux to another across finitely many points in space.

Keywords and phrases. Uncertainty quantification, conservation laws, discontinuous flux, numerical methods.

[1] Department of Mathematics, University of Würzburg, Würzburg, Germany.
[2] Department of Mathematics, University of Oslo, Oslo, Norway.
[3] Seminar for Applied Mathematics, ETH Zürich, Switzerland.
*Corresponding author: adrian.ruf@sam.math.ethz.ch

Equations of type (1.1) arise in a number of areas of application including vehicle traffic flow in the presence of abruptly varying road conditions (see [35]), polymer flooding in oil recovery (see [48]), two-phase flow through heterogeneous porous media (see [22, 23, 44]), and sedimentation processes (see [9, 14]).

Even in the absence of flux discontinuities, and even if the initial datum is smooth, solutions of (1.1) develop discontinuities in finite time and for this reason weak solutions are sought. Weak solutions to (1.1) are not unique, so the weak formulation of the problem is augmented with an additional entropy condition. In the case where $x \mapsto f(k(x), u)$ is smooth, uniqueness follows from the classical Kružkov entropy conditions [33]. In the presence of spatial flux discontinuities, standard Kružkov entropy conditions no longer make sense. This difficulty is usually resolved by requiring that Kružkov entropy conditions hold away from the spatial flux discontinuities and imposing additional jump conditions along the spatial interfaces [1, 4, 5, 14, 21, 22, 26, 29, 30, 49, 50] or by adapting the Kružkov entropy conditions in a suitable way [6–8, 42, 46, 51]. In the present paper we will focus on the second approach of so-called adapted entropy solutions for which we need to require that the flux function f is strictly monotone in u.

In the last two decades, there has been a large interest in the numerical approximation of entropy solutions of (1.1) under various assumptions on k and f. We refer to [3, 4, 9, 10, 12, 20–22, 25–28, 30, 31, 37, 49, 50, 55] for a partial list of references regarding finite volume methods respectively the front tracking method. Specifically, in the adapted entropy framework we want to highlight the results of [7, 8, 15–17, 42, 46, 51] regarding finite volume methods and the front tracking method. We refer to [5, 7, 46] for an overview of the literature concerning conservation laws with discontinuous flux.

The classical paradigm for designing efficient numerical schemes assumes that *data for* (1.1)*, i.e., the initial datum* u_0*, the flux* f*, and the spatial dependency coefficient* k*, are known exactly.*

However, in many situations of practical interest, there is an inherent uncertainty in the modeling and measurement of physical parameters. For example, in two-phase flow through a heterogeneous porous medium the position of the interface between two rock types is typically not known exactly. Often these parameters are only known up to certain statistical quantities of interest like the mean, variance, or higher moments. In such cases, a mathematical framework of (1.1) is required which allows for *random data.*

For standard conservation laws without spatial flux dependency, *i.e.*, for

$$\begin{aligned} u_t + f(u)_x &= 0, \quad x \in \mathbb{R},\ t > 0, \\ u(x,0) &= u_0(x), \quad x \in \mathbb{R}, \end{aligned} \tag{1.2}$$

such a framework was developed in a series of papers allowing for random initial datum [38], random (spatially independent) flux [41], and even random source terms [39] and random diffusion [32].

The first aim of the current paper is to *extend this mathematical framework to include scalar conservation laws with discontinuous flux with random discontinuous spatial dependency.* To that end, we define random entropy solutions and provide an existence and uniqueness result, which generalizes the well-posedness results for (1.2) to the case of uncertain initial datum, flux, and discontinuous spatial dependency. In particular, our framework allows for uncertain positions of the flux discontinuities.

The second aim of this paper is to *design fast and robust numerical algorithms for computing the mean of random entropy solutions of conservation laws with discontinuous flux.* Specifically, we propose and analyze a multilevel combination of Monte Carlo (MC) sampling and a "pathwise" finite volume method (FVM) to approximate the mean of random entropy solutions of conservation laws with discontinuous flux. The multilevel Monte Carlo finite volume method (MLMCFVM) for (1.1) is non-intrusive (in the sense that it requires only repeated applications of existing solvers for input data samples), easy to implement and to parallelize, and well suited for random solutions with low spatial regularity. Solutions exhibiting spatial discontinuities are generic for conservation laws and, in particular, for conservation laws with discontinuous flux. This reduced regularity poses some challenges to the design of efficient so-called stochastic Galerkin methods for example which are based on generalized polynomial chaos. These methods are well-developed for conservation laws – albeit without flux discontinuities – (see [2, 11, 36, 43, 52, 54] and references therein), but they are more intrusive, generally harder to implement and to parallelize. Thus, in the present paper, we focus on the design and mathematical analysis of statistical MC-type methods. Our analysis includes the proof of convergence rates at which the MCFVM and the MLMCFVM converge towards the mean of the random entropy solution of (1.1). The analysis is complicated by the fact that adapted entropy solutions of (1.1) do not possess the same stability properties as entropy solutions of (1.2). Moreover, we determine the number of MC samples needed to minimize the computational work for a given error tolerance.

We want to emphasize that the framework of adapted entropy solutions and more specifically the setting of the present paper is currently the only setting for which we simultaneously have existence [51], uniqueness [6], stability with respect to the modeling parameters [46], and numerical methods with a provable convergence rate [7, 46] – the essential components for an uncertainty quantification framework (*cf.* [41]).

The remainder of this paper is organized as follows. In Section 2 we introduce preliminary results regarding the MC approximation of Banach space-valued random variables. Section 3 is devoted to a review of existence and stability results regarding entropy solutions of (deterministic) conservation laws with discontinuous flux of the form (1.1). In Section 4 we introduce random entropy solutions of (1.1) where the initial datum u_0, the flux f, and the discontinuous coefficient k are subject to randomness. In particular, we prove existence and uniqueness of random entropy solutions. In Section 5, we first review a FVM which was introduced in [7] for the deterministic problem, prove certain stability estimates, and then extend the FVM to MC as well as MLMC versions for (1.1) with random parameters. In Section 6 we perform numerical experiments motivated by two-phase reservoir simulations for reservoirs with varying geological properties to validate our error estimates. Finally, we summarize the findings of this paper in Section 7.

2. Preliminaries on the Monte Carlo method

We first introduce some preliminary concepts which are needed in the exposition. To that end, we follow [34, 53], see also Section 2 of [32] and Section 5 of [13].

Given a probability space $(\Omega, \mathcal{F}, \mathbb{P})$, a Banach space V, and a random variable $X\colon \Omega \to V$ we are interested in approximating the mean $\mathbb{E}[X]$ of X *via* Monte Carlo sampling. To this end, let $(\hat{X}^i)_{i=1}^M$, $i = 1, \dots, M$, be M independent, identically distributed samples of X. Then, the Monte Carlo estimator $E_M[X]$ of $\mathbb{E}[X]$ is defined as the sample average

$$E_M[X] \coloneqq \frac{1}{M}\sum_{i=1}^{M} \hat{X}^i.$$

We are interested in deriving a rate at which

$$\|\mathbb{E}[X] - E_M[X]\|_{\mathrm{L}^q(\Omega;V)} = \mathbb{E}[\|\mathbb{E}[X] - E_M[X]\|_V^q]^{\frac{1}{q}}$$

converges as $M \to \infty$ for some $1 \le q < \infty$ and some Banach space V (typically a Lebesgue space). For general Banach spaces V such convergence rate estimates depend on the type of the Banach space.

Definition 2.1 (Banach space of type q [34], p. 246)**.** Assume that Ω permits a sequence of independent Rademacher random variables $Z_i, i \in \mathbb{N}$. We say that a Banach space V is a Banach space of type $1 \le q \le 2$ if there is a constant $\kappa > 0$ such that for all finite sequences $(x_i)_{i=1}^M \subseteq V$

$$\left(\mathbb{E}\left\|\sum_{i=1}^{M} Z_i x_i\right\|_V^q\right)^{\frac{1}{q}} \le \kappa\left(\sum_{i=1}^{M}\|x_i\|_V^q\right)^{\frac{1}{q}}.$$

We will refer to κ as the type constant of V.

Every Banach space is a Banach space of type 1 and every Hilbert space a Banach space of type 2 ([34], Thm. 9.10). Moreover, L^p spaces are Banach spaces of type $q = \min(2, p)$ for $1 \leq p < \infty$ ([34], p. 247). We will need the following results regarding Lebesgue spaces of functions with values in a Banach space of type q.

Lemma 2.2 ([34], p. 247)**.** *Let $1 \leq r \leq \infty$, $(\Omega, \mathcal{F}, \mathbb{P})$ be a measure space, and V be a Banach space of type q. Then the space $\mathrm{L}^r(\Omega, V)$ is a Banach space of type $\min(r, q)$.*

Proposition 2.3 ([34], Prop. 9.11)**.** *Let V be a Banach space of type q with type constant κ. Then, for every finite sequence $(X_i)_{i=1}^M$ of independent mean zero random variables in $\mathrm{L}^q(\Omega, V)$, we have*

$$\mathbb{E}\left[\left\|\sum_{i=1}^M X_i\right\|_V^q\right] \leq (2\kappa)^q \sum_{i=1}^M \mathbb{E}\left[\|X_i\|_V^q\right].$$

Corollary 2.4 ([32], Cor. 2.5)**.** *Let V be a Banach space of type q with type constant κ and let $X \in \mathrm{L}^q(\Omega; V)$ be a zero mean random variable. Then for every finite sequence $(X_i)_{i=1}^M$ of independent, identically distributed random variables with zero mean and with $X_i \overset{D}{=} X$, we have*

$$\mathbb{E}\left[\|E_M[X]\|_V^q\right] = \mathbb{E}\left[\left\|\frac{1}{M}\sum_{i=1}^M X_i\right\|_V^q\right] \leq (2\kappa)^q M^{1-q} \mathbb{E}\left[\|X\|_V^q\right].$$

We can use Corollary 2.4 to derive a convergence rate of the Monte Carlo estimator in $\mathrm{L}^q(\Omega; \mathrm{L}^p(\mathbb{R}))$ for random variables in $\mathrm{L}^r(\Omega; \mathrm{L}^p(\mathbb{R}))$.

Theorem 2.5. *Let $1 \leq r, p \leq \infty$ and $X \in \mathrm{L}^r(\Omega; \mathrm{L}^p(\mathbb{R}))$, then for $q := \min\{2, p, r\}$ we have the Monte Carlo error estimate*

$$\|\mathbb{E}[X] - E_M[X]\|_{\mathrm{L}^q(\Omega; \mathrm{L}^p(\mathbb{R}))} \leq C M^{\frac{1-q}{q}} \|X\|_{\mathrm{L}^q(\Omega; \mathrm{L}^p(\mathbb{R}))}.$$

In particular, if $p, r > 1$ (and thus $q > 1$) the Monte Carlo estimator $E_M[X]$ converges towards $\mathbb{E}[X]$ in $\mathrm{L}^q(\Omega; \mathrm{L}^p(\mathbb{R}))$.

The proof of this theorem is an adaptation of Theorem 4.1 from [32].

Proof. We have

$$\begin{aligned}\|\mathbb{E}[X] - E_M[X]\|_{\mathrm{L}^q(\Omega; \mathrm{L}^p(\mathbb{R}))}^q &= \mathbb{E}\left[\left\|\mathbb{E}[X] - \frac{1}{M}\sum_{i=1}^M \hat{X}^i\right\|_{\mathrm{L}^p(\mathbb{R})}^q\right] \\ &= \mathbb{E}\left[\left\|\frac{1}{M}\sum_{i=1}^M \left(\mathbb{E}[X] - \hat{X}^i\right)\right\|_{\mathrm{L}^p(\mathbb{R})}^q\right].\end{aligned}$$

If we define $Y = \mathbb{E}[X] - X$ and $Y_i = \mathbb{E}[X] - \hat{X}^i$ we see that Y is in $\mathrm{L}^r(\Omega; \mathrm{L}^p(\mathbb{R}))$ with zero mean and Y_i are i.i.d. random variables with zero mean satisfying $Y_i \overset{D}{=} Y$. Therefore, we can apply Corollary 2.4 since $\mathrm{L}^r(\Omega; \mathrm{L}^p(\mathbb{R}))$ is of type $\min(2, r, p)$ and $\mathrm{L}^p(\mathbb{R})$ is of type $\min(2, p)$ and thus in particular also of type $\min(2, r, p)$. Hence,

$$\mathbb{E}\left[\left\|\frac{1}{M}\sum_{i=1}^M \left(\mathbb{E}[X] - \hat{X}^i\right)\right\|_{\mathrm{L}^p(\mathbb{R})}^q\right] \leq (2\kappa)^q M^{1-q} \mathbb{E}\left[\|\mathbb{E}[X] - X\|_{\mathrm{L}^p(\mathbb{R})}^q\right]$$

where κ is the type constant of $\mathrm{L}^p(\mathbb{R})$. It remains to show $\mathbb{E}\left[\|\mathbb{E}[X]-X\|^q_{\mathrm{L}^p(\mathbb{R})}\right] \leq C\mathbb{E}\left[\|X\|^q_{\mathrm{L}^p(\mathbb{R})}\right]$. This follows from standard estimates and Jensen's inequality in the following way:

$$\begin{aligned}\mathbb{E}\left[\|\mathbb{E}[X]-X\|^q_{\mathrm{L}^p(\mathbb{R})}\right] &\leq C\mathbb{E}\left[\|\mathbb{E}[X]\|^q_{\mathrm{L}^p(\mathbb{R})} + \|X\|^q_{\mathrm{L}^p(\mathbb{R})}\right] \\ &\leq C\left(\left(\mathbb{E}\left[\|X\|_{\mathrm{L}^p(\mathbb{R})}\right]\right)^q + \mathbb{E}\left[\|X\|^q_{\mathrm{L}^p(\mathbb{R})}\right]\right) \\ &\leq C\mathbb{E}\left[\|X\|^q_{\mathrm{L}^p(\mathbb{R})}\right].\end{aligned}$$

□

Note that Corollary 2.4 and Theorem 2.5 do not imply convergence if $q = 1$, *i.e.*, if r or p are equal to 1 in the latter case.

3. Deterministic conservation laws with discontinuous flux

In this section, we present the main existence and stability results for deterministic conservation laws with spatially discontinuous flux from [8, 46, 51].

We consider the Cauchy problem for conservation laws with discontinuous flux of the form

$$\begin{aligned} u_t + f(k(x),u)_x &= 0, \quad x\in\mathbb{R},\ t>0 \\ u(x,0) &= u_0(x), \quad x\in\mathbb{R}. \end{aligned} \tag{3.1}$$

Here, we require that f, k, and u_0 satisfy the following:

Assumption 3.1. *We assume that the flux $f\in\mathcal{C}^2(\mathbb{R}^2;\mathbb{R})$ is strictly monotone in u in the sense that $f_u \geq \alpha > 0$, and that $f(k^*,0) = 0$ for all $k^*\in\mathbb{R}$. Furthermore, we assume that k is piecewise constant with finitely many discontinuities and that the initial datum u_0 is in $(\mathrm{L}^\infty\cap\mathrm{BV})(\mathbb{R})$.*

In the deterministic setting, we consider entropy solutions in the following sense (*cf.* [6, 8]). For $p\in\mathbb{R}$ we define the function $c_p\colon\mathbb{R}\to\mathbb{R}$ through the equation

$$f(k(x),c_p(x)) = p, \qquad \text{for all } x\in\mathbb{R}.$$

Since $f_u\geq\alpha>0$ this equation has a unique solution for each $x\in\mathbb{R}$. Note that in the case of piecewise constant k the function c_p is piecewise constant as well.

Definition 3.2 (Entropy solution)**.** We say $u\in\mathcal{C}([0,T];\mathrm{L}^1(\mathbb{R}))\cap\mathrm{L}^\infty((0,T)\times\mathbb{R})$ is an entropy solution of (3.1) if

$$\begin{aligned}\int_0^T\int_{\mathbb{R}} (|u-c_p(x)|\varphi_t + \operatorname{sgn}(u-c_p(x))(f(k(x),u)-f(k(x),c_p(x)))\varphi_x)\,\mathrm{d}x\,\mathrm{d}t \\ -\int_{\mathbb{R}}|u(x,T)-c_p(x)|\varphi(x,T)\,\mathrm{d}x + \int_{\mathbb{R}}|u_0(x)-c_p(x)|\varphi(x,0)\,\mathrm{d}x \geq 0\end{aligned}$$

for all $p\in\mathbb{R}$ and for all nonnegative $\varphi\in\mathcal{C}_c^\infty(\mathbb{R}\times[0,T])$.

Note that a Rankine–Hugoniot-type argument shows that across a discontinuity ξ of k the entropy solution u satisfies the Rankine–Hugoniot condition

$$f(k(\xi-),u(\xi-,t)) = f(k(\xi+),u(\xi+,t)) \qquad \text{for almost every } t\in(0,T) \tag{3.2}$$

where $k(\xi\mp)$ and $u(\xi\mp,\cdot)$ denote the left and right traces of k respectively u both of which exist due to Remark 2.3 of [5]. In our subsequent analysis we will rely on the following two results concerning existence and stability of entropy solutions.

Theorem 3.3 (Existence and uniqueness of entropy solutions [7,8,51])**.** *Let f, k, and u_0 satisfy Assumption 3.1. Then there exists a unique entropy solution u of (3.1) which satisfies*

$$\begin{aligned} \|u(\cdot,t)\|_{\mathrm{L}^\infty(\mathbb{R})} &\le \frac{C_f}{\alpha}\|u_0\|_{\mathrm{L}^\infty(\mathbb{R})} \\ \mathrm{TV}(u(\cdot,t)) &\le C(\mathrm{TV}(k)+\mathrm{TV}(u_0)) \end{aligned} \tag{3.3}$$

for all $0 \le t \le T$ and

$$\mathrm{TV}_{[0,T]}(u(x,\cdot)) \le C\,\mathrm{TV}(u_0)$$

for all $x \in \mathbb{R}$. Here C_f denotes the maximal Lipschitz constant of f and α is as in Assumption 3.1.

Proof. The existence and uniqueness statement follows from the theory developed by Baiti and Jenssen [8]. The L^∞ and TV bounds follow from Theorem 1.4 of [51] and Lemma 4.6 of [7]. ☐

Theorem 3.4 (Stability of entropy solutions [46])**.** *Let f, k, and u_0 satisfy Assumption 3.1 and u be the corresponding entropy solution of (3.1). If v is the entropy solution of (3.1) with flux g, coefficient l, and initial datum v_0 satisfying Assumption 3.1 then for all $0 \le t \le T$*

$$\|u(\cdot,t)-v(\cdot,t)\|_{\mathrm{L}^1(\mathbb{R})} \le \|u_0-v_0\|_{\mathrm{L}^1(\mathbb{R})} + C\left(\|k-l\|_{\mathrm{L}^\infty(\mathbb{R})} + \|f_u-g_u\|_{\mathrm{L}^\infty(\mathbb{R}^2;\mathbb{R})}\right). \tag{3.4}$$

In particular, entropy solutions of (3.1) satisfy

$$\|u(\cdot,t)\|_{\mathrm{L}^1(\mathbb{R})} \le \|u_0\|_{\mathrm{L}^1(\mathbb{R})}$$

for all $0 \le t \le T$.

Proof. The stability estimate can be found in Theorem 4.1 of [46]. The L^1 bound follows from the stability estimate (3.4) by taking $g = f$, $l = k$, and $v_0 = 0$. ☐

Remark 3.5. We want to mention that the stability result from Theorem 3.4 is not only integral in proving existence and uniqueness of random entropy solutions, but can also be used to show well-posedness of Bayesian inverse problems for conservation laws with discontinuous flux [40].

4. Random conservation laws with discontinuous flux

We now consider conservation laws with discontinuous flux where the flux f, the coefficient k, and the initial datum u_0 in (3.1) are uncertain. To that end, we let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space and denote by $\mathcal{B}(X)$ the Borel σ-algebra on a space X. We define appropriate random data (u_0, k, f) in the following sense.

Definition 4.1 (Random data)**.** Given constants $C_{\mathrm{TV}}, C_f \in \mathbb{R}$, $\alpha \in (0,\infty)$, $N_k \in \mathbb{Z}$, $\delta > 0$ and given a rectangle $R = R_1 \times R_2 \subset \mathbb{R}^2$ let $\mathbb{D}$ be the Banach space

$$\mathbb{D} = (\mathrm{BV} \cap \mathrm{L}^\infty)(\mathbb{R}) \times \mathrm{L}^\infty(\mathbb{R}) \times \mathcal{C}^2(R;\mathbb{R})$$

endowed with the norm

$$\|(u_0,k,f)\|_{\mathbb{D}} = \|u_0\|_{\mathrm{L}^1(\mathbb{R})} + \mathrm{TV}(u_0) + \|u_0\|_{\mathrm{L}^\infty(\mathbb{R})} + \|k\|_{\mathrm{L}^\infty(\mathbb{R})} + \|f\|_{\mathcal{C}^2(R;\mathbb{R})}.$$

We say that a strongly measurable map $(u_0,k,f)\colon (\Omega,\mathcal{F}) \to (\mathbb{D},\mathcal{B}(\mathbb{D}))$ is called random data for (3.1) if for $\mathbb{P}$-*a.e.* ω

$$\begin{aligned} u_0(\omega;x) &\in R_1, && \text{for } a.e.\ x \in \mathbb{R}, \\ \mathrm{TV}(u_0) &\le C_{\mathrm{TV}} < \infty, \end{aligned}$$

$$\begin{aligned} k(\omega;x) &\in R_2, && \text{for } a.e.\ x \in \mathbb{R},\\ k(\omega;\cdot) \text{ is pcw. constant with at most } &N_k \text{ discontinuities}, &&\\ \text{each pair of discontinuities of } k &\text{ is at least } \delta \text{ apart}, &&\\ f_u(\omega,k,u) \geq \alpha > 0 \text{ and } &f(\omega;k,0)=0, && \text{for all } (k,u) \in R,\\ \|f(\omega;\cdot,\cdot)\|_{\mathcal{C}^2(R;\mathbb{R})} &\leq C_f < \infty && \end{aligned}$$

such that for $\mathbb{P}$-*a.e.* ω the data $(u_0(\omega), k(\omega), f(\omega))$ satisfy Assumption 3.1.

We are interested in random entropy solutions of the random conservation law

$$\begin{aligned} \frac{\partial u(\omega;x,t)}{\partial t} + \frac{\partial f(\omega;k(\omega;x),u(\omega;x,t))}{\partial x} &= 0, \quad \omega \in \Omega,\ x \in \mathbb{R},\ t>0,\\ u(\omega;x,0) &= u_0(\omega;x), \quad \omega \in \Omega,\ x \in \mathbb{R}. \end{aligned} \tag{4.1}$$

Definition 4.2 (Random entropy solution)**.** Given random data $(u_0,k,f)\colon \Omega \to \mathbb{D}$, we say that a random variable $u\colon \Omega \to \mathcal{C}([0,T];\mathrm{L}^1(\mathbb{R}))$ is a random entropy solution of (4.1) if u is strongly measurable and for $\mathbb{P}$-*a.e.* $\omega \in \Omega$ the function $u(\omega)$ satisfies

$$\begin{aligned} \int_0^T \int_{\mathbb{R}} (|u(\omega;x,t) - c_p(\omega;x)|\varphi_t + q(\omega;u(\omega;x,t)))\,\mathrm{d}x\,\mathrm{d}t \qquad&\\ - \int_{\mathbb{R}} |u(\omega;x,T) - c_p(\omega;x)|\varphi(x,T)\,\mathrm{d}x + \int_{\mathbb{R}} |u_0(\omega;x) - c_p(\omega;x)|\varphi(x,0)\,\mathrm{d}x &\geq 0 \end{aligned} \tag{4.2}$$

for all $p \in \mathbb{R}$ and nonnegative $\varphi \in \mathcal{C}_c^\infty(\mathbb{R}\times[0,T])$. Here we have used the notation

$$q(\omega;u(\omega;x,t)) = \operatorname{sgn}(u - c_p(\omega;x))(f(\omega;k(\omega;x),u) - f(\omega;k(\omega;x),c_p(\omega;x))).$$

We have the following existence and uniqueness result for random entropy solutions of conservation laws with discontinuous flux.

Theorem 4.3 (Existence and pathwise uniqueness of random entropy solutions)**.** *Let (u_0,k,f) be random data. Then there exists a unique random entropy solution $u\colon \Omega \to \mathcal{C}([0,T];\mathrm{L}^1(\mathbb{R}))$ to (4.1) which is pathwise unique, i.e., if the random data (u_0,k,f) and (v_0,l,g) are $\mathbb{P}$-versions of each other and u and v are corresponding random entropy solutions then u and v are $\mathbb{P}$-versions of each other.*

Proof. Let $S\colon \mathbb{D} \to \mathcal{C}([0,T];\mathrm{L}^1(\mathbb{R}))$ denote the solution operator from Theorem 3.3 that maps (deterministic) $(u_0,k,f) \in \mathbb{D}$ to the unique (deterministic) entropy solution $\hat{u} = S(u_0,k,f)$. Because of the stability estimate (3.4) this solution map is Lipschitz continuous. Now, since the random data $(u_0,k,f)\colon \Omega \to \mathbb{D}$ is strongly measurable the composition $S \circ (u_0,k,f)\colon \Omega \to \mathcal{C}([0,T];\mathrm{L}^1(\mathbb{R}))$ is again strongly measurable (see [53], Cor. 1.13). Hence $u = S \circ (u_0,k,f)$ is a strongly measurable map satisfying (4.2) $\mathbb{P}$-almost surely. Therefore, u is a random entropy solution to (4.1).

Regarding uniqueness of random entropy solutions, let (u_0,k,f) and (v_0,l,g) be $\mathbb{P}$-versions of each other, *i.e.*, $\|(u_0(\omega),k(\omega),f(\omega)) - (v_0(\omega),l(\omega),g(\omega))\|_{\mathbb{D}} = 0$ for $\mathbb{P}$-*a.e.* $\omega \in \Omega$, and u and v corresponding random entropy solutions. Then, the Lipschitz continuity of the solution operator S gives

$$\|u(\omega) - v(\omega)\|_{\mathcal{C}([0,T];\mathrm{L}^1(\mathbb{R}))} \leq C\|(u_0(\omega),k(\omega),f(\omega)) - (v_0(\omega),l(\omega),g(\omega))\|_{\mathbb{D}} = 0.$$

Thus, we have $u(\omega) = v(\omega)$ in $\mathcal{C}([0,T];\mathrm{L}^1(\mathbb{R}))$ for $\mathbb{P}$-*a.e.* $\omega \in \Omega$ which is pathwise uniqueness. □

Note that Theorem 4.3 generalizes the existence result of random entropy solutions of [41] for fluxes which are strictly monotone in u since the present setting allows for a discontinuous spatial dependency of the flux.

Remark 4.4. All existence and continuous dependence results stated so far apply to the deterministic Cauchy problem (3.1). By the usual arguments, verbatim the same results hold for entropy solutions on bounded intervals $D \subset \mathbb{R}$ as well, provided periodic boundary conditions are enforced.

The following probabilistic bound will be important in the numerical approximation of random entropy solutions on bounded domains.

Lemma 4.5. *Let* (u_0, k, f) *be random data and* $D \subset \mathbb{R}$ *a bounded interval. Let further* $u_0 \in \mathrm{L}^r(\Omega; \mathrm{L}^\infty(D))$*, for some* $1 \le r \le \infty$*. Then the random entropy solution* u *of* (4.1) *is in* $\mathrm{L}^r(\Omega; \mathcal{C}([0,T]; \mathrm{L}^p(D)))$ *for all* $1 \le p \le \infty$*. In particular,*

$$\|u(\cdot,t)\|_{\mathrm{L}^r(\Omega;\mathrm{L}^p(D))} \le C \, \|u_0\|_{\mathrm{L}^r(\Omega;\mathrm{L}^\infty(D))}$$

for all $0 \le t \le T$.

Proof. On bounded domains D we have

$$\|u(\cdot,t)\|_{\mathrm{L}^p(D)} \le |D|^{\frac{1}{p}} \, \|u(\cdot,t)\|_{\mathrm{L}^\infty(D)}$$

and thus using the L^∞-bound (3.3) we have for all $0 \le t \le T$

$$\begin{aligned}
\|u(\cdot,t)\|^r_{\mathrm{L}^r(\Omega;\mathrm{L}^p(D))} &= \int_\Omega \|u(\cdot,t)\|^r_{\mathrm{L}^p(D)} \,\mathrm{d}\mathbb{P} \\
&\le C \int_\Omega \|u(\cdot,t)\|^r_{\mathrm{L}^\infty(D)} \,\mathrm{d}\mathbb{P} \\
&\le C \int_\Omega \|u_0\|^r_{\mathrm{L}^\infty(D)} \,\mathrm{d}\mathbb{P} \\
&= C \, \|u_0\|^r_{\mathrm{L}^r(\Omega;\mathrm{L}^\infty(D))}
\end{aligned}$$

which proves the claim. □

5. Numerical approximation of random entropy solutions

In this section, we want to approximate the expectation $\mathbb{E}[u(\cdot,t)]$ of a random entropy solution u of the random conservation law with discontinuous flux (4.1). On the one hand, we will use the Monte Carlo and multilevel Monte Carlo method to approximate in the stochastic domain Ω. On the other hand, since in general exact solutions to (4.1) are not at hand, we will approximate in the physical domain $\mathbb{R} \times [0,T]$ by a finite volume method. To this end, we use a modified version of monotone finite volume methods for conservation laws introduced in [7] which appropriately addresses the presence of the discontinuous parameter k.

The resulting approximation error introduced by the Monte Carlo method depends on the number of samples used, while the error introduced by the finite volume method depends on the resolution of the grid. In the following subsections, we will review the finite volume method for the deterministic problem, detail how to combine it with the Monte Carlo and multilevel Monte Carlo method and prove error estimates for the resulting Monte Carlo and multilevel Monte Carlo finite volume method.

5.1. Finite volume methods for conservation laws with discontinuous flux

We will first consider the (deterministic) conservation law with discontinuous flux (3.1) and present a class of finite volume methods introduced in [7].

For a given (deterministic) function k with discontinuities $\xi_1 < \xi_2 < \ldots < \xi_N$ such that k satisfies the relevant assumptions in Definition 4.1 we denote by $D_i = (\xi_i, \xi_{i+1})$, $i = 0, \ldots, N$, the subdomains where k is constant. Here, we have used the notation $\xi_0 = -\infty$ and $\xi_{N+1} = +\infty$. In the following we will write

$$f^{(i)} = f(k(x), \cdot), \qquad \text{for } x \in D_i,\ i = 0, \ldots, N.$$

We discretize the domain $\mathbb{R} \times [0,T]$ using the spatial and temporal grid discretization parameters Δx and Δt. Here we assume that the spatial discretization parameter Δx is already small with respect to the given minimal distance δ between discontinuities of k, *i.e.*, $\Delta x < \delta$. In order to define the finite volume method we need the spatial grid to be aligned in such a way that all discontinuities of k lie on grid points. We acomplish that in the following way: To the left of ξ_1 and to the right of ξ_N we use a mesh of width Δx that is aligned with ξ_1 respectively ξ_N. Inside each interval $D_i = (\xi_i, \xi_{i+1})$ a mesh of the form $\{\xi_i + j\Delta x\}_{j=1}^{J}$ might not align with the point ξ_{i+1}. This happens precisely when $J\Delta x < \xi_{i+1} - \xi_i < (J+1)\Delta x$ in which case we set up the finer mesh $\{\xi_i + j\Delta x_i\}_{j=1}^{J+1}$ where $\Delta x_i = \frac{\xi_{i+1}-\xi_i}{J+1}$. Note that by definition we have

$$\frac{1}{2}\Delta x \le \Delta x_i \le \Delta x. \tag{5.1}$$

In this way we can set up a spatial grid that is globally non-uniform, but uniform on each subdomain D_i. We want to point out the important fact that while the local grid sizes Δx_i depend on the distance between neighboring discontinuities of k (which we will assume to be random later) the upper and lower bounds of Δx_i given by (5.1) are independ of k.

The resulting grid cells we denote by $\mathcal{C}_j = (x_{j-\frac{1}{2}}, x_{j+\frac{1}{2}})$ for points $x_{j+\frac{1}{2}}$, such that for $j \in \mathbb{Z}$ we have $x_{j+\frac{1}{2}} - x_{j-\frac{1}{2}} = \Delta x_i$ for some $i = 0, \dots, N$. Here we used the notation $\Delta x_0 = \Delta x_N = \Delta x$. The temporal grid cells we denote by $\mathcal{C}^n = [t^n, t^{n+1})$ where $t^n = n\Delta t$ for $n = 0, \dots, M+1$. Since the grid is aligned with the discontinuities of k we have $\xi_i = x_{P_i - \frac{1}{2}}$ for some integers P_i, $i = 1, \dots, N$.

We consider two-point numerical fluxes $F(u,v)$ that have the upwind property such that if $f' \ge 0$ (which is the setting of the present paper), we have $F(u,v) = f(v)$. This includes the upwind flux, the Godunov flux, and the Engquist–Osher flux. The finite volume method we consider is the following [7]:

$$\begin{aligned} u_j^0 &= \frac{1}{\Delta x}\int_{\mathcal{C}_j} u_0(x)\,\mathrm{d}x, \quad j \in \mathbb{Z}, \\ u_j^{n+1} &= u_j^n - \lambda_i\left(f^{(i)}\left(u_j^n\right) - f^{(i)}\left(u_{j-1}^n\right)\right), \quad n \ge 0,\ P_i < j < P_{i+1},\ 0 \le i \le N, \\ u_{P_i}^{n+1} &= \left(f^{(i)}\right)^{-1}\left(f^{(i-1)}\left(u_{P_i-1}^{n+1}\right)\right), \quad n \ge 0,\ 0 < i \le N, \end{aligned} \tag{5.2}$$

where $P_0 = -\infty$, $P_{N+1} = +\infty$, and $\lambda_i = \Delta t/\Delta x$. We assume that the grid discretization parameters satisfy the CFL condition

$$\max_i \max_u \left(f^{(i)}\right)'(u)\frac{\Delta t}{\Delta x} \le \frac{1}{2} \tag{5.3}$$

such that, in particular,

$$\max_i \max_u \left(f^{(i)}\right)'(u)\lambda_i \le 1.$$

Note that the last line of (5.2) represents a discrete version of the Rankine–Hugoniot condition (3.2). Here, we use the ghost cells $\mathcal{C}_{P_i}$, $i = 1, \dots, N$ to explicitly enforce the Rankine–Hugoniot condition on the discrete level.

With the sequence of cell averages $(u_j^n)_{j,n}$ we associate the piecewise constant function $u_{\Delta x}(x,t)$ given by

$$u_{\Delta x}(x,t) = u_j^n, \qquad (x,t) \in \mathcal{C}_j \times \mathcal{C}^n.$$

The following lemma shows that the finite volume method is stable in L^∞ and L^1.

Lemma 5.1 (Stability of the finite volume method)**.** *If the numerical scheme* (5.2) *satisfies the CFL condition* (5.3) *we have the following stability estimates:*

$$\|u_{\Delta x}(\cdot,t)\|_{\mathrm{L}^\infty(\mathbb{R})} \le \frac{C_f}{\alpha}\|u_0\|_{\mathrm{L}^\infty(\mathbb{R})} \tag{5.4}$$

and

$$\|u_{\Delta x}(\cdot,t)\|_{\mathrm{L}^1(\mathbb{R})} \leq \|u_0\|_{\mathrm{L}^1(\mathbb{R})} + C\mathrm{TV}(u_0)\Delta x.$$

Proof. (1) We first prove the L^∞-bound. To that end, we show by induction over $i = 0, \dots, N$ that

$$u_j^n \leq \max_{m=0,\dots,i} \sup_{l=P_m,\dots,P_{m+1}-1} \left(f^{(i)}\right)^{-1}\left(f^{(m)}\left(u_l^0\right)\right) \tag{5.5}$$

for all $j = P_i, \dots, P_{i+1}-1$ and $n = 0, \dots, M+1$. For $i = 0$, standard techniques for finite volume methods for conservation laws show

$$u_j^n \leq \max\left\{u_{j-1}^{n-1}, u_j^{n-1}\right\} \leq \dots \leq \sup_{l<P_1} u_l^0.$$

Assume now that (5.5) holds for some $i \in \{0, \dots, N-1\}$ and all $j = P_i, \dots, P_{i+1}-1$ and $n = 0, \dots, M+1$. Then we have for $j = P_{i+1}$

$$u_{P_{i+1}}^n = \left(f^{(i+1)}\right)^{-1}\left(f^{(i)}\left(u_{P_{i+1}-1}^n\right)\right) \leq \max_{m=0,\dots,i} \sup_{l=P_m,\dots,P_{m+1}-1} \left(f^{(i+1)}\right)^{-1}\left(f^{(m)}\left(u_l^0\right)\right).$$

On the other hand, for $j \in \{P_{i+1}+1, \dots, P_{i+2}-1\}$ we have as before

$$u_j^n \leq \max\left\{u_{j-1}^{n-1}, \dots, u_{j-1}^1, u_{j-1}^0, u_j^0\right\} \leq \dots \leq \max\left\{u_{P_{i+1}}^{n-(j-P_{i+1})}, \dots, u_{P_{i+1}}^1, u_{P_{i+1}}^0, \dots, u_j^0\right\}.$$

By combining both estimates, we obtain for $j \in \{P_{i+1}, \dots, P_{i+2}-1\}$

$$\begin{aligned} u_j^n &\leq \max\left\{\max_{l=P_{i+1},\dots,P_{i+2}-1} u_l^0, \max_{m=0,\dots,i} \sup_{l=P_m,\dots,P_{m+1}-1} \left(f^{(i+1)}\right)^{-1}\left(f^{(m)}\left(u_l^0\right)\right)\right\} \\ &= \max_{m=0,\dots,i+1} \sup_{l=P_m,\dots,P_{m+1}-1} \left(f^{(i+1)}\right)^{-1}\left(f^{(m)}\left(u_l^0\right)\right) \end{aligned}$$

which completes the induction. By taking absolute values in (5.5) we get for $j \in \mathbb{Z}$

$$|u_j^n| \leq \frac{1}{\alpha} \max_{i=0,\dots,N} \left\|f^{(i)}\right\|_{\mathrm{Lip}} \|u_0\|_{\mathrm{L}^\infty(\mathbb{R})}.$$

Taking the supremum over j yields the L^∞-bound (5.4).

(2) In order to prove the L^1-bound note that we have the discrete entropy inequalities

$$|u_j^{n+1} - c| - |u_j^n - c| + \lambda\left(q_j^{(i),n} - q_{j-1}^{(i),n}\right) \leq 0, \qquad i = 0, \dots, N,\ j = P_i+1, \dots, P_{i+1}-1$$

for all $c \in \mathbb{R}$ (see [7]). Here, we have denoted $q_j^{(i),n} = |f^{(i)}(u_j^n) - f^{(i)}(c)|$. Taking $c = 0$ and summing over $j \in \mathbb{Z} \setminus \{P_1, \dots, P_N\}$ yields

$$\sum_{j \neq P_i} |u_j^{n+1}| \leq \sum_{j \neq P_i} |u_j^n| - \lambda \sum_{i=0}^{N} \sum_{j=P_i+1}^{P_{i+1}-1} \left(q_j^{(i),n} - q_{j-1}^{(i),n}\right) = \sum_{j \neq P_i} |u_j^n|.$$

Therefore, we have

$$\sum_{j\in\mathbb{Z}} |u_j^{n+1}| \leq \sum_{j\in\mathbb{Z}} |u_j^n| + \sum_{i=1}^{N} \left(|u_{P_i}^{n+1}| - |u_{P_i}^n|\right) \leq \sum_{j\in\mathbb{Z}} |u_j^n| + \sum_{i=1}^{N} \frac{1}{\alpha} \left\|f^{(i-1)}\right\|_{\mathrm{Lip}} \left|u_{P_i-1}^{n+1} - u_{P_i-1}^n\right|$$

and hence

$$\sum_{j\in\mathbb{Z}} |u_j^{n+1}| \leq \sum_{j\in\mathbb{Z}} |u_j^0| + \sum_{i=0}^{N} \frac{1}{\alpha} \left\| f^{(i-1)} \right\|_{\mathrm{Lip}} \sum_{m=0}^{n} \left| u_{P_i-1}^{m+1} - u_{P_i-1}^{m} \right| .$$

In Lemma 4.6 of [7], it was shown that for all $i = 0, \dots, N$ we have

$$\sum_{m=0}^{n} \left| u_{P_i-1}^{m+1} - u_{P_i-1}^{m} \right| \leq C\mathrm{TV}(u_0)$$

which together with the foregoing estimate finally yields

$$\|u_{\Delta x}(\cdot, t)\|_{\mathrm{L}^1(\mathbb{R})} \leq \|u_0\|_{\mathrm{L}^1(\mathbb{R})} + C\mathrm{TV}(u_0)\Delta x.$$

□

In order to prove error estimates of the Monte Carlo and multilevel Monte Carlo finite volume method we will need the following convergence rate estimate which was proved in [7].

Theorem 5.2 (Convergence rate of the finite volume method [7])**.** *Let f, k, and u_0 satisfy Assumption 3.1 and the discretization parameters satisfy the CFL condition (5.3). Then the finite volume approximation $u_{\Delta x}$ given by the scheme (5.2) converges towards the unique entropy solution u of (4.1) almost everywhere and in* $\mathrm{L}^1(\mathbb{R} \times (0, T))$*. In particular, we have the following convergence rate estimate*

$$\|u(\cdot, t) - u_{\Delta x}(\cdot, t)\|_{\mathrm{L}^1(\mathbb{R})} \leq C\Delta x^{\frac{1}{2}} \tag{5.6}$$

for all $0 \leq t \leq T$.

Note that the convergence rate estimate (5.6) is optimal in the sense that the exponent $\frac{1}{2}$ cannot be improved without further assumptions on the initial datum [7] (see [47] for an overview of the literature regarding optimal convergence rates of finite volume methods for conservation laws without spatial dependency).

Remark 5.3. We want to point out that the constant C in (5.6) depends only on $\mathrm{TV}(u_0)$, $\|u_0\|_{\mathrm{L}^\infty}$, $\|f\|_\infty$, α and the number of discontinuities of k. In particular, for random data given according to Definition 4.1 all those quantities are uniformly bounded and thus for random entropy solutions the constant C in (5.6) is integrable in ω.

Remark 5.4. Reasoning as for entropy solutions, the finite volume approximation satisfies

$$\|u_{\Delta x}(\cdot, t)\|_{\mathrm{L}^p(D)} \leq |D|^{\frac{1}{p}} \|u_{\Delta x}(\cdot, t)\|_{\mathrm{L}^\infty(D)} \leq C \|u_0\|_{\mathrm{L}^\infty(D)}$$

for all $1 \leq p \leq \infty$. Like in Lemma 4.5, this translates into the following probabilistic bound:

$$\|u_{\Delta x}(\cdot, t)\|_{\mathrm{L}^r(\Omega;\mathrm{L}^p(D))} \leq C \|u_0\|_{\mathrm{L}^r(\Omega;\mathrm{L}^\infty(D))} \tag{5.7}$$

for all $0 \leq t \leq T$ and $1 \leq p \leq \infty$.

For the rest of this paper, we will consider entropy solutions on a bounded interval $D \subset \mathbb{R}$ with periodic boundary conditions. With the usual arguments, all previous results concerning entropy solutions and their finite volume approximations carry over to this setting verbatim. Note that restricting ourselves to a bounded domain will enable us to prove error estimates of the Monte Carlo and multilevel Monte Carlo finite volume method also in $\mathrm{L}^2(\Omega; \mathrm{L}^1(D))$ (*cf.* [45]).

5.2. Monte Carlo finite volume method

We now consider the random conservation law with discontinuous flux (4.1) and introduce and analyze the Monte Carlo finite volume method.

Given $M \in \mathbb{N}$, we generate M independent and identically distributed samples $(\hat{f}^i, \hat{k}^i, \hat{u}_0^i)_{i=1}^M$ of given random data (u_0, k, f). Let now $\hat{u}^i_{\Delta x}(\cdot, t)$, $i = 1, \dots, M$, denote the numerical solutions generated by the finite volume method (5.2) at time t corresponding to the sample $(\hat{f}^i, \hat{k}^i, \hat{u}_0^i)$. Then, the M-sample MCFVM approximation to $\mathbb{E}[u(\cdot, t)]$ is defined as

$$E_M[u_{\Delta x}(\cdot, t)] = \frac{1}{M}\sum_{i=1}^M \hat{u}^i_{\Delta x}(\cdot, t).$$

As mentioned earlier the approximation error of the MCFVM has a component coming from the statistical sampling error and one from the deterministic discretization error. We will make this statement precise in the following theorem.

Theorem 5.5 (MCFVM error estimate)**.** *Let (u_0, k, f) be random data and u the corresponding random entropy solution of (4.1). Assume that u_0 satisfies the r-th moment condition*

$$\|u_0\|_{\mathrm{L}^r(\Omega;\mathrm{L}^\infty(D))} < \infty$$

for some $1 < r \le \infty$. Assume further that we are given a FVM (5.2) such that the CFL condition (5.3) holds. Then, for each $1 \le p \le \infty$ and $0 \le t \le T$ and for $q = \min(2, r) > 1$, the MCFVM approximation satisfies the error estimate

$$\|\mathbb{E}[u(\cdot,t)] - E_M[u_{\Delta x}(\cdot,t)]\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))} \le C\left(M^{\frac{1-q}{q}}\|u_0\|_{\mathrm{L}^r(\Omega;\mathrm{L}^\infty(D))} + \|u_0\|^{1-\frac{1}{p}}_{\mathrm{L}^r(\Omega;\mathrm{L}^\infty(D))}\Delta x^{\frac{1}{2p}}\right). \tag{5.8}$$

In particular, the MCFVM approximation converges towards $\mathbb{E}[u(\cdot,t)]$ in $\mathrm{L}^q(\Omega;\mathrm{L}^p(D))$ as $M \to \infty$ and $\Delta x \to 0$.

Proof. We use the triangle inequality to get

$$\begin{aligned}\|\mathbb{E}[u(\cdot,t)] &- E_M[u_{\Delta x}(\cdot,t)]\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))}\\ &\le \|\mathbb{E}[u(\cdot,t)] - E_M[u(\cdot,t)]\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))} + \|E_M[u(\cdot,t)] - E_M[u_{\Delta x}(\cdot,t)]\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))}\end{aligned} \tag{5.9}$$

and estimate the resulting two terms separately. For the first term in (5.9), we distinguish the two cases $p \ge q$ and $p < q$.

(1) We first consider the case $p \ge q$. According to Lemma 4.5 we have

$$\|u(\cdot,t)\|_{\mathrm{L}^r(\Omega;\mathrm{L}^p(D))} \le C\,\|u_0\|_{\mathrm{L}^r(\Omega;\mathrm{L}^\infty(D))}$$

and thus $u(\cdot,t) \in \mathrm{L}^r(\Omega;\mathrm{L}^p(D))$. Therefore, we can apply Theorem 2.5 to get

$$\begin{aligned}\|\mathbb{E}[u(\cdot,t)] - E_M[u(\cdot,t)]\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))} &\le CM^{\frac{1-q}{q}}\|u(\cdot,t)\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))}\\ &\le CM^{\frac{1-q}{q}}\|u(\cdot,t)\|_{\mathrm{L}^r(\Omega;\mathrm{L}^p(D))}\\ &\le CM^{\frac{1-q}{q}}\|u_0\|_{\mathrm{L}^r(\Omega;\mathrm{L}^\infty(D))}.\end{aligned}$$

(2) In the case $p < q$, we can apply Hölder's inequality to estimate

$$\|\mathbb{E}[u(\cdot,t)] - E_M[u(\cdot,t)]\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))} \le C\,\|\mathbb{E}[u(\cdot,t)] - E_M[u(\cdot,t)]\|_{\mathrm{L}^q(\Omega;\mathrm{L}^q(D))}.$$

Again, we want to employ Theorem 2.5. To that end, we note that because of Lemma 4.5 and the fact that $q \le r$ we have

$$\|u(\cdot,t)\|_{\mathrm{L}^q(\Omega;\mathrm{L}^q(D))} \le C\,\|u_0\|_{\mathrm{L}^q(\Omega;\mathrm{L}^\infty(D))} \le C\,\|u_0\|_{\mathrm{L}^r(\Omega;\mathrm{L}^\infty(D))}$$

and therefore $u(\cdot,t) \in \mathrm{L}^q(\Omega;\mathrm{L}^q(D))$ and we can apply Theorem 2.5 to get

$$\|\mathbb{E}[u(\cdot,t)] - E_M[u(\cdot,t)]\|_{\mathrm{L}^q(\Omega;\mathrm{L}^q(D))} \le CM^{\frac{1-q}{q}}\,\|u(\cdot,t)\|_{\mathrm{L}^q(\Omega;\mathrm{L}^q(D))} \le CM^{\frac{1-q}{q}}\,\|u_0\|_{\mathrm{L}^r(\Omega;\mathrm{L}^\infty(D))}\,.$$

Hence, for all $1 \le p \le \infty$, we get

$$\|\mathbb{E}[u(\cdot,t)] - E_M[u(\cdot,t)]\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))} \le CM^{\frac{1-q}{q}}\,\|u_0\|_{\mathrm{L}^r(\Omega;\mathrm{L}^\infty(D))}\,.$$

On the other hand, for the second term in (5.9) we can use the triangle inequality and the linearity of the expected value to obtain

$$\begin{aligned}\|E_M[u(\cdot,t)] - E_M[u_{\Delta x}(\cdot,t)]\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))} &\le \frac{1}{M}\sum_{i=1}^{M}\left\|\hat{u}^i(\cdot,t) - \hat{u}^i_{\Delta x}(\cdot,t)\right\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))}\\ &= \|u(\cdot,t) - u_{\Delta x}(\cdot,t)\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))}\,.\end{aligned}$$

Using the interpolation inequality between L^1 and L^∞, the L^∞-bound for both $u(\cdot,t)$ and $u_{\Delta x}(\cdot,t)$ (see (3.3) respectively (5.4)), and the convergence rate estimate (5.6), we get

$$\begin{aligned}\|u(\cdot,t) - u_{\Delta x}(\cdot,t)\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))} &\le \|u(\cdot,t) - u_{\Delta x}(\cdot,t)\|^{\frac{1}{p}}_{\mathrm{L}^q(\Omega;\mathrm{L}^1(D))}\,\|u(\cdot,t) - u_{\Delta x}(\cdot,t)\|^{1-\frac{1}{p}}_{\mathrm{L}^q(\Omega;\mathrm{L}^\infty(D))}\\ &\le C\,\|u_0\|^{1-\frac{1}{p}}_{\mathrm{L}^r(\Omega;\mathrm{L}^\infty(D))}\,\Delta x^{\frac{1}{2p}},\end{aligned}$$

which completes the proof. □

Note that the computations in the proof of the error estimate (5.8) are also valid if $r = 1$ (and thus $q = 1$). However, in that case the right-hand side does not decrease as $M \to \infty$.

5.3. Multilevel Monte Carlo finite volume method

Instead of just considering Monte Carlo samples of a single fixed resolution of the finite volume method, we now detail the corresponding multilevel variant – the multilevel Monte Carlo finite volume method. The idea of MLMC discretization of differential equations with random parameters was proposed by Giles in [18,19] based upon earlier work by Heinrich on numerical quadrature [24]. The key ingredient of the MLMCFVM is simultaneous MC sampling on different levels of resolution of the finite volume method with level-dependent numbers M_l of MC samples.

To that end, we generate a sequence of finite volume approximations $U(\cdot,t) := (u_l(\cdot,t))_{l=0}^{L}$ on grids with cell sizes Δx_l and time steps Δt_l (subject to the CFL condition (5.3)) and set $u_{\Delta x_{-1}}(\cdot,t) = 0$. Then, we have

$$\mathbb{E}[u_{\Delta x_L}(\cdot,t)] = \mathbb{E}\left[\sum_{l=0}^{L}(u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t))\right] = \sum_{l=0}^{L}\mathbb{E}[u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)].$$

We now approximate each term $\mathbb{E}[u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)]$ by a Monte Carlo estimator with M_l samples. The resulting MLMCFVM approximation to $\mathbb{E}[u(\cdot,t)]$ then is

$$E^L[U(\cdot,t)] = \sum_{l=0}^{L} E_{M_l}\left[u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)\right]. \tag{5.10}$$

In the following convergence analysis, we will assume for simplicity that $\Delta x_l = 2^{-l}\Delta x_0$, $l = 0, \dots, L$, for some $\Delta x_0 > 0$.

As for the MCFVM, we want to obtain a rate at which $E^L[U(\cdot,t)]$ converges towards $\mathbb{E}[u(\cdot,t)]$ in terms of the number of MC samples M_l and the spatial resolution Δx_l on each level $l = 0, \dots, L$.

Theorem 5.6 (MLMCFVM error estimate)**.** *Let $L > 0$, (u_0, k, f) be random data, and u the corresponding random entropy solution of* (4.1)*. Assume that u_0 satisfies*

$$\|u_0\|_{\mathrm{L}^r(\Omega;\mathrm{L}^\infty(D))} < \infty$$

for some $1 < r \le \infty$. Assume further that we are given a FVM (5.2) *such that the CFL condition* (5.3) *holds. Then, for each $0 \le t \le T$, for any sequence $(M_l)_{l=0}^L$ of sample sizes at mesh level l the MLMCFVM approximation* (5.10) *satisfies the following error estimate for $q = \min(2, r) > 1$*

$$\begin{aligned}\left\|\mathbb{E}[u(\cdot,t)] - E^L[U(\cdot,t)]\right\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(\mathbb{R}))}\\ \le C\left(\|u_0\|^{1-\frac{1}{\widetilde{p}}}_{\mathrm{L}^1(\Omega;\mathrm{L}^\infty(D))}\Delta x_L^{\frac{1}{2p}} + \|u_0\|_{\mathrm{L}^q(\Omega;\mathrm{L}^\infty(D))} M_0^{\frac{1-q}{q}} + \|u_0\|^{1-\frac{1}{\widetilde{p}}}_{\mathrm{L}^q(\Omega;\mathrm{L}^\infty(D))}\sum_{l=0}^{L} M_l^{\frac{1-q}{q}}\Delta x_l^{\frac{1}{2\widetilde{p}}}\right)\end{aligned} \tag{5.11}$$

where $\widetilde{p} = \max(p, q)$. In particular, for fixed L the MLMCFVM approximation $E^L[U(\cdot,t)]$ converges towards $\mathbb{E}[u(\cdot,t)]$ in $\mathrm{L}^q(\Omega;\mathrm{L}^p(D))$ as $M_l \to \infty$ and $\Delta x_0 \to 0$.

Proof. Using the triangle inequality and the linearity of the expectation, we get

$$\begin{aligned}
&\|\mathbb{E}[u(\cdot,t)] - E^L[U(\cdot,t)]\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))}\\
&\quad\le \|\mathbb{E}[u(\cdot,t)] - \mathbb{E}[u_{\Delta x_L}(\cdot,t)]\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))} + \left\|\mathbb{E}[u_{\Delta x_L}(\cdot,t)] - E^L[U(\cdot,t)]\right\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))}\\
&\quad= \|\mathbb{E}[u(\cdot,t) - u_{\Delta x_L}(\cdot,t)]\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))}\\
&\qquad+ \left\|\sum_{l=0}^{L}\left(\mathbb{E}[u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)] - E_{M_l}[u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)]\right)\right\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))}\\
&\quad\le \|\mathbb{E}[u(\cdot,t) - u_{\Delta x_L}(\cdot,t)]\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))}\\
&\qquad+ \sum_{l=0}^{L}\left\|\mathbb{E}[u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)] - E_{M_l}[u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)]\right\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))}.
\end{aligned}$$

For the first term, note that the function $\mathbb{E}[u(\cdot,t) - u_{\Delta x_L}(\cdot,t)]$ is deterministic and thus we can use the convergence rate estimate (5.6) to get

$$\begin{aligned}
&\|\mathbb{E}[u(\cdot,t) - u_{\Delta x_L}(\cdot,t)]\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))}\\
&\quad\le \|u(\cdot,t) - u_{\Delta x_L}(\cdot,t)\|_{\mathrm{L}^1(\Omega;\mathrm{L}^p(D))}\\
&\quad\le \|u(\cdot,t) - u_{\Delta x_L}(\cdot,t)\|^{\frac{1}{p}}_{\mathrm{L}^1(\Omega;\mathrm{L}^1(D))}\|u(\cdot,t) - u_{\Delta x_L}(\cdot,t)\|^{1-\frac{1}{p}}_{\mathrm{L}^1(\Omega;\mathrm{L}^\infty(D))}\\
&\quad\le C\|u_0\|^{1-\frac{1}{p}}_{\mathrm{L}^1(\Omega;\mathrm{L}^\infty(D))}\Delta x_L^{\frac{1}{2p}}.
\end{aligned}$$

We now estimate the summands in the second term. Similarly to the proof of Theorem 5.5 we distinguish the two cases $p \ge q$ and $p < q$.

(1) We first consider the case $p \ge q$. Because of the triangle inequality and (5.7) we have

$$\left\|u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)\right\|_{\mathrm{L}^r(\Omega;\mathrm{L}^p(D))} \le C\|u_0\|_{\mathrm{L}^r(\Omega;\mathrm{L}^\infty(D))}$$

and thus $u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t) \in \mathrm{L}^r(\Omega;\mathrm{L}^p(D))$. Therefore we can apply Theorem 2.5 to get

$$\left\|\mathbb{E}[u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)] - E_{M_l}[u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)]\right\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))} \\ \le C M_l^{\frac{1-q}{q}} \left\|u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)\right\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))}.$$

(2) In the case $p < q$, we can apply Hölder's inequality to estimate

$$\left\|\mathbb{E}[u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)] - E_{M_l}[u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)]\right\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))} \\ \le C \left\|\mathbb{E}[u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)] - E_{M_l}[u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)]\right\|_{\mathrm{L}^q(\Omega;\mathrm{L}^q(D))}.$$

Following the same steps as in case (2) in the proof of Theorem 5.5 for $u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)$ instead of $u(\cdot,t)$ and using (5.7) instead of Lemma 4.5, we see that $u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t) \in \mathrm{L}^q(\Omega;\mathrm{L}^q(D))$. Thus, we can apply Theorem 2.5 again and get

$$\left\|\mathbb{E}[u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)] - E_{M_l}[u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)]\right\|_{\mathrm{L}^q(\Omega;\mathrm{L}^q(D))} \\ \le C M_l^{\frac{1-q}{q}} \left\|u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)\right\|_{\mathrm{L}^q(\Omega;\mathrm{L}^q(D))}.$$

Combining both cases, we get

$$\left\|\mathbb{E}[u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)] - E_{M_l}[u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)]\right\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))} \\ \le C M_l^{\frac{1-q}{q}} \left\|u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)\right\|_{\mathrm{L}^q(\Omega;\mathrm{L}^{\widetilde{p}}(D))}$$

where $\widetilde{p} = \max(p,q)$. Now, we can use the triangle inequality to get

$$\left\|u_{\Delta x_l}(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)\right\|_{\mathrm{L}^q(\Omega;\mathrm{L}^{\widetilde{p}}(D))} \le \|u_{\Delta x_l}(\cdot,t) - u(\cdot,t)\|_{\mathrm{L}^q(\Omega;\mathrm{L}^{\widetilde{p}}(D))} + \left\|u(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)\right\|_{\mathrm{L}^q(\Omega;\mathrm{L}^{\widetilde{p}}(D))}.$$

For $l > 0$, we can use the interpolation inequality between L^1 and L^∞, the L^1 and L^∞ bounds of the entropy solution and finite volume approximations (see (3.3) respectively (5.4)), and the convergence rate estimate (5.6) to get

$$\begin{aligned}
&\|u_{\Delta x_l}(\cdot,t) - u(\cdot,t)\|_{\mathrm{L}^q(\Omega;\mathrm{L}^{\widetilde{p}}(D))} + \left\|u(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)\right\|_{\mathrm{L}^q(\Omega;\mathrm{L}^{\widetilde{p}}(D))} \\
&\quad \le \|u_{\Delta x_l}(\cdot,t) - u(\cdot,t)\|_{\mathrm{L}^q(\Omega;\mathrm{L}^1(D))}^{\frac{1}{\widetilde{p}}} \|u_{\Delta x_l}(\cdot,t) - u(\cdot,t)\|_{\mathrm{L}^q(\Omega;\mathrm{L}^\infty(D))}^{1-\frac{1}{\widetilde{p}}} \\
&\qquad + \left\|u(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)\right\|_{\mathrm{L}^q(\Omega;\mathrm{L}^1(D))}^{\frac{1}{\widetilde{p}}} \left\|u(\cdot,t) - u_{\Delta x_{l-1}}(\cdot,t)\right\|_{\mathrm{L}^q(\Omega;\mathrm{L}^\infty(D))}^{1-\frac{1}{\widetilde{p}}} \\
&\quad \le C \|u_0\|_{\mathrm{L}^q(\Omega;\mathrm{L}^\infty(D))}^{1-\frac{1}{\widetilde{p}}} \left(\Delta x_l^{\frac{1}{2\widetilde{p}}} + \Delta x_{l-1}^{\frac{1}{2\widetilde{p}}}\right) \\
&\quad \le C \|u_0\|_{\mathrm{L}^q(\Omega;\mathrm{L}^\infty(D))}^{1-\frac{1}{\widetilde{p}}} \Delta x_l^{\frac{1}{2\widetilde{p}}}.
\end{aligned}$$

Similarly, for $l = 0$ (note that $u_{\Delta x_{-1}} = 0$), the convergence rate estimate (5.6) and the bound from Lemma 4.5 give

$$\begin{aligned}
\|u_{\Delta x_0}(\cdot,t)\|_{\mathrm{L}^q(\Omega;\mathrm{L}^{\widetilde{p}}(D))} &\le \|u_{\Delta x_0}(\cdot,t) - u(\cdot,t)\|_{\mathrm{L}^q(\Omega;\mathrm{L}^{\widetilde{p}}(D))} + \|u(\cdot,t)\|_{\mathrm{L}^q(\Omega;\mathrm{L}^{\widetilde{p}}(D))} \\
&\le C\left(\|u_0\|_{\mathrm{L}^q(\Omega;\mathrm{L}^\infty(D))}^{1-\frac{1}{\widetilde{p}}} \Delta x_0^{\frac{1}{2\widetilde{p}}} + \|u_0\|_{\mathrm{L}^q(\Omega;\mathrm{L}^\infty(D))}\right).
\end{aligned}$$

Combining all estimates finally gives

$$\begin{aligned}\left\|\mathbb{E}[u(\cdot,t)] - E^L[U(\cdot,t)]\right\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(\mathbb{R}))} \\ \leq C\left(\|u_0\|^{1-\frac{1}{p}}_{\mathrm{L}^1(\Omega;\mathrm{L}^\infty(D))}\Delta x_L^{\frac{1}{2p}} + \|u_0\|_{\mathrm{L}^q(\Omega;\mathrm{L}^\infty(D))} M_0^{\frac{1-q}{q}} + \|u_0\|^{1-\frac{1}{p}}_{\mathrm{L}^q(\Omega;\mathrm{L}^\infty(D))}\sum_{l=0}^{L} M_l^{\frac{1-q}{q}}\Delta x_l^{\frac{1}{2p}}\right).\end{aligned}$$

□

5.4. Work estimates and sample number optimization

In order to analyze the efficiency of the MC and MLMCFVM, it is important to estimate the computational work which is needed to compute one approximation of the solution by the deterministic FVM and how it increases with respect to mesh refinement. Here, by computational work, we understand the number of floating point operations performed when executing an algorithm and we assume that this in turn is proportional to the runtime of the algorithm.

In practice, we deal with bounded domains instead of working on the whole real line and thus the number of grid cells scales as $1/\Delta x$. For the deterministic FVM (5.2) the number of floating point operations per time step is proportional to the number of cells in the spatial domain, hence the computational work can be bounded by $C\Delta t^{-1}\Delta x^{-1}$. Considering the CFL condition (5.3), we thus obtain the computational work estimate

$$W^{\mathrm{FVM}}(\Delta x) \leq C\Delta x^{-2}$$

for the deterministic FVM approximation. However, for the sake of generality, we will in the following only assume that the computational work scales as

$$W^{\mathrm{FVM}}(\Delta x) \leq C\Delta x^{-w} \tag{5.12}$$

for some $w > 0$. As seen before, we have the L^p convergence rate estimate

$$\|u(\cdot,t) - u_{\Delta x}(\cdot,t)\|_{\mathrm{L}^p(D)} \leq C\Delta x^{\frac{s}{p}}$$

(for $s = \frac{1}{2}$) which yields the following deterministic convergence rate with respect to work:

$$\|u(\cdot,t) - u_{\Delta x}(\cdot,t)\|_{\mathrm{L}^p(D)} \leq C\left(W^{\mathrm{FVM}}\right)^{-\frac{s}{wp}}. \tag{5.13}$$

In particular, for $p = 1$, $w = 2$, and $s = \frac{1}{2}$ we have

$$\|u(\cdot,t) - u_{\Delta x}(\cdot,t)\|_{\mathrm{L}^1(D)} \leq C\left(W^{\mathrm{FVM}}\right)^{-\frac{1}{4}}.$$

5.4.1. Work estimates for the MCFVM approximation

Since for the Monte Carlo finite volume method M deterministic finite volume approximations need to be computed, each of which require work as in (5.12), the computational work for the MCFVM is bounded as

$$W_M^{\mathrm{MC}} \leq CM\Delta x^{-w}. \tag{5.14}$$

In order to obtain the order of convergence of the approximation error in terms of computational work, we equilibrate the terms $M^{\frac{1-q}{q}}$ and $\Delta x^{\frac{s}{p}}$ in (5.8) by choosing $M = C\Delta x^{\frac{sq}{p(1-q)}}$. Inserting this into the work bound (5.14) yields

$$W_M^{\mathrm{MCFVM}} \leq C\Delta x^{\frac{sq-wp(1-q)}{p(1-q)}}$$

such that we obtain from (5.8)

$$\|\mathbb{E}[u(\cdot,t)] - E_M[u_{\Delta x}(\cdot,t)]\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))} \le C\Delta x^{\frac{s}{p}} \le C\left(W_M^{\mathrm{MC}}\right)^{-\frac{s}{wp+s\frac{q}{q-1}}}. \tag{5.15}$$

Note that, since $q/(q-1)$ is positive, we have

$$\frac{s}{wp+s\frac{q}{q-1}} \le \frac{s}{wp}$$

and thus the rate (5.15) is worse than the error rate in terms of computational work (5.13) of the deterministic finite volume method.

In particular, for $p=1$ and $r\ge 2$ (which implies $q=2$), and taking into account that $w=2$ and $s=\frac{1}{2}$, the rate (5.15) reads

$$\|\mathbb{E}[u(\cdot,t)] - E_M[u_{\Delta x}(\cdot,t)]\|_{\mathrm{L}^2(\Omega;\mathrm{L}^1(D))} \le C\left(W_M^{\mathrm{MC}}\right)^{-\frac{1}{6}}.$$

5.4.2. Optimal sample numbers for the MLMCFVM approximation

In [32], Koley *et al.* showed the following general result for multilevel Monte Carlo finite volume methods which we can apply to our case to determine the number of samples needed at each level l such that, given an error tolerance $\varepsilon>0$, the computational work of the MLMCFVM is minimal.

Lemma 5.7 ([32], Lem. 4.9)**.** *Assume that the work of a multilevel Monte Carlo finite volume method with L discretization levels scales asymptotically as*

$$W_L^{\mathrm{MLMC}} = C\sum_{l=0}^{L} M_l \Delta x_l^{-w}$$

for some $w>0$ and that the approximation error (raised to the q-th power) scales as

$$\mathrm{Err}_L = C\left(\Delta x_L^{\frac{sq}{p}} + M_0^{1-q} + \sum_{l=0}^{L} M_l^{1-q}\Delta x_l^{\frac{sq}{\widetilde{p}}}\right)$$

where $\widetilde{p}=\max(p,q)$ (cf. (5.11)). Then, given an error tolerance $\varepsilon>0$, the optimal sample numbers M_l minimizing the computational work given the error tolerance ε are given by

$$M_0 \simeq \left(\frac{1+\Delta x_0^{\frac{s}{\widetilde{p}}}\sum_{l=1}^{L} 2^{l\left(w\frac{q-1}{q}-\frac{s}{\widetilde{p}}\right)}}{\varepsilon - \Delta x_L^{\frac{sq}{p}}}\right)^{\frac{1}{q-1}} \tag{5.16}$$

and

$$M_l \simeq M_0 \Delta x_0^{\frac{s}{\widetilde{p}}} 2^{-l\left(\frac{s}{\widetilde{p}}+\frac{w}{q}\right)}, \qquad \text{for } l>0, \tag{5.17}$$

where $\simeq$ indicates that this is the number of samples up to a constant which is independent of l and L. The minimal amount of work then is

$$W_L^{\mathrm{MLMC}} \simeq \Delta x_0^{-w}\left(1+\Delta x_0^{\frac{s}{\widetilde{p}}}\sum_{l=1}^{L} 2^{l\left(w\frac{q-1}{q}-\frac{s}{\widetilde{p}}\right)}\right)\left(\frac{1+\Delta x_0^{\frac{s}{\widetilde{p}}}\sum_{l=1}^{L} 2^{l\left(w\frac{q-1}{q}-\frac{s}{\widetilde{p}}\right)}}{\varepsilon - \Delta x_0^{\frac{sq}{p}} 2^{-L\frac{qs}{p}}}\right)^{\frac{1}{q-1}}.$$

Lemma 5.7 can be used to derive a rate for the approximation error of the MLMCFVM in terms of the computational work.

Corollary 5.8. *In addition to the assumptions of Lemma 5.7, assume that* $w\frac{q-1}{q} - \frac{s}{\widetilde{p}} > 0$ *and that* L *and* Δx_0 *are large enough such that*

$$\Delta x_L^{\frac{s}{p}\frac{q}{q-1}-w} > \Delta x_0^{-w}$$

where $\widetilde{p} = \max(p,q)$ *and* w *is as in* (5.7)*. Then, for each* $0 \le t \le T$ *and for* $q = \min(2,r)$ *the* $\mathrm{L}^q(\Omega;\mathrm{L}^p(D))$*-approximation error of the MLMCFVM* (5.10) *scales with respect to computational work as*

$$\left\|\mathbb{E}[u(\cdot,t)] - E^L[U(\cdot,t)]\right\|_{\mathrm{L}^q(\Omega;\mathrm{L}^p(D))} \le C\left(W_L^{\mathrm{MLMC}}\right)^{-\frac{s}{wp+s\frac{\widetilde{p}-p}{\widetilde{p}}\frac{q}{q-1}}}. \tag{5.18}$$

Proof. Since $\left(w\frac{q-1}{q} - \frac{s}{\widetilde{p}}\right) > 0$ the sums in the expression for W_M^{MLMC} from Lemma 5.7 are dominated by $2^{L\left(w\frac{q-1}{q}-\frac{s}{\widetilde{p}}\right)}$. Choosing $\varepsilon = 2\Delta x_L^{\frac{sq}{p}}$ and using that $\Delta x_L^{\frac{s}{p}\frac{q}{q-1}-w} > \Delta x_0^{-w}$ in the last step, we find

$$\begin{aligned}
W_L^{\mathrm{MLMC}} &\simeq \Delta x_0^{-w}\left(1+\Delta x_0^{\frac{s}{\widetilde{p}}}2^{L\left(w\frac{q-1}{q}-\frac{s}{\widetilde{p}}\right)}\right)\left(\frac{1+\Delta x_0^{\frac{s}{\widetilde{p}}}2^{L\left(w\frac{q-1}{q}-\frac{s}{\widetilde{p}}\right)}}{\Delta x_L^{\frac{sq}{p}}}\right)^{\frac{1}{q-1}}\\
&\simeq \Delta x_0^{-w}\Delta x_L^{-\frac{sq}{p(q-1)}}\left(1+\Delta x_0^{\frac{s}{\widetilde{p}}}2^{L\left(w\frac{q-1}{q}-\frac{s}{\widetilde{p}}\right)}\right)^{\frac{q}{q-1}}\\
&\simeq \Delta x_L^{-\frac{sq}{p(q-1)}}\left(\Delta x_0^{-w}+\Delta x_L^{\frac{s}{\widetilde{p}}\frac{q}{q-1}-w}\right)\\
&\simeq \Delta x_L^{s\left(\frac{1}{\widetilde{p}}-\frac{1}{p}\right)\frac{q}{q-1}-w}.
\end{aligned}$$

Thus, we have

$$\left\|\mathbb{E}[u(\cdot,t)] - E^L[U(\cdot,t)]\right\|_{\mathrm{L}^2(\Omega;\mathrm{L}^1(D))} = \varepsilon^{\frac{1}{q}} \simeq \Delta x_L^{\frac{s}{p}} \simeq \left(W_L^{\mathrm{MLMC}}\right)^{-\frac{s}{wp+s\frac{\widetilde{p}-p}{\widetilde{p}}\frac{q}{q-1}}}.$$

□

Since $\frac{(\widetilde{p}-p)}{\widetilde{p}}$ and $\frac{q}{(q-1)}$ are nonnegative, we have

$$\frac{s}{wp+s\frac{\widetilde{p}-p}{\widetilde{p}}\frac{q}{q-1}} \le \frac{s}{wp}$$

and thus the error rate in terms of the computational work (5.18) of the MLMCFVM is worse than the error rate (5.13) for the deterministic scheme. However, since $\frac{\widetilde{p}-p}{\widetilde{p}} \le 1 - \frac{p}{q} \le 1$, we have

$$\frac{s}{wp+s\frac{\widetilde{p}-p}{\widetilde{p}}\frac{q}{q-1}} \ge \frac{s}{wp+s\frac{q}{q-1}}$$

and thus the error rate (5.18) of the MLMCFVM constitutes an improvement over the (single-level) MCFVM, *cf.* (5.15).

Note that, in particular, for $p = 1$ and $r \ge 2$ (which implies $q = 2$ and $\widetilde{p} = 2$), and taking into account that $w = 2$ and $s = \frac{1}{2}$, the error rate (5.18) reads

$$\left\|\mathbb{E}[u(\cdot,t)] - E^L[U(\cdot,t)]\right\|_{\mathrm{L}^2(\Omega;\mathrm{L}^1(D))} \le C\left(W_L^{\mathrm{MLMC}}\right)^{-\frac{1}{5}}.$$

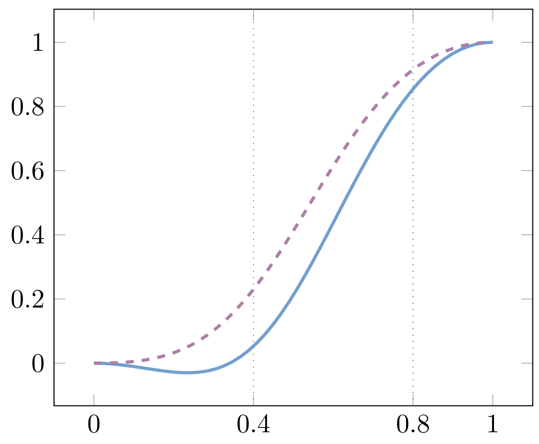


FIGURE 1. Two possible fluxes of the form (6.1) for $k(x) = 0.7$ (dashed line) and $k(x) = 2.3$ (solid line)

6. Numerical experiments

In this section, we present numerical experiments motivated by two-phase flow in a heterogeneous porous medium[1]. The time evolution of the oil saturation $u \in [0,1]$ can be modeled by (1.1) where the flux is given by

$$f(k(x),u) = \frac{\lambda_o(u)}{\lambda_o(u)+\lambda_w(u)}(1-k(x)\lambda_w(u)), \tag{6.1}$$

see Example 8.2 of [25]. Here, the functions λ_o and λ_w denote the phase mobilities/relative permeabilities of the oil and the water phase, respectively. Typically, one uses the simple expressions

$$\lambda_o(u) = u^2, \qquad \lambda_w(u) = (1-u)^2$$

which we will also do in the first two subsequent experiments. The coefficient k in (6.1) corresponds to the absolute permeability of the medium. Since the medium is usually layered to some extent throughout the reservoir and even continuously varying geology is typically mapped onto some grid, the coefficient k is often modeled as a piecewise constant function [23].

Since numerical experiments for conservation laws where the initial datum or the flux is uncertain have been reported in other works (albeit without spatially discontinuous flux), we will here focus on numerical experiments where, in particular, the discontinuous coefficient k is subject to randomness. We consider the initial datum

$$u_0(x) = \begin{cases} 0.8, & -0.9 < x < -0.2, \\ 0.4, & \text{otherwise,} \end{cases} \tag{6.2}$$

on the spatial domain $D = [-1,1]$ with periodic boundary conditions. Figure 1 shows two examples of fluxes of the form (6.1) and indicates the relevant domain determined by the initial datum (6.2). In all experiments we use $\lambda = \frac{\Delta t}{\Delta x} = 0.2$ in the finite volume approximation (5.2).

When choosing the number of samples for the MLMC estimator we use the formulae (5.16) and (5.17) with "=" replacing "$\simeq$" and rounding to the next biggest integer. Here we use $p = 1$, $r = q = 2$, $w = 2$, $s = \frac{1}{2}$, and $\varepsilon = 2\Delta x_L^{2s}$ in (5.16) and (5.17)[2].

[1]The code used to produce these experiments can be found at `https://github.com/adrianmruf/MLMC_discontinuous_flux`

[2]For example, for $L = 7$ and $\Delta x_0 = 2^{-4}$ we use $(M_l)_{l=0}^L = (95\,646, 20\,107, 8\,454, 3\,555, 1\,495, 629, 265, 112)$ samples.

In order to compute an estimate of the approximation error

$$\left\|\mathbb{E}[u(\cdot,T)] - E^L[U(\cdot,T)]\right\|_{\mathrm{L}^2(\Omega;\mathrm{L}^1(D))} = \left(\mathbb{E}\left[\left\|\mathbb{E}[u(\cdot,T)] - E^L[U(\cdot,T)]\right\|^2_{\mathrm{L}^1(D)}\right]\right)^{\frac{1}{2}}$$

we use the root mean square estimator introduced in [38]: We denote by $U_{\mathrm{ref}}(\cdot,T)$ a reference solution and by $(U_i(\cdot,T))_{i=1}^K$ a sequence of independent approximate solutions $E^L[U(\cdot,T)]$ obtained by running the MLMCFVM estimator with L levels K times. Then, we estimate the relative error by

$$\mathcal{RMS} = \left(\frac{1}{K}\sum_{i=1}^{K}(\mathcal{RMS}_i)^2\right)^{\frac{1}{2}}$$

where

$$\mathcal{RMS}_i = 100 \times \frac{\|U_{\mathrm{ref}}(\cdot,T) - U_i(\cdot,T)\|_{\mathrm{L}^1(D)}}{\|U_{\mathrm{ref}}(\cdot,T)\|_{\mathrm{L}^1(D)}}.$$

Here, as suggested in [38], we use $K = 30$ which was shown to be sufficient for most problems. In each experiment, as a reference approximation $U_{\mathrm{ref}}(\cdot,T)$ of $\mathbb{E}[u(\cdot,T)]$, we use a solution computed by the MLMCFVM with $\Delta x_0 = 2^{-4}$ and $L = 8$ which entails using 2^{13} cells on the finest level.

In our figures we also indicate the approximated standard deviation. To that end, we approximate the variance by

$$V_L = \sum_{l=0}^{L} E_{M_l}\left[(u_{\Delta x_l}(\cdot,T) - u_{\Delta x_{l-1}}(\cdot,T) - E_{M_l}[u_{\Delta x_l}(\cdot,T) - u_{\Delta x_{l-1}}(\cdot,T)])^2\right].$$

6.1. Uncertain position of rock layer interface

For our first numerical experiment we will model the absolute permeability parameter as

$$k(x) = \begin{cases} 1, & x < \sigma(\omega), \\ 2, & x > \sigma(\omega) \end{cases}$$

corresponding to an uncertain position of the interface between two rock types in the reservoir. Here, the random variable σ is uniformly distributed in $[-0.3, 0.3]$. Figure 2a shows two samples of the approximate random entropy solution (with $\sigma = -0.3$ and $\sigma = 0.3$, respectively) calculated using 2^{10} grid points at time $T = 0.2$ and Figure 2b shows an estimate of the expectation $\mathbb{E}[u(\cdot,T)]$ computed by the MLMCFVM with $\Delta x_0 = 2^{-4}$ and $L = 7$.

Table 1 and Figure 3 show the estimated $\mathcal{RMS}$ error as a function of the number of levels. In particular, Table 1a shows the observed order of convergence (OOC) with respect to Δx_L while Table 1b shows the observed order of convergence with respect to the computational work calculated based on a best linear fit under the assumptions that $\mathcal{RMS} \sim (\Delta x_L)^{r_1}$ and $\mathcal{RMS} \sim (\mathrm{work})^{r_2}$. Here, we use the runtime as a surrogate for the computational work. We observe that in Experiment 1 both rates are better than the rates guaranteed by our convergence analysis.

6.2. Uncertain absolute permeabilities

For our second numerical experiment we will model the absolute permeability parameter as

$$k(x) = \begin{cases} \xi_1(\omega), & x < 0, \\ \xi_2(\omega), & x > 0 \end{cases}$$

corresponding to uncertain absolute permeabilities of two rock layers. Here, the random variables ξ_1 and ξ_2 are independent and uniformly distributed in $[0.7, 1.3]$ and $[1.7, 2.3]$, respectively.

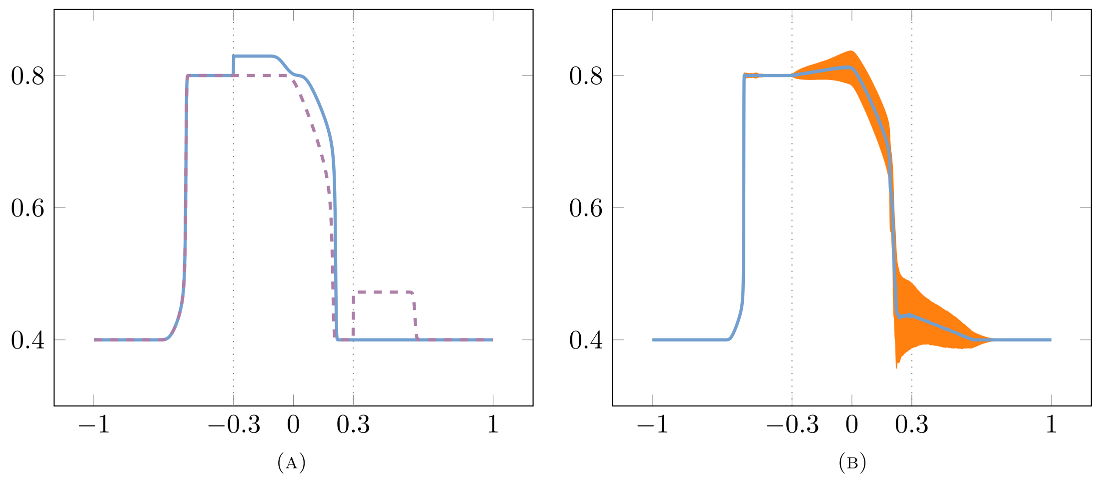


FIGURE 2. Two samples and a MLMCFVM approximation of the (mean of the) random entropy solution for Experiment 1 with $T = 0.2$ and $\lambda = 0.2$. The orange area indicates the area between the mean ± standard deviation. For each sample the discontinuity of k is located in the interval between the dotted lines. (A) Two samples of the random entropy solution ($\sigma = -0.3$ (solid line), $\sigma = 0.3$ (dashed line), $\Delta x = 2^{-9}$). (B) MLMCFVM approximation ($\Delta x_0 = 2^{-4}$, $L = 7$).

TABLE 1. $\mathcal{RMS}$ error in Experiment 1 as a function of the finest grid resolution Δx_L and as a function of the work (here measured by the runtime in s) for various values of L and for $\Delta x_0 = 2^{-4}$.

(a) $\mathcal{RMS}$ *versus* Δx_L

L	Δx_L	$\mathcal{RMS}$	OOC
1	2^{-5}	4.03	
2	2^{-6}	2.53	
3	2^{-7}	1.53	
4	2^{-8}	0.88	
5	2^{-9}	0.49	
6	2^{-10}	0.24	0.80

(b) $\mathcal{RMS}$ *versus* work.

L	Runtime	$\mathcal{RMS}$	OOC
1	0.09	4.03	
2	0.23	2.53	
3	0.73	1.53	
4	2.65	0.88	
5	10.12	0.49	
6	39.23	0.24	−0.45

Figure 4a shows two samples of the approximate random entropy solution (with $(\xi_1, \xi_2) = (1.3, 1.7)$ and $(\xi_1, \xi_2) = (0.7, 2.3)$, respectively) calculated using 2^{10} grid points at time $T = 0.2$ and Figure 4b shows an estimate of the expectation $\mathbb{E}[u(\cdot, T)]$ computed by the MLMCFVM with $\Delta x_0 = 2^{-4}$ and $L = 7$.

Table 2 and Figure 5 again show the root mean square error estimate and the observed order of convergence with respect to Δx_L and with respect to the computational work. As before, we observe that the observed convergence rates are better than the theoretical bounds.

6.3. Uncertain position of rock layer interface and absolute and relative permeabilities

In our last numerical experiment we will model the absolute permeability parameter as

$$k(x) = \begin{cases} \xi_1(\omega), & x < \sigma(\omega), \\ \xi_2(\omega), & x > \sigma(\omega) \end{cases}$$

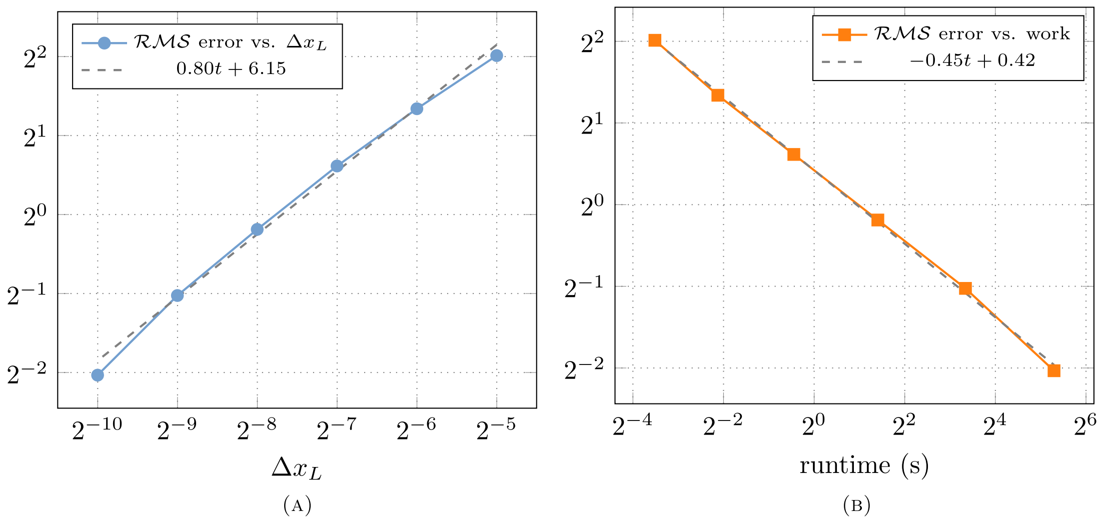


FIGURE 3. $\mathcal{RMS}$ error in Experiment 1 as a function of the finest grid resolution Δx_L and as a function of the work (here measured by the runtime in s) corresponding to the values in Table 1. The dashed lines indicate the observed order of convergence based on a best linear fit. (A) $\mathcal{RMS}$ error *versus* Δx_L. (B) $\mathcal{RMS}$ error *versus* work.

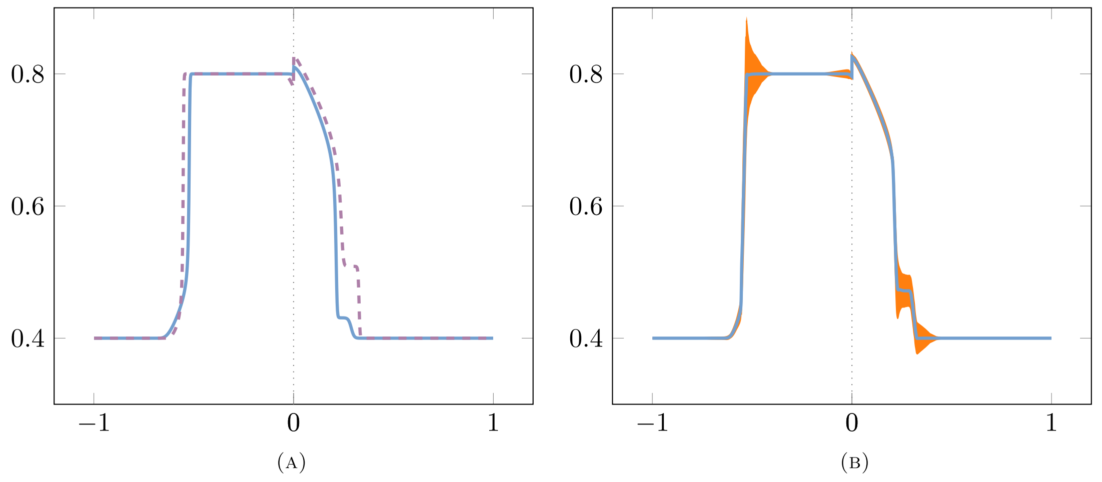


FIGURE 4. Two samples and a MLMCFVM approximation of the (mean of the) random entropy solution for Experiment 2 with $T = 0.2$ and $\lambda = 0.2$. The orange area indicates the area between the mean $\pm$ standard deviation and the dotted line marks the (fixed) position of the discontinuity of k. (A) Two samples of the random entropy solution ($(\xi_1, \xi_2) = (1.3, 1.7)$ (solid line), $(\xi_1, \xi_2) = (0.7, 2.3)$ (dashed line), $\Delta x = 2^{-9}$). (B) MLMCFVM approximation ($\Delta x_0 = 2^{-4}$, $L = 7$).

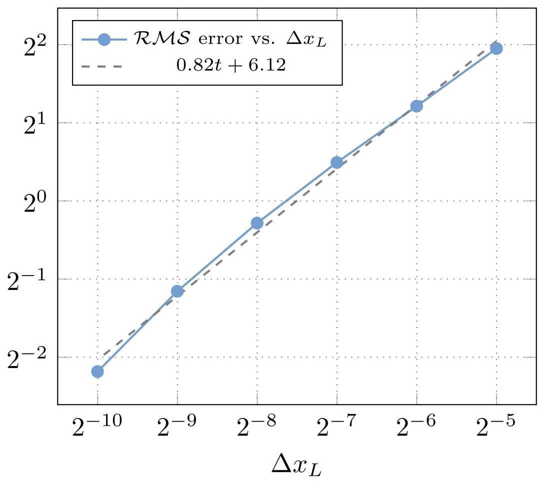
TABLE 2. $\mathcal{RMS}$ error in Experiment 2 as a function of the finest grid resolution Δx_L and as a function of the work (here measured by the runtime in s) for various values of L and for $\Delta x_0 = 2^{-4}$.

(a) $\mathcal{RMS}$ *versus* Δx_L.

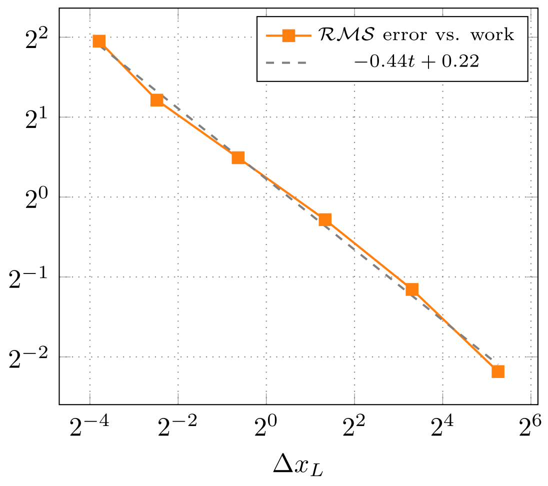
L	Δx_L	$\mathcal{RMS}$	OOC
1	2^{-5}	3.86	
2	2^{-6}	2.32	
3	2^{-7}	1.41	
4	2^{-8}	0.82	
5	2^{-9}	0.45	
6	2^{-10}	0.22	0.82

(b) $\mathcal{RMS}$ *versus* work.

L	Runtime (s)	$\mathcal{RMS}$	OOC
1	0.07	3.86	
2	0.18	2.32	
3	0.64	1.41	
4	2.52	0.82	
5	9.88	0.45	
6	38.16	0.22	−0.44



(A)



(B)

FIGURE 5. $\mathcal{RMS}$ error in Experiment 2 as a function of the finest grid solution Δx_L and as a function of the work (here measured by the runtime in s) corresponding to the values in Table 2. The dotted lines indicate the observed order of convergence based on a best linear fit. (A) $\mathcal{RMS}$ error *versus* Δx_L. (B) $\mathcal{RMS}$ error *versus* work.

corresponding to an uncertain position of the interface between two rock types as well as uncertain absolute permeabilities of the rock layers. Here, the random variables ξ_1, ξ_2, and σ are uniformly distributed in $[0.7, 1.3]$, $[1.7, 2.3]$ and $[-0.3, 0.3]$, respectively. Furthermore, we will model the relative permeabilities λ_o and λ_w in (6.1) as

$$\lambda_o(u) = u^{p(\omega)}, \qquad \lambda_w(u) = (1-u)^{p(\omega)}$$

where the random exponent p is uniformly distributed in $[1.5, 2.5]$. Here, ξ_1, ξ_2, σ and p are mutually independent.

Figure 6a shows two samples of the approximate random entropy solution (with $(\xi_1, \xi_2, \sigma, p) = (0.3, -0.3, -0.3, 1.5)$ and $(\xi_1, \xi_2, \sigma, p) = (-0.3, 0.3, 0.3, 2.5)$, respectively) calculated using 2^{10} grid points at time $T = 0.2$ and Figure 6b shows an estimate of the expectation $\mathbb{E}[u(\cdot, T)]$ computed by the MLMCFVM with $\Delta x_0 = 2^{-4}$ and $L = 7$.

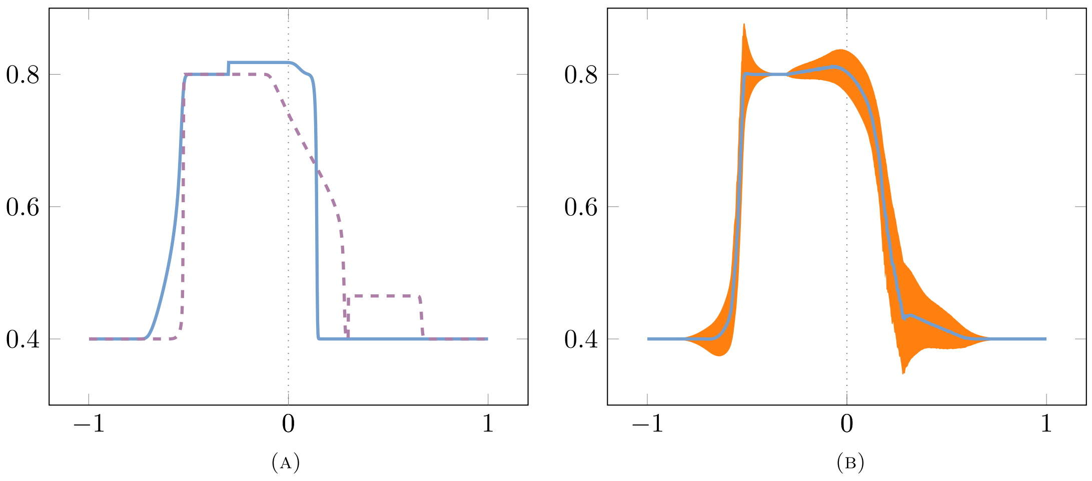


FIGURE 6. Two samples and a MLMCFVM approximation of the (mean of the) random entropy solution for Experiment 3 with $T = 0.2$ and $\lambda = 0.2$. The orange area indicates the area between the mean $\pm$ standard deviation and the dotted line marks the (fixed) position of the discontinuity of k. (A) Two samples of the random entropy solution $((\xi_1, \xi_2, \sigma, p) = (1.3, 1.7, -0.3, 1.5)$ (solid line), $(\xi_1, \xi_2, \sigma, p) = (0.7, 2.3, 0.3, 2.5)$ (dashed line), $\Delta x = 2^{-9})$. (B) MLMCFVM approximation ($\Delta x_0 = 2^{-4}$, $L = 7$).

TABLE 3. $\mathcal{RMS}$ error in Experiment 3 as a function of the finest grid resolution Δx_L and as a function of the work (here measured by the runtime in s) for various values of L and for $\Delta x_0 = 2^{-4}$.

(a) $\mathcal{RMS}$ *versus* Δx_L.

L	Δx_L	$\mathcal{RMS}$	OOC
1	2^{-5}	3.44	
2	2^{-6}	1.97	
3	2^{-7}	1.14	
4	2^{-8}	0.63	
5	2^{-9}	0.33	
6	2^{-10}	0.17	0.86

(b) $\mathcal{RMS}$ *versus* work.

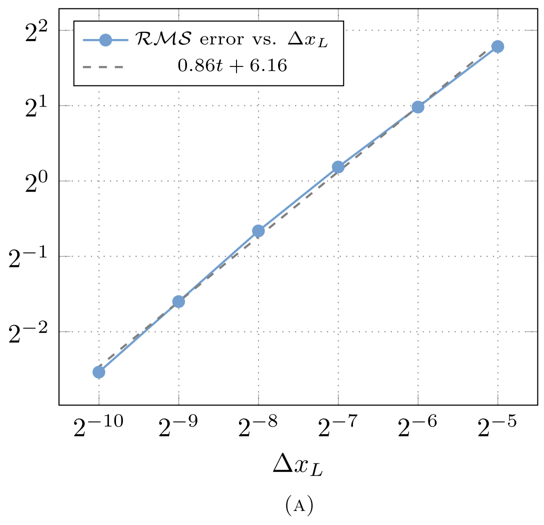
L	Runtime (s)	$\mathcal{RMS}$	OOC
1	0.08	3.44	
2	0.21	1.97	
3	0.74	1.14	
4	3.07	0.63	
5	12.93	0.33	
6	55.93	0.17	−0.45

Table 3 and Figure 7 again show the root mean square error estimate and the observed order of convergence with respect to Δx_L and with respect to the computational work. Notably, the observed convergence rates are very similar to those in Experiments 1 and 2 despite the four dimensional parameter space.

7. CONCLUSION

In this paper, we have considered conservation laws with discontinuous flux where the model parameters, *i.e.*, the initial datum, the flux function, and the discontinuous spatial dependency coefficient, are uncertain. Based on adapted entropy solutions for the deterministic case, we have introduced a notion of random entropy solutions and have proved well-posedness.

To numerically approximate the mean of a random entropy solution, we have proposed Monte Carlo methods coupled with a class of finite volume methods suited for conservation laws with discontinuous flux. Our



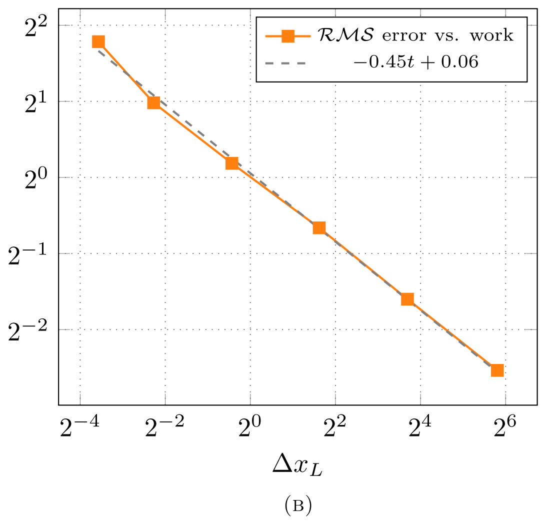


FIGURE 7. $\mathcal{RMS}$ error in Experiment 3 as a function of the finest grid solution Δx_L and as a function of the work (here measured by the runtime in s) corresponding to the values in Table 3. The dotted lines indicate the observed order of convergence based on a best linear fit. (A) $\mathcal{RMS}$ error *versus* Δx_L. (B) $\mathcal{RMS}$ error *versus* work.

convergence analysis includes convergence rate estimates for the Monte Carlo and multilevel Monte Carlo finite volume method. Further, we have provided error *versus* work rates which show that the multilevel Monte Carlo finite volume method is more efficient than the (single-level) Monte Carlo finite volume method.

We have presented numerical experiments motivated by two-phase flow in heterogeneous porous media, *e.g.*, oil reservoirs with different rock layers. The numerical experiments verify our theoretical results concerning convergence rates of the multilevel Monte Carlo finite volume method.

As a possible direction of future research, we want to mention that – from a practical standpoint – it would be desirable to design multilevel Monte Carlo finite volume methods based on finite volume methods that require no processing of the flux discontinuities. Such numerical methods have been considered in [15, 51], however, there are currently no convergence rate results available for these methods.

Acknowledgements. The work of JB was supported by German Priority Programme 1648 (SPPEXA) and the ModCompShock EU Project. The work of NHR was performed while visiting the University of Würzburg supported by the Giovanni-Prodi Chair Position. JB, CK, and NHR were supported by DAAD (German Academic Exchange Service) and the Research Council of Norway.

References

[1] R. Aae Klausen and N.H. Risebro, Stability of conservation laws with discontinuous coefficients. *J. Differ. Equ.* **157** (1999) 41–60.

[2] R. Abgrall, A simple, flexible and generic deterministic approach to uncertainty quantifications in non linear problems: application to fluid flow problems (2008).

[3] S.M. Adimurthi and G.V. Gowda, Conservation law with the flux function discontinuous in the space variable – II: convex–concave type fluxes and generalized entropy solutions. *J. Comput. Appl. Math.* **203** (2007) 310–344.

[4] S.M. Adimurthi and G.V. Gowda, Optimal entropy solutions for conservation laws with discontinuous flux-functions. *J. Hyperbolic Differ. Equ.* **2** (2005) 783–837.

[5] B. Andreianov, K.H. Karlsen and N.H. Risebro, A theory of L^1-dissipative solvers for scalar conservation laws with discontinuous flux. *Arch. Ration. Mech. Anal.* **201** (2011) 27–86.

[6] E. Audusse and B. Perthame, Uniqueness for scalar conservation laws with discontinuous flux via adapted entropies. *Proc. R. Soc. Edinburgh Sect. A: Math.* **135** (2005) 253–265.

[7] J. Badwaik and A.M. Ruf, Convergence rates of monotone schemes for conservation laws with discontinuous flux. *SIAM J. Numer. Anal.* **58** (2020) 607–629.

[8] P. Baiti and H.K. Jenssen, Well-posedness for a class of 2×2 L^∞ data. *J. Differ. Equ.* **140** (1997) 161–185.

[9] R. Bürger, K. Karlsen, C. Klingenberg and N. Risebro, A front tracking approach to a model of continuous sedimentation in ideal clarifier–thickener units. *Nonlinear Anal.: Real World App.* **4** (2003) 457–481.

[10] R. Bürger, K.H. Karlsen and J.D. Towers, An Engquist–Osher-type scheme for conservation laws with discontinuous flux adapted to flux connections. *SIAM J. Numer. Anal.* **47** (2009) 1684–1712.

[11] Q.-Y. Chen, D. Gottlieb and J.S. Hesthaven, Uncertainty analysis for the steady-state flows in a dual throat nozzle. *J. Comput. Phys.* **204** (2005) 378–398.

[12] G.M. Coclite and N.H. Risebro, Conservation laws with time dependent discontinuous coefficients. *SIAM J. Math. Anal.* **36** (2005) 1293–1309.

[13] S. Cox, M. Hutzenthaler, A. Jentzen, J. van Neerven and T. Welti, Convergence in Hölder norms with applications to Monte Carlo methods in infinite dimensions. *IMA J. Numer. Anal.* **41** (2021) 493–548.

[14] S. Diehl, A conservation law with point source and discontinuous flux function modelling continuous sedimentation. *SIAM J. Appl. Math.* **56** (1996) 388–419.

[15] S.S. Ghoshal, A. Jana and J.D. Towers, Convergence of a Godunov scheme to an Audusse–Perthame adapted entropy solution for conservation laws with BV spatial flux. *Numer. Math.* **146** (2020) 629–659.

[16] S.S. Ghoshal, J.D. Towers and G. Vaidya, Convergence of a Godunov scheme for degenerate conservation laws with BV spatial flux and a study of Panov type fluxes. Preprint: arXiv:2011.10946 (2020).

[17] S.S. Ghoshal, J.D. Towers and G. Vaidya, Well-posedness for conservation laws with spatial heterogeneities and a study of BV regularity. Preprint: arXiv:2010.13695 (2020).

[18] M. Giles, Improved multilevel Monte Carlo convergence using the Milstein scheme. In: Monte Carlo and Quasi-Monte Carlo Methods 2006. Springer (2008) 343–358.

[19] M.B. Giles, Multilevel Monte Carlo path simulation. *Oper. Res.* **56** (2008) 607–617.

[20] T. Gimse, Conservation laws with discontinuous flux functions. *SIAM J. Math. Anal.* **24** (1993) 279–289.

[21] T. Gimse and N.H. Risebro, Riemann problems with a discontinuous flux function. In: Vol. 1 of *Proceedings of Third International Conference on Hyperbolic Problems* (1991) 488–502.

[22] T. Gimse and N.H. Risebro, Solution of the Cauchy problem for a conservation law with a discontinuous flux function. *SIAM J. Math. Anal.* **23** (1992) 635–648.

[23] T. Gimse and N.H. Risebro, A note on reservoir simulation for heterogeneous porous media. *Transp. Porous Media* **10** (1993) 257–270.

[24] S. Heinrich, Multilevel Monte Carlo methods. In: International Conference on Large-Scale Scientific Computing. Springer (2001) 58–67.

[25] H. Holden and N.H. Risebro, Front Tracking for Hyperbolic Conservation Laws. Springer **152** (2015).

[26] K.H. Karlsen and J.D. Towers, Convergence of the Lax-Friedrichs scheme and stability for conservation laws with a discontinuous space-time dependent flux. *Chin. Ann. Math.* **25** (2004) 287–318.

[27] K.H. Karlsen and J.D. Towers, Convergence of a Godunov scheme for conservation laws with a discontinuous flux lacking the crossing condition. *J. Hyperbolic Differ. Equ.* **14** (2017) 671–701.

[28] K. Karlsen, N. Risebro and J. Towers, Upwind difference approximations for degenerate parabolic convection–diffusion equations with a discontinuous coefficient. *IMA J. Numer. Anal.* **22** (2002) 623–664.

[29] K.H. Karlsen, N.H. Risebro and J.D. Towers, L^1 stability for entropy solutions of nonlinear degenerate parabolic convection-diffusion equations with discontinuous coefficients. Preprint Series. *Pure Mathematics* http://urn.nb.no/URN:NBN:no-8076 (2003).

[30] C. Klingenberg and N.H. Risebro, Convex conservation laws with discontinuous coefficients. Existence, uniqueness and asymptotic behavior. *Commun. Part. Differ. Equ.* **20** (1995) 1959–1990.

[31] C. Klingenberg and N.H. Risebro, Stability of a resonant system of conservation laws modeling polymer flow with gravitation. *J. Differ. Equ.* **170** (2001) 344–380.

[32] U. Koley, N.H. Risebro, C. Schwab and F. Weber, A multilevel Monte Carlo finite difference method for random scalar degenerate convection–diffusion equations. *J. Hyperbolic Differ. Equ.* **14** (2017) 415–454.

[33] S.N. Kružkov, First order quasilinear equations in several independent variables. *Math. USSR-Sbornik* **10** (1970) 217–243.

[34] M. Ledoux and M. Talagrand, Probability in Banach Spaces: Isoperimetry and Processes. Springer Science & Business Media (2013).

[35] M.J. Lighthill and G.B. Whitham, On kinematic waves II. A theory of traffic flow on long crowded roads. *Proc. R. Soc. London. Ser. A. Math. Phys. Sci.* **229** (1955) 317–345.

[36] G. Lin, C. Su and G. Karniadakis, The stochastic piston problem. *Proc. Nat. Acad. Sci. USA* **101** (2004) 15840–15845.

[37] S. Mishra, Convergence of upwind finite difference schemes for a scalar conservation law with indefinite discontinuities in the flux function. *SIAM J. Numer. Anal.* **43** (2005) 559–577.

[38] S. Mishra and C. Schwab, Sparse tensor multi-level Monte Carlo finite volume methods for hyperbolic conservation laws with random initial data. *Math. Comput.* **81** (2012) 1979–2018.

[39] S. Mishra, C. Schwab and J. Šukys, Multi-level Monte Carlo finite volume methods for uncertainty quantification in nonlinear systems of balance laws. In: Uncertainty Quantification in Computational Fluid Dynamics. Springer (2013) 225–294.

[40] S. Mishra, D. Ochsner, A.M. Ruf and F. Weber, Bayesian inverse problems in the Wasserstein distance and application to conservation laws. in preparation (2021).

[41] S. Mishra, N.H. Risebro, C. Schwab and S. Tokareva, Numerical solution of scalar conservation laws with random flux functions. *SIAM/ASA J. Uncertainty Quant.* **4** (2016) 552–591.

[42] B. Piccoli and M. Tournus, A general BV existence result for conservation laws with spatial heterogeneities. *SIAM J. Math. Anal.* **50** (2018) 2901–2927.

[43] G. Poëtte, B. Després and D. Lucor, Uncertainty quantification for systems of conservation laws. *J. Comput. Phys.* **228** (2009) 2443–2467.

[44] N.H. Risebro and A. Tveito, Front tracking applied to a nonstrictly hyperbolic system of conservation laws. *SIAM J. Sci. Stat. Comput.* **12** (1991) 1401–1419.

[45] N.H. Risebro, C. Schwab and F. Weber, Correction to: Multilevel Monte Carlo front-tracking for random scalar conservation laws. *BIT Numer. Math.* **58** (2018) 247–255.

[46] A.M. Ruf, Flux-stability for conservation laws with discontinuous flux and convergence rates of the front tracking method. *IMA J. Numer. Anal.* **101** (2021) draa101.

[47] A.M. Ruf, E. Sande and S. Solem, The optimal convergence rate of monotone schemes for conservation laws in the Wasserstein distance. *J. Sci. Comput.* **80** (2019) 1764–1776.

[48] W. Shen, On the uniqueness of vanishing viscosity solutions for riemann problems for polymer flooding. *Nonlinear Differ. Equ. App. NoDEA* **24** (2017) 37.

[49] J. Towers, Convergence of a difference scheme for conservation laws with a discontinuous flux. *SIAM J. Numer. Anal.* **38** (2000) 681–698.

[50] J.D. Towers, A difference scheme for conservation laws with a discontinuous flux: the nonconvex case. *SIAM J. Numer. Anal.* **39** (2001) 1197–1218.

[51] J.D. Towers, An existence result for conservation laws having BV spatial flux heterogeneities – without concavity. *J. Differ. Equ.* **269** (2020) 5754–5764.

[52] J. Tryoen, O. Le Maitre, M. Ndjinga and A. Ern, Intrusive Galerkin methods with upwinding for uncertain nonlinear hyperbolic systems. *J. Comput. Phys.* **229** (2010) 6485–6511.

[53] J. Van Neerven, Stochastic evolution equations. ISEM Lecture Notes (2008).

[54] X. Wan and G.E. Karniadakis, Long-term behavior of polynomial chaos in stochastic flow simulations. *Comput. Methods Appl. Mech. Eng.* **195** (2006) 5582–5596.

[55] X. Wen and S. Jin, Convergence of an immersed interface upwind scheme for linear advection equations with piecewise constant coefficients I: L^1-error estimates. *J. Comput. Math.* **26** (2008) 1–22.

Single-Step ALE-DG Methods for the 1-D Euler Equations

Communications on Applied Mathematics and Computation
https://doi.org/10.1007/s42967-019-00054-5

ORIGINAL PAPER

Single-Step Arbitrary Lagrangian–Eulerian Discontinuous Galerkin Method for 1-D Euler Equations

Jayesh Badwaik[1] · **Praveen Chandrashekar**[2] · **Christian Klingenberg**[1]

Received: 6 March 2019 / Revised: 17 September 2019 / Accepted: 31 October 2019

Abstract
We propose an explicit, single-step discontinuous Galerkin method on moving grids using the arbitrary Lagrangian–Eulerian approach for one-dimensional Euler equations. The grid is moved with the local fluid velocity modified by some smoothing, which is found to considerably reduce the numerical dissipation introduced by Riemann solvers. The scheme preserves constant states for any mesh motion and we also study its positivity preservation property. Local grid refinement and coarsening are performed to maintain the mesh quality and avoid the appearance of very small or large cells. Second, higher order methods are developed and several test cases are provided to demonstrate the accuracy of the proposed scheme.

Keywords Discontinuous Galerkin method · Moving meshes · Arbitrary Lagrangian–Eulerian · Euler equations

Mathematics Subject Classification 65M60 · 35L04

1 Introduction

Finite volume schemes based on exact or approximate Riemann solvers are used for solving hyperbolic conservation laws like the Euler equations governing compressible flows. These schemes are able to compute discontinuous solutions in a stable manner since they have implicit dissipation built into them due to the upwind nature of the schemes. Higher

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s42967-019-00054-5) contains supplementary material, which is available to authorized users.

Christian Klingenberg
klingenberg@mathematik.uni-wuerzburg.de

Jayesh Badwaik
badwaik.jayesh@gmail.com

Praveen Chandrashekar
praveen@math.tifrbng.res.in

[1] Department of Mathematics, University of Würzburg, Würzburg, Germany

[2] TIFR Center for Applicable Mathematics, Bangalore, India

order schemes are constructed following a reconstruction approach combined with a high order time integration scheme. Discontinuous Galerkin methods can be considered as higher order generalizations of finite volume methods which also make use of Riemann solver technology but do not need a reconstruction step since they evolve a polynomial solution inside each cell. While these methods are formally high order accurate on smooth solutions, they can still introduce too much numerical dissipation in some situations. Springel [1] gives the example of a Kelvin–Helmholtz instability in which adding a large constant velocity to both states leads to suppression of the instability due to excessive numerical dissipation. This behaviour is attributed to the fact that fixed grid methods based on upwind schemes are not Galilean invariant. Upwind schemes, even when they are formally high order accurate, are found to be too dissipative when applied to turbulent flows [2] since the numerical viscosity can overwhelm the physical viscosity.

For the linear convection equation $u_t + au_x = 0$, the first-order upwind scheme has the modified partial differential equation

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = \frac{1}{2}|a|h(1-\nu)\frac{\partial^2 u}{\partial x^2} + O(h^2), \qquad \nu = \frac{|a|\Delta t}{h},$$

which shows that the numerical dissipation is proportional to $|a|$ which is the wave speed. In case of Euler equations simulated with a Riemann solver, e.g., the Roe scheme, the wave speeds are related to the eigenvalues of the flux Jacobian and the numerical dissipation would be proportional to the absolute values of the eigenvalues, e.g., $|v-c|, |v|, |v+c|$ where v is the fluid velocity and c is the sound speed. This type of the numerical viscosity is not Galilean invariant since the fluid velocity depends on the coordinate frame adopted for the description of the flow. Adding a large translational velocity to the coordinate frame will increase the numerical viscosity and reduce the accuracy of the numerical solution. Such high numerical viscosity can be eliminated or minimized if the grid moves along with the flow as in Lagrangian methods [3–5]. However, pure Lagrangian methods encounter the issue of large grid deformations that occur in highly sheared flows as in the Kelvin–Helmholtz problem requiring some form of re-meshing. A related approach is to use the arbitrary Lagrangian–Eulerian approach [6, 7] where the mesh velocity can be chosen to be close to the local fluid velocity but may be regularized to maintain the mesh quality. Even in the ALE approach, it may be necessary to perform some local remeshing to prevent the grid quality from degrading. In [1], the mesh is regenerated after every time step based on a Delaunay triangulation, which allows it to maintain good mesh quality even when the fluid undergoes large shear deformation. However, these methods have been restricted to second-order accuracy as they rely on unstructured finite volume schemes on general polygonal/polyhedral cells, where achieving higher order accuracy is much more difficult compared to structured grids.

Traditionally, ALE methods have been used for problems involving moving boundaries as in wing flutter, store separation and other problems involving fluid structure interaction [8–12]. In these applications, the main reason to use ALE is not to minimize the dissipation in upwind schemes but to account for the moving boundaries and, hence, the grid velocities are chosen based on boundary motion and with a view to maintain good mesh quality. Another class of methods solves the PDE on moving meshes where the mesh motion is determined based on a monitor function which is designed to detect regions of large gradients in the solution, see [13, 14] and the references therein. These methods achieve automatic clustering of grid points in regions of large gradients. ALE schemes have been used to compute multi-material flows as in [15], since they are useful to accurately track the material interface.

The mesh velocity was chosen to be equal to the contact speed but away from the material contact, the velocity was chosen by linear interpolation and was not close to Lagrangian. There are other methods for choosing the mesh velocity which have been studied in [16, 17]. Lax–Wendroff type ALE schemes for compressible flows have been developed in [18]. Finite volume schemes based on the ADER approach have been developed on unstructured grids [19–21]. The theoretical analysis of ALE-DG schemes in the framework of Runge–Kutta time stepping for conservation laws has been done in [22].

In the present work, we consider only the one-dimensional problem to set down the fundamental principles with which in an upcoming work, we shall solve the multi-dimensional problem. The numerical method developed here will be usable in the multiple dimensions, but additional work is required in multiple dimensions to maintain a good mesh quality under fluid flow deformation. We develop an explicit discontinuous Galerkin scheme that is conservative on moving meshes and automatically satisfies the geometric conservation law. The scheme is a single-step method which is achieved using a predictor computed from a Runge–Kutta scheme that is local to each cell in the sense that it does not require any data from neighbouring cells and belongs to the class of schemes called the ADER method. Due to the single-step nature of the scheme, the TVD limiter has to be applied only once in each time step unlike in multi-stage Runge–Kutta schemes where the limiter is applied after each stage update. This nature of the ADER scheme can reduce its computational expense, especially in multi-dimensional problems and while performing parallel computations. The mesh velocity is specified at each cell face as the local velocity with some smoothing. We analyze the positivity of the first-order scheme using the Rusanov flux and derive a CFL condition. The scheme is shown to be exact for steady moving contact waves and the solutions are invariant to the motion of the coordinate frame. Due to the Lagrangian nature, the Roe scheme does not require any entropy fix. However, we identify the possibility of spurious contact waves arising in some situations. This is due to the vanishing of the eigenvalue corresponding to the contact wave. While the cell averages are well predicted, the higher moments of the solution can be inaccurate. This behaviour of Lagrangian DG schemes does not seem to have been reported in the literature. We propose a fix for the eigenvalue in the spirit of the entropy fix of Harten [23] that prevents the spurious contact waves from occurring in the solution. The methodology developed here will be extended to multi-dimensional flows in a future work with a view towards handling complex sheared flows.

The rest of the paper is organized as follows. Section 2 introduces the Euler equation model that is used in the rest of the paper. In Sect. 3, we explain the derivation of the scheme on a moving mesh together with the quadrature approximations and computation of mesh velocity. The computation of the predicted solution is detailed in Sect. 4. The TVD type limiter is presented in Sect. 5 for a non-uniform mesh, Sect. 6 shows the positivity of the first-order scheme and Sect. 7 shows the preservation of constant states. The grid coarsening and refinement strategy are explained in Sect. 8, while Sect. 9 presents a series of numerical results.

2 Euler Equations

The Euler equations model the conservation of mass, momentum and energy, and can be written as a system of coupled partial differential equations laws of the form

$$\frac{\partial \boldsymbol{u}}{\partial t} + \frac{\partial \boldsymbol{f}(\boldsymbol{u})}{\partial x} = 0, \tag{1}$$

where $\boldsymbol{u}$ is called the vector of conserved variables and $\boldsymbol{f}(\boldsymbol{u})$ is the corresponding flux given by

$$\boldsymbol{u} = \begin{bmatrix} \rho \\ \rho v \\ E \end{bmatrix}, \qquad \boldsymbol{f}(\boldsymbol{u}) = \begin{bmatrix} \rho v \\ p + \rho v^2 \\ \rho H v \end{bmatrix}.$$

In the above expressions, ρ is the density, v is the fluid velocity, p is the pressure and E is the total energy per unit volume, which for an ideal gas is given by $E = p/(\gamma - 1) + \rho v^2/2$, with $\gamma > 1$ being the ratio of specific heats at constant pressure and volume, and $H = (E + p)/\rho$ is the enthalpy. The Euler equations form a hyperbolic system; the flux Jacobian $A(\boldsymbol{u}) = \boldsymbol{f}'(\boldsymbol{u})$ has real eigenvalues and linearly independent eigenvectors. The eigenvalues are $v - c,\ v,\ v + c$ where $c = \sqrt{\gamma p/\rho}$ is the speed of sound and the corresponding right eigenvectors are given by

$$r_1 = \begin{bmatrix} 1 \\ v - c \\ H - vc \end{bmatrix}, \qquad r_2 = \begin{bmatrix} 1 \\ v \\ \frac{1}{2}v^2 \end{bmatrix}, \qquad r_3 = \begin{bmatrix} 1 \\ v + c \\ H + vc \end{bmatrix}. \tag{2}$$

The hyperbolic property implies that A can be diagonalized as $A = R\Lambda R^{-1}$ where R is the matrix formed by the right eigenvectors as the columns and Λ is the diagonal matrix of eigenvalues.

3 Discontinuous Galerkin Method

3.1 Mesh and Solution Space

Consider a partition of the domain into disjoint cells with the jth cell being denoted by $C_j(t) = [x_{j-\frac{1}{2}}(t), x_{j+\frac{1}{2}}(t)]$. As the notation shows, the cell boundaries are time dependent which means that the cell is moving in some specified manner. The time levels are denoted by t_n with the time step $\Delta t_n = t_{n+1} - t_n$. The boundaries of the cells move with a constant velocity in the time interval (t_n, t_{n+1}) given by

$$w_{j+\frac{1}{2}}(t) = w^n_{j+\frac{1}{2}}, \qquad t_n < t < t_{n+1},$$

which defines a cell in space–time as shown in Fig. 1. The algorithm to choose the mesh velocity $w^n_{j+\frac{1}{2}}$ is explained in a later section. The location of the cell boundaries is given by

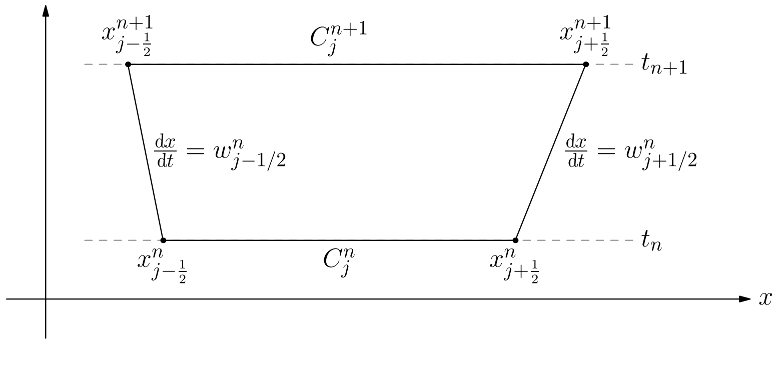

t
$x^{n+1}_{j-\frac{1}{2}}$
C^{n+1}_j
$x^{n+1}_{j+\frac{1}{2}}$
t_{n+1}
$\frac{dx}{dt} = w^n_{j-1/2}$
$\frac{dx}{dt} = w^n_{j+1/2}$
t_n
$x^n_{j-\frac{1}{2}}$
C^n_j
$x^n_{j+\frac{1}{2}}$
x

Fig. 1 Example of a space–time cell in the time interval (t_n, t_{n+1})

$$x_{j+\frac{1}{2}}(t) = x^n_{j+\frac{1}{2}} + (t - t_n)w^n_{j+\frac{1}{2}}, \qquad t_n \le t \le t_{n+1}.$$

Let $x_j(t)$ and $h_j(t)$ denote the center of the cell $C_j(t)$ and its length, i.e.,

$$x_j(t) = \frac{1}{2}(x_{j-\frac{1}{2}}(t) + x_{j+\frac{1}{2}}(t)), \qquad h_j(t) = x_{j+\frac{1}{2}}(t) - x_{j-\frac{1}{2}}(t).$$

Let $w(x, t)$ be the continuous linear interpolation of the mesh velocity which is given by

$$w(x,t) = \frac{x_{j+\frac{1}{2}}(t) - x}{h_j(t)} w^n_{j-\frac{1}{2}} + \frac{x - x_{j-\frac{1}{2}}(t)}{h_j(t)} w^n_{j+\frac{1}{2}}, \qquad x \in C_j(t), \quad t \in (t_n, t_{n+1}).$$

We will approximate the solution of the conservation law by piecewise polynomials which are allowed to be discontinuous across the cell boundaries as shown in Fig. 2. For a given degree $k \ge 0$, the solution in the jth cell is given by

$$\boldsymbol{u}_h(x,t) = \sum_{m=0}^{k} \boldsymbol{u}_{j,m}(t)\varphi_m(x,t), \qquad x \in C_j(t),$$

where $\{\boldsymbol{u}_{j,m} \in \mathbb{R}^3, 0 \le m \le k\}$ are the degrees of freedom associated with the jth cell. The basis functions φ_m are defined in terms of Legendre polynomials

$$\varphi_m(x,t) = \hat{\varphi}_m(\xi) = \sqrt{2m+1}\mathrm{P}_m(\xi), \qquad \xi = \frac{x - x_j(t)}{\frac{1}{2}h_j(t)},$$

where $\mathrm{P}_m : [-1,+1] \to \mathbb{R}$ is the Legendre polynomial of degree m. The above definition of the basis functions implies the following orthogonality property:

$$\int_{x_{j-\frac{1}{2}}(t)}^{x_{j+\frac{1}{2}}(t)} \varphi_l(x,t)\varphi_m(x,t)\mathrm{d}x = h_j(t)\delta_{lm}, \qquad 0 \le l,m \le k. \tag{3}$$

We will sometimes also write the solution in the jth cell in terms of the reference coordinates ξ as

$$\boldsymbol{u}_h(\xi,t) = \sum_{m=0}^{k} \boldsymbol{u}_{j,m}(t)\hat{\varphi}_m(\xi),$$

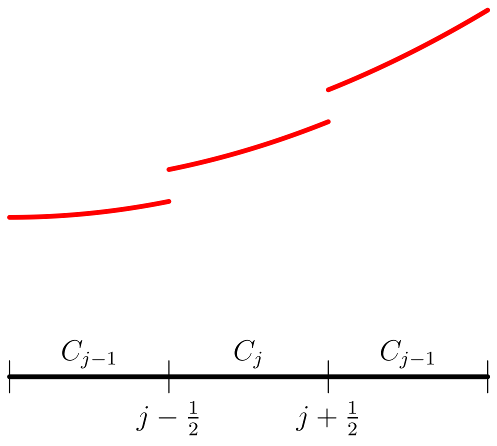


Fig. 2 Example of a discontinuous piecewise polynomial solution

and we will use the same notation $\boldsymbol{u}$ to denote both functions.

3.2 Derivation of the Scheme

To derive the DG scheme on a moving mesh, let us introduce the change of variable $(x, t) \to (\xi, \tau)$ given by

$$\tau = t, \qquad \xi = \frac{x - x_j(t)}{\frac{1}{2}h_j(t)}. \tag{4}$$

For any $0 \leq l \leq k$, we now calculate the rate of change of the lth moment of the solution starting from

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\int_{x_{j-\frac{1}{2}}(t)}^{x_{j+\frac{1}{2}}(t)} \boldsymbol{u}(x,t)\varphi_l(x,t)\mathrm{d}x &= \frac{\mathrm{d}}{\mathrm{d}\tau}\int_{-1}^{+1} \boldsymbol{u}(\xi,\tau)\hat{\varphi}_l(\xi)\frac{1}{2}h_j(\tau)\mathrm{d}\xi \\ &= \frac{1}{2}\int_{-1}^{+1}\left[h_j(\tau)\frac{\partial \boldsymbol{u}}{\partial \tau} + \boldsymbol{u}\frac{\mathrm{d}h_j}{\mathrm{d}\tau}\right]\hat{\varphi}(\xi)\mathrm{d}\xi,\end{aligned}$$

wherein we used the change of variables given by (4). But we also have the inverse transform

$$t = \tau, \qquad x = x_j(\tau) + \frac{\xi}{2}h_j(\tau),$$

and hence

$$\frac{\partial t}{\partial \tau} = 1, \qquad \frac{\partial x}{\partial \tau} = \frac{\mathrm{d}x_j}{\mathrm{d}\tau} + \frac{\xi}{2}\frac{\mathrm{d}h_j}{\mathrm{d}\tau} = \frac{1}{2}(w_{j-\frac{1}{2}} + w_{j+\frac{1}{2}}) + \frac{\xi}{2}(w_{j+\frac{1}{2}} - w_{j-\frac{1}{2}}) = w(x,t).$$

Using the above relations, we can easily show that

$$\frac{\partial \boldsymbol{u}}{\partial \tau}(\xi,\tau) = \frac{\partial \boldsymbol{u}}{\partial t}(x,t) + w(x,t)\frac{\partial \boldsymbol{u}}{\partial x}(x,t).$$

Moreover

$$\frac{\mathrm{d}h_j}{\mathrm{d}\tau} = w_{j+\frac{1}{2}} - w_{j-\frac{1}{2}} = h_j\frac{\partial w}{\partial x},$$

since $w(x, t)$ is linear in x and hence $\frac{\partial w}{\partial x}$ is constant inside each cell. Hence, the lth moment evolves according to

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\int_{x_{j-\frac{1}{2}}(t)}^{x_{j+\frac{1}{2}}(t)} \boldsymbol{u}(x,t)\varphi_l(x,t)\mathrm{d}x &= \int_{-1}^{+1}\left(\frac{\partial \boldsymbol{u}}{\partial t} + w\frac{\partial \boldsymbol{u}}{\partial x} + \boldsymbol{u}\frac{\partial w}{\partial x}\right)\hat{\varphi}_l(\xi)\frac{1}{2}h_j\mathrm{d}\xi \\ &= \int_{x_{j-\frac{1}{2}}(t)}^{x_{j+\frac{1}{2}}(t)}\left(-\frac{\partial \boldsymbol{f}(\boldsymbol{u})}{\partial x} + \frac{\partial}{\partial x}(w\boldsymbol{u})\right)\varphi_l(x,t)\mathrm{d}x,\end{aligned}$$

where we have transformed back to the physical coordinates and made use of the conservation law (1) to replace the time derivative of the solution with the flux derivative. Define the ALE flux

$$\boldsymbol{g}(\boldsymbol{u}, w) = \boldsymbol{f}(\boldsymbol{u}) - w\boldsymbol{u}. \tag{5}$$

Performing an integration by parts in the x variable, we obtain

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\int_{x_{j-\frac{1}{2}}(t)}^{x_{j+\frac{1}{2}}(t)} \boldsymbol{u}_h(x,t)\varphi_l(x,t)\mathrm{d}x = &\int_{x_{j-\frac{1}{2}}(t)}^{x_{j+\frac{1}{2}}(t)} \boldsymbol{g}(\boldsymbol{u}_h, w)\frac{\partial}{\partial x}\varphi_l(x,t)\mathrm{d}x\\ &+ \hat{\boldsymbol{g}}_{j-\frac{1}{2}}(\boldsymbol{u}_h(t))\varphi_l(x^+_{j-\frac{1}{2}}, t) - \hat{\boldsymbol{g}}_{j+\frac{1}{2}}(\boldsymbol{u}_h(t))\varphi_l(x^-_{j+\frac{1}{2}}, t),\end{aligned}$$

where we have introduced the numerical flux

$$\hat{\boldsymbol{g}}_{j+\frac{1}{2}}(\boldsymbol{u}_h(t)) = \hat{\boldsymbol{g}}(\boldsymbol{u}^-_{j+\frac{1}{2}}(t), \boldsymbol{u}^+_{j+\frac{1}{2}}(t), w_{j+\frac{1}{2}}(t)),$$

which provides an approximation to the ALE flux, see Appendix A. Integrating over the time interval (t_n, t_{n+1}) and using (3), we obtain

$$\begin{aligned}h_j^{n+1}\boldsymbol{u}_{j,l}^{n+1} = h_j^n\boldsymbol{u}_{j,l}^n &+ \int_{t_n}^{t_{n+1}}\int_{x_{j-\frac{1}{2}}(t)}^{x_{j+\frac{1}{2}}(t)} \boldsymbol{g}(\boldsymbol{u}_h, w)\frac{\partial}{\partial x}\varphi_l(x,t)\mathrm{d}x\mathrm{d}t\\ &+ \int_{t_n}^{t_{n+1}} [\hat{\boldsymbol{g}}_{j-\frac{1}{2}}(t)\varphi_l(x^+_{j-\frac{1}{2}}, t) - \hat{\boldsymbol{g}}_{j+\frac{1}{2}}(t)\varphi_l(x^-_{j+\frac{1}{2}}, t)]\mathrm{d}t.\end{aligned}$$

The above scheme has an implicit nature since the unknown solution $\boldsymbol{u}_h$ appears on the right-hand side integrals whereas we only know the solution at time t_n. To obtain an explicit scheme, we assume that we have available with us a predicted solution $\boldsymbol{U}_h$ in the time interval (t_n, t_{n+1}), which is used in the time integrals to obtain an explicit scheme. Moreover, the integrals are computed using quadrature in space and time leading to the fully discrete scheme

$$\begin{aligned}h_j^{n+1}\boldsymbol{u}_{j,l}^{n+1} = &h_j^n\boldsymbol{u}_{j,l}^n\\ &+ \Delta t_n \sum_r \theta_r h_j(\tau_r)\sum_q \eta_q \boldsymbol{g}(\boldsymbol{U}_h(x_q, \tau_r), w(x_q, \tau_r))\frac{\partial}{\partial x}\varphi_l(x_q, \tau_r)\\ &+ \Delta t_n \sum_r \theta_r [\hat{\boldsymbol{g}}_{j-\frac{1}{2}}(\boldsymbol{U}_h(\tau_r))\varphi_l(x^+_{j-\frac{1}{2}}, \tau_r) - \hat{\boldsymbol{g}}_{j+\frac{1}{2}}(\boldsymbol{U}_h(\tau_r))\varphi_l(x^-_{j+\frac{1}{2}}, \tau_r)],\end{aligned} \tag{6}$$

where θ_r is the weight for time quadrature and η_q is the weight for spatial quadrature. For the spatial integral, we will use $q = k + 1$ point Gauss quadrature. For the time integral, we will use the mid-point rule for $k = 1$ and two-point Gauss quadrature for $k = 2, 3$. Since the mesh is moving, the spatial quadrature points x_q depend on the quadrature time τ_r though this is not clear from the notation. In practice, the integrals are computed by mapping the cell to the reference cell, and the basis functions and its derivatives are also evaluated on the reference cell. The quadrature points in the reference cell are independent of time due to the linear mesh evolution.

3.3 Mesh Velocity

The mesh velocity must be close to the local fluid velocity to have a Lagrangian character to the scheme. Since the solution is discontinuous, there is no unique fluid velocity at the mesh boundaries. Some researchers, especially in the context of Lagrangian methods, solve a Riemann problem at the cell face to determine the face velocity. Since we use an ALE formulation, we do not require the exact fluid velocity which is anyway not available to use since we only have a predicted solution. Following the exact trajectory of the fluid would also lead to curved trajectories for the grid point, which is an unnecessary complication. In our work, we make two different choices for the mesh velocities.

i. The first choice is to take an average of the two velocities at every face. In the numerical results, we refer to this as ADG

$$\tilde{w}^n_{j+\frac{1}{2}} = \frac{1}{2}\left[v\left(x^-_{j+\frac{1}{2}}, t_n\right) + v\left(x^+_{j+\frac{1}{2}}, t_n\right)\right].$$

ii. The second choice is to solve a linearized Riemann problem at the face at time t_n. In the numerical results, we refer to this as RDG. For simplicity of notation, let the solution to the left of the face $x_{j+\frac{1}{2}}$ be represented as $u^-_{j+\frac{1}{2}}$ and the solution to the right be represented as $u^+_{j+\frac{1}{2}}$. Then,

$$\tilde{w}^n_{j+\frac{1}{2}} = \frac{\rho^n_j c^n_j v^n_j + \rho^n_{j+1} c^n_{j+1} v^n_{j+1}}{\rho^n_j c^n_j + \rho^n_{j+1} c^n_{j+1}} + \frac{p^n_j - p^n_{j+1}}{\rho^n_j c^n_j + \rho^n_{j+1} c^n_{j+1}}.$$

We will also perform some smoothing of the mesh velocity, e.g., the actual face velocity is computed from

$$w^n_{j+\frac{1}{2}} = \frac{1}{3}\left(\tilde{w}^n_{j-\frac{1}{2}} + \tilde{w}^n_{j+\frac{1}{2}} + \tilde{w}^n_{j+\frac{3}{2}}\right).$$

Note that our algorithm to choose the mesh velocity is very local and hence easy and efficient to implement as it does not require the solution of any global problems. In Springel [1], the mesh velocity is adjusted so that the cells remain nearly isotropic which leads to smoothly varying cell sizes. Such an approach leads to many parameters that need to be selected and we did not find a good way to make this choice that works well for a range of problems. Instead, we will make use of mesh refinement and coarsening to maintain the quality of cells, i.e., to prevent very small or large cells from occurring in the grid. The use of a DG scheme makes it easy to perform such local mesh adaptation without loss of accuracy.

Remark 1 Consider the application of the proposed ALE-DG scheme to the linear advection equation $u_t + au_x = 0$. In this case, the mesh velocity is equal to the advection velocity $w_{j+\frac{1}{2}} = a$, i.e., the cells move along the characteristics. This implies that the ALE flux $g(u, w) = au - wu = 0$ and also the numerical flux $\hat{g}_{j+\frac{1}{2}} = 0$. Thus, the DG scheme reduces to

$$\int_{x^{n+1}_{j-\frac{1}{2}}}^{x^{n+1}_{j+\frac{1}{2}}} u_h(x, t_{n+1})\varphi_l(x, t_{n+1})\mathrm{d}x = \int_{x^n_{j-\frac{1}{2}}}^{x^n_{j+\frac{1}{2}}} u_h(x, t_n)\varphi_l(x, t_n)\mathrm{d}x, \qquad l = 0, 1, \cdots, k,$$

so that the solution at time t_n has been advected exactly to the solution at time t_{n+1}. Note that there is no time step restriction involved in this case and the accuracy of the predicted solution is also not relevant. If the initial condition has a discontinuity coinciding with a cell face, then the scheme advects the discontinuity exactly without any diffusion.

4 Computing the Predictor

The predicted solution is used to approximate the flux integrals over the time interval (t_n, t_{n+1}) and the method to compute this must be local, i.e., it must not require solution from neighbouring cells. Several methods for computing the predictor have been reviewed in [24]. The simplest approach is to use a Taylor expansion in space and time. Since the cells are moving, the Taylor expansion has to be performed along the trajectory of the mesh motion. For a second-order scheme, an expansion retaining only linear terms in t and x is sufficient. Consider a quadrature point (x_q, τ_r), the Taylor expansion of the solution around the cell center x_j^n and time level t_n yields

$$\begin{aligned}\boldsymbol{u}_h(x_q,\tau_r) =& \boldsymbol{u}_h(x_j^n,t_n) + (\tau_r - t_n)\frac{\partial \boldsymbol{u}_h}{\partial t}(x_j^n,t_n) + (x_q - x_j^n)\frac{\partial \boldsymbol{u}_h}{\partial x}(x_j^n,t_n)\\ &+ O(\tau_r - t_n)^2 + O(x_q - x_j^n)^2,\end{aligned}$$

and the predicted solution is given by truncating the Taylor expansion at linear terms, leading to

$$\boldsymbol{U}(x_q,\tau_r) = \boldsymbol{u}_h(x_j^n,t_n) + (\tau_r - t_n)\frac{\partial \boldsymbol{u}_h}{\partial t}(x_j^n,t_n) + (x_q - x_j^n)\frac{\partial \boldsymbol{u}_h}{\partial x}(x_j^n,t_n).$$

Using the conservation law, the time derivative is written as $\frac{\partial \boldsymbol{u}}{\partial t} = -\frac{\partial \boldsymbol{f}}{\partial x} = -A\frac{\partial \boldsymbol{u}}{\partial x}$ so that the predictor is given by

$$\boldsymbol{U}_h(x_q,\tau_r) = \boldsymbol{u}_h^n(x_j^n) - (\tau_r - t_n)\Big[A(\boldsymbol{u}_h^n(x_j^n)) - w_q I\Big]\frac{\partial \boldsymbol{u}_h^n}{\partial x}(x_j^n). \tag{7}$$

The above predictor is used for the case of polynomial degree $k = 1$. This procedure can be extended to higher orders by including more terms in the Taylor expansion but the algebra becomes complicated. Instead we will adopt the approach of continuous explicit Runge–Kutta (CERK) schemes [25] to approximate the predictor.

Let us choose a set of $(k+1)$ distinct nodes, e.g., Gauss–Legendre or Gauss–Lobatto nodes, which uniquely define the polynomial of degree k. These nodes are moving with velocity $w(x,t)$, so that the time evolution of the solution at node x_m is governed by

$$\begin{aligned}\frac{\mathrm{d}\boldsymbol{U}_m}{\mathrm{d}t} &= \frac{\partial}{\partial t}\boldsymbol{U}_h(x_m,t) + w(x_m,t)\frac{\partial}{\partial x}\boldsymbol{U}_h(x_m,t)\\ &= -\frac{\partial}{\partial x}\boldsymbol{f}(\boldsymbol{U}_h(x_m,t)) + w(x_m,t)\frac{\partial}{\partial x}\boldsymbol{U}_h(x_m,t)\\ &= -[A(\boldsymbol{U}_m(t)) - w_m(t)I]\frac{\partial}{\partial x}\boldsymbol{U}_h(x_m,t) =: \boldsymbol{K}_m(t),\end{aligned}$$

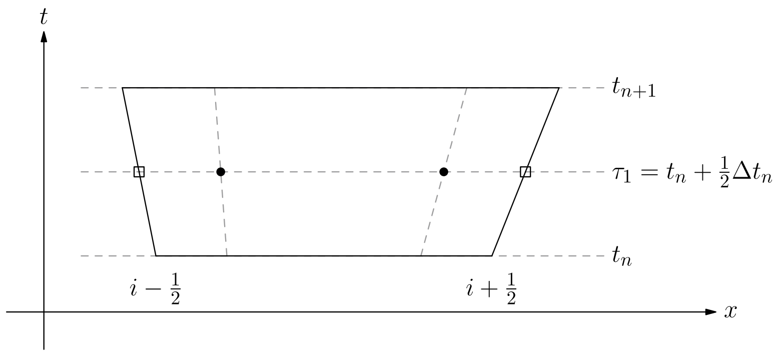


Fig. 3 Quadrature points for second-order scheme

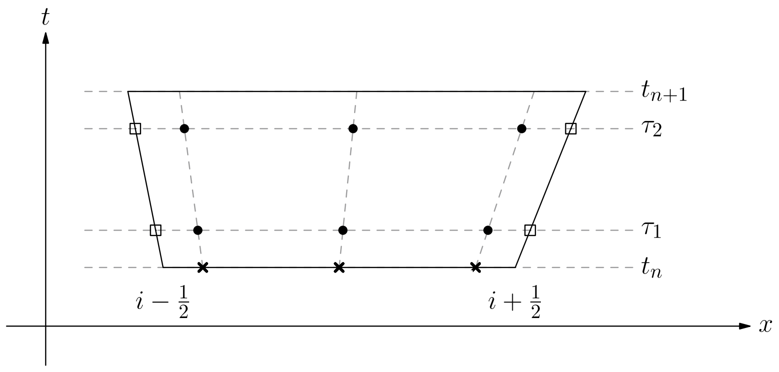


Fig. 4 Quadrature points for third-order scheme

wherein we have made use of the PDE to write the time derivative in terms of spatial derivative of the flux. This equation is solved with the initial condition

$$\boldsymbol{U}_m(t_n) = \boldsymbol{u}_h(x_m, t_n).$$

Using a Runge–Kutta scheme of sufficient order (see Appendix B), we will approximate the solution at these nodes as

$$\boldsymbol{U}_m(t) = \boldsymbol{u}_h(x_m, t_n) + \sum_{s=1}^{n_s} b_s((t-t_n)/\Delta t_n)\boldsymbol{K}_{m,s}, \quad t \in [t_n, t_{n+1}), \quad m = 0, 1, \cdots, k,$$

where $\boldsymbol{K}_{m,s} = \boldsymbol{K}_m(t_n + \theta_s \Delta t_n)$, $\theta_s \Delta t_n$ is the stage time and b_s are certain polynomials related to the CERK scheme and given in Appendix B. Note that we are evolving the nodal values but the computation of $\boldsymbol{K}_{m,s}$ requires the modal representation of the solution in order to calculate spatial derivative of the solution.

Once the predictor is computed as above, it must be evaluated at the quadrature point (x_q, τ_r) as follows. For each time quadrature point $\tau_r \in (t_n, t_{n+1})$,

i. compute nodal values $\boldsymbol{U}_m(\tau_r)$, $m = 0, 1, \cdots, k$;
ii. for each r, convert the nodal values to modal coefficients $\boldsymbol{u}_{m,r}$, $m = 0, 1, \cdots, k$;
iii. evaluate predictor $\boldsymbol{U}_h(x_q, \tau_r) = \sum_{m=0}^{k} \boldsymbol{u}_{m,r}\varphi_m(x_q, \tau_r)$.

The conversion from nodal to modal values is accomplished through a Vandermonde matrix of size $(k+1) \times (k+1)$ which is the same for every cell and can be inverted once before the iterations start. The predictor is also computed at the cell boundaries using the above procedure. Figures 3 and 4 show the quadrature points used in the second-, third- and fourth-order scheme. For the second-order scheme, the values at • and □ points are obtained from the predictor based on Taylor expansion as given in (7). For third- and fourth-order schemes, the nodal values × are evolved forward in time by the CERK scheme and evaluated at the • points. The • point values are converted to modal coefficients using which the solution at the □ points is computed.

Remark 2 By looping over each cell in the mesh, the predicted solution is computed in each cell and the cell integral in (6) is evaluated. The trace values $\boldsymbol{U}^-_{j+\frac{1}{2}}(\tau_r)$ and $\boldsymbol{U}^+_{j-\frac{1}{2}}(\tau_r)$ at the □ points needed for quadrature in time are computed and stored. These are later used in a loop over the cell faces where the numerical flux is evaluated. Thus, the algorithm is easily parallelizable on multiple core machines and/or using threads.

5 Control of Oscillation by Limiting

High order schemes for hyperbolic equations suffer from spurious numerical oscillations when discontinuities or large gradients are present in the solution which cannot be accurately resolved on the mesh. In the case of scalar problems, this is a manifestation of loss of TVD property and hence limiters are used to satisfy some form of TVD condition. In the case of DG schemes, the limiter is used as a post-processor which is applied on the solution after the time update has been performed. If the limiter detects that the solution is oscillatory, then the solution polynomial is reduced to at most a linear polynomial with a limited slope. In the present scheme, the limiter is applied after the solution is updated from time t_n to time t_{n+1}, i.e., the solution $\boldsymbol{u}_h^{n+1}$ obtained from (6) is post-processed by the limiter. Since the mesh is inherently non-uniform due to it being moved with the flow, we modify the standard TVD limiter to account for this non-uniformity. Also, since we are solving a system of conservation laws, the limiter is applied on the local characteristic variables which gives better results than applying it directly on the conserved variables [26].

The solution in cell j can be written as[1]

$$\boldsymbol{u}_h(x) = \bar{\boldsymbol{u}}_j + \frac{x - x_j}{\frac{1}{2}h_j}\boldsymbol{s}_j + \text{higher order terms},$$

where $\bar{\boldsymbol{u}}_j$ is the cell average value and $\boldsymbol{s}_j$ is proportional to the derivative of the solution at the cell center. Let R_j, L_j denote the matrix of right and left eigenvectors evaluated at the cell average value $\bar{\boldsymbol{u}}_j$ which satisfy $L_j = R_j^{-1}$, and the right eigenvectors are given in (2). The local characteristic variables are defined by $\bar{\boldsymbol{u}}^* = L_j\bar{\boldsymbol{u}}$ and $\boldsymbol{s}^* = L_j\boldsymbol{s}$. We first compute the limited slope of the characteristic variables from

$$\boldsymbol{s}_j^{**} = h_j \operatorname{minmod}\left(\frac{\boldsymbol{s}_j^*}{h_j}, \frac{\bar{\boldsymbol{u}}_j^* - \bar{\boldsymbol{u}}_{j-1}^*}{\frac{1}{2}(h_{j-1} + h_j)}, \frac{\bar{\boldsymbol{u}}_{j+1}^* - \bar{\boldsymbol{u}}_j^*}{\frac{1}{2}(h_j + h_{j+1})}\right),$$

where the minmod function is defined by

$$\operatorname{minmod}(a, b, c) = \begin{cases} s\min(|a|, |b|, |c|), & \text{if } s = \operatorname{sign}(a) = \operatorname{sign}(b) = \operatorname{sign}(c), \\ 0, & \text{otherwise.} \end{cases}$$

[1] We suppress the time variable for clarity of notation.

If $\boldsymbol{s}_j^{**} = \boldsymbol{s}_j^*$, then we retain the solution as it is, and otherwise, the solution is modified to

$$\boldsymbol{u}_h(x) = \bar{\boldsymbol{u}}_j + \frac{x - x_j}{\frac{1}{2}h_j} R_j \boldsymbol{s}_j^{**}.$$

This corresponds to a TVD limiter which is known to lose accuracy at smooth extrema [27] since the minmod function returns zero slope at local extrema. The TVB limiter corresponds to replacing the minmod limiter function with the following function:

$$\overline{\text{minmod}}(a, b, c) = \begin{cases} a, & \text{if } |a| \leqslant Mh^2, \\ \text{minmod}(a, b, c), & \text{otherwise}, \end{cases} \tag{8}$$

where the parameter M is an estimate of the second derivative of the solution at smooth extrema [27] and has to be chosen by the user.

6 Positivity Property

The solutions of Euler equations are well defined only if the density and pressure are positive quantities. This is not a priori guaranteed by the DG scheme even when the TVD limiter is applied. In the case of Runge–Kutta DG schemes, a positivity limiter has been developed in [28] which preserves accuracy in smooth regions. This scheme is built on a positive first-order finite volume scheme. Consider the first-order version of the ALE-DG scheme which is a finite volume scheme given by

$$h_j^{n+1} \bar{\boldsymbol{u}}_j^{n+1} = h_j^n \bar{\boldsymbol{u}}_j^n - \Delta t_n [\hat{\boldsymbol{g}}_{j+\frac{1}{2}}^n - \hat{\boldsymbol{g}}_{j-\frac{1}{2}}^n]. \tag{9}$$

The only degree of freedom is the cell average value and the solution is piecewise constant. We will analyze the positivity of this scheme for the case of the Rusanov flux which is given in 1. The update equation can be rewritten as

$$\begin{aligned} h_j^{n+1} \bar{\boldsymbol{u}}_j^{n+1} = & \left[h_j^n - \frac{\Delta t_n}{2} \left(\lambda_{j-\frac{1}{2}}^n + w_{j-\frac{1}{2}}^n + \lambda_{j+\frac{1}{2}}^n - w_{j+\frac{1}{2}}^n \right) \right] \bar{\boldsymbol{u}}_j^n \\ & + \frac{\Delta t_n}{2} \left[\left(\lambda_{j-\frac{1}{2}}^n - w_{j-\frac{1}{2}}^n \right) \bar{\boldsymbol{u}}_{j-1}^n + \boldsymbol{f}_{j-1}^n \right] \\ & + \frac{\Delta t_n}{2} \left[\left(\lambda_{j+\frac{1}{2}}^n + w_{j+\frac{1}{2}}^n \right) \bar{\boldsymbol{u}}_{j+1}^n - \boldsymbol{f}_{j+1}^n \right] \\ = & \, a_j^n \bar{\boldsymbol{u}}_j^n + \frac{\Delta t_n}{2} B_j^n + \frac{\Delta t_n}{2} C_j^n. \end{aligned}$$

From the definition of the Rusanov flux formula, we can easily see that[2]

$$\left(\lambda_{j-\frac{1}{2}} - w_{j-\frac{1}{2}} \right) + v_{j-1} \geqslant c_{j-1} > 0, \qquad \left(\lambda_{j+\frac{1}{2}} + w_{j+\frac{1}{2}} \right) - v_{j+1} \geqslant c_{j+1} > 0.$$

[2] We drop the superscript ^{n} in some of these expressions.

Consider the first component of B_j^n

$$\begin{aligned}\left(\lambda_{j-\frac{1}{2}} - w_{j-\frac{1}{2}}\right)\rho_{j-1} + \rho_{j-1}v_{j-1} \geqslant & \left(|v_{j-1} - w_{j-\frac{1}{2}}| + c_{j-1} - w_{j-\frac{1}{2}} + v_{j-1}\right)\rho_{j-1}\\ \geqslant & c_{j-1}\rho_{j-1} > 0.\end{aligned}$$

Consider the first component of C_j^n

$$\begin{aligned}\left(\lambda_{j+\frac{1}{2}} + w_{j+\frac{1}{2}}\right)\rho_{j+1} - \rho_{j+1}v_{j+1} \geqslant & (|v_j - w_{j+\frac{1}{2}}| + c_{j+1} + w_{j-1} - v_{j+1})\rho_{j+1}\\ \geqslant & c_{j+1}\rho_{j+1} > 0.\end{aligned}$$

The pressure corresponding to B_j^n is

$$p_{j-1}\left(-p_{j-1} + \frac{2\rho_{j-1}(v_{j-1} + \lambda_{j-\frac{1}{2}} - w_{j-\frac{1}{2}})^2}{\gamma - 1}\right) \geqslant p_{j-1}\rho_{j-1}c_{j-1}^2\frac{\gamma+1}{\gamma-1} \geqslant 0,$$

and similarly the pressure corresponding to C_j^n is non-negative. Hence, if the coefficient in term a_j^n is positive, then the scheme is positive. This requires the CFL condition

$$\Delta t_n \leqslant \frac{2h_j^n}{\lambda_{j-\frac{1}{2}}^n + w_{j-\frac{1}{2}}^n + \lambda_{j+\frac{1}{2}}^n - w_{j+\frac{1}{2}}^n}.$$

The time step will also be restricted to ensure that the cell size does not change too much in one time step. If we demand that the cell size does not change by more than a fraction $\beta \in (0, 1)$, then we need to ensure that the time step satisfies

$$\Delta t_n \leqslant \frac{\beta h_j^n}{|w_{j+\frac{1}{2}}^n - w_{j-\frac{1}{2}}^n|}.$$

Combining the previous two conditions, we obtain the following condition on the time step:

$$\Delta t_n \leqslant \Delta t_n^{(1)} := \min_j \left\{ \frac{\left(1 - \frac{1}{2}\beta\right)h_j^n}{\frac{1}{2}\left(\lambda_{j-\frac{1}{2}}^n + \lambda_{j+\frac{1}{2}}^n\right)}, \frac{\beta h_j^n}{\left|w_{j+\frac{1}{2}}^n - w_{j-\frac{1}{2}}^n\right|} \right\}. \tag{10}$$

We can now state the following result on the positivity of the first-order finite volume scheme on moving meshes.

Theorem 1 *The scheme* (9) *with the Rusanov flux is positivity preserving if the time step condition* (10) *is satisfied.*

Remark 3 In this work, we have not attempted to prove the positivity of the scheme for other numerical fluxes. We also do not have a proof of positivity for higher order version of the scheme. In the computations, we use the positivity preserving limiter of [28] which leads to robust schemes which preserve the positivity of the cell average value in all the test cases.

7 Preservation of Constant States

An important property of schemes on moving meshes is their ability to preserve constant states for any mesh motion. This is related to the conservation of cell volumes in relation to the mesh motion. In our scheme, if we start with a constant state $\boldsymbol{u}_h^n = \boldsymbol{c}$, then the predictor is also constant in the space–time interval, i.e., $\boldsymbol{U}_h = \boldsymbol{c}$. The space–time terms in (6) are polynomials with degree $k+1$ in space and degree one in time and these are exactly integrated by the chosen quadrature rule. The flux terms at cell boundaries in (6) are of degree one in time and these are also exactly integrated. Hence, the scheme (6) can be written as

$$h_j^{n+1}\boldsymbol{u}_{j,l}^{n+1} = h_j^n\boldsymbol{u}_{j,l}^n + \int_{t_n}^{t_{n+1}}\int_{x_{j-\frac12}(t)}^{x_{j+\frac12}(t)} \boldsymbol{g}(\boldsymbol{c}, w)\frac{\partial}{\partial x}\varphi_l(x,t)\mathrm{d}x\mathrm{d}t$$
$$+ \int_{t_n}^{t_{n+1}} \left[\hat{\boldsymbol{g}}_{j-\frac12}(t)\varphi_l\left(x_{j-\frac12}^+, t\right) - \hat{\boldsymbol{g}}_{j+\frac12}(t)\varphi_l\left(x_{j+\frac12}^-, t\right)\right]\mathrm{d}t,$$

where $\boldsymbol{u}_{j,0}^n = \boldsymbol{c}$ and $\boldsymbol{u}_{j,l}^n = 0$ for $l = 1, 2, \cdots, k$. Due to the constant predictor and by consistency of the numerical flux

$$\hat{\boldsymbol{g}}_{j+\frac12}(t) = \boldsymbol{f}(\boldsymbol{c}) - w_{j+\frac12}^n\boldsymbol{c}.$$

Moreover, for $l = 1, 2, \cdots, k$,

$$\int_{t_n}^{t_{n+1}}\int_{x_{j-\frac12}(t)}^{x_{j+\frac12}(t)} \boldsymbol{g}(\boldsymbol{c}, w)\frac{\partial}{\partial x}\varphi_l(x,t)\mathrm{d}x\mathrm{d}t + \int_{t_n}^{t_{n+1}} \left[\hat{\boldsymbol{g}}_{j-\frac12}(t)\varphi_l\left(x_{j-\frac12}^+, t\right) - \hat{\boldsymbol{g}}_{j+\frac12}(t)\varphi_l\left(x_{j+\frac12}^-, t\right)\right]\mathrm{d}t$$
$$= \int_{t_n}^{t_{n+1}}\int_{x_{j-\frac12}(t)}^{x_{j+\frac12}(t)} \frac{\partial}{\partial x}\boldsymbol{g}(\boldsymbol{c}, w)\varphi_l(x,t)\mathrm{d}x\mathrm{d}t + \int_{t_n}^{t_{n+1}} \left[\hat{\boldsymbol{g}}_{j-\frac12}(t)\varphi_l\left(x_{j-\frac12}^+, t\right) - \hat{\boldsymbol{g}}_{j+\frac12}(t)\varphi_l\left(x_{j+\frac12}^-, t\right)\right]\mathrm{d}t$$
$$- \int_{t_n}^{t_{n+1}}\int_{x_{j-\frac12}(t)}^{x_{j+\frac12}(t)} \varphi_l(x,t)\frac{\partial}{\partial x}\boldsymbol{g}(\boldsymbol{c}, w)\mathrm{d}x\mathrm{d}t$$
$$= -\int_{t_n}^{t_{n+1}} \frac{\partial}{\partial x}\boldsymbol{g}(\boldsymbol{c}, w)\left(\int_{x_{j-\frac12}(t)}^{x_{j+\frac12}(t)} \varphi_l(x,t)\mathrm{d}x\right)\mathrm{d}t = 0,$$

where we have used the property that w is an affine function of x and φ_l are orthogonal. This implies that $\boldsymbol{u}_{j,l}^{n+1} = 0$ for $l = 1, 2, \cdots, k$. For $l = 0$, we get

$$h_j^{n+1}\boldsymbol{u}_{j,0}^{n+1} = h_j^n\boldsymbol{c} + \int_{t_n}^{t_{n+1}} \left[\hat{\boldsymbol{g}}_{j-\frac12}(t) - \hat{\boldsymbol{g}}_{j+\frac12}(t)\right]\mathrm{d}t = \left[h_j^n + \left(w_{j-\frac12}^n - w_{j+\frac12}^n\right)\Delta t\right]\boldsymbol{c},$$

and since $h_j^{n+1} = h_j^n + \left(w_{j-\frac{1}{2}}^n - w_{j+\frac{1}{2}}^n \right) \Delta t$, we obtain $\boldsymbol{u}_{j,0} = \boldsymbol{c}$ which implies that $\boldsymbol{u}_h^{n+1} = \boldsymbol{c}$.

8 Grid Coarsening and Refinement

The size of the cells can change considerably during the time evolution process due to the near Lagrangian movement of the cell boundaries. Near shocks, the cells will be compressed to smaller sizes which will reduce the allowable time step since a CFL condition has to be satisfied. In some regions, e.g., inside expansion fans, the cell size can increase considerably which may lead to loss of accuracy. To avoid too small or too large cells from occurring in the grid, we implement cell merging and refinement into our scheme. If a cell becomes smaller than some specified size $h_{\min}$, then it is merged with one of its neighbouring cells and the solution is transferred from the two cells to the new cell by performing an L^2 projection. If a cell size becomes larger than some specified size $h_{\max}$, then this cell is refined into two cells by division and the solution is again transferred by the L^2 projection. The use of the L^2 projection for solution transfer ensures the conservation of mass, momentum and energy and preserves the accuracy in smooth regions. We also ensure that the cell sizes do not change drastically between neighbouring cells. To keep a track of refinement of cells, each cell is assigned an initial level equal to 0. The daughter cells created during refinement are assigned a level incremented from the parent cell, while the coarsened cells are assigned a level decremented from the parent cell.

The algorithm for refinement and coarsening is carried out in three sweeps over all the active cells. In the first sweep, we mark the cells for refinement or coarsening based on their size and the level of neighboring cells. Cells are marked for coarsening if the size is less than a pre-specified minimum size. They are marked for refinement if either the size of the cell is larger than the maximum size or if the level of the cell is less than the level of the neighboring cells. If none of the conditions are satisfied, the cells are marked for no change. In the second sweep, a cell is marked for refinement if both the neighboring cells are marked for refinement. A cell is also marked for refinement if the size of the cell is larger than twice the size of either of the neighboring cells, and is also larger than twice the minimum size. The last condition is inserted to prevent a cell being alternately marked for refinement and coarsening in consecutive adaptation cycles. In the third and final sweeps, we again mark cells for refinement if both the neighboring cells are marked for refinement. Further, we ensure that a cell marked for refinement does not have a neighboring cell marked for a coarsening, since this can lead to an inconsistent mesh.

9 Numerical Results

The numerical tests are performed with polynomials of degree one, two and three, together with the linear Taylor expansion, two stage CERK and four stage CERK, respectively, for the computation of the predictor. For the quadrature in time, we use the mid-point rule, two- and three-point Gauss–Legendre quadrature, respectively. The time step is chosen using the CFL condition,

$$\Delta t_n = \frac{\mathrm{CFL}}{2k+1} \Delta t_n^{(1)},$$

Table 1 Order of accuracy study on static mesh using Rusanov flux

N	$k=1$		$k=2$		$k=3$	
	Error	Rate	Error	Rate	Error	Rate
100	4.370E−02	–	3.498E−03	–	3.883E−04	–
200	6.611E−03	2.725	4.766E−04	2.876	1.620E−05	4.583
400	1.332E−03	2.518	6.415E−05	2.885	9.376E−07	4.347
800	3.151E−04	2.372	8.246E−06	2.910	5.763E−08	4.239
1 600	7.846E−05	2.280	1.031E−06	2.932	3.595E−09	4.180

Table 2 Order of accuracy study on moving mesh using Rusanov flux

N	$k=1$		$k=2$		$k=3$	
	Error	Rate	Error	Rate	Error	Rate
100	2.331E−02	–	3.979E−03	–	8.633E−04	–
200	6.139E−03	1.9250	4.058E−04	3.294	1.185E−05	6.186
400	1.406E−03	2.0258	5.250E−05	3.122	7.079E−07	5.126
800	3.375E−04	2.0366	6.626E−06	3.077	4.340E−08	4.760
1 600	8.278E−05	2.0344	8.304E−07	3.057	2.689E−09	4.573

where $\Delta t_n^{(1)}$ is given by (10), and the factor $(2k+1)$ comes from linear stability analysis [27]. In most of the computations, we use CFL $= 0.9$ unless stated otherwise. We observe that the results using the average or linearized Riemann velocity are quite similar. We use the average velocity for most of the results and show the comparison between the two velocities for some results. The main steps in the algorithm within one time step $t_n \to t_{n+1}$ are as follows:

i. choose the mesh velocity $w_{j+\frac{1}{2}}$;
ii. choose the time step Δt_n;
iii. compute the predictor $\boldsymbol{U}_h$;
iv. update the solution $\boldsymbol{u}_h^n$ to the next time level $\boldsymbol{u}_h^{n+1}$;
v. apply the TVD/TVB limiter on $\boldsymbol{u}_h^{n+1}$;
vi. apply the positivity limiter on $\boldsymbol{u}_h^{n+1}$ from [28];
vii. perform grid refinement/coarsening.

In all the solution plots given below, symbols denote the cell average value.

9.1 Order of Accuracy

We study the convergence rate of the schemes by applying them to a problem with a known smooth solution. The initial condition is taken as

$$\rho(x,0) = 1 + \exp(-10x^2), \qquad u(x,0) = 1, \qquad p(x,0) = 1,$$

whose exact solution is $\rho(x,t) = \rho(x-t,0)$, $u(x,t) = 1$, $p(x,t) = 1$. The initial domain is $[-5,+5]$ and the final time is $t = 1$ units. The results are presented using Rusanov and

Table 3 Order of accuracy study on static mesh using HLLC flux

N	$k=1$		$k=2$		$k=3$	
	Error	Rate	Error	Rate	Error	Rate
100	4.582E−02	–	3.952E−03	–	3.464E−04	–
200	9.611E−03	2.253	4.048E−04	3.287	2.058E−05	4.073
400	2.052E−03	2.240	4.640E−05	3.206	1.287E−06	4.036
800	4.803E−04	2.192	5.623E−06	3.152	8.061E−08	4.023
1 600	1.184E−04	2.149	6.929E−07	3.119	5.050E−09	4.016

Table 4 Order of accuracy study on moving mesh using HLLC flux

N	$k=1$		$k=2$		$k=3$	
	Error	Rate	Error	Rate	Error	Rate
100	1.590E−02	–	1.626E−03	–	1.962E−04	–
200	4.042E−03	1.977	2.072E−04	2.972	1.269E−05	3.950
400	1.014E−03	1.985	2.605E−05	2.982	7.983E−07	3.971
800	2.538E−04	1.990	3.261E−06	2.988	4.997E−08	3.980
1 600	6.349E−05	1.992	4.077E−07	2.991	3.124E−09	3.985

Table 5 Order of accuracy study on moving mesh using Rusanov flux using higher order limiter [29]

N	$k=1$		$k=2$	
	Error	Rate	Error	Rate
100	2.053E−02	–	2.277E−03	–
200	4.312E−03	2.251	3.425E−04	2.732
400	1.031E−03	2.064	4.565E−05	2.907
800	2.550E−04	2.015	5.812E−06	2.973
1 600	6.356E−05	2.004	7.315E−07	2.990

HLLC numerical fluxes. The L^2 norm of the error in density is shown in Tables 1 and 3 for the static mesh and in Tables 2 and 4 for the moving mesh. In each case, we see that the error behaves as $O(h^{k+1})$ which is the optimal rate we can expect for smooth solutions. In Table 5, we show that the ALE DG methods preserve its higher order in the presence of a limiter.

The mesh velocity is constant since the fluid velocity is constant. To study the effect of perturbations in mesh velocity, we add a random perturbation to each mesh velocity, $w_{j+\frac{1}{2}} \leftarrow (1+\alpha r_{j+\frac{1}{2}})w_{j+\frac{1}{2}}$ where $r_{j+\frac{1}{2}}$ is a uniform random variable in $[-1,+1]$ and $\alpha=0.05$ and a sample velocity distribution is shown in Fig. 5. Note that this randomization is performed in each time step with different random variables drawn for each face. For the moving mesh, there is no unique cell size and the convergence rate is computed based on initial mesh spacing which is inversely proportional to the number of cells. From Table 6 which shows results using HLLC flux, we again observe that the error reduces at the optional rate of $k+1$ even when the mesh velocity is not very smooth.

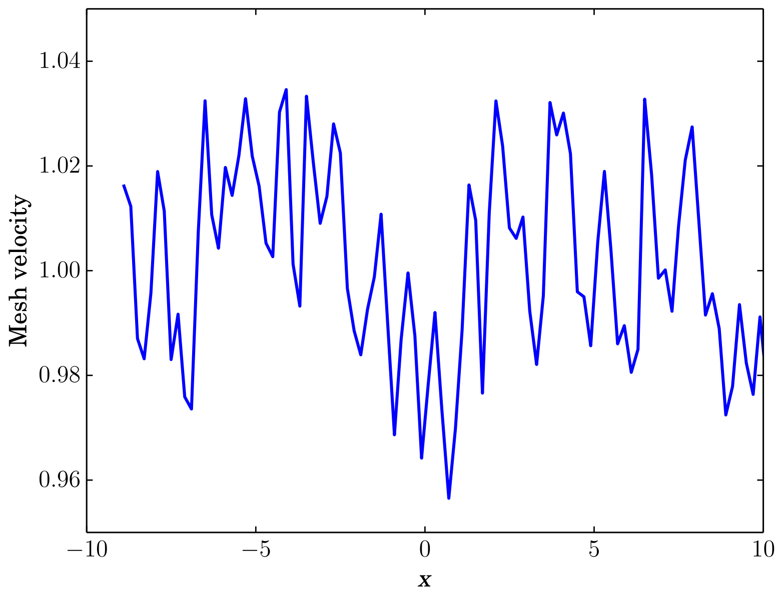


Fig. 5 Example of randomized velocity distribution for smooth test case

Table 6 Order of accuracy study on moving mesh using HLLC flux with randomly perturbed mesh velocity

N	$k=1$		$k=2$		$k=3$	
	Error	Rate	Error	Rate	Error	Rate
100	1.735E−02	–	1.798E−03	–	2.351E−04	–
200	4.179E−03	2.051	2.848E−04	2.676	1.416E−05	4.069
400	1.054E−03	2.035	4.301E−05	2.703	8.578E−07	4.041
800	2.615E−04	1.943	6.012E−06	2.838	5.476E−08	3.958
1 600	7.279E−05	1.852	8.000E−07	2.909	3.505E−09	3.966

Table 7 Order of accuracy study on fixed mesh using Roe flux with non-constant velocity smooth test case

N	$k=1$		$k=2$	
	Error	Rate	Error	Rate
100	8.535E−03	–	1.033E−03	–
200	1.958E−03	2.124	1.221E−04	3.08
400	4.721E−04	2.052	1.581E−05	2.95
800	1.238E−04	1.931	2.14E−06	2.89
1 600	3.563E−05	1.796	2.63E−07	3.02

9.2 Smooth Test Case with Non-constant Velocity

We also test the accuracy of our schemes on a isentropic problem with smooth solutions. In the test case, the initial conditions are given by

$$\rho(x,0)=1+0.999\,999\,5\sin(\pi x),\quad u(x,0)=0,\quad p(x,0)=\rho^{\gamma}(x,0) \tag{11}$$

with $\gamma=3$ and periodic boundary conditions. For this kind of special isentropic problem, the Euler equations are equivalent to the two Burgers equations in terms of their two Riemann invariants which can then be used to derive the analytical solution. The errors are then computed with respect to the given analytical solution. In contrast to the previous test

Table 8 Order of accuracy study on moving mesh using Roe flux with non-constant velocity smooth test case

N	$k=1$		$k=2$	
	Error	Rate	Error	Rate
100	4.235E−03	–	2.238E−04	–
200	1.058E−03	2.001	3.255E−05	2.87
400	2.586E−04	2.035	4.301E−05	3.133
800	5.804E−05	2.155	5.762E−06	2.901
1 600	1.271E−05	2.192	7.401E−07	2.96

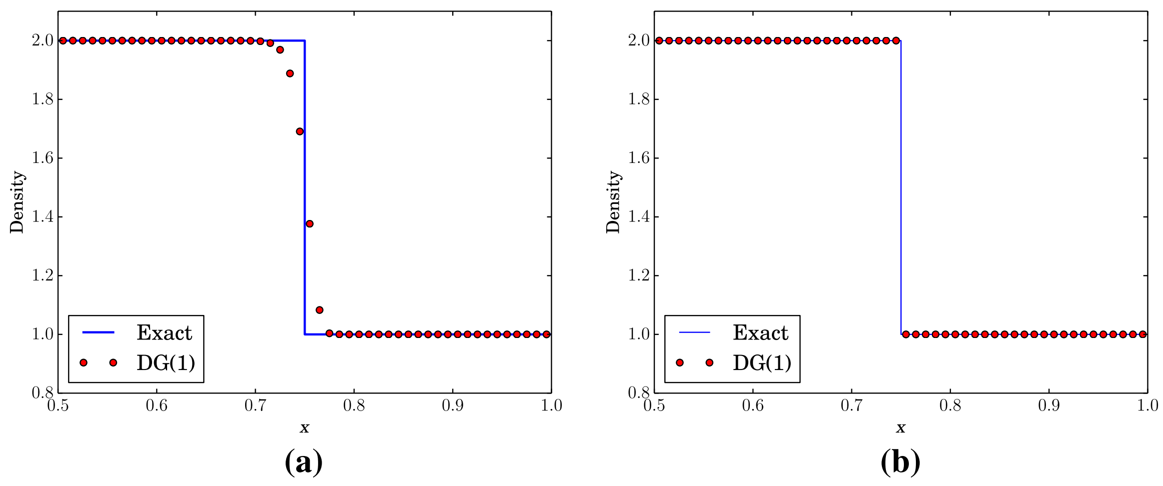


Fig. 6 Single contact wave using Roe flux and 100 cells: **a** static mesh, **b** moving mesh

case, the velocity and pressure are not constant which makes this a more challenging test case. We run the simulation with a WENO-type limiter from [28] and the positivity limiter enabled. As we can see from Tables 7 and 8, the rate of convergence is maintained for the moving mesh method with the moving mesh methods exhibiting much lower errors.

9.3 Single Contact Wave

In this example, we choose a Riemann problem which gives rise to a single contact wave in the solution that propagates with a constant speed. The initial condition is given by

$$(\rho, v, p) = \begin{cases} (2.0, 1.0, 1.0), & \text{if } x < 0.5, \\ (1.0, 1.0, 1.0), & \text{if } x > 0.5, \end{cases}$$

and the contact wave moves with a constant speed of 1.0. The solutions on static and moving meshes are shown in Fig. 6 at time $t = 0.5$ using the Roe flux. The moving mesh is able to exactly resolve the contact wave while the static mesh scheme adds considerable numerical dissipation that smears the discontinuity over many cells. The accurate resolution of contact waves is a key advantage of such moving mesh methods, which are capable of giving very good resolution of the contact discontinuity even on coarse meshes.

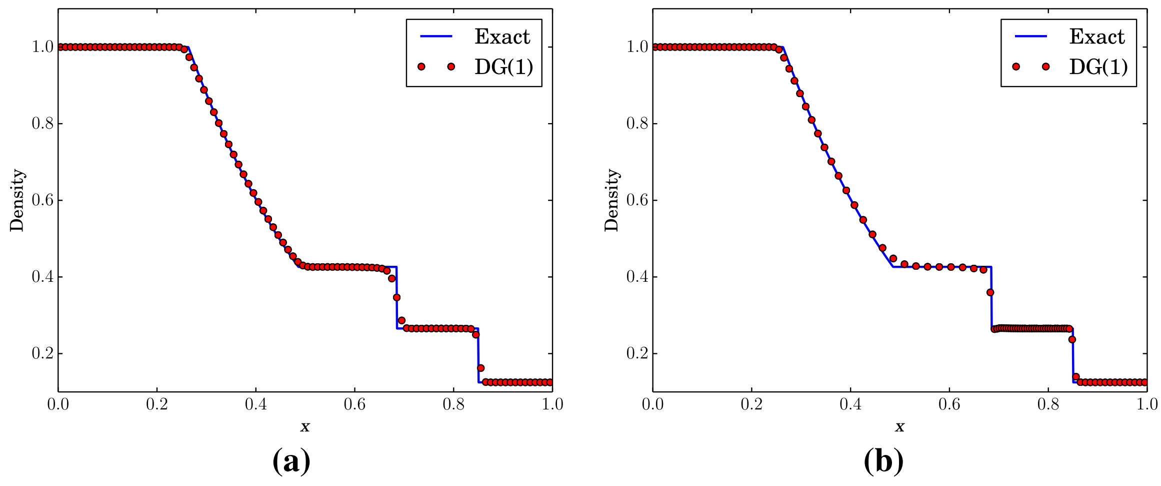


Fig. 7 Sod problem using Roe flux, 100 cells and TVD limiter: **a** static mesh, **b** moving mesh

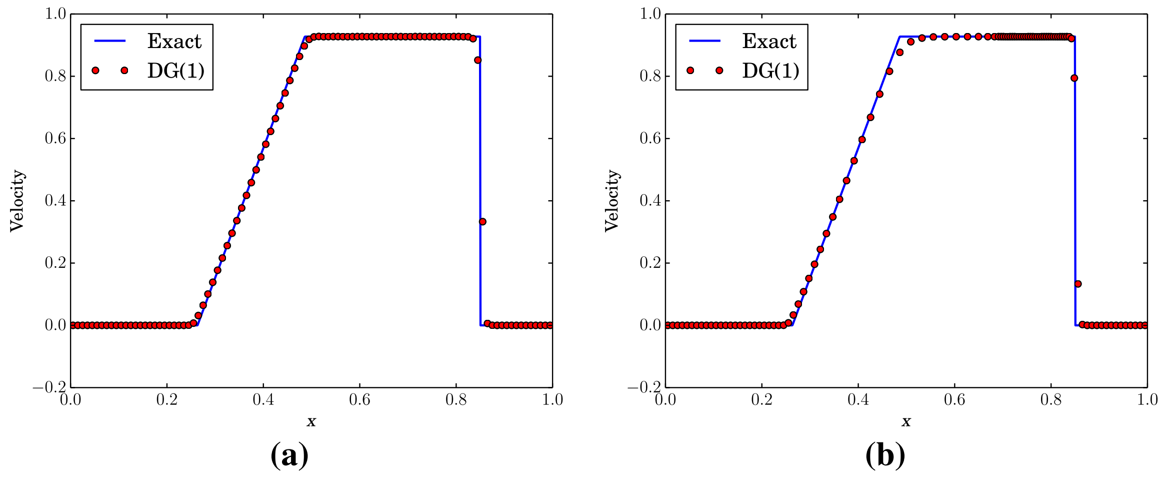


Fig. 8 Sod problem using Roe flux, 100 cells and TVD limiter: **a** static mesh, **b** moving mesh

9.4 Sod Problem

The initial condition for the Sod test case is given by [30]

$$(\rho, v, p) = \begin{cases} (1.0, 0, 1.0), & \text{if } x < 0.5, \\ (0.125, 0, 0.1), & \text{if } x > 0.5, \end{cases}$$

and the solution is computed up to a final time of $T = 0.2$ with the computational domain being [0, 1]. Since the fluid velocity is zero at the boundary, the computational domain does not change with time for the chosen final time. The exact solution consists of a rarefaction fan, a contact wave and a shock wave. In Fig. 7, we show the results obtained using the Roe flux with 100 cells and the TVD limiter on static and moving meshes. The contact wave is considerably well resolved on the moving mesh as compared to the static mesh due to reduced numerical dissipation on moving meshes (Figs. 8 and 9).

To study the Galilean invariance or the dependence of the solution on the choice of coordinate frame, we add a boost velocity of $V = 10$ or $V = 100$ to the coordinate frame, which implies that the initial fluid velocity is $v(x, 0) = V$ and the other quantities remain as before. Figure 10a shows that the accuracy of the static mesh results degrades

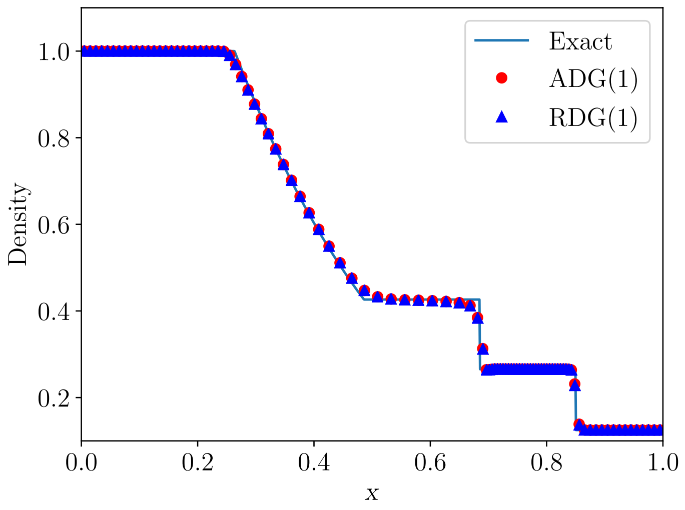


Fig. 9 Sod problem using Roe flux, 100 cells and TVD limiter. *ADG* average velocity, *RDG* linearized riemann velocity

Table 9 Number of iterations required to reach time $t = 0.2$ for Sod test for different boost velocity of the coordinate frame

V	0	10	100
Static mesh	144	810	6 807
Moving mesh	176	176	176

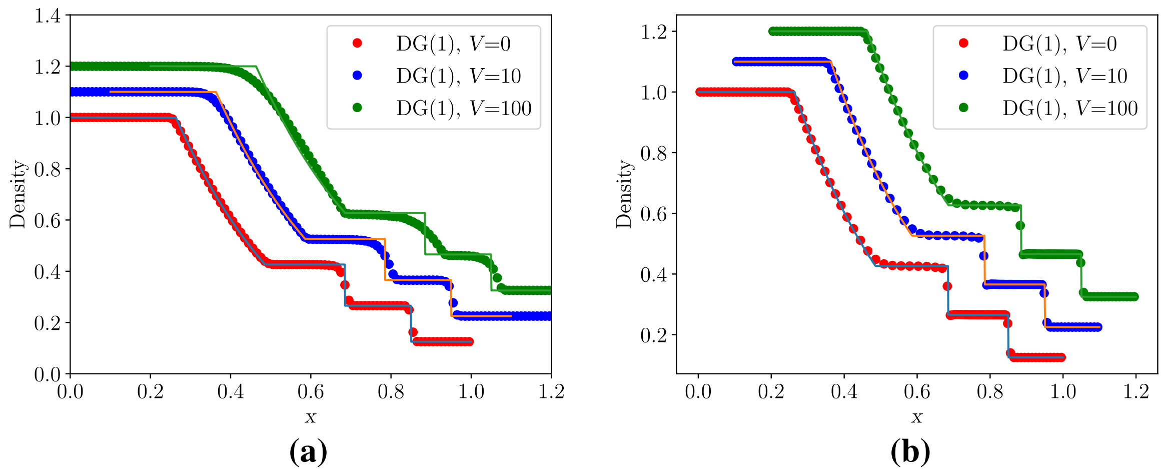


Fig. 10 Effect of coordinate frame motion on Sod problem using Roe flux, 100 cells and TVD limiter: **a** static mesh, **b** moving mesh

with increase in velocity of the coordinate frame, particularly the contact discontinuity is highly smeared. The results given in Fig. 10b clearly show the independence of the results on the moving mesh with respect to the coordinate frame velocity. The allowed time step from CFL condition decreases with increase in coordinate frame speed for the static mesh case, while in case of the moving mesh, it remains invariant. This means that in case of static mesh, we have to perform more time steps to reach the same final time as shown in Table 9, which increases the computational time. Thus, the moving mesh scheme has the additional advantage of allowing a larger time step compared to the fixed mesh scheme.

Finally, we compute the solutions using quadratic and cubic polynomials and the results are shown in Fig. 11. The solutions look similar to the case of linear polynomials and have the same sharp resolution of discontinuities.

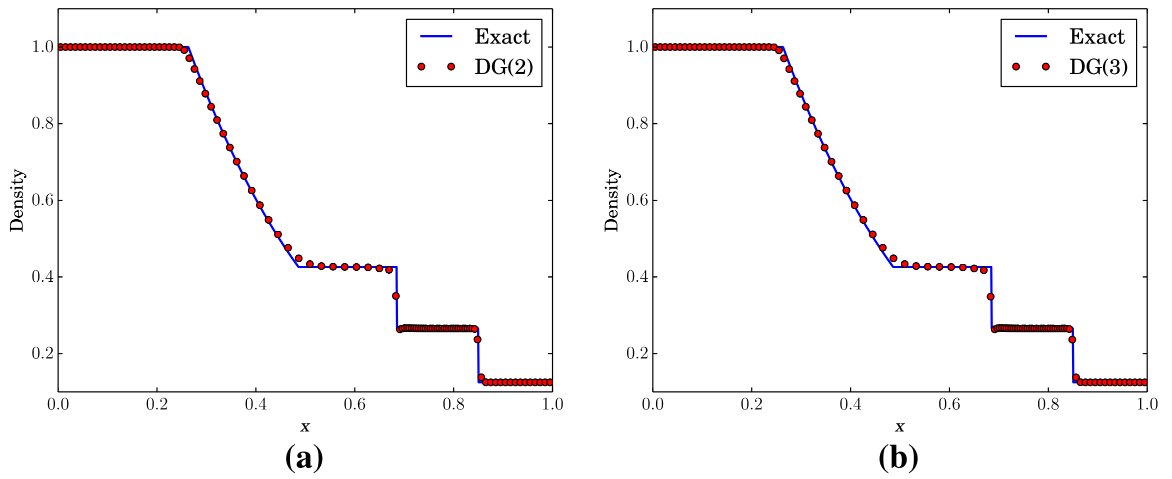


Fig. 11 Sod problem on moving mesh using Roe flux, 100 cells and TVD limiter: **a** Degree = 2, **b** Degree = 3

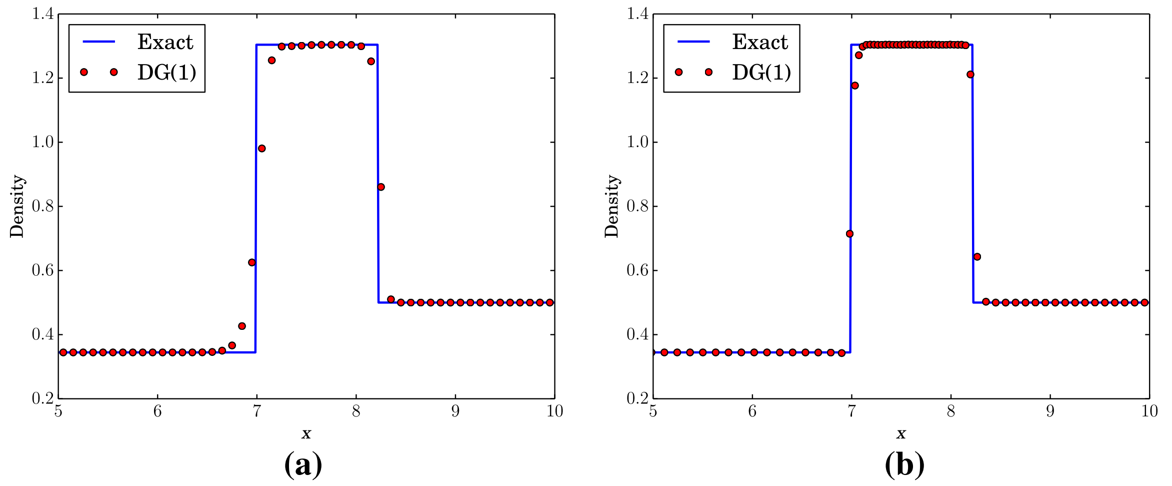


Fig. 12 Lax problem using HLLC flux, 100 cells and TVD limiter: **a** static mesh, **b** moving mesh

9.5 Lax Problem

The initial condition is given by

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

The computational domain is $[-10, +10]$ and we compute the solution up to a final time of $T = 1.3$. This problem has a strong shock and a contact wave that is difficult to resolve accurately. The zoomed view of density is shown at the final time in Fig. 12, and we observe that the moving mesh results are more accurate for the contact wave, which is the first discontinuity in the figure. The second discontinuity is a shock which is equally well resolved in both cases. We can observe that the grid is automatically clustered in the region between the contact and shock wave, but no explicit grid adaptation was used in this simulation (Fig. 13).

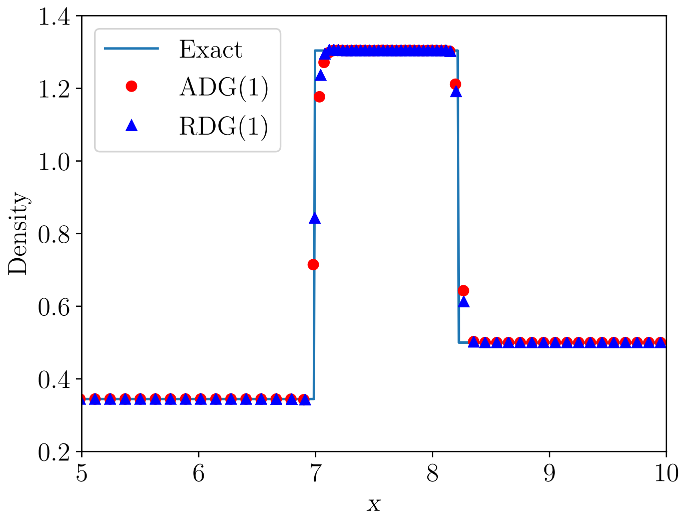


Fig. 13 Lax problem using HLLC flux, 100 cells and TVD limiter. *ADG* average velocity, *RDG* linearized Riemann velocity

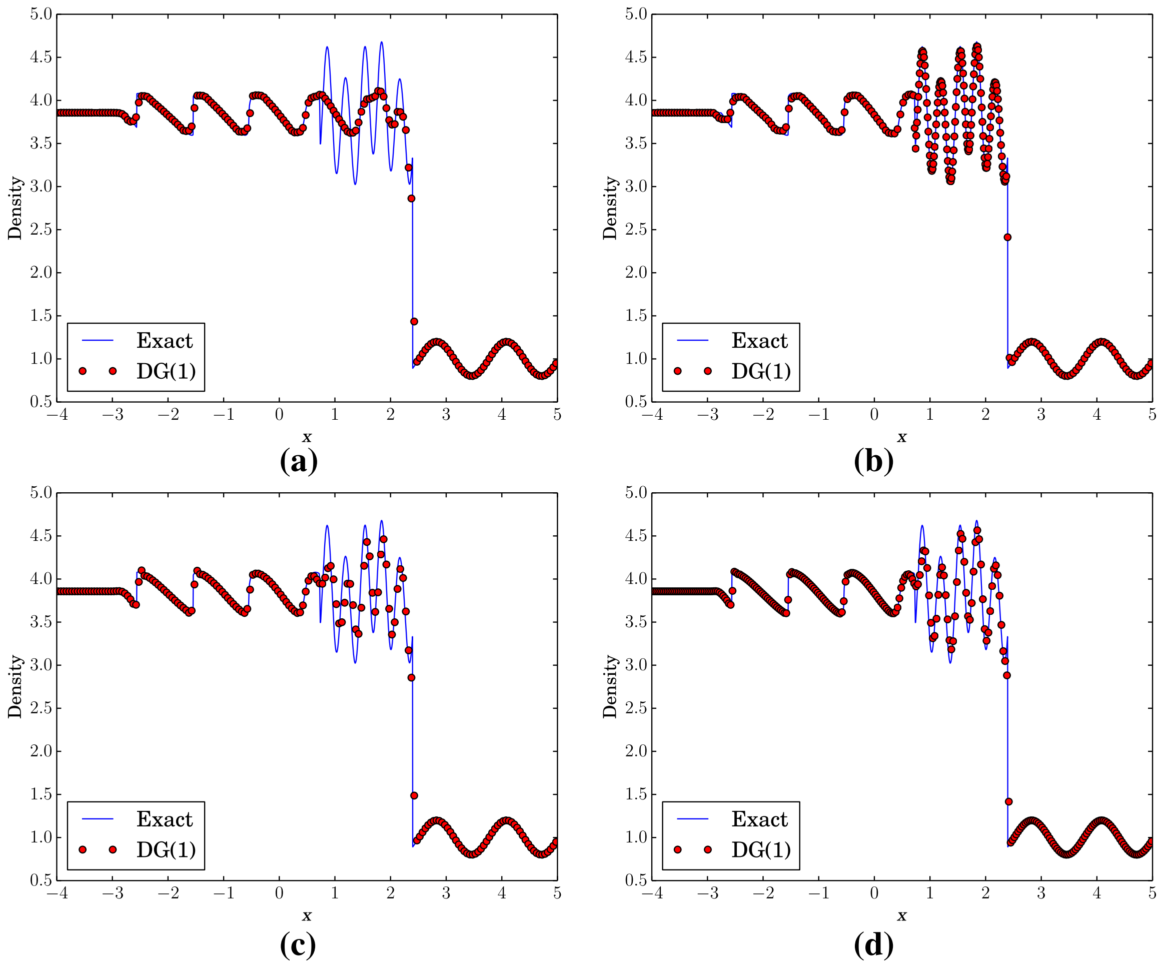


Fig. 14 Shu–Osher problem using Roe flux: **a** static mesh, 200 cells, $M = 0$, **b** moving mesh, 200 cells, $M = 0$, **c** static mesh, 200 cells, $M = 100$, **d** static mesh, 300 cells, $M = 100$

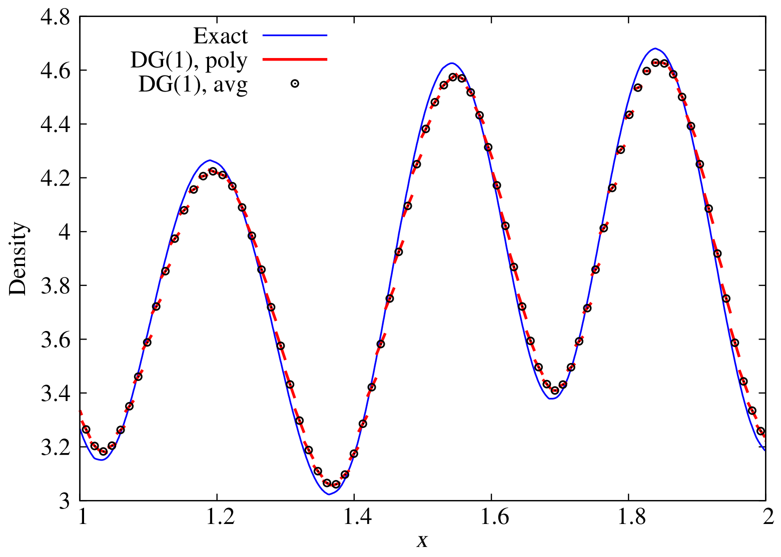


Fig. 15 Shu–Osher problem using Roe flux on moving mesh

9.6 Shu–Osher Problem

The initial condition is given by [31]

$$(\rho, v, p) = \begin{cases} (3.857\,143, 2.629\,369, 10.333\,333), & \text{if } x < -4, \\ (1 + 0.2\sin(5x), 0, 1.0), & \text{if } x > -4, \end{cases}$$

which involves a smooth sinusoidal density wave which interacts with a shock. The domain is $[-5, +5]$ and the solution is computed up to a final time of $T = 1.8$. The solutions are shown in Fig. 14a, b on static and moving meshes using 200 cells and the TVD limiter. The moving mesh scheme is considerably more accurate in resolving the sinusoidal wave structure that arises after interaction with the shock. In Fig. 14c, we compute the solution on static mesh with the TVB limiter and the parameter $M = 100$ in (8). In this case, the solutions on static mesh are more accurate compared to the case of the TVD limiter but still not as good as the moving mesh results. The moving mesh result has more than 200 cells in the interval $[-5, +5]$ at the final time since cells enter the domain from the left side. Hence in Fig. 14d, we show the static mesh results with 300 cells and using the TVB limiter. The results are further improved for the static mesh case but still not as accurate as the moving mesh case. The choice of parameters in the TVB limiter is very critical but we do not have a rigorous algorithm to choose a good value for this. Hence, it is still advantageous to use the moving mesh scheme which gives improved solutions even with the TVD limiter.

The above results show that the ALE method is very accurate in terms of the cell averages. In Fig. 15, we show a zoomed view of density and pressure, where we also plot the linear polynomial solution. The slope of the solution is not accurately predicted with the Roe scheme and there are spurious contact discontinuities as the pressure and velocity are nearly continuous. This behaviour is observed with all contact preserving fluxes like Roe, HLLC and HLL-CPS but not with the Rusanov flux. Due to the almost Lagrangian character of the scheme, the eigenvalue corresponding to the contact wave, $\lambda_2 = v - w$, is nearly zero, which leads to the loss of dissipation in the corresponding characteristic field. If a spurious contact wave is generated during the violent dynamics, then this wave will be preserved by the scheme leading to wrong solutions. We modify the Roe scheme by preventing this eigenvalue from becoming too small or zero, which is similar to the approach

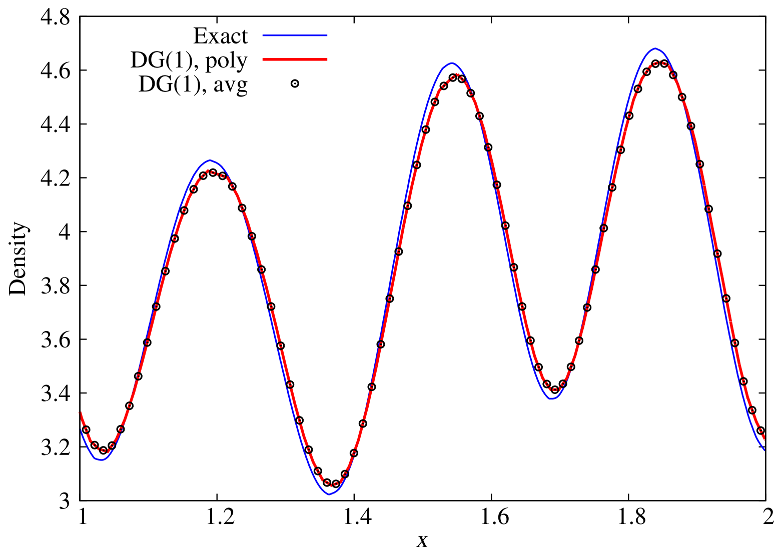


Fig. 16 Shu–Osher problem using modified Roe flux on moving mesh

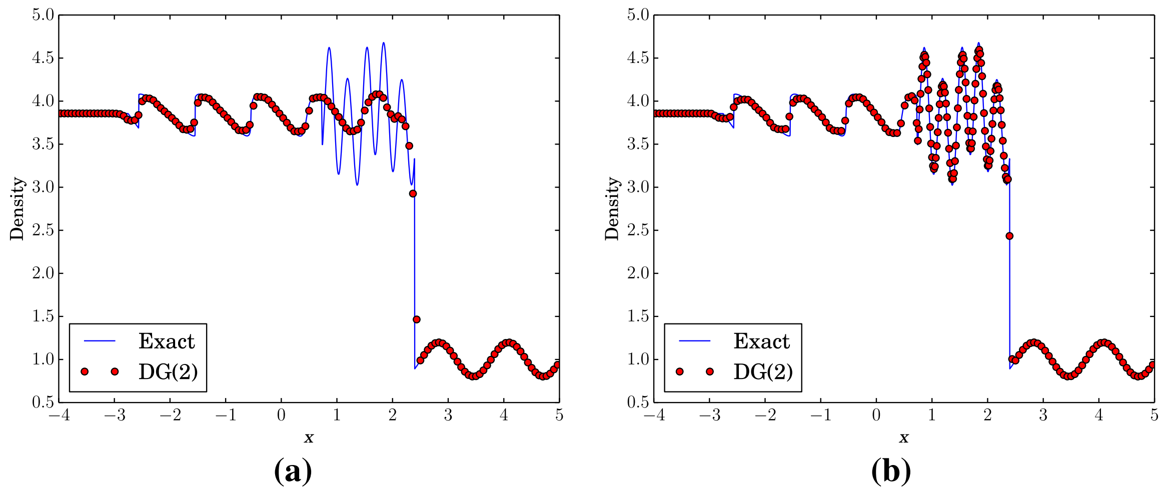


Fig. 17 Shu–Osher problem using the modified Roe flux, the TVD limiter, quadratic polynomials and 150 cells. **a** static mesh, **b** moving mesh

used for the entropy fix. The eigenvalue $|\lambda_2|$ used in the dissipative part of the Roe flux is determined from

$$|\lambda_2| = \begin{cases} |v-w|, & \text{if } |v-w| > \delta = \alpha c, \\ \frac{1}{2}(\delta + |v-w|^2/\delta), & \text{otherwise.} \end{cases}$$

With this modification and using $\alpha = 0.1$, the solution on the moving mesh is shown in Fig. 16 and we do not observe the spurious contact discontinuities which arise with the standard Roe flux, while at the same time, the solution accuracy compares favourably with the previous results that did not use the eigenvalue fix.

We next compute the solutions using quadratic polynomials. Figure 17 shows the results obtained with the TVD limiter which shows the dramatically better accuracy that is achieved on the moving mesh compared to static mesh. In Fig. 18, we perform the same computation with a WENO limiter taken from [29]. The static mesh results are now improved over the case of the TVD limiter but still not as good as the moving mesh results in terms of capturing the extrema. In Fig. 19, we show a zoomed view of the results on the

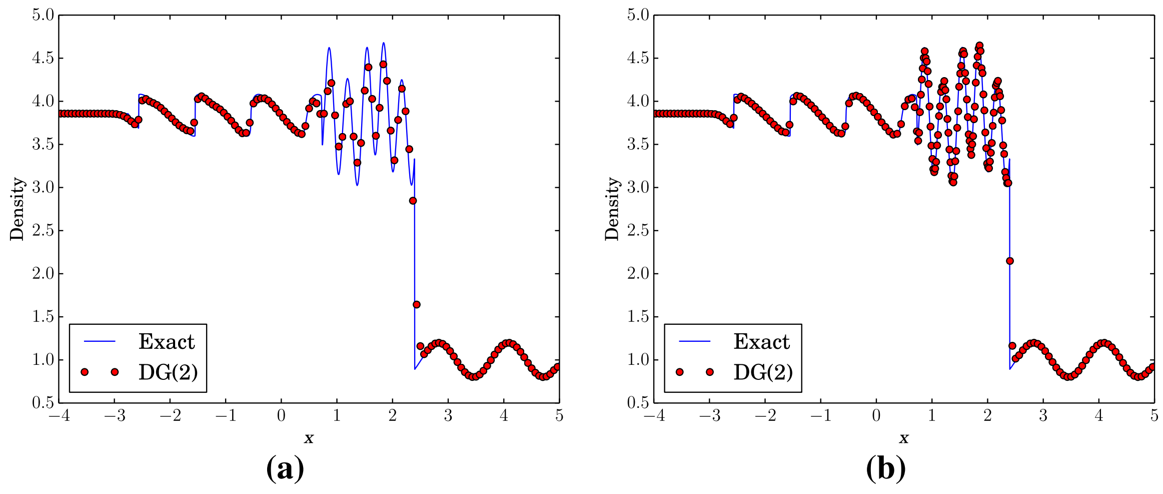


Fig. 18 Shu–Osher problem using modified Roe flux, WENO limiter, quadratic polynomials and 150 cells. **a** Static mesh, **b** moving mesh

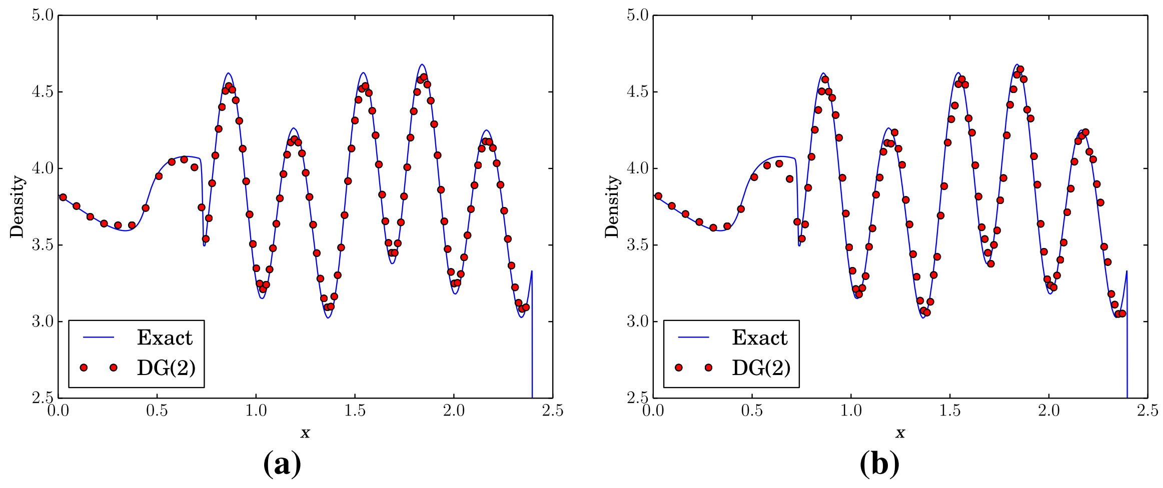


Fig. 19 Shu–Osher problem using modified Roe flux, moving mesh, quadratic polynomials and 150 cells. **a** TVD limiter, **b** WENO limiter

moving mesh with TVD and WENO limiters. We see that the TVD limiter is also able to capture all the features and is almost comparable to the WENO limiter.

9.7 Titarev–Toro Problem

Titarev–Toro problem is an extension of the Shu–Osher problem [32] to test a severely oscillatory wave interacting with a shock wave. It aims to test the ability of higher order methods to capture the extremely high-frequency waves. The initial condition is given by

$$(\rho, v, p) = \begin{cases} (1.515\,695, 0.523\,346, 1.805), & -5 < x \le -4.5, \\ (1 + 0.1\sin(20\pi x), 0, 1), & -4.5 < x \le 5. \end{cases} \tag{12}$$

The computation is carried out on a mesh of 1 000 cells with the final time $T = 5$ and the density at this final time is shown in Figs. 20 and 21. The fixed mesh is not able to resolve the high-frequency oscillations due to dissipation in the fluxes and the TVD limiter, but the ALE scheme gives an excellent resolution of these high-frequency oscillations. Note that

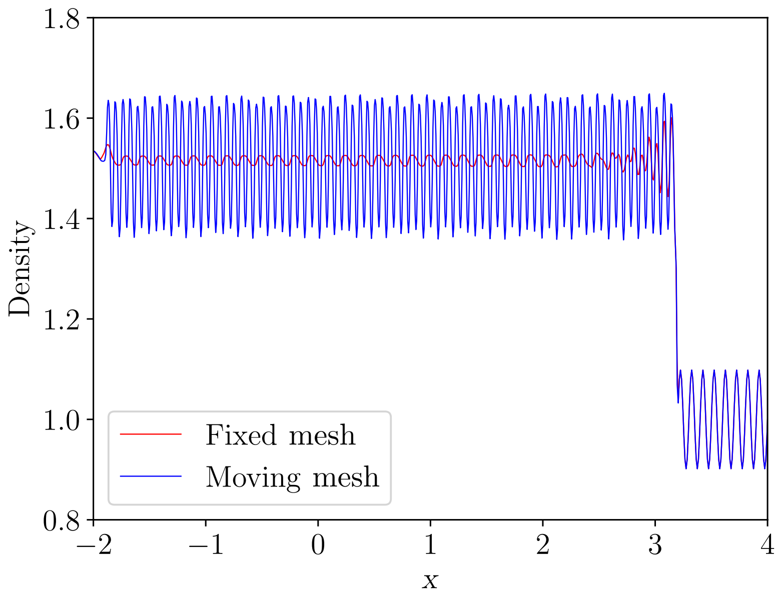


Fig. 20 Titarev problem with HLLC flux, 1 000 cells and TVD limiter

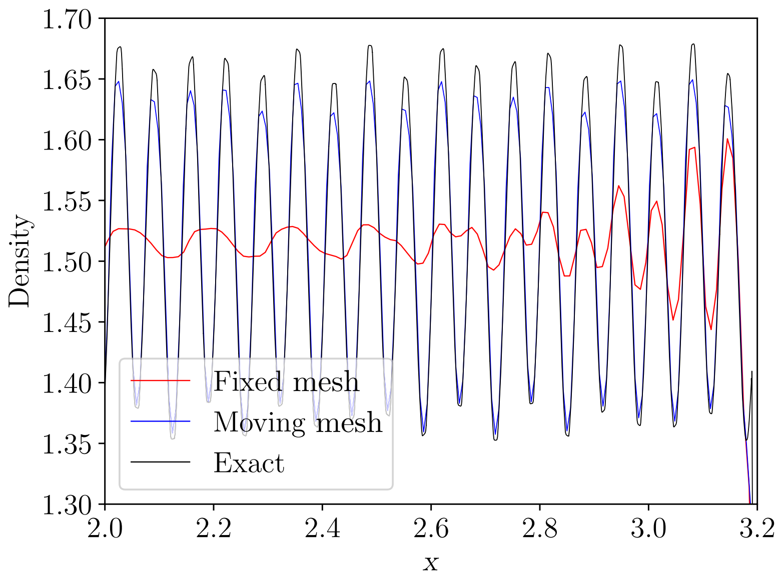


Fig. 21 Titarev problem with HLLC flux, 1 000 cells and TVD limiter (zoomed version)

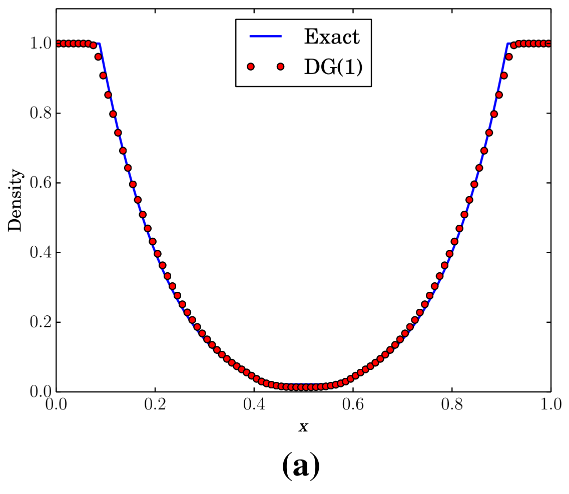
the ALE scheme also uses the same TVD limiter but it is still able to resolve the solution to a very degree of accuracy. This result again demonstrates the superior accuracy that can be achieved using a nearly Lagrangian ALE scheme in problems involving interaction of shocks and smooth flow structures.

9.8 123 Problem

The initial condition is given by [33]

$$(\rho, v, p) = \begin{cases} (1.0, -2.0, 0.4), & x < 0.5, \\ (1.0, +2.0, 0.4), & x > 0.5. \end{cases}$$

The computational domain is [0, 1] and the final time is $T = 0.15$. The density using 100 cells is shown in Fig. 22 with static and moving meshes. The mesh motion does not significantly improve the solution compared to the static mesh case since the solution is smooth.



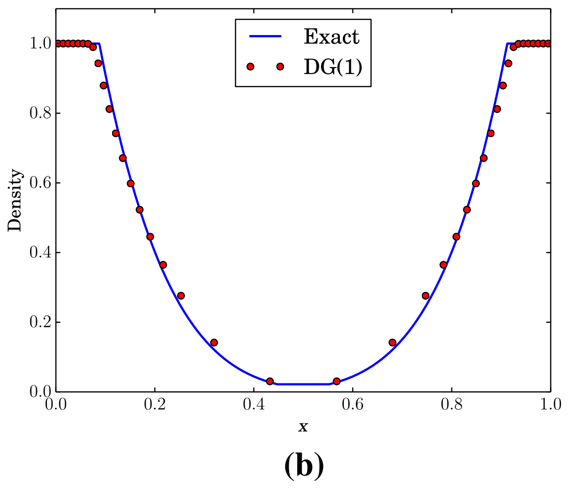
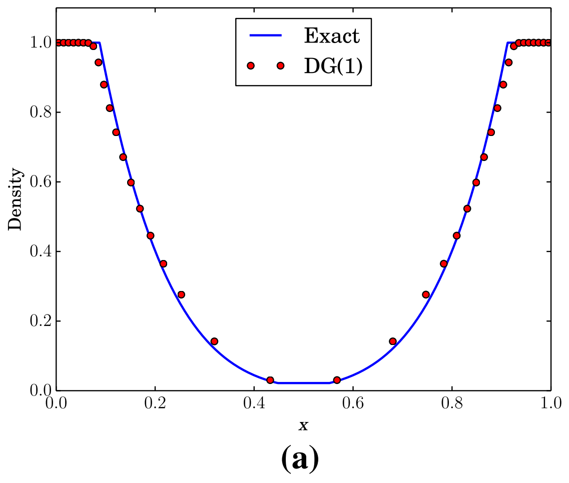


Fig. 22 123 problem using HLLC flux and 100 cells: **a** static mesh, **b** moving mesh



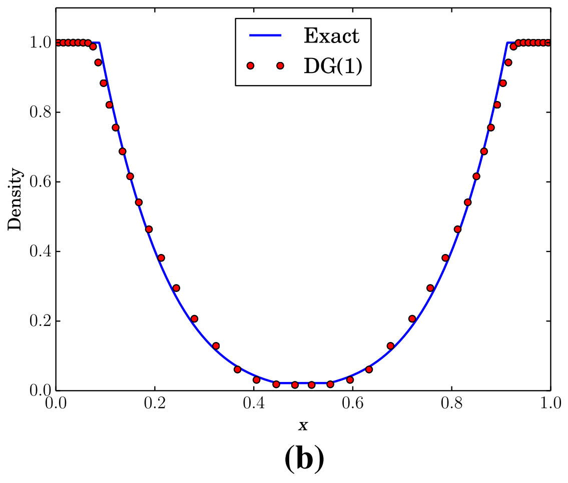
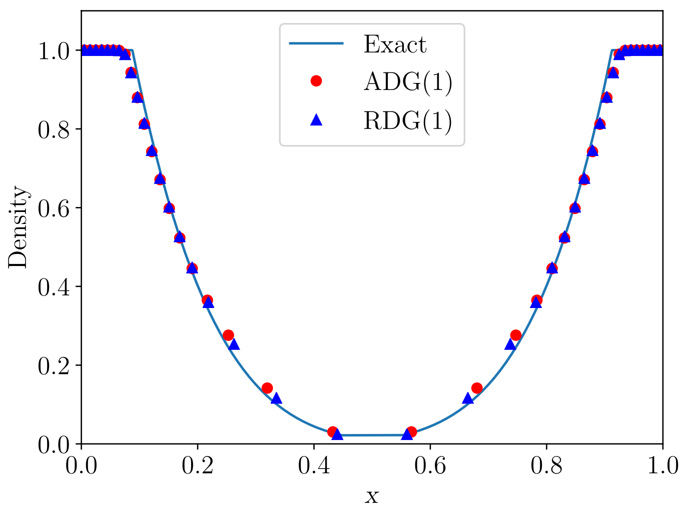


Fig. 23 123 problem using HLLC flux and grid refinement: **a** static mesh, **b** moving mesh with mesh adaptation ($h_{\max}$ = 0.05) leading to 108 cells at final time

Fig. 24 123 problem using HLLC flux, 100 cells and TVD limiter. *ADG* average velocity, *RDG* linearized Riemann velocity



On the contrary, the mesh becomes rather coarse in the expansion region, though the solution is still well resolved. However, severe expansion may lead to very coarse meshes which may be undesirable. To prevent very coarse cells, we switch on the mesh refinement

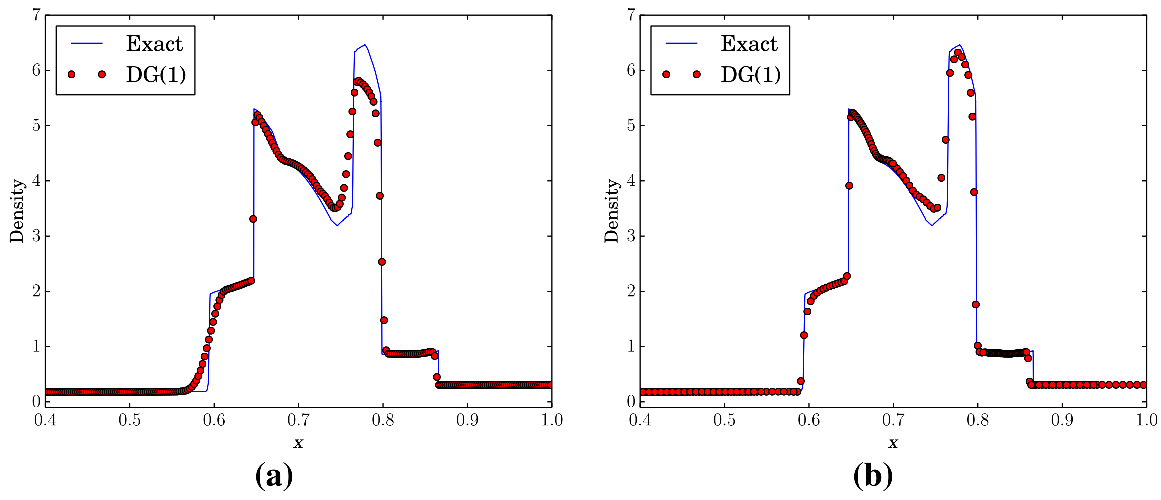


Fig. 25 Blast problem using HLLC flux and 400 cells. **a** Static mesh, **b** moving mesh with adaptation ($h_{\min} = 0.001$) leading to 303 cells at final time

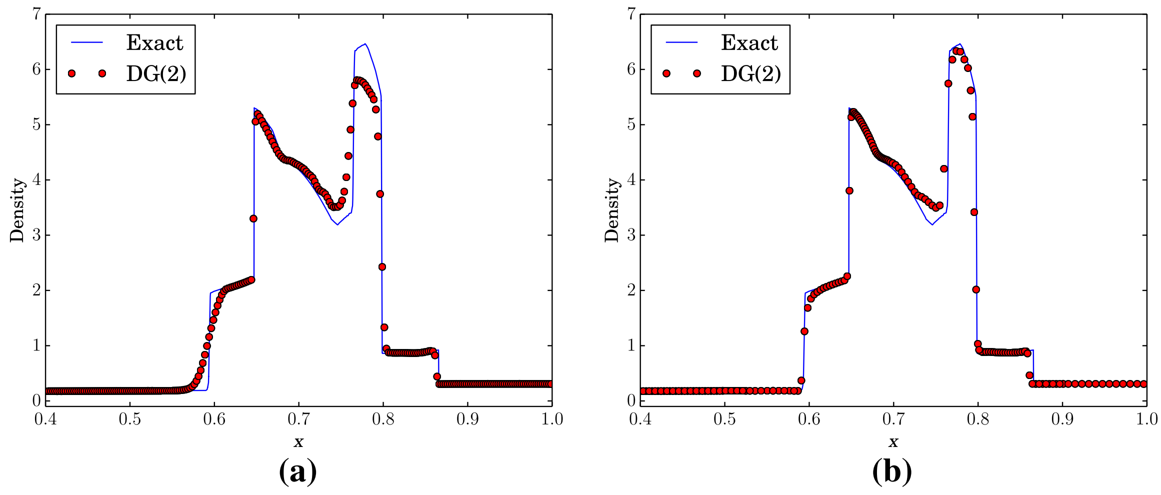


Fig. 26 Blast problem using HLLC flux, quadratic polynomials and 400 cells. **a** Static mesh, **b** moving mesh with adaptation ($h_{\min} = 0.001$) leading to 293 cells at final time

algorithm as described before and use the upper bound on the mesh size as $h_{\max} = 0.05$. The resulting solution is shown in Fig. 23 where the number of cells has increased to 108 at the time shown. The central expansion region is now resolved by more uniformly sized cells compared to the case of no grid refinement (Fig. 24).

9.9 Blast Problem

The initial condition is given by [34]

$$(\rho, v, p) = \begin{cases} (1.0, 0, 1\,000.0), & x < 0.1, \\ (1.0, 0, 0.01), & 0.1 < x < 0.9, \\ (1.0, 0, 100.0), & x > 0.9 \end{cases}$$

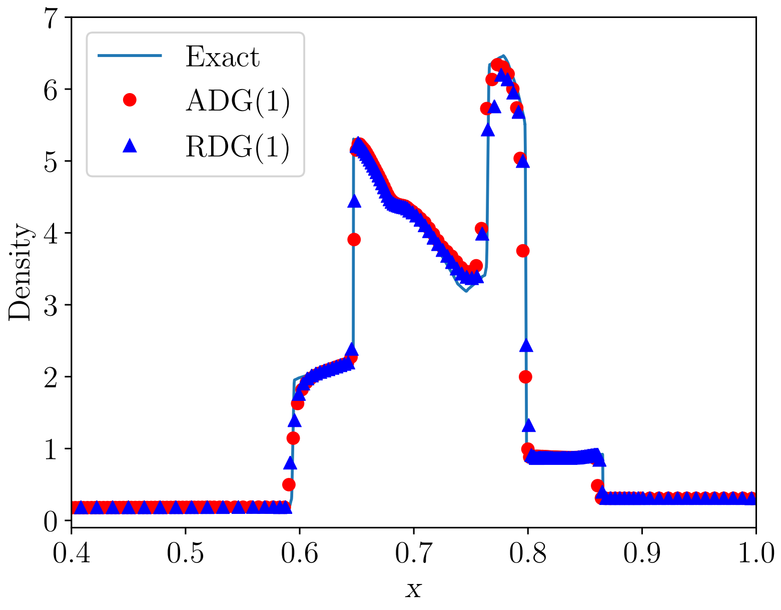


Fig. 27 Blast problem using HLLC flux, 100 cells and TVD limiter. *ADG* average velocity, *RDG* linearized Riemann velocity

with a domain of [0, 1] and the final time is $T = 0.038$. A reflective boundary condition is used at $x = 0$ and $x = 1$. A mesh of 400 cells is used for this simulation and in case of the moving mesh, we perform grid adaptation with $h_{\min} = 0.001$ since some cells become very small during the collision of the two shocks. The positivity preserving limiter of [28] is applied together with the TVD limiter and HLLC flux. The static mesh results shown in Fig. 25a indicate too much numerical viscosity in the contact wave around $x = 0.6$. This wave is more accurately resolved in the moving mesh scheme as seen in Fig. 25b which is an advantage due to the ALE scheme and is a very good indicator of the scheme accuracy as this is a very challenging feature to compute accurately. We next compute the same problem using quadratic polynomials with all other parameters being as before. The solutions are shown in Fig. 26 and indicate that the Lagrangian moving mesh scheme is more accurate in resolving the contact discontinuity. The higher polynomial degree does not show any major improvement in the solution compared to the linear case, which could be a consequence of the strong shock interactions present in this problem, see Fig. (4.11–4.12) in [29] and Fig. (3.7) in [35] in comparison to current results (Fig. 27).

9.10 Le Blanc Shock Tube Test Case

The Le Blanc shock tube test case is an extreme shock tube problem where the initial discontinuity separates a region of high energy and density from one of low energy and density. This is a much more severe test than the Sod problem and hence more challenging for numerical schemes. The computational domain is $0 \leqslant x \leqslant 9$ and is filled with an ideal gas with $\gamma = 5/3$. The gas is initially at rest and we perform the simulation up to a time of $T = 6$ units. The initial discontinuity is at $x = 3$ and the initial condition is given by

$$(\rho, v, p) = \begin{cases} (1.0, 0, 0.1), & \text{if } x < 3, \\ (0.001, 0, 10^{-7}), & \text{if } x > 3. \end{cases} \tag{13}$$

Note that both the density and pressure have a very large jump in the initial condition. The solution that develops from this initial condition consists of a rarefaction wave moving to the left and a contact discontinuity and a strong shock moving to the right. In Fig. 28, we

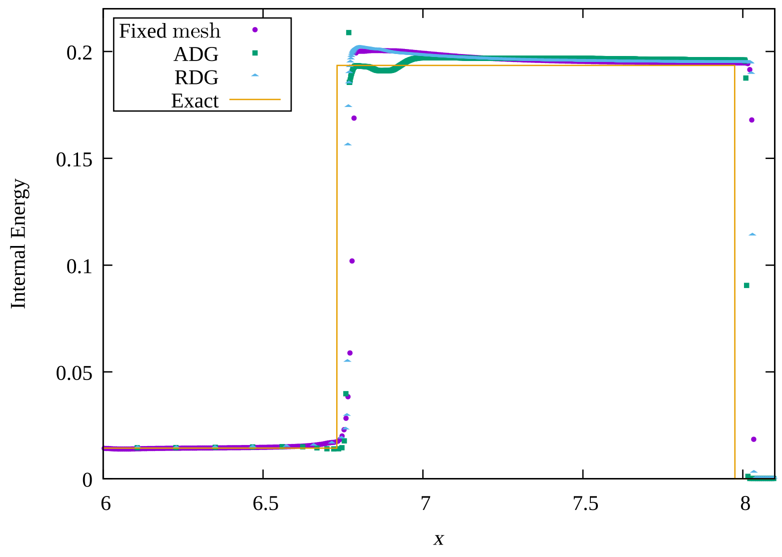


Fig. 28 Internal energy for Le Blanc shock tube with Rusanov flux, 1 400 cells and TVD limiter, *ADG* average velocity, *RDG* linearized Riemann velocity

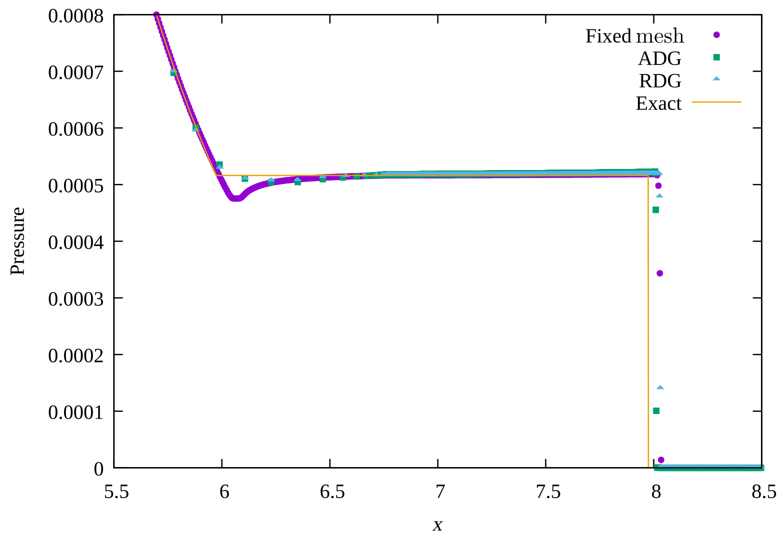


Fig. 29 Pressure for Le Blanc shock tube with Rusanov flux, 1 400 cells and TVD limiter, *ADG* average velocity, *RDG* linearized Riemann velocity

show the comparison of the internal energy profile at final time between a fixed mesh solution and moving mesh solutions with two different mesh velocities as described before. Most methods tend to generate a very large spike in the internal energy in the contact region, e.g., compare with Fig. (11) in [36], while the present ALE method here is able to give a better profile. We plot the pressure profile in Fig. 29 which shows that the ALE scheme is able to better represent the region around the contact wave as compared to fixed mesh method.

9.11 Two-Dimensional Isentropic Vortex Test Case

The extension to two dimensions involves two aspects that need to be addressed. The first issue is how to handle the grid motion and the second is how to formulate the ALE-DG scheme. The second part is a natural generalization of the DG scheme we have described

Table 10 Isentropic vortex in 2D: order of accuracy study on two-dimensional static mesh

N	$k=1$		$k=2$	
	Error	Rate	Error	Rate
50×50	2.230E−03	–	1.762E−04	–
100×100	5.987E−04	1.945	2.305E−05	2.934
200×200	1.498E−04	1.998	2.973E−06	2.955
400×400	3.786E−05	1.984	3.762E−07	2.982
800×800	9.617E−06	1.977	3.474E−08	2.991

for the 1-D case in this paper, except that we have to construct basis functions on triangles and perform some numerical quadrature. The first part involving grid movement is more complicated and we present only very preliminary results in this section to demonstrate that the idea has merit. We now consider the two-dimensional Euler equations written as

$$\frac{\partial u}{\partial t}+\frac{\partial f(u)}{\partial x}+\frac{\partial g(u)}{\partial y}=0, \tag{14}$$

where

$$u=\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ E\end{bmatrix},\quad f(u)=\begin{bmatrix}\rho u\\ p+\rho u^2\\ \rho uv\\ (E+p)u\end{bmatrix},\quad g(u)=\begin{bmatrix}\rho v\\ \rho uv\\ p+\rho v^2\\ (E+p)v\end{bmatrix}, \tag{15}$$

$$p=(\gamma-1)\left[E-\frac{1}{2}\rho(u^2+v^2)\right]. \tag{16}$$

The test case we consider involves an isentropic vortex that is advecting with the constant velocity and is a smooth solution for which error norms can be calculated. The test is carried out on a square domain $[-10,10]\times[-10,10]$ with periodic boundary conditions. The initial condition is an isentropic vortex (Table 10)

$$T=1-\frac{(\gamma-1)\beta^2}{8\gamma\pi^2}\mathrm{e}^{1-r^2}, \tag{17}$$

$$\rho=T^{\frac{1}{\gamma-1}}, \tag{18}$$

$$u=u_\infty-\frac{\beta}{2\pi}y\mathrm{e}^{\frac{1-r^2}{2}}, \tag{19}$$

$$v=v_\infty-\frac{\beta}{2\pi}y\mathrm{e}^{\frac{1-r^2}{2}}, \tag{20}$$

$$p=\rho^\gamma \tag{21}$$

with $u_\infty=1, v_\infty=0, \gamma=1.4, \beta=10$. As the solution evolves in time, the mesh becomes quite deformed because the vortex is continually shearing the mesh, which can lead to

Table 11 Isentropic vortex in 2D: order of accuracy study on two-dimensional moving mesh

N	$k=1$		$k=2$	
	Error	Rate	Error	Rate
50×50	2.230E−03	–	1.762E–04	–
100×100	5.987E−04	1.945	2.305E−05	2.934
200×200	1.498E−04	1.998	2.973E−06	2.955
400×400	3.786E−05	1.984	3.762E−07	2.982
800×800	9.617E−06	1.977	3.474E−08	2.991

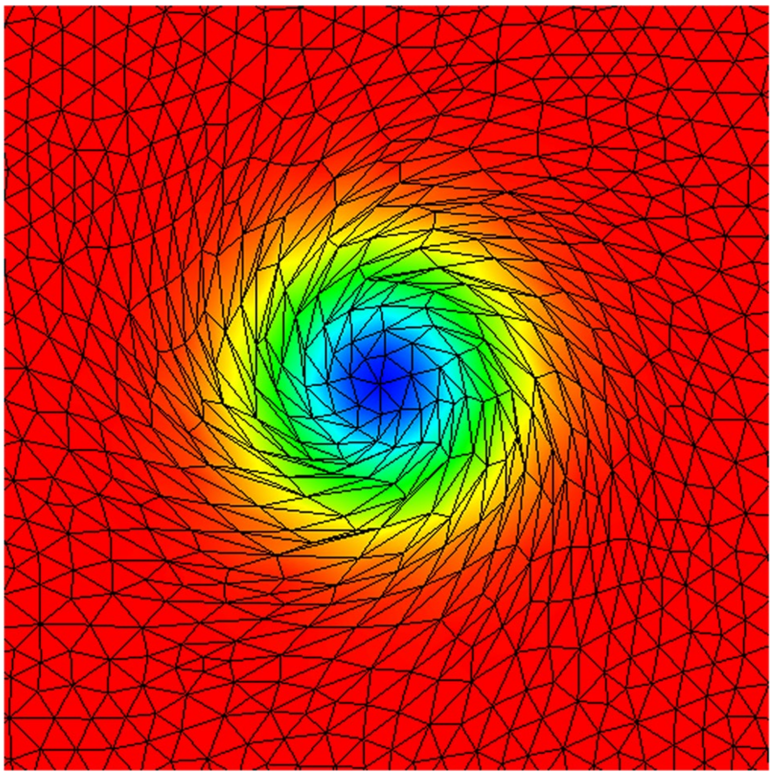


Fig. 30 Isentropic vortex in 2-D: skewed mesh without remeshing $t=2.660\,534$

degenerate meshes, as shown in Fig. 30. We avoid the occurrence of badly shaped triangles using a combination of face swapping and mesh velocity smoothing algorithms [37, 38]. The mesh modification is a very local procedure and does not require global remeshing which is a costly process. With these techniques, we are able to maintain a good mesh quality even after the vortex has rotated 4 times around its center as shown in Fig. 31. As the vortex is translating, we plot the solution in a window centered at the vortex center. We can see that the method maintains its high order of accuracy from the convergence rates of the error shown in Table 11; using linear basis functions yields second-order convergence while quadratic basis functions lead to third-order convergence.

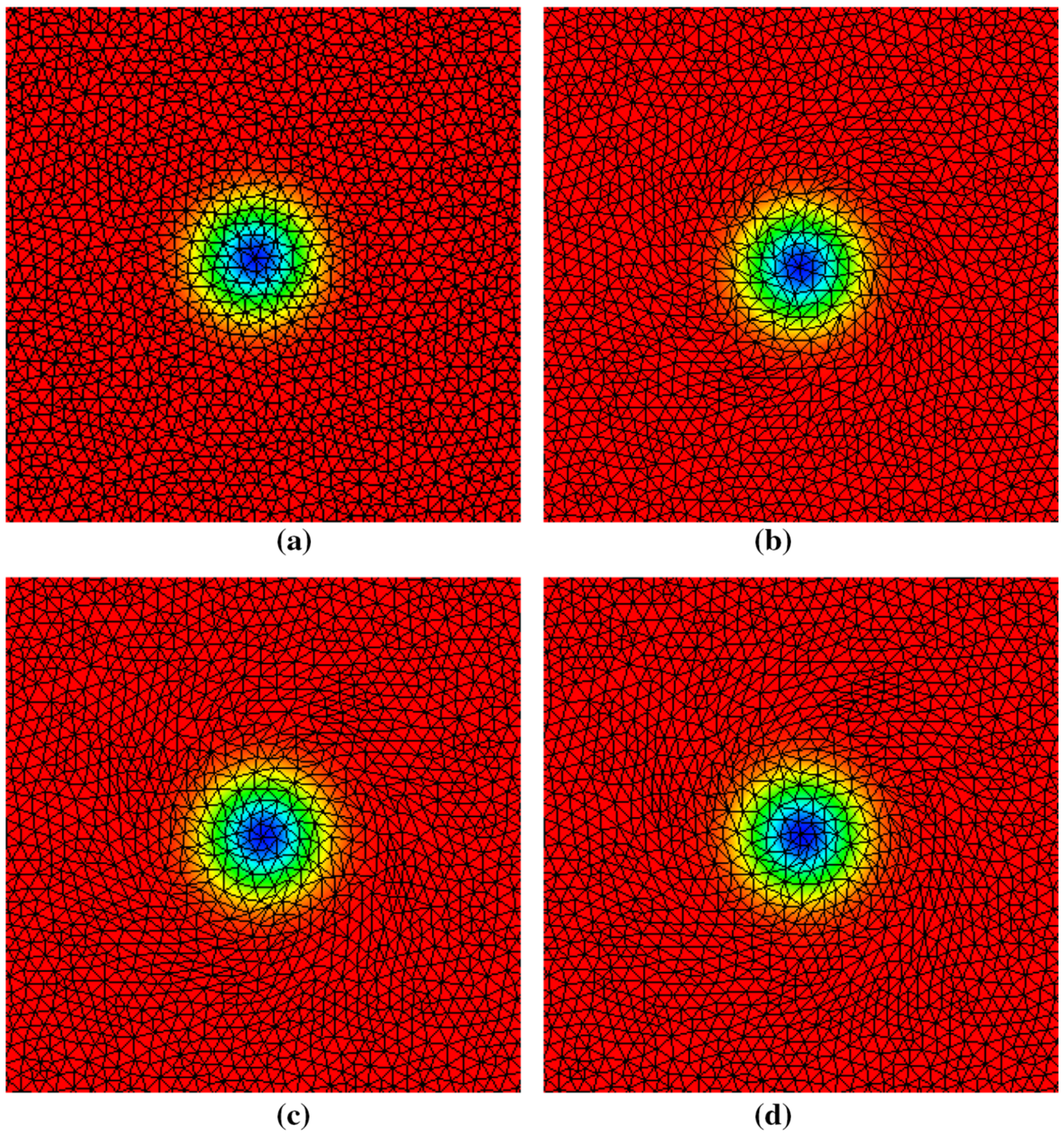


Fig. 31 Isentropic vortex in 2-D: mesh and pressure solution at various times **a** $t = 0$, **b** $t = 6$, **c** $t = 12$, **d** $t = 20$

10 Summary and Conclusions

We have developed an explicit DG scheme on moving meshes using ALE framework and space–time expansion of the solutions within each cell. The near Lagrangian nature of the mesh motion dramatically reduces the numerical dissipation, especially for contact waves. Even moving contact waves can be exactly computed with a numerical flux that is exact for stationary contact waves. The scheme is shown to yield superior results even in the presence of the large boost velocity of the coordinate system indicating its Galilean invariance property. The standard Roe flux does not suffer from entropy violation when applied in the current nearly Lagrangian framework. However, in some problems with strong shocks, spurious contact waves can appear and we propose to fix the dissipation in Roe-type schemes that eliminates this issue. The method yields accurate solutions even in combination with standard TVD limiters, where fixed grid methods perform poorly. The mesh motion provides automatic grid adaptation near shocks but may lead to very coarse cells inside expansion waves. A grid adaptation strategy is developed to handle the problem of very small or very large cells. The presence of the DG polynomials makes it easy to transfer the solution during grid adaptation without loss of accuracy. The proposed methodology is general enough to be applicable to other systems of conservation laws modelling fluid flows. The basic idea can be extended to multi-dimensions but additional considerations are required to maintain good mesh quality under fluid deformations. The preliminary results shown for the isentropic vortex are very promising for the 2-D case.

Acknowledgements This work was started when Jayesh Badwaik was a project assistant at TIFR-CAM, Bangalore. The first two authors gratefully acknowledge the financial support received from the Airbus Foundation Chair on Mathematics of Complex Systems established in TIFR-CAM, Bangalore, for carrying out this work. Christian Klingenberg acknowledges the support of the *Priority Program 1648*: *Software for Exascale Computing* by the German Science Foundation. On behalf of all authors, the corresponding author states that there is no conflict of interest.

Appendix A: Numerical Flux

The ALE scheme requires a numerical flux $\hat{\boldsymbol{g}}(\boldsymbol{u}_l, \boldsymbol{u}_r, w)$ which is usually based on some approximate Riemann solver. The numerical flux function is assumed to be consistent in the sense that

$$\hat{\boldsymbol{g}}(\boldsymbol{u}, \boldsymbol{u}, w) = \boldsymbol{g}(\boldsymbol{u}, w), \qquad \forall \boldsymbol{u} \in \mathbb{R}^3, w \in \mathbb{R}.$$

Since the ALE versions of the numerical fluxes are not so well known, here we list the formulae used in the present work.

Rusanov Flux

The Rusanov flux is a variant of the Lax–Friedrich flux and is given by

$$\hat{\boldsymbol{g}}(\boldsymbol{u}_l, \boldsymbol{u}_r, w) = \frac{1}{2}[\boldsymbol{g}(\boldsymbol{u}_l, w) + \boldsymbol{g}(\boldsymbol{u}_r, w)] - \frac{1}{2}\lambda_{lr}(\boldsymbol{u}_r - \boldsymbol{u}_l),$$

where $\lambda_{lr} = \lambda(\boldsymbol{u}_l, \boldsymbol{u}_r, w)$,

$$\lambda(\boldsymbol{u}_l, \boldsymbol{u}_r, w) = \max\{|v_l - w| + c_l, |v_r - w| + c_r\},$$

which is an estimate of the largest wave speed in the Riemann problem. Since the mesh velocity is close to the fluid velocity, the value of λ is close to the local sound speed. Thus, the numerical dissipation is independent of the velocity scale.

Roe Flux

The Roe scheme [39] is based on a local linearization of the conservation law and then exactly solving the Riemann problem for the linear approximation. The flux can be written as

$$\hat{\boldsymbol{g}}(\boldsymbol{u}_l, \boldsymbol{u}_r, w) = \frac{1}{2}[\boldsymbol{g}(\boldsymbol{u}_l, w) + \boldsymbol{g}(\boldsymbol{u}_r, w)] - \frac{1}{2}|A_w|(\boldsymbol{u}_r - \boldsymbol{u}_l),$$

where the Roe average matrix $A_w = A_w(\boldsymbol{u}_l, \boldsymbol{u}_r)$ satisfies

$$\boldsymbol{g}(\boldsymbol{u}_r, w) - \boldsymbol{g}(\boldsymbol{u}_l, w) = A_w(\boldsymbol{u}_r - \boldsymbol{u}_l),$$

and we define $|A_w| = R|\Lambda - wI|R^{-1}$. This matrix is evaluated at the Roe average state $\boldsymbol{u}(\bar{\boldsymbol{q}})$, $\bar{\boldsymbol{q}} = \frac{1}{2}(\boldsymbol{q}_l + \boldsymbol{q}_r)$, where $\boldsymbol{q} = \sqrt{\rho}[1,\ v,\ H]^{\mathrm{T}}$ is the parameter vector introduced by Roe.

HLLC Flux

This is based on a three wave approximate Riemann solver and the particular ALE version we use can also be found in [15]. Define the relative velocity $q = v - w$; then, the numerical flux is given by

$$\hat{\boldsymbol{g}}(\boldsymbol{u}_l, \boldsymbol{u}_r, w) = \begin{cases} \boldsymbol{g}(\boldsymbol{u}_l, w), & S_l > 0, \\ \boldsymbol{g}^*(\boldsymbol{u}_l^*, w), & S_l \leq 0 < S_M, \\ \boldsymbol{g}^*(\boldsymbol{u}_r^*, w), & S_M \leq 0 \leq S_r, \\ \boldsymbol{g}(\boldsymbol{u}_r, w), & S_r < 0, \end{cases}$$

where the intermediate states are given by

$$\boldsymbol{u}_\alpha^* = \frac{1}{S_\alpha - S_M} \begin{bmatrix} (S_\alpha - q_\alpha)\rho_\alpha \\ (S_\alpha - q_\alpha)(\rho v)_\alpha + p^* - p_\alpha \\ (S_\alpha - q_\alpha)E_\alpha - p_\alpha q_\alpha + p^* S_M \end{bmatrix}, \qquad \alpha = l, r,$$

and

$$\boldsymbol{g}^*(\boldsymbol{u}, w) = S_M \boldsymbol{u} + \begin{bmatrix} 0 \\ p^* \\ (S_M + w)p^* \end{bmatrix},$$

where

$$p^* = \rho_l(q_l - S_l)(q_l - S_M) + p_l = \rho_r(q_r - S_r)(q_r - S_M) + p_r,$$

which gives S_M as

$$S_M = \frac{\rho_r q_r(S_r - q_r) - \rho_l q_l(S_l - q_l) + p_l - p_r}{\rho_r(S_r - q_r) - \rho_l(S_l - q_l)}.$$

The signal velocities are defined as

$$S_l = \min\{q_l - c_l, \hat{v} - w - \hat{c}\}, \qquad S_r = \max\{q_r + c_r, \hat{v} - w + \hat{c}\},$$

where $\hat{v}$, $\hat{c}$ are Roe's average velocity and speed of sound.

Appendix B: Continuous Expansion Runge–Kutta (CERK) Schemes

We use a Runge–Kutta scheme to compute the predicted solution used to compute all the integrals in the DG scheme. In this section, we list down the CERK scheme for the following ODE:

$$\frac{\mathrm{d}u}{\mathrm{d}t} = f(u, t).$$

Given the solution u^n at time t_n, the CERK scheme gives a polynomial solution in the time interval $[t_n, t_{n+1})$ of the form

$$u(t_n + \theta h) = u^n + h \sum_{s=1}^{n_s} b_s(\theta) k_s, \qquad \theta \in [0, 1],$$

where n_s is the number of stages and h denotes the time step.

Second Order (CERK2)

The number of stages is $n_s = 2$ and

$$b_1(\theta) = \theta - \theta^2/2, \qquad b_2(\theta) = \theta^2/2,$$

and

$$\begin{aligned} k_1 &= f(u^n, t_n), \\ k_2 &= f(u^n + hk_1, t_n + h). \end{aligned}$$

Third Order (CERK3)

The number of stages is $n_s = 4$ and

$$\begin{aligned} k_1 &= f(u^n, t_n), \\ k_2 &= f(u^n + (12/23)hk_1, t_n + 12h/23), \\ k_3 &= f(u^n + h((-68/375)k_1 + (368/375)k_2), t_n + 4h/5), \\ k_4 &= f(u^n + h((31/144)k_1 + (529/1\,152)k_2 + (125/384)k_3), t_n + h), \end{aligned}$$

and

$$\begin{aligned} b_1(\theta) &= (41/72)\theta^3 - (65/48)\theta^2 + \theta, \\ b_2(\theta) &= -(529/576)\theta^3 + (529/384)\theta^2, \\ b_3(\theta) &= -(125/192)\theta^3 + (125/128)\theta^2, \\ b_4(\theta) &= \theta^3 - \theta^2. \end{aligned}$$

References

1. Springel, V.: E pur si muove: Galilean-invariant cosmological hydrodynamical simulations on a moving mesh. Mon. Notices R. Astron. Soc. **401**(2), 791–851 (2010)
2. Johnsen, E., Larsson, J., Bhagatwala, A.V., Cabot, W.H., Moin, P., Olson, B.J., Rawat, P.S., Shankar, S.K., Sjögreen, B., Yee, H.C., Zhong, X., Lele, S.K.: Assessment of high-resolution methods for numerical simulations of compressible turbulence with shock waves. J. Comput. Phys. **229**(4), 1213–1237 (2010)

3. Munz, C.D.: On Godunov-type schemes for Lagrangian gas dynamics. SIAM J. Num. Anal. **31**(1), 17–42 (1994)
4. Carré, G., Del Pino, S., Després, B., Labourasse, E.: A cell-centered Lagrangian hydrodynamics scheme on general unstructured meshes in arbitrary dimension. J. Comput. Phys. **228**(14), 5160–5183 (2009)
5. Maire, P.-H.: A high-order cell-centered Lagrangian scheme for two-dimensional compressible fluid flows on unstructured meshes. J. Comput. Phys. **228**(7), 2391–2425 (2009)
6. Hirt, C., Amsden, A., Cook, J.: An arbitrary Lagrangian–Eulerian computing method for all flow speeds. J. Comput. Phys. **14**(3), 227–253 (1974)
7. Donea, J., Huerta, A., Ponthot, J.-P., Rodríguez-Ferran, A.: Arbitrary Lagrangian–Eulerian Methods. Wiley, Oxford (2004)
8. Walter, B.: High order accurate direct arbitrary-Lagrangian–Eulerian ADER-MOOD finite volume schemes for non-conservative hyperbolic systems with stiff source terms. Commun. Comput. Phys. **21**(01), 271–312 (2017)
9. Gaburro, E., Castro, M.J., Dumbser, M.: Well-balanced arbitrary-Lagrangian–Eulerian finite volume schemes on moving nonconforming meshes for the Euler equations of gas dynamics with gravity. Mon. Notices R. Astron. Soc. **477**(2), 2251–2275 (2018). https://doi.org/10.1093/mnras/sty542. 21 June
10. Lomtev, I., Kirby, R., Karniadakis, G.: A discontinuous Galerkin ALE method for compressible viscous flows in moving domains. J. Comput. Phys. **155**(1), 128–159 (1999)
11. Venkatasubban, C.S.: A new finite element formulation for ALE (arbitrary Lagrangian Eulerian) compressible fluid mechanics. Int. J. Eng. Sci. **33**(12), 1743–1762 (1995)
12. Wang, L., Persson, P.-O.: High-order discontinuous Galerkin simulations on moving domains using ALE formulations and local remeshing and projections. American Institute of Aeronautics and Astronautics (2015)
13. Weizhang, H., Russell, R.D.: Adaptive Moving Mesh Methods. Applied Mathematical Sciences. Springer, Berlin (2011)
14. He, P., Tang, H.: An adaptive moving mesh method for two-dimensional relativistic hydrodynamics. Commun. Comput. Phys. **11**(1), 114–146 (2012)
15. Luo, H., Baum, J.D., Löhner, R.: On the computation of multi-material flows using ALE formulation. J. Comput. Phys. **194**(1), 304–328 (2004)
16. Boscheri, W., Dumbser, M., Balsara, D.S.: High-order ADER-WENO ALE schemes on unstructured triangular meshes–application of several node solvers to hydrodynamics and magnetohydrodynamics. Int. J. Numer. Meth. Fluids **76**, 737–778 (2014). https://doi.org/10.1002/fld.3947
17. Luo, H., Baum, J.D., Löhner, R.: On the computation of multi-material flows using ALE formulation. J. Comput. Phys. **194**, 304–328 (2004). https://doi.org/10.1016/j.jcp.2003.09.026
18. Liu, W., Cheng, J., Shu, C.-W.: High order conservative Lagrangian schemes with Lax–Wendroff type time discretization for the compressible Euler equations. J. Comput. Phys. **228**(23), 8872–8891 (2009)
19. Boscheri, W., Dumbser, M.: Arbitrary-Lagrangian–Eulerian one-step WENO finite volume schemes on unstructured triangular meshes. Commun. Comput. Phys. **14**(5), 1174–1206 (2013)
20. Boscheri, W., Dumbser, M.: High order accurate direct arbitrary-Lagrangian–Eulerian ADER-WENO finite volume schemes on moving curvilinear unstructured meshes. Comput. Fluids **136**, 48–66 (2016)
21. Boscheri, W., Dumbser, M., Balsara, D.S.: High-order ADER-WENO ALE schemes on unstructured triangular meshes–application of several node solvers to hydrodynamics and magnetohydrodynamics. Int. J. Num. Methods Fluids **76**, 737–778 (2014)
22. Klingenberg, C., Schnücke, G., Xia, Y.: Arbitrary Lagrangian–Eulerian discontinuous Galerkin method for conservation laws: analysis and application in one dimension. Math. Comput. **86**, 1203–1232 (2017)
23. Harten, A., Hyman, J.M.: Self adjusting grid methods for one-dimensional hyperbolic conservation laws. J. Comput. Phys. **50**, 253–269 (1983)
24. Gassner, G., Dumbser, M., Hindenlang, F., Munz, C.-D.: Explicit one-step time discretizations for discontinuous Galerkin and finite volume schemes based on local predictors. J. Comput. Phys. **230**(11), 4232–4247 (2011)
25. Owren, B., Zennaro, M.: Derivation of efficient, continuous, explicit Runge–Kutta methods. SIAM J. Sci. Stat. Comput. **13**(6), 1488–1501 (1992)
26. Cockburn, B., Lin, S.-Y., Shu, C.-W.: TVB Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws III: One-dimensional systems. J. Comput. Phys. **84**, 90–113 (1989)
27. Cockburn, B., Shu, C.-W.: TVB Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws II: General framework. Math. Comput. **52**(186), 411–435 (1989)

28. Zhang, X., Shu, C.-W.: On positivity-preserving high order discontinuous Galerkin schemes for compressible Euler equations on rectangular meshes. J. Comput. Phys. **229**(23), 8918–8934 (2010)
29. Zhong, X., Shu, C.-W.: A simple weighted essentially nonoscillatory limiter for Runge–Kutta discontinuous Galerkin methods. J. Comput. Phys. **232**(1), 397–415 (2013)
30. Sod, G.: A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws. J. Comput. Phys. **27**, 1–31 (1978)
31. Shu, C.-W., Osher, S.: Efficient implementation of essentially non-oscillatory shock-capturing schemes. J. Comput. Phys. **77**(2), 439–471 (1988)
32. Titarev, V.A., Toro, E.F.: Finite volume WENO schemes for three-dimensional conservation laws. J. Comput. Phys. **201**, 238–260 (2014)
33. Toro, E.F.: Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction. Springer, Berlin (1999)
34. Woodward, P., Colella, P.: The numerical simulation of two-dimensional fluid flow with strong shocks. J. Comput. Phys. **54**(1), 115–173 (1984)
35. Zhu, J., Qiu, J.: A new fifth order finite difference WENO scheme for solving hyperbolic conservation laws. J. Comput. Phys. **318**, 110–121 (2016)
36. Loubere, R., Shashkov, M.J.: A subcell remapping method on staggered polygonal grids for arbitrary-Lagrangian–Eulerian methods. J. Comput. Phys. **209**(1), 105–138 (2005)
37. Olivier, G., Alauzet, F.: A new changing-topology ALE scheme for moving mesh unsteady simulations. In: 49th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition, p. 474 (2011)
38. Frederic, Alauzet: A changing-topology moving mesh technique for large displacements. Eng. Comput. **30**(2), 175–200 (2014)
39. Roe, P.L.: Approximate Riemann solvers, parameter vectors, and difference schemes. J. Comput. Phys. **43**(2), 357–372 (1981)

ALE-DG Methods for the 1-D Euler equations

Arbitrary Lagrangian-Eulerian discontinuous Galerkin method for 1-D Euler equations

Praveen Chandrashekar and Jayesh Badwaik

Abstract We propose an explicit in time discontinuous Galerkin scheme on moving grids using the arbitrary Lagrangian-Eulerian approach for one dimensional Euler equations. The grid is moved with a velocity that is close to the local fluid velocity, which considerably reduces the numerical dissipation in the Riemann solvers. Local grid refinement and coarsening are performed to maintain the mesh quality and avoid very small or large cells. Second, third and fourth order methods are developed and several test cases are provided to demonstrate the accuracy of the proposed scheme.

1 Introduction

Finite volume schemes based on exact or approximate Riemann solvers are able to compute discontinuous solutions in a stable manner since they have implicit dissipation built into them due to the upwind nature of the schemes. Higher order schemes are constructed following a reconstruction approach combined with a high order time integration scheme. While formally high order methods can converge at high rates for smooth solutions, they can still introduce too much numerical dissipation on coarse meshes. Springel [5] gives the example of a Kelvin-Helmholtz instability in which adding a large constant velocity to both states leads to suppression of the instability due to excessive numerical dissipation. This behaviour is attributed to the fact that fixed grid methods based on upwind schemes are not Galilean invariant. Upwind schemes, even when they are formally high order accurate, are found to be

Praveen Chandrashekar
TIFR Center for Applicable Mathematics, Bangalore, India. e-mail: praveen@math.tifrbng.res.in

Jayesh Badwaik
Department of Mathematics, University of Würzburg, Würzburg, Germany. e-mail: jayesh.badwaik@mathematik.uni-wuerzburg.de

too dissipative when applied to turbulent flows [2] since the numerical viscosity can overwhelm the physical viscosity.

The inherent numerical dissipation in upwind schemes can be reduced if the grid moves along with the flow as in Lagrangian methods or arbitrary Lagrangian-Eulerian approach [1], where the mesh velocity can be chosen to be close to the local fluid velocity but may be regularized to maintain the mesh quality. In [5], the mesh is regenerated after every time step based on a Delaunay triangulation, which allows it to maintain good mesh quality even when the fluid undergoes large shear deformation. However these methods have been restricted to second order accuracy.

Traditionally, ALE methods have been used for problems involving moving boundaries as in wing flutter, store separation and other problems involving fluid structure interaction. Another class of methods solve the PDE on moving meshes where the mesh motion is determined based on a monitor function which is designed to detect regions of large gradients in the solution, see [6] and the references therein. These methods achieve automatic clustering of grid points in regions of large gradients. ALE schemes have been used to compute multi-material flows as in [3], since they are useful to accurately track the material interface.

In the present work, we consider the one dimensional problem and propose an explicit discontinuous Galerkin scheme that is conservative on moving meshes and automatically satisfies the geometric conservation law. The scheme is a single step method which is achieved by using a predictor. Numerical results show the dramatic improvement in resolving discontinuities, especially contact waves. Apart from the geometric complexity, the proposed scheme can be extended to multiple dimensions.

2 Euler equations

The Euler equations are a hyperbolic system of conservation laws for mass, momentum and energy, and can be written as

$$\frac{\partial \boldsymbol{u}}{\partial t} + \frac{\partial \boldsymbol{f}(\boldsymbol{u})}{\partial x} = 0 \tag{1}$$

where $\boldsymbol{u}$ is called the vector of *conserved variables* and $\boldsymbol{f}(\boldsymbol{u})$ are the corresponding fluxes given by

$$\boldsymbol{u} = \begin{bmatrix} \rho \\ \rho v \\ E \end{bmatrix}, \qquad \boldsymbol{f}(\boldsymbol{u}) = \begin{bmatrix} \rho v \\ p + \rho v^2 \\ \rho H v \end{bmatrix}$$

In the above expressions, ρ is the density, v is the velocity, p is the pressure and E is the total energy per unit volume, which for an ideal gas is given by $E = p/(\gamma - 1) + \rho v^2/2$, with $\gamma > 1$ being the ratio of specific heats at constant pressure and volume, and $H = (E + p)/\rho$ is the enthalpy.

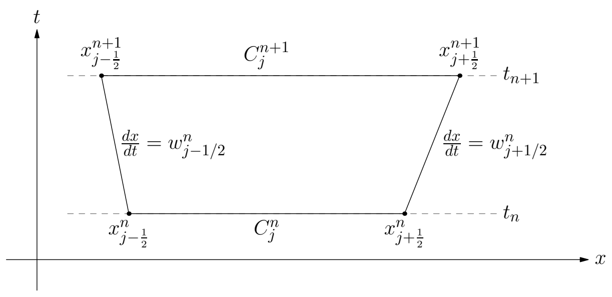

t
$x^{n+1}_{j-\frac{1}{2}}$
C^{n+1}_j
$x^{n+1}_{j+\frac{1}{2}}$
t_{n+1}
$\frac{dx}{dt} = w^n_{j-1/2}$
$\frac{dx}{dt} = w^n_{j+1/2}$
t_n
$x^n_{j-\frac{1}{2}}$
C^n_j
$x^n_{j+\frac{1}{2}}$
x

Fig. 1 Space-time cell

3 Mesh and solution space

Consider a partition of the domain into disjoint cells with the j'th cell being denoted by $C_j(t) = (x_{j-\frac{1}{2}}(t), x_{j+\frac{1}{2}}(t))$. As the notation shows, the cell boundaries are time dependent which means that the cell is moving in some specified manner. The time levels are denoted by t_n with the time step $\Delta t_n = t_{n+1} - t_n$. The boundaries of the cells move with a constant velocity in the time interval (t_n, t_{n+1}) given by

$$w_{j+\frac{1}{2}}(t) = w^n_{j+\frac{1}{2}}, \qquad t_n < t < t_{n+1}$$

which defines a cell in space-time as shown in figure (1). The algorithm to choose the mesh velocity $w^n_{j+\frac{1}{2}}$ is explained in a later section. Let $w(x,t)$ be the continuous linear interpolation of the mesh velocity. We approximate the solution of the conservation law by piecewise polynomials which are allowed to be discontinuous across the cell boundaries. For a given degree $k \geq 0$, the solution in the j'th cell is given by

$$\boldsymbol{u}_h(x,t) = \sum_{m=0}^{k} \boldsymbol{u}_{j,m}(t)\varphi_m(x,t), \qquad x \in C_j(t)$$

where $\{\boldsymbol{u}_{j,m} \in \mathbb{R}^3, 0 \leq m \leq k\}$ are the *degrees of freedom* associated with the j'th cell. The basis functions φ_m are defined in terms of Legendre polynomials by mapping to a reference cell.

4 Discontinuous Galerkin method

Define the *ALE flux* as $\boldsymbol{g}(\boldsymbol{u}, w) = \boldsymbol{f}(\boldsymbol{u}) - w\boldsymbol{u}$. The weak formulation after performing an integration by parts in the x variable, leads to

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\int_{x_{j-\frac{1}{2}}(t)}^{x_{j+\frac{1}{2}}(t)} \boldsymbol{u}_h(x,t)\varphi_l(x,t)\mathrm{d}x = &\int_{x_{j-\frac{1}{2}}(t)}^{x_{j+\frac{1}{2}}(t)} \boldsymbol{g}(\boldsymbol{u}_h,w)\frac{\partial}{\partial x}\varphi_l(x,t)\mathrm{d}x \\ &+\hat{\boldsymbol{g}}_{j-\frac{1}{2}}(\boldsymbol{u}_h(t))\varphi_l(x_{j-\frac{1}{2}}^+,t) - \hat{\boldsymbol{g}}_{j+\frac{1}{2}}(\boldsymbol{u}_h(t))\varphi_l(x_{j+\frac{1}{2}}^-,t)\end{aligned}$$

where we have introduced the numerical flux

$$\hat{\boldsymbol{g}}_{j+\frac{1}{2}}(\boldsymbol{u}_h(t)) = \hat{\boldsymbol{g}}(\boldsymbol{u}_{j+\frac{1}{2}}^-(t), \boldsymbol{u}_{j+\frac{1}{2}}^+(t), w_{j+\frac{1}{2}}(t))$$

which provides an approximation to the ALE flux. The above scheme has an implicit nature since the unknown solution $\boldsymbol{u}_h$ appears on the right hand side integrals whereas we only know the solution at time t_n. In order to obtain an explicit scheme, we assume that we have available with us a *predicted solution* $\boldsymbol{U}_h$ in the time interval (t_n, t_{n+1}), which is used in the time integrals to obtain an explicit scheme. Moreover, the integrals are computed using quadrature in space and time leading to the fully discrete scheme

$$\begin{aligned}h_j^{n+1}\boldsymbol{u}_{j,l}^{n+1} &= h_j^n\boldsymbol{u}_{j,l}^n + \Delta t_n \sum_r \theta_r h_j(\tau_r)\sum_q \eta_q \boldsymbol{g}(\boldsymbol{U}_h(x_q,\tau_r), w(x_q,\tau_r))\frac{\partial}{\partial x}\varphi_l(x_q,\tau_r) \\ &+\Delta t_n \sum_r \theta_r[\hat{\boldsymbol{g}}_{j-\frac{1}{2}}(\boldsymbol{U}_h(\tau_r))\varphi_l(x_{j-\frac{1}{2}}^+,\tau_r) - \hat{\boldsymbol{g}}_{j+\frac{1}{2}}(\boldsymbol{U}_h(\tau_r))\varphi_l(x_{j+\frac{1}{2}}^-,\tau_r)]\end{aligned}$$

where θ_r are weights for time quadrature and η_q are weights for spatial quadrature. In practice, the integrals are computed by mapping the cell to the reference cell, and the basis functions and its derivatives are also evaluated on the reference cell.

4.1 Mesh velocity

The mesh velocity must be close to the local fluid velocity in order to have a Lagrangian character to the scheme. Since the solution is discontinuous, there is no unique fluid velocity at the mesh boundaries. Some researchers, especially in the context of Lagrangian methods, solve a Riemann problem at the cell face to determine the face velocity. Since we use an ALE formulation, we do not require the exact velocity and in our work we make a simple choice which is to take an average of the two velocities at every cell face

$$\tilde{w}_{j+\frac{1}{2}}^n = \frac{1}{2}[v(x_{j+\frac{1}{2}}^-,t_n) + v(x_{j+\frac{1}{2}}^+,t_n)]$$

We will also perform some smoothing of the mesh velocity, e.g., the actual face velocity is computed from

$$w_{j+\frac{1}{2}}^n = \frac{1}{3}(\tilde{w}_{j-\frac{1}{2}}^n + \tilde{w}_{j+\frac{1}{2}}^n + \tilde{w}_{j+\frac{3}{2}}^n)$$

Note that our algorithm to choose the mesh velocity is very local and hence easy and efficient to implement as it does not require the solution of any global problems.

5 Computing the predictor

The predicted solution is used to approximate the flux integrals over the time interval (t_n, t_{n+1}) and the method to compute this must be local, i.e., it must not require solution from neighbouring cells. For a second order scheme, a Taylor expansion retaining only linear terms in t and x is sufficient. For higher order schemes, we adopt the approach of continuous explicit Runge-Kutta (CERK) schemes [4] to approximate the predictor. Let us choose a set of $(k+1)$ distinct nodes, e.g., Gauss-Legendre or Gauss-Lobatto nodes, which uniquely define the polynomial of degree k. These nodes are moving with velocity $w(x,t)$, so that the time evolution of the solution at node x_m is governed by

$$\frac{\mathrm{d}\boldsymbol{U}_m}{\mathrm{d}t} = -[A(\boldsymbol{U}_m(t)) - w_m(t)I]\frac{\partial}{\partial x}\boldsymbol{U}_h(x_m,t) =: \boldsymbol{K}_m(t)$$

with initial condition $\boldsymbol{U}_m(t_n) = \boldsymbol{u}_h(x_m,t_n)$. Using a Runge-Kutta scheme of sufficient order, we will approximate the solution at these nodes as

$$\boldsymbol{U}_m(t) = \boldsymbol{u}_h(x_m,t_n) + \sum_{s=1}^{n_s} b_s((t-t_n)/\Delta t_n)\boldsymbol{K}_{m,s}, \quad t \in [t_n, t_{n+1}), \quad m = 0,1,\ldots,k$$

where $\boldsymbol{K}_{m,s} = \boldsymbol{K}_m(t_n + \theta_s \Delta t_n)$, $\theta_s \Delta t_n$ is the stage time and b_s are certain polynomials related to the CERK scheme.

6 Positivity property

The solutions of Euler equations are well defined only if the density and pressure are positive quantities. This is not a priori guaranteed by the DG scheme even when the TVD limiter is applied. In the case of Runge-Kutta DG schemes, a positivity limiter has been developed in [7] which preserves accuracy in smooth regions. This scheme is built on a positive first order finite volume scheme. For the first order ALE-DG scheme using Rusanov flux, we can show positivity property provided the time step satisfies

$$\Delta t_n \le \Delta t_n^{(1)} := \min_j \left\{ \frac{(1-\frac{1}{2}\beta)h_j^n}{\frac{1}{2}(\lambda_{j-\frac{1}{2}}^n + \lambda_{j+\frac{1}{2}}^n)}, \frac{\beta h_j^n}{|w_{j+\frac{1}{2}}^n - w_{j-\frac{1}{2}}^n|} \right\} \tag{2}$$

Here $\beta \in (0,1)$ is the maximum allowed change in cell size during one time step relative to the previous size.

Theorem 1. *The scheme first order ALE-DG scheme with Rusanov flux is positivity preserving if the time step condition (2) is satisfied.*

Remark 1. In the computations, we use the positivity preserving limiter of [7] which leads to robust schemes which preserve the positivity of the cell average value in all the test cases.

Remark 2. An important property of schemes on moving meshes is their ability to preserve constant states for any mesh motion. This is related to the conservation of cell volumes in relation to the mesh motion. In our scheme, if we start with a constant state $\boldsymbol{u}_h^n = \boldsymbol{c}$, then we can prove that the solution remains constant.

7 Grid coarsening and refinement

The size of the cells can change considerably during the time evolution process due to the near Lagrangian movement of the cell boundaries. Near shocks, the cells will be compressed to smaller sizes which will reduce the allowable time step since a CFL condition has to be satisfied. In some regions, e.g., inside expansion fans, the cell size can increase considerably which may lead to loss of accuracy. In order to avoid too small or too large cells from occuring in the grid, we implement cell merging and refinement. If a cell becomes smaller than some specified size h_{min}, then it is merged with one of its neighbouring cells and the solution is transfered from the two cells to the new cell by performing an L^2 projection. If a cell size becomes larger than some specified size h_{max}, then this cell is refined into two cells by division and the solution is again transfered by L^2 projection.

8 Numerical results

The numerical tests are performed with polynomials of degree one, two and three, together with the linear Taylor expansion, two stage CERK and four stage CERK, respectively, for the predictor. For the quadrature in time, we use the mid-point rule, two and three point Gauss-Legendre quadrature, respectively. High order schemes for hyperbolic equations suffer from spurious numerical oscillations when discontinuities or large gradients are present in the solution which cannot be accurately resolved on the mesh. To control these oscillations, we use standard TVD and TVB limiters applied to characteristic variable and account for non-uniform meshes. The time step is chosen based on equation (2),

$$\Delta t_n = \frac{\text{CFL}}{2k+1} \Delta t_n^{(1)}$$

N	$k=1$		$k=2$		$k=3$	
	Error	Rate	Error	Rate	Error	Rate
100	4.370E-02	-	3.498E-03	-	3.883E-04	-
200	6.611E-03	2.725	4.766E-04	2.876	1.620E-05	4.583
400	1.332E-03	2.518	6.415E-05	2.885	9.376E-07	4.347
800	3.151E-04	2.372	8.246E-06	2.910	5.763E-08	4.239
1600	7.846E-05	2.280	1.031E-06	2.932	3.595E-09	4.180

Table 1 Order of accuracy study on static mesh using Rusanov flux

N	$k=1$		$k=2$		$k=3$	
	Error	Rate	Error	Rate	Error	Rate
100	2.331E-02	-	3.979E-03	-	8.633E-04	-
200	6.139E-03	1.925	4.058E-04	3.294	1.185E-05	6.186
400	1.406E-03	2.0258	5.250E-05	3.122	7.079E-07	5.126
800	3.375E-04	2.0366	6.626E-06	3.077	4.340E-08	4.760
1600	8.278E-05	2.0344	8.304E-07	3.057	2.689E-09	4.573

Table 2 Order of accuracy study on moving mesh using Rusanov flux

where the factor $(2k+1)$ comes from linear stability analysis, and in most of the computations we use CFL $= 0.9$. In all the solutions plots given below, symbols denote the cell average value.

8.1 Order of accuracy

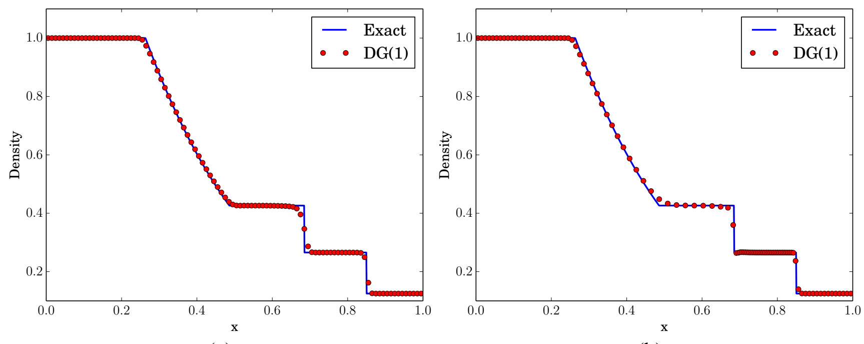
We study the convergence rate of the schemes by applying them to a problem with a known smooth solution. The initial condition is taken as

$$\rho(x,0) = 1 + \exp(-10x^2), \qquad u(x,0) = 1, \qquad p(x,0) = 1$$

whose exact solution is $\rho(x,t) = \rho(x-t,0)$, $u(x,t) = 1$, $p(x,t) = 1$. The initial domain is $[-5,+5]$ and the final time is $t = 1$ units. The L^2 norm of the error in density are shown in table (1) for the static mesh and in table (2) for the moving mesh. In each case, we see that the error behaves as $O(h^{k+1})$ which is the optimal rate for smooth solutions.

8.2 Sod problem

The initial condition for the Sod test case is given by



(a) (b)

Fig. 2 Sod problem using Roe flux, 100 cells and TVD limiter: (a) static mesh (b) moving mesh

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

and the solution is computed upto a final time of $T = 0.2$ where the domain is $[0, 1]$. Since the fluid velocity is zero at the boundary, the computational domain does not change with time for the chosen final time. In figure (2), we show the results obtained using Roe flux with 100 cells and TVD limiter on static and moving mesh. The contact wave is considerably well resolved on the moving mesh as compared to the static mesh.

To study the Galilean invariance or the dependance of the solution on the choice of coordinate frame, we add a boost velocity of $V = 10$ or $V = 100$ to the coordinate frame, while implies the initial fluid velocity if $v(x, 0) = V$ and the other quantities remain as before. The results given in figure (3b) clearly show the independance of the results on the moving mesh. Figure (3a) shows that the accuracy of the static mesh results degrades with increase in velocity of the coordinate frame, particularly the contact discontinuity is highly smeared.

8.3 Shu-Osher problem

The initial condition is given by

$$(\rho, v, p) = \begin{cases} (3.857143, 2.629369, 10.333333) & \text{if } x < -4 \\ (1 + 0.2\sin(5x), 0.0, 1.0) & \text{if } x > -4 \end{cases}$$

which involves a smooth sinusoidal density wave which interacts with a shock. The domain is $[-5, +5]$ and the solution is computed upto a final time of $T = 1.8$. The solutions are shown in figure (4a)-(4b) on static and moving meshes using 200 cells

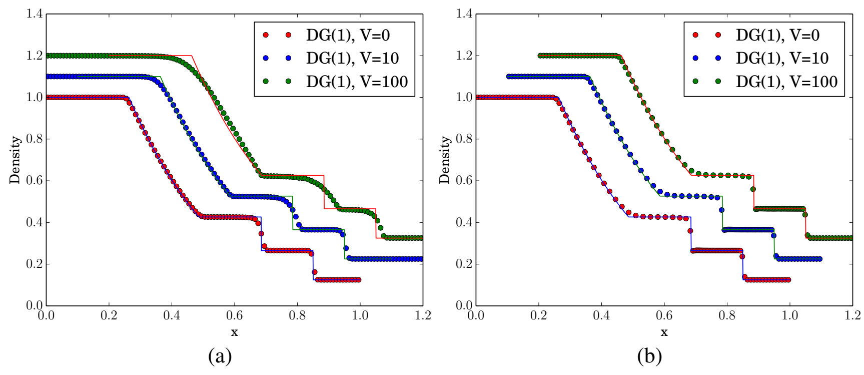


Fig. 3 Effect of coordinate frame motion on Sod problem using Roe flux, 100 cells and TVD limiter: (a) static mesh (b) moving mesh

and TVD limiter. The moving mesh scheme is considerably more accurate in resolving the sinusoidal wave structure that arises after interaction with the shock. In figure (4c) we compute the solution on static mesh with TVB limiter and the parameter $M = 100$ is used. In this case the solutions on static mesh are more accurate compared to the case TVD limiter but still not as good as the moving mesh results. The moving mesh results has more than 200 cells in the interval $[-5, +5]$ at the final time since cells enter the domain from the left side. Hence in figure (4d), we show the static mesh results with 300 cells and using TVB limiter. The results are further improved but still not as accurate as the moving mesh case. The choice of parameters in the TVB limiter is very heuristic and hence it is still advantageous to use the moving mesh scheme which gives improved solutions even with TVD limiter.

8.4 Low density problem

The initial condition is given by

$$(\rho, v, p) = \begin{cases} (1.0, -2.0, 0.4) & x < 0.5 \\ (1.0, +2.0, 0.4) & x > 0.5 \end{cases}$$

The computational domain is $[0, 1]$ and the final time is $T = 0.15$. The density using 100 cells is shown in figure (5) with static and moving meshes. The mesh motion does not significantly improve the solution. On the contrary, the mesh becomes rather coarse in the expansion region, though the solution is still accurate. By enabling grid refinement when $h > 0.05$, we obtain the result shown in figure (6), where better resolution of the low density region is obtained.

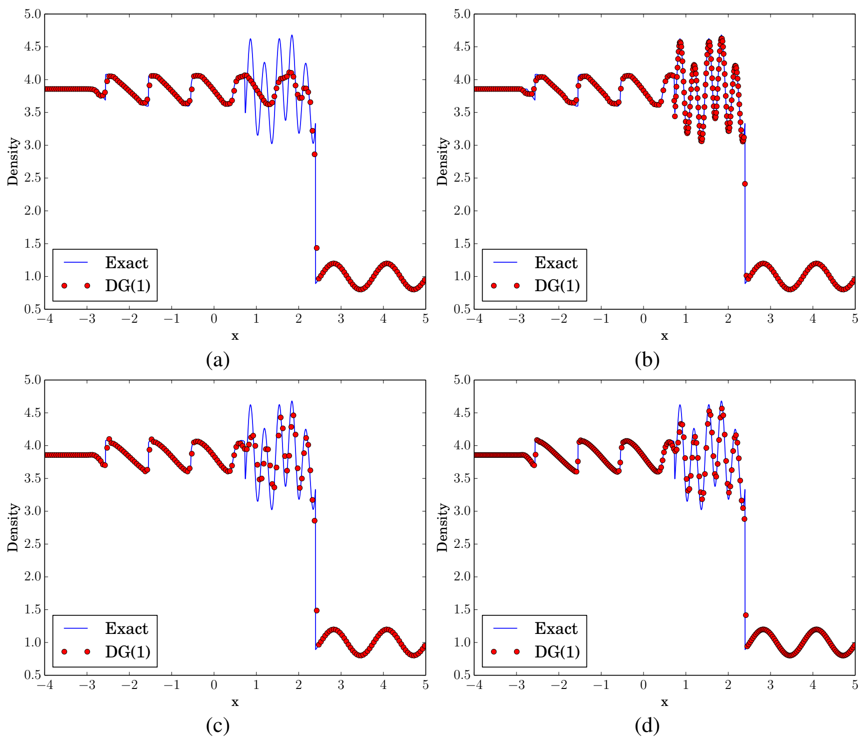


Fig. 4 Shu-Osher problem using Roe flux: (a) static mesh, 200 cells, $M = 0$ (b) moving mesh, 200 cells, $M = 0$ (c) static mesh, 200 cells, $M = 100$ (d) static mesh, 300 cells, $M = 100$

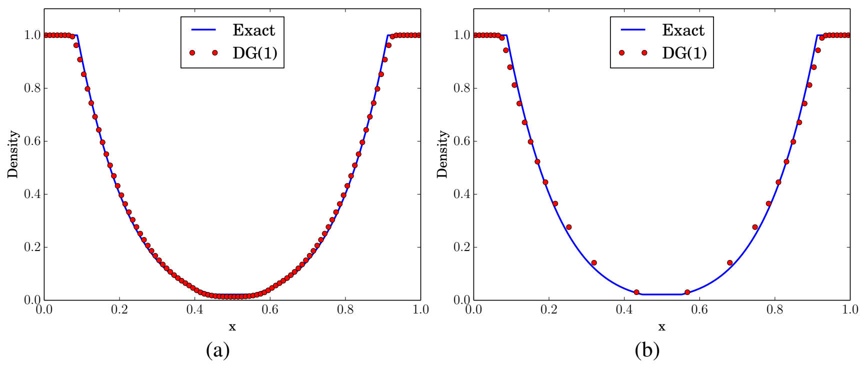


Fig. 5 Low density problem using HLLC flux: (a) static mesh, (b) moving mesh

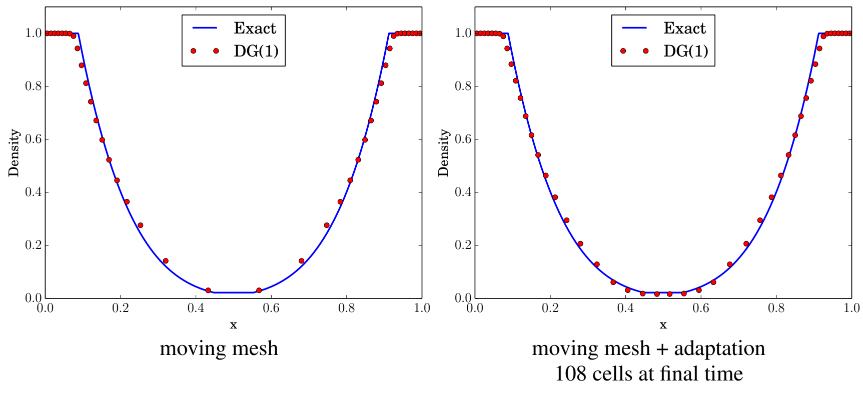


Fig. 6 Low density problem using HLLC flux with 100 cells and moving mesh (a) no adaptation (b) with adaptation.

8.5 *Blast problem*

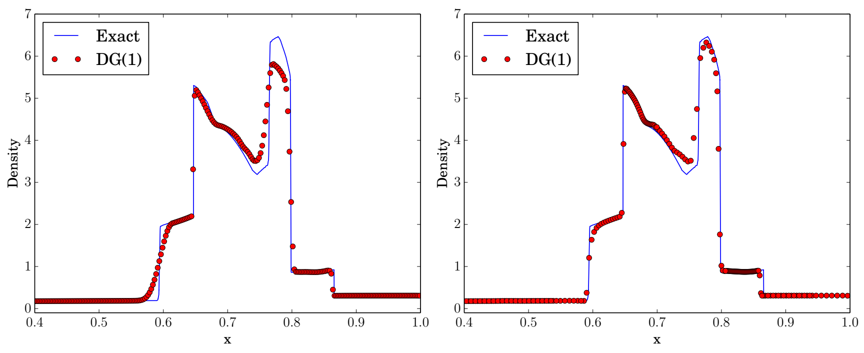
The initial condition is given by

$$(\rho, v, p) = \begin{cases} (1.0, 0.0, 1000.0) & x < 0.1 \\ (1.0, 0.0, 0.01) & 0.1 < x < 0.9 \\ (1.0, 0.0, 100.0) & x > 0.9 \end{cases}$$

and the final time is $T = 0.038$. As shown in figure (7), static mesh results suffer from too much dissipation especially in the contact wave, while the moving mesh is able to resolve this more accurately. Since some cells can become very small in this problem, we have enabled mesh coarsening whenever $h < 0.001$ for any cell.

9 Summary

We have developed an explicit DG scheme on moving meshes using ALE framework and space-time expansion of the solutions within each cell. The near Lagrangian nature of the mesh motion dramatically reduces the numerical dissipation especially for contact waves. The scheme is shown to yield superior results in the presence of large boost velocity of the coordinate system indicating its Galilean invariance property. The mesh motion provides automatic grid adaptation near shocks but may lead to very coarse cells inside expansion waves. A grid adaptation strategy is developed to handle the problem of very small or very large cells. The proposed methodology is general enough to be applicable to other systems of conservation laws modeling fluid flows.



static mesh

moving mesh
303 cells at final time

Fig. 7 Blast problem using HLLC flux with 400 cells on static and moving meshes

Acknowledgements The authors gratefully acknowledge the financial support received from the *Airbus Foundation Chair on Mathematics of Complex Systems* established in TIFR-CAM, Bangalore, for carrying out this work.

References

1. J. Donea, A. Huerta, J.-P. Ponthot, A. Rodríguez-Ferran, Arbitrary Lagrangian–Eulerian Methods, John Wiley & Sons Ltd, 2004.
2. E. Johnsen, J. Larsson, A. V. Bhagatwala, W. H. Cabot, P. Moin, B. J. Olson, P. S. Rawat, S. K. Shankar, B. Sjögreen, H. C. Yee, X. Zhong, S. K. Lele, Assessment of high-resolution methods for numerical simulations of compressible turbulence with shock waves, J. Comput. Phys. 229 (4) (2010) 1213–1237.
3. H. Luo, J. D. Baum, R. Löhner, On the computation of multi-material flows using ALE formulation, Journal of Computational Physics 194 (1) (2004) 304 – 328.
4. B. Owren, M. Zennaro, Derivation of efficient, continuous, explicit runge-kutta methods, SIAM J. Sci. Stat. Comput. 13 (6) (1992) 1488–1501.
5. V. Springel, E pur si muove: Galilean-invariant cosmological hydrodynamical simulations on a moving mesh, Monthly Notices of the Royal Astronomical Society 401 (2) (2010) 791–851.
6. H. Weizhang, R. D. Russell, Adaptive Moving Mesh Methods, Applied Mathematical Sciences, Springer, 2011.
7. X. Zhang, C.-W. Shu, On positivity-preserving high order discontinuous galerkin schemes for compressible euler equations on rectangular meshes, J. Comput. Phys. 229 (23) (2010) 8918–8934.

Task-based Parallelization of an Implicit Kinetic Scheme

ESAIM: PROCEEDINGS AND SURVEYS, June 2018, Vol. 63, p. 60-77
L. Grigori, C. Japhet, P. Moireau, Editors

TASK-BASED PARALLELIZATION OF AN IMPLICIT KINETIC SCHEME [*]

Jayesh Badwaik[1], Matthieu Boileau[2], David Coulette[2], Emmanuel Franck[2], Philippe Helluy[2], Christian Klingenberg[1], Laura Mendoza[3] and Herbert Oberlin[3]

Abstract. In this paper we present and implement the Palindromic Discontinuous Galerkin (PDG) method in dimensions higher than one. The method has already been exposed and tested in [4] in the one-dimensional context. The PDG method is a general implicit high order method for approximating systems of conservation laws. It relies on a kinetic interpretation of the conservation laws containing stiff relaxation terms. The kinetic system is approximated with an asymptotic-preserving high order DG method. We describe the parallel implementation of the method, based on the StarPU runtime library. Then we apply it on preliminary test cases.

1. Introduction

In this work we consider the time discretization of compressible fluid models that appear in gas dynamics, biology, astrophysics or plasma physics for tokamaks. These models can be unified in the following form

$$\partial_t \mathbf{w} + \sum_{k=1}^{D} \partial_k \mathbf{q}^k(\mathbf{w}) = \mathbf{s}, \tag{1}$$

where $\mathbf{w} : \mathbb{R}^D \times [0, T_{\max}] \longrightarrow \mathbb{R}^m$ is the vector of conservative variables, $\mathbf{q}^k(\mathbf{w}) : \mathbb{R}^m \longrightarrow \mathbb{R}^m$ is the flux and $\mathbf{s} : \mathbb{R}^D \times \mathbb{R} \times \mathbb{R}^m \longrightarrow \mathbb{R}^m$ is a source term. D represents the physical space dimension and m the number of unknowns.

In many physical applications such as MHD flows, low Mach Euler equations, Shallow-Water with sedimentation, the model presents several time scales associated to the propagation of different waves. When the time scale of the fast phenomena, which constrains the explicit CFL condition, is very small compared to the time scale of the most relevant phenomena, it becomes necessary to switch to implicit schemes. However standard implicit schemes are very costly in 2D or 3D because they require the resolution of linear or non-linear systems at each time step. In addition, the matrices associated with the hyperbolic systems are generally ill-conditioned.

In this paper, we propose to follow another approach for avoiding the resolution of complicated linear systems. Instead of solving the full fluid model (1) directly, we replace it by a simpler kinetic interpretation made of a set of transport equations coupled through a stiff relaxation term [1, 3, 7]. See also [4] and included references.

[*] *This work has benefited from several supports: ANR/SPPEXA (EXAMAG project), French Public Bank of Investment (BPI HOROCH project), Euratom (Eurofusion project).*

[1] Departement of Mathematics, Würzburg University, Germany
[2] IRMA, University of Strasbourg, Inria TONUS, France
[3] Max-Planck-Institut für Plasmaphysik, Garching, Germany

The kinetic system is then solved by a splitting method where the transport and relaxation stages are treated separately. The method is then well adapted to parallel implementation. The method is already presented in [4] in the one-dimensional case. Here we present its implementation in higher dimensions. We particularly focus our presentation on the massive parallelization of the method with the StarPU runtime system [2].

The paper is organized as following.

First we recall that it is possible to provide a general kinetic interpretation of any system of conservation laws. The interest of this representation is that the complicated non-linear system is replaced by a (larger) set of scalar linear transport equations that are much easier to solve. The transport equations are coupled through a non-linear source term that is fully local in space.

Then, we detail the approximation which allows to solve the transport equation in an efficient way. We adopt a Discontinuous Galerkin (DG) method based on an upwind numerical flux and Gauss-Lobatto quadrature points. Thanks to the upwind flux, the matrix of the discretized transport operator has a block-triangular structure.

The main part of this work is devoted to the task parallelization based on the StarPU library for treating the transport and relaxation steps efficiently. We use the MPI version of StarPU, which allows to address clusters of multicore processors. We also describe the domain decomposition and the macrocell approach that we have used to achieve better performance.

Finally, we present the performance of our parallel implementation on a cluster 4 MPI nodes. Each node contains 24 CPU cores.

2. Kinetic model

We consider the following kinetic equation

$$\partial_t \mathbf{f} + \sum_{k=1}^{D} \mathbf{V}^k \partial_k \mathbf{f} = \frac{1}{\tau}(\mathbf{f}^{eq}(\mathbf{f}) - \mathbf{f}) + \mathbf{g}. \tag{2}$$

The unknown is a vectorial distribution function $\mathbf{f}(\mathbf{x},t) \in \mathbb{R}^{n_v}$ depending on the space variable $\mathbf{x} = (x^1 \dots x^D) \in \mathbb{R}^D$ and time $t \in \mathbb{R}$. $\mathbf{g}(\mathbf{x},t,\mathbf{f})$ is a vectorial source term, possibly depending on space, time and $\mathbf{f}$. The partial derivatives are noted

$$\partial_t = \frac{\partial}{\partial t}, \quad \partial_k = \frac{\partial}{\partial x^k}.$$

The relaxation time τ is a small positive constant. The constant matrices $\mathbf{V}^k$, $1 \le k \le D$ are diagonal

$$\mathbf{V}^k = \begin{pmatrix} v_1^k & & & \\ & v_2^k & & \\ & & \ddots & \\ & & & v_{n_v}^k \end{pmatrix}$$

In other words, (2) is a set of n_v transport equations at constant velocities $\mathbf{v}_i = (v_i^1, \dots, v_i^D)$, coupled through a stiff BGK relaxation, and with an optional additional source term. We denote by $\mathbf{V} \cdot \boldsymbol{\partial} = \sum_{k=1}^{D} \mathbf{V}^k \partial_k$ the transport operator, and by $\mathbf{N}\mathbf{f} = (\mathbf{f}^{eq}(\mathbf{f}) - \mathbf{f})/\tau$ the BGK relaxation term (also called the "collision" term).

Generally, this kinetic model represents an underlying hyperbolic system of conservation laws. The macroscopic conservative variables $\mathbf{w}(\mathbf{x},t) \in \mathbb{R}^m$ are obtained through a linear transformation

$$\mathbf{w} = \mathbf{P}\mathbf{f}, \tag{3}$$

where $\mathbf{P}$ is an $m \times n_v$ matrix. Generally the number of conservative variables is smaller than the number of kinetic data, and thus we have $m < n_v$. The equilibrium (or "Maxwellian") distribution $\mathbf{f}^{eq}(\mathbf{f})$ is such that

$$\mathbf{P}\mathbf{f} = \mathbf{P}\mathbf{f}^{eq}(\mathbf{f}), \tag{4}$$

and

$$\mathbf{w} = \mathbf{P}\mathbf{f}_1 = \mathbf{P}\mathbf{f}_2 \Rightarrow \mathbf{f}^{eq}(\mathbf{f}_1) = \mathbf{f}^{eq}(\mathbf{f}_2), \tag{5}$$

which states that the equilibrium actually depends only on the macroscopic data $\mathbf{w}$. We could have used the notation $\mathbf{f}^{eq} = \mathbf{f}^{eq}(\mathbf{w}) = \mathbf{f}^{eq}(\mathbf{P}\mathbf{f})$, but we have decided to respect a well-established tradition.

When $\tau \to 0$, the kinetic equations provide an approximation of the system of conservation laws

$$\partial_t \mathbf{w} + \sum_{k=1}^{D} \partial_k \mathbf{q}^k(\mathbf{w}) = \mathbf{s}, \tag{6}$$

where the flux is given by

$$\mathbf{q}^k(\mathbf{w}) = \mathbf{P}\mathbf{V}^k \mathbf{f}^{eq}(\mathbf{f}).$$

The flux is indeed a function of $\mathbf{w}$ only because of (5).

Similarly the source term is given by

$$\mathbf{s}(\mathbf{x}, t, \mathbf{w}) = \mathbf{P}\mathbf{g}(\mathbf{x}, t, \mathbf{f}^{eq}) \tag{7}$$

System (2) has to be supplemented with conditions at the boundary $\partial\Omega$ of the computational domain Ω. We denote by $\mathbf{n} = (n_1 \dots n_D)$ the outward normal vector on $\partial\Omega$. For simplicity, we shall only consider very simple imposed and time-independent boundary conditions $\mathbf{f}^b$. We note

$$\mathbf{V} \cdot \mathbf{n} = \sum_{k=1}^{D} \mathbf{V}^k n_k, \quad \mathbf{V} \cdot \mathbf{n}^+ = \max(\mathbf{V} \cdot \mathbf{n}, 0), \quad \mathbf{V} \cdot \mathbf{n}^- = \min(\mathbf{V} \cdot \mathbf{n}, 0).$$

A natural boundary condition, which is compatible with the transport operator $\mathbf{V} \cdot \boldsymbol{\partial}$, is

$$\mathbf{V} \cdot \mathbf{n}^- \mathbf{f}(\mathbf{x}, t) = \mathbf{V} \cdot \mathbf{n}^- \mathbf{f}^b(\mathbf{x}), \quad \mathbf{x} \in \partial\Omega. \tag{8}$$

It states that for a given velocity $\mathbf{v}_i$, the corresponding boundary data f_i^b is used only at the inflow part of the boundary.

Let us point out that the programming optimization that we propose in this paper rely in an essential way on the nature of the boundary condition (8). For other boundary conditions, such as periodic or wall conditions, additional investigations are still needed.

3. Numerical method

3.1. Discontinuous Galerkin approximation

For solving (2) we shall treat the transport operator $\mathbf{V} \cdot \boldsymbol{\partial}$ and the collision operator $\mathbf{N}$ efficiently, thanks to a splitting approach. This allows to achieve a better parallelism. Let us start with the description of the transport solver.

For a simple exposition, we only consider one single scalar transport equation for $f(\mathbf{x}, t) \in \mathbb{R}$ at constant velocity $\mathbf{v}$

$$\partial_t f + \mathbf{v} \cdot \nabla f = 0. \tag{9}$$

The general vectorial case is easily deduced.

We consider a mesh $\mathcal{M}$ of Ω made of open sets, called "cells", $\mathcal{M} = \{L_i,\ i = 1 \dots N_c\}$. In the most general setting, the cells satisfy

(1) $L_i \cap L_j = \emptyset$, if $i \neq j$,
(2) $\overline{\cup_i L_i} = \overline{\Omega}$.

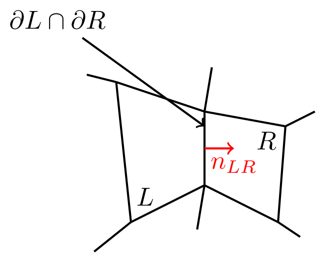


FIGURE 1. Convention for the L and R cells orientation.

In each cell $L \in \mathcal{M}$ we consider a basis of functions $(\varphi_{L,i}(\mathbf{x}))_{i=0\dots N_d-1}$ constructed from polynomials of order d. We denote by h the maximal diameter of the cells. With an abuse of notation we still denote by f the approximation of f, defined by

$$f(\mathbf{x},t) = \sum_{j=0}^{N_d-1} f_{L,j}(t)\varphi_{L,j}(\mathbf{x}), \quad \mathbf{x} \in L.$$

The DG formulation then reads: find the $f_{L,j}$'s such that for all cell L and all test function $\varphi_{L,i}$

$$\int_L \partial_t f \varphi_{L,i} - \int_L f\mathbf{v}\cdot\nabla\varphi_{L,i} + \int_{\partial L}(\mathbf{v}\cdot\mathbf{n}^+ f_L + \mathbf{v}\cdot\mathbf{n}^- f_R)\varphi_{L,i} = 0. \tag{10}$$

In this formula (see Figure 22):

- R denotes the neighboring cell to L along its boundary $\partial L \cap \partial R$, or the exterior of Ω on $\partial L \cap \partial\Omega$.
- $\mathbf{n} = \mathbf{n}_{LR}$ is the unit normal vector on ∂L oriented from L to R.
- f_R denotes the value of f in the neighboring cell R on $\partial L \cap \partial R$.
- If L is a boundary cell, one may have to use the boundary values instead: $f_R = f^b$ on $\partial L \cap \partial\Omega$.
- $\mathbf{v}\cdot\mathbf{n}^+ f_L + \mathbf{v}\cdot\mathbf{n}^- f_R$ is the standard upwind numerical flux encountered in many finite volume or DG methods.

In our application, we consider hexahedral cells. We have a reference cell

$$\hat{L} =]-1,1[^D$$

and a smooth transformation $\mathbf{x} = \boldsymbol{\tau}_L(\hat{\mathbf{x}})$, $\hat{\mathbf{x}} \in \hat{L}$, that maps $\hat{L}$ on L

$$\boldsymbol{\tau}_L(\hat{L}) = L.$$

We assume that $\boldsymbol{\tau}_L$ is invertible and we denote by $\boldsymbol{\tau}'_L$ its (invertible) Jacobian matrix. We also assume that $\boldsymbol{\tau}_L$ is a direct transformation

$$\det \boldsymbol{\tau}'_L > 0.$$

In our implementation $\boldsymbol{\tau}_L$ is a quadratic map based on hexahedral curved "H20" finite elements with 20 nodes. The mesh of H20 finite elements is generated by `gmsh` [6].

On the reference cell, we consider the Gauss-Lobatto points $(\hat{\mathbf{x}}_i)_{i=0\dots N_d-1}$, $N_d = (d+1)^D$ and associated weights $(\omega_i)_{i=0\dots N_d-1}$. They are obtained by tensor products of the $(d+1)$ one-dimensional Gauss-Lobatto (GL) points on $]-1,1[$. The reference GL points and weights are then mapped to the physical GL points of cell L by

$$\mathbf{x}_{L,i} = \boldsymbol{\tau}_L(\hat{\mathbf{x}}_i), \quad \omega_{L,i} = \omega_i \det \boldsymbol{\tau}'_L(\hat{\mathbf{x}}_i) > 0. \tag{11}$$

In addition, the six faces of the reference hexahedral cell are denoted by F_ϵ, $\epsilon = 1\dots6$ and the corresponding outward normal vectors are denoted by $\hat{\mathbf{n}}_\epsilon$. A big advantage of choosing the GL points is that the volume and the faces share the same quadrature points. A special attention is necessary for defining the face quadrature weights. If a GL point $\hat{\mathbf{x}}_i \in F_\epsilon$, we denote by μ_i^ϵ the corresponding quadrature weight on face F_ϵ. We also use the convention that $\mu_i^\epsilon = 0$ if $\hat{\mathbf{x}}_i$ does not belong to face F_ϵ. A given GL point $\hat{\mathbf{x}}_i$ can belong to several faces when it is on an edge or in a corner of $\hat{L}$. Because of symmetry, we observe that if $\mu_i^\epsilon \neq 0$, then the weight μ_i^ϵ does not depend on ϵ.

We then consider basis functions $\hat{\varphi}_i$ on the reference cell: they are the Lagrange polynomials associated to the Gauss-Lobatto point and thus satisfy the interpolation property

$$\hat{\varphi}_i(\hat{\mathbf{x}}_j) = \delta_{ij}.$$

The basis functions on cell L are then defined according to the formula

$$\varphi_{L,i}(\mathbf{x}) = \hat{\varphi}_i(\boldsymbol{\tau}_L^{-1}(\mathbf{x})).$$

In this way, they also satisfy the interpolation property

$$\varphi_{L,i}(\mathbf{x}_{L,j}) = \delta_{ij}. \tag{12}$$

In this paper, we only consider conformal meshes: the GL points on cell L are supposed to match the GL points of cell R on their common face. Dealing with non-matching cells is the object of a forthcoming work.

Let L and R be two neighboring cells. Let $\mathbf{x}_{L,j}$ be a GL point in cell L that is also on the common face between L and R. In the case of conformal meshes, it is possible to define the index j' such that

$$\mathbf{x}_{L,j} = \mathbf{x}_{R,j'}.$$

Applying a numerical integration to (10), using (11) and the interpolation property (12), we finally obtain

$$\partial_t f_{L,i}\omega_{L,i} - \sum_{j=0}^{N_d-1} \mathbf{v}\cdot\nabla\varphi_{L,i}(\mathbf{x}_{L,j}) f_{L,j}\omega_{L,j} + \sum_{\epsilon=1}^{6} \mu_i^\epsilon \left(\mathbf{v}\cdot\mathbf{n}_\epsilon(\mathbf{x}_{L,i})^+ f_{L,i} + \mathbf{v}\cdot\mathbf{n}_\epsilon(\mathbf{x}_{L,i})^- f_{R,i'}\right) = 0. \tag{13}$$

We have to detail how the gradients and normal vectors are computed in the above formula. Let $\mathbf{A}$ be a square matrix. We recall that the cofactor matrix of $\mathbf{A}$ is defined by

$$\operatorname{co}(\mathbf{A}) = \det(\mathbf{A})\left(\mathbf{A}^{-1}\right)^T. \tag{14}$$

The gradient of the basis function is computed from the gradients on the reference cell using (14)

$$\nabla\varphi_{L,i}(\mathbf{x}_{L,j}) = \frac{1}{\det \boldsymbol{\tau}'_L(\hat{\mathbf{x}}_i)} \operatorname{co}(\boldsymbol{\tau}'_L(\hat{\mathbf{x}}_j))\hat{\nabla}\hat{\varphi}_i(\hat{\mathbf{x}}_j).$$

In the same way, the scaled normal vectors $\mathbf{n}_\epsilon$ on the faces are computed by the formula

$$\mathbf{n}_\epsilon(\mathbf{x}_{L,i}) = \operatorname{co}(\boldsymbol{\tau}'_L(\hat{\mathbf{x}}_i))\hat{\mathbf{n}}_\epsilon.$$

We introduce the following notation for the cofactor matrix

$$\mathbf{c}_{L,i} = \operatorname{co}(\boldsymbol{\tau}'_L(\hat{\mathbf{x}}_i)).$$

The DG scheme then reads

$$\partial_t f_{L,i} - \frac{1}{\omega_{L,i}} \sum_{j=0}^{N_d-1} \mathbf{v} \cdot \mathbf{c}_{L,j} \hat{\nabla}\hat{\varphi}_i(\hat{\mathbf{x}}_j) f_{L,j}\omega_j + \frac{1}{\omega_{L,i}} \sum_{\epsilon=1}^{6} \mu_i^\epsilon \left(\mathbf{v} \cdot \mathbf{c}_{L,i}\hat{\mathbf{n}}_\epsilon{}^+ f_{L,i} + \mathbf{v} \cdot \mathbf{c}_{L,i}\hat{\mathbf{n}}_\epsilon{}^- f_{R,i'}\right) = 0. \quad (15)$$

On boundary GL points, the value of $f_{R,i'}$ is given by the boundary condition

$$f_{R,i'} = f^b(\mathbf{x}_{L,i}), \quad \mathbf{x}_{L,i} = \mathbf{x}_{R,i'}.$$

For practical reasons, it is interesting to also consider $f_{R,i'}$ as an artificial unknown in the fictitious cell. The fictitious unknown is then a solution of the differential equation

$$\partial_t f_{R,i'} = 0. \quad (16)$$

In the end, if we put all the unknowns in a large vector $\mathbf{F}(t)$, (15), (16) read as a large system of coupled differential equations

$$\partial_t \mathbf{F} = \mathbf{L}_h \mathbf{F}. \quad (17)$$

In the following, we call $\mathbf{L}_h$ the transport matrix. The transport matrix satisfies the following properties:

- $\mathbf{L}_h\mathbf{F} = 0$ if the components of $\mathbf{F}$ are all the same.
- Let $\mathbf{F}$ be such that the components corresponding to the boundary term vanish. Then $\mathbf{F}^T\mathbf{L}_h\mathbf{F} \leq 0$. This dissipation property is a consequence of the choice of an upwind numerical flux [9].
- In many cases, and with a good numbering of the unknowns in $\mathbf{F}$, $\mathbf{L}_h$ has a block triangular structure, with small blocks corresponding to the local cell unknowns. This aspect is discussed in Subsection 4.1.

As stated above, we actually have to apply a transport solver for each constant velocity $\mathbf{v}_i$.

Let L be a cell of the mesh $\mathcal{M}$ and $\mathbf{x}_i$ a GL point in L. As in the scalar case, we denote by $\mathbf{f}_{L,i}$ the approximation of $\mathbf{f}$ in L at GL point i. In the sequel, with an abuse of notation and according to the context, we may continue to note $\mathbf{F}(t)$ the big vector made of all the vectorial values $\mathbf{f}_{L,j}$ at all the GL points j in all the (real or fictitious) cells L.

We may also continue to denote by $\mathbf{L}_h$ the matrix made of the assembly of all the transport operators for all velocities $\mathbf{v}_i$. With a good numbering of the unknowns it is possible in many cases to suppose that $\mathbf{L}_h$ is block-triangular. More precisely, because in the transport step the equations are uncoupled, we see that $\mathbf{L}_h$ can be made block-diagonal, each diagonal block being itself block-triangular. See Section 4.1.

3.2. **Palindromic time integration**

We can also define an approximation $\mathbf{N}_h$ of the collision operator $\mathbf{N}$. We define by $\mathbf{F}^{eq}(\mathbf{F})$ the big vector made of all the $\mathbf{f}^{eq}(\mathbf{f}_{L,i})$, $L \in \mathcal{M}$, $i = 0 \ldots N_d - 1$.

We set

$$\mathbf{N}_h\mathbf{F} = \frac{1}{\tau}(\mathbf{F}^{eq}(\mathbf{F}) - \mathbf{F}). \quad (18)$$

Similarly we note $\mathbf{G}_h$ the discrete approximation of the kinetic source term $\mathbf{g}$.

The DG approximation of (2) finally reads

$$\partial_t \mathbf{F} = \mathbf{L}_h\mathbf{F} + \mathbf{N}_h\mathbf{F} + \mathbf{G}_h\mathbf{F}.$$

We use the following Crank-Nicolson second order time integrator for the transport equation:

$$\exp(\Delta t \mathbf{L}_h) \simeq T_2(\Delta t) := (\mathbf{I} + \frac{\Delta t}{2}\mathbf{L}_h)(\mathbf{I} - \frac{\Delta t}{2}\mathbf{L}_h)^{-1}. \quad (19)$$

Similarly, for the collision integrator, we use

$$\exp(\Delta t \mathbf{N}_h) \simeq C_2(\Delta t) := (\mathbf{I} + \frac{\Delta t}{2}\mathbf{N}_h)(\mathbf{I} - \frac{\Delta t}{2}\mathbf{N}_h)^{-1}.$$

Because during the collision step, the conservative variables $\mathbf{w} = \mathbf{Pf}$ do not change and the equilibrium function is explicit, the collision integrator is solved with a local and linear implicit scheme. We have the explicit formula:

$$C_2(\Delta t)\mathbf{F} = \frac{(2\tau - \Delta t)\mathbf{F}}{2\tau + \Delta t} + \frac{2\Delta t \mathbf{F}^{eq}(\mathbf{F})}{2\tau + \Delta t}. \tag{20}$$

The source operator is also approximated by a Crank-Nicolson integrator

$$\exp(\Delta t \mathbf{G}_h) \simeq S_2(\Delta t) := (\mathbf{I} + \frac{\Delta t}{2}\mathbf{G}_h)(\mathbf{I} - \frac{\Delta t}{2}\mathbf{G}_h)^{-1},$$

requiring to solve a nonlinear local equation whenever $\mathbf{g}$ depends on $\mathbf{f}$.

If $\tau > 0$, we observe that the operators T_2 and C_2 are *time-symmetric*: if we set $O_2 = T_2$, $O_2 = C_2$, or $O_2 = S_2$, then O_2 satisfies

$$O_2(-\Delta t) = O_2(\Delta t)^{-1}, \quad O_2(0) = Id. \tag{21}$$

This property implies that O_2 is necessarily a second order approximation of the exact integrator [8, 10]. When $\tau = 0$, we also remark that

$$C_2(\Delta t)\mathbf{F} = 2\mathbf{F}^{eq}(\mathbf{F}) - \mathbf{F} \neq \mathbf{F}$$

and then C_2 does not satisfy (21) anymore.

For $\tau > 0$, the Strang formula permits us to construct a five steps second order time-symmetric approximation

$$M_2^s(\Delta t) = T_2\left(\frac{\Delta t}{2}\right) S_2\left(\frac{\Delta t}{2}\right) C(\Delta t) S_2\left(\frac{\Delta t}{2}\right) T_2\left(\frac{\Delta t}{2}\right) = \exp\left(\Delta t\left(\mathbf{L}_h + \mathbf{N}_h + \mathbf{S}_h\right)\right) + O(\Delta t^3),$$

and a three step one

$$M_2(\Delta t) = T_2\left(\frac{\Delta t}{2}\right) C(\Delta t) T_2\left(\frac{\Delta t}{2}\right) = \exp\left(\Delta t\left(\mathbf{L}_h + \mathbf{N}_h\right)\right) + O(\Delta t^3),$$

in the source-less case.

However this formula is no more a second order approximation of (2) when $\tau \to 0$. Indeed, when $\tau = 0$

$$M_2(0)\mathbf{F} = 2\mathbf{F}^{eq}(\mathbf{F}) - \mathbf{F}.$$

As explained in [4] it is better to consider the following method, which remains second order accurate even for infinitely fast relaxation:

$$M_2^{kin}(\Delta t) = T_2\left(\frac{\Delta t}{4}\right) C_2\left(\frac{\Delta t}{2}\right) T_2\left(\frac{\Delta t}{2}\right) C_2\left(\frac{\Delta t}{2}\right) T_2\left(\frac{\Delta t}{4}\right).$$

By palindromic compositions of the second order method M_2^{kin} it is then very easy to achieve any even order of accuracy (see [4]). However, in this paper, we concentrate on the parallel optimization of the method and we shall only present numerical results at second order for the limit system 6. To that end it is sufficient to use the method M_2, as $PM_2(0)$ properly converges towards identity on the macroscopic variable space when $\tau \to 0$.

4. Optimization of the kinetic solver

In this section, we describe the optimizations that can be applied in the implementation of the previous numerical method.

4.1. **Triangular structure of the transport matrix**

Because of the upwind structure of the numerical flux, it appears that the transport matrix is often block-triangular. The size of the diagonal blocks is the number of nodes N_d in a cell. This is very interesting because this allows to apply implicit schemes to (17) at a low cost [11]. Indeed, the block-triangular solver only requires the factorization of the small the diagonal blocks and a simple forward substitution algorithm. We can provide the formal structure of $\mathbf{L}_h$ through the construction of a directed graph $\mathcal{G}$ with a set of vertices $\mathcal{V}$ and a set of edges $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$. The vertices of the graph are associated to the (real or fictitious) cells of $\mathcal{M}$. Consider now two cells L and R with a common face F_{LR}. We denote by $\mathbf{n}_{LR}$ the normal vector on F_{LR} oriented from L to R. If there is at least one GL point $\mathbf{x}$ on F_{LR} such that

$$\mathbf{n}_{LR}(\mathbf{x}) \cdot \mathbf{v} > 0,$$

then the edge from L to R belongs to the graph:

$$(L, R) \in \mathcal{E},$$

see Figure 2.

In (15) we can distinguish between several kinds of terms. We write

$$\partial_t f_L + \Gamma_{L\leftarrow L} f_L + \sum_{(R,L)\in\mathcal{E}} \Gamma_{L\leftarrow R} f_R,$$

with

$$\begin{aligned}\Gamma_{L\leftarrow L} f_L = \quad & -\frac{1}{\omega_{L,i}} \sum_{j=0}^{N_d-1} \mathbf{v} \cdot \mathbf{c}_{L,j} \hat{\nabla} \hat{\varphi}_i(\hat{\mathbf{x}}_j) f_{L,j} \omega_j \\ & + \frac{1}{\omega_{L,i}} \sum_{\epsilon=1}^{6} \mu_i^\epsilon \mathbf{v} \cdot \mathbf{c}_{L,i} \hat{\mathbf{n}}_\epsilon{}^{+} f_{L,i},\end{aligned}$$

and, if $(R, L) \in \mathcal{E}$,

$$\Gamma_{L\leftarrow R} f_R = \frac{1}{\omega_{L,i}} \mu_i^\epsilon \mathbf{v} \cdot \mathbf{c}_{L,i} \hat{\mathbf{n}}_\epsilon{}^{-} f_{R,i'}.$$

We can use the following convention

$$(R, L) \notin \mathcal{E} \Rightarrow \Gamma_{L\leftarrow R} = 0. \tag{22}$$

$\Gamma_{L\leftarrow L}$ contains the terms that couple the values of f inside the cell L. They correspond to diagonal blocks of size $N_d \times N_d$ in the transport matrix $\mathbf{L}_h$. $\Gamma_{L\leftarrow R}$ contains the terms that couple the values inside cell L with the values in the neighboring upwind cell R. If R is a downwind cell relatively to L then $\mu_i^\epsilon \mathbf{v} \cdot C_{L,i} \hat{\mathbf{n}}_\epsilon{}^{-} = 0$ and $\Gamma_{L\leftarrow R} = 0$ is indeed compatible with the above convention (22).

Once the graph $\mathcal{G}$ is constructed, we can analyze it with standard tools. If it contains no cycle, then it is called a Directed Acyclic Graph (DAG). Any DAG admits a topological ordering of its nodes. A topological ordering is a numbering of the cells $i \mapsto L_i$ such that if there is a path from L_i to L_j in $\mathcal{G}$ then $j > i$. In practice, it is useful to remove the fictitious cells from the topological ordering. In our implementation they are put at the end of the list.

Once the new ordering of the graph vertices is constructed, we can construct a numbering of the components of $\mathbf{F}$ by first numbering the unknowns in L_0 then the unknowns in L_1, *etc.* More precisely, we set

$$F_{kN_d+i} = f_{L_k,i}.$$

Then, with this ordering, the matrix $\mathbf{L}_h$ is lower block-triangular with diagonal blocks of size $N_d \times N_d$. It means that we can apply implicit schemes to (17) without inverting large linear systems.

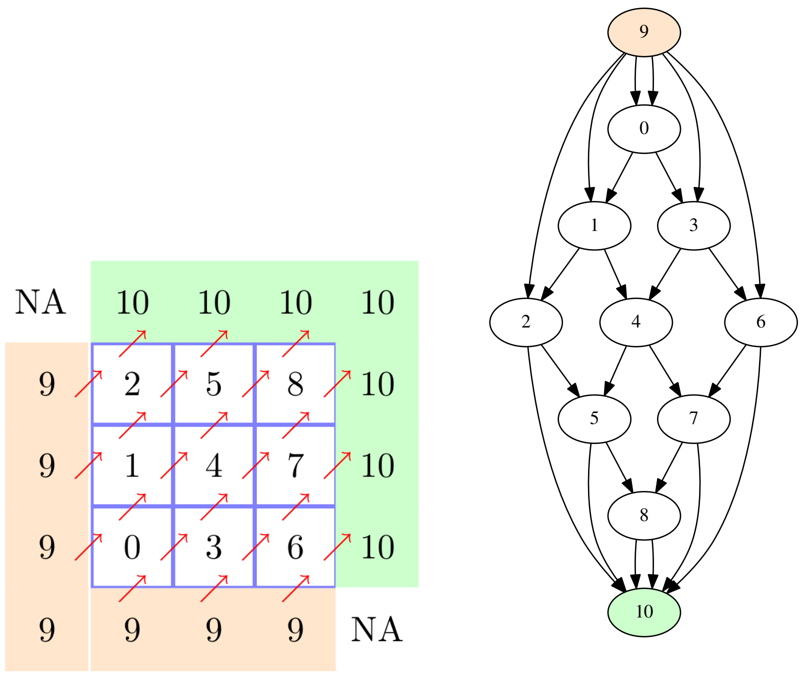


FIGURE 2. Construction of the dependency graph. Left: example of mesh (it is structured here but it is not necessary) with 9 interior cells. The velocity field v is indicated by red arrows. We add two fictitious cells: one for the upwind boundary condition (cell 9) and one for the outflow part of $\partial\Omega$ (cell 10). Right: the corresponding dependency graph $\mathcal{G}$. By examining the dependency graph, we observe that the values of $\mathbf{F}^{n+1}$ in cell 0 have to be computed first, using the boundary conditions. Then cells 1 and 3 can be computed in parallel, then cells 2, 4, and 6 can be computed in parallel, then *etc.*

As stated above, we actually have to apply a transport solver for each constant velocity $\mathbf{v}_i$. In the sequel, with another abuse of notation and according to the context, we continue to note $\mathbf{F}$ the vector of large dimension made of all the vectorial values $\mathbf{f}_{L,j}$ at all the GL points j in all the (real or fictitious) cells L.

We may also continue to denote by $\mathbf{L}_h$ the matrix made of the assembly of all the transport operators for all velocities $\mathbf{v}_i$. With a good numbering of the unknown it is still possible to suppose that $\mathbf{L}_h$ is block-triangular. More precisely, as in the transport step the equations are uncoupled, we see that $\mathbf{L}_h$ can be made a block-diagonal matrix, each diagonal block being itself block-triangular.

4.2. Parallelization of the implicit solver

In this section, we explain how it is possible to parallelize the transport solver. Here again we consider the single transport equation (9) and the associated differential equation (17). We apply a second order Crank-Nicolson implicit scheme. We have explained in Section 3.2 how to increase the order of the scheme. We compute an approximation $\mathbf{F}^n$ of $\mathbf{F}(n\Delta t)$. The implicit scheme reads

$$(\mathbf{I} - \Delta t \mathbf{L}_h)\mathbf{F}^{n+1} = (\mathbf{I} + \Delta t \mathbf{L}_h)\mathbf{F}^n. \tag{23}$$

As explained above, the matrices $(\mathbf{I} - \Delta t \mathbf{L}_h)$ and $(\mathbf{I} + \Delta t \mathbf{L}_h)$ are lower triangular. It is thus possible to solve the linear system explicitly cell after cell, assuming that the cells are numbered in a topological order.

It is possible to perform further optimization by harnessing the parallelism exhibited by the dependency graph. Indeed, once the values of f in the first cell are computed, it is generally possible to compute in parallel the values of f in neighboring downwind cells. For example, as can be seen on Figure 2, once the values in cells 0, 1 and 2 are known, we can compute independently, and in parallel, the values in cells 2, 4 and 6.

We observe that at the beginning and at the end of the time step, the computations are "less parallel" than in the middle of the time step, where the parallelism is maximal.

Implementing this algorithm with OpenMP or using pthread is not very difficult. However, it requires to compute the data dependencies between the computational tasks carefully, and to set adequate synchronization points in order to get correct results. In addition, a rough implementation will probably not exhibit optimized memory access. Therefore, we have decided to rely on a more sophisticated tool called StarPU[1] for submitting the parallel tasks to the available computational resources.

StarPU is a runtime system library developed at Inria Bordeaux [2]. It relies on the data-based parallelism paradigm.

The user has first to split its whole problem into elementary computational tasks. The elementary tasks are then implemented into *codelets*, which are simple C functions. The same task can be implemented differently into several codelets. This allows the user to harness special acceleration devices, such as vectorial CPU cores, GPUs or Intel KNL devices, for example. In the StarPU terminology these devices are called *workers*.

For each task, the user has also to describe precisely what are the input data, in *read* mode, and the output data, in *write* or *read-write* mode. The user then submits the task in a sequential way to the StarPU system. StarPU is able to construct at runtime a task graph from the data dependencies. The task graph is analyzed and the tasks are scheduled automatically to the available workers (CPU cores, GPUs, *etc.*). If possible, they are executed in parallel into concurrent threads. The data transfer tasks between the threads are automatically generated and managed by StarPU, which greatly simplifies the programming.

When a StarPU program is executed, it is possible to choose among several schedulers. The simplest *eager* scheduler adopts a very simple strategy, where the tasks are executed in the order of submission by the free workers, without optimization. More sophisticated schedulers, such as the *dmda* scheduler, are able to measure the efficiency of the different codelets and the data transfer times, in order to apply a more efficient allocation of tasks.

Recently a new data access mode has been added to StarPU: the *commute* mode. In a task, a buffer of data can now be accessed in commute mode, in addition to the write or read-write modes. A commute access tells to StarPU that the execution of the corresponding task may be executed before or after other tasks containing commutative access. This allows StarPU to perform additional optimizations.

There exists also a MPI version of StarPU. In the MPI version, the user has to decide an initial distribution of data among the MPI nodes. Then the tasks are submitted as usual (using the function starpu_mpi_insert_task instead of starpu_insert_task). Required MPI communications are automatically generated by StarPU. For the moment, this approach does not guarantee a good load balancing. It is the responsibility of the user to migrate data from one MPI node to another for improving the load balancing, if necessary.

4.3. **Macrocell approach**

StarPU is quite efficient, but there is an unavoidable overhead due to the task submissions and to the on-the-fly construction and analysis of the task graph. Therefore it is important to ensure that the computational tasks are not too small, in which case the overhead is not amortized, or not too big, in which case some workers are idle.

In order to achieve the right balance, we have decided not to apply directly the above task submission algorithm to the cells but to groups of cells instead.

[1]`http://starpu.gforge.inria.fr`

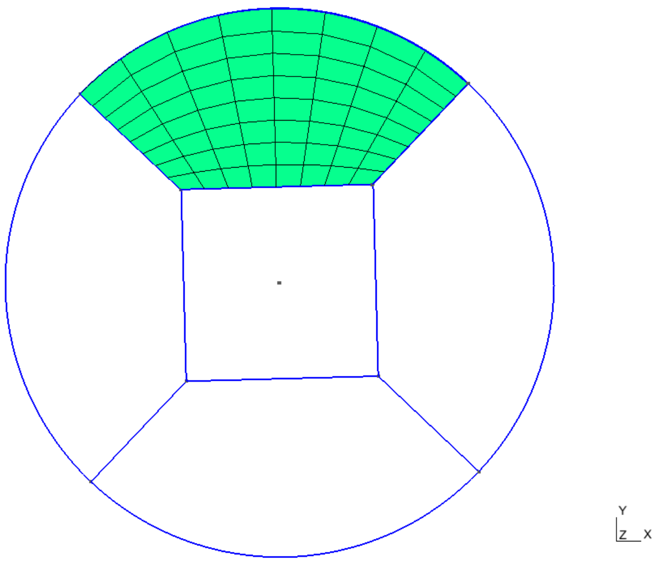


FIGURE 3. Macrocell approach: an example of a mesh made of five macrocells. Each macrocell is then split into several subcells. Only the subcells of the top macrocell are represented here (in green).

The implementation of the whole kinetic scheme is done into the `schnaps` software[2]. `schnaps` is a C99 software dedicated to the numerical simulation of conservation laws.

In `schnaps` we construct first a *macromesh* of the computational domain. Then each *macrocell* of the macromesh is split into subcells. See Figure 3. We also arrange the subcells into a regular sub-mesh of the macrocells. In this way, it is possible to apply additional optimizations. For instance, the subcells L of a same macrocell $\mathcal{L}$ can now share the same geometrical transformation $\boldsymbol{\tau}_L$, which saves memory.

In `schnaps` we define an *interface* structure in order to manage data communications between the macrocells. An interface contains the faces that are common to two neighboring macrocells. We do not proceed exactly as in Section 4.1 where the vertices of graph $\mathcal{G}$ were associated to cells and the edges to faces. Instead, we construct an upwind graph whose vertices are associated to macrocells, and edges to interfaces. This graph is then sorted, and the macrocells are numbered in a topological order.

For solving one time step of one transport equation (23), we split the computations into several elementary operations: for each macrocell $\mathcal{L}$ taken in a topological order, we perform the following tasks:

(1) Volume residual assembly: this task computes in the macrocell $\mathcal{L}$ the part of the right hand side of (23) that comes from the values of f inside $\mathcal{L}$;
(2) Interface residual assembly: this task computes, in the macrocell $\mathcal{L}$, the part of the right hand side of (23) that comes from upwind interface values;
(3) Boundary residual assembly: this task computes, in the macrocell $\mathcal{L}$, the part of the right hand side of (23) that comes from upwind boundaries values.
(4) Volume solve: this task solves the local transport linear system in the macrocell.
(5) Extraction: this task copies the boundary data of $\mathcal{L}$ to the neighbor downwind interfaces.

Let us point out that in step 4 above, the macrocell local transport solver is reassembled and refactorized at each time step: we have decided not to store a sparse linear system in the macrocell for each velocity $\mathbf{v}_i$, in order to save memory. The local sparse linear system is solved thanks to the KLU library [5]. This library

[2]`http://schnaps.gforge.inria.fr/`

is able to detect efficiently sparse triangular matrix structures, which makes the resolution quite efficient. In practice, the factorization and resolution time of the KLU solver is of the same order as the residual assembly time.

In `schnaps`, we use the MPI version of StarPU. The macromesh is initially split into several subdomains and the subdomains are distributed to the MPI nodes. Then the above tasks are launched asynchronously with the starpu_mpi_insert_task function. MPI communications are managed automatically by StarPU.

It is clear that if we were solving a single transport equation our strategy would be very inefficient. Indeed, the downwind subdomains would have to wait for the end of the computations of the upwind subdomains. We are saved by the fact that we have to solve many transport equations in different directions. This helps the MPI nodes to be equally occupied. Our approach is more efficient if we avoid a domain decomposition with internal subdomains, because these subdomains have to wait the results coming from the boundaries.

In our approach it is also essential to launch the tasks in a completely asynchronous fashion. In this way, if a MPI node is waiting for results of other subdomains for a given velocity $\mathbf{v}_i$ it is not prevented from starting the computation for another velocity $\mathbf{v}_j$.

4.4. **Collisions**

In this section we explain how is computed the collision step (20). The computations are purely local to each GL point, which makes the collision step embarrassingly parallel. However it is not so obvious to attain high efficiency because of memory access. If the values of $\mathbf{F}$ are well arranged in memory in the transport stage, it means that the values of $\mathbf{f}$ attached to a given velocity $\mathbf{v}_i$ are close in memory, for a better data locality. On the contrary, in the collision step at a given GL point, a better locality is achieved if the values of $\mathbf{f}$ corresponding to different velocities are close in memory. Additional investigations and tests are needed in order to evaluate the importance of data locality in our algorithm.

In our implementation, we adopt the following strategy. We first identify the following task:

(1) Reduction task for a velocity $\mathbf{v}_i$: this task is associated with one macrocell $\mathcal{L}$. It computes the contribution to $\mathbf{w}$ of the components of $\mathbf{f}$ that have been transported at velocity $\mathbf{v}_i$ with formula (3). The StarPU access to the buffer containing $\mathbf{w}$ is performed in read-write and commute modes. In this way the contribution from each velocity can be added to $\mathbf{w}$ as soon as it is available.
(2) Relaxation task for a velocity $\mathbf{v}_i$: this task is associated to one macrocell $\mathcal{L}$. Once $\mathbf{w}$ is known, it computes the components of $\mathbf{f}^{eq}$ corresponding to velocity $\mathbf{v}_i$. Then it computes the relaxation step (20) for the associated component of $\mathbf{f}$.

In step 2 we can separate the computations for each velocity because the collision term (18) is diagonal. Some Lattice Boltzmann Methods rely on non-diagonal relaxations. It can be useful for representing more general viscous terms for instance. For non-diagonal relaxation we would have to change a little the algorithm.

We can now make a few comments about the storage cost of the method. In the end, we have to store at each GL point $\mathbf{x}_i$ and each cell L the values of $\mathbf{f}_{L,i}$ and $\mathbf{w}_{L,i}$. We do not have to keep the values of the previous time-step, $\mathbf{f}_{L,i}$ and $\mathbf{w}_{L,i}$ can be replaced by the new values as soon as they are available. In this sense, our method is "low-storage". As explained in [4] it is also possible to increase the time order of the method, without increasing the storage.

We can also comment the complexity of a whole implicit step. Because relaxation is local to each node, its complexity is fully linear: the number of floating point operations is proportional to the number of DOF, which is $N_v \times N_d \times N_c$. The most expensive part of the algorithm is the transport step, whose complexity is $N_v \times N_d^3 \times N_c$. The N_d^3 comes from the inversion of the diagonal blocks in the block-triangular linear system. For instance, for a 3D computation of order $d = 3$, we have $N_d = 64$. For the moment, we do not store the factorization of the diagonal blocks. If we stored the factorizations, the complexity would become $N_v \times N_d^2 \times N_c$ at the cost of more memory access. We do not know for the moment which solution is the best.

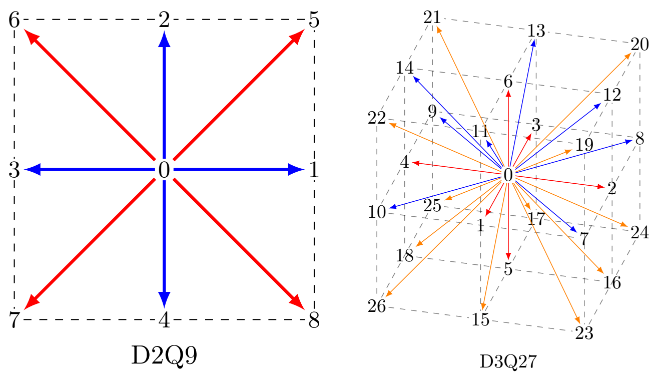


Figure 4. D2Q9 and D3Q27 velocity grids.

4.5. **Scaling test**

For all performance tests presented in this section, we used standard models from the family Lattice-Boltzmann-Method (LBM) kinetic models devised for the simulation of Euler/Navier Stokes systems. We will not give a detailed description of their properties from the modeling point of view: we simply take them as good representative of the typical workload of kinetic relaxation schemes. The most relevant feature impacting the performance of our algorithm is the discrete velocity set of the kinetic model, which determines the task graph structure of the transport step when combined with a particular mesh topology. In standard LBM models, velocity sets are usually built-up from a sequence of pairs of opposite velocities with an additional zero velocity node. On Figure 4 we show the two representative velocity sets of the $D2Q9$ and $D3Q27$ LBM models.

4.5.1. *Multithread performance (D2Q9, D3Q*)*

We first test the multithread performance of our implementation for the full (transport + relaxation) scheme for the standard D2Q9 model. All tests are performed on a single node of the IRMA-ATLAS cluster, with 24 available cores. We consider several square meshes build-up from 1 to 64 macrocells. The number of geometric degrees of freedom per element is kept constant with a value of 3375 points per macrocell, so that the workload per macrocell does not change. For each mesh, we allow StarPu to use from 1 to the full 24 cores of the node and measure the total wall time. The results for this first batch of performance measurements are given in Fig. 5. First we verify that for 1 unique macrocell, parallel performance saturates when the number of cores roughly equals the number of velocities in the model. This is to be expected, as no topological parallelism can be exploited in that case. Increasing the number of macro-element allows to take advantage of topological parallelism. For that workload, parallel efficiency saturates at about 80, which is quite good. On Fig. 6, we consider on the same cubic mesh three different models differing by the number of velocity values. Those cases exhibit a large amount of potential parallelism, due to the large number of velocities combined with the macrocell decomposition. On an ideal machine, they could in theory scale perfectly up to 24 cores. The observed saturation, still around 80 efficiency, is still quite good and comes from the unavoidable concurrency in memory access between the various cores and the scheduling overhead. It is important to note, when considering those results, that the bulk of the computational cost occurs in the transport step of the algorithm. The collision step forces synchronization between all the fields corresponding to individual velocities for the computation of the macroscopic fields. However its actual cost is negligible with respect to the transport step.

4.5.2. *MPI scaling: D3Q15 in a torus*

Having verified the good multithread performance of our code on a single node, we now check whether for larger problem sizes the workload can be distributed among several computing nodes. To that end, accounting for the fact that we aim notably at performing simulations for Tokamak physics, we considered a toroidal

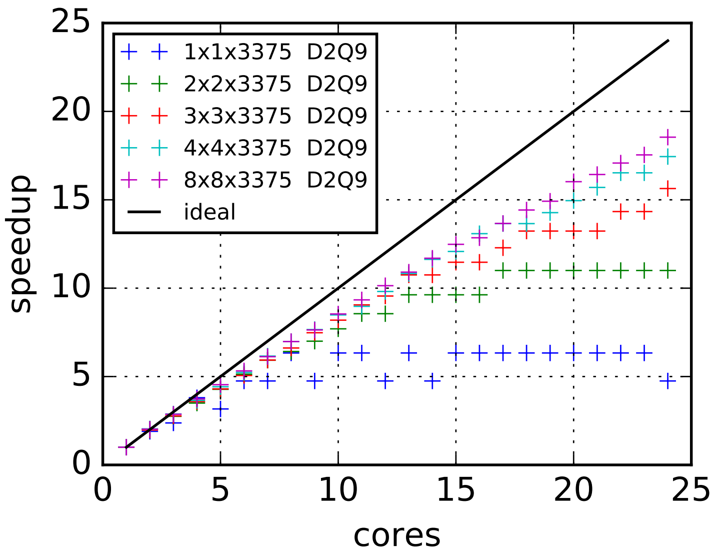


FIGURE 5. Multithread scaling for the D2Q9 model and a collection of square meshes from 1 to 64 macro-elements.

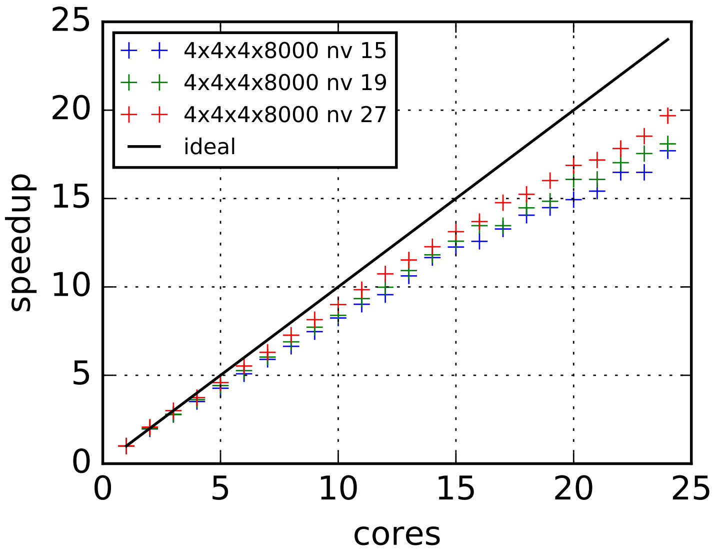


FIGURE 6. Multithread scaling on a 4x4x4 macro-element mesh for models D3Q15, D3Q19 and D3Q27

mesh subdivided into 720 macrocells. The workload distribution across nodes is done using a standard domain decomposition approach: the mesh is partitioned statically into as many sub-domains as computing nodes, ranging from 1 to 4 for our experiment on the IRMA-ATLAS cluster. From an implementation point of view, the transition from a multithreaded code to a hybrid MPI/multithread one is made fairly easy by StarPU. When

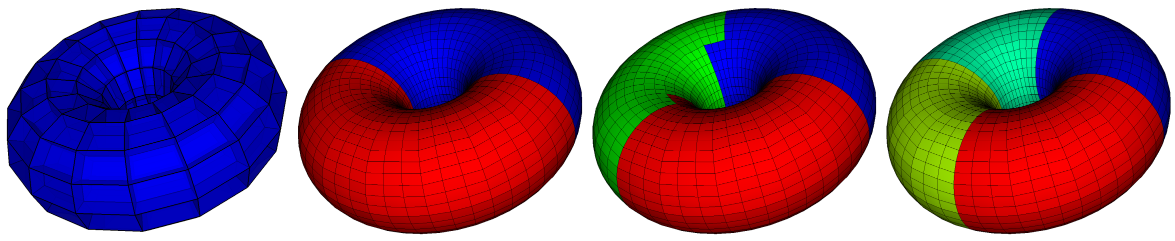


FIGURE 7. Toroidal macromesh (720 macrocells) - Mesh partitions used in the MPI scaling tests.

Nthreads/Nmpi	1	2	3	4
14	6862	2772	1491	1014

TABLE 1. Wall time (in seconds) for the D3Q15 model for 1 to 4 mpi processes with 14 threads per process.

declaring data to StarPU, one simply has to specify the MPI process owning the data. At runtime, each MPI process hosts a local scheduler instance which acts only on data relevant to the local execution graph. All MPI communications are handled transparently by the local scheduler when inter-node data transfers are necessary. In table 1 we show the wall time for a hundred iterations of the full scheme for the D3Q15 model. The number of available threads per node is set to 14, matching the number of velocities actually participating in transport (there is one null-velocity in the model). We observe a super-linear scaling when the load is spread from 1 to 4 nodes. This is not surprising for such an experiment with fixed total problem size. Indeed, both the memory load and size of the local task graph for each decrease when the number of sub-domains increases.

5. Numerical results

5.1. **Euler with gravity**

For this test case we consider the isothermal Euler equations in two dimensions with a constant gravity source term

$$\partial_t \rho + \partial_k(\rho \mathbf{u^k}) = 0, \tag{24}$$

$$\partial_t(\rho \mathbf{u}^k) + \partial_j \mathbf{\Pi^{kj}} = \rho \mathbf{g}, \tag{25}$$

with $\mathbf{\Pi} = \begin{bmatrix} \rho c^2 + \rho u_x^2 & \rho u_x u_y \\ \rho u_x u_y & \rho c^2 + u_y^2 \end{bmatrix}$ and $\mathbf{g} = -g\mathbf{e}_y$.

The conservative variables vector is thus $\mathbf{w} = [\rho, \rho u_x, \rho u_y]^t$. The kinetic model is the standard $D2Q9$ one with nine velocities

$$\mathbf{V} = \lambda \text{diag}\,[(0,0),(1,0),(0,1),(-1,0),(0,-1),(1,1),(-1,1),(-1,-1),(1,-1)] \tag{26}$$

and the (3×9) projection matrix P reads

$$P = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & \lambda & 0 & -\lambda & 0 & \lambda & -\lambda & -\lambda & \lambda \\ 0 & 0 & \lambda & 0 & -\lambda & \lambda & \lambda & -\lambda & -\lambda \end{bmatrix}, \tag{27}$$

i.e $\rho = \sum_i f_i$, $\rho \mathbf{u} = \sum_i f_i \mathbf{v}_i$.

The equilibrium distribution function is given by

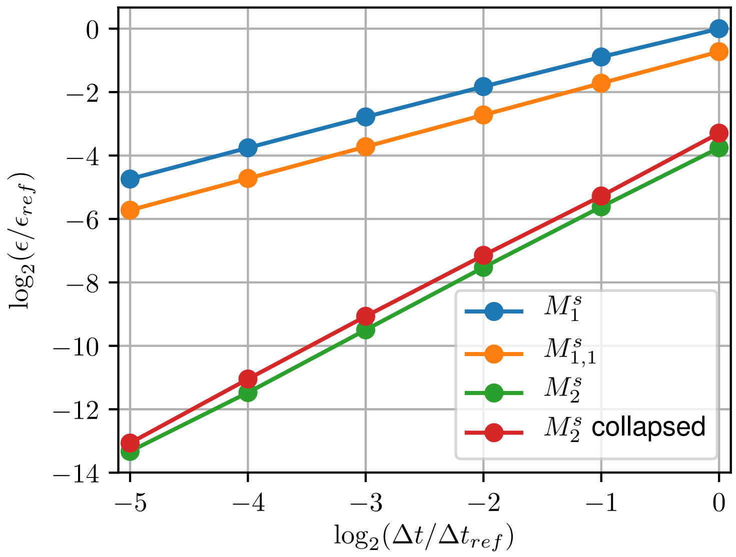


FIGURE 8. Time order convergence for the $2D$ Euler gravity test case with $D2Q9$ model. Convergence is estimated using the relative L^2 error ϵ on macroscopic variables with respect to the analytical solution at $t_{max} \approx 0.12$. The reference values $(\Delta t_{ref}, \epsilon_{ref})$ of the logarithmic scale are $\Delta t_{ref} = 0.024, \epsilon_{ref} = 0.0143$.

$$f_i = w_i \rho \left(1 + \frac{(\mathbf{u}\cdot\mathbf{v}_i)}{c^2} + \frac{(\mathbf{u}\cdot\mathbf{v}_i)^2}{2c^4} - \frac{\mathbf{u}\cdot\mathbf{u}}{2c^2}\right) \tag{28}$$

with $c = \lambda/\sqrt{3}$, and the weights $w_0 = \frac{4}{9}$, $w_i = \frac{1}{9}$ for $i = 1, \dots, 4$, $w_i = \frac{1}{36}$ for $i = 5, \dots, 8$.
The stationary solution for a fluid at rest in the gravity field is

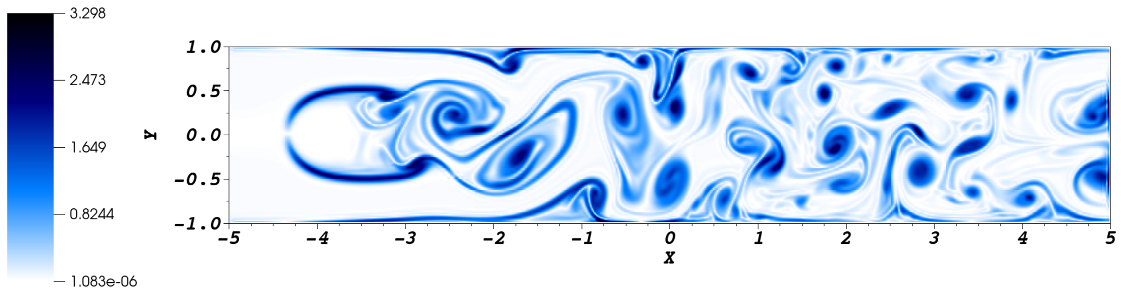
$$\rho = \rho_0 \exp(-gy/c^2), \qquad \mathbf{u} = 0. \tag{29}$$

For this test case, the numerical scheme is made up of three stages: a transport step (T), a source step (S) where the source is applied on the equilibrium part of the distribution function, and the collision step (C). Due to the absence of explicit time dependency and the linearity in $\mathbf{w}$ of the source, the local nonlinear resolution of the source operator converges in one Picard iteration. All steps are implemented as weighted implicit schemes, parametrized by a weight θ and a time step Δt.

We compared several 1^{st} and 2^{nd} order splitting schemes built up from either fully implicit ($\theta = 1$) first order or Crank-Nicolson ($\theta = \frac{1}{2}$) steps:

- Lie first order splitting scheme $M_1^s = T_1(\Delta t)S_1(\Delta t)C_1(\Delta t)$ with first order building blocks.
- Lie first order splitting scheme $M_{1,2}^s = T_2(\Delta t)S_2(\Delta t)C_2(\Delta t)$ with second order building blocks, for which the order loss comes from the splitting error.
- a palindromic second order Strang scheme $M_2^s = T_2(\frac{\Delta t}{2})S_2(\frac{\Delta t}{2})C_2(\Delta t)S_2(\frac{\Delta t}{2})T_2(\frac{\Delta t}{2})$.
- a collapsed version of the second order Strang scheme M_2^s for which the last transport substep of each global step and the first transport substep of the next one are fused in a single $T_2(\Delta t)$ substep except obviously for the first and last time steps of the simulation.

We perform time order convergence tests on a $2D$ square mesh partitioned into 4×4 macrocells with 1024 points per macrocell. As shown by Fig. 8, we obtain the expected convergence orders for each of the splitting schemes used.



was

Figure 9. . Flow around cylindrical obstacle. Vorticity norm $|\nabla \times u|$ at $t = 83$, showing the turbulent field behind the obstacle.

5.2. **2D Flow around a cylinder using a penalization method**

In this test case we consider the flow of a fluid in a rectangular duct with a cylindrical solid obstacle. The simulation domain is the rectangle $[-1, 1] \times [-5, 5]$.

The effect of the obstacle on the flow is modeled using a volumic source term of the form

$$\mathbf{s} = K(\mathbf{x})(\mathbf{w} - \mathbf{w}_s), \tag{30}$$

with $\mathbf{w}_s = [1.0, 0, 0]^t$ the target fluid state in the “solid” part of the domain and the relaxation frequency $K(\mathbf{x})$ is given by

$$K(\mathbf{x}) = K_s \exp(-\kappa(\mathbf{x} - \mathbf{x}_c)^2), \tag{31}$$

with $K_s = 300$, $\mathbf{x}_c = [-4, 0]^t$and $\kappa = 40$. The net effect is a very stiff relaxation towards a flow with zero velocity and the reference density near the center $\mathbf{x}_c$ of the frequency mask. The effective diameter of the cylinder for this simulation is about 0.5. The initial state, which is also applied at the duct boundaries for the whole simulation is $\rho = 1, u_x = 0.03, u_y = 0$. Accounting for the fact that for this model the sound speed is $1/\sqrt{3}$, the Mach number of the unperturbed flow is approximately 0, 017. The simulation is performed on a macromesh with 16×16 macrocells stretched with a 1 : 5 aspect ratio to match the domain dimension; each macrocell contains 12×60 integration points. On figure 9 we show the vorticity norm at $t = 83$, when turbulence is well developed in the wake of the obstacle.

6. Conclusion

In this paper, we have presented an optimized implementation of the Palindromic Discontinuous Galerkin Method for solving kinetic equations with stiff relaxation. The method presents the following interesting features:

- It can be used for solving any hyperbolic system of conservation laws.
- It is asymptotic-preserving with respect to the stiff relaxation.
- It is implicit and thus is not limited by CFL conditions.
- Despite being formally implicit, it requires actual implicit computations only at the subcell level, while the global computation is basically explicit.
- It is easy to increase the time order with a composition method.
- It presents many opportunities for parallelization and optimization: in this paper we have presented the parallelization of the method with the aid of the MPI version of the StarPU runtime system. In this way we address both shared memory and distributed memory MIMD parallelism.

Our perspectives are now to apply the method for computing MHD instabilities in tokamaks. We will also try to extend the method to more general boundary conditions.

References

[1] Denise Aregba-Driollet and Roberto Natalini. Discrete kinetic schemes for multidimensional systems of conservation laws. *SIAM Journal on Numerical Analysis*, 37(6):1973–2004, 2000.

[2] Cédric Augonnet, Olivier Aumage, Nathalie Furmento, Raymond Namyst, and Samuel Thibault. StarPU-MPI: Task Programming over Clusters of Machines Enhanced with Accelerators. In Siegfried Benkner Jesper Larsson Träff and Jack Dongarra, editors, *EuroMPI 2012*, volume 7490 of *LNCS*. Springer, September 2012. Poster Session.

[3] François Bouchut, François Golse, and Mario Pulvirenti. *Kinetic equations and asymptotic theory*. Elsevier, 2000.

[4] David Coulette, Emmanuel Franck, Philippe Helluy, Michel Mehrenberger, and Laurent Navoret. *Palindromic Discontinuous Galerkin Method*, pages 171–178. Springer International Publishing, Cham, 2017.

[5] Timothy A Davis and Ekanathan Palamadai Natarajan. Algorithm 907: KLU, a direct sparse solver for circuit simulation problems. *ACM Transactions on Mathematical Software (TOMS)*, 37(3):36, 2010.

[6] Christophe Geuzaine and Jean-François Remacle. Gmsh: A 3-D finite element mesh generator with built-in pre-and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 79(11):1309–1331, 2009.

[7] Benjamin Graille. Approximation of mono-dimensional hyperbolic systems: A lattice Boltzmann scheme as a relaxation method. *Journal of Computational Physics*, 266:74–88, 2014.

[8] Ernst Hairer, Christian Lubich, and Gerhard Wanner. *Geometric numerical integration: structure-preserving algorithms for ordinary differential equations*, volume 31. Springer Science & Business Media, 2006.

[9] Claes Johnson, Uno Nävert, and Juhani Pitkäranta. Finite element methods for linear hyperbolic problems. *Computer methods in applied mechanics and engineering*, 45(1):285–312, 1984.

[10] Robert I McLachlan and G Reinout W Quispel. Splitting methods. *Acta Numerica*, 11:341–434, 2002.

[11] Salli Moustafa, Mathieu Faverge, Laurent Plagne, and Pierre Ramet. 3D cartesian transport sweep for massively parallel architectures with PARSEC. In *Parallel and Distributed Processing Symposium (IPDPS), 2015 IEEE International*, pages 581–590. IEEE, 2015.