Received: 10 October 2018 | Revised: 15 July 2019 | Accepted: 17 July 2019

DOI: 10.1002/fld.4761

RESEARCH ARTICLE

WILEY

A discontinuous Galerkin immersed boundary solver for compressible flows: Adaptive local time stepping for artificial viscosity–based shock-capturing on cut cells

Markus Geisenhofer[1,2] | **Florian Kummer**[1] | **Björn Müller**[1,2]

[1]Chair of Fluid Dynamics, Technische Universität Darmstadt, Darmstadt, Germany

[2]Graduate School of Computational Engineering, Technische Universität Darmstadt, Darmstadt, Germany

Correspondence
Markus Geisenhofer, Chair of Fluid Dynamics, Technische Universität Darmstadt, Germany.
Email: geisenhofer@fdy.tu-darmstadt.de

Present Address
Markus Geisenhofer, Otto-Berndt-Straße 2,
D-64287 Darmstadt, Germany

Funding information
German Research Foundation (Deutsche Forschungsgemeinschaft), Grant/Award Number: 1194/B06 and WA 2610/2-1

Summary

We present a higher-order cut cell immersed boundary method (IBM) for the simulation of high Mach number flows. As a novelty on a cut cell grid, we evaluate an adaptive local time stepping (LTS) scheme in combination with an artificial viscosity–based shock-capturing approach. The cut cell grid is optimized by a nonintrusive cell agglomeration strategy in order to avoid problems with small or ill-shaped cut cells. Our approach is based on a discontinuous Galerkin discretization of the compressible Euler equations, where the immersed boundary is implicitly defined by the zero isocontour of a level set function. In flow configurations with high Mach numbers, a numerical shock-capturing mechanism is crucial in order to prevent unphysical oscillations of the polynomial approximation in the vicinity of shocks. We achieve this by means of a viscous smoothing where the artificial viscosity follows from a modal decay sensor that has been adapted to the IBM. The problem of the severe time step restriction caused by the additional second-order diffusive term and small nonagglomerated cut cells is addressed by using an adaptive LTS algorithm. The robustness, stability, and accuracy of our approach are verified for several common test cases. Moreover, the results show that our approach lowers the computational costs drastically, especially for unsteady IBM problems with complex geometries.

KEYWORDS

compressible flow, discontinuous Galerkin, immersed boundary, level set, supersonic, time integration

1 | INTRODUCTION

Within the past decades, higher-order discontinuous Galerkin (DG) methods[1-3] have gained a lot of popularity in the fluid dynamics research community due to their favorable properties such as cell locality, applicability to arbitrary geometries on unstructured grids, and an efficient parallelization. This also makes them a suitable candidate for high performance computing (HPC) applications.[4]

When handling complex realistic flow situations, eg, high Mach number flows that involve different discontinuous flow phenomena, DG methods are still susceptible to stability issues caused by under-resolution or oscillating polynomial solutions in the vicinity of shocks.[5-7] To overcome this problem, several approaches have been developed, eg, limiting procedures such as weighted essentially nonoscillatory schemes[8,9] or a posteriori limiting approaches.[10,11]

While these approaches have their advantages, the ultimate simplicity of the artificial viscosity approach originally introduced by Von Neumann and Richtmyer[12] remains appealing to this day. The basic idea is the addition of a second-order diffusive term in order to smooth the shocks over a layer that can be adequately resolved by the numerical scheme and, thus, preventing undesired oscillations. Bassi and Rebay[13,14] along with Bey and Oden[15,16] already demonstrated its potential for higher-order simulations in the early days of DG methods. However, determining the "correct" amount of artificial viscosity across a large range of parameters and flow configurations has remained a delicate issue. Persson and Peraire[5] proposed an extremely appealing solution for this issue that uses the modal decay of the DG coefficients[17] as a robust indicator for the smoothness of the solution.

Overcoming the problems of a piecewise-constant artificial viscosity, which introduces additional oscillations in the vicinity of the discontinuities, Barter and Darmofal[6,18] introduced a new higher-order state-based artificial viscosity formulation by means of a partial differential equation (PDE). Moreover, Persson[19] extended his work[5] with the introduction of an at least C^0-continuous artificial viscosity field formulation in order to deal with the aforementioned stability issues. In addition, other authors analyzed the performance of the modal-decay sensor and enhanced the basic version by introducing additional constraints or by developing a dynamic threshold setting, see the work of Klöckner et al[7] and Lv et al,[20] respectively.

The maximum admissible time step size of a DG method with a standard explicit global time stepping method is dictated by the smallest grid cell and the order of the polynomial approximation.[21,22] A shortcoming of artificial viscosity-based methods is the severe time step restriction caused by the second-order term, eg, see the work of Gassner et al.[23,24] The time step size has a quadratic dependency on the characteristic grid size and the polynomial degree, plus a linear dependency on the (artificial) viscosity.[23] This can lead to a decrease of the stable time step size by up to two orders of magnitude and, thus, increases the computational cost enormously. This motivates the use of local time stepping (LTS) methods where the cells are grouped into cell clusters and updated according to their local maximum stable time step size while still keeping time accuracy.[25-28] Besides that, the cell locality allows for a straightforward combination of DG methods with LTS approaches.[29] Winters and Kopriva[30] developed an explicit LTS scheme for a DG spectral element method on moving meshes using Adams-Bashforth multirate methods. They confirmed spectral and temporal convergence along with providing speedup and memory estimates for test cases on static and moving meshes. Krämer-Eis[31] extended their approach by implementing a conservative flux interpolation method in the context of compressible flows. Another conservative second-order LTS formulation was proposed by Krivodonova[32] who used Heun's method in the context of nonlinear conservation laws.

Immersed boundary methods (IBMs) are another well-suited field of application for LTS approaches. These were originally presented by Peskin[33] in the context of blood flows. Here, Peskin used a Cartesian grid whose boundary was not aligned with the geometrical topology. This makes IBM especially advantageous for simulations with underlying moving or complex geometries. Starting from this work, many enhancements for several numerical schemes were published, eg, see the work of Mittal and Iaccarino[34] for an overview. When representing the immersed boundary as the zero isocontour of a level set function, eg, as done by Qin and Krivodonova,[35] the main work is shifted from the representation of the geometry and the meshing procedure to the treatment of cells that are cut by the immersed boundary. Small or ill-shaped cut cells can cause significant stability problems stemming from ill-conditioning and also increase the computational load due to their cell local time step restriction. Müller et al generalized the work of Qin and Krivodonova[35] by introducing a hierarchical moment-fitting (HMF) strategy[36] for integration in cut cells and extended their approach to the compressible Navier-Stokes equations.[37] Here, a nonintrusive cell agglomeration strategy[35,38,39] was also being used to avoid ill-shaped and small cut cells.

Within this work, we propose a novel combination of a versatile IBM, a shock-capturing scheme, and an adaptive LTS strategy. For simplicity, we focus on inviscid high-speed flows in two dimensions, even though the algorithm directly generalizes to more general applications. For resolving all occurring discontinuous physical phenomena properly, we use an artificial viscosity–based shock-capturing strategy[5] and embed this approach into the IBM.[37] In order to improve the stability of the underlying DG scheme and to lower the computational costs, we present an explicit adaptive LTS strategy[30,31] that is based on a multirate Adams-Bashforth time integration scheme. Our LTS approach features a dynamic reclustering procedure in time, which makes the proposed method especially applicable for unsteady flow configurations. We do not focus on the pure "standalone" quality of the results as we simply use the standard shock-capturing approach by Persson and Peraire,[5,19] but rather on the interaction of the different methods (IBM, shock-capturing, and adaptive LTS).

This work is organized as follows. In Section 2, we present the compressible Euler equations in a nondimensional conservative form. In Section 3, we derive a standard DG discretization, before introducing an extension to the IBM. Here, we briefly outline the used cell agglomeration strategy. Section 4 deals with the artificial viscosity–based shock-capturing approach. The adaptive LTS scheme is described in Section 5. We assess the results of common two-dimensional test cases for the unsteady compressible Euler equations in terms of robustness, accuracy, and computational costs in Section 6. We end this work with a conclusion in Section 7.

2 | GOVERNING EQUATIONS

We consider the two-dimensional Euler equations for inviscid, compressible flow in a nondimensional conservative form

$$\frac{\partial \vec{U}}{\partial t} + \frac{\partial \vec{F}_1(\vec{U})}{\partial x} + \frac{\partial \vec{F}_2(\vec{U})}{\partial y} = 0, \tag{1}$$

where $\vec{U}$ is the vector of conserved quantities

$$\vec{U} = \begin{pmatrix} \rho \\ \rho u_1 \\ \rho u_2 \\ \rho E \end{pmatrix}, \tag{2}$$

and $\vec{F}_1$ and $\vec{F}_2$ are the convective fluxes given by

$$\vec{F}_1 = \frac{1}{\gamma \mathrm{M}_\infty^2} \begin{pmatrix} \rho u_1 \\ \rho u_1 u_1 + p \\ \rho u_1 u_2 \\ u_1(\rho E + p) \end{pmatrix}, \qquad \vec{F}_2 = \frac{1}{\gamma \mathrm{M}_\infty^2} \begin{pmatrix} \rho u_2 \\ \rho u_1 u_2 \\ \rho u_2 u_2 + p \\ u_2(\rho E + p) \end{pmatrix}, \tag{3}$$

where ρ is the fluid density, u_1 and u_2 are the components of the velocity vector $\vec{u}$, ρE is the total energy per unit mass, p is the pressure, γ the heat capacity ratio, and M_∞ is the reference Mach number. We set $\mathrm{M}_\infty = 1/\sqrt{\gamma}$ for the sake of simplicity. With this specific choice, equation system (1) matches its corresponding version with dimensions. Equation system (1) is supplemented with suitable initial and boundary conditions.

For closing equation system (1), we consider the ideal gas law

$$p(\rho, e) = (\gamma - 1)\rho e, \tag{4}$$

where we set the heat capacity ratio γ to 1.4 to model standard air conditions and denote e as the specific inner energy.

When the local Mach number $M = \|\vec{u}\|/a$ is equal or larger than 1, discontinuous flow phenomena like shocks can occur. We denote $a = \sqrt{\gamma p/\rho}$ as the local speed of sound. The polynomial DG approximation will start to oscillate in the vicinity of the discontinuities. In order to prevent the simulation from a failure, an appropriate shock-capturing strategy has to be applied. In this work, we follow an artificial viscosity–based approach[5] that adds a second order diffusive term to the original equation system (1)

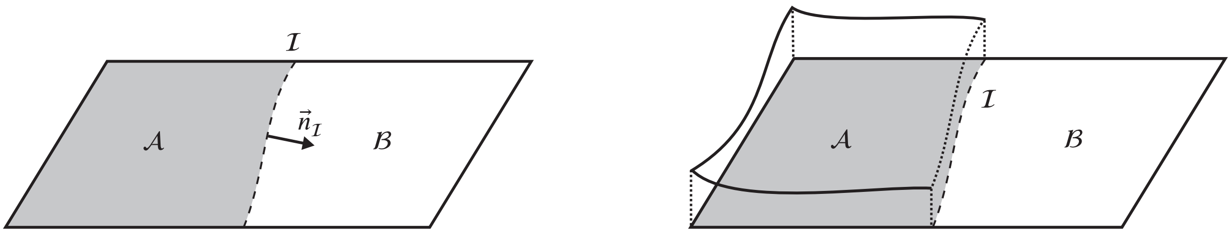
$$\frac{\partial \vec{U}}{\partial t} + \frac{\partial \vec{F}_1(\vec{U})}{\partial x} + \frac{\partial \vec{F}_2(\vec{U})}{\partial y} = \varepsilon \left(\frac{\partial^2 \vec{U}}{\partial x^2} + \frac{\partial^2 \vec{U}}{\partial y^2} \right), \tag{5}$$

where ε is the artificial viscosity parameter that still needs to be determined (cf Section 4.2). Artificial viscosity smoothens the undesired oscillations around the discontinuities. Note that equation system (5) is not purely hyperbolic anymore.

3 | DISCRETIZATION

In this section, we briefly introduce the discretization of the underlying DG IBM solver. Interested readers are referred to the works by Müller et al[37] and Kummer[40] for more details. The present solver is implemented in the open source software package *BoSSS*[*] that features a variety of applications in the context of computational fluid dynamics, eg, a solver for multiphase flows with a sharp interface approach[40] or an incompressible IBM solver for particle flows.[41]

[*] https://github.com/FDYdarmstadt/BoSSS



(A) The interface $\mathcal{I}$ separates a cut cell into two subdomains $\mathcal{A}$ and $\mathcal{B}$

(B) The polynomial approximation is cut at the interface $\mathcal{I}$

FIGURE 1 The configuration in a cut cell. The interface $\mathcal{I}$ (dashed line) with its normal vector $\vec{n}_\mathcal{I}$ is implicitly defined by the zero isocontour of a level set function. We consider the subdomain $\mathcal{A}$ (gray) as the fluid domain and the subdomain $\mathcal{B}$ (white) as void[37]

We discretize the entire computational domain $\Omega \subset \mathbb{R}^2$, which has to be polygonal and simply connected, into a discrete set of nonoverlapping cells $\mathscr{K}_h = \{\mathcal{K}_1, \ldots, \mathcal{K}_J\}$ with $\int_{K_i \cap K_j} 1\,\mathrm{d}x = 0 \quad \text{for} \quad i \neq j$, where h is a characteristic cell length scale. On this domain, we additionally define a set of edges $\mathscr{E}_i = \{\mathcal{E}_{i,1}, \ldots, \mathcal{E}_{i,E}\}$ for each cell $\mathcal{K}_i$ with $\mathcal{E}_{i,j} \subseteq \partial\mathcal{K}_i$. Next, we introduce a DG space

$$\mathbb{P}_P(\mathscr{K}_h) := \left\{ f \in L^2(\Omega) : \forall \mathcal{K}_i \in \mathscr{K}_h : f|_{\mathcal{K}_i \cap \Omega} = \sum_{0 \le k+l \le P} x^k y^l b_{kl},\, b_{kl} \in \mathbb{R} \right\}, \tag{6}$$

with a maximum total polynomial degree P. To incorporate the IBM into the discretization, we introduce a sufficiently smooth level function $\varphi(\vec{x}) \in C^2(\Omega)$ that partitions the numerical domain into a physical fluid domain

$$\mathcal{A} = \left\{ \vec{x} \in \Omega : \varphi(\vec{x}) > 0 \right\}, \tag{7}$$

the immersed boundary

$$\mathcal{I} = \partial\mathcal{A} \setminus \partial\Omega = \{\vec{x} \in \Omega : \varphi(\vec{x}) = 0\}, \tag{8}$$

and a void domain

$$\mathcal{B} = \{\vec{x} \in \Omega : \varphi(\vec{x}) < 0\}. \tag{9}$$

A cut scenario is illustrated in Figure 1.

Moreover, we define a discrete set of cut cells $\mathscr{K}_h^c = \{\mathcal{K}_1^c, \ldots, \mathcal{K}_J^c\}$ by writing the fluid segment as

$$\mathcal{K}_i^c = \mathcal{K}_i \cap \mathcal{A}. \tag{10}$$

The respective fluid surface of a cut cell $\partial\mathcal{K}_i^c$ consists of a set of edges

$$\left\{ \mathcal{E}_{i,e}^{\mathcal{A}} \right\}_{e=1,\ldots,E} = \{\mathcal{E}_{i,e} \cap \mathcal{A}\}_{e=1,\ldots,E}, \tag{11}$$

and the corresponding boundary segment of the interface

$$\mathcal{I}_i = \mathcal{K}_i \cap \mathcal{I}. \tag{12}$$

Up to this point, we have not introduced any limitation to the shape or volume fraction

$$\operatorname{frac}\left(\mathcal{K}_i^c\right) = \frac{\int_{\mathcal{K}_i^c} 1\,dV}{\int_{\mathcal{K}_i} 1\,dV}, \tag{13}$$

of cut cells. However, this can cause significant issues when the cut cells are ill-shaped or extremely small. In the context of DG, the maximum stable time step size of explicit time stepping schemes scales like $\Delta t \sim (h/P)^2$ for diffusive terms. Until here, we have only introduced a characteristic cell length scale h for uncut cells. In nonagglomerated cut cells, the length scales can be significantly smaller than in uncut cells, which leads to a severe diffusive time step restriction and, thus, to a drastic increase in computational costs. For details, the reader is referred to Section 5.1.

To overcome these challenges, we use a nonintrusive cell agglomeration strategy,[37] which shall be briefly outlined in the following. In principle, we follow the algorithm presented by Kummer.[42] We set an agglomeration threshold $0 \leq \delta < 1$ based on the volume fraction.[43] For every cut cell $\mathcal{K}_i^c$, we determine the edge neighbors

$$\mathcal{N}\left(\mathcal{K}_i^c\right) = \left\{ \mathcal{K}_j^c \in \mathscr{K}_h^c : \oint_{\mathcal{K}_i^c \cap \mathcal{K}_j^c \cap \mathcal{A}} 1\,dS > 0, i \neq j \right\}, \tag{14}$$

with the largest volume fraction. If $\text{frac}(\mathcal{K}_i^c) \leq \delta$, we define a *target* cell

$$\text{tar}\left(\mathcal{K}_i^c\right) = \underset{\substack{\mathcal{K}_j^c \in \mathcal{N}(\mathcal{K}_i^c) \\ \text{frac}\left(\mathcal{K}_j^c\right) > 0}}{\arg\max} \left(\text{frac}\left(\mathcal{K}_j^c\right)\right) \tag{15}$$

for every cell in the fluid domain $\mathcal{A}$. If $\text{frac}(\mathcal{K}_i^c) > \delta$,

$$\text{tar}\left(\mathcal{K}_i^c\right) = \mathcal{K}_i^c, \tag{16}$$

which means that a nonagglomerated cut cell is, formally, also a target cell. Based on this, we build an *agglomerated graph* consisting of nodes $\mathcal{K}_i^c$ and nondirectional edges $\{\mathcal{K}_i^c, \text{tar}(\mathcal{K}_i^c)\}$ for all cut cells $\mathcal{K}_i^c$ with $\text{tar}(\mathcal{K}_i^c) \neq \mathcal{K}_i^c$. These edges are called *agglomeration pairs*. The agglomeration graph is defined by the following properties.

- Isolated nodes $\mathcal{K}_i^c$ (nodes without agglomeration pairs) are nonagglomerated cut cells; we define $\mathcal{K}_i^{agg} = \mathcal{K}_i^c$.
- Each connected subset of the agglomerated graph forms one agglomerated cell $\mathcal{K}_i^{agg}$.
  - Let $\mathcal{K}_{j_1}^c, \mathcal{K}_{j_2}^c, \ldots, \mathcal{K}_{j_N}^c$ be any maximal connected subset of $\mathscr{K}_h^c$, ie, we have agglomeration pairs that connect these nodes/cells, and there are no further nodes/cells that can be connected through agglomeration pairs.
  - Out of the numbers $j_1, \ldots, j_N$, we pick one representative, usually one of those where $\text{frac}(\mathcal{K}_{j_i}^c)$ is maximal. Without loss of generality, let this index be j_1. We define an *agglomerated cell* as

$$\begin{aligned} \mathcal{K}_{j_1}^{agg} &= \bigcup_{i=1}^{N} \mathcal{K}_{j_i}^c, \\ \mathcal{K}_{j_2}^{agg}, \ldots, \mathcal{K}_{j_N}^{agg} &= \emptyset, \end{aligned} \tag{17}$$

    where we admit empty cells for the sake of simplicity.
- The agglomerated grid $\mathscr{K}_h^{agg}$ is the set that contains all cells $\mathcal{K}_i^{agg}$.

Our actual implementation exploits the locality of the outlined operations, requiring only a few more cell local matrix-vectors products per time step. We set $\delta = 0.3$, which is a good compromise between a reasonable conditioning of the mass matrices (this holds true for large δ values), also resulting in larger maximum stable time step sizes, and larger spatial discretization errors near the immersed boundary. An example for the presented cell agglomeration procedure is shown in Figure 2.

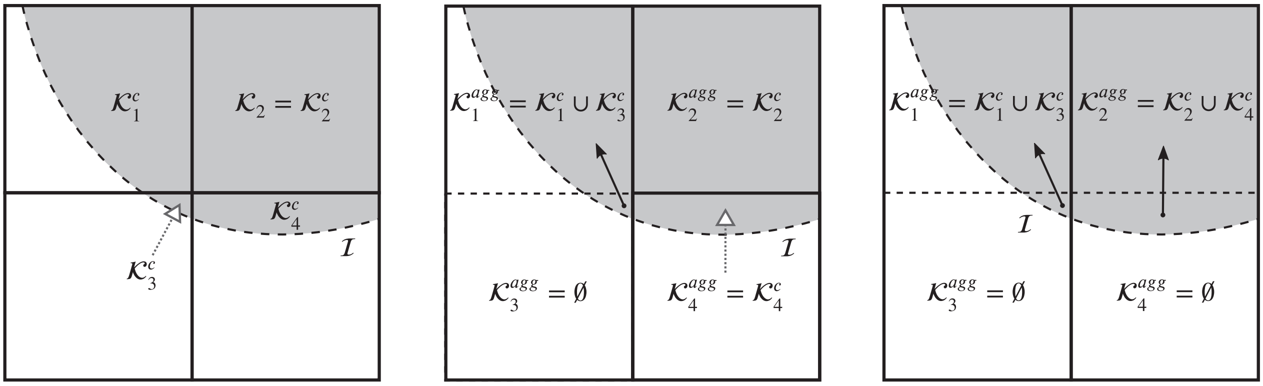
Finally, the IBM simulations presented in Sections 6.3 and 6.4 are simply DG simulations on the space

$$\mathbb{P}_P\left(\mathscr{K}_h^{agg}\right). \tag{18}$$

The accuracy and performance of an IBM strongly depends on the applied quadrature procedure over the fluid segments A_i and interface segments I_i of cut cells. In this work, we use the HMF strategy as first published by Müller et al.[36]

4 | SHOCK CAPTURING

We follow the artificial viscosity–based shock-capturing approach presented by Persson and Peraire.[5] The idea of artificial viscosity is to spread the discontinuities over a length scale that can be adequately resolved by the numerical scheme. The


$\mathcal{K}_1^c$
$\mathcal{K}_2 = \mathcal{K}_2^c$
$\mathcal{K}_4^c$
$\mathcal{I}$
$\mathcal{K}_3^c$
$\mathcal{K}_1^{agg} = \mathcal{K}_1^c \cup \mathcal{K}_3^c$
$\mathcal{K}_2^{agg} = \mathcal{K}_2^c$
$\mathcal{I}$
$\mathcal{K}_3^{agg} = \emptyset$
$\mathcal{K}_4^{agg} = \mathcal{K}_4^c$
$\mathcal{K}_1^{agg} = \mathcal{K}_1^c \cup \mathcal{K}_3^c$
$\mathcal{K}_2^{agg} = \mathcal{K}_2^c \cup \mathcal{K}_4^c$
$\mathcal{I}$
$\mathcal{K}_3^{agg} = \emptyset$
$\mathcal{K}_4^{agg} = \emptyset$

(A) Initial configuration on the standard grid

(B) Agglomerated grid for a small agglomeration threshold δ

(C) Agglomerated grid for a large agglomeration threshold δ

FIGURE 2 Illustration of the agglomeration procedure in cut cells. Black arrows denote the agglomeration direction (source → target), gray dotted arrows are only used for naming purposes. (A) Small cut cells will be agglomerated to their direct neighbor with the largest fluid volume fraction. (B) For small values of the agglomeration threshold δ, cut cell $\mathcal{K}_3^c$ (source) is agglomerated to cell $\mathcal{K}_1^c$ (target). (C) For large values of δ, cell $\mathcal{K}_4^c$ (source) is additionally agglomerated to cell $\mathcal{K}_2^c$ (target)[37]

resolution of a higher-order polynomial scales with $\mathcal{O}(h/P)$ for DG methods. Therefore, Persson and Peraire argue that it is reasonable to introduce a dependency of the artificial viscosity parameter on the resolution of the DG approximation space, ie, $\varepsilon = \mathcal{O}(h/P)$. They aim at the term *subcell resolution*, meaning the shock profile can be significantly thinner than the cell size when keeping the grid size h fixed while simultaneously increasing the polynomial degree P. The two-step strategy consists of a detection step (cf Section 4.1) and a smoothing step (cf Section 4.2).

4.1 | Shock sensor

First, let us write the cell local polynomial solution c_h in a cell $\mathcal{K}_i \in \mathscr{K}_h^{agg}$, $\mathcal{K}_i \neq \emptyset$, on the agglomerated grid as

$$c_{h,i} := c_h|_{\mathcal{K}_i} \in \mathbb{P}_P\left(\left\{\mathcal{K}_i^{agg}\right\}\right). \tag{19}$$

Moreover, we introduce a truncated solution on a basis with order $P-1$

$$\hat{c}_{h,i} = \Pi_{P-1}(c_{h,i}), \tag{20}$$

by using the projection operator

$$\begin{aligned}\Pi_{P-1} : \mathbb{P}_P\left(\left\{\mathcal{K}_i^{agg}\right\}\right) &\to \mathbb{P}_{P-1}\left(\left\{\mathcal{K}_i^{agg}\right\}\right)\\ u &\mapsto \hat{u},\end{aligned} \tag{21}$$

with the essential property $\langle u - \hat{u} | v \rangle = 0, \forall v \in \mathbb{P}_{P-1}(\{\mathcal{K}_i^{agg}\})$, and $\langle \cdot | \cdot \rangle$ denoting the standard L^2 scalar product.

Persson and Peraire[5] now define a cell local shock sensor

$$S_i(t) = \frac{\langle c_{h,i} - \hat{c}_{h,i} | c_{h,i} - \hat{c}_{h,i} \rangle}{\langle c_{h,i} | c_{h,i} \rangle}. \tag{22}$$

For a smooth and at least continuous cell local solution, we can expect the coefficients of the polynomial expansion to decay quickly with a rate of $\sim 1/P^4$ in analogy to spectral analysis.[44] Thus, the same holds for the shock sensor S_i, which was confirmed through numerical experiments.[5] In this work, we use the density as the numerical input variable for determining the sensor values $S(t)$.

When considering an orthogonal basis Φ_j for the cell local solution (we skip the index h for better readability),

$$c_i = \sum_{j=1}^{N(P)} \tilde{c}_{i,j} \Phi_{i,j}, \tag{23}$$

with the coefficients $\tilde{c}_{i,j}$ and $N(P)$ being the number of terms in the polynomial solution, the truncated expansion (20) reads

$$\hat{c}_i = \sum_{j=1}^{N(P-1)} \tilde{c}_{i,j} \Phi_{i,j}. \tag{24}$$

For the usage on cut cells, we reformulate the cell local shock sensor (22) as

$$S_i(t) = \frac{\sum_{j=N(P-1)}^{N(P)} \sum_{k=N(P-1)}^{N(P)} \tilde{c}_{i,j}\tilde{c}_{i,k} \int \Phi_{i,j}\Phi_{i,k}\, dV}{\sum_{j=1}^{N(P)} \sum_{k=1}^{N(P)} \tilde{c}_{i,j}\tilde{c}_{i,k} \int \Phi_{i,j}\Phi_{i,k}\, dV}, \tag{25}$$

where $\int \Phi_{i,j}\Phi_{i,k}\, dV$ is the cell local mass matrix $\mathbf{M}_i$. Note that the structure of $\mathbf{M}_i$ strongly depends on the shape of the cut cells and is therefore different from the mass matrix in uncut cells. For uncut cells on the standard grid, a simplification can be obtained as the mass matrix is equal to the identity matrix ($\mathbf{M}_i = \mathbf{I}$) in the considered setting

$$S_i(t) = \frac{\sum_{j=N(P-1)}^{N(P)} \tilde{c}_j^2}{\sum_{j=1}^{N(P)} \tilde{c}_j^2}. \tag{26}$$

In the remainder of this work, we use this sensor as input variable for the calculation of the artificial viscosity due to its favorable properties such as the h/p-scaling and the straightforward extension for the usage on an agglomerated cut cell grid. In our numerical experiments (cf Section 6), there was no limitation in the applicability or performance of this sensor on an agglomerated cut cell grid. Furthermore, we do not focus on the improvements[7] or other variants of this shock sensor[6,20] but rather apply the basic version[5,19] to demonstrate the applicability in the IBM.

4.2 | Artificial viscosity

When a troubled cell has been detected, artificial viscosity has to be added depending on the strength of the shock. The underlying approach[5] uses a smooth Heaviside function for determining the preliminary amount of artificial viscosity (indicated by a prime)

$$\text{for } \vec{x} \in \mathcal{K}_i: \qquad \varepsilon'(\vec{x},t) = \begin{cases} 0, & \text{if} \quad s_i < s_0 - \kappa \\ \frac{\varepsilon_0}{2}\left(1 + \sin\frac{\pi(s_i - s_0)}{2\kappa}\right), & \text{if} \quad s_0 - \kappa \leq s_i \leq s_0 + \kappa \\ \varepsilon_0, & \text{if} \quad s_i > s_0 + \kappa, \end{cases} \tag{27}$$

where $s_i = \log_{10}(S_i)$. Additionally, as suggested by Persson and Peraire,[5] we add scaling factors to the maximum viscosity value $\varepsilon_0 \sim h/P$ and to the user-defined sensor value $s_0 \sim \log_{10}(1/P^4)$. Several authors[7] hinted on a typographical error in their publication[5] as the logarithm of the sensor value should not scale with $s_0 \sim 1/P^4$ but rather with $s_0 \sim \log_{10}(1/P^4)$. The parameter $\kappa \sim \mathcal{O}(1)$ has to be adjusted in order to obtain the desired sharp shock profile.

The added amount of artificial viscosity also depends on some local characteristic velocity λ. Klöckner et al[7] started from investigating the fundamental solution of the diffusion equation $\partial u/\partial t = \varepsilon\Delta u$ and derived scaling expressions of $\varepsilon_0 \sim \lambda$ and $\varepsilon_0 \sim h/P$ verifying the result in the work of Barter and Darmofal.[6] The Euler equations (1) feature several characteristic velocities, eg, the supersonic propagation speed of a shock wave or the sonic background propagation velocity of a contact discontinuity. The question remains which velocity to choose. Klöckner et al[7] state that it is reasonable to take the maximum local characteristic velocity $\lambda_{\max}$ in order to get a running implementation with the price of smoothing weak features, such as contact discontinuities, more than necessary. Klöckner et al[7] empirically derived a correction factor of 0.5 for the compressible Euler equations (1). The formula for the determination of artificial viscosity in a cell $\mathcal{K}_i$ can now be written as

$$\varepsilon_i(\vec{x},t) = \varepsilon_i'(\vec{x},t) \cdot 0.5 \cdot \lambda_{\max} \cdot \frac{h}{P}. \tag{28}$$

We tested several boundary conditions for the artificial viscosity flux. Our implementation is based on a standard symmetric interior penalty (SIP) formulation, which was first introduced by Arnold[45] (for a detailed review, see the subsequent work[46]). From a theoretical point of view, the boundary condition for the viscous terms is a delicate issue that affects the physical as well as the mathematical properties of the governing PDE. As this work is focused on the applicability in the

IBM, we abstain from going into details. We note, however, that we achieved best results in terms of stability and accuracy when using zero Neumann boundary conditions, which have thus been applied in the remainder of this work.

There are several publications[6,7,20] about the advantages and drawbacks of the original approach,[5] each of them having their own improvements and tunings of the underlying method. We stick to the standard approach except for the modifications presented above. Anyway, we would like to mention that it is essential to enforce C^0-continuity for the artificial viscosity field (27) in order to obtain a robust and reliable method.[6,19]

First, we define a piecewise linear continuous space

$$\mathbb{Q}_1(\mathscr{K}_h) := \left\{ f \in L^2(\Omega) \, : \, \forall \mathcal{K}_i \in \mathscr{K}_h \, : \, f|_{\mathcal{K}_i \cap \Omega} = \sum_{0 \leq k,l \leq 1} x^k y^l b_{kl}, b_{kl} \in \mathbb{R} \right\} \cap C^0(\Omega), \tag{29}$$

with a maximum polynomial degree of $P = 1$ (which is different from space (6)). We define the corresponding projection operator

$$\begin{aligned} \Pi_{C^0} \, : \, L^2(\Omega) &\rightarrow \mathbb{Q}_1(\mathscr{K}_h) \\ u &\mapsto u_{C^0}, \end{aligned} \tag{30}$$

with the essential property $\langle u - u_{C^0} | v \rangle = 0, \forall v \in \mathbb{Q}_1$. Finally, we project the cellwise constant artificial viscosity field (28) onto the space $\mathbb{Q}_1$

$$\varepsilon_{C^0}(\vec{x}, t) = \Pi_{C^0}(\varepsilon(\vec{x}, t)), \tag{31}$$

in order to enforce continuity. The sensor values (22) and the artificial viscosity values (31) are updated in each (local) time step (cf Section 5). Note that we chose $\mathscr{K}_h$ instead of $\mathscr{K}_h^{agg}$ for technical reasons. Kinks of the artificial viscosity field at cell boundaries are present on the background grid as well as on the agglomerated grid. After the C^0-projection, the spatial operator is first evaluated on the background grid before starting the cell agglomeration routine. Since the operator is evaluated on $\mathbb{P}(\mathscr{K}_h^c)$ and then projected onto $\mathbb{P}(\mathscr{K}_h^{agg})$, no, respectively, a little quadrature error is introduced by a kink in ϵ_{C^0} on an agglomerated cell.

5 | ADAPTIVE LOCAL TIME STEPPING

First, we outline the calculation of the time step restrictions in Section 5.1. Second, we present the strategy of an adaptive LTS algorithm in Section 5.2, where we focus on the extension to a dynamic reclustering procedure in time in Section 5.3.

5.1 | Time step restrictions

For a DG method, the maximum admissible time step size is dictated by the characteristic length scale h and the order of the approximating polynomial P, eg, as derived by Cockburn and Shu.[21,22]

Analogous to the early publication by Courant et al[47] who first postulated the *CFL restriction*, a sufficiently accurate pendant for convection dominated problems was defined by Cockburn and Shu[22] in the DG context

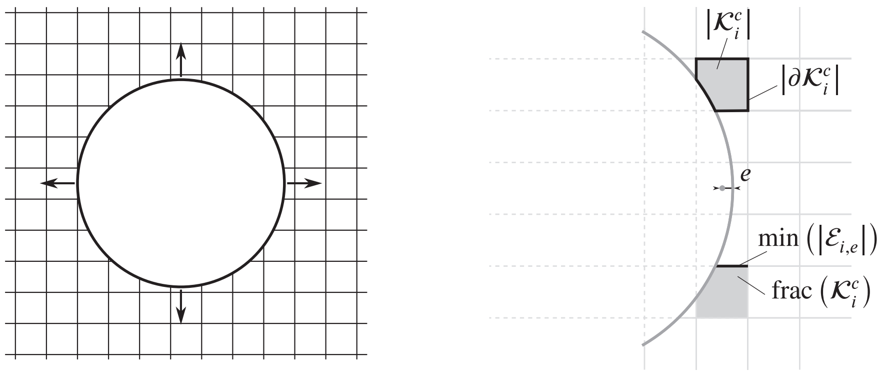
$$\Delta t_c \leq \frac{C_{\mathrm{CFL}}}{2P+1} \frac{h}{||\vec{u}|| + a}, \tag{32}$$

where $0 < C_{\mathrm{CFL}} \leq 1$ is a user-defined scaling factor and $||\vec{u}|| + a$ describes the fastest propagation velocity, eg, of a shock, in the hyperbolic Euler equations (1).

For diffusion dominated problems, Gassner et al[23] derived a similar expression

$$\Delta t_d \leq \frac{C_{\mathrm{DFL}}}{(2P+1)^2} \frac{h^2}{\sqrt{d}\,\nu \max\left(\frac{4}{3}, \frac{\gamma}{\mathrm{Pr}}\right)}, \tag{33}$$

where $0 < C_{\mathrm{DFL}} \leq 1$ is another scaling factor and d is the number of spatial dimensions. Appropriate suggestions for the scaling factors C_{CFL} and C_{DFL} can be found in their work.[23]


$|\mathcal{K}_i^c|$
$|\partial\mathcal{K}_i^c|$
e
$\min\left(|\mathcal{E}_{i,e}|\right)$
$\text{frac}\left(\mathcal{K}_i^c\right)$

(A) The cylinder is slightly shifted in order to produce different cut configurations

(B) Three different measures for h

FIGURE 3 Determination of the cell local characteristic length scale h on cut cells according to Krämer-Eis.[31] The steady compressible flow around a cylinder is investigated, where the maximum stable time step size is determined empirically and afterwards compared to theory for different measure of h

For convection-diffusion problems (cf Equation (5)), we use the harmonic sum

$$\Delta t = \frac{1}{\frac{1}{\Delta t_c} + \frac{1}{\Delta t_d}} \tag{34}$$

as a more reliable measure of the maximum admissible time step size than the minimum of the pure convective and diffusive time step restrictions (32) and (33), see the work of Watkins et al.[48]

In the past, various authors[23,48-50] widely studied the calculation of the maximum stable time step size on structured and unstructured boundary-fitted grids. In contrast, this topic was hardly addressed for a cut cell grid in the IBM. The challenging part is the determination of a suitable choice of h on cut cells. Krämer-Eis[31] investigated this issue for the *BoSSS* IBM solver that has been used for this work. The following three different cell measures:

(i) $|\mathcal{K}_i^c| / |\partial\mathcal{K}_i^c|$,
(ii) $2e$ (e being the distance from the barycenter of a cell to the closest edge or interface),
(iii) $\min(|\mathcal{E}_{i,e}|)\ \text{frac}(\mathcal{K}_i^c)$,

were compared for a steady compressible flow configuration where a cylinder is embedded in a uniform grid, see Figure 3. In order to quantify the influence of the different measures for different cut configurations, the cylinder was slightly shifted in horizontal and vertical direction. For all cut scenarios, the maximum stable time step sizes were empirically determined by numerical simulations. The theoretical time step size was additionally calculated using Equations (32) and (33). The average percentage deviations from the empirically determined time size were 13.6% (measure (i)), 33.4% (measure (ii)), and 25.6% (measure (iii)). This indicates that measure (i) is a suitable candidate for the IBM while measures (ii) and (iii) underestimate the maximum stable time step sizes in cut cells and thus lead to an unnecessarily large amount of time steps.

We abstain from going into details even further as the presented choice of h is sufficient for the purpose of this work. Krivodonova and Qin[51] showed that there is no generally valid choice of h for all cut scenarios. Moreover, they also state that the number and position of small cells on a coarse grid with otherwise identical cells significantly influences the maximum stable time step size.

For the remainder of this work, we choose the cell local characteristic length scale of a cell $\mathcal{K}_i$ as follows:

$$h_i = \begin{cases} \min(|\mathcal{E}_{i,e}|), & \text{if } \mathcal{K}_i \text{ is a standard or agglomeration target cell} \\ \left|\mathcal{K}_i^c\right| / \left|\partial\mathcal{K}_i^c\right|, & \text{if } \mathcal{K}_i \text{ is a nonagglomerated cut cell.} \end{cases} \tag{35}$$

5.2 | Derivation of the basic scheme

Our implementation of an explicit LTS algorithm is based on the approach by Winters and Kopriva.[30] Their scheme is a simplification of the multirate linear multistep method of Gear and Wells[52] that was combined with two-rate Adams-Bashforth schemes, eg, as done by Stock.[53] We use a variable coefficient Adams-Bashforth scheme to evolve the solution in time (cf Appendix A.1). For the standard static case, Krämer-Eis[31] developed a flux correction procedure that results in a conservative LTS formulation.

We make the following assumptions,[30,31] which will be explained in the following.

- The grid cells are grouped into a *set of cell clusters* $\mathscr{M} = \{\mathcal{M}_1, \dots, \mathcal{M}_M\}$, where $\mathcal{M}_i \subseteq \mathscr{K}_h^{agg}$ or $\mathcal{M}_i \subseteq \mathscr{K}_h$. The clustering is based on a cell local characteristic measure. In this work, we use the maximum admissible cell local time step size Δt_i.
- The cell cluster time levels are synchronized at *synchronization levels*.
- The time scales of the cell clusters differ only by *integer factors*.

In the following, we briefly introduce a multirate time integration method for a coupled system of ordinary differential equations (ODEs) $\partial\vec{y}/\partial t = F(t, \vec{y})$, assuming a fast time scale and a slow time scale with superscripts $(\cdot)^f$ and $(\cdot)^s$, as done in previous works.[30,52,53] Thus, the state vector and the right-hand side can be divided into *slow*, *fast*, and *coupled* components

$$\vec{y} = \begin{bmatrix} \vec{y}^f \\ \vec{y}^s \end{bmatrix}, \tag{36}$$

and

$$F = \begin{bmatrix} F_{ff} & F_{fs} \\ F_{sf} & F_{ss} \end{bmatrix}, \tag{37}$$

where $\vec{y}^f$ has to be updated using smaller time steps than $\vec{y}^s$, ie,

$$\Delta t_s > \Delta t_f. \tag{38}$$

In Equation (37), the main diagonal components F_{ff} and F_{ss} can be calculated immediately as they are on the same time scales.[30] In the DG context, this step corresponds to the evaluation of the volume and surface integrals. The coupled terms F_{sf} and F_{fs}, however, require an inter/extrapolation procedures from slow to fast time scales and vice versa, ie, the evaluation of surface integrals along between neighboring cells. An interpolation is needed in order to calculate the coupled component F_{sf}, whereas an extrapolation is needed for F_{fs}, which is an inherently unreliable calculation.[30] Due to the weak coupling between DG elements, the extrapolation F_{fs} can be treated separately, eg, see the work of Winters and Kopriva[30] for details. Therefore, Equation (37) reduces to

$$F = \begin{bmatrix} F_{ff} & 0 \\ F_{sf} & F_{ss} \end{bmatrix}. \tag{39}$$

In the end, we only need to interpolate the right-hand side values from slow to fast time scales F_{sf} at intermediate time levels. In total, less operator evaluations are needed for LTS than for a standard global Adams-Bashforth method, but additional work has to be done in order to calculate the time interpolants and the coupled terms.[30]

We use Equation (34) for the determination of the cell local maximum admissible time step $\Delta\tilde{t}_i$ for a cell $\mathcal{K}_i$ on the standard or agglomerated grid ($\mathcal{K}_i \in \mathscr{K}_h \quad \text{or} \quad \mathcal{K}_i \in \mathscr{K}_h^{agg}$). In the next step, we use the K-means algorithm[54] to cluster the cells into sets $\mathscr{M} = \{\mathcal{M}_1, \dots, \mathcal{M}_M\}$ of similar time step size. These clusters then fulfill the property

$$\underbrace{\min_{\mathcal{K}_i \in \mathcal{M}_1} \Delta\tilde{t}_i}_{:=\Delta t_1} > \min_{\mathcal{K}_i \in \mathcal{M}_2} \Delta\tilde{t}_i > \cdots > \min_{\mathcal{K}_i \in \mathcal{M}_M} \Delta\tilde{t}_i, \tag{40}$$

where Δt_1 is the maximum admissible time step size of the slowest cluster $\mathcal{M}_1$ (with the largest time step size). We also assume that the time scales Δt_m between the different clusters only differ by integer factors f_m

$$\Delta t_m = \frac{\Delta t_1}{f_m}, \qquad \forall m = 2, \dots, M, \tag{41}$$

with

$$f_m = \text{ceil}\left[\frac{\Delta t_1}{\min\limits_{\mathcal{K}_i \in \mathcal{M}_m} \Delta\tilde{t}_i}\right], \qquad f_m \in \mathbb{N}. \tag{42}$$

Equation (41) results in the beneficial fact that all clusters are synchronized at synchronization levels depending on the integer factor of the smallest cluster f_M, ie,

$$t^{n+1} = t^n + \Delta t_1, \qquad n \in \mathbb{N}. \tag{43}$$

At the initial synchronization level $t^0 = 0$, Adams-Bashforth methods for an explicit order of $Q \geq 2$ are not self-starting because no time history is yet available. For the start-up phase, we therefore use a global explicit Runge-Kutta scheme of one order higher than the Adams-Bashforth order in order to guarantee that the temporal error is dominated by the LTS scheme.[30] When the necessary time history is available, we switch back to the Adams-Bashforth scheme.

For a global time stepping method, the cell with the smallest time step size dictates the global time step size. For large differences in the local time step sizes, the computational costs grow enormously. Obviously, this is the case for an artificial viscosity–based shock-capturing approach in combination with a cut cell IBM. The reason for that is that the time step size in nonagglomerated cut cells containing a shock can differ by up to two orders of magnitude from untouched cells in the bulk flow. This motivates the application of an LTS scheme where the clustering is based on the cell local time step sizes.

5.3 | Adaptive reclustering

In unsteady supersonic flows, shocks and other discontinuous phenomena propagate through the computational domain. Accordingly, in numerics, the troubled grid cells with active artificial viscosity and small time step sizes also change in time. This makes an adaption of the clustering necessary. A static clustering would miss the basic idea of LTS, namely, a reasonable update of the cells according to their local time step sizes. Therefore, we introduce a *reclustering interval* $I_{\text{LTS}} \in \mathbb{N}$ where we rebuild the clustering and copy the time history of all cells when using an Adams-Bashforth scheme of higher order. The history contains cell local information about the update times, the change rates (the weighted sum in Equation (A3)), and the DG coordinates.

We outline the procedure for the adaptive LTS algorithm for two time intervals from synchronization levels $t^n \rightarrow t^{n+1}$ and $t^{n+1} \rightarrow t^{n+2}$ in Figure 4. We set $I_{\text{LTS}} = 1$ and assume a one-dimensional grid with four equidistant cells for the sake of simplicity. Note that, in the *x*-*t*-diagram, the height of the cells refers to the time step size in the specific cluster according to Equations (41) and (42). Cells with a small height can be interpreted as cells containing a shock (and, thus, artificial viscosity) and/or being nonagglomerated cut cells.

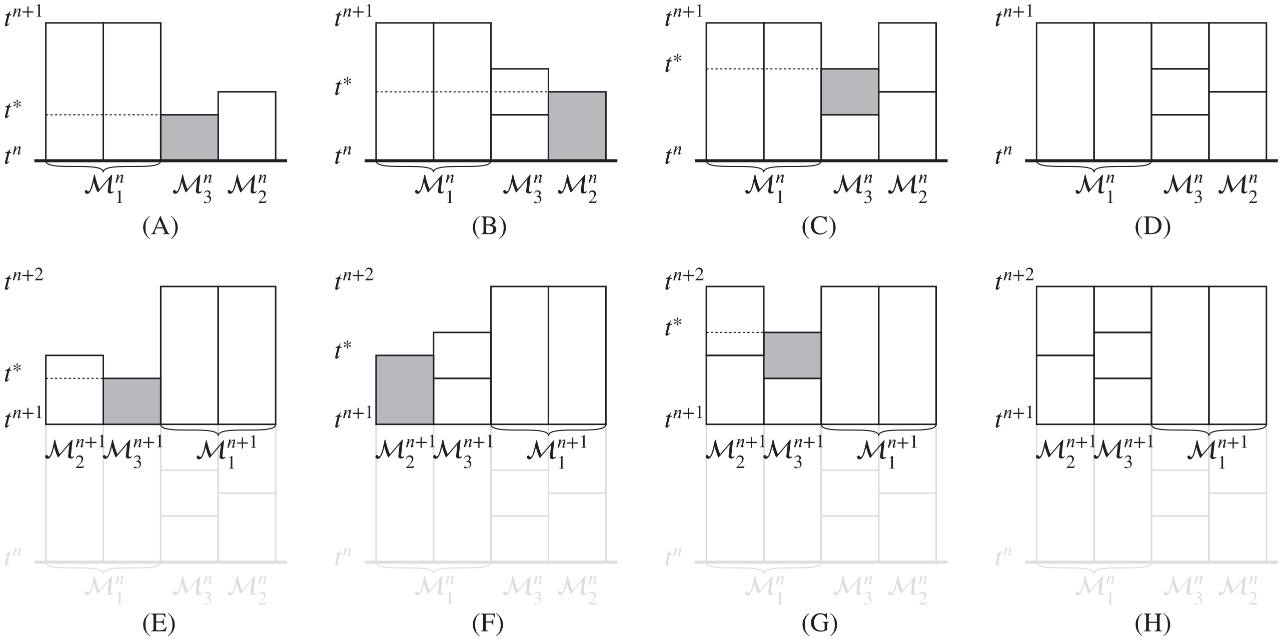


FIGURE 4 Illustration of the adaptive local time stepping procedure as *x*-*t*-diagram. We consider two time intervals $[t^n, t^{n+1}]$ (first row) and $[t^{n+1}, t^{n+2}]$ (second row) and denote the clustering with a set of cell clusters $\mathscr{M} = \{\mathcal{M}_1, \ldots, \mathcal{M}_M\}$. (A)-(C) and (E)-(G) Filled gray cells are at intermediate time t^* and will be updated next. (D) and (H) All cells are at a synchronization level. Additionally, (E) a reclustering is performed based on the current time step restrictions

An LTS method relies on an evolve condition that determines if a cell can be evolved in time. We use the evolve condition as stated by Winters and Kopriva.[30] They define an intermediate time t^* between two synchronization levels t^n and t^{n+1}, ie, $t^* \in [t^n, t^{n+1}]$: *"If the local time on cell $\mathcal{K}_i$ is equal to t^*, then the cell is ready to evolve one local time step."*

At the end of this section, we present a pseudocode for a generic implementation between two synchronization levels $t^n \rightarrow t^{n+1}$ in Algorithm 1.

Algorithm 1 Pseudocode of the adaptive LTS algorithm for the integration between two synchronization levels $t^n \rightarrow t^{n+1}$

```
procedure DoALTSStep( )
    if startUpPhase then
        UseRungeKuttaTimeStepper()
        CreateHistories()                    ▷ Cell local update times, change rates, DG coordinates
    elseif isReclusteringInterval and reclusteringIsNecessary
        CreateNewClustering()
        CopyHistories()
    end if

    UpdateClusterTimeStepSizes()             ▷ This is done in every time interval (cf Equation (41))
    DoAdamsBashforthTimeIntegration()        ▷ All clusters are evolved in time (cf Appendix A.1)
    UpdateHistories()                        ▷ Histories only needed for higher-order time stepping
end procedure
```

6 | NUMERICAL RESULTS

In this section, we present numerical results for several (pseudo-)two-dimensional test cases for high Mach number flows using the supplemented Euler equations (5). We discretize the convective fluxes using the HLLC formulation as presented by Toro[55] and the artificial diffusive fluxes by the SIP formulation as presented by Hartmann and Houston.[56] Their formulation contains two parameters, ie, the cell local characteristic length scale h_i and the global penalty scaling factor C_{IP}. We choose h_i as stated in Equation (35) in combination with the applied cell agglomeration strategy (cf Section 3) and $C_{\text{IP}} = 5.0$ to ensure coercivity.[37,40] For time integration, we use standard explicit time stepping schemes and the adaptive LTS algorithm presented in Section 5.

As we focus on the robustness, stability, and applicability of the IBM in combination with a basic shock-capturing approach and an adaptive LTS scheme, we do not compare our results to the works of other groups in terms of solution accuracy. The feasibility of the underlying artificial viscosity–based shock-capturing approach[5,19] or similar variants has been successfully demonstrated in the past years by various other works.[7,20,57,58] For more details, the interested readers are referred to, eg, the work of Lv et al[20] who presented an extensive study on the performance and accuracy of different shock sensors including the basic version of Persson and Peraire[5] in combination with artificial viscosity.

We selected the numerical test cases according to their increasing complexity and methodology.

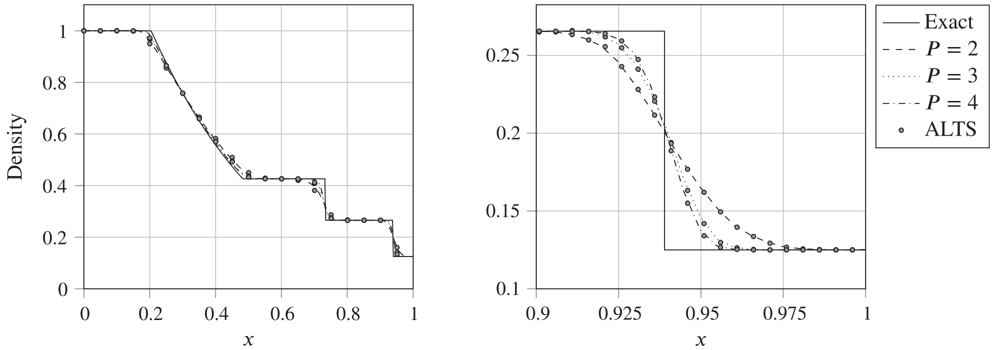
(i) *Shock-capturing + adaptive LTS*: We investigate the Sod shock tube problem in Section 6.1 and a shock-vortex interaction in Section 6.2. These first test cases verify the successful combination of the present shock-capturing approach and the adaptive LTS scheme. Additionally, the computational costs are analyzed in terms of savings through the adaptive LTS scheme.

(ii) *IBM + shock-capturing + adaptive LTS*: The results of the novel extension to a cut cell IBM featuring a cell agglomeration strategy is presented for a Sod shock tube variant with an immersed boundary in Section 6.3 and the double Mach reflection (DMR) test case in Section 6.4. The results are analyzed in the same manner as for part (i).

6.1 | Sod shock tube

This classical test case[55] is often considered for the evaluation of a numerical scheme for its ability to capture simple, non-interacting waves such as rarefaction waves, contact discontinuities, and shock waves. The initial conditions for density,

FIGURE 5 Sod shock tube problem. *P*-refinement study using the artificial viscosity–based shock-capturing mechanism on a grid with 50 × 50 cells and a global Adams-Bashforth scheme of order $Q = 3$ at an end time of $t_{\text{end}} = 0.25$. Further, comparison calculations were conducted using the presented adaptive local time stepping (ALTS) scheme of the same temporal order, an initial number of clusters of $C_{\text{init}} = 3$, and a reclustering interval of $I_{\text{LTS}} = 1$



(A) Entire domain

(B) Zoom-in around the shock wave (right bottom part of (A))

velocity, and pressure are given as

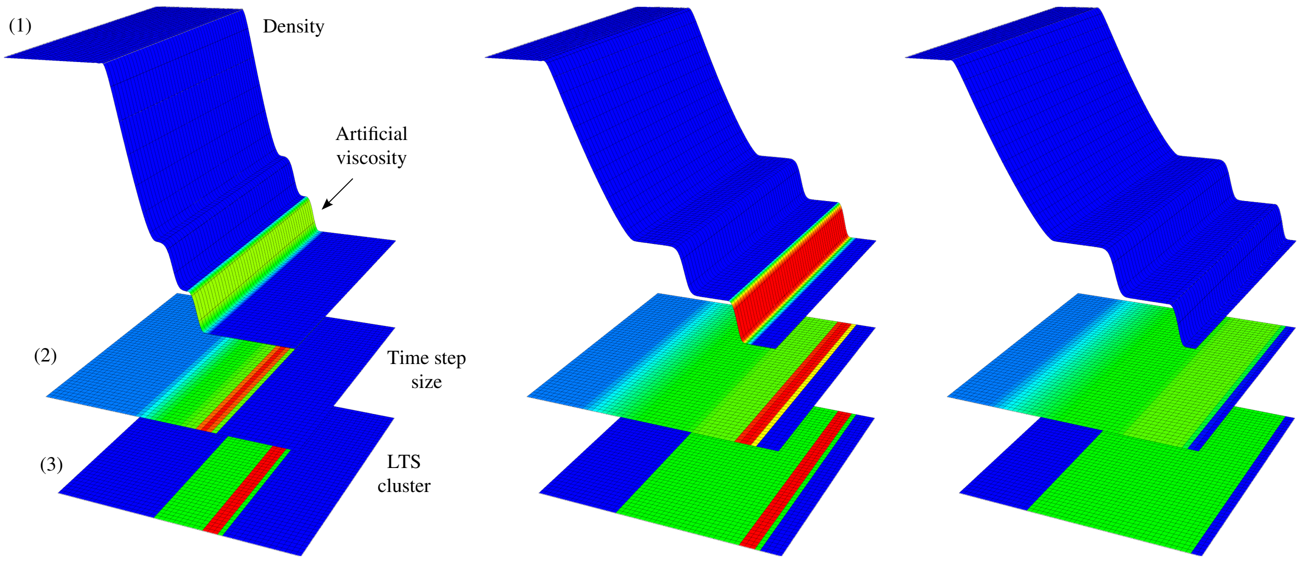
$$(\rho, u_1, u_2, p)^T = \begin{cases} (1, 0, 0, 1)^T, & \text{for} \quad x \leq 0.5, \\ (0.125, 0, 0, 0.1)^T & \text{for} \quad x > 0.5, \end{cases} \tag{44}$$

on the (pseudo-)two-dimensional domain $x \in [0, 1] \times [0, 1]$ with a coarse grid consisting of 50 × 50 cells. The initial discontinuity is located at $x = 0.5$. The simulation end time is $t_{\text{end}} = 0.25$. Dirichlet boundary conditions are imposed on the left and right part of the domain while adiabatic slip wall boundary conditions are used on the top and bottom. We compare our numerical results with the exact solution of the one-dimensional Riemann problem.[55] The shock-capturing parameters were determined by numerical testing. We set $S_0 = 1.0 \cdot 10^{-3}$, $\kappa = 0.5$, and $\varepsilon_0 = 1.0$ (cf Section 4).

Figure 5 shows the density profiles for different polynomial degrees. Results are taken at a horizontal line at $y = 0.5$. For higher polynomial degrees, the solution converges to the exact solution capturing the position and the shape of the different types of waves. Note that the shock-capturing parameters were specified once for a single polynomial degree and were not changed afterwards. This verifies our proposed scaling of the artificial viscosity parameter (28). The *P*-refinement study was conducted with a global Adams-Bashforth time stepping scheme of explicit order $Q = 3$. Moreover, we show the results for the adaptive LTS algorithm of the same order, where we prescribe the initial number of clusters ($C_{\text{init}} = 3$) and set the reclustering interval to $I_{\text{LTS}} = 1$. The adaptive LTS calculations were conducted for the respective polynomial degrees matching the version of the global time stepping scheme without a loss of solution accuracy.

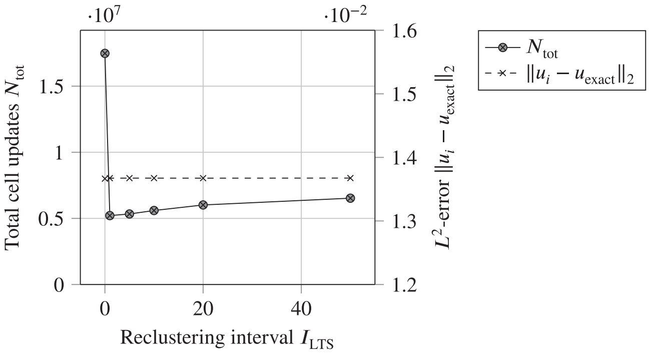
An illustration of the temporal evolution is depicted in Figure 6. The top plane shows the density profile colored with artificial viscosity, the middle plane shows the local time step size, and the bottom plane shows the adaptive LTS clustering. The snapshots are taken at three different points in time ($t \approx 0.09$, $t \approx 0.21$, $t = 0.25$). It is apparent that the artificial viscosity is localized around the shock wave as it is the strongest discontinuous feature, as seen in Figures 6A and 6B. Thus, the time step sizes are very small in the vicinity of the shock (cf Equation (33)). The green area in Figure 6B represents cells whose time step sizes are dictated by the propagation velocity of the different wave types only. The LTS clustering finally consists of three clusters: the untouched area (blue), the segment that is dominated by the convective time step restriction only (green), and the segment where artificial viscosity is activated (red). We choose Figure 6C in order to show that there can be several time steps where artificial viscosity is not active. The solution is sufficiently smooth at these time steps. This yields a more homogeneous distribution of the local time step sizes, which is why the number of clusters automatically reduces to two.

We choose the total number of cell updates N_{tot} as a measure to compare the speedup between a global Adams-Bashforth scheme ($I_{\text{LTS}} = 0$) and our adaptive LTS approach. This measure is independent of the actual implementation (quality) and is therefore chosen as a generic benchmark. We investigate the influence of I_{LTS} on N_{tot} and on the L^2-error $\|u_i - u_{\text{exact}}\|_2$ with respect to the exact solution in Figure 7. A saving of 70.2% in terms of total cell updates compared to the global Adams-Bashforth reference can be observed for $I_{\text{LTS}} = 1$ while maintaining the solution accuracy. Furthermore, the solution accuracy does not change for all depicted values of I_{LTS}. It is reasonable that the number of cell updates increases for larger reclustering intervals. This is due to the clustering remaining in a static state for a longer period of time and not



(A) Time $t \approx 0.09$ (B) Time $t \approx 0.21$ (C) Time $t = 0.25$

FIGURE 6 Sod shock tube problem. Illustration of the adaptive local time stepping (LTS) algorithm using three elevated planes ($P = 3$, 50×50 cells, $Q = 3$, $C_{\text{init}} = 3$, $I_{\text{LTS}} = 1$): (1) density profile colored with artificial viscosity, (2) local time step size, blue/red corresponds to large/small time step sizes, and (3) cell clustering, blue/red corresponds to small/large values [Colour figure can be viewed at wileyonlinelibrary.com]



I_{LTS}	N_{tot}	Speedup
0	17480000	Ref.
1	5211040	70.2 %
5	5333580	69.5 %
10	5596180	67.0 %
20	6015460	65.6 %
50	6528050	62.7 %

FIGURE 7 Sod shock tube problem. Comparison of the total number of cell updates N_{tot} between a global Adams-Bashforth scheme chosen as reference ($I_{\text{LTS}} = 0$) and the adaptive local time stepping (LTS) algorithm for several reclustering intervals I_{LTS}. The L^2-error $\|u_i - u_{\text{exact}}\|_2$ does not change for larger reclustering intervals whereas the total number of cell updates does. (Other parameters are $P = 3$, 50×50 cells, $t_{\text{end}} = 0.25$, $Q = 3$, $C_{\text{init}} = 3$)

considering the physical changes in the flow. This forces the time step size of the largest cluster Δt_1 to become smaller than necessary. In practice, the reclustering interval has to be chosen as a compromise between the total number of cell updates and the computational overhead that is produced by the reclustering procedure itself.

In order to give the readers also a sense for the speedup from a practical point of view, we consider the setting shown in Table 1. Again, we want to stress that this study is strongly influenced by the implementation quality and the underlying setting. Thus, it is not generalizable. We compare the LTS simulation with simulations using a global time stepping scheme. We did two runs with a standard Adams-Bashforth scheme ($Q = 1$) and the LTS code with only one cluster. Both schemes are equal to the explicit Euler scheme in this case. We forced the reclustering in every time step ($I_{\text{LTS}} = 1$) during the LTS run in order to have a "worst case scenario" in terms of speedup. Moreover, the computational overhead of the reclustering procedure was measured with around 8% within the LTS run. It can be stated that the maximum speedup of the adaptive LTS scheme will be between 46.2% and 72.6% for this specific setting. When using other common explicit

TABLE 1 Sod shock tube problem. Run time cost breakdown and comparison between the adaptive local time stepping (LTS) scheme and a global Adams-Bashforth time stepping scheme (GTS). The speedup is listed in terms of run time and the total number of cell updates N_{tot}. (Other parameters are $P = 2$, 50×50 cells, $t_{\mathrm{end}} = 5 \cdot 10^{-3}$, $Q = 3$, $C_{\mathrm{init}} = 3$, $I_{\mathrm{LTS}} = 1$ (forced), single core)

Time stepping scheme	Run time in s	Fraction (run time)	N_{tot}	Fraction (N_{tot})
GTS	10.4 s	100%	212500	100%
GTS with LTS code	12.1 s	−16.3%	212500	100%
LTS	5.6 s	46.2%	58200	72.6%

time integrations schemes like Runge-Kutta schemata, the results will be different. We abstain from going into the details because this issue is beyond the scope of this work.

6.2 | Shock-vortex interaction

Another two-dimensional test case for compressible flows is the interaction of a vortex with a stationary shock wave. A detailed description of the test case can be found, eg, in the works of Dumbser et al[11] and Rault et al[59] and the HiOCFD5 workshop[†]. This test case consists of a complex flow interaction featuring smooth weak and strong discontinuous phenomena and is a challenging test case for higher-order numerical schemes.

The computational domain is set to $\Omega = [0,2] \times [0,1]$. A stationary normal shock wave with a Mach number of M_s is located at $x = 0.5$. A typical range for the shock Mach number is $M_s \in [1.1, 2]$. The values for density, velocity, and pressure in the pre-shock region on the left are given as

$$(\rho_l, u_{1,l}, u_{2,l}, p_l)^T = \left(1, \quad \sqrt{\gamma \frac{p_l}{\rho_l}} \quad M_s, 0, 1\right)^T, \tag{45}$$

with a heat capacity ratio of $\gamma = 1.4$. The quantities in the post-shock region on the right can be calculated using the normal shock wave relations[60(p86-90)]

$$\rho_r = \frac{(\gamma+1)M_s^2}{2+(\gamma-1)M_s^2} \quad \rho_l, \tag{46}$$

$$u_{1,r} = \frac{2+(\gamma-1)M_s^2}{(\gamma+1)M_s^2} \quad u_l, \tag{47}$$

$$u_{2,r} = 0, \tag{48}$$

$$p_r = \left(1 + \frac{2\gamma}{\gamma+1}\left(M_s^2 - 1\right)\right) p_l. \tag{49}$$

Additionally, a moving vortex with an angular velocity profile of

$$v_\Phi(r) = \begin{cases} v_m \ \frac{r}{a} & \text{for } \ r \le a, \\ v_m \ \frac{a}{a^2-b^2}\left(r - \frac{b^2}{r}\right) & \text{for } \ a \le r \le b, \\ 0 & \text{otherwise}, \end{cases} \tag{50}$$

with $r = \sqrt{(x-x_c)^2 + (y-y_c)^2}$, is superimposed onto the initial conditions. The initial center of the vortex is located at $(x_0, y_0) = (0.25, 0.5)$. The initial temperature distribution of the vortex can be obtained by solving the following ODE

$$\frac{\mathrm{d}T}{\mathrm{d}r} = \frac{\gamma-1}{R\gamma}\frac{v_\Phi^2(r)}{r}. \tag{51}$$

[†] https://how5.cenaero.be/content/ci2-%E2%80%93-inviscid-strong-vortex-shock-wave-interaction (02/13/2019).

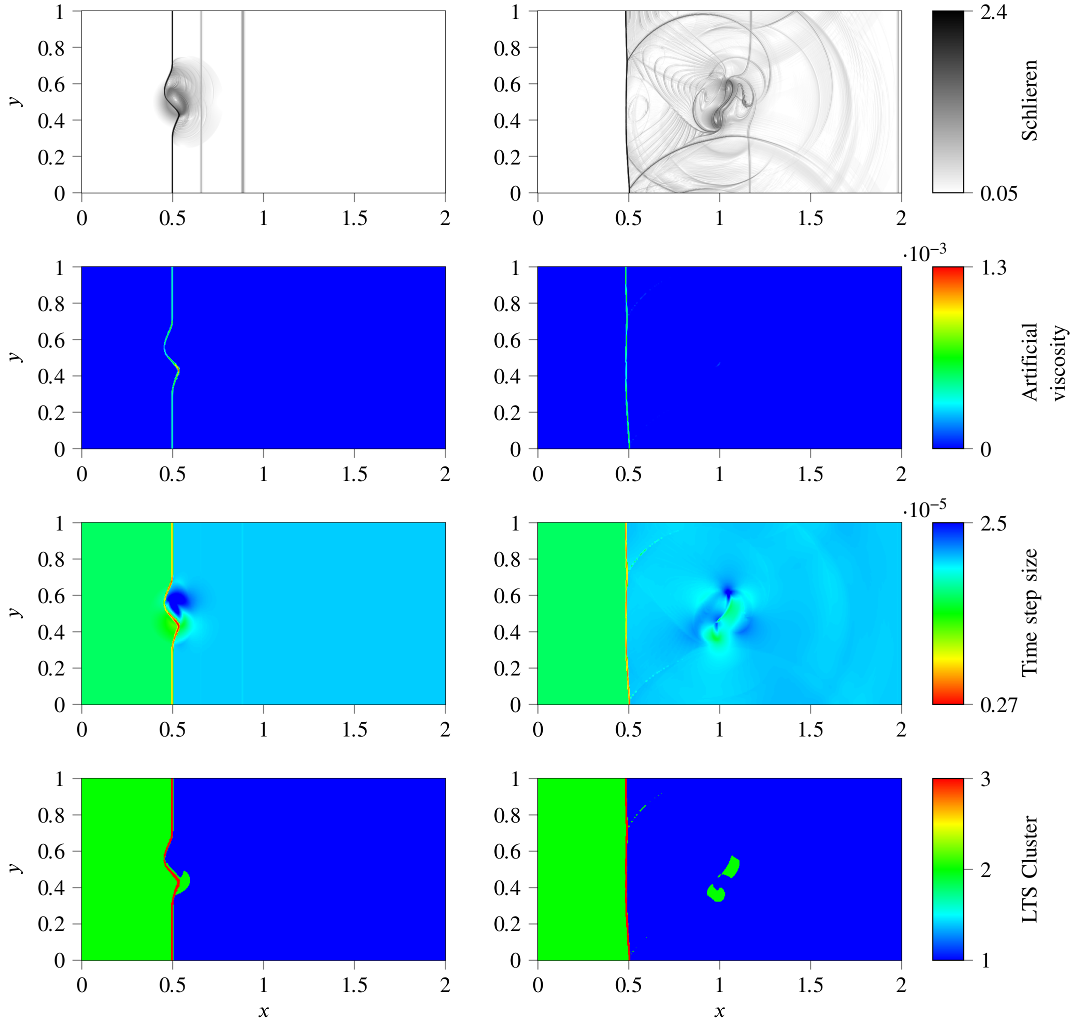
This is done by performing a simple integration with respect to r on Equation (51)

$$T(r) = \frac{\gamma-1}{R\gamma}\begin{cases} T(a) - \frac{\gamma-1}{R\gamma}\frac{v_m^2}{2}\left(1-\frac{r^2}{a^2}\right) & \text{for } r\le a \\ T(b) - \frac{\gamma-1}{R\gamma}\left(\frac{v_m a}{a^2-b^2}\right)^2\left(-\frac{r^2}{2}+2b^2\ln(r)+\frac{b^4}{2r^2}-2b^2\ln(b)\right) & \text{for } a\le r\le b \\ 0 & \text{otherwise}, \end{cases} \tag{52}$$

where the integration constants are obtained by $T(b) = T_l$ for $a \le r \le b$ and $T(a) = T(a)_{a\le r\le b}$ for $r \le a$. In a next step, the density and pressure distributions are calculated using the expressions

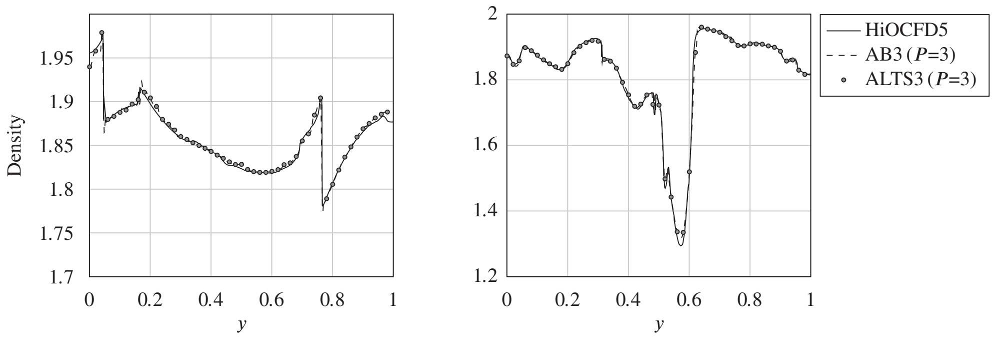
$$\rho = \rho_0\left(\frac{T}{T_0}\right)^{\frac{1}{\gamma-1}}, \tag{53}$$

$$p = p_0\left(\frac{T}{T_0}\right)^{\frac{\gamma}{\gamma-1}}. \tag{54}$$



(A) Time $t = 0.16$ (B) Time $t = 0.7$

FIGURE 8 Shock-vortex interaction. The results have been calculated for a configuration with a vortex Mach number of $M_V = 0.9$ and a shock Mach number of $M_S = 1.5$. We show the Schlieren distribution calculated by $\ln(1 + \|\nabla\rho\|)/\ln(10)$ (first row), the artificial viscosity distribution (second row), the time step size (third row), and the local time stepping (LTS) clustering (last row) using the presented adaptive LTS scheme for the points in time $t = 0.16$ (left column) and $t = 0.7$ (right column). (Other parameters are $Q = 3$, $I_{\text{LTS}} = 1$, $C_{\text{init}} = 3$, $P = 3$, 600×300 cells) [Colour figure can be viewed at wileyonlinelibrary.com]



(A) Vertical line at $x = 0.52 + \varepsilon$ (B) Vertical line at $x = 1.05 + \varepsilon$

FIGURE 9 Shock-vortex interaction. We show the density distribution along two vertical lines for a polynomial degree of $P = 3$ at $t = 0.7$. Line (A) is placed directly behind the shock in the post-shock region. Line (B) goes through the vortex structure. Calculations were conducted using the adaptive local time stepping scheme and their global version for a comparison. (Other parameters are $Q = 3$, $I_{\text{LTS}} = 1$, $C_{\text{init}} = 3$, 600×300 cells)

The reference values are based on the undisturbed flow values upstream the shock wave, yielding $p_0 = R\rho_0 T_0$, where the gas constant R is set to 1 for the sake of simplicity. Typically, the strength of the vortex is varied by its Mach number $M_V = v_m/c_0$, where $c_0 = \sqrt{\gamma p_0/\rho_0}$ represents the speed of sound upstream on the left side of the shock. We set the parameters to $\gamma = 1.4$, $a = 0.075$, $b = 0.175$, $M_S = 1.5$, $M_V = 0.9$. $\rho_0 = \rho_l = 1$ and $p_0 = p_l = 1$ are given by the undisturbed upstream conditions. A typical range for the simulation end time is $t_{\text{end}} = [0.7, 0.8]$.

The results are reported in Figure 8. We show the pressure and artificial viscosity distribution for two different times $t = 0.2$ and $t = 0.7$ (first and second columns). Additionally, we present the time step size (third row) and the respective LTS clustering (last row) for a simulation with a polynomial degree of $P = 3$. Further simulation parameters are listed in the caption of Figure 8. At $t = 0.2$, the shock is distorted by the moving vortex. The distortion of the shocks strongly depends on the strength of the vortex compared to the shock.[61] Furthermore, the interaction between the vortex and the shock produces sound waves that propagate downstream. The initial configuration creates a right running shock wave that is similar to the numerical artifacts encountered in the DMR test case.[62] The vortex breaks up into two parts with the chosen parameters. The vast majority of artificial viscosity is added around the shock, which results in an almost static LTS clustering except for the outliers caused by the generated sound waves.

For a quantitative assessment of the solution accuracy, we show the density distribution along the vertical lines $x = 0.52 + \varepsilon$ and $x = 1.05 + \varepsilon$ in Figure 9 ($\varepsilon = 1.0 \cdot 10^{-4}$). The reference values were extracted from plots of the HiOCFD5 workshop. Our results are in very good accordance with the reference. Moreover, a reference simulation was conducted for a global Adams-Bashforth scheme ($Q = 3$), which needed a total number of $N_{\text{tot}} \approx 31861 \cdot 10^6$ cell updates. The adaptive LTS simulation only performed $N_{\text{tot}} \approx 9821 \cdot 10^6$ updates, which results in an approximate saving of 69%.

6.3 | Sod shock tube for immersed boundaries

We reuse the Sod shock tube problem (cf Section 6.1) in order to show the viability of our novel combination between

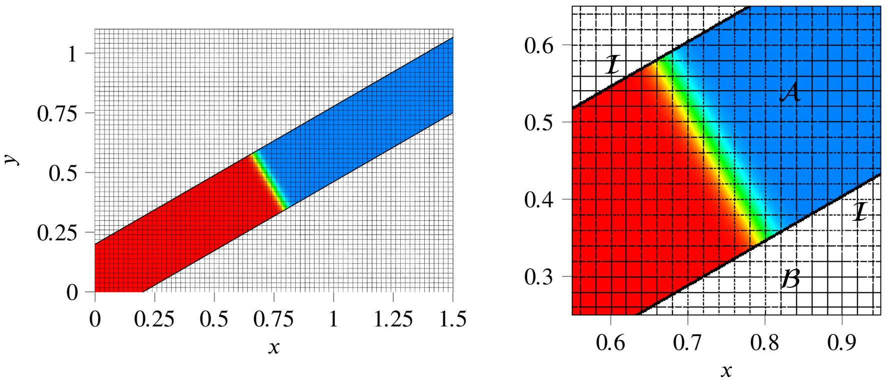
(i) a cut cell IBM that uses a cell agglomeration procedure (cf Section 3),
(ii) an extended shock-capturing scheme (cf Section 4), and
(iii) an adaptive LTS scheme (cf Section 5).

We set the level set function to

$$\varphi(\vec{x}) = -(x_2 - \tan(\pi/6)(x_1 - 0.2))(x_2 - (\tan(\pi/6)x_1 + 0.2)). \tag{55}$$

In combination with a grid consisting of 75×55 cells and a computational domain of $[0.0, 1.5] \times [0.0, 1.1]$, the interface $\mathcal{I}$ cuts the grid as depicted in Figure 10. We choose an agglomeration threshold of $\delta = 0.3$, such that small cut cells are agglomerated to their largest neighbors in the fluid part of the domain due to the definition of the level set function (55). The shock sensor values are calculated using Equation (22).

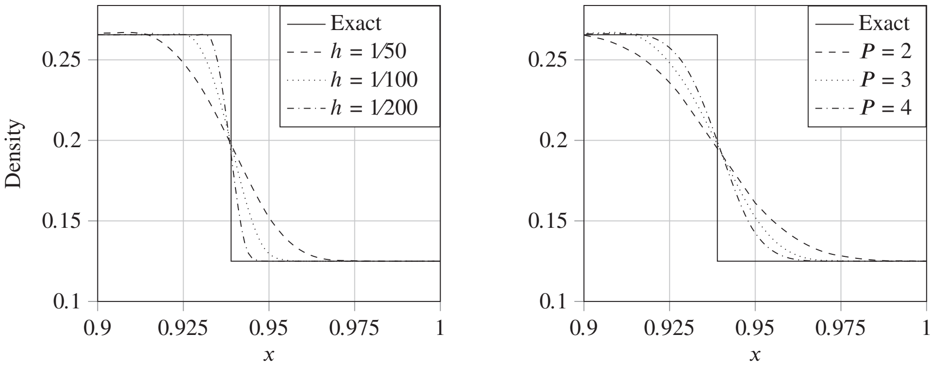
In Section 6.3.1, we briefly evaluate the performance of the sensor on a agglomerated cut cell grid in order to verify its extension to the IBM. Afterwards, we present the speedup obtained by the adaptive LTS scheme in Section 6.3.2 as done in the previous test cases.



(A) Entire domain

(B) Zoom-in

FIGURE 10 Sod shock tube for an immersed boundary method. A Cartesian grid with 75 × 55 cells is used, where the fluid part of the computational domain is cut out by the zero isocontour of the level set function. The initial density distribution is shown, cf Equation (44). The cut cells have a very small fluid fraction and, thus, are agglomerated to their largest (fluid) neighbor cell. The fluid part of the domain is denoted with $\mathcal{A}$, the interface with $\mathcal{I}$, and the void part with $\mathcal{B}$ [Colour figure can be viewed at wileyonlinelibrary.com]



(A) h-refinement study for $P = 3$

(B) P-refinement study for 75 × 55 cells

FIGURE 11 Sod shock tube problem for an IBM. h- and P-refinement study verifying the applicability of the presented modal-decay sensor (25) on an agglomerated cut cell grid. The $1/P^4$-scaling of the critical sensor value S_0 that activates artificial viscosity and the h/P-scaling of the artificial viscosity determination formula (28) work as expected. The plots show the density distributions for $t_{\text{end}} = 0.25$. Moreover, the adaptive local time stepping scheme is applied ($Q = 3$, $I_{\text{LTS}} = 1$, $C_{\text{init}} = 2$)

6.3.1 | Sensor performance on cut cells

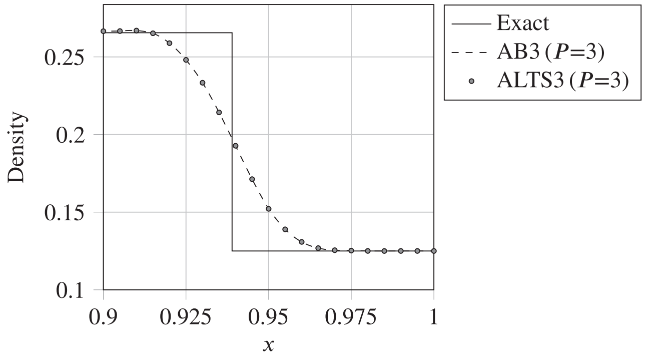
In order to verify the main scaling properties of the presented sensor and also its applicability on an agglomerated cut cell grid (cf Section 4.1), we performed an h- and P-refinement study. Figure 11 shows the density distribution at a zoom-in around the shock wave for an end time of $t_{\text{end}} = 0.25$. In both refinement studies, the density curves approach the exact solution for an increasing h/P-resolution. When keeping h fixed and increasing P even more, the shock can be resolved in one single cell (subcell resolution).

These investigations confirm that the proposed $1/P^4$-scaling of the critical sensor value S_0 that activates artificial viscosity and the artificial viscosity determination formula (28) are also valid in the IBM.

6.3.2 | Speedup on cut cells

We compare again the accuracy of our adaptive LTS scheme to a reference simulation that uses a global time stepping scheme and also to the exact solution[55] in Figure 12. The results have been obtained for the following setting: $P = 3$, $Q = 3$, $I_{\text{LTS}} = 1$, and $C_{\text{init}} = 2$. We reuse the shock-capturing parameters stated in Section 6.1 without any modifications, which clearly indicates the robustness of the proposed approaches in combination with the IBM. The adaptive LTS scheme performs well, also in the context of an IBM with active cell agglomeration, resulting in a saving of ≈ 63.0% in terms of total cell updates for different reclustering intervals. In contrast to the results shown in Figure 7, the speedup remains almost the same. This fact strongly depends on the LTS clustering that is dominated by the small nonagglomerated cut cells in IBM simulations. Here, the clustering only consists of two clusters, whereas it consists of three clusters in the boundary-fitted case. The number of substeps between the different LTS clusters can vary up to two orders of magnitude in IBM simulations. We did not experience any problems with caused by this issue during our numerical experiments. The

FIGURE 12 Sod shock tube problem for an immersed boundary method. The grid is defined as shown in Figure 10. We compare the adaptive local time stepping scheme ($Q = 3$, $I_{\text{LTS}} = 1$, $C_{\text{init}} = 2$) with a global time stepping scheme as reference ($I_{\text{LTS}} = 0$) for a polynomial degree of $P = 3$. Moreover, we list the speedup in terms of the total number of cell updates N_{tot} for several reclustering intervals I_{LTS}



I_{LTS}	N_{tot}	Speedup
0	309109936	Ref.
1	114171407	63.1 %
5	114426437	63.0 %
10	114474955	63.0 %
20	114483524	63.0 %
50	114520668	63.0 %

TABLE 2 Sod shock tube problem for an IBM. We list the speedup in terms of the total number of cell updates N_{tot} for a reclustering interval of $I_{\text{LTS}} = 1$ for several polynomial degrees P. We compare the adaptive LTS scheme ($Q = 3$, $C_{\text{init}} = 2$) with a global time stepping scheme of the same temporal order for every single polynomial degree

P	I_{LTS}	N_{tot}	Ref.	Speedup
2	1	63935248	171615808	62.7%
3	1	114520668	309109936	63.0%
4	1	230659014	640100976	64.0%

underlying K-means clustering algorithm[54] could be tuned in a way that it produces a clustering with a different number of clusters which could be more suitable for a higher speedup. We abstained from doing additional studies because we focus on the applicability of the method itself.

Moreover, we list the speedup for the P-refinement study that has been reported in Section 6.3.1 in Table 2. Again, the speedup is calculated with respect to a reference simulation with a global Adams-Bashforth scheme of the same temporal order ($Q = 3$). The results indicate that higher savings in terms of the total number of cell updates can be obtained for higher polynomial degrees when using the adaptive LTS scheme. The reason for this is the diffusive time step restriction (33) that scales with $\mathcal{O}(1/P^2)$ while keeping h fixed. This effect strongly depends on the topology of the test case and, thus, on the LTS clustering. This effect is only slightly present for this specific test case.

6.4 | Double Mach reflection

As a final test case, we investigate the complex two-dimensional DMR that was first proposed by Woodward and Colella.[62] This self-similar flow configuration contains several challenging features like two moving triple points and complicated shock reflections and interactions. The setting is based on a shock wave with a Mach number of $M_S = 10$ hitting a reflecting wall, which is inclined at an angle of 30°. Experimentally, the DMR can be set up by creating a supersonic flow that hits a ramp or a wedge. We consider an ideal gas with a heat capacity ratio of $\gamma = 1.4$. The right side of the incident shock is set to be at rest, while on the left side, the corresponding post-shock conditions are prescribed satisfying the normal shock wave relations[60(p86-90)]

$$(\rho, u_1, u_2, p)^T = \begin{cases} (8, 8.25, 0, 116.5)^T & \text{for} \quad x < 0.16, \\ (1, 0, 0, 1)^T & \text{for} \quad x \geq 0.16. \end{cases} \tag{56}$$

The computational domain is chosen as $\Omega = [0, 3] \times [0, 2]$ together with a grid consisting of 300×200 cells for the IBM case. We assign supersonic boundary conditions to the inlet, the bottom boundary ($x < 0.16$), the top boundary, and the outlet. Additionally, we prescribe the exact movement of the incident shock wave at the top boundary. For $x \geq 0.16$, we apply an adiabatic slip wall boundary condition along the ramp. If the position where the initial conditions change ($x = 0.16$) is not exactly located on a cell boundary, we apply a smoothing over several cells in order not to introduce additional oscillations by the initial projection. We choose an agglomeration threshold of $\delta = 0.3$ in order to agglomerate small cut cells to their nearest fluid neighbor cells. The sensor value that is responsible for activating artificial viscosity is set to $S_0 = 1.0 \cdot 10^{-4}$. Severe oscillations can occur due to the abrupt, unphysical change of the boundary conditions at $x = 0.16$. While these

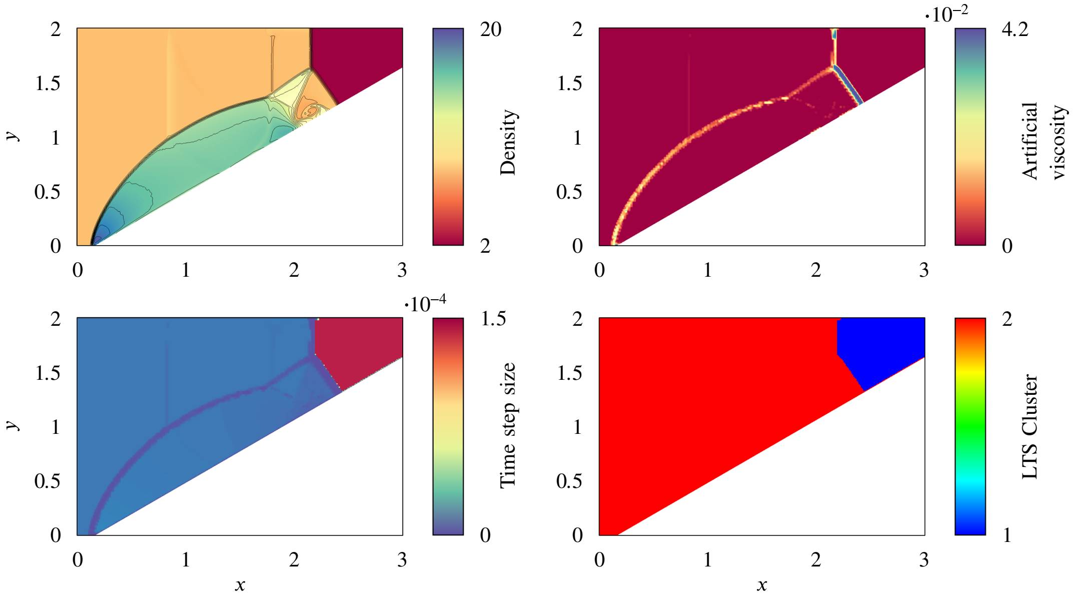


FIGURE 13 Double Mach reflection using an immersed boundary method (IBM). The results were obtained for a polynomial degree of $P = 3$ using the presented IBM with a cell agglomeration threshold of $\delta = 0.3$. We show the density distribution with 20 equally distributed contour lines, the artificial viscosity distribution, the time step size, and the local time stepping (LTS) clustering using the adaptive LTS scheme for $t = 0.2$. (Other parameters are $Q = 3$, $I_{\text{LTS}} = 1$, $C_{\text{init}} = 2$, 300 × 200 cells) [Colour figure can be viewed at wileyonlinelibrary.com]

can be damped via an excessive amount of artificial viscosity, we choose to fix the characteristic velocity ($\lambda_{\max} = 25$) in order to guarantee a sufficient smoothing during the development phase of the self-similar structure.

The results for the IBM case are depicted in Figure 13. The density and artificial viscosity distribution is being shown again with the corresponding time stepping information for a polynomial degree of $P = 3$. We choose another color scale for some plots for a better visibility. Please note that the thin vertical stripes, eg, visible at $x \approx 0.9$ and $x \approx 1.8$, are numerical artifacts caused by the initial condition and the top boundary condition, respectively.[62] The top boundary condition is dynamically adapted with the exact shock velocity. Additionally, we apply a smoothing over several cells as done for the initial condition. Most troubled cells are located around the incident shock wave, the reflected shock waves, and the Mach stem that emanates from the first triple point. The most challenging part is the wall jet at the bottom right part and the Kelvin-Helmholtz instabilities along the first slip line. In order to resolve these flow phenomena, adjustments of the underlying shock-capturing scheme are necessary, eg, by a dynamic threshold setting[20] or an appropriate a posteriori limiting,[11] when not willing to increase the resolution to a maximum. We would like to note that we could resolve the instabilities with our basic shock-capturing approach for a finer resolution or a larger end time that has the same effect due to the self-similarity of the flow structure (results are not shown here).

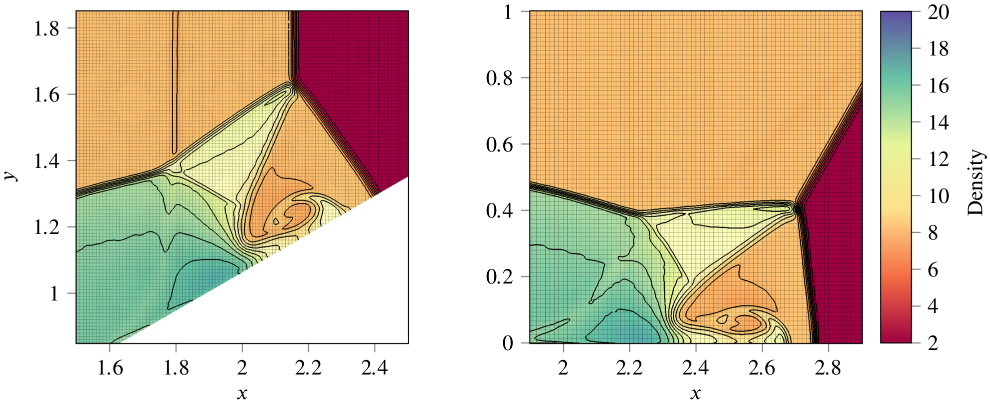


FIGURE 14 Double Mach reflection. Comparison between the immersed boundary method (IBM) simulation (left, cf Figure 13) and a boundary-fitted simulation (right) using a global Adams-Bashforth time stepping scheme of order $Q = 3$ for a polynomial degree of $P = 3$. Right: the domain size is $[0, 4] \times [0, 1]$ with 300 × 75 cells (a zoom-in is plotted). Other parameters are chosen as for the IBM test case [Colour figure can be viewed at wileyonlinelibrary.com]

Especially, in this IBM test case with a very strong incident shock wave, the adaptive LTS scheme together with the presented cell agglomeration strategy shows its potential compared to a global time stepping method. This is due to the fact that the time step sizes in nonagglomerated cut cells with activated artificial viscosity can differ up to two orders of magnitude compared to nontroubled standard cells. Suitable parallelization techniques in combination with the LTS strategy start to play an important role for these large test cases. A detailed assessment of the computational time is therefore not presented for this test case as it is beyond the scope of this work. The reader is referred to the previous sections for a basic analysis of the computational costs.

For a comparison of the solution accuracy, we additionally present the results for a global Adams-Bashforth time stepping scheme on a boundary-fitted mesh in Figure 14. We chose a comparable spatial resolution to the IBM setting. The results are in very good agreement, eg, when considering the position of the triple points or the shape of the wall-near jet. It can be stated that the combination of the IBM, the adaptive LTS method, and the basic shock-capturing approach maintains the accuracy of the numerical solution. See Section 6.3 for a more detailed analysis and a proof of concept of the presented combination of methods.

7 | CONCLUSION

In this work, we have presented a novel efficient combination in the context of a cut cell DG IBM for high Mach number flows consisting of an artificial viscosity–based shock-capturing mechanism and an adaptive explicit LTS scheme. Our approach also makes use of a nonintrusive cell agglomeration strategy that avoids problems with small or ill-shaped cut cells.

Nonagglomerated cut cells with activated artificial viscosity represent a worst-case scenario in terms of the maximum admissible time step size because the time step size of these cells can differ up to two orders of magnitude compared to standard cells. Both, the adaptive LTS scheme and the cell agglomeration strategy, significantly decrease the otherwise immense computational costs while not introducing additional spatial or temporal errors.

The performance of our approach was validated in three common two-dimensional test cases in terms of robustness, stability, and accuracy. For example, we compared our numerical results with the exact solution of the (pseudo-)two-dimensional Sod shock tube problem for several polynomial degrees $P \geq 2$ in a boundary-fitted and an immersed boundary configuration. In both cases, the same error levels were obtained as for a global time stepping scheme while saving up to 60% to 80% in terms of total cell updates depending on the adaptive LTS reclustering interval. Moreover, we presented numerical results for complex two-dimensional test cases, such as for the shock-vortex interaction and the DMR test case, which showed the geometrical flexibility of the presented cut cell IBM.

We applied the basic version of the shock-capturing approach presented by Persson and Peraire[5,19] without major modifications except for the calculation of the actual viscosity used.[7,18] Before that, the artificial viscosity field had been projected on a C^0-continuous field.

Our future work will be focused on the parallel performance by means of implementing an effective dynamic load balancing strategy. For test cases in the context of compressible flows, expensive cells are typically the ones containing shocks. We conducted first numerical experiments[63] where we used space-filling Hilbert curves[64] together with suitable cell cost estimators in order to partition the grid. It is clear that troubled cells with artificial viscosity and also small nonagglomerated cut cells dictate the maximum admissible time step size and therefore the computational costs. These expensive cells typically occur very locally at some parts in the computational domain. Consequently, possible cell cost estimators should consist of multiple balance constraints and could be based on the artificial viscosity values or the LTS clusters. The implementation of the LTS cell update strategy (cf Figure 4) also begins to play an important role as soon as LTS clusters are distributed among several processors.

ACKNOWLEDGEMENTS

The work of M. Geisenhofer and B. Müller is supported by the “Excellence Initiative” of the German Federal and State Governments and the Graduate School of Computational Engineering at Technische Universität Darmstadt. The work of F. Kummer is supported by the German Research Foundation (Deutsche Forschungsgemeinschaft) through the Collaborative Research Center 1194/B06. The work of B. Müller is supported by the German Research Foundation (Deutsche Forschungsgemeinschaft) through research grant WA 2610/2-1. Some of the calculations for this research were conducted on the Lichtenberg high performance computer at Technische Universität Darmstadt. Thanks to P. Cabos, P. Steinberg, and J. Weber for their valuable comments on this work.

ORCID

Markus Geisenhofer https://orcid.org/0000-0002-0632-7237
Florian Kummer https://orcid.org/0000-0002-2827-7576

REFERENCES

1. Sherwin SJ, Karniadakis GE. A new triangular and tetrahedral basis for high-order (hp) finite element methods. *Int J Numer Methods Eng*. 1995;38(22):3775-3802.
2. Warburton T, Lomtev I, Du Y, Sherwin SJ, Karniadakis GE. Galerkin and discontinuous Galerkin spectral/*hp* methods. *Comput Methods Appl Mech Eng*. 1999;175(3):343-359.
3. Cockburn B, Karniadakis GE, Shu C-W. The development of discontinuous Galerkin methods. In: *Discontinuous Galerkin Methods*. Berlin, Germany: Springer; 2000:3-50. *Lecture Notes in Computational Science and Engineering*.
4. Altmann C, Beck AD, Hindenlang F, Staudenmaier M, Gassner GJ, Munz C-D. An efficient high performance parallelization of a discontinuous Galerkin spectral element method. *Facing the Multicore-Challenge III*. Berlin, Germany: Springer; 2013:37-47. *Lecture Notes in Computer Science*.
5. Persson PO, Peraire J. Sub-cell shock capturing for discontinuous Galerkin methods. Paper presented at: 44th AIAA Aerospace Sciences Meeting and Exhibit; 2006; Reno, NV.
6. Barter G, Darmofal D. Shock capturing with PDE-based artificial viscosity for DGFEM: part I. Formulation. *J Comput Phys*. 2010;229(5):1810-1827.
7. Klöckner A, Warburton T, Hesthaven JS. Viscous shock capturing in a time-explicit discontinuous Galerkin method. *Math Model Nat Phenom*. 2011;6(3):57-83.
8. Shu C-W, Osher S. Efficient implementation of essentially non-oscillatory shock-capturing schemes. *J Comput Phys*. 1988;77(2):439-471.
9. Shu C-W, Osher S. Efficient implementation of essentially non-oscillatory shock-capturing schemes, II. In: Hussaini MY, Van Leer B, Van Rosendale J, eds. *Upwind and High-Resolution Schemes*. Berlin, Germany: Springer; 1989:328-374.
10. Giani S, Schötzau D, Zhu L. An a-posteriori error estimate for -adaptive DG methods for convection-diffusion problems on anisotropically refined meshes. *Comput Math Appl*. 2014;67(4):869-887.
11. Dumbser M, Zanotti O, Loubère R, Diot S. A posteriori subcell limiting of the discontinuous Galerkin finite element method for hyperbolic conservation laws. *J Comput Phys*. 2014;278:47-75.
12. Von Neumann J, Richtmyer RD. A method for the numerical calculation of hydrodynamic shocks. *J Appl Phys*. 1950;21(3):232-237.
13. Bassi F, Rebay S. A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier-Stokes equations. *J Comput Phys*. 1997;131(2):267-279.
14. Bassi F, Rebay S. GMRES discontinuous Galerkin solution of the compressible Navier-Stokes equations. In: *Discontinuous Galerkin Methods*. : Springer; 2000:197-208. *Lecture Notes in Computational Science and Engineering*.
15. Bey KS, Oden JT. A Runge-Kutta discontinuous finite element method for high speed flows. Paper presented at: 10th Computational Fluid Dynamics Conference; 1991; Honolulu, HI.
16. Bey KS, Oden JT. *hp*-version discontinuous Galerkin methods for hyperbolic conservation laws. *Comput Methods Appl Mech Eng*. 1996;133(3):259-286.
17. Krivodonova L, Xin J, Remacle JF, Chevaugeon N, Flaherty JE. Shock detection and limiting with discontinuous Galerkin methods for hyperbolic conservation laws. *Appl Numer Math*. 2004;48(3-4):323-338.
18. Barter G, Darmofal D. Shock capturing with higher-order, PDE-based artificial viscosity. Paper presented at: 18th AIAA Computational Fluid Dynamics Conference; 2007; Miami, FL.
19. Persson P-O. Shock capturing for high-order discontinuous Galerkin simulation of transient flow problems. Paper presented: 21st AIAA Computational Fluid Dynamics Conference; 2013; San Diego, CA.
20. Lv Y, See YC, Ihme M. An entropy-residual shock detector for solving conservation laws using high-order discontinuous Galerkin methods. *J Comput Phys*. 2016;322:448-472.
21. Cockburn B, Shu C-W. The Runge-Kutta local projection P1-discontinuous-Galerkin finite element method for scalar conservation laws. *Modél Math Anal Numér*. 1991;25:26.
22. Cockburn B, Shu C-W. Runge–Kutta discontinuous Galerkin Methods for convection-dominated problems. *J Sci Comput*. 2001;16(3):173-261.
23. Gassner G, Lörcher F, Munz C-D. A discontinuous Galerkin scheme based on a space-time expansion II. Viscous Flow equations in multi dimensions. *J Sci Comput*. 2008;34(3):260-286.
24. Gassner G, Altmann C, Hindenlang F, Staudenmeier M, Munz C-D. Explicit discontinuous Galerkin schemes with adaptation in space and time. In: *36th CFD/ADIGMA Course on HP-Adaptive and HP-Multigrid Methods, VKI LS*. Rhode Saint Genèse, Belgium: Von Karman Institute for Fluid Dynamics; 2009.
25. Sandu A, Constantinescu EM. Multirate explicit Adams methods for time integration of conservation laws. *J Sci Comput*. 2009;38(2):229-249.
26. Gassner G, Hindenlang F, Munz C-D. A Runge-Kutta based discontinuous Galerkin method with time accurate local time stepping. In: *Adaptive High-Order Methods in Computational Fluid Dynamics*. Vol. 2. Singapore: World Scientific; 2011:95-118. *Advances in Computational Fluid Dynamics*.

27. Dumbser M, Käser M, Toro EF. An arbitrary high-order Discontinuous Galerkin method for elastic waves on unstructured meshes - V. Local time stepping and *p*-adaptivity. *Geophys J Int*. 2007;171:695-717.
28. Grote MJ, Mitkova T. High-order explicit local time-stepping methods for damped wave equations. *J Comput Appl Math*. 2013;239:270-289.
29. Kopriva DA. *Implementing Spectral Methods for Partial Differential Equations: Algorithms for Scientists and Engineers*. Berlin, Germany: Springer Science & Business Media; 2009.
30. Winters AR, Kopriva DA. High-order local time stepping on moving DG spectral element meshes. *J Sci Comput*. 2014;58(1):176-202.
31. Krämer-Eis S. *A High-Order Discontinuous Galerkin Method for Unsteady Compressible Flows With Immersed Boundaries* [dissertation]. Darmstadt, Germany: Technische Universität Darmstadt; 2017.
32. Krivodonova L. An efficient local time-stepping scheme for solution of nonlinear conservation laws. *J Comput Phys*. 2010;229(22):8537-8551.
33. Peskin CS. *Flow Patterns around Heart Valves: A digital Computer Method to Solve the Equations of Motion* [PhD thesis]. New York, NY: Albert Einstein College of Medicine; 1972.
34. Mittal R, Iaccarino G. Immersed boundary methods. *Annu Rev Fluid Mech*. 2005;37(1):239-261.
35. Qin R, Krivodonova L. A discontinuous Galerkin method for solutions of the Euler equations on Cartesian grids with embedded geometries. *J Comput Sci*. 2013;4(1):24-35.
36. Müller B, Kummer F, Oberlack M. Highly accurate surface and volume integration on implicit domains by means of moment-fitting. *Int J Numer Methods Eng*. 2013;96(8):512-528.
37. Müller B, Krämer-Eis Stephan, Kummer F, Oberlack M. A high-order discontinuous Galerkin method for compressible flows with immersed boundaries. *Int J Numer Methods Eng*. 2016;110(1):3-30.
38. Modisette J, Darmofal D. Toward a robust, higher-order cut-cell method for viscous flows. Paper presented at: 48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition; 2010; Orlando, FL.
39. Sun H, Darmofal DL. An adaptive simplex cut-cell method for high-order discontinuous Galerkin discretizations of elliptic interface problems and conjugate heat transfer problems. *J Comput Phys*. 2014;278:445-468.
40. Kummer F. *The BoSSS Discontinuous Galerkin Solver for Incompressible Fluid Dynamics and an Extension to Singular Equations* [PhD thesis]. Darmstadt, Germany: Technische Universität Darmstadt; 2012.
41. Krause D, Kummer F. An incompressible immersed boundary solver for moving body flows using a cut cell discontinuous Galerkin method. *Comput Fluids*. 2017;153:118-129.
42. Kummer F. Extended discontinuous Galerkin methods for two-phase flows: The spatial discretization. *Int J Numer Methods Eng*. 2016;109(2):259-289.
43. Ventura G. On the elimination of quadrature subcells for discontinuous functions in the extended finite-element method. *Int J Numer Methods Eng*. 2006;66(5):761-795.
44. Mavriplis C. Adaptive mesh strategies for the spectral element method. *Comput Methods Appl Mech Eng*. 1994;116(1-4):77-86.
45. Arnold DN. An interior penalty finite element method with discontinuous elements. *SIAM J Numer Anal*. 1982;19(4):742-760.
46. Arnold DN, Brezzi F, Cockburn B, Marini LD. Unified analysis of discontinuous Galerkin methods for elliptic problems. *SIAM J Numer Anal*. 2002;39(5):1749-1779.
47. Courant R, Friedrichs K, Lewy H. Über die partiellen differenzengleichungen der mathematischen physik. *Mathematische Annalen*. 1928;100(1):32-74.
48. Watkins J, Asthana K, Jameson A. A Numerical Analysis of the Nodal Discontinuous Galerkin Scheme via Flux Reconstruction for the Advection-Diffusion Equation. *Comput Fluids*. 2016;139:233-247.
49. Kubatko EJ, Dawson C, Westerink JJ. Time step restrictions for Runge–Kutta discontinuous Galerkin methods on triangular grids. *J Comput Phys*. 2008;227(23):9697-9710.
50. Toulorge T, Desmet W. CFL conditions for Runge–Kutta discontinuous Galerkin methods on triangular grids. *J Comput Phys*. 2011;230(12):4657-4678.
51. Krivodonova L, Qin R. An analysis of the spectrum of the discontinuous Galerkin method II: nonuniform grids. *Appl Numer Math*. 2013;71:41-62.
52. Gear CW, Wells DR. Multirate linear multistep methods. *BIT Numer Math*. 1984;24(4):484-502.
53. Stock A. *Development and Application of a Multirate Multistep AB Method to a Discontinuous Galerkin Method Based Particle in Cell Scheme* [diploma thesis]. Stuttgart, Germany: Institut für Aerodynamik und Gasdynamik, Universität Stuttgart and Brown University; 2009.
54. Macqueen J. Some methods for classification and analysis of multivariate observations. In: Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability; 1967; Berkeley, CA.
55. Toro EF. *Riemann Solvers and Numerical Methods for Fluid Dynamics*. 3rd ed. Berlin, Germany: Springer; 2009.
56. Hartmann R, Houston P. An optimal order interior penalty discontinuous Galerkin discretization of the compressible Navier-Stokes equations. *J Comput Phys*. 2008;227(22):9670-9685.
57. Huerta A, Casoni E, Peraire J. A simple shock-capturing technique for high-order discontinuous Galerkin methods. *Int J Numer Meth Fluids*. 2012;69(10):1614-1632.
58. Nguyen N, Persson P-O, Peraire J. RANS solutions using high order discontinuous Galerkin methods. Paper presented at: 45th AIAA Aerospace Sciences Meeting and Exhibit; 2007; Reno, NV.
59. Rault A, Chiavassa G, Donat R. Shock-Vortex Interactions at High Mach Numbers. *J Sci Comput*. 2003;19(1-3):347-371.
60. Anderson JD. *Modern compressible flow: With Historical Perspective*. Singapore: McGraw-Hill; 1990.

61. Chatterjee A. Shock wave deformation in shock-vortex interactions. *Shock Waves*. 1999;9(2):95-105.
62. Woodward P, Colella P. The numerical simulation of two-dimensional fluid flow with strong shocks. *J Comput Phys*. 1984;54(1):115-173.
63. Weber J. *Dynamic Load Balancing for the Simulation of Transonic Flows Using a Discontinuous Galerkin Discretization With Local Time Stepping* [master's thesis. Darmstadt, Germany: Technische Universität Darmstadt; 2018.
64. Butz AR. Convergence with Hilbert's space filling curve. *J Comput Syst Sci*. 1969;3(2):128-146.

How to cite this article: Geisenhofer M, Kummer F, Müller B. A discontinuous Galerkin immersed boundary solver for compressible flows: Adaptive local time stepping for artificial viscosity-based shock-capturing on cut cells. *Int J Numer Meth Fluids*. 2019;91:448–472. https://doi.org/10.1002/fld.4761

APPENDIX A

TIME INTEGRATION

A.1 | A variable coefficient Adams-Bashforth time integration scheme

In this section, we briefly present a variable coefficient Adams-Bashforth time integration scheme.[31] Since the LTS clustering is dynamically adapted in time (cf Section 5.3), we need a variable coefficient implementation that has also been used, eg, for simulations on moving meshes.[30]

We start from the definition of a general ODE system

$$\frac{d\vec{y}}{dt} = \vec{f}(\vec{y}, t), \tag{A1}$$

which we integrate in time from t_n to t_{n+1}

$$\vec{y}_{n+1} = \vec{y}_n + \int_{t_n}^{t_{n+1}} \vec{f}(\vec{y}, t)\, dt, \tag{A2}$$

with $\vec{y}_n = \vec{y}(t_n)$. The integrand in Equation (A2) is replaced by the weighted sum of current and previous flux evaluations $\vec{f}_{n-q}$ with some coefficients β_q

$$\vec{y}_{n+1} = \vec{y}_n + \sum_{q=0}^{Q-1} \beta_q \vec{f}_{n-q}\,. \tag{A3}$$

We determine the coefficients β_q by means of the Lagrange formula

$$\beta_q(t) = \int_{t_n}^{t_{n+1}} l_q(s)\, ds, \quad \text{where} \quad l_q(t) = \prod_{\substack{i=0 \\ i \neq q}}^{Q-1} \frac{t - t_{n-1}}{t_{n-q} - t_{n-i}}\,. \tag{A4}$$

As an example, we expand the weighted sum in Equation (A3) for an order of $Q = 2$

$$\sum_{q=0}^{1} \beta_q(t) \vec{f}_{n-q} = \int_{t_n}^{t_{n+1}} \frac{s - t_{n-1}}{\Delta t_n}\, ds\, \vec{f}_n + \int_{t_n}^{t_{n+1}} \frac{s - t_n}{-\Delta t_n}\, ds\, \vec{f}_{n-1}\,, \tag{A5}$$

with $\Delta t_n = t_n - t_{n-1}$. We introduce the substitution $u = (t_{n+1} - s)/\Delta t_{n+1}$ to solve for the integrals in Equation (A5) (not shown), which yields

$$\beta_0 = \frac{\Delta t_{n+1}}{2}\left(\frac{\Delta t_{n+1}}{\Delta t_n} + 2\right), \quad \beta_1 = -\frac{\Delta t_{n+1}}{2}\left(\frac{\Delta t_{n+1}}{\Delta t_n}\right). \tag{A6}$$

The final form reads

$$\vec{y}_{n+1} = \vec{y}_n + \left(\beta_0 \vec{f}_n + \beta_1 \vec{f}_{n-1}\right). \tag{A7}$$