INVARIANT DOMAINS PRESERVING ALE APPROXIMATION OF HYPERBOLIC SYSTEMS WITH CONTINUOUS FINITE ELEMENTS *

JEAN-LUC GUERMOND†, BOJAN POPOV†, LAURA SAAVEDRA‡, AND YONG YANG†

Abstract. A conservative invariant domain preserving Arbitrary Lagrangian Eulerian method for solving nonlinear hyperbolic systems is introduced. The method is explicit in time, works with continuous finite elements and is first-order accurate in space. One originality of the present work is that the artificial viscosity is unambiguously defined irrespective of the mesh geometry/anisotropy and does not depend on any ad hoc parameter. The proposed method is meant to be a stepping stone for the construction of higher-order methods in space by using appropriate limitation techniques.

Key words. Conservation equations, hyperbolic systems, Arbitrary Lagrangian Eulerian, moving schemes, invariant domain, first-order method, finite element method.

AMS subject classifications. 65M60, 65M10, 65M15, 35L65

1. Introduction.

Consider the following hyperbolic system in conservative form

$$\begin{cases} \partial_t \boldsymbol{u} + \nabla\cdot\boldsymbol{f}(\boldsymbol{u}) = 0, \quad \text{for } (\boldsymbol{x},t) \in \mathbb{R}^d\times\mathbb{R}_+. \\ \boldsymbol{u}(\boldsymbol{x},0) = \boldsymbol{u}_0(\boldsymbol{x}), \quad \text{for } \boldsymbol{x} \in \mathbb{R}^d, \end{cases} \tag{1.1}$$

where the dependent variable $\boldsymbol{u}$ is $\mathbb{R}^m$-valued and the flux $\boldsymbol{f}$ is $\mathbb{R}^{m\times d}$-valued. The objective of this paper is to investigate an approximation technique for solving (1.1) using an Arbitrary Lagrangian Eulerian (ALE) formulation with continuous finite elements and explicit time stepping on non-uniform meshes in any space dimension.

The interest for finite elements in the context of compressible Lagrangian hydrodynamics has been recently revived by the work of Dobrev et al. [13], where the authors have demonstrated that high-order finite elements have good properties in terms of geometry representation, symmetry preservation, resolution of shock fronts and high-order convergence rate for smooth solutions. The finite element formalism has been combined with staggered grid hydrodynamics methods in Barlow [3], Scovazzi et al. [32] and with cell-centered hydrodynamics methods in Vilar et al. [35] in the form of a Discontinuous Galerkin scheme. One common factor of all these papers is that the stabilization is done by introducing some artificial viscosity to control post-shock oscillations, and high-order convergence in space is achieved by restricting the diffusion to be active only in the singular regions of the solution. This can be done in many ways, for instance by measuring smoothness like in the ENO/WENO literature like in Cheng and Shu [10] or by using entropies like in the entropy viscosity methodology of Guermond et al. [21]. A detailed list of requirements and specific artificial viscosity expressions for Lagrangian hydrodynamics have been proposed by Caramana et al. [6], Caramana and Loubère [7], Shashkov and Campbell [33], Kolev

*This material is based upon work supported in part by the National Science Foundation grants DMS-1217262, by the Air Force Office of Scientific Research, USAF, under grant/contract number FA99550-12-0358, and by the Army Research Office under grant/contract number W911NF-15-1-0517. Draft version, February 24, 2016

†Department of Mathematics, Texas A&M University 3368 TAMU, College Station, TX 77843, USA.

‡Departamento Fundamentos Matemáticos, Universidad Politécnica de Madrid, E.T.S.I. Aeronáuticos, 28040 Madrid, Spain

and Rieben [25], Lipnikov and Shashkov [30]. The artificial viscosity can also be implicitly introduced by using Godunov type or cell-centered type methods based on Riemann solvers, see e.g., Boscheri and Dumbser [4], Carré et al. [8].

In the present paper we revisit the artificial viscosity problem for general hyperbolic systems like (1.1) using an Arbitrary Lagrangian Eulerian (ALE) formulation and explicit time stepping. The originality of the present work is that (i) The approximation in space is done with continuous finite elements of arbitrary order; (ii) The local shape functions can be linear Lagrange elements, Bernstein-Bezier elements of any order or any other nonnegative functions (not necessarily polynomials) that have the partition of unity property; (iii) The finite element meshes considered are nonuniform, curvilinear and the space dimension is arbitrary; (iv) The artificial viscosity is unambiguously defined irrespective of the mesh geometry/anisotropy, does not depend on any ad hoc parameter, contrary to what has been previously done in the finite element literature, and leads to precise invariant domain properties and entropy inequalities; (v) The methods works for *any* (reasonable in the sense of §2) hyperbolic system. Although entirely based on continuous finite elements, our work is deeply rooted in the work of Lax [28] and Hoff [24], and in some sense mimics well established schemes from the finite volume literature, e.g., Guillard and Farhat [22], Farhat et al. [14] or the discontinuous Galerkin literature, e.g., Vilar et al. [35]. The proposed method is meant to be a stepping stone for the construction of higher-order methods in space by using, for instance, the flux transport correction methodology à la Boris-Book-Zalesak, or any generalization thereof, to implement limitation. None of these generalizations are discussed in the paper. The sole objective of the present work is to give a firm, undisputable, theoretical footing to the first-order, invariant-domain preserving, method.

The paper is organized as follows. We introduce some notation and recall important properties about the one-dimensional Riemann problem in §2. We introduce notation relative to mesh motion and Lagrangian mappings in §3. The results established in §2 and §3 are standard and will be invoked in §4 and §5. The reader who is familiar with these notions is invited to go directly to §4. We describe in §4 two versions of an ALE algorithm to approximate the solution of (1.1). The first algorithm, henceforth referred to as version 1, is composed of the steps (4.9)-(4.10)-(4.11)-(4.15). The second algorithm, i.e., version 2, is composed of the steps (4.32)-(4.33)-(4.34)-(4.35)-(4.15). The key difference between these two algorithms is the way the mass carried by the shape functions is updated (compare (4.10) and (4.34)). Only version 1 can be easily made high-order in time by means of the strong stability preserving technology (see Ferracina and Spijker [15], Higueras [23], Gottlieb et al. [18] for details on SSP techniques). It is proved in §5 that under the appropriate CFL condition both algorithms are invariant domain preserving, conservative and satisfy a local entropy inequality for any admissible entropy pair. The main results of this section are Theorem 5.2 and Theorem 5.6. The SSP RK3 extension of scheme 1 is tested numerically in §6 on scalar conservation equations and on the compressible Euler equations using two different finite element implementations of the method. In all the cases the ALE velocity is ad hoc and no particular effort has been made to optimize this quantity. The purpose of this paper is not to design an optimal ALE velocity but to propose an algorithm that is conservative and invariant domain preserving for *any* reasonable ALE velocity.

2. Riemann problem and invariant domain. We recall in this section elementary properties of Riemann problems that will be used in the paper.

2.1. Notation and boundary conditions. In this paper the dependent variable $\boldsymbol{u}$ in (1.1) is considered as a column vector $\boldsymbol{u} = (u_1, \dots, u_m)^\mathsf{T}$. The flux is a matrix with entries $f_{ij}(\boldsymbol{u})$, $1 \le i \le m$, $1 \le j \le d$. We denote $\boldsymbol{f}_i$ the row vector $(f_{i1}, \dots, f_{id})$, $i \in \{1{:}m\}$. We denote by $\nabla{\cdot}\boldsymbol{f}$ the column vector with entries $(\nabla{\cdot}\boldsymbol{f})_i = \sum_{1\le j\le d} \partial_{x_j} f_{ij}$. For any $\boldsymbol{n} = (n_1 \dots, n_d)^\mathsf{T} \in \mathbb{R}^d$, we denote $\boldsymbol{f}(\boldsymbol{u}){\cdot}\boldsymbol{n}$ the column vector with entries $\boldsymbol{f}_i(\boldsymbol{u}){\cdot}\boldsymbol{n} = \sum_{1\le l\le d} n_l f_{il}(\boldsymbol{u})$, where $i \in \{1{:}m\}$. Given two vector fields, say $\boldsymbol{u} \in \mathbb{R}^m$ and $\boldsymbol{v} \in \mathbb{R}^d$, we define $\boldsymbol{u} \otimes \boldsymbol{v}$ to be the $m{\times}d$ matrix with entries $u_i v_j$, $i \in \{1{:}m\}$, $j \in \{1{:}d\}$. We also define $\nabla{\cdot}(\boldsymbol{u} \otimes \boldsymbol{v})$ to be the column vector with entries $\nabla{\cdot}(\boldsymbol{u} \otimes \boldsymbol{v})_i = \sum_{j=1}^d \partial_j(u_i v_j)$. The unit sphere in $\mathbb{R}^d$ centered at 0 is denoted by $S^{d-1}(\boldsymbol{0}, 1)$.

To simplify questions regarding boundary conditions, we assume that the initial data is constant outside a compact set and we solve the Cauchy problem in $\mathbb{R}^d$ or we use periodic boundary conditions.

2.2. One-dimensional Riemann problem. We are not going to try to define weak solutions to (1.1), but instead we assume that there is a clear notion for the solution of the Riemann problem. To stay general we introduce a generic hyperbolic flux $\boldsymbol{h}$ and we say that $(\eta, \boldsymbol{q})$ is an entropy pair associated with the flux $\boldsymbol{h}$ if η is convex and the following identity holds:

$$\partial_{v_k}(\boldsymbol{q}(\boldsymbol{v}){\cdot}\boldsymbol{n}) = \sum_{i=1}^m \partial_{v_i}\eta(\boldsymbol{v})\partial_{v_k}(\boldsymbol{h}_i(\boldsymbol{v}){\cdot}\boldsymbol{n}), \qquad \forall k \in \{1{:}m\},\ \forall \boldsymbol{n} \in S^{d-1}(\boldsymbol{0}, 1). \tag{2.1}$$

We refer to Chen [9, §2] for more details on convex entropies and symmetrization. In the rest of the paper we assume that there exists a nonempty admissible set $\mathcal{A}_{\boldsymbol{h}} \subset \mathbb{R}^m$ such that the following one-dimensional Riemann problem

$$\partial_t \boldsymbol{u} + \partial_x(\boldsymbol{h}(\boldsymbol{u}){\cdot}\boldsymbol{n}) = 0, \quad (x,t) \in \mathbb{R}{\times}\mathbb{R}_+, \qquad \boldsymbol{u}(x,0) = \begin{cases} \boldsymbol{u}_L, & \text{if } x < 0 \\ \boldsymbol{u}_R, & \text{if } x > 0, \end{cases} \tag{2.2}$$

has a unique entropy satisfying solution for any pair of states $(\boldsymbol{u}_L, \boldsymbol{u}_R) \in \mathcal{A}_{\boldsymbol{h}}{\times}\mathcal{A}_{\boldsymbol{h}}$ and any unit vector $\boldsymbol{n} \in S^{d-1}(\boldsymbol{0}, 1)$. We henceforth denote the solution to this problem by $\boldsymbol{u}(\boldsymbol{h}, \boldsymbol{n}, \boldsymbol{u}_L, \boldsymbol{u}_R)$. We also say that $\boldsymbol{u}$ is an entropy satisfying solution of (2.2) if the following holds in the distribution sense

$$\partial_t \eta(\boldsymbol{u}) + \partial_x(\boldsymbol{q}(\boldsymbol{u}){\cdot}\boldsymbol{n}) \le 0. \tag{2.3}$$

for any entropy pair $(\eta, \boldsymbol{q})$.

It is unrealistic to expect a general theory of the Riemann problem (2.2) for arbitrary nonlinear hyperbolic systems with large data, we henceforth make the following assumption:

$$\begin{aligned} &\text{The unique solution of (2.2) has a finite speed of} \\ &\text{propagation for any } \boldsymbol{n} \text{ and any } (\boldsymbol{u}_L, \boldsymbol{u}_R) \in \mathcal{A}_{\boldsymbol{h}}{\times}\mathcal{A}_{\boldsymbol{h}}, \text{ i.e.,} \\ &\text{there are } \lambda_L(\boldsymbol{h}, \boldsymbol{n}, \boldsymbol{u}_L, \boldsymbol{u}_R) \le \lambda_R(\boldsymbol{h}, \boldsymbol{n}, \boldsymbol{u}_L, \boldsymbol{u}_R) \text{ such} \\ &\boldsymbol{u}(x,t) = \begin{cases} \boldsymbol{u}_L, & \text{if } x \le t\lambda_L(\boldsymbol{h}, \boldsymbol{n}, \boldsymbol{u}_L, \boldsymbol{u}_R) \\ \boldsymbol{u}_R, & \text{if } x \ge t\lambda_R(\boldsymbol{h}, \boldsymbol{n}, \boldsymbol{u}_L, \boldsymbol{u}_R). \end{cases} \end{aligned} \tag{2.4}$$

This assumption is known to hold for small data when the system is strictly hyperbolic with smooth flux and all the characteristic fields are either genuinely nonlinear or

linearly degenerate. More precisely there exists $\delta > 0$ such that the Riemann problem has a unique self-similar weak solution in Lax's form for any initial data such that $\|\boldsymbol{u}_L - \boldsymbol{u}_R\|_{\ell^2} \leq \delta$, see Lax [29] and Bressan [5, Thm 5.3]. In particular there are $2m$ numbers $\lambda_1^- \leq \lambda_1^+ \leq \lambda_2^- \leq \lambda_2^+ \leq \ldots \leq \lambda_m^- \leq \lambda_m^+$, defining up to $2m+1$ sectors (some could be empty) in the (x,t) plane:

$$\frac{x}{t} \in (-\infty, \lambda_1^-), \quad \frac{x}{t} \in (\lambda_1^-, \lambda_1^+), \ldots, \quad \frac{x}{t} \in (\lambda_m^-, \lambda_m^+), \quad \frac{x}{t} \in (\lambda_m^+, \infty). \tag{2.5}$$

where the Riemann solution is $\boldsymbol{u}_L$ in the sector $\frac{x}{t} \in (-\infty, \lambda_1^-)$, $\boldsymbol{u}_R$ in the last sector $\frac{x}{t} \in (\lambda_m^+, \infty)$, and either a constant state or an expansion in the other sectors, see [5, Chap. 5]. In this case we have $\lambda_L := \lambda_L(\boldsymbol{h}, \boldsymbol{n}, \boldsymbol{u}_L, \boldsymbol{u}_R) = \lambda_1^-$ and $\lambda_R := \lambda_R(\boldsymbol{h}, \boldsymbol{n}, \boldsymbol{u}_L, \boldsymbol{u}_R) = \lambda_m^+$. The sector $\lambda_1^- t < x < \lambda_m^+ t$, $0 < t$, is henceforth referred to as the Riemann fan. The maximum wave speed in the Riemann fan is $\lambda_{\max} := \lambda_{\max}(\boldsymbol{h}, \boldsymbol{n}, \boldsymbol{u}_L, \boldsymbol{u}_R) := \max(|\lambda_L|, |\lambda_R|)$. For brevity, when there is no ambiguity, we will omit the dependence of λ_L, λ_R and $\lambda_{\max}$ on the parameters $\boldsymbol{h}, \boldsymbol{n}, \boldsymbol{u}_L, \boldsymbol{u}_R$. The finite speed assumption (2.4) holds in the case of strictly hyperbolic systems that may have characteristic families that are either not genuinely nonlinear or not linearly degenerate, see e.g., Dafermos [12, Thm. 9.5.1].

2.3. Invariant sets and domains. The following elementary result is an important, well-known, consequence of the Riemann fan assumption (2.4):

LEMMA 2.1. *Let $\boldsymbol{h}$ be a hyperbolic flux over the admissible set $\mathcal{A}_{\boldsymbol{h}}$ and satisfying the finite wave speed assumption* (2.4)*. Let $\boldsymbol{v}(\boldsymbol{h}, \boldsymbol{n}, \boldsymbol{v}_L, \boldsymbol{v}_R)$ be the unique solution to the problem $\partial_t \boldsymbol{v} + \partial_x(\boldsymbol{h}(\boldsymbol{v})\cdot\boldsymbol{n}) = 0$ with initial data $\boldsymbol{v}_L, \boldsymbol{v}_R \in \mathcal{A}_{\boldsymbol{h}}$. Let $(\eta, \boldsymbol{q})$ be an entropy pair associated with the flux $\boldsymbol{h}$. Assume that $t\,\lambda_{\max}(\boldsymbol{h}, \boldsymbol{n}, \boldsymbol{v}_L, \boldsymbol{v}_R) \leq \frac{1}{2}$ and let $\overline{\boldsymbol{v}}(t, \boldsymbol{h}, \boldsymbol{n}, \boldsymbol{v}_L, \boldsymbol{v}_R) := \int_{-\frac{1}{2}}^{\frac{1}{2}} \boldsymbol{v}(\boldsymbol{h}, \boldsymbol{n}, \boldsymbol{v}_L, \boldsymbol{v}_R)(x,t)\,\mathrm{d}\boldsymbol{x}$, then*

$$\overline{\boldsymbol{v}}(t, \boldsymbol{h}, \boldsymbol{n}, \boldsymbol{v}_L, \boldsymbol{v}_R) = \tfrac{1}{2}(\boldsymbol{v}_L + \boldsymbol{v}_R) - t\big(\boldsymbol{h}(\boldsymbol{v}_R)\cdot\boldsymbol{n} - \boldsymbol{h}(\boldsymbol{v}_L)\cdot\boldsymbol{n}\big). \tag{2.6}$$

$$\eta(\overline{\boldsymbol{v}}(t, \boldsymbol{h}, \boldsymbol{n}, \boldsymbol{v}_L, \boldsymbol{v}_R)) \leq \tfrac{1}{2}(\eta(\boldsymbol{v}_L) + \eta(\boldsymbol{v}_R)) - t(\boldsymbol{q}(\boldsymbol{v}_R)\cdot\boldsymbol{n} - \boldsymbol{q}(\boldsymbol{v}_L)\cdot\boldsymbol{n}). \tag{2.7}$$

We now introduce the notions of invariant sets. The definitions that we adopt are slightly different from what is usually done in the literature (see e.g., in Chueh et al. [11], Hoff [24], Frid [16].

DEFINITION 2.2 (Invariant set). *Let $\boldsymbol{h}$ be a hyperbolic flux over the admissible set $\mathcal{A}_{\boldsymbol{h}}$ and satisfying the finite wave speed assumption* (2.4)*. We say that a convex set $A \subset \mathcal{A}_{\boldsymbol{h}} \subset \mathbb{R}^m$ is invariant for the problem $\partial_t \boldsymbol{v} + \nabla\cdot\boldsymbol{h}(\boldsymbol{v}) = 0$ if for any pair $(\boldsymbol{v}_L, \boldsymbol{v}_R) \in A \times A$, any unit vector $\boldsymbol{n} \in \mathcal{S}^{d-1}(\boldsymbol{0}, 1)$, the average of the entropy solution of the Riemann problem $\partial_t \boldsymbol{v} + \nabla\cdot(\boldsymbol{h}(\boldsymbol{v})\cdot\boldsymbol{n}) = 0$ over the Riemann fan $\frac{1}{t(\lambda_R - \lambda_L)} \int_{\lambda_L t}^{\lambda_R t} \boldsymbol{v}(\boldsymbol{h}, \boldsymbol{n}, \boldsymbol{v}_L, \boldsymbol{v}_R)(x,t)\,\mathrm{d}\boldsymbol{x}$, remains in A for all $t > 0$.*

Remark 2.1. The above definition implies that $\frac{1}{I}\int_I \boldsymbol{v}(\boldsymbol{h}, \boldsymbol{n}, \boldsymbol{v}_L, \boldsymbol{v}_R)(x,t)\,\mathrm{d}\boldsymbol{x} \in A$ for any $t > 0$ and any interval I such that $(\lambda_L t, \lambda_R t) \subset I$.

LEMMA 2.3 (Translation). *Let $\mathbf{W} \in \mathbb{R}^d$ and let $\boldsymbol{g}(\boldsymbol{v}) := \boldsymbol{f}(\boldsymbol{v}) - \boldsymbol{v} \otimes \mathbf{W}$.*

(i) The two problems: $\partial_t \boldsymbol{u} + \nabla\cdot\boldsymbol{f}(\boldsymbol{u}) = 0$ and $\partial_t \boldsymbol{v} + \nabla\cdot\boldsymbol{g}(\boldsymbol{v}) = 0$ have the same admissible sets and the same invariant sets.

(ii) $(\eta(\boldsymbol{u}), \boldsymbol{q}(\boldsymbol{u}))$ is an entropy pair for the flux $\boldsymbol{f}$ if and only if $(\eta(\boldsymbol{v}), \boldsymbol{q}(\boldsymbol{v}) - \eta(\boldsymbol{v})\mathbf{W})$ is an entropy pair for the flux $\boldsymbol{g}$.

Proof. (i) Given $\boldsymbol{n} \in \mathcal{S}^{d-1}(\boldsymbol{0}, 1)$, the solutions of the Riemann problems $\partial_t \boldsymbol{u} + \partial_x(\boldsymbol{f}(\boldsymbol{u})\cdot\boldsymbol{n}) = 0$ and $\partial_t \boldsymbol{v} + \partial_x(\boldsymbol{g}(\boldsymbol{v})\cdot\boldsymbol{n}) = 0$, with the same initial data, are such that $\boldsymbol{v}(x,t) = \boldsymbol{u}(x + (\mathbf{W}\cdot\boldsymbol{n})t, t)$. Therefore the admissible sets and the invariant sets are

identical. (ii) Let $\boldsymbol{n} \in \mathcal{S}^{d-1}(\mathbf{0},1)$ and $k \in \{1{:}d\}$, then using the definition (2.1), we have

$$\sum_{i=1}^{m} \partial_{v_i}\eta(\boldsymbol{v})\partial_{v_k}(\boldsymbol{f}_i(\boldsymbol{v})\cdot\boldsymbol{n} - v_i\mathbf{W}\cdot\boldsymbol{n}) = \partial_{v_k}(\boldsymbol{q}(\boldsymbol{v})\cdot\boldsymbol{n}) - \mathbf{W}\cdot\boldsymbol{n}\sum_{i=1}^{m}\partial_{v_i}\eta(\boldsymbol{v})\partial_{v_k}(v_i)$$
$$= \partial_{v_k}(\boldsymbol{q}(\boldsymbol{v})\cdot\boldsymbol{n}) - \mathbf{W}\cdot\boldsymbol{n}\partial_{v_k}(\eta(\boldsymbol{v})) = \partial_{v_k}((\boldsymbol{q}(\boldsymbol{v}) - \eta(\boldsymbol{v})\mathbf{W})\cdot\boldsymbol{n}).$$

The conclusion follows readily. ∎

3. Geometric preliminaries. In this section we introduce some notation and recall some general results about Lagrangian mappings. The key results, which will be invoked in §4 and §5, are lemmas 3.1, 3.2, and 3.3. The reader who is familiar with these notions is invited to skip this section and to go directly to §4.

3.1. Jacobian of the coordinate transformation. Let $\boldsymbol{\Phi} : \mathbb{R}^d \times \mathbb{R}_+ \longrightarrow \mathbb{R}^d$ be a uniformly Lipschitz mapping. We additionally assume that there is $t^* > 0$ such that the mapping $\boldsymbol{\Phi}_t : \mathbb{R}^d \ni \boldsymbol{\xi} \longmapsto \boldsymbol{\Phi}_t(\boldsymbol{\xi}) := \boldsymbol{\Phi}(\boldsymbol{\xi},t) \in \mathbb{R}^d$ is invertible for all $t \in [0,t^*]$. Let $\boldsymbol{v}_{\mathrm{A}} : \mathbb{R}^d \times [0,t^*] \longrightarrow \mathbb{R}^d$ be the vector field implicitly defined by

$$\boldsymbol{v}_{\mathrm{A}}(\boldsymbol{\Phi}(\boldsymbol{\xi},t),t) := \partial_t\boldsymbol{\Phi}(\boldsymbol{\xi},t), \quad \forall(\boldsymbol{\xi},t) \in \mathbb{R}\times[0,t^*]. \tag{3.1}$$

Note that this definition makes sense owing to the inversibility assumption on the mapping $\boldsymbol{\Phi}_t$; actually (3.1) is equivalent to $\boldsymbol{v}_{\mathrm{A}}(\boldsymbol{x},t) := \partial_t\boldsymbol{\Phi}(\boldsymbol{\Phi}_t^{-1}(\boldsymbol{x}),t)$ for any $t \in [0,t^*]$.

LEMMA 3.1 (Liouville's formula). *Let $\mathbb{J}(\boldsymbol{\xi},t) = \nabla_{\boldsymbol{\xi}}\boldsymbol{\Phi}(\boldsymbol{\xi},t)$ be the Jacobian matrix of $\boldsymbol{\Phi}$, then*

$$\partial_t \det(\mathbb{J}(\boldsymbol{\xi},t)) = (\nabla\cdot\boldsymbol{v}_{\mathrm{A}})(\boldsymbol{\Phi}(\boldsymbol{\xi},t),t)\det(\mathbb{J}(\boldsymbol{\xi},t)). \tag{3.2}$$

Proof. This result is wellknown but we give the proof for completeness. Let S_d be the set of all permutations of the set $\{1,\dots,d\}$ and $\mathrm{sgn}(\sigma)$ denotes the signature of $\sigma \in S_d$. The Leibniz formula for the determinant gives $\det(\mathbb{J}(\boldsymbol{\xi},t)) = \sum_{\sigma\in S_n}\mathrm{sgn}(\sigma)\partial_{\sigma(1)}\Phi_1\dots\partial_{\sigma(d)}\Phi_d$. Then using the definition of the field $\boldsymbol{v}_{\mathrm{A}}$, with Cartesian coordinates $v_1,\dots,v_d$, we have

$$\partial_t\det(\mathbb{J}(\boldsymbol{\xi},t)) = \sum_{\sigma\in S_n}\mathrm{sgn}(\sigma)\partial_{\sigma(1)}\partial_t\Phi_1\dots\partial_{\sigma(d)}\Phi_d + \dots + \partial_{\sigma(1)}\Phi_1\dots\partial_{\sigma(d)}\partial_t\Phi_d$$
$$= \sum_{\sigma\in S_n}\mathrm{sgn}(\sigma)\partial_{\sigma(1)}(v_1(\boldsymbol{\Phi}(\boldsymbol{\xi},t),t))\dots\partial_{\sigma(d)}\Phi_d + \dots + \partial_{\sigma(1)}\Phi_1\dots\partial_{\sigma(d)}(v_d(\boldsymbol{\Phi}(\boldsymbol{\xi},t),t))$$
$$= \sum_{k=1}^{d}(\partial_k v_1)(\boldsymbol{\Phi}(\boldsymbol{\xi},t),t)\sum_{\sigma\in S_n}\mathrm{sgn}(\sigma)\partial_{\sigma(1)}\Phi_k\partial_{\sigma(2)}\Phi_2\dots\partial_{\sigma(d)}\Phi_d +$$
$$\dots + \sum_{k=1}^{d}(\partial_k v_d)(\boldsymbol{\Phi}(\boldsymbol{\xi},t),t)\sum_{\sigma\in S_n}\mathrm{sgn}(\sigma)\partial_{\sigma(1)}\Phi_1\dots\partial_{\sigma(d-1)}\Phi_{d-1}\partial_{\sigma(d)}\Phi_k.$$

Upon observing that $\sum_{\sigma\in S_n}\mathrm{sgn}(\sigma)\partial_{\sigma(1)}\Phi_k\partial_{\sigma(2)}\Phi_2\dots\partial_{\sigma(d)}\Phi_d$ is zero unless $k=1$, etc., we infer that

$$\partial_t\det(\mathbb{J}(\boldsymbol{\xi},t)) = (\partial_1 v_1)(\boldsymbol{\Phi}(\boldsymbol{\xi},t),t)\det(\mathbb{J}(\boldsymbol{\xi},t)) + \dots(\partial_d v_d)(\boldsymbol{\Phi}(\boldsymbol{\xi},t),t)\det(\mathbb{J}(\boldsymbol{\xi},t)),$$

which proves the result. ∎

Remark 3.1. Note that in (3.2) the expression $(\nabla\cdot\boldsymbol{v}_{\mathrm{A}})(\boldsymbol{\Phi}(\boldsymbol{\xi},t),t)$ should not be confused with $\nabla\cdot(\boldsymbol{v}_{\mathrm{A}}(\boldsymbol{\Phi}(\boldsymbol{\xi},t),t))$.

3.2. Mass transformation. Let $\psi \in C_0^0(\mathbb{R}^d;\mathbb{R})$ be a continuous compactly supported function. We define $\varphi(\boldsymbol{x},t) := \psi(\boldsymbol{\Phi}_t^{-1}(\boldsymbol{x}))$ for all $t \in [0,t^*]$, i.e., $\varphi(\boldsymbol{\Phi}(\boldsymbol{\xi},t),t) = \psi(\boldsymbol{\xi})$ for all $(\boldsymbol{\xi},t) \in \mathbb{R}^d \times [0,t^*]$. We want to compute $\int_{\mathbb{R}^d} \varphi(\boldsymbol{x},t)\,\mathrm{d}\boldsymbol{x}$ and relate it to $\int_{\mathbb{R}^d} \psi(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi}$.

LEMMA 3.2. *Assume that $t \longmapsto \det(\mathbb{J}(\boldsymbol{\xi},t))$ is a polynomial function in t of degree at most $r \in \mathbb{N}$ for any $\boldsymbol{\xi} \in \mathbb{R}^d$. Let $(\omega_l,\zeta_l)_{l\in\mathcal{L}}$ be a quadrature such that $\int_0^1 f(\zeta)\,\mathrm{d}\zeta \simeq \sum_{l\in\mathcal{L}} \omega_l f(\zeta_l)$ is exact for all polynomials of degree at most $\max(r-1,0)$, then, using the notation $\varphi(\boldsymbol{x},t) = \psi(\boldsymbol{\Phi}_t^{-1}(\boldsymbol{x}))$, we have*

$$\int_{\mathbb{R}^d} \varphi(\boldsymbol{x},t)\,\mathrm{d}\boldsymbol{x} - \int_{\mathbb{R}^d} \psi(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi} = t\sum_{l\in\mathcal{L}} \omega_l \int_{\mathbb{R}^d} \varphi(\boldsymbol{x},t\zeta_l)\nabla\cdot\boldsymbol{v}_{\mathrm{A}}(\boldsymbol{x},t\zeta_l)\,\mathrm{d}\boldsymbol{x} \tag{3.3}$$

Proof. Using the definitions we infer that

$$\begin{aligned}\int_{\mathbb{R}^d} \varphi(\boldsymbol{x},t)\,\mathrm{d}\boldsymbol{x} - \int_{\mathbb{R}^d} \psi(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi} &= \int_{\mathbb{R}^d} \varphi(\boldsymbol{\Phi}_t(\boldsymbol{\xi}),t)\det(\mathbb{J}(\boldsymbol{\xi},t))\,\mathrm{d}\boldsymbol{\xi} - \int_{\mathbb{R}^d} \psi(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi}\\ &= \int_{\mathbb{R}^d} \psi(\boldsymbol{\xi})\left[\int_0^t \partial_\zeta \det(\mathbb{J}(\boldsymbol{\xi},\zeta))\,\mathrm{d}\zeta\right]\mathrm{d}\boldsymbol{\xi}.\end{aligned}$$

Since by assumption $\det(\mathbb{J}(\boldsymbol{\xi},\zeta))$ is a polynomial of degree at most r in ζ, and since the quadrature $(\omega_l,\zeta_l)_{l\in\mathcal{L}}$ is exact for all polynomials of degree at most $\max(r-1,0)$, we infer that

$$\begin{aligned}\int_{\mathbb{R}^d} \varphi(\boldsymbol{x},t)\,\mathrm{d}\boldsymbol{x} - \int_{\mathbb{R}^d} \psi(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi} &= t\sum_{l\in\mathcal{L}} \omega_l \int_{\mathbb{R}^d} \psi(\boldsymbol{\xi})\partial_\zeta \det(\mathbb{J}(\boldsymbol{\xi},t\zeta_l))\,\mathrm{d}\boldsymbol{\xi}\\ &= t\sum_{l\in\mathcal{L}} \omega_l \int_{\mathbb{R}^d} \psi(\boldsymbol{\xi})(\nabla\cdot\boldsymbol{v}_{\mathrm{A}})(\boldsymbol{\Phi}(\boldsymbol{\xi},t\zeta_l),t\zeta_l)\det(\mathbb{J}(\boldsymbol{\xi},t\zeta_l))\,\mathrm{d}\boldsymbol{\xi}\\ &= t\sum_{l\in\mathcal{L}} \omega_l \int_{\mathbb{R}^d} \varphi(\boldsymbol{x},t)\nabla\cdot\boldsymbol{v}_{\mathrm{A}}(\boldsymbol{x},t\zeta_l)\,\mathrm{d}\boldsymbol{x},\end{aligned}$$

where we used Lemma 3.1. ∎

Remark 3.2. The statement of Lemma 3.2 is somewhat similar in spirit to Eq. (26) in Farhat et al. [14] or Eq. (7) in Guillard and Farhat [22]. The identity (3.3) will allow us to prove a statement that is known in the ALE literature as the Discrete Geometric Conservation Law (DGCL).

3.3. Arbitrary Lagrangian Eulerian formulation. Let $\boldsymbol{\Phi} : \mathbb{R}^d \times \mathbb{R}_+ \longrightarrow \mathbb{R}^d$ be a uniformly Lipschitz mapping as defined in §3.1 and let $[0,t^*]$ be the interval where the mapping $\mathbb{R}^d \ni \boldsymbol{\xi} \longmapsto \boldsymbol{\Phi}(\boldsymbol{\xi},t) \in \mathbb{R}^d$ is invertible for all $t \in [0,t^*]$. Let $\boldsymbol{v}_{\mathrm{A}}$ be the vector field defined in (3.1), i.e., $\boldsymbol{v}_{\mathrm{A}}(\boldsymbol{x},t) = \partial_t\boldsymbol{\Phi}(\boldsymbol{\Phi}_t^{-1}(\boldsymbol{x}),t)$, and let $\boldsymbol{u}$ be a weak solution to (1.1). The following result is the main motivation for the arbitrary Lagrangian Eulerian formulation that we are going to use in the paper.

LEMMA 3.3. *The following identity holds in the distribution sense (in time) over the interval $[0,t^*]$ for every function $\psi \in C_0^0(\mathbb{R}^d;\mathbb{R})$ (with the notation $\varphi(\boldsymbol{x},t) := \psi(\boldsymbol{\Phi}_t^{-1}(\boldsymbol{x}))$):*

$$\partial_t \int_{\mathbb{R}^d} \boldsymbol{u}(\boldsymbol{x},t)\varphi(\boldsymbol{x},t)\,\mathrm{d}\boldsymbol{x} = \int_{\mathbb{R}^d} \nabla\cdot(\boldsymbol{u}\otimes\boldsymbol{v}_{\mathrm{A}} - \boldsymbol{f}(\boldsymbol{u}))\varphi(\boldsymbol{x},t)\,\mathrm{d}\boldsymbol{x}. \tag{3.4}$$

Proof. Using the chain rule and Lemma 3.1, we have

$$\begin{aligned}\partial_t \int_{\mathbb{R}^d} \boldsymbol{u}(\boldsymbol{x},t)\varphi(\boldsymbol{x},t)\,\mathrm{d}\boldsymbol{x} &= \partial_t \int_{\mathbb{R}^d} \boldsymbol{u}(\boldsymbol{\Phi}_t(\boldsymbol{\xi}),t)\det(\mathbb{J}(\boldsymbol{\xi},t))\psi(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi}\\ &= \int_{\mathbb{R}^d} \big\{\partial_t(\boldsymbol{u}(\boldsymbol{\Phi}_t(\boldsymbol{\xi}),t))\det(\mathbb{J}(\boldsymbol{\xi},t)) + \boldsymbol{u}(\boldsymbol{\Phi}_t(\boldsymbol{\xi}),t)\partial_t(\det(\mathbb{J}(\boldsymbol{\xi},t)))\big\}\psi(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi}\\ &= \int_{\mathbb{R}^d} \big\{(\partial_t\boldsymbol{u})(\boldsymbol{\Phi}_t(\boldsymbol{\xi}),t) + \partial_t\boldsymbol{\Phi}(\boldsymbol{\xi},t)\cdot(\nabla\boldsymbol{u})(\boldsymbol{\Phi}_t(\boldsymbol{\xi}),t)\big\}\det(\mathbb{J}(\boldsymbol{\xi},t))\psi(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi}\\ &\qquad + \int_{\mathbb{R}^d} \boldsymbol{u}(\boldsymbol{\Phi}_t(\boldsymbol{\xi}),t)(\nabla\cdot\boldsymbol{v}_{\mathrm{A}})(\boldsymbol{\Phi}_t(\boldsymbol{\xi}),t)\det(\mathbb{J}(\boldsymbol{\xi},t))\psi(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi}.\end{aligned}$$

Then using (1.1) and the definition of the vector field $\boldsymbol{v}_{\mathrm{A}}$ yields

$$\begin{aligned}\partial_t \int_{\mathbb{R}^d} \boldsymbol{u}(\boldsymbol{x},t)\varphi(\boldsymbol{x},t)\,\mathrm{d}\boldsymbol{x} &= \int_{\mathbb{R}^d} -\nabla\cdot\boldsymbol{f}(\boldsymbol{u})(\boldsymbol{\Phi}_t(\boldsymbol{\xi}),t)\psi(\boldsymbol{\xi})\det(\mathbb{J}(\boldsymbol{\xi},t))\,\mathrm{d}\boldsymbol{\xi}\\ &+ \int_{\mathbb{R}^d} \big\{\boldsymbol{v}_{\mathrm{A}}(\boldsymbol{\Phi}_t(\boldsymbol{\xi}),t)\cdot(\nabla\boldsymbol{u})(\boldsymbol{\Phi}_t(\boldsymbol{\xi}),t) + (\nabla\cdot\boldsymbol{v}_{\mathrm{A}})(\boldsymbol{\Phi}_t(\boldsymbol{\xi}),t)\boldsymbol{u}(\boldsymbol{\Phi}_t(\boldsymbol{\xi}),t)\big\}\psi(\boldsymbol{\xi})\det(\mathbb{J}(\boldsymbol{\xi},t))\,\mathrm{d}\boldsymbol{\xi}\\ &= \int_{\mathbb{R}^d} \big\{-\nabla\cdot\boldsymbol{f}(\boldsymbol{u})(\boldsymbol{\Phi}_t(\boldsymbol{\xi}),t) + \nabla\cdot(\boldsymbol{u}\otimes\boldsymbol{v}_{\mathrm{A}})(\boldsymbol{\Phi}_t(\boldsymbol{\xi}),t)\big\}\psi(\boldsymbol{\xi})\det(\mathbb{J}(\boldsymbol{\xi},t))\,\mathrm{d}\boldsymbol{\xi}.\end{aligned}$$

We conclude by making the change of variable $\boldsymbol{x} = \boldsymbol{\Phi}(\boldsymbol{\xi},t)$. ∎

We now state a result regarding the notion of entropy solution in the ALE framework. The proof of this result is similar to that of Lemma 3.3 and is therefore omitted for brevity.

LEMMA 3.4. *Let $(\eta,\boldsymbol{q})$ be an entropy pair for (1.1). The following inequality holds in the distribution sense (in time) over the interval $[0,t^*]$ for every non-negative function $\psi\in C_0^0(\mathbb{R}^d;\mathbb{R}_+)$ (with the notation $\varphi(\boldsymbol{x},t) := \psi(\boldsymbol{\Phi}_t^{-1}(\boldsymbol{x}))$):*

$$\partial_t \int_{\mathbb{R}^d} \eta(\boldsymbol{u}(\boldsymbol{x},t))\varphi(\boldsymbol{x},t)\,\mathrm{d}\boldsymbol{x} \le \int_{\mathbb{R}^d} \nabla\cdot(\eta(\boldsymbol{u})\boldsymbol{v}_{\mathrm{A}} - \boldsymbol{q}(\boldsymbol{u}))\varphi(\boldsymbol{x},t)\,\mathrm{d}\boldsymbol{x}. \tag{3.5}$$

4. The Arbitrary Lagrangian Eulerian algorithm.

We describe in this section the ALE algorithm to approximate the solution of (1.1). We use continuous finite elements and explicit time stepping. We are going to use two different discrete settings: one for the mesh motion and one for the approximation of (1.1).

4.1. Geometric finite elements and mesh.

Let $(\mathcal{T}_h^0)_{h>0}$ be a shape-regular sequence of matching meshes. The symbol ^{0} in $\mathcal{T}_h^0$ refers to the initial configuration of the meshes. The meshes will deform over time, in a way that has yet to be defined, and we are going to use the symbol ^{n} to say that $\mathcal{T}_h^n$ is the mesh at time t^n for a given $h>0$. We assume that the elements in the mesh cells are generated from a finite number of reference elements denoted $\widehat{K}_1,\ldots,\widehat{K}_\varpi$. For instance, $\mathcal{T}_h^0$ could be composed of a combination of triangles and parallelograms in two space dimensions ($\varpi=2$ in this case); the mesh $\mathcal{T}_h^0$ could also be composed of a combination of tetrahedra, parallelepipeds, and triangular prisms in three space dimensions ($\varpi=3$ in this case). The diffeomorphism mapping $\widehat{K}_r$ to an arbitrary element $K\in\mathcal{T}_h^n$ is denoted $T_K^n : \widehat{K}_r \longrightarrow K$ and its Jacobian matrix is denoted $\mathbb{J}_K^n$, $1\le r\le\varpi$. We now introduce a set of reference Lagrange finite elements $\{(\widehat{K}_r,\widehat{P}_r^{\mathrm{geo}},\widehat{\Sigma}_r^{\mathrm{geo}})\}_{1\le r\le\varpi}$ (the index $r\in\{1{:}\varpi\}$ will be omitted in the rest of the paper to alleviate the notation). Letting $n_{\mathrm{sh}}^{\mathrm{geo}} :=$

$\dim \widehat{P}^{\mathrm{geo}}$, we denote by $\{\widehat{\boldsymbol{a}}_i\}_{i\in\{1:n_{\mathrm{sh}}^{\mathrm{geo}}\}}$ and $\{\widehat{\theta}_i^{\mathrm{geo}}\}_{i\in\{1:n_{\mathrm{sh}}^{\mathrm{geo}}\}}$ the Lagrange nodes of $\widehat{K}$ and the associated Lagrange shape functions.

The sole purpose of the geometric reference element $\{(\widehat{K},\widehat{P}^{\mathrm{geo}},\widehat{\Sigma}^{\mathrm{geo}})$ is to construct the geometric transformation T_K^n as we now explain. Let $\{\boldsymbol{a}_i^n\}_{i\in\{1:I^{\mathrm{geo}}\}}$ be the collection of all the Lagrange nodes in the mesh $\mathcal{T}_h^n$, which we organize in cells by means of the geometric connectivity array $\mathsf{j}^{\mathrm{geo}}:\mathcal{T}_h^n\times\{1:n_{\mathrm{sh}}^{\mathrm{geo}}\}\longrightarrow\{1:I^{\mathrm{geo}}\}$ (assumed to be independent of the time index n). Given a mesh cell $K\in\mathcal{T}_h^n$, the connectivity array is defined such that $\{\boldsymbol{a}^n_{\mathsf{j}^{\mathrm{geo}}(i,K)}\}_{i\in\{1:n_{\mathrm{sh}}^{\mathrm{geo}}\}}$ is the set of the Lagrange nodes describing K^n. More precisely, upon defining the geometric transformation $T_K^n:\widehat{K}\longrightarrow K$ at time t^n by

$$T_K^n(\widehat{\boldsymbol{x}})=\sum_{i\in\{1:n_{\mathrm{sh}}^{\mathrm{geo}}\}}\boldsymbol{a}^n_{\mathsf{j}^{\mathrm{geo}}(i,K)}\widehat{\theta}_i^{\mathrm{geo}}(\widehat{\boldsymbol{x}}) \tag{4.1}$$

we have $K:=T_K^n(\widehat{K})$. In other words the geometric transformation is *fully described by the motion of geometric Lagrange nodes.* We finally recall that constructing the Jacobian matrix $\mathbb{J}_K^n$ from (4.1) is an elementary operation for any finite element code.

4.2. Approximating finite elements. We now introduce a set of reference finite elements $\{(\widehat{K}_r,\widehat{P}_r,\widehat{\Sigma}_r)\}_{1\le r\le\varpi}$ which we are going to use to construct an approximate solution to (1.1) (the index $r\in\{1:\varpi\}$ will be omitted in the rest of the paper to alleviate the notation). The shape functions on the reference element are denoted $\{\widehat{\theta}_i\}_{i\in\{1:n_{\mathrm{sh}}\}}$. We assume that the basis $\{\widehat{\theta}_i\}_{i\in\{1:n_{\mathrm{sh}}\}}$ has the following key properties:

$$\widehat{\theta}_i(\boldsymbol{x})\ge 0,\qquad\sum_{i\in\{1:n_{\mathrm{sh}}\}}\widehat{\theta}_i(\widehat{\boldsymbol{x}})=1,\qquad\forall\widehat{\boldsymbol{x}}\in\widehat{K}. \tag{4.2}$$

These properties hold for linear Lagrange elements. It holds true also for Bernstein-Bezier finite elements, see e.g., Lai and Schumaker [27, Chap. 2], Ainsworth [1].

Given the mesh $\mathcal{T}_h^n$, we denote by D^n the computational domain generated by $\mathcal{T}_h^n$ and define the scalar-valued space

$$P(\mathcal{T}_h^n):=\{v\in\mathcal{C}^0(D^n;\mathbb{R})\mid v_{|K}\circ T_K^n\in\widehat{P},\ \forall K\in\mathcal{T}_h^n\}, \tag{4.3}$$

where $\widehat{P}$ is the reference polynomial space. We also introduce the vector-valued spaces

$$\boldsymbol{P}_d(\mathcal{T}_h^n):=[P(\mathcal{T}_h^n)]^d,\quad\text{and}\quad\boldsymbol{P}_m(\mathcal{T}_h^n):=[P(\mathcal{T}_h^n)]^m. \tag{4.4}$$

We are going to approximate the ALE velocity in $\boldsymbol{P}_d(\mathcal{T}_h^n)$ and the solution of (1.1) in $\boldsymbol{P}_m(\mathcal{T}_h^n)$. The global shape functions in $P(\mathcal{T}_h^n)$ are denoted by $\{\psi_i^n\}_{i\in\{1:I\}}$. Recall that these functions form a basis of $P(\mathcal{T}_h^n)$. Let $\mathsf{j}:\mathcal{T}_h^n\times\{1:n_{\mathrm{sh}}\}\longrightarrow\{1:I\}$ be the connectivity array, which we assume to be independent of n. This array is defined such that

$$\psi^n_{\mathsf{j}(i,K)}(\boldsymbol{x})=\widehat{\theta}_i((T_K^n)^{-1}(\boldsymbol{x})),\quad\forall i\in\{1:n_{\mathrm{sh}}\},\ \forall K\in\mathcal{T}_h^n. \tag{4.5}$$

This definition together with (4.2) implies that

$$\psi_i^n(\boldsymbol{x})\ge 0,\qquad\sum_{i\in\{1:I\}}\psi_i^n(\boldsymbol{x})=1,\qquad\forall\boldsymbol{x}\in\mathbb{R}^d. \tag{4.6}$$

We denote by S_i^n the support of ψ_i^n and by $|S_i^n|$ the measure of S_i, $i \in \{1{:}I\}$. We also define $S_{ij}^n := S_i^n \cap S_j^n$ the intersection of the two supports S_i^n and S_j^n. Let E be a union of cells in $\mathcal{T}_h^n$; we define $\mathcal{I}(E) := \{j \in \{1{:}I\} \mid |S_j^n \cap E| \neq 0\}$ the set that contains the indices of all the shape functions whose support on E is of nonzero measure. Note that the index set $\mathcal{I}(E)$ does not depend on the time index n since we have assumed that the connectivity of the degrees of freedom is fixed once for all. We are going to regularly invoke $\mathcal{I}(K)$ and $\mathcal{I}(S_i^n)$ and the partition of unity property: $\sum_{i\in\mathcal{I}(K)} \psi_i^n(\boldsymbol{x}) = 1$ for all $\boldsymbol{x} \in K$.

LEMMA 4.1. *For all $K \in \mathcal{T}_h^n$, all $\boldsymbol{x} \in K$, and all $\boldsymbol{v}_h := \sum_{i\in\{1:I\}} \mathbf{V}_i\psi_i^n \in \boldsymbol{P}_m(\mathcal{T}_h^n)$, $\boldsymbol{v}_h(\boldsymbol{x})$ is in the convex hull of $(\mathbf{V}_i)_{i\in\mathcal{I}(K)}$ (henceforth denoted $\operatorname{conv}(\mathbf{V}_i)_{i\in\mathcal{I}(K)}$). Moreover for any convex set A in $\mathbb{R}^m$, we have*

$$\big((\mathbf{V}_i)_{i\in\mathcal{I}(K)} \in A\big) \Rightarrow (\boldsymbol{v}_h(\boldsymbol{x}) \in A,\ \forall \boldsymbol{x} \in K)\,. \tag{4.7}$$

Proof. The positivity and partition of unity assumption (4.6) and the definition $\boldsymbol{v}_h(\boldsymbol{x}) = \sum_{i\in\mathcal{I}(K)} \mathbf{V}_i\psi_i^n(\boldsymbol{x})$ implies that $\boldsymbol{v}_h(\boldsymbol{x})$ is a convex combination of $(\mathbf{V}_i)_{i\in\mathcal{I}(K)}$, whence the conclusion. The statement (4.7) follows readily since the convexity assumption on A implies that $\operatorname{conv}(\mathbf{V}_i)_{i\in\mathcal{I}(K)} \subset A$. ∎

Let $\mathcal{M}^n \in \mathbb{R}^{I\times I}$ be the consistent mass matrix with entries $\int_{S_{ij}^n} \psi_i^n(\boldsymbol{x})\psi_j^n(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}$, and let $\mathcal{M}^{L,n}$ be the diagonal lumped mass matrix with entries

$$m_i^n := \int_{S_i^n} \psi_i^n(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}. \tag{4.8}$$

The partition of unity property implies that $m_i^n = \sum_{j\in\mathcal{I}(S_i^n)} \int \psi_j^n(\boldsymbol{x})\psi_i^n(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}$, i.e., the entries of $\mathcal{M}^{L,n}$ are obtained by summing the rows of $\mathcal{M}^n$. Note that the positivity assumption (4.6) implies that $m_i^n > 0$ for any $i \in \{1{:}I\}$.

4.3. The ALE algorithm, version 1. Let $\mathcal{T}_h^0$ be the mesh at the initial time $t = 0$. Let $(\mathfrak{m}_i^0)_{i\in\{1:I\}}$ be the approximations of the mass of the shape functions at time t^0 defined by $\mathfrak{m}_i^0 = m_i^0 := \int_{\mathbb{R}^d} \psi_i^0(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}$. Let $\boldsymbol{u}_{h0} := \sum_{i\in\{1:I\}} \mathbf{U}_i^0\psi_i^0 \in \boldsymbol{P}_m(\mathcal{T}_h^0)$ be a reasonable approximation of the initial data $\boldsymbol{u}_0$ (we shall make a more precise statement later).

Let $\mathcal{T}_h^n$ be the mesh at time t^n, $(\mathfrak{m}_i^n)_{1\le i\le I}$ be the approximations of the mass of the shape functions at time t^n, and $\boldsymbol{u}_h^n := \sum_{i\in\{1:I\}} \mathbf{U}_i^n\psi_i^n \in \boldsymbol{P}_m(\mathcal{T}_h^n)$ be the approximation of $\boldsymbol{u}$ at time t^n. We denote by $\mathfrak{M}^{L,n}$ the approximate lumped matrix, i.e., $\mathfrak{M}_{ij}^{L,n} = \mathfrak{m}_i^n\delta_{ij}$. We now make the assumption that the given ALE velocity field is a member of $\boldsymbol{P}_d(\mathcal{T}_h^n)$, i.e., $\boldsymbol{w}^n = \sum_{i\in\{1:I\}} \mathbf{W}_i^n\psi_i^n \in \boldsymbol{P}_d(\mathcal{T}_h^n)$. Then the Lagrange nodes of the mesh are moved by using

$$\boldsymbol{a}_i^{n+1} = \boldsymbol{a}_i^n + \tau\boldsymbol{w}^n(\boldsymbol{a}_i^n). \tag{4.9}$$

This fully defines the mesh $\mathcal{T}_h^{n+1}$ as explained at the end of §4.1. We now estimate the mass of the shape function ψ_i^{n+1}. Of course we could use $m_i^{n+1} = \int_{\mathbb{R}^d} \psi_i^{n+1}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}$, this option will be explored in §4.5, but to make the method easier to extend with higher-order strong stability preserving (SSP) time stepping techniques, we define $\mathfrak{m}_i^{n+1}$ by approximating (3.3) with a first-order quadrature rule,

$$\mathfrak{m}_i^{n+1} = \mathfrak{m}_i^n + \tau \int_{S_i^n} \psi_i^n(\boldsymbol{x})\nabla\cdot\boldsymbol{w}^n(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}. \tag{4.10}$$

Taking inspiration from (3.4), we propose to compute $\boldsymbol{u}_h^{n+1}$ by using the following explicit technique:

$$\frac{\mathfrak{m}_i^{n+1}\mathbf{U}_i^{n+1}-\mathfrak{m}_i^n\mathbf{U}_i^n}{\tau}-\sum_{j\in\mathcal{I}(S_i^n)}d_{ij}^n\mathbf{U}_j^n + \int_{\mathbb{R}^d}\nabla\cdot\Big(\sum_{j\in\{1:I\}}(\boldsymbol{f}(\mathbf{U}_j^n)-\mathbf{U}_j^n\otimes\mathbf{W}_j^n)\psi_j^n(\boldsymbol{x})\Big)\psi_i^n(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}=0, \tag{4.11}$$

where $\boldsymbol{u}_h^{n+1}:=\sum_{i\in\{1:I\}}\mathbf{U}_i^{n+1}\psi_i^{n+1}\in\boldsymbol{P}_m(\mathcal{T}_h^{n+1})$. Notice that we have replaced the consistent mass matrix by an approximation of the lumped mass matrix to approximate the time derivative. The coefficient d_{ij}^n is an artificial viscosity for the pair of degrees of freedom (i,j) that will be identified by proceeding as in Guermond and Popov [20]. We henceforth assume that $d_{ij}^n=0$ if $j\notin\mathcal{I}(S_i^n)$ and

$$d_{ij}^n\geq 0,\ \text{if } i\neq j,\quad d_{ij}^n=d_{ji}^n,\quad\text{and}\quad d_{ii}:=\sum_{i\neq j\in\mathcal{I}(S_i^n)}-d_{ji}^n. \tag{4.12}$$

The entire process is described in Algorithm 1.

Let us reformulate (4.11) in a form that is more suitable for computations. Let us introduce the vector-valued coefficients

$$\boldsymbol{c}_{ij}^n:=\int_{S_i^n}\nabla\psi_j^n(\boldsymbol{x})\psi_i^n(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}. \tag{4.13}$$

We define the unit vector $\boldsymbol{n}_{ij}^n:=\frac{\boldsymbol{c}_{ij}^n}{\|\boldsymbol{c}_{ij}^n\|_{\ell^2}}$. Then we can rewrite (4.11) as follows

$$\frac{\mathfrak{m}_i^{n+1}\mathbf{U}_i^{n+1}-\mathfrak{m}_i^n\mathbf{U}_i^n}{\tau}+\sum_{j\in\mathcal{I}(S_i^n)}(\boldsymbol{f}(\mathbf{U}_j^n)-\mathbf{U}_j^n\otimes\mathbf{W}_j^n)\cdot\boldsymbol{c}_{ij}^n-d_{ij}^n\mathbf{U}_j^n=0. \tag{4.14}$$

It will be shown in the proof of Theorem 5.2 that an admissible choice for d_{ij}^n is

$$d_{ij}^n=\max(\lambda_{\max}(\boldsymbol{g}_j^n,\boldsymbol{n}_{ij}^n,\mathbf{U}_i^n,\mathbf{U}_j^n)\|\boldsymbol{c}_{ij}^n\|_{\ell^2},\lambda_{\max}(\boldsymbol{g}_i^n,\boldsymbol{n}_{ji}^n,\mathbf{U}_j^n,\mathbf{U}_i^n)\|\boldsymbol{c}_{ji}^n\|_{\ell^2}). \tag{4.15}$$

where $\lambda_{\max}(\boldsymbol{g}_j^n,\boldsymbol{n}_{ij}^n,\mathbf{U}_i^n,\mathbf{U}_j^n)$ is the largest wave speed in the following one-dimensional Riemann problem:

$$\partial_t\boldsymbol{v}+\partial_x(\boldsymbol{g}_j^n(\boldsymbol{v})\cdot\boldsymbol{n}_{ij}^n)=0,\quad(x,t)\in\mathbb{R}\times\mathbb{R}_+,\quad\boldsymbol{v}(x,0)=\begin{cases}\mathbf{U}_i^n & \text{if } x<0\\ \mathbf{U}_j^n & \text{if } x>0.\end{cases} \tag{4.16}$$

where we have defined the flux $\boldsymbol{g}_j^n(\boldsymbol{v}):=\boldsymbol{f}(\boldsymbol{v})-\boldsymbol{v}\otimes\mathbf{W}_j^n$.

Remark 4.1. (Fastest wave speed) The fastest wave speed in (4.16) can be obtained by estimating the fastest wave speed in the Riemann problem (2.2) with the flux $\boldsymbol{f}(\boldsymbol{v})\cdot\boldsymbol{n}_{ij}^n$ and initial data $(\mathbf{U}_i^n,\mathbf{U}_j^n)$. Let $\lambda_L(\boldsymbol{f},\boldsymbol{n}_{ij}^n,\mathbf{U}_i^n,\mathbf{U}_j^n)$ and $\lambda_R(\boldsymbol{f},\boldsymbol{n}_{ij}^n,\mathbf{U}_i^n,\mathbf{U}_j^n)$ be the speed of the leftmost and rightmost waves in (2.2), respectively. Then

$$\lambda_{\max}(\boldsymbol{g}_j^n,\boldsymbol{n}_{ij}^n,\mathbf{U}_i^n,\mathbf{U}_j^n)=\max(|\lambda_L(\boldsymbol{f},\boldsymbol{n}_{ij}^n,\mathbf{U}_i^n,\mathbf{U}_j^n)-\mathbf{W}_j^n\cdot\boldsymbol{n}_{ij}^n|, |\lambda_R(\boldsymbol{f},\boldsymbol{n}_{ij}^n,\mathbf{U}_i^n,\mathbf{U}_j^n)-\mathbf{W}_j^n\cdot\boldsymbol{n}_{ij}^n|). \tag{4.17}$$

A very fast algorithm to compute $\lambda_L(\boldsymbol{f},\boldsymbol{n}_{ij}^n,\mathbf{U}_i^n,\mathbf{U}_j^n)$ and $\lambda_R(\boldsymbol{f},\boldsymbol{n}_{ij}^n,\mathbf{U}_i^n,\mathbf{U}_j^n)$ for the compressible Euler equations is described in Guermond and Popov [19]; see also Toro [34].

Algorithm 1

Require: $\boldsymbol{u}_h^0$ and $\mathfrak{M}^{L,0}$
1: **while** $t^n < T$ **do**
2: Use CFL condition to estimate τ.
3: **if** $t^n + \tau > T$ **then**
4: $\tau \leftarrow T - t^n$
5: **end if**
6: Estimate/choose $\boldsymbol{w}^n$ and make sure that the transformation $\boldsymbol{\Phi}_t$ defined in (4.25) is invertible over the interval $[t^n, t^{n+1}]$.
7: Move mesh from t^n to t^{n+1} using (4.9).
8: Compute $\mathfrak{m}_i^{n+1}$, see (4.10). Check $\mathfrak{m}_i^{n+1} > 0$; otherwise, go to step 6, reduce τ.
9: Compute $\boldsymbol{c}_{ij}^n$ as in (4.13).
10: Compute d_{ij}^n, see (4.15) and (4.12).
11: Check $1 - \sum_{i \neq j \in \mathcal{I}(S_i^n)} 2d_{ij}^n \frac{\tau}{\mathfrak{m}_i^{n+1}}$ positive. Otherwise, go to step 6 and reduce τ.
12: Compute $\boldsymbol{u}_h^{n+1}$ by using (4.14).
13: $t^n \leftarrow t^n + \tau$
14: **end while**

Since it will be important to compare $\mathbf{U}_j^{n+1}$ and $\mathbf{U}_j^n$ to establish the invariant domain property, we rewrite the scheme in a form that is more suitable for this purpose.

LEMMA 4.2 (Non-conservative form). *The scheme* (4.11) *is equivalent to*

$$\mathfrak{m}_i^{n+1} \frac{\mathbf{U}_i^{n+1} - \mathbf{U}_i^n}{\tau} = \sum_{j \in \mathcal{I}(S_i^n)} ((\mathbf{U}_j^n - \mathbf{U}_i^n) \otimes \mathbf{W}_j^n - \boldsymbol{f}(\mathbf{U}_j^n)) \cdot \boldsymbol{c}_{ij}^n + d_{ij}^n \mathbf{U}_j^n, \tag{4.18}$$

Proof. We rewrite (4.14) as follows:

$$\mathfrak{m}_i^{n+1} \frac{\mathbf{U}_i^{n+1} - \mathbf{U}_i^n}{\tau} + \frac{\mathfrak{m}_i^{n+1} - \mathfrak{m}_i^n}{\tau} \mathbf{U}_i^n = \sum_{j \in \mathcal{I}(S_i^n)} (\mathbf{U}_j^n \otimes \mathbf{W}_j^n - \boldsymbol{f}(\mathbf{U}_j^n)) \cdot \boldsymbol{c}_{ij}^n + d_{ij}^n \mathbf{U}_j^n,$$

Then, recalling the expression $\boldsymbol{w}^n = \sum_{i \in \{1:I\}} \mathbf{W}_i^n \psi_i^n$, and using (4.10), we infer that $\mathfrak{m}_i^{n+1} = \mathfrak{m}_i^n + \tau \sum_{j \in \mathcal{I}(S_i^n)} \mathbf{W}_j^n \cdot \boldsymbol{c}_{ij}^n$, which in turn implies that

$$(\mathfrak{m}_i^{n+1} - \mathfrak{m}_i^n) \mathbf{U}_i^n = \tau \mathbf{U}_i^n \sum_{j \in \mathcal{I}(S_i^n)} \mathbf{W}_j^n \cdot \boldsymbol{c}_{ij}^n = \tau \sum_{j \in \mathcal{I}(S_i^n)} (\mathbf{U}_i^n \otimes \mathbf{W}_j^n) \cdot \boldsymbol{c}_{ij}^n,$$

whence the result. ∎

Remark 4.2. (Other discretizations) Note that the method for computing the artificial diffusion is quite generic, i.e., it is not specific to continuous finite elements. The above method can be applied to any type of discretization that can be put into the form (4.14).

4.4. Continuous mesh motion. We introduce in this section some technicalities regarding the mesh motion that will be used in the second version of the algorithm and which will be described in §4.5. Our main motivation is to replace the approximate mass conservation (4.10) by the exact quadrature (3.3). For this purpose, we need to consider the continuous motion of the mesh over the time interval $[t^n, t^{n+1}]$.

Given a mesh $\mathcal{T}_h^n$ we denote by D^n the computational domain generated by $\mathcal{T}_h^n$. Then using the standard constructions of continuous finite element spaces, we define a new scalar-valued space based on the geometric Lagrange finite elements $\{(\widehat{K}_r, \widehat{P}_r^{\rm geo}, \widehat{\Sigma}_r^{\rm geo})\}_{1\le r\le \varpi}$:

$$P^{\rm geo}(\mathcal{T}_h^n) = \{v \in \mathcal{C}^0(D^n;\mathbb{R}) \mid v_{|K}\circ T_K^n \in \widehat{P}^{\rm geo},\ \forall K \in \mathcal{T}_h^n\}, \tag{4.19}$$

where $\widehat{P}^{\rm geo}$ is the reference polynomial space defined on $\widehat{K}$ (note that the index r has been omitted). We also introduce the vector-valued spaces

$$\boldsymbol{P}^{\rm geo}(\mathcal{T}_h^n) := [P^{\rm geo}(\mathcal{T}_h^n)]^d. \tag{4.20}$$

We denote by $\{\psi_i^{{\rm geo},n}\}_{i\in\{1:I\}}$ the global shape functions in $P^{\rm geo}(\mathcal{T}_h^n)$. Recall that $\{\psi_i^{{\rm geo},n}\}_{i\in\{1:I\}}$ is a basis of $P^{\rm geo}(\mathcal{T}_h^n)$ and

$$\psi^{{\rm geo},n}_{{\rm j}^{\rm geo}(i,K)}(\boldsymbol{x}) = \widehat{\theta}_i^{\rm geo}((T_K^n)^{-1}(\boldsymbol{x})), \quad \forall i \in \{1{:}n_{\rm sh}\},\ \forall K \in \mathcal{T}_h^n,\ \forall \boldsymbol{x} \in K. \tag{4.21}$$

The key difference with version 1 of the algorithm is that now we are going to construct exactly the mapping $\boldsymbol{\Phi}_t : \mathcal{T}_h^n \longrightarrow \mathcal{T}_h(t)$ by using $\boldsymbol{P}^{\rm geo}(\mathcal{T}_h^n)$, and we are going to assume that the given ALE velocity is a member of $\boldsymbol{P}^{\rm geo}(\mathcal{T}_h^n)$ instead of $\boldsymbol{P}_d(\mathcal{T}_h^n)$ as we did in §4.3. Let $\boldsymbol{w}^n = \sum_{i\in\mathcal{I}(K)} \mathbf{W}_i^{{\rm geo},n}\psi_i^{{\rm geo},n} \in \boldsymbol{P}^{\rm geo}(\mathcal{T}_h^n)$ be the ALE velocity.

Let us construct the associated transformation $\boldsymbol{\Phi}_t : \mathbb{R}^d \longrightarrow \mathbb{R}^d$ and velocity field $\boldsymbol{v}_{\rm A}$ for any $t \in [t^n, t^{n+1}]$. We define a continuous deformation of the mesh over the time interval $[t^n, t^{n+1}]$ by moving the nodes $\{\boldsymbol{a}_i^n\}_{i\in\{1:I^{\rm geo}\}}$ as follows:

$$\boldsymbol{a}_i(t) = \boldsymbol{a}_i^n + t\mathbf{W}_i^{{\rm geo},n}, \qquad t \in [t^n, t^{n+1}]. \tag{4.22}$$

This rule completely defines the mesh $\mathcal{T}_h(t)$ owing to the definition of the geometric transformation $T_K(t) : \widehat{K} \longrightarrow K$, with $T_K(t)(\widehat{\boldsymbol{x}}) = \sum_{i\in\{1:n_{\rm sh}^{\rm geo}\}} \boldsymbol{a}_{{\rm j}^{\rm geo}(i,K)}(t)\widehat{\theta}_i^{\rm geo}(\widehat{\boldsymbol{x}})$, for all $K \in \mathcal{T}_h(t)$, see (4.1). The shape functions of $P^{\rm geo}(\mathcal{T}_h(t))$ and $P(\mathcal{T}_h(t))$ are defined as usual by setting

$$\varphi^{\rm geo}_{{\rm j}(i,K)}(\boldsymbol{x},t) := \widehat{\theta}_i^{\rm geo}((T_K(t))^{-1}(\boldsymbol{x})), \quad \forall \boldsymbol{x} \in K,\ \forall K \in \mathcal{T}_h(t),\ \forall i \in \{1{:}I^{\rm geo}\} \tag{4.23}$$

$$\varphi_{{\rm j}(i,K)}(\boldsymbol{x},t) := \widehat{\theta}_i((T_K(t))^{-1}(\boldsymbol{x})), \quad \forall \boldsymbol{x} \in K,\ \forall K \in \mathcal{T}_h(t),\ \forall i \in \{1{:}I\}. \tag{4.24}$$

We recall that $\mathcal{T}_h^{n+1} := \mathcal{T}_h(t^{n+1})$. Notice that $\varphi_i^{\rm geo}(\cdot,t^n) = \psi_i^{{\rm geo},n}$ and $\varphi_i^{\rm geo}(\cdot,t^{n+1}) = \psi_i^{{\rm geo},n+1}$ for any $i \in \{1{:}I^{\rm geo}\}$, and $\varphi_i(\cdot,t^n) = \psi_i^n$ and $\varphi_i(\cdot,t^{n+1}) = \psi_i^{n+1}$ for any $i \in \{1{:}I\}$.

For any $t \in [t^n, t^{n+1}]$ we now define the mapping $\boldsymbol{\Phi}_t : \mathcal{T}_h^n \longrightarrow \mathcal{T}_h(t)$ by

$$\boldsymbol{\Phi}_t(\boldsymbol{\xi})_{|K} = \sum_{i\in\{1:n_{\rm sh}^{\rm geo}\}} \boldsymbol{a}_{{\rm j}^{\rm geo}(i,K)}(t)\psi^{{\rm geo},n}_{{\rm j}^{\rm geo}(i,K)}(\boldsymbol{\xi}), \qquad \forall \boldsymbol{\xi} \in K,\ \forall \in \mathcal{T}_h^n. \tag{4.25}$$

Lemma 4.3. *The following properties hold for any $K \in \mathcal{T}_h^n$, any $t \in [t^n, t^{n+1}]$ and any $i \in \{1{:}I\}$:*

$$T_K(t) = \boldsymbol{\Phi}_t \circ T_K^n, \tag{4.26}$$

$$\varphi_i(\boldsymbol{\Phi}_t(\boldsymbol{\xi}),t) = \psi_i^n(\boldsymbol{\xi}), \quad \varphi_i^{\rm geo}(\boldsymbol{\Phi}_t(\boldsymbol{\xi}),t) = \psi_i^{{\rm geo},n}(\boldsymbol{\xi}), \quad \forall \boldsymbol{\xi} \in K \in \mathcal{T}_h^n. \tag{4.27}$$

$$\boldsymbol{v}_{\rm A}(\boldsymbol{x},t) = \sum_{i\in\{1:I\}} \mathbf{W}_i^{{\rm geo},n}\varphi_i^{\rm geo}(\boldsymbol{x},t), \qquad \forall t \in [t^n, t^{n+1}],\ \forall \boldsymbol{x} \in \mathbb{R}^d. \tag{4.28}$$

Proof. Let us observe first that the definition (4.25) together with the definitions of $T_K(t)$ and $\psi^{\mathrm{geo},n}_{\mathrm{j}^{\mathrm{geo}}(i,K)} = \widehat{\theta}^{\mathrm{geo}}_i \circ (T^n_K)^{-1}$ implies that

$$\boldsymbol{\Phi}_t(\boldsymbol{\xi})_{|K} = \sum_{i\in\{1:\, n^{\mathrm{geo}}_{\mathrm{sh}}\}} \boldsymbol{a}_{\mathrm{j}^{\mathrm{geo}}(i,K)}(t)\psi^{\mathrm{geo},n}_{\mathrm{j}^{\mathrm{geo}}(i,K)}(\boldsymbol{\xi}) = \sum_{i\in\{1:\, n^{\mathrm{geo}}_{\mathrm{sh}}\}} \boldsymbol{a}_{\mathrm{j}^{\mathrm{geo}}(i,K)}(t)\widehat{\theta}^{\mathrm{geo}}_i \circ (T^n_K)^{-1}(\boldsymbol{\xi}),$$

which implies that $\boldsymbol{\Phi}_{t|K} = T_K(t) \circ (T^n_K)^{-1}$. This proves the first statement. Second, the definition of the shape functions (4.24) together with the above result implies that

$$\varphi_{\mathrm{j}(i,K)}(\boldsymbol{x},t) = \widehat{\theta}_i((T_K(t))^{-1}(\boldsymbol{x})) = \widehat{\theta}_i((\boldsymbol{\Phi}_t \circ T^n_K)^{-1}(\boldsymbol{x})) = \widehat{\theta}_i((T^n_K)^{-1} \circ (\boldsymbol{\Phi}_t)^{-1}(\boldsymbol{x})).$$

This proves that $\varphi_{\mathrm{j}(i,K)}(\boldsymbol{\Phi}_t(\boldsymbol{\xi}),t) = \psi^n_{\mathrm{j}(i,K)}(\boldsymbol{\xi})$ for every $\boldsymbol{\xi} \in K \in \mathcal{T}^n_h$. Proceed similarly to prove $\varphi^{\mathrm{geo}}_{\mathrm{j}(i,K)}(\boldsymbol{\Phi}_t(\boldsymbol{\xi}),t) = \psi^{\mathrm{geo},n}_{\mathrm{j}(i,K)}(\boldsymbol{\xi})$. Now let us compute $\partial_t\boldsymbol{\Phi}$. Using the definition of the motion of the nodes (4.22) and the definition of $\boldsymbol{\Phi}$, (4.25), we infer that

$$\partial_t\boldsymbol{\Phi}(\boldsymbol{\xi},t)_{|K} = \sum_{i\in\{1:\, n^{\mathrm{geo}}_{\mathrm{sh}}\}} \mathbf{W}^{\mathrm{geo},n}_{\mathrm{j}^{\mathrm{geo}}(i,K)}\psi^{\mathrm{geo},n}_{\mathrm{j}^{\mathrm{geo}}(i,K)}(\boldsymbol{\xi}) = \boldsymbol{w}^n_{|K}.$$

Hence the definition of $\boldsymbol{v}_{\mathrm{A}}$ gives

$$\boldsymbol{v}_{\mathrm{A}}(\boldsymbol{x},t) = \partial_t\boldsymbol{\Phi}(\boldsymbol{\Phi}^{-1}_t(\boldsymbol{x}),t) = \boldsymbol{w}^n(\boldsymbol{\Phi}^{-1}_t(\boldsymbol{x})) = \sum_{i\in\{1:\, I^{\mathrm{geo}}\}} \mathbf{W}^{\mathrm{geo},n}_i\psi^{\mathrm{geo},n}_i(\boldsymbol{\Phi}^{-1}_t(\boldsymbol{x})).$$

We then conclude by invoking (4.27), i.e., $\varphi^{\mathrm{geo}}_i(\boldsymbol{x},t) = \psi^{\mathrm{geo},n}_i(\boldsymbol{\Phi}^{-1}_t(\boldsymbol{x}))$. ∎

Before writing the complete algorithm we need to make a change of basis to express the ALE velocity in the approximation basis. We further assume that

$$\widehat{P}^{\mathrm{geo}} \subset \widehat{P}. \tag{4.29}$$

This assumption implies that $\boldsymbol{P}^{\mathrm{geo}}(\mathcal{T}^n_h) \subset \boldsymbol{P}_d(\mathcal{T}^n_h)$; hence there is a sparse matrix $\mathbb{B}$, independent of n, such that $\psi^{\mathrm{geo},n}_i = \sum_{j\in s(i)} \mathbb{B}_{ij}\psi^n_j$, were $s(i)$ is a sparse set of indices for any $i \in \{1:I^{\mathrm{geo}}\}$. We then define

$$\mathbf{W}^n_j := \sum_{\{i\,\mid\, j\in s(i)\}} \mathbb{B}_{ij}\mathbf{W}^{\mathrm{geo},n}_i, \tag{4.30}$$

which, owing to (4.28), gives the following alternative representation of $\boldsymbol{v}_{\mathrm{A}}$:

$$\boldsymbol{v}_{\mathrm{A}}(\boldsymbol{x},t) = \sum_{i\in\{1:\, I\}} \mathbf{W}^n_i\psi^n_i(\boldsymbol{\Phi}^{-1}_t(\boldsymbol{x})). \tag{4.31}$$

4.5. The ALE algorithm, version 2. It may look odd to some readers that in version 1 of the algorithm we update the mass of the shape function ψ^{n+1}_i by using (4.10) instead of using $m^{n+1}_i = \int_{\mathbb{R}^d} \psi^{n+1}_i(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}$. We propose in this section an alternative form of the algorithm that does exactly that. This algorithm is henceforth referred to as version 2. For reasons that will be detailed in §4.6, we have not been able so far to construct an SSP extension of this algorithm that is both conservative and invariant domain preserving, whereas the SSP extension of version 1 is trivial.

Let $(\omega_l,\zeta_l)_{l\in\mathcal{L}}$ be a quadrature such that $\int_0^1 f(\zeta)\,\mathrm{d}\zeta \simeq \sum_{l\in\mathcal{L}} \omega_l f(\zeta_l)$ is exact for all polynomial function f of degree at most $d-1$. We denote $t^n_l = t^n + \tau\zeta_l$. Given

the ALE field $\boldsymbol{w}^n \in \boldsymbol{P}^{\rm geo}(\mathcal{T}_h^n)$, the Lagrange nodes of the mesh are moved for each time t_l^n, $l \in \mathcal{L}$, by using (4.22):

$$\boldsymbol{a}_i(t_l^n) = \boldsymbol{a}_i^n + \tau\zeta_l \mathbf{W}_i^{{\rm geo},n}. \tag{4.32}$$

This defines the new meshes $\mathcal{T}_h(t_l^n)$. This allows us to compute

$$\boldsymbol{c}_{ij}(t_l^n) := \int_{S_i^n(t_l^n)} \nabla\varphi_j(\boldsymbol{x},t_l^n)\varphi_i(\boldsymbol{x},t_l^n)\,{\rm d}\boldsymbol{x}, \qquad \boldsymbol{c}_{ij}^n := \sum_{l\in\mathcal{L}} \omega_l \boldsymbol{c}_{ij}(t_l^n). \tag{4.33}$$

After constructing $\mathcal{T}_h(t^{n+1})$ by setting $\boldsymbol{a}_i^{n+1} = \boldsymbol{a}_i^n + \tau\mathbf{W}_i^{{\rm geo},n}$, we define the mass of ψ_i^{n+1} by

$$m_i^{n+1} := \int_{\mathbb{R}^d} \psi_i^{n+1}(\boldsymbol{x})\,{\rm d}\boldsymbol{x}. \tag{4.34}$$

Then the change of basis (4.30) is applied to obtain the representation of $\boldsymbol{w}^n$ in $\boldsymbol{P}_d(\mathcal{T}_h^n)$. Following (3.4), we compute $\boldsymbol{u}_h^{n+1}$ by using the following explicit technique:

$$\frac{m_i^{n+1}\mathbf{U}_i^{n+1} - m_i^n\mathbf{U}_i^n}{\tau} + \sum_{j\in\mathcal{I}(S_i^n)} (\boldsymbol{f}(\mathbf{U}_j^n) - \mathbf{U}_j^n\otimes\mathbf{W}_j^n)\cdot\boldsymbol{c}_{ij}^n - d_{ij}^n\mathbf{U}_j^n = 0, \tag{4.35}$$

where d_{ij}^n is computed by using (4.15).

LEMMA 4.4 (Non-conservative form). *The scheme* (4.35) *is equivalent to*

$$m_i^{n+1}\frac{\mathbf{U}_i^{n+1} - \mathbf{U}_i^n}{\tau} = \sum_{j\in\mathcal{I}(S_i^n)} ((\mathbf{U}_j^n - \mathbf{U}_i^n)\otimes\mathbf{W}_j^n - \boldsymbol{f}(\mathbf{U}_j^n))\cdot\boldsymbol{c}_{ij}^n + d_{ij}^n\mathbf{U}_j^n. \tag{4.36}$$

Proof. We rewrite (4.35) as follows:

$$m_i^{n+1}\frac{\mathbf{U}_i^{n+1} - \mathbf{U}_i^n}{\tau} + \frac{m_i^{n+1} - m_i^n}{\tau}\mathbf{U}_i^n = \sum_{j\in\mathcal{I}(S_i^n)} (\mathbf{U}_j^n\otimes\mathbf{W}_j^n - \boldsymbol{f}(\mathbf{U}_j^n))\cdot\boldsymbol{c}_{ij}^n + d_{ij}^n\mathbf{U}_j^n.$$

Then, recalling the expression (4.31) of $\boldsymbol{v}_{\rm A}(\boldsymbol{x},t) = \sum_{i\in\{1:I\}}\mathbf{W}_i^n\varphi_i(\boldsymbol{x},t)$, and using Lemma 3.2 with $\psi = \psi_i^n$ and $\varphi(\boldsymbol{x},t) = \varphi_i(\boldsymbol{x},t)$ we infer that

$$\begin{aligned}(m_i^{n+1} - m_i^n)\mathbf{U}_i^n &= \tau\mathbf{U}_i^n\sum_{l\in\mathcal{L}}\omega_l\int_{\mathbb{R}^d}\varphi_i(\boldsymbol{x},t_l^n)\sum_{j\in\mathcal{I}(S_i^n)}\mathbf{W}_j^n\cdot\nabla\varphi_j(\boldsymbol{x},t_l^n)\,{\rm d}\boldsymbol{x}\\ &= \tau\mathbf{U}_i^n\sum_{j\in\mathcal{I}(S_i^n)}\mathbf{W}_j^n\cdot\boldsymbol{c}_{ij}^n = \tau\sum_{j\in\mathcal{I}(S_i^n)}(\mathbf{U}_i^n\otimes\mathbf{W}_j^n)\cdot\boldsymbol{c}_{ij}^n,\end{aligned}$$

whence the result. ∎

4.6. Version 1 vs. version 2 and SSP extension. We now give an overview of what has been done in the previous sections by highlighting the main differences between the two versions of the algorithm.

- Both versions of the algorithm use the two sets of reference elements: we use $\{(\widehat{K}_r, \widehat{P}_r^{\rm geo}, \widehat{\Sigma}_r^{\rm geo})\}_{1\le r\le\varpi}$ for the geometric mappings (see (4.1)), and we use $\{(\widehat{K}_r, \widehat{P}_r, \widehat{\Sigma}_r)\}_{1\le r\le\varpi}$ for the approximation of $\boldsymbol{u}$.

- We assume that $\boldsymbol{w}^n \in \boldsymbol{P}_d(\mathcal{T}_h^n)$ in version 1, whereas we assume that $\boldsymbol{w}^n \in \boldsymbol{P}^{\rm geo}(\mathcal{T}_h^n)$ in version 2. We also must assume that $\boldsymbol{P}^{\rm geo}(\mathcal{T}_h^n) \subset \boldsymbol{P}_d(\mathcal{T}_h^n)$ (i.e., $\widehat{P}^{\rm geo} \subset \widehat{P}$, see (4.29)) in version 2, which is not the case for version 1; actually the space $\boldsymbol{P}^{\rm geo}(\mathcal{T}_h^n)$ does not play any role in version 1.
- Only the meshes $\mathcal{T}_h^n$ and $\mathcal{T}_h^{n+1}$ are considered in version 1, whereas one must construct all the intermediate meshes $\mathcal{T}_h(t_l^n)$, $l \in \mathcal{L}$, in version 2.
- The mass of ψ_i^{n+1} is updated by setting $\mathfrak{m}_i^{n+1} = \mathfrak{m}_i^n + \tau \int_{S_i^n} \psi_i^n(\boldsymbol{x}) \nabla\cdot\boldsymbol{w}^n(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}$ in version 1, whereas it is updated by setting $m_i^{n+1} = \int_{S_i^{n+1}} \psi_i^{n+1}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}$ in version 2.

Retaining the invariant domain property (see §5.2) and increasing the time accuracy can be done by using so-called Strong Stability Preserving (SSP) time discretization methods. The key is to achieve higher-order accuracy in time by making convex combination of solutions of forward Euler sub-steps. More precisely each time step of a SSP method is decomposed into substeps that are all forward Euler solutions, and the end of step solution is constructed as a convex combination of the intermediate solutions; we refer to Ferracina and Spijker [15], Higueras [23], Gottlieb et al. [18] for reviews on SPP techniques. Algorithm 2 illustrates one Euler step for either version 1 or version 2 of the scheme. SSP techniques are useful when combined with reasonable limitation strategies since the resulting methods are both high-order, in time and space, and invariant domain preserving.

Algorithm 2 Euler step (version 1 and version 2)

Require: $\mathcal{T}_h^0$, $\boldsymbol{u}_h^0$, ($\mathfrak{m}^0$ or m^0), $\boldsymbol{w}^0$, τ

1: Compute $\widetilde{\boldsymbol{a}}_i^1 = \boldsymbol{a}_i^0 + \tau \boldsymbol{w}^0$, ($\widetilde{\mathfrak{m}}^1$ or $\widetilde{m}^1$), $\widetilde{\boldsymbol{u}}_h^1$, and build new mesh $\widetilde{\mathcal{T}}_h^1$

2: **return** $\widetilde{\mathcal{T}}_h^1$, $\widetilde{\boldsymbol{u}}_h^1$, ($\widetilde{\mathfrak{m}}^1$ or $\widetilde{m}^1$)

We describe the SSP RK3 implementation of version 1 of the scheme in Algorithm 3. Generalizations to other SSP techniques are left to the reader.

Algorithm 3 SPP RK3, version 1

Require: $\mathcal{T}_h^0$, $\boldsymbol{u}_h^0$, $\mathfrak{m}^0$, t^0

1: Define the ALE velocity $\boldsymbol{w}^0$ at t^0

2: Call Euler step($\mathcal{T}_h^0$, $\boldsymbol{u}_h^0$, $\mathfrak{m}^0$, $\boldsymbol{w}^0$, τ, $\mathcal{T}_h^1$, $\boldsymbol{u}_h^1$, $\mathfrak{m}^1$)

3: Define the ALE velocity $\boldsymbol{w}^1$ at $t^0 + \tau$

4: Call Euler step($\mathcal{T}_h^1$, $\boldsymbol{u}_h^1$, $\mathfrak{m}^1$, $\boldsymbol{w}^1$, τ, $\widetilde{\mathcal{T}}_h^2$, $\widetilde{\boldsymbol{u}}_h^2$, $\widetilde{\mathfrak{m}}^2$)

5: Set $\boldsymbol{a}^2 = \frac{3}{4}\boldsymbol{a}^0 + \frac{1}{4}\widetilde{\boldsymbol{a}}^2$, $\mathfrak{m}^2 = \frac{3}{4}\mathfrak{m}^0 + \frac{1}{4}\widetilde{\mathfrak{m}}^2$, build mesh $\mathcal{T}_h^2$, $\boldsymbol{u}_h^2 = \frac{3}{4}\frac{\mathfrak{m}^0}{\mathfrak{m}^2}\boldsymbol{u}_h^0 + \frac{1}{4}\frac{\widetilde{\mathfrak{m}}^2}{\mathfrak{m}^2}\widetilde{\boldsymbol{u}}_h^2$

6: Define the ALE velocity $\boldsymbol{w}^2$ at $t^0 + \frac{1}{2}\tau$

7: Call Euler step($\mathcal{T}_h^2$, $\boldsymbol{u}_h^2$, $\mathfrak{m}^2$, $\boldsymbol{w}^2$, τ, $\widetilde{\mathcal{T}}_h^3$, $\widetilde{\boldsymbol{u}}_h^3$, $\widetilde{\mathfrak{m}}^3$)

8: Set $\boldsymbol{a}^3 = \frac{1}{3}\boldsymbol{a}^0 + \frac{2}{3}\widetilde{\boldsymbol{a}}^3$, $\mathfrak{m}^3 = \frac{1}{3}\mathfrak{m}^0 + \frac{2}{3}\widetilde{\mathfrak{m}}^3$, build mesh $\mathcal{T}_h^3$, $\boldsymbol{u}_h^3 = \frac{1}{3}\frac{\mathfrak{m}^0}{\mathfrak{m}^3}\boldsymbol{u}_h^0 + \frac{2}{3}\frac{\widetilde{\mathfrak{m}}^3}{\mathfrak{m}^3}\widetilde{\boldsymbol{u}}_h^3$

9: **return** $\mathcal{T}_h^3$, $\boldsymbol{u}_h^3$, $\mathfrak{m}^3$, $t^1 = t^0 + dt$

Note that $\boldsymbol{u}_h^2$ is a convex combination of $\boldsymbol{u}_h^0$ and $\widetilde{\boldsymbol{u}}_h^2$ since $1 = \frac{3}{4}\frac{\mathfrak{m}_i^0}{\mathfrak{m}_i^2} + \frac{1}{4}\frac{\widetilde{\mathfrak{m}}_i^2}{\mathfrak{m}_i^2}$. The same observation holds for $\boldsymbol{u}_h^3$, i.e., $\boldsymbol{u}_h^3$ is a convex combination of $\boldsymbol{u}_h^0$ and $\widetilde{\boldsymbol{u}}_h^3$ since $1 = \frac{1}{3}\frac{\mathfrak{m}_i^0}{\mathfrak{m}_i^3} + \frac{2}{3}\frac{\widetilde{\mathfrak{m}}_i^3}{\mathfrak{m}_i^3}$, for any $i \in \{1{:}I\}$.

Remark 4.3. (Version 2+SSP) The above properties do not hold for version 2 of the scheme, since in general $m_i^2 \neq \frac{3}{4}m_i^0 + \frac{1}{4}\widetilde{m}_i^2$ and $m_i^3 \neq \frac{1}{3}m_i^0 + \frac{2}{3}\widetilde{m}_i^3$. Notice though

that it can be shown that $m_i^2 = \frac{3}{4}m_i^0 + \frac{1}{4}\widetilde{m}_i^2$ and $m_i^3 = \frac{1}{3}m_i^0 + \frac{2}{3}\widetilde{m}_i^3$ in one space dimension if the ALE velocity is kept constant over the entire Runge Kutta step. So far, we are not aware of any SSP technique for version 2 of the algorithm (at least second-order in time) that is both conservative and invariant domain preserving in the multidimensional case.

5. Stability analysis. We establish the conservation and the invariant domain property of the two schemes (4.11) and (4.35) in this section.

5.1. Conservation. We first discuss the conservation properties of the two schemes.

LEMMA 5.1. (i) *For the scheme* (4.35)*, the quantity* $\int_{\mathbb{R}^d} \boldsymbol{u}_h^n \,\mathrm{d}\boldsymbol{x}$ *is conserved for all* $n \geq 0$*, i.e.,* $\int_{\mathbb{R}^d} \boldsymbol{u}_h^n \,\mathrm{d}x = \int_{\mathbb{R}^d} \boldsymbol{u}_h^0 \,\mathrm{d}\boldsymbol{x}$ *for all* $n \geq 0$*.* (ii) *For the scheme* (4.11)*, the quantity* $\sum_{i\in\{1:I\}} \mathfrak{m}_i^n \mathbf{U}_i^n$ *is conserved, i.e., it is independent of* n*.*

Proof. We start with by proving (i). We observe first that

$$\sum_{i\in\{1:I\}} m_i^n \mathbf{U}_i^n = \sum_{i\in\{1:I\}} \mathbf{U}_i^n \sum_{j\in\{1:I\}} \int_{\mathbb{R}^d} \psi_i^n(\boldsymbol{x})\psi_j^n(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x}$$
$$= \sum_{j\in\{1:I\}} \int_{\mathbb{R}^d} \sum_{i\in\{1:I\}} \mathbf{U}_i^n \psi_i^n(\boldsymbol{x})\psi_j^n(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} = \sum_{j\in\{1:I\}} \int_{\mathbb{R}^d} \boldsymbol{u}_h^n(\boldsymbol{x})\psi_j^n(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x}.$$

Then the partition of unity property gives $\sum_{i\in\{1:I\}} m_i^n \mathbf{U}_i^n = \int_{\mathbb{R}^d} \boldsymbol{u}_h^n \,\mathrm{d}\boldsymbol{x}$. We now sum over the index i in (4.11) and we use again the partition of unity property to infer that

$$\frac{\int_{\mathbb{R}^d} \boldsymbol{u}_h^{n+1} \,\mathrm{d}\boldsymbol{x} - \int_{\mathbb{R}^d} \boldsymbol{u}_h^n \,\mathrm{d}x}{\tau} - \sum_{j\in\mathcal{I}(S_i^n)} \Big(\sum_{i\in\{1:I\}} d_{ij}^n \Big) \mathbf{U}_j^n$$
$$+ \sum_{l\in\mathcal{L}} \omega_l \int_{\mathbb{R}^d} \nabla\cdot \Big(\sum_{j\in\{1:I\}} (\boldsymbol{f}(\mathbf{U}_j^n) - \mathbf{U}_j^n \otimes \mathbf{W}_j^n)\varphi_j(\boldsymbol{x}, t_l^n) \Big) \,\mathrm{d}\boldsymbol{x} = 0.$$

The boundary conditions and the structure assumptions on d_{ij}^n, see (4.12), imply the desired result. The proof of (ii) follows the same lines. ∎

5.2. Invariant domain property. We can now prove a result somewhat similar in spirit to Thm 5.1 from Farhat et al. [14], although the present result is more general since it applies to any hyperbolic system.

We start with version 2 of the scheme by defining the local minimum mesh size $\underline{h}_{ij}(t)$ associated with an ordered pair of shape functions $(\varphi_i(\cdot,t), \varphi_j(\cdot,t))$ at time t as follows: $\underline{h}_{ij}(t) := \frac{1}{\|\|\nabla\varphi_j\|_{\ell^2}\|_{L^\infty(S_{ij}(t))}}$, where $S_{ij}(t) = S_i(t) \cap S_j(t)$. We also define $\underline{h}_i(t) = \min_{j\in\mathcal{I}(S_i(t))} \underline{h}_{ij}(t)$. Given a time t^n, we define a local minimum mesh size $\underline{h}_i^n$ and a local mesh structure parameter κ_i^n by

$$\underline{h}_i^n := \min_{l\in\mathcal{L}} \underline{h}_i(t_l^n), \qquad \kappa_i^n := \frac{\sum_{i\neq j\in\mathcal{I}(S_i^n)} \sum_{l\in\mathcal{L}} \omega_l \int_{S_{ij}(t_l^n)} \varphi_i(\boldsymbol{x}, t_l^n) \,\mathrm{d}\boldsymbol{x}}{\sum_{l\in\mathcal{L}} \omega_l m_i(t_l^n)}. \tag{5.1}$$

For version 1 of the algorithm we set

$$\underline{h}_i^n := \underline{h}_i(t^n), \qquad \kappa_i^n := \frac{\sum_{i\neq j\in\mathcal{I}(S_i^n)} \int_{S_{ij}^n} \psi_i^n(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x}}{\int_{S_i^n} \psi_i^n(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x}}. \tag{5.2}$$

Note that the upper estimate $\kappa_i^n \le \max_{j\in\{1:I\}} \mathrm{card}(\mathcal{I}(S_j(0))) - 1$ implies that κ_i^n is uniformly bounded with respect to n and i for both algorithms.

THEOREM 5.2 (Local invariance). *Let $n \ge 0$, and let $i \in \{1{:}I\}$. $\lambda_{i,\max}^n := \max_{j\in\mathcal{I}(S_i^n)}(\lambda_{\max}(\boldsymbol{g}_j^n, \boldsymbol{n}_{ij}^n, \mathbf{U}_i^n, \mathbf{U}_j^n)$, $\lambda_{\max}(\boldsymbol{g}_i^n, \boldsymbol{n}_{ji}^n, \mathbf{U}_j^n, \mathbf{U}_i^n))$. Depending on the version of the algorithm, version 2 or version 1 respectively, assume that τ is such that*

$$2\tau \frac{\lambda_{i,\max}^n}{\underline{h}_i^n}\kappa_i^n \frac{\sum_{l\in\mathcal{L}}\omega_l m_i(t_l^n)}{m_i^{n+1}} \le 1, \quad \text{or} \quad 2\tau \frac{\lambda_{i,\max}^n}{\underline{h}_i^n}\kappa_i^n \frac{\mathfrak{m}_i^n}{\mathfrak{m}_i^{n+1}} \le 1. \tag{5.3}$$

Let $B \subset \mathcal{A}_{\boldsymbol{f}}$ be a convex invariant set for the flux $\boldsymbol{f}$ such that $\{\mathbf{U}_j^n \mid j \in \mathcal{I}(S_i^n)\} \subset B$, then $\mathbf{U}_i^{n+1} \in B$.

Proof. We do the proof for version 2 of the algorithm. The proof for version 1 is similar. Let $i \in \{1{:}I\}$ and invoke (4.36) from Lemma 4.4 (or (4.18) from Lemma 4.2 for version 1) to express $\mathbf{U}_i^{n+1}$ into the following from

$$\mathbf{U}_i^{n+1} = \mathbf{U}_i^n + \frac{\tau}{m_i^{n+1}} \sum_{j\in\mathcal{I}(S_i^n)} ((\mathbf{U}_j^n - \mathbf{U}_i^n)\otimes \mathbf{W}_j^n - \boldsymbol{f}(\mathbf{U}_j^n))\cdot \boldsymbol{c}_{ij}^n + d_{ij}^n \mathbf{U}_j^n.$$

Since the partition of unity property implies that $\sum_{j\in\mathcal{I}(S_i^n)} \boldsymbol{c}_{ij}^n = 0$ and we have $\sum_{j\in\mathcal{I}(S_i^n)} d_{ij}^n = 0$ from (4.12), we can rewrite the above equation as follows:

$$\begin{aligned}
\mathbf{U}_i^{n+1} &= \mathbf{U}_i^n + \sum_{j\in\mathcal{I}(S_i^n)} d_{ij}^n(\mathbf{U}_i^n + \mathbf{U}_j^n) \\
&\quad + \frac{\tau}{m_i^{n+1}} \sum_{j\in\mathcal{I}(S_i^n)} ((\mathbf{U}_j^n - \mathbf{U}_i^n)\otimes \mathbf{W}_j^n + \boldsymbol{f}(\mathbf{U}_i^n) - \boldsymbol{f}(\mathbf{U}_j^n))\cdot \boldsymbol{c}_{ij}^n \\
&= \mathbf{U}_i^n \left(1 + 2d_{ii}^n \frac{\tau}{m_i^{n+1}}\right) + \sum_{i\ne j\in\mathcal{I}(S_i^n)} d_{ij}^n(\mathbf{U}_i^n + \mathbf{U}_j^n) \\
&\quad + \frac{\tau}{m_i^{n+1}} \sum_{i\ne j\in\mathcal{I}(S_i^n)} ((\mathbf{U}_j^n - \mathbf{U}_i^n)\otimes \mathbf{W}_j^n + \boldsymbol{f}(\mathbf{U}_i^n) - \boldsymbol{f}(\mathbf{U}_j^n))\cdot \boldsymbol{c}_{ij}^n.
\end{aligned}$$

Recall that $\boldsymbol{n}_{ij}^n := \boldsymbol{c}_{ij}^n/\|\boldsymbol{c}_{ij}^n\|_{\ell^2} \in S^{d-1}(\mathbf{0}, 1)$, and let us introduced the auxiliary state $\overline{\mathbf{U}}_{ij}^{n+1}$ defined by

$$\overline{\mathbf{U}}_{ij}^{n+1} = (\boldsymbol{f}(\mathbf{U}_i^n) - \boldsymbol{f}(\mathbf{U}_j^n) - (\mathbf{U}_i^n - \mathbf{U}_j^n)\otimes \mathbf{W}_j^n)\cdot \boldsymbol{n}_{ij}^n \frac{\|\boldsymbol{c}_{ij}^n\|_{\ell^2}}{2d_{ij}^n} + \frac{1}{2}(\mathbf{U}_i^n + \mathbf{U}_j^n).$$

Then, provided we establish that $1 - \sum_{i\ne j\in\mathcal{I}(S_i^n)} 2d_{ij}^n \frac{\tau}{m_i^{n+1}} \ge 0$, we have proved that $\mathbf{U}_i^{n+1}$ is a convex combination of $\mathbf{U}_i^n$ and $(\overline{\mathbf{U}}_{ij}^{n+1})_{i\ne j\in\mathcal{I}(S_i^n)}$:

$$\mathbf{U}_i^{n+1} = \mathbf{U}_i^n \left(1 - \sum_{i\ne j\in\mathcal{I}(S_i^n)} 2d_{ij}^n \frac{\tau}{m_i^{n+1}}\right) + \frac{\tau}{m_i^{n+1}} \sum_{i\ne j\in\mathcal{I}(S_i^n)} 2d_{ij}^n \overline{\mathbf{U}}_{ij}^{n+1}. \tag{5.4}$$

Let us now consider the Riemann problem (4.16). Let $\boldsymbol{v}(\boldsymbol{g}_j^n, \boldsymbol{n}_{ij}^n, \mathbf{U}_i^n, \mathbf{U}_j^n)$ be the solution to (4.16) with $\boldsymbol{g}_j^n(\boldsymbol{v}) := \boldsymbol{f}(\boldsymbol{v}) - \boldsymbol{v}\otimes \mathbf{W}_j^n$. Let $\lambda_{\max}(\boldsymbol{g}_j^n, \boldsymbol{n}_{ij}^n, \mathbf{U}_i^n, \mathbf{U}_j^n)$ be the fastest wave speed in (4.16), see (4.17). Using the notation of Lemma 2.1, we then observe that

$$\overline{\mathbf{U}}_{ij}^{n+1} = \overline{\boldsymbol{v}}(t, \boldsymbol{g}_j^n, \boldsymbol{n}_{ij}^n, \mathbf{U}_i^n, \mathbf{U}_j^n)$$

with $t = \frac{\|\boldsymbol{c}_{ij}^n\|_{\ell^2}}{2d_{ij}^n}$, provided $t\lambda_{\max}(\boldsymbol{g}_j^n, \boldsymbol{n}_{ij}^n, \mathbf{U}_i^n, \mathbf{U}_j^n) \le \frac{1}{2}$. Note that the definition of d_{ij}^n, (4.15), implies that the condition $t\lambda_{\max}(\boldsymbol{g}_j^n, \boldsymbol{n}_{ij}^n, \mathbf{U}_i^n, \mathbf{U}_i^n) \le \frac{1}{2}$ is satisfied. Since B is an invariant set for the flux $\boldsymbol{f}$, by Lemma 2.3, B is also an invariant set for the flux $\boldsymbol{g}_j^n$. Since, in addition, B contains the data $(\mathbf{U}_i^n, \mathbf{U}_j^n)$, we conclude that $\overline{\mathbf{U}}_{ij}^{n+1} = \overline{\boldsymbol{v}}(t, \boldsymbol{g}_j^n, \boldsymbol{n}_{ij}^n, \mathbf{U}_i^n, \mathbf{U}_j^n) \in B$; see Remark 2.1. In conclusion, $\mathbf{U}_i^{n+1} \in B$ since $\mathbf{U}_i^{n+1}$ is a convex combination of objects in B.

Setting $\mathcal{Y} := \sum_{i\neq j\in\mathcal{I}(S_{ij}(t_l^n))} \frac{2\tau d_{ij}^n}{m_i^{n+1}}$, there remains to establish that $1-\mathcal{Y} \ge 0$ to complete the proof for version 2 of the algorithm. Note first that

$$\|\boldsymbol{c}_{ij}(t_l^n)\|_{\ell^2} \le \int_{S_{ij}(t_l^n)} \|\nabla\varphi_j(\boldsymbol{x}, t_l^n)\|_{\ell^2}\varphi_i(\boldsymbol{x}, t_l^n)\,\mathrm{d}\boldsymbol{x} \le \underline{h}_{ij}^{-1}(t_l^n)\int_{S_{ij}(t_l^n)} \varphi_i(\boldsymbol{x}, t_l^n)\,\mathrm{d}\boldsymbol{x}.$$

Notice that $\boldsymbol{c}_{ij}(t_l^n) = -\boldsymbol{c}_{ji}(t_l^n)$ because there are no boundary conditions (i.e., we solve the Cauchy problem in $\mathbb{R}^d$, or the domain is periodic); hence $\|\boldsymbol{c}_{ji}(t_l^n)\|_{\ell^2} = \|\boldsymbol{c}_{ij}(t_l^n)\|_{\ell^2}$. Recalling the definition of d_{ij}^n, we have

$$\begin{aligned}\mathcal{Y} &\le 2\tau\frac{\lambda_{i,\max}^n}{\underline{h}_i^n}\frac{\sum_{l\in\mathcal{L}}\omega_l m_i(t_l^n)}{m_i^{n+1}}\frac{\sum_{i\neq j\in\mathcal{I}(S_{ij}(t_l^n))}\sum_{l\in\mathcal{L}}\omega_l\int_{S_{ij}(t_l^n)}\varphi_i(\boldsymbol{x}, t_l^n)\,\mathrm{d}\boldsymbol{x}}{\sum_{l\in\mathcal{L}}\omega_l m_i(t_l^n)}\\ &\le 2\tau\frac{\lambda_{\max}^n}{\underline{h}_i^n}\frac{\sum_{l\in\mathcal{L}}\omega_l m_i(t_l^n)}{m_i^{n+1}}\kappa_i^n \le 1,\end{aligned}$$

which is the desired result. The proof of the CFL condition for version 1 of the algorithm follows the same lines. This concludes the proof. ∎

COROLLARY 5.3. *Let $n \in \mathbb{N}$. Assume that τ is small enough so that the CFL condition* (5.3) *holds for all $i \in \{1{:}I\}$. Let $B \subset \mathcal{A}_{\boldsymbol{f}}$ be a convex invariant set. Assume that $\{\mathbf{U}_i^n \mid i \in \{1{:}I\}\} \subset B$. Then* (i) $\{\mathbf{U}_i^{n+1} \mid i \in \{1{:}I\}\} \subset B$ (ii) $\boldsymbol{u}_h^n \in B$ *and* $\boldsymbol{u}_h^{n+1} \in B$.

Proof. The statement (i) is a direct consequence of Theorem 5.2. The statement (ii) is a consequence of (4.7) from Lemma 4.1. ∎

COROLLARY 5.4. *Let $B \subset \mathcal{A}_{\boldsymbol{f}}$ be a convex invariant set containing the initial data $\boldsymbol{u}_0$. Assume that $\{\mathbf{U}_i^0 \mid i \in \{1{:}I\}\} \subset B$. Let $N \in \mathbb{N}$. Assume that τ is small enough so that the CFL condition* (5.3) *holds for all $i \in \{1{:}I\}$ and all $n \in \{0{:}N\}$. Then $\{\mathbf{U}_i^n \mid i \in \{1{:}I\}\} \subset B$ and $\boldsymbol{u}_h^n \in B$ for all $n \in \{0{:}N+1\}$.*

Remark 5.1. (Construction of $\boldsymbol{u}_h^0$) Let $B \subset \mathcal{A}_{\boldsymbol{f}}$ be a convex invariant set containing the initial data $\boldsymbol{u}_0$. If $\boldsymbol{P}_m(\mathcal{T}_h^0)$ is composed of piecewise Lagrange elements, then defining $\boldsymbol{u}_h^0$ to be the Lagrange interpolant of $\boldsymbol{u}_0$, we have $\{\mathbf{U}_i^0 \mid i \in \{1{:}I\}\} \subset B$. Similarly if $\boldsymbol{P}_m(\mathcal{T}_h^0)$ is composed of Bernstein finite elements of degree two and higher, then defining $\boldsymbol{u}_h^0$ to be the Bernstein interpolant of $\boldsymbol{u}_0$ we have $\{\mathbf{U}_i^0 \mid i \in \{1{:}I\}\} \subset B$; see Lai and Schumaker [27, Eq. (2.72)]. Note that the approximation of $\boldsymbol{u}_0$ is only second-order accurate in this case independently of the polynomial degree of the Bernstein polynomials; see [27, Thm. 2.45]. In both cases the assumptions of Corollary 5.4 hold.

5.3. Discrete Geometric Conservation Law. Both the scheme (4.11) and the scheme (4.35) satisfy the so-called Discrete Geometric Conservation Law (DGCL), i.e., they preserve constant states.

COROLLARY 5.5 (DGCL). *The schemes* (4.11) *and* (4.35) *preserve constant states. In particular if $\mathbf{U}_j^n = \mathbf{U}_i^n$ for all $j \in \mathcal{I}(S_i^n)$, then $\mathbf{U}_i^{n+1} = \mathbf{U}_i^n$.*

Proof. The partition of unity property implies that $\sum_{j\in\mathcal{I}(S_i^n)} \boldsymbol{c}_{ij}^n = 0$ for both schemes. Moreover, the definition d_{ij}^n, which is common for both schemes, implies that $\sum_{j\in\mathcal{I}(S_i^n)} d_{ij}^n = 0$ (see (4.12)). For the scheme (4.35), Lemma 4.4, which we recall is a consequence of Lemma 3.2, implies that

$$\begin{aligned}\mathbf{U}_i^{n+1} = \mathbf{U}_i^n &+ d_{ij}^n(\mathbf{U}_j^n - \mathbf{U}_i^n)\\ &+ \frac{\tau}{m_i^{n+1}} \sum_{j\in\mathcal{I}(S_i^n)} ((\mathbf{U}_j^n - \mathbf{U}_i^n)\otimes \mathbf{W}_j^n + \boldsymbol{f}(\mathbf{U}_i^n) - \boldsymbol{f}(\mathbf{U}_j^n))\cdot \boldsymbol{c}_{ij}^n.\end{aligned}$$

For the scheme (4.11), Lemma 4.2 implies that

$$\begin{aligned}\mathbf{U}_i^{n+1} = \mathbf{U}_i^n &+ d_{ij}^n(\mathbf{U}_j^n - \mathbf{U}_i^n)\\ &+ \frac{\tau}{\mathfrak{m}_i^{n+1}} \sum_{j\in\mathcal{I}(S_i^n)} ((\mathbf{U}_j^n - \mathbf{U}_i^n)\otimes \mathbf{W}_j^n + \boldsymbol{f}(\mathbf{U}_i^n) - \boldsymbol{f}(\mathbf{U}_j^n))\cdot \boldsymbol{c}_{ij}^n.\end{aligned}$$

It is now clear that if $\mathbf{U}_j^n = \mathbf{U}_i^n$ for all $j \in \mathcal{I}(S_i^n)$, then $\mathbf{U}_i^{n+1} = \mathbf{U}_i^n$. ∎

Remark 5.2. (DGCL) Note that although the DGCL seems to be given some importance in the literature, Corollary 5.5 has no particular significance. For scheme 2, it is a direct consequence of Lemma 3.2 which is invoked to rewrite the scheme (4.35) from the conservative form to the equivalent nonconservative form (4.36). For scheme 1, it is a direct consequence of the definition of the mass update (4.10) which is invoked to rewrite the scheme (4.11) from the conservative form to the equivalent nonconservative form (4.18). The nonconservative form of both schemes is essential to prove the invariant domain property. In other words, *the DGCL is just a consequence of the equivalence of the discrete conservative and nonconservative formulations.*

5.4. Discrete entropy inequality. In this section we prove a discrete entropy inequality which is consistent with the inequality stated in Lemma 3.4.

THEOREM 5.6. *Let $(\eta, \boldsymbol{q})$ be an entropy pair for* (1.1)*. Let $n \in \mathbb{N}$ and $i \in \{1{:}I\}$. Assume that all the assumptions of Theorem 5.2 hold. Then the following discrete entropy inequality holds for scheme 1:*

$$\begin{aligned}\frac{1}{\tau}\big(\mathfrak{m}_i^{n+1}\eta(\mathbf{U}_i^{n+1}) - \mathfrak{m}_i^n\eta(\mathbf{U}_i^n)\big) \le &- \sum_{j\in\mathcal{I}(S_i^n)} d_{ij}^n\eta(\mathbf{U}_j^n)\\ &- \int_{\mathbb{R}^d} \nabla\cdot\Big(\sum_{j\in\mathcal{I}(S_i^n)} (\boldsymbol{q}(\mathbf{U}_j^n) - \eta(\mathbf{U}_j^n)\mathbf{W}_j^n)\psi_j^n(\boldsymbol{x})\Big)\psi_i^n(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}\end{aligned} \tag{5.5}$$

and the following holds for scheme 2:

$$\begin{aligned}\frac{1}{\tau}\big(m_i^{n+1}\eta(\mathbf{U}_i^{n+1}) - m_i^n\eta(\mathbf{U}_i^n)\big) \le &- \sum_{j\in\mathcal{I}(S_i^n)} d_{ij}^n\eta(\mathbf{U}_j^n)\\ &- \sum_{l\in\mathcal{L}}\omega_l \int_{\mathbb{R}^d} \nabla\cdot\Big(\sum_{j\in\mathcal{I}(S_i^n)} (\boldsymbol{q}(\mathbf{U}_j^n) - \eta(\mathbf{U}_j^n)\mathbf{W}_j^n)\varphi_j(\boldsymbol{x}, t_l^n)\Big)\varphi_i(\boldsymbol{x}, t_l^n)\,\mathrm{d}\boldsymbol{x}\end{aligned} \tag{5.6}$$

Proof. We only do the proof for scheme 2. The proof for scheme 1 is similar. Let $(\eta, \boldsymbol{q})$ be an entropy pair for the hyperbolic system (1.1). Let $i \in \{1{:}I\}$ and let $n \in \mathbb{N}$.

Then using (5.4), the CFL condition and the convexity of η, we have

$$\eta(\mathbf{U}_i^{n+1}) \le \eta(\mathbf{U}_i^n)\bigg(1 - \sum_{i\neq j\in\mathcal{I}(S_i^n)} 2d_{ij}^n \frac{\tau}{m_i^{n+1}}\bigg) + \frac{\tau}{m_i^{n+1}} \sum_{i\neq j\in\mathcal{I}(S_i^n)} 2d_{ij}^n \eta(\overline{\mathbf{U}}_{ij}^{n+1}).$$

This can also be rewritten as follows:

$$\frac{m_i^{n+1}}{\tau}\big(\eta(\mathbf{U}_i^{n+1}) - \eta(\mathbf{U}_i^n)\big) \le \sum_{i\neq j\in\mathcal{I}(S_i^n)} 2d_{ij}^n (\eta(\overline{\mathbf{U}}_{ij}^{n+1}) - \eta(\mathbf{U}_i^n)).$$

Owing to (2.7) from Lemma 2.1, and recalling that the entropy flux of the Riemann problem (4.16) is $(\boldsymbol{q}(v) - \eta(v)\mathbf{W}_j^n)\cdot\boldsymbol{n}_{ij}^n$ we infer that

$$\eta(\overline{\mathbf{U}}_{ij}^{n+1}) \le \tfrac{1}{2}(\eta(\mathbf{U}_i^n) + \eta(\mathbf{U}_j^n)) - t\big(\boldsymbol{q}(\mathbf{U}_j^n) - \eta(\mathbf{U}_j^n)\mathbf{W}_j^n - \boldsymbol{q}(\mathbf{U}_i^n) + \eta(\mathbf{U}_i^n)\mathbf{W}_j^n\big)\cdot\boldsymbol{n}_{ij}^n$$

with $t = \|\boldsymbol{c}_{ij}^n\|_{\ell^2}/2d_{ij}^n$. Inserting this inequality in the first one, we have

$$\begin{aligned}\frac{m_i^{n+1}}{\tau}\big(\eta(\mathbf{U}_i^{n+1}) - \eta(\mathbf{U}_i^n)\big) \le{}& \sum_{j\in\mathcal{I}(S_i^n)} d_{ij}^n(\eta(\mathbf{U}_j^n) - \eta(\mathbf{U}_i^n)) \\ &- \sum_{j\in\mathcal{I}(S_i^n)} \|\boldsymbol{c}_{ij}^n\|_{\ell^2}\big(\boldsymbol{q}(\mathbf{U}_j^n) - \boldsymbol{q}(\mathbf{U}_i^n) - (\eta(\mathbf{U}_j^n) - \eta(\mathbf{U}_i^n))\mathbf{W}_j^n\big)\cdot\boldsymbol{n}_{ij}^n.\end{aligned}$$

By proceeding as in the proof of Lemma 4.4, we observe that

$$\frac{m_i^{n+1} - m_i^n}{\tau} = \sum_{l\in\mathcal{L}} \omega_l \int_{\mathbb{R}^d} \varphi_i(\boldsymbol{x}, t_l^n) \sum_{j\in\mathcal{I}(S_i^n)} \mathbf{W}_j^n\cdot\nabla\varphi_j(\boldsymbol{x}, t_l^n)\,\mathrm{d}\boldsymbol{x} = \sum_{j\in\mathcal{I}(S_i^n)} \mathbf{W}_j^n\cdot\boldsymbol{c}_{ij}^n.$$

Then using that $\|\boldsymbol{c}_{ij}^n\|_{\ell^2}\boldsymbol{n}_{ij}^n = \boldsymbol{c}_{ij}^n$, we conclude that

$$\begin{aligned}\frac{1}{\tau}\big(m_i^{n+1}\eta(\mathbf{U}_i^{n+1}) - m_i^n\eta(\mathbf{U}_i^n)\big) \le{}& -\sum_{j\in\mathcal{I}(S_i^n)} d_{ij}\eta(\mathbf{U}_j^n) \\ &- \sum_{l\in\mathcal{L}} \omega_l \int_{\mathbb{R}^d} \nabla\cdot\bigg(\sum_{j\in\mathcal{I}(S_i^n)} (\boldsymbol{q}(\mathbf{U}_j^n) - \eta(\mathbf{U}_j^n)\mathbf{W}_j^n)\varphi_j(\boldsymbol{x}, t_l^n)\bigg)\varphi_i(\boldsymbol{x}, t_l^n)\,\mathrm{d}\boldsymbol{x}\end{aligned}$$

This concludes the proof. □

6. Numerical tests. In this section, we numerically illustrate the performance of scheme 1 using SSP RK3. All the tests reported below have also been done with version 2 and we have observed that the method works as advertised when used with Euler time stepping, but we do not show the results for brevity. As expected from Remark 4.3, we have indeed observed very small violations of the invariant domain (maximum principle in the scalar case) when version 2 is combined with SSP RK3.

All the tests have been done with two different codes. One code is written in F95 and uses $\mathbb{P}_1$ Lagrange elements on triangles. The other code is based on deal.ii [2], is written in C++ and uses $\mathbb{Q}_1$ Lagrange elements on quadrangles. The mesh composed of triangles is obtained by dividing all the quadrangles into two triangles. The same numbers of degrees of freedom are used for both codes.

6.1. Analytical scalar-valued solution. To test the convergence property of the SSP RK3 version of scheme 1, as described in Algorithm 3, we solve the linear transport equation in the domain $D^0 = (0,1)^2$:

$$\partial_t u + \nabla\cdot(\boldsymbol{\beta} u) = 0, \quad u_0(\boldsymbol{x}) = x_1 + x_2, \tag{6.1}$$

where $\boldsymbol{\beta} = (\sin(\pi x_1)\cos(\pi x_2)\cos(2\pi t), -\cos(\pi x_1)\sin(\pi x_2)\cos(2\pi t))^{\mathsf{T}}$. In both codes the ALE velocity is chosen by setting $\mathbf{W}_i^n = \boldsymbol{\beta}(\boldsymbol{a}_i^n)$, i.e., $\boldsymbol{w}_h^n$ is the Lagrange interpolant of $\boldsymbol{\beta}$ on $\mathcal{T}_h^n$. Notice that there is no issue with boundary condition since $\boldsymbol{\beta}\cdot\boldsymbol{n}_{|\partial D^0} = 0$. We first test the accuracy in time of the algorithm by setting $d_{ij}^n = 0$, i.e., the

Table 6.1: Rotation problem (6.1) with Lagrangian formulation, CFL=1.0

	Without viscosity				With viscosity			
# dofs	$\mathbb{Q}_1$, L^1-norm		$\mathbb{P}_1$, L^1-norm		$\mathbb{Q}_1$, L^1-norm		$\mathbb{P}_1$, L^1-norm	
81	6.46E-04	-	1.76E-03	-	1.31E-02	-	1.13E-02	-
289	1.16E-04	2.48	2.46E-04	2.85	4.28E-03	1.61	3.63E-03	1.64
1089	1.41E-05	3.03	3.23E-05	2.93	1.23E-03	1.80	1.04E-03	1.80
4225	1.76E-06	3.01	4.20E-06	2.94	3.29E-04	1.90	2.78E-04	1.90
16641	2.26E-07	2.96	5.76E-07	2.87	8.50E-05	1.95	7.19E-05	1.95
66049	2.82E-08	3.00	9.57E-08	2.59	2.16E-05	1.97	1.83E-05	1.98

viscosity is removed. We report the error measured in the L^1-norm at time $t = 0.5$ in the left part of Table 6.1. The computations are done with $CFL = 1$. The third-order convergence in time is confirmed. Note that there is no space error due to the particular choice of the ALE velocity and the initial data.

In the second test we put back the viscosity d_{ij}^n. Notice that the particular choice of the ALE velocity implies that $\lambda_{\max}(\boldsymbol{g}_j^n, \boldsymbol{n}_{ij}^n, \mathbf{U}_i^n, \mathbf{U}_j^n) = |(\boldsymbol{\beta}_i^n - \boldsymbol{\beta}_j^n)\cdot\boldsymbol{n}_{ij}^n|$; hence the viscosity is second-order in space instead of being first-order. This phenomenon makes the algorithm second-order in space (in addition to being conservative and maximum principle preserving). The error in the L^1-norm at time $t = 0.5$ is shown in the right part of Table 6.1.

6.2. Nonlinear scalar conservation equations. We now test scheme 1 on nonlinear scalar conservation equations.

6.2.1. Definition of the ALE velocity. In nonlinear conservation equations, solutions may develop shocks in finite time. In this case, using the purely Lagrangian velocity leads to a breakdown on the method in finite time. The breakdown manifests itself by a time step that goes to zero as the current time approaches the time of formation of the shock. One way to avoid this breakdown is to use an ALE velocity that is a modified version of the Lagrangian velocity.

Many techniques have been proposed in the literature to construct an ALE velocity. For instance, in Gastaldi [17], the ALE velocity is obtained by modeling the deformation of the domain as an "elastic" solid, see Gastaldi [17, Eq. (4.5)-(4.6)]. In Yang and Mavriplis [37], several mesh moving strategies are mentioned, including tension spring analogy, torsion spring analogy, truss analogy and linear elasticity analogy. In Wells et al. [36, Eq. (7)], an elliptic problem is used to construct an ALE velocity for the Euler equations. The purpose of the present paper is not to design an optimal ALE velocity but to propose an algorithm that is conservative and invariant

domain preserving for any reasonable ALE velocity. We now propose an algorithm to compute an ALE velocity based on ideas from Loubère et al. [31]. The only purpose of this algorithm is to be able to run the nonlinear simulations of §6.2.2 and §6.2.3 past the time of formation of shocks. We refer the reader to the abundant ALE literature to design other ALE velocities that better suit the reader's goals.

We first deform the mesh by using the Lagrangian motion, i.e., we set $\boldsymbol{a}_{i,\mathrm{Lg}}^{n+1} = \boldsymbol{a}_i^n + \tau \nabla_u \boldsymbol{f}(\mathbf{U}_i^n)$; we recall that $\mathbf{U}_i^n \in \mathbb{R}$ and $\nabla_u \boldsymbol{f}(\mathbf{U}_i^n) \in \mathbb{R}^d$ for scalar equations. Then, given $L \in \mathbb{N} \setminus \{0\}$, we define a smooth version of the Lagrangian mesh by smoothing the position of the geometric Lagrange nodes as follows:

$$\left\{\begin{array}{l} \boldsymbol{a}_i^{n+1,0} := \boldsymbol{a}_{i,\mathrm{Lg}}^{n+1},\ i \in \{1{:}I\} \\ \left(\boldsymbol{a}_i^{n+1,l} := \dfrac{1}{|\mathcal{I}(\mathcal{S}_i)|-1} \displaystyle\sum_{i\neq j\in\mathcal{I}(\mathcal{S}_i)} \boldsymbol{a}_j^{n+1,l-1},\ i \in \{1{:}I\}\right),\ l \in \{1{:}L\} \\ \boldsymbol{a}_{i,\mathrm{Sm}}^{n+1} := \boldsymbol{a}_i^{n+1,L},\ i \in \{1{:}I\}. \end{array}\right. \tag{6.2}$$

Finally, the actual ALE motion is defined by

$$\boldsymbol{a}_i^{n+1} = \omega \boldsymbol{a}_{i,\mathrm{Lg}}^{n+1} + (1-\omega)\boldsymbol{a}_{i,\mathrm{Sm}}^{n+1}, \quad i \in \{1{:}I\}$$

where ω is a user-defined constant. In all our computations, we use $\omega = 0.9$ and $L = 2$. The above method is similar to that used in Loubère et al. [31]. As mentioned in [31], a more advanced method consists of choosing ω pointwise by using the right Cauchy-Green strain tensor. We have not implemented this version of the method since the purpose of the tests in the next sections is just to show that the present method works as it should for any reasonable ALE velocity. The objective of this work is not to construct a sophisticated algorithm for the ALE velocity.

6.2.2. Burgers equation. We consider the inviscid Burgers equation in two space dimensions

$$\partial_t u + \nabla\cdot(\tfrac{1}{2}u^2\boldsymbol{\beta}) = 0, \quad u_0(\boldsymbol{x}) = \mathbb{1}_S, \tag{6.3}$$

where $\boldsymbol{\beta} = (1,1)^{\mathsf{T}}$, S is the unit square $(0,1)^2$, and $\mathbb{1}_E$ denotes the characteristic function of the set $E \subset \mathbb{R}^d$. The solution to this problem at time $t > 0$ and at $\boldsymbol{x} = (x_1, x_2)$ is as follows. Assume first that $x_2 \le x_1$, then define $\alpha = x_1 - x_2$. Let $\alpha_0 = 1 - \frac{t}{2}$. There are three cases. If $\alpha > 1$, then $u(x_1, x_2, t) = 0$.

$$\text{If } \alpha \le \alpha_0, \text{ then } \quad u(x_1,x_2,t) = \begin{cases} \frac{x_2}{t} & \text{if } 0 \le x_2 < t \\ 1 & \text{if } t \le x_2 < \frac{t}{2} + 1 - \alpha \\ 0 & \text{otherwise.} \end{cases} \tag{6.4}$$

$$\text{If } \alpha_0 < \alpha \le 1, \text{ then } \quad u(x_1,x_2,t) = \begin{cases} \frac{x_2}{t} & \text{if } 0 \le x_2 < \sqrt{2t(1-\alpha)} \\ 0 & \text{otherwise.} \end{cases} \tag{6.5}$$

If $x_2 > x_1$, then $u(x_1, x_2, t) := u(x_2, x_1, t)$. The computation are done up to $T = 1$ in the initial computational domain $D^0 = (-0.25, 1.75)^2$. The boundary of D^n does not move in the time interval $(0, 1)$, i.e., $\partial D^0 = \partial D^n$ for any $n \ge 0$. The results of the convergence tests are reported in Table 6.2. The solution computed on a 128×128 mesh and the mesh at $T = 1$ are shown in Figure 6.1.

Table 6.2: Burgers equation, convergence tests, $CFL = 0.1$

	$\mathbb{Q}_1$				$\mathbb{P}_1$			
# dofs	L^2-error		L^1-error		L^2-error		L^1-error	
81	5.79E-01	-	6.00E-01	-	5.80E-01	-	6.17E-01	-
289	4.20E-01	0.46	3.88E-01	0.63	4.43E-01	0.39	4.68E-01	0.40
1089	2.96E-01	0.51	2.32E-01	0.74	3.12E-01	0.51	2.86E-01	0.71
4225	2.14E-01	0.47	1.32E-01	0.82	2.17E-01	0.53	1.55E-01	0.88
16641	1.56E-02	0.45	7.40E-02	0.83	1.23E-01	0.82	7.57E-02	1.04

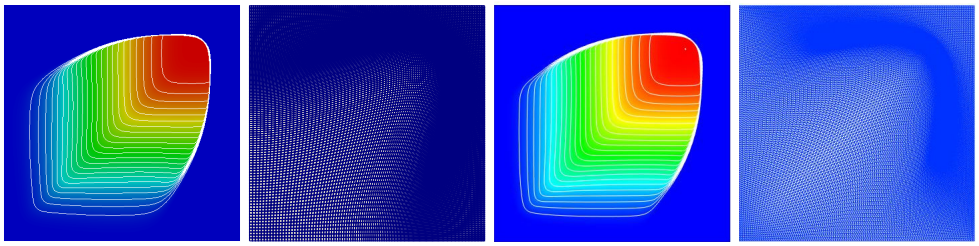

Fig. 6.1: Burgers equation. Left: $\mathbb{Q}_1$ FEM with 25 contours; Center left: Final $\mathbb{Q}_1$ mesh; Center right: $\mathbb{P}_1$ FEM with 25 contours; Right: Final $\mathbb{P}_1$ mesh.

6.2.3. Nonconvex flux. Our last scalar example is a nonlinear scalar conservation law with a non-convex flux

$$\partial_t u + \nabla \cdot \boldsymbol{f}(u) = 0, \quad u_0(\boldsymbol{x}) = 3.25\pi \mathbb{1}_{\|\boldsymbol{x}\|_{\ell^2}<1} + 0.25\pi. \tag{6.6}$$

where $\boldsymbol{f}(u) = (\sin u, \cos u)^\mathsf{T}$. This test, henceforth referred to as KPP, was proposed

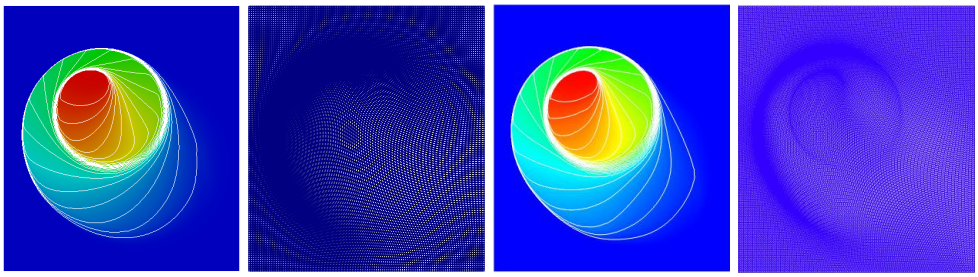

Fig. 6.2: KPP problem. Left: $\mathbb{Q}_1$ FEM with 25 contours; Center left: Final $\mathbb{Q}_1$ mesh; Center right: $\mathbb{P}_1$ FEM with 25 contours; Right: Final $\mathbb{P}_1$ mesh.

in Kurganov et al. [26]. It is a challenging test for many high-order numerical schemes because the solution has a two-dimensional composite wave structure. The initial computational domain is $D^0 = [-2.5, 1.5] \times [-2.0, 2.5]$. Note that the background velocity is constant and equal to $\boldsymbol{\beta} = (\frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2})^\mathsf{T}$. It can be shown that the computational domain keeps a rectangular shape in the time interval $(0, 1)$. The computation is done

up to time $T = 1$ using $\mathbb{Q}_1$ and $\mathbb{P}_1$ finite elements on structured meshes 128×128 with $CFL = 0.1$. The results are shown in Figure 6.2

6.3. Compressible Euler equations. We finish the series of tests by solving the compressible Euler equations in $\mathbb{R}^2$

$$\begin{cases} \partial_t \rho + \nabla\cdot(\rho \boldsymbol{u}) & = 0, \\ \partial_t(\rho \boldsymbol{u}) + \nabla\cdot(\rho \boldsymbol{u}\otimes\boldsymbol{u} + p\mathbb{I}) & = 0, \\ \partial_t E + \nabla\cdot(\boldsymbol{u}(E+p)) & = 0, \end{cases} \tag{6.7}$$

with an ideal gas equation of state, $p = (\gamma-1)(E - \frac{1}{2}\rho\|\boldsymbol{u}\|^2_{\ell^2})$ where $\gamma > 1$, and appropriate initial and boundary conditions. The motion of the mesh is done as described in (6.2) with $\boldsymbol{a}^{n+1}_{i,\mathrm{Lg}} = \boldsymbol{a}^n_i + \tau \boldsymbol{u}^n_h(\boldsymbol{a}^n_i)$ where $\boldsymbol{u}^n_h$ is the approximate fluid velocity at time t^n.

6.3.1. Sod problem. We use the so-called Sod shocktube problem to test the convergence of our algorithm (version-1 only); it is a Riemann problem with the following initial data:

$$\rho_0(\boldsymbol{x}) = 1.0, \quad \boldsymbol{u}_0(\boldsymbol{x}) = 0.0, \quad p_0(\boldsymbol{x}) = \mathbb{1}_{x_1<0.5} + 0.1\,\mathbb{1}_{x_1>0.5}. \tag{6.8}$$

and $\gamma = 1.4$, see e.g., Toro [34, §4.3.3]. The computational domain at the initial time is the unit square $(0,1)^2$. Dirichlet boundary conditions are enforced on the left and right sides of the domain and we do not enforce any boundary conditions on the upper and lower sides. The computation is done up to $T = 0.2$. Since no wave reaches the left and the right boundaries in the time interval $0 < t < T = 0.2$, the computational domain remains a square for the whole duration of the simulation. The solution being one-dimensional, the convergence tests are done on five meshes with refinements made only along the x_1-direction. These meshes have 20×4, 40×4, ..., 1280×4 cells and are uniform at $t = 0$. The results of the convergence test are shown in Table 6.3. We show in this table the L^1- and L^2-norm of the error on the density. The convergence orders are compatible with what is usually obtained in the literature for this problem.

Table 6.3: SOD problem, ALE, convergence test, $T = 0.2$, $CFL = 0.1$

	$\mathbb{Q}_1$				$\mathbb{P}_1$			
# dofs	L^2-norm		L^1-norm		L^2-norm		L^1-norm	
1605	2.47E-02	-	1.51E-02	-	2.82E-02	-	1.77E-02	-
3205	1.84E-02	0.43	9.99E-03	0.60	2.07E-02	0.45	1.15E-02	0.61
6405	1.36E-02	0.42	6.42E-03	0.64	1.56E-02	0.40	7.45E-03	0.63
12805	1.05E-02	0.39	4.07E-03	0.66	1.26E-02	0.32	4.82E-03	0.63

6.4. Noh problem. The last problem that we consider is the so-called Noh problem, see e.g., Caramana et al. [6, §5]. The computational domain at the initial time is $D^0 = (-1,1)^2$ and the initial data is

$$\rho_0(\boldsymbol{x}) = 1.0, \quad \boldsymbol{u}_0(\boldsymbol{x}) = -\frac{\boldsymbol{x}}{\|\boldsymbol{x}\|_{\ell^2}}, \quad p_0(\boldsymbol{x}) = 10^{-15}. \tag{6.9}$$

A Dirichlet boundary condition is enforced on all the dependent variables at the boundary of the domain for the entire simulation. We use $\gamma = \frac{5}{3}$. The solution to this problem is known; for instance, the density is equal to

$$\rho(t, \boldsymbol{x}) = 16\mathbb{1}_{\{\|\boldsymbol{x}\|_{\ell^2} < \frac{t}{3}\}} + (1 + \frac{t}{\|\boldsymbol{x}\|_{\ell^2}})\mathbb{1}_{\{\|\boldsymbol{x}\|_{\ell^2} > \frac{t}{3}\}}. \tag{6.10}$$

The ALE velocity at the boundary of the computational domain is prescribed to be equal to the fluid velocity, i.e., the boundary moves inwards in the radial direction with speed 1. The final time is chosen to be $T = 0.6$ in order to avoid that the shockwave collides with the moving boundary of the computational domain which happens at $t = \frac{3}{4}$ since the shock moves radially outwards with speed $\frac{1}{3}$.

We show in Table 6.4 the L^1- and the L^2-norm of the error on the density for various meshes which are uniform at $t = 0$: 30×30, 60×60, etc.

Table 6.4: Noh problem, convergence test, $T = 0.6$, $CFL = 0.2$

	$\mathbb{Q}_1$				$\mathbb{P}_1$			
# dofs	L^2-norm		L^1-norm		L^2-norm		L^1-norm	
961	2.60	-	1.44	-	2.89	-	1.71	-
3721	1.81	0.52	8.45E-01	0.77	2.21	0.39	1.09	0.64
14641	1.16	0.64	4.21E-01	1.01	1.42	0.64	5.15E-01	1.08
58081	7.66E-01	0.60	2.10E-01	0.99	9.39E-01	0.59	2.60E-01	0.99
231361	5.21E-01	0.56	1.06E-01	0.98	6.33E-01	0.57	1.28E-01	1.02

Preserving the radial symmetry of the solution as best as possible on non-uniform meshes is an important property for Lagrangian hydrocodes in the context of the inertial confinement fusion project, which involves simulating implosion problems, see Caramana et al. [6]. In these problems, mesh-induced violation of the spherical symmetry may artificially trigger the Rayleigh-Taylor instability and thereby may hamper the understanding of the real dynamics of the implosion. We show in the top row of Figure 6.3 simulations that are done on a uniform mesh composed of 96×96 square cells for the $\mathbb{Q}_1$ approximation ((2×96)×(2×96) triangular cells for the $\mathbb{P}_1$ approximation), and we compare them with simulations done on a nonuniform mesh constructed as follows: The initial square D^0 is divided in four quadrants; in the bottom left quadrant the mesh is composed of 32×32 square cells; in the top left quadrant the mesh is composed of 32×64 rectangular cells; in the top right quadrant the mesh is composed of 64×64 square cells; the bottom right quadrant is composed of 64×32 rectangular cells. This is a generic test for many Lagrangian hydrocodes, see e.g., Dobrev et al. [13, §8.4]. We notice a slight break of symmetry, but the solution does not develop any Rayleigh-Taylor-type instability as it is often the case for many other Lagrangian algorithms.

We show in Figure 6.4 a zoom around the center of the computational domain for both the $\mathbb{Q}_1$ and the $\mathbb{P}_1$ approximations. We notice a slight motion of the center, but there is no dramatic breakdown of the structure of the solution.

7. Concluding remarks. In this paper we have developed a framework for constructing ALE algorithms using continuous finite elements. The method is invariant domain preserving on any mesh in arbitrary space dimension. The methodology applies to any hyperbolic system which has such intrinsic property. If the system at

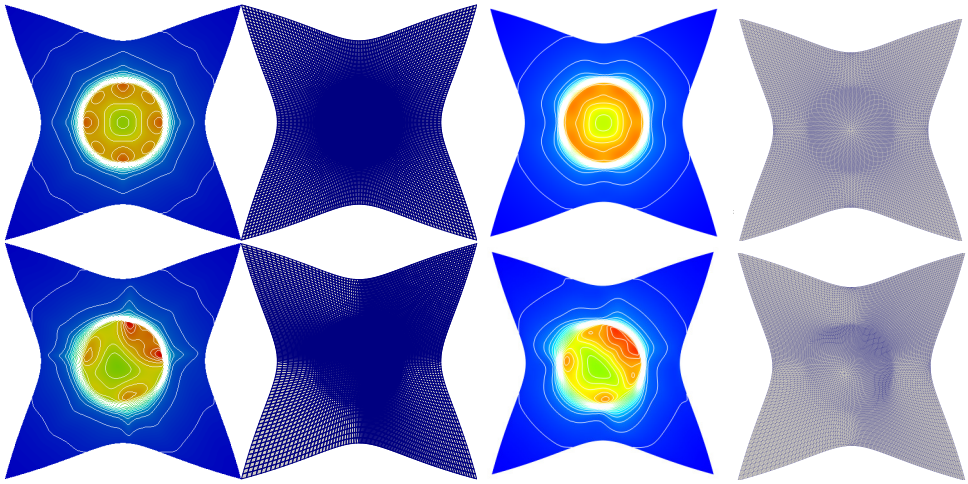

Fig. 6.3: Noh problem. Uniform meshes in top row, nonuniform meshes in bottom row. From left to right: density field with $\mathbb{Q}_1$ approximation (25 contour lines); mesh with $\mathbb{Q}_1$ approximation; density field with $\mathbb{P}_1$ approximation (25 contour lines); mesh with $\mathbb{P}_1$ approximation.

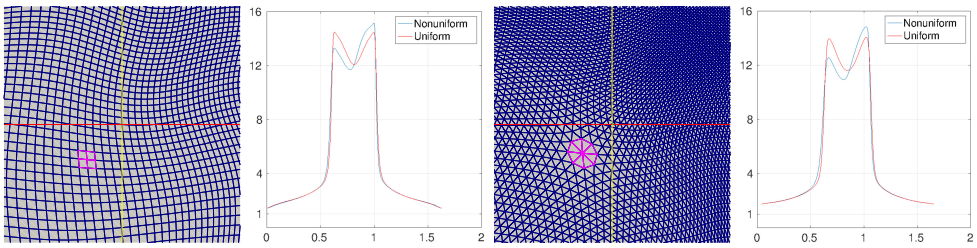

Fig. 6.4: Noh problem. From left to right: Zoom around the center of the nonuniform $\mathbb{Q}_1$ mesh at $T = 0.6$ (notice the small displacement of the center); Cross section along the line connecting the points $(-1,-1)$ and $(1,1)$ for the $\mathbb{Q}_1$ solutions on the uniform mesh and on the nonuniform mesh; Zoom around the center of the nonuniform $\mathbb{P}_1$ mesh at $T = 0.6$; Cross section along the line connecting the points $(-1,-1)$ and $(1,1)$ for the $\mathbb{P}_1$ solutions on the uniform mesh and on the nonuniform mesh;

hand has an entropy pair, then the method also satisfies a discrete entropy inequality. The time accuracy of one of the methods (scheme 1) can be increased by using SSP time discretization techniques. This makes the method appropriate to use as a safe-guard when constructing high-order accurate discretization of the system which may violate the invariant domain property. The equivalence between the conservative and non-conservative formulations implies the DGCL condition (preservation of constant states). The new methods have been tested on a series of benchmark problems and the observed convergence orders and numerical performance are compatible with what is reported in the literature.

References.

[1] M. Ainsworth. Pyramid algorithms for Bernstein-Bézier finite elements of high,

nonuniform order in any dimension. *SIAM J. Sci. Comput.*, 36(2):A543–A569, 2014.

[2] W. Bangerth, R. Hartmann, and G. Kanschat. deal.II – a general purpose object oriented finite element library. *ACM Trans. Math. Softw.*, 33(4):24/1–24/27, 2007.

[3] A. Barlow. A compatible finite element multi-material ALE hydrodynamics algorithm. *Internat. J. Numer. Methods Fluids*, 56:953–964, 2007.

[4] W. Boscheri and M. Dumbser. A direct arbitrary-Lagrangian-Eulerian ADER-WENO finite volume scheme on unstructured tetrahedral meshes for conservative and non-conservative hyperbolic systems in 3D. *J. Comput. Phys.*, 275:484–523, 2014.

[5] A. Bressan. *Hyperbolic systems of conservation laws*, volume 20 of *Oxford Lecture Series in Mathematics and its Applications*. Oxford University Press, Oxford, 2000. The one-dimensional Cauchy problem.

[6] E. Caramana, M. Shashkov, and P. Whalen. Formulations of artificial viscosity for multi-dimensional shock wave computations. *J. Comput. Phys.*, 144(1):70–97, 1998.

[7] E. J. Caramana and R. Loubère. "Curl-q": a vorticity damping artificial viscosity for essentially irrotational Lagrangian hydrodynamics calculations. *J. Comput. Phys.*, 215(2):385–391, 2006.

[8] G. Carré, S. Del Pino, B. Després, and E. Labourasse. A cell-centered Lagrangian hydrodynamics scheme on general unstructured meshes in arbitrary dimension. *J. Comput. Phys.*, 228(14):5160–5183, 2009.

[9] G.-Q. Chen. Euler equations and related hyperbolic conservation laws. In *Evolutionary equations. Vol. II*, Handb. Differ. Equ., pages 1–104. Elsevier/North-Holland, Amsterdam, 2005.

[10] J. Cheng and C.-W. Shu. Positivity-preserving Lagrangian scheme for multi-material compressible flow. *J. Comput. Phys.*, 257(part A):143–168, 2014.

[11] K. N. Chueh, C. C. Conley, and J. A. Smoller. Positively invariant regions for systems of nonlinear diffusion equations. *Indiana Univ. Math. J.*, 26(2):373–392, 1977.

[12] C. M. Dafermos. *Hyperbolic conservation laws in continuum physics*, volume 325 of *Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]*. Springer-Verlag, Berlin, 2000.

[13] V. Dobrev, T. Kolev, and R. Rieben. High-order curvilinear finite element methods for Lagrangian hydrodynamics. *SIAM J. Sci. Comput.*, 34(5):B606–B641, 2012.

[14] C. Farhat, P. Geuzaine, and C. Grandmont. The discrete geometric conservation law and the nonlinear stability of ALE schemes for the solution of flow problems on moving grids. *J. Comput. Phys.*, 174(2):669–694, 2001.

[15] L. Ferracina and M. N. Spijker. An extension and analysis of the Shu-Osher representation of Runge-Kutta methods. *Math. Comp.*, 74(249):201–219, 2005.

[16] H. Frid. Maps of convex sets and invariant regions for finite-difference systems of conservation laws. *Arch. Ration. Mech. Anal.*, 160(3):245–269, 2001.

[17] L. Gastaldi. *A priori* error estimates for the Arbitrary Lagrangian Eulerian formulation with finite elements. *J. Num. Math.*, 9(2):123–156, 2001.

[18] S. Gottlieb, D. I. Ketcheson, and C.-W. Shu. High order strong stability preserving time discretizations. *J. Sci. Comput.*, 38(3):251–289, 2009.

[19] J.-L. Guermond and B. Popov. Fast estimation of the maximum wave speed in the

Riemann problem for the euler equations. 2015. arXiv:1511.02756, Submitted.

[20] J.-L. Guermond and B. Popov. Invariant domains and first-order continuous finite element approximation for hyperbolic systems. 2015. arXiv:1509.07461, Submitted.

[21] J.-L. Guermond, B. Popov, and V. Tomov. Entropy-viscosity method for the single material euler equations in Lagrangian frame. *Computer Methods in Applied Mechanics and Engineering*, 300:402 – 426, 2016.

[22] H. Guillard and C. Farhat. On the significance of the geometric conservation law for flow computations on moving meshes. *Comput. Methods Appl. Mech. Engrg.*, 190(11-12):1467–1482, 2000.

[23] I. Higueras. Representations of Runge-Kutta methods and strong stability preserving methods. *SIAM J. Numer. Anal.*, 43(3):924–948, 2005.

[24] D. Hoff. Invariant regions for systems of conservation laws. *Trans. Amer. Math. Soc.*, 289(2):591–610, 1985.

[25] T. Kolev and R. Rieben. A tensor artificial viscosity using a finite element approach. *J. Comput. Phys.*, 228(22):8336–8366, 2009.

[26] A. Kurganov, G. Petrova, and B. Popov. Adaptive semidiscrete central-upwind schemes for nonconvex hyperbolic conservation laws. *SIAM J. Sci. Comput.*, 29 (6):2381–2401 (electronic), 2007.

[27] M.-J. Lai and L. L. Schumaker. *Spline functions on triangulations*, volume 110 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, Cambridge, 2007.

[28] P. Lax. Weak solutions of nonlinear hyperbolic equations and their numerical computation. *Comm. Pure Appl. Math.*, 7:159–193, 1954.

[29] P. D. Lax. Hyperbolic systems of conservation laws. II. *Comm. Pure Appl. Math.*, 10:537–566, 1957.

[30] K. Lipnikov and M. Shashkov. A framework for developing a mimetic tensor artificial viscosity for Lagrangian hydrocodes on arbitrary polygonal meshes. *J. Comput. Phys.*, 229(20):7911–7941, 2010.

[31] R. Loubère, P.-H. Maire, M. Shashkov, J. Breil, and S. Galera. ReALE: A reconnection-based arbitrary-Lagrangian-Eulerian method. *Journal of Computational Physics*, 229(12):4724 – 4761, 2010.

[32] G. Scovazzi, E. Love, and M. Shashkov. Multi-scale Lagrangian shock hydrodynamics on Q1/P0 finite elements: Theoretical framework and two-dimensional computations. *Comput. Methods Appl. Mech. Engrg.*, 197:1056–1079, 2008.

[33] M. Shashkov and J. Campbell. A tensor artificial viscosity using a mimetic finite difference algorithm. *J. Comput. Phys.*, 172(2):739–765, 2001.

[34] E. F. Toro. *Riemann solvers and numerical methods for fluid dynamics*. Springer-Verlag, Berlin, third edition, 2009. A practical introduction.

[35] F. Vilar, P.-H. Maire, and R. Abgrall. A discontinuous Galerkin discretization for solving the two-dimensional gas dynamics equations written under total Lagrangian formulation on general unstructured grids. *J. Comput. Phys.*, 276(0): 188 – 234, 2014.

[36] B. Wells, M. Baines, and P. Glaister. Generation of Arbitrary Lagrangian-Eulerian (ALE) velocities, based on monitor functions, for the solution of compressible uid equations. *Int. J. Numer. Meth. Fluids*, 47:1375–1381, 2005.

[37] Z. Yang and D. J. Mavriplis. Unstructured dynamic meshes with higher-order time integration schemes for the unsteady Navier-Stokes equations. AIAA Paper 2005–1222, 2005.