Journal of Computational Physics 228 (2009) 5160–5183

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

A cell-centered Lagrangian hydrodynamics scheme on general unstructured meshes in arbitrary dimension

G. Carré, S. Del Pino, B. Després *, E. Labourasse

CEA/DIF, Bruyères le Chatel, 91297 Arpajon Cedex, France

ARTICLE INFO

Article history:
Received 9 July 2008
Received in revised form 6 April 2009
Accepted 8 April 2009
Available online 22 April 2009

Keywords:
Lagrangian scheme
Compressible gas dynamics
Godunov scheme

ABSTRACT

We describe a cell-centered Godunov scheme for Lagrangian gas dynamics on general unstructured meshes in arbitrary dimension. The construction of the scheme is based upon the definition of some geometric vectors which are defined on a moving mesh. The finite volume solver is node based and compatible with the mesh displacement. We also discuss boundary conditions. Numerical results on basic 3D tests problems show the efficiency of this approach. We also consider a quasi-incompressible test problem for which our nodal solver gives very good results if compared with other Godunov solvers. We briefly discuss the compatibility with ALE and/or AMR techniques at the end of this work. We detail the coefficients of the isoparametric element in the appendix.

1. Introduction

In this work we describe a method to solve the equations of Lagrangian hydrodynamics [3,28] on general unstructured meshes in arbitrary dimension. The physical variables are the density ρ, the velocity $\mathbf{u}$, the total energy e and the pressure p. In integral form these equations are

$$\begin{cases} \frac{D}{Dt}\int_{V_j(t)} \rho\, dV = 0, & \text{mass conservation,} \\ \frac{D}{Dt}\int_{V_j(t)} \rho\mathbf{u}\, dV + \int_{S_j(t)} p\mathbf{n}\, dS = \mathbf{0}, & \text{momentum conservation,} \\ \frac{D}{Dt}\int_{V_j(t)} \rho e\, dV + \int_{S_j(t)} p\mathbf{u}.\mathbf{n}\, dS = 0, & \text{total energy conservation,} \\ \frac{D}{Dt}\int_{V_j(t)} dV - \int_{S_j(t)} \mathbf{u}.\mathbf{n}\, dS = 0, & \text{volume conservation.} \end{cases} \tag{1}$$

Here $V_j(t)$ is a moving domain, in practice it is a moving cell with constant mass. The operator $\frac{D}{Dt} = \partial_t + \mathbf{u}\cdot\nabla$ is the material derivative. The first three equations are physical equations, and the last one is a geometric one.

We call the scheme **GLACE** since it is a **G**odunov method (in dimension one the scheme is equal to the acoustic Godunov scheme [17,18]), the scheme is **LA**grangian, and finally the method is **C**onservative for the total **E**nergy variable. The method is cell-centered like any Godunov type scheme [17,18,37,20].

Classically, Lagrangian hydrodynamics equations are solved with staggered schemes (that is kinematic variables at nodes and thermodynamic variables within the cells), following the seminal idea of Von Neumann [26]. We refer to the works [4,3,9,6,16,15,27,39] and references therein for a presentation of this topic. These methods require the use of artificial viscosity techniques to capture shocks. The so-called compatible hydrodynamics [6] Lagrangian scheme is a recent improvement: it is by construction conservative for all physical variables. To our knowledge this is the only staggered Lagrangian

* Corresponding author.
E-mail address: bruno.despres@cea.fr (B. Després).

0021-9991/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2009.04.015

scheme with this highly desirable property. Another approach that we do not discuss in this paper is the use of a Finite Element formulation of Lagrangian hydrodynamics, see [33] where such a method (also staggered for the final scheme, and conservative for all physical variables) is described. We also mention the work on stabilized and variational multi-scale formulations for Lagrangian hydrodynamics in [32,31,29,30], where a globally conservative formulation is obtained from a nodal-based finite element method.

The *a priori* good properties of Godunov cell-centered methods are: a natural conservation of physical quantities; no need for artificial viscosity techniques; easy implementation of fully conservative remapping and remeshing techniques. The method described in this paper originates from a series of work [21,25,13,22] in which a new approach for the definition of cell-centered Godunov schemes for Lagrangian hydrodynamics is pursued. In particular we follow [12,13], further extended in [22], where the idea of having nodes based fluxes is developed. This was recognized as a fundamental issue for the design of Godunov schemes for Lagrangian hydrodynamics. On the contrary the Caveat approach [1], see also [3], suffers from serious theoretical incompatibilities mainly because the fluxes are edge based and not node based. In a different direction we also refer to the recent work [10] in which a new class of ENO Lagrangian type schemes is developed for quadrilateral cells and to [11] where an analysis of Godunov schemes on staggered meshes is performed.

In the following, we describe a general formalism which aims at simplifying the understanding of what is a cell-centered Godunov scheme for Lagrangian hydrodynamics in any dimension on general polygonal (or polyhedral) meshes: the cells can have warped faces as well in this new formalism, which was not the case in [13]. The presentation is no more partial differential equations (PDE) based as it was the case in [13,19] for instance. Instead we directly consider some basic formulas that one can write for a moving mesh. This constitutes a framework which can provide a toolbox for the discretization of PDE operators on moving meshes. It shows the importance of some vectors $\mathbf{C}_{jr}$ (j stands for the cells, r stands for the vertices) which contain all the needed geometric information. These vectors are all we need for the discretization of the divergence operator and the gradient operator. Our aim is that it will be an aid to practitioners because these definitions are immediate from the geometry of the mesh. In some sense the nodal solver that we construct is a kind of multiD Riemann solver and is a Lagrangian generalization of Eulerian 1D Riemann solvers for which we refer the reader to the standard textbooks [37,20]. This solver is not a cure for the well known problems attached to Godunov solvers: on 1D test problems it behaves like any other Godunov solvers. Our concern is the multiD generalization on general meshes. The design principle of our approach makes a priori possible the use of any type of cells, not only simplicial meshes or combination of simplicial elements as in the reference [24]. Simplicial elements are very convenient in order to have an immediate geometric interpretation. However in some 3D cases simplicial elements introduce an arbitrary non symmetric decomposition of the faces into triangles (particularly in the case where all the nodes of the triangles must be vertices of the mesh, as in [24]). It can break the natural symmetries of the problem and this is something we do not want to do for the kind of 3D meshes we use in the numerical section. In the numerical section basic examples show indeed a very good behavior for meshes with such a symmetry. In the appendix we detail the formulas for the 3D isoparametric element [36,14].

The outline of this work is the following. In Section 2 we define and explain the basic properties of the vectors $\mathbf{C}_{jr}$. The next section is devoted to the presentation of the first-order GLACE scheme. Section 4 focuses on the discretization of boundary conditions. Then we give some theoretical properties of the scheme in Section 5. Time step considerations are developed in Section 6. The second-order "Monotone Upstream-centered Scheme for Conservation Laws" (MUSCL) extension is given in Section 7. Some basic test problems are discussed in the numerical experiments Section 8. Finally the appendix gives more technical material about the definition of the vectors $\mathbf{C}_{jr}$ for isoparametric cells. We adopt the notation that all vectors and matrices are bold faces. We use the convention that the time derivative of $t \mapsto f(t)$ is f', while the time derivative of a composed function $g(\mathbf{x}(t))$ is $\frac{d}{dt}g = (\mathbf{x}', \nabla g)$.

2. Geometry on a moving mesh

The aim of this section is to provide basic relations on a moving geometry, which will be the foundations of the PDE discretization. Let us consider a computational domain $\mathscr{V} \subset \mathbb{R}^d$. This domain is composed of cells indexed by j. In practice the dimension of the problem can be $d = 1, 2$ or 3. In the sequel we refer systematically to the volume of the cell. This is a convenient abuse of notation. In dimension $d = 2$ the volume becomes an area. In dimension $d = 1$ it is a length.

In a Lagrangian computation the mesh moves with the flow. The volume of cell j at time $t_k = t_{k-1} + \Delta t$ is denoted as V_j^k. The vertices of the mesh are nodes $\mathbf{x}_r^k \in \mathbb{R}^d$. Let us denote $\mathbf{x}^k = (\mathbf{x}_1^k, \ldots, \mathbf{x}_r^k, \ldots)$ the collection of all vertices. We assume that the volume V_j^k is defined as a function of the vertices $\mathbf{x}^k \mapsto V_j(\mathbf{x}^k)$. It means that we have a formula to do so, whatever this formula is. In the sequel we concentrate on the consequences of this assumption. For the simplicity of the presentation, we shall also consider the semi-discrete scheme which is continuous in time and fully discrete in space: in this case the volume is $V_j(\mathbf{x}(t))$.

A first problem is to compute the derivative $\frac{d}{dt}V_j$ of the volume as a function of the nodal velocities $\mathbf{u}_r(t) = \mathbf{x}_r'(t)$. This problem is fundamental in any Lagrangian computation. Let us define the gradient of the volume V_j with respect to the nodal positions $\mathbf{x}_r$

$$\mathbf{C}_{jr} = \nabla_{\mathbf{x}_r} V_j = \left(\frac{\partial}{\partial \mathbf{x}_{r,1}} V_j, \ldots, \frac{\partial}{\partial \mathbf{x}_{r,d}} V_j \right)^t \in \mathbb{R}^d. \tag{2}$$

This definition is the cornerstone of our approach, as in the seminal work [38] (more references to be found in [6,24]). By construction one has

$$\frac{d}{dt}V_j = (\nabla_{\mathbf{x}} V_j, \mathbf{x}') = \sum_r (\mathbf{C}_{jr}, \mathbf{u}_r). \tag{3}$$

This equation expresses that the change of volume is due to a discrete divergence, see the last equation of (1). By duality it can be used to define a discrete gradient.

Also fundamental is the homogeneity of the volume with respect to the vertices $V_j(\lambda \mathbf{x}) = \lambda^d V_j(\mathbf{x})$: for example in dimension $d = 3$ a scaling of the mesh size by a factor 2 implies a multiplication by $2^3 = 8$ of all volumes. The Euler relation for homogeneous functions holds, so

$$V_j = \frac{1}{d}(\nabla_{\mathbf{x}} V_j, \mathbf{x}) = \frac{1}{d}\sum_r (\mathbf{C}_{jr}, \mathbf{x}_r). \tag{4}$$

It means that the knowledge of the geometrical vectors $\mathbf{C}_{jr}$ is all we need to compute the volume of the cell and the variation in time of this volume. Two additional properties can be deduced.

Proposition 1. *For all cell j one has*

$$\sum_r \mathbf{C}_{jr} = \mathbf{0}. \tag{5}$$

If the vertex $\mathbf{x}_r$ is in the interior of the computational domain and the volumes of the cells are compatible (that is their sum is equal to the total volume) then

$$\sum_j \mathbf{C}_{jr} = \mathbf{0}. \tag{6}$$

An interpretation of these properties is possible for simple geometries. We will show with an example in dimension $d = 2$ (see Section 2.3) that these algebraic properties are equivalent to a finite volume geometrical integration. In simple cases, Eq. (6) can also be deduced from (5) after the definition of control volumes around the vertices.

Consider an arbitrary constant velocity field $\mathbf{u}_r = \mathbf{u}_{\text{ref}}$. Then the volume of any cell is constant that is $0 = \frac{d}{dt}V_j = \sum_r (\mathbf{C}_{jr}, \mathbf{u}_{\text{ref}})$ for all $\mathbf{u}_{\text{ref}} \in \mathbb{R}^d$. It proves relation (5). It remains to prove relation (6). Assume that the total volume $\mathscr{V} = \sum_j V_j$ is constant: $0 = \frac{d}{dt}\mathscr{V} = \sum_j \frac{d}{dt}V_j = \sum_j (\sum_r (\mathbf{C}_{jr}, \mathbf{u}_r))$, that is $0 = \sum_{s \neq r} \left(\sum_j (\mathbf{C}_{js}, \mathbf{u}_s)\right) + \sum_j (\mathbf{C}_{jr}, \mathbf{u}_r)$. Set $\mathbf{u}_s = \mathbf{0}$ for $s \neq r$, and $\mathbf{u}_r \neq \mathbf{0}$ arbitrary for an interior node: it implies that $\sum_j \mathbf{C}_{jr} = \mathbf{0}$ which proves the result.

We detail the computation of $\mathbf{C}_{jr}$ in the subsequent paragraphs. In dimension $d = 1$ the length of a cell is uniquely defined. The area of a polygonal cell is also uniquely defined according to the coordinates of its vertices in dimension $d = 2$. So $\mathbf{C}_{jr}$ is uniquely defined for $d = 1, 2$. In dimension $d = 3$, several constructions can be derived to describe the geometry of a polyhedron defined by its vertices (this is essentially related to the presence of non-planar faces). For instance, one can consider that an hexahedron is the image by an isoparametric transformation of the unit cube. But it can also be defined as the union of several tetrahedra. In Appendix A we construct $\mathbf{C}_{jr}$ for the particular case of hexahedron Fig. 1: one is based on a isoparametric representation. In all cases once the choice of a representation has been made, the $\mathbf{C}_{jr}$ is uniquely defined by relation (2).

2.1. Computation of $\mathbf{C}_{jr}$ in 1D

To simplify, we denote by $r = j + \frac{1}{2}$ the vertex between j and $j + 1$. Then of course $V_j = x_{j+\frac{1}{2}} - x_{j-\frac{1}{2}}$. So $\mathbf{C}_{jj+\frac{1}{2}} = 1$ and $\mathbf{C}_{jj-\frac{1}{2}} = -1$.

2.2. Computation of $\mathbf{C}_{jr}$ in the planar 2D case

Consider the typical situation of Fig. 2. By convention the vertices are listed counterclockwise $\mathbf{x}_{r-1}, \mathbf{x}_r, \mathbf{x}_{r+1}, \ldots$ with coordinates $\mathbf{x}_r = (x_r, y_r)$. The quantity $\frac{1}{2}(x_r y_{r+1} - y_r x_{r+1})$ is the oriented area of the triangle with vertices $\mathbf{x}_r, \mathbf{x}_{r+1}$ and $O = (0, 0)$. The sum of these oriented area is the total area $V_j = \sum_r \frac{1}{2}(x_r y_{r+1} - y_r x_{r+1})$. Therefore the formula (2) implies (after elementary manipulations) the formula used in [13]

$$\mathbf{C}_{jr} = \frac{1}{2}(-y_{r-1} + y_{r+1}, x_{r-1} - x_{r+1})^t. \tag{7}$$

2.3. Geometrical interpretation of (5)–(7) in the planar 2D case

For the completeness of the presentation, we give a geometrical interpretation of (5)–(7) in dimension 2, filling the gap with the standard Finite Volume approach developed for example in [13,22]. Consider Fig. 3 where edges have been cut into two half pieces. For example cells p and k are neighbors to the cell j. The interface between j and p (resp. k) is characterized by

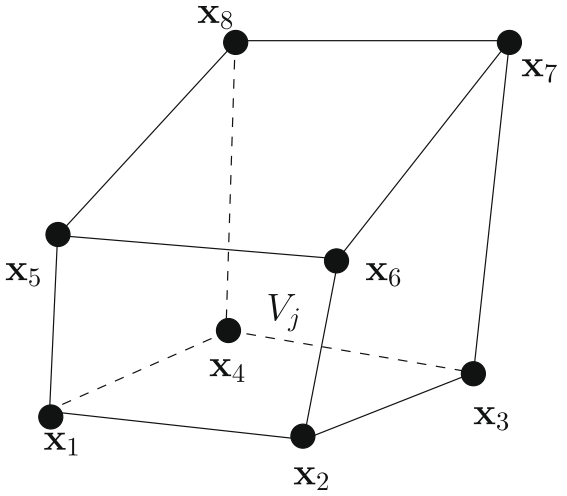


Fig. 1. In dimension $d = 3$, an hexahedron V_j and its 8 vertices.

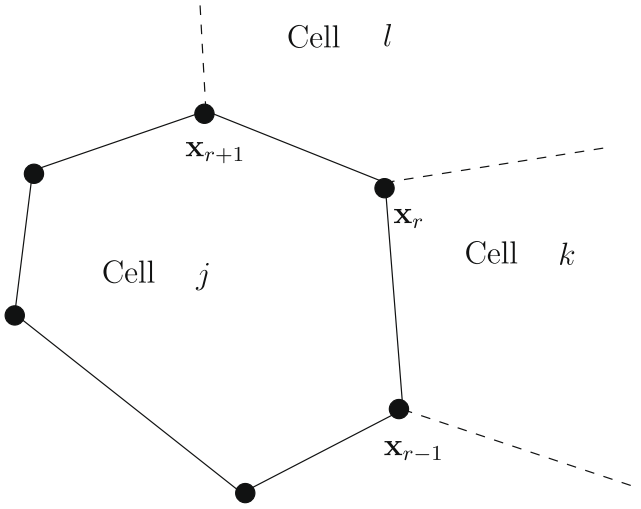


Fig. 2. A mesh in dimension $d = 2$.

the vector $l_{jp}\mathbf{n}_{jp}$ (resp. $l_{jk}\mathbf{n}_{jk}$): it is the product of the length of the interface by the normal. If one cuts these edges in two pieces and glue them at the corner then one gets

$$l_{jr}\mathbf{n}_{jr} = \frac{1}{2}l_{jp}\mathbf{n}_{jp} + \frac{1}{2}l_{jk}\mathbf{n}_{jk}. \tag{8}$$

The unit vector $\mathbf{n}_{jr}$ is a definition of a normal at the node, and l_{jr} is the half of the length of the segment $[\mathbf{x}_{r-1}, \mathbf{x}_{r+1}]$. By construction one has

$$l_{jr}\mathbf{n}_{jr} = \frac{1}{2}(-y_{r-1} + y_{r+1}, x_{r-1} - x_{r+1})^t = \mathbf{C}_{jr}.$$

In this framework the relation (5) simply means that the sum of the outgoing normals multiplied by the length of the interfaces is equal to zero. It is a well known property of any finite volume method: the cell j is a closed contour.

Consider now Fig. 4 where we have displayed in bold the pieces of boundaries which are used to define $l_{jr}\mathbf{n}_{jr}$ but also $l_{pr}\mathbf{n}_{pr}$ and $l_{kr}\mathbf{n}_{kr}$. It is clear by construction that $l_{jr}\mathbf{n}_{jr} + l_{pr}\mathbf{n}_{pr} + l_{kr}\mathbf{n}_{kr} = \left(\frac{1}{2}l_{jp}\mathbf{n}_{jp} + \frac{1}{2}l_{jk}\mathbf{n}_{jk}\right) + \left(\frac{1}{2}l_{pj}\mathbf{n}_{pj} + \frac{1}{2}l_{pk}\mathbf{n}_{pk}\right) + \left(\frac{1}{2}l_{kp}\mathbf{n}_{kp} + \frac{1}{2}l_{kj}\mathbf{n}_{kj}\right)$. However, since $l_{jp}\mathbf{n}_{jp} + l_{pj}\mathbf{n}_{pj} = 0$, all terms cancel and therefore $l_{jr}\mathbf{n}_{jr} + l_{pr}\mathbf{n}_{pr} + l_{kr}\mathbf{n}_{kr} = 0$. This equality is the geometrical interpretation of the general property (6).

In theory it is also possible to reinterpret the standard finite volume construction in our framework: by a standard Finite Volume method we mean a scheme for which the fluxes are computed and applied through the edges. Geometrically it corresponds to the Fig. 5. Then (5) is a sum over all the neighboring cells: it is still zero since the cell is a closed contour. On the other hand (6) becomes $l_{jk}\mathbf{n}_{jk} + l_{kj}\mathbf{n}_{kj} = 0$ for all k. A partial conclusion of the previous discussion is that node based

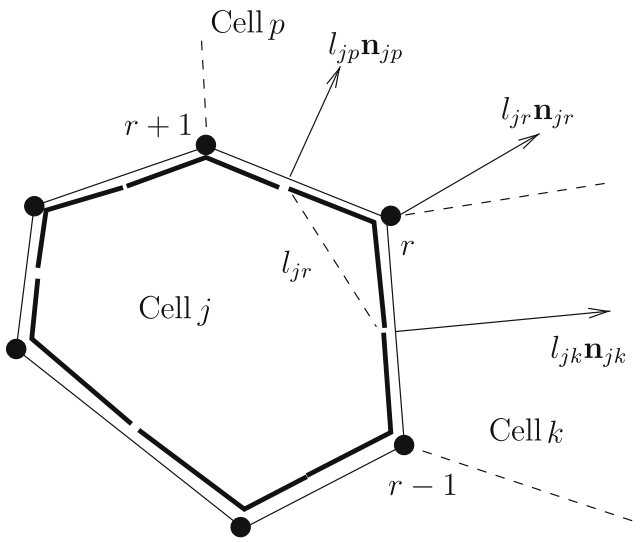


Fig. 3. Cutting the edges in two pieces.

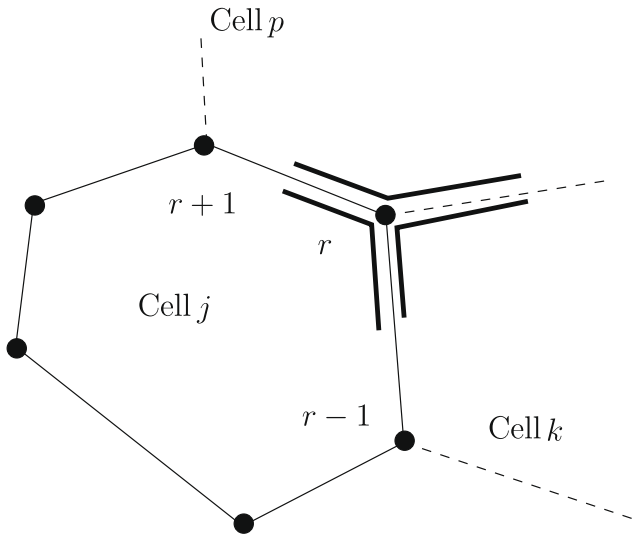


Fig. 4. Checking formula (6).

quantities are convenient for 2D Lagrangian computations. So we base our extension in dimension $d = 3$ directly on the node based vectors $\mathbf{C}_{jr}$.

2.4. Computation of $\mathbf{C}_{jr}$ in 3D

In dimension $d = 3$, the computation of the vectors $\mathbf{C}_{jr}$ is slightly more difficult. Indeed non-planar faces are possible: the computation of the total volume of a cell requires to define precisely the geometry of the polyhedron as it was discussed before. Assuming that one is able to decompose the volume in a sum of tetrahedra, then it is enough to compute the $\mathbf{C}_{jr}$ in a consistent way for all tetrahedra and then to sum up these vectors. In Appendix A we indicate possibilities for the computation of the $\mathbf{C}_{jr}$ for hexahedrons with warped faces. The case of a tetrahedron is based on the following formulas. Consider a tetrahedron j with vertices $r = 1, 2, 3$ and 4. One has

$$V_j = \frac{1}{6}\begin{vmatrix} x_2 - x_1 & x_3 - x_1 & x_4 - x_1 \\ y_2 - y_1 & y_3 - y_1 & y_4 - y_1 \\ z_2 - z_1 & z_3 - z_1 & z_4 - z_1 \end{vmatrix} = \frac{1}{6}\begin{vmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \\ z_1 & z_2 & z_3 & z_4 \\ 1 & 1 & 1 & 1 \end{vmatrix}.$$

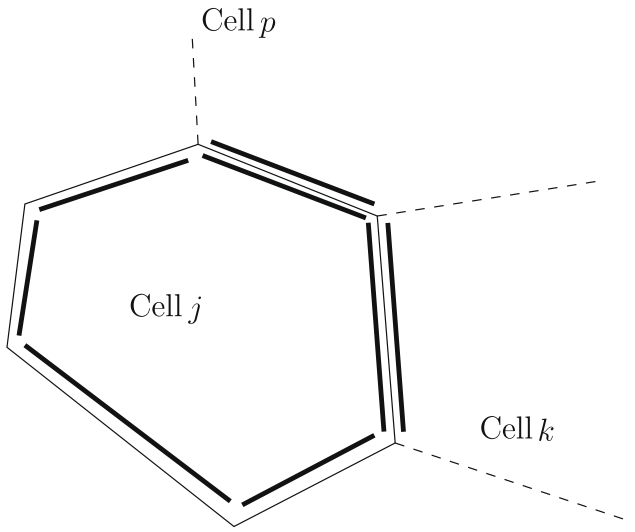


Fig. 5. Standard finite volume construction.

So using (2) and differentiating according to $\mathbf{x}_1$, one gets

$$\mathbf{C}_{j1} = \frac{1}{6}\left(\begin{vmatrix} y_2 & y_3 & y_4 \\ z_2 & z_3 & z_4 \\ 1 & 1 & 1 \end{vmatrix}, -\begin{vmatrix} x_2 & x_3 & x_4 \\ z_2 & z_3 & z_4 \\ 1 & 1 & 1 \end{vmatrix}, \begin{vmatrix} x_2 & x_3 & x_4 \\ y_2 & y_3 & y_4 \\ 1 & 1 & 1 \end{vmatrix}\right)^t.$$

One notes that $\mathbf{C}_{j1}$ is normal to the opposite face $(\mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4)$ of $\mathbf{x}_1$. Its norm is one third of the area of the opposite face. The other vectors $\mathbf{C}_{jr}$ for $r = 2, 3, 4$ are identical up to a permutation of the indices.

3. The GLACE scheme

The GLACE scheme is a one step forward Euler method that discretizes the Lagrangian equations (1). The only information we need at the beginning of time step k is the knowledge of the vertices $\mathbf{x}_r^k$ of the mesh, the vectors $\mathbf{C}_{jr}^k$ and the values of the physical unknowns inside the cells

$$\rho_j^k, \mathbf{u}_j^k, e_j^k, \quad \forall j.$$

The unknowns are constants in each cell, that is the scheme is cell-centered. We show below how to compute the unknowns at the end of the time step

$$\rho_j^{k+1}, \mathbf{u}_j^{k+1}, e_j^{k+1}, \quad \forall j,$$

in a compatible way with the mesh displacement $\mathbf{x}_r^{k+1} = \mathbf{x}_r^k + \Delta t\, \mathbf{u}_r^k$ where the node velocity $\mathbf{u}_r^k$ is to be defined.

3.1. Discretization of the divergence operator

The formula for the variation of the volume implicitly contains a discretization of the divergence operator. For the sake of simplicity of the presentation, we discuss this property in the semi-discrete context. Indeed the mass of a Lagrangian cell is constant in time: $V_j(\mathbf{x}(t))\rho_j(t) = M_j$ is independent of the time t. Let us now define $\tau_j = \frac{1}{\rho_j}$ the specific volume. Then

$$M_j \tau_j'(t) = \frac{d}{dt} V_j(\mathbf{x}(t)) = \sum_r (\mathbf{C}_{jr}, \mathbf{u}_r).$$

The hydrodynamics equations (1) imply $\rho \frac{d}{dt}\tau = \nabla \cdot \mathbf{u}$. It means that $\sum_r (\mathbf{C}_{jr}, \mathbf{u}_r)$ is an approximation in the finite volume sense of $\nabla \cdot \mathbf{u}$ over the moving cell j

$$\int_{V_j} \nabla \cdot \mathbf{u}\, dx = \int_{\partial V_j} (\mathbf{u}, \mathbf{n})\, d\sigma \approx \sum_r (\mathbf{C}_{jr}, \mathbf{u}_r).$$

3.2. Discretization of the gradient operator

Consider the bilinear form $(\mathbf{u}, p) \mapsto \int_{\partial V_j} (\mathbf{u}, \mathbf{n}) p\, d\sigma$ which represents the work of the pressure across the moving boundary. At the discrete level this quantity is

$$\int_{V_j} \nabla \cdot (p\mathbf{u}) d\sigma = \int_{\partial V_j} (\mathbf{u}, \mathbf{n}) p \, d\sigma \approx \sum_r (\mathbf{C}_{jr}, \mathbf{u}_r) p_{jr} = \sum_r (\mathbf{u}_r, \mathbf{C}_{jr} p_{jr}),$$

where p_{jr} is the pressure at vertex $\mathbf{x}_r$ seen from the cell j which will be defined later. Assuming that the velocity is constant (=$\mathbf{u}$), one gets

$$\int_{V_j} \nabla \cdot (p\mathbf{u}) d\sigma = \left(\mathbf{u}, \int_{V_j} \nabla p \, dx \right) \approx \left(\mathbf{u}, \sum_r \mathbf{C}_{jr} p_{jr} \right).$$

It implies that

$$\int_{V_j} \nabla p \, dx = \int_{\partial V_j} \mathbf{n} p d\sigma \approx \sum_r \mathbf{C}_{jr} p_{jr}$$

is the discrete approximation of the gradient of the pressure. The discrete gradient operator is the adjoint of the discrete divergence operator, as in the support operator method [34,5]. In principle one could argue that the node pressure should be the same for all cells around the same vertex $\mathbf{x}_r$: that is $p_{jr} = p_r$ for all j. This is not possible in the general case for reasons that have been explained in [13]. At this point we obtain a possible semi-discrete numerical approximation of the continuous system (1)

$$\begin{cases} M_j \tau_j'(t) = \sum_r (\mathbf{C}_{jr}, \mathbf{u}_r), \\ M_j \mathbf{u}_j'(t) = -\sum_r \mathbf{C}_{jr} p_{jr}, \\ M_j e_j'(t) = -\sum_r (\mathbf{C}_{jr}, \mathbf{u}_r) p_{jr}. \end{cases} \tag{9}$$

The first equation is the discretization of the mass conservation relation, and reflects the compatibility of the method with the discretization of the volume conservation equation which is the fourth equation of (1). The last two equations are the semi-discrete version of the momentum and total energy equations. This formulation is similar but different to the one pursued in [22,24]. It remains to close the problem with a convenient definition of the velocity of the vertices $\mathbf{u}_r$ and the pressures p_{jr} which are both node based quantities.

3.3. The nodal solver

In order to determine the velocity $\mathbf{u}_r$ of the vertices and the node pressures p_{jr} (see Fig. 6), we generalize what has been proposed in [13].

The nodal solver is based on two different formulas. The first formula (10) is a multidimensional generalization of a first-order Riemann solver, but in the direction parallel to the vector $\mathbf{C}_{jr}$. So we assume a linearized-Riemann-invariant relation in the direction of $\mathbf{C}_{jr}$

$$p_{jr} - p_j + \alpha_{jr}(\mathbf{u}_r - \mathbf{u}_j, \mathbf{n}_{jr}) = 0. \tag{10}$$

By definition the normal vector $\mathbf{n}_{jr}$ is such that

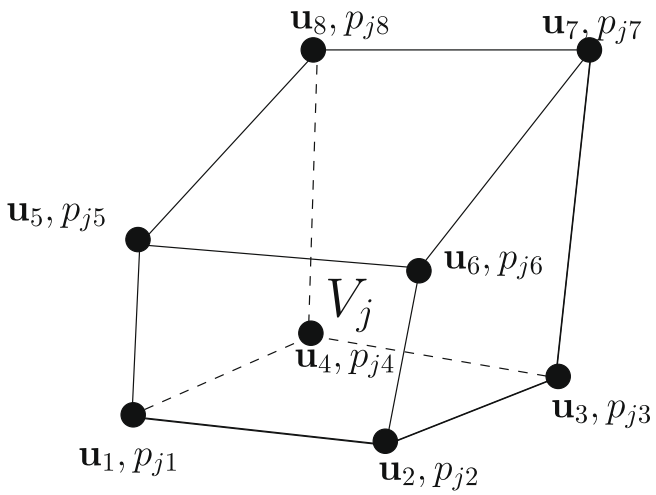


Fig. 6. Nodal velocities and nodal pressures.

$$\mathbf{n}_{jr} = \frac{\mathbf{C}_{jr}}{|\mathbf{C}_{jr}|}, \quad |\mathbf{n}_{jr}| = \sqrt{(\mathbf{n}_{jr}, \mathbf{n}_{jr})} = 1.$$

This relation is an approximation of the Rankine Hugoniot relations for shock hydrodynamics in dimension one, and is commonly used for the design of Godunov solvers. *A priori* the coefficient α_{jr} is defined as the acoustic impedance

$$\alpha_{jr} = \rho_j c_j \tag{11}$$

where c_j is the sound speed in cell j: $c^2 = \frac{\partial p}{\partial \rho|S}$ and S is the entropy. In this formula α_{jr} does not depend upon the node index r. However other determinations of α_{jr} are possible. In Section 6.2 we will describe one of them. The second formula, needed to construct the nodal solver for the integration of (9), expresses that the sum of all forces around the vertex $\mathbf{x}_r$ is zero

$$\sum_j \mathbf{C}_{jr} p_{jr} = 0. \tag{12}$$

This formula is natural in the context of Lagrangian methods because it enforces the conservation of momentum, see Proposition 3.

The solution of the set of equations (10)–(12) is easy to compute. Using (10) one can eliminate the pressures in (12). One gets a linear system

$$\mathbf{A}_r \mathbf{u}_r = \mathbf{b}_r. \tag{13}$$

The unknown vector is the node velocity $\mathbf{u}_r \in \mathbb{R}^d$. The matrix is

$$\mathbf{A}_r = \sum_j \rho_j c_j \frac{\mathbf{C}_{jr} \otimes \mathbf{C}_{jr}}{|\mathbf{C}_{jr}|} \in \mathbb{R}^{d\times d}. \tag{14}$$

By construction $\mathbf{A}_r = \mathbf{A}_r^t$ is non negative. If the mesh is non degenerate then $\mathbf{A}_r > 0$. Let us be more precise. Assume for example that $\mathbf{A}_r \mathbf{u}_r = 0$, which implies that

$$(\mathbf{u}_r, \mathbf{A}_r \mathbf{u}_r) = \sum_j \rho_j c_j \frac{(\mathbf{C}_{jr}, \mathbf{u}_r)^2}{|\mathbf{C}_{jr}|} = 0 \Rightarrow (\mathbf{C}_{jr}, \mathbf{u}_r) = 0 \quad \forall j. \tag{15}$$

Due to the equality (6) the vectors $\mathbf{C}_{jr}$ are linearly dependent. If the number of these linearly dependent vectors is greater or equal to $d+1$ then it is enough, in the general case, to generate a basis of $\mathbb{R}^d$, which implies that $\mathbf{u}_r = 0$. In summary the general case is that if $d+1$ cells connect to the vertex $\mathbf{x}_r$, then the matrix $\mathbf{A}_r$ is non-singular. The right hand side is

$$\mathbf{b}_r = \sum_j \mathbf{C}_{jr} p_j + \sum_j \rho_j c_j \frac{\mathbf{C}_{jr} \otimes \mathbf{C}_{jr}}{|\mathbf{C}_{jr}|} \mathbf{u}_j \in \mathbb{R}^d. \tag{16}$$

The solution of the linear system is

$$\mathbf{u}_r = \mathbf{A}_r^{-1} \mathbf{b}_r. \tag{17}$$

Once the nodal velocities $\mathbf{u}_r$ have been calculated, one computes the nodal pressures p_{jr} using (10)

$$p_{jr} = p_j + \rho_j c_j (\mathbf{u}_r - \mathbf{u}_j, \mathbf{n}_{jr}). \tag{18}$$

3.4. First-order GLACE scheme

The first-order GLACE scheme is the following:

(1) At the beginning of the time step one computes the geometrical vectors $\mathbf{C}_{jr}^k$ for all j, r, as a function of the vertices $\mathbf{x}_r^k$ (Sections 2.1, 2.2, 2.4 and Appendix A).
(2) Then one determines the nodal velocities $\mathbf{u}_r^k$ and the nodal pressures p_{jr}^k using the nodal solver (17) and (18).
(3) It is enough to update the total momentum and the total energy. For the momentum one uses

$$M_j \frac{\mathbf{u}_j^{k+1} - \mathbf{u}_j^k}{\Delta t} = -\sum_r \mathbf{C}_{jr}^k p_{jr}^k. \tag{19}$$

The total energy is updated with

$$M_j \frac{e_j^{k+1} - e_j^k}{\Delta t} = -\sum_r \left(\mathbf{C}_{jr}^k, \mathbf{u}_r^k\right) p_{jr}^k. \tag{20}$$

(4) Then the vertices are moved

$$\mathbf{x}_r^{k+1} = \mathbf{x}_r^k + \Delta t\, \mathbf{u}_r^k \tag{21}$$

and one computes the new volume V_j^{k+1}.

(5) Finally the new density in the cell is computed

$$\rho_j^{k+1} = \frac{M_j}{V_j^{k+1}}. \tag{22}$$

4. Boundary conditions

Since GLACE is based on a nodal solver, the implementation of boundary conditions requires a special treatment. The major difficulty comes from the corners of the computational domain: at such a corner two different boundary conditions can interact. Another practical problem is that the matrix $\mathbf{A}_r$ defined in (14) may become singular at the boundary because $\mathbf{A}_r$ does not contain any information about the boundary condition. It is illustrated in Fig. 7. For this very simple mesh $\mathbf{A}_r$ is by construction singular at all vertices: $\mathbf{A}_r$ is the sum of two rank one matrices at $\mathbf{x}_4$ and is a rank one matrix at $\mathbf{x}_1$. But of course if a proper boundary condition is considered, it can be enough to obtained a well posed problem at the boundary. This is indeed the case. We propose to rely on the following approach which is adapted to many practical situations. The method aims at a correct discretization of the boundary condition by means of the definition of a new non-singular matrix $\widetilde{\mathbf{A}_r}$ and a new right hand side $\widetilde{\mathbf{b}_r}$, so that the node velocity is the solution of the non-singular discrete linear system

$$\widetilde{\mathbf{A}_r}\mathbf{u}_r = \widetilde{\mathbf{b}_r}. \tag{23}$$

The main motivation for using this method is that it is convenient for implementation considerations since the boundary nodes can be treated like interior nodes, once $\widetilde{\mathbf{A}_r}$ and $\widetilde{\mathbf{b}_r}$ have been defined.

4.1. Sliding on a plane

Consider Fig. 8 where node $\mathbf{x}_r$ is constrained to move only on the boundary plane xOy. The node velocity $\mathbf{u}_r$ is orthogonal to the exterior normal $\mathbf{n}$

$$(\mathbf{u}_r, \mathbf{n}) = 0. \tag{24}$$

We consider that the projection, tangentially to the plane, of the sum of forces must vanish

$$\sum_j (\mathbf{C}_{jr} p_{jr}, \mathbf{t}) = 0, \text{ for all } \mathbf{t} \text{ such that } (\mathbf{t}, \mathbf{n}) = 0. \tag{25}$$

This is a projected version of the action–reaction law (12). Let us assume that relation (18) holds as before

$$p_{jr} = p_j + \rho_j c_j (\mathbf{u}_r - \mathbf{u}_j, \mathbf{n}_{jr}). \tag{26}$$

So we get a new linear system (24)–(26). It can be proved that this linear system has a unique solution in the general case (that is when d cells impinge on $\mathbf{x}_r$). Our method to prove this property is based on a reformulation of the boundary condition with a new matrix $\widetilde{\mathbf{A}_r}$ (and a new right hand side $\widetilde{\mathbf{b}_r}$) such that $\widetilde{\mathbf{A}_r}$ is non-singular by construction.

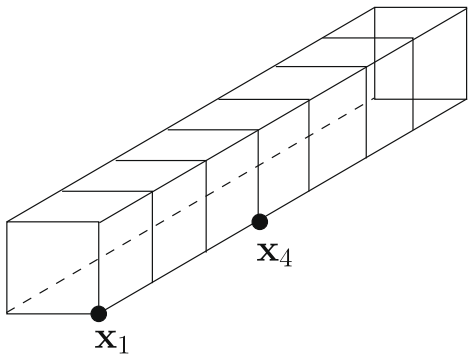


Fig. 7. Degenerated matrix $\mathbf{A}_r$.

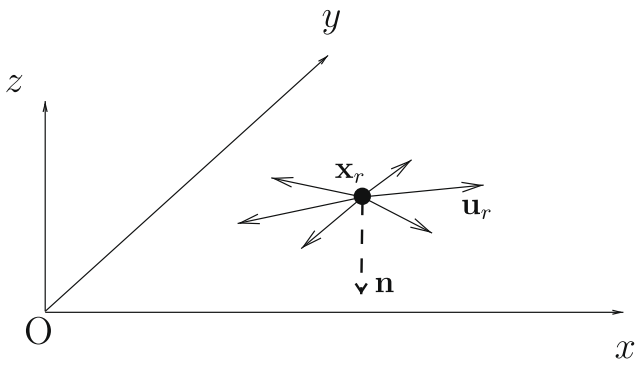


Fig. 8. Sliding on a plane.

Let us assume that $\mathbf{A}_r$ and $\mathbf{b}_r$ have already been calculated by (17) and (18). Then we define a new matrix

$$\widetilde{\mathbf{A}_r} = (I - \mathbf{n}\otimes\mathbf{n})\mathbf{A}_r(I - \mathbf{n}\otimes\mathbf{n}) + \mathrm{tr}(\mathbf{A}_r)\mathbf{n}\otimes\mathbf{n} = \widetilde{\mathbf{A}_r}^t \tag{27}$$

and a new right hand side

$$\widetilde{\mathbf{b}_r} = (I - \mathbf{n}\otimes\mathbf{n})\mathbf{b}_r. \tag{28}$$

Remark 1. In Eq. (27), $\mathrm{tr}(\mathbf{A}_r)$ is used to ensure that the condition number of $\widetilde{\mathbf{A}_r}$ is not too large. At least the condition number is independent of the scaling $\mathbf{A}_r \leftarrow \lambda\mathbf{A}_r$.

In the general case the matrix $\widetilde{\mathbf{A}_r}$ is non-singular. This is a consequence of the following property.

Proposition 2. *The vector $\mathbf{u}_r$ solution of $\widetilde{\mathbf{A}_r}\mathbf{u}_r = \widetilde{\mathbf{b}_r}$ is also the solution of discrete equations at the boundary, that is (24) which is the sliding condition, (25) which is the action–reaction law tangentially to the plane, and (26) which is the acoustic solver in the corner direction.*

The new linear system writes

$$[(I - \mathbf{n}\otimes\mathbf{n})\mathbf{A}_r(I - \mathbf{n}\otimes\mathbf{n}) + \mathrm{tr}(\mathbf{A}_r)\mathbf{n}\otimes\mathbf{n}]\mathbf{u}_r = (I - \mathbf{n}\otimes\mathbf{n})\mathbf{b}_r. \tag{29}$$

Take the scalar product of (29) with the normal vector $\mathbf{n}$: then $\mathrm{tr}(\mathbf{A}_r)(\mathbf{n},\mathbf{u}_r) = 0$. Since $\mathbf{A}_r$ is symmetric non negative, then $\mathbf{A}_r = 0$ if and only if $\mathrm{tr}(\mathbf{A}_r) = 0$. But there exists always at least one cell impinging on vertex $\mathbf{x}_r$. This is clear on Fig. 7 which is perhaps the most singular configuration. So $\mathbf{A}_r \neq 0$ and $\mathrm{tr}(\mathbf{A}_r) \neq 0$. It implies the sliding condition (24). Since $(I - \mathbf{n}\otimes\mathbf{n})\mathbf{t} = \mathbf{t}$ and $(\mathbf{n}\otimes\mathbf{n})\mathbf{t} = 0$, the Eq. (29) simplifies into $(I - \mathbf{n}\otimes\mathbf{n})\mathbf{A}_r\mathbf{u}_r = (I - \mathbf{n}\otimes\mathbf{n})\mathbf{b}_r$. Take the scalar product of (29) against any tangent vector $\mathbf{t}$ orthogonal to the normal $\mathbf{n}$. One gets $(\mathbf{t},\mathbf{A}_r\mathbf{u}_r) = (\mathbf{t},\mathbf{b}_r)$. By construction of $\mathbf{A}_r$ and since the pressures are computed using (26), one has $\mathbf{A}_r\mathbf{u}_r = \sum_j \mathbf{C}_{jr}p_{jr}$. It implies (25). This ends the proof.

4.2. Sliding on a line

This case is treated with the same method. We modify the matrix and the right hand side.
The node slides parallel to the tangent vector $\mathbf{t}$ which is orthogonal to both normals

$$(\mathbf{t},\mathbf{n}_1) = (\mathbf{t},\mathbf{n}_2) = 0.$$

The planes are not necessarily orthogonal: that is $(\mathbf{n}_1,\mathbf{n}_2) \neq 0$ is possible. Let us define the new non-singular matrix

$$\widetilde{\mathbf{A}_r} = (\mathbf{t}\otimes\mathbf{t})\mathbf{A}_r(\mathbf{t}\otimes\mathbf{t}) + \mathrm{tr}(\mathbf{A}_r)(I - \mathbf{t}\otimes\mathbf{t})$$

and the new right hand side

$$\widetilde{\mathbf{b}_r} = (\mathbf{t}\otimes\mathbf{t})\mathbf{b}_r = (\mathbf{t},\mathbf{b}_r)\mathbf{t}.$$

The nodal velocity on the line is defined as the unique solution of $\widetilde{\mathbf{A}_r}\mathbf{u}_r = \widetilde{\mathbf{b}_r}$ (see Fig. 9).

4.3. Given pressure

The situation is different from the previous ones. Let us assume that an external pressure p_{ext} is imposed on the vertex $\mathbf{x}_r$, as described in Fig. 10.

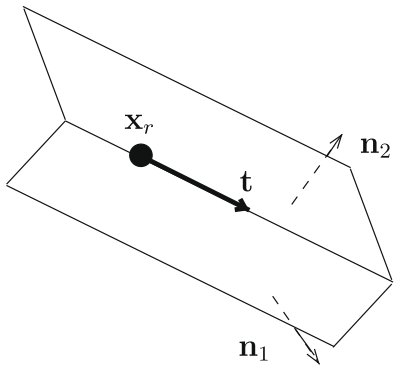


Fig. 9. Sliding on a line.

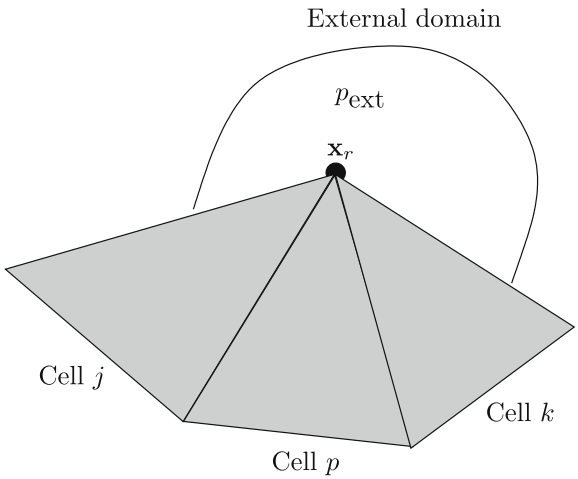


Fig. 10. Given pressure p_{ext} at the boundary.

We consider that the sum of the forces around $\mathbf{x}_r$ is zero

$$\sum_j \mathbf{C}_{jr} p_{jr} + \left(-\sum_j \mathbf{C}_{jr} \right) p_{\text{ext}} = 0.$$

The vector $-\sum_j \mathbf{C}_{jr}$ is the external surface at $\mathbf{x}_r$ on which the external given pressure must be applied. One has $\sum_j \mathbf{C}_{jr} \neq 0$ because the total volume changes (the mesh moves at $\mathbf{x}_r$). In this case it is sufficient to keep the same matrix $\widetilde{\mathbf{A}_r} = \mathbf{A}_r$ but to modify the right hand side

$$\widetilde{\mathbf{b}_r} = \mathbf{b}_r - \sum_j \mathbf{C}_{jr} p_{\text{ext}}. \tag{30}$$

4.4. Mixed condition

The problem arises at the corners, as described in Fig. 11 in dimension $d = 2$. The problem is the same in dimension $d = 3$. At $\mathbf{x}_1$ two different boundaries interact. A given external pressure p_{ext} is imposed on the boundary. The other boundary condition on the wall is a sliding condition.

A solution is easily computed as follows. In a first step we modify the right hand side with (30) keeping the same matrix. Then we modify the matrix and the right hand side using the sliding method (27) and (28). So the right hand side is modified twice. Then we solve (23).

5. Main theoretical properties

As for any Lagrangian method, conservation and compatibility with the entropy condition are essential properties.

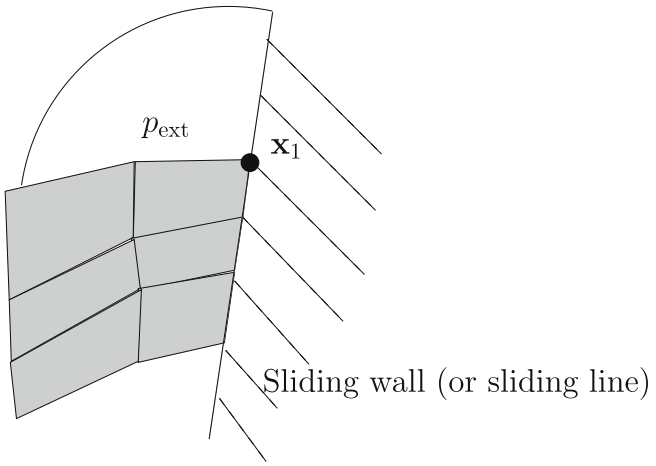


Fig. 11. Mixed conditions applied on vertex $\mathbf{x}_1$.

5.1. Conservativity

Proposition 3. *By construction, the GLACE scheme is conservative in total mass, total momentum and total energy.*

This property is true up to boundary conditions of course. Therefore we forget about boundary conditions and concentrate on the key features of the scheme. By construction GLACE is of course conservative for the mass of each cell, so the total mass is also preserved.

Concerning the total momentum one has

$$\sum_j M_j \frac{\mathbf{u}_j^{k+1} - \mathbf{u}_j^k}{\Delta t} = \sum_j \left(-\sum_r \mathbf{C}_{jr}^k p_{jr}^k \right) = -\sum_r \left(\sum_j \mathbf{C}_{jr}^k p_{jr}^k \right).$$

One equation of the nodal solver is (12): $\sum_j \mathbf{C}_{jr}^k p_{jr}^k = \mathbf{0}$. Hence the total momentum is preserved $\sum_j M_j \mathbf{u}_j^{k+1} = \sum_j M_j \mathbf{u}_j^k$. Concerning the total energy one has

$$\sum_j M_j \frac{e_j^{k+1} - e_j^k}{\Delta t} = \sum_j \left(-\sum_r \left(\mathbf{C}_{jr}^k, \mathbf{u}_r^k \right) p_{jr}^k \right) = -\sum_r \left(\sum_j \mathbf{C}_{jr}^k p_{jr}^k, \mathbf{u}_r^k \right).$$

Under the same condition, one gets the conservation relation $\sum_j M_j e_j^{k+1} = \sum_j M_j e_j^k$.

5.2. Compatibility with the entropy condition

The compatibility of the scheme with the entropy condition is an important property of any Lagrangian method. It guarantees a correct physical behavior of the cell, and also gives some indications that the method is stable. In what follows we show that the semi-discrete GLACE scheme is such that the entropy increases in the cell, that is $\frac{d}{dt} S_j \geqslant 0$. For a discrete scheme in space and time, the estimate is much more difficult to obtain [13]: it involves the CFL condition.

Let us assume that the pressure law is compatible with the fundamental principle of thermodynamics

$$TdS = d\varepsilon + pd\tau.$$

The entropy is $S, \varepsilon = e - \frac{1}{2}|\mathbf{u}|^2$ is the internal energy and the temperature is $T > 0$.

Proposition 4. *The semi-discrete GLACE scheme (9) is compatible with the entropy condition. This property remains valid with any boundary conditions such that these boundary conditions are compatible with* Eq. (10).

One has $T_j \frac{d}{dt} S_j = \varepsilon_j' + p_j \tau_j' = p_j \tau_j' - \left(\mathbf{u}_j, \mathbf{u}_j' \right) + e_j'$. Therefore $M_j T_j \frac{d}{dt} S_j = p_j \sum_r (\mathbf{C}_{jr}, \mathbf{u}_r) + (\mathbf{u}_j, \sum_r \mathbf{C}_{jr} p_{jr}) - \sum_r (\mathbf{C}_{jr}, \mathbf{u}_r) p_{jr}$. On the other hand one has $\sum_r (\mathbf{C}_{jr}, \mathbf{u}_j) p_j = 0$ since $\sum_r \mathbf{C}_{jr} = \mathbf{0}$. After elementary manipulations one gets

$$M_j T_j \frac{d}{dt} S_j = \sum_r (\mathbf{C}_{jr}, \mathbf{u}_r - \mathbf{u}_j)(p_j - p_{jr}).$$

Eq. (10) implies that $(\mathbf{C}_{jr}, \mathbf{u}_r - \mathbf{u}_j)$ and $p_j - p_{jr}$ have the same sign, so that $M_j T_j \frac{d}{dt} S_j \geqslant 0$, and the proof is completed.

This property explains why the scheme has no need of any artificial viscosity technique to be able to handle shocks. By construction the scheme generates entropy and is conservative. The theory of conservation laws [20] shows that it is sufficient to be compatible with the Rankine Hugoniot relations for discontinuous solutions.

Nevertheless the compatibility of the entropy, as stated in the proposition, is not enough. It is important to verify that the increase of entropy is not too large, in particular for smooth compression flows or for rarefaction waves. We have two answers: (a) we have checked on a large number of test problems that the entropy production is moderate. In particular we have compared with the Caveat scheme [3] and also with the first-order scheme [22]. See the discussion for smooth compression flow in Section 8.6; (b) as usual in computational fluid dynamics, a second-order MUSCL extension of the scheme allows for a better accuracy for smooth flows, in particular for rarefaction waves. We have developed such a MUSCL procedure that we will explain below.

6. Time step considerations and geometrical correction

For Lagrangian computations, an efficient control of the time step is important to obtain the hydrodynamic stability of the method (that is stability of thermodynamic quantities like density, pressure,...) and the stability of the mesh (because a moving mesh is very sensitive to spurious mesh displacements). We also show how to introduce a simple geometrical correction so that GLACE is equal to the standard 1D Godunov scheme on Cartesian meshes (grids).

6.1. CFL restriction

Since the method is explicit, a CFL condition is needed [3]. In practice this condition takes the form $c_j \frac{\Delta t}{\Delta x_j} \leqslant \text{CFL}$ for all j, where Δx_j is a local length for cell j, c_j is the local sound speed and $\text{CFL} \leqslant 1$. As usual the definition of Δx_j is not obvious for an unstructured mesh. In practice a local evaluation based on the radius of the inscribed circle is possible, following the so-called "worst guess" principle. Since our second-order extension, described in Section 7, is based on a Lax–Wendroff procedure (the second-order Lax–Wendroff scheme and the first-order Donor cell scheme for linear transport are stable under the same CFL condition) we impose the same CFL restriction for both the first-order GLACE scheme and the second-order extension. It is also possible to add time step restriction based on volume variation as described in [24].

6.2. The geometrical correction λ_j

The idea which is pursued in this section is that it may be worthwhile that the first-order GLACE scheme degenerates exactly to the first-order Godunov acoustic solver on Cartesian geometries (grid). Let us point out that there is no theoretical reason to impose that the scheme degenerates in simple geometries to the exact acoustic solver: in our mind it is rather a comfortable framework since it is more easy to calibrate the scheme on simple 1D test problems with this modification. The proposed procedure does not change the consistency of the scheme, in the sense that the solution at convergence is the same. On the other hand this procedure modifies the viscosity of the scheme. So it may have impact on the stability and on the effective diffusivity of the scheme. Another interest of this study is that it shows the influence of the aspect ratio of the mesh on the resulting scheme.

The main idea is to introduce λ_j a free parameter (a priori close to 1) and to modify α_{jr} which becomes $\alpha_{jr} = \rho_j c_j \lambda_j$. What we propose is to replace the discrete relation (18) by

$$p_{jr} = p_j + \rho_j c_j \lambda_j (\mathbf{u}_r - \mathbf{u}_j \mathbf{n}_{jr}), \text{ where } \lambda_j = \sqrt{\frac{\sum_r |\mathbf{C}_{jr}|}{\mu_j}} \tag{31}$$

and μ_j is the largest eigenvalue of the matrix $\mathbf{B}_j = \sum_r \frac{\mathbf{C}_{jr} \otimes \mathbf{C}_{jr}}{|\mathbf{C}_{jr}|}$. The matrix $\mathbf{B}_j \in \mathbb{R}^{3\times 3}$ is symmetric positive definite in the general case. Note that λ_j is non dimensional. We have observed in all our simulations that $\lambda_j \in \left[1, \sqrt{3}\right]$. A proof that $\lambda_j \in \left[1, \sqrt{2}\right]$ exists in dimension $d = 2$. In practice we first compute μ_j with Cardan's formulas, with a careful elimination of the singularities. Then we compute λ_j.

Proposition 5. *Assume that the mesh is structured, that is $j = (j_1, j_2, j_3)$ with $j_1, j_2, j_3 \in \mathbb{Z}$. Consider a 1D flow in the x direction, and assume that the mesh size is everywhere smaller in the x direction than in the y and z directions, that is $\Delta x_j < \Delta y_j$ and $\Delta x_j < \Delta z_j$. Consider the scheme with λ_j given by (31). Then the scheme is equivalent to the 1D Godunov acoustic scheme*

$$\begin{cases} p_{j_1+\frac{1}{2}} = \frac{\rho_{j_1+1} c_{j_1+1} p_j + \rho_{j_1} c_{j_1} p_{j_1+1}}{\rho_{j_1} c_{j_1} + \rho_{j_1+1} c_{j_1+1}} + \frac{\rho_{j_1} c_{j_1} \rho_{j_1+1} c_{j_1+1}}{\rho_{j_1} c_{j_1} + \rho_{j_1+1} c_{j_1+1}} (u_{j_1} - u_{j_1+1}), \\ u_{j_1+\frac{1}{2}} = \frac{\rho_{j_1} c_{j_1} u_j + \rho_{j_1+1} c_{j_1+1} u_{j+1}}{\rho_{j_1} c_{j_1} + \rho_{j_1+1} c_{j_1+1}} + \frac{1}{\rho_{j_1} c_{j_1} + \rho_{j_1+1} c_{j_1+1}} (p_{j_1} - p_{j_1+1}). \end{cases} \tag{32}$$

The proof is a bit technical and is given in the appendix. If λ_j is not set to the optimal value (31), then the viscosity coefficient in the formulas (32) is premultiplied by a bounded factor λ_j and $\frac{1}{\lambda_j}$.

Assume now that the grid is such that

$$10 \min(\Delta y, \Delta z) \geqslant \Delta x > \max(\Delta y, \Delta z)$$

(this is just an indication, it depends on the test problem), then the previous proposition does not apply. However we have observed that the scheme behaves well in this case and we see a similar behavior with respect to the standard Godunov

scheme in the x direction. But if $\Delta x \gg \max(\Delta y, \Delta z)$, then the results may show a important dependence with respect to the aspect ratio, both in term of stability and accuracy. However, it must be noticed that all the theoretical properties are true (entropy, conservativity,...) with or without this small geometrical correction.

In practice we use the correction (31) systematically on all meshes. If the second-order extension is used, it is reasonable to think that the influence of the aspect ratio of the mesh is less important.

7. Second-order extension

In this section we describe the MUSCL type second-order extension of the first-order GLACE scheme. Essentially this is based on the seminal ideas of Van Leer applied in the Lagrangian context [3]. The method that we have chosen has the advantage to be a one step scheme, which means we do not use a Runge–Kutta two-step integration scheme. The results in 1D for the Sod shock tube problem show that it provides the usual gain of accuracy with respect to a first-order scheme.

7.1. Slope reconstruction and limitation

The idea is very natural. In all cells, one reconstructs an affine function $\mathbf{x} \mapsto \bar{p}_j(\mathbf{x})$ for the pressure

$$\bar{p}_j(\mathbf{x}) = p_j + (\nabla p_j, \mathbf{x} - \mathbf{x}_j),$$

and also for the velocity $\bar{\mathbf{u}}_j(\mathbf{x}) = \mathbf{u}_j + \nabla \mathbf{u}_j(\mathbf{x} - \mathbf{x}_j)$. The quantity ∇p_j is a pressure gradient. We reconstruct this gradient from the neighboring values. Many ways are possible to do so. In practice we prefer to construct first a node gradient

$$\nabla p_r = \frac{\sum_j \mathbf{C}_{jr} p_j}{V_{jr}},$$

where V_{jr} is a control volume around node $\mathbf{x}_r$: the only constraint is $\sum_r V_{jr} = V_j$. One can use the standard definition of the control volume in staggered schemes [3,6,9] where the centers of the faces determines the limit of control volumes. Then we average the node gradient to obtain an approximate cell gradient

$$\nabla p_j = \frac{\sum_r \nabla p_r}{\sum_r 1}.$$

This is quite a crude way to compute a local gradient. However it has been proved to be accurate enough and very robust. We perform the same reconstruction for the velocity. It gives $\nabla \mathbf{u}_r = \frac{\sum_j \mathbf{C}_{jr} \otimes \mathbf{u}_j}{V_{jr}}$ and $\nabla \mathbf{u}_j = \frac{\sum_r \nabla \mathbf{u}_r}{\sum_r 1}$.

Once this is done, we limit these quantities with the Dukowicz algorithm [15]. That is ∇p_j is damped and becomes: $\nabla p_j \leftarrow \sigma_j \nabla p_j, \quad 0 \leqslant \sigma_j \leqslant 1$. Many other limitation procedures are possible [37], as usual in hydrodynamics codes.

However the spatial limitation might not be enough to obtain stability in all cases. By analogy with the Lax–Wendroff scheme for linear advection, we introduce another factor $1 - v_j$ (v_j is the local CFL number). For example the final reconstructed pressure is now

$$\bar{p}_j(\mathbf{x}) = p_j + (1 - v_j)\sigma_j(\nabla p_j, \mathbf{x} - \mathbf{x}_j), \quad 0 \leqslant v_j \leqslant 1.$$

The same factor is applied to the reconstructed velocity, with the exception that for flows and meshes which rotational symmetries it can be better to limit the gradient of the velocity only in the radial direction. This tensorial limitation procedure is known to be much better in 2D on equi-sectorial meshes. It must be noticed that the situation is less clear in 3D because such perfect equi-sectorial meshes cannot exist.

7.2. Second-order nodal solver

Assume now that the reconstructed cell pressure and the reconstructed cell velocity are available. Then we use them directly in the nodal solver, that is we consider the solution $(\mathbf{u}_r, p_{jr})$ of: $p_{jr} - \bar{p}_j(\mathbf{x}_r) + \alpha_{jr}(\mathbf{u}_r - \bar{\mathbf{u}}_j(\mathbf{x}_r), \mathbf{n}_{jr}) = 0$, and $\sum_j \mathbf{C}_{jr} p_{jr} = 0$. Up to these small modifications, we proceed as in Section 3.3. That is p_{jr} and $\mathbf{u}_r$ are computed after inversion of a linear system.

An advantage of the overall procedure is its natural compatibility with the boundary conditions. The method is simple to use, and inexpensive with respect to CPU considerations. We found an important gain of accuracy using it, specially in rarefaction fans.

8. Numerical results

We discuss basic test problems for Lagrangian flows.

A first series of problems is computed on grids. The 1D test problem has been computed with the 3D scheme with a mesh similar to the one depicted in Fig. 7 and in the configuration described in Proposition 5. We show the results computed on a 3D grid for the Saltzmann problem and for the Sedov problem. These problems are challenging ones for Lagrangian methods

since the mesh is not aligned with the flow, see for example [8] where it is shown that spurious jets may exist along the axis. We have observed that if the calculation is done with a second-order extension, then the numerical results are quite good.

A second series of tests is characteristic of calculations on 3D spherical meshes for Inertial Confinement Fusion applications. The 2D and 3D Kidder test problem is to assess the numerical convergence of the method. The meshes of the $\frac{1}{8}$ of the sphere that we use are symmetric with respect to the reduced connectivity line. The reduced connectivity line is the set of vertices behind the reduced connectivity point which is visible at the internal face of the mesh for the thin shell problem: it coincides with the middle of the internal face. It must be noted that the design of such meshes in 3D makes necessary the incorporation of this reduced connectivity point. The $\mathbf{C}_{jr}$ are computed with the isoparametric representation.

8.1. Sod 1D

This test case is the very classical 1D Sod shock tube [35]. It is a simple Riemann problem. It involves a perfect gas with an adiabatic constant $\gamma = 1.4$. The initial conditions are $U_0(x < 0.5) = U_l$ and $U_0(x > 0.5) = U_r$ with $U_l = (\rho_l = 1, u_l = 0, p_l = 1)$ and $U_r = (\rho_r = 0.125, u_r = 0, p_r = 0.1)$. The results for the density are compared to the analytical solution on Fig. 12 for the Lagrangian second-order GLACE scheme. We observe a very good accuracy between the discrete solution and the exact solution. It is an immediate consequence of the property 5 which guarantees that the scheme is equal to a standard Godunov scheme in 1D, provided the mesh is fine in the longitudinal direction which is the case since $\Delta x \leqslant \min(\Delta y, \Delta z)$. So the result is similar to the results obtained with any 1D second-order Godunov method. In Fig. 13 we plot the result computed with the

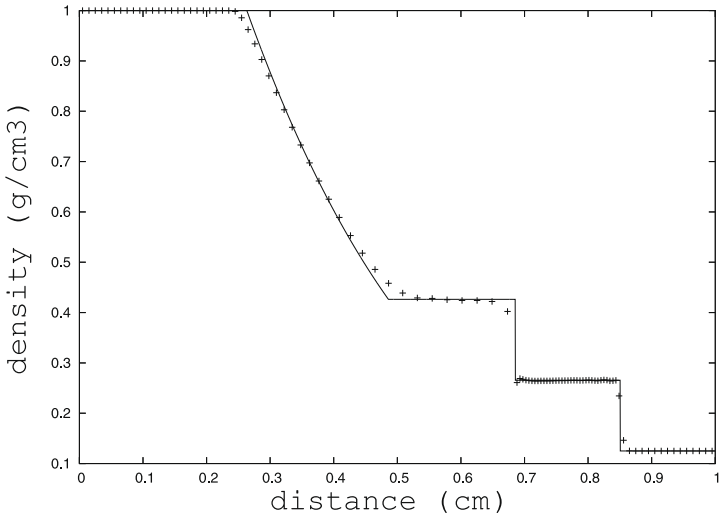


Fig. 12. The 1D Sod shock tube problem. Comparison with the reference solution (plain line) of the density profiles at $t = 0.2s$ for the order 2 scheme with a Dukowicz limiter (symbols).

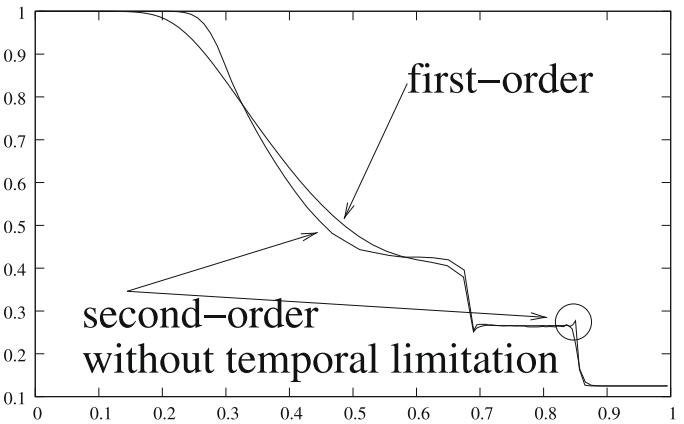


Fig. 13. The 1D Sod shock tube problem. First- and second-order but without temporal limitation (some oscillations are visible at the shock). The second-order scheme without the temporal limitation is already more accurate in the rarefaction fan.

first-order scheme (the dissipativity is visible in the rarefaction fan) and the results computed with the second-order scheme but without the temporal limitation. An extra oscillation is visible at the shock. This is one of the reason why we use the temporal limitation in all calculations.

8.2. Saltzmann 3D on a skewed grid

The Saltzmann test problem is a severe shock tube problem computed on a distorted mesh. It is known to be quite demanding in term of stability. The data are described in [7]. At $t = 0$, the density is $\rho = 1$, the pressure is ≈ 0 (10^{-6} in practice), the velocity is 0. The piston's velocity is 1. The mesh is skewed, that is

$$\begin{cases} x = 0.1(l-1) + 0.01(11-k)(6-m)/5\sin(0.01(l-1)), & 1 \leqslant m \leqslant 6, \\ x = 0.1(l-1) + 0.01(k-1)(m-6)/5\sin(0.01(l-1)), & 7 \leqslant m \leqslant 11, \\ y = 0.01(m-1), \\ z = 0.01(k-1) \end{cases}$$

where $1 \leqslant k, m \leqslant 11$ and $1 \leqslant l \leqslant 101$. The analytical value of the density after the first shock is $\frac{\gamma+1}{\gamma-1} = 4$ since $\gamma = \frac{5}{3}$. For this problem we noticed that the deformations of the mesh are more important with the first-order scheme than with the second-order scheme. The result is provided in Fig. 14 at time $t = 0.7$. We observe an average density of about 4 with some spurious values near the skewed sliding lines.

8.3. 3D blast wave

A challenging problem for Lagrangian methods is the 3D blast wave problem (Sedov problem) on a grid. A very strong shock propagates from the center on the domain. We refer to the recent works [33,23,8] for uptodate references. The problem can be quite difficult for Lagrangian methods because spurious jets might introduce some important non accuracy. This behavior is common to many Lagrangian schemes: it has forced some authors to use regularization methods or to add extra-viscous contributions in the solver. In the following numerical experiments, we have used that data given in [33]. The mesh is made of 20^3 cells on a grid $[0, 1.1]^3$. The parameter of the pressure law is $\gamma = \frac{7}{5}$. The distribution of the internal energy that was adopted is the regularized one from [33], that is

$$\begin{cases} p_{(0,0,0)} = \frac{27}{64} \times 10^{10}, & p_{(1,0,0)} = p_{(0,1,0)} = p_{(0,0,1)} = \frac{9}{64} \times 10^{10}, \\ p_{(1,1,1)} = \frac{1}{64} \times 10^{10}, & p_{(1,1,0)} = p_{(0,1,1)} = p_{(1,0,1)} = \frac{3}{64} \times 10^{10}. \end{cases}$$

By convention (0,0,0) is the numbering of the corner cell. Our results computed with the second-order scheme is provided on the left part of Fig. 15. No other regularization was necessary. The accuracy provided by the second-order extension is enough to capture the correct solution. Some discrepancy is nevertheless visible on the figure at 45 degrees on the facets of the domain.

We also compute the an equivalent 3D Sedov-like blast wave problem on the $\frac{1}{8}$th of the sphere (see also Fig. 15). The mesh is 100 layers $\times$ 75 sectors $=$ 7500 cells. The angular discretization of the mesh is 9 degrees (measured on the boundaries). The initial data are $\rho = p = 1$ everywhere except in the central layer ($r \leqslant 0.01$) where $\rho = 1$ and $p = 10^{10}$. We used the first-order scheme with sliding walls on the boundaries. With the second-order scheme the mesh is almost the same. We compare in Fig. 16 the result with a reference solution. The final time is $t = 0.00145$. We have plot the density for all the radius of the simulation on the Fig. 16. At the final time the symmetry of the mesh is very good since all curves are almost identical, despite of the reduce connectivity point (and also very close to the reference solution).

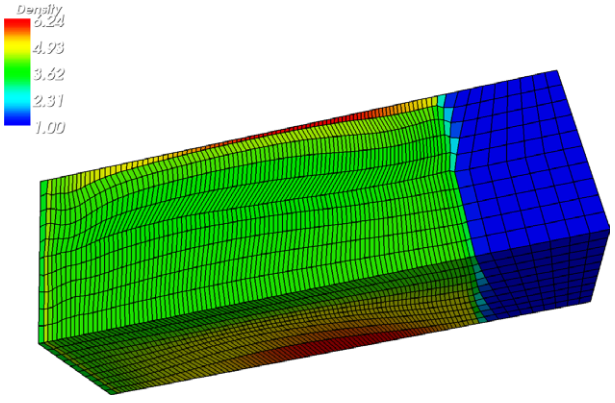


Fig. 14. The 3D Saltzmann problem computed with the second-order GLACE scheme. The time is $t = 0.7$. The analytical solution is $\rho = 4$ after the shock, and $\rho = 1$ before the shock.

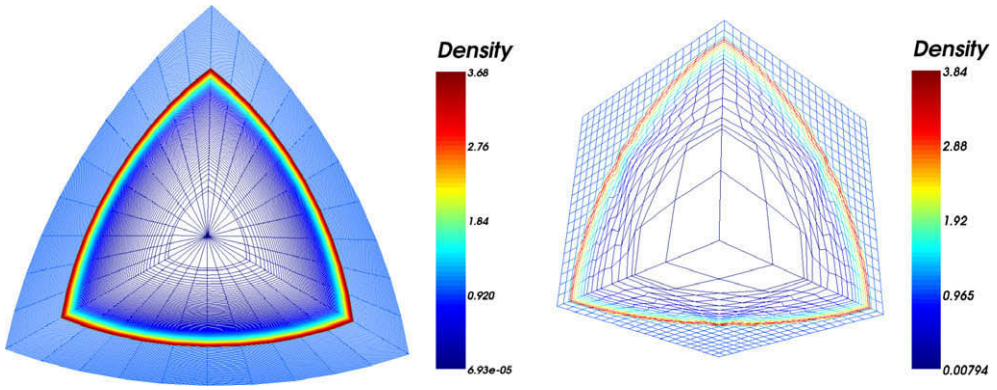
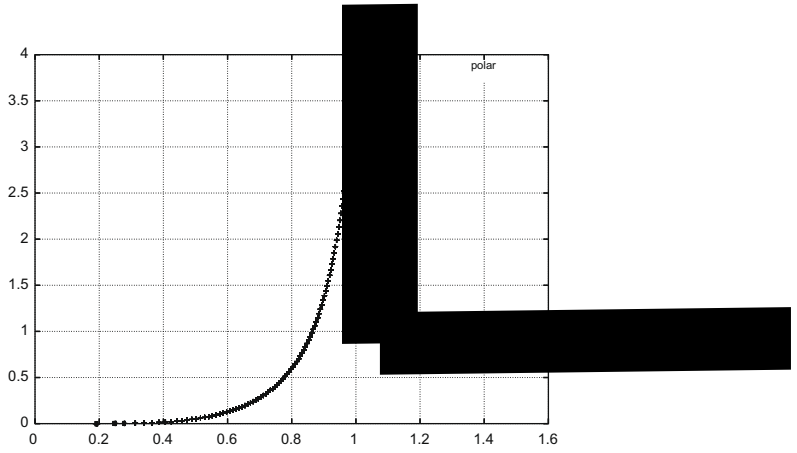


Fig. 15. A 3D blast wave at time $t = 1.510^{-4}$. **Left:** using a spherical mesh. **Right:** using a grid made with 20^3 cells.



8.4. The Kidder problem

We give the result of convergence tests for the Kidder problem in 2D and 3D (see [23] for instance). A portion of a shell $r_i = 0.9 \leqslant r \leqslant 1 = r_e$ is filled with a perfect gas. The adiabatic constant of the gas is $\gamma = 2$ in 2D and $\gamma = \frac{5}{3}$ in 3D. At $t = 0$, the initial density is $\rho_0(r) = \left(\frac{r_e^2 - r^2}{r_e^2 - r_i^2}\rho_i^{\gamma-1} + \frac{r^2 - r_i^2}{r_e^2 - r_i^2}\rho_e^{\gamma-1}\right)^{\frac{1}{\gamma-1}}$, with $\rho_i = 1$ and $\rho_e = 2$. The initial pressure is $p_0(r) = \rho_0(r)^\gamma$. The initial entropy is uniform $s_0 = \frac{p_0}{\rho_0^\gamma} = 1$. The velocity is $\mathbf{u}_0 = 0$ at $t = 0$. The boundary conditions are: sliding walls on the lateral faces; a given exterior pressure at the internal frontier $p_i(t) = p_0(r_i)h(t)^{-\frac{2\gamma}{\gamma-1}}$; and a given exterior pressure at the external frontier $p_e(t) = p_0(r_e)h(t)^{-\frac{2\gamma}{\gamma-1}}$. Let us denote c the sound speed. The solution is known [23] to be an isentropic compression $\left(s = \frac{p}{\rho^\gamma} = 1\right)$ such that the position at time $t > 0$ is $R(r,t) = h(t)r$, where the homothety rate is $h(t) = \sqrt{1 - \frac{t^2}{\tau_{\text{foc}}^2}}$ and the focalisation time is $\tau_{\text{foc}} = \sqrt{\frac{(\gamma-1)(r_e^2 - r_i^2)}{2(c_i^2 - c_e^2)}}$.

In Fig. 17 we display the mesh at $t = 0$ and at the final time of the simulation $t_f = \frac{\sqrt{3}}{2}\tau_{\text{foc}}$, so that the compression rate is $h(t_f) = 0.5$ in both cases. Therefore the analytical solution at t_f is a shell $0.45 \leqslant R \leqslant 0.5$. In Fig. 18 we compare the analytical position of the external and internal boundaries with the numerical ones calculated with the low resolution meshes (see below the definition of M_1 and N_1) and with the first-order scheme.

Finally we record in Table 1 the mean value of the external and internal radiuses. In 2D we used four meshes: M_1 is a $10 \times 10 = 100$ cells mesh, that is 10 sectors and 10 layers; M_2 is $20 \times 20 = 400$ cells; M_3 is $40 \times 40 = 1600$ cells; and finally M_4 is $80 \times 80 = 6400$ cells. In 3D we use three meshes: the first mesh N_1 has 10 sectors per facets and 5 layers, since the exterior and interior boundary are designed with 3 square meshes, then the total number of cells is $3 \times 5 \times 10 \times 10 = 1500$ cells; then we double the number of cells in each direction, that N_2 is a 12,000 cells mesh; and finally N_3 is a 96,000 cells mesh. The order is one (O_1) or two (O_2). We observe convergence of the numerical solution to the exact one in 2D and 3D. The accuracy of the numerical solution computed with the second order scheme O_2 is similar or better than the accuracy of the O_1 scheme with the mesh just coarser in the list. The experimental order of the convergence is

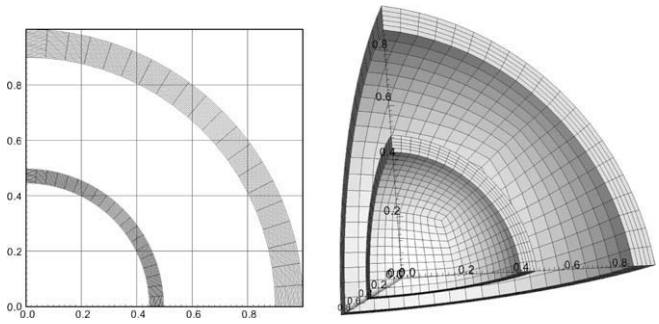


Fig. 17. Initial condition and final solution for the Kidder problem. 2D on the left, 3D on the right. The data are normalized so that the homothety factor is exactly $\frac{1}{2}$ in both cases.

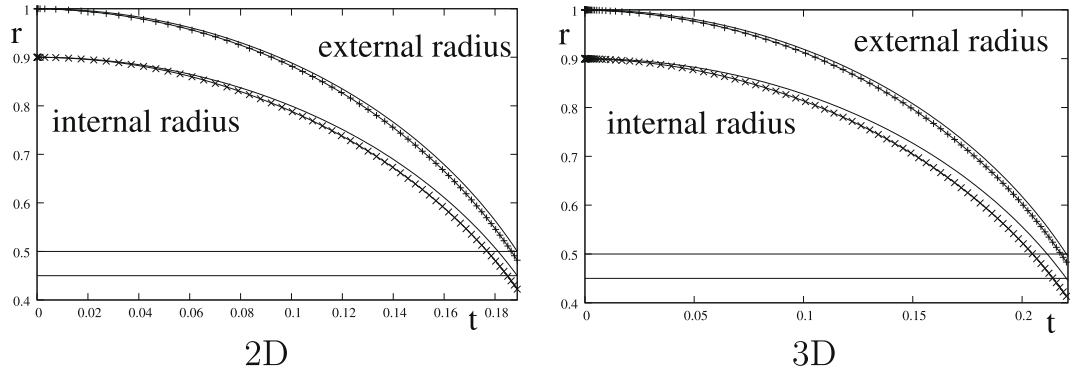


Fig. 18. The analytical solution is the continuous line, the discrete solution is plotted with symbols. On the left the 2D curves for the mesh M_1 and the first-order O_1 scheme. On the right the 3D curves for the mesh N_1 and the order O_1.

Table 1
Convergence of the numerical solution towards the exact solution, in function of the order of the scheme (O_1 and O_2) and of the mesh. The order of convergence is systematically computed with the two last meshes in the list (that is M_3M_4 or N_2N_3).

Dimension	Mesh	Order	$r_i(t_f)$	$r_e(t_f)$	Order	$r_i(t_f)$	$r_e(t_f)$
2	M_1	O_1	0.4223	0.4820	O_2	0.4343	0.4880
2	M_2	O_1	0.4392	0.4937	O_2	0.4458	0.4966
2	M_3	O_1	0.4453	0.4975	O_2	0.4487	0.4991
2	M_4	O_1	0.4478	0.4989	O_2	0.4495	0.4997
Convergence order			≈1.09	≈1.18		≈1.37	≈1.58
3	N_1	O_1	0.4133	0.4833	O_2	0.4269	0.4885
3	N_2	O_1	0.4339	0.4929	O_2	0.4422	0.4963
3	N_3	O_1	0.4424	0.4967	O_2	0.4472	0.4987
Convergence order			≈1.08	≈1.10		≈1.47	≈1.50

≈ 1 for O_1, and is ≈ 1.5 for the second-order extension O_2. In some cases the first order Glace scheme behaves like a second-order scheme. The reason is that the approximation of the geometry may be the dominant error: therefore smaller cells allow for a natural second-order approximation of the geometry. Our experience is that it is always an important contribution to the global error.

8.5. Noh 3D on a spherical mesh

The Noh test problem is run on the same type of spherical mesh as for the blast wave problem. The size of the internal cells is multiplied by a factor 5. It helps to stabilize the computation at the center of the domain and to obtain a more regular mesh at the final time. The number of cells is 95 layers $\times$ 75 sectors $=$ 7125. It has been observed in the past that spurious hourglass modes can arise for the Noh test problem in 2D [25]. Anyway for the low resolution 3D mesh depicted in Fig. 19 we did not observe any spurious modes.

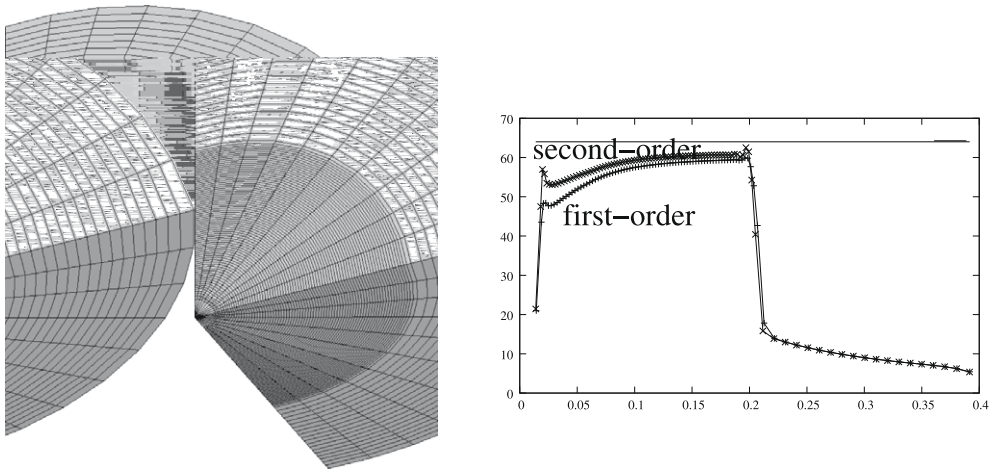


Fig. 19. The 3D Noh problem. Left: the mesh. Right: a cut of the density computed with the first-order scheme and the second-order scheme. The exact solution of the plateau is the plain line.

8.6. A thin shell problem in 3D

Next we compute a quasi-incompressible flow in 3D. The initial mesh is characteristic of a shell $R^- = 0.95 \leqslant r \leqslant 1 = R^+$. The initial values are: $\rho = 1, p = 2$, and the velocity is directed towards the center $\mathbf{u} = -\frac{1}{r^3}\mathbf{r}$ (in 2D $\mathbf{u} = -\frac{1}{r^2}\mathbf{r}$). The initial velocity is divergence free. The boundary conditions are the following: sliding walls on the planes, and a given pressure $p_{\text{ext}} = 2$ on the internal and external boundaries. This is why the flow is quasi-incompressible. Notice also that this problem can be run with a stiffened gas pressure law, using the change of dimensions and variables: $x^+ = R^* x, t^+ = T^* t, \rho^+ = \rho^* \rho, p^+ = \left(\frac{R^*}{T^*}\right)^2 \rho^* p - \gamma \pi^*$. If one takes $\gamma \pi^* = \left(\frac{R^*}{T^*}\right)^2 \rho^* p_{\text{ext}}$, then the problem is a high velocity implosion problem with a material modeled with a stiffened gas equation of state: in this case the exterior pressure is $p^+_{\text{ext}} = 0$. We refer to [13] for such an example. A typical result in 3D is given in Fig. 20. The mesh is made of 40 layers $\times$ 1200 sectors $=$ 4800 cells. The results are very accurate. On the figure we also notice a very stable behavior of the reduced connectivity point which is at the center of the internal boundary.

For such problems the standard first-order Godunov method is known to be very dissipative. In Fig. 21 we compare the result computed with the first-order GLACE scheme and with the first-order Chic 2D scheme (on this problem [24] and the CAVEAT scheme are equivalent). We see the extra viscosity for these schemes.

An explanation of this phenomenon is as follows. Consider the mesh depicted in 2D in Fig. 22. For the Godunov scheme the fluxes are computed on the edges of the mesh. The pressure flux at the edge is $p^* \approx \frac{p_j + p_k}{2} + (\rho c)^* (\mathbf{u}_j - \mathbf{u}_k, \mathbf{n}_2)$. The term $(\rho c)^* (\mathbf{u}_j - \mathbf{u}_k, \mathbf{n}_2)$ is a viscous correction to the mean value. For a radial flow one has $\mathbf{u}_j = -u\mathbf{e}_j \left(\mathbf{e}_j = \frac{\mathbf{Ox}_j}{|\mathbf{Ox}_j|}\right)$ and $\mathbf{u}_k = -u\mathbf{e}_k \left(\mathbf{e}_k = \frac{\mathbf{Ox}_k}{|\mathbf{Ox}_k|}\right)$, where $u > 0$ is the absolute value of the velocity. Therefore the viscous contribution is

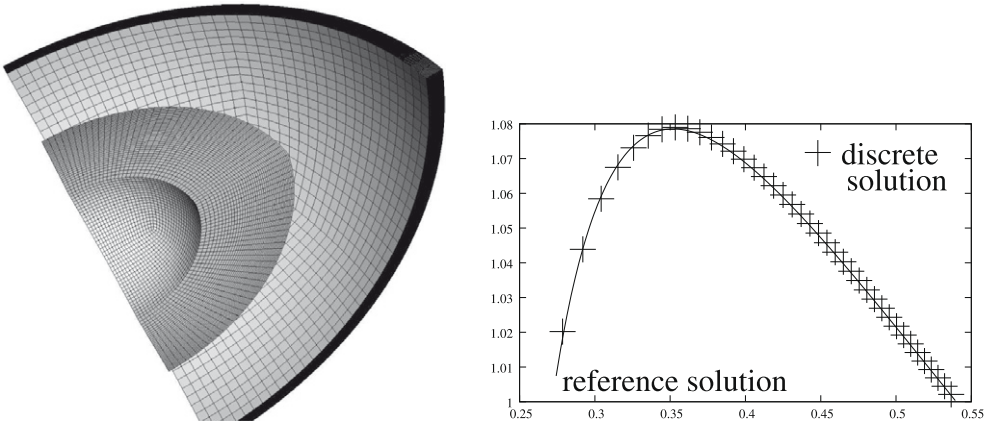


Fig. 20. The thin shell problem in 3D. The mesh is made with 4800 cells. Top: the mesh at $t = 0$ and $t = 0.52$. Bottom: a cut of the density compared with a reference solution computed with a 1D spherical code.

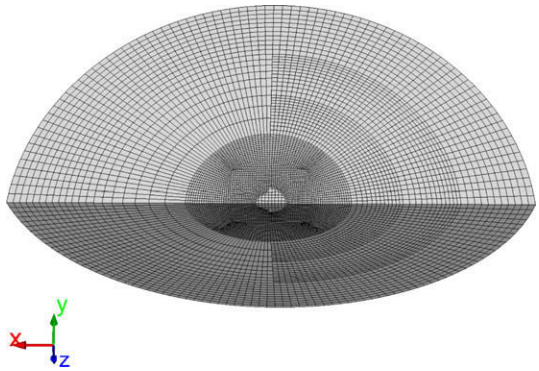


Fig. 23. The 3D Sod shock tube problem. Mesh at $t = 0.2$. On the left purely Lagrangian, on the right with layer refinement.

8.7. Sod 3D on a spherical mesh

The initial data is the Sod shock tube problem in a 3D spherical geometry. The initial mesh is layered in the high-density/high-pressure exterior zone. In the low-density/low-pressure interior zone, the mesh is continuously transformed into a logical mesh. The initial number of layers in the exterior zone is 20, therefore the resolution is poor in the radial direction.

We perform two calculations. The first calculation is purely Lagrangian. For the second one we use a adaptive mesh refinement (AMR) technique in the exterior zone: we compare the radial length of the layers with a predefined length equal to the length at $t = 0$. If the layer is stretched too much in the radial direction, then we refine it and define two news layers. The projection method that we used in this calculation is very basic: we simply project with a first-order method. That is the solution (the conservative quantities) in the refined cells is equal to its value in the mother cell. By construction the total mass, the total impulse and the total energy is the same before and after the AMR stage of the algorithm. For this problem, we have observed that such a first-order method is enough to increase a lot the quality of the numerical solution. The development of a second-order projection method is less evident.

The results are displayed in Fig. 23. On the left is the purely Lagrangian calculation with a constant number of layers, on the right is the AMR-Lagrange calculation with an increasing number of layers (32 at the end of the calculation). On Fig. 24 we plot the density profile for a radial cut at the final time $t = 0.2$. We also plot a reference solution computed with a 1D code in spherical geometry, and with a large number of cells to get an almost converged solution. We observe that the purely Lagrangian calculation displays large errors at the focusing shock. But with the refinement technique in the exterior zone, the shock is captured at the correct position. Our interpretation is of course the better accuracy in the rarefaction fan, even with the first-order AMR projection method. We can also remark that the refinement technique allows to keep a good cell aspect ratio (see paragraph 6.2) during the whole calculation which limits the transverse numerical dissipation (which is known to be penalizing for such computation). Notice also the good aspect ratio of the cells in the radial direction for the AMR computation, which is compatible with the Proposition 5. It shows the interest of layer refinement for Lagrangian computations.

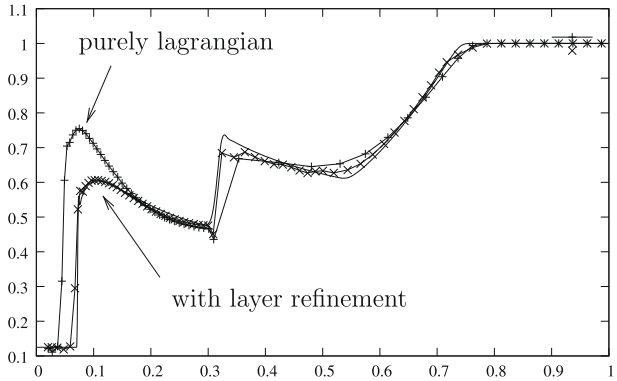


Fig. 24. Density profiles for the 3D Sod shock tube problem. Three curves: purely Lagrangian, with layer refinement and reference solution. The calculation with layer refinement is closer to the reference solution.

9. Conclusion

In this paper we have presented in detail the mathematical structure of the discrete GLACE scheme for the numerical integration of Lagrangian compressible fluid dynamics. A basic idea is to have a Finite Volume node based discrete numerical method, compatible by construction with the displacement of the cells, as in the 2D schemes [13,22]. The 3D GLACE method is an extension of [13] in any dimension and on arbitrary meshes. In this appendix we describe the formulas for the isoparametric representation of an hexahedron. We have also described a simple second-order extension. The numerical tests show that this procedure is accurate and stable, so it allows to compute basic test problems in 3D. We have explained an important feature of GLACE: for a simple isentropic compression problem which is a thin shell test problem, GLACE is much less dissipative than the edge based first-order Godunov scheme. The reason is that the normals are corner-based in GLACE, on the contrary they are edge based for other more standard Godunov methods [1], see also the recent works [22,24]. These issues are linked with the way the boundary conditions are implemented: for the more standard Lagrangian Godunov methods [1,22,24] the treatment of the boundary conditions is more immediate since the boundary conditions never interact. For GLACE we have described in Section 4 how to incorporate all standard Lagrangian boundary conditions in a coherent way. Since the scheme has low viscosity, the price to pay is somehow less damping of spurious modes (the famous hourglass modes). With this respect GLACE behaves as the Von Neumann–Richtmyer scheme [26]. For meshes which are portions of the sphere, the results show a good behavior of the scheme near the reduced connectivity point (or line). Since GLACE is cell-centered, it is compatible by construction with ALE techniques [3,2] and/or with AMR techniques [20]. It can dramatically enhance the robustness and accuracy of Lagrangian computations. We will report in a further work dedicated to the coupling of Glace with these techniques.

Appendix A. $\mathbf{C}_{jr}$ for hexahedra in 3D

Considering eight distinct vertices, there are many ways to define what is an hexahedron. Once a representation and then a volume are defined, the $\mathbf{C}_{jr}$ vectors are uniquely defined thanks to (2). Notice that the verification of $V_j = \frac{1}{3}\sum_r(\mathbf{C}_{jr}, \mathbf{x}_r)$ is not sufficient to establish to $\mathbf{C}_{jr} = \nabla_{\mathbf{x}_r} V_j$. In 3D one has as well $V_j = \frac{1}{3}\sum_r(\mathbf{C}_{jr} + \mathbf{d}_r \wedge \mathbf{x}_r, \mathbf{x}_r)$ where $\mathbf{d}_r$ are arbitrary vectors. In what follows we give a possibility for the definition the geometry of the hexahedron and the associated $\mathbf{C}_{jr}$. We recall that the problem comes from non-planar faces. The method uses an isoparametric representation of the hexahedron, for which the normals are not well defined on the faces of the cell. We recover also the so-called eight nodes hexahedral element [36,14]. The advantage of our approach with respect to [36] is in the use of the formula (A.1) which simplifies a lot the calculations.

The isoparametric representation of the hexahedron with possible warped faces represented in Fig. 1 is: $\mathbf{x}(\alpha,\beta,\gamma) = \sum_{r=1}^{8} \mathbf{x}_r \lambda_r(\alpha,\beta,\gamma)$ where $0 \leqslant \alpha,\beta,\gamma \leqslant 1$. The functions λ_r are the 8 barycentric functions of the unit reference cube: $\lambda_1 = (1-\alpha)(1-\beta)(1-\gamma)$, $\lambda_2 = \alpha(1-\beta)(1-\gamma)$, and so on. The vertices of the hexahedron are the $\mathbf{x}_r = (x_r, y_r, z_r)$. The Jacobian of the transformation is $J = \det\left(\sum_{r=1}^{8} \mathbf{x}_r \otimes \nabla\lambda_r\right) = \det\left(\sum_{r=1}^{8} x_r \nabla\lambda_r, \sum_{r=1}^{8} y_r \nabla\lambda_r, \sum_{r=1}^{8} z_r \nabla\lambda_r\right)$. So the volume of the hexahedron is

$$V = \int_{0\leqslant\alpha,\beta,\gamma\leqslant 1} \det\left(\sum_{r=1}^{8} x_r \nabla\lambda_r, \sum_{r=1}^{8} y_r \nabla\lambda_r, \sum_{r=1}^{8} z_r \nabla\lambda_r\right) d\alpha d\beta d\gamma.$$

Let us compute C_1^1 the gradient of V with respect to x_1 which is the first coordinate of $\mathbf{x}_1$. One has

$$\begin{aligned} C_1^1 = \nabla_{x_1} V &= \int_{0\leqslant\alpha,\beta,\gamma\leqslant 1} \det\left(\nabla\lambda_1, \sum_{r=1}^{8} b_r \nabla\lambda_r, \sum_{s=1}^{8} c_s \nabla\lambda_s\right) d\alpha d\beta d\gamma \\ &= \sum_{r=2}^{8} \sum_{s=r+1}^{8} (b_r c_s - c_r b_s) \int_{0\leqslant\alpha,\beta,\gamma\leqslant 1} \det(\nabla\lambda_1, \nabla\lambda_r, \nabla\lambda_s) d\alpha d\beta d\gamma. \end{aligned}$$

More generally

$$\mathbf{C}_1 = \nabla_{\mathbf{x}_1} V = \sum_{r=2}^{8} \sum_{s=r+1}^{8} \mathbf{x}_r \wedge \mathbf{x}_s \int_{0\leqslant\alpha,\beta,\gamma\leqslant 1} \det(\nabla\lambda_1, \nabla\lambda_r, \nabla\lambda_s) d\alpha d\beta d\gamma.$$

One has the identity: $\det(\nabla\lambda_1, \nabla\lambda_r, \nabla\lambda_s) = \nabla.(\lambda_1 \nabla\lambda_r \wedge \nabla\lambda_s)$. Therefore

$$\int_{0\leqslant\alpha,\beta,\gamma\leqslant 1} \det(\nabla\lambda_1, \nabla\lambda_r, \nabla\lambda_s) d\alpha d\beta d\gamma = \int_{\partial\{0\leqslant\alpha,\beta,\gamma\leqslant 1\}} \lambda_1 (\nabla\lambda_r \wedge \nabla\lambda_s, \mathbf{n}) d\sigma \tag{A.1}$$

where $\mathbf{n}$ is the outgoing normal of the unit cube. This formula is used to simplify the volume integral. The boundary of the unit cube is made of 6 different faces. Notice first that the face integrals vanish if the first node $\alpha = \beta = \gamma = 0$ (which corresponds to $\mathbf{x}_1$ in our numbering) does not belong to the face because λ_1 vanishes on these faces. So it is sufficient to compute the face integrals on the three faces which have the first node as a common vertice. Second consider one of these three faces, for example the face delimited by the vertices 1, 2, 3 and 4 with the notations of Fig. 1. The calculation of the integral can be

split in three different cases: $r=2$ and $s=3; r=2$ and $s=4; r=3$ and $s=4$. On this face the outgoing normal is $\mathbf{n}=(0,0,-1)$. The first case corresponds to $A=-\int_{0\leqslant\alpha,\beta\leqslant 1}\lambda_1(\partial_\alpha\lambda_2\partial_\beta\lambda_3-\partial_\beta\lambda_2\partial_\alpha\lambda_3)d\alpha d\beta$. The second case corresponds to $B=-\int_{0\leqslant\alpha,\beta\leqslant 1}\lambda_1(\partial_\alpha\lambda_2\partial_\beta\lambda_4-\partial_\beta\lambda_2\partial_\alpha\lambda_4)d\alpha d\beta$ and the third one to $C=-\int_{0\leqslant\alpha,\beta\leqslant 1}\lambda_1(\partial_\alpha\lambda_3\partial_\beta\lambda_4-\partial_\beta\lambda_3\partial_\alpha\lambda_4)d\alpha d\beta$. Recall that $\lambda_1=(1-\alpha)(1-\beta), \lambda_2=\alpha(1-\beta), \lambda_3=\alpha\beta$ and $\lambda_4=(1-\alpha)\beta$. Finally $A=B=C=\frac{1}{12}$. All coefficients on the other faces are computed by rotation of the indices.

Appendix B. Proof of property 5

Each cell is parallelepipedic $V_j=\Delta x_j\Delta y_j\Delta z_j$. The 8 vectors $\mathbf{C}_{jr}$ are

$$\mathbf{C}_{jr}=\frac{1}{4}\Delta y_j\Delta z_j(\pm\mathbf{e}_x)+\frac{1}{4}\Delta x_j\Delta z_j(\pm\mathbf{e}_y)+\frac{1}{4}\Delta x_j\Delta y_j(\pm\mathbf{e}_z)$$

where $\mathbf{e}_x,\mathbf{e}_y$ and $\mathbf{e}_z$ are the basis vectors. So for all r

$$|\mathbf{C}_{jr}|=\frac{\sqrt{\Delta x_j^2\Delta z_j^2+\Delta y_j^2\Delta z_j^2+\Delta x_j^2\Delta x_j^2}}{4}.$$

Therefore the matrix $\mathbf{B}_j$ is

$$\mathbf{B}_j=\frac{2}{\sqrt{\Delta x_j^2\Delta z_j^2+\Delta y_j^2\Delta z_j^2+\Delta x_j^2\Delta y_j^2}}\begin{pmatrix}\Delta y_j^2\Delta z_j^2 & 0 & \\ 0 & \Delta x_j^2\Delta z_j^2 & 0\\ 0 & 0 & \Delta x_j^2\Delta y_j^2\end{pmatrix} \tag{B.1}$$

In this case the largest eigenvalue of $\mathbf{B}_j$ is $\mu_j=\frac{2\Delta y_j^2\Delta z_j^2}{\sqrt{\Delta x_j^2\Delta z_j^2+\Delta y_j^2\Delta z_j^2+\Delta x_j^2\Delta y_j^2}}$ and the coefficient λ_j defined in (31) is $\lambda_j=\frac{\sqrt{\Delta x_j^2\Delta z_j^2+\Delta y_j^2\Delta z_j^2+\Delta x_j^2\Delta y_j^2}}{\Delta y_j\Delta z_j}$. Now consider the correction term $\lambda_j(\mathbf{u}_r-\mathbf{u}_j,\mathbf{n}_{jr})=(\mathbf{u}_r-\mathbf{u}_j,\lambda_j\mathbf{n}_{jr})$ in (31). By hypothesis the velocity $\mathbf{u}_j$ is oriented in the x direction. By symmetry it is also the case for the vertex velocity $\mathbf{u}_r$. It remains to evaluate the corrected normal $\lambda_j\mathbf{n}_{jr}$

$$\lambda_j\mathbf{n}_{jr}=\lambda_j\frac{\mathbf{C}_{jr}}{|\mathbf{C}_{jr}|}=\begin{pmatrix}\pm1\\0\\0\end{pmatrix}+\frac{\Delta x_j}{\Delta y_j}\begin{pmatrix}0\\\pm1\\0\end{pmatrix}+\frac{\Delta x_j}{\Delta z_j}\begin{pmatrix}0\\0\\\pm1\end{pmatrix}.$$

The important consequence is that the coefficient in the x direction is now ±1. So the scalar product of $\lambda_j\mathbf{n}_{jr}$ with $\mathbf{u}_r-\mathbf{u}_j$ is equal to $\pm|\mathbf{u}_r-\mathbf{u}_j|$. Therefore the Eqs. (10)–(12) degenerate (we just drop j_2 and j_3) to

$$\begin{cases}p_{j_1+\frac{1}{2}}-p_{j_1}+\rho_{j_1}c_{j_1}\left(u_{j_1+\frac{1}{2}}-u_{j_1}\right)=0,\\ p_{j_1+\frac{1}{2}}-p_{j_1+1}-\rho_{j_1+1}c_{j_1+1}\left(u_{j_1+\frac{1}{2}}-u_{j_1+1}\right)=0.\end{cases} \tag{B.2}$$

The solution is the standard Godunov acoustic scheme [17,18] in 1D (32). It ends the proof. □

References

[1] F.L. Addessio, J.R. Baumgardner, Dukowicz, N.L.J.K., Johnson, B.A. Kashiwa, R.M. Rauenzahn, C. Zemach. CAVEAT: A Computer Code for Fluid Dynamics Problems with Large Distortion and Internal Slip. Technical report, Los Alamos National Laboratory LA-10613, 1990.
[2] R.W. Anderson, N.S. Elliott, R.B. Pember, An arbitrary Lagrangian–Eulerian method with adaptive mesh refinement for the solution of Euler equations, J. Comput. Phys. 199 (2) (2005).
[3] D.J. Benson, Computational methods in Lagrangian and Eulerian hydrocodes, Comput. Method Appl. Mech. Eng. 99 (1992) 235–394.
[4] D.E Burton, Multidimensional discretization of conservation laws for unstructured polyhedral grids. Technical Report, Lawrence Livermore National Laboratory, UCRL-JC-118306, 1994.
[5] J. Campbell, M.J. Shashkov, The compatible Lagrangian hydrodynamics algorithm on unstructured grids. Technical Report, Los Alamos National Labs, LA-UR-00-323, 2000.
[6] E.J. Caramana, D.E. Burton, M.J. Shashkov, P.P. Whalen, The construction of compatible hydrodynamics algorithms utilizing conservation of total energy, J. Comput. Phys. 146 (1998) 227–262.
[7] E.J. Caramana, C.L. Roulscup, D.E. Burton, A compatible, energy and symmetry preserving Lagrangian hydrodynamics algorithm in three-dimensional Cartesian geometry, J. Comput. Phys. 157 (2000) 89–119.
[8] E.J. Caramana, R. Loubere, Curl-Q: a vorticity damping artificial viscosity for Lagrangian hydrodynamics calculations, J. Comput. Phys. 215 (2) (2006) 385–391.
[9] E.J. Caramana, M.J. Shashkov, P.P. Whalen, Formulations of artificial viscosity for multidimensional shock wave computations, J. Comput. Phys. 144 (1998) 70–97.
[10] J. Cheng, C.W. Shu, A high order ENO conservative Lagrangian type scheme for the compressible Euler equations, J. Comput. Phys. 227 (2007) 1567–1596.
[11] R.B. Christensen, Godunov methods on a staggered mesh, an improved artificial viscosity. Tech. Rep. UCRL-JC 105269 LLNL, 1990.
[12] B. Després, C. Mazeran, Symmetrization of Lagrangian gas dynamics and Lagrangian solvers, Comptes Rendus Académie des Sciences (Paris) 331 (2003) 475–480.
[13] B. Després, C. Mazeran, Lagrangian gas dynamics in 2D and Lagrangian systems, Arch. Rat. Mech. Anal. 178 (2005) 327–372.

[14] J.K. Dukowicz, Efficient volume computation for three-dimensional hexahedral cells, J. Comput. Phys. 74 (1988) 493–496.
[15] J.K. Dukowicz, B. Meltz, Vorticity errors in multidimensional Lagrangian codes, J. Comput. Phys. 99 (1992).
[16] D.P. Flanaghan, T. Belytshko, A uniform strain hexaedron and quadrilateral and orthogonal hourglass control, Int. J. Numer. Method Eng. 17 (1982) 679–706.
[17] S. Godunov, A difference scheme for numerical computation of discontinuous solution of hydrodynamic equations, Math. Sib. 47 (1959).
[18] S. Godunov, Reminiscences about difference schemes, J. Comput. Phys. 153 (1999) 6–25.
[19] W.H. Hui, Unified coordinate system in computational fluid dynamics, Commun. Comput. Phys. 2 (2006) 577–610.
[20] J.R. Leveque, Numerical methods for conservation laws. Lectures in Mathematics, Birkhauser, 1991.
[21] R. Loubere, J. Ovadia, R. Abgrall, A Lagrangian discontinuous Galerkin type method on unstructured meshes to solve hydrodynamics problems, Int. J. Numer. Method Fluids (2000).
[22] P.H. Maire, R. Abgrall, J. Breil, J. Ovadia, A cell-centered Lagrangian scheme for 2D compressible flow problems, Siam. J. Sci. Comput. 29 (2007).
[23] P.H. Maire, A high-order cell centered Lagrangian scheme for two-dimensional compressible fluid flows on unstructured meshes, J. Comput. Phys., 2009, online.
[24] P.H. Maire, B. Nkonga, Multi-scale Godunov type method for cell-centered discrete Lagrangian hydrodynamics, J. Comput. Phys., 2008, online (october).
[25] C. Mazeran, Sur la structure mathématique et l'approximation numérique de l'hydrodynamique lagrangienne bidimensionnelle. Ph.D. Thesis, Université de Bordeaux I, November 2007.
[26] J. von Neumann, R.D. Richtmyer, A method for the calculation of hydrodynamics shocks, J. Appl. Phys. 21 (1950) 232–237.
[27] W.E. Pracht, Calculating three-dimensional fluid flows at all speeds with an Eulerian–Lagrangian computing mesh, J. Comput. Phys. 17 (1975) 132–159.
[28] B. Scheurer, Quelques schémas numériques pour l'hydrodynamique Lagrangienne. Technical Report, Commissariat à l'Energie Atomique, CEA-R-5942, 2000.
[29] G. Scovazzi, A discourse on Galilean invariance, SUPG stabilization, and the variational multi-scale framework, Comput. Method Appl. Mech. Eng. 54 (6-8) (2007) 1108–1132.
[30] G. Scovazzi, Galilean invariance and stabilized methods for compressible flows, Int. J. Numer. Method Fluids 54 (6-8) (2007) 757–778.
[31] G. Scovazzi, Stabilized shock hydrodynamics: II. Design and physical interpretation of the SUPG operator for Lagrangian computations, Comput. Method Appl. Mech. Eng. 196 (4–6) (2007) 967–978.
[32] G. Scovazzi, M.A. Christon, T.J.R. Hugues, J.N. Shadid, Stabilized shock hydrodynamics: I.A Lagrangian method, Comput. Method Appl. Mech. Eng. 196 (4–6) (2007) 923–966.
[33] G. Scovazzi, E. Love, M.J. Shashkov, Multi-scale Lagrangian shock hydrodynamics on Q1/P0 finite elements: Theoretical framework and two-dimensional computations, Comput. Method Appl. Mech. Eng. 197 (2008) 1056–1079.
[34] M. Shashkov, Conservative Finite Difference Methods on General Grids, CRC Press, 1996.
[35] G.A. Sod, A survey of finite difference methods for systems of nonlinear conservation laws, J. Comput. Phys. 27 (1978) 1–31.
[36] L.M. Taylor, D.P. Flanaghan, PRONTO 3D, A three-dimensional transient solid dynamics program. Technical Report, Sandia National Laboratories Sand87-1912, 1989.
[37] E.F. Toro, Riemann Solvers and Numerical Methods for Fluid Dynamics, Springer, 1997.
[38] P.P. Whalen, Algebraic limitations on two-dimensional hydrodynamics simulations, J. Comput. Phys. 124 (1996) 46–54.
[39] M. Wilkins, Use of artificial viscosity in multidimensional shock wave problems, J. Comput. Phys. 3 (36) (1980) 281–303.