A CELL-CENTERED ARBITRARY LAGRANGIAN EULERIAN (ALE) METHOD FOR MULTIMATERIAL COMPRESSIBLE FLOWS

P.-H. MAIRE[1], M. DE BUHAN[2], A. DIAZ [3], C. DOBRZYNSKI[4], G. KLUTH[5] AND F. LAGOUTIÈRE[6]

Abstract. We present an original and accurate unstructured cell-centered ALE algorithm devoted to the simulation of two-dimensional multi-material compressible fluid flows.

Résumé. Nous présentons dans cet article une méthode du type ALE appliquée à la résolution d'écoulements multimatériaux compressibles. Les équations de la dynamique des gaz sont discrétisées sur des maillages non structurés en utilisant un schéma du type volume fini.

INTRODUCTION

Fidelity in tracking multifluid interfaces makes the Lagrangian frame of reference attractive for a wide variety of hydrocode applications. Unfortunately, since the mesh moves with the flow, convolution of the flow can lead to mesh deformation and tangling. The Arbitrary Lagrangian-Eulerian (ALE) framework, introduced by [5], allows a computation to be broken up into a Lagrangian phase, a mesh rezoning phase intended to correct for mesh deformation, and a remapping phase where computed physical values are advanced to the rezoned mesh. The Arbitrary Lagrangian-Eulerian framework has demonstrated the ability to simultaneously elicit the advantages of both the Lagrangian and Eulerian frames of reference. The goal of this paper is to describe an original cell-centered ALE strategy for multi-material fluid flows and to focus more precisely on the description of the rezoning phase. Classically, Lagrangian step uses a staggered scheme, in which velocities are vertex-centered and the other variables are cell-centered [3]. The main difficulty with this approach lies in the fact that one needs special treatment for momentum remapping [9]. As our Lagrangian step is fully cell-centered [11,12], we avoid such special treatment. Our rezoning step utilizes the "local" minimization of nodally based objectives functions [6]. The remapping step is based on an unstructured extension of the "simplified face-based donor-cell" method of [14]. The remainder of this paper is organized as follows. We first describe the different steps of the ALE formulation. Then, computational results are given to access the robustness and the accuracy of this method.

---

[1] UMR CELIA, Université Bordeaux I, 351 Cours de la Libération, 33 405 Talence, France

[2] Université Pierre et Marie Curie

[3] Department of Mathematics and Computer Science, Santa Clara University, 500 El Camino Real, Santa Clara, CA 95053-0290, USA

[4] IMB, Université Bordeaux I, Université Bordeaux I, 351 Cours de la Libération, 33 405 Talence, France

[5] CEA-DIF/DSSI, BP 12 91680 Bruyères Le Châtel, France

[6] Université Denis Diderot

1. Governing equations

Let $\mathcal{D}$ be a region of the two-dimensional space $\mathbb{R}^2$, filled with an inviscid fluid and equipped with an orthonormal frame. It is convenient, from the point of view of subsequent discretization to write the unsteady compressible Euler equations in the control volume formulation which holds for an arbitrary moving control volume

$$\frac{d}{dt} \int_{\Omega(t)} \mathrm{d}\Omega - \int_{\partial\Omega(t)} \dot{\boldsymbol{X}} \cdot \boldsymbol{N} \, \mathrm{d}l = 0, \tag{1a}$$

$$\frac{d}{dt} \int_{\Omega(t)} \rho \, \mathrm{d}\Omega + \int_{\partial\Omega(t)} \rho \left(\boldsymbol{V} - \dot{\boldsymbol{X}} \right) \cdot \boldsymbol{N} \rho \, \mathrm{d}l = 0, \tag{1b}$$

$$\frac{d}{dt} \int_{\Omega(t)} \rho \boldsymbol{V} \, \mathrm{d}\Omega + \int_{\partial\Omega(t)} \left[\left(\boldsymbol{V} - \dot{\boldsymbol{X}} \right) \cdot \boldsymbol{N} \rho \boldsymbol{V} + P\boldsymbol{N} \right] \, \mathrm{d}l = \boldsymbol{0}, \tag{1c}$$

$$\frac{d}{dt} \int_{\Omega(t)} \rho E \, \mathrm{d}\Omega + \int_{\partial\Omega(t)} \left[\left(\boldsymbol{V} - \dot{\boldsymbol{X}} \right) \cdot \boldsymbol{N} \rho E + P\boldsymbol{V} \cdot \boldsymbol{N} \right] \, \mathrm{d}l = 0. \tag{1d}$$

Here, $\Omega(t)$ is the moving control volume, and $\partial\Omega(t)$ its boundary. $\rho, \boldsymbol{V}, P, E$ are the density, velocity, pressure, and specific total energy of the fluid. $\boldsymbol{N}$ denotes the unit outward normal vector to the moving boundary $\partial\Omega(t)$ whose velocity is denoted $\dot{\boldsymbol{X}}$. The set of previous equations is referred to as the ALE integral form of the Euler equations and can be found in many papers [1, 5]. The first equation, (1a), is a geometric conservation law (GCL), whereas (1b), (1c) and (1d) express the conservation of mass, momentum and total energy. When $\dot{\boldsymbol{X}} = \boldsymbol{V}$, we recover the Lagrangian description of the fluid flow for which the control volume moves with the fluid velocity. On the other hand, when $\dot{\boldsymbol{X}} = \boldsymbol{0}$, we get the Eulerian description.

The thermodynamical closure of the set of previous equations is obtained by the addition of an equation of state which is taken to be of the form

$$P = P(\rho, \varepsilon), \tag{2}$$

where the specific internal energy, ε, is related to the specific total energy by $\varepsilon = E - \frac{1}{2} \mid \boldsymbol{V} \mid^2$.

It is possible to formulate the ALE scheme as a single algorithm [10] based on solving the previous equations in a moving coordinate frame. However it is more usual to separate it into three phases. These are: a Lagrangian phase in which the solution and the grid are updated; a rezoning phase in which the nodes of the computational grid are moved to a more optimal position; and a remapping phase in which the Lagrangian solution is interpolated onto the rezoned grid.

2. Lagrangian phase

The Lagrangian phase consists in computing the rates of change of volume, mass, momentum and energy, assuming that the computational volumes are following the material motion. By setting $\dot{\boldsymbol{X}} = \boldsymbol{V}$ in the set of equations (1) one gets

$$\frac{d}{dt} \int_{\Omega_L} \mathrm{d}\Omega - \int_{\partial\Omega_L} \boldsymbol{V} \cdot \boldsymbol{N} \, \mathrm{d}l = 0, \tag{3a}$$

$$\frac{d}{dt} \int_{\Omega_L} \rho \, \mathrm{d}\Omega = 0, \tag{3b}$$

$$\frac{d}{dt} \int_{\Omega_L} \rho \boldsymbol{V} \, \mathrm{d}\Omega + \int_{\partial\Omega_L} P\boldsymbol{N} \, \mathrm{d}l = \boldsymbol{0}, \tag{3c}$$

$$\frac{d}{dt} \int_{\Omega_L} \rho E \, \mathrm{d}\Omega + \int_{\partial\Omega_L} P\boldsymbol{V} \cdot \boldsymbol{N} \, \mathrm{d}l = 0, \tag{3d}$$
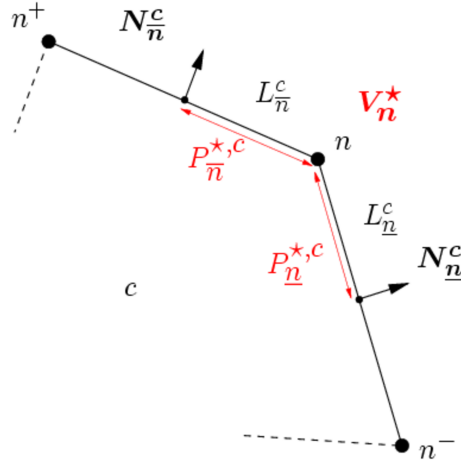
Figure 1. Notations in a cell

where Ω_L is the Lagrangian (material) control volume, $\partial\Omega_L$ is its surface, and the time rates of change refers to quantities associated with this volume. We notice that equation (3b) implies that the mass of the control volume remains constant during the Lagrangian phase.

2.1. Spatial approximation

In order to recall the spatial approximation of the Lagrangian phase derived in [11], we introduce the following notations. Let $\{c\}$ be a collection of non overlapping polygons whose reunion covers the domain filled by the fluid. Each cell is labelled with a unique index c. We denote by $\{n\}$ the set of all the vertices of the cells. Each vertex is labelled with a unique index n. If we consider a given cell c, we introduce the set of all the vertices of the cell c and denote it by $\mathcal{N}(c)$. For a given node n, we also define the set of all the cells that share this vertex and denote it by $\mathcal{C}(n)$. The sets $\mathcal{N}(c)$ and $\mathcal{C}(n)$ are counterclockwise ordered. For a node $n \in \mathcal{N}(c)$, n^- and n^+ are the previous/next nodes with respect to n in the list of vertices of cell c, see Figure 1. We denote by $L_{\underline{n}}^c$ and $L_{\overline{n}}^c$ the half length of the edges $[n^-, n]$ and $[n, n^+]$. We use the same notations to define the unit normal outward $\boldsymbol{N_{\underline{n}}^c}$ and $\boldsymbol{N_{\overline{n}}^c}$, see Figure 1.

All fluid variables are assumed to be constant in cell c and we denote them by using subscript c. Therefore, we obtain a spatial approximation which is first order accurate. A set of evolution equations is written for the the discrete unknowns $(\tau_c, \boldsymbol{V_c}, E_c)$ using the Lagrangian conservation equations in control volume form applied for cell c, see [11].

$$m_c \frac{d}{dt}\tau_c - \sum_{n\in\mathcal{N}(c)} \left(L_{\underline{n}}^c \boldsymbol{N_{\underline{n}}^c} + L_{\overline{n}}^c \boldsymbol{N_{\overline{n}}^c} \right) \cdot \boldsymbol{V_n^\star} = 0, \tag{4a}$$

$$m_c \frac{d}{dt}\boldsymbol{V_c} + \sum_{n\in\mathcal{N}(c)} \left(L_{\underline{n}}^c P_{\underline{n}}^{\star,c} \boldsymbol{N_{\underline{n}}^c} + L_{\overline{n}}^c P_{\overline{n}}^{\star,c} \boldsymbol{N_{\overline{n}}^c} \right) = \boldsymbol{0}, \tag{4b}$$

$$m_c \frac{d}{dt}E_c + \sum_{n\in\mathcal{N}(c)} \left(L_{\underline{n}}^c P_{\underline{n}}^{\star,c} \boldsymbol{N_{\underline{n}}^c} + L_{\overline{n}}^c P_{\overline{n}}^{\star,c} \boldsymbol{N_{\overline{n}}^c} \right) \cdot \boldsymbol{V_n^\star} = 0. \tag{4c}$$

In these equations $\tau_c = \frac{1}{\rho_c}$ is the specific volume and m_c is the mass in the cell c that is constant during time. $\boldsymbol{V_n^\star}$ denotes the nodal velocity. The main feature of our discretization is the introduction of the two pressure fluxes $P_{\underline{n}}^{\star,c}$ and $P_{\overline{n}}^{\star,c}$ at the node n, see Figure 1. $P_{\underline{n}}^{\star,c}$ (resp. $P_{\overline{n}}^{\star,c}$) represents the pressure acting between the midpoint of edge $[n^-, n]$ (resp. $[n, n^+]$) and the node n, seen from cell c. The Cartesian coordinates of node n

are $\boldsymbol{X_n} = (X_n, Y_n)^t$. The motion of the nodes is given by the trajectories equation $\frac{d}{dt}\boldsymbol{X_n} = \boldsymbol{V_n^\star}$, $\boldsymbol{X_n}(0) = \boldsymbol{x_n}$. We note that (4a) is consistent with the nodes motion.

2.2. Nodal solver

The nodal fluxes $\boldsymbol{V_n^\star}$, $P_{\underline{n}}^{\star,c}$ and $P_{\overline{n}}^{\star,c}$ are computed in a consistent manner due to a solver located at the nodes. We give hereafter a summary of fluxes computation. Let Z_c denotes the acoustic impedance of cell c and $\mathsf{M}_n^c = Z_c \left(L_{\underline{n}}^c \boldsymbol{N_{\underline{n}}^c} \otimes \boldsymbol{N_{\underline{n}}^c} + L_{\overline{n}}^c \boldsymbol{N_{\overline{n}}^c} \otimes \boldsymbol{N_{\overline{n}}^c} \right)$. The nodal velocity is written

$$\boldsymbol{V_n^\star} = \left(\sum_{c \in \mathcal{C}(n)} \mathsf{M}_n^c \right)^{-1} \sum_{c \in \mathcal{C}(n)} \left[\left(L_{\underline{n}}^c \boldsymbol{N_{\underline{n}}^c} + L_{\overline{n}}^c \boldsymbol{N_{\overline{n}}^c} \right) P_c + \mathsf{M}_n^c \boldsymbol{V_c} \right], \tag{5}$$

where $\mathcal{C}(n)$ is the set of the cells that share node n. The pressure fluxes are given by the following discrete Riemann invariants

$$P_c - P_{\underline{n}}^{\star,c} = Z_c \left(\boldsymbol{V_n^\star} - \boldsymbol{V_c} \right) \cdot \boldsymbol{N_{\underline{n}}^c},$$
$$P_c - P_{\overline{n}}^{\star,c} = Z_c \left(\boldsymbol{V_n^\star} - \boldsymbol{V_c} \right) \cdot \boldsymbol{N_{\overline{n}}^c}.$$

It is shown in [11] that these fluxes lead to a conservative and entropy consistent scheme. This scheme computes in a coherent way the vertex motion as well as the nodal fluxes, and for 1D flows, it recovers exactly the Godunov acoustic fluxes.

2.3. Second order extension

The spatial second order extension is obtained by a piecewise linear monotonic reconstruction of the pressure and velocity, given by their mean values over mesh cells. Let ϕ denotes a fluid variable (pressure or velocity components), we assume a linear variation for ϕ in cell c

$$\phi_c(\boldsymbol{X}) = \phi_c + \boldsymbol{\nabla}\phi_c \cdot (\boldsymbol{X} - \boldsymbol{X_c}). \tag{6}$$

Here, $\boldsymbol{X_c} = \frac{1}{|c|} \int_c \boldsymbol{X} \, d\Omega$ is the cell centroid and $|c| = \int_c d\Omega$ is the volume of cell c. The gradient in (6) is approximated by a least squares procedure [2]. To preserve monotonicity, we limit the value that the gradient is allowed to take, using a multidimensional extension of the van Leer's classical method. We introduce a limiting coefficient α_c ($0 \le \alpha_c \le 1$) for each cell c such that

$$\boldsymbol{\nabla}\phi_c = \alpha_c \widehat{\boldsymbol{\nabla}\phi_c},$$

where the hat symbol denotes the approximated gradient obtained by least squares. The coefficient α_c is determined by enforcing a local monotonicity criterion so that the quantity ϕ in the cell c does not lie outside the range of the average quantities in the neighboring cells. Let $\phi_c^{\max}$, $\phi_c^{\min}$ be the maximum and the minimum values of ϕ in the neighboring cells. We also define $\phi_c^{\star,\max}$, $\phi_c^{\star,\min}$ the maximum and the minimum values of ϕ in cell c obtained from (6) with $\widehat{\boldsymbol{\nabla}\phi_c}$. Then, the coefficient α_c is determined by

$$\alpha_c = \min(1, \alpha_c^{\min}, \alpha_c^{\max}),$$

where

$$\alpha_c^{\max} = \max(0, \frac{\phi_c^{\max} - \phi_c}{\phi_c^{\star,\max} - \phi_c}),$$

$$\alpha_c^{\min} = \max(0, \frac{\phi_c^{\min} - \phi_c}{\phi_c^{\star,\min} - \phi_c}).$$

Finally, instead of using the mean values of the pressure and the velocity in our nodal solver, we use their nodal extrapolated values deduced from the linear monotonic reconstruction. The time discretization is explicit and based on a classical two steps Runge-Kutta procedure.

3. Rezoning phase

The rezoning phase consists in moving the node of the Lagrangian grid to improve the geometric quality of the grid while keeping the rezoned grid a close as possible to the Lagrangian grid. This constraint must be taken into account to maintain the accuracy of the computation brought by the Lagrangian phase [7]. In fact, by requiring the rezoned grid to remain as close as possible to the Lagrangian grid, we minimize the error of the remap phase, and we justify employing a local remapper in which mass, momentum and energy are simply exchanged between neighboring cells, see Section 4.

We will describe two types of smoothing algorithm, the first one is the well known Laplacian smoothing and the second one is the condition number smoothing which is closely related to the Winslow algorithm [6]. For both types of rezoning we make the assumption that the input grid (the grid produced by the Lagrangian phase) is unfolded. If the Lagrangian phase produces non valid cells then we have to use an untangling procedure [18].

3.1. **Laplacian smoothing**

The regularization methods based on the resolution of a discrete Laplacian were first introduced in image processing (see [4]) for smoothing and denoising curves and surfaces. In fact, these methods can be seen as filters on the highest frequencies and are used to eliminate the random noise in data. The Laplacian smoothing (or barycentric regularization) is so far the most commonly used method for mesh smoothing mainly because it is easy to implement and rather inexpensive to carry out (the complexity is linear in the number of edges). This method consists in locally relaxing the position of each vertex in the direction of the weighted barycenter of its neighbors. It can be obtained by minimizing the following quadratic local objective function related to node n

$$F_l(\boldsymbol{X_n}) = \sum_{m \in \mathcal{V}(n)} \frac{1}{2} \omega_{nm} \|\boldsymbol{nm}\|^2, \tag{7}$$

where $\mathcal{V}(n)$ is the set of vertices that are connected to vertex n and $\omega_{nm} \in [0,1]$ is a weight such that $\sum_{m \in \mathcal{V}(n)} \omega_{nm} = 1$. The minimum of this functional is obtained for

$$\bar{\boldsymbol{X}}_{\boldsymbol{n}} = \sum_{m \in \mathcal{V}(n)} \omega_{nm} \boldsymbol{X_m}.$$

The rezoned position of node n, $\tilde{\boldsymbol{X}}_{\boldsymbol{n}}$ is defined as

$$\tilde{\boldsymbol{X}}_{\boldsymbol{n}} = \boldsymbol{X_n} + \lambda_n(\bar{\boldsymbol{X}}_{\boldsymbol{n}} - \boldsymbol{X_n}), \tag{8}$$

where the relaxing parameter,$\lambda_n \in [0,1]$, is a number which can be specified for each vertex or global ($\lambda_n = \lambda$). The weights can be chosen in many different ways. A specific trivial choice, although producing good results is to set $\omega_{nm} = \frac{1}{N(n)}$, where $N(n)$ is the the number of nodes connected to node n.

The main advantage of this method is related to its simplicity. But, in image processing, it is well known (see [15]) that this simple method produces shrinkage, that is a diminution of the area (resp. the volume) of the closed curve (resp. surface) when the smoothing method is applied a large number of times. This can be explained by the fact that the single Laplacian smoothing not only suppresses the hight frequencies but also attenuates the low frequencies (except zero). In mesh regularization, this results in a loss of desirable geometric features (privileged directions, rezoning zone...) in the mesh which becomes fast over-smoothed, see Figure 2 (d). In many cases, this can be avoided by choosing suitable weights. An empirical choice that gives good results is to set the weights to the inverse distances between the vertex n and its neighbors m :

$$\omega_{nm} = \frac{1}{\|\boldsymbol{nm}\|} \tag{9}$$

and to normalize them by the sum of all the weights $\displaystyle\sum_{m \in \mathcal{V}(n)} \omega_{nm}$ to ensure $\omega_{nm} \in [0,1]$. This choice allows to keep the distance ratios of the initial mesh, see Figure 2 (f). In order to cure the shrinkage problem observed

(a)   (c)   (e)

(b)   (d)   (f)
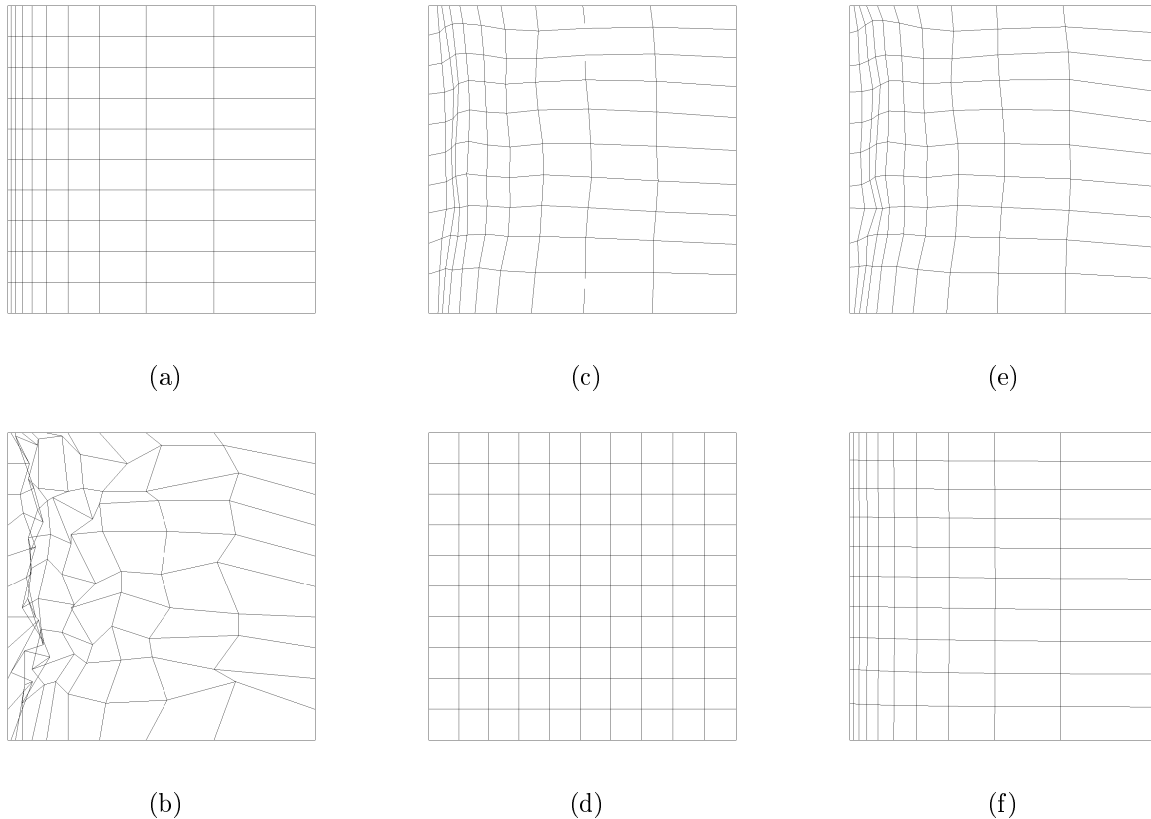
FIGURE 2. Laplacian smoothing with $\lambda = 0.5$, (a) initial mesh, (b) randomly perturbed mesh, (c) after 3 iterations without weights, (d) after 300 iterations (convergence) without weights, (e) after 3 iterations with weights, (e) after 300 iterations (convergence) with weights

with a single Laplacian smoothing G. Taubin has introduced an alternative method [16]. It consists of two

consecutive Laplacian smoothing steps:

$$\boldsymbol{X_n^\star} = \boldsymbol{X_n} + \lambda(\boldsymbol{\bar{X}_n} - \boldsymbol{X_n}),$$
$$\boldsymbol{\tilde{X}_n} = \boldsymbol{X_n^\star} - \mu(\boldsymbol{\bar{X}_n^\star} - \boldsymbol{X_n^\star}).$$

After a first Laplacian smoothing step with a positive scale factor λ applied to all vertices of the mesh, a second Laplacian smoothing step is applied to all the vertices, but with a negative scale factor $-\mu$, greater in magnitude than the first scale factor: $0 < \lambda < \mu$.

To produce a significant smoothing effect, these two steps must be repeated, alternating the positive and negative scale factors, a number of times. In fact, this method produces a low pass filter effect, the two scale factors determining the pass-band limit (see below). The Taubin's method presents the same advantages (simplicity and linear complexity) than the Laplacian smoothing. It suppresses high frequencies of a discrete Laplacian operator defined in the mesh but it does not produce shrinkage. Nevertheless, when applied a large number of times, Taubin's smoothing enhances low-frequency oscillations which can invalid the mesh and this can not be avoided even by introducing weights.

Finally, we briefly explain why the Laplacian smoothings can be considered as low pass filters. For more explanations, we refer to [17]. Let N denotes the total number of nodes, we introduce W the $N \times N$ matrix of weights under the convention that $\omega_{nm} = 0$ if $m \notin \mathcal{V}(n)$, that is, m is not connected to n

$$\mathsf{W} = (\omega_{nm})_{1 \leq n,m \leq N}\,.$$

This matrix contains all the information relative on the mesh connectivity and on the distances for instance, if the weights have been chosen as the inverse of the distances between the vertices. Let $\boldsymbol{X}$ be the vector of the positions of the n mesh vertices. Then, the Laplacian operator can be written as

$$\Delta\boldsymbol{X} = (\mathsf{W} - \mathsf{I})\boldsymbol{X}.$$

Let $\mathsf{K} = -(\mathsf{W} - \mathsf{I})$, if we assume that K is a normal matrix, one can show that it is a positive matrix, hence it has N real positive eigenvalues. This depends on the mesh connectivity and it is the case in all the examples we have presented. Then, the eigenvectors of the matrix K can be seen as the vibration mode of the mesh and the corresponding eigenvalues k_n as the related frequencies. Filtering a signal $\boldsymbol{X}$ is equivalent to change its frequency according to a transfer function $f(k)$:

$$\tilde{\boldsymbol{X}} = f(\mathsf{K})\boldsymbol{X},$$

where $f(\mathsf{K})$ is a matrix having the same eigenvectors as K but with eigenvalues equal to $f(k_n)$. The component of $\boldsymbol{X}$ corresponding to the frequency k_n is enhanced or attenuated by a factor $f(k_n)$. The transfer function for an ideal low pass filter with pass band frequency k_{PB} is :

$$f(k) = \begin{cases} 1 & \text{for } 0 \leq k \leq k_{\mathrm{PB}}, \\ 0 & \text{for } k_{\mathrm{PB}} \leq k. \end{cases}$$

It suppresses all the frequencies greater than k_{PB} and do not change the others. But it is very expensive to compute. For the Laplacian smoothing, the transfer function corresponding to the L-th iteration is $f_L(k) = (1 - \lambda k)^L$. This function produces shrinkage because:

$$\lim_{L \to \infty} (1 - \lambda k)^L = \begin{cases} 1 & \text{for } k = 0, \\ 0 & \text{for } k > 0. \end{cases}$$

That is, as the number of iterations grows, all the frequencies are attenuated. For Taubin's smoothing, the
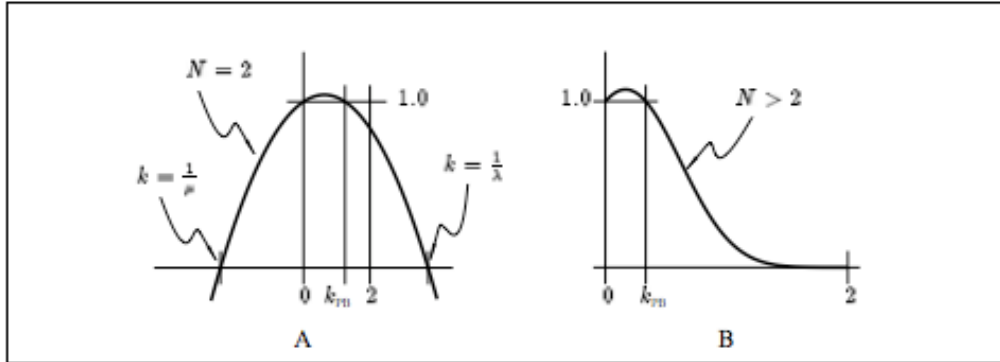
FIGURE 3. Graph of transfer function for Taubin's smoothing - reprinted from [17]
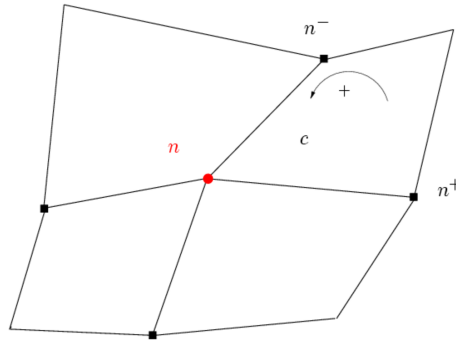

FIGURE 4. Notations for condition number smoothing

transfer function is $f_L(k) = ((1 - \lambda k)(1 + \mu k))^L$, see Figure 3. This simple modification still produces smoothing but prevents shrinkage. The pass band frequency of this filter is defined as the unique value of k such that $f_L(k) = 1$. Such a value exists when $0 < \lambda < \mu$ and turns out to be equal to $k_{\mathrm{PB}} = \frac{1}{\lambda} - \frac{1}{\mu}$. So, all the frequencies greater than k_{PB} will be attenuated when those smaller than k_{PB} will be slightly enhanced.

Laplacian type smoothing methods are attractive because of their simplicity. Acting as low pass filters, they permits to smooth fast the mesh. Unfortunately, these methods operate heuristically and there is no theoretical guarantee that they lead to valid meshes or elements of better quality than the original mesh. So, the process can not be fully automated. Then, we have to adjust manually the coefficients and the number of iterations for each case.

3.2. Condition number smoothing

This smoothing method has been introduced by Knupp [6] for mesh quality optimization. We recall hereafter its main features. Let c be a given cell of the Lagrangian grid, $n \in \mathcal{N}(c)$ a node of this cell and n^-, n^+ the previous and next nodes with respect to n in the list of the vertices of cell c, see Figure 4. We define $\mathsf{J}_{\mathsf{cn}} = [\boldsymbol{nn}^+ \mid \boldsymbol{nn}^-]$ the 2×2 Jacobian matrix associated with each corner at a vertex n of cell c. The condition number of matrix J_{cn} in $\mathbb{R}^2$ is defined as

$$\kappa(\mathsf{J}_{\mathsf{cn}}) = \frac{\|\boldsymbol{nn}^+\|^2 + \|\boldsymbol{nn}^-\|^2}{A_{cn}}, \tag{10}$$

where A_{cn} is the area of the triangle between edges $[nn^+]$ and $[nn^-]$. We note that κ is minimal for a 90 degree corner with edges of equal length. With this condition number, we define a local objective function associated to node n

$$F_{\mathrm{cn}}(\boldsymbol{X_n}) = \sum_{c\in\mathcal{C}(n)} \sum_{m\in\mathcal{V}(n)} \kappa(\mathsf{J}_{\mathsf{cm}}), \tag{11}$$

where $\mathcal{C}(n)$ is the set of cells that share node n and $\mathcal{V}(n)$ is the set of vertices that are connected to vertex n (including n itself). By summing this functional over all the grid nodes, we can also define a global objective function.

One very important feature that can be seen explicitly in the definition of $\kappa(\mathsf{J}_{\mathsf{cn}})$ is that the functional has a barrier, i.e., the value of the functional approaches infinity when the new grid approaches a non valid grid where $A_{cn} = 0$. This explains how the condition number smoothing produces unfolded grids.

The rezoned position of node n, $\tilde{\boldsymbol{X}}_{\boldsymbol{n}}$ is determined by minimizing the local functional (11) using a single step procedure

$$\tilde{\boldsymbol{X}}_{\boldsymbol{n}} = \boldsymbol{X_n} - \mathsf{H}_{\mathsf{n}}^{-1}(\boldsymbol{X_n}) \boldsymbol{\nabla} F_n(\boldsymbol{X_n}), \tag{12}$$

where H_{n} is the 2×2 Hessian matrix related to F_n. We separately minimize each of the local objective functions and iterating over all the nodes. This approach is termed as Jacobi sweep minimization. Its main advantage lies in the fact that the algorithm does not depend on the vertex order.

It is well known that the condition number smoothing has the tendency of driving the mesh to a square mesh. Therefore, it is unable to maintain non-uniformity which is beneficial when a higher node density is desired in the area of the flow feature of interest such as a shock boundary. To address this concern, we introduce the following local quadratic objective function

$$F_{\mathrm{w}}(\boldsymbol{X_n}) = \sum_{c\in\mathcal{C}(n)} \sum_{m\in\mathcal{V}(n)} w_c A_{cm}^2 \tag{13}$$

where w_c is a given positive weight define in cell c. By minimizing this functional, the node n displacement tends to adapt to the distribution of the weight function. The motion of node n is obtained by solving an equation analogous to (12).

Let $\delta\boldsymbol{X_n}^{\mathrm{cn}}$ (resp. $\delta\boldsymbol{X_n}^{\mathrm{w}}$) denotes the displacement of node n due to condition number smoothing (resp. weight smoothing). The final displacement of node n is defined as

$$\delta\boldsymbol{X_n} = \alpha\delta\boldsymbol{X_n}^{\mathrm{cn}} + (1-\alpha)\delta\boldsymbol{X_n}^{\mathrm{w}}, \tag{14}$$

where $\alpha \in]0,1]$ is a user defined parameter. For numerical applications, we will set the weight function w equal to the density of the fluid flow ρ in order to maintain a higher node density in the shock region. Finally, the displacement vector may be rescaled to the fluid velocity to prevent unnecessary mesh movement in domains which have not become dynamically active. This is achieved by setting

$$\delta\boldsymbol{X_n} \to \delta\boldsymbol{X_n} \min\left(1, \kappa\frac{\|\boldsymbol{V_n^\star}\|\Delta t}{\|\delta\boldsymbol{X_n}\|}\right),$$

where κ is an arbitrary scale constant, Δt the physical Lagrange time step, and $\boldsymbol{V_n^\star}$ the Lagrangian velocity of node n.

4. Remapping phase

The remapping phase is a conservative interpolation procedure of mass, momentum and total energy from the grid obtained at the termination of the Lagrangian phase to the new grid obtained from the rezoning phase. In the context of our ALE method, we consider that the Lagrangian grid and the rezoned one have the same
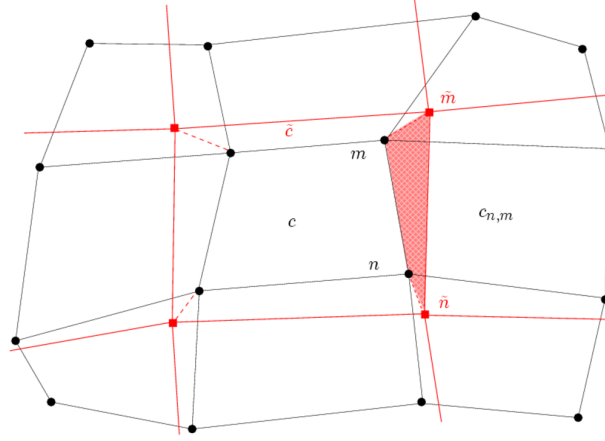
FIGURE 5. Lagrangian grid (thin lines and solid circles) and rezoned grid (thick lines and solid squares)

connectivity-i.e., the same number of cells and vertices, and the same neighbor relations. Let $\tilde{c}$ denotes the cell of the rezoned grid that corresponds to the cell c of the Lagrangian grid, see Figure 5.

Let $\psi \equiv \psi(\boldsymbol{X})$ be a density function of the flow, $\psi = (\rho, \rho\boldsymbol{V}, \rho E)$. At the end of the Lagrangian phase, for each cell c, we know $M_c = \int_c \psi(\boldsymbol{X})\,\mathrm{d}\Omega$, the mass associated to the density function and, also its corresponding mean value $\psi_c = \frac{M_c}{|c|}$. The conservative remapping consists in computing the masses $\tilde{M}_c$ and the mean values $\tilde{\psi}_c$ for each cell of the rezoned grid. Since the density distribution is unknown on the rezoned grid we have to define it by using an approximation. This approximation should be conservative and sufficiently accurate to limit the numerical diffusion. First, we compute $\psi_c(\boldsymbol{X}) = \psi_c + \boldsymbol{\nabla}\psi_c \cdot (\boldsymbol{X} - \boldsymbol{X_c})$, the piecewise linear monotonic reconstruction for ψ in the cell c following the methodology described in Section 2.3. Then, $\psi_c(\boldsymbol{X})$ could be exactly integrated over the new cells $\tilde{c}$. This approach, which needs to compute the intersection of the Lagrangian grid and the rezoned one, is very expensive and will not be used in the sequel. To minimize the computational cost, we prefer to use an approximate remapping that is an unstructured extension of the simplified face-based donor cell method [8, 14]. The new mass $\tilde{M}_c$ is defined in terms of the old mass M_c plus the mass exchange with the neighboring cells

$$\tilde{M}_c = M_c + \sum_{n,m} \mathcal{F}^c_{n,m} \tag{15}$$

where $\mathcal{F}^c_{n,m}$ represents the flux exchanged by the cell c with its neighbors, through the oriented edge $[n, m]$. To define this flux more precisely, we introduce the oriented quadrilateral $K_{n,m} = \{n, \tilde{n}, \tilde{m}, m\}$, see Figure 5, that is obtained when following the displacement of the nodes n and m to their new locations $\tilde{n}$ and $\tilde{m}$. This quadrilateral is called a fluxing area or a swept region. Its area $\mid K_{n,m} \mid$ is signed and, its sign depends on the ordering of its vertices, for situation shown in Figure 5, $\mid K_{n,m} \mid > 0$. With these notations, we define the flux as follows

$$\mathcal{F}_{n,m} = \int_{K_{n,m}} \psi_{K_{n,m}}(\boldsymbol{X})\,\mathrm{d}\Omega, \tag{16}$$

where

$$\psi_{K_{n,m}}(\boldsymbol{X}) = \begin{cases} \psi_c(\boldsymbol{X}) & \text{if } \mid K_{n,m} \mid \geq 0, \\ \psi_{c_{n,m}}(\boldsymbol{X}) & \text{if } \mid K_{n,m} \mid < 0. \end{cases}$$

Here, $c_{n,m}$ denotes the cell that shared the edge $[n, m]$ with cell c. The computation of the integral in (16) is made using quadrature formula which is exact for linear function. The formula (15) is conservative since $\mathcal{F}^c_{n,m} = -\mathcal{F}^{c_{n,m}}_{n,m}$.

5. NUMERICAL RESULTS

In this section, we address the following three materials instability problem. We consider a shocktube problem filled in with three different ideal gazes. initial conditions and geometry are displayed in the Figure 6. At the
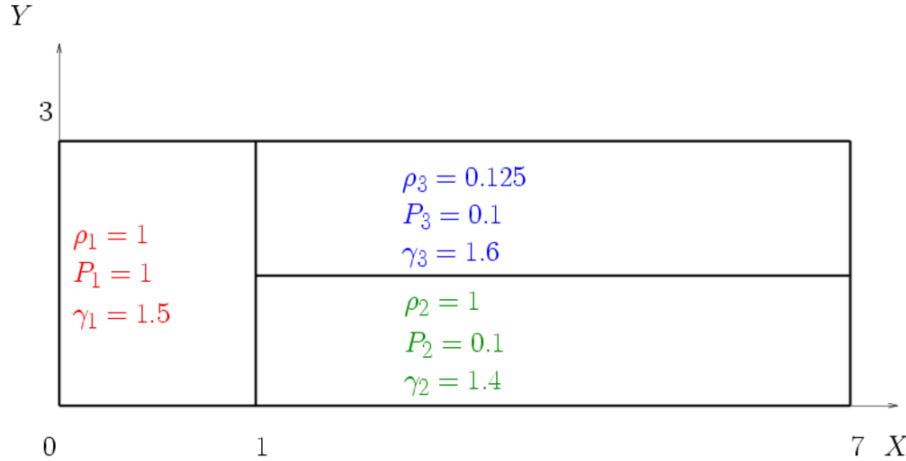

FIGURE 6. Initial conditions for the three materials instability problem.

first time, near the vertical interface located at $X = 1$, the solution consists of a rarefaction moving to the left, and a contact discontinuity and a two shock waves moving to the right. These two shock waves travel with different speeds since the densities on both sides of the horizontal interface are different. This difference in shock wave velocities create a strong shear at the horizontal interface and initiate an instability.

We compute the different solutions on a quadrangular mesh which is composed of a 70×30 square cells until time $t_{\text{end}} = 5$. We use equations of concentration for ALE computations, with iso-pressure and iso-temperature assumptions.

We have displayed in Figure 7 the grids obtained at $t = 4.3$ for a pure Lagrangian computation and an ALE computation. The Lagrangian computation stops at this time due to mesh tangling caused by the high level of vorticity, see Figure 8. For the ALE computation, we perform rezoning and remapping at each time step. The rezoning utilizes only the condition number smoothing. We can observe in Figure 7 (right) that the ALE mesh has a better quality than the Lagrangian. Hence, the ALE computation is more robust than the Lagrangian one. However, the ALE mesh has been over-smoothed and a lot of details of the features of the flow has been lost. We perform another ALE simulation combining condition number and weighted smoothing with parameter $\alpha = 0.5$, as it is described in Section 3.2. We notice that the addition of the weighted smoothing tends to increase the density of the mesh in the vicinity of the shocked regions and improve the accuracy of the computation, see Figure 9.
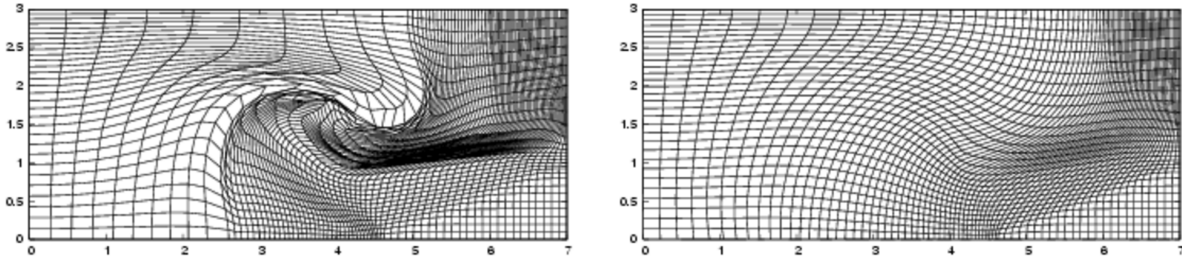
FIGURE 7. Comparison of the Lagrangian (left) and the ALE (right) solutions at $t = 4.3$.
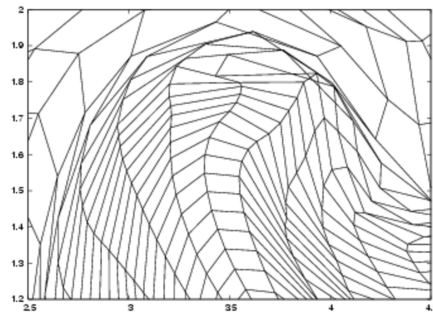

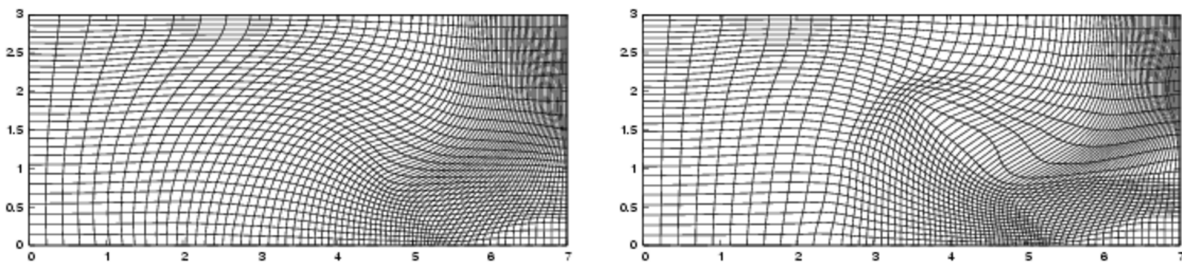FIGURE 8. Zoom on the tangled Lagrangian mesh at $t = 4.3$.

FIGURE 9. Comparison of ALE solution with condition number smoothing (left) and with condition number plus weighted smoothing (right), at $t = 5$.

References

[1] F.L. Adessio, D.E. Carroll, J.K. Dukowicz, F.H. Harlow, J.N. Johnson, B. A. Kashiwa, M. E. Maltrud, H. M. Ruppel. *CAVEAT: A Computer code for fluid dynamics problems with large distortion and internal slip* Report LA-10613-MS, Los Alamos National Laboratory, 1986.

[2] T. Barth. *Numerical Methods for Conservation Laws on Structured and Unstructured Meshes* VKI Lecture Series, 2003.

[3] E. J. Caramana, D. E. Burton, M. J. Shashkov and P. P. Whalen. The construction of compatible hydrodynamics algorithms utilizing conservation of total energy. *Journal of Computional Physics* 1998;**146**: 227–262.

[4] D. A. Field. Laplacian smoothing and Delaunay triangulations. *Comm. Applied Numer. Methods* 1987; **4**.

[5] C. W. Hirt, A. A. Amsden and J. L. Cook. An arbitrary Lagrangian-Eulerian computing method for all flow speeds. *Journal of Computational Physics* 1974;**14**: 227–253.

[6] P. Knupp. Achieving finite element mesh quality via optimization of the Jacobian matrix norm and associated quantities. Part I – a framework for surface mesh optimization. *Int. J. Numer. Meth. Engng.* 2000; **48**: 401–420.

[7] P. Knupp, L. G. Margolin and M. J. Shashkov. Reference Jacobian optimization-based rezone strategies for arbitrary Lagrangian Eulerian methods. *Journal of Computational Physics* 2002; **176**: 93–128.

[8] M. Kucharic, M. J. Shashkov and B. Wendroff An efficient linearity-and-bound-preserving remapping method *Journal of Computional Physics* 2003; **188**: 462–471.

[9] R. Loubère and M. J. Shashkov. A subcell remapping method on staggered polygonal grids for arbitrary-Lagrangian-Eulerian methods. *Journal of Computational Physics* 2005; **209**: 105–138.

[10] H. Luo, J.D. Baum and R. Lhner. On the computation of multi-material flows using ALE formulation. *Journal of Computational Physics* 2004; **194**: 304–328.

[11] P. H. Maire, R. Abgrall, J. Breil and J.Ovadia. A centered Lagrangian scheme for multidimensional compressible flow problems. *SIAM Journal of Scientific Computing* 2007; **29**: 1781–1824.

[12] P. H. Maire and J. Breil. A second order cell-centered Lagrangian scheme for two-dimensional compressible flows problems. *International Journal for Numerical Methods in Fluids* 2007; DOI: 10.1002/fld.1564

[13] P. H. Maire, J. Breil and S. Galera. A cell-centered arbitrary Lagrangian-Eulerian method. *International Journal for Numerical Methods in Fluids* 2007; DOI: 10.1002/fld.1557

[14] L. G. Margolin, M. J. Shashkov. Second-order sign-preserving remapping on general grids. *Journal of Computational Physics* 2003; **184**: 266–298.

[15] Y. Ohtake, A. Belyaev and I. Bogaevski. Mesh regularization and adaptative smoothing. *Computer-Aided Design* 2001; **33**.

[16] G. Taubin. Curve and surface smoothing without shrinkage. *Proc. Int. Conf. on Computer Vision* 1995.

[17] G. Taubin, T. Zhang and G. Golub. Optimal surface smoothing as filter design. *Proc. Fourth European Conf. on Computer Vision* 1996.

[18] P. Vachal, R. V. Garimella and M. J. Shashkov. Untangling of 2D meshes in ALE simulations. *Journal of Computational Physics* 2004; **196**: 627–644.