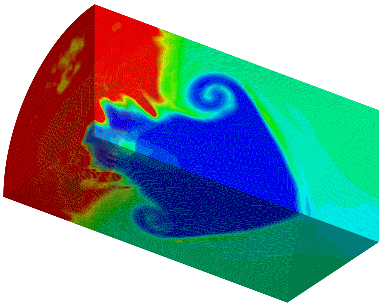
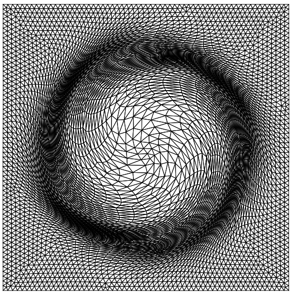
Doctoral School in Environmental Engineering

High Order Direct Arbitrary-Lagrangian-Eulerian (ALE) Finite Volume Schemes for Hyperbolic Systems on Unstructured Meshes

Walter Boscheri





UNIVERSITÁ DEGLI STUDI DI TRENTO
Dipartimento di Ingegneria Civile, Ambientale e Meccanica

2015

Doctoral thesis in **Environmental Engineering, XXVII cycle.**

Department of Civil, Environmental and Mechanical Engineering,
University of Trento.

Academic year **2014/2015**.

Internal supervisor	:	**Prof. Dr.-Ing. Michael Dumbser**
External supervisors	:	**Prof. Dr. Claus-Dieter Munz** *IAG-Stuttgart (Germany)*
	:	**Prof. Dr. Bruno Després** *University of Paris (France)*

University of Trento
Trento, Italy
2015

Ai miei genitori Anna Maria e Roberto

“*dos est magna parentium virtus*”
Orazio

Preface

This thesis has been developed during the three-year Doctoral program of the Doctoral School in Environmental Engineering at the Department of Civil, Environmental and Mechanical (DICAM) Engineering of the University of Trento.

The presented research has been financed by the European Research Council (ERC) under the European Union's Seventh Framework Programme (FP7/2007-2013) with the research project *STiMulUs*, ERC Grant agreement no. 278267.

Most of the work has been carried out at DICAM (Trento) under the supervision of Prof. Dr.-Ing. Michael Dumbser. Furthermore I spent two months (August 2013 - September 2013) at the Department of Physics of the Notre Dame University (South Bend, IN - USA) for a collaboration with Prof. Dinshaw Balsara, while for three months (September 2014 - November 2014) my work took place at the Institute de Mathématiques de Toulouse (Toulouse, France) with Dr. Raphaël Loubère in order to be assigned with the *Doctor Europaeus* label.

Some of the two-dimensional simulations shown in this thesis have been run on the AMD Opteron cluster of the *STiMulUs* project located in Povo (Trento, Italy), while the rest of the numerical results have been collected using the SuperMUC supercomputer of the Leibniz Rechenzentrum (LRZ) in Munich (Germany), for which we gained access under the project "STiMulUs - Lagrangian Space-Time Methods for Multi-Fluid Problems on Unstructured Meshes", proposals no. 2012071312, 2013091889 and 2014112638.

Trento, January 2015

Walter Boscheri

Acknowledgments

I would like to thank Prof. Dumbser for his outstanding supervision as well as for his friendship, that allowed my research to be carried out within excellent working conditions.

Many thanks also to Prof. Balsara and Dr. Loubère for the collaborations done together and to the external supervisors Prof. Munz and Prof. Després, who spent part of their time to read and review this thesis.

Last but not least I warmly thank my future wife, Eloina, who enlightens every day of my life.

Contents

Symbols

c	Speed of sound
d	Number of space dimensions
i	Element index
j	Neighbor element index
k	Node index
M	Maximum degree of the reconstruction
n	Current time level
$n+1$	Future time level
p	Pressure
t	Time
T	Control volume
u	Velocity component in x-direction
v	Velocity component in y-direction
w	Velocity component in z-direction
x	Horizontal direction in the physical system
y	Lateral direction in the physical system
z	Vertical direction in the physical system
β	Linear basis function
γ	Ratio of specific heats
δ_{ij}	Kronecker symbol
Δt	Timestep
ρ	Density
ρE	Total energy per mass unit
ε^*	Internal energy per mass unit
θ	Space-time basis function and test function
ψ	Space basis function
$\boldsymbol{\Psi}$	Integration path
ξ	Horizontal direction in the reference system
η	Lateral direction in the reference system
ζ	Vertical direction in the reference system
τ	Relative time with respect to timelevel t^n

χ_1, χ_2	Face parameters on the element boundary
Ω	Computational domain
$\boldsymbol{B}$	Non-conservative nonlinear flux tensor
$\boldsymbol{F}$	Conservative nonlinear flux tensor
$\boldsymbol{I}$	Identity matrix
$\boldsymbol{n}$	Element boundary normal vector in space
$\tilde{\boldsymbol{n}}$	Element boundary normal vector in space and time
$\mathcal{A}$	Algorithm efficiency
$\mathcal{NS}_{cs}$	Node solver of Cheng and Shu
$\mathcal{NS}_{m}$	Node solver of Maire
$\mathcal{NS}_{b}$	Node solver of Balsara
$\mathcal{N}_i$	Neumann neighborhood of element i
$\mathcal{O}$	Order of the scheme
$\mathcal{V}_k$	Voronoi neighborhood of node k
$\mathcal{V}_i$	Voronoi neighborhood of element i
$\boldsymbol{Q}$	Vector of conserved variables
$\boldsymbol{U}$	Vector of primitive variables
$\boldsymbol{S}$	Algebraic source term
$\boldsymbol{v}$	Vector of the local fluid velocity
$\boldsymbol{V}$	Vector of the local mesh velocity
$\boldsymbol{X}_k$	Vertex physical coordinates in space
$\boldsymbol{x}$	Physical spatial coordinate vector
$\tilde{\boldsymbol{x}}$	Physical space-time coordinate vector
$\boldsymbol{\xi}$	Reference spatial coordinate vector
$\tilde{\boldsymbol{\xi}}$	Reference space-time coordinate vector
$\boldsymbol{q}_h$	High order space-time predictor solution
$\boldsymbol{w}_h$	High order reconstruction polynomial

Abbreviations

ADER	Arbitrary high order scheme using derivatives
ALE	Arbitrary-Lagrangian-Eulerian scheme
CG	Continuous Galerkin method
CFL	Courant-Friedrichs-Levy number
CPU	Central processing unit
DG	Discontinuous Galerkin method
ENO	Essentially non-oscillatory
FV	Finite Volume
GRP	Generalized Riemann problem
HD	Hydrodynamics
IC	Initial condition
LTS	Local time stepping
MHD	Magneto-Hydrodynamics
MOOD	Multi-dimensional optimal order detection
ODE	Ordinary differential equation
PDE	Partial differential equation
TVD	Total variation diminishing
WENO	Weighted essentially non-oscillatory

Abstract

In this work we develop a new class of high order accurate Arbitrary-Lagrangian-Eulerian (ALE) one-step finite volume schemes for the solution of nonlinear systems of conservative and non-conservative hyperbolic partial differential equations. The numerical algorithm is designed for two and three space dimensions, considering *moving* unstructured triangular and tetrahedral meshes, respectively.
As usual for finite volume schemes, data are represented within each control volume by piecewise constant values that evolve in time, hence implying the use of some strategies to improve the order of accuracy of the algorithm. In our approach high order of accuracy in space is obtained by adopting a *WENO reconstruction* technique, which produces piecewise polynomials of higher degree starting from the known cell averages. Such spatial high order accurate reconstruction is then employed to achieve high order of accuracy also in time using an element-local space-time *finite element predictor*, which performs a one-step time discretization. Specifically, we adopt either the continuous Galerkin (CG) predictor, which does not allow discontinuities in time and is suitable for smooth time evolutions, or the discontinuous Galerkin (DG) predictor which can handle stiff source terms that might produce jumps in the local space-time solution. Since we are dealing with moving meshes the elements deform while the solution is evolving in time, hence making the use of a reference system very convenient. Therefore, within the space-time predictor, the physical element is mapped onto a reference element using a high order isoparametric approach, where the space-time basis and test functions are given by the Lagrange interpolation polynomials passing through a predefined set of space-time nodes.
The computational mesh continuously changes its configuration in time, following as closely as possible the flow motion. The entire mesh motion procedure is composed by three main steps, namely the Lagrangian step, the rezoning step and the relaxation step. In order to obtain a continuous mesh configuration at any time level, the mesh motion is evaluated by assigning each node of the computational mesh with a unique velocity vector at each timestep. The *node solver* algorithm preforms the Lagrangian stage, while we rely on a *rezoning algorithm* to improve the mesh quality when the flow motion becomes very complex, hence producing highly deformed computational elements. A so-called *relaxation algorithm* is finally employed to partially recover the optimal Lagrangian accuracy where the computational elements are not distorted too much. We underline that our scheme is supposed to be an ALE algorithm, where the local mesh velocity can be chosen independently from the local fluid

velocity. Once the vertex velocity and thus the new node location has been determined, the old element configuration at time t^n is connected with the new one at time t^{n+1} with *straight edges* to represent the local mesh motion, in order to maintain algorithmic simplicity.
The final *ALE finite volume scheme* is based directly on a space-time conservation formulation of the governing system of hyperbolic balance laws. The nonlinear system is reformulated more compactly using a space-time divergence operator and is then integrated on a moving space-time control volume. We adopt a linear parametrization of the space-time element boundaries and Gaussian quadrature rules of suitable order of accuracy to compute the integrals. In our algorithm either a simple and robust Rusanov-type numerical flux or a more sophisticated and less dissipative Osher-type numerical flux is employed. We apply the new high order direct ALE finite volume schemes to several hyperbolic systems, namely the multidimensional Euler equations of compressible gas dynamics, the ideal classical and relativistic magneto-hydrodynamics (MHD) equations and the non-conservative seven-equation Baer-Nunziato model of compressible multi-phase flows with stiff relaxation source terms. Numerical convergence studies as well as several classical test problems will be shown to assess the accuracy and the robustness of our schemes.
Furthermore we focus on the following issues to improve the algorithm efficiency: the time evolution, the numerical flux computation across element boundaries and the high order WENO reconstruction procedure. First, a *time-accurate local time stepping* (LTS) algorithm for unstructured triangular meshes is derived and presented, where each element can run at its own optimal time step, given by a local CFL stability condition. Then, we propose a new and efficient *quadrature-free* formulation for the flux computation, in which the space-time boundaries of each element are split into simplex sub-elements. This leads to space-time normal vectors as well as Jacobian matrices that are constant within each sub-element, hence allowing the flux integrals to be evaluated on the space-time reference control volume once and for all analytically during a preprocessing step. Finally, we consider the very new *a posteriori MOOD paradigm*, recently proposed for the Eulerian framework, to overcome the expensive WENO approach on moving meshes. The MOOD technique requires the use of only one central reconstruction stencil because the limiting procedure is carried out *a posteriori* instead of *a priori*, as done in the WENO formulation.

1 Introduction

Many real world processes are modeled using time-dependent partial differential equations (PDE), which are based on the conservation of some physical quantities. Therefore these mathematical and physical models are typically addressed as *conservation laws* and they cover a wide range of phenomena, such as environmental and meteorological flows, hydrodynamic and thermodynamic problems, plasma flows as well as the dynamics of many industrial and mechanical processes. In any case the governing equations can generally be solved using either an *Eulerian* or a *Lagrangian* approach. In the first case the fluid flow is observed and computed in a fixed reference system, while in the latter case the reference system is moving *together* with the local fluid velocity. In general any conservation law assumes that the modeled medium is a continuum and describes the evolution of the *conserved quantity* $u(\mathbf{x}, t)$ in the control volume ω, which can be chosen arbitrarily. The conserved quantity depends both on space ($\mathbf{x}$) and time (t) and any change of u, i.e. the time evolution of u, is assumed to be due to some *fluxes* $F(u)$ across the boundary $\partial\omega$ of the control volume and, in some cases, also to a so-called *source term* $S(u)$ that may affect the evolution of u by either increasing or decreasing the conserved quantity. A very general formulation for a conservation law reads

$$\frac{\partial}{\partial t}\int_{\omega} u\, dV + \int_{\partial\omega} F(u)\,\overline{n}\, dS = \int_{\omega} S(u), \tag{1.1}$$

where $\overline{n}$ represents the outward pointing unit normal vector on the boundary $\partial\omega$. The above expression must be valid for any control volume, hence leading to the following partial differential equation:

$$\frac{\partial u}{\partial t} + \nabla \cdot F(u) = S(u), \tag{1.2}$$

where Gauss' theorem has been used to rewrite the boundary integral as the volume integral of the divergence of the fluxes $\nabla \cdot F(u)$.
The quantity u might also be a vector, hence involving more conserved quantities. For instance, fluid dynamics is governed by conservation laws which describe the evolution of three conserved quantities, namely mass, momentum

and total energy. As a consequence we obtain a *system of conservation laws*, whenever the quantity u is given by a vector. In this case a system matrix $\boldsymbol{A}$ can be defined as

$$\boldsymbol{A} = \frac{\partial F}{\partial u}\overline{n} \tag{1.3}$$

and the system is considered *hyperbolic* if for all $\overline{n}$ all eigenvalues of matrix $\boldsymbol{A}$ are real and if a complete set of eigenvectors exists.

This work focuses on the solution of hyperbolic systems of conservation laws of the form (1.2), considering a Lagrangian-like approach, where the control volume $\omega(t)$ is moving and therefore is time-dependent. Specifically, our task is to design high order accurate finite volume schemes for the solution of hyperbolic systems adopting an *Arbitrary-Lagrangian-Eulerian* approach. Section 1.1 provides a general overview of high order numerical methods for the solution of hyperbolic PDEs in the Eulerian framework, while Section 1.2 presents a literature review of the state-of-the-art in the field of Lagrangian numerical schemes. Finally, Section 1.3 provides the introduction to this work.

1.1 High order finite volume methods on fixed grids

The Eulerian approach implies the introduction of nonlinear convective terms in the governing equations because the flow is observed in a fixed reference system, which does not neither change nor move in time. These terms are considered within the flux term $F(u)$ of the conservation law (1.2). A lot of research has been carried out in the past decades in order to solve conservation laws of the form (1.2) numerically, starting from the one-dimensional case. A very famous and widespread approach is given by *Godunov*-type finite volume methods [133,240], where the discrete solution is stored as constant data within each control volume of the computational mesh and is evolved in time by using the integral form of the conservation law (1.1). Since the discrete solution in general exhibits jumps at the element interfaces, the introduction of numerical fluxes across the discontinuities of each cell is necessary. Godunov suggested to obtain these numerical fluxes by solving *Riemann problems* at each interface. Early work regarded the exact solution of the Riemann problem [73,133], that was followed by the development of approximate Riemann solvers, such as the linearized Riemann solver of Roe [200], the HLL and HLLE Riemann solvers [112,139] and the local Lax-Friedrichs (LLF) solver proposed by Rusanov [201], which can be reinterpreted as an HLL-type flux with a particular choice of the signal speeds. While the above-mentioned HLL schemes are very robust, they smear out contact discontinuities. An improvement was made by Einfeldt and

Munz in [113] with the introduction of the HLLEM Riemann solver, where the intermediate state was assumed piecewise linear instead of piecewise constant. Another well-known improvement of the original HLL scheme is due to Toro et al. in [231] with the design of the HLLC Riemann solvers that use an enhanced wave model that is able to capture also the intermediate contact wave. In [190] Osher et al. introduced a class of approximate Riemann solvers based on path integrals, where the paths were obtained by an approximation of the solution of the Riemann problem by rarefaction fans. A simpler and more general version of the Osher flux has recently been forwarded by Dumbser and Toro in [106, 107]. All those one-dimensional Riemann solvers can be used even in two- and three-dimensional problems, where the discontinuities are resolved at each boundary of the control volume along the normal direction.
In order to design *high order accurate* finite volume numerical schemes, a high order reconstruction operator in space is needed as well as a time evolution of the conserved quantities that allows the method to achieve high order of accuracy even in time. Since linear monotone schemes are at most of order one, as stated by the Godunov theorem [134], a first contribution for the improvement of the order of accuracy has been provided by the class of second order accurate TVD schemes, which adopts a linear reconstruction in space and time, like the MUSCL scheme of van Leer [241] and the second order method of Barth and Jespersen on unstructured meshes [30]. Later on nonlinear ENO reconstructions on unstructured grids have been introduced [4, 217] as well as WENO reconstructions [126, 144, 215]. Once the high order spatial reconstruction is available, a suitable time stepping technique has to be used to guarantee the final order of accuracy. Runge-Kutta (RK) methods perform a multi-stage time-integration to evolve the numerical solution from the current time level t^n to the next time level t^{n+1}. The higher is the order of accuracy, the higher is the number of substages which are needed. Furthermore, the reconstruction operator must be recomputed at each substage, hence drastically decreasing the efficiency of the algorithm. For this reason RK methods are at most of order four, because of the so-called Butcher barriers [50], which cause the number of intermediate RK substages to become larger than the formal order of accuracy. In recent years a valid alternative was proposed by Toro et al., who developed the ADER approach [28, 54, 94, 98, 178, 222, 223]. ADER is the abbreviation for "Arbitrary high order schemes using DERivatives" and the basic idea is to use the high order reconstructed states, which are available from the reconstruction operator, to evaluate the numerical fluxes at element interfaces. In this way the initial data for the local Riemann problems occurring at element boundaries are given by high order piecewise polynomials, instead of

piecewise constants as in the original formulation of Godunov [134]. The first ADER algorithms [151, 178, 210, 211, 222, 223, 233, 234] follow the concept of Ben-Artzi and Falcovitz [31] based on the solution of the generalized Riemann problem (GRP) at zone boundaries. The time evolution is carried out by using repeatedly the governing conservation law in differential form to replace time derivatives by space derivatives, which is the so-called Cauchy-Kovalewski or Lax-Wendroff procedure. The idea behind the GRP approach is a temporal Taylor series expansion of the state at the interface. However, problems arise when the solution is discontinuous. Since in general jumps are admitted at element boundaries, conventional homogeneous Riemann problems for the state and all space derivatives have first to be solved at the interface, then the obtained results are plugged into the Cauchy-Kovalewski procedure to obtain high order accurate time derivatives. The resulting ADER schemes are one-step fully discrete and of arbitrary order of accuracy in space and time, and have been successfully used in the framework of both finite volume (FV) and Discontinuous Galerkin (DG) methods, see [102,103,105,210,211]. An efficient quadrature-free approach for the numerical flux integration has been proposed in [103].
The most recent ADER methods [27, 28, 94, 98] evolve the spatially high order accurate reconstruction polynomial locally in time using a weak integral formulation of the conservation law in space-time, hence obtaining space-time accurate representation of the solution within a cell. This most recent version of the ADER schemes is more similar to the original ENO scheme proposed by Harten et al. [138], since it first evolves the data in each element by solving a local Cauchy problem in the small, i.e. without accounting for the neighbor cells, and then solves the interactions at the zone boundaries. The main advantages of this time evolution are: (i) the cumbersome Cauchy-Kovalewski procedure is no more needed, and (ii) the resulting technique can handle very general and different hyperbolic systems of conservation laws. Furthermore stiff sources are also treated properly, as highlighted in [99, 141].

1.2 Lagrangian methods on moving meshes

Any Lagrangian method aims at following the fluid motion as closely as possible, with a computational mesh that is moving with the local fluid velocity. In the Lagrangian description of the fluid the nonlinear convective terms disappear and Lagrangian schemes exhibit virtually no numerical dissipation at contact waves and material interfaces. Therefore the Lagrangian approach allows such

discontinuities to be precisely located and tracked during the computation, achieving a much more accurate resolution of these waves compared to classical Eulerian methods on fixed grids. For this reason a lot of efforts has been made in the last decades in order to develop Lagrangian methods. Already John von Neumann and Richtmyer were working on Lagrangian schemes in the 1950ies [242], using a formulation of the governing equations in primitive variables, which was also used later in [32, 51]. However, most of the modern Lagrangian finite volume schemes use the conservation form of the equations based on the physically conserved quantities like mass, momentum and total energy in order to compute shock waves properly, see e.g. [53,172,182,216]. Lagrangian schemes can be also classified according to the location of the physical variables on the mesh: when all variables are defined on a collocated grid the so-called *cell-centered* approach is adopted [71,171–173,202], while in the *staggered mesh* approach [167,168] the velocity is defined at the cell interfaces and the other variables at the cell center.
Cell-centered Lagrangian Godunov-type schemes of the Roe-type and of the HLL-type for the Euler equations of compressible gas dynamics have first been considered by Munz in [182]. A cell-centered Godunov-type scheme has also been introduced by Carré et al. in [53], who developed a Lagrangian finite volume algorithm on general multidimensional unstructured meshes. The resulting finite volume scheme is node based and compatible with the mesh displacement. In the work of Després et al. [77,78] the physical part of the system of equations is coupled and evolved together with the geometrical part, hence obtaining a weakly hyperbolic system of conservation laws that is solved using a node-based finite volume scheme. Furthermore they presented a cell-centered Lagrangian method [71] that is translation invariant and suitable for curved meshes. In [169–171] Maire proposed first and second order accurate cell-centered Lagrangian schemes in two- and three- space dimensions on general polygonal grids, where the time derivatives of the fluxes are obtained using a node-centered solver that may be considered as a multidimensional extension of the Generalized Riemann problem methodology introduced by Ben-Artzi and Falcovitz [31], Le Floch et al. [46,124] and Titarev and Toro [222,224,227]. Cell-centered discontinuous Galerkin methods for solving the Lagrangian equations of gas dynamics have been considered in [127–129,161]. Since Lagrangian schemes may lead to severe mesh deformation after a finite time, it is necessary to remesh (or at least to rezone) the computational grid from time to time. A very popular approach consists therefore in Lagrangian remesh and remap schemes, such as the family of cell-centered ALE remap algorithms introduced by Shashkov et al. and Maire et al. in [35, 37, 156, 157, 162, 202]. In

[48,125,203,244] purely Lagrangian and Arbitrary-Lagrangian-Eulerian (ALE) numerical schemes with remapping for multi-phase and multi-material flows are discussed. All the Lagrangian schemes listed so far are at most second order accurate in space and time.

Higher order of accuracy in space was first achieved in [65–67,163] by Cheng and Shu, who introduced a third order accurate essentially non-oscillatory (ENO) reconstruction operator into Godunov-type Lagrangian finite volume schemes. High order of accuracy in time was guaranteed either by the use of a Runge-Kutta or by a Lax-Wendroff-type time stepping. The mesh velocity is simply computed as the arithmetic average of the corner-extrapolated values in the cells adjacent to a mesh vertex. Such a node solver algorithm is very simple and general and can be easily applied to different complicated nonlinear systems of hyperbolic PDE in multiple space dimensions. Cheng and Toro [68] also investigated Lagrangian ADER-WENO schemes in one space dimension. In the finite element framework high order Lagrangian schemes have been developed for example by Scovazzi et al. [187,213] and also by Dobrev et al. [82–84], who solved the equations for Lagrangian hydrodynamics using high order curvilinear finite element methods.

In the literature there are also other methods using a Lagrangian approach and these schemes are at least briefly mentioned in the following. For example, also meshless particle schemes, such as the smooth particle hydrodynamics (SPH) method, belong to the category of fully Lagrangian schemes, see e.g. [118–121, 179]. SPH is generally used to follow the fluid motion in very complex deforming domains. Since it is a particle method, no rezoning or remeshing has to be applied. Furthermore, also semi-Lagrangian methods should be mentioned. They are typically adopted to solve transport equations [140, 199]. Although these schemes use a fixed mesh, as in the classical Eulerian approach, the Lagrangian trajectories of the fluid are followed backward in time in order to compute the numerical solution at the the new time level, see for example [42, 59, 62, 145, 160, 196]. There is also the class of Arbitrary-Lagrangian-Eulerian (ALE) methods [64, 89, 116, 117, 142, 194, 216], where the mesh moves with a velocity that does not necessarily have to coincide with the local fluid velocity. This method is often used for fluid-structure interaction (FSI) problems, but it is also used together with Lagrangian remap schemes.

1.3 Towards high order ALE ADER-WENO schemes

The aim of this work is to design a new family of high order accurate Arbitrary-Lagrangian-Eulerian (ALE) one-step ADER-WENO finite volume schemes for the solution of nonlinear systems of conservative and non-conservative hyperbolic partial differential equations.
The work is based on the already existing high order ADER finite volume solver [94,98] mentioned in Section 1.1, which is used here as a starting point for the development of the new Lagrangian algorithms. From Section 1.2 we know that no better than third order accurate non-oscillatory Lagrangian schemes have ever been proposed on unstructured meshes in two and three space dimensions, hence leading to a challenging task that matches the research frontier of numerical methods on moving mesh. Furthermore, the new algorithm emerging from this work will be so *general* that it will be applicable to a wide range of scientific fields, since it is based on a very general formulation of the governing PDE, which many hyperbolic systems can be cast into.
The first contribution to this new class of numerical methods, which will be addressed as *direct ALE ADER-WENO schemes*, has been presented by Dumbser et al. in [108], where the authors proposed the first one-dimensional high order ALE ADER-WENO finite volume schemes for hyperbolic balance laws with stiff source terms. In this case high order of accuracy in time was achieved by using the local space-time Galerkin predictor method introduced in [98, 141] for the Eulerian case, whereas a high order WENO reconstruction algorithm was used to obtain high order of accuracy in space.
Then, this work contains all the contributions for ALE ADER-WENO methods that have been done in the last three years of research. In [39, 96] Boscheri and Dumbser extended the one-dimensional algorithm [108] to unstructured triangular meshes for conservative and non-conservative hyperbolic systems with stiff source terms. In [41] three different node solver techniques have been applied to the Euler equations of compressible gas dynamics as well as to the equations for magnetohydrodynamics and have been compared with each other. The multidimensional HLL Riemann solver presented in [88] for the Eulerian framework on fixed grids has been used as a node solver for the computation of the mesh velocity in [41] and for the computation of the space-time fluxes of a high order Lagrangian-like finite volume scheme in [38]. In the latter reference it has been shown that the use of a multidimensional Riemann solver allows the use of larger time steps in multiple space dimensions and therefore leads to a computationally more efficient scheme compared to a method based on classical one-dimensional Riemann solvers. In [40] the ALE ADER-WENO finite volume

schemes have been applied to conservative and non-conservative hyperbolic systems on unstructured tetrahedral moving meshes, while in [44] Boscheri and Dumbser introduce a quadrature-free formulation for the numerical flux computation in the ALE context. In order to reduce the computational efforts, which is typically higher for Lagrangian schemes than for Eulerian methods, in [43, 93] the first high order time-accurate local time stepping ALE ADER-WENO schemes have been presented in one and two space dimensions, while in [45] the expensive WENO reconstruction procedure has been replaced with the very recently developed MOOD paradigm [69,79,81,165], which requires the use of only one central reconstruction stencil because the limiting procedure is carried out *a posteriori* instead of *a priori*, as done in the WENO formulation.
The rest of the work is structured as follows. In Chapter 2 we describe in detail the new high order ALE ADER-WENO finite volume schemes, considering what has been done in [39, 40, 96]. The algorithm will be presented in a very general way, treating both conservative and non-conservative hyperbolic systems as well as the presence of algebraic source terms which are allowed to be stiff. Next, Chapter 3 focuses on the techniques used to carry out the mesh motion, i.e. the numerical strategies adopted to evaluate the mesh velocity and consequently to compute the new node location. Three different node solvers will be considered, according to [41], and a rezoning technique with a relaxation algorithm will also be detailed. Chapter 4 aims at introducing some modifications of the direct ALE WENO algorithm, presented in Chapter 2, in order to improve the overall algorithm efficiency. Specifically, we present (i) a local time stepping (LTS) algorithm for moving unstructured triangular meshes [43], (ii) the genuinely multidimensional HLL Riemann solvers for the flux computation in the ALE ADER-WENO framework according to [38], (iii) an efficient quadrature-free approach for the numerical flux integration [44] and finally (iv) the MOOD version [45] of our original ALE ADER scheme, which adopts an efficient *a posteriori* limiting technique. In Chapter 5 several hyperbolic systems of conservation laws will be described as well as the associated test cases used to validate our new finite volume schemes. The corresponding numerical results are given in Chapter 6, where we present some of the numerical simulations that have been run during the last three years of our research activity.
For a more detailed discussion about any of the topics illustrated and described within this work, we refer the reader to the above mentioned references. For the sake of generality, the new family of high order direct ALE ADER finite volume schemes presented in this thesis uses an ALE approach, so that the local mesh velocity can in principle be chosen *independently* from the local fluid velocity.

As a consequence the method in general allows a mass flux and even when the mesh velocity is set to be equal to the fluid velocity the proposed scheme is *not* meant to be a pure Lagrangian method *in sensu stricto*. In this sense, our scheme falls into the category of *direct* ALE methods.

2 High Order ALE One-Step ADER-WENO Finite Volume Schemes

In this chapter we provide a detailed description of our numerical method, presenting and analyzing each part of the algorithm.

In Section 2.1 we propose an introduction to the finite volume framework, showing how such approach applies to moving unstructured meshes. Sections 2.2 and 2.3 are devoted to explain how the algorithm can achieve high order of accuracy both in space and time, respectively. Therefore a WENO reconstruction procedure as well as a one-step element-local space-time predictor are fully detailed. For the one-step space-time predictor we take into account not only the case of smooth solutions in Section 2.3.1, but in Section 2.3.2 we also give a description of a local space-time discontinuous Galerkin predictor which is suitable in case of stiff sources and discontinuous time evolutions.

Since we are dealing with moving meshes, the mesh velocity plays an important role and should be evaluated very accurately. For this reason in Section 2.4 we limit us to provide a very simple solution for the computation of the velocity vector for each node of the computational mesh. Instead, we refer the reader to Chapter 3, where we present in detail all the steps which are needed to move the mesh to the next time level.

This allows us to proceed with the description of the high order finite volume schemes in Section 2.5, considering both conservative and non-conservative systems of balance laws.

2.1 Finite volume framework on moving unstructured meshes

In this work we consider nonlinear systems of hyperbolic balance laws which may also contain non-conservative products and stiff source terms. In our approach we rely on a very general formulation of the governing equations which can apply to several hyperbolic systems. This gives our algorithm the possibility to cover a wide range of physical phenomena, namely all the ones that are governed by equations which can be cast into the following form:

$$\frac{\partial \boldsymbol{Q}}{\partial t} + \nabla \cdot \boldsymbol{F}(\boldsymbol{Q}) + \boldsymbol{B}(\boldsymbol{Q}) \cdot \nabla \boldsymbol{Q} = \boldsymbol{S}(\boldsymbol{Q}), \qquad \mathbf{x} \in \Omega \subset \mathbb{R}^d, t \in \mathbb{R}_0^+, \tag{2.1}$$

where $\boldsymbol{Q} = (q_1, q_2, ..., q_\nu)$ denotes the vector of conserved variables, $\boldsymbol{F} = (\boldsymbol{f}, \boldsymbol{g}, \boldsymbol{h})$ is the conservative nonlinear flux tensor, $\boldsymbol{B} = (\boldsymbol{B}_1, \boldsymbol{B}_2, \boldsymbol{B}_3)$ contains the purely non-conservative part of the system written in block-matrix notation and $\boldsymbol{S}(\boldsymbol{Q})$ represents a nonlinear algebraic source term that is allowed to be stiff. We furthermore introduce the abbreviation $\boldsymbol{P} = \boldsymbol{P}(\boldsymbol{Q}, \nabla \boldsymbol{Q}) = \boldsymbol{B}(\boldsymbol{Q}) \cdot \nabla \boldsymbol{Q}$ to make notation easier. The balance law (2.1) is defined in the multidimensional physical computational domain Ω, where $d \in [2, 3]$ denotes the number of space dimension and $\mathbf{x} = (x, y, z)$ is the position vector. In the following we present the algorithm for $d = 3$, since for the two-dimensional case the method can be easily derived setting to zero the third spatial coordinate, i.e. $z = 0$, as well as all its related quantities.
The finite volume approach is based on the integral formulation of the conservation law (2.1), hence providing discrete evolution equations for integral cell averages. As a consequence, data are represented and stored as cell averages which are evolved in time. The main advantage of working within the finite volume framework is that the integral formulation of the governing equations must hold for arbitrary control volumes, hence yielding almost no restrictions in the discretization of the computational domain Ω. In a Lagrangian framework the computational domain $\Omega(t) \subset \mathbb{R}^d$ is time-dependent and is discretized at the current time t^n by a set of non-overlapping control volumes T_i^n that can be either triangles ($d = 2$) or tetrahedra ($d = 3$). N_E denotes the total number of elements contained in the domain and the union of all elements is called the *current mesh configuration* $\mathcal{T}_\Omega^n$ of the domain

$$\mathcal{T}_\Omega^n = \bigcup_{i=1}^{N_E} T_i^n. \tag{2.2}$$

Since we are dealing with a moving computational domain where the mesh configuration continuously changes in time, it is more convenient to map the physical element T_i^n to a reference element T_E via a *local* reference coordinate system $\xi - \eta - \zeta$. The spatial reference element T_E is the unit tetrahedron (or the unit triangle in 2D) shown in Figure 2.1 and is defined by the nodes $\boldsymbol{\xi}_{e,1} = (\xi_{e,1}, \eta_{e,1}, \zeta_{e,1}) = (0, 0, 0)$, $\boldsymbol{\xi}_{e,2} = (\xi_{e,2}, \eta_{e,2}, \zeta_{e,2}) = (1, 0, 0)$, $\boldsymbol{\xi}_{e,3} = (\xi_{e,3}, \eta_{e,3}, \zeta_{e,3}) = (0, 1, 0)$ and $\boldsymbol{\xi}_{e,4} = (\xi_{e,4}, \eta_{e,4}, \zeta_{e,4}) = (0, 0, 1)$, where $\boldsymbol{\xi} = (\xi, \eta, \zeta)$ is the vector of the spatial coordinates in the reference system, while the position vector $\mathbf{x} = (x, y, z)$ is defined in the physical system. Let furthermore $\boldsymbol{X}^n_{k,i} = (X^n_{k,i}, Y^n_{k,i}, Z^n_{k,i})$ be the vector of physical spatial coordinates of the k-th vertex of element T^n_i. Then the linear mapping from T^n_i to T_e is given by

$$\mathbf{x} = \boldsymbol{X}^n_{1,i} + \left(\boldsymbol{X}^n_{2,i} - \boldsymbol{X}^n_{1,i}\right)\xi + \left(\boldsymbol{X}^n_{3,i} - \boldsymbol{X}^n_{1,i}\right)\eta + \left(\boldsymbol{X}^n_{4,i} - \boldsymbol{X}^n_{1,i}\right)\zeta. \tag{2.3}$$

When $d = 2$ the same transformation applies for the coordinates x and y, setting $\zeta = 0$. The vertices of T^n_i are given a connectivity $\mathcal{C}$ with a counter-clockwise convention, as illustrated in Figure 2.1, hence

$$\mathcal{C} = \begin{cases} (1,2,3), & \text{if} \quad d = 2, \\ (1,2,3,4), & \text{if} \quad d = 3. \end{cases} \tag{2.4}$$

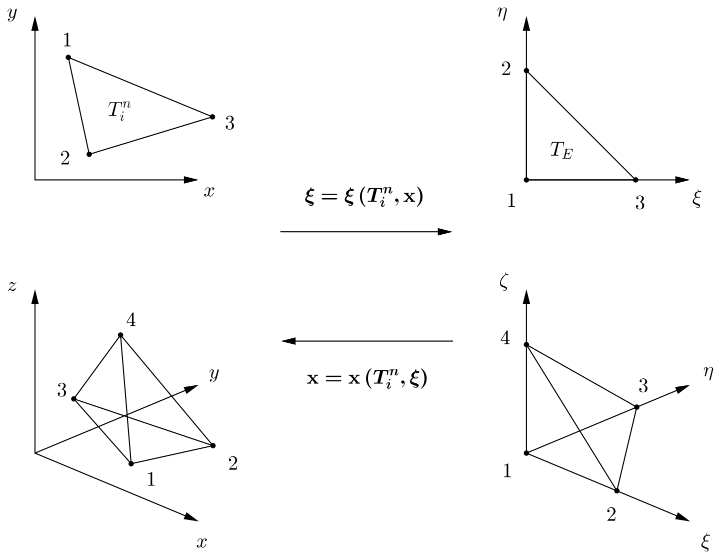
The piecewise constant cell averages, which represent the data that are stored and evolved in time within a finite volume scheme, are defined at each time level t^n within the control volume T^n_i as

$$\boldsymbol{Q}^n_i = \frac{1}{|T^n_i|}\int_{T^n_i} \boldsymbol{Q}(\mathbf{x}, t^n)d\mathbf{x}, \tag{2.5}$$

with $|T^n_i|$ denoting the volume of element T^n_i. The key point of any finite volume schemes is the so-called numerical flux function, which computes the fluxes across the boundaries of the control volume T^n_i. According to Godunov's idea [134], the numerical flux function can be defined by solving local Riemann problems at the interfaces of the control volumes. If we limit us to use only the values given by (2.5) to evaluate the numerical fluxes, we obtain a first order accurate numerical scheme. In order to construct higher order finite volume schemes we need to improve the order of accuracy of the solution employed for the computation of the numerical flux function. In the next Section 2.2 a WENO reconstruction technique is described and used to obtain piecewise higher order polynomials $\mathbf{w}_h(\mathbf{x}, t^n)$ from the known cell averages $\boldsymbol{Q}^n_i$. High order of accuracy in time is achieved later in Section 2.3 by applying a local space-time Galerkin predictor method to the reconstruction polynomials $\mathbf{w}_h(\mathbf{x}, t^n)$.

2.2 Polynomial WENO reconstruction

The WENO reconstruction operator produces piecewise polynomials $\mathbf{w}_h(\mathbf{x}, t^n)$ of degree M. The $\mathbf{w}_h(\mathbf{x}, t^n)$ are computed for each control volume T^n_i from the



y
1
T_i^n
3
2
x
η
2
T_E
1
3
ξ
$\boldsymbol{\xi} = \boldsymbol{\xi}\left(T_i^n, \mathbf{x}\right)$
z
4
3
y
2
1
x
$\mathbf{x} = \mathbf{x}\left(T_i^n, \boldsymbol{\xi}\right)$
ζ
4
3
η
1
2
ξ

Figure 2.1: Spatial mapping from the physical element T_i^n defined with $\mathbf{x}$ to the unit reference element T_E in $\boldsymbol{\xi}$ for triangles (top) and tetrahedra (bottom). Vertices are numbered according to the local connectivity $\mathcal{C}$ given by (2.4).

known cell averages within a so-called *reconstruction stencil* $\mathcal{S}_i^s$, which is composed of an appropriate neighborhood of element T_i^n and contains a prescribed total number n_e of elements which depends on the order M of the polynomial. We do not use the original *pointwise* WENO method first introduced by Shu et al. [144, 146, 249], but we adopt the *polynomial* formulation proposed in [101, 102, 126, 151] and also used in [225, 236], which is relatively simple to code and which allows the scheme to reach very high order of accuracy even on unstructured tetrahedral meshes in three space dimensions.

In [29, 151, 186] it has been shown that the total number of elements n_e must be greater than the smallest number $\mathcal{M}$ needed to reach the formal order of accuracy $M+1$. As suggested in [101, 102] we normally take $n_e = d \cdot \mathcal{M}$, with

$$\mathcal{M} = \prod_{k=1}^{d} \frac{(M+k)}{d!}. \tag{2.6}$$

According to [102] we always use seven ($1 \leq s \leq 7$) and nine ($1 \leq s \leq 9$) reconstruction stencils in two and three space dimensions, respectively. Specifically, $s = 1$ denotes the central stencil, while one half of the remaining stencils are the so-called forward stencils and the others are the backward reconstruction stencils, as depicted in Figures 2.2-2.3. For reconstruction, each element T_i^n and its surrounding elements are first mapped to the reference coordinate system $\xi - \eta - \zeta$ using the mapping (2.3) in order to avoid ill-conditioned reconstruction matrices, see [4]. The three types of stencils (central, forward and backward) are then obtained by a recursive algorithm which adds recursively neighbor elements to the stencil until the prescribed number n_e is reached. Therefore:

- for the central stencil ($s = 1$), we first add the Neumann neighbors of T_i^n (i.e. the direct side neighbors surrounding element T_i^n) to the stencil, and then recursively continue adding the neighbors of these neighbors, until the desired total number of elements in the stencil n_e is reached;

- each of the three forward stencils ($2 \leq s \leq 4$ in 2D and $2 \leq s \leq 5$ in 3D) is filled with elements using the same recursive algorithm, but adding only those elements whose barycenters are located in the corresponding forward sector. On triangular meshes ($d = 2$) the three forward sectors are spanned by a vertex of the triangle and the pair of vectors connecting this vertex with the two vertices of the opposite edge, while for tetrahedra ($d = 3$) the four forward stencils are defined by a vertex k of the tetrahedron T_i^n and the triplet of vectors connecting k to the three vertices of the opposite face;

- the three backward stencils ($5 \leq s \leq 7$ in 2D and $6 \leq s \leq 9$ in 3D) are constructed in the same way as the forward stencils. The associated backward sectors cover the remaining part of $\mathbb{R}^d$ that has not been covered by the forward stencils and are spanned by the negative vectors of the forward stencils and the opposite edge or face barycenter in two and three space dimensions, respectively.

For the central stencil we use a simple Neumann-type neighbor search algorithm that recursively adds direct face neighbors to the stencil, until the desired number n_e is reached. For the remaining one-sided stencils we use a Voronoi-type search algorithm, which fills the stencil starting from the vertex neighborhood of the control volume and then using recursively vertex neighbors of stencil elements. Figures 2.2-2.3 show the stencils used for the WENO reconstruction technique on triangular and tetrahedral meshes, respectively.

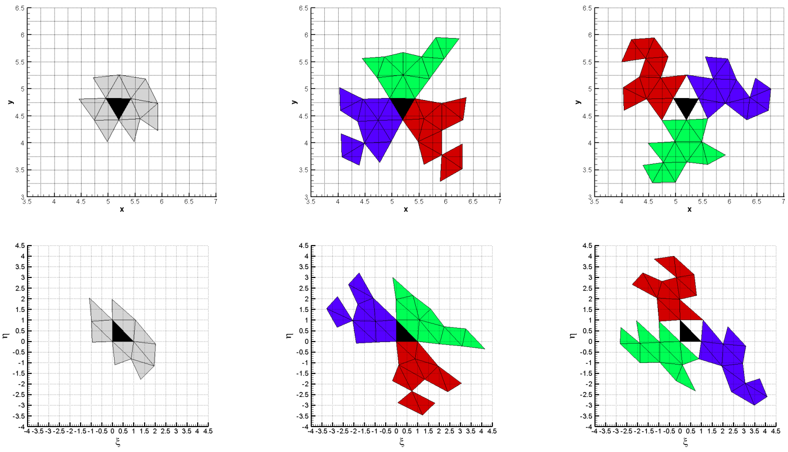


Figure 2.2: Two-dimensional WENO reconstruction stencils in the physical (top row) and in the reference (bottom row) coordinate system for $M = 2$, hence $n_e = 12$: one central stencil (left), three forward stencils (center) and three backward stencils (right).

Once the stencil search procedure has been carried out, each stencil contains a total number of elements n_e that depends on the reconstruction degree M given by (2.6), hence

$$\mathcal{S}_i^s = \bigcup_{j=1}^{n_e} T_{m(j)}^n, \tag{2.7}$$

where $1 \leq j \leq n_e$ is a local index which progressively counts the elements in the stencil number s and $m(j)$ represents a mapping from the local index j to

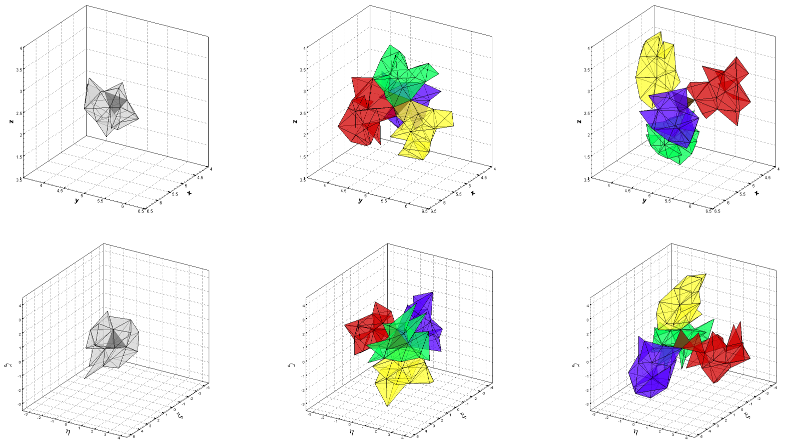


Figure 2.3: Three-dimensional WENO reconstruction stencils in the physical (top row) and in the reference (bottom row) coordinate system for $M = 2$, hence $n_e = 30$: one central stencil (left), four forward stencils (center) and four backward stencils (right).

the global index of the element in $\mathcal{T}_\Omega^n$.

The high order reconstruction polynomial for each candidate stencil $\mathcal{S}_i^s$ for element T_i^n is written in terms of the *orthogonal* Dubiner-type basis functions $\psi_l(\xi, \eta, \zeta)$ [72, 90, 149] on the reference element T_e, i.e.

$$\mathbf{w}_h^s(\mathbf{x}, t^n) = \sum_{l=1}^{\mathcal{M}} \psi_l(\boldsymbol{\xi}) \hat{\mathbf{w}}_{l,i}^{n,s} := \psi_l(\boldsymbol{\xi}) \hat{\mathbf{w}}_{l,i}^{n,s}, \tag{2.8}$$

where the mapping to the reference coordinate system is given by (2.3) and $\hat{\mathbf{w}}_{l,i}^{n,s}$ denote the *unknown* degrees of freedom (expansion coefficients) of the reconstruction polynomial on stencil $\mathcal{S}_i^s$ for element T_i^n at time t^n. In the rest of this manuscript we will use classical tensor index notation based on the Einstein summation convention, which implies summation over two equal indices. For a more details on the space basis functions $\psi_l(\boldsymbol{\xi})$ we refer to Section A.1 of Appendix A.

Integral conservation is required for the reconstruction on each element T_j^n of the stencil $\mathcal{S}_i^s$, yielding

$$\frac{1}{|T_j^n|} \int\limits_{T_j^n} \psi_l(\boldsymbol{\xi}) \hat{\mathbf{w}}_{l,i}^{n,s} d\boldsymbol{x} = \boldsymbol{Q}_j^n, \qquad \forall T_j^n \in \mathcal{S}_i^s. \tag{2.9}$$

Inserting the transformation (2.3) into the above expression (2.9), an analytical integration formula can be obtained that is a function of the physical vertex coordinates $\boldsymbol{X}_{k,j}^n$ of the element. The resulting algebraic expressions of the integrals appearing in (2.9) can be obtained for example at the aid of a symbolic computer algebra system like MAPLE. Up to $M = 3$ we use the aforementioned analytical integration, while for higher reconstruction degrees the integrals in (2.9) are simply evaluated using Gaussian quadrature formulae of suitable order, see [218] for details, since the analytical expressions become too cumbersome. The reconstruction matrix, which is given by the integrals of the linear system (2.9), depends on the geometry of the control volumes in stencil $\mathcal{S}_i^s$. Therefore, since in the Lagrangian framework the mesh is moving in time, the reconstruction matrix can *not* be inverted and stored once and for all during a pre-processing stage, like in the Eulerian case. As a consequence, we assemble and solve the small reconstruction system (2.9) for each element T_i^n directly at the beginning of each time step t^n using optimized LAPACK subroutines. This makes the ALE WENO reconstruction *computationally more expensive* but at the same time also *much less memory consuming* compared to the original Eulerian WENO algorithm presented in [101, 102], since no reconstruction matrices are stored.
While the mesh is moving in time, we always assume that the connectivity of the mesh and therefore also the topology of each reconstruction stencil remains constant in time. Therefore, the definition of the stencils $\mathcal{S}_i^s$ does *not* need to be updated during the simulation. This is a very important simplification, since the stencil search may be quite time consuming in multiple space dimensions on unstructured meshes.
Since each stencil $\mathcal{S}_i^s$ is filled with a total number of $n_e = d \cdot \mathcal{M}$ elements, system (2.9) results in an overdetermined linear system that has to be solved properly by either using a constrained least-squares technique (LSQ), see [102], or a more sophisticated singular value decomposition (SVD) algorithm. The use of the reference coordinate system ensures the matrix of the linear system (2.9) to be reasonably well conditioned.
As stated by the Godunov theorem [134], linear monotone schemes are at most of order one and if the scheme is required to be higher order accurate and

non-oscillatory, it must be *nonlinear*. Therefore a nonlinear formulation has to be used for the final WENO reconstruction polynomial. We first measure the *smoothness* of each reconstruction polynomial obtained on stencil $\mathcal{S}_i^s$ by a so-called oscillation indicator $\boldsymbol{\sigma}_s$ [146],

$$\boldsymbol{\sigma}_s = \Sigma_{lm} \hat{\mathbf{w}}_{l,i}^{n,s} \hat{w}_{m,i}^{n,s}, \tag{2.10}$$

which is computed on the reference element using the (universal) oscillation indicator matrix Σ_{lm}, which, according to [102], is given by

$$\Sigma_{lm} = \sum_{1 \leq \alpha+\beta+\gamma \leq M} \int_{T_e} \frac{\partial^{\alpha+\beta+\gamma} \psi_l(\xi,\eta,\zeta)}{\partial \xi^\alpha \partial \eta^\beta \partial \zeta^\gamma} \cdot \frac{\partial^{\alpha+\beta+\gamma} \psi_m(\xi,\eta,\zeta)}{\partial \xi^\alpha \partial \eta^\beta \partial \zeta^\gamma} d\xi d\eta d\zeta. \tag{2.11}$$

In two space dimensions the above expression holds with $\zeta = 0$ and $\gamma = 0$. The nonlinearity is then introduced into the scheme by the WENO weights $\boldsymbol{\omega}_s$, which read

$$\tilde{\boldsymbol{\omega}}_s = \frac{\lambda_s}{(\boldsymbol{\sigma}_s + \epsilon)^r}, \qquad \boldsymbol{\omega}_s = \frac{\tilde{\boldsymbol{\omega}}_s}{\sum_k \tilde{\boldsymbol{\omega}}_k}, \tag{2.12}$$

with the parameters $r = 8$ and $\epsilon = 10^{-14}$. According to [102] the linear weights are chosen as $\lambda_1 = 10^5$ for the central stencil and $\lambda_s = 1$ for the remaining one-sided stencils. Formula (2.12) is intended to be read componentwise. For a WENO reconstruction based on characteristic variables see [101]. A weighted nonlinear combination of the reconstruction polynomials obtained on each candidate stencil $\mathcal{S}_i^s$ yields the final WENO reconstruction polynomial and its coefficients:

$$\mathbf{w}_h(\mathbf{x}, t^n) = \sum_{l=1}^{\mathcal{M}} \psi_l(\boldsymbol{\xi}) \hat{\mathbf{w}}_{l,i}^n, \qquad \text{with} \qquad \hat{\mathbf{w}}_{l,i}^n = \sum_s \boldsymbol{\omega}_s \hat{\mathbf{w}}_{l,i}^{n,s}. \tag{2.13}$$

Within this work we may also refer to an $\mathcal{M}$-th order accurate reconstruction polynomial with the notation $\mathbb{P}_M$, meaning that the reconstruction procedure has been carried out using polynomial of degree M.

2.3 Local space-time Galerkin predictor on moving curved meshes

The reconstructed polynomials $\mathbf{w}_h(\mathbf{x}, t^n)$ computed at the current time t^n are then *evolved* during one time step, i.e. up to time t^{n+1}, *locally* within each element $T_i(t)$ without requiring any neighbor information. As a result, one

obtains piecewise space-time polynomials of degree M, denoted by $\boldsymbol{q}_h(\mathbf{x}, t)$. This allows the scheme to achieve also high order of accuracy in time. Such an element-local time-evolution procedure has also been used within the MUSCL scheme of van Leer [241] and the original ENO scheme of Harten et al. [138], who called this element-local predictor with initial data $\mathbf{w}_h(\mathbf{x}, t^n)$ the solution of a Cauchy problem *in the small*, since no information from neighbor elements is used. The coupling with the neighbor elements occurs only later in the final one-step finite volume scheme (see Section 2.5). While the original ENO scheme of Harten et al. uses a higher order Taylor series in time together with the *strong* differential form of the PDE to substitute time–derivatives with space derivatives (the so–called Cauchy–Kovalewski or Lax–Wendroff procedure [159]), here a *weak* formulation of the governing PDE (2.1) in space-time is derived (see Eqn. 2.28 below). The resulting method does not require the computation of higher order derivatives, but just pointwise evaluations of the fluxes, source terms and non-conservative products appearing in the PDE. This approach has first been developed for the Eulerian framework on fixed grids in [95, 98, 99, 141] and here we extend it to moving unstructured meshes in multiple space dimensions.
Let $\mathbf{x} = (x, y, z)$ and $\boldsymbol{\xi} = (\xi, \eta, \zeta)$ be the spatial coordinate vectors defined in the physical and in the reference system, respectively, and let $\tilde{\mathbf{x}} = (x, y, z, t)$ and $\tilde{\boldsymbol{\xi}} = (\xi, \eta, \zeta, \tau)$ be the corresponding space-time coordinate vectors. Let furthermore $\theta_l = \theta_l(\tilde{\boldsymbol{\xi}}) = \theta_l(\xi, \eta, \zeta, \tau)$ be a space-time basis function defined by the Lagrange interpolation polynomials passing through a set of space-time nodes $\tilde{\boldsymbol{\xi}}_m = (\xi_m, \eta_m, \zeta_m, \tau_m)$. For the Discontinuous Galerkin (DG) predictor, illustrated in Section 2.3.2, the space-time nodes are defined by the tensor product of the spatial nodes of classical conforming high order finite elements and the Gauss-Legendre quadrature points in time, while in the Continuous Galerkin (CG) approach the coordinates of the space-time points are chosen according to [94]. The two-dimensional reference and physical space-time element configuration as well as the associated space-time nodes for the case $M = 2$ are depicted in Figures 2.4 and 2.5 for the CG and the DG predictor algorithm, respectively. More details are also given in Section A.2 of Appendix A.
Since the Lagrange interpolation polynomials define a *nodal* basis, the functions θ_l satisfy the following interpolation property:

$$\theta_l(\tilde{\boldsymbol{\xi}}_m) = \delta_{lm}, \tag{2.14}$$

where δ_{lm} denotes the usual Kronecker symbol. Following [95] the local solution $\boldsymbol{q}_h$, the fluxes $\boldsymbol{F}_h = (\boldsymbol{f}_h, \boldsymbol{g}_h, \boldsymbol{h}_h)$, the source term $\boldsymbol{S}_h$ and the non-conservative product $\boldsymbol{P}_h = \boldsymbol{B}(\boldsymbol{q}_h) \cdot \nabla \boldsymbol{q}_h$ are approximated within the space-time element $T_i(t) \times [t^n; t^{n+1}]$ with

$$\begin{aligned} \boldsymbol{q}_h &= \boldsymbol{q}_h(\tilde{\boldsymbol{\xi}}) = \theta_l(\tilde{\boldsymbol{\xi}})\,\widehat{\boldsymbol{q}}_{l,i}, \qquad & \boldsymbol{F}_h = \boldsymbol{F}_h(\tilde{\boldsymbol{\xi}}) = \theta_l(\tilde{\boldsymbol{\xi}})\,\widehat{\boldsymbol{F}}_{l,i}, \\ \boldsymbol{S}_h &= \boldsymbol{S}_h(\tilde{\boldsymbol{\xi}}) = \theta_l(\tilde{\boldsymbol{\xi}})\,\widehat{\boldsymbol{S}}_{l,i}, \qquad & \boldsymbol{P}_h = \boldsymbol{P}_h(\tilde{\boldsymbol{\xi}}) = \theta_l(\tilde{\boldsymbol{\xi}})\,\widehat{\boldsymbol{P}}_{l,i}. \end{aligned} \tag{2.15}$$

Because of the interpolation property (2.14) we evaluate the degrees of freedom for $\boldsymbol{F}_h$, $\boldsymbol{S}_h$ and $\boldsymbol{P}_h$ in a *pointwise* manner from $\boldsymbol{q}_h$ as

$$\widehat{\boldsymbol{F}}_{l,i} = \boldsymbol{F}(\widehat{\boldsymbol{q}}_{l,i}), \quad \widehat{\boldsymbol{S}}_{l,i} = \boldsymbol{S}(\widehat{\boldsymbol{q}}_{l,i}), \quad \widehat{\boldsymbol{P}}_{l,i} = \boldsymbol{P}(\widehat{\boldsymbol{q}}_{l,i}, \nabla\widehat{\boldsymbol{q}}_{l,i}), \quad \nabla\widehat{\boldsymbol{q}}_{l,i} = \nabla\theta_m(\tilde{\boldsymbol{\xi}}_l)\widehat{\boldsymbol{q}}_{m,i}. \tag{2.16}$$

The degrees of freedom $\nabla\widehat{\boldsymbol{q}}_{l,i}$ represent the gradient of $\boldsymbol{q}_h$ in node $\tilde{\boldsymbol{\xi}}_l$.
An *isoparametric* approach is used, where the mapping between the physical space-time coordinate vector $\tilde{\boldsymbol{x}}$ and the reference space-time coordinate vector $\tilde{\boldsymbol{\xi}}$ is represented by the *same* basis functions θ_l used for the discrete solution $\boldsymbol{q}_h$ itself. Therefore

$$\mathbf{x}(\tilde{\boldsymbol{\xi}}) = \theta_l(\tilde{\boldsymbol{\xi}})\,\widehat{\mathbf{x}}_{l,i}, \qquad t(\tilde{\boldsymbol{\xi}}) = \theta_l(\tilde{\boldsymbol{\xi}})\,\widehat{t}_l, \tag{2.17}$$

where $\widehat{\mathbf{x}}_{l,i} = (\widehat{x}_{l,i}, \widehat{y}_{l,i}, \widehat{z}_{l,i})$ are the degrees of freedom of the spatial physical coordinates of the moving space-time control volume, which are unknown, while $\widehat{t}_l$ denote the *known* degrees of freedom of the physical time at each space-time node $\tilde{\boldsymbol{x}}_{l,i} = (\widehat{x}_{l,i}, \widehat{y}_{l,i}, \widehat{z}_{l,i}, \widehat{t}_l)$. The mapping in time is linear and simply reads

$$t = t_n + \tau\,\Delta t, \qquad \tau = \frac{t - t^n}{\Delta t}, \qquad \Rightarrow \qquad \widehat{t}_l = t_n + \tau_l\,\Delta t, \tag{2.18}$$

where t^n represents the current time and Δt is the current time step, which is computed under a classical Courant-Friedrichs-Levy number (CFL) stability condition, i.e.

$$\Delta t = \text{CFL} \min_{T_i^n} \frac{d_i}{|\lambda_{\max,i}|}, \qquad \forall T_i^n \in \Omega^n, \tag{2.19}$$

with d_i denoting the insphere or incircle diameter of element T_i^n and $|\lambda_{\max,i}|$ corresponding to the maximum absolute value of the eigenvalues computed from the solution $\boldsymbol{Q}_i^n$ in T_i^n. In multiple space dimensions, the CFL condition must satisfy the inequality CFL $\leq \frac{1}{d}$ if one-dimensional Riemann solvers are used, see [229].

The Jacobian of the transformation from the physical space-time element to the reference space-time element reads

$$J_{st} = \frac{\partial \tilde{\mathbf{x}}}{\partial \tilde{\boldsymbol{\xi}}} = \begin{pmatrix} x_\xi & x_\eta & x_\zeta & x_\tau \\ y_\xi & y_\eta & y_\zeta & y_\tau \\ z_\xi & z_\eta & z_\zeta & z_\tau \\ 0 & 0 & 0 & \Delta t \end{pmatrix} \tag{2.20}$$

and its inverse is given by

$$J_{st}^{-1} = \frac{\partial \tilde{\boldsymbol{\xi}}}{\partial \tilde{\mathbf{x}}} = \begin{pmatrix} \xi_x & \xi_y & \xi_z & \xi_t \\ \eta_x & \eta_y & \eta_z & \eta_t \\ \zeta_x & \zeta_y & \zeta_z & \zeta_t \\ 0 & 0 & 0 & \frac{1}{\Delta t} \end{pmatrix}. \tag{2.21}$$

We point out that in the Jacobian matrix $t_\xi = t_\eta = t_\zeta = 0$ and $t_\tau = \Delta t$, as can be easily derived from the time mapping (2.18).
In the following we introduce the notation adopted for the nabla operator ∇ in the reference space $\boldsymbol{\xi} = (\xi, \eta, \zeta)$ and in the physical space $\mathbf{x} = (x, y, z)$:

$$\nabla_{\boldsymbol{\xi}} = \begin{pmatrix} \frac{\partial}{\partial \xi} \\ \frac{\partial}{\partial \eta} \\ \frac{\partial}{\partial \zeta} \end{pmatrix}, \quad \nabla = \begin{pmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{pmatrix} = \begin{pmatrix} \xi_x & \eta_x & \zeta_x \\ \xi_y & \eta_y & \zeta_y \\ \xi_z & \eta_z & \zeta_z \end{pmatrix} \begin{pmatrix} \frac{\partial}{\partial \xi} \\ \frac{\partial}{\partial \eta} \\ \frac{\partial}{\partial \zeta} \end{pmatrix} = \left(\frac{\partial \boldsymbol{\xi}}{\partial \mathbf{x}} \right)^T \nabla_{\boldsymbol{\xi}}. \tag{2.22}$$

Furthermore let us introduce the two integral operators

$$\begin{aligned} [f,g]^\tau &= \int\limits_{T_e} f(\xi,\eta,\zeta,\tau) g(\xi,\eta,\zeta,\tau) d\xi d\eta d\zeta, \\ \langle f,g \rangle &= \int\limits_0^1 \int\limits_{T_e} f(\xi,\eta,\zeta,\tau) g(\xi,\eta,\zeta,\tau) d\xi d\eta d\zeta d\tau, \end{aligned} \tag{2.23}$$

that denote the scalar products of two functions f and g over the spatial reference element T_E at time τ and over the space-time reference element $T_E \times [0,1]$, respectively.
The governing PDE (2.1) is then reformulated in the reference coordinate system (ξ, η, ζ) using the inverse of the associated Jacobian matrix (2.21) with

$\tau_x = \tau_y = 0$ and $\tau_t = \frac{1}{\Delta t}$ according to (2.18) and adopting the gradient notation illustrated in (2.22) above:

$$\frac{\partial \boldsymbol{Q}}{\partial \tau} + \Delta t \left[\frac{\partial \boldsymbol{Q}}{\partial \boldsymbol{\xi}} \cdot \frac{\partial \boldsymbol{\xi}}{\partial t} + \left(\frac{\partial \boldsymbol{\xi}}{\partial \mathbf{x}} \right)^T \nabla_{\boldsymbol{\xi}} \cdot \boldsymbol{F} + \boldsymbol{B}(\boldsymbol{Q}) \cdot \left(\frac{\partial \boldsymbol{\xi}}{\partial \mathbf{x}} \right)^T \nabla_{\boldsymbol{\xi}} \boldsymbol{Q} \right] = \Delta t \boldsymbol{S}(\boldsymbol{Q}). \tag{2.24}$$

Note that the Lagrangian nature of the scheme, i.e. the moving space–time control volume, leads to the term $\frac{\partial \boldsymbol{Q}}{\partial \boldsymbol{\xi}} \cdot \frac{\partial \boldsymbol{\xi}}{\partial t}$, which is not present in the Eulerian case introduced in [95]. By introducing the following abbreviation

$$\boldsymbol{H} = \frac{\partial \boldsymbol{Q}}{\partial \boldsymbol{\xi}} \cdot \frac{\partial \boldsymbol{\xi}}{\partial t} + \left(\frac{\partial \boldsymbol{\xi}}{\partial \mathbf{x}} \right)^T \nabla_{\boldsymbol{\xi}} \cdot \boldsymbol{F} + \boldsymbol{B}(\boldsymbol{Q}) \cdot \left(\frac{\partial \boldsymbol{\xi}}{\partial \mathbf{x}} \right)^T \nabla_{\boldsymbol{\xi}} \boldsymbol{Q}, \tag{2.25}$$

Eqn. (2.24) simplifies to

$$\frac{\partial \boldsymbol{Q}}{\partial \tau} + \Delta t \boldsymbol{H} = \Delta t \boldsymbol{S}(\boldsymbol{Q}). \tag{2.26}$$

The numerical approximation of $\boldsymbol{H}$ is computed by the same isoparametric approach used in (2.15) for the solution and the flux representation, i.e.

$$\boldsymbol{H}_h = \theta_l(\tilde{\boldsymbol{\xi}})\, \widehat{\boldsymbol{H}}_{l,i}. \tag{2.27}$$

Inserting (2.15) and (2.27) into (2.24), then multiplying Eqn. (2.24) with the space-time test functions $\theta_k(\boldsymbol{\xi})$ and integrating the resulting equation over the space-time reference element $T_E \times [0,1]$, one obtains a *weak formulation* of the governing PDE (2.1):

$$\left\langle \theta_k, \frac{\partial \theta_l}{\partial \tau} \right\rangle \widehat{\boldsymbol{q}}_{l,i} = \left\langle \theta_k, \theta_l \right\rangle \Delta t \left(\widehat{\boldsymbol{S}}_{l,i} - \widehat{\boldsymbol{H}}_{l,i} \right). \tag{2.28}$$

Since the mesh is moving, we also have to evolve in time the geometry of the space-time control volume, i.e. the vertex coordinates of element T_i^n, together with the predictor solution $\boldsymbol{q}_h(\mathbf{x}, t)$. The mesh motion is simply described by the ODE system

$$\frac{d\mathbf{x}}{dt} = \boldsymbol{V}(\boldsymbol{Q}, \mathbf{x}, t), \tag{2.29}$$

with $\boldsymbol{V} = \boldsymbol{V}(\boldsymbol{Q}, \mathbf{x}, t)$ denoting the local mesh velocity. In this work we are developing an *Arbitrary-Lagrangian-Eulerian* (ALE) method, which allows the mesh velocity to be chosen independently from the local fluid velocity, so that the scheme may reduce either to a pure Eulerian approach in the case where

$\boldsymbol{V} = 0$ or to a more Lagrangian-type algorithm if $\boldsymbol{V}$ coincides with the local fluid velocity $\boldsymbol{v}$. Any other choice for the mesh velocity is possible. The velocity inside element $T_i(t)$ is also expressed in terms of the space-time basis functions θ_l as

$$\boldsymbol{V}_h = \theta_l(\boldsymbol{\xi}, \tau)\widehat{\boldsymbol{V}}_{l,i}, \tag{2.30}$$

with $\widehat{\boldsymbol{V}}_{l,i} = \boldsymbol{V}(\hat{\boldsymbol{q}}_{l,i}, \hat{\mathbf{x}}_{l,i}, \hat{t}_l)$.
The weak formulation (2.28), which gives the local evolution of the solution, and the ODE system (2.29), which governs the element motion, constitute a *coupled* set of equations that has to be solved *simultaneously* with an iterative procedure, until the residuals of the predicted solution $\hat{\boldsymbol{q}}_{l,i}$ and the new vertex position $\hat{\mathbf{x}}_{l,i}$ at iteration r are less than a prescribed tolerance, typically set to 10^{-12}. All the details regarding the iterative algorithm will be given later in Sections 2.3.1 and 2.3.2. Therefore the element-local predictor strategy on moving meshes can be summarized by the following steps:

- first we compute the local mesh velocity with (2.30), usually by choosing the local fluid velocity, hence obtaining $\widehat{\boldsymbol{V}}_{l,i} = \boldsymbol{V}(\hat{\boldsymbol{q}}_{l,i}, \hat{\mathbf{x}}_{l,i}, \hat{t}_l)$;
- knowing the mesh velocity, the geometry is updated *locally* within the predictor stage, i.e. obtaining the element-local space-time coordinates $\hat{\mathbf{x}}_{l,i}$;
- the Jacobian matrix and its inverse are then evaluated by using (2.20)-(2.21);
- finally we compute the term $\boldsymbol{H}$ according to (2.24) and the new solution is evolved according to (2.28).

Once we have carried out the above procedure for all the elements of the computational domain, we end up with an *element-local predictor* for the numerical solution $\boldsymbol{q}_h$, for the fluxes $\boldsymbol{F}_h = (\boldsymbol{f}_h, \boldsymbol{g}_h, \boldsymbol{h}_h)$, for the source term $\boldsymbol{S}_h$ and also for the mesh velocity $\boldsymbol{V}_h$.

2.3.1 Local Continuous Galerkin (CG) predictor

The local CG predictor does not allow the numerical solution to be discontinuous in time within the local time evolution, therefore the weak formulation of the governing PDE can be taken in its current form given by (2.28) and solved. To ease the notation, expression (2.28) is shortened by adopting a more compact matrix notation as

$$\boldsymbol{K}_\tau \widehat{\boldsymbol{q}}_{l,i} = \Delta t \boldsymbol{M} \left(\widehat{\boldsymbol{S}}_{l,i} - \widehat{\boldsymbol{H}}_{l,i} \right), \tag{2.31}$$

where the following definitions hold:

$$\boldsymbol{K}_\tau = \left\langle \theta_k, \frac{\partial \theta_l}{\partial \tau} \right\rangle \qquad \text{and} \qquad \boldsymbol{M} = \langle \theta_k, \theta_l \rangle \, . \tag{2.32}$$

The unknown degrees of freedom for $\tau > 0$ of the numerical solution $\widehat{\boldsymbol{q}}_{l,i}$ are addressed with $\widehat{\boldsymbol{q}}^1_{l,i}$, while $\widehat{\boldsymbol{q}}^0_{l,i}$ denote the degrees of freedom which are known from the initial condition $\boldsymbol{w}_h$. Therefore they are moved onto the right hand side of (2.31) by setting the corresponding degrees of freedom of $\boldsymbol{w}_h$ to the known values, like a standard Dirichlet boundary condition in the continuous finite element framework. The resulting nonlinear algebraic equation system (2.31) is solved by an iterative procedure, according to [95], which reads

$$\boldsymbol{K}_\tau \widehat{\boldsymbol{q}}^{r+1}_{l,i} = \Delta t \boldsymbol{M} \left(\widehat{\boldsymbol{S}}_{l,i} - \widehat{\boldsymbol{H}}_{l,i} \right)^r, \tag{2.33}$$

where the superscript r denotes the iteration number. For an efficient initial guess ($r = 0$) based on a second order MUSCL-type scheme we refer the reader to [141], otherwise one can also simply use the reconstruction polynomial $\boldsymbol{w}_h$ at the initial time level t^n.

The local space-time Galerkin method is used again to solve the system (2.29), hence

$$\left\langle \theta_k, \frac{\partial \theta_l}{\partial \tau} \right\rangle \widehat{\mathbf{x}}_{l,i} = \Delta t \left\langle \theta_k, \theta_l \right\rangle \widehat{\boldsymbol{V}}_{l,i}, \tag{2.34}$$

which leads to the following iteration scheme for the unknown coordinate vector $\widehat{\mathbf{x}}_l = (x_l, y_l, z_l)$:

$$\boldsymbol{K}_\tau \widehat{\mathbf{x}}^{r+1}_{l,i} = \Delta t \boldsymbol{M} \widehat{\boldsymbol{V}}^r_{l,i}. \tag{2.35}$$

The initial condition of the ODE system is given by the nodal degrees of freedom $\widehat{\mathbf{x}}_l$ at relative time $\tau = 0$, which are known from the geometry of element T^n_i at time t^n. The iterative procedure described above stops when the residuals of (2.33) and (2.35) are less than a prescribed tolerance.

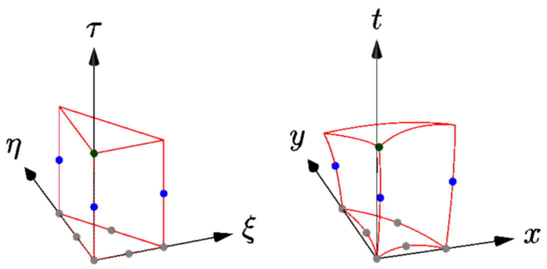


Figure 2.4: Iso-parametric mapping of the space-time reference element (left) to the physical space-time element (right) used within the local space-time Continuous Galerkin (CG) predictor for a triangular control volume.

2.3.2 Local Discontinuous Galerkin (DG) predictor

The local DG predictor has been designed for the treatment of *stiff source terms* [98, 109, 141] in the governing equations (2.1), which might lead to a local time discontinuity of the solution. Therefore the term on the left hand side of the weak formulation (2.28) is integrated by parts in time, which also allows to introduce the initial condition of the local Cauchy problem in a weak form as follows:

$$\begin{aligned} \left[\theta_k(\boldsymbol{\xi},1),\theta_l(\boldsymbol{\xi},1)\right]^1 \widehat{\mathbf{q}}_{l,i} \quad & - \left\langle \frac{\partial \theta_k}{\partial \tau}, \theta_l \right\rangle \widehat{\mathbf{q}}_{l,i} = \\ \left[\theta_k(\boldsymbol{\xi},0),\psi_l(\boldsymbol{\xi})\right]^0 \hat{\mathbf{w}}^n_{l,i} \quad & + \left\langle \theta_k,\theta_l \right\rangle \Delta t \left(\widehat{\mathbf{S}}_{l,i} - \widehat{\boldsymbol{H}}_{l,i} \right). \end{aligned} \tag{2.36}$$

Adopting the following matrix-vector notation

$$\boldsymbol{K}_1 = \left[\theta_k(\boldsymbol{\xi},1),\theta_l(\boldsymbol{\xi},1)\right]^1 - \left\langle \frac{\partial \theta_k}{\partial \tau}, \theta_l \right\rangle, \quad \boldsymbol{F}_0 = \left[\theta_k(\boldsymbol{\xi},0),\psi_l(\boldsymbol{\xi})\right], \quad \boldsymbol{M} = \left\langle \theta_k,\theta_l \right\rangle, \tag{2.37}$$

the system (2.36) is reformulated as

$$\boldsymbol{K}_1 \widehat{\mathbf{q}}_{l,i} = \boldsymbol{F}_0 \hat{\mathbf{w}}^n_{l,i} + \Delta t \boldsymbol{M} \left(\widehat{\mathbf{S}}_{l,i} - \widehat{\boldsymbol{H}}_{l,i} \right). \tag{2.38}$$

Eqn. (2.38) constitutes an element-local nonlinear algebraic equation system for the unknown space-time expansion coefficients $\widehat{\boldsymbol{q}}_{l,i}$ which can be solved using an iterative scheme as

$$\widehat{\boldsymbol{q}}_{l,i}^{r+1} - \Delta t \boldsymbol{K}_1^{-1} \boldsymbol{M}\, \widehat{\mathbf{S}}_{l,i}^{r+1} = \boldsymbol{K}_1^{-1} \left(\boldsymbol{F}_0 \hat{\mathbf{w}}_{l,i}^{n} - \Delta t \boldsymbol{M} \widehat{\boldsymbol{H}}_{l,i}^{r} \right), \tag{2.39}$$

where r denotes the iteration number. In case of stiff algebraic source terms, the discretization of $\boldsymbol{S}$ must be *implicit*, see [98, 108, 109, 141]. For an efficient initial guess of this iterative procedure we refer the reader to [141].
The system (2.29), which governs the local mesh motion, can be conveniently solved for the unknown coordinate vector $\widehat{\mathbf{x}}_l = (x_l, y_l, z_l)$ using the same approach, hence

$$\boldsymbol{K}_1 \widehat{\mathbf{x}}_{l,i} = [\theta_k(\boldsymbol{\xi}, 0), \mathbf{x}(\boldsymbol{\xi}, t^n)]^0 + \Delta t \boldsymbol{M}\, \widehat{\mathbf{V}}_{l,i}, \tag{2.40}$$

where $\mathbf{x}(\boldsymbol{\xi}, t^n)$ is given by the mapping (2.3) based on the known vertex coordinates of element T_i^n at time t^n. Even in this case we monitor the residuals of (2.39) and (2.40), so that the iterative procedure stops when the prescribed tolerance has been reached.

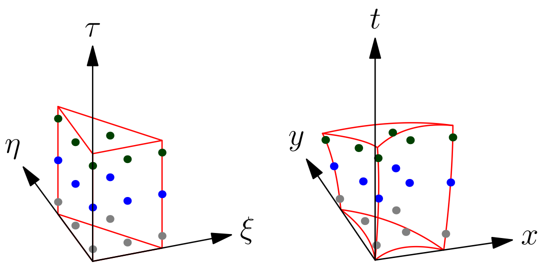


Figure 2.5: Iso-parametric mapping of the space-time reference element (left) to the physical space-time element (right) used within the local space-time Discontinuous Galerkin (DG) predictor for a triangular control volume.

2.4 Mesh motion

Lagrangian schemes have been designed and developed in order to compute the flow variables by moving together with the fluid. As a consequence, the computational mesh continuously changes its configuration in time, following as closely as possible the flow motion.
Once the local predictor procedure has been carried out, at each vertex k different velocity vectors $\boldsymbol{V}_{k,j}^n$ are defined, depending on the number of elements T_j^n that belong to the Voronoi neighborhood $\mathcal{V}_k$ of node k, as depicted in Figure 2.6. In order to obtain a *continuous* mesh configuration at the new time level t^{n+1}, one has to fix a *unique* time-averaged node velocity $\boldsymbol{V}_k^n$ using a *node solver* algorithm. Moreover, the flow motion may become very complex, hence highly deforming the computational elements, that are compressed, twisted or even tangled. Therefore a suitable *rezoning algorithm* is typically used to improve the mesh quality together with a so-called *relaxation algorithm* to partially recover the optimal Lagrangian accuracy where the computational elements are not distorted too much.
The entire mesh motion is then composed by three main steps, namely the Lagrangian step, the rezoning step and the relaxation step. Since an exhaustive description of such stages would become to cumbersome to be carried out at this point of the overall algorithm description, we refer the reader to Chapter 3 for a detailed presentation of the mesh motion procedure adopted in our ALE numerical method.
Here we only provide the most simple and straightforward solution to determine a unique mesh velocity for each node k of the mesh, that has been proposed by Cheng and Shu in [65]. In such a very general approach the node velocity is chosen to be the arithmetic average velocity among all the contributions coming from the neighbor elements, hence yielding an *arithmetic node solver*. The local velocity contribution $\boldsymbol{V}_{k,j}^n$ of element T_j^n that belongs to the Voronoi neighborhood $\mathcal{V}_k$ is evaluated as the time integral of the high order vertex-extrapolated velocity at node k, i.e.

$$\boldsymbol{V}_{k,j}^n = \left(\int_0^1 \theta_l(\xi_{m(k)}^e, \eta_{m(k)}^e, \zeta_{m(k)}^e, \tau) d\tau \right) \widehat{\boldsymbol{V}}_{l,j}, \tag{2.41}$$

where $m(k)$ denotes a mapping from the global node number k defined in the mesh configuration $\mathcal{T}_\Omega^n$ to the local vertex number in element T_j^n, according to the local connectivity $\mathcal{C}$ (2.4). If $d = 2$ we simply set $\zeta_{m(k)}^e = 0$ and the

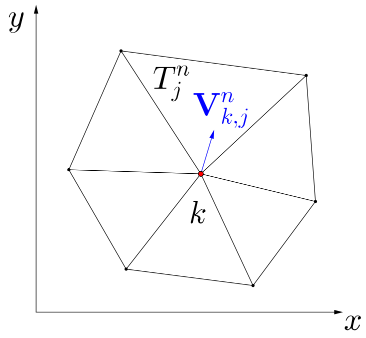


Figure 2.6: Geometrical notation for the arithmetic node solver: k is the local node, T_j^n denotes one element of the neighborhood $\mathcal{V}_k$ and $\boldsymbol{V}_{k,j}^n$ is the local velocity contribution of element T_j^n given by (2.41).

space-time basis functions θ_l are defined on the reference triangle T_E with the mappings (2.3)-(2.18). The final node velocity $\boldsymbol{V}_k^n$ is then given by

$$\boldsymbol{V}_k^n = \frac{1}{N_{j(k)}} \sum_{T_j^n \in \mathcal{V}_k} \boldsymbol{V}_{k,j}^n, \tag{2.42}$$

with $N_{j(k)}$ representing the number of Voronoi neighbors T_j of node k.
As a result of the node solver, we obtain a unique high order accurate time-averaged vertex velocity $\boldsymbol{V}_k^n$ for each vertex k, that is used to evaluate the *new* node position $\boldsymbol{X}_k^{n+1}$ of node k at time t^{n+1} as

$$\boldsymbol{X}_k^{n+1} = \boldsymbol{X}_k^n + \Delta t\, \boldsymbol{V}_k^n, \tag{2.43}$$

where $\boldsymbol{X}_k^n$ denotes the coordinates of node k at the current time level t^n. With the new vertex positions we are able to update all the other geometric quantities needed for the computation, e.g. normal vectors, volumes, side lengths, *etc.*

2.5 High order ALE finite volume schemes

Since finite volume schemes are based on the integral formulation of the conservation law, we first have to clearly define the control volumes where integration will be carried out. For each element T_i the new vertex coordinates $\boldsymbol{X}_k^{n+1}$ are connected to the old coordinates $\boldsymbol{X}_k^n$ with *straight* line segments, yielding a multidimensional space-time control volume $C_i^n = T_i(t) \times \left[t^n; t^{n+1}\right]$, that involves overall five space-time sub-surfaces in 2D or six sub-volumes in 3D, as depicted in Figure 2.7. Specifically, the space-time volume C_i^n is bounded on the bottom and on the top by the element configuration at the current time level T_i^n and at the new time level T_i^{n+1}, respectively, while it is closed with a total number of $\mathcal{N}_i = (d+1)$ lateral sub-volumes $\partial C_{ij}^n = \partial T_{ij}(t) \times [t^n; t^{n+1}]$ that are given by the evolution of each face $\partial T_{ij}(t)$ of element T_i within the timestep $\Delta t = (t^{n+1} - t^n)$. Therefore the space-time volume C_i^n is bounded by its surface ∂C_i^n which is given by

$$\partial C_i^n = \left(\bigcup_{T_j(t) \in \mathcal{N}_i} \partial C_{ij}^n \right) \cup \, T_i^n \, \cup \, T_i^{n+1}. \tag{2.44}$$

For the sake of clarity from now on we will present the finite volume scheme for the three-dimensional case, hence addressing the sub-volumes with ∂C_{ij}^n and the faces with $\partial T_{ij}(t)$. If $d = 2$ then volumes reduce to surfaces, while surfaces reduce to segments and the algorithm formulation can be easily derived by setting the $z-$aligned coordinate to zero as well as all its related physical quantities.
In order to develop a Lagrangian-type finite volume schemes on moving unstructured meshes, we rely on a space-time divergence operator $\tilde{\nabla}$ that allows the governing PDE (2.1) to be reformulated more compactly as

$$\tilde{\nabla} \cdot \tilde{\boldsymbol{F}} + \tilde{\boldsymbol{B}}(\boldsymbol{Q}) \cdot \tilde{\nabla}\boldsymbol{Q} = \boldsymbol{S}(\boldsymbol{Q}), \qquad \tilde{\nabla} = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}, \frac{\partial}{\partial t} \right)^T, \tag{2.45}$$

where the space-time flux tensor $\tilde{\boldsymbol{F}}$ and the system matrix $\tilde{\boldsymbol{B}}$ explicitly read

$$\tilde{\boldsymbol{F}} = (\boldsymbol{f}, \, \boldsymbol{g}, \, \boldsymbol{h}, \, \boldsymbol{Q}), \qquad \tilde{\boldsymbol{B}} = (\boldsymbol{B}_1, \boldsymbol{B}_2, \boldsymbol{B}_3, 0). \tag{2.46}$$

For the computation of the state vector at the new time level $\boldsymbol{Q}^{n+1}$, the balance law (2.45) is integrated over a *four-dimensional* space-time control volume

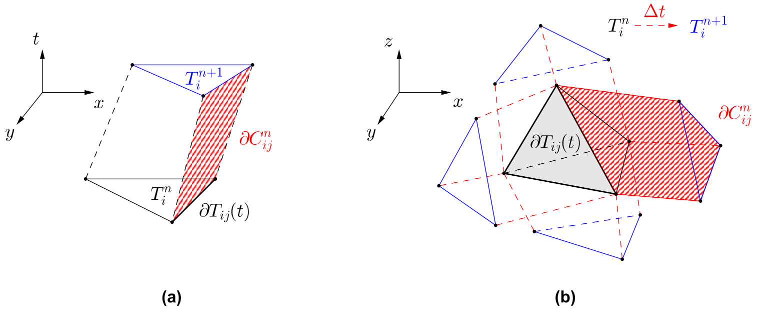


Figure 2.7: Space-time evolution of element T_i within one timestep Δt in two (a) and three (b) space dimensions. The dashed red lines denote the evolution in time of the faces ∂C_{ij}^n of the control volume T_i, whose configuration at the current time level t^n and at the new time level t^{n+1} is depicted in black and blue, respectively.

$\mathcal{C}_i^n = T_i(t) \times \left[t^n; t^{n+1}\right]$, i.e.

$$\int\limits_{\mathcal{C}_i^n} \tilde{\nabla} \cdot \tilde{\boldsymbol{F}} \, d\boldsymbol{x}dt + \int\limits_{\mathcal{C}_i^n} \tilde{\boldsymbol{B}}(\boldsymbol{Q}) \cdot \tilde{\nabla} \boldsymbol{Q} \, d\boldsymbol{x}dt = \int\limits_{\mathcal{C}_i^n} \boldsymbol{S}(\boldsymbol{Q}) \, d\boldsymbol{x}dt. \tag{2.47}$$

Application of the theorem of Gauss yields

$$\int\limits_{\partial\mathcal{C}_i^n} \tilde{\boldsymbol{F}} \cdot \; \tilde{\boldsymbol{n}} \, dS + \int\limits_{\mathcal{C}_i^n} \tilde{\boldsymbol{B}}(\boldsymbol{Q}) \cdot \tilde{\nabla} \boldsymbol{Q} \, d\boldsymbol{x}dt = \int\limits_{\mathcal{C}_i^n} \boldsymbol{S}(\boldsymbol{Q}) \, d\boldsymbol{x}dt, \tag{2.48}$$

where the space-time volume integral on the left of (2.47) has been rewritten as the sum of the fluxes computed over the *three-dimensional* space-time volume $\partial\mathcal{C}_i^n$, given by the evolution of each face of element $T_i(t)$ within the timestep Δt, as depicted in Figure 2.7. The symbol $\tilde{\boldsymbol{n}} = (\tilde{n}_x, \tilde{n}_y, \tilde{n}_z, \tilde{n}_t)$ denotes the outward pointing space-time unit normal vector on the space-time surface ∂C_i^n.

In order to simplify the integral computation, each of the space-time sub-volumes is mapped to a reference element. For the configurations at the current and at the new time level, T_i^n and T_i^{n+1}, we use the mapping (2.3) with $(\xi, \eta, \zeta) \in [0;1]$. The space-time unit normal vectors simply read $\tilde{\boldsymbol{n}} = (0,0,0,-1)$ for T_i^n and $\tilde{\boldsymbol{n}} = (0,0,0,1)$ for T_i^{n+1}, since these volumes are orthogonal to the time coordinate. For the lateral sub-volumes ∂C_{ij}^n we adopt a linear parametrization to map the physical volume to a three-dimensional space-time reference prism, as shown in Figure 2.8. Starting from the old vertex coordinates $\boldsymbol{X}_{ik}^n$ and the new ones $\boldsymbol{X}_{ik}^{n+1}$, that are *known* from the mesh motion algorithm (briefly described in Section 2.4 and detailed in next Chapter 3), the lateral sub-volumes are parametrized using a set of linear basis functions $\beta_k(\chi_1, \chi_2, \tau)$ that are defined on a local reference system (χ_1, χ_2, τ) which is oriented orthogonally w.r.t. the face $\partial T_{ij}(t)$ of element T_i^n, e.g. the reference time coordinate τ is orthogonal to the reference space coordinates (χ_1, χ_2) that lie on $\partial T_{ij}(t)$. The temporal mapping is simply given by $t = t^n + \tau\,\Delta t$, hence $t_{\chi_1} = t_{\chi_2} = 0$ and $t_\tau = \Delta t$. The lateral space-time sub-volume ∂C_{ij}^n is defined by a total number N_k of vertices of physical coordinates $\tilde{\boldsymbol{X}}_{ij,k}^n$, namely $N_k = 4$ or $N_k = 6$ in two or three space dimensions, respectively. The first three vectors $(\boldsymbol{X}_{ij,1}^n, \boldsymbol{X}_{ij,2}^n, \boldsymbol{X}_{ij,3}^n)$ are the nodes defining the common face $\partial T_{ij}(t^n)$ at time t^n, while the same procedure applies at the new time level t^{n+1}. Therefore the six vectors $\tilde{\boldsymbol{X}}_{ij,k}^n$ are given by

$$\tilde{\boldsymbol{X}}_{ij,1}^n = \left(\boldsymbol{X}_{ij,1}^n, t^n\right), \qquad \tilde{\boldsymbol{X}}_{ij,2}^n = \left(\boldsymbol{X}_{ij,2}^n, t^n\right), \qquad \tilde{\boldsymbol{X}}_{ij,3}^n = \left(\boldsymbol{X}_{ij,3}^n, t^n\right),$$

$$\tilde{\boldsymbol{X}}_{ij,4}^n = \left(\boldsymbol{X}_{ij,1}^{n+1}, t^{n+1}\right), \qquad \tilde{\boldsymbol{X}}_{ij,5}^n = \left(\boldsymbol{X}_{ij,2}^{n+1}, t^{n+1}\right), \qquad \tilde{\boldsymbol{X}}_{ij,6}^n = \left(\boldsymbol{X}_{ij,3}^{n+1}, t^{n+1}\right), \tag{2.49}$$

and the parametrization for ∂C_{ij}^n reads

$$\partial C_{ij}^n = \tilde{\boldsymbol{x}}\left(\chi_1, \chi_2, \tau\right) = \sum_{k=1}^{N_k} \beta_k(\chi_1, \chi_2, \tau)\, \tilde{\boldsymbol{X}}_{ij,k}^n, \tag{2.50}$$

with $0 \leq \chi_1 \leq 1$, $0 \leq \chi_2 \leq 1 - \chi_1$ and $0 \leq \tau \leq 1$. The basis functions

$\beta_k(\chi_1, \chi_2, \tau)$ for the reference space-time element in 3D are given by

$$\begin{aligned}
\beta_1(\chi_1, \chi_2, \tau) &= (1 - \chi_1 - \chi_2)(1 - \tau), \\
\beta_2(\chi_1, \chi_2, \tau) &= \chi_1(1 - \tau), \\
\beta_3(\chi_1, \chi_2, \tau) &= \chi_2(1 - \tau), \\
\beta_4(\chi_1, \chi_2, \tau) &= (1 - \chi_1 - \chi_2)(\tau) \\
\beta_5(\chi_1, \chi_2, \tau) &= \chi_1 \tau, \\
\beta_6(\chi_1, \chi_2, \tau) &= \chi_2 \tau. \qquad (2.51)
\end{aligned}$$

The corresponding basis functions for the two dimensional case can be easily obtained by setting $\chi_2 = 0$, since the space-time reference element is defined in the reference system (χ_1, τ), as shown in Figure 2.8.
The coordinate transformation is associated with a matrix $\mathcal{T}$ that reads

$$\mathcal{T} = \left(\hat{\mathbf{e}}, \frac{\partial \tilde{\mathbf{x}}}{\partial \chi_1}, \frac{\partial \tilde{\mathbf{x}}}{\partial \chi_2}, \frac{\partial \tilde{\mathbf{x}}}{\partial \tau} \right)^T, \qquad (2.52)$$

with $\hat{\mathbf{e}} = (\hat{\mathbf{e}}_1, \hat{\mathbf{e}}_2, \hat{\mathbf{e}}_3, \hat{\mathbf{e}}_4)$ and where $\hat{\mathbf{e}}_p$ represents the unit vector aligned with the p-th axis of the physical coordinate system (x, y, z, t). In the following $\tilde{x}_q$ denotes the q-th component of vector $\tilde{\mathbf{x}}$. The determinant of $\mathcal{T}$ produces at the same time the space-time volume $|\partial C^n_{ij}|$ of the space-time sub-volume ∂C^n_{ij} and the associated space-time normal vector $\tilde{\boldsymbol{n}}_{ij}$, as

$$\tilde{\boldsymbol{n}}_{ij} = \left(\epsilon_{pqrs}\, \hat{\mathbf{e}}_p \frac{\partial \tilde{x}_q}{\partial \chi_1} \frac{\partial \tilde{x}_r}{\partial \chi_2} \frac{\partial \tilde{x}_s}{\partial \tau} \right) / |\partial C^n_{ij}|, \qquad (2.53)$$

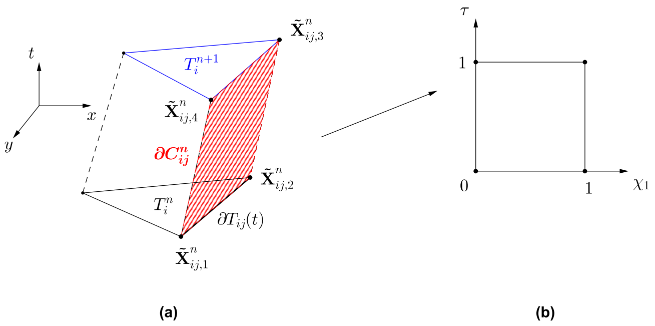
where the *Levi-Civita* symbol has been used according to the usual definition

$$\epsilon_{pqrs} = \begin{cases} +1, & \text{if } (p,q,r,s) \text{ is an } \textit{even} \text{ permutation of } (1,2,3,4), \\ -1, & \text{if } (p,q,r,s) \text{ is an } \textit{odd} \text{ permutation of } (1,2,3,4), \\ 0, & \text{otherwise,} \end{cases} \qquad (2.54)$$

and with

$$|\partial C^n_{ij}| = \left\| \epsilon_{pqrs}\, \hat{\mathbf{e}}_p \frac{\partial \tilde{x}_q}{\partial \chi_1} \frac{\partial \tilde{x}_r}{\partial \chi_2} \frac{\partial \tilde{x}_s}{\partial \tau} \right\|. \qquad (2.55)$$

We now need to discretize the integral form (2.48) to obtain an evolution equation of the cell averages of the state vector $\boldsymbol{Q}$. Depending on the governing PDE system (2.1), we might deal either with non-conservative products or not. Therefore in next Sections 2.5.1 and 2.5.2 we will present an ALE finite volume method which is suitable for each one of the aforementioned cases.


t
x
y
$\tilde{\mathbf{X}}^n_{ij,3}$
T_i^{n+1}
$\tilde{\mathbf{X}}^n_{ij,4}$
∂C^n_{ij}
$\tilde{\mathbf{X}}^n_{ij,2}$
T_i^n
$\partial T_{ij}(t)$
$\tilde{\mathbf{X}}^n_{ij,1}$
τ
1
0
1
χ_1
(a)
(b)

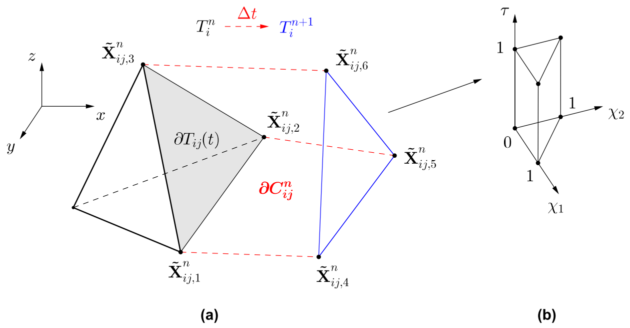

$T_i^n \xrightarrow{\Delta t} T_i^{n+1}$
z
x
y
$\tilde{\mathbf{X}}^n_{ij,3}$
$\tilde{\mathbf{X}}^n_{ij,6}$
$\tilde{\mathbf{X}}^n_{ij,2}$
$\partial T_{ij}(t)$
$\tilde{\mathbf{X}}^n_{ij,5}$
∂C^n_{ij}
$\tilde{\mathbf{X}}^n_{ij,1}$
$\tilde{\mathbf{X}}^n_{ij,4}$
τ
1
1
χ_2
0
1
χ_1
(a)
(b)

Figure 2.8: Physical space-time element (a) and parametrization of the lateral space-time sub-volume ∂C^n_{ij} (b) for triangles (top) and tetrahedra (bottom). The dashed red lines denote the evolution in time of the faces of the element, whose configuration at the current time level t^n and at the new time level t^{n+1} is depicted in black and blue, respectively.

Before proceeding with the description of the evolution equation of the cell averages of the state vector $\boldsymbol{Q}$, let us underline that the integration over a closed space-time control volume, as done in (2.48), automatically satisfies the so-called geometric conservation law (GCL), since application of Gauss' theorem yields

$$\int_{\partial \mathcal{C}_i^n} \tilde{\boldsymbol{n}} \, dS = 0. \tag{2.56}$$

Note that (2.56) is the *time-integrated* (fully discrete) version of the *classical GCL relation* typically used in the Lagrangian community, see for example equation (2.9i) on page 1785 of reference [172]. For the proof of this equivalence, see Appendix C. For *all* the applications and the test problems shown later in Chapter 6 the integral appearing in (2.56) has been evaluated for each element and at each timestep using Gaussian quadrature rules of sufficient accuracy. We could verify that condition (2.56) has been always satisfied on the discrete level up to machine precision.
Last but not least, we would like to state clearly that within the family of high order one-step direct ALE methods proposed in this work the choices of the Riemann solver, the reconstruction technique and the mesh velocity are deliberately independent from each other, hence the method in general allows a mass flux. This means that even for $\boldsymbol{V} = \boldsymbol{v}$ the proposed scheme is *not* meant to be a pure Lagrangian method *in sensu stricto*. However, the family of schemes presented in this framework is able to resolve material interfaces and contact waves very well, much better than traditional high order Eulerian methods on fixed meshes.

2.5.1 Formulation for non-conservative systems

The non-conservative term $\tilde{\boldsymbol{B}}(\boldsymbol{Q}) \cdot \tilde{\nabla} \boldsymbol{Q}$ appearing in (2.48) is integrated by using a *path-conservative* approach [56, 57, 97, 99, 107, 180, 192, 193, 198, 235], which follows the theory of Dal Maso-Le Floch and Murat [175] and defines the non-conservative term as a Borel measure. For a more detailed discussion on the known limitations and problems associated with path-conservative schemes see [7, 58]. One thus obtains

$$\int\limits_{\partial \mathcal{C}_i^n} \left(\tilde{\boldsymbol{F}} + \tilde{\boldsymbol{D}} \right) \cdot \, \tilde{\boldsymbol{n}} \, dS + \int\limits_{\mathcal{C}_i^n \backslash \partial \mathcal{C}_i^n} \tilde{\boldsymbol{B}}(\boldsymbol{Q}) \cdot \tilde{\nabla} \boldsymbol{Q} \, d\boldsymbol{x} dt = \int\limits_{\mathcal{C}_i^n} \boldsymbol{S}(\boldsymbol{Q}) \, d\boldsymbol{x} dt, \tag{2.57}$$

where a new term $\tilde{\boldsymbol{D}}$ has been introduced in order to take into account potential jumps of the solution $\boldsymbol{Q}$ on the space-time element boundaries $\partial \mathcal{C}_i^n$. This term is computed by the path integral

$$\tilde{\boldsymbol{D}} \cdot \tilde{\boldsymbol{n}} = \frac{1}{2} \int_0^1 \tilde{\boldsymbol{B}} \left(\boldsymbol{\Psi}(\boldsymbol{Q}^-, \boldsymbol{Q}^+, s) \right) \cdot \tilde{\boldsymbol{n}} \, \frac{\partial \boldsymbol{\Psi}}{\partial s} \, ds. \tag{2.58}$$

The integration path $\boldsymbol{\Psi}$ in (2.58) is chosen to be a simple straight-line segment [57, 99, 107, 192], although other choices are possible. Therefore it reads

$$\boldsymbol{\Psi} = \boldsymbol{\Psi}(\boldsymbol{Q}^-, \boldsymbol{Q}^+, s) = \boldsymbol{Q}^- + s(\boldsymbol{Q}^+ - \boldsymbol{Q}^-), \tag{2.59}$$

and the jump term (2.58) simply reduces to

$$\tilde{\boldsymbol{D}} \cdot \tilde{\boldsymbol{n}} = \frac{1}{2} \left(\int_0^1 \tilde{\boldsymbol{B}} \left(\boldsymbol{\Psi}(\boldsymbol{Q}^-, \boldsymbol{Q}^+, s) \right) \cdot \tilde{\boldsymbol{n}} \, ds \right) \left(\boldsymbol{Q}^+ - \boldsymbol{Q}^- \right), \tag{2.60}$$

with $\left(\boldsymbol{Q}^-, \boldsymbol{Q}^+\right)$ representing the two vectors of conserved variables within element T_i^n and its direct neighbor T_j^n, respectively.
The final one-step ALE finite volume scheme for non-conservative hyperbolic systems takes the following form:

$$\begin{aligned} |T_i^{n+1}| \, \boldsymbol{Q}_i^{n+1} = |T_i^n| \, \boldsymbol{Q}_i^n \quad & - \sum_{T_j \in \mathcal{N}_i} \int_0^1 \int_0^1 \int_0^{1-\chi_1} |\partial C_{ij}^n| \tilde{\boldsymbol{G}}_{ij} \, d\chi_2 d\chi_1 d\tau \\ & + \int_{\mathcal{C}_i^n \setminus \partial \mathcal{C}_i^n} \left(\boldsymbol{S}_h - \boldsymbol{P}_h \right) \, d\mathbf{x} dt, \end{aligned} \tag{2.61}$$

where the term $\tilde{\boldsymbol{G}}_{ij} \cdot \tilde{\boldsymbol{n}}_{ij}$ contains the Arbitrary-Lagrangian-Eulerian numerical flux function as well as the path-conservative jump term, hence allowing the discontinuity of the predictor solution $\boldsymbol{q}_h$ that occurs at the space-time sub-volume ∂C_{ij}^n to be properly resolved. The volume and surface integrals appearing in (2.61) are approximated using multidimensional Gaussian quadrature rules, see [218] for details. The term $\tilde{\boldsymbol{G}}_{ij}$ can be evaluated using a simple ALE Rusanov-type scheme [108] as

$$\tilde{\boldsymbol{G}}_{ij} = \frac{1}{2} \left(\tilde{\boldsymbol{F}}(\boldsymbol{q}_h^+) + \tilde{\boldsymbol{F}}(\boldsymbol{q}_h^-) \right) \cdot \tilde{\boldsymbol{n}}_{ij} + \frac{1}{2} \left(\int_0^1 \tilde{\boldsymbol{B}}(\boldsymbol{\Psi}) \cdot \tilde{\boldsymbol{n}} \, ds - |\lambda_{\max}| \boldsymbol{I} \right) \left(\boldsymbol{q}_h^+ - \boldsymbol{q}_h^- \right), \tag{2.62}$$

where $\boldsymbol{q}_h^-$ and $\boldsymbol{q}_h^+$ are the local space-time predictor solution inside element $T_i(t)$ and the neighbor $T_j(t)$, respectively, and $|\lambda_{\max}|$ denotes the maximum absolute value of the eigenvalues of the matrix $\tilde{\boldsymbol{A}} \cdot \tilde{\boldsymbol{n}}$ in space-time normal direction. Using the normal mesh velocity $\boldsymbol{V} \cdot \boldsymbol{n}$, matrix $\tilde{\boldsymbol{A}}_{\tilde{\boldsymbol{n}}}$ reads

$$\tilde{\boldsymbol{A}}_{\tilde{\boldsymbol{n}}} = \tilde{\boldsymbol{A}} \cdot \tilde{\boldsymbol{n}} = \left(\sqrt{\tilde{n}_x^2 + \tilde{n}_y^2 + \tilde{n}_z^2}\right) \left[\left(\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{Q}} + \boldsymbol{B} \right) \cdot \boldsymbol{n} - (\boldsymbol{V} \cdot \boldsymbol{n}) \boldsymbol{I} \right], \tag{2.63}$$

with $\boldsymbol{I}$ denoting the $\nu \times \nu$ identity matrix, $\boldsymbol{A} = \partial \boldsymbol{F} / \partial \boldsymbol{Q} + \boldsymbol{B}$ representing the classical Eulerian system matrix and $\boldsymbol{n}$ being the spatial unit normal vector given by

$$\boldsymbol{n} = \frac{(\tilde{n}_x, \tilde{n}_y, \tilde{n}_z)^T}{\sqrt{\tilde{n}_x^2 + \tilde{n}_y^2 + \tilde{n}_z^2}}. \tag{2.64}$$

The numerical flux term $\tilde{\boldsymbol{G}}_{ij}$ can be also computed relying on a more sophisticated Osher-type scheme [189], introduced in the Eulerian framework for conservative and non-conservative hyperbolic systems in [106, 107]. It reads

$$\tilde{\boldsymbol{G}}_{ij} = \frac{1}{2} \left(\tilde{\boldsymbol{F}}(\boldsymbol{q}_h^+) + \tilde{\boldsymbol{F}}(\boldsymbol{q}_h^-) \right) \cdot \tilde{\boldsymbol{n}}_{ij} + \frac{1}{2} \left(\int_0^1 \left(\tilde{\boldsymbol{B}}(\boldsymbol{\Psi}) \cdot \tilde{\boldsymbol{n}} - \left| \tilde{\boldsymbol{A}}_{\tilde{\boldsymbol{n}}}(\boldsymbol{\Psi}) \right| \right) ds \right) \left(\boldsymbol{q}_h^+ - \boldsymbol{q}_h^- \right), \tag{2.65}$$

where the matrix absolute value operator is computed as usual as

$$|\boldsymbol{A}| = \boldsymbol{R} |\boldsymbol{\Lambda}| \boldsymbol{R}^{-1}, \qquad |\boldsymbol{\Lambda}| = \operatorname{diag}\left(|\lambda_1|, |\lambda_2|, ..., |\lambda_\nu|\right), \tag{2.66}$$

with the right eigenvector matrix $\boldsymbol{R}$ and its inverse $\boldsymbol{R}^{-1}$. According to [106,107] Gaussian quadrature formulae of sufficient accuracy are adopted to evaluate the path integral present in (2.65).

2.5.2 Formulation for conservative systems

If the governing PDE (2.1) does not involve any non-conservative term, then $\boldsymbol{B}(\boldsymbol{Q}) = 0$ and the integral form (2.48) simply reduces to

$$\int_{\partial \mathcal{C}_i^n} \tilde{\boldsymbol{F}} \cdot \tilde{\boldsymbol{n}} \, dS = \int_{\mathcal{C}_i^n} \boldsymbol{S}(\boldsymbol{Q}) \, d\boldsymbol{x} dt, \tag{2.67}$$

which is then discretized as

$$\begin{aligned} |T_i^{n+1}|\,\boldsymbol{Q}_i^{n+1} = |T_i^n|\,\boldsymbol{Q}_i^n \quad &- \sum_{T_j \in \mathcal{N}_i} \int\limits_0^1 \int\limits_0^1 \int\limits_0^{1-\chi_1} |\partial C_{ij}^n| \tilde{\boldsymbol{F}}_{ij} \cdot \tilde{\boldsymbol{n}}_{ij} \, d\chi_2 d\chi_1 d\tau \\ &+ \int\limits_{\mathcal{C}_i^n} \boldsymbol{S}_h \, d\boldsymbol{x} dt, \end{aligned} \tag{2.68}$$

where the discontinuity of the predictor solution $\boldsymbol{q}_h$ at the space-time sub-volume ∂C_{ij}^n is resolved by a numerical flux function $\tilde{\boldsymbol{F}}_{ij} \cdot \tilde{\boldsymbol{n}}_{ij}$.
The numerical flux $\tilde{\boldsymbol{F}}_{ij} \cdot \tilde{\boldsymbol{n}}_{ij}$ appearing in (2.68) is chosen to be either a simple Rusanov-type ALE flux or a more sophisticated Osher-type ALE flux, as done for the non-conservative ALE finite volume scheme presented in the previous section. Here, the expression for the Rusanov flux is given by

$$\tilde{\boldsymbol{F}}_{ij} \cdot \tilde{\boldsymbol{n}}_{ij} = \frac{1}{2}\left(\tilde{\boldsymbol{F}}(\boldsymbol{q}_h^+) + \tilde{\boldsymbol{F}}(\boldsymbol{q}_h^-)\right) \cdot \tilde{\boldsymbol{n}}_{ij} - \frac{1}{2}|\lambda_{\max}| \left(\boldsymbol{q}_h^+ - \boldsymbol{q}_h^-\right), \tag{2.69}$$

with $|\lambda_{\max}|$ denoting the maximum eigenvalue of the ALE Jacobian matrix w.r.t. the normal direction in space, i.e.

$$\boldsymbol{A}_{\boldsymbol{n}}^{\boldsymbol{V}}(\boldsymbol{Q}) = \left(\sqrt{\tilde{n}_x^2 + \tilde{n}_y^2 + \tilde{n}_z^2}\right) \left[\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{Q}} \cdot \boldsymbol{n} - (\boldsymbol{V} \cdot \boldsymbol{n})\, \boldsymbol{I}\right], \qquad \boldsymbol{n} = \frac{(\tilde{n}_x, \tilde{n}_y, \tilde{n}_z)^T}{\sqrt{\tilde{n}_x^2 + \tilde{n}_y^2 + \tilde{n}_z^2}}, \tag{2.70}$$

where $\boldsymbol{I}$ is the identity matrix and $\boldsymbol{V} \cdot \boldsymbol{n}$ represents the local normal mesh velocity.
The Osher-type flux formulation reads

$$\tilde{\boldsymbol{F}}_{ij} \cdot \tilde{\boldsymbol{n}}_{ij} = \frac{1}{2}\left(\tilde{\boldsymbol{F}}(\boldsymbol{q}_h^+) + \tilde{\boldsymbol{F}}(\boldsymbol{q}_h^-)\right) \cdot \tilde{\boldsymbol{n}}_{ij} - \frac{1}{2}\left(\int\limits_0^1 \left|\boldsymbol{A}_{\boldsymbol{n}}^{\boldsymbol{V}}(\boldsymbol{\Psi}(s))\right| ds\right) \left(\boldsymbol{q}_h^+ - \boldsymbol{q}_h^-\right), \tag{2.71}$$

where the left and the right state across the discontinuity are connected using the simple straight-line segment path (2.59), hence

$$\boldsymbol{\Psi}(s) = \boldsymbol{q}_h^- + s\left(\boldsymbol{q}_h^+ - \boldsymbol{q}_h^-\right), \qquad 0 \le s \le 1. \tag{2.72}$$

According to [106] the integral in (2.71) is evaluated numerically using Gaussian quadrature formulae of suitable order of accuracy. The absolute value of the dissipation matrix in (2.71) is given by (2.66).

3 Mesh Motion

This chapter is meant to focus on the detailed description of the procedure needed to determine how the computational mesh moves. Indeed, that is a crucial point in any moving mesh algorithm.
Lagrangian schemes generally aim at following as closely as possible the fluid flow, hence implying mesh motion. In the Lagrangian description of the fluid the nonlinear convective terms disappear and Lagrangian schemes exhibit virtually no numerical dissipation at material interfaces. As a consequence the mesh velocity, i.e. the velocity vector of each vertex of the mesh, has to be evaluated very accurately in order to guarantee the high resolution properties for contact waves and material discontinuities.
However, the fluid flow may become sometimes very complex, leading to highly deformed control volumes that may drastically reduce the admissible timestep, which is computed under a classical Courant-Friedrichs-Levy number (CFL) stability condition (see Eqn. (2.19) in Chapter 2). If the grid keeps moving with the fluid velocity, one will end up with mesh elements which are almost degenerate, i.e. with zero volume, or even inverted, that is with negative Jacobian.
Therefore, the challenge of any Lagrangian scheme is to preserve at the same time its excellent resolution features together with a good mesh quality without invalid elements. It is evident that the mesh motion becomes a key point in such a difficult task. For pure Lagrangian algorithms the mesh motion procedure is typically composed by three main steps:

- the *Lagrangian step* allows each node of the computational mesh to be assigned with a unique mesh velocity which is as Lagrangian as possible, i.e. the most accurate local fluid velocity that the scheme is able to capture. This step is carried out by a so-called *node solver*, which considers the entire Voronoi neighborhood of the node to get the physical state at the vertex. Several node solvers have been proposed in literature and in Section 3.1 we present the ones adopted in our ALE method;

- the *rezoning step* is used to improve the local mesh quality. It is based on the optimization of a geometrical goal function that does not take into account any physical information. Sometimes the rezoning stage is replaced by a *remeshing* algorithm, where the global mesh configuration is reshaped, hence allowing the grid topology and its associated connectivity to change. Furthermore the rezoning and remeshing algorithm must be able to maintain the exact location of material interfaces and contact waves. In our algorithm this would be extremely inefficient, since we should recompute all the WENO stencils needed for the high order reconstruction procedure, see Section 2.2 in Chapter 2;

- the *remapping step* is devoted to interpolate the numerical solution, which is defined on the old mesh, to the new grid configuration obtained by the rezoning or remeshing step. This is necessary in order to maintain the high quality features of Lagrangian schemes ensuring zero mass flux.

In our direct ALE framework the geometrical quantities are not coupled to the physical quantities as strictly as for pure Lagrangian algorithms [53, 78, 163, 169, 170, 174], since our scheme in general allows a mass flux. Therefore we avoid the use of the remapping step and we rely only on a rezoning algorithm to improve the mesh quality. On the other hand, we still try to perform an almost Lagrangian-like algorithm and we do not want to rezone the mesh nodes where it is not strictly necessary in order to carry on with the computation. For this purpose we adopt a *relaxation algorithm* [130] that mitigates the rezoning procedure where the grid deformation is not too strong.
As a consequence our direct ALE finite volume schemes use the following three steps to compute the mesh motion, which will be presented in this chapter: the Lagrangian step, the rezoning step and the relaxation step. Since the two dimensional version for the mesh motion follows straightforward from the three dimensional algorithm, we will only provide the description for the latter case, we will consider the three dimensional case, as done in the previous chapter.

3.1 The Lagrangian step: node solvers

At the end of the element-local predictor procedure illustrated in Section 2.3 of Chapter 2, each vertex k is assigned with several velocity vectors $\boldsymbol{V}_{k,j}$, each of them coming from the Voronoi neighborhood which is composed by the neighbor elements that share the common node k, according to Figure 2.6. Moving the same vertex k to the next time level t^{n+1} with different velocities would lead to a discontinuity in the geometry, that is not admissible in our direct Arbitrary-Lagrangian-Eulerian algorithm. Therefore a node solver technique

is adopted in order to fix a *unique* velocity for each node of the computational grid. This is a common feature in all cell-centered Lagrangian schemes.
We compare three different methods to compute the node velocity with each other and briefly describe each of them in the following. Once a unique high order accurate time-averaged vertex velocity $\boldsymbol{V}_k^n$ is known, the new *Lagrangian coordinates* $\mathbf{X}_k^{Lag}$ are simply given by

$$\mathbf{X}_k^{Lag} = \mathbf{X}_k^n + \Delta t\, \boldsymbol{V}_k^n. \tag{3.1}$$

with $\boldsymbol{X}_k^n$ representing the coordinates of node k at the current time level t^n. Note that (3.1) is the weak integral form of the ODE (2.29) that governs the vertex position, hence it is not just simply a first order Euler method, but it is high order accurate if the time-averaged node velocity $\boldsymbol{V}_k^n$ is computed with high order of accuracy in time.

3.1.1 The node solver of Cheng and Shu $\mathcal{NS}_{cs}$.

In [65–67] Cheng and Shu introduced a very simple and general formulation for obtaining the final node velocity, which is chosen to be the arithmetic average velocity amongst all the contributions coming from the neighbor elements, see Eqn. (2.42). Since the mesh might be locally highly deformed, we propose to define the node solver $\mathcal{NS}_{cs}$ using the idea of Cheng and Shu, but taking a *mass weighted* average velocity among the neighborhood $\mathcal{V}_k$ of node k, i.e.

$$\boldsymbol{V}_k^n = \frac{1}{\mu_k} \sum_{T_j^n \in \mathcal{V}_k} \mu_{k,j}\, \boldsymbol{V}_{k,j}^n, \tag{3.2}$$

with

$$\mu_k = \sum_{T_j^n \in \mathcal{V}_k} \mu_{k,j}, \qquad \mu_{k,j} = \rho_j^n |T_j^n|. \tag{3.3}$$

The local weights $\mu_{k,j}$, which are the masses of the elements T_j^n, are defined multiplying the cell averaged value of density ρ_j^n with the cell volume $|T_j^n|$, while the local velocity contributions $\boldsymbol{V}_{k,j}^n$ are computed integrating in time the high order vertex-extrapolated velocity at node k as

$$\boldsymbol{V}_{k,j}^n = \left(\int\limits_0^1 \theta_l(\xi_{m(k)}^e, \eta_{m(k)}^e, \zeta_{m(k)}^e, \tau) d\tau \right) \widehat{\boldsymbol{V}}_{l,j}, \tag{3.4}$$

where $m(k)$ is a mapping from the global node number k to the local node number in element T_j^n given by the connectivity (2.4). Recall that the $(\xi_{m(k)}^e, \eta_{m(k)}^e, \zeta_{m(k)}^e)$

denote the coordinates of the vertices of the reference element in space, see Figure 2.1.

3.1.2 The node solver of Maire $\mathcal{NS}_m$.

In [169–171] Maire et al. developed the node solver $\mathcal{NS}_m$ for hydrodynamics, while in [53] Després presented a similar approach. In [53] the node velocity is computed, within the GLACE scheme, using a nodal solver which is based on two different formulae, namely a linearized-Riemann-invariant relation and an expression for the conservation of momentum around the vertex. The first one can be seen as a multidimensional generalization of a first order Riemann solver in the direction given by the gradient of the volume with respect to the node position, evaluated for each cell which shares the vertex. The second equation ensures the conservation of momentum by construction, since it expresses the that the sum of all forces around the node is zero. This set of equations yields an algebraic linear system, whose solution is the node velocity. Such a technique looks similar to the algorithm proposed in [169–171], which is considered within our work. Indeed, the node solver $\mathcal{NS}_m$ is based on the conservation of total energy in the equations for compressible hydrodynamics. All the details can be found in the above-mentioned references, hence we limit us here only to a brief overview of this nodal solver. According to Figure 3.1, k is the node index, T_j^n denotes the neighbor element j of vertex k and the subscripts (j^R, j^L, j^B) represent the three faces of tetrahedron T_j^n which share node k, ordered adopting a counterclockwise convention. Furthermore $(S_{j^R}, S_{j^L}, S_{j^B})$ are assumed to be one third of the corresponding face areas and $(\boldsymbol{n}_{j^R}, \boldsymbol{n}_{j^L}, \boldsymbol{n}_{j^B})$ denote the associated outward pointing unit normal vectors. In two space dimensions we adopt a coherent notation, hence the two edges of triangle T_j^n are denoted by (j^R, j^L) and (S_{j^R}, S_{j^L}) represent one half of the corresponding edge lengths which are given the normal vectors $(\boldsymbol{n}_{j^R}, \boldsymbol{n}_{j^L})$. Finally p_j is the fluid pressure and c_j is the speed of sound for hydrodynamics. The total energy at the generic node k is conserved only if the sum of the forces acting on node k is zero, i.e.

$$\sum_{T_j^n \in \mathcal{V}_k} \boldsymbol{F}_{k,j} = 0. \tag{3.5}$$

In (3.5) the sub-cell force $\boldsymbol{F}_{k,j}$ exerted by each neighbor element T_j^n onto vertex k, is evaluated solving approximately three half Riemann problems on the faces (j^R, j^L, j^B). The acoustic Riemann solver of Dukowicz et al. [92] is used to

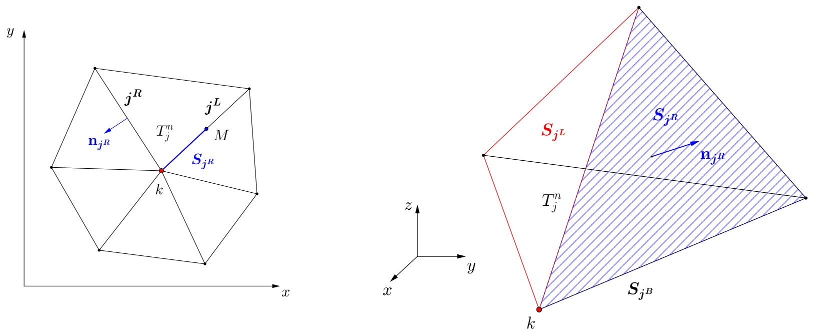


Figure 3.1: Geometrical notation for the node solver $\mathcal{NS}_m$. Left: in 2D T_j^n is the neighbor element of node k, S_{j^R}, S_{j^L} represent one half of the corresponding edges j^R, j^L with the associated normal vectors $\boldsymbol{n}_{j^R}, \boldsymbol{n}_{j^L}$. Right: in 3D only one neighbor element T_j^n of node k is depicted; $S_{j^R}, S_{j^L}, S_{j^R}$ denote one third of the total area of the faces j^R, j^L, j^B of T_j^n that share vertex k, while $\boldsymbol{n}_{j^R}, \boldsymbol{n}_{j^L}, \boldsymbol{n}_{j^B}$ are the corresponding outward pointing unit normal vectors.

obtain the final expression for the sub-cell force, which reads

$$\boldsymbol{F}_{k,j} = S_{k,j} p_{k,j} \boldsymbol{n}_{k,j} - \boldsymbol{M}_{k,j} \left(\boldsymbol{V}_k^n - \boldsymbol{V}_{k,j}^n \right), \tag{3.6}$$

with $S_{k,j} \boldsymbol{n}_{k,j} = S_{j^R} \boldsymbol{n}_{j^R} + S_{j^L} \boldsymbol{n}_{j^L} + S_{j^B} \boldsymbol{n}_{j^B}$ denoting the *corner vector* related to node k. $\boldsymbol{V}_{k,j}^n$ represents the known vertex velocity of cell j according to (3.4), while $\boldsymbol{V}_k^n$ denotes the unknown velocity of node k. $\boldsymbol{M}_{k,j}$ is a $(d \times d)$ symmetric positive definite matrix that is evaluated as

$$\boldsymbol{M}_{k,j} = z_{j^R} S_{j^R} \left(\boldsymbol{n}_{j^R} \otimes \boldsymbol{n}_{j^R} \right) + z_{j^L} S_{j^L} \left(\boldsymbol{n}_{j^L} \otimes \boldsymbol{n}_{j^L} \right) + z_{j^L} S_{j^B} \left(\boldsymbol{n}_{j^B} \otimes \boldsymbol{n}_{j^B} \right), \tag{3.7}$$

where $z_j = \rho_j c_j$ is the acoustic impedance. The equation for the total energy conservation (3.5) can be reformulated using the expression for the sub-cell force (3.6), hence obtaining a linear algebraic system for the unknown node velocity $\boldsymbol{V}_k^n$:

$$\boldsymbol{M}_k \boldsymbol{V}_k^n = \sum_{T_j^n \in \mathcal{V}_k} \left(S_{k,j} p_{k,j} \boldsymbol{n}_{k,j} + \boldsymbol{M}_{k,j} \boldsymbol{V}_{k,j}^n \right), \quad \boldsymbol{M}_k = \sum_{T_j^n \in \mathcal{V}_k} \boldsymbol{M}_{k,j}. \tag{3.8}$$

Since matrix $\boldsymbol{M}_k$ is always invertible, this system admits a unique solution and the node velocity can always be evaluated. Instead of taking the above-defined acoustic impedance, one can compute it as originally proposed by Dukowicz in [92]:

$$z_{j^+} = \rho_j \left[c_j + \Gamma_j \left| \left(\boldsymbol{V}_k^n - \boldsymbol{V}_{k,j}^n \right) \cdot \boldsymbol{n}_{j^+} \right| \right], \tag{3.9}$$

where $\Gamma_j = \frac{\gamma+1}{2}$ is a material dependent parameter which is a function of the ratio of specific heats γ. In this case the system (3.8) becomes nonlinear, due to the dependency of the acoustic impedance on the unknown node velocity, and a suitable iterative algorithm has to be used to obtain the solution.
For magneto-hydrodynamics we adopt the same procedure as for hydrodynamics, where we add the magnetic pressure in the sub-cell force computation (3.6) and we use the fastest magnetospeed for the acoustic impedance evaluation (3.9). The final time-averaged node velocity $\boldsymbol{V}_k^n$ is obtained using Gaussian quadrature in time, where the node solver is invoked at each Gaussian point in time with the corresponding vertex-extrapolated states from the cells surrounding node k.

3.1.3 The node solver of Balsara et al. $\mathcal{NS}_b$.

In a recent series of papers [17, 18, 25, 26, 88] Balsara et al. have proposed a genuinely multidimensional formulation of HLL and HLLC Riemann solvers for nonlinear hyperbolic conservation laws on Cartesian grids and general unstructured meshes in two space dimensions. In [25, 26] the multidimensional Riemann problem is formulated in similarity variables, hence allowing any self-similar one-dimensional Riemann solver to be employed as a building block for the multidimensional Riemann solver, while in [17, 18, 88] the method is based on one-dimensional HLL and HLLC Riemann solvers. There, a family of node-based HLL Riemann solvers is developed that considers a genuinely multidimensional flow structure developing at each grid vertex, in contrast to the classical edge-based Riemann solvers used in traditional Eulerian Godunov-type finite volume schemes. At a grid vertex k one can indeed take into account more physical information because multiple elements T_j^n come together from all possible directions. In this work, we rely on the fact that the genuinely multidimensional HLL Riemann solver can be used as one of the essential building blocks in the *two-dimensional* cell-centered ALE framework, namely as alternative node solver to the two previously mentioned ones. Hence, the multidimensional HLL Riemann solver also allows the node to be assigned a unique node velocity vector $\boldsymbol{V}_k^n$ after the element-local space-time predictor stage. Once the multidimensional HLL state $\boldsymbol{Q}^*$ is computed, the velocity components can be extracted from this so-called strongly interacting state and can be integrated in time in order to move the node to its location at the new time level t^{n+1}. For our purposes we always use the HLL version of the multidimensional Riemann solver proposed in [88]. Figure 3.2, which is taken from [88], shows the neighborhood $\mathcal{V}_k$ of vertex k, where three different states $(\boldsymbol{Q}_1, \boldsymbol{Q}_2, \boldsymbol{Q}_3)$ come together at a node. The method is designed to handle an arbitrary number of states coming together at a node, hence we will use the generic states $\boldsymbol{Q}_j$ being the vertex-extrapolated states from element T_j^n at node k. The edge-aligned unit vector η_j separates the states $\boldsymbol{Q}_j$ and $\boldsymbol{Q}_{j+1}$, which have to be ordered in a counterclockwise fashion. Associated with vectors η_j, we define τ_j in such a way that $\eta_j \cdot \tau_j = 0$. The fastest waves propagate along the η_j direction with speeds $\boldsymbol{S}_j$ and within the time interval $\boldsymbol{T} = \Delta t = t^{n+1} - t^n$ they are contained in the polygon bounded by vertexes P_j, defined as the intersection between the lines orthogonal to η_j and located at a distance $d_j = \boldsymbol{S}_j \boldsymbol{T}$ from vertex k along direction η_j. These wavefronts define a polygonal area Ω_{HLL} which circumscribes the strongly interacting state and which evolves in time. In the space-time coordinate system it forms an inverted prism, as depicted in Figure 3.3.

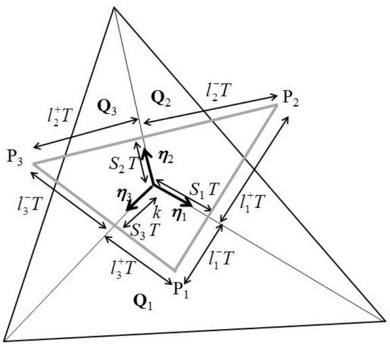


Figure 3.2: Multidimensional Riemann problem at vertex k, where three different states $(\boldsymbol{Q}_1, \boldsymbol{Q}_2, \boldsymbol{Q}_3)$ come together. The control volume generated by the propagation of the wavespeeds $(\boldsymbol{S}_1, \boldsymbol{S}_2, \boldsymbol{S}_3)$ within a time step Δt is highlighted by the grey lines.

The multidimensional state $\boldsymbol{Q}^*$ can be computed following three main steps:

1. first we solve the one-dimensional Riemann problems perpendicular to η_j, hence along the τ_j directions. For this purpose we adopt a rotated reference system to solve the one-dimensional Riemann problems arising at each side j using a classical one-dimensional HLL solver. The dark shaded areas on the side panels of Figure 3.3 represent the resolved one-dimensional states;

2. the interacting state $\boldsymbol{Q}^*$ should then fully contain all the wave speeds starting from vertex k, originated from all the one-dimensional Riemann problems resolved during step 1. Thus, we use the wave speeds to obtain the *multidimensional wave model* as shown in Figure 3.3 and the extremal wavefronts move with speed $\boldsymbol{S}_j$;

3. finally, the two-dimensional conservation law (2.1) is integrated over the three-dimensional prism in space-time (an inverted triangular pyramid in Figure 3.3) in order to calculate the strongly interacting multidimensional HLL state $\boldsymbol{Q}^*$.

The details of the computation of the above reported steps can be found in [88] and [17,18]. The final value of the velocity vector $\boldsymbol{V}_k^n$ for node k is then easily extracted from the multidimensional state $\boldsymbol{Q}^*$. Also in this case, the final time-averaged node velocity $\boldsymbol{V}_k^n$ is obtained by Gaussian quadrature in time. For this node solver algorithm only the two-dimensional version has been developed so far, therefore it will be employed for numerical applications on unstructured triangular meshes.

3.2 The rezoning step

The Lagrangian step allows the nodes to follow the fluid motion as closely as possible. However, this may lead to bad quality elements, where the Jacobians become very small or even negative. This either drastically decreases the admissible timestep, according to (2.19), or even leads to a failure of the computation. Therefore, also a *rezoned* position should be computed for each node k in order to improve the *local* mesh quality without taking into account any physical information. We use a different treatment for internal nodes and boundary nodes. Specifically, the rezoning algorithm presented in [130,154] is adopted for inner nodes, while a variant of the feasible set method proposed by Berndt et al. [36] is used for the boundary nodes.

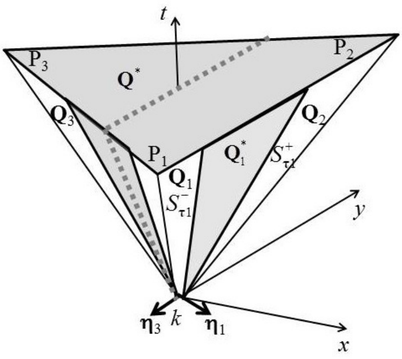


Figure 3.3: Inverted prism in space and time where the strongly interacting state $\boldsymbol{Q}^*$ is on the top surface. Along the side panels are depicted the one-dimensional Riemann problems.

The rezoning algorithm aims at improving the mesh quality *locally*, i.e. in the Voronoi neighborhood $\mathcal{V}_k$ of node k, considering all the neighbor elements T_j^{n+1}, which for sake of simplicity will be addressed by j. The starting point is the Lagrangian coordinate vector $\mathbf{X}_k^{Lag}$ obtained at the end of the Lagrangian step, illustrated in Section 3.1. The rezoning procedure consists in optimizing a goal function $\mathcal{K}_k$ that has to be defined for each node k as

$$\mathcal{K}_k = \sum_{T_j^{n+1} \in \mathcal{V}_k} \kappa_j, \tag{3.10}$$

where κ_j is the condition number of the Jacobian matrix $\boldsymbol{J}_j$ of the mapping from the reference tetrahedron to the physical element j:

$$\boldsymbol{J}_j = \left(\begin{array}{ccc} x_{j,2} - x_k & y_{j,2} - y_k & z_{j,2} - z_k \\ x_{j,3} - x_k & y_{j,3} - y_k & z_{j,3} - z_k \\ x_{j,4} - x_k & y_{j,4} - y_k & z_{j,4} - z_k \end{array} \right). \tag{3.11}$$

In (3.11) the coordinate vector $\mathbf{x}_{j,l} = (x_{j,l}, y_{j,l}, z_{j,l})$ represents the four nodes $l = 1, 2, 3, 4$ of the neighbor tetrahedron T_j^{n+1}, which are counterclockwise ordered in such a way that node k corresponds to $l = 1$. Note that this is a local connectivity *related* to node k, hence it may be different from the standard connectivity given by (2.4). Then, the condition number of matrix $\boldsymbol{J}_j$ is given by

$$\kappa_j = \left\|\boldsymbol{J}_j^{-1}\right\| \left\|\boldsymbol{J}_j\right\|. \tag{3.12}$$

The goal function $\mathcal{K}_k$ is computed according to [154] as the sum of the local condition numbers of the neighbors, see Eqn. (3.10), and its minimization leads to a *locally* optimal position of the free node k. As proposed in [130], the optimized *rezoned coordinates* $\mathbf{x}_k^{Rez}$ for vertex k are computed using the first step of a Newton algorithm, hence

$$\mathbf{x}_k^{Rez} = \mathbf{x}_k^{Lag} - \boldsymbol{H}_k^{-1}\left(\mathcal{K}_k\right) \cdot \nabla\mathcal{K}_k, \tag{3.13}$$

where $\boldsymbol{H}_k$ and $\nabla\mathcal{K}_k$ represent the Hessian and the gradient of the goal function $\mathcal{K}_k$, respectively:

$$\boldsymbol{H}_k = \sum_{T_j^{n+1} \in \mathcal{V}_k} \begin{pmatrix} \frac{\partial^2 \kappa_j}{\partial x^2} & \frac{\partial^2 \kappa_j}{\partial x \partial y} & \frac{\partial^2 \kappa_j}{\partial x \partial z} \\ \frac{\partial^2 \kappa_j}{\partial y \partial x} & \frac{\partial^2 \kappa_j}{\partial y^2} & \frac{\partial^2 \kappa_j}{\partial y \partial z} \\ \frac{\partial^2 \kappa_j}{\partial z \partial x} & \frac{\partial^2 \kappa_j}{\partial z \partial y} & \frac{\partial^2 \kappa_j}{\partial z^2} \end{pmatrix}, \quad \nabla\mathcal{K}_k = \sum_{T_j^{n+1} \in \mathcal{V}_k} \left(\frac{\partial \kappa_j}{\partial x}, \frac{\partial \kappa_j}{\partial y}, \frac{\partial \kappa_j}{\partial z}\right). \tag{3.14}$$

For the boundary nodes we present a simplified but very efficient version of the feasible set method proposed in [36] for two-dimensional unstructured meshes. The original feasible set method has been designed in order to find the convex polygon on which a vertex can lie without invalid elements in its neighborhood. In three space dimensions such an algorithm becomes very complex and highly demanding in terms of computational efforts. In our simplified procedure the rezoned coordinates $\mathbf{x}_k^{Rez,b}$ of the boundary node k are evaluated as a volume weighted average among the barycenter coordinates $\mathbf{x}_{c,j}^{Lag}$ of each neighbor element j, which have been projected onto the boundary face. Hence,

$$\mathbf{x}_k^{Rez,b} = \frac{1}{\alpha_k} \sum_{T_j^{n+1} \in \mathcal{V}_k} \mathbf{x}_{c,j}^{Lag} \cdot \alpha_{k,j}, \tag{3.15}$$

with the weights

$$\alpha_{k,j} = |T_j^{n+1}|, \qquad \alpha_k = \sum_{T_j^{n+1} \in \mathcal{V}_k} \alpha_{k,j} \tag{3.16}$$

and the barycenter defined as usual as

$$\mathbf{x}_{c,j}^{n+1} = \frac{1}{d+1} \sum \mathbf{x}_k^{Lag}. \tag{3.17}$$

3.3 The relaxation step

Since our ALE scheme is supposed to be as Lagrangian as possible, we do not want to rezone the mesh nodes where it is not strictly necessary in order to carry on with the computation. Therefore the *final node position* $\boldsymbol{X}_k^{n+1}$ is obtained applying the relaxation algorithm of Galera et al. [130], that performs a convex combination between the Lagrangian position and the rezoned position of node k, hence

$$\boldsymbol{X}_k^{n+1} = \boldsymbol{X}_k^{Lag} + \omega_k \left(\boldsymbol{X}_k^{Rez} - \boldsymbol{X}_k^{Lag} \right), \tag{3.18}$$

where ω_k is a node-based coefficient associated to the deformation of the Lagrangian grid over the time step Δt. The values for ω_k are bounded in the interval $[0,1]$, so that when $\omega_k = 0$ a fully Lagrangian mesh motion occurs, while if $\omega_k = 1$ the new node location is defined by the pure rezoned coordinates $\boldsymbol{X}_k^{Rez}$. We point out that the coefficient ω_k is designed to result in $\omega_k = 0$ for rigid body motion, namely rigid translation and rigid rotation, where no element deformation occurs.
Let $\boldsymbol{X}^0$ and $\boldsymbol{X}^1$ be the spatial coordinates of a generic point associated to the Lagrangian mesh at time t^n and t^{n+1}, that is $\boldsymbol{X}^0 = \boldsymbol{X}_k^n$ and $\boldsymbol{X}^1 = \boldsymbol{X}_k^{Lag}$ for vertex k. We can define a map $\boldsymbol{R}$ of these two flow configurations that yields

$$\boldsymbol{X}^1 = \boldsymbol{R}\left(\boldsymbol{X}^0, \Delta t\right), \qquad F = \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{X}^0}, \tag{3.19}$$

with F denoting the associated Jacobian matrix, which is also called the deformation gradient tensor. Let us introduce the determinant J of matrix F, i.e. $J = |F|$, and the right Cauchy-Green strain tensor C that is evaluated by right-multiplying matrix F by its transpose as $C = F^t F$. This symmetric positive definite tensor has dimensions (d x d) and admits a total number $N_\lambda = d$ of eigenvalues λ_q which may be interpreted as the rates of dilation in the eigenvectors directions of the transformation from $\boldsymbol{X}^0$ to $\boldsymbol{X}^1$.
According to [130], we adopt a node-centered approximation for the Cauchy-Green tensor by considering the change of surface ($d = 3$) or length ($d = 2$) between the two Lagrangian configurations. Therefore

$$J \left(F^{-1}\right)^t \boldsymbol{n}^0 \, dS^0 = \boldsymbol{n}^1 \, dS^1, \tag{3.20}$$

with dS^0, dS^1 representing the element surfaces in the two configurations and $\boldsymbol{n}^0, \boldsymbol{n}^1$ their associated outward pointing unit normal vectors. Expression (3.20) can be shortened to

$$K\boldsymbol{n}^0\, dS^0 = \boldsymbol{n}^1\, dS^1, \tag{3.21}$$

by introducing the abbreviation

$$K = J\left(F^{-1}\right)^t. \tag{3.22}$$

From (3.21) we can compute matrix K, which will be used later to obtain an approximation for the Jacobian matrix F. Finally we will be able to obtain an approximation for the Cauchy-Green strain tensor C.
To this end we assemble matrix $\mathcal{A}_k$ and matrix $\mathcal{B}_k$ *for each node k* of the computational mesh. They are given by

$$\begin{aligned} \mathcal{A}_k &= \sum_{T_j^{n+1}\in\mathcal{V}_k} \left[S_{jR}\left(\boldsymbol{n}_{jR}\otimes\boldsymbol{n}_{jR}\right) + S_{jL}\left(\boldsymbol{n}_{jL}\otimes\boldsymbol{n}_{jL}\right) + S_{jB}\left(\boldsymbol{n}_{jB}\otimes\boldsymbol{n}_{jB}\right)\right]^0, \\ \mathcal{B}_k &= \sum_{T_j^{n+1}\in\mathcal{V}_k} \left[S_{jR}\left(\boldsymbol{n}_{jR}\otimes\boldsymbol{n}_{jR}\right) + S_{jL}\left(\boldsymbol{n}_{jL}\otimes\boldsymbol{n}_{jL}\right) + S_{jB}\left(\boldsymbol{n}_{jB}\otimes\boldsymbol{n}_{jB}\right)\right]^1. \end{aligned} \tag{3.23}$$

Inserting the above definitions (3.23) in (3.21) and solving for the unknown matrix K_k, one simply gets

$$K_k = \mathcal{A}_k^{-1}\mathcal{B}_k, \tag{3.24}$$

which allows the node Jacobian matrix F_k to be determined from (3.22). The Cauchy-Green tensor centered at vertex k is then obtained by computing

$$C_k = F_k^t F_k. \tag{3.25}$$

Since C_k is symmetric and positive definite, we compute the real positive eigenvalues $\lambda_{i,k}$ with $i \in [1, d]$ and we describe the deformation of the mesh associated to node k with a parameter α_k, which according to [130], reads

$$\alpha_k = \frac{\min \lambda_{i,k}}{\max \lambda_{i,k}}. \tag{3.26}$$

Finally, the relaxation factor ω_k is given by

$$\omega_k = 1 - \frac{\alpha_k - \alpha_{min}}{1 - \alpha_{min}}, \tag{3.27}$$

where α_{min} is the minimum value of α_k among all the vertices of the mesh. Further details about the computation of ω_k can be found in [130].

4 Algorithm Efficiency Improvements

In previous chapters we have designed and presented a new family of direct Arbitrary-Lagrangian-Eulerian (ALE) finite volume schemes, called *direct ALE ADER-WENO* schemes, where high order of accuracy in time is obtained by using a local space-time Galerkin predictor on moving curved meshes, while a high order accurate nonlinear WENO method is adopted to produce high order essentially non-oscillatory reconstruction polynomials in space. The mesh is moved at each time step according to the solution of a node solver algorithm that assigns a unique velocity vector to each node of the mesh. A rezoning procedure can also be applied when mesh distortions and deformations become too severe. The space-time mesh is then constructed by *straight* edges connecting the vertex positions at the old time level t^n with the new ones at the next time level t^{n+1}, yielding closed space-time control volumes, on the boundary of which the numerical flux must be integrated.
The entire algorithm can be divided into three main parts, namely the WENO reconstruction, the local space-time predictor and the numerical flux evaluation. In order to investigate the efficiency of our numerical method, we perform the simulation of a very simple test problem, i.e. the three-dimensional smooth isentropic vortex described later in Chapter 5 (Section 5.1.1). We run the second, third and fourth order accurate version of the numerical scheme and we measure the computational cost of each part of the algorithm, which is reported in Table 4.1.
The most expensive part of the algorithm is the flux evaluation, since in the Lagrangian framework no quadrature-free approach is in principle possible, due to the continuous evolution of the geometry configuration that does not allow the flux computation to be treated as done for the Eulerian case in [101], where the space-time basis used for the flux integrals in (2.61) and (2.68) are integrated on the reference space-time element in a pre-processing step and stored only once. As the order of accuracy increases the relative cost of the WENO reconstruction procedure also increases because the reconstruction stencils become larger, while the local space-time predictor step is the least expensive part of the whole algorithm.
In this chapter we will present and discuss some modifications of the direct

Table 4.1: Computational cost of the second, third and fourth order version of the direct ALE WENO finite volume schemes presented in this work. The times used for the WENO reconstruction, the local space-time predictor and the flux evaluation are given in percentage w.r.t. the total time of the computation. The isentropic vortex test case (Section 5.1.1 of Chapter 5) has been used on a coarse grid with 60157 tetrahedra until the final time $t_f = 1.0$. The simulation has run in parallel on four Intel Core i7-2600 CPUs with a clock-speed of 3.40GHz.

Component of the algorithm	$\mathcal{O}(2)$	$\mathcal{O}(3)$	$\mathcal{O}(4)$
WENO Reconstruction	22 %	30 %	40 %
Space-Time Predictor	5 %	9 %	3 %
Flux Evaluation	73 %	61 %	57 %
Total time $[s]$	135	423	2040

ALE WENO algorithm that have been designed starting from the analysis and the data highlighted in Table 4.1. Specifically, the following strategies have been investigated in order to improve the overall algorithm efficiency:

- in Section 4.1 we propose a local time stepping (LTS) algorithm for moving unstructured triangular meshes, where each element of the mesh has to obey only a less restrictive *local* CFL stability condition, hence using its own optimal local timestep to reach the final time of the simulation. The new algorithm illustrated in Section 4.1 is based on a non-conforming mesh in time, with hanging nodes that are continuously moving and in principle never match the same time level, unless either an intermediate output time or the final time of the simulation is reached. As a consequence, the reconstruction is carried out locally, i.e. within each control volume, using a virtual geometry and a virtual set of cell averages of the surrounding elements that are both computed using the high order space-time predictor solution;

- then, in Section 4.2, we use the genuinely multidimensional HLL Riemann solvers developed by Balsara et al. in [88] as a building block for genuinely multidimensional numerical flux evaluation that allows the scheme to run with *larger time steps* compared to conventional finite volume schemes that use classical one-dimensional Riemann solvers in normal direction. The space-time flux integral computation is carried out at the boundaries of each triangular space-time control volume using the *Simpson* quadrature rule in space and Gauss-Legendre quadrature in time. A brief description of the multidimensional HLL Riemann solvers has already been provided in Section 3.1.3 of Chapter 3, where they have been used as a node solver algorithm;

- a new and efficient *quadrature-free* approach for the numerical flux integration is then presented in Section 4.3. The space-time boundaries of the space-time control volumes are split into *simplex* sub-elements, i.e. either triangles in 2D or tetrahedra in 3D, hence leading to space-time normal vectors as well as Jacobian matrices that are *constant* within each sub-element. Within the space-time Galerkin predictor stage (see Section 2.3) that solves the Cauchy problem inside each element in the small, the discrete solution and the flux tensor are approximated using a nodal space-time basis. Since these space-time basis functions are defined on a reference element and do not change, their integrals over the simplex sub-surfaces of the space-time reference control volume can be integrated once and for all *analytically* during a pre-processing step. The resulting integrals are then used together with the space-time degrees of freedom of the predictor in order to compute the numerical flux that is needed in the finite volume scheme;

- Section 4.4 is devoted to the improvement of the reconstruction part of the algorithm. The expensive WENO approach on moving meshes, used to obtain high order of accuracy in space, is replaced by the very recent *a posteriori* MOOD paradigm [69, 79, 81, 165] which is shown to be less expensive but still as accurate. This a posteriori MOOD strategy ensures the numerical solution in each cell at any discrete time level to fulfill a set of user-defined detection criteria. If one cell value is not satisfying the detection criteria, then the solution is locally re-computed by progressively decrementing the order of the polynomial reconstructions, following the so-called *cascade* of schemes. A very robust scheme is employed as a last resort for genuinely problematic cells. The cascade of schemes defines how the decrementing process is carried out, i.e. how many schemes are tried and which orders are adopted for the polynomial reconstructions. Furthermore the iterative MOOD loop allows the numerical solution to maintain some interesting properties such as positivity, mesh validity, *etc.*

4.1 Time-accurate local time stepping on moving meshes

Almost all algorithms, including the direct ALE ADER-WENO finite volume methods presented in Chapter 2, use an explicit *global* time stepping scheme in which the timestep is computed under a classical *global* CFL stability condition according to (2.19), so that the timestep is essentially determined by the smallest control volume appearing in the mesh. In the Lagrangian context, where the mesh follows as closely as possible the local fluid motion, very severe deformations and distortions may occur in the computational cells, especially at shocks and shear waves. As a consequence, the computational efficiency of the algorithm drastically decreases, because the smallest timestep imposed by the most deformed control volumes dictates the timestep for the entire computational grid, including those elements which are much bigger or which lie in a zone where the fluid is moving uniformly.
In the Eulerian framework such a problem can be partially avoided controlling the mesh quality *a priori* and designing a high quality mesh once in a preprocessing step, since the grid will not change anymore during the simulation. Of course, the CFL condition can be circumvented by using implicit or semi-implicit schemes, see for example [42, 60, 61, 63, 85–87], but this approach does not yet seem to be very popular in the context of cell-centered Lagrangian-type finite volume methods. An alternative to overcome the global CFL condition consists in the development of numerical schemes that allow for time-accurate *local* time stepping (LTS), where each element has to obey only a less restrictive *local* CFL stability condition, hence using its own optimal local timestep. Therefore, many efforts have been devoted to the construction of high order accurate Eulerian schemes with time-accurate LTS, developing either discontinuous Galerkin finite element methods [104,122,123,131,155,164,220] or high order accurate finite volume schemes with LTS [13, 14, 33, 34, 49, 55, 100, 110, 122, 237]. The finite volume schemes with LTS adopt mainly classical adaptive mesh refinement (AMR) techniques in space and time or block-clustered local time stepping algorithms. In [136, 137] also high order accurate Runge-Kutta time integrators with local time stepping (so-called multi-rate integrators) can be found. To our knowledge, the first high order accurate *Lagrangian-like* algorithm with *time accurate local time stepping* on moving grids has been proposed very recently in [93], where the equations of hydrodynamics and of classical magnetohydrodynamics (MHD) have been solved in one spatial dimension. In the following we extend the algorithm presented in [93] to *moving unstructured triangular meshes*.
The finite volume framework of the LTS algorithm is the same described in

Section 2.1 of Chapter 2, hence storing and evolving data according to (2.5). In the time-accurate LTS algorithm a cell T_i^n is allowed to evolve the solution in time only if the so-called *update criterion* [93,103,164] is satisfied, namely if

$$\max_{j\in\mathcal{N}_i}\left(t_j^n\right) \leq \left(t_i^n + \Delta t_i^n\right) \leq \min_{j\in\mathcal{N}_i}\left(t_j^n + \Delta t_j^n\right), \tag{4.1}$$

where $\mathcal{N}_i$ denotes the *Neumann neighborhood* of element T_i, i.e. the three direct side neighbors T_j of the cell, while t_i^n and Δt_i^n represent the current local time and the local timestep of triangle T_i, respectively. Hence, $(t_i^n + \Delta t_i^n)$ is the future time of element T_i and to make notation easier it will be addressed with t_i^{n+1}.
There are two important issues that need to be clarified:

1. in order to develop a numerical scheme that evolves the cell averages (2.5) with high order of accuracy in space and in time in one single step, two strategies are followed. For the accuracy in space we implement a suitable Weighted Essentially Non-Oscillatory (WENO) reconstruction technique that is able to deal with LTS and which is presented in detail in the next Section 4.1.1, while for the accuracy in time we use the element-local space-time Galerkin predictor approach, already illustrated in Section 2.3 of Chapter 2;

2. in a time-accurate LTS finite volume scheme, each element T_i^n evolves the solution $\boldsymbol{Q}_i^n$ in time with a local timestep Δt_i^n that is computed according to a local CFL stability condition. As a result, the WENO reconstruction will be carried out *locally*, i.e. considering only the element T_i^n which is currently updating the solution to its new time level t_i^{n+1}, as well as an appropriate neighborhood of T_i^n that is necessary to carry out the reconstruction, the so-called reconstruction stencil $\mathcal{S}_i^W$. Since the neighbor elements of T_i in general have a different local time, the reconstruction needs to get time-accurate *virtual* cell averages from the neighbor cells as input. These virtual cell averages are readily available from the local space-time Galerkin predictor solution inside the neighbors.

4.1.1 High order WENO reconstruction for local time stepping

All the details of the high order WENO reconstruction procedure are contained in Section 2.2 of Chapter 2, therefore we present here only a brief summary of the main features of the scheme, highlighting the modifications that are necessary to handle a time accurate local time stepping formulation.

The reconstructed solution $\mathbf{w}_h(\mathbf{x}, t_i^n)$ is given again in terms of piecewise polynomials of degree M and is computed *locally* for each control volume T_i^n. First, one has to construct a set of reconstruction stencils S_i^s relative to the element T_i, namely

$$\mathcal{S}_i^s = \bigcup_{j=1}^{n_e} T_{m(j)}, \tag{4.2}$$

where $1 \leq j \leq n_e$ denotes a local index which counts the elements belonging to the stencil, while $m(j)$ maps the local counter j to the global element number used in the triangulation (2.2). According to [29,102,152,186] on unstructured two-dimensional meshes we set $n_e = 2\mathcal{M}$, where $\mathcal{M} = (M+1)(M+2)/2$ represents the smallest number of elements needed to reach the formal order of accuracy $M+1$. Moreover we need a total number of stencils $s = 7$ [102,152], hence the update criterion (4.1) must be extended to the *total* WENO stencil $\mathcal{S}_i^W$ given by

$$\mathcal{S}_i^W = \bigcup_{s=1}^{7} \mathcal{S}_i^s, \tag{4.3}$$

hence obtaining

$$\max\left(t_j^n\right) \leq t_i^{n+1} \leq \min\left(t_j^{n+1}\right), \qquad \forall T_j \in \mathcal{S}_i^W. \tag{4.4}$$

In order to guarantee that at least one element in the entire mesh satisfies condition (4.4), the total stencils $\mathcal{S}_i^W$ need to be constructed in such a way that they are *symmetric*, i.e. each element $T_j \in \mathcal{S}_i^W$ inside the stencil of T_i *must* contain in its own WENO stencil $\mathcal{S}_j^W$ the element T_i. In other words, if $T_j \in \mathcal{S}_i^W$ then $T_i \in \mathcal{S}_j^W$. It is always possible to construct such symmetric stencils by adding elements to the stencils until the condition of symmetry is satisfied for all elements.
For the sake of clarity we give a simple example of what could happen if we take *non-symmetric* stencils. Let element T_j be *not* contained in the stencil of T_i and let T_i belong to the stencil $\mathcal{S}_j^W$ of element T_j. Let furthermore the current time level of T_i and T_j be t_i^n and t_j^n, respectively, with the corresponding future times t_i^{n+1} and t_j^{n+1}. Without loss of generality we assume $t_i^n = t_j^n$, while the future time levels are chosen such that $t_i^{n+1} > t_j^{n+1}$. If the update criterion on the *non-symmetric* stencil $\mathcal{S}_i^W$ is supposed to be satisfied, then element T_i is allowed to update the numerical solution to its future time, which will subsequently become the *current* time of T_i, i.e. $t_i^n \to t_i^{n+1}$. The resulting situation will lead to a *dead lock* in the algorithm, where element T_j will never

obey condition (4.4) since $t_j^{n+1} < t_i^n$. A simple solution is to always build a *symmetric* stencil. In this case element T_j performs the update *first* and does not prevent element T_i from updating its solution. The drawback of this approach is that slightly larger stencils are required.
Due to (4.4), the current time t_j^n of the neighbor elements belonging to the WENO stencil $\mathcal{S}_i^W$ must be lower than the current time level t_i^n of the triangle T_i for which the reconstruction has to be performed. Moreover, in Lagrangian algorithms the mesh is moving in time, therefore the local WENO reconstruction is carried out on a *virtual geometry* with *virtual cell averages*, as suggested in [93]. These virtual cell averages, which are needed for the reconstruction, are obtained from the local space-time predictor solution $\boldsymbol{q}_h(\mathbf{x}, t_i^n)$ inside the neighbor elements T_j using a simple integral *projection* (averaging). The way how this predictor solution is computed has been described in Section 2.3 of Chapter 2. A similar projection is used also for the virtual geometry of the elements inside the total WENO stencil, where all elements $T_j^n \in \mathcal{S}_i^W$ are moved *virtually* until time t_i^n is reached. We emphasize that the projection of the stencil geometry and of the cell averages is done only virtually, just for the purpose of reconstruction, because the real mesh motion and the real conservative update of the cell averages will be performed individually by each element at its scheduled time according to the update criterion (4.4). The geometry of each stencil element T_j^n, i.e. the vertex coordinates, are projected and also all the other geometric quantities used for the computation, e.g. normal vectors, volumes, side lengths, *etc.*. For the sake of clarity, the projected quantities will be denoted by a tilde symbol in the following, hence

$$\tilde{\boldsymbol{X}}_{k,j}^{n+1} = \boldsymbol{X}_{k,j}^n + \left(t_i^n - t_j^n\right)\ \boldsymbol{V}_{k,j}^n, \qquad \forall T_j^n \in \mathcal{S}_i^W, \quad k = 1,2,3 \tag{4.5}$$

and

$$\tilde{\boldsymbol{Q}}_j^n = \left\{ \begin{array}{ccc} \boldsymbol{Q}_i^n, & \text{if} & j = i, \\ \frac{1}{|\tilde{T}_j^n|} \int_{\tilde{T}_j^n} \boldsymbol{q}_h(\mathbf{x}, t_i^n) dV, & \text{if} & j \neq i, \end{array} \right. \qquad \forall T_j \in \mathcal{S}_i^W. \tag{4.6}$$

In (4.5) the time-averaged node velocity $\boldsymbol{V}_{k,j}^n$ is computed according to the node solver algorithm, see also Chapter 3, which will be briefly described in Section 4.1.2, while in (4.6) the virtual cell averages $\tilde{\boldsymbol{Q}}_j^n$ of the neighbor elements are given as the spatial integral of the predicted solution at time t_i^n over the virtual control volumes $\tilde{T}_j^n$.
Once the virtual geometry and the virtual cell averages have been computed for the entire stencil $\mathcal{S}_i^W$, we are in the position to carry out the *local* high

order WENO reconstruction procedure. To obtain the reconstruction polynomial $\mathbf{w}_h(\mathbf{x}, t_i^n)$, integral conservation of the projected cell averages $\tilde{\boldsymbol{Q}}_j^n$ in each reconstruction stencil $\mathcal{S}_i^s$ is required, i.e.

$$\frac{1}{|\tilde{T}_j^n|} \int\limits_{\tilde{T}_j^n} \mathbf{w}_h^s(\mathbf{x}, t_i^n) dV = \frac{1}{|\tilde{T}_j^n|} \int\limits_{\tilde{T}_j^n} \psi_l(\xi, \eta) \hat{\mathbf{w}}_{l,i}^{n,s} = \tilde{\boldsymbol{Q}}_j^n, \qquad \forall T_j^n \in \mathcal{S}_i^s, \tag{4.7}$$

which is the same equation of (2.9) presented in Section 2.2 of Chapter 2. According to (2.8) the reconstruction polynomial on each stencil is expressed in terms of a set of orthogonal spatial basis functions $\psi_l(\xi, \eta)$ on the reference element [72,90,149] and $\mathcal{M}$ unknown degrees of freedom $\hat{\mathbf{w}}_{l,i}^{n,s}$. The linear system (4.7) is solved using the same procedure described in Section 2.2, hence obtaining the final nonlinear WENO reconstruction polynomial and its coefficients according to (2.13) with the definitions (2.12).

4.1.2 Mesh motion with local time stepping

A high order time accurate predictor for the numerical solution is obtained by the local space-time Galerkin predictor technique (see Section 2.3 in Chapter 2), hence producing piecewise space-time polynomials $\boldsymbol{q}_h(\mathbf{x}, t)$ of degree M. The local time evolution is carried out within each control volume from the current time level t_i^n of element T_i up to its next time level $t_i^{n+1} = t_i^n + \Delta t_i^n$.
At the end of the local predictor stage each node k of the computational mesh needs to be assigned a *uniquely* defined velocity vector. The Voronoi neighborhood $\mathcal{V}_k$ of node k is composed by all those elements T_j which share the node k. The node k will be moved each time the update criterion (4.4) is satisfied by one element $T_i \in \mathcal{V}_k$. Therefore the future time to which node k moves will coincide with the future time t_i^{n+1} of that element T_i.
Here we consider the node solver $\mathcal{NS}_{cs}$, which has been detailed in Section 3.1.1 of Chapter 3. It adopts the idea of Cheng and Shu [65,163] and the node velocity is computed according to (3.2). The local weights $\mu_{k,j}$ are obtained by multiplying the cell averages of the density ρ_j with the cell area $|T_j|$ at the current neighbor time level t_j^n.
The mesh motion plays an important role in Lagrangian schemes, because it allows interfaces and shear waves to be precisely identified. For this reason an accurate computation of the node velocity represents a crucial step and in our approach the local velocity contributions $\boldsymbol{V}_{k,j}^n$ are taken to be the time integrals of the high order vertex-extrapolated velocities at node k. We can use the space-time reference system $\xi - \eta - \tau$ and the velocity approximation

given by (2.30) to evaluate the time integral. Since each node k can be moved by any of the Voronoi neighbors T_j, the vertex time level of node k is not known *a priori* when an element T_i satisfies (4.4) and is ready to update the geometry. Therefore, it is much more convenient to define a *node time* variable t_k^n, that is independent of the time evolution of the elements and advances in time whenever the node is moved by any of its Voronoi neighbors T_j. As a result, the high order velocity integration for each element $T_j \in \mathcal{V}_k$ must be done within the time interval $\Delta t_k = [t_k^n, t_k^{n+1}]$, that has to be *rescaled* to the corresponding reference time interval $\Delta \tau_k = [\tau_{k,j}^0, \tau_{k,j}^1]$ as

$$\tau_{k,j}^0 = \frac{t_k^n - t_j^n}{\Delta t_j^n} \qquad \tau_{k,j}^1 = \frac{t_k^{n+1} - t_j^n}{\Delta t_j^n}, \qquad \forall T_j \in \mathcal{V}_k, \tag{4.8}$$

where Δt_j^n is the local timestep of element T_j. Recall that $t_k^{n+1} = t_i^{n+1}$, if the node is moved by element T_i which is supposed to satisfy the update criterion. Finally the local velocity contributions $\boldsymbol{V}_{k,j}^n$ are given by

$$\boldsymbol{V}_{k,j}^n = \left(\int\limits_{\tau_{k,j}^0}^{\tau_{k,j}^1} \theta_l(\xi_{m(k)}^e, \eta_{m(k)}^e, \tau) d\tau \right) \widehat{\boldsymbol{V}}_{l,j}, \tag{4.9}$$

where $m(k)$ is a mapping from the global node number k to the local node number in element T_j according to (2.4), while ξ_m^e and η_m^e represent the coordinates of the vertices of the reference triangle in space. $\widehat{\boldsymbol{V}}_{l,j}$ are the space-time degrees of freedom which are *known* from the local space-time predictor solution $\boldsymbol{q}_{h,j}$. Since no rezoning nor relaxation step is considered within the framework of the local time stepping algorithm, each node k belonging to element T_i is finally moved to the new position $\boldsymbol{X}_k^{n+1}$ with (3.1).

4.1.3 Finite volume scheme with local time stepping

In this section we consider conservative hyperbolic balance laws, hence the vector of conserved variables $\boldsymbol{Q}_i^n$ is evolved to its own next time level t_i^{n+1} according to (2.67), which carries on the space-time flux integration over the space-time volume shown in Figure 4.1. The evolution is done only when element T_i^n obeys the update criterion (4.4).
In the time-accurate local time stepping (LTS) algorithm, when the element T_i is ready to update its numerical solution $\boldsymbol{Q}_i^n$, it might well be the case that the vertices of T_i have already been moved by another element T_j sharing one or

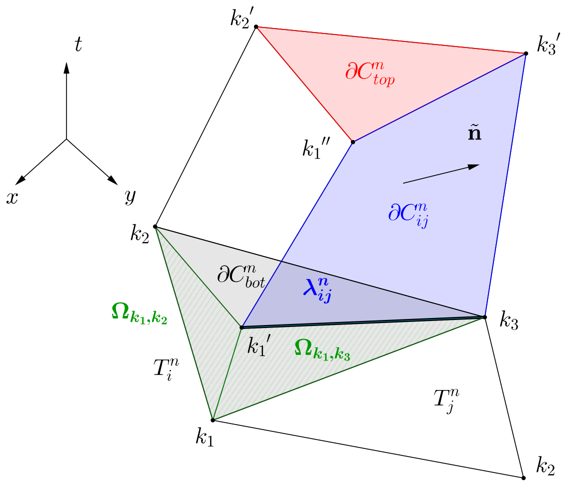


Figure 4.1: Space-time evolution of element T_i from time t_i^n (black triangle) to time t_i^{n+1} (red triangle). The triangular sub-surfaces Ω_{k_1,k_2} and Ω_{k_2,k_3} (already computed in the past by some Voronoi neighbors of the vertices of T_i) are highlighted in green, while the trapezoidal space-time sub-surfaces ∂C^n_{ij} computed with the current element update are highlighted in blue.

more nodes with T_i. This situation generates hanging nodes in time, as shown in Figure 4.1, where vertex k_1 has changed its position to k_1'. In order to design a suitable finite volume scheme on moving meshes with LTS, some parts of the flux integral appearing in (2.67) will be computed using a *memory variable* $\boldsymbol{Q}_i^M$, according to [93]. The memory variable contains all fluxes through the element space-time sub-surfaces ∂C^n_{ij} in the *past*, e.g. the fluxes through the space-time triangular surfaces Ω_{k_1,k_2} and Ω_{k_1,k_3} depicted in Figure 4.1. Therefore, from (2.67) the following high order ALE one-step finite volume scheme with LTS is obtained:

$$|T_i^{n+1}|\,\boldsymbol{Q}_i^{n+1} = |T_i^n|\,\boldsymbol{Q}_i^n \quad - \quad \sum_{T_j\in\mathcal{N}_i}\int_0^1\int_0^1 |\partial C_{ij}^n|\tilde{\boldsymbol{F}}_{ij}\cdot\tilde{\boldsymbol{n}}_{ij}\,d\tau d\chi_1$$
$$+ \quad \int_{t_i^n}^{t_i^{n+1}}\int_{T_i(t)} \boldsymbol{S}(\boldsymbol{q}_h)\,d\boldsymbol{x}dt + \boldsymbol{Q}_i^M\,, \qquad (4.10)$$

which reads very similar to expression (2.68) apart from the memory variable $\boldsymbol{Q}_i^M$. In the above equation $|T_i^n|$ and $|T_i^{n+1}|$ represent the surface of triangle T_i at the current and at the future time level, i.e. t_i^n and t_i^{n+1}, and $|\partial C_{ij}^n|$ denote the determinant of the coordinate transformation of each lateral sub-surface ∂C_{ij}^n. Furthermore $\tilde{\boldsymbol{F}}_{ij}\cdot\tilde{\boldsymbol{n}}_{ij}$ is the numerical flux used to resolve the discontinuity of the predictor solution $\boldsymbol{q}_h$ at the space-time sub-face ∂C_{ij}^n. In the finite volume scheme (4.10) the flux integral across the quadrilateral sub-surface ∂C_{ij}^n is computed in an edge-based unit reference system $(\chi_1,\tau)\in[0,1]^2$ (see Figure 2.8) that is linked to the physical coordinates of the four space-time nodes that define ∂C_{ij}^n. Note that in the edge-aligned system the relative time coordinate τ is in general *different* from the ones in the adjacent left and right elements T_i and T_j, respectively, since the two nodes that define the edge may have already been moved before the update of element T_i. Let us denote the common edge between element T_i and $T_j\in\mathcal{N}_i$ with λ_{ij} and the global number of the first node on λ_{ij} with L and the one of the second node on the same edge with R. According to the notation illustrated in Figure 4.1, we would have $L=k_1$ and $R=k_2$, therefore the space-time coordinates of the four space-time nodes defining the sub-surface ∂C_{ij}^n in (4.10) are given by

$$\tilde{\mathbf{x}}_{ij}^1 = (\boldsymbol{X}_L^n, t_L^n)\,, \qquad \tilde{\mathbf{x}}_{ij}^2 = (\boldsymbol{X}_R^n, t_R^n)\,,$$
$$\tilde{\mathbf{x}}_{ij}^3 = \left(\boldsymbol{X}_R^{n+1}, t_R^{n+1}\right)\,, \qquad \tilde{\mathbf{x}}_{ij}^4 = \left(\boldsymbol{X}_L^{n+1}, t_L^{n+1}\right)\,. \qquad (4.11)$$

Note that $L=L(i,j)$ and $R=R(i,j)$ are functions of the numbers of element T_i and the neighbor T_j, respectively, but to ease notation this explicit dependency is dropped. The associated space-time integral of the numerical flux over ∂C_{ij}^n is also called *edge flux* and denoted by $\boldsymbol{E}_{ij}^n$ in the following. The physical times of the four space-time nodes (4.11) have then to be rescaled to each individual reference space-time coordinate system associated with element T_i and its neighbor T_j, respectively, using the time transformation (2.18).

In order to obtain a conservative scheme, the task of the memory variable $\boldsymbol{Q}_i^M$ in (4.10) is to accumulate (sum) all past fluxes through the lateral space-time sub-surfaces, from the current element time t_i^n to the current local node times t_L^n and t_R^n, respectively, see [93]. The edge flux $\boldsymbol{E}_{ij}^n$ through the sub-surface ∂C_{ij}^n is given by

$$\boldsymbol{E}_{ij}^n = \int\limits_{\partial C_{ij}^n} \tilde{\boldsymbol{F}}_{ij} \cdot \tilde{\boldsymbol{n}}_{ij}\, dS = \int\limits_0^1 \int\limits_0^1 |\partial C_{ij}^n| \tilde{\boldsymbol{F}}_{ij} \cdot \tilde{\boldsymbol{n}}_{ij}\, d\tau d\chi. \tag{4.12}$$

Then, if element T_i is updated according to (4.10), the memory variable of the element itself is reset to zero and the memory variables of the *neighbor* elements $T_j \in \mathcal{N}_i$ are updated by summing (accumulating) the contribution of the edge-flux $\boldsymbol{E}_{ij}^n$ to $\boldsymbol{Q}_j^M$. Note that for element T_i the contribution $\boldsymbol{E}_{ij}^n$ has negative sign. Like in the 1D case presented in [93] we therefore have after each update of element T_i:

$$\boldsymbol{Q}_i^M := 0, \qquad \boldsymbol{Q}_j^M := \boldsymbol{Q}_j^M + \boldsymbol{E}_{ij}^n, \qquad \forall T_j \in \mathcal{N}_i. \tag{4.13}$$

The implementation of the finite volume scheme (4.10) requires that a numerical flux is specified through an approximate Riemann solver and we rely either on the Rusanov-type (2.69) or the Osher-type (2.71) numerical flux.
When element T_i performs its local time update, the geometry of cell T_i is also updated, because all three vertices of T_i are moved according to (3.1). Using the memory variable $\boldsymbol{Q}_i^M$ we ensure conservation of the edge-fluxes, since the numerical fluxes computed over the space-time sub-surfaces ∂C_{ij} are immediately saved (with opposite sign) in the memory variables of the neighbor elements $T_j \in \mathcal{N}_i$. While the consideration of edge fluxes is sufficient for the Lagrangian LTS algorithm presented in [93], its extension to moving unstructured triangular meshes requires an important modification due to the increased topological complexity of a two-dimensional mesh. As shown in Figure 4.2, each vertex k of element T_i is shared among the Voronoi neighbors $T_j \in \mathcal{V}_k$. Hence, we must also compute a numerical flux $\boldsymbol{E}_{k,m}$ across each edge defined by the vertices k and m which does *not* belong to element T_i, i.e.

$$\boldsymbol{E}_{k,m} = \int\limits_{\partial\Omega_{k,m}} \tilde{\boldsymbol{F}}_{l,r} \cdot \tilde{\boldsymbol{n}}_{l,r}\, d\tilde{\boldsymbol{x}}. \tag{4.14}$$

This *vertex flux* will also be stored (with the proper sign) in the corresponding memory variables $\boldsymbol{Q}_l^M$ and $\boldsymbol{Q}_r^M$ of elements T_l and T_r, where l denotes the left element and l denotes the right element on the corresponding edge composed of vertices $k - m$, respectively. As shown in Figure 4.2, the numerical flux is integrated over the *triangular* space-time surfaces $\Omega_{j,j+1}$, defined by vertices $(\tilde{\mathbf{x}}(k), \tilde{\mathbf{x}}(k'), \tilde{\mathbf{x}}(m))$, that represent the space-time coordinates of vertex k at the old and at the new time level, and the space-time location of vertex k_m, respectively.

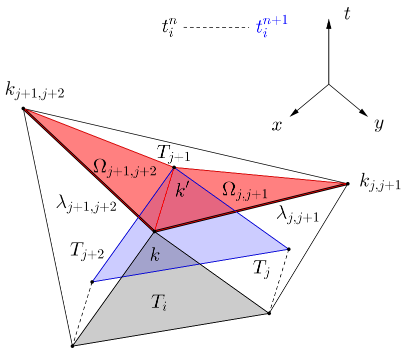


Figure 4.2: Space-time evolution of element T_i from time t_i^n (black triangle) to time t_i^{n+1} (blue triangle). We consider the vertex k shared among elements $T_i, T_j, T_{j+1}, T_{j+2}$. Once element T_i has advanced up to its local next time level t_i^{n+1}, the edge fluxes $\boldsymbol{E}_{k,k_{j,j+1}}$ and $\boldsymbol{E}_{k,k_{j,j+1}}$ are evaluated according to (4.14) by integration over the triangular sub-surfaces $\Omega_{k,k_{j,j+1}}$ and $\Omega_{k,k_{j+1,j+2}}$, which are highlighted in red. For sake of clarity when we consider $\Omega_{k,k_{j,j+1}}$ we set $T_l = T_j$ and $T_r = T_{j+1}$ with $m = k_{j,j+1}$, while for $\Omega_{k,k_{j+1,j+2}}$ it follows that $T_l = T_{j+1}$ and $T_r = T_{j+2}$ with $m = k_{j+1,j+2}$.

In order to verify whether the GCL (2.56) is also satisfied in the practical implementation of our Lagrangian-like LTS algorithm, we need to compute the integral (2.56) whenever element T_i performs an update. For this purpose, we also compute a variable H_i^M that behaves like the memory variable $\boldsymbol{Q}_i^M$, but for the GCL. All past contributions to the integral (2.56) relative to the cell T_i are recorded in the *geometrical memory variable* H_i^M, which is reset to zero when the local timestep procedure has been completed by element T_i. Strictly speaking this this is not needed, since Eqn. (2.56) is always satisfied at the end of a local time step because the final space-time control volume is always closed.

4.1.4 Description of the high order Lagrangian LTS algorithm in multiple space dimensions

The aim of this Section is to give an overall overview of the entire LTS algorithm that has been previously described in all its parts. By placing each portion of the algorithm in a context, this presentation should clarify how the numerical scheme can be practically implemented. Due to the LTS approach, where elements are updated in the order given by the update criterion (4.4), we can no longer speak of *timesteps* but we have to consider *cycles*, as done in [93]. In each cycle the scheme runs over all elements and only those which obey condition (4.4) are allowed to update the numerical solution, while the others are simply skipped to the next cycle. In the *pre-processing phase* all elements of the mesh are assigned with the initial condition of the problem at the common time level $t = 0$, i.e. the cell averages $\boldsymbol{Q}_i^n$ are defined according to (2.5) from the known initial condition. For each element the *first* WENO reconstruction procedure presented in Section 4.1.1 is carried out. Since all elements are at the same time $t = 0$, for this first reconstruction no virtual geometry or virtual cell averages $\tilde{\boldsymbol{Q}}$ are needed. As a result, we obtain the high order spatial polynomial $\boldsymbol{w}_h$ for each element. Then, the *element-local timestep* Δt_i^n is computed for each cell T_i according to a classical CFL stability condition, considering only cell number i and its Neumann neighborhood $\mathcal{N}_i$, i.e.

$$\Delta t_i^n = \min\left(\text{CFL}\,\frac{\tilde{d}_i}{|\tilde{\lambda}_{\max,i}|}, \text{CFL}\,\frac{\tilde{d}_j}{|\tilde{\lambda}_{\max,j}|}\right), \qquad \forall T_j \in \mathcal{N}_i, \tag{4.15}$$

with $\tilde{d}_j = d_j^0$ denoting the incircle diameter of element T_j and $|\tilde{\lambda}_{\max,j}| = |\lambda_{\max,j}|^0$ representing the maximum absolute value of the eigenvalues computed from the initial condition $\tilde{\boldsymbol{Q}}_j = \boldsymbol{Q}_j^0$ in T_j. CFL is the Courant-Friedrichs-Levy number that must satisfy the inequality CFL ≤ 0.5 in the two-dimensional case, as stated in [229]. In the last part of the pre-processing stage, since the local element timestep Δt_i^n as well as the local reconstruction polynomial $\boldsymbol{w}_h$ have already been computed, we are able to carry out the local space-time Galerkin predictor procedure described in Section 2.3 of Chapter 2, which gives the high order local space-time predictor solution $\boldsymbol{q}_h$. All cells are now at the same

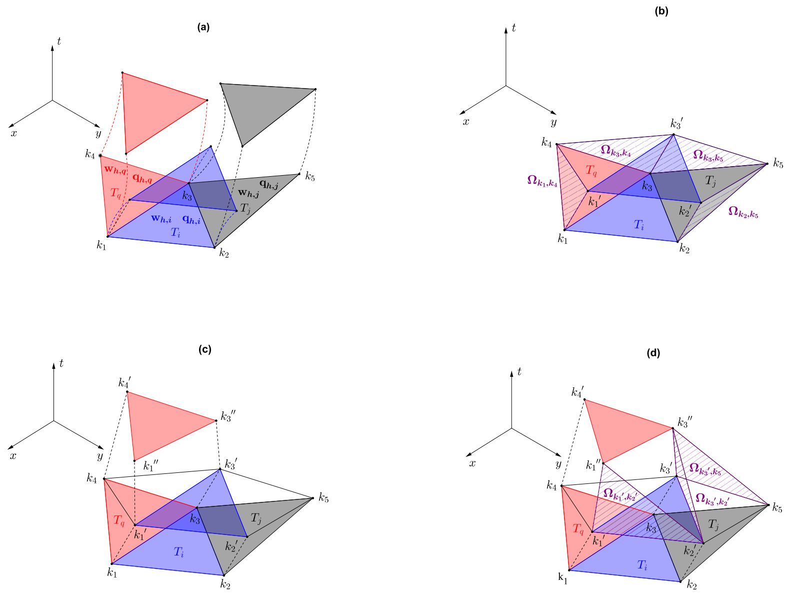


Figure 4.3: Update of element T_i and T_q according to the high order direct ALE LTS algorithm presented in Section 4.1. At the beginning we assume the *same* current time for each element, i.e. $t_i^n = t_j^n = t_q^n = t$. (***a***) At the current time level t each element is given its own reconstruction and predictor solution $\boldsymbol{w}_h$ and $\boldsymbol{q}_h$, respectively. (***b***) Update of element T_i to the new time level t_i^{n+1}. Computation of the necessary edge fluxes with the direct neighbors *and* computation of the associated vertex fluxes $\Omega_{k_1,k_4}, \Omega_{k_3,k_4}, \Omega_{k_3,k_5}, \Omega_{k_2,k_5}$. (***c***) Update of element T_q, where the edge fluxes are evaluated only over the space-time surfaces that exceeds the vertex fluxes previously calculated and stored in the memory variable $\boldsymbol{Q}_q^M$. (***d***) Computation of the vertex fluxes related to the update of element T_q.

current time level $t = 0$ and for each element T_i the local predictor solution $\boldsymbol{q}_h$, the local reconstruction polynomial $\boldsymbol{w}_h$ and the cell average $\boldsymbol{Q}_i^n$ are given (Figure 4.3 (a)). We underline that also each node k of the entire computational mesh is assigned the initial time level $t_k^0 = 0$.
The algorithm proceeds with the *computational phase*, during which each element T_i will reach the imposed final time of the simulation $t = t_f$ in a certain number of necessary cycles, according to its own optimal timestep. The first cycle starts by looping over all elements to check in which elements the update criterion (4.4) is satisfied. If an element T_i obeys condition (4.4), then it performs the local timestep until its future time $t_i^{n+1} = t_i^n + \Delta t_i^n$ (Figure 4.3 (b)) through the following sub-steps:

- *mesh motion*: each vertex k of element T_i is moved to the new position at time $t_k^{n+1} = t_i^{n+1}$ using the node solver algorithm illustrated in Section 4.1.2 and all other geometric quantities of element T_i are also updated;

- *edge flux computation*: we compute the numerical fluxes $\boldsymbol{F}_{ij}^n$ through the quadrilateral space-time sub-surfaces and using the high order Lagrangian-like finite volume scheme (4.10) we obtain the numerical solution $\boldsymbol{Q}_i^{n+1}$. Subsequently, we reset the memory variable of element T_i to zero, i.e. $\boldsymbol{Q}_i^M := 0$ and accumulate the edge-fluxes into the memory variables of the neighbor elements to maintain conservation ($\boldsymbol{Q}_j^M := \boldsymbol{Q}_j^M + \boldsymbol{E}_{ij}^n$). Also the geometry variable H_i^M is reset to zero, after assuring that condition (2.56) is satisfied;

- *vertex flux computation*: as explained in Section 4.1.3, for each vertex k of the element T_i we also need to evaluate for each edge $k - m$ the additional fluxes $\boldsymbol{E}_{k,m}$ using (4.14) (Figure 4.3 (b)). The numerical fluxes evaluated over the space-time triangular sub-surface $\Omega_{j,j+1}$ (see Figure 4.2) are immediately stored into the memory variable of the adjacent elements T_j, T_{j+1}, while the part of the geometry integral (2.56) is stored into H_j^M and H_{j+1}^M. In this way we ensure that the numerical scheme is fully conservative;

- *virtual projection*: all the elements T_j belonging to the entire reconstruction stencil $\boldsymbol{S}_i^W$ of element T_i are now moved *virtually* to the future time level of cell i, i.e. t_i^{n+1}, and also the virtual cell averages $\tilde{\boldsymbol{Q}}_j$ are *estimated* from the local predictor solution $\boldsymbol{q}_h$ in the neighbors T_j;

- *local WENO reconstruction*: once the *virtual* geometry and cell averages have been projected to the future time t_i^{n+1}, the local WENO reconstruction technique described in Section 4.1.1 can be carried out for element T_i, hence obtaining the new reconstruction polynomial $\boldsymbol{w}_h$ at time t_i^{n+1};

- *local timestep computation*: using the virtual geometry and the virtual solution of the Neumann neighbors, the next local timestep Δt_i^{n+1} is evaluated according to (4.15);
- *local space-time predictor*: finally we compute the high order space-time predictor solution $\boldsymbol{q}_h$ valid within the next timestep of element T_i.

This procedure is repeated for all elements, until all of them reach the final time of the simulation t_f. As soon as an element T_i has finished its own computation because it has reached the final time t_f, it is automatically skipped at the beginning of each cycle, waiting for the remaining elements to reach the final time, too.
This brief description summarizes how our high order direct Arbitrary-Lagrangian-Eulerian LTS algorithm is organized. During the simulation hanging nodes in time appear because each node is moved *physically* only by the updating element T_i which the vertex belongs to. As a consequence, the resulting space-time mesh is computed *dynamically*, producing a *non-conforming* space-time mesh. Due to our high order approach, the edge and vertex fluxes have to be evaluated using higher order Gaussian quadrature rules, hence increasing the computational cost. In practical applications, for which first or second order accurate finite volume schemes are considered adequate, one could rely on the fast and simple mid-point rule that would significantly improve the computational efficiency of our LTS algorithm.

4.2 Genuinely multidimensional HLL Riemann solvers for ALE methods

An important branch of research in finite volume methods has put a lot of effort into the introduction of multidimensional effects into Riemann solvers [1–3]. The aim was the formulation of genuinely multidimensional Riemann solvers for the solution of hyperbolic conservation laws of the form (2.1) without taking into account non-conservative products, hence $\boldsymbol{B}(\boldsymbol{Q}) \cdot \nabla \boldsymbol{Q} = 0$. In a series of very recent papers [17,18,25,26,88], Balsara et al. presented multidimensional HLL and HLLC Riemann solvers for hydrodynamics and magnetohydrodynamics on both structured and unstructured meshes. In [88] the multidimensional Riemann solver is designed to work also on moving meshes, incorporating the

mesh velocity in the signal speeds for the Riemann problem. Here, we use the above-mentioned strategy twice in our one-step ALE algorithm: first as a node solver that assigns a unique velocity vector to each vertex, where the node velocity can be directly extracted from the so-called *strongly interacting state*, or, multidimensional HLL state, produced by the multidimensional HLL Riemann solver (see Section 3.1.3); second, the multidimensional HLL Riemann solvers are employed to evaluate the numerical fluxes across element boundaries. Numerical evidence has been shown in [88] that with the use of multidimensional HLL schemes the CFL number can be chosen of the order of unity in two space-dimensions instead of the usual limit of CFL≤ 0.5, see [229].
The new numerical scheme uses the same ingredients provided in Chapter 2, namely the high order WENO reconstruction procedure is performed according to Section 2.2, the element-local Galerkin predictor follows the description given in Section 2.3 and the *conservative* formulation of the governing PDE (2.1) is integrated over the moving space-time control volume $C_i^n = T_i(t) \times \left[t^n; t^{n+1}\right]$, yielding the direct ALE finite volume discretization (2.68).

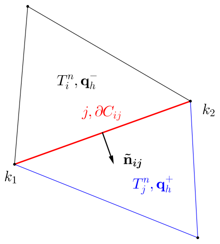


Figure 4.4: Notation used for ALE ADER-WENO finite volume schemes based on genuinely multidimensional HLL Riemann solvers. Element T_i^n and its direct neighbor T_j^n share edge j, which is bounded by vertices k_1 and k_2.

What is different here is how the numerical flux function $\tilde{\boldsymbol{F}}_{ij} \cdot \tilde{\boldsymbol{n}}_{ij}$ is computed. According to Figure 4.4, let (k_1, k_2) be the two vertices that bound edge j and let $(\boldsymbol{q}_h^-, \boldsymbol{q}_h^+)$ be the numerical solution inside element $T_i(t)$ and inside the neighbor element $T_j(t)$, respectively. The numerical flux is evaluated at each sub-face by taking into account a multidimensional vertex-based flux and a one-dimensional edge-based flux, therefore the term $\tilde{\boldsymbol{F}}_{ij} \cdot \tilde{\boldsymbol{n}}_{ij}$ in (2.68) is computed as follows:

- first we solve the Riemann problem around the two vertices (k_1, k_2) of the space-time face ∂C_{ij} using the multidimensional HLL Riemann solver [88], hence obtaining the multidimensional states $(\boldsymbol{Q}_1^*, \boldsymbol{Q}_2^*)$ and the multidimensional numerical fluxes $(\tilde{\boldsymbol{F}}_1^*, \tilde{\boldsymbol{F}}_2^*)$. The multidimensional HLL formulation adopted here is the same algorithm used as node solver and explained in Section 3.1.3 of Chapter 3. Now we do not limit to evaluate the interacting state $\boldsymbol{Q}^*$, but we also compute the multidimensional fluxes $\tilde{\boldsymbol{F}}^*$ for each vertex (k_1, k_2) of edge j. Positivity of density and pressure is guaranteed by using the self-adjusting positivity preserving scheme of [23], extended to unstructured meshes. According to [88], the final expression for the multidimensional vertex-based fluxes $\tilde{\boldsymbol{F}}^*$ are computed by a blending between the multidimensional HLL and HLLC fluxes, with the blending factor given by the flattener variable introduced and presented in details in [23] (see also Appendix B);

- then a classical Godunov-type one-dimensional edge flux $\tilde{\boldsymbol{F}}_{edge}$ has to be determined, that is projected orthogonally w.r.t. the edge j, as usually done on unstructured meshes. The one-dimensional ALE-type HLL flux can be formulated as

$$\tilde{\boldsymbol{F}}_{edge} \cdot \tilde{\boldsymbol{n}}_{ij} = \frac{1}{s_R - s_L} \left[\left(s_R \tilde{\boldsymbol{F}}(\boldsymbol{q}_h^-) - s_L \tilde{\boldsymbol{F}}(\boldsymbol{q}_h^+) \right) \cdot \tilde{\boldsymbol{n}}_{ij} + s_L s_R \left(\boldsymbol{q}_h^+ - \boldsymbol{q}_h^- \right) \right], \tag{4.16}$$

  where s_L and s_R are the usual HLL estimates of the left and right signal speeds, associated with the ALE Jacobian matrix in spatial normal direction, which is given by (2.70);

- the spatial part of the space-time surface integral at the space-time sub-face ∂C_{ij}^n is computed using the Simpson rule, which achieves up to fourth order of accuracy. In time, classical Gauss-Legendre quadrature with two quadrature points is used. The final approximation of the lateral space-time surface integrals reads

$$\int_0^1\int_0^1 |\partial C_{ij}^n| \tilde{\boldsymbol{F}}_{ij} \cdot \tilde{\mathbf{n}}_{ij} d\tau d\chi_1 \approx \sum_j \omega_j \Big(\frac{1}{6} |\partial C_{ij}^n|(0,\tau_j) \tilde{\boldsymbol{F}}_1^*(\tau_j) \cdot \tilde{\mathbf{n}}_{ij}(0,\tau_j) + \frac{4}{6} |\partial C_{ij}^n|(\frac{1}{2},\tau_j) \tilde{\boldsymbol{F}}_{edge}(\tau_j) \cdot \tilde{\mathbf{n}}_{ij}(\frac{1}{2},\tau_j) + \frac{1}{6} |\partial C_{ij}^n|(1,\tau_j) \tilde{\boldsymbol{F}}_2^*(\tau_j) \cdot \tilde{\mathbf{n}}_{ij}(1,\tau_j) \Big), \tag{4.17}$$

where τ_j and ω_j are the temporal quadrature points and weights, respectively. For unsplit Godunov-type schemes in two space dimensions based on *one-dimensional Riemann solvers* the Courant number CFL must satisfy CFL < 0.5 for linear stability, as mentioned in [229]. However, numerical evidence indicates that our finite volume schemes based on *multidimensional Riemann solvers* are able to run in a stable manner also with a much less restrictive CFL condition of CFL < 1, because of the multidimensionality introduced in the numerical flux evaluation, see [88]. Hence, for the test problems run with the above-described multidimensional HLL finite volume scheme and presented in Chapter 6, the CFL number has been actually set very close to this experimentally observed limit by choosing CFL $= 0.95$. Thus, the multidimensional finite volume scheme can run the same test case much more efficiently than a classical edge-based finite volume algorithm. This leads to a significant improvement in terms of computational efforts, especially in the Lagrangian framework, which is typically characterized by very small timesteps caused by strongly deformed and distorted elements. Furthermore, the high order WENO reconstruction on moving unstructured meshes is very expensive, since the reconstruction equations can no longer be solved once and for all in a pre-processing stage, as it was the case for Eulerian schemes on fixed meshes in [101, 102]. Hence, the possibility to use larger time steps leads to less reconstructions to be done when running a simulation to a given final time reducing thus the total computational effort.

4.3 Quadrature-free ALE ADER schemes

Since the geometry is continuously changing in time, Lagrangian schemes need to update all the geometric quantities (e.g. normal vectors, volumes, side

lengths, *etc.*) at *each* timestep. Therefore, all geometry-related functions of the algorithm can not be called only once in a pre-processing stage, as for Eulerian methods on fixed meshes. As a consequence, Lagrangian methods are typically much more demanding in terms of computational effort compared to classical Eulerian algorithms. In Table 4.1 we noticed that for direct ALE ADER-WENO schemes both the high order WENO reconstruction algorithm and the Gaussian quadrature of the numerical flux integrals on the space-time boundaries require about 90% of the total computational time.
In [102] a WENO reconstruction technique was presented where the reconstruction matrix is calculated and stored for all elements once and for all in a pre-processing step, while in [103] the authors used a quadrature-free formulation for the numerical flux computation. As already stated in Section 2.2 of Chapter 2, for ADER-WENO algorithms on moving meshes the reconstruction matrix cannot be pre-computed and stored, but it must be recomputed at each timestep again, together with the solution of the associated linear system (2.9).
A first attempt in improving the efficiency of the flux computation has been presented in the previous section, where the use of a genuinely multidimensional HLL-type Riemann solver [21, 22, 88] yields larger time steps and therefore leads to a computationally more efficient scheme compared to a method based on classical one-dimensional Riemann solvers. Another possibility is given by the adoption of a local time stepping scheme (see Section 4.1), that allows each control volume to reach the final time of the simulation using its own optimal timestep, which obeys only a local CFL stability condition instead of a global one. Nevertheless, a quadrature-free formulation could definitely further improve the efficiency of a high order Lagrangian algorithm. Thus, the present section aims at introducing a quadrature-free formulation of the numerical flux integration within the family of high order one-step direct ALE ADER-WENO schemes, following the original work of Dumbser et al. [103] carried out in the Eulerian context.
This new approach differs from the original ALE ADER-WENO method only for the numerical flux evaluation. Thus, the WENO reconstruction, the local Galerkin predictor and the mesh motion algorithm are exactly carried out in the *same* manner as illustrated in the corresponding sections of Chapter 2. We adopt again the space-time divergence operator $\tilde{\nabla}$ to reformulate the governing PDE (2.1) in the form given by 2.45. Then, integration over the moving space-time control volume $C_i^n = T_i(t) \times \left[t^n; t^{n+1}\right]$ and application of Gauss theorem leads to expression (2.48).
In Section 2.5, according to Figure 2.8, each lateral space-time sub-volume ∂C_{ij}^n has been mapped to a reference element, here addressed with ∂C_E, defined on

a local reference system (χ_1, χ_2, τ) and then parametrized with a set of *bilinear* (in 2D) or *trilinear* (in 3D) basis functions $\beta_k(\chi_1, \chi_2, \tau)$ given by (2.51). The flux integral appearing in (2.48) has been computed on the reference element ∂C_E using multidimensional Gaussian quadrature rules of suitable order of accuracy, see [218] for details. Since such a parametrization is *not linear*, the outward pointing normal vector $\tilde{\boldsymbol{n}}$ as well as the Jacobian of the transformation between ∂C_{ij}^n and ∂C_E are *not constant.* As a consequence no optimization can be done in order to improve the efficiency of the integral computation, unless a sort of *linearization* of the multidimensional space-time control volume C_i^n is performed. For this purpose each lateral sub-volume ∂C_{ij}^n is *split* into a total number of N_l sub-elements, yielding either *two space-time triangles* in 2D or *three space-time tetrahedra* in 3D, as shown in Figures 4.5 and 4.6, respectively. In two space dimensions the lateral space-time sub-surface ∂C_{ij}^n is bounded by a total number $N_k = 4$ of vertices with physical space-time coordinates $\tilde{\boldsymbol{X}}_{ij,k}^n$ and is divided into two triangles $\mathcal{T}_{ij,l}^{2D}$ $(N_l = 2)$, defined according to Figure 4.5 by the following connectivity:

$$\mathcal{T}_{ij,1}^{2D} = \left[\tilde{\boldsymbol{X}}_{ij,IV}^n, \tilde{\boldsymbol{X}}_{ij,I}^n, \tilde{\boldsymbol{X}}_{ij,III}^n\right], \quad \mathcal{T}_{ij,2}^{2D} = \left[\tilde{\boldsymbol{X}}_{ij,II}^n, \tilde{\boldsymbol{X}}_{ij,III}^n, \tilde{\boldsymbol{X}}_{ij,I}^n\right]. \tag{4.18}$$

The same philosophy applies for the three-dimensional case, where $N_k = 6$ and the lateral space-time sub-volume ∂C_{ij}^n is decomposed into three tetrahedra $\mathcal{T}_{ij,l}^{3D}$ $(N_l = 3)$ given by

$$\begin{aligned}
\mathcal{T}_{ij,1}^{3D} &= \left[\tilde{\boldsymbol{X}}_{ij,I}^n, \tilde{\boldsymbol{X}}_{ij,II}^n, \tilde{\boldsymbol{X}}_{ij,III}^n, \tilde{\boldsymbol{X}}_{ij,IV}^n\right], \\
\mathcal{T}_{ij,2}^{3D} &= \left[\tilde{\boldsymbol{X}}_{ij,IV}^n, \tilde{\boldsymbol{X}}_{ij,VI}^n, \tilde{\boldsymbol{X}}_{ij,V}^n, \tilde{\boldsymbol{X}}_{ij,III}^n\right], \\
\mathcal{T}_{ij,3}^{3D} &= \left[\tilde{\boldsymbol{X}}_{ij,II}^n, \tilde{\boldsymbol{X}}_{ij,IV}^n, \tilde{\boldsymbol{X}}_{ij,V}^n, \tilde{\boldsymbol{X}}_{ij,III}^n\right].
\end{aligned} \tag{4.19}$$

In three space dimensions an additional difficulty occurs, which is given by the space-time evolution of the *edges* of the tetrahedron T_i^n. The space-time sub-volumes ∂C_{ij}^n are bounded by space-time surfaces that are generated by the evolution in space and time of the element edges. Particular care must be taken in order to assure that the space-time control volume C_{ij}^n remains *closed*, hence automatically satisfying the Geometric Conservation Law (GCL). For this reason the space-time evolution of each edge of the mesh has to be split in the same manner, i.e. with the same *orientation*, for all those elements which share the same edge. This is not the case in two space-dimensions, where the lateral sub-surfaces ∂C_{ij}^n are separated one from the others by *segments*, which are the space-time evolution of the vertexes of the triangular elements. In 3D, for each face of the tetrahedron, we may have six different orientation of the splitting scheme proposed in (4.19) and during the pre-processing stage each face is assigned with the corresponding orientation that makes the space-time volume C_{ij}^n *closed* for each computational element. For all the numerical test problems shown later in the next chapter it has been explicitly verified that the GCL has always been satisfied for all elements and for all timesteps up to machine precision.

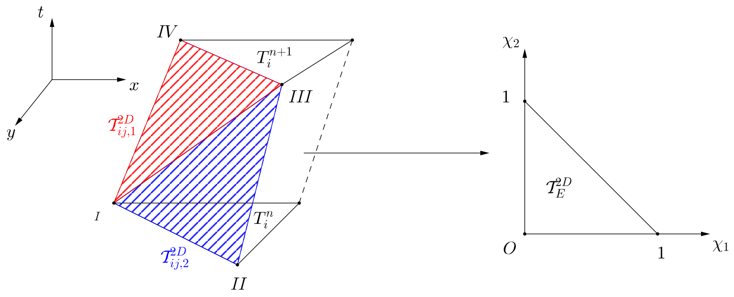


Figure 4.5: Splitting of the quadrilateral space-time sub-surface ∂C_{ij}^n into two space-time sub-triangles $\mathcal{T}_{ij,1}^{2D}$ (red) and $\mathcal{T}_{ij,2}^{2D}$ (blue). Each sub-triangle is mapped to the reference sub-element $\mathcal{T}_E^{2D}$ defined on the local reference system (χ_1, χ_2).

In order to make the computation easier, integration is carried out on a reference sub-element, which is chosen according to the value of d. The mapping of the element configurations T_i^n and T_i^{n+1} is very simple, since these control volumes are orthogonal to the time coordinate. Therefore we use again the transformation (2.3), where $(\xi, \eta, \zeta) \in [0; 1]$, with the space-time unit normal vectors given by $\tilde{\boldsymbol{n}} = (0, 0, 0, -1)$ for T_i^n and $\tilde{\boldsymbol{n}} = (0, 0, 0, 1)$ for T_i^{n+1}. Each lateral sub-element l of ∂C_{ij}^n is then mapped to the reference triangle $\mathcal{T}_E^{2D}$ or the reference tetrahedron $\mathcal{T}_E^{3D}$ defined in the local reference coordinate system (χ_1, χ_2, χ_3), see Figures 4.5-4.6. Furthermore each sub-element is assigned with the same *local* connectivity $\mathcal{C}$ (2.4) among the N_k vertices, according to Figure 2.1. The Jacobian of the transformation from the physical system to the reference system (χ_1, χ_2, χ_3) for the space-time sub-volume $\partial C_{ij}(t)$ and the

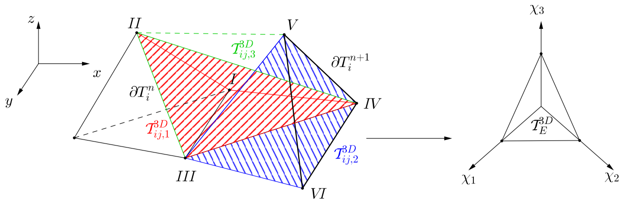


Figure 4.6: Splitting of the space-time sub-volume ∂C_{ij}^n into three space-time sub-tetrahedra $\mathcal{T}^{3D}_{ij,1}$ (red), $\mathcal{T}^{3D}_{ij,2}$ (blue) and $\mathcal{T}^{3D}_{ij,3}$ (green). Each sub-tetrahedron is mapped to the reference sub-element $\mathcal{T}^{3D}_E$ defined on the local reference system (χ_1, χ_2, χ_3).

associated outward pointing normal vector can be evaluated *simultaneously* by computing the determinant of the matrix $\mathcal{S}_l$, which, for each sub-element $\mathcal{T}^{3D}_{ij,l}$, reads

$$\mathcal{S}_l = \begin{pmatrix} \hat{\mathbf{e}} \\ \tilde{\mathbf{X}}_{ij,\mathcal{P}(2)} - \tilde{\mathbf{X}}_{ij,\mathcal{P}(1)} \\ \tilde{\mathbf{X}}_{ij,\mathcal{P}(3)} - \tilde{\mathbf{X}}_{ij,\mathcal{P}(1)} \\ \tilde{\mathbf{X}}_{ij,\mathcal{P}(4)} - \tilde{\mathbf{X}}_{ij,\mathcal{P}(1)} \end{pmatrix}_l := \begin{pmatrix} \hat{\mathbf{e}} \\ \tilde{\mathbf{s}}_1 \\ \tilde{\mathbf{s}}_2 \\ \tilde{\mathbf{s}}_3 \end{pmatrix}_l , \tag{4.20}$$

with $\hat{\mathbf{e}} = (\hat{\mathbf{e}}_1, \hat{\mathbf{e}}_2, \hat{\mathbf{e}}_3, \hat{\mathbf{e}}_4)$. We address with $\hat{\mathbf{e}}_p$ the unit vector aligned with the p-th axis of the physical coordinate system (x, y, z, t), while the q-th component of vector $\tilde{\mathbf{s}}_i$ with $i = 1, 2, 3$ is denoted by $\tilde{s}_{i,q}$. Therefore the expression for the space-time normal vector is given by

$$\tilde{\boldsymbol{n}}_{ij} = \left(\epsilon_{pqrt}\, \hat{\mathbf{e}}_p\, \tilde{s}_{1,q}\ \tilde{s}_{2,r}\ \tilde{s}_{2,t}\right) / |\partial C_{ij}^n|, \tag{4.21}$$

where the *Levi-Civita* is evaluated according to (2.54) and with the determinant defined as

$$|\partial C_{ij}^n| = \left\| \epsilon_{pqrt}\, \hat{\mathbf{e}}_p\ \tilde{s}_{1,q}\ \tilde{s}_{2,r}\ \tilde{s}_{3,t} \right\| . \tag{4.22}$$

To ease notation and to let the reader focus on the quadrature-free approach for the numerical flux computation, we consider the *conservative* formulation of the governing hyperbolic balance laws (2.1) with no source terms, hence

$\boldsymbol{S}(\boldsymbol{Q}) = 0$. However, the extension of the algorithm to the non-conservative case follows straightforward.
Discretization of Eqn.(2.48) yields (2.68), when $\tilde{\boldsymbol{B}}(\boldsymbol{Q}) \cdot \tilde{\nabla}\boldsymbol{Q} = 0$. Here, the space-time flux integrals on the space-time boundaries ∂C_{ij}^n are split into a total number of N_l integrals on the corresponding simplex sub-elements. Therefore the finite volume scheme for the *conservative* formulation (2.68) is given by

$$|T_i^{n+1}|\, \boldsymbol{Q}_i^{n+1} = |T_i^n|\, \boldsymbol{Q}_i^n - \sum_{T_j \in \mathcal{N}_i} \sum_{\mathcal{T}_{ij,l}^{3D} \in N_l} \int\limits_{\mathcal{T}_{E,l}^{3D}} |\partial C_{ij}^n| \tilde{\boldsymbol{F}}_{ij} \cdot \tilde{\boldsymbol{n}}_{ij}\, d\chi_1 d\chi_2 d\chi_3, \tag{4.23}$$

where each lateral physical space-time integral has been replaced by N_l contributions given by (4.18)-(4.19) on the corresponding reference element and the source term has disappeared according to the assumption made within this section. As usual, we use either a Rusanov-type (2.69) or an Osher-type (2.71) numerical flux to evaluate the term $\tilde{\boldsymbol{F}}_{ij} \cdot \tilde{\boldsymbol{n}}_{ij}$ in (4.23).
Since the space-time sub-elements $\mathcal{T}_{ij,l}^{3D}$ in (4.23) have been designed to be either triangles or tetrahedra, the transformation from the physical system (x, y, z) to the reference system (χ_1, χ_2, χ_3) is linear, hence implying *constant Jacobian* and *constant normal vector* within each sub-element. As a consequence the integrals appearing in (4.23) can be evaluated using a *quadrature-free formulation*: the physical flux $\tilde{\boldsymbol{F}}(\boldsymbol{q}_h^+)$ contained in the expression of the numerical flux $\tilde{\boldsymbol{F}}_{ij} \cdot \tilde{\boldsymbol{n}}_{ij}$ for (2.69) and (2.71) is written in terms of the space-time basis functions $\theta_l(\tilde{\boldsymbol{\xi}})$ used for the local predictor solution (see Section 2.3 of Chapter 2), and the integral of the physical flux on the reference sub-element results in

$$\int\limits_{\mathcal{T}_{E,l}^{3D}} \tilde{\boldsymbol{F}}(\boldsymbol{q}_h^+)\, d\tilde{\xi} = \int\limits_{\mathcal{T}_{E,l}^{3D}} \theta_m(\tilde{\boldsymbol{\xi}}) \widehat{\boldsymbol{F}}_m^+\, d\tilde{\xi} = \widehat{\boldsymbol{F}}_m^+ \int\limits_{\mathcal{T}_{E,l}^{3D}} \theta_m(\tilde{\boldsymbol{\xi}})\, d\tilde{\xi} := \mathcal{I}_{l,m}^+\, \widehat{\boldsymbol{F}}_m^+, \tag{4.24}$$

where the degrees of freedom $\widehat{\boldsymbol{F}}_m^+$ are known from the local CG predictor procedure and the term $\mathcal{I}_{l,m}^+$ represents the integral of the basis functions $\theta_m(\tilde{\boldsymbol{\xi}})$ over the reference sub-element $\mathcal{T}_{E,l}^{3D}$ of the sub-element $\mathcal{T}_{ij,l}^{3D}$. The same procedure applies for the left physical flux $\tilde{\boldsymbol{F}}(\boldsymbol{q}_h^-)$, yielding $\mathcal{I}_{l,m}^- \widehat{\boldsymbol{F}}_m^-$. The basis functions $\theta_m(\tilde{\boldsymbol{\xi}})$ do *not* change in time and they are always defined in the *same* space-time reference system $\tilde{\boldsymbol{\xi}}$, hence allowing the volume integral of (4.24) to be evaluated only *once* in the pre-processing step for each sub-element $\mathcal{T}_{ij,l}^{3D}$ after the mapping to the reference sub-element $\mathcal{T}_{E,l}^{3D}$ has been carried out. The same procedure applies for the numerical solution $\boldsymbol{q}_h^+ = \theta_l(\tilde{\boldsymbol{\xi}})\, \widehat{\boldsymbol{q}}_l^+$ appearing in the expression of the numerical flux, either (2.69) or (2.71), leading to

$$\int\limits_{\mathcal{T}^{3D}_{E,l}} \boldsymbol{q}_h^{\pm}\, d\tilde{\xi} := \mathcal{I}^{\pm}_{l,m}\, \widehat{\boldsymbol{q}}^{\pm}_{m}. \tag{4.25}$$

The ALE Jacobian matrix $\boldsymbol{A}^{\mathbf{V}}_{\boldsymbol{n}}$ can *not* be pre-computed. However it is evaluated at each timestep only once for each space-time sub-volume ∂C^n_{ij} in its space-time barycenter $G_{ij,l}$, i.e. $\boldsymbol{A}^{\mathbf{V}}_{\boldsymbol{n}}(G_{ij,j})$, and is then *frozen* over the entire space-time sub-element, like in the Eulerian quadrature-free schemes proposed in [103]. Therefore, the final expression for the high order quadrature-free numerical flux of the finite volume scheme (2.68) on moving unstructured meshes simply reads

$$\int\limits_{T^{3D}_{E,l}} \tilde{\boldsymbol{F}}_{ij}\cdot\tilde{\boldsymbol{n}}_{ij} = \frac{1}{2}\left(\mathcal{I}^{+}_{l,m}\widehat{\boldsymbol{F}}^{+}_{m} + \mathcal{I}^{-}_{l,m}\widehat{\boldsymbol{F}}^{-}_{m}\right)\cdot\tilde{\boldsymbol{n}}_{ij} - \frac{1}{2}\boldsymbol{D}_{ij}\left(\mathcal{I}^{+}_{l,m}\widehat{\boldsymbol{q}}^{+}_{m} - \mathcal{I}^{-}_{l,m}\widehat{\boldsymbol{q}}^{-}_{m}\right), \tag{4.26}$$

with the dissipation matrix being either $\boldsymbol{D}_{ij} = |\lambda_{\max,i}|\boldsymbol{I}$ for the Rusanov-type flux (2.69) or $\boldsymbol{D}_{ij} = \int\limits_0^1 \left|\boldsymbol{A}^{\mathbf{V}}_{\boldsymbol{n}}(\boldsymbol{\Psi}(s))\right| ds$ for the Osher-type flux (2.71).

4.4 Direct ALE ADER-MOOD finite volume schemes

It is evident from Table 4.1 that the WENO reconstruction step is the only remaining part of the direct ALE ADER algorithm for which no improvements have been done so far. As clearly stated in Section 2.2 of Chapter 2, the WENO approach requires the blending of six (in 2D) or nine (in 3D) polynomial reconstructions per cell per variable using nonlinear weights to obtain essentially-non-oscillatory (ENO) schemes. This computational cost is almost incompressible if the WENO paradigm is not questioned.

In the finite volume context a new concept has been recently proposed, namely the Multi-dimensional Optimal Order Detection (MOOD) approach, which is an *a posteriori* approach to the problem of limiting. Indeed, the key idea of this paradigm is to run a spatially unlimited high-order finite volume scheme in order to produce a so-called *candidate solution*. Then, the validity of the candidate solution is tested against a set of pre-defined *admissibility criteria*. Some cells are marked as “acceptable” and are therefore valid. Some others may be locally marked as “problematic” if they do not pass the detection process.

These cells are consequently *locally recomputed* using polynomial reconstructions of a lower degree. Thus, after decrementing the polynomial degree and locally recomputing the solution, a new candidate solution is obtained. That solution is again tested for validity and the decrementing procedure re-applies, if necessary. Such degree decrementing can occur several times within one time step for the same cell, but it will always stop after a finite number of steps: either the cell is valid for a polynomial degree greater than 0, or the degree zero is reached. In the latter case, which is the worst case scenario, the cell is updated with the robust and stable first order accurate Godunov finite volume scheme, that is supposed to produce always valid (monotone and positivity-preserving) solutions under a CFL stability condition. This *a posteriori* detection and decrementing iterative loop is called the *MOOD loop*. Recently in [165] the *a posteriori* MOOD method has been successfully substituted to WENO within a 3D ADER high order finite volume scheme designed for fixed grid and it has been applied to different systems of conservation laws. Even more recently the *a posteriori* MOOD concept has been revamped and used as a subcell limiter for high order accurate Discontinuous Galerkin schemes in [111]. We refer the reader to [69, 79, 81, 165] for more details.

Contrarily to WENO, the MOOD paradigm does not require several polynomial reconstructions per cell: only one central stencil is considered to perform the high order reconstruction in space. Moreover the detection of problematic cells is made *a posteriori*, i.e. on a candidate solution at time t^{n+1}. Consequently it drastically eases the test of the candidate solution against any desirable property (positivity, mesh validity, etc.). In addition, a list of problematic cells, that need to be re-updated, can be constructed from the previous checks. If the detection criteria as well as the decrementing procedure are designed properly, for a $\mathcal{M}$-th order accurate ALE-MOOD scheme we may expect to retrieve at least the same accuracy and ENO behavior than the equivalent $\mathcal{M}$-th order accurate ALE WENO scheme. On the other hand we could expect genuine gains in CPU time and memory consumption in favor of MOOD. Thereby the main purpose of this section is to design a MOOD approach within our existing high order direct ALE ADER framework. The new algorithm, which will be addressed with ALE ADER-MOOD, is expected to be as accurate as the original ALE ADER-WENO formulation, but more efficient.

The reconstruction step is now performed according to Section 2.2, but we do not need to build more than *one central* ($s = 1$) reconstruction stencil $\mathcal{S}_i^s$ for both $d = 2$ and $d = 3$. Then, the local CG or DG predictor, the mesh motion and the finite volume scheme are carried out as exhaustively explained in Sections 2.3, 2.4 and 2.5, respectively. Since the high order reconstruction polynomials $\mathbf{w}_h(\mathbf{x}, t^n)$ of degree M are *not limited*, the numerical solution $\boldsymbol{Q}_i^{n+1}$ produced by the discrete evolution equation (2.61) or (2.68) may not be valid, i.e. it might not satisfy either some physical requirements, as positivity, or some numerical requirements, as *Not-a-Number* (NaN) values. Therefore the result evaluated by Eqn. (2.61) or Eqn. (2.68) is addressed as a *candidate solution* $\boldsymbol{Q}_i^*$, which will be marked as "acceptable" only when it fulfills both the physical and the numerical requirements. In the following we give an overview of the entire MOOD loop which has been adopted within the framework of the direct ALE ADER finite volume schemes proposed in this work. We apply this new approach to the Euler equations of compressible gas dynamics, fully described later in Section 5.1 of Chapter 5.

4.4.1 MOOD paradigm as stabilization technique

The *a posteriori* MOOD paradigm is based on the fact that it is relatively simple to check *a posteriori* the validity of a numerical discrete solution at the end of a timestep, provided some physical and numerical criteria. From this check we can also easily extract a list of problematic cells, i.e. cells which have not pass the checks. Then, going back in time at the beginning of the timestep, we can re-update *only* these cells with a more viscous, dissipative and robust numerical scheme. Thus, a new candidate solution, locally updated with two different schemes, is available to be tested against the validity criteria. If problematic cells are still detected, then, at last, an ultra-dissipative robust scheme is used on these cells to obtain the solution at the next time level t^{n+1}. Therefore these elements are re-updated for the second time still within the current timestep. This iterative MOOD loop ends with cells updated either with a high accurate but less robust scheme or with a low accurate but more robust one.
The main three entities which must be given to an iterative MOOD loop are the detection criteria, the cascade of schemes and the parachute scheme.

The detection criteria. The detection criteria are a list of properties that have to be checked to assess whether a numerical solution in a cell is acceptable or not.
The first set of criteria, called the Physical Admissible Detection (PAD) criteria, is based on the physics underlying the simulation and these requirements correspond to properties that must be fulfilled to ensure physical admissibility of a numerical solution. Therefore they strongly depend on the system of conservation laws which is solved. For the Euler equations of compressible gas

dynamics (see Section 5.1) we set the following PAD:

$$\rho_i^* > 0, \qquad \varepsilon_i^* = E_i^* - \frac{1}{2}|\mathbf{v}_i^*|^2 > 0, \tag{4.27}$$

with ε_i^* representing the specific internal energy, according to (5.3). Furthermore in a moving mesh framework the volume of any element T_i must also be strictly positive, thus

$$|T_i^*| > 0. \tag{4.28}$$

The second set of criteria, called Numerical Admissible Detection (NAD) criteria, is based on numerical properties and it contains all the checks needed to ensure that the numerical solution is essentially non-oscillatory. In previous works [69, 79, 79, 165] the NAD criteria adopt a sort of relaxed discrete maximum principle (DMP) with the so-called u2 criteria [81] applied on each conservative variable. Let us briefly describe the DMP+u2 detection process on a generic variable A and a candidate solution A_i^* at time t^{n+1} in the control volume T_i. We consider the Voronoi neighborhood $\mathcal{V}_i$ of element T_i and the Voronoi neighbor element T_j belonging to $\mathcal{V}_i$. The set of vertex neighbors $\mathcal{V}_i$ contains all neighbors of cell T_i that have a common vertex with T_i. To ease notation we will refer to cell T_i with i and to the neighbor element $T_j \in \mathcal{V}_i$ with j. First, if A_i^* fulfills the DMP, i.e.

$$\min_{j\in\mathcal{V}_i}(A_j^n, A_i^n) \leq A_i^* \leq \max_{j\in\mathcal{V}_i}(A_j^n, A_i^n), \tag{4.29}$$

then the cell is valid for this variable. If the DMP is not fulfilled, then we check the u2 criterion [79,81], which allows to determine whether this new extremum is smooth or not. A candidate solution A_i^* in cell T_i^* which violates the DMP is nonetheless eligible if the following holds

$$\mathcal{X}_i^{max}\mathcal{X}_i^{min} > 0 \quad \text{and} \quad \left|\frac{\mathcal{X}_i^{min}}{\mathcal{X}_i^{max}}\right| \geq 1-\varepsilon, \tag{4.30}$$

where ε is a smoothness parameter set to 1/2 and $\mathcal{X}_i^*$ represents a "measure" of the local discrete directional curvature, e.g. the second derivative in x direction of the third order polynomial reconstruction. Moreover we set

$$\mathcal{X}_i^{min} = \min_{j\in\mathcal{V}_i}\left(\mathcal{X}_i^*,\ \mathcal{X}_j^*\right) \quad \text{and} \quad \mathcal{X}_i^{max} = \max_{j\in\mathcal{V}_i}\left(\mathcal{X}_i^*,\ \mathcal{X}_j^*\right). \tag{4.31}$$

The same check is done for the y and z components. Another alternative is to compute the true local curvatures, see [165]. Note that the detection of smooth

local extrema has also been discussed in the context of extremum preserving PPM schemes [74] and MPWENO schemes [19,219]. Furthermore we also check if the candidate solution is an acceptable data, i.e. we verify that *Not-a-Number* (NaN) situations do not occur.
In our implementation if any of the conservative variables does not fulfill either the PAD or the NAD criteria, then one considers that all variables need correction on the next MOOD iteration and the solution $\boldsymbol{Q}_i^*$ must be recomputed by *locally* supplementing the scheme with more dissipation.

The cascade of schemes. The cascade of schemes is a list of ordered numerical schemes, from the most accurate one up to the least accurate but robust one [165]. This sequence is related to the accuracy which the polynomial reconstructions have been evaluated with. We set a maximal polynomial degree M meaning that, in an ideal situation, the corresponding scheme is $M+1$-th order accurate in space and time. Then, several intermediate polynomial degrees for the reconstructions can be tried up to the last scheme, which is supposed to be the most dissipative one, called the parachute scheme. According to [165], we employ the cascade given by $\mathbb{P}_M \rightarrow \mathbb{P}_1^{\text{LIM}} \rightarrow \mathbb{P}_0$ with $M = 4$, but other values could be used as well. The $\mathbb{P}_1^{\text{LIM}}$ scheme uses $\mathbb{P}_1$ reconstructions with Barth & Jespersen slope limiting [30].
The MOOD loop first computes the unlimited $\mathbb{P}_M$ candidate solution $\boldsymbol{Q}_i^*$ for each element T_i and checks if any cell is problematic according to the detection criteria. Then, all invalid cells, i.e. all cells which have not passed the detection criteria, are recomputed with $\mathbb{P}_1^{\text{LIM}}$ reconstructions. For the numerical flux evaluation we also need $\mathbb{P}_1^{\text{LIM}}$ reconstructions for each direct neighbor T_j of T_i. This is to ensure that the numerical solution within the control volume T_i is updated with fluxes of the *same* order of accuracy for each boundary. This new candidate solution is checked again for validity. Such a candidate solution may be given by both $\mathbb{P}_M$-updated cells and $\mathbb{P}_1^{\text{LIM}}$-updated ones. Nonetheless some cells can be invalid again because the $\mathbb{P}_1^{\text{LIM}}$ may still produce some non-admissible states. Therefore, for these still invalid cells, we rely on the parachute scheme $\mathbb{P}_0$. In the worst case scenario all cells in the domain are updated with the $\mathbb{P}_0$ scheme. Contrarily, in an ideal situation, all cells are updated with the unlimited $\mathbb{P}_M$ scheme.

The parachute scheme. The parachute scheme is the last scheme of the cascade, which is used as a last resort when the detection criteria are not fulfilled. As such, the candidate numerical solution it provides must always be considered as a valid one. According to [69, 79, 79, 165] we use the first order Godunov finite volume scheme, addressed with $\mathbb{P}_0$. Note that the MOOD loop always converges to an acceptable discrete solution provided that the parachute scheme produces such a solution.

4.4.2 The MOOD loop

In the new ALE ADER-MOOD algorithm the MOOD loop simply embraces the main evolution routines of the high order ADER method and iterates to recompute those cells marked as "problematic" by the detection criteria described in the previous section.

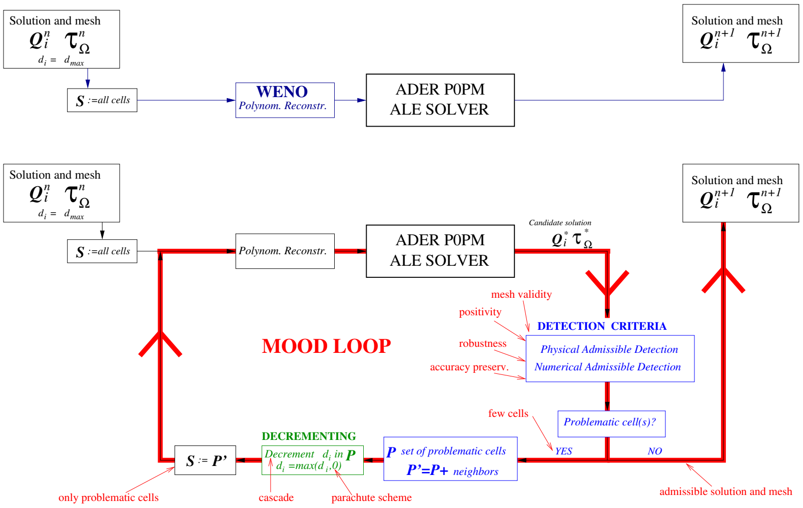


Figure 4.7: Sketch of the direct ALE ADER-$\mathbb{P}_0\mathbb{P}_M$ simulation code. Top: original ADER-WENO solver. Bottom: sketch of the MOOD loop embracing the existing ALE ADER-$\mathbb{P}_0\mathbb{P}_M$ solver.

Figure 4.7 shows a sketch of the MOOD loop. The efficiency of the *a posteriori* MOOD paradigm is mainly due to the fact that usually few cells need a cell polynomial degree decrementing, as proved by the numerical results given later in Chapter 6. Therefore the extra-work needed to recompute a new candidate

solution on problematic cells is low. Moreover, for a given polynomial degree M, only one polynomial reconstruction per variable and per cell is computed, which reduces the CPU time but also the memory consumption compared to an *a priori* WENO reconstruction.
The flexibility of the *a posteriori* MOOD paradigm is based on:

1. any robust and preferred parachute scheme can be kept as the last and safest scheme;

2. only light modifications are usually needed to implement a MOOD loop within an existing high-order finite volume code because it embraces the existing solver, then it is not genuinely invasive (see Figure 4.7);

3. any constraint or property can be added to the list of detection criteria, should it be based on either physical or numerical requirements.

5 Applications to Hyperbolic Systems

In order to validate the unstructured multidimensional direct ALE ADER schemes presented in this work, we solve a wide set of benchmark test problems using different hyperbolic systems of governing equations that can all be cast into form (2.1). We apply the new schemes proposed in this work to several hyperbolic systems, namely the multidimensional Euler equations of compressible gas dynamics, the ideal classical and relativistic magneto-hydrodynamics (MHD) equations and the non-conservative seven-equation Baer-Nunziato model of compressible multi-phase flows with stiff relaxation source terms.

In this chapter we present each system of balance laws used for our simulations together with a detailed description of the associated numerical test cases that have been run to assess the accuracy and the robustness of our schemes. We always consider the system of equations in three space dimensions, from which one can easily derive the corresponding two-dimensional formulation.

The numerical results will be shown next, in Chapter 6. This is meant to ease the reading and to avoid repetition of the same test case description.

5.1 The Euler equations of compressible gas dynamics

The first set of equations considered within this work are the so-called Euler equations of compressible gas dynamics, also known as hydrodynamics equations. They constitute a *conservative* system of hyperbolic conservation laws with no sources and they govern the fluid flow in case of neutral, i.e. non-charged, fluids.

Let $\boldsymbol{Q} = (\rho, \rho u, \rho v, \rho w, \rho E)$ be the vector of conserved variables with ρ denoting the fluid density, $\boldsymbol{v} = (u, v, w)$ representing the velocity vector and ρE being the total energy density. Let furthermore p be the fluid pressure and γ the ratio of specific heats of the ideal gas, so that the speed of sound is $c = \sqrt{\frac{\gamma p}{\rho}}$.

The three-dimensional Euler equations of compressible gas dynamics can be cast into form (2.1), with the state vector $\boldsymbol{Q}$ previously defined and the flux tensor $\boldsymbol{F} = (\boldsymbol{f}, \boldsymbol{g}, \boldsymbol{h})$ given by

$$\boldsymbol{f} = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ u(\rho E + p) \end{pmatrix}, \quad \boldsymbol{g} = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vw \\ v(\rho E + p) \end{pmatrix}, \quad \boldsymbol{h} = \begin{pmatrix} \rho w \\ \rho uw \\ \rho vw \\ \rho w^2 + p \\ w(\rho E + p) \end{pmatrix}. \tag{5.1}$$

The term $\boldsymbol{B}$ appearing in (2.1) is zero for this hyperbolic conservation law, because the system does not involve any non-conservative product. The system is then closed by the equation of state for an ideal gas, which reads

$$p = (\gamma - 1)\left(\rho E - \frac{1}{2}\rho \mathbf{v}^2\right). \tag{5.2}$$

Moreover we define the specific internal energy ε^* as

$$\varepsilon^* = E - \frac{1}{2}|\mathbf{v}^2| \tag{5.3}$$

In Section 5.1.12 we present a modification of the two-dimensional Euler equations (5.1) that is able to handle *multi-material* flows, hence focusing on the simulation of two-material fluids.

5.1.1 Numerical convergence studies

The convergence studies of our direct ALE ADER finite volume schemes for the Euler equations of compressible gas dynamics (5.1) are carried out considering the solution of a smooth convected isentropic vortex first proposed on unstructured meshes by Hu and Shu [144] in two space dimensions. The initial computational domain for the three-dimensional case is the box $\Omega(0) = [0; 10] \times [0; 10] \times [0; 5]$ with periodic boundary conditions imposed on each face. The domain reduces to the square $\Omega(0) = [0; 10] \times [0; 10]$ if $d = 2$. The initial condition is the same given in [144], where we set to zero the $z-$aligned velocity component w, and it is given as a linear superposition of a homogeneous background field and some perturbations δ:

$$\boldsymbol{U} = (\rho, u, v, w, p) = (1 + \delta\rho, 1 + \delta u, 1 + \delta v, 1 + \delta w, 1 + \delta p). \tag{5.4}$$

The perturbation of the velocity vector $\mathbf{v} = (u, v, w)$ as well as the perturbation of temperature T read

$$\begin{pmatrix} \delta u \\ \delta v \\ \delta w \end{pmatrix} = \frac{\epsilon}{2\pi} e^{\frac{1-r^2}{2}} \begin{pmatrix} -(y-5) \\ (x-5) \\ 0 \end{pmatrix}, \qquad \delta T = -\frac{(\gamma-1)\epsilon^2}{8\gamma\pi^2} e^{1-r^2}, \tag{5.5}$$

where the radius of the vortex has been defined on the $x-y$ plane as $r^2 = (x-5)^2 + (y-5)^2$, the vortex strength is $\epsilon = 5$ and the ratio of specific heats is set to $\gamma = 1.4$. The entropy perturbation is assumed to be zero, i.e. $S = \frac{p}{\rho^\gamma} = 0$, while the perturbations for density and pressure are given by

$$\delta\rho = (1+\delta T)^{\frac{1}{\gamma-1}} - 1, \quad \delta p = (1+\delta T)^{\frac{\gamma}{\gamma-1}} - 1. \tag{5.6}$$

The vortex is furthermore convected with constant velocity $\mathbf{v}_c = (1, 1, 0)$. The final time of the simulation is chosen to be $t_f = 1.0$, otherwise the deformations occurring in the mesh due to the Lagrangian motion would stretch and twist the control volumes so highly that a rezoning stage would be necessary. Here we want the convergence studies to be done with an almost pure Lagrangian motion, hence no rezoning procedure is admitted and the final time t_f has been set to a sufficiently small value. The exact solution $\boldsymbol{Q}_e$ can be simply computed as the time-shifted initial condition, e.g. $\boldsymbol{Q}_e(\mathbf{x}, t_f) = \boldsymbol{Q}(\mathbf{x} - \mathbf{v}_c t_f, 0)$, with the convective mean velocity $\mathbf{v}_c$ previously defined. The error is measured at time t_f using the continuous L_2 norm with the high order reconstructed solution $\mathbf{w}_h(\mathbf{x}, t_f)$, hence

$$\epsilon_{L_2} = \sqrt{\int\limits_{\Omega(t_f)} \left(\boldsymbol{Q}_e(\mathbf{x}, t_f) - \mathbf{w}_h(\mathbf{x}, t_f)\right)^2 d\mathbf{x}}, \tag{5.7}$$

where $h(\Omega(t_f))$ represents the mesh size which is taken to be the maximum diameter of the circumspheres or the circumcircles of the elements in the final domain configuration $\Omega(t_f)$. Sometimes we might also use the L_1 norm, whose expression is given by

$$\epsilon_{L_1} = \int\limits_{\Omega(t_f)} \left(\boldsymbol{Q}_e(\mathbf{x}, t_f) - \mathbf{w}_h(\mathbf{x}, t_f)\right) d\mathbf{x}. \tag{5.8}$$

Figures 5.1-5.2 show some of the successively refined meshes used for this test case. Convergence rates up to sixth order of accuracy will be shown in Section 6.1.1 of Chapter 6. This test problem is also used to measure the efficiency of the algorithm (see Table 4.1).

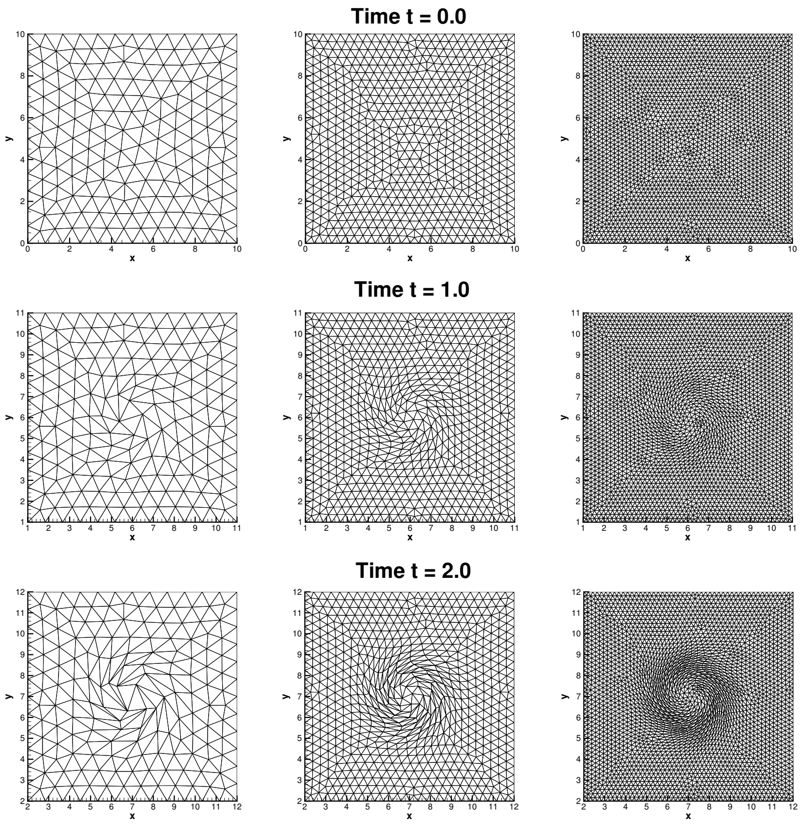


Figure 5.1: Sequence of triangular meshes used for the numerical convergence studies for the two-dimensional Euler equations of compressible gas dynamics at different time outputs: $t = 0$ (top row), $t = 1$ (middle row) and $t = 2$ (bottom row). The total number of elements N_E is increasing from the left grid ($N_E = 320$), passing through the middle one ($N_E = 1298$), to the right one ($N_E = 5180$).

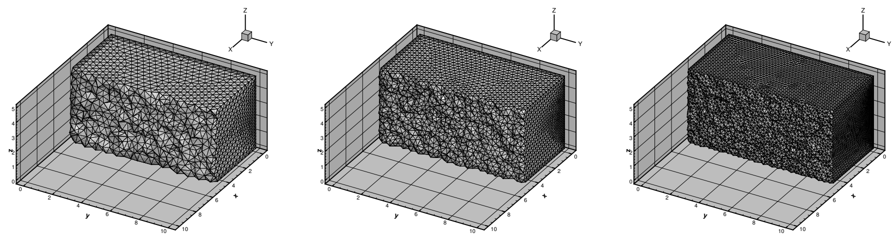


Figure 5.2: Sequence of tetrahedral meshes at the initial time $t = 0$ used for the numerical convergence studies for the three-dimensional Euler equations of compressible gas dynamics. The total number of elements N_E is increasing from the left grid ($N_E = 60157$), passing through the middle one ($N_E = 227231$), to the right one ($N_E = 801385$).

5.1.2 The Sod shock tube problem

Here we propose in a multidimensional setting the well-known Sod shock tube problem, which is a classical one-dimensional test problem that involves a rarefaction wave traveling towards the left boundary as well as a right-moving contact discontinuity and a shock wave traveling to the right. The initial computational domain is the box $\Omega(0) = [-0.5; 0.5] \times [-0.05; 0.05] \times [-0.05; 0.05]$, which is discretized with a characteristic mesh size of $h = 1/100$. In two space dimensions the z coordinate vanishes. We set periodic boundaries in the y and z directions, while transmissive boundaries are imposed along the x direction. The final time of the simulation is chosen to be $t_f = 0.2$. The initial condition consists in a discontinuity located at $x_0 = 0$ between two different states $\boldsymbol{U}_L$ and $\boldsymbol{U}_R$, where $\boldsymbol{U} = (\rho, u, v, w, p)$ denotes the vector of *primitive* variables:

$$\boldsymbol{U}(\mathbf{x}, 0) = \begin{cases} \boldsymbol{U}_L = (1.0, 0, 0, 0, 1.0)\,, & \text{if} \quad x \leq x_0, \\ \boldsymbol{U}_R = (0.125, 0, 0, 0, 0.1)\,, & \text{if} \quad x > x_0. \end{cases} \tag{5.9}$$

Although the Sod problem is a one-dimensional test case, it becomes multidimensional when applied to unstructured meshes, where in general the element faces are not aligned with the coordinate axis or the fluid motion. Hence, it is actually a non trivial test problem. Moreover, a contact wave is present in the solution, so that one can check how well it is resolved by the ALE scheme. In order to reduce numerical diffusion, we employ the Osher-type flux (2.71). The exact solution can be obtained with the exact Riemann solver presented in [229].

5.1.3 Multidimensional explosion problem

The explosion problem can be seen as a fully multidimensional extension of the Sod problem presented in the previous section. The initial domain is the circle (in 2D) or the sphere (in 3D) of radius $R_o = 1$, i.e. $\Omega(0) = \{\mathbf{x} : \|\mathbf{x}\| \leq R_o\}$, in which the associated circle or sphere of radius $R = 0.5$ separates two different constant states:

$$\boldsymbol{U}(\mathbf{x}, 0) = \begin{cases} \boldsymbol{U}_i = (1, 0, 0, 0, 1), & \text{if} \quad \|\mathbf{x}\| \leq R, \\ \boldsymbol{U}_o = (0.125, 0, 0, 0, 0.1), & \text{if} \quad \|\mathbf{x}\| > R. \end{cases} \tag{5.10}$$

The *inner state* $\boldsymbol{U}_i$ and the *outer state* $\boldsymbol{U}_o$ correspond to the ones of the 1D Sod problem, according to (5.9).
For cylindrically or spherically symmetric problems, the multidimensional Euler system (2.1)-(5.1) can be simplified to a one-dimensional system with geometric source terms, see [39, 229]. It reads

$$\boldsymbol{Q}_t + \boldsymbol{F}(\boldsymbol{Q})_r = \boldsymbol{S}(\boldsymbol{Q}), \tag{5.11}$$

with

$$\boldsymbol{Q} = \begin{pmatrix} \rho \\ \rho u \\ \rho E \end{pmatrix}, \quad \boldsymbol{F} = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ u(\rho E + p) \end{pmatrix}, \quad \boldsymbol{S} = -\frac{d-1}{r} \begin{pmatrix} \rho u \\ \rho u^2 \\ u(\rho E + p) \end{pmatrix}. \tag{5.12}$$

The radial direction is denoted as usual by r, while u represents here the radial velocity and d is the number of space dimensions. In order to compute a suitable reference solution we fix the value of $d \in [2, 3]$ and a classical second order TVD scheme [229] with Rusanov flux has been used to solve the inhomogeneous system of equations (5.11) on a one-dimensional mesh of 15000 points in the radial interval $r \in [0; 1]$.
The ratio of specific heats is assumed to be $\gamma = 1.4$ and the final time is $t_f = 0.25$. Transmissive boundary conditions have been imposed on the external boundary and we use the Osher-type numerical flux (2.71). The solution

involves three different waves, namely one spherical shock wave traveling towards the external boundary of the domain, the rarefaction fan which is moving to the opposite direction and the contact wave in between, that is very well resolved due to the Lagrangian-like approach (see Section 6.1.3).

5.1.4 The Kidder problem

In [153] Kidder proposed this test problem, which has become a benchmark for Lagrangian schemes [53, 169]. It consists in an isentropic compression of a portion of a shell filled with a prefect gas which is assigned with the following initial condition:

$$\begin{pmatrix} \rho_0(r) \\ \mathbf{v}_0(r) \\ p_0(r) \end{pmatrix} = \begin{pmatrix} \left(\frac{r_{e,0}^2 - r^2}{r_{e,0}^2 - r_{i,0}^2} \rho_{i,0}^{\gamma-1} + \frac{r^2 - r_{i,0}^2}{r_{e,0}^2 - r_{e,0}^2} \rho_{e,0}^{\gamma-1} \right)^{\frac{1}{\gamma-1}} \\ 0 \\ s_0 \rho_0(r)^\gamma \end{pmatrix}, \tag{5.13}$$

where $r = \sqrt{x^2 + y^2 + z^2}$ represents the general radial coordinate, $(r_i(t), r_e(t))$ are the time-dependent internal and external frontier that delimit the shell, $\rho_{i,0} = 1$ and $\rho_{e,0} = 2$ are the corresponding initial values of density and $\gamma = 2$ or $\gamma = \frac{5}{3}$ is the ratio of specific heats in two and three space dimensions, respectively. Furthermore s_0 denotes the initial entropy distribution, that is assumed to be uniform, i.e. $s_0 = \frac{p_0}{\rho_0^\gamma} = 1$. If $d = 2$ we set $z = 0$, as usual.
The initial computational domain $\Omega(0)$ is either one fourth (in 2D) or one eighth (in 3D) of the entire shell and is depicted in Figure 5.3. Sliding wall boundary conditions are imposed on the lateral faces and on the bottom, while on the internal and on the external frontier a space-time dependent state is assigned according to the exact analytical solution $R(r, t)$ [153], which is defined at the general time t for a fluid particle initially located at radius r as a function of the radius and the homothety rate $h(t)$, i.e.

$$R(r, t) = h(t) r, \qquad h(t) = \sqrt{1 - \frac{t^2}{\tau^2}}, \tag{5.14}$$

where τ is the focalisation time

$$\tau = \sqrt{\frac{\gamma - 1}{2} \frac{(r_{e,0}^2 - r_{i,0}^2)}{c_{e,0}^2 - c_{i,0}^2}} \tag{5.15}$$

with $c_{i,e} = \sqrt{\gamma \frac{p_{i,e}}{\rho_{i,e}}}$ representing the internal and external sound speeds. As done in [53, 169], the final time of the simulation is chosen in such a way that the compression rate is $h(t_f) = 0.5$, hence $t_f = \frac{\sqrt{3}}{2}\tau$ and the the exact location of the shell is bounded with $0.45 \leq R \leq 0.5$.

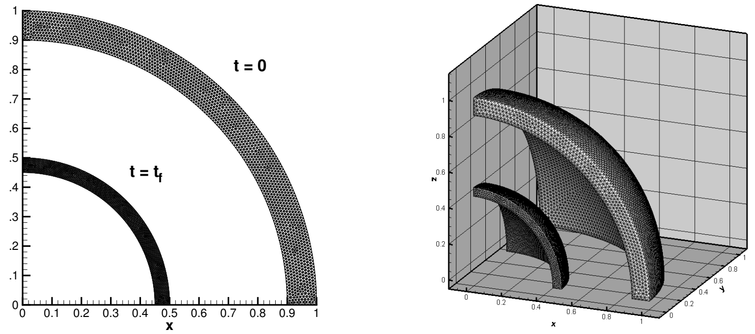


Figure 5.3: Position and mesh configuration of the shell at times $t = 0$ and at $t = t_f$ in two (left) and three (right) space dimensions.

The Osher-type flux (2.71) is adopted to perform the simulation of the Kidder problem, in order to precisely identified the location of the internal and external frontier. In the simulations of the Kidder problem we also measure the associated absolute error $|err|$, that can be evaluated as the difference between the analytical and the numerical location of the internal and external radius at the final time t_f (see Section 6.1.4).

5.1.5 The Saltzman problem

The Saltzman problem involves a strong shock wave that is caused by the motion of a piston traveling along the main direction of a rectangular box. This test case was first proposed in [91] for a two-dimensional Cartesian grid that has been skewed and it represents a very challenging test problem that allows the robustness of any Lagrangian scheme to be validated, because the mesh is not aligned with the fluid motion. According to [174], we consider the three-dimensional extension of the original problem [52, 91], hence the initial computational domain is the box $\Omega(0) = [0; 1] \times [0; 0.1] \times [0; 0.1]$. The computational mesh is obtained as follows:

- the domain is initially meshed with a uniform Cartesian grid composed by $100 \times 10 \times 10$ cubic elements, as done in [174];
- each cube is then split into five tetrahedra;
- finally we use the mapping given in [52,174] to transform the uniform grid, defined by the coordinate vector $\mathbf{x} = (x, y, z)$, to the skewed configuration $\mathbf{x}' = (x', y', z')$:

$$\begin{aligned} x' &= x + (0.1 - z)\,(1 - 20y)\sin(\pi x) \quad \text{for} \quad 0 \le y \le 0.05, \\ x' &= x + z\,(20y - 1)\sin(\pi x) \quad \text{for} \quad 0.05 < y \le 0.1, \\ y' &= y, \\ z' &= z. \end{aligned} \tag{5.16}$$

In 2D we consider the computational domain $\Omega(0) = [0; 1] \times [0; 0.1]$ and the computational mesh is composed of 200×20 triangular elements, obtained as follows:

- first we build a Cartesian mesh with 100×10 square elements, as done in [163, 169];
- each square element is then split into two right triangles;
- finally the uniform grid, defined by the coordinate vector $\mathbf{x} = (x, y)$, is skewed with the mapping

$$\begin{aligned} x' &= x + (0.1 - y)\sin(\pi x), \\ y' &= y, \end{aligned} \tag{5.17}$$

where x' and y' denote the deformed coordinates, respectively. The initial mesh configuration as well as the final mesh configuration for $d \in [2, 3]$ are depicted in Figure 5.4.
According to [163], the computational domain is filled with a perfect gas with the initial state $\boldsymbol{Q}_0$ given by

$$\boldsymbol{Q}_0 = (1, 0, 0, 0, \epsilon)\,. \tag{5.18}$$

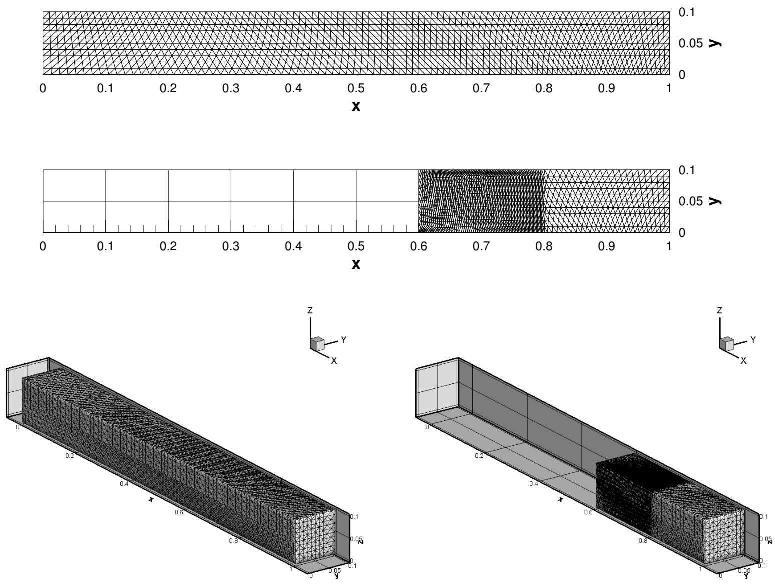


Figure 5.4: Initial and final mesh configuration for the Saltzman problem in 2D (top) and in 3D (bottom).

The ratio of specific heats is taken to be $\gamma = \frac{5}{3}$, $\epsilon = 10^{-4}$ and the final time is set to $t_f = 0.6$. The piston is traveling from the left to the right side of the domain with velocity $\boldsymbol{v_p} = (u_p, v_p, w_p) = (1, 0, 0)$ and it starts moving at the initial time while the gas is at rest. In the initial time steps the scheme must obey a geometric CFL condition, i.e. the piston must not move more than one element per time step. Sliding wall boundary conditions have been set everywhere, except for the piston, which has been assigned with moving slip wall boundary condition.

The exact solution $\boldsymbol{Q}_{ex}(\mathbf{x}, t_f)$ is a one-dimensional infinite strength shock wave and it can be computed by solving the Riemann problem given in Table 5.1.

The details of the algorithm that computes the exact solution of the Riemann problem are given in the book of Toro [229]. The exact solution has then to be shifted by a certain quantity d to the right, corresponding to the movement of the piston during the time of the simulation t_f, i.e.

$$d = u_p \cdot t_f. \tag{5.19}$$

	Left state	*Right state*
ρ	1.0	1.0
u	1.0	-1.0
v	0.0	0.0
w	0.0	0.0
p	$6.67 \cdot 10^{-7}$	$6.67 \cdot 10^{-7}$

Table 5.1: One-dimensional Riemann problem for obtaining the exact solution of the Saltzman problem.

It reads

$$\boldsymbol{Q}_{ex}(\mathbf{x}, t_f) = \begin{cases} (4,4,0,0,4) & \text{if} \quad x \leq x_f, \\ (1,0,0,0,\epsilon) & \text{if} \quad x > x_f, \end{cases} \tag{5.20}$$

where $x_f = 0.8$ is the shock location at time $t_f = 0.6$.

5.1.6 The Sedov problem

Another classical test case for hydrodynamics is the Sedov problem. We consider the spherical symmetric Sedov problem, which describes the evolution of a blast wave generated at the origin $\boldsymbol{O} = (x, y, z) = (0, 0, 0)$ of the initial cubic computational domain $\Omega(0) = [0; 1.2] \times [0; 1.2] \times [0; 1.2]$. It is a well-known test case for Lagrangian schemes [168,169,174] that becomes very challenging in the three-dimensional case. An analytical solution which is based on self-similarity arguments is furthermore available from the work of Kamm et al. [147]. The computational domain is first discretized with cubic elements, then each cube is split into five tetrahedra. The computational domain is filled with a prefect gas with $\gamma = 1.4$, which is initially at rest and is assigned with a uniform density $\rho_0 = 1$. The total energy E_{tot} is concentrated only in the cell c_{or} containing the origin $\boldsymbol{O}$, therefore the initial pressure is given by

$$p_{or} = (\gamma - 1)\rho_0 \frac{E_{tot}}{8 \cdot V_{or}}, \tag{5.21}$$

where V_{or} is the volume of the cell c_{or}, which is composed by five tetrahedra, and the factor $\frac{1}{8}$ takes into account the spherical symmetry, since the computational domain $\Omega(0)$ is only the eighth part of the entire domain, which would have to be considered if we did not assume the spherical symmetry. According to [168] we set $E_{tot} = 0.851072$, while in the rest of the domain the initial pressure is $p_0 = 10^{-6}$. At the final time of the simulation $t_f = 1.0$ the exact solution is a symmetric spherical shock wave located at radius $r = 1$ with a density peak of $\rho = 6$.
The two-dimensional Sedov initial condition can be set up as follows: the initial computational domain is now the square $\Omega(0) = [0; 1.2] \times [0; 1.2]$ and the initial mesh is initially composed by square elements, each of those has been split into two right triangles. The ratio of specific heats is again taken to be $\gamma = 1.4$, while the total amount of released energy is $E_{tot} = 0.244816$. The volume of the cell c_{or} is here composed by two triangles and the cylindric symmetry implies to consider a factor of $\frac{1}{4}$ in Eqn. (5.21).

5.1.7 The Noh problem

In [184] Noh proposed this test case in order to validate Lagrangian schemes in the regime of strong shock waves. The initial computational domain is given by $\Omega(0) = [0; 1.0]^d$ and the initial mesh is composed either by squares or by cubes, that are split into either two right triangles (in 2D) or five right tetrahedra (in 3D). A gas with $\gamma = \frac{5}{3}$ is initially assigned with a unity density $\rho_0 = 1$ and a unity radial velocity which is moving the gas towards the origin of the domain $O = (0, 0, 0)$. Hence, the velocity components are initialized with

$$u = -\frac{x}{r}, \qquad v = -\frac{y}{r}, \qquad w = -\frac{z}{r}, \tag{5.22}$$

and the initial pressure is $p = 10^{-6}$ everywhere. The generic radial position is given as usual as $r = \sqrt{x^2 + y^2 + z^2}$. As time advances, an outward moving cylindrical or spherical shock wave is generated which travels with velocity $v_{sh} = \frac{1}{3}$ in radial direction. According to [169, 174, 184], the final time is chosen to be $t_f = 0.6$, therefore the shock wave is located at radius $R = 0.2$ and the maximum density value is either $\rho_f = 16$ ($d = 2$) or $\rho_f = 64$ ($d = 3$), which occurs on the plateau behind the shock wave. Since the problem is set up in order to take into account cylindrical or spherical symmetry, we impose no-slip wall boundary conditions on the *logically* internal faces of the domain, while moving boundaries have been used on the remaining sides.

5.1.8 The Gresho vortex problem

We also consider the well-known Gresho vortex problem [135] in the variant proposed by Vilar et al. [128]. This test problem involves a *steady* vortex flow, which is characterized by a perfect balance between inertia and pressure gradient in the momentum equation of system (2.1)-(5.1). The mathematical procedure to derive such a solution is fully described in [128], hence we provide here only the information needed to setup this test case. In *two* space dimensions the initial computational domain is the square $\Omega(0) = [-0.5; 0.5] \times [-0.5; 0.5]$. The initial condition $\boldsymbol{U}(\boldsymbol{x}, 0)$ is given in terms of primitive variables and reads

$$
\begin{aligned}
\rho_0 &= 1.0, \\
u_0 &= \begin{cases} -5y & \text{if} \quad r < 0.2 \\ -\left(\frac{2}{r} - 5\right) y & \text{if} \quad 0.2 \leq r < 0.4 \\ 0.0 & \text{if} \quad r \geq 0.4 \end{cases}, \\
v_0 &= \begin{cases} 5x & \text{if} \quad r < 0.2 \\ \left(\frac{2}{r} - 5\right) x & \text{if} \quad 0.2 \leq r < 0.4 \\ 0.0 & \text{if} \quad r \geq 0.4 \end{cases}, \\
p_0 &= \begin{cases} 5 + 12.5r^2 & \text{if} \quad r < 0.2 \\ 9 - 4\log(0.2) + 12.5r^2 - 20r + 4\log(r) & \text{if} \quad 0.2 \leq r < 0.4 \\ 3 + 4\log(2) & \text{if} \quad r \geq 0.4 \end{cases},
\end{aligned}
\tag{5.23}
$$

with $r = \sqrt{x^2 + y^2}$ denoting the generic radial position. Since the vortex flow is stationary, the analytical solution is simply given by the initial condition. The Gresho vortex test case is typically used to asses the robustness and the accuracy of a Lagrangian scheme, since strong mesh deformations occur during the simulation. As done in [128] we set the final time $t_f = 1.0$ so that the vortex does one complete rotation and we impose no-slip wall boundary conditions everywhere.

5.1.9 The Taylor-Green vortex problem

The Taylor-Green test case consists in a smooth vortex flow and was presented for the two-dimensional case by Dobrev et al. [83], who considered an analytical solution of the incompressible Navier-Stokes equations and proposed a modified version which applies to the Euler equations of compressible gas dynamics. All the details on the construction of this test case can be found in the above mentioned reference, hence we limit us in the following to recalling only the necessary information to setup the test problem in *three* space dimensions, with the cubic computational domain defined by $\Omega(0) = [0;1.0] \times [0;1.0] \times [0;0.25]$. The initial condition is given in terms of primitive variables and reads

$$\boldsymbol{U}(\boldsymbol{x},0) = \begin{pmatrix} \rho_0 \\ u_0 \\ v_0 \\ w_0 \\ p_0 \end{pmatrix} = \begin{pmatrix} \rho_0 \\ U_0\left[sin(\pi x)cos(\pi y)\right] \\ U_0\left[-cos(\pi x)sin(\pi y)\right] \\ 0.0 \\ \frac{1}{4}\rho_0 U_0^2\left[cos(2\pi x)+cos(2\pi y)\right]+C_0 \end{pmatrix}, \tag{5.24}$$

with $\rho_0 = 1$, $U_0 = 1$ and $C_0 = 1$, according to [128]. The velocity field is initially divergence-free and in order to satisfy the total energy equation of the Euler system we have to insert an additional source term for the last equation, hence obtaining

$$\boldsymbol{S}(\boldsymbol{Q}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \frac{\pi}{4}\frac{\rho_0 U_0^3}{\gamma-1}\left[cos(3\pi x)cos(\pi y) - cos(3\pi y)cos(\pi x)\right] \end{pmatrix}, \tag{5.25}$$

where the ratio of specific heats is chosen to be $\gamma = 1.4$, as done in [128]. Periodic boundaries are set on the top and on the bottom of the domain, while we impose no-slip wall boundary conditions on the lateral sides. The final time of the simulation is $t_f = 0.7$ and the exact solution is simply given by the initial condition, since this test problem involves a *steady* flow.

5.1.10 The two-dimensional double Mach reflection problem

We propose to solve a very difficult and challenging test case, namely the double Mach reflection problem (DMR), which was originally proposed in [243]. This test problem involves a very strong shock wave with a shock Mach number

$M_s = 10$ which is moving along the $x-$direction of the computational domain, where a ramp with angle $\alpha = \frac{\pi}{6}$ is located. The strong shock wave that hits the ramp leads to the development of small-scale structures that are highly deforming the computational mesh. The initial computational domain $\Omega(0)$ is discretized with a characteristic mesh size of $h = 1/50$ (similar to [163]), with a total number of elements of $N_E = 41456$ and is depicted in Figure 5.5.
The initial condition $\boldsymbol{U}(\mathbf{x}, 0)$ is given in terms of primitive variables and reads

$$\boldsymbol{U}(\mathbf{x}, 0) = (\rho_0, u_0, v_0, p_0) = \begin{cases} (8.0, 8.25, 0, 116.5)\,, & \text{if} \quad x < x_0, \\ (1.4, 0, 0, 1.0)\,, & \text{if} \quad x \geq x_0, \end{cases} \tag{5.26}$$

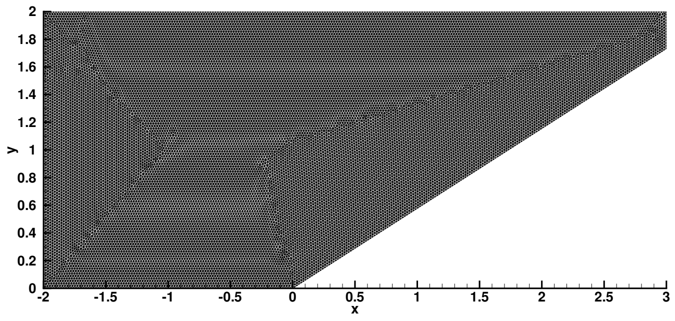
with the initial discontinuity located at $x_0 = 0$. Sliding wall boundary conditions have been imposed on the upper and the lower side of the domain, while transmissive boundaries are set on the remaining sides. The final time is chosen to be $t_f = 0.2$ and the ratio of specific heats is $\gamma = 1.4$.

5.1.11 Mono-material triple point problem

This problem is a three state one material 2D Riemann problem in a vessel, which is a variation of the triple point problem from [166]. We first present the two-dimensional setup, then we consider the extension to three space dimensions.
The triple point problem computational domain is $\Omega(0) = [0; 7] \times [0; 3]$ in 2D, which is split into three subdomains filled with a perfect gas with $\gamma = 1.4$. The high pressure high density state in $\Omega_1(0) = [0; 1] \times [0; 3]$ is defined by $(\rho_1, p_1) = (1, 1)$, the low pressure high density state in $\Omega_2(0) = [1; 7] \times [0; 1.5]$ is given by $(\rho_2, p_2) = (0.125, 0.1)$, the low pressure low density in $\Omega_3(0) = [1; 7] \times [1.5; 3]$ is $(\rho_3, p_3) = (1.0, 0.1)$, see Figure 5.6. The initial gas is at rest and the final time of the simulation is $t_f = 5.5$.
The 3D triple point problem [83,84] is constructed by considering the 2D setup, detailed above, as a 2D axi-symmetric $r - z$ setup, with r denoting the radial coordinate of the vessel according to Figure 5.6. Consequently subdomain Ω_3 becomes an internal cylinder to cylindrical subdomain Ω_2. Subdomain Ω_1 (blue colored) can be seen as a cylindrical cap to these embedded cylinders.

5.1.12 Multi-material flow

One of the major advantages in adopting a Lagrangian approach is that material interfaces can be precisely located, since the computational mesh moves



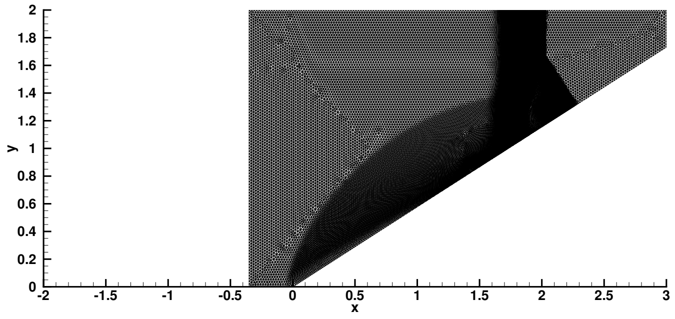


Figure 5.5: Computational domain and mesh at the initial time $t = 0$ (top) and at the final time $t = 0.2$ (bottom) for the double Mach reflection problem.

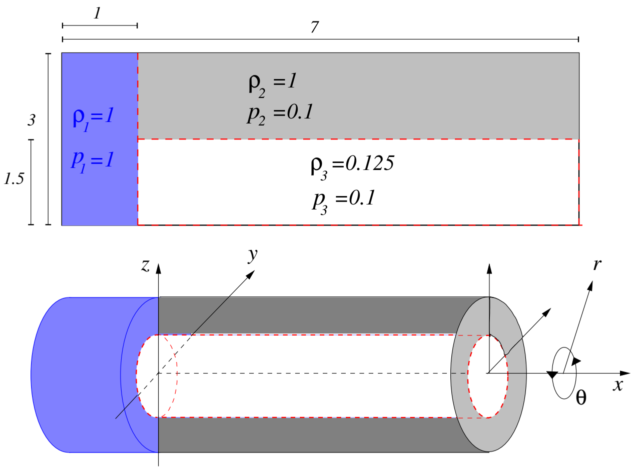


Figure 5.6: Sketch of the triple point problem initialization. Top: 2D setup. Bottom: 3D axi-symmetric version.

together with the fluid and the resulting numerical scheme is typically less dissipative than a classical Eulerian method. Such an advantage becomes even more evident when multi-material flows are considered, as done in the following. As we will show later, our direct ALE ADER-WENO schemes can be successfully applied to multi-material flows in the framework of diffusive interface methods using the seven equation Baer-Nunziato model for compressible two-phase flow, see [96] and Section 5.4. Here we would like to present an alternative way to consider multi-material flows, hence focusing on the simulation of two-material fluids using the Euler equations for compressible gas dynamics (5.1).

If the Lagrangian method is designed in order to assure no mass flux through the edges of the computational control volumes [47, 130, 166], then no mixing cells occur, i.e. cells which contain more than one material. Nevertheless the mixing might be generated by the rezoning and remapping procedure that is normally carried out in order to improve the quality of the mesh during the computation. Due to the direct Arbitrary-Lagrangian-Eulerian (ALE) formulation of our scheme, that in principle allows mass flux, we have to deal with mixing cells which require an appropriate thermodynamical closure. We propose to use the so-called *concentration approach*, in which the multi-material flow is considered as a mixture of miscible fluids. This assumption is normally used for plasma flows, i.e. gases at very high temperatures. Each component k is assumed to be an ideal gas which obeys the equation of state (5.2) with its own ratio of specific heats γ_k. According to [130] we assume a pressure-temperature equilibrium in order to define the effective polytropic index of the mixture as

$$\gamma = 1 + \frac{\frac{\phi_1}{M_1} + \frac{\phi_2}{M_2}}{\frac{\phi_1}{(\gamma_1-1)M_1} + \frac{\phi_2}{(\gamma_2-1)M_2}}, \tag{5.27}$$

where M_1, M_2 are the molar masses of the materials that are supposed to be the same, i.e. $M_1 = M_2$, while ϕ_1, ϕ_2 denote the mass fractions. Mixing cells can be properly treated using the above-defined effective ratio of specific heats γ together with the closure equation (5.2). Please note that where no mixing occurs, i.e. one of the two mass fraction is zero, expression (5.27) reduces to the corresponding polytropic index of the phase and the ideal single fluid behavior is fully recovered. For each fluid k the corresponding mass fraction ϕ_k, which stands for its concentration, is computed according to [130] as the ratio between the mass of fluid m_k and the total mass of the mixture m, i.e.

$$\phi_k = \frac{m_k}{m}, \tag{5.28}$$

with the concentrations for the two materials that sum to the unity, i.e. $\phi_1 + \phi_2 = 1$. Therefore the mass fraction can be viewed as a passive scalar, which allows to track the location of each material inside the flow. In order to take into account the evolution of ϕ_k in our direct ALE scheme, an additional transport equation must be added to the Euler system (5.1), hence

$$\frac{\partial \rho\phi_1}{\partial t} + \frac{\partial \rho\phi_1 u}{\partial x} + \frac{\partial \rho\phi_1 v}{\partial y} = 0, \tag{5.29}$$

where the additional conserved variable is given by $\rho\phi_1$. We point out that the choice of the Riemann solver plays a crucial role in our direct ALE method, since the mixing depends on the numerical dissipation produced by the numerical fluxes applied at the cell edges. In order to reduce as much as possible the number of mixing cells in the computation, we rely on the Osher-type numerical flux (2.71), which exactly preserves moving isolated contact waves without mass flux. Note that the solution of the Euler equations together with Eqn. (5.29) may lead to spurious pressure oscillations, which can be treated using the approach presented in [5]. Since we do not explicitly reconstruct the material interface, our method remains a *diffuse interface approach*, subject to numerical diffusion like the one used previously in [96].

The two-material Sod problem. We propose to solve the well-known Sod shock tube problem in the two-material variant described in [214]. As the original single-fluid Sod problem (see Section 5.1.2), this one-dimensional test case involves three different kinds of waves, namely one rarefaction wave, one shock wave and one contact wave which separates the two fluids. The initial computational domain is the box $\Omega(0) = [-0.5; 0.5] \times [-0.05; 0.05]$ which is meshed with a characteristic mesh size of $h = 1/100$, while the initial condition $\boldsymbol{U}(\boldsymbol{x}, 0)$ for the two materials is given in terms of primitive variables (ρ, u, v, p, ϕ_1) and reads

$$\boldsymbol{U}(\boldsymbol{x}, 0) = \begin{cases} \boldsymbol{U}_L = (1.0, 0, 0, 2.0, 1.0 - \epsilon), & \text{if} \quad x \leq x_0, \\ \boldsymbol{U}_R = (0.125, 0, 0, 0.1, \epsilon), & \text{if} \quad x > x_0 \end{cases}, \tag{5.30}$$

where the initial discontinuity between the two fluids is located at $x_0 = 0$ and we set the parameter $\epsilon = 10^{-6}$. The ratio of specific heats for the left and right material are $\gamma_1 = 2.0$ and $\gamma_2 = 1.4$, respectively, according to [130, 214], and the final time of the simulation is chosen to be $t_f = 0.2$. Periodic boundary conditions have been imposed in the y-direction and transmissive boundaries in x-direction. The exact solution can be obtained by the exact Riemann solver for the Euler equations of compressible gas dynamics [229] using two different values of γ on the left and on the right of the contact discontinuity.

5.2 The ideal magnetohydrodynamics (MHD) equations

The equations of ideal classical magnetohydrodynamics (MHD) are used to describe the motion of charged fluids like *plasma fluids*. They constitute a more complicated hyperbolic conservation law compared to the Euler equations presented in Section 5.1, especially because this system introduces an additional constraint regarding the divergence of the magnetic field that must remain zero in time, i.e.

$$\nabla \cdot \boldsymbol{B} = 0. \tag{5.31}$$

If the magnetic field $\boldsymbol{B}$ is initialized with data that are guaranteed to be divergence-free, then Eqn. (5.31) is always satisfied for the exact solution.

The difficulty appears at the discrete level, where the numerical divergence-free constraint has to be carefully taken into account and properly treated. To overcome this problem, we adopt the hyperbolic version of the generalized Lagrangian multiplier (GLM) divergence cleaning approach proposed by Dedner et al. [75], hence adding to the MHD system one more variable Ψ as well as one more linear scalar PDE that aims at transporting the divergence errors out of the computational domain with an artificial divergence cleaning speed c_h. The augmented MHD system can be cast into form (2.1) and reads

$$\frac{\partial}{\partial t}\begin{pmatrix} \rho \\ \rho \mathbf{v} \\ \rho E \\ \boldsymbol{B} \\ \psi \end{pmatrix} + \nabla \cdot \begin{pmatrix} \rho \mathbf{v} \\ \rho \mathbf{v}\mathbf{v} + p_{tot}\boldsymbol{I} - \frac{1}{4\pi}\boldsymbol{B}\boldsymbol{B} \\ \mathbf{v}(\rho E + p_{tot}) - \frac{1}{4\pi}\boldsymbol{B}(\mathbf{v}\cdot\boldsymbol{B}) \\ \mathbf{v}\boldsymbol{B} - \boldsymbol{B}\mathbf{v} + \psi\boldsymbol{I} \\ c_h^2\boldsymbol{B} \end{pmatrix} = 0. \tag{5.32}$$

The non-conservative part of the ideal MHD system is zero, the velocity vector is denoted by $\boldsymbol{v} = v_i = (u, v, w)$ and similarly the vector of the magnetic field is addressed with $\boldsymbol{B} = B_i = (B_x, B_y, B_z)$. The system is then closed by the equation of state

$$p = (\gamma - 1)\left(\rho E - \frac{1}{2}\mathbf{v}^2 - \frac{\boldsymbol{B}^2}{8\pi}\right), \tag{5.33}$$

with γ representing the ratio of specific heats and the total pressure being defined as $p_{tot} = p + \frac{1}{8\pi}\boldsymbol{B}^2$.
We define also the fastest magnetosonic speed, needed for the node solver $\mathcal{NS}_m$ presented in Section 3.1.2 of Chapter 3. It reads

$$c = \sqrt{\frac{1}{2}\left(\frac{\gamma p}{\rho} + (B_x + B_y + B_z) + \sqrt{\left(\frac{\gamma p}{\rho} + (B_x + B_y + B_z)\right)^2 - 4\frac{\gamma p}{\rho}\frac{B_x^2}{4\pi\rho}}\right)}. \tag{5.34}$$

5.2.1 Numerical convergence studies

We use the convected smooth vortex test problem proposed by Balsara et al. [16] in order to carry out the numerical convergence studies for the ideal classical MHD equations (5.32) in *two* space dimensions. This test case is defined on a square computational domain $\Omega(0) = [0; 10] \times [0; 10]$ with periodic boundaries everywhere. As for the hydrodynamic isentropic vortex presented in Section

5.1.1, the initial condition is given by a linear superposition of a constant flow and some fluctuations in the velocity and magnetic fields, which read

$$(\rho, u, v, p, B_x, B_y, \Psi) = (1+\delta\rho, 1+\delta u, 1+\delta v, 1+\delta p, 1+\delta B_x, 1+\delta B_y, 0), \quad (5.35)$$

with the following perturbations:

$$\begin{bmatrix} \delta u \\ \delta v \\ \delta p \\ \delta B_x \\ \delta B_y \end{bmatrix} = \begin{bmatrix} \frac{\epsilon}{2\pi} e^{\frac{1}{2}(1-r^2)}(5-y) \\ \frac{\epsilon}{2\pi} e^{\frac{1}{2}(1-r^2)}(x-5) \\ \frac{1}{8\pi}\left(\frac{\mu}{2\pi}\right)^2 (1-r^2) e^{(1-r^2)} - \frac{1}{2}\left(\frac{\epsilon}{2\pi}\right)^2 e^{(1-r^2)} \\ \frac{\mu}{2\pi} e^{\frac{1}{2}(1-r^2)}(5-y) \\ \frac{\mu}{2\pi} e^{\frac{1}{2}(1-r^2)}(x-5) \end{bmatrix}. \quad (5.36)$$

According to [16], we set the parameters $\epsilon = 1$ and $\mu = \sqrt{4\pi}$ as well as the ratio of specific heats $\gamma = \frac{5}{3}$. The speed for the divergence cleaning is taken to be $c_h = 2$ and the velocity $\mathbf{v}_c = (1, 1)$ convects the vortex. The fluid motion of the vortex would lead to high element distortions and deformations, as clearly depicted in Figure 5.7, therefore the final time of the computation is $t_f = 1.0$, because we do not want the rezoning step to be used for the convergence rate studies. The exact solution is given by the initial condition shifted in space by a factor $\boldsymbol{s} = (s_x, s_y) = \mathbf{v} \cdot t_f$.

5.2.2 The MHD rotor problem

The first test case for the ideal classical MHD equations is the MHD rotor problem proposed by Balsara et al. in [20]. It consists in a fluid of high density that is rotating very quickly, surrounded by a fluid at rest with low density. The initial computational domain $\Omega(0)$ is a sphere of radius $R_0 = 0.5$ and transmissive boundary conditions are imposed at the external boundary. The generic radial position is denoted by $r = \sqrt{x^2 + y^2 + z^2}$ and at radius $R = 0.1$ the *inner* region with the high density fluid is separated by the *outer* region. Therefore the initial density distribution is $\rho = 10$ for $0 \le r \le R$ and $\rho = 1$ in the rest of the domain, while the angular velocity ω of the rotor is assumed to be constant and it is chosen in such a way that at $r = R$ the toroidal velocity is $v_t = \omega \cdot R = 1$. The initial discontinuity for density and velocity occurring at the frontier $r = R$ is smeared out according to [20], where a linear taper bounded by $0.1 \le r \le 0.13$ is applied in such a way that the internal values for density and velocity match exactly those ones of the outer region. The pressure is $p = 1$ in the whole computational domain and a constant magnetic

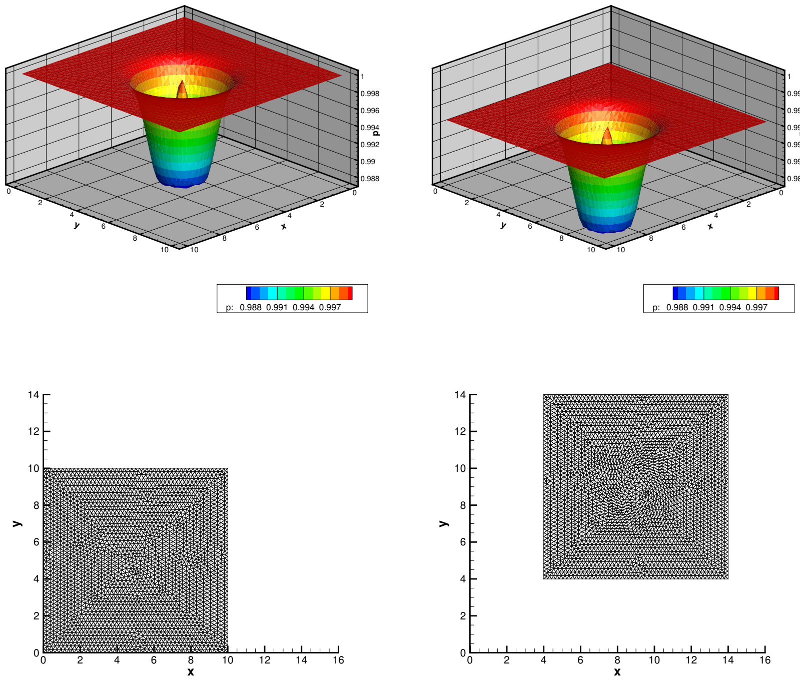


Figure 5.7: Top: pressure distribution for the ideal MHD vortex problem at time $t = 0.0$ and $t = 4.0$. Bottom: mesh configuration at time $t = 0.0$ and $t = 4.0$.

field $\boldsymbol{B} = (2.5, 0, 0)^T$ is imposed everywhere. The divergence cleaning velocity is taken to be $c_h = 2$, while the ratio of specific heats is set to $\gamma = 1.4$ and the final time is $t_f = 0.25$. In two space dimensions the computational domain reduces to a circle and the z coordinate as well as all its related quantities disappear.

5.2.3 The MHD blast wave problem

The blast wave problem constitutes a benchmark in magnetohydrodynamics. A strong circular fast magnetosonic shock wave is traveling from the center to the boundaries of the initial computational domain $\Omega(0)$, which is a sphere (in 3D) or a circle (in 2D) of radius $R_0 = 0.5$. The frontier delimited by radius $R = 0.1$ splits the domain into two parts, hence defining an *inner* state $\boldsymbol{U}_i$ and an *outer* state $\boldsymbol{U}_o$, that are initially assigned in terms of primitive variables $\boldsymbol{U} = (\rho, u, v, w, p, B_x, B_y, B_z, \psi)$ as

$$\boldsymbol{U}(\mathbf{x}, 0) = \begin{cases} \boldsymbol{U}_i = (1.0, 0.0, 0.0, 0.1, 70, 0.0, 0.0) & \text{if} \quad r \leq R, \\ \boldsymbol{U}_o = (1.0, 0.0, 0.0, 1000, 70, 0.0, 0.0) & \text{if} \quad r > R, \end{cases} \tag{5.37}$$

where $r = \sqrt{x^2 + y^2 + z^2}$. We set transmissive boundary conditions at the external boundary. The final time of the computation is $t_f = 0.01$ and the ratio of specific heats is taken to be $\gamma = 1.4$.

5.3 The relativistic MHD equations (RMHD)

For very high flow velocities, i.e. $|\mathbf{v}|/c_l \gg 1$ with c_l denoting the speed of light, we have to consider the relativistic MHD equations (RMHD). All the details regarding this physical model can be found in [15, 132, 197, 245]. Let ρ be the density and $\mathbf{v} = (u, v, w)$ be the velocity vector, then p is the hydrodynamic pressure while p_{tot} is the total pressure, obtained adding to p also the contribution of the magnetic pressure. Furthermore let e represent the internal energy, E the total energy and let denote the magnetic field with $\boldsymbol{B} = (B_x, B_y, B_z)$ and the Lorenz factor with γ, while the ratio of specific heats will be addressed with the symbol Γ *only* when considering the RMHD equations. Again we take care of the divergence constraint for the magnetic field using the hyperbolic divergence-cleaning approach of Dedner et al. [75], as done for the ideal MHD equations presented in Section 5.2. The vector of conserved variables of the RMHD system reads

$$\boldsymbol{Q} = \begin{pmatrix} D \\ \boldsymbol{q} \\ E \\ \boldsymbol{B} \\ \Psi \end{pmatrix} = \begin{pmatrix} \gamma\rho \\ \gamma w_{tot} \mathbf{v} - b^0 \boldsymbol{b} \\ \gamma^2 w_{tot} - b^0 b^0 - p_{tot} \\ \boldsymbol{B} \\ \Psi \end{pmatrix}, \tag{5.38}$$

and the flux tensor $\boldsymbol{F}(\boldsymbol{Q})$ is given by

$$\boldsymbol{F} = \begin{pmatrix} \gamma\rho\mathbf{v}^T \\ \gamma^2 w_{tot}\mathbf{v}\mathbf{v} - \boldsymbol{b}\boldsymbol{b} + p_{tot}\boldsymbol{I} \\ \gamma^2 w_{tot}\mathbf{v}^T - b^0\boldsymbol{b}^T \\ \mathbf{v}\boldsymbol{B} - \boldsymbol{B}\mathbf{v} + \Psi\boldsymbol{I} \\ c_h^2\boldsymbol{B}^T \end{pmatrix}. \tag{5.39}$$

Here, $\boldsymbol{I}$ is the identity matrix, the enthalpy w_{tot} and the total pressure p_{tot} are defined as

$$w_{tot} = e + p + |b|^2, \qquad p_{tot} = p + \frac{1}{2}|b|^2, \tag{5.40}$$

where the internal energy is given by the following equation of state

$$e = \rho + \frac{p}{\Gamma - 1}. \tag{5.41}$$

The Lorenz factor is

$$\gamma = \frac{1}{\sqrt{1 - \mathbf{v}^2}}, \tag{5.42}$$

and the other quantities appearing in (5.39) are

$$b^0 = \gamma\left(\mathbf{v}\cdot\boldsymbol{B}\right), \quad \boldsymbol{b} = \frac{\boldsymbol{B}}{\gamma} + \gamma\mathbf{v}\left(\mathbf{v}\cdot\boldsymbol{B}\right), \quad |b^2| = \frac{\boldsymbol{B}^2}{\gamma} + \left(\mathbf{v}\cdot\boldsymbol{B}\right)^2. \tag{5.43}$$

We assume a speed of light normalized to unity. The computation of the primitive variables $\boldsymbol{U} = (\rho, \mathbf{v}, p, \boldsymbol{B})$ from the conserved quantities $\boldsymbol{Q}$ has to be done *numerically*, by using an iterative Newton or bisection method, as explained in [95, 245].
In the present work, only the *two-dimensional* version of the RMHD equations is considered. For all simulations we use the node solver $\mathcal{NS}_{cs}$ (see Section 3.1.1 of Chapter 3) for the calculation of the mesh velocity, due to its simple and very general formulation, which allows this node solver to be applied to any general nonlinear hyperbolic conservation law. This flexibility is not available with the other two node solvers presented in Chapter 3, which have to be designed specifically for each hyperbolic system under consideration.

5.3.1 Large Amplitude Alfvén wave

The relativistic MHD equations are an extremely challenging and highly non-linear hyperbolic system, for which the development of accurate and robust numerical methods is very difficult. To assess the accuracy of our high order cell-centered one-step direct ALE ADER-WENO finite volume schemes we perform a numerical convergence study of the third, fourth and fifth order version of our scheme on a very nice time-dependent test case proposed originally by Del Zanna et al. in [246] and which has subsequently also been used for the assessment of other high order schemes in [95,108,109,191]. It consists in a space-time periodic Alfvén wave with large amplitude. The initial condition for the primitive variables is chosen as the exact solution of the problem at time $t = 0$. In particular, one has $\rho = p = 1$, $u = B_x = \Psi = 0$, $B_y = \eta B_0 \cos(kx - v_A t)$, $B_z = \eta B_0 \sin(kx - v_A t)$ and $v = -v_A B_y / B_0$, $w = -v_A B_z / B_0$. We use the wavenumber $k = 2\pi$, the 2D computational domain is $\Omega = [0; 1] \times [-0.1; +0.1]$ with four periodic boundary conditions and $\Gamma = \frac{5}{3}$. With these parameters and $B_0 = \eta = 1$, the speed of the Alfvén wave in positive x-direction is $v_A = 0.433892047069424$, see [246] for a closed analytical expression for v_A. The final time is set to $t = 0.5$ and the mesh velocity is defined as $\boldsymbol{V} = (\frac{1}{10}\left(1 + \cos(\pi x)^2\right., 0)$ so that the total computational domain $\Omega(t)$ remains constant in time and the periodic boundary conditions can be applied. In the ALE framework proposed in this work, the mesh velocity can indeed be chosen independently of the fluid velocity. Due to the smooth mesh motion imposed here, a rezoning strategy is not needed for this test problem. Note that in the RMHD system, the minimum and maximum eigenvalues of the ALE Jacobian must remain between $\lambda_{\min} = -1$ and $\lambda_{\max} = +1$, since the relativistic MHD equations are no longer Galilean invariant as the previous PDE systems based on classical Newtonian mechanics. In all the computations we use a Courant number of 0.5.

5.3.2 The RMHD rotor problem

The initially circular computational domain is of radius $R_0 = 0.5$. As for the ideal MHD rotor problem, radius $R = 0.1$ splits again the domain into an internal and an external region. The rotor, which is in the internal region, is here spinning with an angular frequency of $\omega = 8$, hence yielding the maximal toroidal velocities of $v_t = (\omega \cdot R) = 0.8$. The initial density is $\rho = 1$ in the external region and $\rho = 10$ in the inner state, while the pressure $p = 1$ is constant throughout the entire computational domain, as well as the magnetic field $\boldsymbol{B} = (1, 0, 0)$. We use again the taper described in Section 5.2.2. The speed of divergence-cleaning is set to $c_h = 1$ and the ratio of specific heats is taken to be $\Gamma = \frac{5}{3}$. For this test problem, the rezoning step is necessary according to Section 3.2.

5.3.3 The RMHD blast wave problem

This problem is similar to the classical MHD blast wave problem described in Section 5.2.3. It was also used in the context of resistive RMHD equations in [109]. The initial computational domain is again a circle of radius $R_0 = 0.5$ and the initial condition reads

$$\boldsymbol{Q}(\mathbf{x},0) = \begin{cases} \boldsymbol{Q}_i & \text{if} \quad r \leq R, \\ \boldsymbol{Q}_o & \text{if} \quad r > R. \end{cases} \tag{5.44}$$

The *inner state* is defined in the central circle of radius $R = 0.1$, while the *outer state* $\boldsymbol{Q}_o$ is defined outside. We assume $\gamma = 4/3$ and the final time of the simulation is chosen to be $t_f = 0.3$. The divergence cleaning speed is $c_h = 1$. We use the initial condition reported in Table 5.2 and transmissive boundary conditions are imposed everywhere.

Table 5.2: Initial condition for the RMHD blast wave problem.

	ρ	u	v	w	p	B_x	B_y	B_z	Ψ
Inner state ($\boldsymbol{Q}_i$)	1.0	0.0	0.0	0.0	1.0	0.05	0.0	0.0	0.0
Outer state ($\boldsymbol{Q}_o$)	1.0	0.0	0.0	0.0	10^{-3}	0.05	0.0	0.0	0.0

5.4 The Baer-Nunziato model of compressible two-phase flows

Multi-phase flow problems, such as liquid-vapor and solid-gas flows are encountered in numerous natural processes, such as avalanches, meteorological flows with cloud formation, volcano explosions, sediment transport in rivers and on the coast, granular flows in landslides, etc., as well as in many industrial applications, e.g., in aerospace engineering, automotive industry, petroleum and chemical process engineering, nuclear reactor safety, paper and food manufacturing and renewable energy production. Most of the industrial applications are concerned with *compressible* multi-phase flows as they appear for example in combustion processes of liquid and solid fuels in car, aircraft and rocket engines, but also in solid bio-mass combustion processes. Already the mathematical description of such flows is quite complex and up to now there is no universally agreed model for such flows. One wide-spread model is the

Baer-Nunziato model for compressible two-phase flow, which has been developed by Baer and Nunziato in [12] for describing detonation waves in solid-gas combustion processes. High resolution shock capturing finite volume schemes combined with a stiff relaxation approach have been successfully applied to this system by Saurel and Abgrall in [9, 204]. In this work we will use the original choice of Baer-Nunziato, which has also been adopted in several papers about the exact solution of the Riemann-Problem of the Baer-Nunziato model, see [11, 76, 212]. A reduced five-equation model has been proposed in [148], for the solution of which a Godunov type scheme has been presented in [183]. Approximate Riemann solvers of Baer-Nunziato-type models of compressible multi-phase flows can be found for example in [99,107,221,226]. Numerical schemes for compressible multi phase flows on moving meshes have been considered for the one-dimensional case in [207] and an efficient Eulerian approach on fixed unstructured grids has been proposed in [8]. For further work on numerical methods for compressible multiphase flows see, for example, [6,10,176,185,195,206,208,209,247].

The first phase is normally addressed as the *solid* phase, while the second one as the *gas* phase and in this thesis we will use the subscripts 1 and 2 to define them. We will use equivalently also the subscripts s and g to denote the solid and the gas phase. Let $k = 1, 2$ be the phase number and ϕ_k be the volume fraction of phase k with the condition $\phi_1 + \phi_2 = 1$, while ρ_k and p_k represent the corresponding density and pressure, respectively. Let furthermore the velocity vector of each phase be addressed with $\mathbf{u}_k = (u_k, v_k, w_k)$. The full seven-equation Baer-Nunziato model with relaxation source terms results in a *non-conservative system* of nonlinear hyperbolic PDE that can be written as

$$\left.\begin{array}{l}
\frac{\partial}{\partial t}\left(\phi_1\rho_1\right) + \nabla\cdot\left(\phi_1\rho_1\mathbf{u_1}\right) = 0,\\
\frac{\partial}{\partial t}\left(\phi_1\rho_1\mathbf{u_1}\right) + \nabla\cdot\left(\phi_1\rho_1\mathbf{u_1}\mathbf{u_1}\right) + \nabla\phi_1 p_1 = p_I\nabla\phi_1 - \lambda\left(\mathbf{u_1} - \mathbf{u_2}\right),\\
\frac{\partial}{\partial t}\left(\phi_1\rho_1 E_1\right) + \nabla\cdot\left(\left(\phi_1\rho_1 E_1 + \phi_1 p_1\right)\mathbf{u_1}\right) = -p_I\partial_t\phi_1 - \lambda\,\boldsymbol{u_I}\cdot\left(\boldsymbol{u_1} - \boldsymbol{u_2}\right),\\
\frac{\partial}{\partial t}\left(\phi_2\rho_2\right) + \nabla\cdot\left(\phi_2\rho_2\mathbf{u_2}\right) = 0,\\
\frac{\partial}{\partial t}\left(\phi_2\rho_2\mathbf{u_2}\right) + \nabla\cdot\left(\phi_2\rho_2\mathbf{u_2}\mathbf{u_2}\right) + \nabla\phi_2 p_2 = p_I\nabla\phi_2 - \lambda\left(\mathbf{u_2} - \mathbf{u_1}\right),\\
\frac{\partial}{\partial t}\left(\phi_2\rho_2 E_2\right) + \nabla\cdot\left(\left(\phi_2\rho_2 E_2 + \phi_2 p_2\right)\mathbf{u_2}\right) = p_I\partial_t\phi_1 - \lambda\,\boldsymbol{u_I}\cdot\left(\boldsymbol{u_2} - \boldsymbol{u_1}\right),\\
\frac{\partial}{\partial t}\phi_1 + \mathbf{u_I}\nabla\phi_1 = \mu(p_1 - p_2),
\end{array}\right\} \tag{5.45}$$

where only strongly simplified interphase drag and pressure relaxation source terms are considered. Further details on the choice and the formulation of such terms can be found in [148]. The so-called stiffened gas equation of state is then used for each of the two phases to close the system:

$$e_k = \frac{p_k + \gamma_k \pi_k}{\rho_k(\gamma_k - 1)}. \tag{5.46}$$

The specific total energy of each phase is $E_k = e_k + \frac{1}{2}\boldsymbol{u_k}^2$ with e_k denoting the corresponding internal energy, while in the system (5.45) μ is a parameter which characterizes pressure relaxation and λ is related to the friction between the phases. According to [12, 148] the velocity at the interface I is taken to be the *solid* velocity, while for the interface pressure we choose the *gas* pressure, hence

$$\boldsymbol{u_I} = \boldsymbol{u_1} \qquad p_I = p_2. \tag{5.47}$$

Other choices are possible, see [204, 205] for a detailed discussion.
The resolution of material interfaces, which are given by jumps in the volume fraction ϕ_k, is a challenging task for the numerical methods applied to the Baer-Nunziato model (5.45). We stress that the present approach is a so-called *diffuse interface approach*, which may not be suitable for all situations occurring in the simulation of multi-fluid and multi-material problems. For so-called *sharp interface approaches*, the reader is referred to aforementioned references.

5.4.1 Numerical convergence studies

The numerical convergence study is performed for the compressible Baer-Nunziato model (5.45) in two space dimensions. We rely on the test problem proposed in [99], which has also been used in [107]. This test case is similar to the one described in [144] and [16]. The exact solution of this smooth unsteady test problem is obtained in two steps: first, an exact *stationary* and rotationally symmetric solution of the governing PDE is sought and then the problem is made *unsteady* by superimposing a constant, uniform velocity field $\bar{\boldsymbol{v}}$ using the principle of Galilean invariance of Newtonian mechanics. The exact solution is then simply given by the advection of the nontrivial initial condition with the superimposed constant velocity field $\bar{\boldsymbol{v}}$. The rotationally symmetric solution is found by writing the governing equations (5.45) in polar coordinates $(r-\beta)$ and by imposing angular symmetry $\partial/\partial\beta = 0$. What remains is an ODE system in the radial coordinate r that can be solved analytically, see [99].
In the following we denote with u_k^β the angular velocities and with u_k^r the radial velocities. Since we are interested in a vortex–type solution, we furthermore

suppose that $u_k^r = 0$. From the radial momentum equations we then obtain the following ODE system:

$$\begin{aligned} \frac{\partial}{\partial r}(\phi_1 p_1) &= p_2 \frac{\partial}{\partial r}\phi_1 + \frac{1}{r}\left(u_1^\beta\right)^2 \phi_1 \rho_1, \\ \frac{\partial}{\partial r}(\phi_2 p_2) &= p_2 \frac{\partial}{\partial r}\phi_2 + \frac{1}{r}\left(u_2^\beta\right)^2 \phi_2 \rho_2. \end{aligned} \tag{5.48}$$

If ϕ_1, p_1 and p_2 are known, e.g. by simply *prescribing them*, then (5.48) is just a simple algebraic equation system for the angular velocities u_k^β. As in [99] we choose

$$p_k = p_{k0}\left(1 - \frac{1}{4}e^{\left(1 - r^2/s_k^2\right)}\right), \qquad (k = 1, 2)\,, \tag{5.49}$$

and

$$\phi_1 = \frac{1}{3} + \frac{1}{2\sqrt{2\pi}}e^{-r^2/2}, \tag{5.50}$$

hence the angular velocities of each phase result as

$$\begin{aligned} u_1^\beta &= \frac{1}{2s_1 D}\sqrt{rD\left[p_{10}\left(4\sqrt{2\pi}F_1 + 6H_1 - 12Gs_1^2 + 3H_1 s_1^2\right) + 3p_{20}s_1^2\left(4G - H_2\right)\right]}\,, \\ u_2^\beta &= \frac{r\sqrt{2}}{2\rho_2 s_2}\sqrt{\rho_2 p_{20} F_2}\,, \end{aligned} \tag{5.51}$$

with the auxiliary variables

$$H_k = e^{-\dfrac{2r^2 + r^2 s_k^2 - 2s_k^2}{2s_k^2}}, \qquad F_k = e^{-\dfrac{(r - s_k)(r + s_k)}{s_k^2}}, \qquad (k = 1, 2),$$

and

$$G = e^{-r^2/2}, \qquad D = \rho_1\left(2\sqrt{2\pi} + 3G\right).$$

Note that the vector velocity field is rotating about the center of the vortex. To this steady, rotationally symmetric solution of the compressible Baer-Nunziato equations we now add a constant uniform velocity field $\bar{\mathbf{v}} = (\bar{u}, \bar{v})$ to make the test problem unsteady, as already mentioned above. This can be done since Newtonian mechanics is Galilean invariant. With this manufactured analytical solution we can now calculate the convergence rates of the high order direct ALE one-step ADER-WENO finite volume schemes presented in Chapter 2. For the computational setup, we use the following parameters:

$$\gamma_1 = 1.4, \quad \gamma_2 = 1.35, \quad \pi_1 = \pi_2 = 0, \quad \bar{u} = \bar{v} = 2, \quad \mu = \lambda = 0,$$

$$\rho_1 = 1, \quad \rho_2 = 2, \quad p_{10} = 1, \quad p_{20} = \frac{3}{2}, \quad s_1 = \frac{3}{2}, \quad s_2 = \frac{7}{5}. \tag{5.52}$$

The computational domain is the square $\Omega = [-10; 10] \times [-10; 10]$ and is depicted in Figure 5.8. Periodic boundaries are imposed everywhere.

5.4.2 Riemann problems

The high order ALE ADER-WENO finite volume schemes proposed in this work are validated by applying them to 1D Riemann problems that are solved in a 2D and 3D geometry on unstructured triangular and tetrahedral meshes. The exact solution for these 1D Riemann problems can be found in [11,76,212]. From the above mentioned articles we have chosen a subset of four Riemann problems, whose initial conditions are listed in Table 5.3. Some of the test cases use the stiffened gas EOS, some of them consider just a mixture of two ideal gases.

Table 5.3: Initial states left (L) and right (R) for the Riemann problems solved in 2D and 3D with the Baer-Nunziato model. Values for γ_i, π_i and the final time t_e are also given.

	ρ_s	u_s	p_s	ρ_g	u_g	p_g	ϕ_s	t_e
RP1 [76]:	$\gamma_s = 1.4$,		$\pi_s = 0$,	$\gamma_g = 1.4$,		$\pi_g = 0$,	$\lambda = \mu = 0$	
L	1.0	0.0	1.0	0.5	0.0	1.0	0.4	0.10
R	2.0	0.0	2.0	1.5	0.0	2.0	0.8	
RP2 [76]:	$\gamma_s = 3.0$,		$\pi_s = 100$,	$\gamma_g = 1.4$,		$\pi_g = 0$,	$\lambda = \mu = 0$	
L	800.0	0.0	500.0	1.5	0.0	2.0	0.4	0.10
R	1000.0	0.0	600.0	1.0	0.0	1.0	0.3	
RP4 [212]:	$\gamma_s = 3.0$,		$\pi_s = 3400$,	$\gamma_g = 1.35$,		$\pi_g = 0$,	$\lambda = \mu = 0$	
L	1900.0	0.0	10.0	2.0	0.0	3.0	0.2	0.15
R	1950.0	0.0	1000.0	1.0	0.0	1.0	0.9	
RP5 [99]:	$\gamma_s = 1.4$,		$\pi_s = 0$,	$\gamma_g = 1.67$,		$\pi_g = 0$,	$\lambda = 10^3$,	$\mu = 10^2$
L	1.0	0.0	1.0	1.0	0.0	1.0	0.99	0.2
R	0.125	0.0	0.1	0.125	0.0	0.1	0.01	

The initial computational domain is given either by $\Omega(0) = [-0.5; 0.5] \times [-0.05; 0.05]$ if $d = 2$ or $\Omega(0) = [-0.5; 0.5] \times [-0.05; 0.05] \times [-0.05; 0.05]$ if

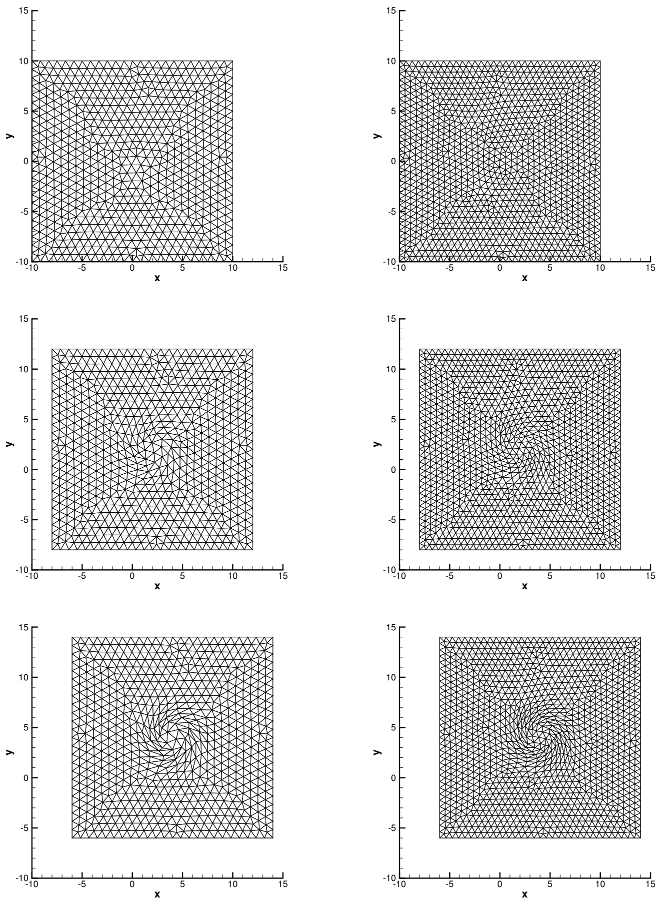


Figure 5.8: Moving Lagrangian meshes used for the numerical convergence test for the compressible Baer-Nunziato model in 2D at times $t = 0$ (top), $t = 1$ (center) and $t = 2$ (bottom) with resolution 24×24 (left) and 32×32 (right).

$d = 3$. The domain is discretized using a characteristic mesh size of $h = 1/200$, corresponding to an equivalent one-dimensional resolution of 200 cells. The initial discontinuity is located at $x = 0$ and the final simulation times are listed in Table 5.3. In x-direction we use transmissive boundaries, while periodic boundary conditions are imposed along the remaining directions.
Friction and pressure relaxation are neglected in the first three Riemann problems RP1, RP2 and RP4, while for RP5 we use a moderately stiff interphase drag $\lambda = 10^3$ and pressure relaxation $\mu = 10^2$. RP5 involves two almost pure ideal gases that differ in their value of γ. As done in [96,99] the exact solution for RP5 is computed using the exact Riemann solver for the Euler equations of compressible gas dynamics [229] with two different values of γ on the left and on the right of the contact discontinuity, respectively. In this test problem the algebraic source term in the full Baer-Nunziato system (5.45) becomes stiff, but it can be properly treated by the local space-time discontinuous Galerkin (DG) predictor presented in Section 2.3.2 of Chapter 2.

5.4.3 Explosion problems

We use the same initial condition given for the Riemann problems in Table 5.3 to solve the compressible Baer-Nunziato equations either on a circular or a spherical computational domain $\Omega(t)$ with initial radius $R = 1.0$ in 2D and $R = 0.9$ in 3D. In all cases the initial state $\boldsymbol{Q}(\mathbf{x}, 0)$ is assigned taking

$$\boldsymbol{Q}(\mathbf{x}, 0) = \left\{ \begin{array}{lll} \boldsymbol{Q}_i, & \text{if} & |\mathbf{x}| < r_c \\ \boldsymbol{Q}_o, & \text{else} & \end{array} \right. , \qquad (5.53)$$

with $r_c = 0.5$ representing the location of the initial discontinuity. The left state reported in Table 5.3 is assumed to be the inner state $\boldsymbol{Q}_i$, while the right state represents here the outer state $\boldsymbol{Q}_o$. In particular, the first explosion problem EP1 uses the initial condition of RP1, EP2 corresponds to RP2 and EP3 to RP4, respectively. In the fourth explosion problem EP4 we use again the initial values of RP2 and we set $\lambda = 10^5$ and $\mu = 0$, hence adopting a stiff interphase drag. The reference solution is obtained by solving an equivalent non-conservative one-dimensional PDE in radial direction with geometric reaction source terms, see [229] for the Euler equations and [230] for the Baer-Nunziato model for details. In our case here the reference solution has been obtained by using a path-conservative second order TVD scheme [107] on a very fine (fixed) 1D mesh consisting of 10,000 cells.

5.4.4 Two-Dimensional Riemann Problems

In [158] Kurganov and Tadmor have collected a very nice set of numerical solutions for two–dimensional Riemann problems of the compressible Euler equations [248]. Here, we propose two 2D Riemann problems for the compressible Baer–Nunziato model, however, without following the guidelines laid out in [158, 248], which lead to exactly one elementary wave at each interface, but we just simply take as initial data some of the data used for the 1D Riemann problems before, see Table 5.3. The initial computational domain is the square $\Omega(0) = [-0.5; 0.5] \times [-0.5; 0.5]$ and reflective wall boundaries are applied everywhere. The initial condition is given by four piecewise constant states defined in each quadrant of the two–dimensional coordinate system:

$$\boldsymbol{Q}(x,0) = \begin{cases} \boldsymbol{Q}_1 & \text{if} \quad x > 0 \wedge y > 0, \\ \boldsymbol{Q}_2 & \text{if} \quad x \leq 0 \wedge y > 0, \\ \boldsymbol{Q}_3 & \text{if} \quad x \leq 0 \wedge y \leq 0, \\ \boldsymbol{Q}_4 & \text{if} \quad x > 0 \wedge y \leq 0. \end{cases} \tag{5.54}$$

The initial conditions for the two configurations presented in this work are listed in Table 5.4.

Table 5.4: Initial conditions for the two–dimensional Riemann problems.

Configuration C1 ($\gamma_s = 1.4, \gamma_g = 1.67, \pi_s = \pi_g = 0,\ \lambda = 10^5,\ \mu = 10^2$)									
	ρ_s	u_s	v_s	p_s	ρ_g	u_g	v_g	p_g	ϕ_s
$\boldsymbol{Q}_1:\ (x > 0, y > 0)$	2.0	0.0	0.0	2.0	1.5	0.0	0.0	2.0	0.8
$\boldsymbol{Q}_2:\ (x < 0, y > 0)$	1.0	0.0	0.0	1.0	0.5	0.0	0.0	1.0	0.4
$\boldsymbol{Q}_3:\ (x < 0, y < 0)$	2.0	0.0	0.0	2.0	1.5	0.0	0.0	2.0	0.8
$\boldsymbol{Q}_4:\ (x > 0, y < 0)$	1.0	0.0	0.0	1.0	0.5	0.0	0.0	1.0	0.4
Configuration C2 ($\gamma_s = 3.0, \gamma_g = 1.4, \pi_s = 100, \pi_g = 0,\ \lambda = \mu = 0$)									
	ρ_s	u_s	v_s	p_s	ρ_g	u_g	v_g	p_g	ϕ_s
$\boldsymbol{Q}_1:\ (x > 0, y > 0)$	1000.	0.0	0.0	600.0	1.0	0.0	0.0	1.0	0.3
$\boldsymbol{Q}_2:\ (x < 0, y > 0)$	800.	0.0	0.0	500.0	1.5	0.0	0.0	2.0	0.4
$\boldsymbol{Q}_3:\ (x < 0, y < 0)$	1000.	0.0	0.0	600.0	1.0	0.0	0.0	1.0	0.3
$\boldsymbol{Q}_4:\ (x > 0, y < 0)$	800.	0.0	0.0	500.0	1.5	0.0	0.0	2.0	0.4

6 Numerical Results

In this chapter we show the numerical results collected with the high order direct ALE ADER finite volume schemes presented in this work. We will consider *all* the hyperbolic systems listed in Chapter 5 and *all* the associated numerical test problems described there.
In order to avoid any confusion or misunderstanding, we rigorously illustrate the numerical results for each test case following the *same* order given in Chapter 5. Furthermore each section is divided into subsections, which allows us to introduce the results contained therein. For the numerical scheme that has been used in each test problem we adopt the following abbreviations:

- ALE ADER-WENO refers to the original numerical method described in Chapter 2 and presented in [39, 41, 96] for $d = 2$ and in [40] for $d = 3$. Therefore this algorithm adopts the WENO reconstruction technique detailed in Section 2.2, the local Galerkin predictor illustrated in Section 2.3 and the finite volume scheme described in Section 2.5;

- MDRS stands for Multi-Dimensional Riemann solvers and it will be used to address the algorithm proposed in Section 4.2 of Chapter 4 (see [38] for further details);

- QF denotes the use of the quadrature-free flux computation that has been introduced in Section 4.3 of Chapter 4. A full presentation of the algorithm can be found in [44];

- ALE ADER-MOOD means that the WENO reconstruction procedure has been replaced by the MOOD paradigm, according to Section 4.4 of Chapter 4 and [45];

- LTS is the notation adopted for the Local Time Stepping numerical scheme [43] highlighted in Section 4.1 of Chapter 4. On the contrary, GTS represents the Global Time Stepping method which is employed, by default, by all the other versions listed so far. To ease notation, such abbreviation will be used only when an explicit comparison against the LTS algorithm is intended.

We will present only some samples among all the numerical results obtained with our algorithm, aiming at avoiding tedious repetitions of figures and tables which look very similar. For further results we refer the reader to the abovementioned references.
Since the aim of this work is to design almost Lagrangian-like algorithms with our direct ALE formulation, for each of the test cases we choose the local mesh velocity as the local fluid velocity, hence

$$\boldsymbol{V} = \boldsymbol{v}. \tag{6.1}$$

Furthermore, for each simulation, we explicitly write the numerical flux that has been adopted as well as the order of accuracy of the scheme and the mesh size. The node solver technique is addressed according to the notation given in Chapter 3, hence using $\mathcal{NS}_{cs}$ (node solver of Cheng and Shu, see Section 3.1.1), $\mathcal{NS}_{m}$ (node solver of Maire, see Section 3.1.2) or $\mathcal{NS}_{b}$ (node solver of Balsara, see Section 3.1.3).

6.1 The Euler equations of compressible gas dynamics

6.1.1 Numerical convergence studies

In the following we present numerical convergence studies as well as efficiency comparisons among the different versions of our direct ALE ADER schemes illustrated in Chapter 4.

Multidimensional convergence studies for ALE ADER-WENO schemes.

Tables 6.1 and 6.2 show the numerical convergence studies carried out for ALE ADER-WENO schemes in two and three space dimensions, respectively. We consider the test problem described in Section 5.1.1 and the Osher-type (2.71) numerical flux has been used in all computations. The convergence behavior of the sixth order scheme in 3D is not optimal for the last refinement, but the errors are nevertheless significantly lower than the ones of the fifth order method. A similar behavior was also observed for the Eulerian case, see [95].

2D convergence studies for LTS ALE ADER-WENO schemes.

Since the Local Time Stepping (LTS) algorithm, which has been described in Section 4.1, does involve a non-trivial procedure to evolve the numerical solution in time, we require the validity and the accuracy of this scheme to be assessed properly. Thus, Table 6.3 reports the numerical convergence studies for the two-dimensional LTS ALE ADER-WENO method. The designed order of accuracy is achieved very well using the Rusanov-type numerical flux (2.69) together with the node solver $\mathcal{NS}_{cs}$.

Comparison between ALE ADER-WENO and MDRS ALE ADER-WENO schemes.

Here we run the smooth vortex test problem (see Section 5.1.1) again using both, the original ALE ADER-WENO algorithm with the Rusanov-type (2.69) numerical flux and the genuinely multidimensional Riemann solver (MDRS), comparing in detail CPU time and accuracy. The behavior of the different solvers is depicted in Figure 6.1: the ALE ADER-WENO scheme with CFL $= 0.5$ is drawn by the black lines, while red and blue lines refer to the multidimensional HLL solver with CFL $= 0.5$ and CFL $= 1.0$, respectively. The error has been evaluated in L_1 norm according to (5.8) and the CPU time has been measured as the accumulated time obtained running the simulation in

Table 6.1: Numerical convergence results for the two-dimensional compressible Euler equations using the first up to sixth order ALE ADER-WENO finite volume schemes with the Osher-type numerical flux (2.71) and the node solver $\mathcal{NS}_{cs}$. The error norms refer to the variable ρ (density) at time $t = 1.0$.

$h(\Omega(t_f))$	ϵ_{L_2}	$\mathcal{O}(L_2)$	$h(\Omega, t_f)$	ϵ_{L_2}	$\mathcal{O}(L_2)$
	$\mathcal{O}1$			$\mathcal{O}2$	
3.73E-01	9.525E-02	-	3.43E-01	1.716E-02	-
2.63E-01	6.907E-02	0.9	2.49E-01	1.109E-02	1.4
2.14E-01	5.700E-02	0.9	1.69E-01	5.766E-03	1.7
1.74E-01	4.752E-02	0.9	1.28E-01	3.027E-03	2.3
	$\mathcal{O}3$			$\mathcal{O}4$	
3.28E-01	1.614E-02	-	3.29E-01	4.717E-03	-
2.51E-01	6.943E-03	3.0	2.51E-01	1.822E-03	3.5
1.68E-01	2.290E-03	2.7	1.67E-01	4.379E-04	3.5
1.28E-01	9.274E-04	3.3	1.28E-01	1.313E-04	4.4
	$\mathcal{O}5$			$\mathcal{O}6$	
3.29E-01	4.946E-03	-	3.29E-01	2.051E-03	-
2.51E-01	1.465E-03	4.5	2.51E-01	5.803E-04	4.7
1.67E-01	2.594E-04	4.3	1.67E-01	8.317E-05	4.8
1.28E-01	6.966E-05	4.9	1.31E-01	1.994E-05	5.9

parallel on four Intel Core i7-2600 CPUs with a clock-speed of 3.40GHz. The multidimensional Riemann solver allows the scheme to be run with CFL condition of unity, hence representing clearly the most efficient algorithm in terms of computational efficiency (blue lines in the right panel of Figure 6.1), although the most accurate one on a given mesh remains the classical one-dimensional Riemann solver (black lines in the left panel of Figure 6.1). Numerical values of the mesh size h, the L1 error norm and the corresponding CPU time are reported in [38] for each of the simulations contained in Figure 6.1.

The multidimensional Riemann solver is intended to approximate the structure of the strongly interacting state that forms when multiple one-dimensional Riemann problems come together at a vertex. Predicting the entire structure of that state is a difficult enterprise, with the result that approximations, based on the structure of conservation laws, are inevitable. Even in the simpler case

Table 6.2: Numerical convergence results for the three-dimensional compressible Euler equations using the first up to sixth order ALE ADER-WENO finite volume schemes with the Osher-type numerical flux (2.71) and the node solver $\mathcal{NS}_{cs}$. The error norms refer to the variable ρ (density) at time $t = 1.0$.

$h(\Omega(t_f))$	ϵ_{L_2}	$\mathcal{O}(L_2)$	$h(\Omega, t_f)$	ϵ_{L_2}	$\mathcal{O}(L_2)$
	$\mathcal{O}1$			$\mathcal{O}2$	
3.43E-01	1.081E-01	-	2.89E-01	2.214E-02	-
2.85E-01	9.159E-02	0.9	2.16E-01	1.202E-02	2.1
2.09E-01	6.875E-02	0.9	1.52E-01	5.865E-03	2.0
1.47E-01	4.899E-02	1.0	1.13E-01	3.254E-03	2.0
	$\mathcal{O}3$			$\mathcal{O}4$	
2.89E-01	1.718E-02	-	2.89E-01	4.116E-03	-
2.17E-01	7.641E-03	2.8	2.17E-01	1.369E-03	3.8
1.52E-01	2.601E-03	3.1	1.52E-01	3.273E-04	4.1
1.13E-01	1.049E-03	3.1	1.13E-01	9.802E-05	4.1
	$\mathcal{O}5$			$\mathcal{O}6$	
2.89E-01	2.272E-03	-	2.89E-01	1.015E-03	-
2.17E-01	6.605E-04	4.3	2.17E-01	2.312E-04	5.1
1.52E-01	1.234E-04	4.8	1.52E-01	3.090E-05	5.7
1.13E-01	2.932E-05	4.9	1.13E-01	6.576E-06	5.2

where the individual one-dimensional Riemann problems are weak, the wave model would approximate a Monge cone. Our wave model approaches the form of a Monge cone as the angular distribution of triangles around a vertex becomes more isotropic. In the Lagrangian framework the control volumes, i.e. the triangles which compose the computational grid, may become highly compressed and distorted, due to strong deformations that might occur during the simulation. Even the test case used for obtaining the convergence rates considers a vortex, whose motion stretches the triangles of the mesh yielding highly deformed elements with edges that may be either very short or very long. If the triangles of the primal grid are highly distorted and if the mesh experiences strong deviations from isotropy, the Voronoi polygons Ω_{HLL} may have some edges with very small edge length. Since the multidimensional wave model must contain all the 1D Riemann problems along the edges, the multidimen-

Table 6.3: Numerical convergence results for the compressible Euler equations using second to fourth order ALE ADER-WENO finite volume schemes with time accurate local time stepping (LTS). The error norms refer to the variable ρ (density) at time $t = 1.0$.

$h(\Omega(t_f))$	ϵ_{L_2}	$\mathcal{O}(L_2)$	$h(\Omega, t_f)$	ϵ_{L_2}	$\mathcal{O}(L_2)$
	$\mathcal{O}1$			$\mathcal{O}2$	
3.48E-01	1.921E-01	-	3.58E-01	5.286E-02	-
2.49E-01	1.524E-01	-	2.48E-01	3.558E-02	1.1
1.71E-01	1.112E-01	-	1.70E-01	1.514E-02	2.3
1.31E-01	8.691E-02	-	1.28E-01	8.193E-03	2.1
	$\mathcal{O}3$			$\mathcal{O}4$	
3.32E-01	3.471E-02	-	7.00E-01	6.419E-02	-
2.51E-01	1.789E-02	2.4	3.28E-01	1.030E-02	2.4
1.68E-01	6.346E-03	2.6	2.51E-01	3.598E-03	3.9
1.28E-01	2.935E-03	2.8	1.68E-01	7.706E-04	3.8

sional signal speeds $\boldsymbol{S}_j$ must be chosen large enough, which leads to additional numerical dissipation compared to a classical one-dimensional Riemann solver, which does not have this dependence on the geometry. Therefore the numerical errors of the one-dimensional Riemann solver are smaller than the values referring to the multidimensional scheme (see [38] for details). A possible solution could be to introduce more substructure into the strongly interacting region, and that is a topic of current research. However, in the overall comparison the ALE schemes based on multidimensional Riemann solvers are computationally still more efficient since they allow a larger time step.

Multidimensional convergence studies for QF ALE ADER-WENO schemes.

We perform the convergence studies for the quadrature-free (QF) approach, presented in Section 4.3, from first up to fourth order of accuracy on a series of successively refined meshes. The Rusanov-type numerical flux (2.69) with the node solver $\mathcal{NS}_m$ has been used in all computations and the results are reported in Tables 6.4 and 6.5 for unstructured triangular and tetrahedral meshes, respectively.

Table 6.4: Numerical convergence results for the equations of hydrodynamics using the *two-dimensional* quadrature-free (QF) ALE-ADER-WENO finite volume schemes presented in Section 4.3. The error norms refer to the variable ρ (density) at time $t = 1.0$ for first up to fourth order of accuracy.

$h(\Omega(t_f))$	ϵ_{L_2}	$\mathcal{O}(L_2)$	$h(\Omega, t_f)$	ϵ_{L_2}	$\mathcal{O}(L_2)$
	$\mathcal{O}1$			$\mathcal{O}2$	
3.47E-01	1.9526E-01	-	3.64E-01	5.5398E-02	-
2.50E-01	1.5567E-01	0.7	2.48E-01	3.5933E-02	1.1
1.71E-01	1.1334E-01	0.8	1.71E-01	1.5749E-02	2.2
1.33E-01	8.7917E-02	1.0	1.28E-01	8.6411E-03	2.1
	$\mathcal{O}3$			$\mathcal{O}4$	
3.31E-01	3.4896E-02	-	3.28E-01	1.0919E-02	-
2.51E-01	1.8049E-02	2.4	2.51E-01	3.7978E-03	3.9
1.68E-01	6.4878E-03	2.5	1.67E-01	7.5383E-04	4.0
1.28E-01	2.9128E-03	3.0	1.28E-01	2.4142E-04	4.2

Table 6.5: Numerical convergence results for the equations of hydrodynamics using the *three-dimensional* quadrature-free (QF) ALE-ADER-WENO finite volume schemes presented in Section 4.3. The error norms refer to the variable ρ (density) at time $t = 1.0$ for first up to fourth order of accuracy.

$h(\Omega(t_f))$	ϵ_{L_2}	$\mathcal{O}(L_2)$	$h(\Omega, t_f)$	ϵ_{L_2}	$\mathcal{O}(L_2)$
	$\mathcal{O}1$			$\mathcal{O}2$	
5.92E-01	5.9956E-01	-	5.93E-01	1.1771E-01	-
2.89E-01	3.4632E-01	0.8	2.89E-01	3.9396E-02	1.5
2.17E-01	2.7426E-01	0.8	2.16E-01	2.4336E-02	1.7
1.33E-01	2.0076E-01	0.9	1.52E-01	1.1719E-02	2.1
	$\mathcal{O}3$			$\mathcal{O}4$	
5.92E-01	1.5658E-01	-	5.92E-01	7.3869E-02	-
2.89E-01	3.7319E-02	2.0	2.89E-01	7.4072E-03	3.2
2.16E-01	1.7780E-02	2.6	2.16E-01	2.3191E-03	4.0
1.52E-01	6.2188E-03	3.0	1.52E-01	6.2111E-04	3.7

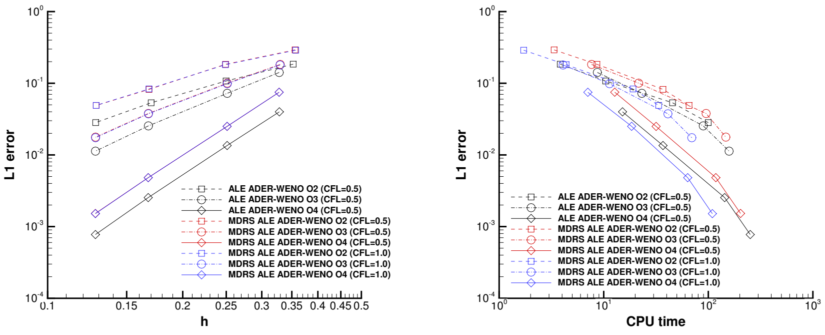


Figure 6.1: Comparison between ALE ADER-WENO and MDRS ALE ADER-WENO two-dimensional schemes from first up to fourth order of accuracy with different CFL number. The Rusanov-type (2.69) numerical flux and the node solver $\mathcal{NS}_b$ have been used. Left: dependency of the error norm on the mesh size. Right: dependency of the error norm on the CPU time.

Convergence rates and comparison between ALE ADER-WENO and ALE ADER-MOOD schemes.

A CPU time comparison between ALE ADER-WENO and ALE ADER-MOOD schemes for $d \in [2, 3]$ has been carried out using a single Intel Core i7-2600 CPU with a clock-speed of 3.40GHz, in order to assess the pure serial performance, without accounting for the MPI overhead. Figure 6.2 represents in the left panel the convergence rates for the L_2 norms (see Eqn. (5.7)) in log scale versus the expected fourth and fifth order lines, while in the right panel we also show the CPU time as a function of the initial element characteristics length h for ALE ADER-WENO and ALE ADER-MOOD schemes using $\mathbb{P}_3$ and $\mathbb{P}_4$ polynomial reconstructions. One can clearly notice that, for this problem, the fifth order accurate ALE ADER-MOOD scheme has approximately the same cost than the fourth order accurate ALE ADER-WENO scheme. We can also observe that MOOD costs about 2 times less than WENO in this configuration. The corresponding data can be found in [45].

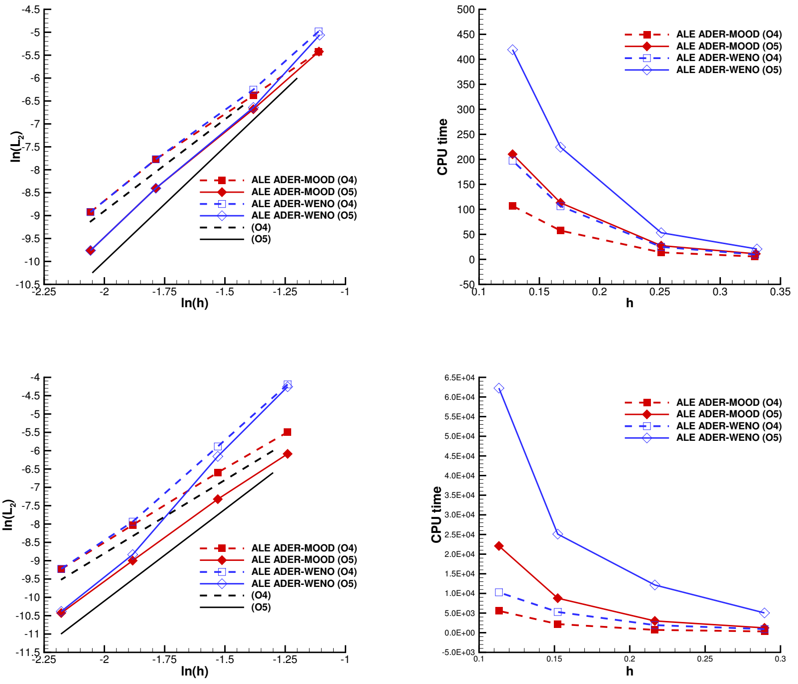


Figure 6.2: Comparison between ALE ADER-WENO and ALE ADER-MOOD schemes in 2D (top row) and in 3D (bottom row) for fourth and fifth order of accuracy using the Osher-type (2.71) numerical flux and the node solver $\mathcal{NS}_m$. Left: convergence rate for the L_2 norm in log scale versus expected 4th and 5th order lines. Right: CPU time [s] as a function of the final element characteristics size $h(\Omega(t_f))$.

6.1.2 The Sod shock tube problem

2D ALE ADER-WENO numerical flux comparison.

In order to study how the choice of the numerical flux does affect the solution, we consider the Sod shock tube problem presented in Section 5.1.2 of Chapter 5. The two-dimensional computational domain Ω is discretized with a total number of elements of $N_E = 18018$ of characteristic size $h = 1/200$. The numerical results plotted in Figure 6.3 have been collected with three different numerical fluxes, namely the Rusanov-type flux (2.69) and the Osher-type flux given by (2.71). Furthermore we also use an HLLC-type numerical flux, first introduced by Toro et al. [232] for the Eulerian case and then extended to moving meshes by Van der Vegt et al. [238, 239] in the framework of ALE space-time DG finite element schemes. For a detailed description of the HLLC flux and its extension to dynamic grid motion we refer to [238].

Figure 6.3 shows a comparison between the exact solution and the numerical results at time $t = 0.25$ obtained with a third order accurate direct ALE ADER-WENO finite volume scheme. We extend the final time of the simulation to $t_f = 0.25$ (instead of $t_f = 0.2$ as given in Section 5.1.2) in order to highlight the differences arising from the use of different numerical fluxes. The Rusanov flux is more diffusive if compared with the Osher and the HLLC fluxes, especially looking at the contact wave located at $x \approx 0.23$. The Rusanov-type flux visibly smooths the contact discontinuity even in the case of a moving mesh, while the contact wave is almost perfectly resolved by the Osher and the HLLC flux, which both preserve a sharp numerical solution with only one point inside the wave.

Numerical convergence rates for the three different numerical fluxes are reported in Table 6.6. As test problem we use again the isentropic vortex fully described previously in section 5.1.1, but here we consider only a third order scheme as example. The lowest L_2 errors are achieved by the Osher-type flux and the HLLC flux. The Rusanov flux, being the most dissipative one, yields the highest error, as expected.

Looking at the computational times reported in Table 6.7, the Rusanov flux leads to the fastest scheme, but also to the least accurate one, while the Osher-type flux gives the lowest errors. The CPU effort for the HLLC flux and the Osher-type flux are comparable. In Table 6.7 η denotes the CPU time normalized by the one needed by the Rusanov flux.

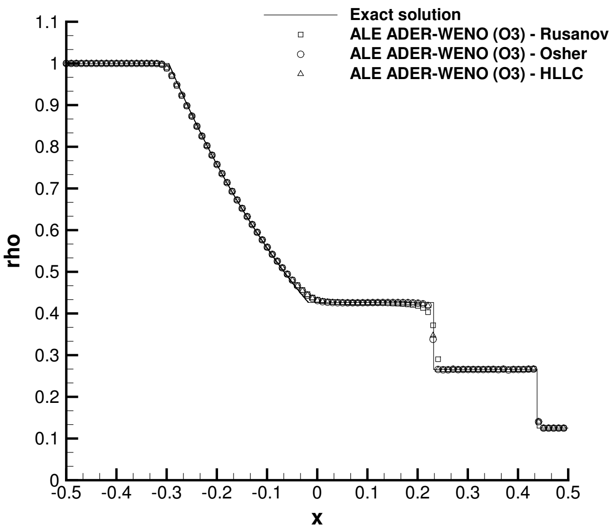


Figure 6.3: Sod shock tube test for $d = 2$. Third order accurate numerical results obtained using three different numerical fluxes (Rusanov, Osher and HLLC) and comparison with the exact solution (solid line) at time $t = 0.25$. The node solver $\mathcal{NS}_{cs}$ has been employed.

3D ALE ADER-WENO results.

Figure 6.4 displays the results for the Sod problem obtained with a third order accurate ALE ADER-WENO scheme. We use a characteristic mesh size of $h = 1/100$ with a total number of tetrahedra $N_E = 70453$. The contact wave has been resolved very well with only one intermediate point and overall a very good agreement with the exact solution is achieved for density, as well as for pressure and for the horizontal velocity component u.

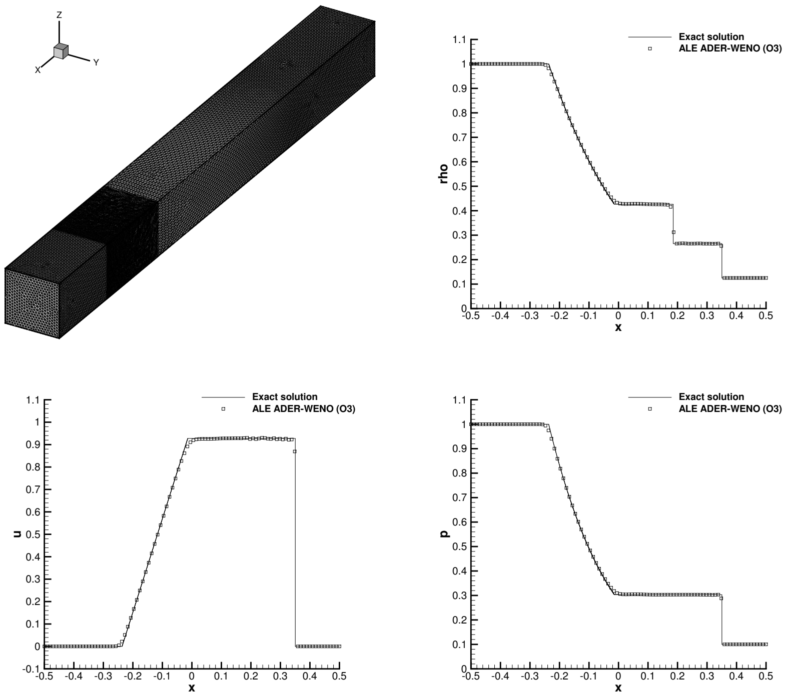


Figure 6.4: Final 3D mesh configuration together with a 1D cut along the x-axis through the third order numerical results and comparison with exact solution for the three-dimensional Sod shock tube problem at time $t = 0.2$. We use the Osher-type (2.71) numerical flux with the node solver $\mathcal{NS}_m$.

Table 6.6: Comparison of numerical convergence results for the compressible Euler equations using the third order version of the two-dimensional ALE ADER-WENO finite volume schemes and three different types of numerical fluxes (Rusanov, Osher and HLLC). The error norms refer to the variable ρ (density) at time $t = 1.0$.

Rusanov		Osher		HLLC	
ϵ_{L_2}	$\mathcal{O}(L_2)$	ϵ_{L_2}	$\mathcal{O}(L_2)$	ϵ_{L_2}	$\mathcal{O}(L_2)$
1.076E-01	-	1.614E-02	-	1.818E-02	-
2.315E-02	4.2	6.943E-03	3.2	7.897E-03	3.0
8.658E-03	2.4	2.290E-03	2.7	2.621E-03	2.7
3.950E-03	2.9	9.274E-04	3.3	1.068E-03	3.3

Table 6.7: Computational time for the convergence studies and the Sod shock tube results obtained with three different numerical fluxes (Rusanov, Osher and HLLC). CPU Time is measured in seconds $[s]$ on one Intel Core i7-2600 CPU with a clock-speed of 3.40GHz and η denotes the ratio w.r.t. the computational time of the Rusanov flux.

	Rusanov	Osher		HLLC	
	CPU time	CPU time	η	CPU time	η
	6.21E+00	9.11E+00	1.5	1.06E+01	1.7
Isentropic	2.22E+01	2.49E+01	1.1	2.77E+01	1.2
vortex	9.64E+01	1.02E+02	1.1	1.12E+02	1.1
	1.55E+02	1.69E+02	1.1	1.93E+02	1.2
Sod problem	1.22E+04	1.44E+04	1.2	1.59E+04	1.3

Riemann problems using the LTS ALE ADER-WENO algorithm.

Here we solve two classical Riemann problems, namely the shock tube problems of Sod and of Lax, which are in the following addressed as RP1 and RP2, respectively, and which are widely adopted to validate numerical algorithms for the solution of the compressible Euler equations. They both include the formation of a left-propagating rarefaction wave, an intermediate contact discontinu-

ity and a right-propagating shock wave. Though intrinsically one-dimensional, these tests become non-trivial and multidimensional when applied to unstructured meshes, where in general the element edges are not aligned with the fluid motion. Since a contact wave is present in the solution, we can also check how well it is resolved by the LTS ALE ADER-WENO scheme described in Section 4.1.
The initial computational domain is given by the box $\Omega(0) = [-0.5; 0.5] \times [-0.05; 0.05]$ that is discretized with a characteristic mesh size of $h = 1/200$, leading to a total number of $N_E = 8862$ elements, while the initial conditions are given in terms of the primitive variables $\boldsymbol{U} = (\rho, u, v, p)$. Table 6.8 reports the relevant data for the setup of the two tests, where t_f represents the final time of the simulation while x_d gives the position of the initial discontinuity which splits the computational domain, as well as the initial conditions, in the two left and right states $\boldsymbol{U}_L$ and $\boldsymbol{U}_R$. For both Riemann problems we set periodic boundary conditions in the y direction, while transmissive boundaries are imposed along the x direction.

Table 6.8: Initial condition for the Sod (RP1) and the Lax (RP2) shock tube problem. t_f is the final time of the simulation and x_d denotes the position of the initial discontinuity.

Case	ρ_L	u_L	v_L	p_L	ρ_R	u_R	v_R	p_R	t_f	x_d
RP1	1.0	0.0	0.0	1.0	0.125	0.0	0.0	0.1	0.2	0.0
RP2	0.445	0.698	0.0	3.528	0.5	0.0	0.0	0.571	0.1	0.0

We use the third order version of our LTS ALE ADER-WENO schemes with the Osher-type numerical flux (2.71) to obtain the results depicted in Figures 6.5-6.6, where a comparison between the exact and the numerical solution is shown. We observe an excellent resolution of the contact wave with only one intermediate point for both RP1 and RP2, and a very good agreement with the analytical solution can also be noticed for density, as well as for pressure and for the horizontal velocity component.

Multidimensional ALE ADER-MOOD results.

Figure 6.7 shows two- and three-dimensional numerical results for the Sod problem using the fifth order accurate ALE ADER-MOOD algorithm. The meshes are constituted of $N_E = 8862$ triangles for $d = 2$ and $N_E = 70453$

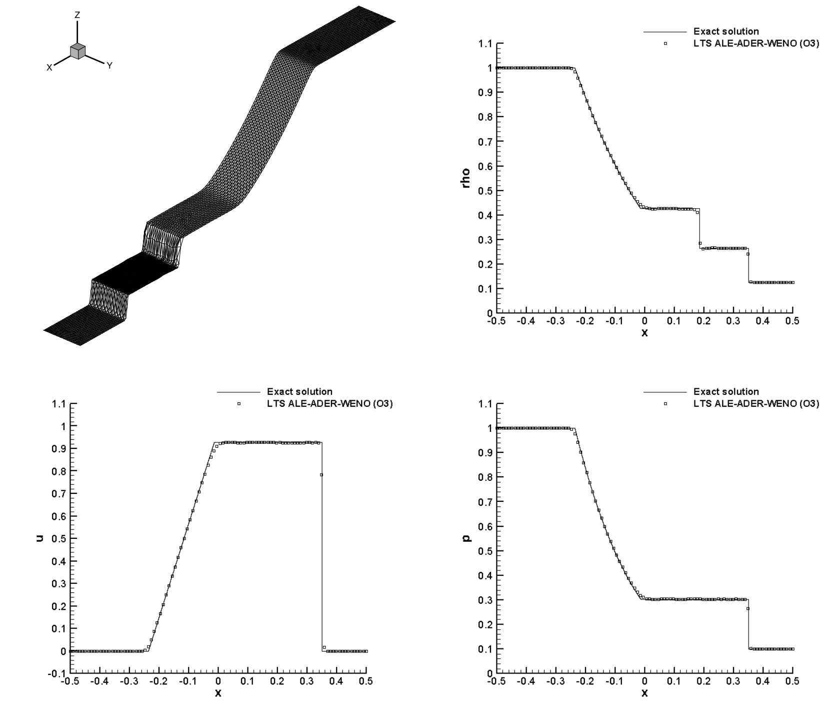


Figure 6.5: Comparison between exact and third order accurate numerical solution for the Sod shock tube problem RP1. Density (top right), velocity (bottom left) and pressure (bottom right) distribution are shown as well as a 3D view of the density solution at the final time $t_f = 0.2$ (top left).

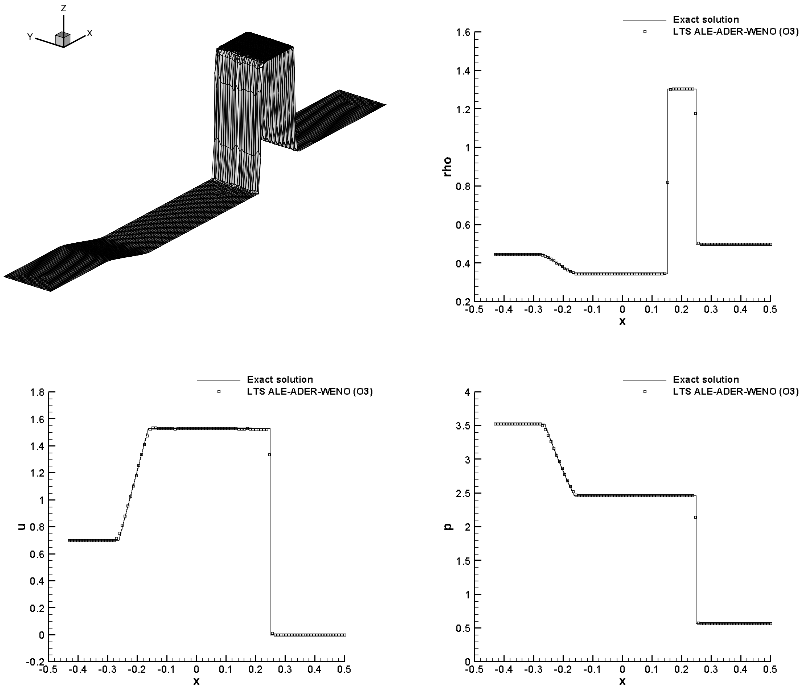


Figure 6.6: Comparison between exact and third order accurate numerical solution for the Lax shock tube problem RP2. Density (top right), velocity (bottom left) and pressure (bottom right) distribution are shown as well as a 3D view of the density solution at the final time $t_f = 0.1$ (top left).

tetrahedral elements for $d = 3$. One can notice that the MOOD version of our scheme is able to capture the expected solution. In order to visually estimate how the MOOD loop affects the simulation, we have plotted in Figure 6.7 on the left column the percentage of cells updated with $\mathbb{P}_1^{\text{LIM}}$ (red) or $\mathbb{P}_0$ (blue) polynomial reconstructions as a function of the iteration number. Moreover a cubic fit of these sample points has been computed to emphasize the general behavior of the scheme. As expected, only very few elements are updated with the low order schemes, namely less than 7% in 2D and 1% in 3D. Therefore we can deduce that the computational efficiency of the whole algorithm is almost no affected by the MOOD loop.

6.1.3 Multidimensional explosion problem

Multidimensional ALE ADER-WENO results.

Figure 6.8 shows the comparison between the reference solution and the numerical solution obtained with the third order version of the two-dimensional direct ALE ADER-WENO schemes for density and velocity at time $t_f = 0.20$: a circular shock wave is traveling away from the center together with a contact wave, while a circular rarefaction wave is running towards the origin. The contact wave is very well resolved due to the use of the little diffusive Osher-type flux (2.71). The mesh contains a total number of $N_E = 68,324$ triangles of characteristic mesh spacing of $h = 1/100$ in 2D. A 3D view of the numerical solution as well as a very coarse version of the mesh are depicted in the top row of Figure 6.8.
The three-dimensional results are displayed in Figure 6.9 and they have been obtained using the fourth order accurate version of the ALE ADER-WENO algorithm. The computational domain is discretized with a total number of elements $N_E = 7225720$ and the Osher-type numerical flux (2.71) has been used to run the simulation up to the final time $t_f = 0.25$. Even in 3D the contact discontinuity is very well resolved due to the Lagrangian approach. A slice of the entire mesh configuration at four different output times is depicted in Figure 6.10, where the progressively compression of the tetrahedra located at the shock frontier can be clearly identified.

Explosion problems using the LTS ALE ADER-WENO algorithm.

The two initial states needed for the setup of the two-dimensional explosion problem are addressed here as the *inner* state $\boldsymbol{U}_i$ and the *outer* state $\boldsymbol{U}_o$, respectively. They are given in Table 6.9 and have been taken from [229]. In

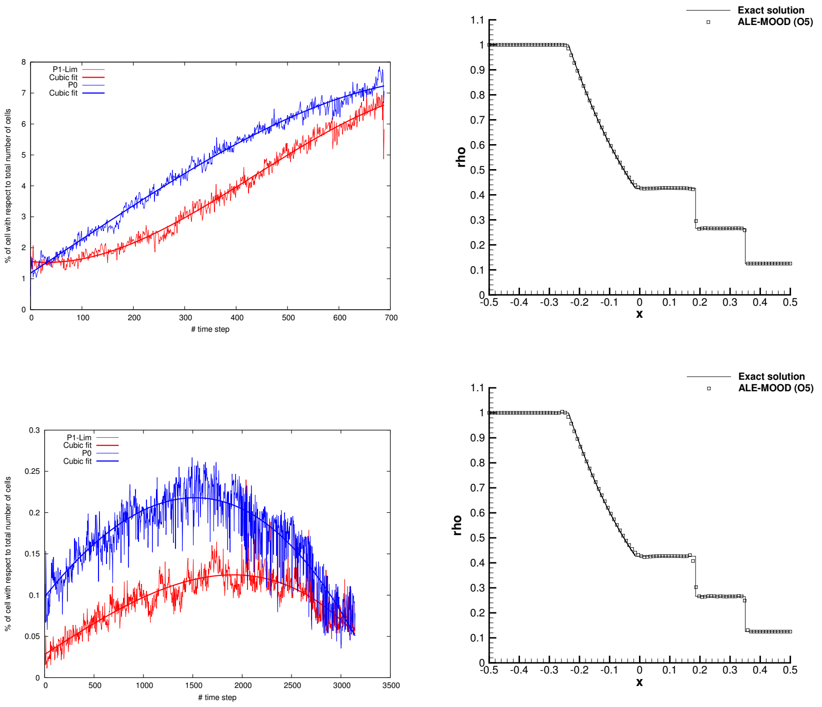


Figure 6.7: Numerical results for the Sod shock tube problem at the final time $t_f = 0.2$ in 2D (top row) and in 3D (bottom row) using the fifth order accurate ALE ADER-MOOD schemes with the Osher-type numerical flux and the node solver $\mathcal{NS}_m$. Left: percentage of cells updated with $\mathbb{P}_1^{\text{LIM}}$ (red) or $\mathbb{P}_0$ (blue) reconstruction as a function of the iteration number and associated cubic fits. Right: density distribution.

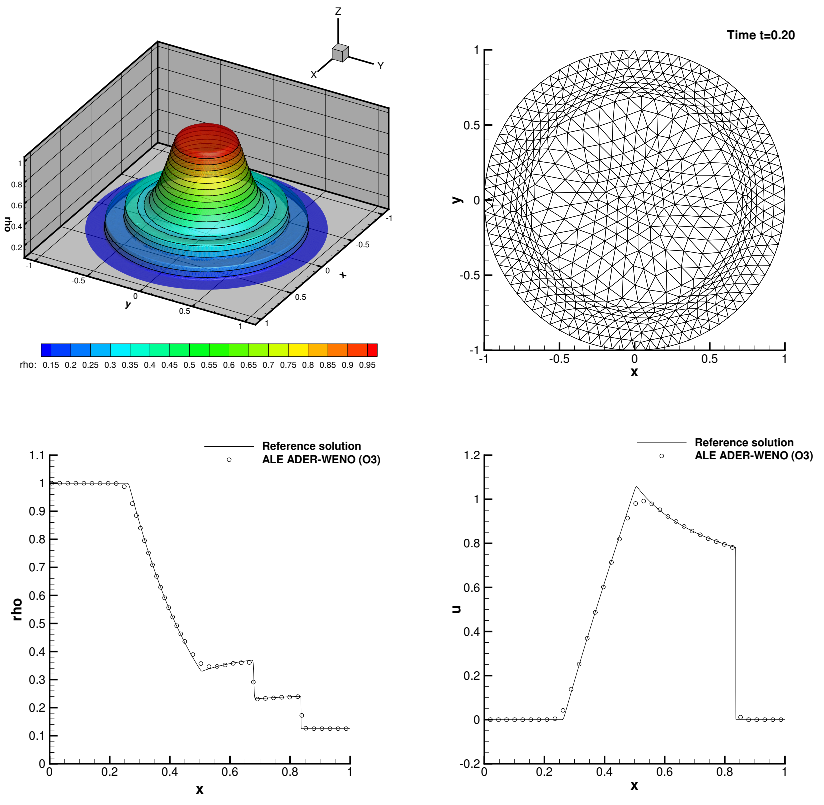


Figure 6.8: Third order numerical results for the *two-dimensional* explosion problem at the final time $t_f = 0.2$. The Osher-type numerical flux has been used as well as the node solver $\mathcal{NS}_{cs}$.

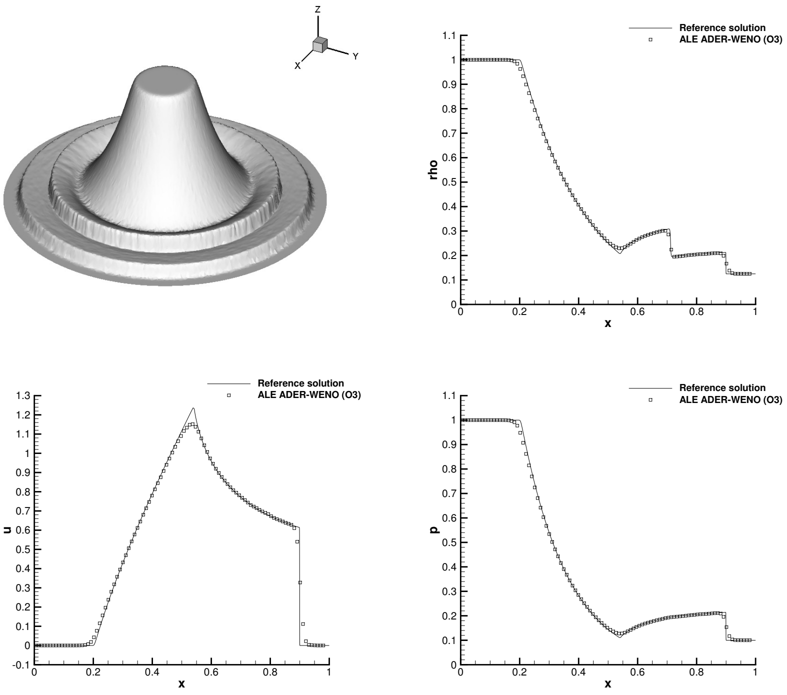


Figure 6.9: Fourth order numerical results and comparison with the reference solution for the three-dimensional explosion problem at time $t_f = 0.25$. The Osher-type numerical flux has been used as well as the node solver $\mathcal{NS}_m$.

both cases we use the same computational mesh with a characteristic mesh size of $h = 1/100$ for $r \leq R$ and $h = 1/50$ for $r > R$, hence obtaining a total number $N_E = 43756$ of triangles. $R = 0.5$ is the initial position of the discontinuity which separates the states $\boldsymbol{U}_i$ and $\boldsymbol{U}_o$.

Third order accurate LTS ALE ADER-WENO schemes have been used together with the Osher-type numerical flux (2.71) to compute the explosion problems

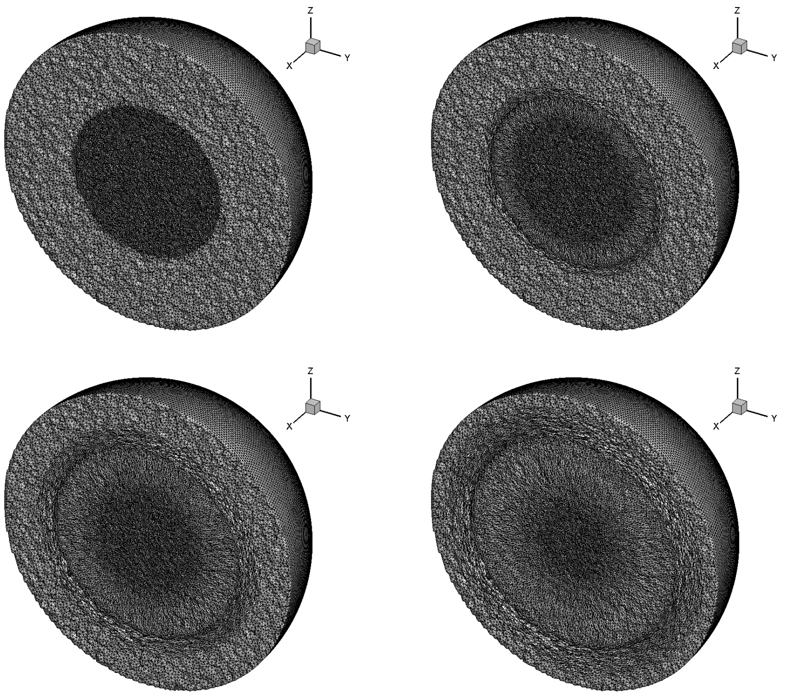


Figure 6.10: Mesh configuration for the three-dimensional explosion problem at times $t = 0.00$, $t = 0.08$, $t = 0.16$ and $t = 0.25$.

Table 6.9: Initial conditions for the two-dimensional explosion problems EP1 and the EP2 with t_f denoting the final time of the simulation.The LTS ALE ADER-WENO algorithm is used together with the node solver $\mathcal{NS}_{cs}$.

Case	ρ_i	u_i	v_i	p_i	ρ_o	u_o	v_o	p_o	t_f
EP1	1.0	0.0	0.0	1.0	0.125	0.0	0.0	0.1	0.2
EP2	1.0	0.0	0.0	1000.0	1.0	0.0	0.0	0.01	0.012

EP1 and EP2. Figures 6.11-6.12 show a comparison between the numerical solution obtained with the Lagrangian LTS scheme and the 1D reference solution, computed as explained in 5.1.3. One can appreciate the very good resolution of the contact wave in the density distribution and a good agreement with the reference solution is achieved also for horizontal velocity and pressure. We point out that EP2 is more challenging than EP1 because it involves a strong shock wave which causes a high compression of some elements in the mesh, as clearly depicted in Figure 6.13. By using the LTS approach we can avoid that those small triangles dictate the timestep for the entire mesh, hence allowing the other control volumes to reach the end of the simulation much faster and with a lower number of element updates, as highlighted in Table 6.14 of Section 6.1.13.

6.1.4 The Kidder problem

We present numerical results for the Kidder problem described in the previous chapter in Section 5.1.4.

2D numerical results.

The computational domain is discretized with a characteristic mesh size of $h = 1/100$ for a total number of $N_E = 3180$. The Osher-type (2.71) numerical flux is used in all computations. Figure 6.14 displays the numerical results obtained with the MDRS version of the direct ALE ADER-WENO algorithm, where a CFL number of CFL $= 0.95$ was set. The evolution of the density distribution has been plotted as well as the time-dependent location of the internal and the external frontier, which has been compared against the exact solution of Kidder given in Section 5.1.4. The node solver $\mathcal{NS}_b$ has been employed.
The two-dimensional Kidder problem has been run also using the original formulation of our algorithm (see Chapter 2) with each of the node solvers described in Section 3.1 of Chapter 3. We have computed the absolute error $|err|$, reported in Table 6.10 and defined as the difference between the analytical and the numerical location of the internal and external radius at the final time.

3D numerical results.

A total number of $N_E = 111534$ tetrahedra has been used to discretize the computational domain. We use the fourth order version of our ALE ADER-WENO scheme together with the Osher-type flux (2.71) and the node solver

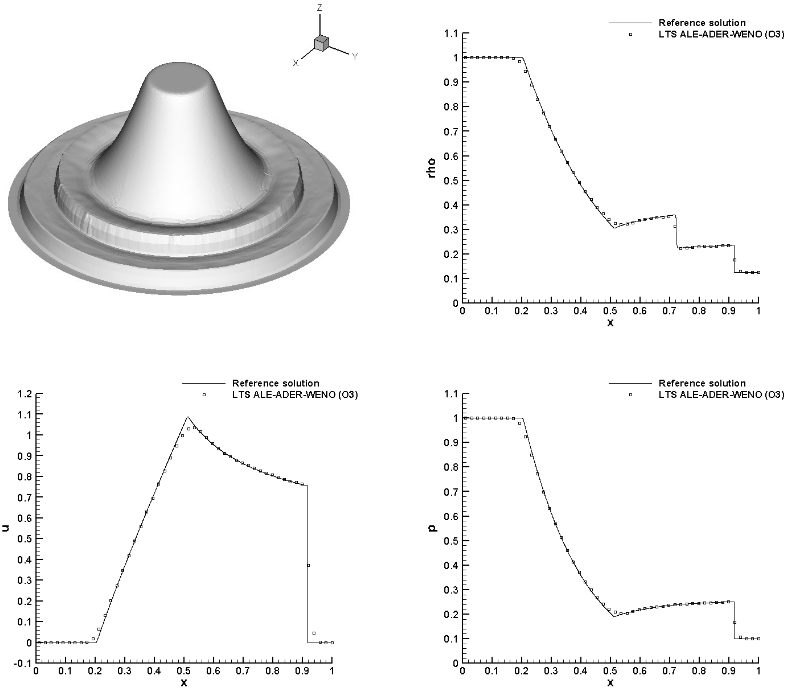


Figure 6.11: Comparison between reference and third order accurate numerical solution for the explosion problem EP1 obtained with the LTS ALE ADER-WENO algorithm. Density (top right), velocity (bottom left) and pressure (bottom right) distribution are shown as well as a 3D view of the density solution at the final time $t_f = 0.25$ (top left).

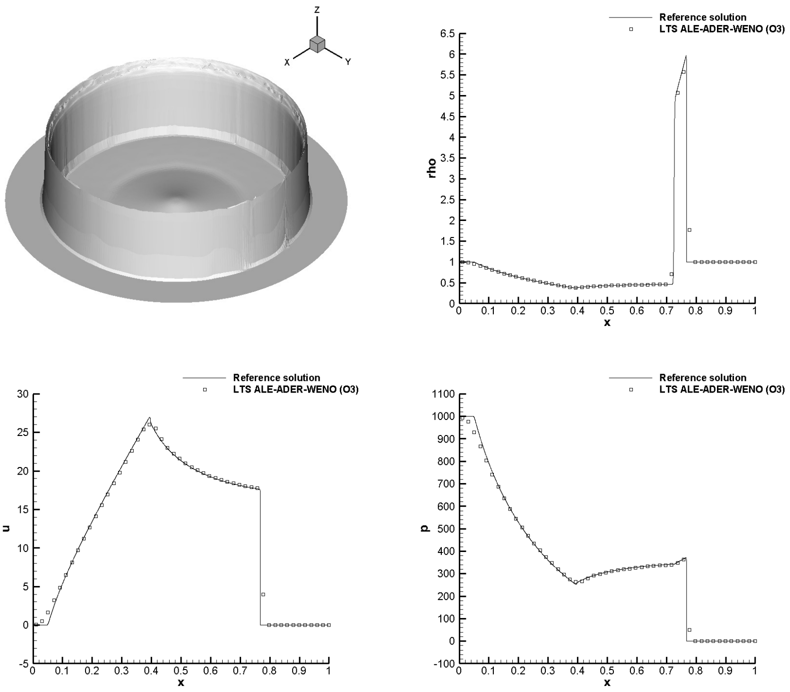


Figure 6.12: Comparison between reference and third order accurate numerical solution for the explosion problem EP2 obtained with the LTS ALE ADER-WENO algorithm. Density (top right), velocity (bottom left) and pressure (bottom right) distribution are shown as well as a 3D view of the density solution at the final time $t_f = 0.012$ (top left).

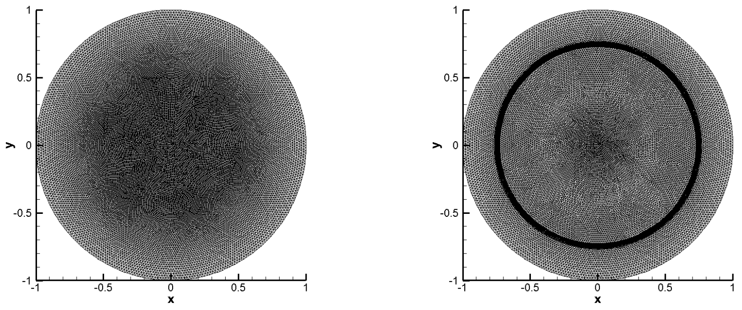


Figure 6.13: Initial (left) and final (right) mesh configuration for the explosion problem EP2 run with the LTS ALE ADER-WENO algorithm. The strong shock generates a high compression of those elements which follow the wave.

Table 6.10: Absolute error for the internal and external radius location between exact and numerical solution for the three different node solvers ($\mathcal{NS}_{cs}$, $\mathcal{NS}_m$ and $\mathcal{NS}_b$), obtained with the original direct ALE ADER-WENO algorithm. The numerical value has been evaluated as an average of the position of all the nodes lying on the internal and on the external frontier.

	$\mathcal{NS}_{cs}$	$\mathcal{NS}_m$	$\mathcal{NS}_b$
$\lvert err_{int}\rvert$	7.72443E-06	7.72460E-06	7.73181E-06
$\lvert err_{ext}\rvert$	1.01812E-05	1.01811E-05	1.01867E-05

$\mathcal{NS}_m$. Figure 6.15 shows the initial and the final density distribution of the shell as well as the evolution of the internal and external frontier location during the simulation.

We also did the simulation adopting the quadrature-free (QF) approach presented in Section 4.3 and Table 6.11 reports the absolute error $|err|$ associated

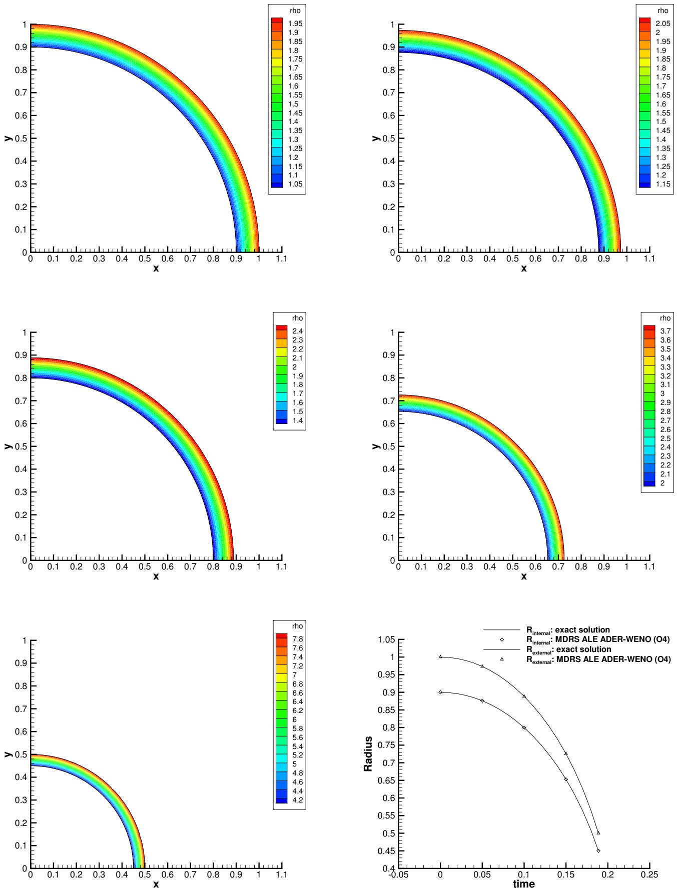


Figure 6.14: Density distribution for the Kidder problem at output times $t = 0.00$, $t = 0.05$, $t = 0.10$, $t = 0.15$ and $t = t_f$ (from top left to bottom left). Evolution of the internal and external radius of the shell and comparison between analytical and numerical solution (bottom right)for the MDRS ALE ADER-WENO scheme.

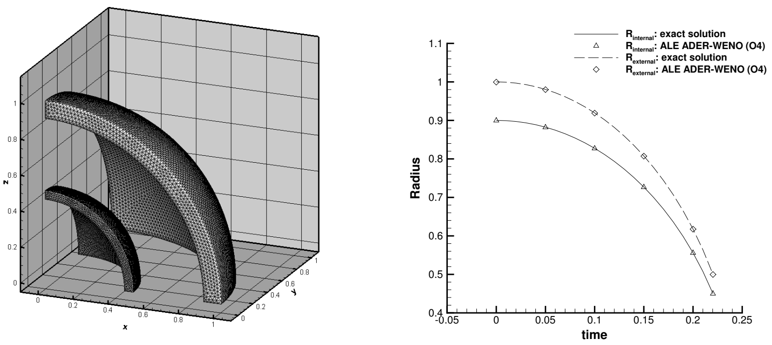


Figure 6.15: Left: position and mesh configuration of the shell at times $t = 0$ and at $t = t_f$. Right: Evolution of the internal and external radius of the shell and comparison between analytical and numerical solution for the three-dimensional ALE ADER-WENO scheme.

to the internal and external frontier evolution. A comparison is done between the quadrature-free algorithm and the classical Gaussian-quadrature (GQ) formulation.

	QF ALE-ADER-WENO		ALE-ADER-WENO	
r_{ex}	r_{num}	$\|err\|_{QF}$	r_{num}	$\|err\|_{GQ}$
0.450000	0.450347	3.47E-04	0.449749	2.51E-04
0.500000	0.499412	5.88E-04	0.499720	2.80E-04

Table 6.11: Absolute error for the internal and external radius location between exact (r_{ex}) and numerical (r_{num}) solution computed with (QF) and without (GQ) the quadrature-free formulation.

6.1.5 The Saltzman problem

In the following we display some numerical results for the challenging Saltzman test case introduced previously in in Section 5.1.4. We give separately the results for $d = 2$ and $d = 3$. For all simulations the robust Rusanov-type (2.69) has been used.

2D numerical results.

Figure 6.16 shows the evolution of the density solution obtained using the third order accurate quadrature-free (QF) version of the algorithm, while Figure 6.17 plots a comparison between analytical and numerical solution for density and velocity which has been computed with the LTS and GTS ALE ADER-WENO scheme with third order of accuracy. The decrease of density near the piston, which affects all computations, is due to the well known *wall-heating problem*, see [228].
Furthermore the Saltzman problem has been run using the different node solvers $\mathcal{NS}_{cs}$, $\mathcal{NS}_m$ and $\mathcal{NS}_b$ within the original numerical method presented in Chapter 2, i.e. direct ALE ADER-WENO schemes. Adopting the node solver $\mathcal{NS}_b$ we were able to run the simulation with a time step size that was 10% larger with respect to the other node solvers. Moreover, with the node solver $\mathcal{NS}_b$ it was possible to run the simulation until a final time of $t = 0.74$, which was not possible with the other node solvers $\mathcal{NS}_m$ and $\mathcal{NS}_{cs}$, that required smaller time steps and reached only $t_f = 0.69$.

3D numerical results.

The three-dimensional results for the Saltzman problem have been obtained with the third order version of the ALE ADER-WENO algorithm using the node solver $\mathcal{NS}_m$. Density and velocity distribution are displayed in Figure 6.18. A good agreement with the exact solution can be noticed regarding both density and velocity distribution at the final time $t_f = 0.6$. The positivity preserving technique described in Appendix B has been used to smear out some unphysical oscillations occurring at the shock.
A fifth order accurate simulation has been run with the ALE ADER-MOOD scheme and the results are depicted in Figure 6.19, where we present the final grid configuration and cell orders (top line). The original skewness of the mesh has not generated any spurious oscillations or lack of symmetry and the mesh is nicely shaped. Moreover the cell orders are almost at maximal value and few cells are decremented, meaning that the $\mathbb{P}_4$ reconstruction is used almost

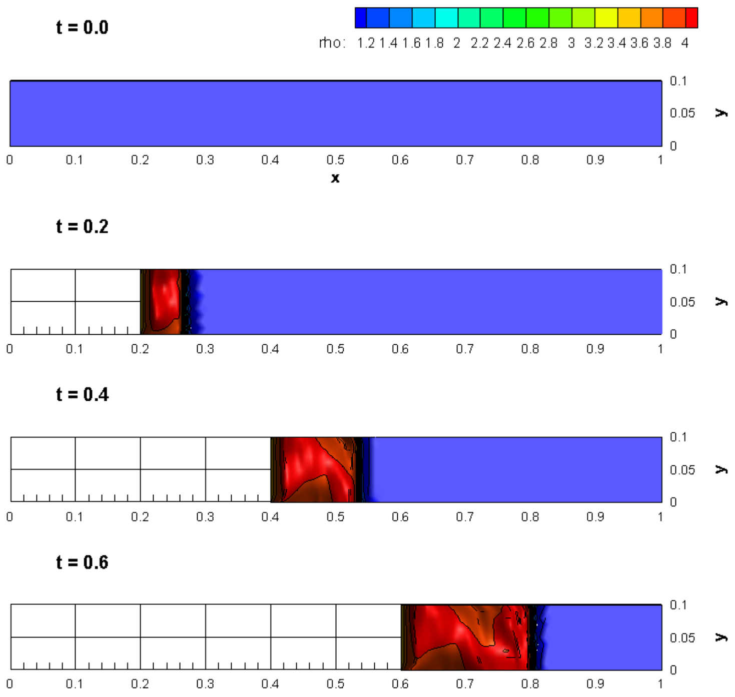


Figure 6.16: Evolution of the density solution for the Saltzman problem at output times $t = 0$, $t = 0.2$, $t = 0.4$ and $t = 0.6$. The numerical simulation has been run with the QF ALE ADER-WENO formulation and the node solver $\mathcal{NS}_m$.

everywhere. The bottom panels present the density (left) and the horizontal component of the velocity (right) as a function of x for all cells versus the exact solution (red line). Apart from the classical wall heating effect close to the moving boundary condition on the right of the figure and a slight overshoot after the shock wave, the results are in good agreement with the exact solution.

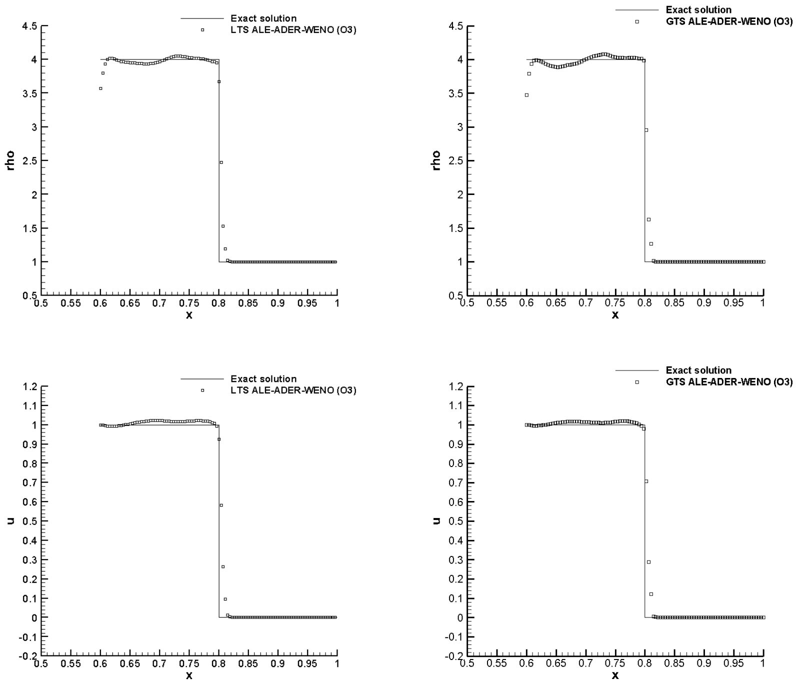


Figure 6.17: Third order accurate numerical solution for the Saltzman problem at the final time $t_f = 0.6$ with the node solver $\mathcal{NS}_{cs}$. *Left panels*: solution obtained with LTS. *Right panels*: solution obtained with GTS.

6.1.6 The Sedov problem

The numerical results for the Sedov problem (see Section 5.1.6) will be presented in the following. Two- and three-dimensional results are given in separate subsections. We always use the Rusanov-type (2.69) numerical flux.

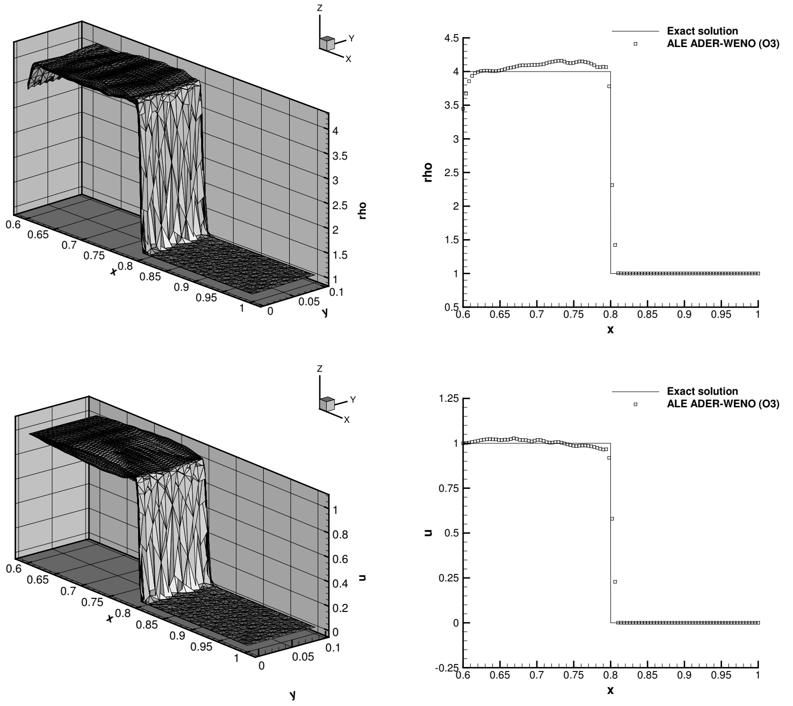


Figure 6.18: Third order numerical results with ALE ADER-WENO schemes for the Saltzman problem: density (top) and velocity (bottom) distribution and comparison with analytical solution at time $t = 0.6$.

2D numerical results.

Figure 6.20 contains the numerical results obtained with a third order ALE ADER-WENO scheme using the multidimensional HLL flux and the node solver $\mathcal{NS}_b$. The initial grid is composed by (30×30) square elements, each of them divided into two right triangles, hence giving a total number of $N_E = 1800$ control volumes. The mesh is highly distorted and compressed by the shock

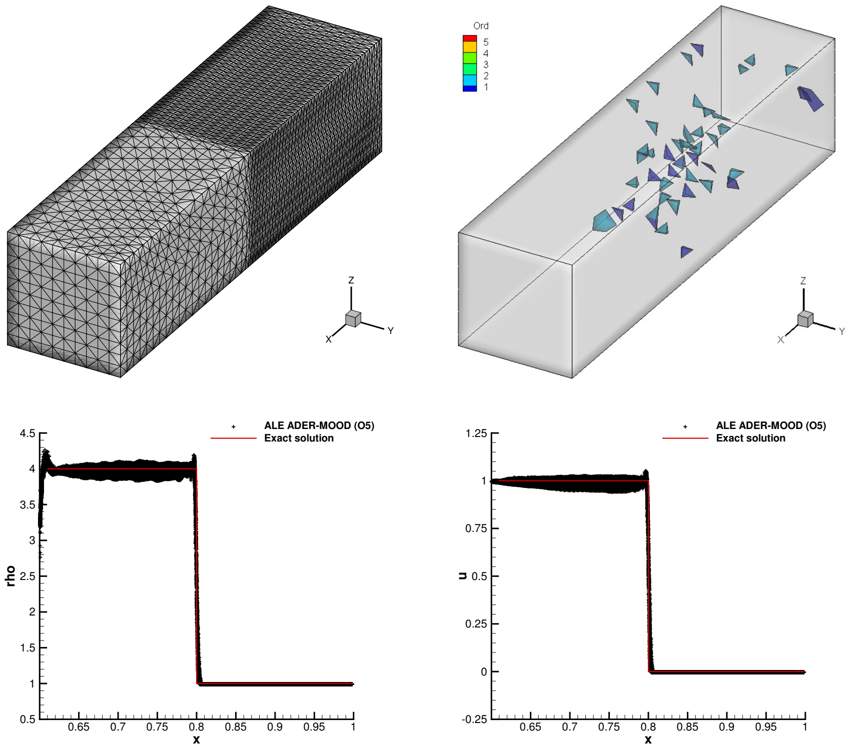


Figure 6.19: Saltzman problem in 3D at $t_f = 0.6$ for ALE ADER-MOOD with $\mathbb{P}_4$ polynomial reconstruction results. Top: final mesh configuration and cell orders. Bottom: density and horizontal velocity component u as a function of x for all cells.

wave, but the numerical solution agrees well with the exact solution, as depicted in Figure 6.20. The rezoning step described in Section 3.2 of Chapter 3 was necessary in order to reduce the mesh deformation and to avoid tangled elements.

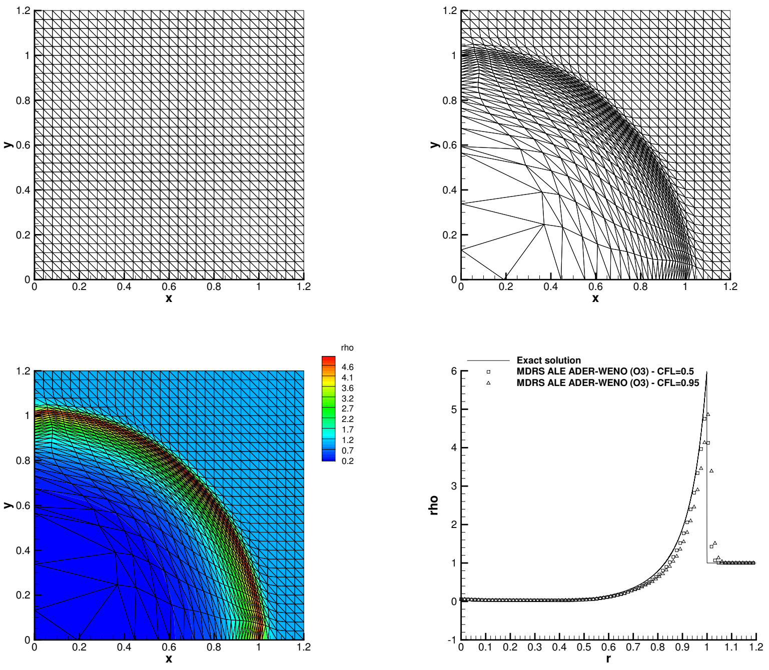


Figure 6.20: Numerical results for the Sedov problem with a third order accurate MDRS ALE ADER-WENO scheme. Top: initial and final mesh configuration. Bottom: density distribution at the final time $t_f = 1.0$ and comparison between the exact solution (solid line) and two different third order accurate numerical solution obtained with CFL = 0.5 and CFL = 0.95.

Higher order simulations have been carried out with the ALE ADER-MOOD version of the algorithm. Figure 6.21 shows the evolution of the blast wave and the final density distribution in *all* elements of the same mesh used before. We can notice that the *a posteriori* limiting is active only on the shock frontier, where the order of the scheme has been lowered to 1, while in the rest of the computational domain the method can run at the maximum fifth order of accuracy.

3D numerical results.

As done in [168] we consider two different meshes, the first one m_1 is composed by $20 \times 20 \times 20$ cubes, while the second one m_2 involves $40 \times 40 \times 40$ elements. We use the third order accurate version of the ALE ADER-WENO schemes together with the Rusanov-type numerical flux (2.69) and the node solver $\mathcal{NS}_m$. The positivity preserving algorithm illustrated in Appendix B has been employed. The numerical solution for the Sedov problem has been computed on both meshes m_1 and m_2. Figure 6.22 shows the solution for density at the final time of the simulation as well as the mesh configuration and a comparison between the numerical and the exact density distribution along the radial direction.
We also run the three-dimensional Sedov problem using the fifth order accurate ALE ADER-MOOD algorithm on the computational mesh m_2. To emphasize the gain in accuracy, in Figure 6.23 we compare these results with the corresponding fifth order results obtained with the original ALE ADER-WENO formulation. The density as a function of cell radius for all cells is reported for both schemes against the exact solution in red line. One can clearly notice that the density peak is better retrieved by the MOOD approach.

6.1.7 The Noh problem

Here we consider the Noh problem (see Section 5.1.7) using the Rusanov-type (2.69) numerical flux. Two- and three-dimensional results are given in separate subsections.

2D numerical results.

We solve the Noh test case with the MDRS version of our numerical method. The domain is discretized with a total number $N_E = 5000$ of triangles and we use from second up to fourth order accurate finite volume schemes with the

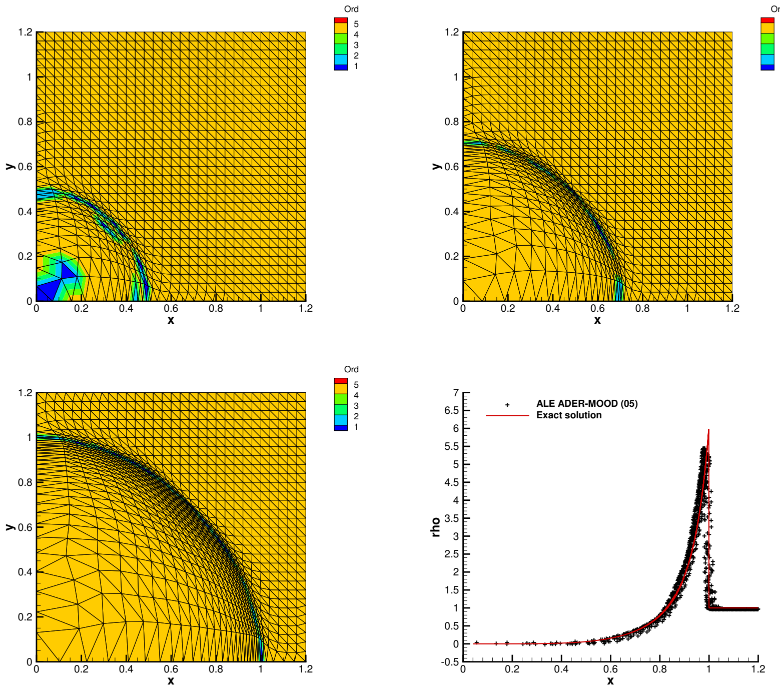


Figure 6.21: Sedov problem at $t_f = 1.0$ for fifth order accurate ALE ADER-MOOD schemes. From top-left to bottom-right panels: mesh and cell polynomial degrees at intermediate times $t = 0.25$, $t = 0.5$, $t = 1.0$ and scatter plot of the cell density as a function of cell radius versus the exact solution (red line).The node solver $\mathcal{NS}_m$ has been used.

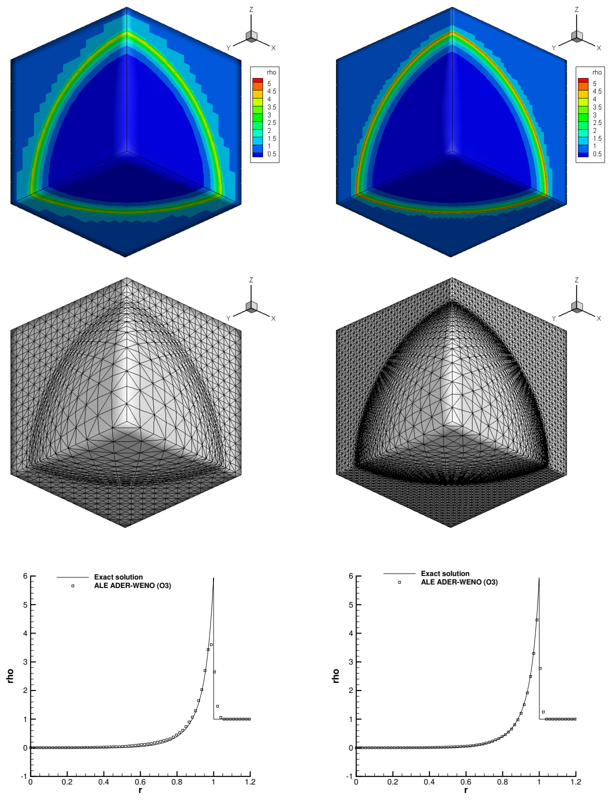


Figure 6.22: Third order results for the Sedov problem with ALE ADER-WENO schemes on the coarse grid m_1 (left column) and on the fine grid m_2 (right column). From top to bottom: solution for density at the final time of the simulation (top row), mesh configuration at the final time $t_f = 1.0$ (middle row) and comparison between analytical and numerical density distribution along the diagonal straight line that crosses the cubic computational domain (bottom row).

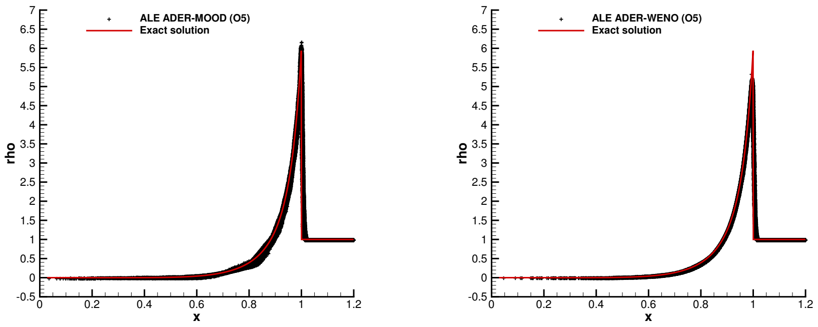


Figure 6.23: Fifth order numerical results for the Sedov problem in 3D at $t_f = 1.0$. Left: ALE ADER-MOOD results. Right: ALE ADER-WENO results. Comparison between density distribution as a function of cell radius for all cells and the exact solution (red line).

node solver $\mathcal{NS}_b$. Figure 6.24 shows the initial and the final mesh configuration and a comparison between the exact solution and three high order accurate numerical results obtained with the ALE ADER-WENO finite volume schemes based on genuinely multidimensional HLL Riemann solvers. A Courant number of CFL $= 0.9$ has been used for all the numerical simulations and one can notice that the quality of the solution becomes the better as the order of accuracy of the scheme increases.

We also use the fifth order version of our ALE ADER-MOOD schemes to run the Noh test problem. The percentage of cells updated with $\mathbb{P}_1^{\text{LIM}}$ (red) or $\mathbb{P}_0$ (blue) polynomial reconstruction is plotted in Figure 6.25 as a function of the iteration number. The panel on the left presents the first 50 iterations when the explosion occurs. We clearly see that about $50\% - 60\%$ of the total number of cells are recomputed with a low order scheme for the first 15 timesteps. This is a large amount of extra work which, nonetheless, is mandatory to stabilize the scheme. However, the overall efficiency is not too drastically affected, see the CPU time and memory consumptions provided in Table 6.16 of Section 6.1.13. Then, when the numerical scheme has generated enough numerical

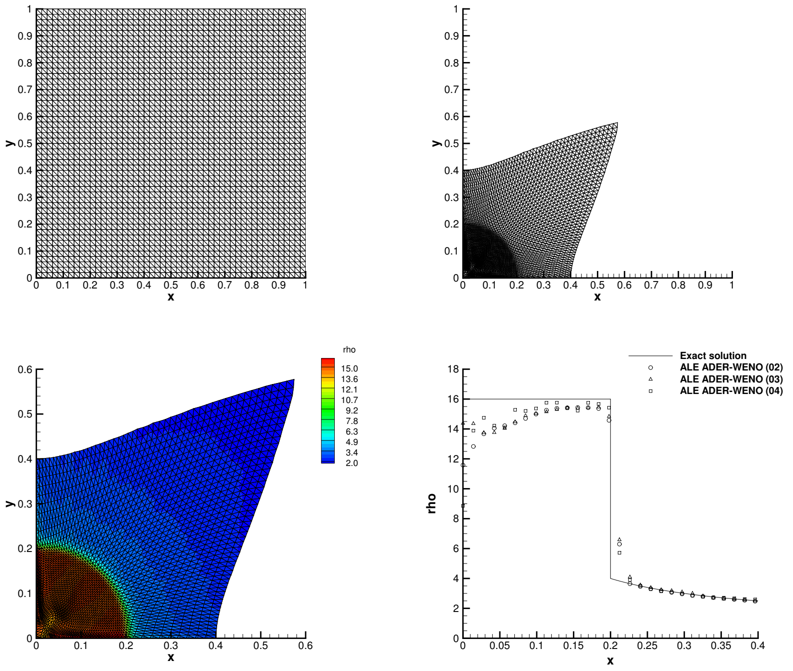


Figure 6.24: Top: mesh configuration for the Noh problem at the initial time $t = 0$ and at the final time $t_f = 0.6$. Bottom: fourth order accurate density distribution at the final time and comparison between the exact solution (solid line) and three different high order accurate numerical results, i.e. 2^{nd}, 3^{rd} and 4^{th} order ALE ADER-WENO finite volume schemes using the multidimensional HLL Riemann solver with CFL = 0.9.

dissipation to deal with the initial shock wave, the number of detected and decremented cells drops to few percents. From iteration ≈ 100 (see right panel of Figure 6.25) the number of problematic cells is back to a small amount, about $0.5\% - 1.5\%$. This figure illustrates that the Noh problem is difficult to handle at the very beginning, but, as soon as the shock wave has emerged, no more dramatic action has to be taken by the MOOD process. Moreover, we have plotted a cubic fit of these sample points in the right panel taken into account only the sample points after iteration 50.

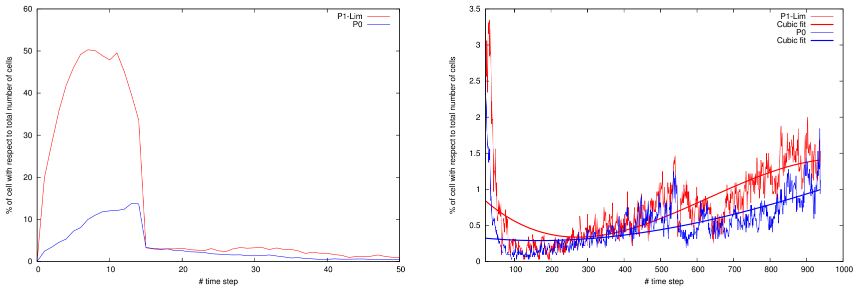


Figure 6.25: Noh problem in 2D with fifth order accurate ALE ADER-MOOD schemes. Percentage of cells updated with $\mathbb{P}_1^{\text{LIM}}$ (red) or $\mathbb{P}_0$ (blue) reconstruction as a function of iteration number. Left: first part of the simulation before iteration 50 corresponding to the generation of the shock wave. Right: second part of the simulation. Cubic fits are presented on the second panel only.

3D numerical results.

We use the ALE ADER-MOOD schemes to perform a three-dimensional fifth order accurate simulation of the Noh problem with the node solver $\mathcal{NS}_m$. The computational mesh is constructed with $N^3 = 40^3$ hexahedra which are further split into 5 tetrahedra leading to a total number of elements $N_E = 32 \times 10^4$. For this difficult problem we observe that the solution is slightly perturbed by parasitical phenomenon mostly arising from the no-slip boundary conditions. Nevertheless the spherical shock wave is well located. The number of problematic cells seems low and they are mostly located close to the shock front.

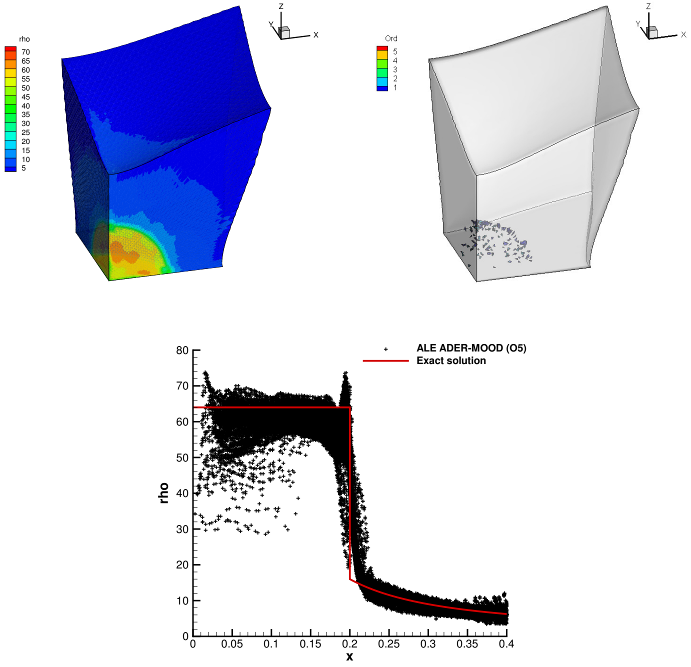


Figure 6.26: Noh problem in 3D at $t_f = 0.6$ with ALE ADER-MOOD schemes using $\mathbb{P}_4$ polynomial reconstruction. On the top row we plot the final mesh configuration (left) and the final cell order (right). A comparison between density as a function of cell radius for all cells and the exact solution is depicted in the bottom row.

6.1.8 The Gresho vortex problem

The square shaped computational domain is discretized with two different meshes m_1 and m_2 with a characteristic mesh size of $h = 1/30$ and $h = 1/60$, respectively. We use the second up to fourth order version of our two-dimensional ALE ADER-WENO schemes with the Rusanov-type numerical flux (2.69) and the node solver $\mathcal{NS}_m$ to obtain the numerical results depicted in Figure 6.27, where the numerical solution is compared against the analytical solution. The higher is the order of accuracy the more accurate is the numerical distribution for both density and pressure. The order of accuracy of the scheme is furthermore validated thorough the convergence studies listed in Table 6.12. The mesh configurations at the initial time as well as at the final time of the simulation for each order of accuracy are presented in Figure 6.28: the computational grid is highly distorted especially for high order schemes because the flow trajectories are approximated and followed much better, hence resulting in a more complicated deformation path.

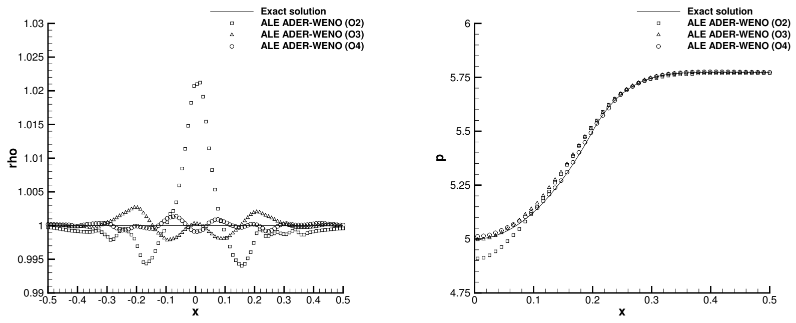


Figure 6.27: Comparison between analytical solution and second, third and fourth order accurate numerical results for density (left) and pressure (right) distribution of the Gresho vortex problem. The original ALE ADER-WENO algorithm (in 2D) has been used together with the node solver $\mathcal{NS}_m$ and the very robust Rusanov-type numerical flux (2.69).

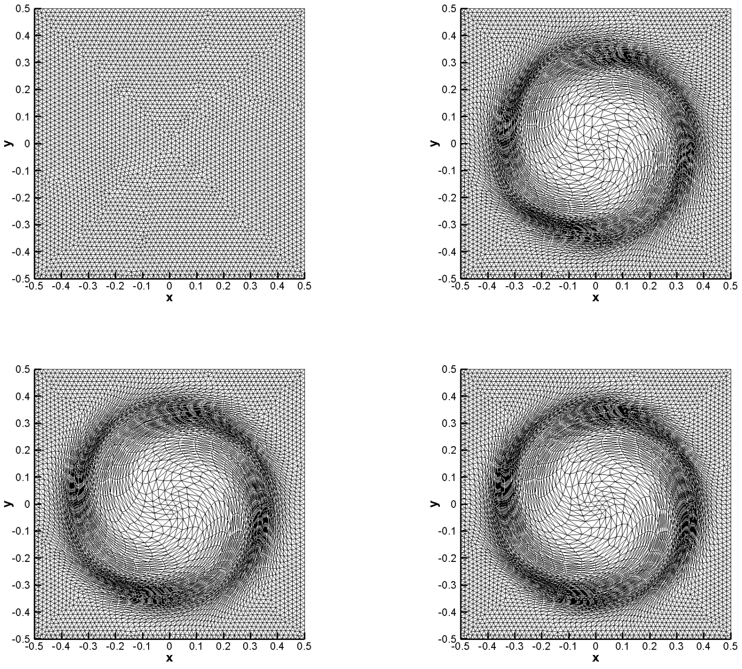


Figure 6.28: Mesh configuration at the initial time $t = 0.0$ (top left) and at the final time $t = 1.0$ for the *two-dimensional* Gresho vortex problem using second (top right), third (bottom left) and fourth (bottom right) order accurate direct ALE ADER-WENO schemes.

Table 6.12: L_2 error norms and convergence rates for the *two-dimensional* Gresho vortex test case. The norms refer to density and have been computed for the second up to fourth order numerical results at the final time $t_f = 1.0$. $h(\Omega(t_f))$ denotes the final mesh size.

$h(\Omega(t_f))$	ϵ_{L_2}	$\mathcal{O}(L_2)$
	$\mathcal{O}2$	
3.59E-02	3.347E-03	-
2.84E-02	2.213E-03	1.8
	$\mathcal{O}3$	
3.71E-02	2.474E-03	-
2.58E-02	9.531E-04	2.6
	$\mathcal{O}4$	
3.70E-02	7.999E-04	-
2.70E-02	2.251E-04	4.0

6.1.9 The Taylor-Green vortex problem

For the *three-dimensional* Taylor-Green vortex problem the computational domain is represented by a box which is discretized with a characteristic mesh size of $h = 0.05$, see Figure 6.29.
We run the Taylor-Green vortex test with the second up to fourth order accurate version of our ALE ADER-WENO schemes using the Rusanov-type flux (2.69) as Riemann solver together with the positivity preserving technique described in Appendix B. Pressure distribution as well as the velocity field at the final time are depicted in Figure 6.30 for the fourth order accurate numerical results. A strong mesh deformation occurs, as clearly shown in Figure 6.29, with the corner elements that are highly stretched by the vortex flow. We also perform the same test case on a coarser grid with a characteristic mesh size of $h = 0.10$ in order to check that the order of accuracy is still preserved properly. The convergence rates as well as the L_2 error norms for density are listed in Table 6.13 and the results show that the designed order of the scheme has been reached in this test case. This also means that the flattener variable of the positivity preserving technique (see Appendix B) has not been triggered within the computation of such a smooth flow problem, despite the severe mesh distortion that occurs here.

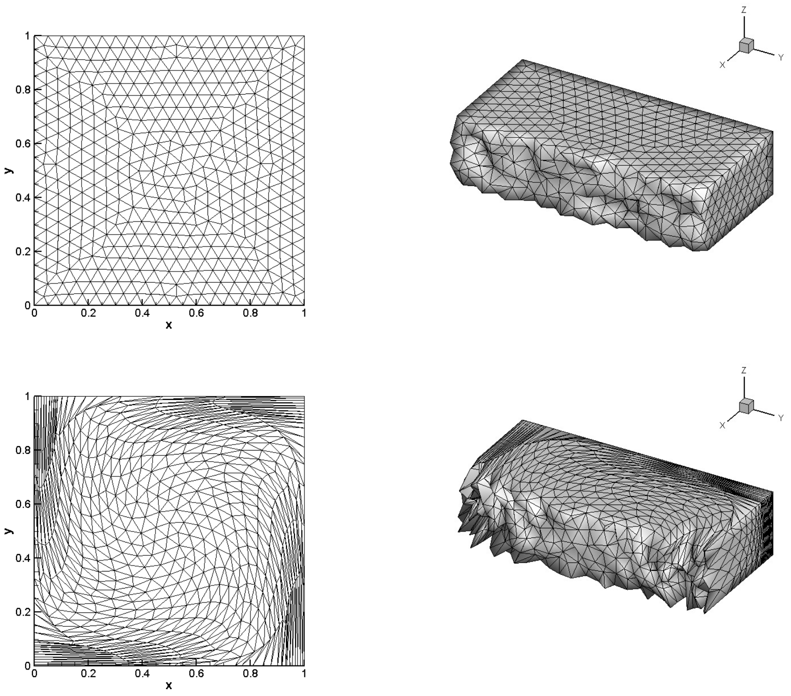


Figure 6.29: Initial (top) and final (bottom) configuration of the computational grid used for the *three-dimensional* Taylor-Green vortex test problem. The mesh size is $h = 0.05$ and the total number of elements is $N_E = 15846$.

6.1.10 Mono-material triple point problem

We use the ALE ADER-MOOD schemes to run the mono-material triple point problem both in 2D and 3D, using a fifth order accurate scheme together with the Rusanov-type (2.69) numerical flux and the node solver $\mathcal{NS}_m$. In 2D the mesh is made by $N_E = 19098$ triangles in such a way that it perfectly matches the discontinuities among the three subdomains, while in 3D we use

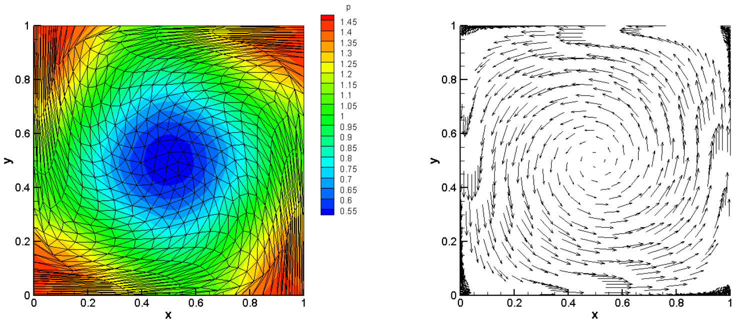


Figure 6.30: Fourth order accurate numerical results for the three-dimensional Taylor-Green vortex at the final time $t_f = 0.7$ obtained using the ALE ADER-WENO finite volume schemes with the node solver $\mathcal{NS}_m$ and the Rusanov-type (2.69) numerical flux. Left: pressure contours and final mesh configuration. Right: vectors of the velocity field.

$N_E = 304246$ tetrahedra to discretize the cylindrical computational domain by exactly matching the state interfaces.

Figure 6.31 shows the *two-dimensional* results, where density, specific internal energy and final mesh configuration are depicted. The vortex shape is clearly captured and the mesh seems to follow the flow field. This behavior is also illustrated in Figure 6.32 where we have colored the cells according to the initial subdomain index $(1, 2 \text{ or } 3)$. For a mesh moving scheme one expects that the mesh follows the flow with a quasi-vortex like velocity field. Nevertheless the numerical diffusion and rezoning procedure (see Section 3.2) of the ALE scheme can not allow such a vortex motion as mesh tangling will inexorably occur. This is why we can not observe in Figure 6.32 a perfect vortex shape.

The numerical results for the 3D triple point problem are displayed in Figure 6.33, where we present the specific internal energy, the grid configuration and the cell order at the final time of the simulation. Two 3D view of the specific internal energy are displayed: one on the left showing the boundary faces, and the other one on the right displaying the result configuration for all cells located below the plane $z = x$. The mesh and cell order figures adopt the same view.

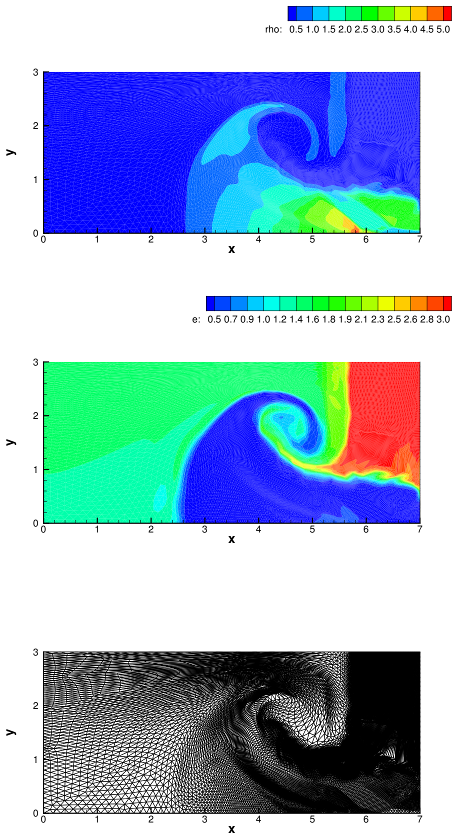


Figure 6.31: Fifth order ALE ADER-MOOD numerical results for the triple point problem in 2D at $t_f = 5.5$. From top to bottom: cell density, cell specific internal energy and mesh configuration.

Table 6.13: L_2 error norms and convergence rates for the Taylor-Green vortex test case in 3D. The norms refer to density and have been computed for the second up to fourth order numerical results with ALE ADER-WENO at the final time $t_f = 0.7$. $h(\Omega(t_f))$ denotes the final mesh size.

$h(\Omega(t_f))$	ϵ_{L_2}	$\mathcal{O}(L_2)$
	$\mathcal{O}2$	
7.66E-02	1.518E-02	-
3.99E-02	3.866E-03	2.1
	$\mathcal{O}3$	
7.35E-02	1.890	-
4.16E-02	3.269	3.1
	$\mathcal{O}4$	
7.34E-02	7.461E-03	-
4.15E-02	4.627E-04	4.8

The mesh is clearly following the flow field, hence arising to highly deformed and stretched elements. Nevertheless the final cell orders reach the maximal value for almost all cells. At last we present the percentage of cells updated with $\mathbb{P}_1^{\text{LIM}}$ (red) or $\mathbb{P}_0$ (blue) reconstructions as a function of iteration number. Left panel presents the first 100 iterations whereas the right panel displays the remaining iterations along with cubic fits of the data in straight lines. From these data we observe that for both simulations, for any timestep, less than 1% of cells are decremented, apart from the first 10 iterations when the shocks emanate from the discontinuities.

This problem is a difficult one for mesh moving schemes with fixed connectivity as the vortex motion of the flow implies that sooner or later invalid or highly stretched elements will appear. At this point, to avoid the failure of the code, a stronger mesh relaxation is mandatory leading to a more Eulerian-like end of the simulation.

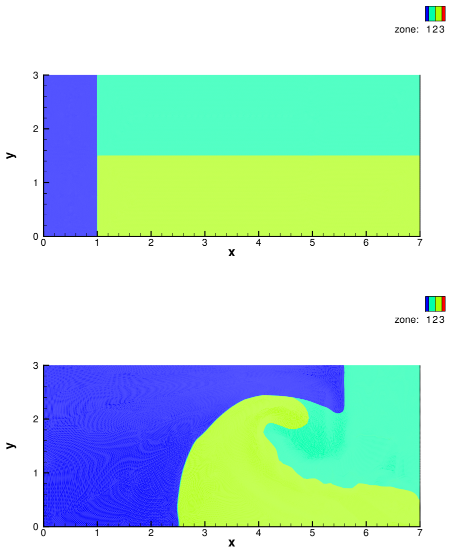


Figure 6.32: Fifth order ALE ADER-MOOD numerical results for the triple point problem in 2D at $t_f = 5.5$. Cells colored corresponding to their initial subdomains, according to the notation given in Section 5.1.11.

6.1.11 Multi-material flow

For the solution of the two-material Sod problem described in Section 5.1.12, the third order version together with the node solver $\mathcal{NS}_{cs}$ have been used to obtain the numerical results shown in Figure 6.34, where a general good agreement with the exact solution is achieved for density as well as for pressure. The use of the Osher-type numerical flux (2.71) allows the contact wave to be very well resolved, hence admitting at most three mixing cells along the x-direction, as clearly shown by the concentration profile in Figure 6.34. Since we use an additional transport equation for the mass fractions, also the present way to handle multi-material flows can be interpreted as a *diffuse interface method*, like the one used in our previous paper [96], where we have adopted

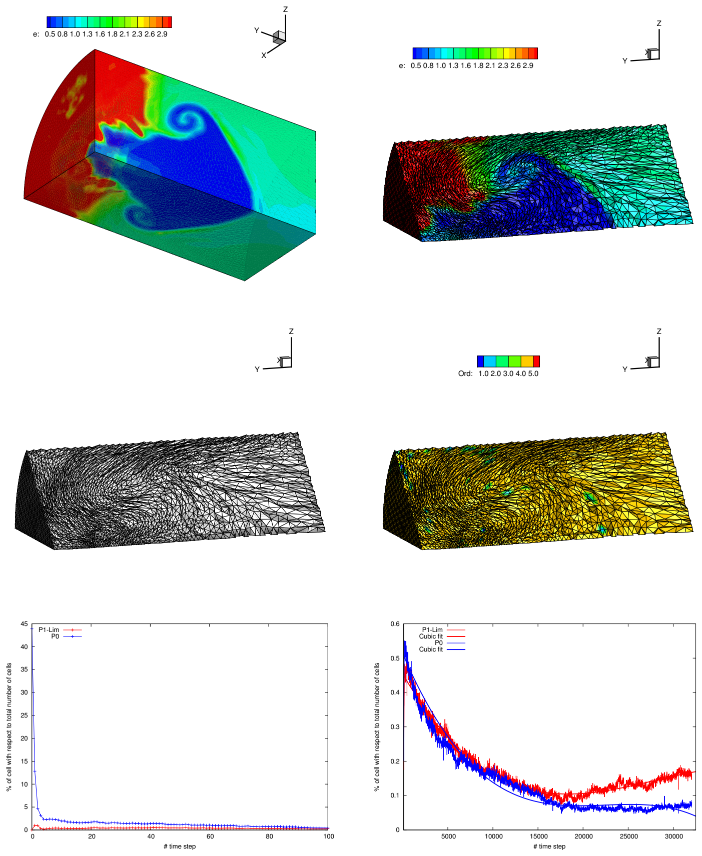

e: 0.5 0.8 1.0 1.3 1.6 1.8 2.1 2.3 2.6 2.9
X
Y
Z
Ord: 1.0 2.0 3.0 4.0 5.0
P1-Lim
P0
Cubic fit
% of cell with respect to total number of cells
# time step

Figure 6.33: Fifth order ALE ADER-MOOD numerical results for the triple point problem in 3D at $t_f = 5.5$. Top: specific internal energy. Middle: mesh and cell orders. Bottom: percentage of cells updated with $\mathbb{P}_1^{\text{LIM}}$ (red) or $\mathbb{P}_0$ (blue) reconstructions as a function of the iteration number. Left panel presents the first 100 iterations whereas the right panel displays the remaining iterations along with cubic fits of the data in straight lines.

the framework of the full seven-equation Baer-Nunziato model of compressible multi-phase flows.

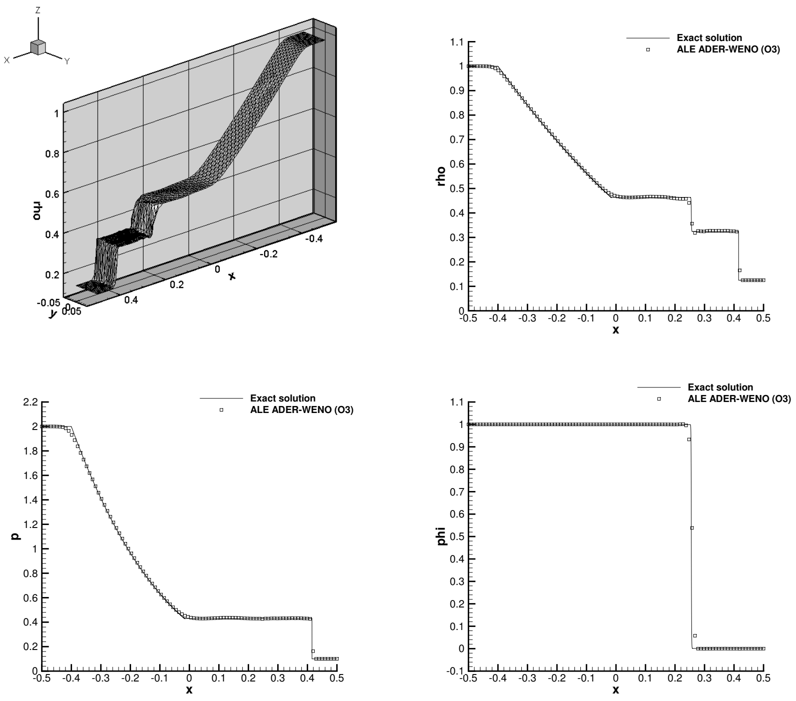


Figure 6.34: Numerical results for the two-material Sod shock tube problem at time $t = 0.2$ using a third order accurate ALE ADER-WENO scheme with the node solver ($\mathcal{NS}_{cs}$) in 2D. Density (top right), pressure (bottom left) and mass fraction ϕ_1 (bottom right) are shown as well as a three-dimensional view of the density distribution (top left).

6.1.12 The two-dimensional double Mach reflection problem

We solve the double Mach reflection problem (DMR) presented in Section 6.1.12 using the third order version of the classical ALE ADER-WENO schemes together with the Rusanov type numerical flux (2.69) and the node solver $\mathcal{NS}_m$. The numerical results are shown in Figure 6.35, while Figure 6.36 the final mesh configuration is depicted. We notice a very strong mesh compression induced by the shock wave. For this test case, the use of the rezoning strategy introduced in Section 3.2 of Chapter 3 is vital.

This test case has also been run with the *quadrature-free* approach, described in Section 4.3 of Chapter 4. A comparison in terms of computational efficiency can be shown later in Table 6.15 of Section 6.1.13.

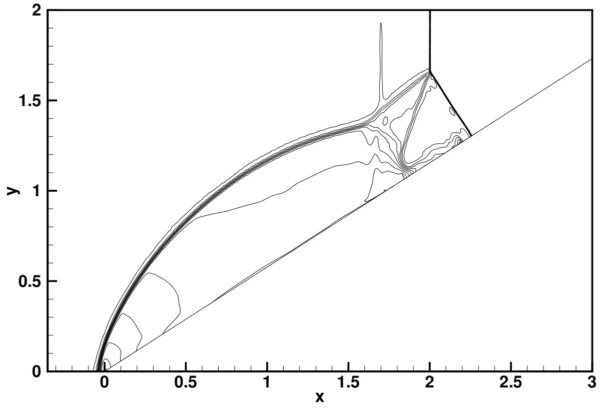
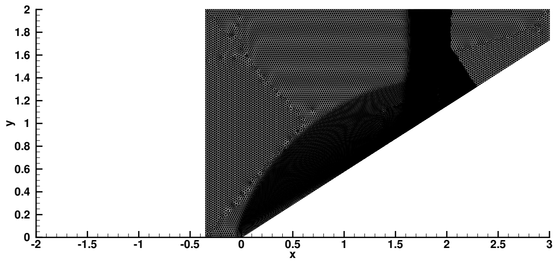


Figure 6.35: Density contours for the double Mach reflection problem at the final time $t = 0.2$.



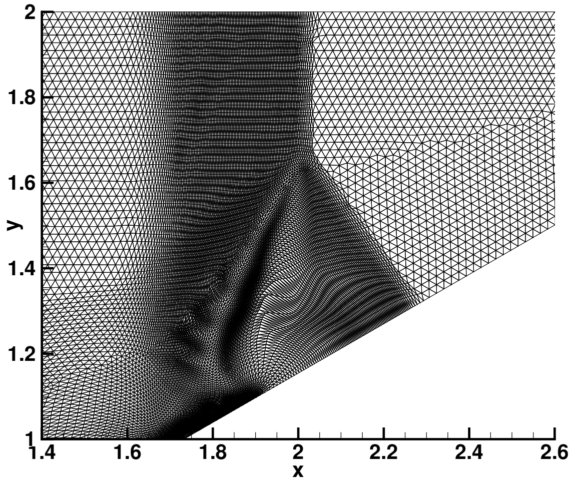


Figure 6.36: Computational domain and mesh at the final time $t = 0.2$ (top) and zoom into the final mesh for the double Mach reflection problem.

6.1.13 Efficiency comparison

Comparison between LTS and GTS ALE ADER-WENO schemes.

Table 6.14 aims at showing the computational efficiency of the LTS algorithm w.r.t. the classical ALE ADER-WENO schemes with global time stepping (GTS) presented in Chapter 2. In order to give a fair comparison between LTS and GTS schemes, the efficiency is not measured in terms of computational time, which may depend on the machine hardware or on the algorithm implementation, but rather we count the total number of element updates needed to reach the final time of the simulation, as done in [93]. We consider the Riemann problems and the explosion problems presented in Sections 6.1.2 and 6.1.3, respectively. All the details of each simulation can be found there. Looking at Table 6.14, we notice that the ALE algorithm with global time stepping requires a total number of element updates that is a factor of 3-4 times larger than the one of our new ALE scheme with LTS, approaching a factor of 5 for the explosion problem EP2.

Table 6.14: Comparison of the computational efficiency between GTS (**GTS AAW**) and LTS (**LTS AAW**) ALE ADER-WENO algorithm in terms of the total number of element updates for RP1,RP2,EP1 and EP2. A third order scheme has been adopted for each test case as well as the Osher-type (2.71) numerical flux with the node solver $\mathcal{NS}_{cs}$. The efficiency $\mathcal{A}$ is measured as the ratio between the total number of element updates needed for the GTS and for the LTS approach.

	Number of element updates		
2D Tests	**GTS AAW $\mathcal{O}3$**	**LTS AAW $\mathcal{O}3$**	$\mathcal{A}$
RP1	$10.120404 \cdot 10^6$	$3.257847 \cdot 10^6$	3.11
RP2	$23.349964 \cdot 10^6$	$5.020780 \cdot 10^6$	4.45
EP1	$30.804224 \cdot 10^6$	$12.206887 \cdot 10^6$	2.52
EP2	$181.412376 \cdot 10^6$	$38.274477 \cdot 10^6$	4.74

Comparison between ALE ADER-WENO and QF ALE ADER-WENO schemes.

This section is meant to assess the gain in terms of computational efficiency that arises from the quadrature-free (QF) ADER-WENO algorithm presented in Section 4.3 of Chapter 4. Table 6.15 reports the CPU time needed to run some of the test cases for the Euler equations of compressible gas dynamics shown in previous sections. Furthermore we report also the computational time τ which was necessary to update the solution of one element within one timestep, in order to get rid of any mesh dependency. The results have been collected running each of the test problem in parallel on four CPU cores of an Intel i7-2600 processor with 3.4 GHz of clock speed. We could verify that the QF ALE ADER-WENO formulation allows the computation to be carried out almost 2 times faster in 2D, while we gain an average factor of 2.7 in 3D. For a more detailed analysis we refer the reader to [44].

Comparison between ALE ADER-WENO and ALE ADER-MOOD schemes.

In this section we summarize the CPU time and memory consumption for a wide range of 2D and 3D tests for the Euler equations of compressible gas dynamics by comparing the ALE ADER-WENO algorithm (see Chapter 2) with the ALE ADER-MOOD version (see Section 4.4 of Chapter 4). These schemes are of the same nominal order of accuracy and are implemented within the same framework. We use fifth order of accuracy and we consider all the test problem proposed in [45]. All the data listed in Table 6.16 refers to the two-dimensional version of the schemes and have been collected running each of the test case using a single CPU core of an Intel i7-2600 processor with 3.4 GHz of clock speed and 16 GB of RAM, in order to assess the pure serial performance, without accounting for the MPI overhead. Table 6.17 considers the three-dimensional results, where data have been collected running each of the test problem in parallel on 1024 processors. Further details can be found in [45].
From Table 6.16 and 6.16 we observe that the gain is systematically in favor of a MOOD approach in our ALE framework for these test cases. On average the acceleration is about 1.9 for $d = 2$ and 2.6 for $d = 3$, meaning that MOOD needs about $2 - 2.5$ times less CPU time than WENO for the same mesh configuration. The saving in terms of memory is about $1.6 - 1.8$ in favor of MOOD. In [165] for a fixed grid Eulerian MOOD and WENO schemes the ratio was different, namely a bigger memory saving (ratio 3) and a smaller

Table 6.15: Computational efficiency of the QF (quadrature-free) and the original GQ (Gauss-quadrature) ALE ADER-WENO algorithm, addressed with **QF AAW** and **GQ AAW**, respectively. We consider the Saltzman problem, the Noh problem and the double Mach reflection problem (DMR) in 2D, while in 3D we present the explosion problem, the Kidder problem and the Sedov problem. Each test case has been run using a third (in 2D) and a fourth (in 3D) order accurate scheme with the Rusanov type flux (2.69) and the node solver $\mathcal{NS}_m$. $\tau_E = \frac{CPUtime}{N_E \cdot N}$ gives the time used per element update, while $\mathcal{A} = \tau_E^{GF}/\tau_E^{QF}$ indicates the efficiency of the new QF ALE schemes w.r.t. the original formulation of the algorithm described in Chapter 2.

2D Tests	**QF AAW $\mathcal{O}3$**		**GQ AAW $\mathcal{O}3$**		**Efficiency**
	CPU time	τ_E^{QF}	CPU time	τ_E^{GQ}	$\mathcal{A}$
Saltzman	$4.73 \cdot 10^3$	$3.27 \cdot 10^{-4}$	$8.74 \cdot 10^3$	$6.11 \cdot 10^{-4}$	1.9
Noh	$1.65 \cdot 10^4$	$4.89 \cdot 10^{-3}$	$6.64 \cdot 10^6$	$1.20 \cdot 10^{-2}$	2.5
DMR	$1.73 \cdot 10^6$	$3.88 \cdot 10^{-4}$	$2.57 \cdot 10^6$	$5.45 \cdot 10^{-4}$	1.4
				Average	***1.9***

3D Tests	**QF AAW $\mathcal{O}4$**		**GQ AAW $\mathcal{O}4$**		**Efficiency**
	CPU time	τ_E^{QF}	CPU time	τ_E^{GQ}	$\mathcal{A}$
Explosion	$1.67 \cdot 10^8$	$6.18 \cdot 10^{-3}$	$3.99 \cdot 10^7$	$3.67 \cdot 10^{-2}$	3.7
Kidder	$9.56 \cdot 10^5$	$7.60 \cdot 10^{-3}$	$1.22 \cdot 10^6$	$3.38 \cdot 10^{-2}$	1.4
Sedov	$3.21 \cdot 10^6$	$2.02 \cdot 10^{-3}$	$5.49 \cdot 10^6$	$4.67 \cdot 10^{-3}$	2.9
				Average	***2.7***

acceleration (1.3), both in favor of MOOD. Contrarily to an Eulerian context, here in our ALE framework we can not pre-compute and store all WENO reconstruction matrices. Indeed they change at each time step according to the geometry evolution. Then, in our ALE framework more computation and less storing is performed by WENO approach. On the other hand the MOOD approach, which always needs less reconstructions than WENO, needs less on-the-fly computation to solve the linear reconstruction systems (2.9).
The information provided by Tables 6.16-6.17 only provide a general idea of the gain brought by the use of an *a posteriori* MOOD treatment in replacement of a WENO reconstruction and limiting. Both approaches have different

Table 6.16: Summary of CPU time needed to update one cell within one timestep ($\times 10^{-2}$ in seconds) and memory consumption (in MB) for the *two-dimensional* test cases proposed in [45]. ALE ADER-WENO (**AAW**) versus ALE ADER-MOOD (**AAM**) fifth order accurate schemes. The last two columns show the ratio between the CPU times ("acceleration" $\mathcal{A}$) and the memory consumption ("saving" $\mathcal{M}$).

2D Tests	**AAW** $\mathcal{O}5$		**AAM** $\mathcal{O}5$		**Efficiency**	
	CPU time $\times 10^{-2}$s	Memory $\times 10^{4}$MB	CPU time $\times 10^{-2}$s	Memory $\times 10^{4}$MB	$\mathcal{A}$	$\mathcal{M}$
Sod	2.3018	62.97	1.1882	40.42	1.94	1.56
Sedov	4.7396	53.76	2.7457	35.67	1.73	1.51
Noh	8.6977	79.97	4.5598	47.96	1.91	1.67
Kidder	8.9423	78.54	4.1643	47.53	2.15	1.65
Saltzman	7.2889	56.23	3.5914	36.04	2.03	1.56
Triple point	39.65	307.20	23.32	192.00	1.70	1.60
				Average	***1.9***	***1.6***

Table 6.17: Summary of CPU time needed to update one cell within one timestep ($\times 10^{-2}$ in seconds) and memory consumption (in MB) for the *three-dimensional* test cases proposed in [45]. ALE ADER-WENO (**AAW**) versus ALE ADER-MOOD (**AAM**) fifth order accurate schemes. The last two columns show the ratio between the CPU times ("acceleration" $\mathcal{A}$) and the memory consumption ("saving" $\mathcal{M}$).

3D Tests	**AAW** $\mathcal{O}5$		**AAM** $\mathcal{O}5$		**Efficiency**	
	CPU time $\times 10^{-2}$s	Memory $\times 10^{4}$MB	CPU time $\times 10^{-2}$s	Memory $\times 10^{4}$MB	$\mathcal{A}$	$\mathcal{M}$
Sod	37.89	1.55	15.16	0.85	2.50	1.83
Sedov	7.67	2.59	3.26	1.52	2.35	1.70
Noh	6.83	2.55	3.12	1.54	2.19	1.66
Kidder	7.92	1.84	2.19	0.98	3.64	1.87
Saltzman	10.53	0.05	4.14	0.03	2.54	1.85
Triple point	34.10	3.28	14.09	1.94	2.42	1.69
				Average	***2.6***	***1.8***

user/developer parameters that may be ticked to improve the general efficiency. We would like to emphasize the fact that this comparison refers to our own implementation of both approaches.

6.2 The ideal magnetohydrodynamics (MHD) equations

6.2.1 Numerical convergence studies

We use the two-dimensional vortex test case described in Section 5.2.1 to perform the convergence studies for the MHD equations presented in Section 5.2 of Chapter 5.
Table 6.18 reports the convergence rates from first up to fifth order accurate ALE ADER-WENO schemes using *each* of the node solvers illustrated in Section 3.1 of Chapter 3. The Osher-type (2.71) numerical flux has been used in all computations and the designed order of accuracy is well preserved with each node solver.
We also perform the vortex problem with the MDRS version of our algorithm, see Section 4.2. We run this test case on four successively refined meshes from first up to fourth order of accuracy and for each simulation we compute the error in L_2 norm, according to (5.7). The multidimensional HLLC Riemann solver for the MHD equations has been used, see [88], together with the node solver $\mathcal{NS}_b$.

6.2.2 The MHD rotor problem

2D ALE ADER-WENO results.

We use a computational grid with a characteristic mesh size of $h = 1/200$ to run the two-dimensional MHD rotor problem. Numerical results obtained with a fourth order ALE ADER-WENO scheme with the Rusanov-type flux (2.69) and the node solver $\mathcal{NS}_b$ are depicted in Figure 6.37. We can notice a good agreement with the solution presented in [20], although the mesh used for the simulation is coarser than the one adopted by Balsara and Spicer.
The MHD rotor problem has also been tested in the framework of genuinely multidimensional Riemann solvers (MDRS) for moving meshes, hence adopting the MDRS ALE ADER-WENO algorithm illustrated in Section 4.2 of Chapter 4. There, we could use a CFL number of CFL = 0.95 and we produce third order accurate results, which look in good agreement with the solution presented in [20]. We refer the reader to [38] for further details.

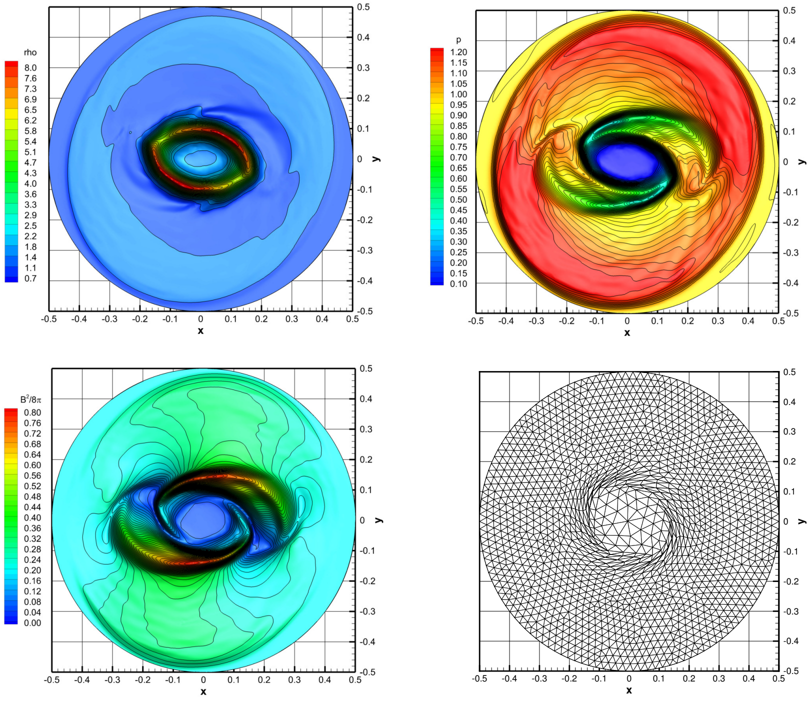


Figure 6.37: Numerical results for the two-dimensional ideal MHD rotor problem: density, pressure, magnetic pressure and a coarse mesh configuration at time $t = 0.25$. A 4^{th} order direct ALE ADER-WENO scheme has been used with the rezoning stage and the multidimensional node solver $\mathcal{NS}_b$ (see Section 3.1.3 of Chapter 3).

Table 6.18: Numerical convergence results for the ideal MHD equations. The first up to fifth order version of the two-dimensional ALE one-step ADER-WENO finite volume scheme has been used for each node solver type. The error norms refer to the variable ρ (density) at time $t = 1.0$.

	$\mathcal{NS}_{cs}$		$\mathcal{NS}_m$		$\mathcal{NS}_b$	
$h(\Omega(t_f))$	ϵ_{L_2}	$\mathcal{O}(L_2)$	ϵ_{L_2}	$\mathcal{O}(L_2)$	ϵ_{L_2}	$\mathcal{O}(L_2)$
			$\mathcal{O}1$			
3.26E-01	2.7330E-03	-	2.7059E-03	-	2.7381E-03	-
2.37E-01	2.0111E-03	0.96	2.0173E-03	0.90	2.0173E-03	0.93
1.64E-01	1.3081E-03	1.17	1.3055E-03	1.20	1.3113E-03	1.20
1.28E-01	9.5497E-04	1.26	9.5150E-04	1.30	9.5617E-04	1.28
			$\mathcal{O}2$			
3.26E-01	4.8091E-03	-	4.7707E-03	-	5.5971E-03	-
2.35E-01	2.8382E-03	1.61	2.8571E-03	1.58	2.7874E-03	2.13
1.64E-01	1.4212E-03	1.91	1.4239E-03	1.88	1.3789E-03	1.94
1.28E-01	6.4686E-04	3.24	6.4610E-04	3.26	7.2141E-04	2.67
			$\mathcal{O}3$			
3.25E-01	1.1417E-03	-	1.1376E-03	-	1.1265E-03	-
2.36E-01	1.8935E-04	5.57	1.8930E-04	5.56	1.8632E-04	5.56
1.63E-01	7.1734E-05	2.65	7.1740E-05	2.65	7.1912E-05	2.60
1.28E-01	3.1651E-05	3.38	3.1653E-05	3.38	3.1738E-05	3.38
			$\mathcal{O}4$			
3.26E-01	2.4858E-04	-	2.4864E-04	-	2.4472E-04	-
2.35E-01	7.9871E-05	3.50	7.9875E-05	3.50	7.9884E-05	3.45
1.63E-01	2.1790E-05	3.55	2.1791E-05	3.55	2.1795E-05	3.55
1.28E-01	8.2013E-06	4.03	8.2014E-06	4.03	8.1998E-06	4.03
			$\mathcal{O}5$			
3.26E-01	1.2010E-04	-	1.2010E-04	-	1.1992E-04	-
2.35E-01	2.7365E-05	4.56	2.7359E-05	4.56	2.7327E-05	4.56
1.63E-01	4.8779E-06	4.71	4.8778E-06	4.71	4.8898E-06	4.70
1.28E-01	1.3947E-06	5.17	1.3947E-06	5.17	1.3935E-06	5.18

3D ALE ADER-WENO results.

The computational domain is discretized with a total number of tetrahedra of $N_E = 1089071$. The numerical results for the three-dimensional MHD rotor problem have been obtained using the third order version of the ALE ADER-WENO schemes presented in Chapter 2 with the Rusanov-type flux (2.69) and they are depicted in Figure 6.38. Furthermore we use the $\mathcal{NS}_m$ which has been

Table 6.19: Numerical convergence results for the ideal MHD equations using the ALE ADER-WENO finite volume schemes with genuinely multidimensional HLL Riemann solvers presented in Section 4.2 of Chapter 4. The error norms refer to the variable ρ (density) at time $t = 1.0$ for first up to fourth order version of the scheme.

$h(\Omega(t_f))$	ϵ_{L_2}	$\mathcal{O}(L_2)$	$h(\Omega, t_f)$	ϵ_{L_2}	$\mathcal{O}(L_2)$
	$\mathcal{O}1$			$\mathcal{O}2$	
3.26E-01	5.4032E-03	-	3.25E-01	1.2393E-02	-
2.36E-01	4.7048E-03	0.4	2.46E-01	9.5840E-03	0.9
1.63E-01	4.0697E-03	0.4	1.63E-01	5.7617E-03	1.2
1.28E-01	3.5298E-03	0.6	1.28E-01	3.5875E-03	2.0
	$\mathcal{O}3$			$\mathcal{O}4$	
6.75E-01	1.6836E-02	-	6.73E-01	1.9276E-02	-
3.25E-01	3.3009E-03	2.2	3.26E-01	1.0209E-03	4.1
2.47E-01	1.0170E-03	4.3	2.47E-01	2.6494E-04	4.9
1.63E-01	2.9097E-04	3.0	1.63E-01	5.1003E-05	4.0

extended to the MHD equations, following the procedure described in Section 3.1.2 of Chapter 3. Although the mesh adopted for the simulation is coarser than the one used in [20], we can observe a good qualitative agreement with the solution presented in [20] (note that the present simulation is carried out in 3D). The rezoning procedure described in Section 3.2 allows the mesh to be reasonably well shaped, even with the strong deformations produced by the velocity field of the rotor. Figure 6.39 shows the initial and the final mesh configuration and the corresponding density distribution.

6.2.3 The MHD blast wave problem

For the MHD blast wave problem we use the same mesh adopted for the MHD rotor problem described in the previous section both in two and in three space dimensions.

2D MDRS ALE ADER-WENO results.

Here we show the numerical results obtained with the MDRS version (see Section 4.2) of the original ALE ADER-WENO method. The numerical results

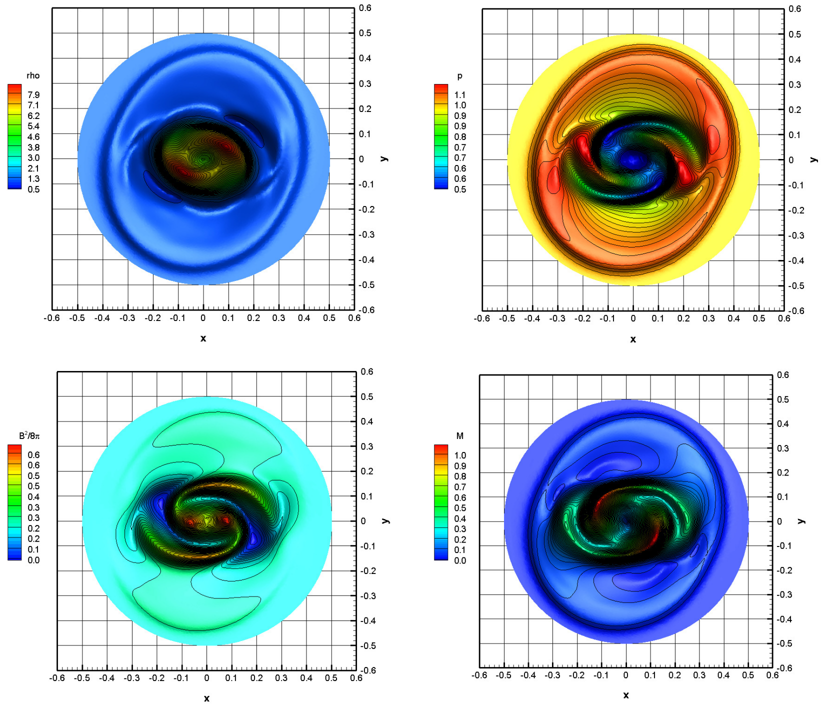


Figure 6.38: Third order ALE ADER-WENO numerical results for the ideal MHD rotor problem at time $t = 0.25$. Top: density and pressure. Bottom: magnitude of the magnetic field and Mach number.

depicted in Figure 6.40 have been computed using the third order accurate version of the ALE ADER-WENO finite volume scheme with the multidimensional HLL flux for the MHD equations [88] and CFL = 0.95. We plot the logarithm of density and pressure, as well as the magnitude of both the velocity and the magnetic field, and the solution looks very similar to the results given in [16].

Due to the very strong shock wave, the velocity of the flow is quite high and the fluid is pushed towards the left and the right part of the computational domain.

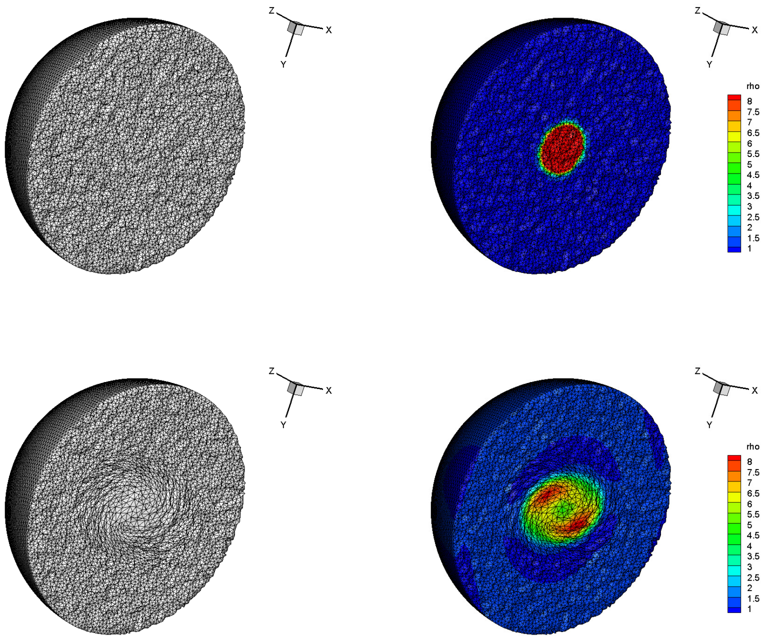


Figure 6.39: Mesh configuration and density distribution for the three-dimensional MHD rotor problem at the initial time $t = 0.0$ (top) and at the final time $t = 0.25$ (bottom).

Therefore we used the rezoning algorithm presented in Section 3.2, which allows the mesh elements to recover a more regular shape in order to carry on the simulation until the final time t_f. Figure 6.41 shows a comparison between the fully Lagrangian mesh configuration and the rezoned mesh configuration at time $t = 0.004$.

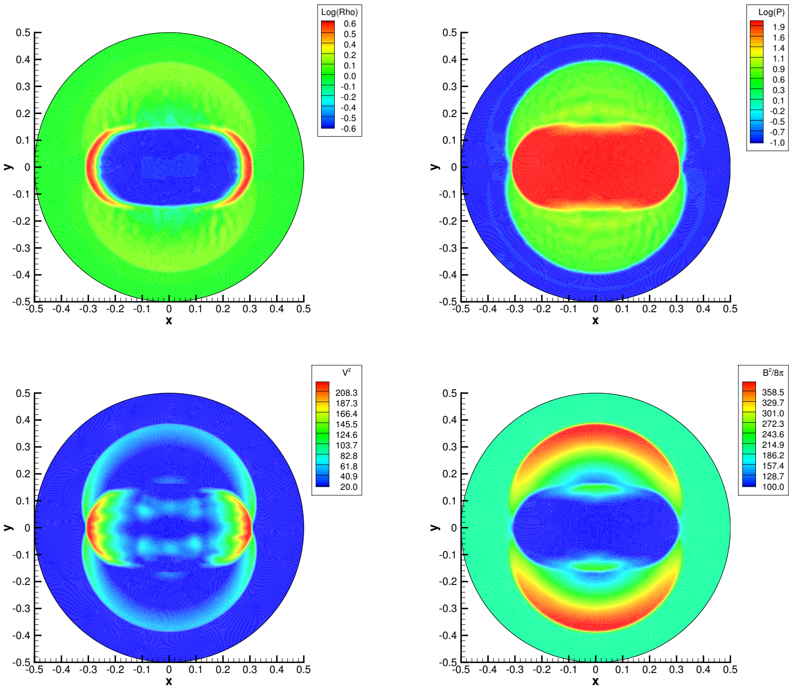
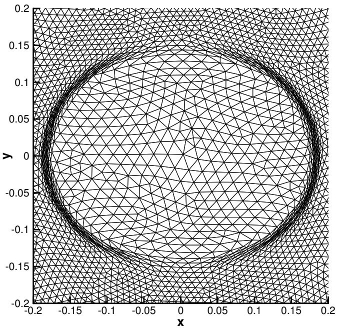


Figure 6.40: Numerical results for the *two-dimensional* MHD blast wave problem at time $t = 0.01$ obtained with a third order accurate MDRS ALE ADER-WENO scheme. Top: logarithm (base 10) of the density and logarithm (base 10) of the pressure. Bottom: magnitude of the velocity and the magnetic field.



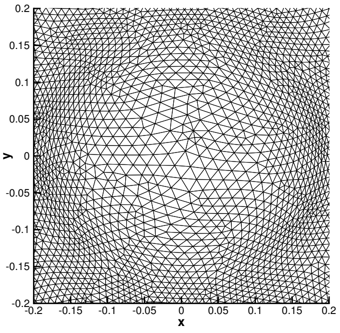


Figure 6.41: Mesh configurations for the MHD blast wave problem in 2D at time $t = 0.004$. Left: fully Lagrangian mesh motion. Right: Lagrangian mesh motion with the rezoning stage (Section 3.2 of Chapter 3).

3D ALE ADER-WENO results.

The numerical results in 3D have been obtained with the third order version of the ALE ADER-WENO schemes using the Rusanov-type flux (2.69) and the node solver $\mathcal{NS}_m$. The numerical solution is depicted in Figure 6.42, where the logarithm of density and pressure, as well as the magnitude of both the velocity and the magnetic field are reported. The solution is in qualitative agreement with the results shown previously for $d = 2$ in Figure 6.40, where the two-dimensional version of our Lagrangian-like WENO algorithm with MDRS has been used to run this test case. The tetrahedral mesh at the final time $t = 0.01$ is depicted in Figure 6.43.

6.3 The relativistic MHD equations (RMHD)

Next we show the numerical results obtained with the ALE ADER-WENO algorithm for the *two-dimensional* RMHD equations presented in Section 5.3 of the previous chapter.

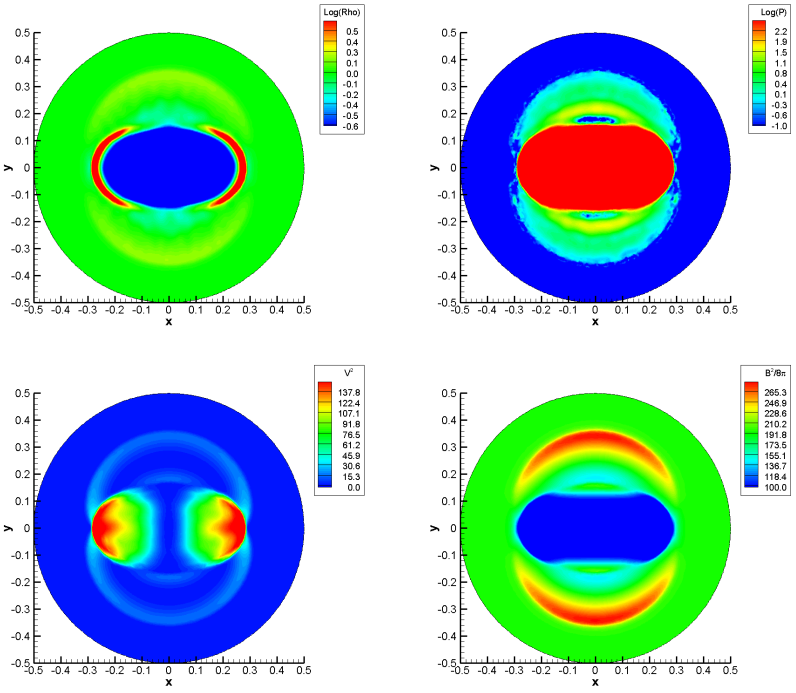


Figure 6.42: Third order ALE ADER-WENO numerical results for the *three-dimensional* Blast problem at time $t = 0.01$. Top: logarithm (base 10) of the density and logarithm (base 10) of the pressure. Bottom: magnitude of the velocity field and the magnetic field.

6.3.1 Large Amplitude Alfvén wave

This test case is used to assess the accuracy of our numerical scheme for the RMHD equations. Table 6.20 reports the errors ϵ_{L2} and the measured convergence orders $\mathcal{O}_{L2}$ in L^2 norm for the flow variable B_y. The number $1/h$ denotes the reciprocal characteristic mesh spacing along each coordinate direction. We underline that a very high level of accuracy can be achieved on very coarse

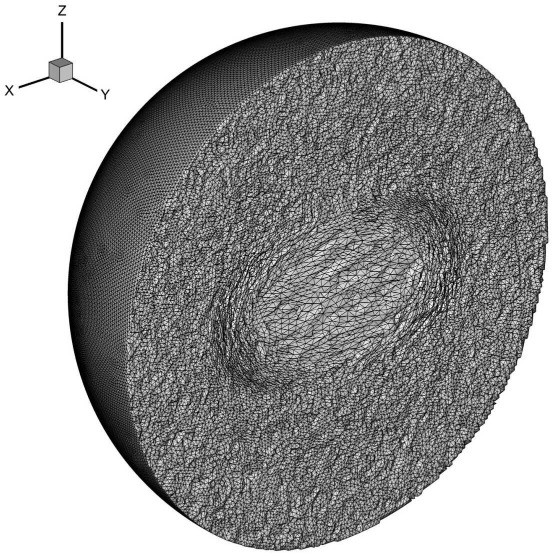


Figure 6.43: View of the unstructured tetrahedral grid at time $t = 0.01$ for the 3D MHD blast wave problem.

meshes with the fourth and fifth order scheme compared to the third order method even if the latter is run on much finer grids.

6.3.2 The RMHD rotor problem

The circular computational domain is discretized with a total number of elements $N_E = 71046$. Figure 6.44 displays the evolution of the pressure distribution up to the final time $t_f = 0.4$, while in Figure 6.45 the corresponding mesh configurations are reported, obtained with a third order ALE ADER-WENO scheme and the Rusanov-type (2.69) flux. The results obtained with the high order direct ALE WENO scheme on a moving unstructured mesh agree qualitatively well with those obtained previously by an Eulerian WENO method on a fixed mesh in [95]. As far as we know, these are the first results obtained for the RMHD equations with a high order Lagrangian-type finite volume scheme.

Table 6.20: Numerical convergence study of third, fourth and fifth order direct ALE ADER-WENO finite volume schemes for the relativistic MHD equations (RMHD) in 2D. Errors refer to the variable B_y.

	$\mathcal{O}3$			$\mathcal{O}4$			$\mathcal{O}5$	
$1/h$	ϵ_{L2}	$\mathcal{O}_{L2}$	$1/h$	ϵ_{L2}	$\mathcal{O}_{L2}$	$1/h$	ϵ_{L2}	$\mathcal{O}_{L2}$
50	1.4270E-04		25	6.9640E-05		25	1.0749E-05	
100	1.7436E-05	3.03	50	3.0158E-06	4.53	50	4.5265E-07	4.57
150	5.1826E-06	2.99	75	5.8315E-07	4.05	75	4.1669E-08	5.88
200	2.1831E-06	3.01	100	1.7717E-07	4.09	100	9.8553E-09	5.52

6.3.3 The RMHD blast wave problem

We use a mesh with a characteristic mesh size of $h = 1/200$ and a total number of $N_E = 71046$ triangles to discretize the initial computational domain. We rely on the third order accurate version of the ALE ADER-WENO finite volume scheme with the simple Rusanov-type flux (2.69) and the simple node solver $\mathcal{NS}_{cs}$ to run this test problem, since more sophisticated Riemann solvers and node solvers are very difficult to obtain for this very complicated system. For an HLLC-type Riemann solver of the RMHD equations see the papers by Mignone and Bodo [177] and of Honkkila and Janhunen [143]. An Osher-Solomon-type flux for RMHD has been recently proposed by Dumbser and Toro in the framework of universal Osher-type fluxes for general nonlinear hyperbolic conservation laws in [106], see eqn. (2.71), however, for this stringent test problem the more dissipative and more robust Rusanov flux (2.69) was needed. The numerical results obtained with the rezoning strategy (see Section 3.2) switched on are depicted in Figure 6.46. The contour colors of the magnetic field component B_y are reported, together with a fine grid Eulerian reference simulation carried out with a third order ADER-WENO scheme on a mesh with 282860 elements and characteristic mesh spacing $h = 1/400$.

6.4 The Baer-Nunziato model of compressible two-phase flows

In the following we present some numerical results obtained with the full seven-equation Baer-Nunziato model with relaxation source terms described in Section 5.4 of Chapter 5. Since this physical model involves two phases, namely

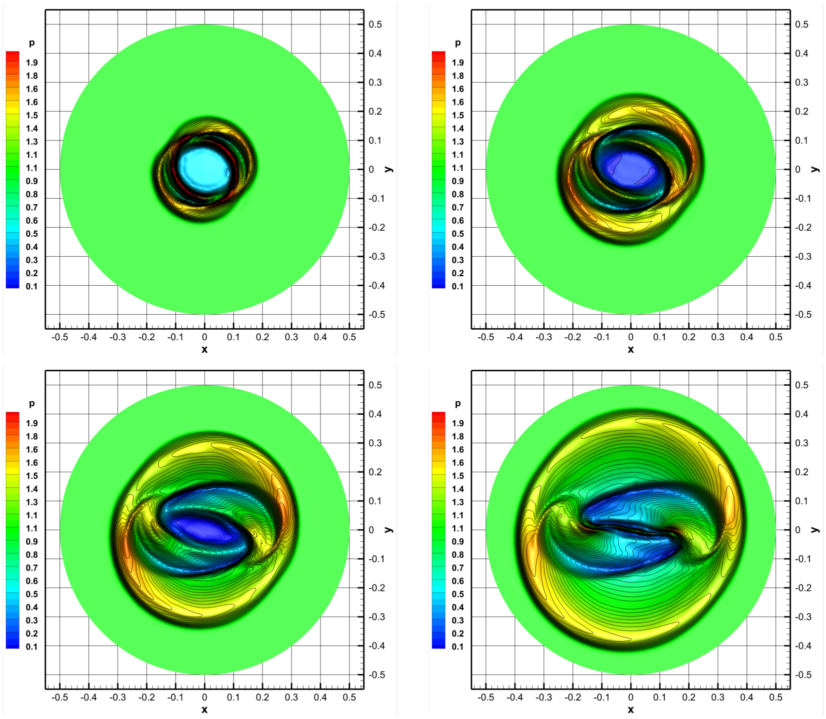


Figure 6.44: Results for the pressure p for the RMHD rotor problem in 2D at output times $t = 0.10$, $t = 0.20$, $t = 0.30$ and $t = 0.40$.

the solid phase and the gas phase, we decide to move the mesh with the *solid phase velocity*, hence

$$\mathbf{V} = \boldsymbol{u_I} = \boldsymbol{u}_1, \tag{6.2}$$

which coincides with the interface velocity, according to our assumptions (see Section 5.4). Furthermore the node solver $\mathcal{NS}_{cs}$ presented in Section 3.1.1 of Chapter 3 is adopted because it can be easily applied to different hyperbolic systems due to its very general formulation. If not stated otherwise, we use a third order direct ALE ADER-WENO scheme with the Osher-type method

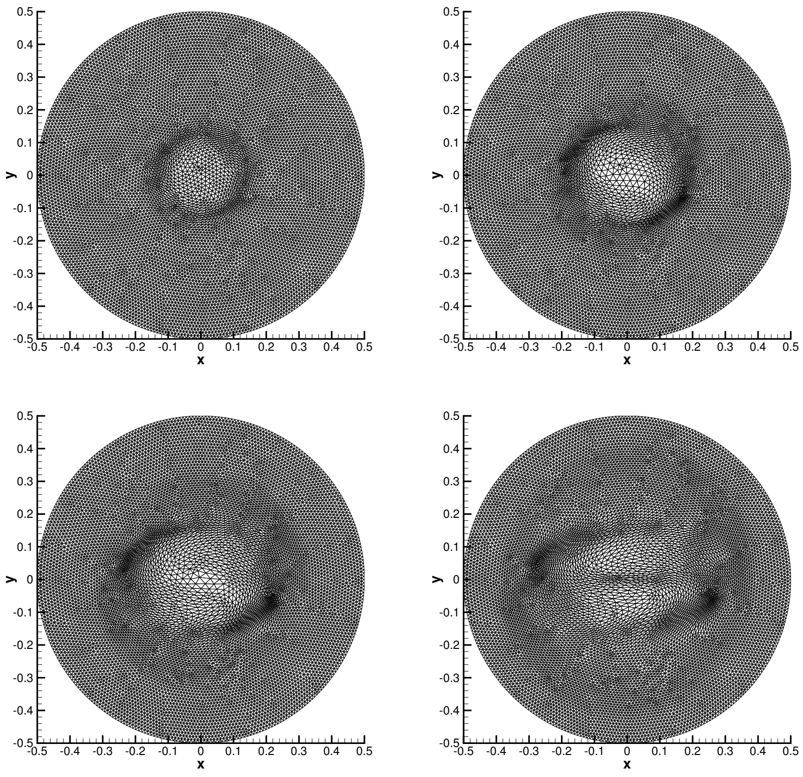


Figure 6.45: Evolution of a coarse version of the moving unstructured Lagrangian mesh with rezoning for the RMHD rotor problem at output times $t = 0.10$, $t = 0.20$, $t = 0.30$ and $t = 0.40$.

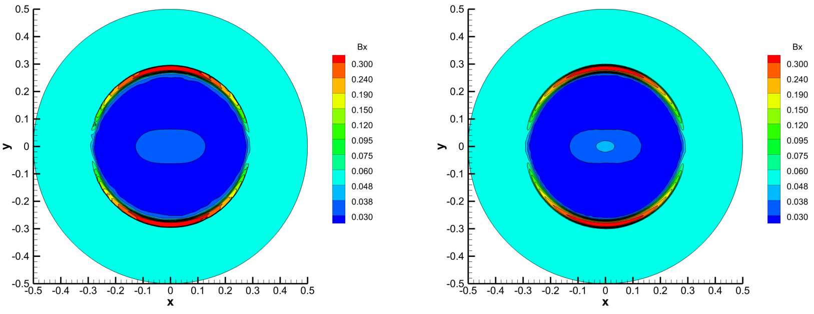


Figure 6.46: Results for the magnetic field component B_x for the RMHD blast wave problem in 2D at the final time $t = 0.30$. 11 color contours are exponentially distributed between 0.03 and 0.3. Left: Third order direct ALE ADER-WENO scheme on a grid with $h = 1/200$. Right: Eulerian reference solution computed with a third order ADER-WENO scheme on a fine grid ($h = 1/400$).

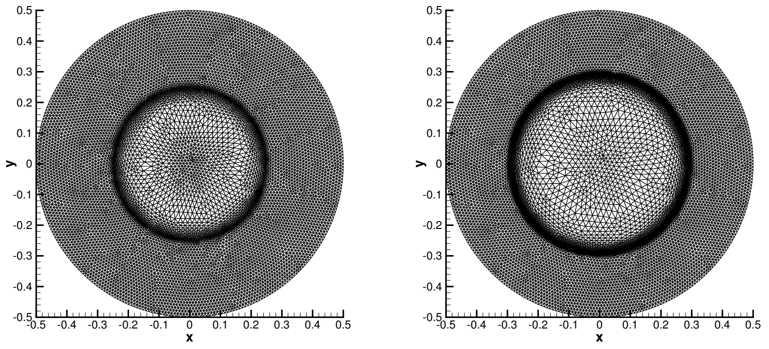


Figure 6.47: Evolution of a coarse version of the moving unstructured Lagrangian mesh with rezoning for the RMHD blast wave problem at output times $t = 0.20$ and $t = 0.30$.

(2.65). This setting is the same for *all* the test cases listed below. Finally, in all figures the abbreviation ALE ADER-WENO has been replaced with ALE WENO to ease the visualization.

In Section 6.4.5 we show a comparison between the direct ALE ADER-WENO algorithm presented in Chapter 2 and the corresponding Eulerian version proposed in [99, 102, 103]. Both accuracy and efficiency will be compared and the analysis demonstrates that Lagrangian-like algorithms are more accurate but also more expensive in terms of computational efforts, as expected.

6.4.1 Numerical convergence studies

The numerical convergence rates for the compressible Baer-Nunziato model have been carried out using the Osher-type (2.65) numerical flux and the results are shown for the solid volume fraction ϕ_s at time $t = 2.0$ in Table 6.21 for a sequence of successively refined meshes with a number of elements N_G along each direction. Similar results have been obtained for all variables of the state vector $\boldsymbol{Q}$. One observes that the schemes reach their designed order of accuracy quite well. To our knowledge, this is the first time ever that a better than second order accurate Lagrangian-like WENO finite volume scheme is presented for non-conservative hyperbolic systems on unstructured triangular meshes with applications to the Baer-Nunziato model of compressible multi-phase flows.

6.4.2 Riemann problems

The numerical results for the Riemann problems presented in Section 5.4.2 are shown in Figures 6.48 - 6.51 and are compared with the exact solution. RP1 and RP4 have been run in 2D, while for RP2 and RP5 the three-dimensional results are displayed. On the top left of each figure a sketch of the mesh is depicted, while the other subfigures contain a one-dimensional cut through the reconstructed numerical solution $\boldsymbol{w}_h$ along the x-axis, evaluated at the final time on 200 *equidistant* sample points. Due to the Lagrangian-like formulation of the method, the solid contact is resolved in a very sharp manner in all cases, which was actually the main aim in the design of a high order ALE schemes for the compressible Baer-Nunziato model. Also for the other waves we can note in general a very good agreement between our numerical results and the exact reference solutions given in [11, 76, 212].

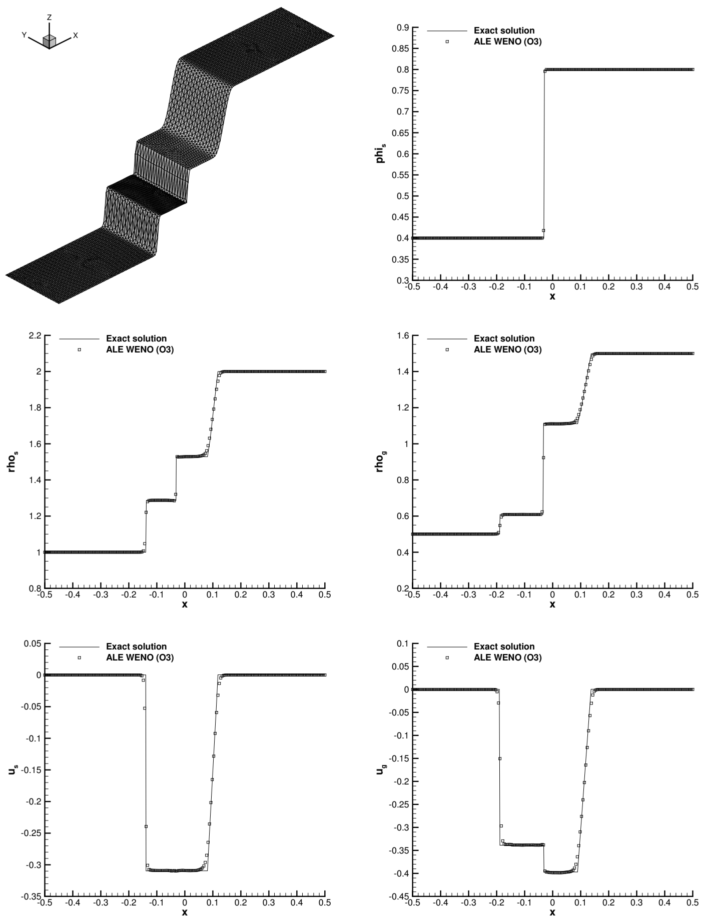


Figure 6.48: Results for Riemann problem RP1 of the seven-equation Baer-Nunziato model in 2D at time $t = 0.1$.

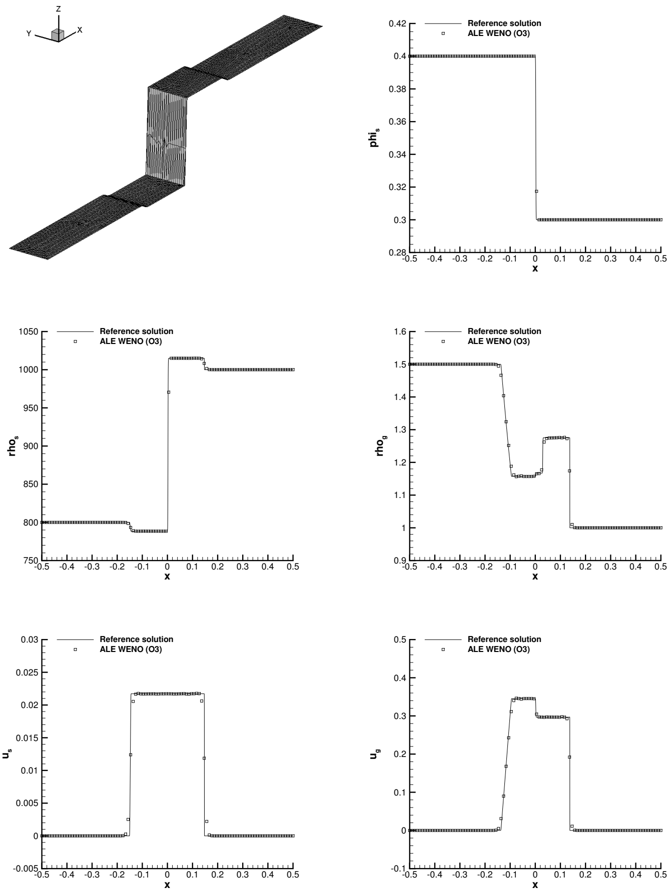


Figure 6.49: Results for Riemann problem RP2 of the seven-equation Baer-Nunziato model in 3D at time $t = 0.1$.

Table 6.21: Numerical convergence results for the compressible Baer-Nunziato model in 2D using the third to sixth order version of the direct Arbitrary-Lagrangian-Eulerian one-step ADER-WENO finite volume schemes presented in this work. The error norms refer to the variable ϕ_s (solid volume fraction) at time $t = 2.0$.

N_G	ϵ_{L_2}	$\mathcal{O}(L_2)$	N_G	ϵ_{L_2}	$\mathcal{O}(L_2)$
	$\mathcal{O}3$			$\mathcal{O}4$	
24	2.6916E-02	-	24	1.5993E-02	-
32	1.0906E-02	3.1	32	3.8281E-03	5.0
64	1.9750E-03	2.5	64	3.0900E-04	3.6
128	2.5442E-04	3.0	128	2.0855E-05	3.9
	$\mathcal{O}5$			$\mathcal{O}6$	
24	1.4493E-02	-	24	8.3869E-03	-
32	3.8912E-03	4.6	32	1.9504E-03	5.1
64	2.5564E-04	3.9	64	6.1843E-05	5.0
128	8.7457E-06	4.9	96	7.4509E-06	5.2

6.4.3 Explosion problems

In two space dimensions the initial mesh spacing is of characteristic size $h = 1/250$, leading to a total number of $N_E = 431224$ triangular elements used to discretize $\Omega(t)$, while in 3D we use a characteristic mesh size of $h = 1/100$ for $r \leq r_c$ and $h = 1/50$ for $r > r_c$ for a total number of tetrahedra of $N_E = 2632305$. Numerical results for EP1 and EP3 have been collected using the two-dimensional version of the algorithm, whereas we perform a three-dimensional computation for EP2 and EP4. The numerical results are compared with the 1D reference solution in Figures 6.52 - 6.55. On the top left of each figure a 3D visualization of either the solid or the gas density is shown, in order to verify that either the cylindrical or the spherical symmetry is reasonably maintained on the unstructured meshes used here. The other subfigures show a one-dimensional cut through the reconstructed numerical solution $\boldsymbol{w}_h$ on 100 *equidistant* sample points along the x-axis. We use the path-conservative Osher-type method (2.65) since it is less dissipative than the Rusanov-type scheme (2.62), hence a better resolution of the material contact can be achieved. Since the mesh is moving with the interface velocity $\boldsymbol{u_I}$, i.e. $\boldsymbol{V} = \boldsymbol{u_I} = \boldsymbol{u}_1$, the

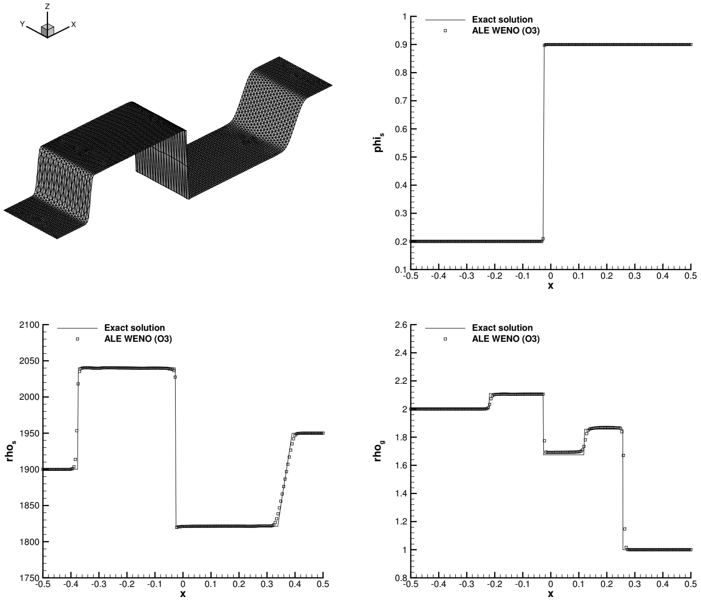


Figure 6.50: Results for Riemann problem RP4 of the seven-equation Baer-Nunziato model in 2D at time $t = 0.15$.

contact discontinuity of the first phase ϕ_1 is very well resolved in all cases. The quality of the three-dimensional results is lower towards the external boundary of the computational domain because a coarser grid with $h = 1/50$ has been used there. This was necessary to reduce the amount of computational resources needed to carry on the computation.

6.4.4 Two-Dimensional Riemann Problems

We use the *two-dimensional* direct ALE ADER-WENO schemes presented in Chapter 2 to perform the multidimensional Riemann problems illustrated in Section 5.4.4. The computational domain is discretized using an unstructured triangular mesh composed of $N_E = 90080$ elements with an initial characteristic

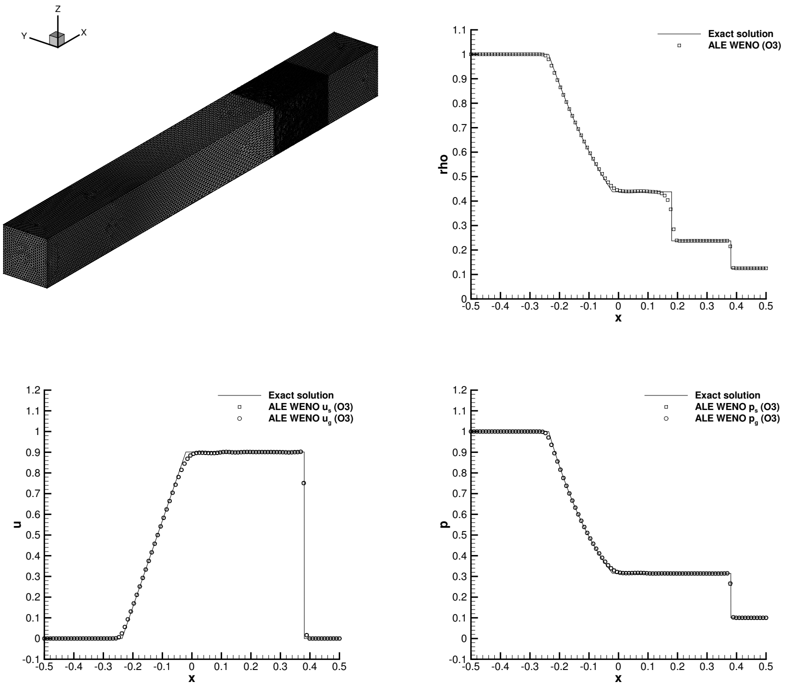


Figure 6.51: Results for Riemann problem RP5 of the seven-equation Baer-Nunziato model in 3D with drag and pressure relaxation ($\lambda = 10^3, \mu = 10^2$) at time $t = 0.2$ and comparison with the exact solution.

mesh spacing of $h = 1/200$. The reference solution is computed with a high order Eulerian one-step scheme as presented in [107,230], using a very fine mesh composed of 2,277,668 triangles with characteristic mesh spacing $h = 1/1000$. The obtained results together with the Eulerian reference solution are depicted in Figures 6.57 - 6.58, where we can observe a very good qualitative agreement of the Lagrangian-like solution with the Eulerian fine-grid reference solution.

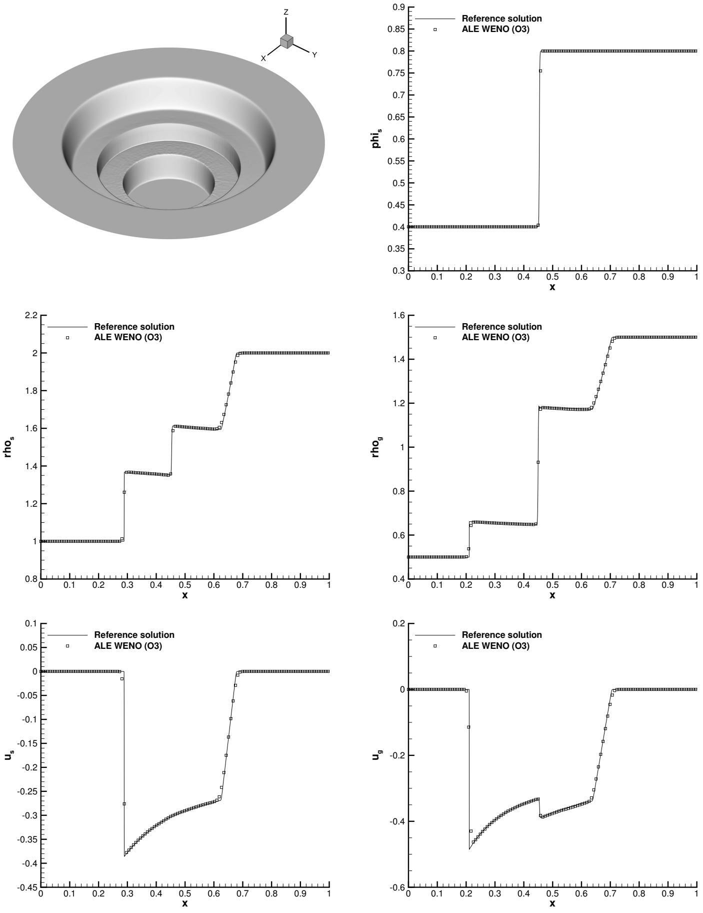


Figure 6.52: Results obtained for the cylindrical explosion problem EP1 of the seven-equation Baer-Nunziato model in 2D at time $t = 0.15$ and comparison with the reference solution.

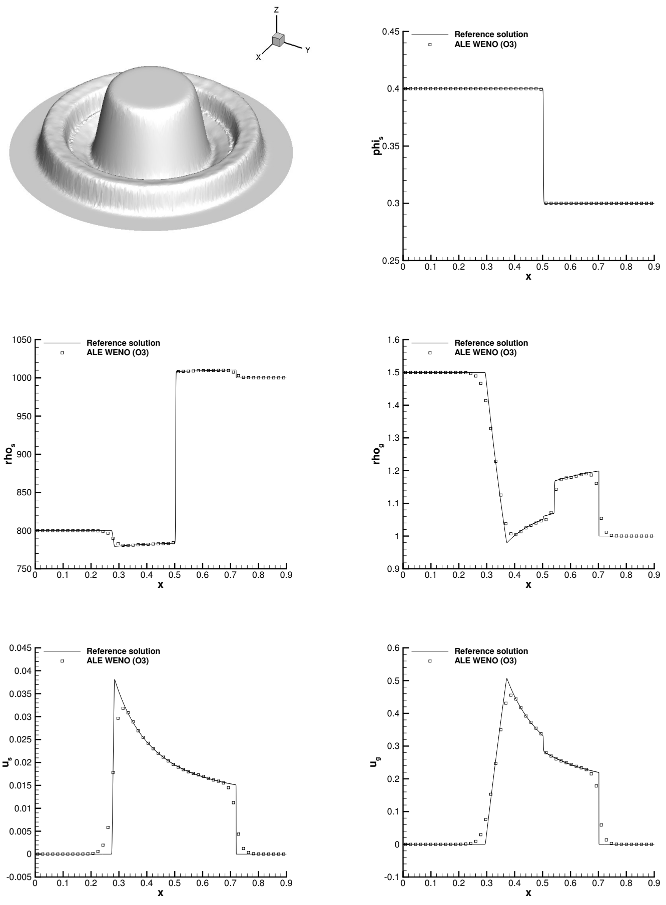


Figure 6.53: Results obtained for the cylindrical explosion problem EP2 of the seven-equation Baer-Nunziato model in 3D at time $t = 0.15$ and comparison with the reference solution.

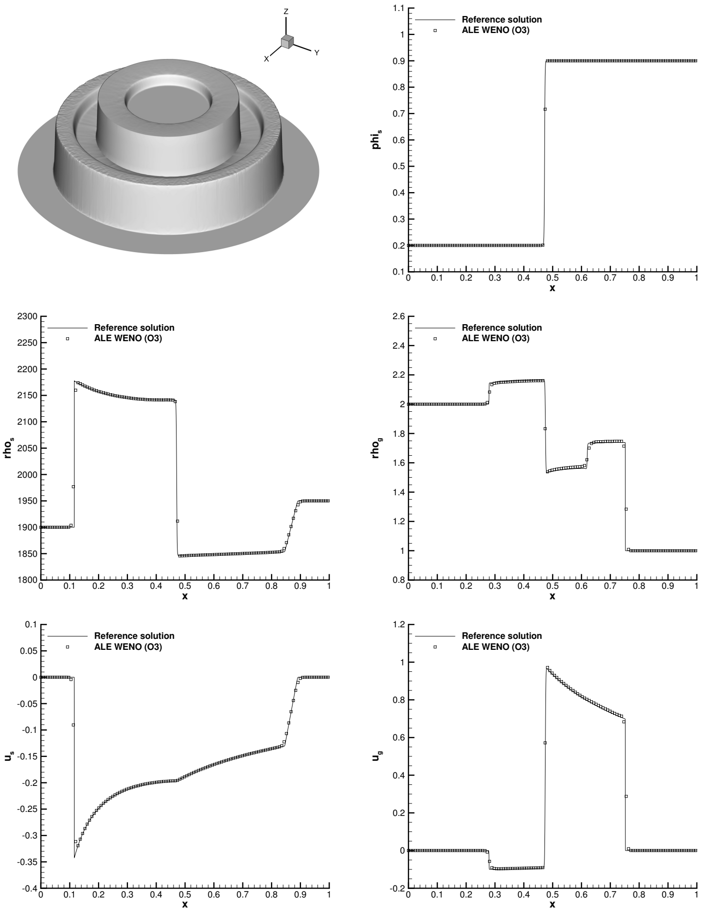


Figure 6.54: Results obtained for the cylindrical explosion problem EP3 of the seven-equation Baer-Nunziato model in 2D at time $t = 0.15$ and comparison with the reference solution.

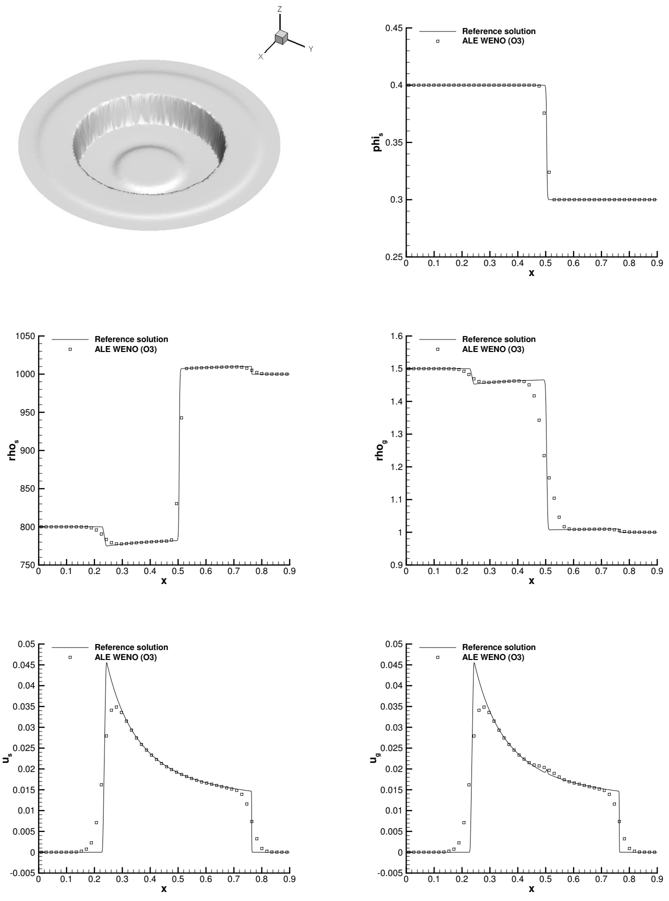


Figure 6.55: Results obtained for the cylindrical explosion problem EP4 of the seven-equation Baer-Nunziato model in 2D with $\lambda = 10^5$ at time $t = 0.18$ and comparison with the reference solution.

For the first test problem, the initial and the final mesh are depicted in Figure 6.56.

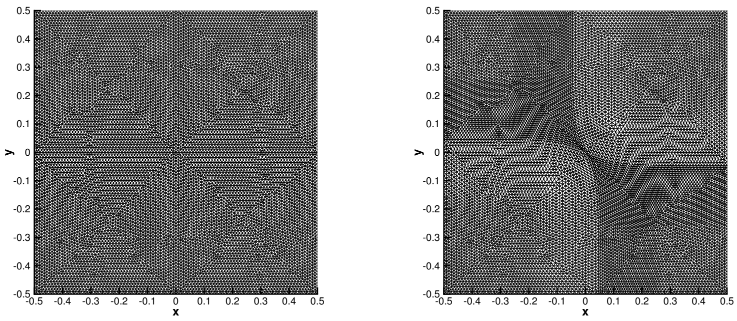


Figure 6.56: Mesh for configuration C1 at times $t = 0$ (left) and $t = 0.15$ (right).

6.4.5 Computational efficiency comparison of Eulerian and ALE schemes

In the previous sections we have shown numerical results obtained with the high order path-conservative direct ALE ADER-WENO schemes presented in Chapter 2, highlighting the excellent resolution of the solid contact wave due to the use of a Lagrangian framework. Such a sharp resolution of the material interface cannot be achieved by a classical Eulerian formulation on fixed meshes without the use of additional techniques, such as the level-set method coupled with the ghostfluid approach [114, 115, 122, 181, 188]. However, the update of the mesh and its associated geometric quantities in time makes Lagrangian algorithms typically much more demanding in terms of computational effort compared to Eulerian schemes. In order to investigate how much the computational efficiency may decrease in high order Lagrangian-like methods, a comparison between our ALE ADER-WENO algorithm and the corresponding Eulerian version (see [99, 102, 103]) is carried out in the following. The numerical test problem is chosen to be the cylindrical explosion problem EP1 and we run the same test case with the ALE and the Eulerian code on the same sequence of successively refined meshes with a total number of elements N_E.

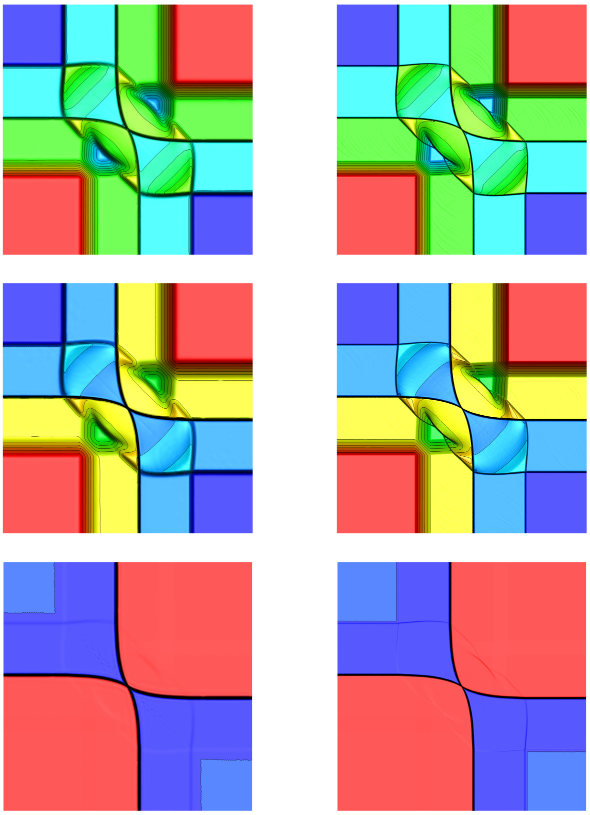


Figure 6.57: Results obtained with the *two-dimensional* third order direct ALE ADER-WENO scheme for the 2D Riemann problem C1 at time $t = 0.15$ (left column). The reference solution computed with an Eulerian method on a very fine mesh is also shown (right column). 30 equidistant contour lines are shown for the solid density ρ_s (top row), the gas density ρ_g (middle row) and the solid volume fraction ϕ_s (bottom row). In this test problem stiff relaxation source terms are used setting $\lambda = 10^5$ and $\mu = 10^2$.

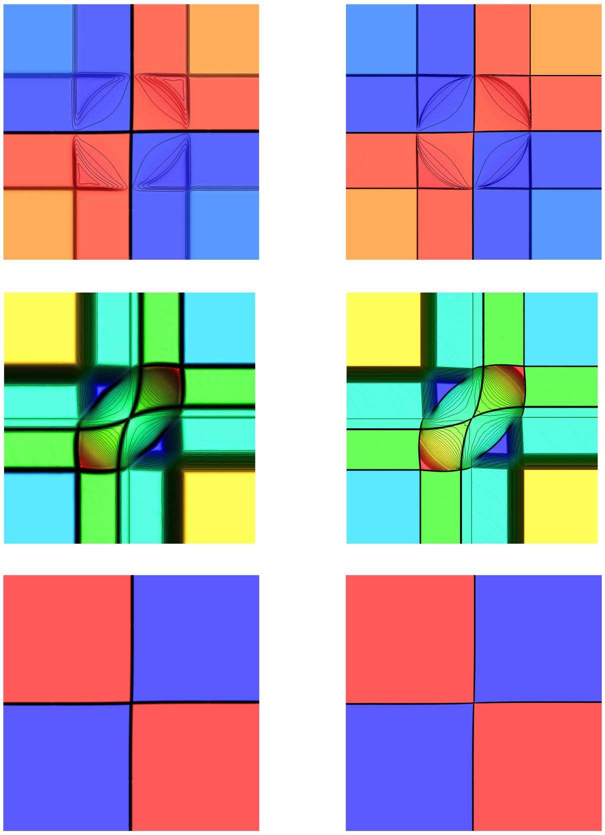
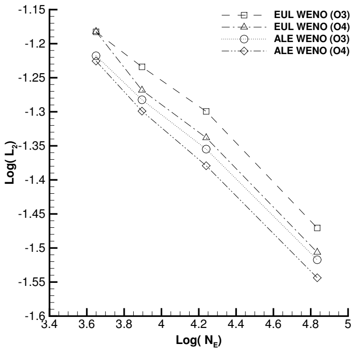


Figure 6.58: Results obtained with the *two-dimensional* third order direct ALE ADER-WENO scheme for the 2D Riemann problem C2 at time $t = 0.15$ (left column). The reference solution computed with an Eulerian method on a very fine mesh is also shown (right column). 30 equidistant contour lines are shown for the solid density ρ_s (top row), the gas density ρ_g (middle row) and the solid volume fraction ϕ_s (bottom row).



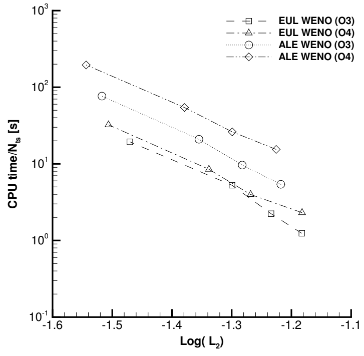


Figure 6.59: Comparison between Lagrangian and Eulerian WENO schemes. Left: error norm versus mesh size (total number of element N_E). Right: CPU time per timestep versus error norm.

Convergence rates and efficiency curves are depicted in Figure 6.59 for a third and a fourth order scheme. Furthermore Table 6.22 contains the error norms ϵ_{L_2} and the CPU time of each run. For a fair comparison, the representative computational time τ_{CPU} (time per element and time step) has been evaluated in microseconds by dividing the total wallclock time t_{CPU} in seconds obtained on a single CPU core (Intel i7-2600 with 3.4 GHz) by the number of timesteps N_{ts} and by the total number of elements N_E, i.e.

$$\tau_{\text{CPU}} = \frac{t_{\text{CPU}}}{N_{ts} \cdot N_E} \cdot 10^6 \tag{6.3}$$

expressed in microseconds (μs). Finally the ratio $\tau^* = \frac{\tau_{\text{CPU,Lag}}}{\tau_{\text{CPU,Eul}}}$ between the two methods shows the significantly larger computational cost of the ALE scheme with respect to the Eulerian scheme. This is mainly due to the fact that in the Lagrangian framework the reconstruction equations (2.9) can *not* be solved once and for all beforehand in a pre-processing step, as in the Eulerian case, but must assembled and solved again for each element in each time step. As a consequence, the ratio τ^* increases with increasing order of accuracy. Since the high order WENO reconstruction step is by far the most expensive part of the algorithm, the use of a high order one-step time integrator like the local space-time Galerkin predictor presented in Section 2.3 is highly recommended. The use of the MOOD paradigm (see Section 4.4) allows the overall algorithm efficiency to be improved, since the cost of the reconstruction drastically decreases because in the MOOD approach we only need one central reconstruction stencil. These results are confirmed by the data reported in Tables 6.16-6.17.

Table 6.22: Comparison between *two-dimensional* ALE and Eulerian WENO algorithm. The compressible Baer-Nunziato model (5.45) has been used to run the cylindrical explosion problem EP1 given in Table 5.3 with third and fourth order of accuracy.

$\mathcal{O}3$									
	Eulerian WENO				*ALE WENO*				
N_E	t_{CPU}	N_{ts}	τ_{CPU}	ϵ_{L_2}	t_{CPU}	N_{t_s}	τ_{CPU}	ϵ_{L_2}	τ^*
4460	33.48	27	278	6.5666E-02	167.67	31	1213	6.0566E-02	4.4
7862	80.50	36	284	5.8334E-02	395.42	41	1227	5.2193E-02	4.3
17366	331.33	63	303	5.0181E-02	1403.76	67	1206	4.4176E-02	4.0
68362	2366.30	122	284	3.3837E-02	9878.44	129	1120	3.0387E-02	3.9

$\mathcal{O}4$									
	Eulerian WENO				*ALE WENO*				
N_E	t_{CPU}	N_{ts}	τ_{CPU}	ϵ_{L_2}	t_{CPU}	N_{t_s}	τ_{CPU}	ϵ_{L_2}	τ^*
4460	62.29	27	517	6.5737E-02	478.27	31	3459	5.9497E-02	6.7
7862	143.26	36	506	5.3914E-02	1098.50	42	3327	5.0234E-02	6.6
17366	535.08	63	489	4.5909E-02	3589.55	66	3132	4.1764E-02	6.4
68362	3942.47	122	473	3.1165E-02	24790.93	127	2855	2.8592E-02	6.0

7 Conclusions and Outlook

In this thesis we have developed a new family of high order Arbitrary-Lagrangian-Eulerian one-step ADER-WENO finite volume schemes on unstructured triangular and tetrahedral meshes [39, 40, 96]. The algorithm is formulated in a very general manner so that it can be applied to both conservative and non-conservative hyperbolic systems of balance laws, with and without stiff source terms. A WENO reconstruction technique is used to achieve high order of accuracy in space, while an element-local space-time Galerkin finite element predictor on moving curved meshes is used to obtain a high order accurate one-step time discretization. To the knowledge of the author, this is the first better than second order accurate Lagrangian-type finite volume scheme ever presented on unstructured tetrahedral meshes. The final ALE finite volume scheme belongs to the category of *direct* ALE methods, because an additional remapping stage, which is typically used in the context of indirect ALE and pure Lagrangian schemes, is unnecessary in our case. This is possible because the new class of ALE algorithms proposed within this thesis is based directly on a space-time conservation formulation of the governing PDE system, which furthermore allows the geometric conservation law (GCL) to be satisfied by the scheme by construction (see Appendix C for details).

The mesh motion procedure has been described in details, considering all the steps needed to move the mesh, namely the Lagrangian step, the rezoning step and the relaxation step. In the Lagrangian step three different *node solvers* have been considered for the computation of the node velocity in order to move the grid in time [41]. The first solver $\mathcal{NS}_{cs}$ is simply defined as the mass weighted average of the states in the cells surrounding the node, while the solver $\mathcal{NS}_m$ has been introduced in our scheme for hydrodynamics and for the classical MHD equations. Finally, we have used the recent multidimensional HLL Riemann solver [88] as a new node solver type for the first time in the ALE context.

In order to improve the overall algorithm efficiency several variants of the original scheme have been presented.

First, we have developed a novel cell-centered ALE ADER-WENO method that allows for *time-accurate local time stepping* (LTS) algorithm [43]. It applies to unstructured triangular meshes uses the following basic ingredients: a high order WENO reconstruction in space on unstructured meshes, an element-local high-order accurate space-time Galerkin predictor that performs the time evolution of the reconstructed polynomials within each element, the computation of numerical ALE fluxes at the moving element interfaces through approximate Riemann solvers, and a one-step finite volume scheme for the time update which is directly based on the integral form of the conservation equations in space-time. The inclusion of the LTS algorithm requires a number of crucial extensions, such as a proper scheduling criterion for the time update of each element and for each node; a virtual projection of the elements contained in the reconstruction stencils of the element that has to perform the WENO reconstruction; and the proper computation of the fluxes through the space-time boundary surfaces that will inevitably contain hanging nodes in time due to the LTS algorithm.

Then, the first two-dimensional high-order unstructured ALE ADER-WENO finite volume schemes based on *genuinely multidimensional HLL Riemann solvers* [38] have been considered because the multidimensionality in the Riemann solver allows the numerical method to run with a less severe CFL condition on the timestep, namely taking CFL $= 0.95$ in two space dimensions, instead of setting it to the usual limit of CFL ≤ 0.5 typical for unsplit Godunov schemes in two space dimensions using conventional one-dimensional Riemann solvers.

We also focused on the numerical flux integration, proposing a new and efficient *quadrature-free* way to compute the space-time integrals of the numerical fluxes within the framework of high order direct ALE ADER-WENO finite volume schemes on unstructured triangular and tetrahedral meshes [44]. The boundaries of the Lagrangian space-time control volumes are divided into smaller *simplex* sub-elements, which allow an analytical integration to be carried out only once and for all in a pre processing step. This yields a big reduction of the computational time needed to carry out numerical simulations.

Finally, we have applied the novel *a posteriori* Multi-dimensional Optimal Order Detection (MOOD) approach [69, 79, 81, 165] to the context of direct Arbitrary-Lagrangian-Eulerian (ALE) ADER schemes solving the multidimensional hydrodynamics equations. Starting from the unlimited ALE ADER-WENO scheme previously designed in Chapter 2, we have replaced the *a priori* WENO polynomial limiting technique by the MOOD paradigm [45]. These ALE ADER-MOOD schemes have been implemented to solve the Euler equations for compressible gas dynamics on unstructured grids, but the scheme may in principle apply also to non-conservative hyperbolic systems.

For each version of the algorithm high order numerical convergence studies in space and time have been carried out, hence showing the accuracy of the new family of high order ALE ADER finite volume schemes. In Chapter 5 we have considered different hyperbolic systems of conservation laws, namely the multidimensional Euler equations of compressible gas dynamics, the ideal classical and relativistic magneto-hydrodynamics (MHD) equations and the non-conservative seven-equation Baer-Nunziato model of compressible multi-phase flows with stiff relaxation source terms. Several classical test problems have been run for each system of PDEs in order to assess the robustness of the new schemes. The obtained numerical results have been carefully compared with exact or other numerical reference solutions and some of them have been presented in Chapter 6.
Further work will regard the extension of the presented direct ALE ADER-WENO finite volume schemes to the more general framework of the new P_NP_M method proposed in [95], which can deal with either pure finite volume or pure discontinuous Galerkin finite element methods, or with a hybridization of both. This will allow the mesh motion to be described by *curvilinear* trajectories, instead of limiting us to use straight lines as done within this work. Therefore material interfaces and contact waves will be located even more precisely by using curvilinear unstructured meshes and the quality of the numerical results will improve even more.
We also plan to investigate *pure* Lagrangian algorithms, i.e. numerical methods with zero mass flux across element interfaces, in the context of ADER schemes. The use of curvilinear meshes will allow the first better than second order pure Lagrangian finite volume schemes to be developed on unstructured meshes, using the already existing framework presented in this thesis work.
Last but not least, another important topic will be the application of the present scheme to more realistic real world simulations in engineering and physics.

A Basis Functions

In this appendix we provide the information needed to compute the basis functions used in the direct ALE ADER-WENO algorithm. For the reconstruction technique, illustrated in Section 2.2 of Chapter 2, the basis functions ψ_l depend only on space and constitute a *modal basis*, while in the local predictor step, described in Section 2.3 we use the space-time basis functions θ_l that lead to a *nodal approach*. The details for the computation of both ψ_l and θ_l are presented in the following Sections A.1 and A.2, respectively.

A.1 Space basis functions

The basis functions ψ_l constitute an orthogonal hierarchical tensor-type basis and they are evaluated by extending the original idea of Dubiner [90], as fully explained in [150]. The basis is a combination of structured and unstructured domains consisting in *polymorphic subdomains* and it can be applied to both 2D and 3D subdomains, i.e. simplicial elements like triangles and tetrahedra as used within our ALE algorithm. For each subdomain, defined in the reference system $\boldsymbol{\xi}$, the polynomial expansion is based upon a new local coordinate system $\boldsymbol{\eta}$.
The basis functions are given in terms of the Jacobi polynomials $P_n^{\alpha,\beta}(z)$ on the interval $[-1;1]$ by

$$P_n^{\alpha,\beta}(z) = \frac{(-1)^n}{2^n n!}(1-z)^{-\alpha}(1+z)^{-\beta}\frac{d^n}{dz^n}\left[(1-z)^{\alpha+n}(1+z)^{\beta+n}\right], \quad \text{(A.1)}$$

which is the solution of the Jacobi differential equation

$$\left(1-z^2\right)y'' + \left[\beta-\alpha-(\alpha+\beta+2)z\right]y' + n(n+\alpha+\beta+1)y = 0. \quad \text{(A.2)}$$

According to [150] we set $\alpha=\beta=0$, so that the Jacobi polynomials $P_n^{0,0}(z)$ reduce to the Legendre polynomials. Then, we define three principle functions $\Theta_i^a(z)$, $\Theta_{ij}^b(z)$ and $\Theta_{ijk}^c(z)$ as

$$\Theta_i^a(z) = P_i^{0,0}(z), \qquad \Theta_{ij}^b(z) = \left(\frac{1-z}{2}\right)^i P_j^{2i+1,0}(z),$$

$$\Theta^c_{ijk}(z) = \left(\frac{1-z}{2}\right)^{i+j} P_k^{2i+2j+2,0}(z), \tag{A.3}$$

and we use them to construct the polynomial expansions, which are given as a product of the principle functions. The expansions have the property to be orthogonal in the Legendre inner product.

A.1.1 Triangles

The reference triangle T_E is defined in the two-dimensional reference system $\boldsymbol{\xi} = (\xi_1, \xi_2)$ as

$$T_E = \left\{(\xi_1, \xi_2) \in \mathbb{R}^2 \mid 0 \le \xi_1 \le 1 \ \wedge \ 0 \le \xi_2 \le 1 - \xi_1\right\}. \tag{A.4}$$

The triangular expansion reads

$$\psi_l\left(\xi_1, \xi_2\right) = \Theta^a_p\left(\eta_1\right) \cdot \Theta^b_{pq}\left(2\eta_2 - 1\right) \tag{A.5}$$

with the local coordinates

$$\eta_1 = \frac{2\xi_1}{1-\xi_2} - 1 \quad \text{and} \quad \eta_2 = \xi_2. \tag{A.6}$$

For a complete polynomial basis of order N we have

$$0 \le p \le N \quad \wedge \quad 0 \le q \le N \quad \wedge \quad (p+q) \le N. \tag{A.7}$$

The mono-index l in (A.5) is function of the indexes (p, q) and is bounded in the interval $[0, L-1]$, where L represents the numbers of degrees of freedom. For the two-dimensional case it can be evaluated as

$$L = \frac{(N+1)(N+2)}{2}, \tag{A.8}$$

hence yielding

$$0 \le l \le \frac{(N+1)(N+2)}{2} - 1. \tag{A.9}$$

A.1.2 Tetrahedra

The reference tetrahedron T_E is defined in the three-dimensional reference system $\boldsymbol{\xi} = (\xi_1, \xi_2, , \xi_3)$ as

$$T_E = \left\{ (\xi_1, \xi_2, \xi_3) \in \mathbb{R}^3 \mid 0 \le \xi_1 \le 1 \ \wedge \ 0 \le \xi_2 \le 1 - \xi_1 \ \wedge \ 0 \le \xi_3 \le 1 - \xi_1 - \xi_2 \right\}. \tag{A.10}$$

The polynomial expansion for the tetrahedron element is given by

$$\psi_l \left(\xi_1, \xi_2 \right) = \Theta_p^a \left(\eta_1 \right) \cdot \Theta_{pq}^b \left(\eta_2 \right) \cdot \Theta_{pqr}^c \left(\eta_3 \right) \tag{A.11}$$

with the local coordinates

$$\eta_1 = \frac{2 \left(1 + \xi_1 \right)}{-\xi_2 - \xi_3}, \quad \eta_2 = \frac{2 \left(1 + \xi_2 \right)}{1 - \xi_3} - 1 \quad \text{and} \quad \eta_3 = \xi_3. \tag{A.12}$$

For a complete polynomial basis of order N for the tetrahedron expansion (A.11) we have

$$0 \le p \le N \quad \wedge \quad 0 \le q \le N \quad \wedge \quad 0 \le r \le N \quad \wedge \quad (p + q + r) \le N. \tag{A.13}$$

As for triangles, the mono-index l in (A.11) is function of the indexes (p, q, r) and it ranges from 0 to $L - 1$ as maximum value, with L denoting the numbers of degrees of freedom. In three space dimensions it is given by

$$L = \frac{(N+1)(N+2)(N+3)}{6}, \tag{A.14}$$

according to expression (2.6) introduced in Section 2.2.

A.2 Space-time basis functions

The basis functions θ_l are given by the Lagrange interpolation polynomials that are passing through a set of space-time nodes $\tilde{\boldsymbol{\xi}}_m$, leading to the use of a *nodal basis*. Therefore the explicit formula for the basis θ_l simply reads

$$\theta_l \left(\tilde{\boldsymbol{\xi}} \right) = \frac{\prod\limits_{l \neq k} \tilde{\boldsymbol{\xi}} - \tilde{\boldsymbol{\xi}}_k}{\prod\limits_{l \neq k} \tilde{\boldsymbol{\xi}}_l - \tilde{\boldsymbol{\xi}}_k}, \qquad \forall l, k \in [1, L], \tag{A.15}$$

where L denotes the number of degrees of freedom which depend on the order N. In the following we will detail how the space-time nodes $\tilde{\boldsymbol{\xi}}_m$ are chosen and computed. We underline that the Lagrange interpolation polynomials satisfy by construction the interpolation property given by (2.14), as required by the nodal formulation of the basis.

Local Continuous Galerkin (CG) predictor. The space-time nodes are defined according to [94], where special care has been taken in order to use only the minimal number of space-time nodes necessary to reach the formal order of accuracy. According to [94], the optimal number of degrees of freedom L for a polynomial of degree N is

$$L(d,N) = \frac{1}{(d+1)!} \prod_{j=1}^{d+1} N + j, \tag{A.16}$$

with d representing the number of space dimensions. The degrees of freedom are then located for the reference triangle and the reference tetrahedron at the space-time points defined by

$$\begin{aligned} \tilde{\boldsymbol{\xi}} &= (\xi_{qrs}, \eta_{qrs}, \tau_{qrs}) = \left(\frac{q}{N-s}, \frac{r}{N-s}, \frac{s}{N} \right) \text{ in } \mathbb{R}^2, \\ \tilde{\boldsymbol{\xi}} &= (\xi_{pqrs}, \eta_{pqrs}, \zeta_{pqrs}, \tau_{pqrs}) = \left(\frac{p}{N-s}, \frac{q}{N-s}, \frac{r}{N-s}, \frac{s}{N} \right) \text{ in } \mathbb{R}^3, \end{aligned} \tag{A.17}$$

with

$$0 \le s \le N, \quad 0 \le r \le (N-s), \quad 0 \le q \le (N-s-r), \quad 0 \le p \le (N-s-r-q). \tag{A.18}$$

The last degree of freedom, which corresponds to the singular case $s = N$, is located at the spatial barycenter of the reference element T_E at the new time level $\tau = 1$, hence

$$(\boldsymbol{\xi}_{0..0N}, \tau_{0..0N}) = \left(\frac{1}{d+1}, ..., \frac{1}{d+1}, 1 \right). \tag{A.19}$$

Local Discontinuous Galerkin (DG) predictor. In this case the space-time nodes are given by the tensor product of the spatial nodes of classical conforming high order finite elements and the Gauss-Legendre quadrature points in time. More precisely, with the multi-index $m = (m_1, m_2, m_3, m_4)$ and $1 \le m_i \le N+1$, the space-time nodes for the three-dimensional case are given by

$$\tilde{\boldsymbol{\xi}}_m = \left(\frac{(m_1-1)}{N}, \frac{(m_2-1)}{N}, \frac{(m_3-1)}{N}, \tau_{m_4} \right),$$

where τ_{m_4} is the m_4-th root of the Legendre polynomial of degree $N+1$, rescaled to the unit interval $[0;1]$. If $d=2$ we can derive the associated space-time coordinates simply setting $\boldsymbol{\xi}(m_3)=0$.
The Gauss-Legendre quadrature points are defined by the roots of the associated Legendre polynomials of order $N+1$, which are defined in the interval $[-1,1]$, and using Rodrigues formula they read

$$P_n(z) = \frac{1}{2^n n!}\frac{d^n}{dx^n}\left[\left(z^2-1\right)^n\right]. \tag{A.20}$$

The above expression is the solution of the Legendre differential equation in the special case when index n is an integer number:

$$\frac{d}{dz}\left[\left(1-z^2\right)\frac{d}{dx}P_n(z)\right] + n(n+1)P_n(z) = 0. \tag{A.21}$$

Since the Gauss-Legendre quadrature points are used along the reference time coordinate $\tau \in [0,1]$, we use the *shifted* Legendre polynomials $\tilde{P}_n(z)$ which can be obtained considering the shifting function $z \to 2z+1$:

$$\tilde{P}_n(z) = P_n(2z+1). \tag{A.22}$$

Thus, the corresponding Rodrigues formula reads

$$\tilde{P}_n(z) = \frac{1}{n!}\frac{d^n}{dx^n}\left[\left(z^2-z\right)^n\right]. \tag{A.23}$$

Recall that with a total number of N_G Gauss-Legendre quadrature points, an order of accuracy of $\mathcal{O} = 2N_G+1$ is retrieved, see [218] for details.

B Positivity preserving technique

Several phenomena in physics and engineering as well as many classical benchmark test cases in the Lagrangian framework are involving strong shock waves, which may lead to a loss of positivity for density and pressure in the numerical scheme. Such a problem typically occurs after carrying out the high-order reconstruction algorithm presented in Section 2.2 of Chapter 2, which is designed to be *essentially* but not *absolutely* non-oscillatory. For this reason we rely on the positivity preserving technique of Balsara [24], where a flattener variable is computed in order to smear out the oscillations and to bring back density and pressure values to their physically admissible range if the positivity constraint has been violated. In [24] the equations for both hydrodynamics and magnetohydrodynamics have been considered on two- and three-dimensional Cartesian grids and in this paper we extend the method to moving unstructured tetrahedral meshes.
First we have to detect those regions of the computational domain $\Omega(t)$ which are characterized by strong shocks. Let us consider a tetrahedron T_i^n and its Neumann neighborhood $\mathcal{N}_i$, i.e. all the elements T_j^n that are attached to a face of T_i^n. Let furthermore $\boldsymbol{Q}_i^n$ and $\boldsymbol{Q}_j^n$ be the vectors of conserved variables of element T_i^n and its direct neighbor T_j^n, respectively, and let ρ^n denote the density and p^n the pressure. A shock can be identified by comparing the divergence of the velocity field $\nabla \cdot \mathbf{v}^n$ with the minimum of the sound speed $c_{i,\min}^n$ obtained by considering the element T_i^n itself as well as its neighborhood $\mathcal{N}_i$. Hence,

$$\nabla \cdot \mathbf{v}^n = \frac{1}{|T_i^n|} \sum_{T_j^n \in \mathcal{N}_i} S_j^n \left(\mathbf{v}_j^n - \mathbf{v}_i^n \right) \cdot \boldsymbol{n}_{ij}^n, \qquad c_{i,\min}^n = \min_{T_j^n \in \mathcal{N}_i} \left(c_i^n, c_j^n \right), \tag{B.1}$$

where $|T_i^n|$ represents as usual the volume of the tetrahedron T_i^n, S_j^n denotes the surface shared between element T_i^n and the neighbor T_j^n, $\boldsymbol{n}_{ij}^n$ is the associated unit normal vector w.r.t. the surface S_j^n and $c_{i,j}^n = \sqrt{\frac{\gamma p_{i,j}^n}{\rho_{i,j}^n}}$ are the sound speeds of T_i^n and the neighbor element T_j^n, respectively, with γ representing the ratio of specific heats. The divergence of the velocity field is estimated from

the cell-averaged states $\boldsymbol{Q}_{i,j}^n$ and *not* from the reconstructed states $\mathbf{w}_h(\mathbf{x}, t^n)$ obtained from (2.13).
The flattener variable f_i^n is then computed according to [24] as

$$f_i^n = \min\left[1, \max\left(0, -\frac{\nabla \cdot \mathbf{v}^n + k_1 c_{i,\min}^n}{k_1 c_{i,\min}^n}\right)\right], \tag{B.2}$$

with the coefficient k_1 that is set to the value of $k_1 = 0.1$ for all our computations. For rarefaction waves the divergence of the velocity field is positive, i.e. $\nabla \cdot \mathbf{v}^n \geq 0$, hence obtaining $f_i^n = 0$ and leaving the reconstruction polynomial $\mathbf{w}_h(\mathbf{x}, t^n)$ as it is. Even when shocks of modest strength occur, i.e. $-k_1 c_{i,\min}^n \leq \nabla \cdot \mathbf{v}^n \leq 0$, the reconstruction remains untouched.
In the work of Balsara [24] the flattener variable is propagated even to those elements that are about to be crossed by a shock, but have still to enter the wave, i.e. the neighbors of an element which has already experienced the shock. Due to the more complex computational domain on unstructured meshes, we propose to define a *node based* flattener $\tilde{f}_k^n$: the Voronoi neighborhood $\mathcal{V}_k$ of each vertex k of element T_i^n is also considered in order to propagate the flattener, hence taking into account all those elements that share vertex k of tetrahedron T_i^n. The node based flattener results in the maximum value among the flattener values f_j^n of the attached tetrahedra that has been previously computed according to (B.2):

$$\tilde{f}_k^n = \max_{j \in \mathcal{V}_k} f_j^n. \tag{B.3}$$

Each element T_i^n is then assigned again with the maximum value of the node based flattener among the set $\mathcal{K}_i$ of the four vertices that define the tetrahedron T_i^n, i.e.

$$f_i^n = \max_{k \in \mathcal{K}_i} \tilde{f}_k^n. \tag{B.4}$$

Once the flattener variable has been computed for each element of the computational domain $\Omega(t)$, the WENO reconstruction polynomials are corrected with the following expression:

$$\mathbf{w}_h(\mathbf{x}, t^n) := (1 - f_i^n)\psi_l(\boldsymbol{\xi})\hat{\mathbf{w}}_{l,i}^n + f_i^n \cdot \boldsymbol{Q}_i^n. \tag{B.5}$$

If positivity is still violated even after using (B.5), then $f_i^n := 1$ is set, thus recovering a (positivity preserving) first order finite volume scheme. This strategy resembles to some extent the recently developed MOOD algorithm of Diot et al. [70, 80, 165], however, it is still used as an *a priori* limiter here, while the MOOD approach uses an innovative *a posteriori* limiting philosophy.

Indeed, with the MOOD version of the direct ALE ADER-WENO algorithm, presented in Section 4.4 of Chapter 4, the preserving technique described in this appendix is *no longer* needed. The positivity preservation falls in fact in the Physical Admissible Criteria (PAD) of the MOOD loop, hence it will be always check *a posteriori* for *each* control volume. The presented flattener technique is by default switched off and has been used only for those test problems where it was absolutely necessary in order to run the simulation to the final time.

C Geometric Conservation Law (GCL)

In the literature about classical pure Lagrangian schemes the geometric conservation law (GCL) is usually written in its *semi discrete* form as follows, see for example [172]:

$$\frac{d}{dt}\int\limits_{T_i(t)} dV - \int\limits_{\partial T_i(t)} \boldsymbol{V}\cdot\boldsymbol{n}\,dS = 0, \tag{C.1}$$

where $\boldsymbol{V} = (U, V, W)$ denotes the mesh velocity and $\boldsymbol{n}$ is the outward-pointing spatial unit normal vector $\boldsymbol{n} = (n_x, n_y, n_z)^T$. The corresponding *fully discrete* form can be obtained by integrating (C.1) in time, which yields

$$|T_i^{n+1}| - |T_i^n| - \int\limits_{t^n}^{t^{n+1}}\int\limits_{\partial T_i(t)} \boldsymbol{V}\cdot\boldsymbol{n}\,dSdt = 0, \tag{C.2}$$

where $|T_i^n|$ denotes the volume of element T_i at time t^n. Each face ∂T_{ij} of the tetrahedron $T_i(t)$ is now parametrized using a mapping of the type $\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{\chi})$ with the coordinate vector $\boldsymbol{x} = (x, y, z)^T$ and the vector of parameters $\boldsymbol{\chi} = (\chi_1, \chi_2)^T$ with $0 \le \chi_1 \le 1 - \chi_2$ and $0 \le \chi_2 \le 1$. The normal vector $\boldsymbol{n}$ multiplied with the area differential dS is denoted by $\boldsymbol{dn} = \boldsymbol{n}dS$ and reads with $\boldsymbol{e_x} = (1,0,0)^T$, $\boldsymbol{e_y} = (0,1,0)^T$ and $\boldsymbol{e_z} = (0,0,1)^T$ for each face ∂T_{ij} as

$$\boldsymbol{dn} = \begin{vmatrix} \boldsymbol{e_x} & x_{\chi_1} & x_{\chi_2} \\ \boldsymbol{e_y} & y_{\chi_1} & y_{\chi_2} \\ \boldsymbol{e_z} & z_{\chi_1} & z_{\chi_2} \end{vmatrix} d\chi_1 d\chi_2 = \begin{pmatrix} y_{\chi_1} z_{\chi_2} - z_{\chi_1} y_{\chi_2} \\ -(x_{\chi_1} z_{\chi_2} - z_{\chi_1} x_{\chi_2}) \\ x_{\chi_1} y_{\chi_2} - x_{\chi_1} y_{\chi_2} \end{pmatrix} d\chi_1 d\chi_2. \tag{C.3}$$

We furthermore parametrize the physical time as $t = t^n + \tau\Delta t$, hence, the fully discrete form of the GCL (C.2) becomes

$$|T_i^{n+1}| - |T_i^n| - \sum_{\partial T_{ij}} \int\limits_0^1 \int\limits_0^1 \int\limits_0^{1-\chi_2} \Delta t\, \boldsymbol{V}\cdot\boldsymbol{dn}\, d\tau = 0. \tag{C.4}$$

Note also that

$$\begin{aligned}\boldsymbol{V}\cdot\boldsymbol{dn} &= U\left(y_{\chi_1}z_{\chi_2}-z_{\chi_1}y_{\chi_2}\right)d\chi_1 d\chi_2\\ &- V\left(x_{\chi_1}z_{\chi_2}-z_{\chi_1}x_{\chi_2}\right)d\chi_1 d\chi_2\\ &+ W\left(x_{\chi_1}y_{\chi_2}-x_{\chi_1}y_{\chi_2}\right)d\chi_1 d\chi_2.\end{aligned} \tag{C.5}$$

We now prove that Eqn. (C.3) is equivalent to the *fourth component* (i.e. the time component) of the vector equation (2.56), which reads

$$\int_{\partial C_i^n}\tilde{\boldsymbol{n}}dS=\int_{\partial C_i^n}\boldsymbol{d\tilde{n}}=0, \tag{C.6}$$

with the outward-pointing space-time normal vector $\boldsymbol{d\tilde{n}}=(dn_x,dn_y,dn_z,dn_t)^T$. The bounding surface $\partial\mathcal{C}_i^n$ of the space-time control volume is delimited by the two space-time faces T_i^n (tetrahedron at the initial time t^n) and T_i^{n+1} (element at the final time t^{n+1} of a time step) and by the four lateral space-time sub-surfaces ∂C_{ij}^n, see Eqn. (2.50). The associated space-time normal vectors for the first two space-time sub-surfaces are $\tilde{\boldsymbol{n}}=(0,0,0,-1)^T$ and $\tilde{\boldsymbol{n}}=(0,0,0,+1)^T$, respectively. The space-time unit vectors along the physical coordinate axis are $\tilde{\boldsymbol{e}}_{\boldsymbol{x}}=(1,0,0,0)^T$, $\tilde{\boldsymbol{e}}_{\boldsymbol{y}}=(0,1,0,0)^T$, $\tilde{\boldsymbol{e}}_{\boldsymbol{z}}=(0,0,1,0)^T$ and $\tilde{\boldsymbol{e}}_{\boldsymbol{t}}=(0,0,0,1)^T$, therefore Eqn. (C.6), which is the same as Eqn. (2.56), can also be written as

$$\int_{T_i^{n+1}}\tilde{\mathbf{e}}_t dV-\int_{T_i^n}\tilde{\mathbf{e}}_t dV+\sum_j\int_{\partial C_{ij}^n}\boldsymbol{d\tilde{n}}=0. \tag{C.7}$$

The parametrization of each lateral space-time sub-face ∂C_{ij}^n is of the type $\tilde{\boldsymbol{x}}=\tilde{\boldsymbol{x}}(\tilde{\boldsymbol{\chi}})$ with the Cartesian space-time coordinate vector $\tilde{\boldsymbol{x}}=(x,y,z,t)^T$ and the vector of parameters $\tilde{\boldsymbol{\chi}}=(\chi_1,\chi_2,\tau)^T$ with $0\le\chi_1\le 1-\chi_2$ and $0\le\chi_2\le 1$ and $0\le\tau\le 1$. We still use $t=t^n+\tau\Delta t$, hence $t_{\chi_1}=t_{\chi_2}=0$. Furthermore, the mesh velocity vector is given by

$$\boldsymbol{V}=\frac{1}{\Delta t}\left(\begin{array}{c}x_\tau\\ y_\tau\\ z_\tau\end{array}\right). \tag{C.8}$$

The vector $\boldsymbol{d\tilde{n}} = \boldsymbol{\tilde{n}} dS$ for the lateral space-time sub-surfaces reads

$$\boldsymbol{d\tilde{n}} = \begin{pmatrix} d\tilde{n}_x \\ d\tilde{n}_y \\ d\tilde{n}_z \\ d\tilde{n}_t \end{pmatrix} = \begin{vmatrix} \boldsymbol{\tilde{e}_x} & x_{\chi 1} & x_{\chi 2} & x_\tau \\ \boldsymbol{\tilde{e}_y} & y_{\chi 1} & y_{\chi 2} & y_\tau \\ \boldsymbol{\tilde{e}_z} & z_{\chi 1} & z_{\chi 2} & z_\tau \\ \boldsymbol{\tilde{e}_t} & t_{\chi 1} & t_{\chi 2} & t_\tau \end{vmatrix} d\chi_1 d\chi_2 d\tau$$
$$= \begin{pmatrix} \left(y_{\chi 1} z_{\chi 2} - z_{\chi 1} y_{\chi 2}\right) \Delta t \\ -\left(x_{\chi 1} z_{\chi 2} - z_{\chi 1} x_{\chi 2}\right) \Delta t \\ \left(x_{\chi 1} y_{\chi 2} - x_{\chi 1} y_{\chi 2}\right) \Delta t \\ \tilde{n}_t \end{pmatrix} d\chi_1 d\chi_2 d\tau, \quad \text{(C.9)}$$

with

$$\tilde{n}_t = -x_\tau \left(y_{\chi 1} z_{\chi 2} - z_{\chi 1} y_{\chi 2}\right) + y_\tau \left(x_{\chi 1} z_{\chi 2} - z_{\chi 1} x_{\chi 2}\right) - z_\tau \left(x_{\chi 1} y_{\chi 2} - x_{\chi 1} y_{\chi 2}\right). \quad \text{(C.10)}$$

Using (C.10), (C.9), (C.8) and (C.5) one finds that

$$d\tilde{n}_t = -\Delta t\, \boldsymbol{V} \cdot \boldsymbol{dn}\, d\tau. \quad \text{(C.11)}$$

Therefore, the fourth component of equation (2.56) reads

$$|T_i^{n+1}| - |T_i^n| - \sum_{\partial T_{ij}} \int_0^1 \int_0^1 \int_0^{1-\chi_2} \Delta t\, \boldsymbol{V} \cdot \boldsymbol{dn}\, d\tau = 0, \quad \text{(C.12)}$$

which is therefore *identical* with the fully discrete form of the usual geometric conservation law, see Eqn. (C.2).

D Boundary conditions

In the following we briefly describe the type of boundary conditions used within the high order ALE ADER-WENO framework to carry on the numerical simulations of the test problems presented in Chapter 5. Unlike in the context of finite element or discontinuous Galerkin methods, where boundary conditions can be imposed in a very rigorous and simple way just by choosing a suitable numerical flux at the domain boundary, for finite volume schemes the setting of boundary conditions is not trivial and in some case it might become cumbersome. This is mainly due to the fact that for high order finite volume methods reconstruction is necessary, hence yielding the need to introduce ghost cells for the boundary treatment that allows the numerical flux at the domain boundary to be computed consistently with the order of accuracy of the reconstruction, thus of the entire scheme.
From the practical viewpoint of implementation, the boundary conditions setting means that we have to assign a suitable *boundary state* $\boldsymbol{Q}_g$ for the ghost neighbor T_g of element T_i, which lies on boundary and is given the state $\boldsymbol{Q}_i$. The entire range of boundary condition types that may be imposed for all test cases considered in this work (see Chapters 5 - 6) are the following:

- *transmissive* boundary conditions are adopted to let the fluid flow across the domain boundary. The flow is governed by the internal state, hence yielding the simple setting $\boldsymbol{Q}_g = \boldsymbol{Q}_i$;

- *wall* (or reflective) boundary conditions are used for the treatment of wall boundaries. In this case the orthogonal flux across the domain boundary is zero, therefore we first set $\boldsymbol{Q}_g = \boldsymbol{Q}_i$ and then the velocity vector $\boldsymbol{v}_g$ for the boundary state is computed as

$$\boldsymbol{v}_g = \boldsymbol{v}_i - 2\,(\boldsymbol{v}_i \cdot \boldsymbol{n})\,\boldsymbol{n}, \tag{D.1}$$

  where $\boldsymbol{n}$ denotes as usual the outward pointing unit normal vector on the boundary edge of element T_i and $\boldsymbol{v}_i$ represents the velocity vector given by the internal state $\boldsymbol{Q}_i$, according to Figure D.1. This treatment is also called *no-slip wall* boundary condition and for *inviscid* flows the fluid is still allowed to flow along the boundary, i.e. tangential to the boundary

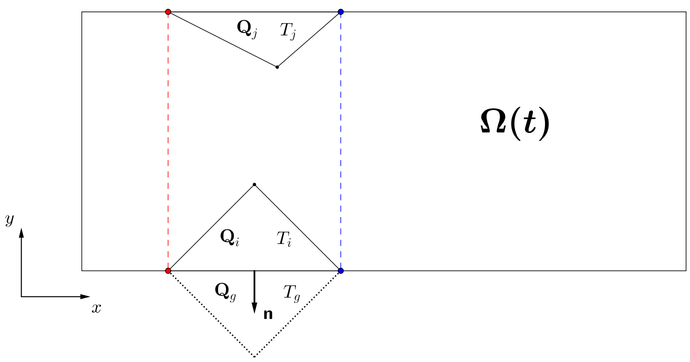


Figure D.1: Sketch of the time-dependent computational domain $\Omega(t)$ for the treatment of boundary conditions. $\boldsymbol{Q}_i$ represents the state of the internal element T_i, $\boldsymbol{Q}_g$ is the state of the ghost neighbor T_g, while $\boldsymbol{Q}_j$ denotes the boundary neighbor when periodic boundary conditions are imposed in the $y-$direction. In this case particular attention must be taken to build a logically connected mesh, where boundary vertexes along the periodic direction match exactly (red and blue vertexes).

edge. On the contrary, for *viscous* flows the velocity is set to zero on the boundary edge, thus leading to $\boldsymbol{v}_g = -\boldsymbol{v}_i$. The factor 2 in the above expression (D.1) has to be taken into account because of the factor $\frac{1}{2}$ in the numerical flux formulation, either the Rusanov type (2.69) or the Osher-type (2.71) flux;

- *space-time dependent* boundary conditions are used whenever the boundary is moving during the simulation, according to a prescribed boundary motion. Since the law which governs the boundary motion must be known, i.e. $\mathbf{x}_b(t)$, we simply set $\boldsymbol{Q}_g = \boldsymbol{Q}_b(\mathbf{x}_b(t))$, where $\boldsymbol{Q}_b(\mathbf{x}_b(t))$ denotes the state that has been computed according to the explicit expression for the boundary motion $\mathbf{x}_b(t)$;

- we rely on *periodic* boundary conditions to simulate an infinite computational domain. Although the domain is *de facto* finite, it is *logically* connected in one or more directions, hence yielding a logical infinite space in which the fluid flows. In this case we have $\boldsymbol{Q}_g = \boldsymbol{Q}_j$, with T_j representing the logical Neumann neighbor of element T_i, as depicted in Figure D.1. Particular care should be taken in order to build a logically connected mesh, ensuring that each vertex which lies on a periodic boundary has its own corresponding vertex on the other side of the domain, as highlighted by the red and blue lines in Figure D.1.

We underline that for finite volume schemes no "canonical" procedure is available to set boundary conditions. Thus, different ways are possible and they all are, in principle, correct.

Bibliography

[1] R. Abgrall. Approximation du problème de Riemann vraiment multidimensionnel des équations d'Euler par une méthode de type Roe, I: La linéarisation. *C.R. Acad. Sci. Ser. I*, 319:499 – 504, 1994.

[2] R. Abgrall. Approximation du problème de Riemann vraiment multidimensionnel des équations d'Euler par une méthode de type Roe, II: Solution du problème de Riemann approché. *C.R. Acad. Sci. Ser. I*, 319:625 – 629, 1994.

[3] R. Abgrall. On essentially non-oscillatory schemes on unstructured meshes : Analysis and implementation. *Journal of Computational Physics*, 114:45 – 58, 1994.

[4] R. Abgrall. On essentially non-oscillatory schemes on unstructured meshes: analysis and implementation. *Journal of Computational Physics*, 144:45–58, 1994.

[5] R. Abgrall. How to prevent pressure oscillations in multicomponent flow calculations: A quasi conservative approach. *Journal of Computational Physics*, 125:150–160, 1996.

[6] R. Abgrall and S. Karni. Computations of Compressible Multifluids. *Journal of Computational Physics*, 169:594–623, 2001.

[7] R. Abgrall and S. Karni. A comment on the computation of non-conservative products. *Journal of Computational Physics*, 229:2759–2763, 2010.

[8] R. Abgrall, B. Nkonga, and R. Saurel. Efficient numerical approximation of compressible multi-material flow for unstructured meshes. *Computers and Fluids*, 32:571–605, 2003.

[9] R. Abgrall and R. Saurel. Discrete equations for physical and numerical compressible multiphase mixtures. *Journal of Computational Physics*, 186:361–396, 2003.

[10] N. Andrianov, R. Saurel, and G. Warnecke. A simple method for compressible multiphase mixtures and interfaces. *International Journal for Numerical Methods in Fluids*, 41:109–131, 2003.

[11] N. Andrianov and G. Warnecke. The Riemann problem for the Baer-Nunziato two-phase flow model. *Journal of Computational Physics*, 212:434–464, 2004.

[12] M.R. Baer and J.W. Nunziato. A two-phase mixture theory for the deflagration-to-detonation transition (DDT) in reactive granular materials. *J. Multiphase Flow*, 12:861–889, 1986.

[13] A. Baeza, A. Martínez-Gavara, and P. Mulet. Adaptation based on interpolation errors for high order mesh refinement methods applied to conservation laws . *Applied Numerical Mathematics*, 62:278–296, 2012.

[14] A. Baeza and P. Mulet. Adaptive mesh refinement techniques for high-order shock capturing schemes for multi-dimensional hydrodynamic simulations. *International Journal for Numerical Methods in Fluids*, 52:455–471, 2006.

[15] D. Balsara. Total variation diminishing scheme for relativistic magneto-hydrodynamics. *The Astrophysical Journal Supplement Series*, 132:83–101, 2001.

[16] D. Balsara. Second-order accurate schemes for magnetohydrodynamics with divergence-free reconstruction. *The Astrophysical Journal Supplement Series*, 151:149–184, 2004.

[17] D. Balsara. Multidimensional hlle riemann solver; application to euler and magnetohydrodynamic flows. *Journal of Computational Physics*, 229:1970 – 1993, 2010.

[18] D. Balsara. A two-dimensional hllc riemann solver for conservation laws: Application to euler and magnetohydrodynamic flows. *Journal of Computational Physics*, 231:7476 – 7503, 2012.

[19] D. Balsara and C.W. Shu. Monotonicity preserving weighted essentially non-oscillatory schemes with increasingly high order of accuracy. *Journal of Computational Physics*, 160:405–452, 2000.

[20] D. Balsara and D. Spicer. A staggered mesh algorithm using high order godunov fluxes to ensure solenoidal magnetic fields in magnetohydrodynamic simulations. *Journal of Computational Physics*, 149:270–292, 1999.

[21] D.S. Balsara. Multidimensional HLLE Riemann solver: Application to Euler and magnetohydrodynamic flows. *Journal of Computational Physics*, 229:1970–1993, 2010.

[22] D.S. Balsara. A two-dimensional HLLC Riemann solver for conservation laws: Application to Euler and magnetohydrodynamic flows. *Journal of Computational Physics*, 231:7476–7503, 2012.

[23] D.S. Balsara. Self-adjusting, positivity preserving high order schemes for hydrodynamics and magnetohydrodynamics. *Journal of Computational Physics*, 231:7504–7517, 2012.

[24] D.S. Balsara. Self-adjusting, positivity preserving high order schemes for hydrodynamics and magnetohydrodynamics. *Journal of Computational Physics*, 231:7504 – 7517, 2012.

[25] D.S. Balsara. Multidimensional Riemann problem with self-similar internal structure. Part I - Application to hyperbolic conservation laws on structured meshes. *Journal of Computational Physics*, 277:163–200, 2014.

[26] D.S. Balsara and M. Dumbser. Multidimensional Riemann problem with self-similar internal structure. Part II - Application to hyperbolic conservation laws on unstructured meshes. *Journal of Computational Physics*, 2015. in press: DOI:10.1016/j.jcp.2014.11.004.

[27] D.S. Balsara, C. Meyer, M. Dumbser, H. Du, and Z. Xu. Efficient implementation of ader schemes for euler and magnetohydrodynamical flows on structured meshes speed comparisons with runge-kutta methods. *Journal of Computational Physics*, 235:934–969, 2013.

[28] D.S. Balsara, T. Rumpf, M. Dumbser, and C.D. Munz. Efficient, high accuracy ADER-WENO schemes for hydrodynamics and divergence-free magnetohydrodynamics. *Journal of Computational Physics*, 228:2480–2516, 2009.

[29] T.J. Barth and P.O. Frederickson. Higher order solution of the euler equations on unstructured grids using quadratic reconstruction. *28th Aerospace Sciences Meeting*, pages AIAA paper no. 90–0013, January 1990.

[30] T.J. Barth and D.C. Jespersen. The design and application of upwind schemes on unstructured meshes. *AIAA Paper 89-0366*, pages 1–12, 1989.

[31] M. Ben-Artzi and J. Falcovitz. A second-order godunov-type scheme for compressible fluid dynamics. *Journal of Computational Physics*, 55:1–32, 1984.

[32] D.J. Benson. Computational methods in lagrangian and eulerian hydrocodes. *Computer Methods in Applied Mechanics and Engineering*, 99:235–394, 1992.

[33] M. J. Berger and P. Colella. Local adaptive mesh refinement for shock hydrodynamics. *Journal of Computational Physics*, 82:64–84, May 1989.

[34] M. J. Berger and J. Oliger. Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations. *Journal of Computational Physics*, 53:484, March 1984.

[35] M. Berndt, J. Breil, S. Galera, M. Kucharik, P.H. Maire, and M. Shashkov. Two-step hybrid conservative remapping for multimaterial arbitrary Lagrangian-Eulerian methods. *Journal of Computational Physics*, 230:6664–6687, 2011.

[36] M. Berndt, M. Kucharik, and M.J. Shashkov. Using the feasible set method for rezoning in ALE. *Procedia Computer Science*, 1:1879 – 1886, 2010.

[37] P. Bochev, D. Ridzal, and M.J. Shashkov. Fast optimization-based conservative remap of scalar fields through aggregate mass transfer. *Journal of Computational Physics*, 246:37–57, 2013.

[38] W. Boscheri, D.S. Balsara, and M. Dumbser. Lagrangian ADER-WENO Finite Volume Schemes on Unstructured Triangular Meshes Based On Genuinely Multidimensional HLL Riemann Solvers. *Journal of Computational Physics*, 267:112–138, 2014.

[39] W. Boscheri and M. Dumbser. Arbitrary–Lagrangian–Eulerian One–Step WENO Finite Volume Schemes on Unstructured Triangular Meshes. *Communications in Computational Physics*, 14:1174–1206, 2013.

[40] W. Boscheri and M. Dumbser. A Direct Arbitrary-Lagrangian-Eulerian ADER-WENO Finite Volume Scheme on Unstructured Tetrahedral Meshes for Conservative and Nonconservative Hyperbolic Systems in 3D. *Journal of Computational Physics*, 275:484–523, 2014.

[41] W. Boscheri, M. Dumbser, and D.S. Balsara. High Order Lagrangian ADER-WENO Schemes on Unstructured Meshes – Application of Several Node Solvers to Hydrodynamics and Magnetohydrodynamics. *International Journal for Numerical Methods in Fluids*, 76:737–778, 2014.

[42] W. Boscheri, M. Dumbser, and M. Righetti. A semi-implicit scheme for 3d free surface flows with high order velocity reconstruction on unstructured voronoi meshes. *International Journal for Numerical Methods in Fluids*, 72:607–631, 2013.

[43] W. Boscheri, M. Dumbser, and O. Zanotti. High order cell-centered lagrangian-type finite volume schemes with time-accurate local time stepping on unstructured triangular meshes. *Journal of Computational Physics*, 2014. submitted to.

[44] W. Boscheri, R. Loubère, and M. Dumbser. An Efficient Quadrature-Free Formulation for High Order Arbitrary-Lagrangian-Eulerian ADER-WENO Finite Volume Schemes on Unstructured Meshes. *Journal of Scientific computing*, 2014. submitted to.

[45] W. Boscheri, R. Loubère, and M. Dumbser. Direct Arbitrary-Lagrangian-Eulerian ADER-MOOD Finite Volume Schemes for Multidimensional Hyperbolic Conservation Laws. *Journal of Computational Physics*, 2014. submitted to.

[46] A. Bourgeade, P. LeFloch, and P.A. Raviart. An asymptotic expansion for the solution of the generalized riemann problem. Part II: application to the gas dynamics equations. *Annales de l'institut Henri Poincaré (C) Analyse non linéaire*, 6:437–480, 1989.

[47] J. Breil, T. Harribey, P.H. Maire, and M. Shashkov. A multi-material reale method with mof interface reconstruction. *Computers and Fluids*, 83:115–125, 2013.

[48] J. Breil, T. Harribey, P.H. Maire, and M.J. Shashkov. A multi-material ReALE method with MOF interface reconstruction. *Computers and Fluids*, 83:115–125, 2013.

[49] R. Bürger, P. Mulet, and L.M. Villada. Spectral weno schemes with adaptive mesh refinement for models of polydisperse sedimentation. *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für angewandte Mathematik und Mechanik*, pages n/a–n/a, 2012.

[50] J.C. Butcher. *The Numerical Analysis of Ordinary Differential Equations: Runge-Kutta and General Linear Methods.* Wiley, 1987.

[51] 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 Computational Physics*, 146:227–262, 1998.

[52] E.J. Caramana, C.L. Rousculp, and D.E. Burton. A compatible, energy and symmetry preserving lagrangian hydrodynamics algorithm in three-dimensional cartesian geometry. *Journal of Computational Physics*, 157:89 – 119, 2000.

[53] G. Carré, S. Del Pino, B. Després, and E. Labourasse. A cell-centered Lagrangian hydrodynamics scheme on general unstructured meshes in arbitrary dimension. *Journal of Computational Physics*, 228:5160–5183, 2009.

[54] C. C. Castro and E. F. Toro. Solvers for the high-order riemann problem for hyperbolic balance laws. *Journal of Computational Physics*, 227:2481–2513, 2008.

[55] C.E. Castro, M. Käser, and E.F. Toro. Space–time adaptive numerical methods for geophysical applications. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 367:4613–4631, 2009.

[56] M.J. Castro, J.M. Gallardo, J.A. López, and C. Parés. Well-balanced high order extensions of godunov's method for semilinear balance laws. *SIAM Journal of Numerical Analysis*, 46:1012–1039, 2008.

[57] M.J. Castro, J.M. Gallardo, and C. Parés. High-order finite volume schemes based on reconstruction of states for solving hyperbolic systems

with nonconservative products. applications to shallow-water systems. *Mathematics of Computation*, 75:1103–1134, 2006.

[58] M.J. Castro, P.G. LeFloch, M.L. Muñoz-Ruiz, and C. Parés. Why many theories of shock waves are necessary: Convergence error in formally path-consistent schemes. *Journal of Computational Physics*, 227:8107–8129, 2008.

[59] V. Casulli. Semi-implicit finite difference methods for the two-dimensional shallow water equations. *Journal of Computational Physics*, 86:56–74, 1990.

[60] V. Casulli. A semi-implicit finite difference method for non-hydrostatic free-surface flows. *International Journal for Numerical Methods in Fluids*, 30:425–440, 1999.

[61] V. Casulli. A semi–implicit numerical method for the free–surface Navier–Stokes equations. *International Journal for Numerical Methods in Fluids*, 74:605–622, 2014.

[62] V. Casulli and R.T. Cheng. Semi-implicit finite difference methods for three-dimensional shallow water flow. *International Journal of Numerical Methods in Fluids*, 15:629–648, 1992.

[63] V. Casulli and G. S. Stelling. Semi-implicit subgrid modelling of three-dimensional free-surface flows. *International Journal for Numerical Methods in Fluids*, 67:441–449, 2011.

[64] J. Cesenek, M. Feistauer, J. Horacek, V. Kucera, and J. Prokopova. Simulation of compressible viscous flow in time-dependent domains. *Applied Mathematics and Computation*, 219:7139–7150, 2013.

[65] J. Cheng and C.W. Shu. A high order ENO conservative Lagrangian type scheme for the compressible Euler equations. *Journal of Computational Physics*, 227:1567–1596, 2007.

[66] J. Cheng and C.W. Shu. A cell-centered Lagrangian scheme with the preservation of symmetry and conservation properties for compressible fluid flows in two-dimensional cylindrical geometry. *Journal of Computational Physics*, 229:7191–7206, 2010.

[67] J. Cheng and C.W. Shu. Improvement on spherical symmetry in two-dimensional cylindrical coordinates for a class of control volume Lagrangian schemes. *Communications in Computational Physics*, 11:1144–1168, 2012.

[68] J. Cheng and E.F. Toro. A 1D conservative Lagrangian ADER scheme. *Chinese Journal of Computational Physics*, 30:501–508, 2013.

[69] S. Clain, S. Diot, and R. Loubère. A high-order finite volume method for systems of conservation laws - Multi-dimensional Optimal Order Detection (MOOD). *ournal of Computational Physics*, 230:4028–4050, 2011.

[70] S. Clain, S. Diot, and R. Loubère. A high-order finite volume method for systems of conservation laws - multi-dimensional optimal order detection (mood). *Journal of Computational Physics*, 230:4028 – 4050, 2011.

[71] A. Claisse, B. Després, E.Labourasse, and F. Ledoux. A new exceptional points method with application to cell-centered Lagrangian schemes and curved meshes. *Journal of Computational Physics*, 231:4324–4354, 2012.

[72] B. Cockburn, G. E. Karniadakis, and C.W. Shu. *Discontinuous Galerkin Methods*. Lecture Notes in Computational Science and Engineering. Springer, 2000.

[73] P. Colella. A direct eulerian muscl scheme for gas dynamics. *SIAM J. Sci. Statist. Comput.*, 6:104 – 117, 1985.

[74] P. Colella and M.D. Sekora. A limiter for PPM that preserves accuracy at smooth extrema. *Journal of Computational Physics*, 227:7069–7076, 2008.

[75] A. Dedner, F. Kemm, D. Kröner, C.-D. Munz, T. Schnitzer, and M. Wesenberg. Hyperbolic divergence cleaning for the MHD equations. *Journal of Computational Physics*, 175:645–673, 2002.

[76] V. Deledicque and M.V. Papalexandris. An exact Riemann solver for compressible two-phase flow models containing non-conservative products. *Journal of Computational Physics*, 222:217–245, 2007.

[77] B. Després and C. Mazeran. Symmetrization of Lagrangian gas dynamic in dimension two and multimdimensional solvers. *C.R. Mecanique*, 331:475–480, 2003.

[78] B. Després and C. Mazeran. Lagrangian gas dynamics in two-dimensions and Lagrangian systems. *Archive for Rational Mechanics and Analysis*, 178:327–372, 2005.

[79] S. Diot, S. Clain, and R. Loubère. Improved detection criteria for the Multi-dimensional Optimal Order Detection (MOOD) on unstructured meshes with very high-order polynomials. *Computers and Fluids*, 64:43–63, 2012.

[80] S. Diot, S. Clain, and R. Loubère. Improved detection criteria for the multi-dimensional optimal order detection (mood) on unstructured meshes with very high-order polynomials. *Journal of Computational Physics*, 64:43 – 63, 2012.

[81] S. Diot, R. Loubère, and S. Clain. The MOOD method in the three-dimensional case: very high-order finite volume method for hyperbolic systems. *International Journal for numerical Methods in Fluids*, 73:362–392, 2013.

[82] V.A. Dobrev, T.E. Ellis, Tz.V. Kolev, and R.N. Rieben. Curvilinear Finite elements for Lagrangian hydrodynamics. *International Journal for Numerical Methods in Fluids*, 65:1295–1310, 2011.

[83] V.A. Dobrev, T.E. Ellis, Tz.V. Kolev, and R.N. Rieben. High Order Curvilinear Finite Elements for Lagrangian Hydrodynamics. *SIAM Journal on Scientific Computing*, 34:606–641, 2012.

[84] V.A. Dobrev, T.E. Ellis, Tz.V. Kolev, and R.N. Rieben. High Order Curvilinear Finite Elements for axisymmetric Lagrangian Hydrodynamics. *Computers and Fluids*, 83:58–69, 2013.

[85] V. Dolejsi. Semi-implicit interior penalty discontinuous Galerkin methods for viscous compressible flows. *Communications in Computational Physics*, 4:231–274, 2008.

[86] V. Dolejsi and M. Feistauer. A semi-implicit discontinuous Galerkin finite element method for the numerical solution of inviscid compressible flow. *Journal of Computational Physics*, 198:727–746, 2004.

[87] V. Dolejsi, M. Feistauer, and J. Hozman. Analysis of semi-implicit DGFEM for nonlinear convection-diffusion problems on nonconforming meshes. *Computer Methods in Applied Mechanics and Engineering*, 196:2813–2827, 2007.

[88] M. Dumbser D.S. Balsara and R. Abgrall. Multidimensional HLLC Riemann Solver for Unstructured Meshes - With Application to Euler and MHD Flows. *Journal of Computational Physics*, 261:172–208, 2014.

[89] L. Dubcova, M. Feistauer, J. Horacek, and P. Svacek. Numerical simulation of interaction between turbulent flow and a vibrating airfoil. *Computing and Visualization in Science*, 12:207–225, 2009.

[90] M. Dubiner. Spectral methods on triangles and other domains. *Journal of Scientific Computing*, 6:345–390, 1991.

[91] J.K. Dukovicz and B. Meltz. Vorticity errors in multidimensional lagrangian codes. *Journal of Computational Physics*, 99:115 – 134, 1992.

[92] J.K. Dukowicz. A general non-iterative riemann solver for godunov's method. *Journal of Computational Physics*, 61:119 – 137, 1984.

[93] M. Dumbser. Arbitrary-Lagrangian-Eulerian ADER-WENO Finite Volume Schemes with Time-Accurate Local Time Stepping for Hyperbolic Conservation Laws. *Computational Methods in Applied Mechanics and Engineering*, 280:57–83, 2014.

[94] M. Dumbser, D. Balsara, E.F. Toro, and C.D. Munz. A unified framework for the construction of one-step finite-volume and discontinuous Galerkin schemes. *Journal of Computational Physics*, 227:8209–8253, 2008.

[95] M. Dumbser, D.S. Balsara, E.F. Toro, and C.-D. Munz. A unified framework for the construction of one-step finite volume and discontinuous galerkin schemes on unstructured meshes. *Journal of Computational Physics*, 227:8209 – 8253, 2008.

[96] M. Dumbser and W. Boscheri. High-order unstructured Lagrangian one–step WENO finite volume schemes for non–conservative hyperbolic systems: Applications to compressible multi–phase flows. *Computers and Fluids*, 86:405 – 432, 2013.

[97] M. Dumbser, M. Castro, C. Parés, and E.F. Toro. ADER schemes on unstructured meshes for non-conservative hyperbolic systems: Applications to geophysical flows. *Computers and Fluids*, 38:1731–1748, 2009.

[98] M. Dumbser, C. Enaux, and E.F. Toro. Finite volume schemes of very high order of accuracy for stiff hyperbolic balance laws. *Journal of Computational Physics*, 227:3971–4001, 2008.

[99] M. Dumbser, A. Hidalgo, M. Castro, C. Parés, and E.F. Toro. FORCE schemes on unstructured meshes II: Non-conservative hyperbolic systems. *Computer Methods in Applied Mechanics and Engineering*, 199:625–647, 2010.

[100] M. Dumbser, A. Hidalgo, and O. Zanotti. High order space-time adaptive ADER-WENO finite volume schemes for non-conservative hyperbolic systems. *Computer Methods in Applied Mechanics and Engineering*, 268:359–387, 2014.

[101] M. Dumbser, M. Kaeser, V.A. Titarev, and E.F. Toro. Quadrature-free non-oscillatory finite volume schemes on unstructured meshes for nonlinear hyperbolic systems. *Journal of Computational Physics*, 226:204 – 243, 2007.

[102] M. Dumbser and M. Käser. Arbitrary high order non-oscillatory finite volume schemes on unstructured meshes for linear hyperbolic systems. *Journal of Computational Physics*, 221:693–723, 2007.

[103] M. Dumbser, M. Käser, V.A Titarev, and E.F. Toro. Quadrature-free non-oscillatory finite volume schemes on unstructured meshes for nonlinear hyperbolic systems. *Journal of Computational Physics*, 226:204–243, 2007.

[104] M. Dumbser, M. Käser, and E. F. Toro. An arbitrary high order discontinuous Galerkin method for elastic waves on unstructured meshes V: Local time stepping and p-adaptivity. *Geophysical Journal International*, 171:695–717, 2007.

[105] M. Dumbser and C.D. Munz. Building blocks for arbitrary high order discontinuous Galerkin schemes. *Journal of Scientific Computing*, 27:215–230, 2006.

[106] M. Dumbser and E. F. Toro. On universal Osher-type schemes for general nonlinear hyperbolic conservation laws. *Communications in Computational Physics*, 10:635–671, 2011.

[107] M. Dumbser and E. F. Toro. A simple extension of the Osher Riemann solver to non-conservative hyperbolic systems. *Journal of Scientific Computing*, 48:70–88, 2011.

[108] M. Dumbser, A. Uuriintsetseg, and O. Zanotti. On Arbitrary-Lagrangian-Eulerian One-Step WENO Schemes for Stiff Hyperbolic Balance Laws. *Communications in Computational Physics*, 14:301–327, 2013.

[109] M. Dumbser and O. Zanotti. Very high order PNPM schemes on unstructured meshes for the resistive relativistic MHD equations. *Journal of Computational Physics*, 228:6991–7006, 2009.

[110] M. Dumbser, O. Zanotti, A. Hidalgo, and D. S. Balsara. ADER-WENO finite volume schemes with space-time adaptive mesh refinement. *Journal of Computational Physics*, 248:257–286, 2013.

[111] M. Dumbser, O. Zanotti, R. Loubère, and S. Diot. A posteriori subcell limiting of the discontinuous Galerkin finite element method for hyperbolic conservation laws. *Journal of Computational Physics*, 278:47 – 75, 2014.

[112] B. Einfeldt. On godunov-type methods for gas dynamics. *SIAM J. Numer. Anal.*, 25:294 – 318, 1988.

[113] B. Einfeldt, C. D. Munz, P. L. Roe, and B. Sjögreen. On godunov-type methods near low densities. *Journal of Computational Physics*, 92:273–295, 1991.

[114] R. Fedkiw, T. Aslam, B. Merriman, and S. Osher. A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method). *Journal of Computational Physics*, 152:457–492, 1999.

[115] R.P. Fedkiw, T. Aslam, and S. Xu. The Ghost Fluid method for deflagration and detonation discontinuities. *Journal of Computational Physics*, 154:393–427, 1999.

[116] M. Feistauer, J. Horacek, M. Ruzicka, and P. Svacek. Numerical analysis of flow-induced nonlinear vibrations of an airfoil with three degrees of freedom. *Computers and Fluids*, 49:110–127, 2011.

[117] M. Feistauer, V. Kucera, J. Prokopova, and J. Horacek. The ALE discontinuous Galerkin method for the simulatio of air flow through pulsating human vocal folds. *AIP Conference Proceedings*, 1281:83–86, 2010.

[118] A. Ferrari. SPH simulation of free surface flow over a sharp-crested weir. *Advances in Water Resources*, 33:270–276, 2010.

[119] A. Ferrari, M. Dumbser, E.F. Toro, and A. Armanini. A New Stable Version of the SPH Method in Lagrangian Coordinates. *Communications in Computational Physics*, 4:378–404, 2008.

[120] A. Ferrari, M. Dumbser, E.F. Toro, and A. Armanini. A new 3D parallel SPH scheme for free surface flows. *Computers & Fluids*, 38:1203–1217, 2009.

[121] A. Ferrari, L. Fraccarollo, M. Dumbser, E.F. Toro, and A. Armanini. Three-dimensional flow evolution after a dambreak. *Journal of Fluid Mechanics*, 663:456–477, 2010.

[122] A. Ferrari, C.D. Munz, and B. Weigand. A high order sharp interface method with local timestepping for compressible multiphase flows. *Communications in Computational Physics*, 9:205–230, 2011.

[123] J. Flaherty, R. Loy, M. Shephard, B. Szymanski, J. Teresco, and L. Ziantz. Adaptive local refinement with octree load–balancing for the parallel solution of three–dimensional conservation laws. *Journal of Parallel and Distributed Computing*, 47:139–152, 1997.

[124] P. Le Floch and P.A. Raviart. An asymptotic expansion for the solution of the generalized riemann problem. Part I: General theory. *Annales de l'institut Henri Poincaré (C) Analyse non linéaire*, 5:179–207, 1988.

[125] M.M. Francois, M.J. Shashkov, T.O. Masser, and E.D. Dendy. A comparative study of multimaterial Lagrangian and Eulerian methods with pressure relaxation. *Computers and Fluids*, 83:126–136, 2013.

[126] O. Friedrich. Weighted essentially non-oscillatory schemes for the interpolation of mean values on unstructured grids. *Journal of Computational Physics*, 144:194–212, 1998.

[127] F.Vilar. Cell-centered discontinuous Galerkin discretization for two-dimensional Lagrangian hydrodynamics. *Computers and Fluids*, 64:64–73, 2012.

[128] F.Vilar, P.H. Maire, and R. Abgrall. A discontinuous Galerkin discretization for solving the two-dimensional gas dynamics equations written under total Lagrangian formulation on general unstructured grids. *INRIA Research Report N° 8483 2014*. available at http://hal.inria.fr/docs/00/95/07/82/PDF/RR-8483.pdf.

[129] F.Vilar, P.H. Maire, and R. Abgrall. Cell-centered discontinuous Galerkin discretizations for two-dimensional scalar conservation laws on unstructured grids and for one-dimensional Lagrangian hydrodynamics. *Computers and Fluids*, 46(1):498–604, 2010.

[130] S. Galera, P.H. Maire, and J. Breil. A two-dimensional unstructured cell-centered multi-material ale scheme using vof interface reconstruction. *Journal of Computational Physics*, 229:5755–5787, 2010.

[131] G. Gassner, F. Lörcher, and C. D. Munz. A discontinuous Galerkin scheme based on a space-time expansion II. viscous flow equations in multi dimensions. *Journal of Scientific Computing*, 34:260–286, 2008.

[132] B. Giacomazzo and L. Rezzolla. The exact solution of the Riemann problem in relativistic magnetohydrodynamics. *Journal of Fluid Mechanics*, 562:223–259, 2006.

[133] S. K. Godunov. Finite difference methods for the computation of discontinuous solutions of the equations of fluid dynamics. *Mathematics of the USSR*, 47:271–306, 1959.

[134] S.K. Godunov. Finite difference methods for the computation of discontinuous solutions of the equations of fluid dynamics. *Mathematics of the USSR: Sbornik*, 47:271–306, 1959.

[135] P.M. Gresho and S.T. Chan. On the theory of semi-implicit projection methods for viscous incompressible flow and its implementation via a finite element method that also introduces a nearly consistent mass matrix. Part 2: Implementation. *International Journal for Numerical Methods in Fluids*, 11(5):621–659, 1990.

[136] M.J. Grote and T. Mitkova. Explicit local time-stepping methods for Maxwell's equations. *Journal of Computational and Applied Mathematics*, 234:3283–3302, 2010.

[137] M.J. Grote and T. Mitkova. High-order explicit local time-stepping methods for damped wave equations. *Journal of Computational and Applied Mathematics*, 239:270–289, 2013.

[138] A. Harten, B. Engquist, S. Osher, and S. Chakravarthy. Uniformly high order essentially non-oscillatory schemes, III. *Journal of Computational Physics*, 71:231–303, 1987.

[139] A. Harten, P. D. Lax, and B. van Leer. On upstream differencing and godunov-type schemes for hyperbolic conservation laws. *SIAM Rev*, 25:289 – 315, 1983.

[140] R.W. Healy and T.F. Russel. Solution of the advection-dispersion equation in two dimensions by a finite-volume eulerian-lagrangian localized adjoint method. *Advances in Water Resources*, 21:11–26, 1998.

[141] A. Hidalgo and M. Dumbser. ADER schemes for nonlinear systems of stiff advection-diffusion-reaction equations. *Journal of Scientific Computing*, 48:173–189, 2011.

[142] C. Hirt, A. Amsden, and J. Cook. An arbitrary lagrangian-eulerian computing method for all flow speeds. *Journal of Computational Physics*, 14:227–253, 1974.

[143] V. Honkkila and P. Janhunen. HLLC solver for ideal relativistic MHD. *Journal of Computational Physics*, 223:643–656, 2007.

[144] C. Hu and C.W. Shu. Weighted essentially non-oscillatory schemes on triangular meshes. *Journal of Computational Physics*, 150:97–127, 1999.

[145] C.S. Huang, T. Arbogast, and J. Qiu. An Eulerian-Lagrangian WENO finite volume scheme for advection problems. *Journal of Computational Physics*, 231:4028–4052, 2012.

[146] G.S. Jiang and C.W. Shu. Efficient implementation of weighted ENO schemes. *Journal of Computational Physics*, 126:202–228, 1996.

[147] J.R. Kamm and F.X. Timmes. On efficient generation of numerically robust sedov solutions. *Technical Report LA-UR-07-2849,LANL*, 2007.

[148] A.K. Kapila, R. Menikoff, J.B. Bdzil, S.F. Son, and D.S. Stewart. Two-phase modelling of DDT in granular materials: reduced equations. *Physics of Fluids*, 13:3002–3024, 2001.

[149] G. E. Karniadakis and S. J. Sherwin. *Spectral/hp Element Methods in CFD.* Oxford University Press, 1999.

[150] G.E. Karniadakis and S.J. Sherwin. *Spectral/hp Element Methods in CFD.* Oxford University Press, 1999.

[151] M. Käser and A. Iske. ADER schemes on adaptive triangular meshes for scalar conservation laws. *Journal of Computational Physics*, 205:486–508, 2005.

[152] M. Käser and A. Iske. ADER schemes on adaptive triangular meshes for scalar conservation laws. *Journal of Computational Physics*, 205:486 – 508, 2005.

[153] R.E. Kidder. Laser-driven compression of hollow shells: power requirements and stability limitations. *Nucl. Fus.*, 1:3 – 14, 1976.

[154] P.M. Knupp. Achieving finite element mesh quality via optimization of the jacobian matrix norm and associated quantities. part ii - a framework for volume mesh optimization and the condition number of the jacobian matrix. *Int. J. Numer. Meth. Engng.*, 48:1165 – 1185, 2000.

[155] L. Krivodonova. An efficient local time–stepping scheme for solution of nonlinear conservation laws. *Journal of Computational Physics*, 229:8537–8551, 2010.

[156] M. Kucharik, J. Breil, S. Galera, P.H. Maire, M. Berndt, and M.J. Shashkov. Hybrid remap for multi-material ALE. *Computers and Fluids*, 46:293–297, 2011.

[157] M. Kucharik and M.J. Shashkov. One-step hybrid remapping algorithm for multi-material arbitrary Lagrangian-Eulerian methods. *Journal of Computational Physics*, 231:2851–2864, 2012.

[158] A. Kurganov and E. Tadmor. Solution of two-dimensional Riemann problems for gas dynamics without Riemann problem solvers. *Numer. Methods Partial Differential Equations*, 18:584–608, 2002.

[159] P.D. Lax and B. Wendroff. Systems of conservation laws. *Communications in Pure and Applied Mathematics*, 13:217–237, 1960.

[160] M. Lentine, J.T. Grétarsson, and R. Fedkiw. An unconditionally stable fully conservative semi-Lagrangian method. *Journal of Computational Physics*, 230:2857–2879, 2011.

[161] Z. Li, X. Yu, and Z. Jia. The cell-centered discontinuous Galerkin method for Lagrangian compressible Euler equations in two dimensions. *Computers and Fluids*, 96:152–164, 2014.

[162] R. Liska, M.J. Shashkov P. Váchal, and B. Wendroff. Synchronized flux corrected remapping for ALE methods. *Computers and Fluids*, 46:312–317, 2011.

[163] W. Liu, J. Cheng, and C.W. Shu. High order conservative Lagrangian schemes with Lax-Wendroff type time discretization for the compressible Euler equations. *Journal of Computational Physics*, 228:8872–8891, 2009.

[164] F. Lörcher, G. Gassner, and C. D. Munz. A discontinuous Galerkin scheme based on a space-time expansion. I. inviscid compressible flow in one space dimension. *Journal of Scientific Computing*, 32:175–199, 2007.

[165] R. Loubère, M. Dumbser, and S. Diot. A New Family of High Order Unstructured MOOD and ADER Finite Volume Schemes for Multidimensional Systems of Hyperbolic Conservation Laws. *Communications in Computational Physics*, 16:718–763, 2014.

[166] R. Loubère, P.H. Maire, M. Shashkov, J. Breil, and S. Galera. Reale: A reconnection-based arbitrary-lagrangian-eulerian method. *Journal of Computational Physics*, 229:4724–4761, 2010.

[167] R. Loubère, P.H. Maire, and P. Váchal. A second-order compatible staggered Lagrangian hydrodynamics scheme using a cell-centered multidimensional approximate Riemann solver. *Procedia Computer Science*, 1:1931–1939, 2010.

[168] R. Loubère, P.H. Maire, and P. Váchal. 3D staggered Lagrangian hydrodynamics scheme with cell-centered Riemann solver-based artificial viscosity. *International Journal for Numerical Methods in Fluids*, 72:22 – 42, 2013.

[169] P.H. Maire. A high-order cell-centered lagrangian scheme for two-dimensional compressible fluid flows on unstructured meshes. *Journal of Computational Physics*, 228:2391–2425, 2009.

[170] P.H. Maire. A high-order one-step sub-cell force-based discretization for cell-centered lagrangian hydrodynamics on polygonal grids. *Computers and Fluids*, 46(1):341–347, 2011.

[171] P.H. Maire. A unified sub-cell force-based discretization for cell-centered lagrangian hydrodynamics on polygonal grids. *International Journal for Numerical Methods in Fluids*, 65:1281–1294, 2011.

[172] P.H. Maire, R. Abgrall, J. Breil, and J. Ovadia. A cell-centered lagrangian scheme for two-dimensional compressible flow problems. *SIAM Journal on Scientific Computing*, 29:1781–1824, 2007.

[173] P.H. Maire and J. Breil. A second-order cell-centered lagrangian scheme for two-dimensional compressible flow problems. *International Journal for Numerical Methods in Fluids*, 56:1417–1423, 2007.

[174] P.H. Maire and B. Nkonga. Multi-scale Godunov-type method for cell-centered discrete Lagrangian hydrodynamics. *Journal of Computational Physics*, 228:799–821, 2009.

[175] G. Dal Maso, P.G. LeFloch, and F. Murat. Definition and weak stability of nonconservative products. *J. Math. Pures Appl.*, 74:483–548, 1995.

[176] O. Le Métayer, J. Massoni, and R. Saurel. Modelling evaporation fronts with reactive Riemann solvers. *Journal of Computational Physics*, 205:567–610, 2005.

[177] A. Mignone and G. Bodo. An HLLC Riemann solver for relativistic flows - II. Magnetohydrodynamics. *Monthly Notices of the Royal Astronomical Society*, 368:1040–1054, 2006.

[178] R.C. Millington, E.F. Toro, and L.A.M. Nejad. *Arbitrary High Order Methods for Conservation Laws I: The One Dimensional Scalar Case.* PhD thesis, Manchester Metropolitan University, Department of Computing and Mathematics, June 1999.

[179] J.J. Monaghan. Simulating free surface flows with SPH. *Journal of Computational Physics*, 110:399–406, 1994.

[180] M.L. Muñoz and C. Parés. Godunov method for nonconservative hyperbolic systems. *Mathematical Modelling and Numerical Analysis*, 41:169–185, 2007.

[181] W. Mulder, S. Osher, and J.A. Sethian. Computing interface motion in compressible gas dynamics. *Journal of Computational Physics*, 100:209–228, 1992.

[182] C.D. Munz. On Godunov-type schemes for Lagrangian gas dynamics. *SIAM Journal on Numerical Analysis*, 31:17–42, 1994.

[183] A. Murrone and H. Guillard. A five equation reduced model for compressible two phase flow problems. *Journal of Computational Physics*, 202:664–698, 2005.

[184] W.F. Noh. Errors for calculations of strong shocks using artificial viscosity and an artificial heat flux. *Journal of Computational Physics*, 72:78 – 120, 1987.

[185] R. Saurel O. Le Métayer, J. Massoni. Modelling evaporation fronts with reactive Riemann solvers. *Journal of Computational Physics*, 205:567–610, 2005.

[186] C. Olliver-Gooch and M. Van Altena. A high-order-accurate unstructured mesh finite-volume scheme for the advection-diffusion equation. *Journal of Computational Physics*, 181:729 – 752, 2002.

[187] A. López Ortega and G. Scovazzi. A geometrically-conservative, synchronized, flux-corrected remap for arbitrary Lagrangian-Eulerian computations with nodal finite elements. *Journal of Computational Physics*, 230:6709–6741, 2011.

[188] S. Osher and J.A. Sethian. Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics*, 79:12–49, 1988.

[189] S. Osher and F. Solomon. Upwind difference schemes for hyperbolic conservation laws. *Math. Comput.*, 38:339–374, 1982.

[190] S. Osher and F. Solomon. Upwind difference schemes for hyperbolic systems of conservation laws. *Mathematics of Computation*, 38:339 – 374, 1982.

[191] C. Palenzuela, L. Lehner, O. Reula, and L. Rezzolla. Beyond ideal MHD: towards a more realistic modeling of relativistic astrophysical plasmas. *Mon. Not. R. Astron. Soc.*, 394:1727–1740, 2009.

[192] C. Parés. Numerical methods for nonconservative hyperbolic systems: a theoretical framework. *SIAM Journal on Numerical Analysis*, 44:300–321, 2006.

[193] C. Parés and M.J. Castro. On the well-balance property of roe's method for nonconservative hyperbolic systems. applications to shallow-water systems. *Mathematical Modelling and Numerical Analysis*, 38:821–852, 2004.

[194] J.S. Peery and D.E. Carroll. Multi-material ale methods in unstructured grids,. *Computer Methods in Applied Mechanics and Engineering*, 187:591–619, 2000.

[195] F. Petitpas, J. Massoni, R. Saurel, E. Lapebie, and L. Munier. Diffuse interface model for high speed cavitating underwater systems. *International Journal of Multiphase Flow*, 35:747–759, 2009.

[196] J.M. Qiu and C.W. Shu. Conservative high order semi-Lagrangian finite difference WENO methods for advection in incompressible flow. *Journal of Computational Physics*, 230:863–889, 2011.

[197] L. Rezzolla and O. Zanotti. An improved exact riemann solver for relativistic hydrodynamics. *Journal of Fluid Mechanics*, 449:395–411, 2001.

[198] S. Rhebergen, O. Bokhove, and J.J.W. van der Vegt. Discontinuous Galerkin finite element methods for hyperbolic nonconservative partial differential equations. *Journal of Computational Physics*, 227:1887–1922, 2008.

[199] K. Riemslagh, J. Vierendeels, and E. Dick. An arbitrary Lagrangian-Eulerian finite-volume method for the simulation of rotary displacement pump flow. *Applied Numerical Mathematics*, 32:419–433, 2000.

[200] P.L. Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics*, 43:357–372, 1981.

[201] V. V. Rusanov. Calculation of interaction of non-steady shock waves with obstacles. *J. Comput. Math. Phys. USSR*, 1:267 – 305, 1961.

[202] S.K. Sambasivan, M.J. Shashkov, and D.E. Burton. A finite volume cell-centered Lagrangian hydrodynamics approach for solids in general unstructured grids. *International Journal for Numerical Methods in Fluids*, 72:770–810, 2013.

[203] S.K. Sambasivan, M.J. Shashkov, and D.E. Burton. Exploration of new limiter schemes for stress tensors in Lagrangian and ALE hydrocodes. *Computers and Fluids*, 83:98–114, 2013.

[204] R. Saurel and R. Abgrall. A multiphase Godunov method for compressible multifluid and multiphase flows. *Journal of Computational Physics*, 150:425–467, 1999.

[205] R. Saurel, S. Gavrilyuk, and F. Renaud. A multiphase model with internal degrees of freedom: Application to shock-bubble interaction. *Journal of Fluid Mechanics*, 495:283–321, 2003.

[206] R. Saurel, M. Larini, and J.C. Loraud. Exact and Approximate Riemann Solvers for Real Gases. *Journal of Computational Physics*, 112:126–137, 1994.

[207] R. Saurel, J. Massoni, and F. Renaud. A numerical method for one-dimensional compressible multiphase flows on moving meshes. *International Journal for Numerical Methods in Fluids*, 54:1425–1450, 2007.

[208] R. Saurel, F. Petitpas, and R. Abgrall. Modelling phase transition in metastable liquids: application to cavitating and flashing flows. *Journal of Fluid Mechanics*, 607:313–350, 2008.

[209] R. Saurel, F. Petitpas, and R.A. Berry. Simple and efficient relaxation methods for interfaces separating compressible fluids, cavitating flows and shocks in multiphase mixtures. *Journal of Computational Physics*, 228:1678–1712, 2009.

[210] T. Schwartzkopff, M. Dumbser, and C.D. Munz. Fast high order ADER schemes for linear hyperbolic equations and their numerical dissipation and dispersion. Technical Report 2003/35, Preprint series of SFB404, Stuttgart University, 2003.

[211] T. Schwartzkopff, C.D. Munz, and E.F. Toro. ADER: A high order approach for linear hyperbolic systems in 2d. *Journal of Scientific Computing*, 17(1-4):231–240, 2002.

[212] D.W. Schwendeman, C.W. Wahle, and A.K. Kapila. The Riemann problem and a high-resolution Godunov method for a model of compressible two-phase flow. *Journal of Computational Physics*, 212:490–526, 2006.

[213] G. Scovazzi. Lagrangian shock hydrodynamics on tetrahedral meshes: A stable and accurate variational multiscale approach. *Journal of Computational Physics*, 231:8029–8069, 2012.

[214] M. Shashkov. Closure models for multimaterial cells in arbitrary Lagrangian-Eulerian hydrocodes. *International Journal for Numerical Methods in Fluids*, 56:1497–1504, 2008.

[215] J. Shi, C. Hu, and C.W. Shu. A technique of treating negative weights in WENO schemes. *Journal of Computational Physics*, 175:108–127, 2002.

[216] R.W. Smith. AUSM(ALE): a geometrically conservative arbitrary lagrangian-eulerian flux splitting scheme. *Journal of Computational Physics*, 150:268–286, 1999.

[217] T. Sonar. On the construction of essentially non-oscillatory finite volume approximations to hyperbolic conservation laws on general triangulations: polynomial recovery, accuracy and stencil selection. *Computer Methods in Applied Mechanics and Engineering*, 140:157–181, 1997.

[218] A.H. Stroud. *Approximate Calculation of Multiple Integrals*. Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1971.

[219] A. Suresh and H.T. Huynh. Accurate monotonicity-preserving schemes with runge-kutta time stepping. *Journal of Computational Physics*, 136:83–99, 1997.

[220] A. Taube, M. Dumbser, C.D. Munz, and R. Schneider. A High Order Discontinuous Galerkin Method with Local Time Stepping for the Maxwell Equations. *International Journal Of Numerical Modelling: Electronic Networks, Devices And Fields*, 22:77–103, 2009.

[221] B. Tian, E.F. Toro, and C.E. Castro. A path-conservative method for a five-equation model of two-phase flow with an hllc-type riemann solver. *Computers and Fluids*, 46:122–132, 2011.

[222] V.A. Titarev and E.F. Toro. ADER: Arbitrary high order Godunov approach. *Journal of Scientific Computing*, 17(1-4):609–618, December 2002.

[223] V.A. Titarev and E.F. Toro. ADER schemes for three-dimensional nonlinear hyperbolic systems. *Isaac Newton Institute for Mathematical Sciences Preprint Series*, 2003.

[224] V.A. Titarev and E.F. Toro. ADER schemes for three-dimensional nonlinear hyperbolic systems. *Journal of Computational Physics*, 204:715–736, 2005.

[225] V.A. Titarev, P. Tsoutsanis, and D. Drikakis. WENO schemes for mixed-element unstructured meshes. *Communications in Computational Physics*, 8:585–609, 2010.

[226] S.A. Tokareva and E.F. Toro. Hllc-type riemann solver for the baer-nunziato equations of compressible two-phase flow. *Journal of Computational Physics*, 229:3573–3604, 2010.

[227] E. F. Toro and V. A. Titarev. Derivative Riemann solvers for systems of conservation laws and ADER methods. *Journal of Computational Physics*, 212(1):150–165, 2006.

[228] E.F. Toro. Anomalies of conservative methods: analysis, numerical evidence and possible cures. *International Journal of Computational Fluid Dynamics*, 11:128–143, 2002.

[229] E.F. Toro. *Riemann Solvers and Numerical Methods for Fluid Dynamics: a Practical Introduction.* Springer, 2009.

[230] E.F. Toro, A. Hidalgo, and M. Dumbser. FORCE schemes on unstructured meshes I: Conservative hyperbolic systems. *Journal of Computational Physics*, 228:3368–3389, 2009.

[231] E.F. Toro, M. Spruce, and W. Speares. Restoration of contact surface in the HLL Riemann solver. *Shock Waves*, 4:25 – 34, 1994.

[232] E.F. Toro, M. Spruce, and W. Speares. Restoration of the contact surface in the Harten-Lax-van Leer Riemann solver. *Journal of Shock Waves*, 4:25–34, 1994.

[233] E.F. Toro and V. A. Titarev. Solution of the generalized Riemann problem for advection-reaction equations. *Proc. Roy. Soc. London*, pages 271–281, 2002.

[234] E.F. Toro and V.A. Titarev. ADER schemes for scalar hyperbolic conservation laws in three space dimensions. *Isaac Newton Institute for Mathematical Sciences Preprint Series*, 2003.

[235] I. Toumi. A weak formulation of Roe's approximate Riemann solver. *Journal of Computational Physics*, 102:360–373, 1992.

[236] P. Tsoutsanis, V.A. Titarev, and D. Drikakis. WENO schemes on arbitrary mixed-element unstructured meshes in three space dimensions. *Journal of Computational Physics*, 230:1585–1601, 2011.

[237] J. Utzmann, T. Schwartzkopff, M. Dumbser, and C.D. Munz. Heterogeneous Domain Decomposition for Computational Aeroacoustics. *AIAA Journal*, 44:2231–2250, 2006.

[238] J. J. W. van der Vegt and H. van der Ven. Space-time discontinuous Galerkin finite element method with dynamic grid motion for inviscid compressible flows I. general formulation. *Journal of Computational Physics*, 182:546–585, 2002.

[239] H. van der Ven and J. J. W. van der Vegt. Space-time discontinuous Galerkin finite element method with dynamic grid motion for inviscid compressible flows II. efficient flux quadrature. *Comput. Methods Appl. Mech. Engrg.*, 191:4747–4780, 2002.

[240] B. van Leer. Toward the ultimate conservative difference scheme. v. a second-order sequel to godunov's method. *Journal of Computational Physics*, 32:101 – 136, 1979.

[241] B. van Leer. Towards the ultimate conservative difference scheme V: A second order sequel to Godunov's method. *Journal of Computational Physics*, 32:101–136, 1979.

[242] J. von Neumann and R.D. Richtmyer. A method for the calculation of hydrodynamics shocks. *Journal of Applied Physics*, 21:232–237, 1950.

[243] P. Woodward and P. Colella. The numerical simulation of two-dimensional fluid flow with strong shocks. *Journal of Computational Physics*, 54:115–173, 1984.

[244] Y.V. Yanilkin, E.A. Goncharov, V.Y. Kolobyanin, V.V. Sadchikov, J.R. Kamm, M.J. Shashkov, and W.J. Rider. Multi-material pressure relaxation methods for lagrangian hydrodynamics. *Computers and Fluids*, 83:137–143, 2013.

[245] L. Del Zanna, N. Bucciantini, and P. Londrillo. An efficient shock-capturing central-type scheme for multidimensional relativistic flows II. magnetohydrodynamics. *Astronomy and Astrophysics*, 400:397–413, 2003.

[246] L. Del Zanna, O. Zanotti, N. Bucciantini, and P. Londrillo. ECHO: an Eulerian conservative high order scheme for general relativistic magnetohydrodynamics and magnetodynamics. *Astronomy and Astrophysics*, 473:11–30, 2007.

[247] A. Zein, M. Hantke, and G. Warnecke. Modeling phase transition for compressible two-phase flows applied to metastable liquids. *Journal of Computational Physics*, 229:2964–2998, 2010.

[248] T. Zhang and Y. Zheng. Conjecture on the structure of solutions of the Riemann problem for two-dimensional gas dynamics systems. *SIAM J. Math. Anal.*, 21:593–630, 1990.

[249] Y.T. Zhang and C.W. Shu. Third order WENO scheme on three dimensional tetrahedral meshes. *Communications in Computational Physics*, 5:836–848, 2009.

List of Tables

List of Figures