Mesh adaptation by continuous deformation. Basics: accuracy, efficiency, well balancedness

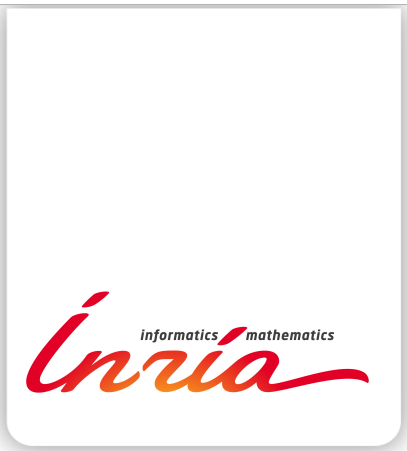
Luca Arpaia, Mario Ricchiuto

Mesh adaptation by continuous deformation. Basics: accuracy, efficiency, well balancedness

Luca Arpaia , Mario Ricchiuto

RESEARCH REPORT
N° 8666
January 2015
Project-Teams CARDAMOM

ISSN 0249-6399 ISRN INRIA/RR--8666--FR+ENG



Mesh adaptation by continuous deformation. Basics: accuracy, efficiency, well balancedness

Luca Arpaia *, Mario Ricchiuto†*

Project-Teams CARDAMOM

Research Report n° 8666 — January 2015 — 55 pages

Abstract: In this document we consider the construction of an adaptation technique based on continuous mesh deformation without re-meshing for conservation laws with source terms. The objective of the report is to analyse all the fundamental aspects of the process. These go from the choice of the appropriate ALE (Arbitrary Lagrangian Eulerian) of the PDE, allowing to preserve at the discrete level the appropriate steady state equilibrium, to the choice of the dicretization method, to the definition of the mesh deformation technique and finally of the coupling scheme-mesh deformation. For each step different alternatives are proposed and evaluated. Several improvements with repeat to the methods existing in literature are validated on properly chosen numerical benchmarks.

Key-words: mesh adaptation, continuous deformation, ALE formulation, source terms, well-balancedness

* Inria BSO, Team CARDAMOM
† Institut de Mathématiques de Bordeaux

RESEARCH CENTRE
BORDEAUX – SUD-OUEST

351, Cours de la Libération
Bâtiment A 29
33405 Talence Cedex

Mesh adaptation by continuous deformation: basics

Résumé : Dans ce document on considère la construction d'une méthode d'adaptation de maillage par deformation continue de la mesh et sans remaillage pour des lois de conservation avec termes de sources. Le but de ce rapport est d'analyser les aspects fondamentaux de la méthode qui vont de l'écriture et discrétisation de l'EDP dans une forme ALE (Arbitrary Lagrangian Eulerian) permettant de préserver des état d'équilibre stationnaire, à la technique de discretisation, aux aspects de deformation adaptative de maillage et finalement au couplage schéma-deformation. Pour chaque étape différentes alternatives sont proposée et évaluée. Plusierus améliorations par rapport aux mèthodes existantes sont validés par des example numériques.

Mots-clés : adaptation de maillage, deformation continue, formulation ALE, termes de sources, well-balancedness

Contents

1 ALE formulation : basic notation

In this second half of the document we deal with the numerical approximation of the following scalar hyperbolic non-homegeneous PDE called balance law

$$\frac{\partial u}{\partial t} + \nabla \cdot \mathbf{f}(u) + \mathbf{a} \cdot \nabla g = 0 \tag{1}$$

where the flux is defined as $\mathbf{f} = \mathbf{a}u$, $\mathbf{a} = \mathbf{a}(u)$ can be also defined as $\mathbf{a} = \frac{d\mathbf{f}}{du}$ and $g = g(\mathbf{x})$ is a given continuous function. Equation (1) admits an invariant $\eta = u + g$ resulting from the steady balance between the flux and the source terms. If u is interpreted as the water height in a lake and g as a sort of bathimetry, the above steady solution closely resembles the lake at rest solution for the Shallow Water system, namely $\eta = const$. Working on the simplified scalar equation we will show some properties of numerical schemes when trying to approximate the "lake at rest". While Residual Distribution (RD) schemes are able of preserving very naturally the above state, which in a Shallow Water context is called well-balanced or C-property, for the same RD approximation in Arbitary

Lagrangian Eulerian form (ALE) it is shown that this property is lost. However a simple change of variable permits to recover it.

Assuming that we are given a domain Ω and field of displacements that brings every point of the domain from the reference position $\mathbf{X}$ to the actual one $\mathbf{x}(t)$ and that this field is governed by an arbitrary given motion law

$$\frac{d\mathbf{x}(t)}{dt} = \sigma(\mathbf{x},t), \tag{2}$$

Solving the ODE (2) starting from the reference configuration $\mathbf{x}(0) = \mathbf{X}$, gives back, at every time instant, the actual configuration through the following mapping

$$A(t) : \Omega_X \to \Omega_x(t), \qquad \mathbf{x} = A(\mathbf{X},t) \tag{3}$$

We define the Jacobian matrix of the mapping as

$$\mathscr{J}_A = \frac{\partial \mathbf{x}}{\partial \mathbf{X}}, \quad J_A = \det \mathscr{J}_A \neq 0$$

We introduce now another set of coordinates, the Lagrangian or material coordinates χ, and a mapping that describes the motion of each particle. This mapping returns the physical location, rapresented by the actual coordinate $\mathbf{x}$, of the particle marked with χ at time t

$$B(t) : \Omega_\chi \to \Omega_x(t), \qquad \mathbf{x} = B(\chi,t)$$

Defining the Jacobian matrix of the mapping

$$\mathscr{J}_B = \frac{\partial \mathbf{x}}{\partial \chi} \quad J_B = \det \mathscr{J}_B \neq 0$$

If u is a conserved quantity it can be expressed as a function of the different coordinates $\mathbf{x}, \mathbf{X}, \chi$ and three different time derivatives can be defined. If the derivation is computed in the actual configuration one has

$$\left.\frac{\partial u(\mathbf{x},t)}{\partial t}\right|_x = \frac{\partial u}{\partial t} \qquad \text{spatial derivative}$$

If it is computed following the particle motion one has

$$\left.\frac{\partial u(\mathbf{x},t)}{\partial t}\right|_\chi = \frac{\partial u(B,t)}{\partial t} = \frac{du}{dt} \qquad \text{material derivative}$$

Finally if it is computed following the domain motion one has

$$\left.\frac{\partial u(\mathbf{x},t)}{\partial t}\right|_X = \frac{\partial u(A,t)}{\partial t} \qquad \text{referential derivative}$$

Moreover two different velocities can be computed: the domain velocity and the particle velocity

$$\left.\frac{\partial \mathbf{x}(t)}{\partial t}\right|_X = \frac{dA(\mathbf{X},t)}{dt} = \sigma \tag{4}$$

$$\left.\frac{\partial \mathbf{x}(t)}{\partial t}\right|_\chi = \frac{dB(\chi,t)}{dt} = \mathbf{a}$$

The transformation theorem provides a relation between the above derivatives and these velocities

$$\frac{du}{dt} = \frac{\partial u}{\partial t} + \mathbf{a}(\mathbf{x},t)\cdot\nabla u(\mathbf{x},t) \tag{5}$$

$$\frac{du}{dt} = \left.\frac{\partial u}{\partial t}\right|_X + (\mathbf{a}(\mathbf{x},t) - \sigma(\mathbf{x},t))\cdot\nabla u(\mathbf{x},t) \tag{6}$$

From continuum mechanics we need also the followings

$$\left.\frac{\partial J_B}{\partial t}\right|_\chi = \frac{dJ_B}{dt} = J_B\nabla\cdot\mathbf{a} \tag{7}$$

$$\left.\frac{\partial J_A}{\partial t}\right|_X = J_A\nabla\cdot\sigma \tag{8}$$

The last one is commonly called Geometric Conservation Law (GCL) and rapresents a constraint the points of the domain have to satisfy during their arbitrary motion. This will be very important when developing a numerical method with a moving grid; up to now we only want to make clear that the movement of the domain is arbitrary but within hypothesis (8).

2 Balance law in ALE form

The balance law of the scalar quantity u can be written, dependig on convenience, in the different coordinates frameworks. If we choose a material control volume $C(t)$ which contains always the same particles, following them throughout all the domain, the balance to express the growth/decay rates of u due to the effect of a generic source/sink S is simply stated in actual coordinates

$$\frac{d}{dt}\int_{C(t)} u(\mathbf{x},t)\,d\mathbf{x} + \int_{C(t)} S(u,\mathbf{x},t)\,d\mathbf{x} = 0 \tag{9}$$

Passing to material coordinates and using (7) together with the transformation theorem (5)

$$\begin{aligned}
\frac{d}{dt}\int_{C_\chi} u(\chi,t)J_B\,d\chi &= \int_{C_\chi}\frac{d}{dt}\left(u(\chi,t)J_B\right)d\chi \\
&= \int_{C_\chi}\left(\frac{du}{dt}J_B + u\frac{dJ_B}{dt}\right)d\chi \\
&= \int_{C_\chi}\left(\frac{\partial u}{\partial t} + \mathbf{a}\cdot\nabla u + u\nabla\cdot\mathbf{a}\right)J_B\,d\chi
\end{aligned} \tag{10}$$

We have derived the conservation law in integral Eulerian form

$$\int_C\left(\frac{\partial u}{\partial t} + \nabla\cdot\mathbf{f} + S\right)d\mathbf{x} = 0 \tag{11}$$

where the flux is defined as $\mathbf{f} = \mathbf{a}u$. Hereinafter we will assume that $\mathbf{a} = \mathbf{a}(u)$ hence the advective velocity can be also defined as $\mathbf{a}(u) = \frac{d\mathbf{f}(u)}{du}$. The above equation models a typical trasport-reaction problem where the transport of information occurs at local velocity $\mathbf{a}(u)$ while the source/sink term S implies the growth/decay of the same travelling information.

Now, in the intermediate passage (10), we use (6) instead of (5)

$$\begin{aligned}\int_{C_\chi}\left(\frac{du}{dt}J_B+u\frac{dJ_B}{dt}\right)d\chi &= \int_{C_\chi}\left(\left.\frac{\partial u}{\partial t}\right|_X+(\mathbf{a}-\sigma)\cdot\nabla u+u\nabla\cdot\mathbf{a}\right)J_B\,d\chi\\ &= \int_{C(t)}\left(\left.\frac{\partial u}{\partial t}\right|_X+\nabla\cdot\mathbf{f}-\sigma\cdot\nabla u\right)d\mathbf{x}\end{aligned}\tag{12}$$

The first term can be rewritten if we compute the derivative of the conserved quantity inside a control volume $C(t)$, which is following the motion of the points of the domain. Note that there is a little abuse in the notation since $C(t)$ has been already used to rapresent a material volume. Transforming into referential coordinate and using the fact that C_X does not depend on time

$$\begin{aligned}\left.\frac{\partial}{\partial t}\right|_X\int_{C(t)}u(\mathbf{x},t)\,d\mathbf{x} &= \left.\frac{\partial}{\partial t}\right|_X\int_{C_X}u(\mathbf{X},t)J_A\,d\mathbf{X}=\int_{C_X}\left.\frac{\partial\,(J_Au)}{\partial t}\right|_X d\mathbf{X}\\ &= \int_{C_X}\left.\frac{\partial u}{\partial t}\right|_X J_A\,d\mathbf{X}+\int_{C_X}\left.\frac{\partial J_A}{\partial t}\right|_X u\,d\mathbf{X}\\ &= \int_{C_X}\left.\frac{\partial u}{\partial t}\right|_X J_A\,d\mathbf{X}+\int_{C_X}J_Au\nabla\cdot\sigma\,d\mathbf{X}\end{aligned}\tag{13}$$

So we have

$$\int_{C(t)}\left.\frac{\partial u}{\partial t}\right|_X d\mathbf{x}=\left.\frac{\partial}{\partial t}\right|_X\int_{C(t)}u\,d\mathbf{x}-\int_{C(t)}u\nabla\cdot\sigma\,d\mathbf{x}\tag{14}$$

Substituting (14) in (12) we get the integral form of conservation law written in Arbitrary Lagrangian Eulerian Formulation (ALE)

$$\left.\frac{\partial}{\partial t}\right|_X\int_{C(t)}u\,d\mathbf{x}+\int_{C(t)}\nabla\cdot(\mathbf{f}-u\sigma)\,d\mathbf{x}+\int_{C(t)}S\,d\mathbf{x}=0\tag{15}$$

which express the conservation of u contained in a control volume which is moving arbitrarly. The equilibrium is reached by the relative flux of u entering and leaving the volume with velocity $\mathbf{a}-\sigma$.

A differential form of conservation law in ALE formulation is needed but its derivation is simple if we start from the integral form (15)

$$\begin{aligned}&\left.\frac{\partial}{\partial t}\right|_X\int_{C_X}uJ_A\,d\mathbf{X}+\int_{C_X}J_A\nabla\cdot(\mathbf{f}-u\sigma)\,d\mathbf{X}+\int_{C_X}J_AS\,d\mathbf{X}=0\\ &=\int_{C_X}\left(\left.\frac{\partial\,(J_Au)}{\partial t}\right|_X+J_A\nabla\cdot(\mathbf{f}-u\sigma)+J_AS\right)d\mathbf{X}=0\end{aligned}$$

Using the localization principle, the differential form of balance law in ALE formulation is derived

$$\left.\frac{\partial\,(J_Au)}{\partial t}\right|_X+J_A\nabla\cdot(\mathbf{f}-u\sigma)+J_AS=0\tag{16}$$

It is easy to see that the requirement for volume conservation (8) can be derived simply by imposing a state of uniform flow in the homegeneous part of (16). In this case we are modelling a situation in which the flow is uniform and the domain is moving from behind.

Developing the derivative in (16) and then substituting (8)

$$J_A\left.\frac{\partial u}{\partial t}\right|_X+J_Au\nabla\cdot\sigma+J_A\nabla\cdot(\mathbf{f}-u\sigma)+J_AS=0$$

which lead to the following equation that we will use extensively hereinafter

$$\left.\frac{\partial u}{\partial t}\right|_X+\nabla\cdot\mathbf{f}-\sigma\cdot\nabla u+S=0\tag{17}$$

2.1 "Lake at rest" solution

We are interested in a very particular source term

$$S(u,\mathbf{x},t) = \mathbf{a}(u)\cdot\nabla g(\mathbf{x})$$

where $g = g(\mathbf{x})$ is an arbitrary given continuous function. In the steady case we reduce to a balance between the flux and the source term

$$\nabla\cdot\mathbf{f}(u) + \mathbf{a}(u)\cdot\nabla g = 0 \tag{18}$$

which can be solved with the change of varibale $\eta = u + g$.

$$\nabla\eta = 0 \Rightarrow \eta = \eta_0 \tag{19}$$

We are interested in these kind of solutions becouse, if u is considered as the water depth, g is the bathimetry, η is the total water height, the above steady state rapresents the well known lake at rest solution for the Shallow Water system. When designing a good numerical method for the Shallow Water equations it is very important to ensure that the same scheme could verify the balance (18). Hereinafter we will refer to Well Balanced property as the ability, for a scalar numerical scheme, to reproduce exactly (19). As before, the term "exactly" means that the numerical error is driven by the quadrature error carried out when computing the integrals.

2.2 ALE form with the change of variable

The ALE formulation of the scalar equation presented in the previous section is obtained for both the variables: the conservative variable u and the invariant η. For a conserved quantity u we have already seen that one can use directly equations (15), (16) or (17). We end up respectively with the integral balance law, the differential balance law and the differential nonconservative form. We recall them for clarity

$$\left.\frac{\partial}{\partial t}\right|_X \int_{C(t)} u\,d\mathbf{x} + \int_{C(t)} \nabla\cdot(\mathbf{f}-u\boldsymbol{\sigma})\,d\mathbf{x} + \int_{C(t)} \mathbf{a}(u)\cdot\nabla g\,d\mathbf{x} = 0$$

$$\left.\frac{\partial\,(J_A u)}{\partial t}\right|_X + J_A\nabla\cdot(\mathbf{f}-u\boldsymbol{\sigma}) + J_A\mathbf{a}(u)\cdot\nabla g = 0 \tag{20}$$

$$\left.\frac{\partial u}{\partial t}\right|_X + \nabla\cdot\mathbf{f} - \boldsymbol{\sigma}\cdot\nabla u + \mathbf{a}(u)\cdot\nabla g = 0 \tag{21}$$

To obtain the ALE form in η we start again from the lagrangian balance (9), but we have to make a change of variable in the intermediate passage (12)

$$\int_{C(t)} \left(\left.\frac{\partial\eta}{\partial t}\right|_X - \left.\frac{\partial g}{\partial t}\right|_X + \nabla\cdot\mathbf{f} - \boldsymbol{\sigma}\cdot\nabla u + \mathbf{a}\cdot\nabla g \right) d\mathbf{x} = 0 \tag{22}$$

Now, repeating passages (13), we also have that

$$\int_{C(t)} \left.\frac{\partial\eta}{\partial t}\right|_X d\mathbf{x} = \left.\frac{\partial}{\partial t}\right|_X \int_{C(t)} \eta\,d\mathbf{x} - \int_{C(t)} \eta\nabla\cdot\boldsymbol{\sigma}\,d\mathbf{x} \tag{23}$$

Moreover using the chain rule to derivate g in time

$$\left.\frac{\partial g(\mathbf{x})}{\partial t}\right|_X = \left.\frac{\partial\mathbf{x}(t)}{\partial t}\right|_X \cdot\nabla g$$

with the formula (4) we get

$$\left.\frac{\partial g}{\partial t}\right|_X = \sigma \cdot \nabla g \tag{24}$$

This last equation rapresents the time variation of the bathimetry measured from an observer which is following the domain motion at local velocity $\sigma(\mathbf{x})$. Substituting results (23) and (24) in equation (22), one gets

$$\left.\frac{\partial}{\partial t}\right|_X \int_{C(t)} \eta \, d\mathbf{x} + \int_{C(t)} \nabla \cdot (\mathbf{f} - u\sigma - g\sigma) \, d\mathbf{x} + \int_{C(t)} \mathbf{a} \cdot \nabla g \, d\mathbf{x} = 0$$

Which can be rewritten compactly in the integral balance law for η in ALE framework

$$\left.\frac{\partial}{\partial t}\right|_X \int_{C(t)} \eta \, d\mathbf{x} + \int_{C(t)} \nabla \cdot (\mathbf{f} - \eta\sigma) \, d\mathbf{x} + \int_{C(t)} \mathbf{a} \cdot \nabla g \, d\mathbf{x} = 0$$

It is important to underline the fact the flux still depends on u, hence we have to make the correct change of variable $\mathbf{f} = \mathbf{f}(u) = \mathbf{f}(\eta - g)$ and $\mathbf{a} = \mathbf{a}(u) = \mathbf{a}(\eta - g)$. The differential forms are easily obtained. First we present the conservative differential balance law for η in ALE framework

$$\left.\frac{\partial (J_A \eta)}{\partial t}\right|_X + J_A \nabla \cdot (\mathbf{f} - \eta\sigma) + J_A \mathbf{a} \cdot \nabla g = 0 \tag{25}$$

Then the nonconservative form

$$\left.\frac{\partial \eta}{\partial t}\right|_X + \nabla \cdot \mathbf{f} - \sigma \cdot \nabla \eta + \mathbf{a} \cdot \nabla g = 0 \tag{26}$$

3 Residual Distribution approximation for balance laws

3.1 Steady case

Consider the steady scalar balance between the flux of u and a continous source term S, this time expressed in Eulerian framework

$$\nabla \cdot \mathbf{f}(u) + S(u, \mathbf{x}) = 0, \quad \mathbf{x} \in \Omega \tag{27}$$

A numerical approximation of the advective and of the source terms in the simpler steady eulerian case is now performed. However, for more complex cases such as unsteady problems in ALE framework, the treatment of these two terms will be similar. Once we have approximated the domain through a suitable triangulation $\mathscr{T}_h$ we propose directly a Residual Distribution approximation of (27). Boundary condition are neglected at this point and we imagine that every element does not share any edge with the domain boundary.

1. With a piecewice linear approximation of the solution and of the source term over each triangle

$$u_h(\mathbf{x}, t) = \sum_{j \in \mathscr{T}_h} \varphi_j(\mathbf{x}) u_j(t) \tag{28}$$

$$S_h(\mathbf{x}) = \sum_{j \in \mathscr{T}_h} \varphi_j(\mathbf{x}) S_j \tag{29}$$

the element residual is computed as

$$\phi^K = \int_K \nabla \cdot \mathbf{f}(u_h) \, d\mathbf{x} + \int_K S_h \, d\mathbf{x} = \int_{\partial K} \mathbf{f}(u_h) \cdot \mathbf{n} \, ds + \int_K S_h \, d\mathbf{x} \tag{30}$$

2. Distribute the residual to the nodes of the element $i,j,k \in K$ through weights that sum up to one for consistency
$$\phi^K = \beta_i^K \phi^K + \beta_j^K \phi^K + \beta_k^K \phi^K = \sum_{j\in K} \phi_j^K \tag{31}$$
with
$$\beta_i^K + \beta_j^K + \beta_k^K = 1$$

3. Assembly the residuals shared by the same node. If $\mathscr{D}_i$ is the domain formed by all the elements of the triangulation that have in common node i, we have
$$\sum_{K\in\mathscr{D}_i} \beta_i^K \phi^K = \sum_{K\in\mathscr{D}_i} \phi_i^K = 0, \quad \forall i \in \mathscr{T}_h \tag{32}$$

Linearization

Being conservative is a delicate issue for every numerical method approximating conservation laws. It means that the numerical solution satisfies the integral form of conservation laws, mimicking what the exact solution does. Summing (32) over all the elements, using consistency condition (31), (30) and the fact that, for a conservative scheme, fluxes cancel out except at the boundaries, we have that
$$\begin{aligned}\sum_{i\in\mathscr{T}_h}\sum_{K\in\mathscr{D}_i} \phi_i^K &= \sum_{K\in\mathscr{T}_h}\sum_{j\in K} \phi_j^K = \sum_{K\in\mathscr{T}_h} \left(\int_{\partial K} \mathbf{f}(u_h)\cdot\mathbf{n}\, ds + \int_K S_h\, d\mathbf{x} \right) \\ &= \int_{\partial\Omega_h} \mathbf{f}(u_h)\cdot\mathbf{n}\, ds + \int_{\Omega_h} S_h\, d\mathbf{x}\end{aligned}$$
which states that, imposing correctly boundary conditions and giving a correct approximation S_h of S, we respect the integral balance (27) over the full domain. We can reassume that a RD method is conservative if it is consistent and we are able to approximate the advective part of the residual with a sufficient degree of precision
$$\phi^{K,adv} = \int_{\partial K} \mathbf{f}(u_h)\cdot\mathbf{n}\, ds$$
This has been interpreted as a constraint on the linearization. To be sure that we are computing correctly the residual, the passage from a conservative form to a linearized one should be exact
$$\begin{aligned}\phi^{K,adv} &= \int_K \nabla\cdot\mathbf{f}(u_h)\, d\mathbf{x} = \int_K \mathbf{a}(u_h)\, d\mathbf{x}\cdot\nabla u_h \\ &= \bar{\mathbf{a}}\cdot\nabla u_h |K|\end{aligned} \tag{33}$$
where
$$\bar{\mathbf{a}} = \frac{1}{|K|}\int_K \mathbf{a}(u_h)\, d\mathbf{x}$$
The approximation of the source integral does not pose particular problems connected to the scheme conservativness.

An important result, which will be used extensively, is the following definition of the linearized residual. From the definition of the gradient of a P1 solution over the element $\nabla u_h = \frac{1}{2|K|}\sum_{j\in K}\mathbf{n}_j u_j$, taking advantage of the fact that S_h is piecewise linear we have that
$$\begin{aligned}\phi^K &= \bar{\mathbf{a}}\cdot\nabla u^h |K| + \sum_{j\in K}\frac{|K|}{3} S_j = \frac{1}{2}\sum_{j\in K}\bar{\mathbf{a}}\cdot\mathbf{n}_j u_j + \sum_{j\in K}|K|\bar{S}_j \\ &= \sum_{j\in K} k_j u_j + \sum_{j\in K}|K|\bar{S}_j\end{aligned} \tag{34}$$

We have introduced the upwind parameter

$$k_i = \frac{1}{2}\bar{\mathbf{a}} \cdot \mathbf{n}_i$$

Equivalently

$$\phi^K = \sum_{j \in K, j \neq i} k_j (u_j - u_i) + \sum_{j \in K} |K| \bar{S}_j$$

3.2 Unsteady case in ALE form

In this paper we are interested in unsteady phenomena, moreover we are interested in unsteady phenomena resolved on moving grids. For this reason we present the scalar balance law in ALE form as it has been derived in (16)

$$\left.\frac{\partial (J_A u)}{\partial t}\right|_X + J_A \nabla \cdot (\mathbf{f} - u\boldsymbol{\sigma}) + J_A S = 0$$

While the advective and the source part has been treated in the previous section, nothing has been said about the time part, infact its approximation within a Residual Distribution method is not a trivial task and has challanged many researcher during the last decade. Here we use the full analogy that exists between Finite Elements and Residual Distribution: omitting the details, it is possible to build a RD approximation of an unsteady balance law, from a Stabilized FE one.

3.2.1 Space approximation: Stabilized FE analogy

We proceed in the construction of a Stabilized FE approximation of (16) choosing an approximation of the solution in the space of piecewise linear polynomials $u \in X_h^1$ and the test function w as

$$w = \varphi + \gamma$$

where $\varphi(\mathbf{x})$ is the Galerkin shape function, and γ_i, called bubble function, belongs to some other functionl space different from X_h^1. Giving also the following piecewise linear approximation of the grid velocity

$$\boldsymbol{\sigma}_h(\mathbf{x},t) = \sum_{j \in \mathcal{T}_h} \varphi_j(\mathbf{x}) \boldsymbol{\sigma}_j(t)$$

the approximate weak form reads

$$\int_{\Omega_h^X} w_i \left(\left.\frac{\partial (J_A u_h)}{\partial t}\right|_X + J_A \nabla \cdot (\mathbf{f}(u_h) - \boldsymbol{\sigma}_h u_h) + J_A S_h \right) d\mathbf{X} = 0$$

Since the the configuration Ω_X does not depend on time and assuming $\left.\frac{\partial w_i}{\partial t}\right|_X = 0$ we can take the time derivative outside the integral

$$\left.\frac{\partial}{\partial t}\right|_X \int_{\Omega_h^X} w_i J_A u_h \, d\mathbf{X} + \int_{\Omega_X} w_i J_A \nabla \cdot (\mathbf{f}(u_h) - \boldsymbol{\sigma}_h u_h) \, d\mathbf{X} + \int_{\Omega_X} w_i J_A S_h \, d\mathbf{X} = 0$$

Passing to the current coordinates $\mathbf{x}$ we have the Petrov-Galerkin approximation for (16)

$$\left.\frac{\partial}{\partial t}\right|_X \int_{\Omega_h(t)} w_i u_h \, d\mathbf{x} + \int_{\Omega_h(t)} w_i \nabla \cdot (\mathbf{f}(u_h) - \boldsymbol{\sigma}_h u_h) \, d\mathbf{x} + \int_{\Omega_h(t)} w_i S_h \, d\mathbf{x} = 0 \tag{35}$$

3.2.2 Time approximation and GCL: Explicit Euler

At this point the time derivative of (123) is discretized with the simpliest approximation, an Explicit Euler scheme (EE). We have

$$\frac{\Delta}{\Delta t}\int_{\Omega_h(t)} w_i u_h \, d\mathbf{x} + \int_{\Omega_h(t)} w_i \nabla\cdot(\mathbf{f}(u_h^n) - \sigma_h u_h^n)\, d\mathbf{x} + \int_{\Omega_h(t)} w_i S_h^n \, d\mathbf{x} = 0 \tag{36}$$

The source term is treated, using (29), consistently with the scheme

$$S_h^n = S_h(u^n, \mathbf{x}(t^n), t^n)$$

Before moving on, we face the fact that the mesh velocity and the time instant when we have to compute the space integrals of the advective and of source terms are still undefined: the necessity to fix both of them, σ_h and $\Omega_h(t)$, arises.

Let's recall what we called the GCL condition (8) for the domain motion. It is naturally to ask, also for the numerical scheme, a similar property, hence a law that the grid has to satisfy during its arbitrary motion. This is referred to as Discrete Geometric Conservation Law (DGCL). In a similar manner such as (8), it is easy to prove that the DGCL condition is obtained imposing a uniform solution in the equivalent time-space discretized homegeneous problem.

Imposing $S_h = 0$ and a uniform flow in (36) one gets

$$\int_{\Omega_h^{n+1}} w_i \, d\mathbf{x} - \int_{\Omega_h^n} w_i \, d\mathbf{x} = \Delta t \int_{\Omega_h(t)} w_i \nabla\cdot\sigma_h \, d\mathbf{x} \tag{37}$$

Farhat [1] demonstrates the following property.

Property 3.2.2 (DGCL). *A numerical method in the form (36) satisfy the DGCL if the following choice is made*

$$\sigma_h = \sigma_h^* = \frac{\mathbf{x}_j^{n+1} - \mathbf{x}_j^n}{\Delta t} \tag{38}$$

$$\Omega_h(t) = \Omega_h(t^{n+1/2}) = \Omega_h^{n+1/2}$$

Proof. With such a choice the DGCL condition, equation (37), is an identity.

Finally, the scheme we are going to present is capable of preserving a uniform solution "exactly", where exactly means that the numerical error is driven by the quadrature error done in the numerical integration step.

Going back, we can develop (36) as

$$\int\limits_{\Omega_h^{n+1}} w_i u_h^{n+1}\, d\mathbf{x} - \int\limits_{\Omega_h^n} w_i u_h^n \, d\mathbf{x} + \Delta t \int\limits_{\Omega_h^{n+1/2}} w_i \left(\nabla\cdot(\mathbf{f}(u_h^n) - \sigma_h^* u_h^n) + S_h^n\right) d\mathbf{x} = 0 \tag{39}$$

Note that the source term does not enter in the GCL balance and it is not necessary to compute its integral at midpoint configuration. However, for compactness, it is treated as the advective part. We can split the ALE flux

$$\frac{\Delta}{\Delta t}\int\limits_{\Omega(t)} w_i u^h \, d\mathbf{x} + \int\limits_{\Omega^{n+1/2}} w_i \left(\nabla\cdot\mathbf{f}(u_h^n) - \sigma_h^*\cdot\nabla u_h^n + S_h^n\right) d\mathbf{x} - \int\limits_{\Omega^{n+1/2}} w_i u_h^n \nabla\cdot\sigma_h^* \, d\mathbf{x} = 0 \tag{40}$$

We recall the following property which come from the fact that, given two quantity A,B, then $\Delta(AB) = \overline{B}\Delta A + \overline{A}\Delta B$

$$\Delta|_n^{n+1}\left(\int_{\Omega(t)} w_i u^h \, d\mathbf{x}\right) = \tag{41}$$

$$= \int_{\Omega^{n+1/2}} w_i \left(u_h^{n+1} - u_h^n\right) d\mathbf{x} + \Delta t \int_{\Omega^{n+1/2}} w_i \frac{\left(u_h^{n+1} + u_h^n\right)}{2} \nabla \cdot \boldsymbol{\sigma}_h^* \, d\mathbf{x}$$

Second, we substitute (41) in (40) and we sum the last term of the above equation with the last one in (40)

$$\left(1 + \frac{\Delta t}{2} \nabla \cdot \boldsymbol{\sigma}_h^*\right) \int_{\Omega^{n+1/2}} w_i \left(u_h^{n+1} - u_h^n\right) d\mathbf{x} + \Delta t \int_{\Omega^{n+1/2}} w_i \left(\nabla \cdot \mathbf{f}(u_h^n) - \boldsymbol{\sigma}_h^* \cdot \nabla u_h^n + S_h^n\right) d\mathbf{x} = 0$$

The second term can be recasted in a RD form within the hypothesis of linearization. If this is true, defining the weight coefficients

$$\frac{1}{|K|} \int_K w_i \, d\mathbf{x} = \beta_i^K \tag{42}$$

and the total residual

$$\phi^K = \int_{K^{n+1/2}} \left(\nabla \cdot \mathbf{f}(u_h^n) - \boldsymbol{\sigma}_h^* \cdot \nabla u_h^n + S_h^n\right) d\mathbf{x}$$

we can exploit a full analogy between FE and RD. Developing the mass matrix with mass lumping we obtain

$$\sum_{K \in \mathscr{D}_i} \left(1 + \frac{\Delta t}{2} \nabla \cdot \boldsymbol{\sigma}_h^*\right) \frac{|K^{n+1/2}|}{3} \left(u_i^{n+1} - u_i^n\right) = -\Delta t \sum_{K \in \mathscr{D}_i} \beta_i \phi^K$$

The final RD algorithm reads

$$u_i^{n+1} = u_i^n - \frac{\Delta t}{|\tilde{S}_i^{n+1/2}|} \sum_{K \in \mathscr{D}_i} \phi_i^K \tag{43}$$

where the median dual cell area, evaluated at midpoint configuration, has to be modified to take into account the grid distortion

$$|\tilde{S}_i^{n+1/2}| = \sum_{K \in \mathscr{D}_i} \left(1 + \frac{\Delta t}{2} \nabla \cdot \boldsymbol{\sigma}_h^*\right) \frac{|K^{n+1/2}|}{3} \tag{44}$$

The total residual is enhanced by a new term due to the ALE part. Since the grid velocity is approximated with a P1 interpolation, this last term is easily linearized in a conservative manner

$$\int_K \boldsymbol{\sigma}_h^* \cdot \nabla u^h \, d\mathbf{x} = \sum_{j \in K} \frac{\boldsymbol{\sigma}_j^*}{3} \nabla u^h |K| = \bar{\boldsymbol{\sigma}} \nabla u^h |K|$$

Adding the ALE term in (34), the total residual is then computed as

$$\phi^K = \sum_{j \in K} k_j^{ALE} u_j + \sum_{j \in K} \frac{|K|^{n+1/2}}{3} S_j^n \tag{45}$$

The upwind parameter with the ALE correction naturally becomes

$$k_i^{ALE} = \frac{1}{2} \left(\bar{\mathbf{a}} - \bar{\sigma}\right) \cdot \mathbf{n}_i^{n+1/2} \tag{46}$$

The method satisfy the DGCL by construction but it is extremely easy to prove this again, by simply imposing a uniform state in the method presented so far. Unfortunately this scheme converges only with first order of accuracy. To achieve second order accuracy and stay explicit we can use the explicit Runge-Kutta presented in Sec.(5).

3.3 Properties of the RD discretisation

The RD scheme (43) can be reformalized in the following abstract form

$$u_i^{n+1} = u_i^n - \frac{\Delta t}{|\tilde{S}_i^{n+1/2}|}\left(\sum_{K\in\mathscr{D}_i}\sum_{j\in K, j\neq i} c_{ij}^K (u_i - u_j) + \sum_{K\in\mathscr{D}_i}\sum_{j\in K} |K^{n+1/2}| c_{ij}^S S_j\right) \tag{47}$$

Proof. Substituting (45) in (43)

$$\begin{aligned} u_i^{n+1} &= u_i^n - \frac{\Delta t}{|\tilde{S}_i^{n+1/2}|}\left(\sum_{K\in\mathscr{D}_i}\beta_i^K\left(\sum_{j\in K} k_j^{ALE} u_j + \sum_{j\in K}\frac{|K|^{n+1/2}}{3} S_j\right)\right) = \\ &= u_i^n - \frac{\Delta t}{|\tilde{S}_i^{n+1/2}|}\left(\sum_{K\in\mathscr{D}_i}\sum_{j\in K, j\neq i}\left(\beta_i^K k_j^{ALE}\right)(u_i - u_j) + \sum_{K\in\mathscr{D}_i}\sum_{j\in K}|K|^{n+1/2}\left(\frac{\beta_i^K}{3}\right) S_j\right) \end{aligned}$$

we get the proof.

Form (47) is referred to as the compact prototype scheme. In the next paragraphs we will study the properties of schemes that can be recasted in the form (47).

3.3.1 Stability

Differently from the homogenous case, a balance law with a source term does not admit a maximum principle. Ricchiuto in [2] searches for uniform bounds on the numerical solution in order to obtain L^∞-stability

$$u_j^{n+1} = \mathscr{H}(u^n; j), \quad \|\mathscr{H}\|_{L^\infty(\Omega)} \leq C_s$$

We start giving the following result:

Property 3.3.1.1 (L^∞ bounds). *The numerical solution of a scheme in the form (47) respects the following bounds*

$$\min_{j\in\mathscr{D}_i} u_j^n + \Delta t \sum_{j\in\mathscr{D}_i} C_{ij}^S \min_{j\in\mathscr{D}_i} S_j^n \leq u_i^{n+1} \leq \max_{j\in\mathscr{D}_i} u_j^n + \Delta t \sum_{j\in\mathscr{D}_i} C_{ij}^S \max_{j\in\mathscr{D}_i} S_j^n \tag{48}$$

if the following conditions are verified:

- *the scheme is LED, hence*

$$\tilde{c}_{ij}^K \geq 0 \quad \forall j \in \mathscr{D}_i,\ j\neq i \quad \text{and} \quad \forall i \in \mathscr{T}_h \tag{49}$$

  with $\tilde{c}_{ij}^K$ defined in (53)

- *the scheme verifies a CFL-like condition, hence there is a time step restiction*

$$\Delta t \leq \frac{|S_i^{n+1/2}|}{\sum_{j\in\mathscr{D}_i, j\neq i}\tilde{c}_{ij}^K} \quad \forall i \in \mathscr{T}_h \tag{50}$$

- *$S(\mathbf{x})$ is uniformly bounded and so it is C_{ij}^S*

$$\sup_\Omega |S(\mathbf{x})| < \infty, \quad C_{ij}^S < \infty \quad \forall i,j \in K \tag{51}$$

The definition of C_{ij}^S is given in the demonstration.

Proof. We can obtain the proof starting directly from (47)

$$\begin{aligned} u_i^{n+1} &= u_i^n - \frac{\Delta t}{|\tilde{S}_i^{n+1/2}|}\left(\sum_{K\in\mathscr{D}_i}\sum_{j\in K, j\neq i} c_{ij}^K(u_i-u_j) + \sum_{K\in\mathscr{D}_i}\sum_{j\in K}|K^{n+1/2}|c_{ij}^S S_j\right) \\ &= u_i^n - \frac{\Delta t}{|\tilde{S}_i^{n+1/2}|}\left(\sum_{j\in\mathscr{D}_i, j\neq i}\left(\sum_{K\in\mathscr{D}_i\cap\mathscr{D}_j} c_{ij}^K\right)(u_i-u_j) + \sum_{j\in\mathscr{D}_i}\left(\sum_{K\in\mathscr{D}_i\cap\mathscr{D}_j}|K^{n+1/2}|c_{ij}^S\right)S_j\right) \end{aligned}$$

In order to lighten the notation we define the terms in brackets

$$\tilde{c}_{ij}^K = \sum_{K\in\mathscr{D}_i\cap\mathscr{D}_j} c_{ij}^K \tag{52}$$

$$\tilde{c}_{ij}^S = \sum_{K\in\mathscr{D}_i\cap\mathscr{D}_j} |K^{n+1/2}|c_{ij}^S \tag{53}$$

We get

$$\begin{aligned} u_i^{n+1} &= \left(1 - \frac{\Delta t}{|\tilde{S}_i^{n+1/2}|}\sum_{j\in\mathscr{D}_i, j\neq i}\tilde{c}_{ij}^K\right)u_i^n + \frac{\Delta t}{|\tilde{S}_i^{n+1/2}|}\sum_{j\in\mathscr{D}_i, j\neq i}\tilde{c}_{ij}^K u_j^n + \frac{\Delta t}{|\tilde{S}_i^{n+1/2}|}\sum_{j\in\mathscr{D}_i}\tilde{c}_{ij}^S S_j \\ &= C_{ii}u_i^n + \sum_{j\in\mathscr{D}_i, j\neq i} C_{ij}u_j^n + \sum_{j\in\mathscr{D}_i} C_{ij}^S S_j^n = \sum_{j\in\mathscr{D}_i} C_{ij}u_j^n + \Delta t\sum_{j\in\mathscr{D}_i} C_{ij}^S S_j^n \end{aligned}$$

LED property (92) ensures that $C_{ij}\geq 0$, while the CFL-like condition (93) ensure that also $C_{ii}\geq 0$, hence we can maximize and minimize the right hand side

$$\begin{aligned} u_i^{n+1} &\geq \sum_{j\in\mathscr{D}_i} C_{ij}\min_{j\in\mathscr{D}_i} u_j^n + \Delta t\sum_{j\in\mathscr{D}_i} C_{ij}^S \min_{j\in\mathscr{D}_i} S_j^n \\ u_i^{n+1} &\leq \sum_{j\in\mathscr{D}_i} C_{ij}\max_{j\in\mathscr{D}_i} u_j^n + \Delta t\sum_{j\in\mathscr{D}_i} C_{ij}^S \max_{j\in\mathscr{D}_i} S_j^n \end{aligned}$$

with the obvious fact that $\sum_{j\in\mathscr{D}_i} C_{ij} = 1$, we get the proof.

Once we have the bounds (48) for all time steps t^n and for all nodes of the domain i we get the following stabilty result.

Property 3.3.1.2 (L^∞-Stability). *If the local bounds (48) hold in all time slabs $[t^n, t^{n+1}]$, $n = 0,\ldots,M-1$ then the scheme (47) is L^∞-stable and the following bounds hold for its numerical solution*

$$\min_{i\in\mathscr{T}_h} u_i^0 + t^n \min_{i\in\mathscr{T}_h,\, n}\left(\sum_{j\in\mathscr{D}_i} C_{ij}^S \min_{j\in\mathscr{D}_i} S_j^n\right) \leq u_i^{n+1} \leq \max_{i\in\mathscr{T}_h} u_i^0 + t^n \max_{i\in\mathscr{T}_h,\, n}\left(\sum_{j\in\mathscr{D}_i} C_{ij}^S \max_{j\in\mathscr{D}_i} S_j^n\right)$$

The numerical solution is bounded by the sum of initial solution and the contribution of the source term. With such bounds the numerical solution of a linear scheme could overcome or drop the initial bounds only due to the growth/decay rates dictated by the source term, thus the discretization of the advective part does not give rise to oscillations near discontinuities. A nice criteria to avoid oscillation has been obtained.

3.3.2 Accuracy

Now we provide some criteria to define if a scheme returns second order accurate solutions for the steady problem

$$\nabla\cdot\mathbf{f} + S = 0$$

The interests in steady problems lies in the fact that one can obtain second order accurate solutions with the compact prototype scheme (47), of course setting $\sigma = 0$, otherwise it is impossible to have a steady solution.

At steady state we want second order of accuracy in some norm V

$$\|u - u_h\|_V = \mathscr{O}(h^2)$$

We give the following result whose demontration is contained in [2].

Property 3.3.2.1 (Second Order Accuracy). *An Eulerian RD scheme in the form (47), hence with $\sigma = 0$, produce a second order accurate solution at steady state if*

$$\phi_i^K = \mathscr{O}(h^3) \quad \forall K \in \mathscr{T}_h \text{ and } \forall i \in K \tag{54}$$

for a continuous second order accurate approximation of the fluxes $\mathbf{f}_h$ and of source term S_h.

Moreover, under the same hypothesis of the above property, the following estimate for the residual holds

$$\phi^K = \mathscr{O}(h^3) \tag{55}$$

Proof. For the linear case the residual can be written

$$\begin{aligned}
\phi^K &= \int_K (\mathbf{a} \cdot \nabla u_h + S_h)\, d\mathbf{x} \\
&= \int_K \nabla \cdot \mathbf{a} u_h \, d\mathbf{x} + \int_K S_h \, d\mathbf{x} \\
&= \int_K \nabla \cdot (\mathbf{a} u_h - \mathbf{a} u)\, d\mathbf{x} + \int_K (S_h - S)\, d\mathbf{x} \\
&= \int_{\partial K} (\mathbf{a} u_h - \mathbf{a} u) \cdot \mathbf{n}\, dl + \int_K (S_h - S)\, d\mathbf{x} \\
&= \mathscr{O}(h^3)
\end{aligned}$$

If u_h can be interpreted as the solution of the Petrov-Galerkin weak form then, in case of P1 approximation and smooth solution the following estimate holds

$$\|u_h - u\|_{L^2(\Omega)} = \mathscr{O}(h^2) \tag{56}$$

The following estimate is also true

$$\mathbf{a}\,(u_h - u) = \mathscr{O}(u_h - u)$$

because $\mathbf{a}$ is bounded. Moreover $dl = \mathscr{O}(h)$. For a P1 approximation of the source term the result then is prooved.

Now the fundamental relation $\phi_i^K = \beta_i \phi^K$, together with (54)(55), lead to the following result

Property 3.3.2.2 (Linearity Preserving scheme). *A RD scheme is linearity preserving if the distribution coefficients are uniformly bounded with respect to the solution and data of the problem, hence exists a constant C such that*

$$\max_{K \in \mathscr{T}_h} \max_{j \in K} |\beta_j^K| < C \quad \forall u_h, \mathbf{a}, u_h^0 \tag{57}$$

A scheme which is linearity preserving is second order accurate at steady state.

Even if the results given are valid only for steady state, in Sec.(5) we will use the same linearity preserving schemes, together with a Runge Kutta two time integrator, to obtain second order accurate numerical solution for unsteady problems in ALE framework.

3.3.3 Well Balancedness : Eulerian RD

A well known property of Eulerian RD schemes is that they verify very naturally the Well Balanced property defined in section (2.1).

Property 3.3.3.1 (Well Balanced on fixed grid). *Linearity Preserving Eulerian RD schemes preserves exactly the steady solution (19), if the same continuous numerical representation is used for u and g.*

Proof. Applying the scheme (43) to equation (1) for a fixed mesh

$$u_i^{n+1} = u_i^n - \frac{\Delta t}{|S_i|} \sum_{K \in \mathscr{D}_i} \phi_i^K$$

using a conservative linearization and using the same P1 approximation for the solution and the "bathimetry"

$$\begin{aligned} \phi^K &= \int_K (\nabla \cdot \mathbf{f}(u_h^n) + \mathbf{a}(u_h^n) \cdot \nabla g_h)\, d\mathbf{x} = \int_K \mathbf{a}(u_h^n) \cdot \nabla \left(u_h^n + g_h\right) d\mathbf{x} \\ &= \frac{1}{2}\bar{\mathbf{a}} \cdot \sum_{j \in K} \mathbf{n}_j \left(u_j^n + g_j\right) = \frac{1}{2}\bar{\mathbf{a}} \cdot \sum_{j \in K} \mathbf{n}_j \eta_j^n \end{aligned}$$

Hence starting from a lake at rest initial solution the total residual is everywhere null

$$\phi^K = \frac{1}{2}\bar{\mathbf{a}} \cdot \left(\sum_{j \in K} \mathbf{n}_j \right) \eta_0 = 0$$

We get the proof $u_i^{n+1} = u_i^n \quad \forall n = 1, ...M-1$.

3.3.4 Well Balancedness : ALE RD

Let's check the same property on the RD scheme approximating the balance law in ALE form (20). Provided that our method is DGCL (property 3.2.2) and it is also Well Balanced with $\sigma = 0$ (property 3.3.3)

$$J_A \left(\left.\frac{\partial u}{\partial t}\right|_X - \sigma \cdot \nabla u \right) + u \underbrace{\left(\left.\frac{\partial J_A}{\partial t}\right|_X - \nabla \cdot \sigma \right)}_{\text{DGCL}} + J_A \underbrace{(\nabla \cdot \mathbf{f} - \mathbf{a} \cdot \nabla g)}_{\text{Well Balanced}} = 0$$

the above equation has an unsteady lake at rest solution

$$\left.\frac{\partial u}{\partial t}\right|_X - \sigma \cdot \nabla u = 0 \tag{58}$$

with u that evolves in time following the domain arbitrary motion

$$u = \eta_0 - g(\mathbf{x}) \Rightarrow u = u_0(\mathbf{x}) = u_0(A(t, \mathbf{X}))$$

We need the RD algorithm in its full version (43) but, since we are not able of solving (58) exactly, we cannot expect a Well Balanced property. It is immediate to check that the residual is not zero

$$\begin{aligned} \phi^K &= \int_{K^{n+1/2}} (\nabla \cdot \mathbf{f}(u_h^n) - \sigma_h^* \cdot \nabla u_h^n + \mathbf{a}(u_h^n) \cdot \nabla g_h^n)\, d\mathbf{x} \\ &= \int_{K^{n+1/2}} (\mathbf{a}(u_h^n) \cdot \nabla \left(u_h^n + g_h^n\right) - \sigma_h^* \cdot \nabla u_h^n)\, d\mathbf{x} \end{aligned}$$

Two terms are not null: beside the residual part consistent with the problem (58), that allows the correct time evolution of the solution, a perturbation term appears which is proportional to the gradient $\nabla\left(u_h^n+g_h^n\right)$. This term is not zero, infact $u_h^n+g_h^n=\eta_h^n\neq\eta_0$ because u_h^n is an approximation of the exact value $u(\mathbf{x},t^n)$ which would have ensured the above equality. This result is independent from the choice of the time instant at which we evaluate the bathimetry.

If we start from the ALE balance law in η variable, equation (25), we show that one can do much better in the approximation of the lake at rest solution. In this case imposing $\eta=\eta_0$

$$J_A\underbrace{\left(\left.\frac{\partial\eta}{\partial t}\right|_X-\sigma\cdot\nabla\eta\right)}_{0}+\eta\underbrace{\left(\left.\frac{\partial J_A}{\partial t}\right|_X-\nabla\cdot\sigma\right)}_{\text{DGCL}}+J_A\underbrace{(\nabla\cdot\mathbf{f}-\mathbf{a}\cdot\nabla g)}_{\text{Well Balanced}}=0$$

we have an identity, and the Well Balanced property on moving grids can be recovered. The scheme (43) in η variable simply writes

$$\eta_i^{n+1}=\eta_i^n-\frac{\Delta t}{|\tilde{S}_i^{n+1/2}|}\sum_{K\in\mathscr{D}_i}\phi_i^K \tag{59}$$

and the following property holds

Property 3.3.4.2 (Well Balanced ALE). *RD-ALE scheme in the form (59) which verifies the DGCL condition (property 3.2.2), preserves exactly the steady solution (19), provided it preserves the same state on fixed grid (property 3.3.3).*

Proof. For RD in the form (59) the residual is computed as

$$\begin{aligned}\phi^K &= \int_{K^{n+1/2}}(\nabla\cdot\mathbf{f}(u_h^n)-\sigma_h^*\cdot\nabla\eta_h^n+\mathbf{a}(u_h^n)\cdot\nabla g_h^n)\,d\mathbf{x}\\ &= \int_{K^{n+1/2}}(\mathbf{a}(u_h^n)\cdot\nabla\left(u_h^n+g_h^n\right)-\sigma_h^*\cdot\nabla\eta_h^n)\,d\mathbf{x}\end{aligned}$$

Since u_h is computed in a post-precessing step once the unknown η_h has been computed

$$u_h^n=\eta_h^n-g_h^n$$

we can further develop the residual

$$\begin{aligned}\phi^K &= \int_{K^{n+1/2}}(\mathbf{a}(u_h^n)\cdot\nabla\eta_h^n-\sigma_h^*\cdot\nabla\eta_h^n)\,d\mathbf{x}\\ &= =\frac{1}{2}(\bar{\mathbf{a}}-\bar{\sigma})\cdot\sum_{j\in K}\mathbf{n}_j^{n+1/2}\eta_j^n\\ &= \frac{1}{2}(\bar{\mathbf{a}}-\bar{\sigma})\cdot\left(\sum_{j\in K}\mathbf{n}_j^{n+1/2}\right)\eta_0=0\end{aligned}$$

Once again, starting from a lake at rest initial solution, we get the proof.

3.4 Distribution schemes

In this section we provide different definitions for ϕ_i^K, in order to substitute it into the RD scheme (43). We use two very known approximations that can be absorbed or even recasted into RD form: the first order FV with Lax-Friederich fluxes and the second order FE with SUPG stabilization. A method to construct second order and stable approximation is presented to treat nonlinear problems.

3.4.1 Lax Friederich scheme

The Lax-Friederich residual, obtained from an heuristic extension of its FV counterpart, is written as

$$\phi_i^{LxF} = \frac{1}{3}\phi^K + \alpha^K \sum_{j\in K, j\neq i} (u_i - u_j), \qquad \alpha^K \geq \max_{j\in K} |k_j^{ALE}| \tag{60}$$

The scheme is the result of a centered Galerkin scheme plus a diffusion term which introduces some form of stabilization to damp oscillations. A great amount of diffusion is introduced, unfortunately much more the one is effectively needed. This scheme does not give rise to oscillation close to discontinities because it respects property 3.3.1.2 but it is not linearity preserving, thus it is only first order accurate.

3.4.2 SUPG scheme

Starting from the classical SUPG method, under the hypothesis of conservative linearization, it is possible to derive the SUPG residual as

$$\phi_i^{SUPG} = \beta_i^{SUPG}\phi^K \tag{61}$$

with

$$\beta_i^{SUPG} = \frac{1}{3} + k_i^{ALE} T, \quad T = \left(\sum_{j\in K} |k_j^{ALE}|\right)^{-1} \tag{62}$$

It is easy to check that SUPG is linearity preserving but does not respect the bounds stated in property 3.3.1.2.

3.4.3 Limited Lax Friederich scheme

A root to the construction of schemes which are stable in the sense of property 3.3.1.2 and second order accurate consists in limiting the unbounded coefficients of a Lax-Friederich scheme. The Limited Lax-Friederich residual reads

$$\phi_i^{LLxF} = \beta_i^{LLxF}\phi^K \tag{63}$$

with bounded coefficients obtained from the following limiting procedure [3]

$$\beta_i^{LLxF} = \frac{\left(\phi_i^{LxF}\right)^+}{\sum_{j\in K}\left(\phi_j^{LxF}\right)^+} \tag{64}$$

3.4.4 Limited Lax Friederich Stabilized scheme

For limited schemes things, at end, turned out not so simple. Infact looking to the solution obtained with LLxF one can observe the appereance of *wiggles* on the isolines, wiggles that gives very poor results in term of accuracy and destroy the convergence property expected. Without giving the details of possible explanations, a solution to cure the problem is suggesed by [4]. A SUPG term is added, with a limiter to tune the diffusion introduced. The final distribution for such a scheme called LLxF stabilized or briefly LLxFs reads

$$\phi_i^{LLxFs} = \beta_i^{LLxFs}\phi^K \qquad \beta_i^{LLxFs} = (1-\delta(u^h))\beta_i^{LLxF} + \delta(u^h)\beta_i^{SUPG} \tag{65}$$

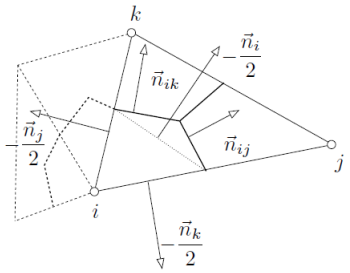


Figure 1: Finite Volume normals $\mathbf{n}_{ij}$ for the interface of the median dual cell area; geometrical relationship with Residual Distribution normals $\mathbf{n}_i$: $\sum_{j\in K, j\neq i}\mathbf{n}_{ij} = -\frac{\mathbf{n}_i}{2}$

For every triangle we can use the following choice

$$l(u^h) = \min\left(1, \frac{1}{\frac{|\phi^K|}{\bar{u}h_K^2} + \varepsilon}\right) \tag{66}$$

with h_K the element reference size, $\bar{u} = \max_{j\in K}|u_j|$ and $\varepsilon = 10^{-10}$. It is easy to show that the definition (66) can detect the discontinuties. Infact $\delta(u^h)$ is of order $\mathscr{O}(1)$ in smooth region where dissipation is needed to damp oscillations and of order $\mathscr{O}(h^{-1})$ across discontinuities where the LLxF scheme behaves nicely computing well-resolved profiles.

4 Finite Volume discretization

It is surprising that many different Finite Volume schemes can be incorporated in a residual-based form, simply computing the total flux across ∂S_i by element instead that by interface

$$|S_i^{n+1}|u_i^{n+1} = |S_i^n|u_i^n - \Delta t \sum_{K\in\mathscr{D}_i}\phi_i^{K,FV} \tag{67}$$

The discretization of the flux and of the source term, even if it is carried out in a similar fashion, here is splitted for clarity

$$\phi_i^{K,FV} = \phi_i^{K,adv} + \phi_i^{K,S}$$

The advective part of the residual is expressed as the sum of the contributions of two fluxes respectively across the interfaces i-j and i-k, both lying into element K

$$\phi_i^{K,adv} = \sum_{j\in K, j\neq i}\left(\frac{\hat{\mathbf{f}}_j + \hat{\mathbf{f}}_i}{2}\cdot\mathbf{n}_{ij} - \frac{\hat{u}_j+\hat{u}_i}{2}\upsilon_{ij} - \frac{|\tilde{\mathbf{a}}\cdot\mathbf{n}_{ij} - \upsilon_{ij}|}{2}(\hat{u}_j - \hat{u}_i)\right) \tag{68}$$

The apex $(\cdot)^r$ stands for some reconstruction/approximation of the variable at the interface while $\tilde{\mathbf{a}} = \frac{\mathbf{f}_j - \mathbf{f}_i}{u_j - u_i}$ is the Roe-averaged advective speed and $\mathbf{n}_{ij}$ the FV normal of the interface i-j belonging to the element K, see Fig. (1). We recognize also the ALE part of the flux which has been discretized defining an *interface velocity*

$$\upsilon_{ij} = \int_{\partial S_{ij}(t)\cup K}\sigma_h\cdot\mathbf{n}_{ij}$$

As seen for RD methods, we face the vagueness in the definition of the time where to evaluate the above integral and, of course we lack a definition for the velocity σ_h. These values are fixed such as to verify the DGCL condition which reads

$$|S_i^{n+1}| - |S_i^n| = \Delta t\sum_{K\in\mathscr{D}_i}\upsilon_{ij}$$

Property 3.2.2 could be used also for schemes in the form (67), thus a closure for DGCL is obtained by the follwoing choice of the interface velocity

$$\upsilon_{ij} = \int_{\partial S_{ij} \cup K} \sigma_h^* \cdot \mathbf{n}_{ij}^{n+1/2}$$

It is interesting to note the full analogy with the interface velocity consistency condition given in [5].

As clearly described in [6], an upwind discretization of the source term ensures the satisfaction of the Well Balanced property which, in turn, prevents from unphysical oscillation in what we called *lake at rest* state

$$\phi_i^{K,S} = \sum_{j \in K, j \neq i} \left(\frac{\hat{g}_j - \hat{g}_i}{2} \mathbf{a} \cdot \mathbf{n}_{ij} - \frac{|\tilde{\mathbf{a}} \cdot \mathbf{n}_{ij} - \upsilon_{ij}|}{2} (\hat{g}_j - \hat{g}_i) \right)$$

Finally isolating in (68) the single contribution relative to the interface i-j, dividing a right state from a left state

$$H(u_R, u_L) = \frac{\mathbf{f}(\hat{u}_R) + \mathbf{f}(\hat{u}_L)}{2} \cdot \mathbf{n}_{ij} - \frac{\hat{u}_R + \hat{u}_L}{2} \upsilon_{ij} - \frac{|\tilde{\mathbf{a}} \cdot \mathbf{n}_{ij} - \upsilon_{ij}|}{2} (\hat{u}_R - u_L) \tag{69}$$

$$Q(g_R, g_L) = \frac{\hat{g}_R - \hat{g}_L}{2} \mathbf{a} \cdot \mathbf{n}_{ij} - \frac{|\tilde{\mathbf{a}} \cdot \mathbf{n}_{ij} - \upsilon_{ij}|}{2} (\hat{g}_R - \hat{g}_L) \tag{70}$$

we can introduce many types of FV schemes equivalent to RD.

4.1 Upwind Finite Volume scheme

The classical first order upwind Finite Volume scheme could be recasted into a RD form if, in (69) and (70), nodal values are used for the left and right state

$$\phi_i^{FV} = \sum_{j \in K, j \neq i} (H(u_j, u_i) + Q(g_j, g_i)) \tag{71}$$

with

$$H(u_j, u_i) = \frac{\mathbf{f}_j + \mathbf{f}_i}{2} \cdot \mathbf{n}_{ij} - \frac{u_j + u_i}{2} \upsilon_{ij} - \frac{|\tilde{\mathbf{a}} \cdot \mathbf{n}_{ij}|}{2} (u_j - u_i)$$

$$Q(g_j, g_i) = \frac{g_j - g_i}{2} \mathbf{a} \cdot \mathbf{n}_{ij} - \frac{|\tilde{\mathbf{a}} \cdot \mathbf{n}_{ij} - \upsilon_{ij}|}{2} (g_j - g_i)$$

With the instruments provided for RD methods one can demonstrate all the well known properties of the upwind FV method including positivity, consistency and conservativeness.

4.2 Finite Volume scheme with centered recostruction

It is possible a better approximation of the left and right state, respect to the choice of (71) where nodal values were used. For example a second order accurate reconstruction of the solution at the interface is obtained with the following piecewice linear approximation over the element

$$u_R = u_i + \frac{1}{2} \mathbf{l}_{ij} \cdot \nabla u_i$$

$$u_L = u_j - \frac{1}{2} \mathbf{l}_{ij} \cdot \nabla u_j \tag{72}$$

where $\mathbf{l}_{ij}$ is the vector connecting nodes i and j while the gradient is recostructed with a centered weighted-area average

$$\nabla u_i = \frac{\sum_{K\in\mathscr{D}_i} (\nabla u)^K |K|}{\sum_{K\in\mathscr{D}_i} |K|}$$

Finally substituting formulas (72) in (69) and in turn in Eq. (71) we obtain a second order linear scheme which is called here Fromm scheme in analogy with the nomenclature of LeVeque in [7]

$$\phi_i^{FROMM} = \sum_{j\in K, j\neq i} (H(u_R,u_L) + Q(g_R,g_L)) \tag{73}$$

As pointed out in [6], even if the reconstruction (72) is used also g, the numerical flux does not balance the source term. A simple correction to cure this problem is provided in the same reference.

4.3 MUSCL scheme

The second order accurate Fromm scheme is much more accurate then the upwind scheme but fails near discontinuties where the same idea of improving accuracy through a piecewise linear reconstruction does not make sense. To avoid unphysical overshoot in the solution, we have to ensure that the reconstruction step is TVD. This is done by limiting the slope with the following procedure

$$\begin{aligned} u_R &= u_i + \frac{1}{2} LIM(a,b) \\ u_L &= u_j - \frac{1}{2} LIM(c,b) \end{aligned} \tag{74}$$

where

$$\begin{aligned} b &= u_j - u_i \\ a &= 2(\nabla u_i)\cdot \mathbf{l}_{ij} - b \\ c &= 2(\nabla u_j)\cdot \mathbf{l}_{ij} - b \end{aligned}$$

For the limiter operator we used the slope limiter proposed by Van Albada and Van Leer

$$LIM(a,b) = \begin{cases} \frac{(a+\varepsilon)b+(b+\varepsilon)a}{a^2+b^2+\varepsilon}, & \text{if } ab > 0 \\ 0, & \text{if } ab \leq 0 \end{cases} \tag{75}$$

5 Explicit Runge Kutta time approximation

Now we search for second order time-accurate discretizations. Balance laws, either in the form (20) or (25), are first discretized in time with Runge Kutta two.

5.1 RD-RK2

The space approximation follows the one done in paragraph (3.2.2) with the only difference that a non conservative form (respectively (21) and (26)) is used in the stabilization bubble. For the details of why this trick is necessary

and why it does not spoil conservativness we refer to [8]. The Galerkin approximation of the time discretized balance law (20) with $\sigma^h = \sigma_h^*$ and midpoint configuartion, writes at $k-th$ time step

$$\frac{\Delta}{\Delta t}\int_{\Omega(t)} \varphi_i u_h^k \, d\mathbf{x} + \int_{\Omega^{n+1/2}} \varphi_i \left(\nabla\cdot\mathbf{f}(u_h) - \sigma_h^*\cdot\nabla u_h + S_h\right)^k d\mathbf{x} - \int_{\Omega^{n+1/2}} \varphi_i \left(u_h \nabla\cdot\sigma_h^*\right)^k d\mathbf{x} = 0 \tag{76}$$

While for the stabilization one, also computed at midpoint configuration, we have

$$\sum_{K\in\mathscr{D}_i}\int_{K^{n+1/2}} \gamma_i \frac{\overline{\Delta u_h^k}}{\Delta t} d\mathbf{x} + \sum_{K\in\mathscr{D}_i}\int_{K^{n+1/2}} \gamma_i \left(\nabla\cdot\mathbf{f}(u_h^n) - \sigma_h^*\cdot\nabla u_h^n + S_h\right)^k d\mathbf{x} = 0 \tag{77}$$

As one can see, the time part in the stabilization undergoes a step shifting in the form $\overline{\Delta u^k} = u^{k-1} - u^n$. Since the term under analysis is now dependent on quantities already computed at the previous step, the shifting operator guarantees to end up with an explicit linear scheme, without deteriorating the overall accuracy. The algebraic details are left aside and we give directly the final formulation that we have implemented in the numerical experiments

$$\begin{cases} |\tilde{S}_i^{n+1/2}| \frac{u_i^1 - u_i^n}{\Delta t} = -\sum_K \Phi_i^{RK(1)} \\ |\tilde{S}_i^{n+1/2}| \frac{u_i^{n+1} - u_i^n}{\Delta t} = -\sum_K \left(\Phi_i^{RK(2)} - \sum_j \tilde{m}_{ij}^{lump} \frac{u_j^1 - u_j^n}{\Delta t}\right) \end{cases} \tag{78}$$

The modified mass-matrix is defined as

$$\tilde{m}_{ij}^{lump} = \left(1 + \frac{\Delta t}{2}\nabla\cdot\sigma_h^*\right)\left(\delta_{ij} + l(\mathsf{u})\left(m_{ij}^{GAL} - \delta_{ij}\right)\right) \tag{79}$$

where the apex $(\cdot)^{lump}$ points out that, depending on the limiter, the Galerkin mass matrix is lumped or not. It is well known that the lumping of the mass matrix corresponds to introduce some numerical diffusion which it is desiderable when computing discontinuities. The total residual computed at the $k-th$ step is

$$\Phi_i^{RK(k)} = \int_{K^{n+1/2}} w_i \left(\frac{\overline{\Delta u_h^k}}{\Delta t} + \left(\nabla\cdot\mathbf{f}(u_h) - \sigma_h^*\cdot\nabla u_h + S_h\right)^k\right) d\mathbf{x} \tag{80}$$

which in a RD form writes

$$\Phi_i^{RK(k)} = \sum_{j\in K} m_{ij}^K \frac{\overline{\Delta u_j^k}}{\Delta t} + \phi_i^{RK(k)} \tag{81}$$

In particular exploiting the two steps

$$\Phi_i^{RK(1)} = \phi_i(u^h) \tag{82}$$

$$\Phi_i^{RK(2)} = \sum_{j\in K} m_{ij}^K \frac{u_j^1 - u_j^n}{\Delta t} + \frac{1}{2}\left(\phi_i(u_h^1) + \phi_i(u_h^n)\right) \tag{83}$$

The steady part of the residual ϕ_i is defined according to the schemes (60), (62),(63),(65). For clarity we explicitly compute the total residual $\Phi_i^{RK(k)}$ for each scheme presented in section (3.4). To avoid further complication in the notation the superscript $RK(k)$ has been changed in $XX(k)$ where XX is the shorthand notation for the scheme. For the LxF scheme the definition of a mass matrix is not so clear because of the unboundness of the distribution

coefficients. Moreover this scheme are only first order accurate, thus one can lump the mass matrix

$$\Phi_i^{LxF(k)} = \frac{|K^{n+1/2}|}{3}\frac{\Delta\overline{u^k}_i}{\Delta t} + \phi_i^{LxF(k)} \tag{84}$$

$$\Phi_i^{SUPG(k)} = \sum_{j\in K} m_{ij}^{SUPG}\frac{\Delta\overline{u^k_j}}{\Delta t} + \phi_i^{SUPG(k)} \tag{85}$$

$$\Phi_i^{LLxFs(k)} = \sum_{j\in K} m_{ij}^{LLxFs}\frac{\Delta\overline{u^k_j}}{\Delta t} + \phi_i^{LLxFs(k)} \tag{86}$$

In order to exploit all the terms of the residual, the last effort consists in the construction of the consistent mass-matrix $m_{ij} = \int_{K^{n+1/2}} \varphi_j w_i \, d\mathbf{x}$.

Two possible choices that satisfy (42) lead to the mass-matrix formulations used in our computations called respectively F1 and F2 are

$$\begin{aligned} w_i^{F1}(\mathbf{x}) &= \beta_i^K \quad \mathbf{x}\in K \\ w_i^{F2}(\mathbf{x}) &= \varphi_i(\mathbf{x}) + \beta_i^K - \frac{1}{3} \quad \mathbf{x}\in K \end{aligned}$$

The mass matrix for both the formulation used in our computation, is then computed

$$\begin{aligned} m_{ij}^{K,F1} &= \int_K \varphi_j w_i^{F1}\, d\mathbf{x} \\ &= \frac{|K|}{3}\beta_i^K = |K|\hat{m}_{ij}^{K,F1} \\ m_{ij}^{K,F2} &= \int_K \varphi_j w_i^{F2}\, d\mathbf{x} \\ &= \frac{|K|}{3}\left(\frac{\delta_{ij}}{4} + \beta_i^K - \frac{1}{12}\right) = |K|\hat{m}_{ij}^{K,F2} \end{aligned}$$

In [9] it shown that the so called F2 mass matrix corresponds to the sum of the the F1 and a dissipation operator

$$m_{ij}^{F2} = m_{ij}^{F1} + \delta m_{ij}, \quad v^T m_{ij} v > 0 \; \forall v \in \mathbb{R}^3$$

Also for this mass matrix an automatic switch from F1 to F2 is performed in correspondence of discontinuities in order to tune the dissipation operator in the numerical scheme

$$m_{ij}^K = m_{ij}^{F1} + (1 - l(\mathsf{u}))\delta m_{ij}$$

Putting $\beta_i^{XX} = \beta_i^{GAL} = 1/3$ the Galerkin mass-matrix is obtained

$$m_{ij}^{GAL} = \frac{|K^{n+1/2}|}{3}\left(\frac{\delta_{ij}}{4} + \frac{1}{4}\right) \tag{87}$$

For the non-linear limited LLxF scheme we use directly the limiting operation (64) with the total residual defined in this contest

$$\beta_i^{LLxF} = \frac{\left(\Phi_i^{LxF(k)}\right)^+}{\sum_{j\in K}\left(\Phi_j^{LxF(k)}\right)^+} \tag{88}$$

Finally the limiter (66) is rewritten according to the modified form of the residual

$$\delta(u^h) = \min\left(1, \frac{1}{\frac{|\Phi^K|}{\Delta t \bar{u} \max\limits_{j\in K} ||\mathbf{a}_j - \sigma_j|| h_K^2} + \varepsilon}\right) \tag{89}$$

The positivity result seen in Sec.(3.3.1) remains valid for the predictor step but must be extended to the corrector step. This does not pose particular problems since the mass matrix defined within the residual, for a positive distribution, is lumped. In the compact prototype scheme (47) we substitute the residual for the second RK step. Here the source term is neglected since its contribution to the stability bounds for the numerical solution remains the same as in the first RK step and can be added by superimposition.

$$u_i^{n+1} = u_i^n - \frac{1}{2}\frac{\Delta t}{|\tilde{S}_i^{n+1/2}|}\sum_{K\in\mathscr{D}_i}\sum_{j\in K, j\neq i} a_{ij}^K\left(u_i^n - u_j^n\right) - \frac{1}{2}\frac{\Delta t}{|\tilde{S}_i^{n+1/2}|}\sum_{K\in\mathscr{D}_i}\sum_{j\in K, j\neq i} b_{ij}^K\left(u_i^1 - u_j^1\right) \tag{90}$$

Proof. We substitute a first order residual such as (85) into (78)

$$\begin{aligned} u_i^{n+1} &= u_i^1 - \frac{\Delta t}{|\tilde{S}_i^{n+1/2}|}\sum_{K\in\mathscr{D}_i}\left(\frac{1}{2}\beta_i^K\sum_{j\in K} k_j^{ALE} u_j^n + \frac{1}{2}\beta_i^K\sum_{j\in K} k_j^{ALE} u_j^1 + \frac{|K|^{n+1/2}}{\Delta t}(u_i^1 - u_i^n)\right) \\ &= u_i^n - \frac{\Delta t}{|\tilde{S}_i^{n+1/2}|}\sum_{K\in\mathscr{D}_i}\frac{1}{2}\left(\sum_{j\in K, j\neq i}\left(\beta_i^K k_j^{ALE}\right)\left(u_i^n - u_j^n\right) + \sum_{j\in K, j\neq i}\left(\beta_i^K k_j^{ALE}\right)\left(u_i^1 - u_j^1\right)\right) \end{aligned}$$

Introducing the coefficient a_{ij}^K and b_{ij}^K we get the proof.

Property 5 (L^∞ bounds). *The numerical solution of a scheme in the form (78) respects the following bounds*

$$\frac{1}{2}\min_{j\in\mathscr{D}_i} u_j^n + \frac{1}{2}\min_{j\in\mathscr{D}_i} u_j^1 \leq u_i^{n+1} \leq \frac{1}{2}\max_{j\in\mathscr{D}_i} u_j^n + \frac{1}{2}\max_{j\in\mathscr{D}_i} u_j^1 \tag{91}$$

if the following conditions are verified:

- *the scheme is LED, hence*

$$\tilde{a}_{ij}^K \geq 0, \quad \tilde{b}_{ij}^K \geq 0, \quad \tilde{c}_{ij}^K \geq 0 \quad \forall j \in \mathscr{D}_i,\ j \neq i \quad \text{and} \quad \forall i \in \mathscr{T}_h \tag{92}$$

  with $\tilde{(\cdot)}_{ij}^K$ defined in (94) and (53)

- *the scheme verifies a CFL-like condition, hence there is a time step restiction*

$$\Delta t \leq \min\left(\frac{|\tilde{S}_i^{n+1/2}|}{\sum_{j\in\mathscr{D}_i, j\neq i}\tilde{a}_{ij}^K}, \frac{|\tilde{S}_i^{n+1/2}|}{\sum_{j\in\mathscr{D}_i, j\neq i}\left(\tilde{b}_{ij}^K + \tilde{c}_{ij}^K\right)}\right) \quad \forall i \in \mathscr{T}_h \tag{93}$$

Proof. Adding and subtracting to (90) the term $\frac{1}{2}u_i^1$ and using the first RK step

$$\begin{aligned} u_i^{n+1} &= \frac{1}{2}\left(u_i^n - \frac{\Delta t}{|\tilde{S}_i^{n+1/2}|}\sum_{K\in\mathscr{D}_i}\sum_{j\in K, j\neq i} a_{ij}^K\left(u_i^n - u_j^n\right)\right) \\ &+ \frac{1}{2}\left(u_i^1 - \frac{\Delta t}{|\tilde{S}_i^{n+1/2}|}\sum_{K\in\mathscr{D}_i}\sum_{j\in K, j\neq i}\left(b_{ij}^K + c_{ij}^K\right)\left(u_i^1 - u_j^1\right)\right) \end{aligned}$$

In order to lighten the notation we define the terms

$$\tilde{a}_{ij}^K = \sum_{K\in\mathscr{D}_i\cap\mathscr{D}_j} a_{ij}^K, \quad \tilde{b}_{ij}^K = \sum_{K\in\mathscr{D}_i\cap\mathscr{D}_j} b_{ij}^K \tag{94}$$

If one switches the order of the summation $\sum_{K\in\mathscr{D}_i}\sum_{j\in K, j\neq i} = \sum_{j\in\mathscr{D}_i, j\neq i}\sum_{K\in\mathscr{D}_i\cap\mathscr{D}_j}$ then can use the definitions (94) and gets

$$\begin{aligned}
u_i^{n+1} &= \frac{1}{2}\left(1-\frac{\Delta t}{|\tilde{S}_i^{n+1/2}|}\sum_{j\in\mathscr{D}_i, j\neq i}\tilde{a}_{ij}^K\right)u_i^n + \frac{1}{2}\frac{\Delta t}{|\tilde{S}_i^{n+1/2}|}\sum_{j\in\mathscr{D}_i, j\neq i}\tilde{a}_{ij}^K u_j^n \\
&+ \frac{1}{2}\left(1-\frac{\Delta t}{|\tilde{S}_i^{n+1/2}|}\sum_{j\in\mathscr{D}_i, j\neq i}\left(\tilde{b}_{ij}^K+\tilde{c}_{ij}^K\right)\right)u_i^1 + \frac{1}{2}\frac{\Delta t}{|\tilde{S}_i^{n+1/2}|}\sum_{j\in\mathscr{D}_i, j\neq i}\left(\tilde{b}_{ij}^K+\tilde{c}_{ij}^K\right)u_i^1 \\
&= C_{ii}^n u_i^n + \sum_{j\in\mathscr{D}_i, j\neq i} C_{ij}^n u_j^n + C_{ii}^1 u_i^1 + \sum_{j\in\mathscr{D}_i, j\neq i} C_{ij}^1 u_j^1
\end{aligned}$$

The LED condition ensures that $C_{ij}^n > 0$ and $C_{ij}^1 > 0$ while CFL-like condition ensure that $C_{ii}^n > 0$ and $C_{ii}^1 > 0$ thus the bounds (91) holds.

5.2 FV-RK2

For the Finite Volume method the RK discretization is more immediate since there is no matrix to be inverted while the second step does not pose any problem for the satisfaction of the DGCL. Thus the method implemented reads as follows

$$\begin{cases} |S_i^{n+1}|u_i^1 = |S_i^n|u_i^n - \Delta t\sum_K \Phi_i^{RK(1)} \\ |S_i^{n+1}|u_i^{n+1} = |S_i^n|u_i^n - \Delta t\sum_K \Phi_i^{RK(2)} \end{cases} \tag{95}$$

exploiting the two steps

$$\Phi_i^{RK(1)} = \phi_i(u^h) \tag{96}$$

$$\Phi_i^{RK(2)} = \frac{1}{2}\left(\phi_i(u_h^1)+\phi_i(u_h^n)\right) \tag{97}$$

The residual can be substituted with the expressions for the different schemes (ϕ_i^{FV}, ϕ_i^{FROMM}, ϕ_i^{MUSCL}) seen in Sec. (4).

6 Preliminary numerical verification

The scalar experiments shown here provide the evidence for the properties seen in Sec.(3.3): stability, accuracy and well balanced.

The scalar balance law in u (16), and in η (25) have been approximated with the RD-RK2 scheme of Sec.(5).

For all the experiments the time step is computed in order to verify the CFL condition

$$\Delta t = \mathrm{CFL}\min_{i\in\mathscr{T}_h}\frac{|\tilde{S}_i^{n+1/2}|}{\sum_{K\in\mathscr{D}_i} 3\alpha^K} \tag{98}$$

where $\mathrm{CFL} = 0.8$.

6.1 Lake at rest

To test the Well Balanced we use a linear problem

$$\begin{cases} \frac{\partial u}{\partial t} + \mathbf{a}\cdot\nabla u + \mathbf{a}\cdot\nabla g = 0, & \mathbf{a} = [0,1]\,, \mathbf{x} \in [0,1]\times[0,2], t \in [0,1] \\ u_0(\mathbf{x}) = 1 - g(\mathbf{x}) \end{cases}$$

The bathimetry is defined by the following continuous function

$$g(\mathbf{x}) = 0.8e^{\psi(x,y)}$$

with

$$\psi = -5\,(y-0.9)^2 - 50\,(x-0.5)^2$$

We choose 4 unstructured grid with characteristic lenghts respectively $h_K = 1/30, 1/50, 1/80, 1/160$. Given a reference domain (X,Y), it is mapped according to the following law

$$\begin{cases} x(t) &= X + 0.1\sin(2\pi X)\sin(\pi Y)\sin(2\pi t) \\ y(t) &= Y + 0.2\sin(2\pi X)\sin(\pi Y)\sin(4\pi t) \end{cases} \tag{99}$$

At $t = 1$, the mapping is the identity $\mathbf{x} = \mathbf{X}$, so we can compare the ALE solution with the Eulerian one on the same grid.

Numerical evidence confirmed the results seen in Sec. (3.3.3) and (3.3.4) : the Well Balanced ALE formulation preserves lake at rest solution while balance law in written in conservative variables not. As seen in figure (2) spurious oscillations appears but the convergence rates for the L^2-norm of the error, seems to shows that this oscillation goes to zero with second order of accuracy. The perturbation term does not spoil the global order of accuracy of the scheme for such a problem.

6.2 Linear advection

To test the accuracy of the method we use the simple case of linear advection of a smooth sinusoidal hill

$$\begin{cases} \frac{\partial u}{\partial t} + \mathbf{a}\cdot\nabla u + \mathbf{a}\cdot\nabla g = 0, & \mathbf{a} = [0,1]\,, \mathbf{x} \in [0,1]\times[0,2], t \in [0,1] \\ u_0(\mathbf{x}) = 1 - g(\mathbf{x}) + \cos^2(2\pi r) & \text{if } r \le 0.25,\ r = \sqrt{(x-0.5)^2 + (y-0.5)^2} \\ u_0(\mathbf{x}) = 1 - g(\mathbf{x}) & \text{otherwise} \end{cases}$$

We use again the RK2-SL-SUPG method, thus expecting second order of accuracy for both the formulation in u and η. This can be seen clearly from figure (3). However the precence of spurious oscillations in the flat region make the scheme in water depth much more inaccurate compared to the total height scheme. This is more clear from figure (4). This example shows why the C-property, that seem meaningless when more complicated problems are approached, is instead fundamental. Even in such a case the inability of the scheme to compute correctly the flat region is translated into poor accuracy, even if the order of accuracy remain the one expected.

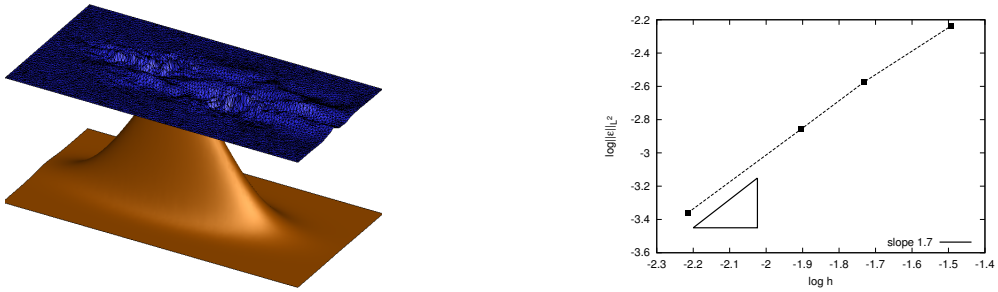


Figure 2: Lake at rest for the ALE formulation in conservative variable u. Failing in verifiyng Well Balanced and convergence

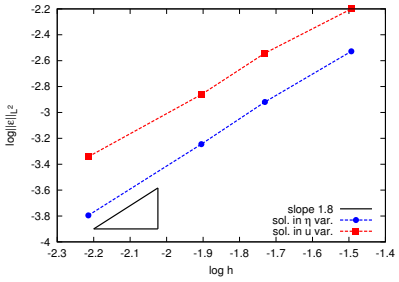


Figure 3: Convergence order for the L^2-norm of the error

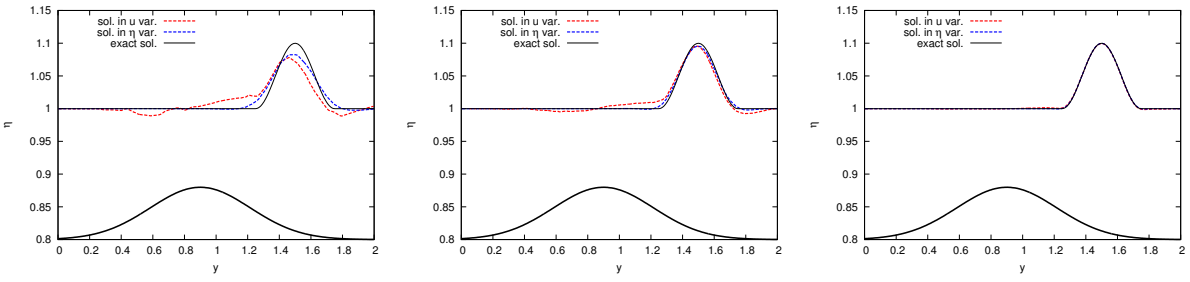


Figure 4: Linear advection with RK2-SL and SUPG scheme. Comparison beteween the numerical solution and exact one on the simmetry line $x = 0.5$. From the top: grid 2,3,5. The bathimetry is out of scale and it is also translated along the y axis.

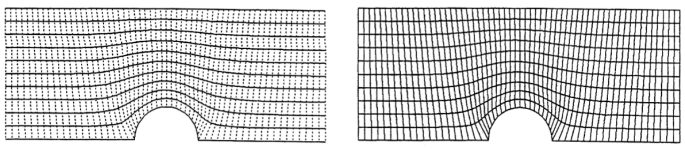


Figure 5: Analogy between flow potential solution and grid points distribution. Left, potential solution with lines at constant ϕ (equipotential lines) and constant ψ (streamlines). Right, actual grid with lines at constant X and constant Y

7 Elliptic equations for grid movement

In this second chapter we exploit the ALE formulation of conservation law proposed in the previous chapter. The aim is to move the nodes refining the grid size in correspondence of the gradients of the solution, obtaining better accuracy respect to a fixed grid computation. For hyperbolic problems, in principle, numerical dispersion or diffusion associated with inadequate resolution of large gradients shoud be reduced allowing sharper discontinuities. A dynamic adaptation is performed, that is the grid continuously adapts to follow developing gradients of unsteady solutions. During the last decades many strategies have been proposed, among them two classical approaches for mesh refinement are, following the nomenclature of [10], nodes redistribution and node insertion/deletion. In this report the first approach is implemented, nodes are gathered into region where the error is larger. If nodes are taken from region with small error/gradient and not from region where other possible gradients occur, the nodes redistribution approach is effective, in particular due to its semplicity. Infact the data structure does not change, resulting in a straigthforward implementation. However it is not trivial to avoid nodes deplation and at the same time obtain everywhere a good element's quality, specially in regions where multiple fronts interact in complex structures. In these cases the method shows a certain lack of robustness which is overcame by the introduction of weights that should enhance smoothness, nodes'concentration and quality.

In this section, following Thompson [10], we fix the equations for adaptive mesh movement. They are developed using the visual analogy between potential solutions and grid points distributions. For example if a flow is described by the velocity potential ϕ and the stream function ψ both of them satisfying a Laplace equation, $\nabla^2\phi = 0$ and $\nabla^2\psi = 0$, we can search for some reference coordinates X and Y which are also solution of a Laplace equation

$$\nabla^2 X = 0, \quad \nabla^2 Y = 0 \tag{100}$$

It is evident the analogy between the solution of the two problems: in particular the overlapping between equipotential lines and lines at constant X, streamlines and lines at constant Y, as seen in figure (5). The solution of equation (100) determines the transformation from the physical or actual domain defined by the cartesian coordinates $\mathbf{x} = (x, y)$ to the reference or computational domain defined by the coordinates $\mathbf{X} = (X, Y)$

$$\mathbf{X} = \mathbf{X}(\mathbf{x}) \tag{101}$$

In order to recover the actual domain from the reference one, the transformation should be invertible

$$\mathbf{x} = \mathbf{x}(\mathbf{X}) \tag{102}$$

We are assuming that the overlapping of the points in the domain is impossible. The last inversion, gives us the actual position of grid points on the reference domain. As a results of the monotone character of X and Y, the reference domain could be mapped as in figure (6). Here we fix discrete values of the coordiantes $\mathbf{X}_k$ and from (102) we get the actual node position $\mathbf{x}_k$, namely the actual grid.

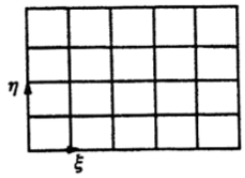


Figure 6: Reference domain

Another way of obtaining the actual grid consists in solving the trasformation of (100). Equations are a little more complicated but we have the advantage of solving them on the reference domain, which allows to get directly the actual grid.

$$\left[\left(\frac{\partial X}{\partial x}\right)^2+\left(\frac{\partial X}{\partial y}\right)^2\right]\frac{\partial^2 x}{\partial X^2}+2\left[\frac{\partial X}{\partial x}\frac{\partial Y}{\partial x}+\frac{\partial Y}{\partial y}\frac{\partial X}{\partial y}\right]\frac{\partial^2 x}{\partial X\partial Y}+\left[\left(\frac{\partial Y}{\partial x}\right)^2+\left(\frac{\partial Y}{\partial y}\right)^2\right]\frac{\partial^2 x}{\partial Y^2}=0 \tag{103}$$

$$\left[\left(\frac{\partial X}{\partial x}\right)^2+\left(\frac{\partial X}{\partial y}\right)^2\right]\frac{\partial^2 x}{\partial X^2}+2\left[\frac{\partial X}{\partial x}\frac{\partial Y}{\partial x}+\frac{\partial Y}{\partial y}\frac{\partial X}{\partial y}\right]\frac{\partial^2 y}{\partial X\partial Y}+\left[\left(\frac{\partial Y}{\partial x}\right)^2+\left(\frac{\partial Y}{\partial y}\right)^2\right]\frac{\partial^2 x}{\partial Y^2}=0 \tag{104}$$

The demonstration of (103) and (104) could be obtained as a special case of the general equation for 2D grid movement given is Sec.(7.2).

Grids obtained by solving elliptic equations have the advantage of being very smooth. For example, imposing correctly Dirichlet conditions, one could get an excellent geometry-driven adaptation of the mesh to moving boundaries. However, sometimes there is the need to concentrate grid points, hence constant$-Y/X$ lines, in region where strong gradients of the solution appears. With Laplace equation one has no control on grid movement making impossible to perform such a solution-driven adaptation. Returning to equation (100), a source term is introduced

$$\nabla^2\mathbf{X}=\mathbf{P} \tag{105}$$

With the proper choice of the monitor function, $\mathbf{P}=(P,Q)$, the above equation prescribes the grid points to move in some specified region. Now all the problems have been transferred to the specification of the monitor functions. As pointed out by Thompson always in [10], this passage implies the fact that one is not dealing anymore with equations derived from physical conservation laws but with *ad hoc* designed equation. In this sense the drawback of a lack of robustness lies within the suggested approach. However during the last decades intuition and experience lead to the design of quite general monitor functions which can ensure a certain grid quality, a certain degree of smoothing and, more important, the adaptation to some particular feature of the solution.

The inversion of (105) leads to a set of equations that can be solved in the actual grid coordinates. The rest of the paragraph is addressed to this not trivial effort.

7.1 1D case

We start with the one dimensional case where vector calculus is avoided, and still the main concept can be highlighted. Starting from the obvious fact that

$$\frac{\partial^2}{\partial x^2}x=0$$

we use the chain rule to transform the differential operator from actual coordinates to reference ones

$$\frac{\partial X}{\partial x}\frac{\partial}{\partial X}\left(\frac{\partial X}{\partial x}\frac{\partial}{\partial X}x\right)=0 \tag{106}$$

Making the derivative of the product in brackets one ends up with the following elliptic differential equation that governs the motion of the grid

$$\left(\frac{\partial X}{\partial x}\right)^2\frac{\partial^2 x}{\partial X^2}+\frac{\partial^2 X}{\partial x^2}\frac{\partial x}{\partial X}=0 \tag{107}$$

Following the approach outlined in the previous paragraph, we let the laplacian of X satisfying a Poisson equation. In particular if the Jacobian is set equal to a certain function $\frac{\partial X}{\partial x}=\omega(x)$, a Poisson equation naturally arise

$$\frac{\partial^2 X}{\partial x^2}=\frac{\partial \omega}{\partial x} \tag{108}$$

From hereinafter we refer to ω as the monitor function, instead of the derivative itself as it emerged in the previous paragraph.

Using the monitor function inside equation (106), the final formulation for the elliptic differential equation reads compactly

$$\frac{\partial}{\partial X}\left(\omega\frac{\partial x}{\partial X}\right)=0 \tag{109}$$

Even if the Poisson equation designed in the reference domain, equation (108), gives us the intuition that grid points are, somehow, moving, it is not that clear where they like to concentrate. From (109) we get the following equidstribution principle

$$\omega\frac{\partial x}{\partial X}=const \tag{110}$$

For a unit increment of X

$$\omega\Delta x=const \tag{111}$$

hence where ω is large the grid points gather, on the opposite where ω is small they stretch. Designing properly the monitor function means we can cluster points in some predefinied region.

Another analogy, different from potential flow theory, emerges from equation (109). Interpreting x as the position of each section in a solid bar of lenght L and ω as the stiffness E of the same bar, one can write the following functional corresponding to the elastic energy

$$F(x)=\frac{1}{2}\int_L \frac{\partial x}{\partial X}\omega\frac{\partial x}{\partial X}\,dX$$

Finding the bar configuration which minimize the above functional corresponds to solve the Euler-Lagrange equation (109). Hence, through an elastic analogy, we have a variational method to set the equation for grid motion. Renaming the displacement respect to the reference configuration $\delta=x-X$ and the axial force $F=-\frac{\partial\omega}{\partial x}$ we exploit the full analogy with the equilibrium equation of the elastic bar

$$\frac{\partial\sigma}{\partial X}=F$$

with the axial stress given by the following linear isotropic law $\sigma=E\frac{\partial\delta}{\partial X}$. Once more the role of ω emerges, controlling both the force and the stiffness of the system.

7.2 2D case

For the two dimensional case we have to make a brief review on how the main differential operators change with the coordinate systems transformation (102). The jacobian of the mapping is

$$\mathscr{J} = \frac{\partial \mathbf{x}}{\partial \mathbf{X}}$$

which is assumed to admit the inverse $\mathscr{J}^{-1}$. Let ϕ be a scalar function of the position. Its gradient could be expressed in both the actual coordinate systems and the reference one. We refer to them respectively with the notation $\nabla_{\mathbf{x}}(\cdot)$ and $\nabla_{\mathbf{X}}(\cdot)$. The following rotation holds

$$\nabla_{\mathbf{x}}\phi = \mathscr{J}^{-1}\nabla_{\mathbf{X}}\phi \tag{112}$$

Let $\mathbf{u} \in \mathbb{R}^2$ be a generic vector of the position. Each component rotates with equation (112), hence in tensor form

$$\nabla_{\mathbf{x}}\mathbf{u} = \mathscr{J}^{-1}\nabla_{\mathbf{X}}\mathbf{u} \tag{113}$$

The divergence of $\mathbf{u}$ is transformed as follows

$$\nabla_{\mathbf{x}} \cdot \mathbf{u} = \frac{\partial \mathbf{u}}{\partial X} \cdot \mathbf{J}_X^{-1} + \frac{\partial \mathbf{u}}{\partial Y} \cdot \mathbf{J}_Y^{-1}$$

where we have defined $\mathbf{J}_X^{-1}$,$\mathbf{J}_Y^{-1}$ as the row vectors of the Jacobian.

$$\mathbf{J}_X^{-1} = \nabla_{\mathbf{x}} X, \quad \mathbf{J}_Y^{-1} = \nabla_{\mathbf{x}} Y \tag{114}$$

Finally the divergence of a tensor $\mathbf{T} \in \mathbb{R}^{2\times 2}$ in actual coordinates reads

$$\nabla_{\mathbf{x}} \cdot \mathbf{T} = \frac{\partial \mathbf{T}}{\partial X}\mathbf{J}_X^{-1} + \frac{\partial \mathbf{T}}{\partial Y}\mathbf{J}_Y^{-1}$$

Once we have established the above trasformations, we use them to derive the set of elliptic equations that governs the motion of the grid points, in actual coordinate. We state the obvious fact that

$$\nabla_{\mathbf{x}}^2 \mathbf{x} = 0$$

Using equation (113) it is possible to transform the inner gradient operator

$$\nabla_{\mathbf{x}} \cdot \left(\mathscr{J}^{-1}\nabla_{\mathbf{X}}\mathbf{x}\right) = 0$$

Equation (7.2) is used to transform the divergence operator

$$\nabla_{\mathbf{x}} \cdot \left(\mathscr{J}^{-1}\nabla_{\mathbf{X}}\mathbf{x}\right) = \frac{\partial\left(\mathscr{J}^{-1}\nabla_{\mathbf{X}}\mathbf{x}\right)}{\partial X}\mathbf{J}_X^{-1} + \frac{\partial\left(\mathscr{J}^{-1}\nabla_{\mathbf{X}}\mathbf{x}\right)}{\partial Y}\mathbf{J}_Y^{-1}$$

After some algebra, we end up with the following

$$\begin{aligned}\left[\left(\frac{\partial X}{\partial x}\right)^2 + \left(\frac{\partial X}{\partial y}\right)^2\right]\frac{\partial^2 x}{\partial X^2} \quad + \quad 2\left[\frac{\partial X}{\partial x}\frac{\partial Y}{\partial x} + \frac{\partial Y}{\partial y}\frac{\partial X}{\partial y}\right]\frac{\partial^2}{\partial X \partial Y} + \left[\left(\frac{\partial Y}{\partial x}\right)^2 + \left(\frac{\partial Y}{\partial y}\right)^2\right]\frac{\partial^2 x}{\partial Y^2} \\ + \left(\nabla_{\mathbf{x}}^2 X\right)\frac{\partial x}{\partial X} + \left(\nabla_{\mathbf{x}}^2 Y\right)\frac{\partial x}{\partial Y} = 0\end{aligned} \tag{115}$$

$$\left[\left(\frac{\partial X}{\partial x}\right)^2+\left(\frac{\partial X}{\partial y}\right)^2\right]\frac{\partial^2 x}{\partial X^2} \quad + \quad 2\left[\frac{\partial X}{\partial x}\frac{\partial Y}{\partial x}+\frac{\partial Y}{\partial y}\frac{\partial X}{\partial y}\right]\frac{\partial^2}{\partial X\partial Y}+\left[\left(\frac{\partial Y}{\partial x}\right)^2+\left(\frac{\partial Y}{\partial y}\right)^2\right]\frac{\partial^2 x}{\partial Y^2}$$
$$+\left(\nabla^2_{\mathbf{x}}X\right)\frac{\partial x}{\partial X}+\left(\nabla^2_{\mathbf{x}}Y\right)\frac{\partial x}{\partial Y}=0 \tag{116}$$

As done in one dimension, a Poisson equation is designed in the reference space setting the following Jacobian

$$\mathscr{J}^{-1}=\begin{matrix}\omega & 0\\ 0 & \omega\end{matrix} \tag{117}$$

As a consequance we have

$$\nabla^2_{\mathbf{x}}\mathbf{X}=\nabla\omega \tag{118}$$

These passages permit to simplify the system which now could be written in a compact form. From (7.2) we get

$$\nabla_{\mathbf{X}}\cdot(\omega\nabla_{\mathbf{X}}\mathbf{x})=0 \tag{119}$$

which rapresents a set of elliptic quasi-linear equations where the unknown is the position in actual coordinates; the adapted grid is directly obtained if one is able of solving (119). A well posed problem is obtained imposing Dirichlet and free-slip boundary conditions. However the treatment of boundary conditions will be addressed in the next paragraph. The monitor function has to be fixed in order that the grid adapts to some particular features. In many unsteady fluid-dynamics problems one could assists to the development of special flow features where large gradients are involved, such as shock waves and boundary layers. A possible root to better resolve numerically these regions consists in a local refinament of the mesh whereas the gradients become important. Winslow monitor function suggested in [11], thus couples the mesh motion with ∇u where u is the solution of the underlying PDE

$$\omega=\sqrt{1+\alpha||\nabla u||^2}$$

α is a non-negative scalar parameter which could be set *ad hoc* and allows the user to have a better control of the mesh distortion. If either α or ∇u are null the mesh PDEs collapse again into Laplace equations for both the actual and the reference domain. For Shallow Water flows the accurate computation of wave crests and troughs could be as important as the sharp computation of bores; to not loose the peaks an hessian is added in the following fashion

$$\omega=\sqrt{1+\alpha\left(\max\left(||\nabla u||,||\nabla^2 u||\right)\right)^2} \tag{120}$$

A last remark concerns again the existence of a variational formulation for equation (119). Given the following

$$F(\mathbf{x})=\frac{1}{2}\int_{\Omega_X}\left(\nabla_{\mathbf{X}}x^T\,\omega\,\nabla_{\mathbf{X}}x+\nabla_{\mathbf{X}}y^T\,\omega\,\nabla_{\mathbf{X}}y\right)d\mathbf{X} \tag{121}$$

(119) can be interpreted as the Euler-Lagrange equation of the above functional.

8 Finite Element approximation with Lagrangian multipliers

In this section we search a numerical solution for equation (119), for which many different methods are available, depending on convenience. Spekreijse [12], once the control function has been specified, uses a Finite Difference approximation for the derivatives in square brackets in equation (115) and (116); then a successive finite difference approximation for the derivatives of x leads to a linear system solved by a fixed-point iteration. Chen [13] prefers to use a Finite Volume approximation since he use the same method for the flow solver. Baines and Hubbard [14],

within a Residual Distribution framework, are interested that each point of the actual grid is moved according to a convex combination of the adiacent points.

Since in this report hyperbolic equations are solved with a Residual Distribution method, a natural choice for approximating equation (119) is to use a standard Finite Element method: the analogy between the two approaches permits to use, in the approximation of both the hyperbolic and the elliptic equation, the same geometrical quantities for the elements (normals, node coordinates, areas).

At this point we fix the rational assumption that the boundary stay fixed in space such as it happens for the case of Shallow Waters equations. A free-slip condition for the displacement of each boundary point completes the differential formulation for (119): boundary points are allowed to move following the inner points but also according to gradients of the solution that develops in correspondence of the same boundary. The imposition of the above condition in the weak formulation for equation (119) is not trivial. Infact [15],[16] bypass the problem redistributing boundary grid points according to the trace of (119), then the new boundary values are used as Dirichlet condition for the inner points. Apart from the fact that the computation of boundary and inner points is carried out separately, the main drawback of this approach consists in the fact that boundary movement is decoupled from the rest of the domain, which could lead to a loss in smoothness.

In this section we consider a coupled formulation of inner and boundary points by means of lagrangian multipliers. We assume that the boundary can be approximated by a closed piecewise linear curve formed by the segments $\Gamma_k, k = 1, N_{bseg}$. In such a special case the imposition of free-slip boundary condition is stated $\mathbf{x}\cdot\mathbf{n}_{\mathbf{X}}|_{\Gamma_k} = c_k$ where $c_k = \mathbf{X}\cdot\mathbf{n}_{\mathbf{X}}|_{\Gamma_k}$. We recall that the solution of equation (119) minimizes the functional (121), thus we can write the following constraint optimization problem

$$\begin{aligned} &\min_{\mathbf{x}} F(\mathbf{x}) \quad \text{with}\, \mathbf{x}\in\Omega_X \\ &\text{subject to} \quad c(\mathbf{x}) = 0 \end{aligned}$$

where the constraint equation consists in the free-slip boundary condition

$$c(\mathbf{x}) = \mathbf{x}\cdot\mathbf{n}_{\mathbf{X}} - c_k \quad \text{in}\,\Gamma_k \subset \partial\Omega_X, \quad \forall k$$

By introducing a lagrangian multiplier λ, the unconstrained optimization problem reads

$$\min_{\mathbf{x},\lambda} \left(F(\mathbf{x}) + \lambda c(\mathbf{x})\right) \quad \text{with}\, \mathbf{x}\in\Omega_X$$

Even if [17], starting from a general functional $F(\mathbf{x},\lambda)$, shows how to develop a FE method stable and optimally convergent, in this report we go through the details of the more practical approach contained in [18]. A standard Galerkin approximation of equation (119) is carried out followed by the application of the constraints on the discretized weak form.

To deal with a well posed problem the boundary is divided into a part where Dirichlet condition is imposed and a part where free-slip is imposed

$$\begin{cases} \nabla_{\mathbf{X}}\cdot(\omega\nabla_{\mathbf{X}}\mathbf{x}) = 0 & \text{in}\,\Omega_X \\ \mathbf{x} = \mathbf{g} & \text{in}\,\partial\Omega_X^{dir} \\ \mathbf{x}\cdot\mathbf{n}_{\mathbf{X}} = c & \text{in}\,\partial\Omega_X^{slip} \end{cases} \tag{122}$$

with $\mathbf{g} = \mathbf{X}$. Searching the solution $\mathbf{x}\in H_g^1(\Omega_X)$, taking a test function as $v\in H_0^1(\Omega_X)$, the weak formulation reads

$$\int_{\Omega_X} v\nabla_{\mathbf{X}}\cdot(\omega\nabla_{\mathbf{X}}\mathbf{x})\,d\mathbf{X} = 0$$

This last expression can be rewritten

$$\int_{\Omega_X} \nabla_{\mathbf{X}}\cdot(v\omega\nabla_{\mathbf{X}}\mathbf{x})\,d\mathbf{X} + \int_{\Omega_X} \omega\nabla_{\mathbf{X}}v\nabla_{\mathbf{X}}\mathbf{x}\,d\mathbf{X} = 0$$

Applying the divergence theorem to the first term we get

$$\int_{\partial\Omega_X} v\omega\nabla_{\mathbf{X}}\mathbf{x}\cdot\mathbf{n}_{\mathbf{X}}\,ds+\int_{\Omega_X}\omega\nabla_{\mathbf{X}}v\nabla_{\mathbf{X}}\mathbf{x}\,d\mathbf{X}=0$$

The points where Dirichlet condition is imposed are the edge points of the borders, which means that the points of a certain segment Γ_k can move along the same segment but can't cross the edges. As a consequences the first term cancels out. Choosing the approximate solution and the test function within the piecewise linear finite element space (P1), $\mathbf{x}_h\in V_g^h=\{\mathbf{x}_h\in X_h^{P1},\mathbf{x}_h|_{\partial\Omega^{dir}}=\mathbf{g}\}$, $v_h\in V_0^h=\{v_h\in X_h^{P1},v_h|_{\partial\Omega^{dir}}=0\}$, we end up with Galerkin method

$$\int_{\Omega_X^h}\omega\nabla_{\mathbf{X}}v_h\nabla_{\mathbf{X}}\mathbf{x}_h\,d\mathbf{X}=0 \tag{123}$$

With a lagrangian basis φ_i, $i\in\mathscr{T}_h$, we can express

$$\mathbf{x}_h=\mathbf{x}_h+\mathbf{x}_h^{dir}=\sum_{j\in\mathscr{T}_h|\mathscr{T}_h^{dir}}\varphi_j\mathbf{x}_j+\sum_{j\in\mathscr{T}_h^{dir}}\varphi_j\mathbf{g}_j$$
$$v_h=\sum_{j\in\mathscr{T}_h}\varphi_j v_j$$

We substitute the last expansions in the discretized weak form (123),

$$\sum_{j\in\mathscr{T}_h|\mathscr{T}_h^{dir}}\underbrace{\int_{\Omega_h}\omega\nabla_{\mathbf{X}}\varphi_i\nabla_{\mathbf{X}}\varphi_j\,d\mathbf{X}}_{k_{ij}}\mathbf{x}_j=\underbrace{\sum_{j\in\mathscr{T}_h^{dir}}\int_{\Omega_h}\omega\nabla_{\mathbf{X}}\varphi_i\nabla_{\mathbf{X}}\varphi_j\,d\mathbf{X}\mathbf{g}_j}_{f_i}\quad\forall i\in\mathscr{T}_h^{in}$$

We partition the nodes $\mathbf{x}_j$ with $j\in\mathscr{T}_h|\mathscr{T}_h^{dir}$ into inner nodes ($\mathscr{T}_h^{in}$) and boundary nodes ($\mathscr{T}_h^{slip}$): the inner nodes coordiantes are collected into the vectors x^{in}, y^{in} while the boundary nodes for which $\mathbf{x}_j\in\partial\Omega_X^{slip}$ are collected into the vectors x^{∂}, y^{∂}. Depending on the fact that $i,j\in\mathscr{T}_h^{in}$ or $i,j\in\mathscr{T}_h^{slip}$ we can also partition the matrix K into submatrices. The same applies for the vector f leading to the following system of equations

$$\begin{pmatrix}\mathsf{K}^{\partial,\partial}&\mathsf{K}^{\partial,in}&0&0\\ \mathsf{K}^{in,\partial}&\mathsf{K}^{in.in}&0&0\\ 0&0&\mathsf{K}^{\partial,\partial}&\mathsf{K}^{\partial,in}\\ 0&0&\mathsf{K}^{in,\partial}&\mathsf{K}^{in,in}\end{pmatrix}\begin{pmatrix}\mathsf{x}^{\partial}\\ \mathsf{x}^{in}\\ \mathsf{y}^{\partial}\\ \mathsf{y}^{in}\end{pmatrix}=\begin{pmatrix}\mathsf{f}_x^{\partial}\\ \mathsf{f}_x^{in}\\ \mathsf{f}_y^{\partial}\\ \mathsf{f}_y^{in}\end{pmatrix} \tag{124}$$

The compact writing $\mathsf{Kx}=\mathsf{f}$, suggests us the analogy with an elastic system with x the vector containing the degrees of freedom and K the stiffness matrix of the system. The system admits a functional representing the deformation energy

$$V=\mathsf{x}^T\mathsf{Kx}$$

Free-slip condition on the boundary nodes is imposed fixing the proper constraints for the system. We search the solution that minimize the total energy $V-\mathsf{x}^T\mathsf{f}$ subjected to the constraint

$$\mathbf{x}_j\cdot\mathbf{n}_{\mathbf{X}_j}-c_j=0\quad\forall j\in\mathscr{T}_h^{slip}$$

In vector notation one can write

$$\begin{pmatrix}\mathsf{B}_x&0&\mathsf{B}_y&0\end{pmatrix}\begin{pmatrix}\mathsf{x}^{\partial}\\ \mathsf{x}^{in}\\ \mathsf{y}^{\partial}\\ \mathsf{y}^{in}\end{pmatrix}=\begin{pmatrix}\dots\\ c_j\\ \dots\\ \dots\end{pmatrix}$$

Using again compact notation we can write $\mathsf{Bx} = \mathsf{b}$ with $\mathsf{B}_x = \mathsf{Diag}(n_{x_j})$ and $\mathsf{B}_y = \mathsf{Diag}(n_{y_j})$. At the end the continuous optimization problem (8), involving differential operators in the definition of the functional, has been discretized in the following discrete optimization problem with an algebraic functional

$$\begin{aligned} &\min_{\mathbf{x}} q(\mathsf{x}) = \mathsf{x}^T\mathsf{K}\mathsf{x} - \mathsf{x}^T\mathsf{f} \\ &\text{subject to} \quad \mathsf{B}\mathsf{x} - \mathsf{c} = 0 \end{aligned} \tag{125}$$

Optimization of quadratic objective function with symmetric and positive definite stiffness matrix $\mathsf{K} \in \mathbb{R}^n \times \mathbb{R}^n$, subjected to m linear constraints, thus $\mathsf{B} \in \mathbb{R}^m \times \mathbb{R}^n$, takes the name of *Convex Quadratic Programming*.

8.1 Optimality conditions

Optimality conditions are well established for such optimization. In particular finding a vector λ which is solution of the following unconstrained optimization is a necessary condition for x to be the solution of (125)

$$\min_{\mathsf{x},\lambda}[V(\mathsf{x}) - \mathsf{x}^T\mathsf{f} + \lambda^T(\mathsf{B}\mathsf{x} - \mathsf{c})]$$

Deriving respect to the unknowns

$$\begin{aligned} \frac{\partial F}{\partial \mathsf{x}} &= \mathsf{K}\mathsf{x} - \mathsf{f} + \mathsf{B}^T\lambda = 0 \\ \frac{\partial F}{\partial \lambda} &= \mathsf{B}\mathsf{x} - \mathsf{c} = 0 \end{aligned}$$

Thus we traced to the resolution of a linear system

$$\begin{pmatrix} \mathsf{K} & \mathsf{B}^T \\ \mathsf{B} & 0 \end{pmatrix} \begin{pmatrix} \mathsf{x} \\ \lambda \end{pmatrix} = \begin{pmatrix} \mathsf{f} \\ \mathsf{c} \end{pmatrix} \tag{126}$$

in the form $\mathsf{A}\mathsf{x} = \mathsf{b}$.

The following lemma points out the conditions under which (126) admits a unique solution which, for what we mention before, represents necessary condition for the original optimization (125). We use $\mathsf{Z} \in \mathbb{R}^n \times \mathbb{R}^{n-m}$ to denote the matrix whose columns are a basis for the null space of B. That is, Z has full rank and $\mathsf{BZ} = 0$.

Lemma *Let* B *have full row rank, and assume that* $\mathsf{Z}^T\mathsf{K}\mathsf{Z}$ *is positive definite, then* A *is nonsingular, and there is a unique vector pair* (x, λ) *satisfying (126).*

It is possible to proof that verifying (126) is also a sufficent condition for x to be solution of (125). This is stated by the following theorem

Theorem. *Suppose that the conditions of the above lemma are verified, if* x *is the solution of (126), it is also the global solution of optimization problem (125).*

Finally we need the following results to show that A is indefinite.

Lemma. *Suppose that* B *has full row rank and that* $\mathsf{Z}^T\mathsf{K}\mathsf{Z}$ *is positive definite. Then* A *has n positive eigenvalues, m negative eigenvalues, and no zero eigenvalues.*

All the proofs are contained in the book of Nocedal [19].

8.2 Details

An explicit expression for the elements of the matrix K is now given. We realize that the integral is not null only on the elements that share node i. We can limit the domain on which we integrate on $\mathscr{D}_i = \cup K, K | \, i \in K$

$$\begin{aligned} k_{ij} &= \int_{\Omega_h} \omega \nabla_{\mathbf{X}} \varphi_i \nabla_{\mathbf{X}} \varphi_j \, d\mathbf{X} = \int_{\mathscr{D}_i} \omega \nabla_{\mathbf{X}} \varphi_i \nabla_{\mathbf{X}} \varphi_j \, d\mathbf{X} \\ &= \sum_{K \in \mathscr{D}_i} \int_K \omega \nabla_{\mathbf{X}} \varphi_i \nabla_{\mathbf{X}} \varphi_j \, d\mathbf{X} = \sum_{K \in \mathscr{D}_i} \int_K \omega \, d\mathbf{X} \, (\nabla_{\mathbf{X}} \varphi_i \nabla_{\mathbf{X}} \varphi_j)^K \quad \forall i \in \mathscr{T}_h^{in} \end{aligned}$$

For the monitor function (120) we use a P1 interpolation of nodal values

$$\omega = \omega_h = \sum_{j \in \mathscr{T}_h} \varphi_j \omega_j, \quad \omega_j = \omega(u_h(\mathbf{x}_j))$$

so that the calculation of the elements is simplified

$$\begin{aligned} k_{ij} &= \sum_{K \in \mathscr{D}_i} \sum_{j \in K} \frac{|K| \omega_j}{3} (\nabla_{\mathbf{X}} \varphi_i \nabla_{\mathbf{X}} \varphi_j)^K = \sum_{K \in \mathscr{D}_i} |K| \, (\bar{\omega} \nabla_{\mathbf{X}} \varphi_i \nabla_{\mathbf{X}} \varphi_j)^K \\ &= \sum_{K \in \mathscr{D}_i} \bar{\omega}^K \left(\frac{\mathbf{n}_{\mathbf{X}_i} \cdot \mathbf{n}_{\mathbf{X}_j}}{4|K|} \right)^K = \sum_{K \in \mathscr{D}_i} a_{ij}^K \quad \forall i \in \mathscr{T}_h^{in} \end{aligned} \tag{127}$$

It is evident that the geometrical quantities that we need are the same normals scaled by the lenght of the edge that are used in the RD method. To assemble the matrix K a simple loop on the elements is performed: for each K a submatrix of elements k_{ij}^K (3×3 for triangular mesh) is computed and then the elements of K are assembled summing for each index i, j the single elemental contributions. Note also that $\mathsf{K} = \mathsf{K}(\omega_h, \mathscr{T}_h^X)$; it consists in a part depending on the reference configuration that can be computed once at the beginning of the computation and a part depending on ω which has to be updated with the solution u_h. As a consequence K is nonlinear with respect to x, thus the system should be written as

$$\mathsf{K} = \mathsf{K}(\omega(\mathbf{x}_j), \mathscr{T}_h^0) = \mathsf{K}(\mathsf{x}), \quad \mathsf{K}(\mathsf{x}) \, \mathsf{x} = \mathsf{f} \tag{128}$$

The monitor function, as it has been presented in formula (120), cannot be used for general problems but should undergo some processing steps.

Gradient/Hessian reconstruction

Up to now nothing has been said regarding the computation of the monitor function through the formula (120). From one of the intermediate steps of (127) emerges that the nodal values of ω are required, which is quickly translated in the availabilty of the nodal values of ∇u and $\nabla^2 u$. The simplest way to obtain such values is through the following weighted area average

$$\nabla u_i = \frac{\sum_{K \in \mathscr{D}_i} (\nabla u)^K |K|}{\sum_{K \in \mathscr{D}_i} |K|}$$

$$\nabla^2 u_i = \frac{\sum_{K \in \mathscr{D}_i} (\nabla \nabla u)^K |K|}{\sum_{K \in \mathscr{D}_i} |K|}$$

$(\nabla \cdot)^K$ is the exact gradient of a P1 function (either u or ∇u) over an element computed with standard formulas. L_2-norms then are used

$$\begin{aligned} ||\nabla u_i|| &= \sqrt{(\nabla_x u_i)^2 + (\nabla_y u_i)^2} \\ ||\nabla^2 u_i|| &= \sqrt{(\nabla^2_{xx} u_i)^2 + (\nabla^2_{yy} u_i)^2 + 2(\nabla^2_{xy} u_i)^2} \end{aligned}$$

Scaling

A more general monitor function is obtained through the following scaling of the reconstructed gradient/hessian suggested in [13]

$$\nabla u_i = \min\left(1, \frac{||\nabla u_i||}{\beta \max_{i\in\mathscr{T}_h}||\nabla u_i||}\right)$$
$$\nabla^2 u_i = \min\left(1, \frac{||\nabla^2 u_i||}{\gamma \max_{i\in\mathscr{T}_h}||\nabla^2 u_i||}\right)$$

the nodal values are limited, $\nabla u_i, \nabla^2 u_i \in [0,1]$, whichever is the problem under study, thus in (120) the upper bound of ω is controlled only through the parameter α. Here $\beta, \gamma \in [0,1]$ in order to cut the peaks above a certain threshold measured in percentage of the maximum value.

8.3 Nonlinear system

Nonlinearity of the block (128) results in the nonlinearity of the system (126), thus some iterations of the Newton method should be used, leading to the solution of a linear system at each k-th Newton iteration

$$\mathsf{A}^{[k]} = \mathsf{A}(\mathsf{x}^{[k]}) \tag{129}$$
$$\mathsf{r}^{[k]} = \mathsf{A}^{[k]}\mathsf{x}^{[k]} - \mathsf{b} \tag{130}$$
$$\mathsf{p}^{[k]} = -\left(\mathsf{A}^{[k]}\right)^{-1}\mathsf{r}^{[k]} \tag{131}$$
$$\hat{\mathsf{x}}^{[k]} = \mathsf{x}^{[k]} + \mathsf{p}^{[k]} \tag{132}$$
$$\mathsf{x}^{[k+1]} = \hat{\mathsf{x}}^{[k]}\mu + \mathsf{x}^{[k]}(1-\mu) \tag{133}$$

A first problem that we face is the fact that $\mathsf{A}^{[k]}$ is not directly accessible because of the following consideration that stems from writing explicitly the dependence in (128)

$$\mathsf{K}^{[k]} = \mathsf{K}(\omega(u_h(\mathsf{x}^{[k]})) \tag{134}$$

thus in order to evaluate expression (134) an intermediate interpolation step at every Newton iteration is necessary to take into account the true nonlinear character of the underlying moving mesh equation

$$u_h^{[k]} = u_h^n(\mathsf{x}^{[k]}) \tag{135}$$

where $u_h^{[k]}$ is the interpolation of u_h^n on the updated mesh $\mathsf{x}^{[k]}$. The interpolation step becomes a delicate issue for an adaptive mesh algorithm since it can be very expensive, leading to a dramatical loss of utility for the whole method proposed, and even worse, it can spoil the numerical solution of positivity, accuracy and conservativeness. This will be addressed in Sec.(9).

An under-relaxation step (133) has been added. If a Jacobi method is used, following the fact that K is diagonally dominant, inner nodes are moved according to a convex combination of the position of the neighbouring grid points at each iteration. Thus an inner point should remain within its dual cell. With a relaxation factor $\mu = 1/2$, Chen in [13] limits further grid points movement since, at each iteration step, a node i must lie within the convex hull formed by the midpoints of the edges sharing i.

However many defintions for the scalar $\mu \in [0,1]$ are possible, also a local definition μ_i. In the present computations we take

$$\mu_i = \max(\sigma, \tau\nabla u_i)$$

with σ and τ introduced to better control mesh distortion, avoiding an excessive nodes' deplation in region where the gradient is not significative. A minimun threshold for the stiffness is tuned by fixing σ, if $\sigma \sim 0$ the stiffness in regions where $\nabla u \sim 0$ is strongly increased. In our computation $\sigma = 0.7$.

8.3.1 Newton-Jacobi method

Methods for general sparse matrix are available, such as LU factorization of the full A matrix, followed by backward and forward substitution with the triangular factors. Using the symmetry of the matrix, it is possible to perform a symmetric indefinite factorization which reduce the computational cost typically about the half respect to Gaussian elimination

$$\mathsf{A} = \mathsf{L}\mathsf{D}\mathsf{L}^T$$

The CPU time however remain too high. A possible way to overcome the above problems consists in using iterative methods. On the other hand the Coniugate Gradient method, is not recommended because it can be unstable for matrices that are not positive definite.

Decoupling the computation of the increment $\mathsf{p}^{[k]}$ corresponds to apply a Jacobi iteration

$$\mathsf{p}^{[k]} = -\left(\mathrm{Diag}(a_{ii}^{[k]})\right)^{-1} \mathsf{r}^{[k]} \tag{136}$$

The last inversion results quite inexpensive. Substituting it into the Newton iteration we obtain a Jacobi iteration

$$\hat{\mathsf{x}}^{[k+1]} = \mathsf{x}^{[k]} - \frac{1}{a_{ii}^{[k]}}\left(\mathsf{A}^{[k]}\mathsf{x}^{[k]} - \mathsf{b}\right) \tag{137}$$

From (126)(127) one can immediatly see that A is symmetric however presents zero entries on the diagonal, thus discouraging a simple Jacobi method. A possible solution is to sum to the last m rows (namely the last block in (126)), the first and the third block rows from matrix K which have non-zero diagonal elements. Unfortunately a continuous projection of the boundary nodes on the respective boundary segments is necessary since accumulation errors drive them to move away, as pointed out also by [18].

8.4 Displacement formulation and Newton-Jacobi method

In this section we suggest another way for imposing boundary conditions. The idea is that free-slip boundary condition could be stated obviuosly in terms of displacement such as

$$\Delta n = \Delta \mathbf{x} \cdot \mathbf{n}_{\mathbf{X}} = 0 \tag{138}$$

Step by step the following procedure is used to impose boundary condition in the form (138)

- rewrite the FE approximation (124) in terms of displacements
- decompose the boundary nodes displacement respectively into a normal and tangent component to the boundary
- impose the Dirichlet condition (138) on the normal displacement
- write back the tangential component of the diplacement in cartesian coordinates

At every time-step we consider the actual position as the sum of the position at the previous time instant and plus a displacement

$$\Delta \mathbf{x}_j \quad = \quad \mathbf{x}_j - \mathbf{x}_j^n$$

which substituted in (124) gives the following nonlinear system

$$\mathsf{K}(\mathsf{x})\,\Delta\mathsf{x} = \mathsf{F}(\mathsf{x}) \tag{139}$$

with $\mathsf{F} = -\mathsf{K}(\mathsf{x})\mathsf{x}^n + \mathsf{f}$ and with $\Delta\mathsf{x}$ collecting all the nodes displecements $\Delta\mathbf{x}_j$. The next step consists in decomposing each boundary node displacement vector subjected to free-slip into a normal displacement and into a tangential displacement. Thus if $\mathbf{x}_j \in \partial\Omega_X^{slip}$, $\Delta\mathbf{x}_j$ can be expressed as

$$\begin{pmatrix} \Delta x_j \\ \Delta y_j \end{pmatrix} = \begin{pmatrix} \hat{\mathbf{n}}_{\mathbf{X}_j} & \hat{\mathbf{n}}^{\perp}_{\mathbf{X}_j} \end{pmatrix} \begin{pmatrix} \Delta n_j \\ \Delta n_j^{\perp} \end{pmatrix}$$

where $\hat{\mathbf{n}}_{\mathbf{X}}$ is the normal versor to the boundary. It is immediate now to impose condition (138)

$$\begin{pmatrix} \Delta x_j^{slip} \\ \Delta y_j^{slip} \end{pmatrix} = \begin{pmatrix} \hat{n}^{\perp}_{X_j} \\ \hat{n}^{\perp}_{Y_j} \end{pmatrix} \Delta n_j^{\perp} \tag{140}$$

where the apex $(\cdot)^{slip}$ means that the displacement component respects a free-slip condition. The tangential component of the displacement is decomposed back into cartesian coordiantes

$$\Delta n_j^{\perp} = \begin{pmatrix} \hat{\mathbf{x}}\cdot\hat{\mathbf{n}}_{\mathbf{X}_j} & \hat{\mathbf{y}}\cdot\hat{\mathbf{n}}_{\mathbf{X}_j} \end{pmatrix} \begin{pmatrix} \Delta x_j \\ \Delta y_j \end{pmatrix} \tag{141}$$

since $\hat{\mathbf{x}} = \begin{pmatrix} 1 & 0 \end{pmatrix}$, $\hat{\mathbf{y}} = \begin{pmatrix} 0 & 1 \end{pmatrix}$ we can write

$$\Delta n_j^{\perp} = \begin{pmatrix} \hat{n}^{\perp}_{X_j} & \hat{n}^{\perp}_{Y_j} \end{pmatrix} \begin{pmatrix} \Delta x_j \\ \Delta y_j \end{pmatrix}$$

Substituting (141) into (140) and using the property of the scalar product $a\,(a^{\perp}\cdot b) = (a\cdot a^{\perp})\,b$

$$\begin{pmatrix} \Delta x_j^{slip} \\ \Delta y_j^{slip} \end{pmatrix} = \begin{pmatrix} \hat{n}^{\perp}_{X_j}\hat{n}^{\perp}_{X_j} & \hat{n}^{\perp}_{X_j}\hat{n}^{\perp}_{Y_j} \\ \hat{n}^{\perp}_{Y_j}\hat{n}^{\perp}_{X_j} & \hat{n}^{\perp}_{Y_j}\hat{n}^{\perp}_{Y_j} \end{pmatrix} \begin{pmatrix} \Delta x_j \\ \Delta y_j \end{pmatrix}$$

Imposing the solution vector of system (139) such as to ensures the satisfaction of free-slip boundary condition at each $j-th$ boundary node $\in \partial\Omega_X$

$$\begin{pmatrix} \Delta\mathsf{x}^{\partial} \\ \Delta\mathsf{x}^{in} \\ \Delta\mathsf{y}^{\partial} \\ \Delta\mathsf{y}^{in} \end{pmatrix} = \begin{pmatrix} \Delta\mathsf{x}^{slip} \\ \Delta\mathsf{x}^{in} \\ \Delta\mathsf{y}^{slip} \\ \Delta\mathsf{y}^{in} \end{pmatrix}$$

results in the following projection $\Delta\mathsf{x}^{slip} = \mathsf{P}\Delta\mathsf{x}$

$$\begin{pmatrix} \Delta\mathsf{x}^{slip} \\ \Delta\mathsf{x}^{in} \\ \Delta\mathsf{y}^{slip} \\ \Delta\mathsf{y}^{in} \end{pmatrix} = \begin{pmatrix} \mathsf{P}_{xx} & 0 & \mathsf{P}_{xy} & 0 \\ 0 & \mathsf{I} & 0 & 0 \\ \mathsf{P}_{yx} & 0 & \mathsf{P}_{yy} & 0 \\ 0 & 0 & 0 & \mathsf{I} \end{pmatrix} \begin{pmatrix} \Delta\mathsf{x}^{\partial} \\ \Delta\mathsf{x}^{in} \\ \Delta\mathsf{y}^{\partial} \\ \Delta\mathsf{y}^{in} \end{pmatrix}$$

Finally the system takes the usual form

$$\mathsf{A}\,\Delta\mathsf{x} = \mathsf{b}$$

where $\mathsf{A} = \mathsf{K}\mathsf{P}$ and $\mathsf{b} = \mathsf{F}$. Note that the problem is now coupled, such as it happens for the case of lagrangian multipliers.

8.4.1 Newton-Jacobi method

The nonlinear system can be solved with the Newton-Jacobi method already presented in Sec.(8.3.1). The only difference lies in the known term which is nonlinear and has to be updated $\mathsf{b}^{[k]} = -\mathsf{K}^{[k]}\mathsf{x}^{[n]} + \mathsf{f}$.

$$\hat{\Delta\mathsf{x}}^{[k+1]} = \Delta\mathsf{x}^{[k]} - \frac{1}{a_{ii}^{[k]}}\left(\mathsf{A}^{[k]}\Delta\mathsf{x}^{[k]} - \mathsf{b}^{[k]}\right) \tag{142}$$

$$\mathsf{x}^{[k+1]} = \mathsf{x}^{[k]} + \hat{\Delta\mathsf{x}}^{[k]}\mu \tag{143}$$

9 Conservative Interpolation

Looking to (135) the implementation of an interpolation algorithm becomes mandatory if one wants to compute an approximation for $u_h^{[k]}$. The problem is the following: given a reference configuration $\mathbf{X}$ and function $u(\mathbf{X})$, assuming that the domain is transformed according to the usual arbitrary mapping $\mathbf{x} = A(\mathbf{X},t)$, we want to evaluate function u over the transfomed domain, and we call it $\tilde{u}$

$$\tilde{u} = u(\mathbf{x},t)$$

The problem is expressed by the following conservation law which state that the integral of any function defined over a close fixed domain C is conserved

$$\frac{\partial}{\partial t}\int_C \tilde{u}\, d\mathbf{x} = 0 \tag{144}$$

If the problem is transformed into ALE form, (cf. eq. (5)(6) with $\mathbf{a} = 0$) the conservation law writes as follows

$$\left.\frac{\partial}{\partial t}\right|_X \int_{C(t)} \tilde{u}\, d\mathbf{x} - \int_{C(t)} \nabla\cdot(\sigma\tilde{u})\, d\mathbf{x} = 0 \tag{145}$$

with σ defined as usual by (2). The interpolation has been reformalized into a typical hyperbolic problem: the solution is updated by the boundary flux of the solution which is advected by the domain movement. It is not a coincidence that it is usually referred to as *advection remap*.

At this point an approximation for (145) is needed. The RD method, largely studied in Chapter 1, can be succesfully applied to this problem. Using the nomenclature of (135), namely $u_h^{[k]} = \tilde{u}_h(\mathbf{x}^{[k]})$ with $u_h^{[0]} = \tilde{u}_h(\mathbf{x}^n) = u_h^n$

$$u_i^{[k+1]} = u_i^{[k]} - \frac{\Delta t}{|\tilde{S}_i^{k+1/2}|}\sum_{K\in\mathscr{D}_i}\phi_i^K \tag{146}$$

with the total residual

$$\phi^K = -\int_{K^{k+1/2}} \sigma_h^*\cdot\nabla u_h^{[k]}\, d\mathbf{x}$$

We recall that σ, as usual, is considered constant within the Newton iteration

$$\sigma_i^* = \frac{\mathbf{x}_i^{[k+1]} - \mathbf{x}_i^{[k]}}{\Delta t}$$

In general, to have a nice interpolated solution, the algorithm (146) has to fulfill many properties. First of all conservativeness, which corresponds to satisy at a discrete level equation (144) for the whole domain. We end up with the following condition

$$\sum_{K\in\mathscr{T}_h}\int_{K^{[k+1]}} u_h^{[k+1]}\, d\mathbf{x} = \sum_{K\in\mathscr{T}_h}\int_{K^n} u_h^n\, d\mathbf{x}$$

which can be shown to be satisfied by the scheme (146).

Secondly, the interpolation algorithm should be accurate. Starting from a second order accurate solution, a few iteration of a first order interpolation scheme could spoil the global order of accuracy. Third, it is natural to demand for some stability bounds for (146). As already seen the accuracy and stability properties of the interpolation step depend on how we distribute the residual ϕ_i^K. In the numerical test we will see how different distributions influence the property of the solution.

Finally consistency. In the limit of $\mathbf{x} \to \mathbf{X}$, or $\mathbf{x}^{[k+1]} \to \mathbf{x}^n$, we expect that $u_i^{[k]} \to u_i^n$ which is also true for (146).

10 Adaptive algorithms

Now that we have given all details of the method, we provide the step by step procedure that we used in the algorithm. The idea is to couple the elliptic equation (119) with u, solution of the hyperbolic balance law (1) recall below

$$\frac{\partial u}{\partial t} + \nabla \cdot \mathbf{f} + S = 0 \tag{147}$$

in order to obtain a mesh that adapts continuously to the evolving solution. Depending on the framework in which we evolve the above PDE, two different algorithms are tested.

10.1 Conservation law in ALE form (ALE alg.)

Conservation law is written, by means of the ALE formulation, directly in a framework coincindent with the moving domain, that is Eq. (16). At every time step we get the solution on the adapted grid and no interpolation step is needed.

The RD method in ALE framework has been studied extensively in Ch. 1 where we have presented stability and accuracy results. In particular for the LLxF-SUPG scheme used in the computations we expect second order of accuracy and a non oscillatory behaviour near discontinuities. To present a fairest and deepest comparison a Well Balanced FV method in ALE form has been also discussed and implemeted. In the MUSCL version it should compute second order accurate solutions and sharp/monotone discontinuities.

We recall that x is the collection of the nodes'x-coordinates x_i into a column vector. Similarly we define u the vector that collects the solution at every node, u_i. The RD method with grid adaptation writes as follows.

`Step 1.` Take an initial triangular mesh $\mathscr{T}_h^0$, from which we compute the vectors x^0, together with the initial solution u_0 which has to be discretized over the same grid. Let $\mathsf{x}^n = \mathsf{x}^0$ and $\mathsf{u}^n = \mathsf{u}^0$

`DO k=1,kmax`

`Step 2.` Compute the interpolated solution $\mathsf{u}^{[k]}$ according to (146) with initial condition $\mathsf{u}^{[0]} = \mathsf{u}^n$. Compute the monitor function $\omega_h^{[k]}(\mathsf{u}^{[k]})$ and matrix $\mathsf{A}^{[k]} = \mathsf{A}\left(\omega_h^{[k]}, \mathscr{T}_h^0\right)$. Move the mesh according to the Newton-Jacobi iteration of Sec. (8.4.1). (Eq. (142) and (143)) with initial condition $\mathsf{x}^{[0]} = \mathsf{x}^n$. At each iteration we get $\mathsf{x}^{[k+1]}$. We used `kmax=5`. Let $\mathsf{x}^{n+1} = \mathsf{x}^{[kmax]}$, thus we get the adapted mesh $\mathscr{T}_h^{n+1}$ at the new time step

`ENDDO`

`Step 3.` Evolve the underlying conservation law in ALE framework with the RD/FV-RK2 scheme (see Eq. (78) and Eq. (95)) on the midpoint grid $\mathscr{T}_h^{n+1/2}$, with initial condition u^n. We get u^{n+1}

Step 4. Let $\mathsf{x}^{n+1} = \mathsf{x}^n$ and $\mathsf{u}^{n+1} = \mathsf{u}^n$

IF ($\mathsf{t} > \mathsf{T}$**) EXIT**

ENDDO

The above algorithm can be implemented in a straigthforward way. What we still miss is the distribution strategy for the interpolation step. Since the interpolated solution is used only to compute correctly the updated mesh and does not influence, at least directly, the numerical solution one could not be interested in its accuracy and stability. Infact, after testing many distributions, the results between linear and nonlinear schemes were very similar, thus demonstrating that a simple linear first order scheme could be succesfully used for the interpolation step. Into a RD context we could use a simple LxF or even a Galerkin scheme while in a FV context a first order upwind FV scheme or a centered approximation of the fluxes.

10.2 Conservation law in Eulerian form (EUL1 alg.)

Conservation law is resolved numerically at every time instant in an Eulerian framework coicident with the adapted mesh. The drawback of this approach lies in the fact that, once the grid has been adapted, we have to interpolate the solution over the new grid to get the initial condition for the new time iteration.

In this case the RD method with grid adaptation reads as follows.

Step 1. Take an initial triangular mesh $\mathscr{T}_h^0$, from which we compute the vectors x^0, together with the initial solution u_0 which has to be discretized over the same grid Let $\mathsf{x}^n = \mathsf{x}^0$ and $\mathsf{u}^n = \mathsf{u}^0$

DO k=1,5

Step 2. Compute the interpolated solution $u_h^{[k]}$ according to (146) with initial condition $\mathsf{u}^{[0]} = \mathsf{u}^n$. Compute the monitor function $\omega_h^{[k]}(\mathsf{u}^{[k]})$ and matrix $\mathsf{A}^{[k]} = \mathsf{A}\left(\omega_h^{[k]}, \mathscr{T}_h^0\right)$. Move the mesh according to the Newton-Jacobi iteration of Sec. (8.4.1). (Eq. (142) and (143)) with initial condition $\mathsf{x}^{[0]} = \mathsf{x}^n$. At each iteration we get $\mathsf{x}^{[k+1]}$. We used `kmax=5`. Let $\mathsf{x}^{n+1} = \mathsf{x}^{[kmax]}$, thus we get the adapted mesh $\mathscr{T}_h^{n+1}$ at the new time step. Moreover we get $\tilde{\mathsf{u}} = \mathsf{u}^{[kmax]}$, the interpolated solution over the new mesh.

ENDDO

Step 3. Evolve the underlying conservation law in Eulerian framework using the RD/FV-RK2 scheme (see Eq. (78) and Eq. (95) with $\sigma = 0$) on the grid $\mathscr{T}_h^{n+1}$ with initial condition $\tilde{\mathsf{u}}$. We get u^{n+1}

Step 4. Let $\mathsf{x}^{n+1} = \mathsf{x}^n$ and $\mathsf{u}^{n+1} = \mathsf{u}^n$

IF ($\mathsf{t} > \mathsf{T}$**) EXIT**

ENDDO

Since this time the interpolated solution will act as the initial condition for the new time iteration, great care has to be put in its computation. The interpolation step does not have to spoil the accuracy property of the numerical scheme, in our case it should be at least second accurate. But also it does not have to introduce oscillations. This means that more complicated distributions respect to the first order ones are used, distributions that typically cost something more in terms of CPU time.

10.3 Conservation law in Eulerian form (EUL2 alg.)

In the previous algorithm, a double role emerges for the interpolation step: we need an interpolated solution $\mathsf{u}^{[k]}$ at every Newton sub-step in order to evolve the mesh and a last interpolated solution $\tilde{\mathsf{u}}$ on the finally updated mesh in order to provide an initial condition for the PDE solver. We will show that the mesh movement is weakly influenced by the interpolation, this leading to a possibile decoupling: a rough Galerkin interpolation is used in the Newton sub-steps and a proper interpolation ensuring the same properties of the flow solver, in terms of accuracy, stabilty ecc..., is used to compute the initial solution. If we are able to perform an accurate one-step interpolation $\tilde{u}_h = u_h^n(\mathsf{x}^{n+1})$

$$\tilde{u}_i = u_i^n - \frac{\Delta t}{|\tilde{S}_i^{n+1/2}|} \sum_{K \in \mathscr{D}_i} \phi_i^K \tag{148}$$

with the total residual

$$\phi^K = -\int_{K^{n+1/2}} \sigma_h^* \cdot \nabla u_h^n \, d\mathbf{x}$$

we could save some CPU time.

In this case the RD method with grid adaptation reads as follows.

Step 1. Take an initial triangular mesh $\mathscr{T}_h^0$, from which we compute the vectors x^0, together with the initial solution u_0 which has to be discretized over the same grid Let $\mathsf{x}^n = \mathsf{x}^0$ and $\mathsf{u}^n = \mathsf{u}^0$

DO k=1,5

Step 2. Compute the interpolated solution $u_h^{[k]}$ according to (146) with initial condition $\mathsf{u}^{[0]} = \mathsf{u}^n$. Compute the monitor function $\omega_h^{[k]}(\mathsf{u}^{[k]})$ and matrix $\mathsf{A}^{[k]} = \mathsf{A}\left(\omega_h^{[k]}, \mathscr{T}_h^0\right)$. Move the mesh according to the Newton-Jacobi iteration of Sec. (8.4.1). (Eq. (142) and (143)) with initial condition $\mathsf{x}^{[0]} = \mathsf{x}^n$. At each iteration we get $\mathsf{x}^{[k+1]}$. We used `kmax=5`. Let $\mathsf{x}^{n+1} = \mathsf{x}^{[kmax]}$, thus we get the adapted mesh $\mathscr{T}_h^{n+1}$ at the new time step.

ENDDO

Step 3. Compute properly the interpolated solution $\tilde{\mathsf{u}}$ according to (148).

Step 4. Evolve the underlying conservation law in Eulerian framework using the RD/FV-RK2 scheme (see Eq. (78) and Eq. (95) with $\sigma = 0$) on the grid $\mathscr{T}_h^{n+1}$ with initial condition $\tilde{\mathsf{u}}$. We get u^{n+1}

Step 5. Let $\mathsf{x}^{n+1} = \mathsf{x}^n$ and $\mathsf{u}^{n+1} = \mathsf{u}^n$

IF (t > T) EXIT

ENDDO

11 Numerical results

11.1 Mesh generators

We begin with testing Step 2 of the algorithms seen in Sec. (10). We performs the test cases contained in [15]. The monitor function is computed according to $\omega = \sqrt{1 + \alpha u^2}$, with u an assigned test function which should simulate

the gradient of some underlying solution.

$$u(x,y) = \exp(-8(x^2+9y^2-1)^2) \quad (149)$$

$$u(x,y) = \exp(-100(y-x^2+0.5)^2) \quad (150)$$

$$u(x,y) = 50\exp(-2500(x^2+y^2) \quad (151)$$

$$u(x,y) = \begin{cases} 1 & \text{if } y = x \\ 0 & \text{if } y \neq x \end{cases} \quad (152)$$

The reference domain is a square $[-1,1]\times[-1,1]$ discretized by means of a structured triangular mesh $\mathscr{T}_h^0$. Step 2 is repeated in the pseudo-time loop until convergence is reached. The control parameters are $\alpha = 100$ and $\beta = 1$. Following always [15], the results with homogeneous Dirichlet boundary condition are collected (see Fig. (7)). In the first two examples (149) and (150) the function u presents a strong gradient upon which the grid is strongly refined. In example (151) we show that also singularities could be well handled. Finally example (152) could be a test case for an oblique shock; the grid motion captures the discontinuty: we see a strong mesh refinement without tangling although the elements present a certain skewness.

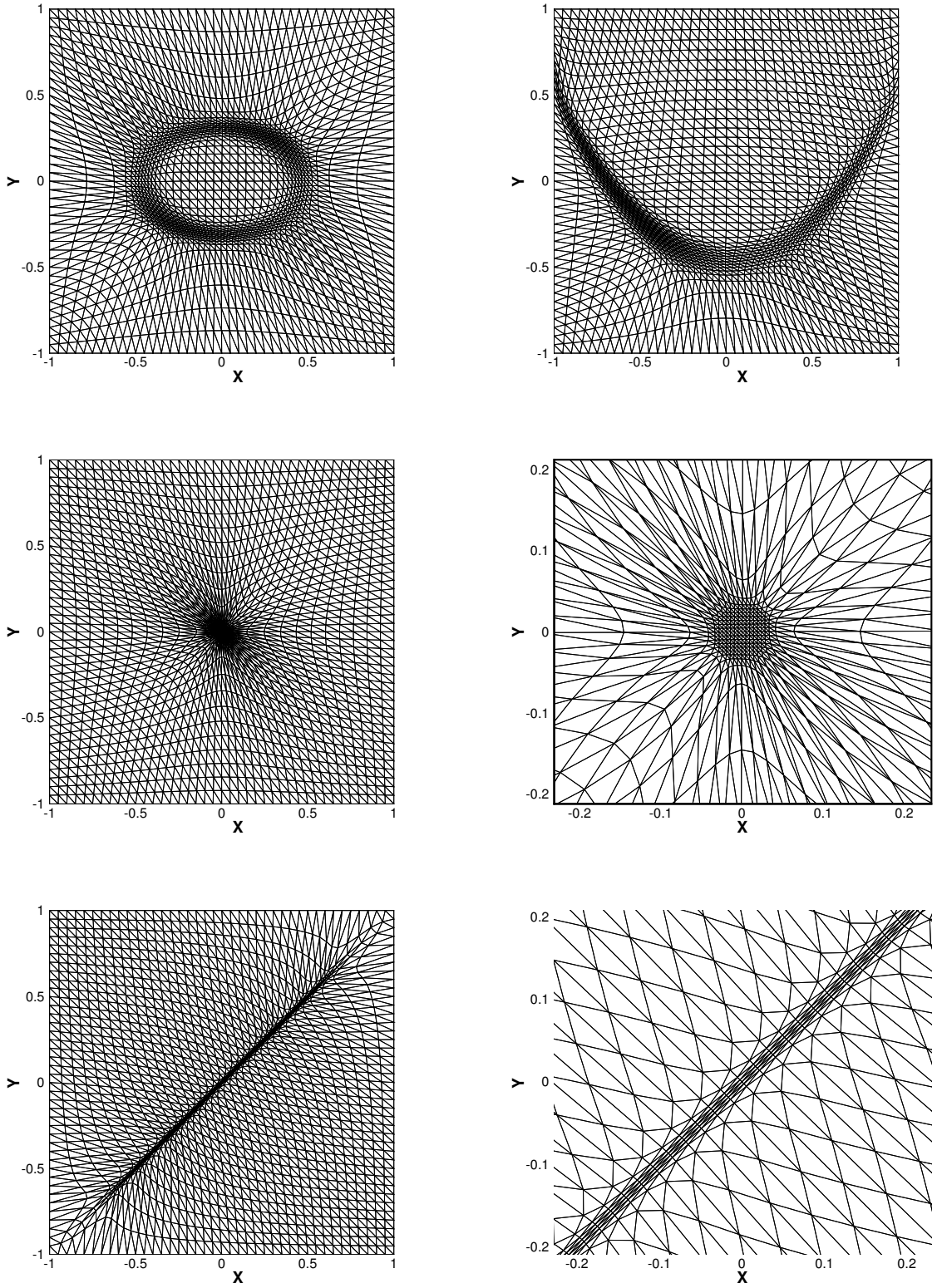


Figure 7: Mesh generator with fixed boundaries. From the top to bottom: test (149),(150), (151) with zoom of the singularity,(152) with zoom of the discontinuity.

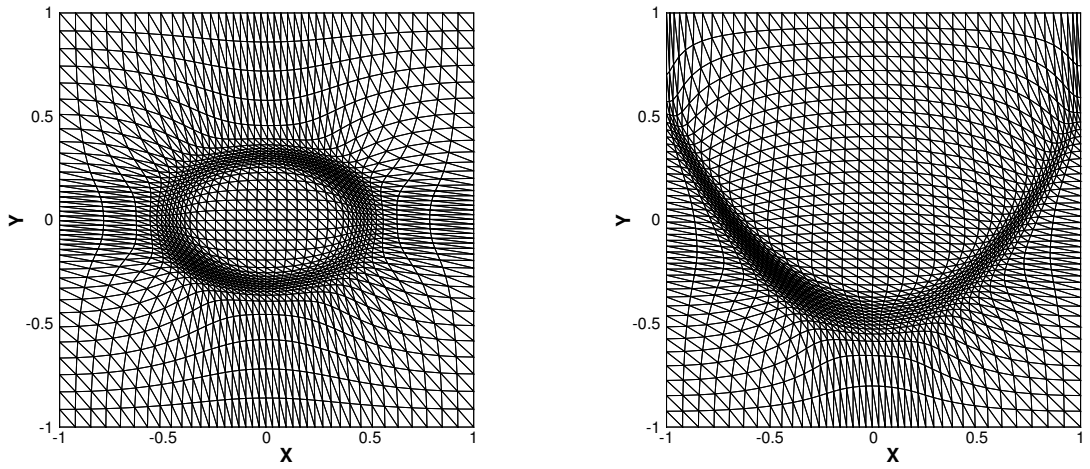


Figure 8: Mesh generator with free-slip boundaries. From the top to bottom: test (149),(150).

Regarding the formulation for the imposition of free-slip boundary condition only the displacement formulation seen in Sec. (8.4) has been considered and implemented for the next test cases. The closure with Lagrangian multipliers gives similar results with additional complexity. In Fig. (8) the first two previous examples are tackled.

11.2 Computational details

With the next test cases the different algorithms of Sec. (10) are tested and compared in terms of accuracy, ability to compute sharp discontinuities and performaces. In particular we focus the attention on the comparison between the numerical solution obtained in ALE framework and the one obtained in Eulerian framework with a conservative interpolation step (EUL1 and EUL2). The underlying PDE (147) is evolved with the numerical method of Sec. (5).

Concerning the adaptive step the following parameter are used $\alpha = 10$ and $\beta = \gamma = 0.15$. Here we did not perform a serious optimization relative to these parameters but we procedeed after a few trials.

11.3 Rotation

To test the accuracy of the different methods proposed in (10) we use the classical test case of rotation of a smooth sinusoidal hill, this time with a source term. In a quasi-linear form

$$\begin{cases} \frac{\partial u}{\partial t} + \mathbf{a}\cdot\nabla u + \mathbf{a}\cdot\nabla g = 0, & \mathbf{a} = [-2y, 2x]\,, \mathbf{x} \in [-1,1]\times[-1,1], t \in [0,\pi] \\ u_0(\mathbf{x}) = 1 - g(\mathbf{x}) + \cos^2(2\pi r) & \text{if } r \leq 0.25,\ r = \sqrt{x^2 + (y+0.5)^2} \\ u_0(\mathbf{x}) = 1 - g(\mathbf{x}) & \text{otherwise} \end{cases}$$

with

$$g(x,y) = 0.8e^{\psi(x,y)}, \quad \psi = -5y^2 - 5x^2$$

The accuracy is tested on 4 different structured meshes with element reference size respectively 1/12, 1/25, 1/50, 1/100. Two linear schemes are tested for the rotation problem: the SUPG method of Sec. (3.4.2) and the Fromm of Sec. (4.2).

In Fig. (9) are reported the convergence curves for the different combinations of moving mesh algorithms and numerical schemes tested. , With the moving mesh parameters that we have choosen an homegenous refinement on the cosine-hill region is perfomed such that the convergence curve of the adapted algorithms is shifted by a constant respect to the fixed grid one. The ALE algorithm is weakly influenced by the interpolation step necessary to evolve the mesh, almost all the curves in blue color are overlapped. For the EUL1 algorithm there is only one interpolation scheme which guarentees second order of accuracy, actually the one using the same scheme which we evolve the PDE with. As expected interpolations with first order scheme such us LxF or FV spoils the global order of convergence while dispersive interpolation with centered scheme (centered flux scheme (CFV) or centered distribution scheme (GAL)) gives erratic results. The EUL2 algorithm also gives accurate results.

In Fig. (10) the performances of the different algorithms are compared in terms of error/time. With the SUPG method, the ALE algorithm shows the lowest CPU time, for a fixed level of error (roughly 4 times faster then a fixed grid computation), respect to the EUL1 and EUL2 algorithm. This is due to the fact that in order to obtain an accurate interpolation the full two stage procedure of the RK had to be implemented, this causing a more expensive algorithm if compared to the quick one step Galerkin scheme implemented for the ALE algorithm. For the Fromm scheme the results between the EUL2 and the ALE algorithm are much more similar because this time, in the interpolation step, the second stage of RK2 seemed to be not necessary, as emerges also from the work of [15].

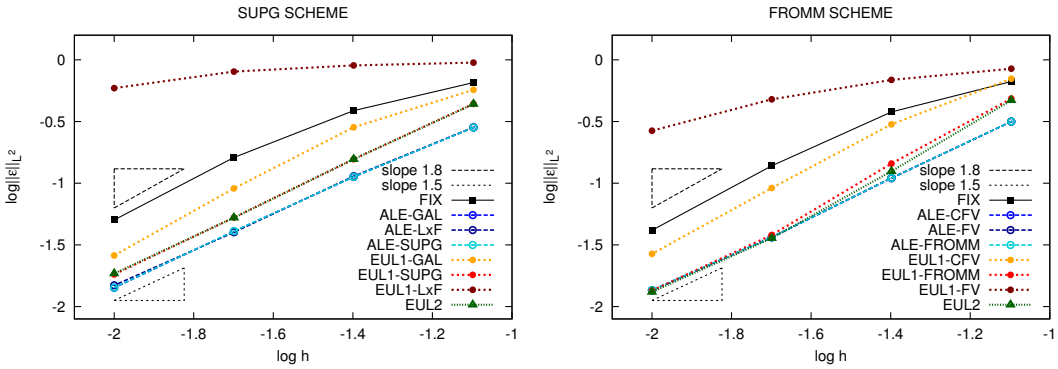


Figure 9: Rotation. Order of convergence: Left, SUPG. Right, Fromm.

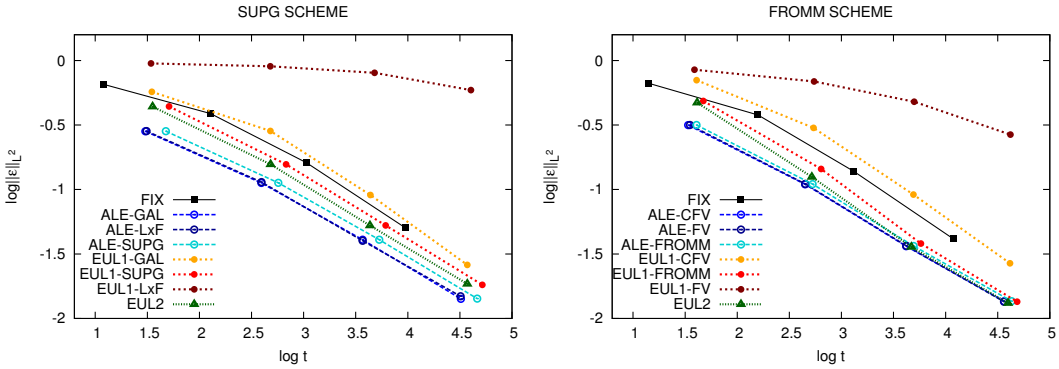


Figure 10: Rotation. Error vs CPU time: Left, SUPG. Right, Fromm.

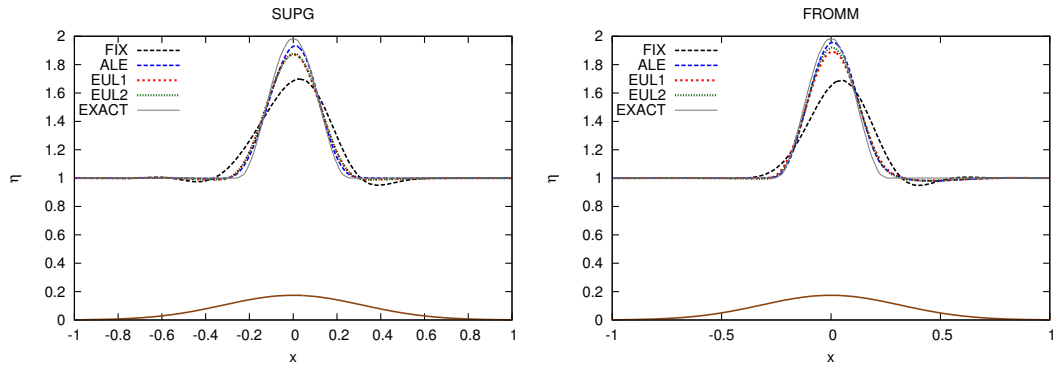


Figure 11: Rotation. Comparison between the algorithms ALE, EUL1 and EUL2 on the mesh with $h_K = 1/25$: Left, SUPG. Right, Fromm.

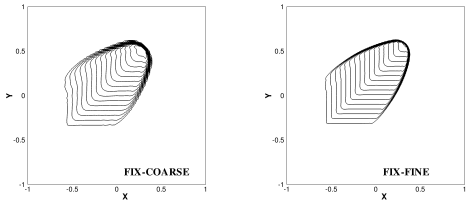


Figure 12: Burgers Equation computed with LLxF-SUPG scheme on fixed grid. Left: coarse mesh $h = 1/40$. Right: fine mesh $h = 1/100$.

11.4 Burgers equation

In this section we test if the adaptive mesh algorithm implemented in this report is effective when discontinuities develop. Solutions with discontinuities are obtained with a Burgers equation and discontinuous initial conditions.

$$\begin{cases} \frac{\partial u}{\partial t} + \nabla \cdot f = 0, & f = \left[u^2/2, u^2/2\right], x \in [-1,1] \times [-1,1], t \in [0,1] \\ u_0 = 1 & \text{if} x \in [-0.6,-0.1] \times [-0.35,0.15] \\ u_0 = 0 & \text{otherwise} \end{cases}$$

A reference solutions is computed on a fixed structured mesh with $h = 1/100$, which is referred to as the fine mesh. To test the effectivness of the adaptation algorithm computations are performed on fixed structured mesh with element reference size $h = 1/40$, which is referred to as coarse.

Again with the moving mesh parameter choosen before, a good mesh refinement is obtained in correspondence of the discontinuity with deformations which are locals, that means only elements near shocks undergo relevant distortions. The comparison between EUL1, EUL2 and ALE strategies is made for both the LLxF-SUPG and the MUSCL schemes.

In Fig. (13), the results for the residual based LLxF-SUPG scheme are showed. For the ALE algorithm we have tested many distribution scheme to perform the interpolation step within the adaptation routine: first order, second order or blended schemes does not affect significantly the mesh adaptation or quality, thus at the end we used the faster ones, the Galerkin and the LxF distributions. The mesh obtained with LxF is slightly smoother and as a results the shock is a little bit more smeared. The adapted algorithm produces a much accurate solution respect to the fixed grid, using roughly six times the CPU time. Apart from the time step which is smaller, the mesh movement algorithm involves two more steps at every Newton iteration: the Jacobi iteration and the interpolation step. However the time is much smaller then the one obtained with the fine grid.

For the Eulerian scheme we have seen that the interpolation step is fundamental for the property of the numerical solution, hence a good remap scheme must be used to not spoil accuracy and positivity properties. This is clear again in Fig.(13) ($3^{\text{rd}}, 4^{\text{th}}, 5^{\text{th}}$ and 6^{th} rows): the Galerkin remap does not spoil accuracy on the frontal discontinuties but, as expected, gives rise to oscillations, the positive LxF remap is not feasible due to its extremely diffusive property. The only distribution that allows an improvement of the adaptive solution respect to the fixed grid are the SUPG and the LLxF-SUPG schemes, which should remap the solution with a formal second order of accuracy.

For the FV method the same results holds. In the ALE algorithm one could use any distribution for the eulerian remap, without great changement in the mesh quality. Figure (15), obtained with a simple upwind FV remap, shows that the adapted solution could be successfully compared with the one obtained with the fine fixed grid.

For the Eulerian schemes, in order to not spoil the global properties of the numerical scheme, a second order and stable remap should be used. The upwind FV remap smear out too much the solution while the Fromm scheme compute an accurate solution but with no guarantees about positivity. In this case oscillation remains limited. Only the MUSCL remap ensure the preservation of both accuracy and positivity during the interpolation step.

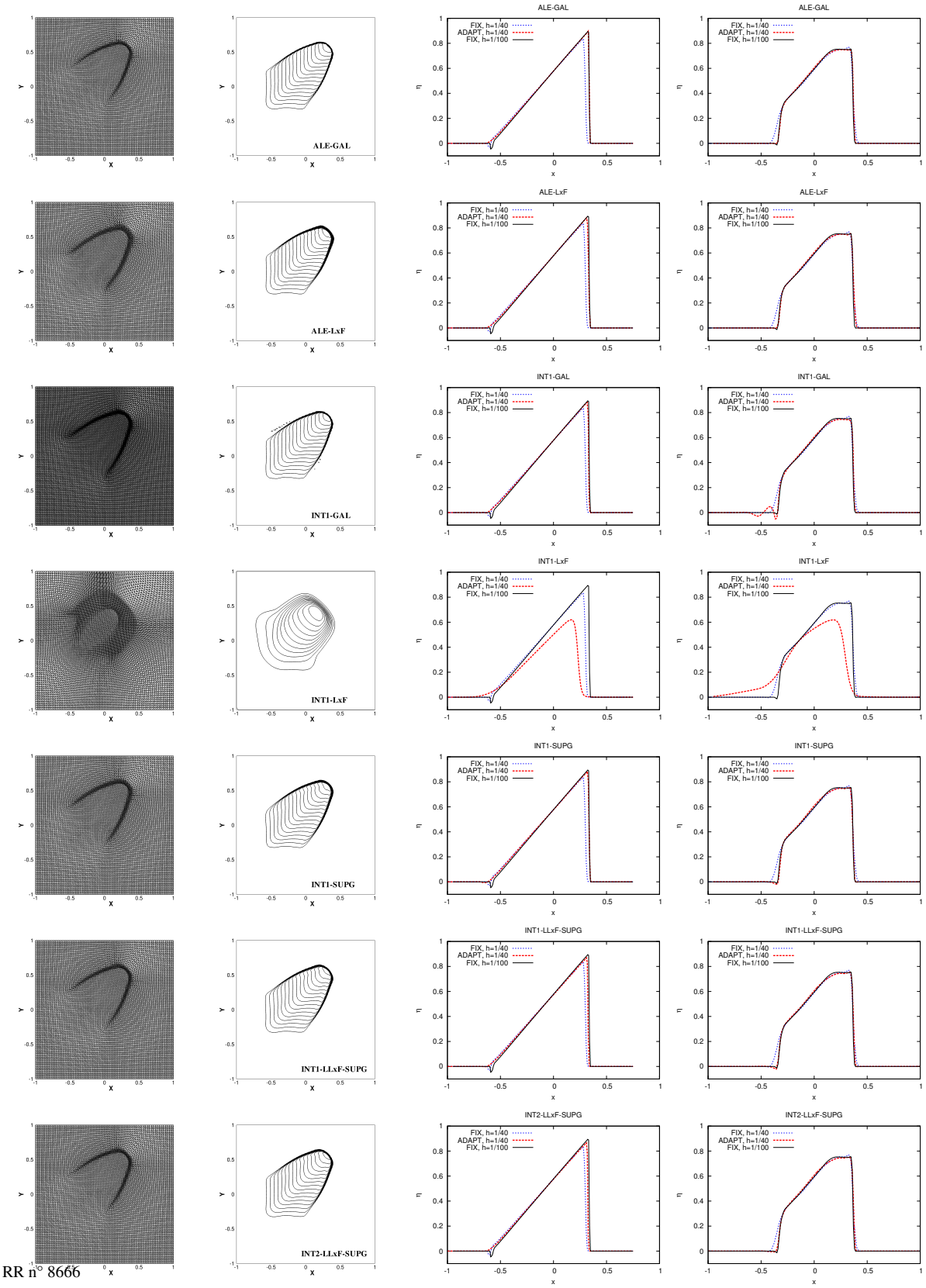


Figure 13: Burgers Equation computed with adaptive LLxF-SUPG scheme. From left to right: adapted grid at final time, 20 equispaced solution isolines between 0 and 1; comparison of the solution along the symmetry lines and the lines at $y = 0.4$

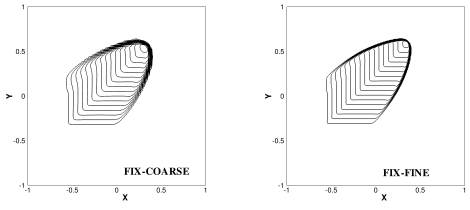


Figure 14: Burgers Equation computed with MUSCL scheme on fixed grid. Left: coarse mesh $h = 1/40$. Right: fine mesh $h = 1/100$.

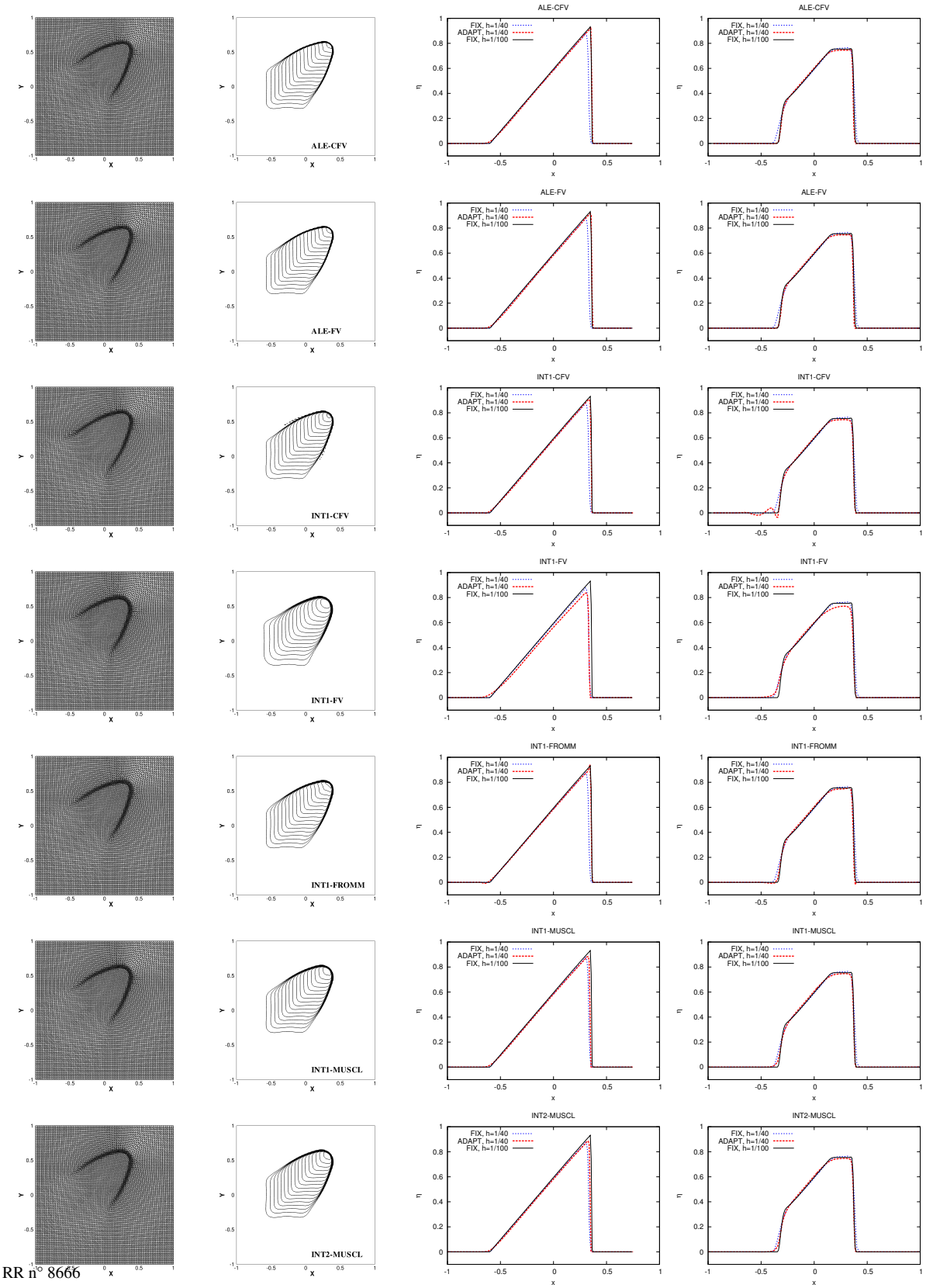


Figure 15: Burgers Equation computed with adaptive MUSCL scheme. From left to right: adapted grid at final time, 20 equispaced solution isolines between 0 and 1; comparison of the solution along the symmetry lines and the lines at $y = 0.4$

MESH	ALG.	PDE SOLVER	INTERP.	u_{min}	u_{max}	CPU TIME [s]
COARSE	FIXED	LLxF-SUPG	-	-0.051	0.841	58.52
FINE	FIXED	LLxF-SUPG	-	-0.051	0.905	619.75
COARSE	ADAPT. ALE	LLxF-SUPG	GAL	-0.015	0.903	346.08
COARSE	ADAPT. ALE	LLxF-SUPG	LxF	-0.003	0.883	378.5
COARSE	ADAPT. EUL1	LLxF-SUPG	GAL	-0.113	0.881	507.72
COARSE	ADAPT. EUL1	LLxF-SUPG	LxF	0.000	0.619	409.21
COARSE	ADAPT. EUL1	LLxF-SUPG	SUPG	-0.033	0.885	720.66
COARSE	ADAPT. EUL1	LLxF-SUPG	LLxF-SUPG	-0.037	0.872	774.60
COARSE	ADAPT. EUL2	LLxF-SUPG	GAL/LLxF-SUPG			

MESH	ALG.	PDE SOLVER	INTERP.	u_{min}	u_{max}	CPU TIME [s]
COARSE	FIX	MUSCL	-	-0.009	0.878	61.89
FINE	FIX	MUSCL	-	-0.007	0.934	647.51
COARSE	ADAPT. ALE	MUSCL	CFV	-0.012	0.935	369.22
COARSE	ADAPT. ALE	MUSCL	FV	-0.012	0.935	366.33
COARSE	ADAPT. EUL1	MUSCL	CFV	-0.114	0.908	517.85
COARSE	ADAPT. EUL1	MUSCL	FV	0.000	0.837	539.99
COARSE	ADAPT. EUL1	MUSCL	FROMM	-0.034	0.942	609.06
COARSE	ADAPT. EUL1	MUSCL	MUSCL	-0.002	0.885	620.00
COARSE	ADAPT. EUL2	MUSCL	CFV/MUSCL	-0.002	0.886	503.92

Acknowledgments

Work partially funded by the TANDEM contract, reference ANR-11-RSNR- 499 0023-01 of the French Programme Investissements d'Avenir.

References

[1] M. Lesoinne and C. Farhat. Geometric conservation laws for flow problems with moving boundaries and deformable meshes, and their impact on aeroelastic computation. *Computer methods in applied mechanics and engineering*, 134(1-2):71–90, 1996.

[2] M. Ricchiuto. *Construction and analysis of compact residual discretizations for conservation laws on unstructured meshes*. PhD thesis, Von Karman Institute, 2005.

[3] R. Struijs. *A multidimensional upwind discretization method for the Euler equations on unstructured grids*. PhD thesis, Von Karman Institute, 1994.

[4] R. Abgrall. Essentially non oscillatory residual distribution schemes for hyperbolic problems. *J. Comput. Phys*, 214(2):773–808, 2006.

[5] D.Isola. *An Interpolation Free Two-Dimensional Conservative ALE scheme over Adaptive Unstructured Grids for Rotorcraf Aerodynamics*. PhD thesis, Politecnico di Milano. Dipartimento di Ingegneria Aerospaziale, 2012. Sec. 2.4.1.

[6] I.K. Nikolos and A.I. Delis. An unstructured node-centered finite volume scheme for shallow water flows with wet/dry fronts over complex topography. *Comput. Methods Appl. Mech. Engrg.*, 198:3723–3750, 2009.

[7] R.J. LeVeque. High-resolution methods. In *Finite Volume Methods for Hyperbolic problems*, chapter 6, page 107. Cambridge University Press, 2004.

[8] L. Arpaia, M. Ricchiuto, and R. Abgrall. An ale formulation for explicit runge-kutta residual distribution. *J. Sci. Comput.*, 190(34):1467–1482, 2014.

[9] M. Ricchiuto and R. Abgrall. Explicit runge-kutta residual distribution schemes for time dependent problems: Second order case. *J. Comput. Phys*, 229(16):5653 – 5691, 2010.

[10] J.F. Thompson and N.P. Weatherill. Fundamental concepts and approaches. In J.F. Thompson, B.K. Soni, and N.P. Weatherill, editors, *Hanbook of Grid Generation*, chapter 1, page 10. CRC Press, 1999.

[11] A. Winslow. Numerical solution of the quasi-linear poisson equation. *J. Comput. Phys*, 1:149–172, 1967.

[12] S.P. Spekreijse. Elliptic generation systems. In J.F. Thompson, B.K. Soni, and N.P. Weatherill, editors, *Hanbook of Grid Generation*, chapter 4, page 5. CRC Press, 1999.

[13] G. Chen, H. Tang, and P. Zhang. Second-order accurate godunov scheme for multicomponent flows on moving triangular meshes. *J. Sci. Comput*, 34:64–86, 2008.

[14] M.J. Baines and M.E. Hubbard. Multidimensional upwinding for grid adaptation. *Numerical Methods for Wave Propagation*, pages 33–54, 1998.

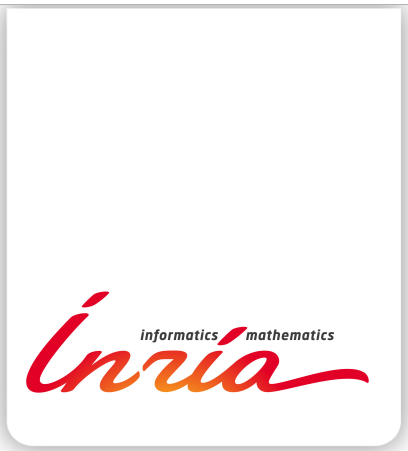
[15] H. Tang and T. Tang. Adaptive mesh methods for one and two-dimensional hyperbolic conservation laws. *SIAM J. Numer. Anal.*, 41(2):487–515, 2003.

[16] W. Cao, W. Huang, and R. D. Russell. An r-adaptive finite element method based upon moving mesh pdes. *J. Comput. Phys*, 149:221–244, 1998.

[17] I. Babuska. A moving mesh finite element algorithm for singular problems in two and three space dimensions. *Numerische Mathematik*, 20:179–192, 1972/73.

[18] R. Li, T. Tang, and P. Zhang. A moving mesh finite element algorithm for singular problems in two and three space dimensions. *J. Comput. Phys*, 177(2):365–393, 2002.

[19] J. Nocedal and S.J. Wright. Quadratic programming. In P.G. Stephen and M. Robinson, editors, *Numerical Optimization*, chapter 16, page 439. Springer, 1999.



RESEARCH CENTRE
BORDEAUX – SUD-OUEST

351, Cours de la Libération
Bâtiment A 29
33405 Talence Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399