An immersed boundary method using unstructured anisotropic mesh adaptation combined with level-sets and penalization techniques

Remi Abgrall, Héloïse Beaugendre, Cécile Dobrzynski

An immersed boundary method using unstructured anisotropic mesh adaptation combined with level-sets and penalization techniques

R. Abgrall, H. Beaugendre , C. Dobrzynski

RESEARCH REPORT
N° 8228
March 2012
Project-Team Bacchus

ISSN 0249-6399 ISRN INRIA/RR--8228--FR+ENG

An immersed boundary method using unstructured anisotropic mesh adaptation combined with level-sets and penalization techniques

R. Abgrall*, H. Beaugendre †, C. Dobrzynski ‡

Project-Team Bacchus

Research Report n° 8228 — March 2012 — ?? pages

Abstract:
The interest on embedded boundary methods increases in Computational Fluid Dynamics (CFD) because they simplify then mesh generation problem in the case of the Navier-Stokes equations. The same simplifications occur for the simulation of multi-physics flows, the coupling of fluid-solid interactions in situation of large motions or deformations, to give a few examples. Nevertheless an accurate treatment of the wall boundary conditions remains an issue of the method. In this work, the wall boundary conditions are easily taken into account through a penalization technique, and the accuracy of the method is recovered using mesh adaptation, thanks to the potential of unstructured meshes. Several classical examples are used used to demonstrate that claim.

Key-words: Penalization technique, unstructured mesh, level-set, anisotropic mesh, mesh adaptation, embedded method, Navier Stokes equations.

* INRIA Bordeaux–Sud-Ouest & Institut de Mathématiques de Bordeaux, Team-project Bacchus, 200 route de la Vieille Tour, 33405 Talence, FRANCE
† Institut de Mathématiques de Bordeaux & INRIA Bordeaux–Sud-Ouest, Team-project Bacchus, 200 route de la Vieille Tour, 33405 Talence, FRANCE
‡ Institut de Mathématiques de Bordeaux & INRIA Bordeaux–Sud-Ouest, Team-project Bacchus, 200 route de la Vieille Tour, 33405 Talence, FRANCE

RESEARCH CENTRE
BORDEAUX – SUD-OUEST

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

Une méthode de frontiéres immergé par pénalisation employant une technique d'adaptation de maillage couplée avec une reprśentation par courbes de niveau des frontières.

Résumé : L'intÃ¨rÃªt, en CFD, pour les mÃ©thodes employant des frontiÃ¨res immergÃ©es, va croissant car elles simplifient les procÃ¨dures de gÃ©nÃ©ration de maillage, en particulier dans le cas des Ã©quations de Navier Stokes. On peut faire le mÃªme constat dans le cas des problÃ¨mes multi-physique, comme, par exemple, pour le couplage fluide-structure, quand il y a de trÃ¨s grandes dÃ©formations du maillage.

NÃ©anmoins, un traitement prÃ©cis des conditions de parois reste un problÃ¨me difficile. Dans ce travail, les conditions d eparoi sont prises en compte par une mÃ©thode de pÃ©nalisation, et la prÃ©cision est obtenue grÃ¢ce Ã une mÃ©thode d'adaptation de maillage. En cela, on utilise le potentiel des mailalges non strcturÃ©s. Plusieurs exemples classiques de simulations sont proposÃ©es pour valider l'approche.

Mots-clés : Technique de pénalisation, mailalges non structurés, level set, maillages anisotropes, adaptaion de maillage, méthode embeded, Équations de Navier Stokes.

Contents

When dealing with CFD simulations two types of grids are most commonly used: body-fitted grids and embedded grids. In the case of body-fitted grids, the external mesh faces match up with the body surfaces and external boundary faces of the domain. This is different for embedded approach also known as fictitious domain, immersed boundary method (IBM) or Cartesian method. Indeed when considering embedded techniques, the bodies are immersed inside a large mesh, most of the time a Cartesian mesh, and special treatments of the elements close to the body surfaces are performed. When considering general cases of moving or deforming surfaces along with topological changes, both approaches have complementary strengths and weaknesses.

When dealing with moving bodies using body-fitted grids, the partial differential equations (PDEs) describing the flow have to be cast in an Arbitrary Lagrangian-Eulerian frame of reference (ALE) , see e.g. [**?**, **?**, **?**, **?**, **?**]. The idea is to move the mesh in such a way as to minimize its distortion and if required the mesh can be regenerated and/or adapted and the solution interpolated [**?**] or not such as in [**?**] where a tricky space-time formalism is used. Each of these steps (ALE, mesh movement, interpolation) have been optimized but the topology reconstruction may fail for singular surface points and the interpolation required between grids may lead to some loss of information.

On the contrary, embedded grids methods are attractive: the PDE formulation remains in an Eulerian frame of reference even when moving bodies are considered, this formulation simplify *a priori* the meshing issue. Most developments made using embedded-grids are performed using structured grids [**?**, **?**, **?**, **?**], though some work using unstructured grids using fixed embedded grids have already been proposed to solve fluid/structure interactions by Wang *et al.* [**?**], or in Löhner *et al.* [**?**] for Computational Structural Dynamics (CSD), with special boundary treatments in order to prevent penetration during contact. In the last decades, different embedded approaches (fictitious domain, IBM, penalization, etc...) have been developed such that flows around complex geometries can be computed [**?**]. Immersed boundary methods can now deal with incompressible flows [**?**] or compressible viscous flows [**?**] and even turbulent flows using mesh stretching and wall law turbulence formulation [**?**]. However, the drawback of embedded boundary methods remains the complicated treatment of wall boundary conditions in general. Recent developments in embedded boundary methods have focused on that point with algorithms for interface treatment to improve high order accuracy [**?**, **?**, **?**, **?**] and/or ghost-cell technique [**?**].

In this work, we introduce the Immersed Boundary Method with Level Sets and Refined Unstructred Meshes (IMB-LS-RM) method. The idea is to combine the strength of mesh adaptation, that is to provide an accurate flow description especially when dealing with wall boundary conditions, to the simplicity of embedded grids techniques, that is to simplify the meshing issue and the wall boundary treatment when combined with penalization term to enforce boundary conditions. The bodies are described using the level-set method [**?**] and are embedded in an unstructured grid. The wall boundary conditions are enforced by a penalization term [**?**]. Once a first numerical solution is computed, mesh adaptation [**?**, **?**] using quality criteria on the level-set and the solution is re-evaluated. In our opinion, this advantage of this strategy is the following: though simple to implement, the IBM techniques suffer a lack of accuracy near the walls. There, the method is consistant but not very accurate. This is why mesh adaptation is employed, in order to remedy to this behavior by a reduced mesh size near the boundaries of interest. Since most of the IMB schemes use Cartesian meshes, this leads to AMR-like grids, see for example [**?**], where the mesh is no longer conformal. Here we have chosen a different strategy. In order to have simple data structure, we have foccussed on a numerical method that uses conformal meshes. Our particular choice is not fundemental. In order to adapt the mesh easily, we foccus on simplicial meshes using triangles and tets. The localisation of the solid bodies is done via a level set method, as in [**?**], but with mesh adaptation in order to improve both the quality of the surface representation and of the solution. The quality of the adapted mesh is controled automaticaly. Of course, the challenge is to control the increase of refined elements, so that the CPU cost of the simulation remains of the same order as the cost of a simulation with a similar mesh, but with a body fitted geometry. Up to our knowledge, none of the existing embedded method combine these three features together.

This paper is organized as follows: we first describe the system we use, including the penalization terms, and we explain the numerical strategy for one given mesh. We discuss the structure of the penalisation. Section 3 describes our anisotropic mesh adaptation strategy and how it is coupled to the system of interest. The IMB-LS-RM method is tested against several classical problems, the results and the discussion are reported in section 4. We discuss the choice of the penalisation parameter and the

cost associated to the method in term of memory requirements. We also discuss qualitatively the quality of the solution, and show that the accuracy of the IMB-LS-RM solver is similar to the same CFD solver where the boundary conditions are weakly imposed, as in [**?**]. Some conclusions and perspectives for future work follow.

2 Penalization

2.1 Penalization method for compressible flows

We consider in this work a laminar flow described by the compressible Navier-Stokes equations. The conservative form of the Navier-Stokes equations can be written as follows (Gravity effects are assumed negligible):

$$\begin{array}{lll} \dfrac{\partial \rho}{\partial t} & +\dfrac{\partial}{\partial \boldsymbol{x}} \cdot (\rho \boldsymbol{u}) & = 0 \\ \dfrac{\partial}{\partial t}(\rho \boldsymbol{u}) & +\dfrac{\partial}{\partial \boldsymbol{x}} \cdot (\rho \boldsymbol{u} \otimes \boldsymbol{u}) + \dfrac{\partial p}{\partial \boldsymbol{x}} & = \dfrac{\partial}{\partial \boldsymbol{x}} \cdot \underline{\boldsymbol{\pi}} \\ \dfrac{\partial}{\partial t}(\rho e) & +\dfrac{\partial}{\partial \boldsymbol{x}} \cdot ((\rho e + p)\, \boldsymbol{u}) & = \dfrac{\partial}{\partial \boldsymbol{x}} \cdot (\underline{\boldsymbol{\pi}} \boldsymbol{u} + \boldsymbol{q}) \end{array} \tag{1a}$$

where ρ is the density, $\boldsymbol{u}$ the velocity, e the specific total energy. The pressure p and the heat flux $\boldsymbol{q}$ are respectively defined by the perfect gas equation of state and the Fourier law:

$$p = (\gamma - 1)\rho T \quad \text{and} \quad \boldsymbol{q} = -\frac{c_p(\mu)}{Pr} \frac{\partial T}{\partial \boldsymbol{x}} \quad \text{with} \quad T = e - \frac{1}{2}|\boldsymbol{u}|^2 .$$

Pr is the laminar Prandtl number, c_p is the heat capacity at constant pressure, T is the temperature and μ is the laminar viscosity. For Newtonian compressible fluids the stress tensor is given by

$$\underline{\boldsymbol{\pi}} = \mu \left(\left[\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{x}} \right] + \left[\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{x}} \right]^T - \frac{2}{3} \left[\frac{\partial}{\partial \boldsymbol{x}} \cdot \boldsymbol{u} \right] \underline{\mathbf{Id}} \right).$$

A penalization method [**?**], is used to enforce the no-slip boundary condition inside the solid wall boundaries. The solids around which the flow is computed are defined using the so-called penalization method or Brinckman-Navier-Stokes equations. Here, the solids are considered as porous media with a very small intrinsic permeability. One level set function, Φ, is used to capture interfaces and compute rigid motions of the solid bodies, [**?**]. Given a computational domain Ω, we consider a compressible flow in Ω_f around rigid solids S^i. A schematic representation of a computational domain composed of six solids is sketched on Figure (**??**).

The fluid-solid interaction problem can be modeled by the compressible Navier-Stokes equations (**??**) along with the following conditions:

$$\begin{array}{ll} \boldsymbol{u} = \boldsymbol{u}_{s^i} & \text{on } \partial S^i, \\ \boldsymbol{u} = \boldsymbol{u}_f & \text{on } \partial \Omega_f. \end{array} \tag{1b}$$

and Robin-like conditions on the temperature

$$\begin{array}{ll} a_i T + b_i \dfrac{\partial T}{\partial x} \cdot \mathbf{n} = g_i & \text{on } \partial S^i, \\ a_f T + b_f \dfrac{\partial T}{\partial x} \cdot \mathbf{n} = g_i s & \text{on } \partial \Omega^f \end{array} \tag{1c}$$

with suitable values of a_i, b_i a_s and b_f so that Dirichlet, Neuman and genuine Robin condition can be handled in the same formalism. Depending on the location in the computational domain, the velocity is either the fluid velocity $\boldsymbol{u}_f$, either the solid velocity $\boldsymbol{u}_{s^i}$. The idea is to extend the velocity field inside the solid body and to solve the flow equations with a penalization term to enforce rigid motion inside the solid [**?**].

Given a penalization parameter $\frac{1}{\eta} \gg 1$, and denoting by χ_{s^i} the characteristic function of the solid S^i, the model equation writes

$$\begin{array}{lll} \frac{\partial \rho}{\partial t} + \frac{\partial}{\partial \boldsymbol{x}} \cdot (\rho \boldsymbol{u}) & = & 0 \\ \frac{\partial}{\partial t}(\rho \boldsymbol{u}) + \frac{\partial}{\partial \boldsymbol{x}} \cdot (\rho \boldsymbol{u} \otimes \boldsymbol{u}) + \frac{\partial p}{\partial \boldsymbol{x}} + \frac{1}{\eta} \sum_{i=1}^{Ns} \chi_{s^i} (\rho \boldsymbol{u} - \rho \boldsymbol{u}_{s^i}) & = & \frac{\partial}{\partial \boldsymbol{x}} \cdot \underline{\boldsymbol{\pi}} \\ \frac{\partial}{\partial t}(\rho e) + \frac{\partial}{\partial \boldsymbol{x}} \cdot ((\rho e + p)\boldsymbol{u}) + \frac{1}{\eta} \sum_{i=1}^{Ns} \chi_{s^i} (\rho e - \rho e_{s^i}) + \frac{1}{\eta} \sum_{i=1}^{Ns} \chi_{s^i} (\rho \boldsymbol{u} - \rho \boldsymbol{u}_{s^i}) \cdot \boldsymbol{u} & = & \frac{\partial}{\partial \boldsymbol{x}} \cdot (\underline{\boldsymbol{\pi}} \boldsymbol{u} + \boldsymbol{q}) \end{array} \tag{2}$$

est ce que le systeme penalise est invariant par transformation galileenne. We recall that $\rho e = \rho c_v T + \frac{1}{2}\rho \boldsymbol{u}^2$, hence if we want to impose a Dirichlet boundary condition on the temperature we extend this temperature inside the solid body and the penalization term for the energy equation becomes

$$\frac{1}{\eta} \sum_{i=1}^{Ns} \chi_{s^i} \left(\rho e - \rho c_v T_{s^i} - \frac{1}{2} \rho \boldsymbol{u}_s^2 \right). \tag{3}$$

Note that, since the penalization parameter is large, the actual value of the temperature inside the solid plays a role only in the viscidity of the solid surface. This gives the possibility to handle non uniform Dirichlet condition. For a Neumann boundary condition on the temperature (adiabatic wall), we do not impose any additional constraint , i.e. we consider (**??**) with $T_s = 0$.???????? The discretization and the integration of the penalization term affect the choice of the penalization parameter $\frac{1}{\eta}$: the larger the parameter, the better the quality of the penalization. In this work χ_{s^i} is computed from a level set function Φ_{s^i}. We initialize Φ_{s^i} as the signed distance function to the boundary of S^i, because Φ_{s^i} is positive outside S^i and negative inside the solid:

$$\chi_{s^i} = H(-\Phi_{s^i}), \tag{4}$$

where H is the Heaviside function.

Doit-on garder de ici Note that if we are considering moving bodies Φ_{s^i} has to satisfy the same advection equation as χ_{s^i}:

$$\frac{\partial \Phi_{s^i}}{\partial t} + (\mathbf{u}_{s^i} \cdot \nabla)\Phi_{s^i} = 0 \qquad \text{for } \mathbf{x} \in \Omega\,. \tag{5}$$

In case, it is important to notice that, since $\boldsymbol{u}_{s^i}$ is a rigid body motion, one can guarantee that Φ_{s^i} remains a signed distance for all time. Que veut-on dire ? To summarize, each body-fluid interface is captured by a level set function. a la, vu qu'on fait pas de moving solids?

The global model considered in this context can be written in the following compact form

$$\partial_t \boldsymbol{U} + \partial_{\boldsymbol{x}} \cdot \underline{\boldsymbol{F}} = \partial_{\boldsymbol{x}} \cdot \underline{\boldsymbol{G}} + \boldsymbol{S} \tag{6}$$

where the vector $\boldsymbol{U}$ of conservative variables, advection $\underline{\boldsymbol{F}}$ and viscous $\underline{\boldsymbol{G}}$ flux tensors are defined as:

$$\boldsymbol{U} = \begin{pmatrix} \rho \\ \rho \boldsymbol{u} \\ \rho e \end{pmatrix}, \quad \underline{\boldsymbol{F}} = \begin{pmatrix} \rho \boldsymbol{u} \\ \rho \boldsymbol{u} \otimes \boldsymbol{u} + p \underline{\mathbf{Id}} \\ (\rho e + p)\boldsymbol{u} \end{pmatrix} \quad \text{and} \quad \underline{\boldsymbol{G}} = \begin{pmatrix} 0 \\ \underline{\boldsymbol{\pi}} \\ \underline{\boldsymbol{\pi}} \boldsymbol{u} + \boldsymbol{q} \end{pmatrix}$$

Finally, the penalization term holds in the vector $\boldsymbol{S}$:

$$\boldsymbol{S} = \frac{1}{\eta} \sum_{i=1}^{Ns} \chi_{s^i} \begin{pmatrix} 0 \\ \rho \boldsymbol{u} - \rho \boldsymbol{u}_{s^i} \\ \rho e - \rho e_{s^i} + \rho \big(\boldsymbol{u} - \boldsymbol{u}_s\big) \cdot \boldsymbol{u} \end{pmatrix}$$

The convective part of this system (when right hand size is set to zero) is hyperbolic. Indeed, for any direction $\boldsymbol{n}$ we define the matrix $\underline{\boldsymbol{A}}(\boldsymbol{n})$ as

$$\underline{\boldsymbol{A}}(\boldsymbol{n}) = \frac{\partial \underline{\boldsymbol{F}} \boldsymbol{n}}{\partial \boldsymbol{U}} \quad \text{with} \quad \underline{\boldsymbol{F}} \boldsymbol{n} = \begin{pmatrix} \rho \boldsymbol{u} \cdot \boldsymbol{n} \\ \rho \boldsymbol{u} (\boldsymbol{u} \cdot \boldsymbol{n}) + p \boldsymbol{n} \\ (\rho e + p) \boldsymbol{u} \cdot \boldsymbol{n} \end{pmatrix}$$

For any $\boldsymbol{n}$ with $|\boldsymbol{n}| \neq 0$, the matrix $\underline{\boldsymbol{A}}(\boldsymbol{n})$ is diagonalisable and have three different eigenvalues

$$\lambda_- = \boldsymbol{u}\cdot\boldsymbol{n} - a\|\boldsymbol{n}\|, \qquad \lambda_0 = \boldsymbol{u}\cdot\boldsymbol{n} \quad \text{and} \quad \lambda_+ = \boldsymbol{u}\cdot\boldsymbol{n} + a\|\boldsymbol{n}\|,$$

where $a = \sqrt{\dfrac{\gamma p}{\rho}}$ is the sound speed. We denote by $\underline{\boldsymbol{P}}(\boldsymbol{n})$ and $\underline{\boldsymbol{\Lambda}}(\boldsymbol{n})$ respectively the matrices of eigenvectors and eigenvalues of $\underline{\boldsymbol{A}}(\boldsymbol{n})$:

$$\underline{\boldsymbol{A}}(\boldsymbol{n}) = [\underline{\boldsymbol{P}}(\boldsymbol{n})]\,\underline{\boldsymbol{\Lambda}}(\boldsymbol{n})\,[\underline{\boldsymbol{P}}(\boldsymbol{n})]^{-1}$$

this spectral decomposition is useful for upwinding, streamlines stabilization and boundary conditions and will be used inside our discretization.

2.2 Discretization

Numerical applications are performed for simplexes meshes (2D-triangles and 3D-tetrahedrons) with piecewise linear Lagrange test functions. The obstacles are localized using the zero isovalue of a static level-set function. The system to be solved writes: find $\mathbf{U}_h$ such that

$$\int_\Omega \mathbf{W}_h \mathcal{R}(\mathbf{U}_h) d\Omega = 0 \tag{7}$$

where $\mathcal{R}(\mathbf{U}_h)$ is the residual of equation (**??**) that is

$$\mathcal{R}(\mathbf{U}_h) = \partial_t \boldsymbol{U}_h + \partial_{\boldsymbol{x}}\cdot\underline{\boldsymbol{F}} - \partial_{\boldsymbol{x}}\cdot\underline{\boldsymbol{G}} - \boldsymbol{S} := \partial_t \boldsymbol{U}_h + \Psi(\mathbf{U}_h) \tag{8}$$

The time domain is subdivided into non overlapping intervals $]t^n, t^{n+1}[$, $\Delta t_{n+1/2} = t_{n+1} - t_n$. The numerical solution at a time t^n is denoted by $\mathbf{U}_h^n$. We are given an initial condition $\mathbf{U}_h^0$ we assume that $\mathbf{U}_h^{-1} \equiv \mathbf{U}_h^0$ and $t^{-1} = t^0$. Therefore the system to be solved at each time step is to find $\left(\mathbf{U}_h^{n+1}\right)_{n\geq 0}$ such that

$$\begin{aligned}\int_\Omega \mathbf{W}_h \frac{\mathbf{U}_h^{n+1} - \mathbf{U}_h^n}{\Delta t_{n+1/2}} d\Omega + \int_\Omega \mathbf{W}_h \Psi\left(\mathbf{U}_h^{n+\theta}\right) d\Omega \\ + \int_{\partial\Omega} \mathbf{Bc}\left(\mathbf{W}_h, \mathbf{U}_h^{n+\theta}\right) = 0\end{aligned} \tag{9}$$

where $\mathbf{U}_h^{n+\theta} = \mathbf{U}_h^n + \theta\left(\mathbf{U}_h^{n+1} - \mathbf{U}_h^n\right)$, $\theta \geq 0$ and the operator $\mathbf{Bc}$ takes into account all contributions for boundary conditions.

We briefly describe our numerical strategy:

1. The system of equations is discretized using a classical mixed Finite Element/Finite Volume scheme. Finite Volumes are used for the convective part of the system, see [**?**], which is a generalization of the MUSCL method on simplicial meshes. Gradient extrapolation is done on the primitive variables, and the van Leer-van Albada limiter is used. The numerical fluxes are Roe's or HLLC's, and Steger Warming's for boundary conditions at the inlet and outlet of the domain. The diffusive part of the system and source terms are discretized using $\mathbb{P}^1$ Finite Elements on the triangulation with the Galerkin approximation. Note there is no need here to use no slip boundary conditions. The approximation of the penalization terms is related to the time approximation for stability reasons.

2. The time derivative is approximated as follows. For time dependent problems, the most accurate second order scheme is in general obtained with the semi-implicit scheme associated to $\theta = \frac{1}{2}$ (Crank-Nicholson scheme). The explicit scheme associated to $\theta = 0$ is subject to CFL stability condition and do not have proper behavior in this context of penalization, because the CFL will be directly linked to the choice of $\dfrac{1}{\eta}$: an Euler explicit time discretization does not allow to use $\dfrac{1}{\eta} > 1/\Delta t$. To obtain accuracy using penalization technique, we need to take $\dfrac{1}{\eta} \gg 1$. Hence, in practice, we always use implicit schemes with $\dfrac{1}{\eta} = 1 \times 10^{12}$.

3. Therefore, we have to solve, at each time step, a nonlinear system. This resolution is achieved by Newton-type relaxations, each relaxation step is defined by a linear sparse system of large size, obtained by a linearized implicit approximation:

$$\begin{aligned}\Psi\left(\mathbf{U}_h^{n+\theta}\right) &\simeq \Psi\left(\mathbf{U}_h^n\right) + \theta\left(\frac{\partial\Psi}{\partial\mathbf{U}_h}\right)^n\left(\mathbf{U}_h^{n+1}-\mathbf{U}_h^n\right)\\ \mathbf{Bc}\left(\mathbf{W}_h,\mathbf{U}_h^{n+\theta}\right) &\simeq \mathbf{Bc}\left(\mathbf{W}_h,\mathbf{U}_h^n\right) + \theta\left(\frac{\partial\mathbf{Bc}}{\partial\mathbf{U}_h}\right)^n\left(\mathbf{U}_h^{n+1}-\mathbf{U}_h^n\right)\end{aligned}$$

The approximated Jacobians are again obtained by the same method as in [**?**].

This numerical strategy has been deployed in the computing platform "Realfluids" that provides tools for high order time integration. Parallelism is achieved by domain decomposition and message passing using MPI. The domain decomposition is performed via Metis [**?**] or the SCOTCH Library [**?**]. See [**?**] for details. The linear systems are solved by standard techniques (preconditioned GMRES). The preconditioning methods are either of the ILU(0) type or by using locally or globally the direct solvers contained in the PASTIX library [**?**, **?**]. Our computations have been performed up to 64 processors on the cluster PlaFRIM from INRIA Bordeaux Sud-Ouest. However, parallel efficiency of Realfluids platform has been demonstrated up to 512 processors.

3 Anisotropic mesh adaptation.

3.1 Mesh adaptation background.

In this section, we briefly explain the theoretical framework for anisotropic meshing and describe the important features of the mesh adaptation algorithm. For more details, we refer to [**?**, **?**, **?**, **?**, **?**, **?**, **?**, **?**, **?**, **?**]. In particular it is shown in these references that the local error can be estimated on each element K by

$$||u-\Pi_h(u)||_\infty \leq c \max_{y\in K}\max_{x\in K}\left(x^T|H_u(y)|x\right)$$

where $\Pi_h(u)$ is the Lagrange interpolant, the constant c only depends on the type of element, and H_u is the Hessian of u. The Hessian is estimated from discrete data by a technique detailed in the above reference, this metric is denoted by M and is local.

The aim of anisotropic mesh adaptation is to control the size, the shape and the orientation of mesh elements. These specifications are usually based on an error estimate and they are written via a metric tensor.

3.1.1 Metric tensors.

This metric tensor $M(x)$ is represented by a $d\times d$ $(d=2,3)$ symmetric definite positive matrix. The eigenvalues (resp. eigenvectors) are related to the desired sizes (resp. directions) of the element edges. This metric tensor is used to generate a quasi-uniform mesh of the domain in this metric. This means we would have unitary volume of an element K in the discretized domain T_h:

$$\int_K \sqrt{\det(M(x))}\,dx = 1 \;\; \forall K\in T_h.$$

The length of the vector $\overrightarrow{e}$ with respect to the metric $\mathcal{M}(x)$ is defined as [**?**]:

$$l_{\mathcal{M}}(e) = \int_0^1 \sqrt{e^T\mathcal{M}(t)e}\,dt. \tag{10}$$

3.1.2 Metric interpolation.

In practice, the metric is often supplied at mesh vertices and we need to define a metric at each point on a mesh edge. To this end, we define a continuous metric field along the mesh edge by using a linear interpolation scheme.

If P and Q are the endpoints of an edge parametrized by

$$e(t) = (1-t)P + tQ, \quad 0 \leq t \leq 1$$

and M_P (resp. M_Q) the metric associated to P (resp. Q), the interpolation scheme is defined by:

$$M(t) = ((1-t)M_P^{-\frac{1}{2}} + tM_Q^{-\frac{1}{2}})^{-2}\,.$$

3.1.3 Metric intersection.

If several metrics are defined at one vertex $P \in T_h$, we define a single metric by using the intersection metric:

$$\mathcal{M}_\cap = \mathcal{M}_1 \cap \mathcal{M}_2\,.$$

Geometrically speaking, it consists in defining the largest ellipsoid included in the intersection of all the ellipsoids associated to the considered vertex. This is illustrated in the case of two metrics on the Figure **??** for which the black ellipse should be used for the adaptation of the mesh. From a practical point of view, we use simultaneous reduction of the two metric tensors, see [**?**] for more details.

3.2 Level-set adaptation: metric definition

Solid bodies around which flow computations are performed are defined by the isovalue 0 of a level-set function. In this case the level-set function is the signed distance function, positive outside the solid and negative inside. Obtaining a good approximation of the contour of the solid bodies means to have an accurate tracking of their boundaries. Those boundaries are given by the isovalue 0, so the idea is to put enough elements in the vicinity of this isovalue and minimize the piecewise affine approximation of those boundaries. To this end, in [**?**], Frey *et al.* have defined a metric to control the geometric approximation of an isovalue of a level-set function φ. Let's ε be an error, h_{min} (resp. h_{max}) the minimal (resp. max.) length edge. The metric is defined by:

$$M = R \begin{pmatrix} \frac{1}{\varepsilon^2} & 0 & 0 \\ 0 & \frac{|\lambda_1|}{\varepsilon} & 0 \\ 0 & 0 & \frac{|\lambda_2|}{\varepsilon} \end{pmatrix} R^T \tag{11}$$

with $R = (\nabla\varphi \;\; v_1 \;\; v_2)$ where $(v_1 \;\; v_2)$ is a basis of the tangent plane to the boundary and λ_i are the eigenvalues of the Hessian of φ (i.e. the curvature of the isoline). Note that $\nabla\varphi$ is the normal of the isoline. We impose this metric to the vertices in the vicinity of the solid objects. For the other vertices, h_{min} and ε are increased linearly up to h_{max}.

3.3 Mesh adaptation to flow features

In order to have a good accuracy of the physical solution with a minimum number of nodes, we use classical anisotropic mesh adaptation based on interpolation error.

Let u be a variable and $\Pi_h u$ its $\mathbb{P}^1$ interpolant. We have the following estimation of the interpolation error on a mesh element K [**?**]:

$$\| u - \Pi_h u \|_{\infty,K} \quad \leq \quad c \max_{e \in E_K} \langle e, \mathcal{M}(K)\, e \rangle\,,$$

with e a mesh edge and E_K the set of mesh edges. In this estimation, $\mathcal{M}(K)$ is a metric tensor computed with the Hessian of u and defined by:

$$\mathcal{M} = \mathcal{R} \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{pmatrix} \mathcal{R}^{-1} \tag{12}$$

where $\mathcal{R}$ is the matrix of the eigenvectors of the Hessian of u and

$$\lambda_i = \min\left(\max(|h_i|, \frac{1}{h_{max}^2}), \frac{1}{h_{min}^2}\right). \tag{13}$$

Here the h_is are the eigenvalues of the Hessian of u.

We would equidistribuate the interpolation error over all the mesh so if ε is the desired error. Thanks to the previous estimate, we would have:

$$\forall \vec{e} \quad \varepsilon = c\,\langle \vec{e},\, \mathcal{M}(K)\vec{e}\rangle \quad \Rightarrow \quad \forall \vec{e} \quad \langle \vec{e},\, \frac{c}{\varepsilon}\mathcal{M}(K)\ \vec{e}\rangle = 1.$$

This means that we search to have edges with their lengths equal to unity in the metric $\frac{c}{\varepsilon}\mathcal{M}(K)$.

More details on the construction of the two metrics can be found in the appendix .

Remark 1. *For practical applications, it is better to consider the relative error:*

$$\left\| \frac{u - \Pi_h u}{u} \right\|_{\infty,K} \leq c \max_{e \in E_K} \langle e, \frac{\mathcal{M}(K)}{u}\ e\rangle\,. \tag{14}$$

The reason behind this choice is to give an equivalent weight to each criterion when several criteria are needed, as it is the case here. We adapt on a physical criterion and on the isovalue 0 of the level-set. Of course the same adimensionalisation is applied for the criteria discussed in section **??**.

4 Numerical results

We propose in this section four test cases to demonstrate the ability of the method to obtain accurate solutions together with an accurate wall treatment starting with initial meshes that does not contain any point on the level-set 0 of the solid body.

In the first test case, we check the ability of the method to model and predict the laminar boundary layer over a flat plate. The solution is then compared to the Blasius solution. The second test case demonstrates the performance of the method to position correctly a shock wave in the laminar regime. The third test case considers the laminar flow around a NACA 0012 airfoil. In this case, the penalized solution is compared to the body fitted one. The last test case is a 3D laminar flow around an ellipse: we demonstrate that the proposed method is suitable for 3D problems.

4.1 Blasius test case

The velocity distribution $U(x)$ of the outer flow of an incompressible flat plate when dealing with laminar flow is given as a power series in x, where x is the coordinate measured along the flat plate. The velocity distribution in the boundary layer is then also expanded in a power series, called the Blasius series, where the coefficients are still functions of the y coordinate. For this first test case the numerical solution of the Navier-Stokes equations in the laminar regime ($Re = 500$) is compared to the analytical solution derived from Blasius series. To investigate the role of the penalization term and verify the quality of the solution along with mesh adaption, the test case is constructed as a backward facing step test case, see figure **??**, where the step is all penalized.

Of course the original mesh does not have necessarily points located on the flat plate. This mesh has 57509 points and 114024 triangles, a zoom is given on figure **??**. Once the solution is computed mesh adaptation constrained by the distance to the level-set and the Hessian of the solution is performed. The criteria on the solution is performed on the component U of the velocity. The adaptation parameters for U are

$$\varepsilon = 10^{-3};\ h_{min} = 10^{-3};\ h_{max} = 2$$

and for the adaptation on the level-set are

$$\varepsilon = 10^{-3};\ h_{min} = 10^{-3};\ h_{max} = 0.1.$$

The adapted mesh has 4906 points and 9610 triangles, see figure **??**. Two-dimensional cuts of the velocity distribution U/U_{inf} as a function of the non dimensional vertical coordinate Y/Y_{inf} are performed at

the end of the flat plate, location $x = 1$. The solution obtained on the original mesh and on the adapted mesh are compared to Blasius analytical solution, see figure **??** and **??**. Using penalization technique together with mesh adaptation we obtain a solution in very good agreement with the theoretical one and similar to a solution obtained using body fitted mesh. Mesh adaptation allows us to improve the quality of the solution at the interface and reduce the number of points needed for the test case, indeed the points are mainly located on the focus zone of interest of the solution.

Note that the size of the step has no influence on the results. We have chosen the worst case scenario taking a large step paying no attention at all to reduce the number of nodes inside the penalized region. The method does not add more than 30% nodes. We have performed several other cases, with different step size, and the same conclusion hold: the number of points on the final adapted mesh is similar to the one of the body fitted mesh, for the same quality of results.

4.2 Supersonic flow around a triangle

We consider the supersonic flow around a solid body of triangular shape with height $h = 0.5$, half angle $\theta = 20$ deg and $S = (0.5\,,\,1)$, see figure **??**. The flow is computed using the penalized Navier-Stokes equation (**??**). The same computational domain as [**?**] (for their computation using FLUENT) is chosen even if a smaller domain at the inlet could have been taken because we are dealing with supersonic flow. The computation is stopped when a steady state is obtained. The Reynolds number is fixed to $Re = 5 \times 10^4$, the Prandtl number to 0.72 and the Mach number to $M_1 = 2$. As in Boiron *and al.* [**?**] the velocity inside the triangle is set to zero and the nondimensional temperature is set to $T_s = 3$.

An oblique shock is predicted by the inviscid flow theory. It can be attached or detached depending on values of the angle θ and the Mach number. If the shock is attached to the triangle, its angle β with the horizontal is given by:

$$\tan\theta = 2\cot\beta \left[\frac{M_1^2 \sin^2\beta - 1}{M_1^2\,(\gamma + \cos 2\beta) + 2} \right]. \tag{15}$$

The initial non-adapted mesh contains 28141 nodes and 56189 triangles, the mesh and the density distribution are presented in figure **??**. The final adapted mesh, after 5 cycles of adaptations, contains 791705 nodes and 1 583 221 triangles. The parameters for the adaptations are the following

$$\text{on the velocity} \qquad \varepsilon = 10^{-4};\ h_{min} = 10^{-4};\ h_{max} = 2$$

$$\text{on the level set} \qquad \varepsilon = 10^{-4};\ h_{min} = 10^{-4};\ h_{max} = 2$$

The component u of the velocity and the adapted mesh are shown in figures **??** and **??**.

According to (**??**), the angle of the shock for this test case is such that

$$\tan\left(\frac{\pi}{9}\right) = 2\cot(\beta)\frac{4.\sin^2(\beta) - 1}{4(\gamma + \cos(2\beta)) + 2} \quad \Rightarrow \beta = 53.46 \text{ deg} \tag{16}$$

In our solution we measure the angle β at the same location as Boiron *and al.* [**?**] , i.e. $y = 0$. We find an angle of about 53.8 deg compared to 53.7 deg found by [**?**]. In [**?**] the computation is performed on a Cartesian grid: 1024×1024, and the domain is $[0, 2] \times [0, 2]$. A 1D cut of the pressure along the line $y = 1.44$ is compared to Boiron *et al* [**?**] and presented in figure **??**. Our results are in good agreement with the theory and the numerical solutions performed in [**?**]. In figure **??**, we notice that our solution is able to compute a sharper shock wave with less diffusion and with less points than Boiron *and al.* Note also that we are computing the solution on a complete domain avoiding blocking effects.

4.3 Flow around a NACA0012 airfoil

The third test case proposed consists on a laminar subsonic flow around a symmetrical NACA 0012 airfoil. The Reynolds number of the solution is set to $Re = 5000$, the Mach number is $M = 0.5$ and there is no angle of attack. To validate our approach we have chosen to perform this numerical simulation using first a body fitted mesh along with mesh adaption on u-velocity. Then our penalized approach is performed on an embedded grid along with mesh adaptation performed on both u-velocity and level-set. The body fitted and embedded initial grids are presented in figure **??**, the body fitted grid contains 44829 vertices and 87823 triangles, the embedded grid contains 49565 vertices and 99062 triangles. As

usual the embedded grid does not contain necessarily points on the level set 0. After 3 cycles of mesh adaptation, the two solutions are compared in figure **??** where for each sub-figure the body fitted solution is located on top and penalized solution on bottom. The adaptation parameters on the u-velocity field for both approaches are taken the same, that is

$$\varepsilon = 5.10^{-4};\ h_{min} = 10^{-4};\ h_{max} = 2$$

and for the embedded grid we use for the level-set adaptation criteria the following parameters

$$\varepsilon = 10^{-4};\ h_{min} = 5.10^{-4};\ h_{max} = 2.$$

The adapted meshes contain respectively 84880 vertices and 165948 triangles for the body fitted grid and 101478 vertices with 202885 triangles for the penalized grid, meaning that the penalized case adds less than 20% of nodes. The u-velocity component drawn in figure **??**, with a zoom at leading edge (figure **??**) and at trailing edge figure **??**) shows a perfect agreement of both approaches, validating the overall proposed technique. Special care has to be given to the trailing edge of the NACA 0012 airfoil because this is a singularity for the level-set, indeed the angle is very narrow, the level-set distance should be verify before performing the penalization technique to avoid convexity default.

4.4 3D test case flow around an ellipse

The method described on this paper can easily be applied on 3D test cases. For this test case we use the mesh generator MMG3D [**?**, **?**]. We show in this section, a preliminary test case in three dimensions, a laminar flow around an ellipse. The ellipse is centered at $(0, 0, 0)$, the axes are aligned with x, y, z and the radius are respectively $(0.5, 0.1, 0.2)$. The complete domain is a sphere of radius 15. The initial mesh is adapted on the isovalue 0 of the level-set function. To obtain the initial mesh we impose

$$\varepsilon = 10^{-4};\ h_{min} = 5.10^{-3};\ h_{max} = 2\ .$$

The initial mesh contains 1 051 541 tetrahedrons and 178 909 vertices, see figure **??**. The flow is then computed using the penalized Navier-Stokes equations (**??**). The Reynolds number is fixed to $Re = 500$, the Prandlt number is set to 0.72 and the Mach number to $M = 0.375$. Once the first solution is computed, we adapt the mesh on two criteria, the x-component of the velocity and the isovalue 0 of the level-set. The adaptation parameters for the velocity are

$$\varepsilon = 10^{-3};\ \ h_{min} = 5.10^{-3};\ \ h_{max} = 2$$

and we keep the parameters of the initial mesh for the level-set adaptation. The adapted mesh is shown on Figure **??**. It contains 1 117 932 tetrahedrons and 187 977 vertices. Figures **??** and **??** show respectively the distribution of x and y-components of the velocity.

5 Conclusion

In this work, we propose to combine penalization techniques to mesh adaption. The idea is to conserve the simplicity of the embedded approaches for grid generation process and overcome the difficulty of wall treatments by using mesh adaptation. Mesh adaptions are performed using two criteria, the distance to the level-set 0 and one component of the flow solution (ρ, u, v, etc...). In our case u has been chosen. Other parameters could have been chosen. The four test cases proposed demonstrate the ability of the proposed method to obtain an accurate solution along with an accurate wall treatment even when the initial mesh does not contain any point on the level-set 0. The obtained numerical solutions are compared to analytical solution for Blasius test case, to other numerical solutions for supersonic flow around a triangle, to body fitted adapted solution for a laminar NACA 0012 airfoil, and the technique is performed on a 3D test case to demonstrate the feasibility of the method even in the case of a more demanding problem. The important conclusion is that, for a given accuracy, the number of points added in the boundary layer by this mesh adaptation technique with level set, though larger to what can be done for body fitted meshes, is still of the same order: the method is about 20% more expensive but completely avoid the construction of meshes having to mach complex geometries. In the near future, we will investigate penalized methods for moving bodies.

Acknowledgments:

R. Abgrall has been funded in part by the EU ERC Advanced grant "ADDECCO" #226616.

A Practical issues for the metric construction

In practice we compute the two metrics on each node of the mesh. The algorithms used to compute these metrics are given in the following sections, then we proceed to a metric intersection using a simultaneous reduction.

A.1 Flow features metric

We choose the flow scalar variable on which we decide to adapt the grid, for example the density, and we proceed with the following steps:

1. We compute the gradient on each node using a least square approximation;
2. We compute the Hessian with a least square approximation on the gradient: on each node the Hessian is a $d \times d$ matrix where d is the space dimension;
3. We normalize the Hessian by dividing by the maximum value of the chosen scalar through the domain;
4. We define the metric using equations (**??**) and (**??**), i.e. we compute the eigenvalues and eigenvectors of the Hessian.

A.2 Level-set metric

The next step is to compute the level-set metric:

1. On each node we compute the gradient of the level-set function using a least square approximation;
2. We compute the Hessian (using least square);
3. We check if we are close to the level-set 0 or not:
   (a) If we are closed to the level-set 0: we compute the curvature of the level-set, using the Hessian, to impose the metric defined by equation (**??**);
   (b) If not we increase the edge length linearly from h_{min} to h_{max} depending on the distance from the level-set 0;
4. We then have our metric for the level-set and we have to intersect it with the previous one.

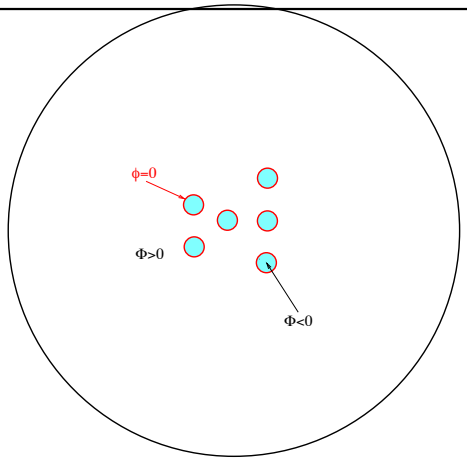


Figure 1: Schematic representation of a computational domain $\Omega = \Omega_f \cup \Omega_s$. The solid domain is $\Omega_s = \cup_{i=1}^{6} S^i$ where the S_is are the individual solid domins. Ω_f is the fluid one. The outer boundary is $\Gamma = \partial\Omega$, The solid boundaries are $\partial\Omega_s = \cup\partial S_i$.

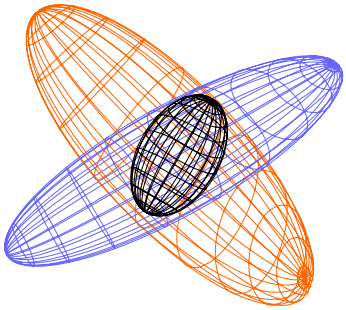


Figure 2: Geometrical representation of metric intersection.

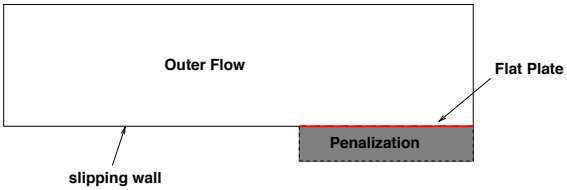
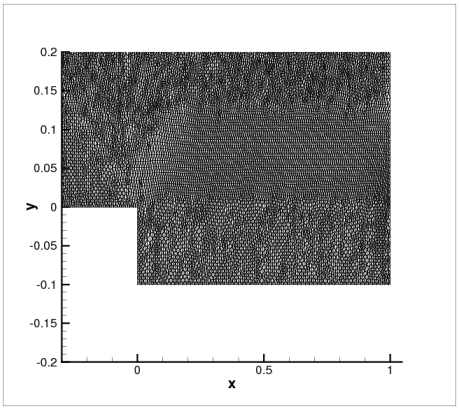
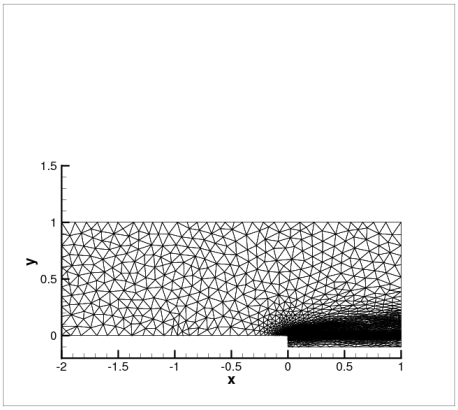


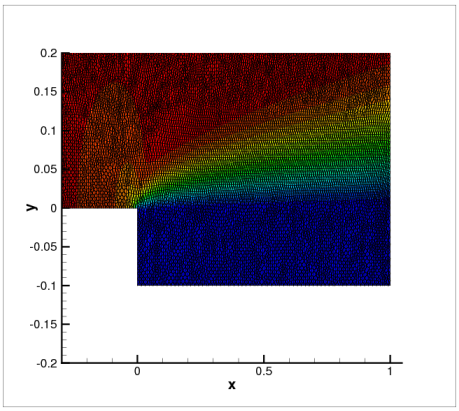
Figure 3: Blasius setup.



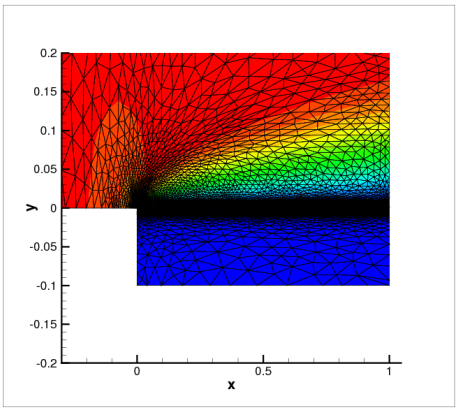
(a) Zoom on the first flat plate mesh for Blasius



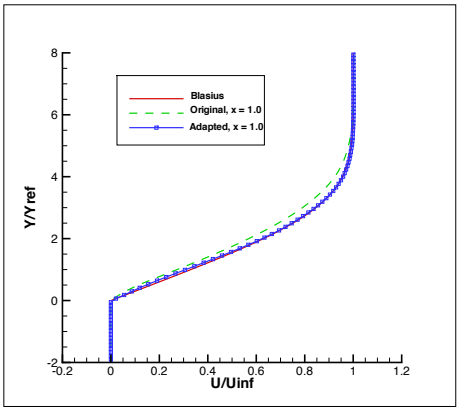
(b) Adapted mesh for the flat plate



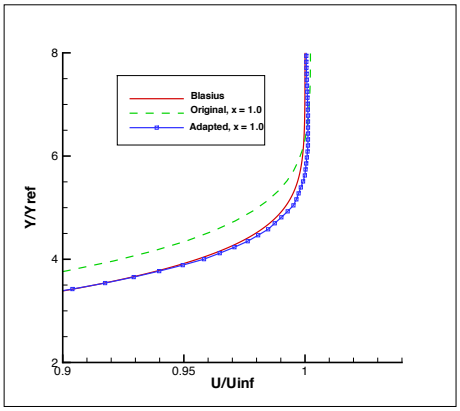
(c) Flat plate velocity distribution u on the original mesh



(d) On the adapted grid

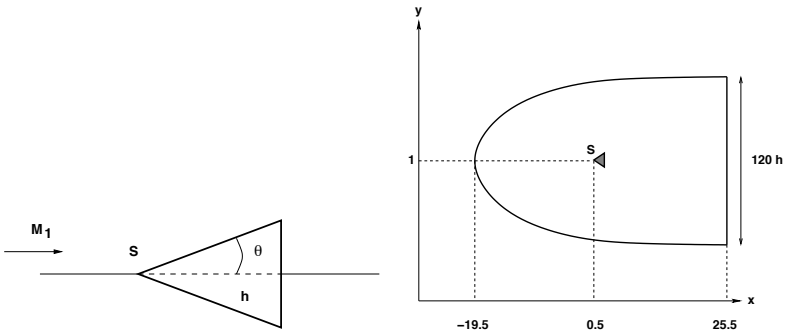


(e) Comparison with the theoretical Blasius solution



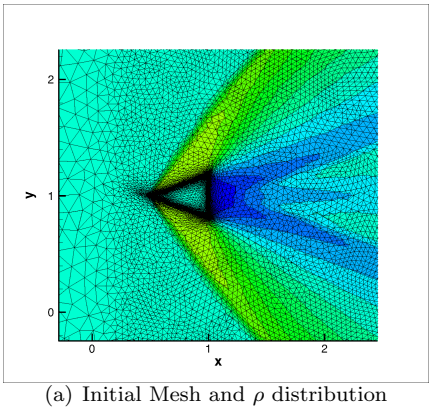
(f) Zoom on the top of the boundary layer

Figure 4: Blasius test case

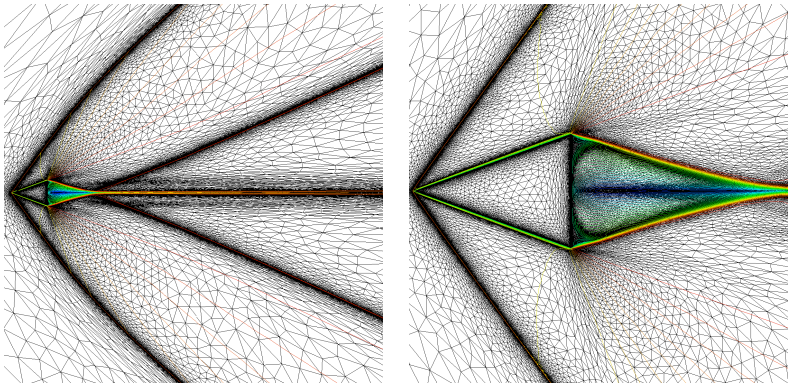


(a) Triangle geometry (b) Triangle domain

Figure 5: Triangle test case setup.



(a) Initial Mesh and ρ distribution



(b) Adapted Mesh along with u velocity (c) zoom on the Adapted Mesh

Figure 6: 2D triangle test case, initial mesh and solution, final mesh and solution

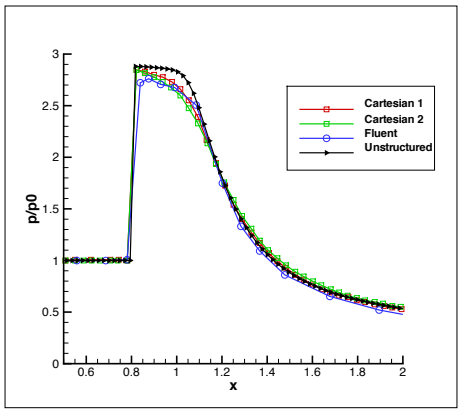


Figure 7: 1D cut of the pressure along the line $y = 1.44$; red squares Boiron Cartesian grid 1024×1024; green squares Boiron Cartesian grid 512×512; blue circles Boiron fluent solution; black triangles our approach.

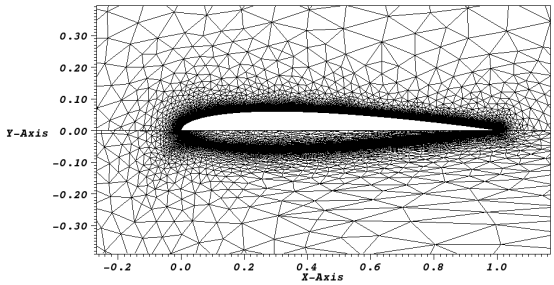
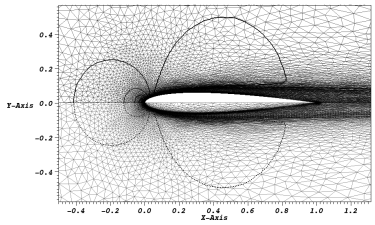
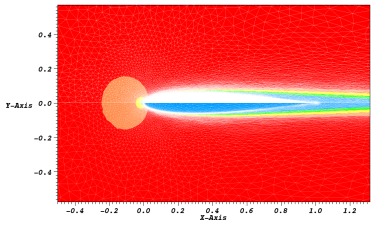


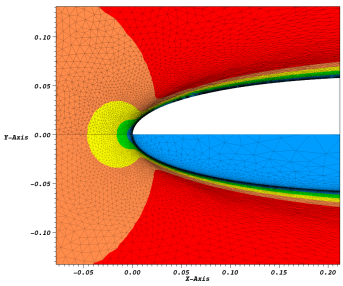
Figure 8: Initial meshes body fitted and embedded



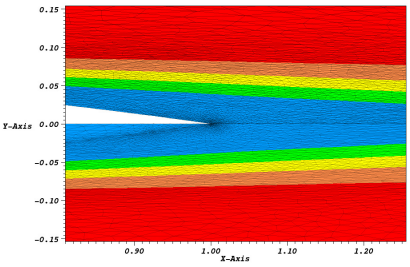
(a) Penalized and non penalized mesh



(b) u-velocity solution

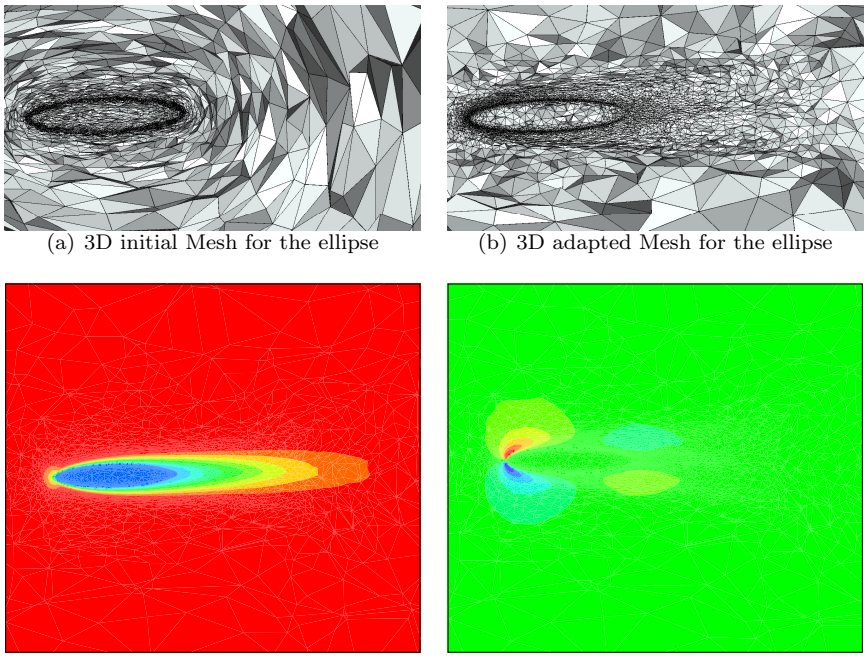


(c) Zoom on the leading edge: mesh and u-velocity



(d) Zoom on the tip

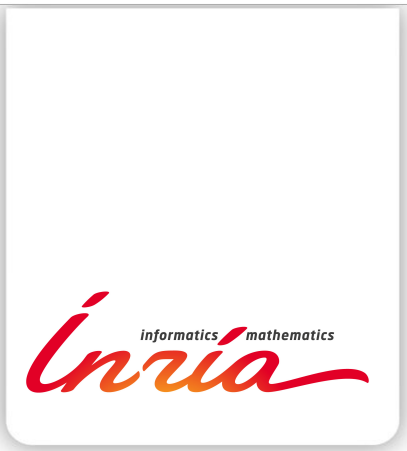
Figure 9: Laminar flow around a NACA0012 airfoil



(a) 3D initial Mesh for the ellipse (b) 3D adapted Mesh for the ellipse

(c) Ellipse: x-component of the velocity (d) Ellipse: y-component of the velocity

Figure 10: Cuts through tetrahedral meshes for the 3D ellipse and 2D cuts of the velocity.



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