Engineering with Computers
https://doi.org/10.1007/s00366-025-02119-x

ORIGINAL ARTICLE

Implementation of immersed boundaries via volume penalization in the industrial aeronautical computational fluid dynamics solver CODA

Jonatan Núñez[1,2] · David Huergo[1,2] · Diego Lodares[1,3] · Suyash Shrestha[1,2] · Juan Guerra[3] · Juan Florenciano[3] · Esteban Ferrer[1,2] · Eusebio Valero[1,2]

Received: 27 July 2024 / Accepted: 10 February 2025

Abstract
We present the implementation and validation of an immersed boundary volume penalization method in the computational fluid dynamics solver CODA (from ONERA, DLR, and Airbus). Our goal is to model and simulate turbulent fluid flows in complex 3D aerodynamic configurations through the numerical solution of the Reynolds-averaged Navier–Stokes equations using the Spalart–Allmaras turbulent model. To do that, an immersed boundary method has been implemented in CODA and an efficient preprocessing tool for the construction of unstructured hexahedral meshes with adaptive mesh refinement around immersed geometries has been developed. We report several numerical examples, including subsonic flow past the NACA0012 airfoil, transonic flow past the RAE2822 airfoil, subsonic flow past the MDA30P30N multi-element airfoil, and subsonic flow around the NASA high-lift CRM aircraft. These simulations have been performed in the CODA solver with a second-order finite volume scheme as spatial discretization and an implicit backward Euler scheme based on the matrix-free GMRES block-Jacobi iterative method. The reported numerical simulations are in good agreement with their corresponding experimental data. These encouraging results allow us to conclude that the implemented immersed boundary method is efficient, flexible, and accurate and can therefore be used for aeronautical applications in industry.

Keywords CODA · Immersed boundaries · Volume penalization · Octree mesh · Computational fluid dynamics

1 Introduction

In the ever-evolving landscape of computational fluid dynamics (CFD) and industrial simulations, the accurate representation of complex geometries and their interaction with fluid flow remains a critical challenge. The ability to model and analyze intricate structures in a computationally efficient manner is paramount, as it directly influences the design, performance, and optimization of various industrial systems. This challenge is particularly relevant in the context of the industrial solver CODA [1], where the interaction between fluid flow and complex geometries plays a central role.

✉ Jonatan Núñez
jonatan.nunez@upm.es

David Huergo
david.huergo.perea@upm.es

Diego Lodares
diego.lodares@airbus.com

Suyash Shrestha
s.shrestha@upm.es

Juan Florenciano
juan.florenciano@airbus.com

Esteban Ferrer
esteban.ferrer@upm.es

Eusebio Valero
eusebio.valero@upm.es

[1] School of Aeronautics, Universidad Politécnica de Madrid, Plaza del Cardenal Cisneros 3, 28040 Madrid, Spain

[2] Center for Computational Simulation, Universidad Politécnica de Madrid, Campus Montegancedo, 28660 Boadilla del Monte, Spain

[3] Flight Physics Department, Airbus Defence and Space S.A.U, Paseo John Lennon, s/n, 28906 Getafe, Spain

Published online: 27 February 2025

CODA is a parallel software framework for multidisciplinary analysis and optimization of aircraft and helicopters based on advanced and accurate numerical methods [2, 3]. The solver is being developed as part of a collaboration between the French National Aerospace Research Center (ONERA), the German Aerospace Center (DLR) and Airbus. CODA provides a robust, scalable and computational efficient integrated design process for aerodynamics and structural analysis. To perform an efficient analysis and optimization of aircraft on state-of-the-art HPC systems, the Navier–Stokes and the Reynolds-averaged Navier–Stokes (RANS) equations are solved for high Reynolds-number flow on unstructured grids with second-order finite-volume and higher-order discontinuous-Galerkin discretizations.

The implementation of immersed boundaries through volume penalization has emerged as a promising technique to address the difficulty to generate body fitted meshes for complex geometries and simulate moving geometries. Immersed boundaries allow for the inclusion of detailed and intricate geometries within the computational domain, even in cases where traditional structured grids might be impractical or prohibitively expensive to generate. In the realm of CFD simulations, the term "immersed boundary" refers to the representation of complex structures, such as solid objects or porous media, as part of the fluid computational domain, without the need for a grid that conforms to their intricate shapes. This not only enhances the capability to model real-world industrial scenarios, but also significantly improves the computational efficiency (while reducing human intervention) of the simulation workflow because the time-consuming body-fitted mesh generation is avoided.

The primary objective of this research is to provide an in-depth exploration of the methodology, mathematical formulations, and computational considerations involved in the implementation of immersed boundaries through volume penalization in the solver CODA. By doing so, this study aims to facilitate a deeper understanding of this technique and its potential applications in aeronautics. It is our hope that the insights and findings presented in this paper will contribute to the advancement of CFD simulations, offering engineers and researchers a valuable tool for the accurate modeling of complex industrial systems and the optimization of their performance.

The immersed boundary method was introduced by Peskin [4] for analyzing the flow through the native mitral heart valve, and since then, the method has been applied to the numerical simulation of flow problems over complex geometries [5], multiphase flows [6], and fluid–structure interaction [7].

The main feature of immersed boundary methods is that the governing equations are solved on a Cartesian background mesh which is not necessarily conforming with the immersed body. Immersed boundary methods can be classified according to the mechanisms used to describe the interaction between the fluid and the immersed geometry. Several approaches are currently available, among them, the more relevant are the cut-cell methods [8–10], the ghost cell methods [11–14], the direct forcing approach [15–18], sharp interface methods [19, 20], and the volume penalization schemes [21–25]. In [26] is discussed the efficiency of the immersed boundary volume penalization for handling moving geometries, like the flow around an airfoil with pitching and plunging motions. More details on different families of immersed boundary methods can be found in the reviews [27–30] and the references cited therein. Recently, immersed boundary methods have been extensively used to simulate turbulent aerodynamic flow problems in the context of RANS simulations [31–38]. This paper is organized as follows: in Sect. 3 we summarize the main numerical schemes implemented in CODA for simulating turbulent fluid flow with immersed boundary methods. In Sect. 4 we discuss the preprocessing and postprocessing tools developed for generating refined meshes around immersed bodies and the analysis of simulation data. Next, in Sect. 5 we present several numerical computations done with CODA with immersed boundary volume penalization methods: subsonic flow past the NACA0012 airfoil, transonic flow past the RAE2822 airfoil, subsonic flow past the MDA30P30N multi-element airfoil and a preliminary simulation of the subsonic flow around the NASA high-lift CRM aircraft. Finally, in Sect. 6, a summary of this work is presented, and advantages and disadvantages of the immersed boundary volume penalization for aerodynamic simulations in industrial scenarios are discussed.

2 Governing equations

2.1 Navier–Stokes equations

The Navier–Stokes equations can be written as a hyperbolic-parabolic system of conservation laws in the following way

$$\frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot \boldsymbol{f}(\boldsymbol{u}, \nabla \boldsymbol{u}) = \boldsymbol{0}.$$

The vector of conserved quantities is defined by $\boldsymbol{u}(\boldsymbol{x}, t) = (\rho, \rho \boldsymbol{v}, \rho E)$, where ρ is the mass density, $\boldsymbol{v} = \left(v_x, v_y, v_z\right)$ is the velocity vector and E is the total energy. The physical flux is defined by

$$\boldsymbol{f}(\boldsymbol{u}, \nabla \boldsymbol{u}) = \boldsymbol{f}^A(\boldsymbol{u}) - \boldsymbol{f}^D(\boldsymbol{u}, \nabla \boldsymbol{u}),$$

with the convective and viscous fluxes given, respectively, by

$$\boldsymbol{f}^A(\boldsymbol{u}) = \begin{pmatrix} \rho \boldsymbol{v} \\ \rho \boldsymbol{v} \otimes \boldsymbol{v} + p\boldsymbol{I} \\ \boldsymbol{v}(\rho E + p) \end{pmatrix}, \quad \boldsymbol{f}^D(\boldsymbol{u}, \nabla \boldsymbol{u}) = \begin{pmatrix} 0 \\ -\boldsymbol{\tau} \\ \boldsymbol{\tau} \cdot \boldsymbol{v} - \boldsymbol{q} \end{pmatrix}.$$

The viscous stresses are described by the stress tensor $\boldsymbol{\tau}$, defined by

$$\boldsymbol{\tau} = 2\mu \boldsymbol{S}^D = \mu\Big(\nabla \boldsymbol{v} + (\nabla \boldsymbol{v})^\intercal - \frac{2}{3}(\nabla \cdot \boldsymbol{v})\boldsymbol{I}\Big),$$

where $\boldsymbol{S}^D$ is the deviatoric component of the strain-rate tensor

$$\boldsymbol{S} = \frac{1}{2}\big(\nabla \boldsymbol{v} + (\nabla \boldsymbol{v})^\intercal\big)$$

and

$$\boldsymbol{q} = -k\nabla T, \quad k = \mu \frac{c_p}{\mathrm{Pr}}.$$

The equation of state

$$p = (\gamma - 1)\Big(\rho E - \frac{1}{2}\rho v^2\Big),$$

where p is the static pressure, γ is the ideal gas index, μ is the dynamic viscosity, T is the temperature, k is the thermal conductivity, and μ denotes the dynamic viscosity coefficient. The kinematic viscosity coefficient is defined by the formula

$$\nu = \mu/\rho.$$

2.2 Reynolds-averaged Navier–Stokes equations

The Navier–Stokes equations govern the motion of fluids in both laminar and turbulent regimes. Because of the large range of spatial and temporal scales present in turbulent flows, directly solving the Navier–Stokes equations is prohibitively expensive. Therefore, the Reynolds–averaged Navier–Stokes (RANS) equations are solved instead to model steady turbulent mean flows. The RANS equations couple the mean flow equations with the one-equation turbulence model of Spallart–Allmaras. These equations can be written also as a hyperbolic-parabolic system of conservation laws with source term in the following way

$$\frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot \boldsymbol{f}(\boldsymbol{u}, \nabla \boldsymbol{u}) = \boldsymbol{s}(\boldsymbol{u}).$$

Here, $\boldsymbol{u}$ is the vector of time-averaged conservative variables over a given time interval and has the following components: $\boldsymbol{u}(\boldsymbol{x}, t) = (\rho, \rho \boldsymbol{v}, \rho E, \rho \tilde{\nu})$, where ρ is the time-averaged mass density, $\boldsymbol{v} = \big(v_x, v_y, v_z\big)$ is the time-averaged velocity vector, E is the time-averaged total energy, and $\rho\tilde{\nu}$ is a new conservative variable relating the time-averaged mass density and the eddy viscosity $\tilde{\nu}$. The advective and diffusive components of the physical flux associated with the quantity $\rho\tilde{\nu}$ are given by

$$\boldsymbol{f}^A[\rho\tilde{\nu}] = \rho\tilde{\nu}\boldsymbol{v}, \quad \boldsymbol{f}^D[\rho\tilde{\nu}] = \frac{1}{\sigma}\big(\mu + f_{n1}\rho\tilde{\nu}\big)\nabla\tilde{\nu}$$

In the diffusive fluxes, the turbulent stress tensor $\boldsymbol{\tau}_t$ and the turbulent heat fluxes $\boldsymbol{q}_t$ are added respectively to $\boldsymbol{\tau}$ and $\boldsymbol{q}$ appearing in the mean flow equations:

$$\boldsymbol{\tau}_t = 2\mu_t \boldsymbol{S}^D, \quad \boldsymbol{q}_t = -\frac{\mu_t}{\mathrm{Pr}_t} c_p \nabla T$$

where $\mathrm{Pr}_t = 0.9$ is the turbulent Prandtl number and μ_t is the turbulent dynamic viscosity

$$\mu_t = \begin{cases} \rho\tilde{\nu} f_{v1}(\chi) & \text{for } \tilde{\nu} \geq 0, \\ 0 & \text{for } \tilde{\nu} < 0, \end{cases}, \quad f_{v1}(\chi) = \frac{\chi^3}{\chi^3 + c_{v1}^3}, \quad \chi = \frac{\rho\tilde{\nu}}{\mu}.$$

The source terms only have nonzero component in the equation for the turbulent variable $\rho\tilde{\nu}$:

$$\boldsymbol{S}[\rho\tilde{\nu}] = -\rho(P - D) - \frac{c_{b2}}{\sigma}\rho\nabla\tilde{\nu} + \frac{1}{\sigma}\big(\nu + f_{n1}\tilde{\nu}\big)\nabla\rho \cdot \nabla\tilde{\nu},$$

where the production and destruction terms, P and D, are defined by:

$$P = \begin{cases} c_{b1}\big(1 - f_{t2}\big)\tilde{\omega}\tilde{\nu} & \text{for } \tilde{\nu} \geq 0, \\ c_{b1}\big(1 - c_{t3}\big)\omega\tilde{\nu} & \text{for } \tilde{\nu} < 0, \end{cases}$$

$$D = \begin{cases} \Big(c_{w1} f_w - \frac{c_{b1}}{\kappa^2} f_{t2}\Big)\Big(\frac{\tilde{\nu}}{d}\Big)^2 & \text{for } \tilde{\nu} \geq 0, \\ -c_{w1}\Big(\frac{\tilde{\nu}}{d}\Big)^2 & \text{for } \tilde{\nu} < 0, \end{cases}$$

and

$$f_{n1} = \begin{cases} 1 & \text{for } \tilde{\nu} \geq 0, \\ \frac{c_{n1} + \chi^3}{c_{n1} - \chi^3} & \text{for } \tilde{\nu} < 0, \end{cases},$$

$$f_{t2} = c_{t3} \exp\big(-c_{t4}\chi^2\big), \quad f_w = g\left(\frac{1 + c_{w3}^6}{g^6 + c_{w3}^6}\right)^{\frac{1}{6}}$$

with

$$g = r + c_{w2}\big(r^6 - r\big), \quad r = \min\Big(r_{\mathrm{lim}}, \frac{\tilde{\nu}}{\omega\kappa^2 d^2}\Big),$$

and d is the distance to the nearest wall and ω the vorticity magnitude. The modified vorticity magnitude $\tilde{\omega}$ is given by

$$\tilde{\omega} = \begin{cases} \omega + \bar{\omega} & \text{for } \bar{\omega} > -c_{v2}\omega, \\ \omega + \frac{\omega\big(c_{v2}^2 + c_{v3}\bar{\omega}\big)}{\big(c_{v3} - 2c_{v2}\big)\omega - \bar{\omega}} & \text{for } \bar{\omega} < -c_{v2}\omega, \end{cases}$$

where $\bar{\omega}$ and f_{v2} are given by

$$\bar{\omega} = \frac{\tilde{\nu}}{\kappa^2 d^2} f_{v2}, \quad f_{v2} = 1 - \frac{\chi}{1 - f_{v1}}$$

For the sake of completeness, we give the values of the constants in the above expressions: $c_{v1} = 7.1$, $\sigma = 2/3$, $c_{b1} = 0.1355$, $c_{b2} = 0.622$, $\kappa = 0.41$, $c_{w2} = 0.3$, $c_{w3} = 2$, $r_{\text{lim}} = 10, c_{t3} = 1.2, c_{t4} = 0.5, c_{v2} = 0.7, c_{v3} = 0.9, c_{n1} = 16$.

3 Numerical methods

CODA is a CFD solver, which integrates an extensive set of algorithms for solving partial differential equations in a multi-physics context. In this work, we focus on the implementation of an immersed boundary methodology for the modelling of turbulent fluid flows by the well-known Reynolds-Averaged Navier–Stokes equations. In our formulation, we have selected the Spalart–Allmaras turbulent model that we briefly reproduce in Sect. 2 for completeness of the paper. Regarding the spatial and time discretization employed in CODA, we summarize the key ingredients in the next section.

3.1 Spatial and time discretization

The spatial discretization schemes used in CODA to solve the Navier–Stokes equations and the Reynolds-Averaged Navier–Stokes equations are the finite volume method [39], the modal discontinuous Galerkin method [40], and the discontinuous Galerkin spectral element method [41, 42]. The partial differential equations of interest can be written as a hyperbolic-parabolic system of conservation laws in differential form (see Sect. 2 for more details)

$$\frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot \boldsymbol{f}(\boldsymbol{u}, \nabla \boldsymbol{u}) = \boldsymbol{s}(\boldsymbol{u}),$$

where $\boldsymbol{u} = \boldsymbol{u}(\boldsymbol{x}, t)$ is the vector of conserved quantities, and $\boldsymbol{f} = \boldsymbol{f}(\boldsymbol{u}, \nabla \boldsymbol{u})$ is the tensor of physical fluxes, which has two terms, the advective flux and the diffusive term, namely

$$\boldsymbol{f}(\boldsymbol{u}, \nabla \boldsymbol{u}) = \boldsymbol{f}^A(\boldsymbol{u}) - \boldsymbol{f}^D(\boldsymbol{u}, \nabla \boldsymbol{u}),$$

and $\boldsymbol{s} = \boldsymbol{s}(\boldsymbol{u})$ represents the source terms. The conservation laws can also be written in integral form as follows

$$\frac{\partial}{\partial t} \int_\Omega \boldsymbol{u} \, \mathrm{d}\Omega + \oint_{\partial\Omega} \left(\boldsymbol{f}^A(\boldsymbol{u}) - \boldsymbol{f}^D(\boldsymbol{u}, \nabla \boldsymbol{u}) \right) \cdot \boldsymbol{n} \, \mathrm{d}\sigma = \int_\Omega \boldsymbol{s} \, \mathrm{d}\Omega.$$

Here, Ω is the spatial domain and $\boldsymbol{n}$ is the normal vector on the surface $\partial\Omega$ enclosing the spatial domain. In this work, we are interested only in the finite-volume schemes. A short description of these methods is outlined; for the discontinuous Galerkin schemes, see [43]. In the finite-volume framework, the computational domain is divided into a set of non-overlapping polyhedral control volumes or cells, that is, $\Omega = \bigcup_i^N \Omega_i$, and the integral form of the equations is discretized for each control volume Ω_i. Defining the mean value of the function $\boldsymbol{u}$ in the cell Ω_i by

$$\boldsymbol{u}_i := \frac{1}{|\Omega_i|} \int_{\Omega_i} \boldsymbol{u}(\boldsymbol{x}, t) \, \mathrm{d}\Omega,$$

we get the so-called semi-discrete formulation

$$\frac{\mathrm{d}\boldsymbol{u}_i}{\mathrm{d}t} := -\frac{1}{|\Omega_i|} \left\{ \sum_l \left[\boldsymbol{f}^A \cdot \hat{\boldsymbol{n}} \right]^* \sigma_l - \sum_l \left[\boldsymbol{f}^D \cdot \hat{\boldsymbol{n}} \right]^* \sigma_l - \boldsymbol{s} |\Omega_i| \right\}.$$

The surface integral has been approximated by a sum of the fluxes through the faces $\partial\Omega_l$ of the cell Ω_i. The σ_l stands for the area of the face $\partial\Omega_l$, $\hat{\boldsymbol{n}}$ is the normal vector to the face $\partial\Omega_l$, and $|\Omega_i|$ is the volume of the cell Ω_i. The flux across the element faces is computed via numerical fluxes (this is represented in the equation as the operator $[\cdot]^*$). The computation of the numerical advective fluxes requires a reconstruction procedure that takes cell-centered values of conserved quantities, momentum and temperature gradients, and interpolate them to the faces. Typically, the Roe schemes with an entropy fix are employed. The gradients are computed based on the Green–Gauss theorem. This procedure reconstructs values needed at face integration points to approximate the Green–Gauss integral linearly. Gradient limiters can be applied in the vecinity of strong discontinuities. The finite volume discretization implemented in CODA is second-order accurate. More details on how are computed the element and face gradients can be found in [1, 44, 45].

Regarding the time discretization, in CODA are available explicit, implicit and implicit-explicit integrators. In this work we have used an implicit backward-Euler scheme based on preconditioned matrix-free GMRES. Preconditioners include incomplete LU and line-inversion [46].

3.2 Immersed boundary methods

Immersed Boundary Methods are numerical techniques for simulating fluid–structure interaction problems involving complex geometries, providing an alternative to numerical methods based on body-fitted meshes. Immersed Boundary Methods were first proposed by Peskin [4] to simulate blood flow in the human heart. These methods provide a flexible framework for modeling fluid flow over complicated immersed bodies on grids that do not conform to the surface of the body by treating these as immersed boundaries within the computational domain. In this way, immersed boundary methods overcome the challenges of generating high-quality meshes around complex geometries, which can be time-consuming and computationally demanding, while accurately resolving complex flows using simple grids.

The geometric components are discretized using a separate representation, such as a surface or volume mesh, which is then immersed within the computational grid. Immersed boundary conditions are applied to take into account the geometric components lying in the flow.

Because the meshes used in the immersed boundary method are, in principle, non-conforming with the geometry of the obstacle, a very high resolution of this background mesh is required to obtain reliable numerical solutions. The employment of very fine meshes in the whole computational domain makes unpractical the use of immersed boundary methods for industrial applications, like the study of the aerodynamics of aircrafts. In fact, very fine meshes are only necessary in specific places in the computational domain, like around the immersed geometry and the wake regions. Fine meshes far away from the immersed bodies are unnecessary. For these reasons, mesh refinement algorithms are used to refine the mesh only in the regions where it is actually essential for accurate computations.

3.2.1 Immersed boundary volume penalization method

In this family of immersed boundary volume penalization methods, the flow equations are solved on the whole computational domain, while a source term is introduced to represent the body as a porous medium with very low permeability [26, 47–50]. This method is based on the idea of penalizing the integration points which are located inside the immersed body using source terms, instead of using a boundary condition on the surface of the body, as it is the case when using conforming meshes. In this way, a simple Cartesian mesh can be used to solve the Navier–Stokes equations over arbitrary geometries, reducing the time required for the mesh generation Fig. 1.

In the IBVP method, a mask function is required to discriminate between those points that are inside or outside the immersed body, as it is represented in Fig. 1. Then, source terms are applied to modify the behavior of the flow field inside the immersed body. Given the governing equations for a compressible viscous fluid:

$$\frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot \boldsymbol{f}(\boldsymbol{u}, \nabla \boldsymbol{u}) = \boldsymbol{s}(\boldsymbol{u}) \tag{1}$$

where $\boldsymbol{s}(\boldsymbol{u})$ is the IBM source term. This source term is written as

$$\boldsymbol{s}(\boldsymbol{u};\chi,\eta) = \frac{\chi}{\eta}\begin{pmatrix} 0 \\ \rho(\boldsymbol{v}-\boldsymbol{v}_s) \\ \frac{1}{2}\rho\left(\boldsymbol{v}\cdot\boldsymbol{v}-\boldsymbol{v}_s\cdot\boldsymbol{v}_s\right) \end{pmatrix}, \tag{2}$$

where $\boldsymbol{v}_s$ is the velocity of the moving geometry (note that for static object $\boldsymbol{v}_s = 0$ and χ represents the mask function and distinguishes between the fluid region Ω_f and the solid region Ω_s:

$$\chi(\boldsymbol{x},t) = \begin{cases} 1, & \text{if } \boldsymbol{x} \in \Omega_s, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

and $0 < \eta \ll 1$ is the penalization parameter. If a RANS Spalart–Allmaras model is used, the eddy viscosity, $\tilde{\nu}$, is set to zero inside the body [51]:

$$\boldsymbol{s}_{SA}(\boldsymbol{u};\chi,\eta) = -\frac{\chi}{\eta}\tilde{\nu}. \tag{4}$$

Furthermore, a wall model has been developed to improve the accuracy of RANS simulations. This model selects a set of points near the immersed body where the wall function will be applied, which are called wall points. For each wall point, the normal vector to the body's surface is computed and a new point is defined, called image point. The image points will provide the required information to the wall function to update the state vector at the wall points. However, image points (in general) are not integration points of the computational mesh. Therefore, the state vector at the image points has to be interpolated from a set of interpolation points. These interpolation points, which are computed through a kd-tree algorithm, are the N-nearest integration points to each image point. An overview of this approach is represented schematically in Fig. 2. The distance which is used to discriminate between the regions where the wall

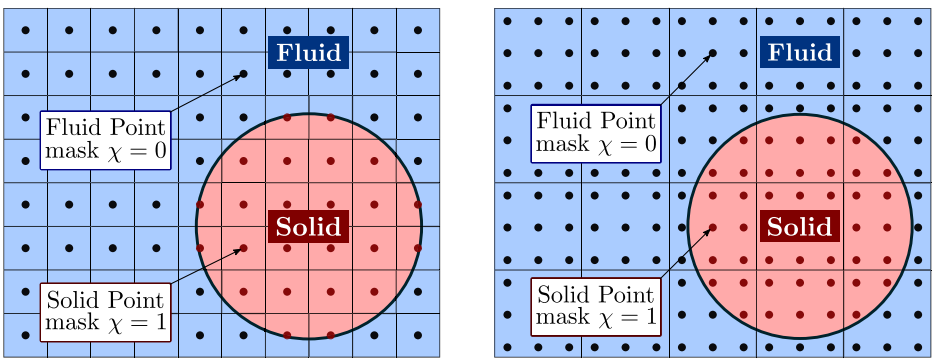

Fig. 1 Schematic diagram of the background mesh for the immersed boundary method as it used along with finite volume schemes (left) and discontinuous Galerkin spectral element methods (right)

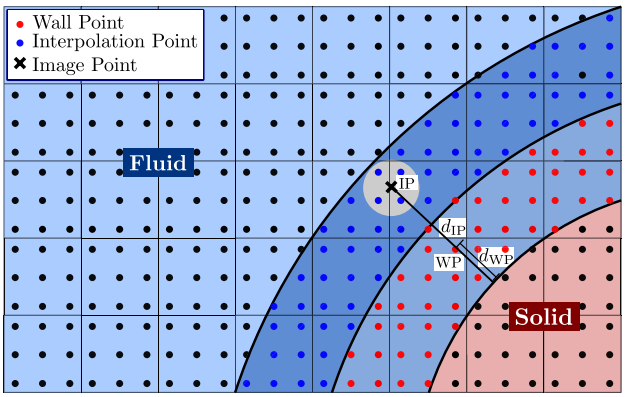

Fig. 2 Schematic definition of wall points, image points and interpolation points

points and the image points live, is defined as the length of the smallest edge of the smallest cell of the mesh.

To interpolate the state vector at each image point, a Radial Basis Function (RBF) interpolator with a Gaussian kernel is used. Then, a wall function is used to compute the friction velocity at each image point [52]:

$$v_{||}^{+}(y^{+}) = \frac{1}{\kappa}\log\left(1+\kappa y^{+}\right) + \left(C - \frac{1}{\kappa}\log\left(\kappa\right)\right) \left(1 - \exp\left(-\frac{y^{+}}{11}\right) - \frac{y^{+}}{11}\exp\left(-\frac{y^{+}}{3}\right)\right), \tag{5}$$

and

$$v_{\mathrm{IP},t} = v_{\tau} v_{||}^{+}\left(y_{\mathrm{IP}}^{+}\right), \quad \text{with} \quad y_{\mathrm{IP}}^{+} = \frac{v_{\tau} d_{\mathrm{IP}}}{\nu_{\mathrm{IP}}}, \tag{6}$$

where the subscript IP refers to each image point and the subscript t refers to the tangential component of the velocity vector.

Finally, the resulting friction velocity v_{τ} is considered to be constant along the normal direction and, hence, it can be used at the wall points (subscript WP). Taking into account this information, the velocity, the eddy viscosity and the non-dimensional wall distance can be obtained at the wall points:

$$y_{\mathrm{WP}}^{+} = \frac{v_{\tau} d_{\mathrm{WP}}}{\nu_{\mathrm{WP}}}, \tag{7}$$

$$v_{\mathrm{WP},t} = v_{\tau} v^{+}\left(y_{\mathrm{WP}}^{+}\right), \tag{8}$$

$$v_{\mathrm{WP},n} = \frac{d_{\mathrm{WP}}}{d_{\mathrm{IP}}} v_{\mathrm{IP},n}, \tag{9}$$

$$\tilde{\nu}_{\mathrm{WP}} = \kappa v_{\tau} d_{\mathrm{WP}}, \tag{10}$$

which can be solved including an additional condition of parallelism:

$$v_{\mathrm{WP},t} \parallel v_{\mathrm{IP},t} \tag{11}$$

The subscript n refers to the normal direction of the velocity vector, $\kappa = 0.41$ is the von Kármán constant and d is the distance from a point to the immersed body. The new velocity and turbulent viscosity at the wall points are considered by including additional source terms for the momentum, energy and turbulent viscosity equations. Finally, this wall model as implemented in CODA along with the IBVP scheme has proved to improve the results of the standard RANS simulation if the y^{+} at the wall points has a value (approximately) of 60 or less.

4 Preprocessing and postprocessing

4.1 Preprocessing

Immersed boundary methods use a background mesh for solving a fluid–structure interaction problem. In 3D applications, this background mesh is typically a simple Cartesian mesh made of hexahedral elements. However, it can be also unstructured and made of hybrid elements (hexahedra, prisms, tetrahedra, etc.), or even a body-fitted mesh. In this section we focus on the construction of a Cartesian background mesh built out of hexahedra. This kind of mesh is assembled in an unstructured fashion by the preprocessing tool.

4.1.1 Background mesh generation

We have developed a preprocessing tool to create meshes made of hexahedral elements. The preprocessing tool has two ways for generating the initial background mesh: the first way consists in creating a new Cartesian mesh from scratch, and the second way consists in importing a mesh generated by the software GMSH [53] (strictly speaking, the preprocessing tool is not generating a mesh but importing an already generated one by an external tool). Our preprocessing tool is only currently capable of handling meshes made of hexahedral elements. Next, we will describe the tasks the preprocessing tool does for constructing an unstructured and adaptively refined Cartesian mesh.

4.1.2 Loading parameters for background mesh generation

The first task to be performed is to load from file the parameters of the mesh to be constructed. Depending on

the type of mesh to be generated (built-in or imported mesh), different parameters are read-in.

For the built-in mesh type, the preprocessing tool requires some parameters that specify the type of built-in box (standard box, curved box, box deformation function, type of target geometry, mesh stretching factors in each spatial direction, etc.). The number of elements in each direction of the Cartesian box, the degree of the mesh (for high-order curved meshes), and the kind of curved boundary are also required.

For the imported mesh type, the filename of the GMSH file is required. Additional parameters are necessary for setting up the name and location of boundary conditions and the output format of the mesh file (formats supported: HDF5, GMSH, and TECPLOT).

4.1.3 Creation of the background mesh

Once the mesh parameters are loaded, the preprocessing tool proceeds to create the background mesh, either a built-in or an imported mesh. The typical workflow of the preprocessing tool is as follows:

- Creation of interpolation matrices between different polynomial bases.
- Creation of a reference box.
- Creation of a curved box from the reference box.
- Application of the stretching functions on each spatial direction, if required.
- Application of mesh deformation functions, if required.
- Application of mesh rotation matrices, if required.
- Storing the final mesh in arrays.
- Numbering of nodes, elements, elements faces and boundary faces.
- Creation of the main data structure of the mesh: the list of elements with their corresponding information (nodes ID, elements ID, local faces definition, boundary conditions). This elements list is a dynamical data structure, namely, a linked list. Taking into account that the number of elements increases dynamically according to the desired target mesh size around the immersed geometry, this data structure makes possible a straightforward application of the mesh refinement algorithm.

4.1.4 Mesh refinement around immersed geometries

Once the background mesh is generated and stored in the element list data structure, the preprocessing tool will refine this mesh provided the geometry of the body is in STL format.

4.1.5 Loading parameters for the mesh refinement

From the same parameter file used in the generation of the background mesh, the preprocessing tool will load all the information necessary for the mesh refinement task: the maximum level of refinement around the immersed geometry, the number and location of special regions that need to be refined, and their corresponding levels of refinement, the type of refinement (either isotropic or anisotropic), and the filenames of the STL files used to describe the geometry of the immersed bodies. After reading this information, the preprocessing tool will perform mesh refinement around the immersed geometry.

4.1.6 Importing geometries from STL files

The immersed geometries are described by STL files. These files contain an unstructured triangulated surface (triangle unit vectors and the coordinates of their corresponding vertices) as generated by the CAD software. The preprocessing tool is responsible for loading the tessellated surface of the immersed body for further use in the refinement stage.

4.1.7 Mesh refinement around immersed geometries

The preprocessing tool has the capability of refining the background hexahedral mesh in special regions within the computational domain (for instance, wake regions) and also around the triangulated surface of the immersed geometry. The preprocessing tool will follow the next steps for refining the background mesh:

- **Flagging the elements that require refinement**. For the refinement of special regions in the computational domain, the flagging procedure marks all hexahedral cells inside the region coordinates provided in the parameter file. For the refinement of the hexahedral elements intersecting the triangulated surface of the immersed body, an efficient algorithm for the triangle-box overlap is implemented [54]. Before mesh refinement takes place, this triangle-box overlap is tested for all triangles of the surface triangulation, and if such overlap takes place, the involved elements store the overlapped triangles. During the mesh refinement loop (refinement level $l > 0$), this triangle-box overlap is tested only for those hexahedra storing overlapped triangles.
- **Flagging the elements for balancing**. The FSDM and, therefore, CODA can handle hanging nodes for different types of face elements (triangular and quadrilateral faces). This hanging nodes capability is restricted in FSDM to elements with a 2:1 refinement level ratio. This means that for two neighboring elements with different refinement levels, only a level difference of at most 1

between neighboring cells is allowed. The preprocessing tool is aware of this restriction and therefore an algorithm for preserving the 2:1 balancing has been implemented.

- A feature implemented in the preprocessing tool is the refinement of elements around the immersed body up to certain number of neighboring cells far away from the truly overlapped elements with the geometry. This allows us to have a thick layer of small elements around the immersed body, which is useful in the simulation of flow problems with the RANS equations and the use of wall functions.
- **Refinement of hexahedral elements**. All flagged cells are split into 4 or 8 new children cells (for anisotropic or isotropic mesh refinement, respectively). Hanging nodes can appear, and the relation between the face nodes of neighboring elements with different levels of refinement has to be computed and exported in such a way that FSDM can handle them. For the hexahedral meshes used within the preprocessing tool and when hanging nodes are present, the Quad2Quad and Quad4Quad interfaces are computed (see Fig. 3). FSDM treats the hanging nodes in the following way: With help of pseudo element types, FSDM stores "hanging" connections. These are ignored by the solver, have no volume and no solution values, but at the same time they are used by the face extractor to create the face based grid. After that, FSDM completes the hanging node grid to a kind of (pseudo) conformity, which enable the adaptation to work on hanging elements in the same way (only with other element types) as on conforming elements.
- **Storing the final mesh in arrays**. In these arrays are stored the nodes, elements, conforming faces and non-conforming faces.
- Exporting the mesh arrays to HDF5 format, following the FSDM guidelines for the HDF5 format.

4.2 Postprocessing

The postprocessing for the IBVP methodology is based on the computation of pressure coefficient, skin friction coefficient, lift coefficient and drag coefficient on the immersed body surface. Two different approaches are considered: integration over the STL and over a modified immersed body tessellation integration.

4.2.1 Integration over the original immersed body tessellation

This approach takes advantage of the information within the STL file to perform the integration. First, for each triangle of the tessellation, the fluid variables are interpolated at the barycenter, at the vertices, and at the middle point of each edge of the triangle. The interpolation can be performed using different approaches: Radial Basis Functions (RBF), Inverse Distance, or Linear Interpolation. In general, a simple linear interpolation has shown good behavior and robustness.

Then the viscous stress tensor (including the pressure) $\tilde{\boldsymbol{\tau}} = \boldsymbol{\tau} - p\boldsymbol{I}$ is computed at those points. Finally, the integration over each triangle is performed as follows:

$$\boldsymbol{F}_t = \frac{S}{60}\left(27\left[\tilde{\boldsymbol{\tau}} \cdot \hat{\boldsymbol{n}}_t\right]_{\text{bary}} + 3\sum_{i=1}^{3}\left[\tilde{\boldsymbol{\tau}} \cdot \hat{\boldsymbol{n}}_t\right]_{\text{vert},i} + 8\sum_{i=1}^{3}\left[\tilde{\boldsymbol{\tau}} \cdot \hat{\boldsymbol{n}}_t\right]_{\text{mid},i} \right). \tag{12}$$

In this case, the integrated force F_t on the triangle t of area S is computed as a weighted average of the stress tensor (including the pressure), $\tilde{\boldsymbol{\tau}}$, projected over the normal vector of the local triangle. The value of $\left[\tilde{\boldsymbol{\tau}} \cdot \hat{\boldsymbol{n}}_t\right]_{\text{bary}}$, $\left[\tilde{\boldsymbol{\tau}} \cdot \hat{\boldsymbol{n}}_t\right]_{\text{vert},i}$, and $\left[\tilde{\boldsymbol{\tau}} \cdot \hat{\boldsymbol{n}}_t\right]_{\text{mid},i}$ correspond to the projected force at the barycenter, at the vertex i and at the middle point of the side i respectively.

4.2.2 Integration over a modified immersed body tessellation

This approach provides an alternative to compute the lift and drag when the STL is too coarse (and the integration over the STL shows a bad quality) or too fine (and the computational cost has to be reduced). In these cases, a new set of points on the surface of the immersed body are defined to perform the integration.

First, a region around the body is defined and every integration point of the mesh within that region is projected over the body's surface. Then, an algorithm is used to compute the weights for each point. Once the points on the surface (and their weights) are known, the fluid variables are interpolated on those points. Finally, the stress tensor (including the

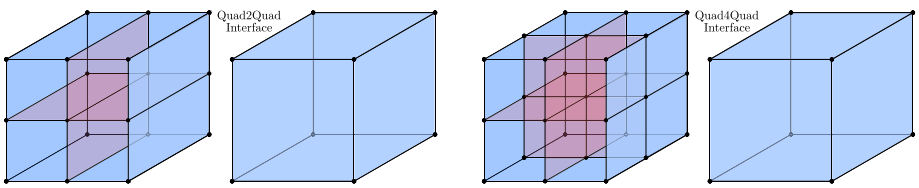

Fig. 3 Hanging nodes distribution in a Quad2Quad interface (left) and Quad4Quad interface (right)

pressure) is computed following the same equations shown above and the projected force is integrated as follows:

$$F = \sum_{i=0}^{\text{nPoints}} \tilde{\tau}_i \cdot \hat{\boldsymbol{n}}_i \omega_i. \tag{13}$$

In this case, the global force is computed as the dot product between the projected force at each point, $\tilde{\boldsymbol{\tau}}$, and the weights, ω.

5 Numerical computations

To prove the validity and performance of this approach, we present in this section several numerical simulations of different fluid flow problems of increasing difficulty. These tests are the subsonic flow around the NACA0012 airfoil at different angles of attack, the transonic flow around the RAE2822 airfoil, the subsonic flow around the MDA30P30N multi-element airfoil, and the more challenging NASA High-Lift Common Research Model (CRM-HL) aircraft. Experimental results of most of those problems are reported in the literature and thus will serve as validation of our methodology. For all simulations, the RANS equations were used for modeling the fluid flow, and the numerical methods employed by CODA were, for the spatial discretization, the second order finite volume method coupled with the immersed boundary volume penalization. A linearized implicit Euler scheme is employed for time discretization. All simulations were executed using 600 cores in the Magerit Supercomputer at Supercomputing and Visualization Center of Madrid (CeSViMa) and 3840 cores in the Marenostrum4 Supercomputer at the Barcelona Supercomputing Center.

5.1 Flow around the NACA0012 airfoil

The computational domain for the fluid flow around the NACA0012 airfoil is the box with dimensions $[-20, +20] \times [-20, +20] \times [0, +1]$. The domain span is sufficient to avoid domain confinement effects and wave reflections from the domain boundaries, which could lead to a significant error when computing the lift and drag forces around the airfoil. The extent of the computational domain corresponds to $40c$, where c is the chord length and is equal to 1. The domain is initially discretized with $n_x n_y n_z = 1600$ hexahedral elements, where $n_x = 40$, $n_y = 40$, and $n_z = 1$ are the number of elements in the direction x, y, and z, respectively. This initial mesh is further refined around the immersed geometry and also in the wake region. The spacing of the grid obeys $h = 2^{-l}$, where l is the level of refinement. Sufficient grid resolution around the airfoil is crucial to obtain accurate results for the lift and drag coefficients. The mesh around the immersed geometry is refined up to the level of refinement $l = 16$, we get a mesh with around 4.99×10^6 elements. The wake region is resolved with a refined subregion with refinement level $l = 8$. This region is approximately 20 airfoil chords downstream of the leading edge of the airfoil. The characteristics of the meshes used for the mesh convergence study are shown in Table 1; ranging from a coarse mesh to a very fine mesh. The refinement level $l_{\max}$ listed in the table is the highest level of refinement around the immersed body. The immersed geometry is located in the center of the computational domain, with its cross-section in the plane z, and this corresponds to the NACA0012 profile with chord length $c = 1$. The triangulation of the surface geometry is made up of 65,468 triangles for the low resolution STL and 4,720,046 triangles for the high resolution STL. The geometry is shown in Fig. 4, and in Fig. 5 the mesh around the immersed geometry is depicted. In Fig. 6 are shown, at 10, 000x zoom, the meshes around the NACA0012 airfoil with maximum level of refinement $l_{\max} = 16$ and with the blanking activated; the leading edge is depicted for a low-resolution STL geometry (left) and a high-resolution STL geometry (right).

5.1.1 Subsonic flow at $M_\infty = 0.15$ and $\alpha = 0°$ and $\alpha = 10°$

The first test corresponds to the subsonic flow around an NACA0012 airfoil. This problem has become a classic test case for RANS solvers due to the simple geometry and the large amount of available numerical and experimental data [55–57]. It is used primarily for the analysis of turbulence models, testing their convergence properties and their effect on the accuracy. We perform simulations for the angle of attack $\alpha = 0°$ and $\alpha = 10°$. The flow satisfies the following conditions: a gas with adiabatic index $\gamma = 1.4$ and Reynolds number $\text{Re} = 6 \times 10^6$ flows with a free-stream Mach number $M_\infty = 0.15$. The non-dimensional density is set to $\rho = 1$, and the non-dimensional pressure to $p = 1$. On the boundaries of the computational domain the following boundary conditions were set: the left face is set to inflow, the right face to outflow, the front and back faces were set to periodic, and the top and bottom to far field. The inflow pressure assumes the value of the stagnation pressure and the outflow pressure to the static pressure.

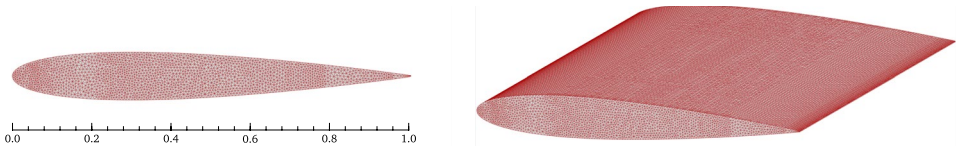

Fig. 4 Geometry used for the numerical simulations of the NACA0012 airfoil

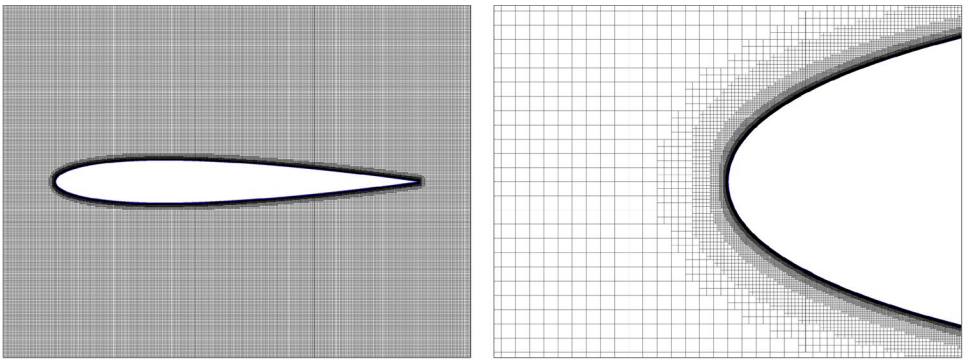

Fig. 5 Mesh refinement around the NACA0012 airfoil. The mesh has been generated with maximum level of refinement $l_{\max} = 16$ and with the blanking activated. Close-up view of the mesh at 50x zoom, depicting the full airfoil (left) and close-up view of the mesh at 500x zoom, depicting the leading edge (right)

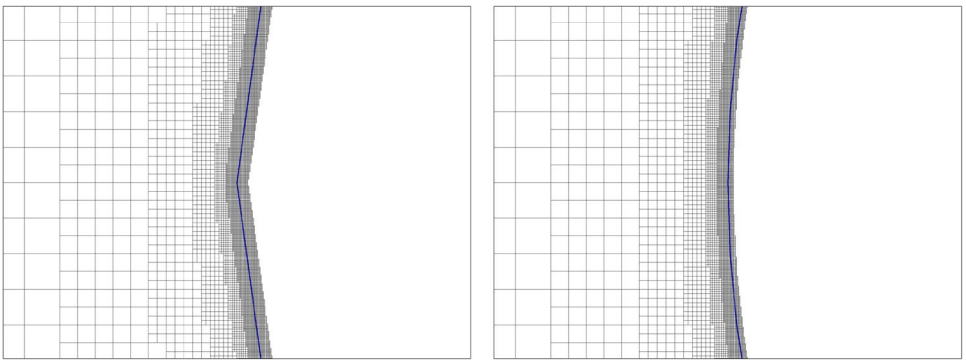

Fig. 6 Mesh refinement around the NACA0012 airfoil. The meshes have been generated with maximum level of refinement $l_{\max} = 16$ and with the blanking activated. Close-up view of the meshes at 10000x zoom, depicting the leading edge and using a low-resolution STL geometry (left) and a high-resolution STL geometry (right)

Table 1 Characteristics of the meshes for the flow around the NACA0012 airfoil

Name	$l_{\max}$	h_{wake}	h_{body}	*nElems*
Coarse	10	3.91×10^{-3}	9.77×10^{-4}	6.53×10^{5}
Medium	12	3.91×10^{-3}	2.44×10^{-4}	8.59×10^{5}
Fine	14	3.91×10^{-3}	6.10×10^{-5}	1.68×10^{6}
Very Fine	16	3.91×10^{-3}	1.53×10^{-5}	4.99×10^{6}

In the Figs. 7 and 8 are depicted the contour plots of the velocity magnitude for the flow past the NACA0012 profile at angle of attack $\alpha = 0°$. The mesh employed in the simulation is the very fine mesh with element's size around the body $h_{\text{body}} = 1.53 \times 10^{-5}$, element's size in the wake region $h_{\text{wake}} = 3.91 \times 10^{-3}$. The Spallart–Allmaras model has been activated. We observe the flow field is symmetric as the fluid past the airfoil. As the fluid flows over the blunt trailing edge, a minor amount of flow separation occurs, but it reattaches shortly after it passes the trailing edge and recovers the mean velocity in the distant far field.

In the Fig. 9 are depicted the contour plots of the velocity magnitude for the flow past the NACA0012 profile at angle of attack $\alpha = 10°$. The mesh employed is the same as for angle of attack $\alpha = 0°$. The Spallart–Allmaras model has been activated. We observe that the stagnation point is

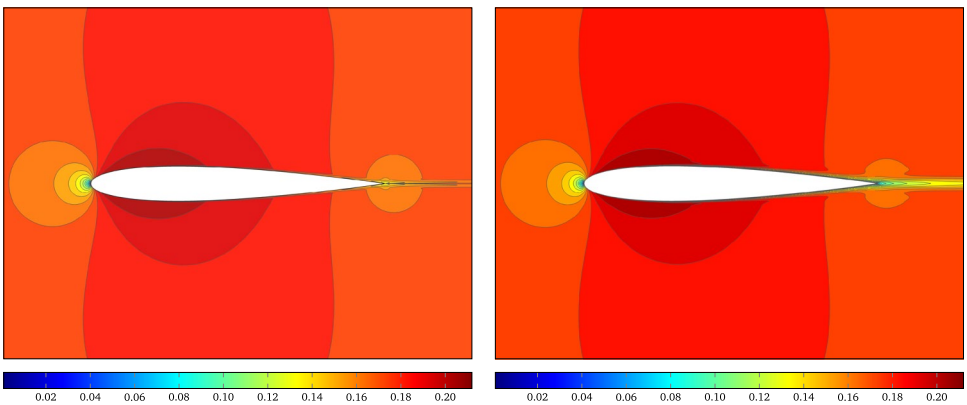

Fig. 7 Subsonic flow around the NACA0012 airfoil. Contour plots of the velocity magnitude for the simulations with angle of attack $\alpha = 0°$. Computations were performed using the CODA solver based on a body-fitted mesh (left) and an IBM Cartesian mesh with maximum level of refinement $l_{\max} = 16$ (right). In both computations, the RANS equations were solved with the wall model deactivated

shifted downward (relative to the airfoil), reflective of the shifted angle of attack. The fluid velocity above the airfoil is clearly different than that below the airfoil, showing a low-velocity region toward the upper side of the trailing edge, which means flow separation.

Now we analyze the pressure coefficient plots. The Figs. 10 and 11 show the pressure coefficient C_p on the airfoil surface for the simulation case with an angle of attack $\alpha = 0°$. All meshes in Table 1 have been used and the cases where the STL geometry has a low resolution and a high resolution. Computations with a deactivated and activated wall model have been considered. The experimental data are also plotted and were obtained from [55]. For an angle of attack $\alpha = 0°$, and for all considered meshes, for computations with a low-resolution STL geometry, the pressure coefficient curve is very close to the experimental data, but some unphysical oscillations are clearly perceptible in all simulations, both with the wall model switched off and switched on. These oscillations tend to diminish as the mesh becomes finer. For simulations with high-resolution geometry, the oscillations are present only for the coarser mesh. The STL geometry resolution has an important impact on the accuracy of the computations along with the mesh refinement. We can observe that the computations with the wall

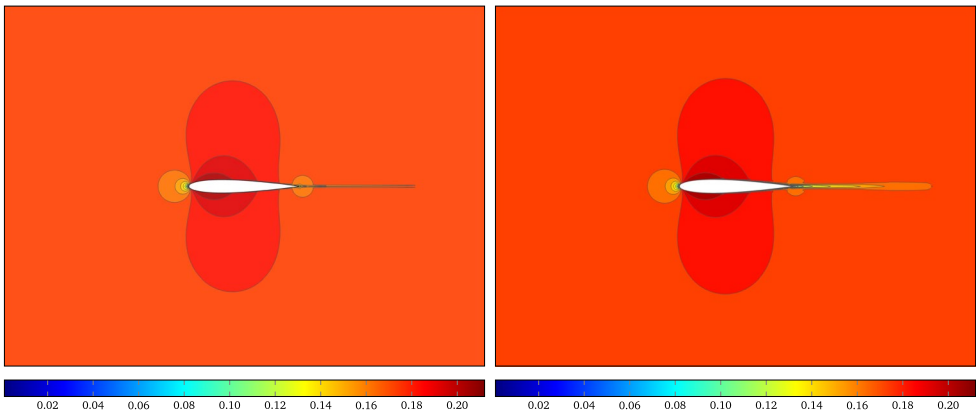

Fig. 8 Subsonic flow around the NACA0012 airfoil. Contour plots of the velocity magnitude for the simulations with angle of attack $\alpha = 0°$. Computations were performed using the CODA solver based on a body-fitted mesh (left) and an IBM Cartesian mesh with maximum level of refinement $l_{\max} = 16$ (right). In both computations, the RANS equations were solved with the wall model deactivated

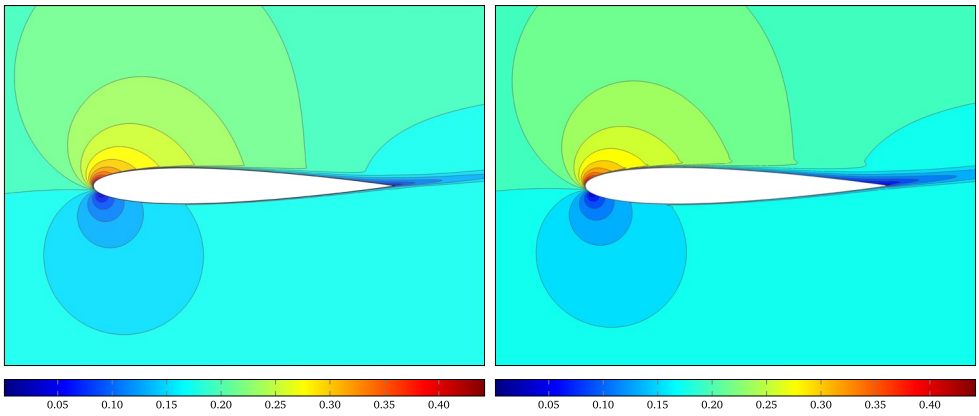

Fig. 9 Subsonic flow around the NACA0012 airfoil. Contour plots of the velocity magnitude for the simulations with angle of attack $\alpha = 10°$. Computations were performed using the CODA solver based on a body-fitted mesh (left) and an IBM Cartesian mesh with maximum level of refinement $l_{\max} = 16$ (right). In both computations, the RANS equations were solved with the wall model deactivated

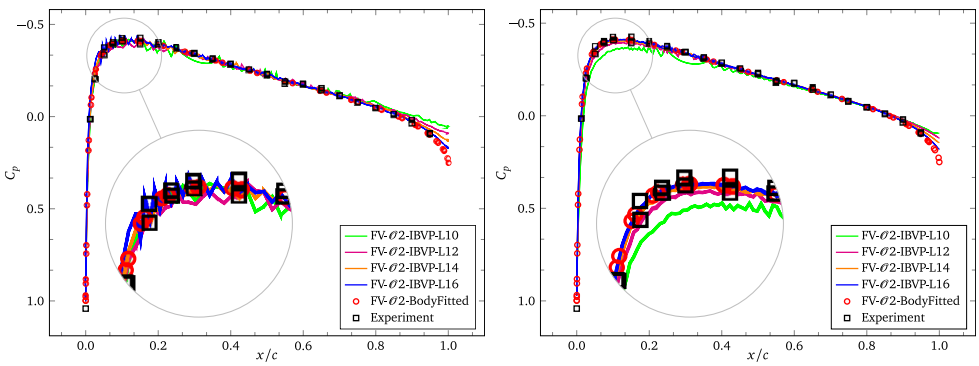

Fig. 10 Subsonic flow around the NACA0012 airfoil. Grid convergence study in terms of the pressure coefficient C_p for the subsonic flow over the NACA0012 airfoil surface at angle of attack $\alpha = 0°$ for the cases with low-resolution STL geometry (left) and high-resolution STL geometry (right). In both computations, the RANS equations were solved with the wall model deactivated

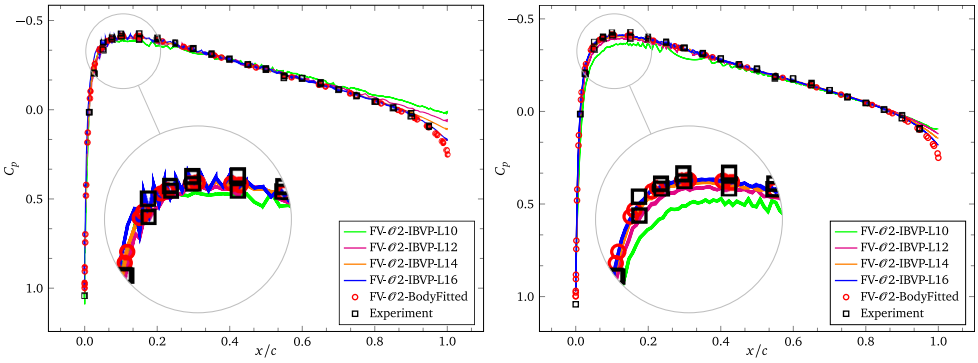

Fig. 11 Subsonic flow around the NACA0012 airfoil. Grid convergence study in terms of the pressure coefficient C_p for the subsonic flow over the NACA0012 airfoil surface at angle of attack $\alpha = 0°$ for the cases with low-resolution STL geometry (left) and high-resolution STL geometry (right). In both computations, the RANS equations were solved with the wall model activated

model deactivated and activated, and for high-resolution STL geometries, show very similar C_p curves, suggesting that for good enough resololutions (in terms of volume mesh and surface tessellation of the STL), wall functions are not necessary.

In Fig. 12 is represented by the pressure coefficient C_p on the airfoil surface for an angle of attack $\alpha = 10°$ and for all meshes reported in Table 1. Only the cases with deactivated wall model have been taken into account. The pressure coefficient curves for the coarse and medium meshes show a large discrepancy with respect to the experimental data, especially on the upper surface of the leading edge. Only the fine and very fine meshes are in good agreement with the experiment. The computations with high-resolution STL geometries show an oscillatory behavior for the coarse and medium meshes. For simulations with low-resolution meshes, at leading edge, the C_p seems to diverge, but this divergence is not present for the high-resolution STL geometry computations. Although C_p curves for computations with wall model activated are not shown, the diverging behavior is also present in the cases with fine and very fine meshes. We are convinced that geometry and mesh resolutions play an important role in simulating turbulent flows accurately with IBVP methods.

In Fig. 13 we show the skin friction coefficient curves at $z = 0.5$. We observe that the surface skin friction coefficient is not very well predicted as the pressure coefficient. However, as the resolution of the mesh increases, C_f is approaching to the reference values taken from the body-fitted simulations. This discrepancy is typically attributed to the way the gradients are interpolated. An appropriate way to reconstruct C_f is still an open question. Several strategies have been proposed in [58] and are worth investigating in future works.

The lift and drag values are shown in Table 2 for the angle of attack $\alpha = 0°$ and $\alpha = 10°$. For the simulation with angle of attack $\alpha = 10°$, the lift coefficient C_L increases monotonically as the grid is refined, getting closer to the experimental value, while the drag coefficient C_D approaches its numerical value to the experimental one. This behavior occurs for simulations with the wall model deactivated and activated. For the simulation with angle of attack $\alpha = 0°$, we do not observe monotonicity in the evolution of the lift coefficient C_L and the drag coefficient C_D as the mesh is refined, however, as the mesh is refined, the C_L and C_D approach to the body-fitted and experimental values.

5.1.2 Influence of mesh blanking on simulations performance

In the context of immersed boundary methods, the use of Cartesian meshes with refinement around the immersed geometry involves the generation of grids with a large number of elements. This represents a fundamental disadvantage

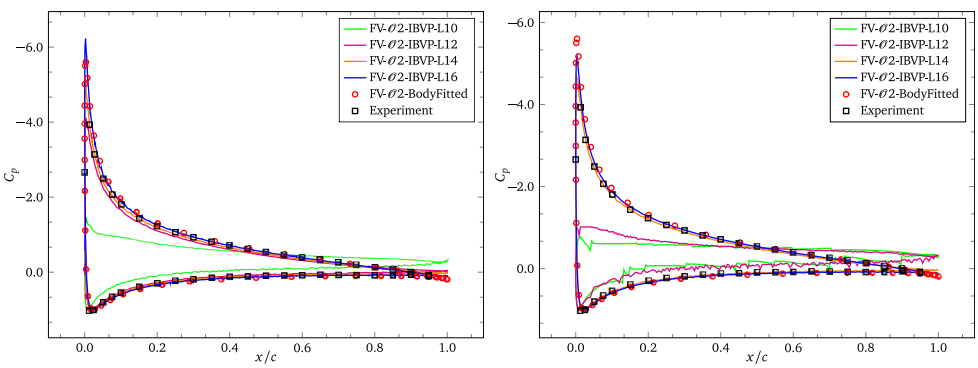

Fig. 12 Subsonic flow around the NACA0012 airfoil. Grid convergence study in terms of the pressure coefficient C_p for the subsonic flow over the NACA0012 airfoil surface at angle of attack $\alpha = 10°$ for the cases with low-resolution STL geometry (left) and high-resolution STL geometry (right). In both computations, the RANS equations were solved with the wall model deactivated

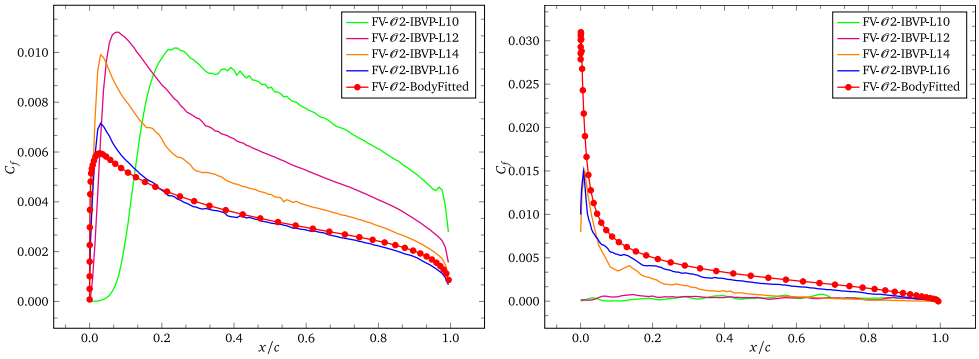

Fig. 13 Subsonic flow around the NACA0012 airfoil. Grid convergence study in terms of the skin friction coefficient C_f for the subsonic flow over the NACA0012 airfoil surface at angles of attack $\alpha = 0°$ (left) and $\alpha = 10°$ (right) and employing a high-resolution STL geometry. In both computations, the RANS equations were solved with the wall model deactivated

Table 2 Subsonic flow around the NACA0012 airfoil

Mesh	$\alpha = 0°$		$\alpha = 10°$	
	C_L	C_D	C_L	C_D
Coarse	4.381×10^{-6}	1.630×10^{-2}	2.031×10^{-1}	9.448×10^{-2}
Medium	2.472×10^{-6}	1.213×10^{-2}	2.511×10^{-1}	1.000×10^{-1}
Fine	6.884×10^{-6}	8.672×10^{-3}	4.627×10^{-1}	9.159×10^{-2}
Very fine	2.078×10^{-5}	6.147×10^{-3}	4.921×10^{-1}	9.323×10^{-2}
Body-fitted	9.430×10^{-6}	8.330×10^{-3}	1.090×10^{0}	1.230×10^{-2}
Experimental	7.240×10^{-3}	8.090×10^{-3}	1.057×10^{0}	1.190×10^{-2}

Lift and drag coefficients for angle of attack $\alpha = 0°$ and $\alpha = 10°$

with respect to CFD solvers based on body-fitted meshes. Mesh blanking in IBM Cartesian meshes is a procedure used to reduce the computation time when similar results are sought among CFD solvers based on IBM techniques and body-fitted meshes. In this work, we explore the mesh blanking to assess the performance of the CODA solver with IBM. In blanked meshes, elements within the immersed geometry are removed, leaving a thin layer of elements inside the body (see Fig. 14). In Table 3 is shown the performance comparison between simulations of the NACA0012 subsonic flow at angle of attack $\alpha = 0°$ with immersed boundary methods and with meshes without blanking and with blanking. Computations done using blanked meshes have a superior efficiency compared to simulations using meshes without blanking. We can see that computations using blanked meshes require two orders of magnitude fewer linear iterations than computations with no blanked meshes in order to converge to a solution with similar residual in the density in computations performed with body-fitted meshes.

5.2 Transonic flow around the RAE2822 airfoil

Now we will consider the transonic flow around the RAE2822 airfoil. This airfoil has been tested in the RAE wind tunnel at different flow conditions in the range of Mach numbers $0.676 - 0.750$ and at several Reynolds numbers. The test case considered in our study is the case 10 from [59], which is representative of a transonic flow with a strong shock-boundary layer interaction with a reattachment upstream of the trailing edge. The occurrence of a strong shock on the upper surface of the airfoil develops an important thickening of the boundary layer. Numerical and experimental data are available due to detailed studies carried out in the framework of the European initiative on validation of CFD codes (EUROVAL) project [60]. Further numerical simulations have been performed in the context of RANS equations with finite volume methods in body-fitted meshes and also immersed boundary methods [32, 34, 35, 61, 62]. The geometry of the configuration is depicted in Fig. 15. The computational domain is similar to that used in the NACA0012 case, that is, a box with dimensions $[-20, +20] \times [-20, +20] \times [0, +1]$. The domain is initially

Table 3 Subsonic flow around the NACA0012 airfoil

Mesh	Blanking deactivated				Blanking activated			
	nElements	Time [h]	nIter	Residual [ρ]	nElements	Time [h]	nIter	Residual [ρ]
Coarse	6.53×10^5	6.68	3000	9.32×10^{-6}	7.19×10^5	0.21	100	3.30×10^{-11}
Medium	8.59×10^5	18.42	3000	5.73×10^{-5}	8.54×10^5	0.17	100	1.77×10^{-10}
Fine	1.68×10^6	23.40	3000	3.62×10^{-5}	1.39×10^6	0.34	100	1.39×10^{-9}
Very fine	4.99×10^6	37.80	3000	3.98×10^{-5}	3.55×10^6	0.71	100	1.02×10^{-7}
Body-fitted	9.18×10^5	0.88	4000	9.36×10^{-7}	9.18×10^5	0.88	4000	9.36×10^{-7}

Influence of mesh blanking on simulations performance for angle of attack $\alpha = 0°$

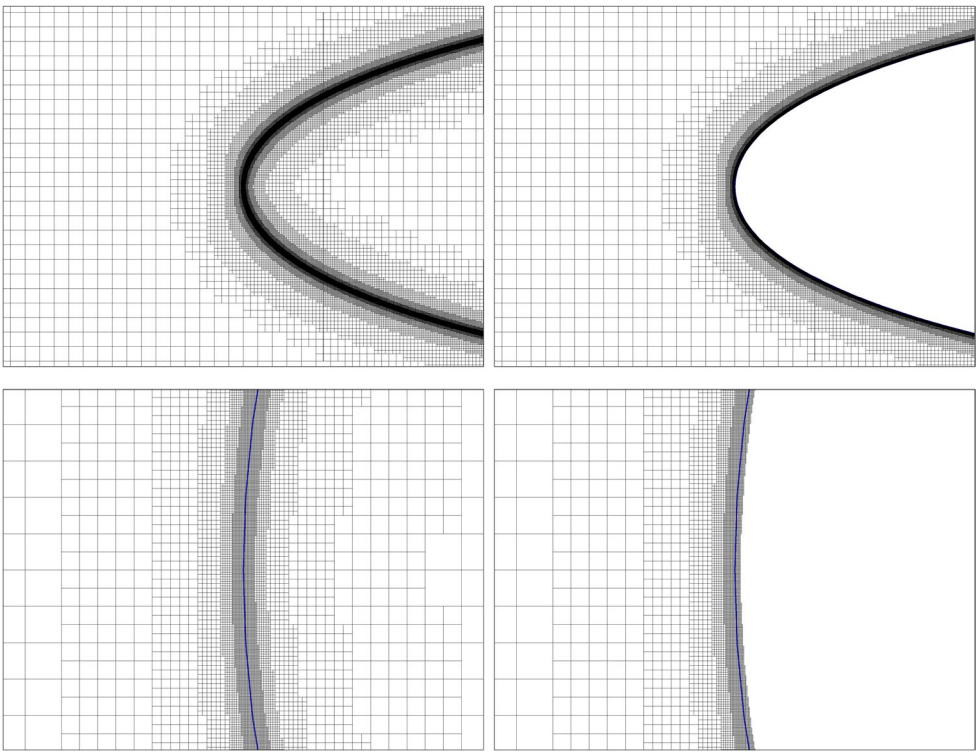

Fig. 14 Mesh refinement around the NACA0012 airfoil without blanking of elements inside the geometry (left) and with blanking of elements inside the geometry (right). The blue lines represent the edges of the STL triangles. In both cases, a high-resolution STL geometry has been employed

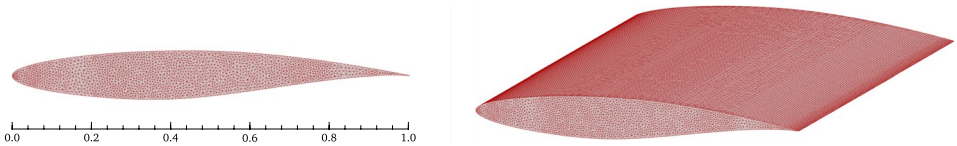

Fig. 15 Geometry used for the numerical simulations of the RAE2822 airfoil

discretized with $n_x n_y n_z = 1600$ hexahedral elements, where $n_x = 40$, $n_y = 40$, and $n_z = 1$. The mesh is then refined around the immersed geometry and in the wake region. The preprocessing tool refines the mesh around the immersed geometry up to the refinement level $l = 16$, and the wake region is resolved with a refined subregion with refinement level $l = 8$. This region is approximately 20 airfoil chords downstream of the lead edge of the airfoil. In Table 4 are shown the characteristics of the meshes used for the mesh convergence study, ranging from a coarse mesh to a very fine mesh. The refinement level $l_{\max}$ listed in the table is the highest level of refinement around the immersed body. The immersed geometry is located in the center of the computational domain, with its cross-section in the plane z, and this corresponds to the RAE2822 profile with chord length $c = 1.0$. The triangulation of the high-resolution surface geometry is made up of 4744 764 triangles. In Fig. 16 the mesh around the immersed geometry is depicted.

Regarding the flow conditions, the gas has adiabatic index $\gamma = 1.4$ and it flows with Reynolds number $\mathrm{Re} = 6.5 \times 10^6$ and freestream Mach number $M = 0.729$. The simulations were performed with the airfoil at an angle of attack $\alpha = 2.31°$. The non-dimensional density is set to $\rho = 1$, and the non-dimensional pressure $p = 1$. On the boundaries of the computational domain the following boundary conditions were set: the left face is set to inflow, the right face to outflow, the front and back faces were set to periodic, and the top and bottom to far field. The inflow pressure assumes the value of the stagnation pressure and the outflow pressure to the static pressure.

Table 4 Characteristics of the meshes for the flow around the RAE2822 airfoil

Name	$l_{\max}$	h_{wake}	h_{body}	*nElems*
Coarse	10	3.91×10^{-3}	9.77×10^{-4}	7.19×10^5
Medium	12	3.91×10^{-3}	2.44×10^{-4}	8.54×10^5
Fine	14	3.91×10^{-3}	6.10×10^{-5}	1.39×10^6
Very fine	16	3.91×10^{-3}	1.53×10^{-5}	3.55×10^6

Contour plots of the Mach number and pressure from the numerical solution on the very fine mesh are shown in Fig. 17 (left and right figures, respectively). In the Fig. 18 is presented the grid convergence in terms of the pressure coefficient C_p on the airfoil surface, with wall model deactivated. The experimental data has been taken from

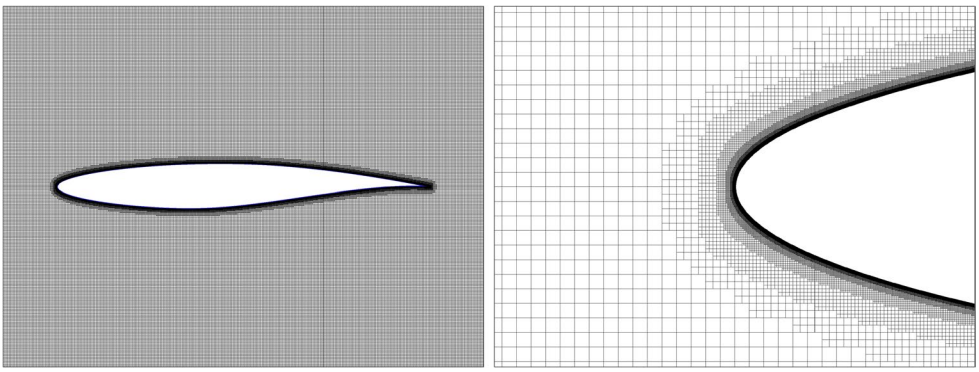

Fig. 16 Mesh refinement around the RAE2822 airfoil. The mesh has been generated with maximum level of refinement $l_{\max} = 16$ and with the blanking activated. Close-up view of the mesh at 50x zoom, depicting the full airfoil (left) and close-up view of the mesh at 500x zoom, depicting the leading edge (right)

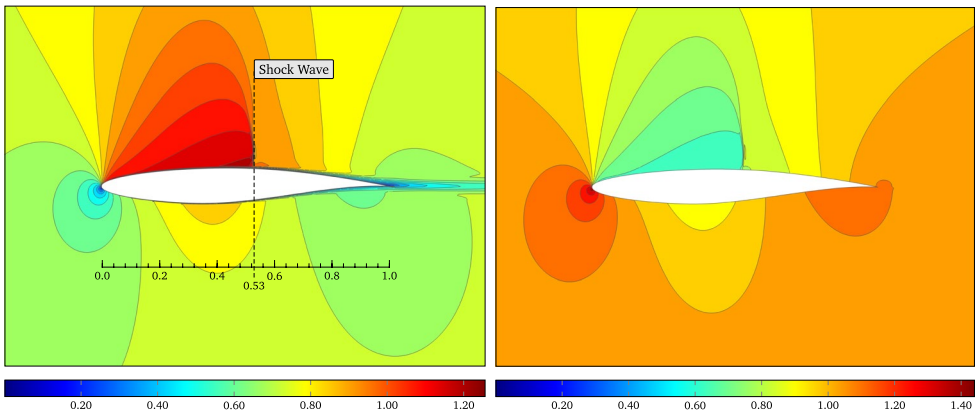

Fig. 17 Transonic flow around the RAE2822 airfoil. Contour plots of the Mach number (left) and pressure (right) for the simulations with angle of attack $\alpha = 2.31°$ and Mach number $M = 0.729$. Computations were performed using the CODA solver based on an IBM Cartesian mesh with maximum level of refinement $l_{\max} = 16$. The RANS equations were solved with the wall model deactivated

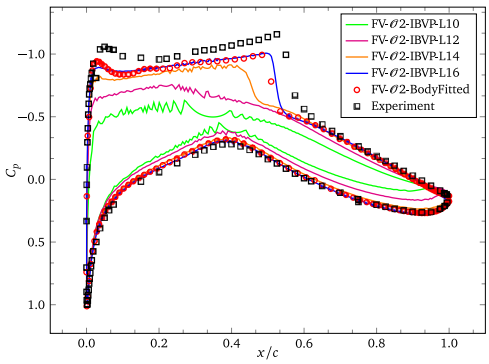

Fig. 18 Transonic flow around the RAE2822 airfoil. Grid convergence study in terms of the pressure coefficient C_p for the transonic flow at angle of attack $\alpha = 2.31°$. Only results with wall model deactivated are shown

[59]. On the upper surface, the results show good agreement for the finer mesh with the body fitted simulation. Let us remind the reader that these results were obtained without including any penalization in the continuity equation, as proposed by other authors [63].

5.3 Subsonic flow around the MDA30P30N multi-element airfoil

The fourth test in the benchmarking is a three-element configuration, specifically the McDonnell Douglas 30P-30N landing configuration (MDA30P30N). Many experimental and computational studies have been performed for the flow past multi-element airfoils in the last decades [64–74]. Accurate prediction of the flow over multi-element airfoils during high-lift operations can improve the performance and the safety factor of aircrafts. The flow around multi-element airfoils is complex, and it is well known to be dominated by different flow mechanisms at different operating conditions, making rather difficult to accurately predict high-lift flow fields. The geometry of the configuration is depicted in Fig. 19. The leading edge slat and the trailing edge flap have a deflection angle of 30°. This airfoil has been extensively tested in the NASA Langley Low Turbulence Pressure Tunnel at various Reynolds and Mach numbers and has also been numerically simulated using a wide range of numerical techniques to solve the Navier–Stokes equations along with different turbulence models.

The computational domain is similar to that used in the NACA0012 case, that is, a box with dimensions $[-20, +20] \times [-20, +20] \times [0, +1]$. The domain is initially discretized with $n_x n_y n_z = 1600$ hexahedral elements, where

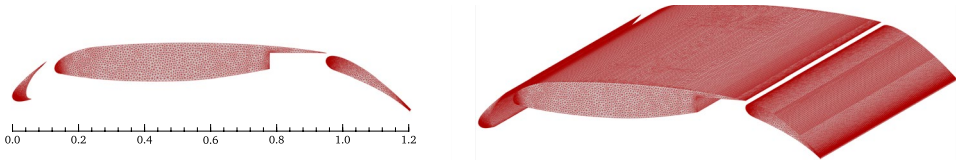

Fig. 19 Geometry used for the numerical simulations of the MDA30P30N multi-element airfoil

Table 5 Characteristics of the meshes for the flow around the MDA30P30N multi-element airfoil

Name	$l_{\max}$	h_{wake}	h_{body}	*nElems*
Coarse	10	3.91×10^{-3}	9.77×10^{-4}	9.64×10^{5}
Medium	12	3.91×10^{-3}	2.44×10^{-4}	1.25×10^{6}
Fine	14	3.91×10^{-3}	6.10×10^{-5}	2.44×10^{6}
Very fine	16	3.91×10^{-3}	1.53×10^{-5}	7.19×10^{6}

$n_x = 40$, $n_y = 40$, and $n_z = 1$. The mesh is then refined around the immersed geometry and in the wake region. The mesh around the immersed geometry is refined to the refinement level $l = 16$, and the wake region is resolved with a refined subregion with refinement level $l = 8$. This region is approximately 20 airfoil chords downstream of the lead edge of the airfoil. In Table 5 are shown the characteristics of the meshes used for the mesh convergence study, ranging from a medium mesh to a very fine mesh. The refinement level $l_{\max}$ listed in the table is the highest level of refinement around the immersed body. The immersed geometry is located in the center of the computational domain, with its cross-section in the plane z, and this corresponds to the MDA30P30N profile with chord length $c = 1.2$. The triangulation of the surface geometry is made up of 6,381,338 triangles. In Fig. 20 the mesh around the immersed geometry is depicted. The inner elements are not shown.

Regarding flow conditions, the gas has an adiabatic index $\gamma = 1.4$ and flows with Reynolds number $\text{Re} = 9 \times 10^6$ and freestream Mach number $M = 0.2$. The simulations were performed with the airfoil at an angle of attack $\alpha = 8°$. The non-dimensional density is set to $\rho = 1$, and the non-dimensional pressure $p = 1$. On the boundaries of the computational domain the following boundary conditions were set: the left face is set to inflow, the right face to outflow, the front and back faces were set to periodic, and the top and bottom to far field. The inflow pressure assumes the value of the stagnation pressure and the outflow pressure to the static pressure.

Contour plots of velocity magnitude from the numerical solution on the very fine mesh are shown in Fig. 21. A small separation region is present on the suction side of the upstream side of the trailing edge of the flap. In Fig. 22 is presented the grid convergence in terms of the pressure coefficient C_p on the multi-element airfoil surface, with wall model deactivated. The experimental data were taken from [75]. In the main part of the airfoil, the pressure coefficient curve agrees well with the experimental data, except at its trailing edge. We can observe a remarkable discrepancy with respect to the experimental data on the upper surface of the slat and the flap for the coarse and medium meshes, but a good agreement is obtained in the simulations with the fine and very fine meshes. We observe that the surface skin friction coefficient is not very well predicted, and presents an oscillatory behavior. The lift and drag values are shown in Table 6 for the angle of attack $\alpha = 8°$. The lift coefficient increases monotonically as the grid is refined, getting closer to the experimental value $C_L = 3.243$ [75]. No experimental data is available for the drag coefficient.

5.4 Subsonic flow around the NASA high-lift CRM

The last test we consider in this work is the NASA high-lift CRM configuration, selected from the Fourth AIAA CFD High Lift Prediction Workshop [76]. The NASA Common Research Model has been used mainly in the Drag Prediction

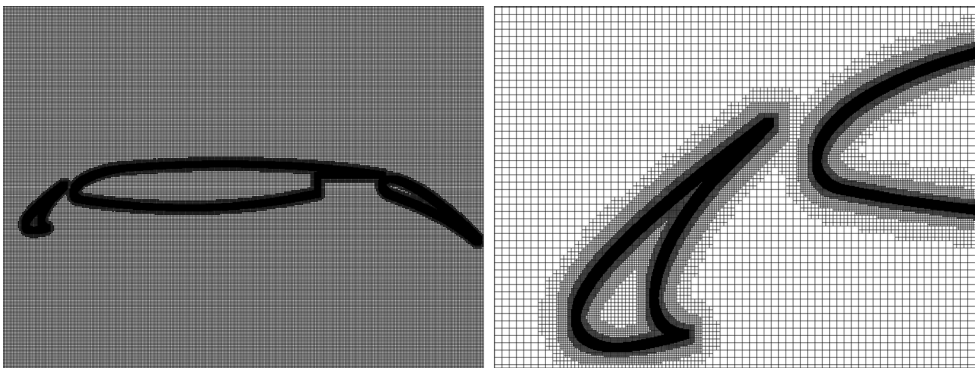

Fig. 20 Mesh refinement around the MDA30P30N airfoil. The mesh has been generated with maximum level of refinement $l_{\max} = 16$ and with the blanking deactivated. Close-up view of the mesh at 50x zoom, depicting the full airfoil (left) and close-up view of the mesh at 500x zoom, depicting the slat and leading edge (right)

Table 6 Subsonic flow around the MDA30P30N multi-element airfoil

Mesh	C_L	C_D
Coarse	2.473	4.490×10^{-2}
Medium	2.743	4.579×10^{-2}
Fine	2.837	4.673×10^{-2}
Very fine	3.017	5.851×10^{-2}
Experimental	3.243	Not available

Lift and drag coefficients for angle of attack $\alpha = 8°$

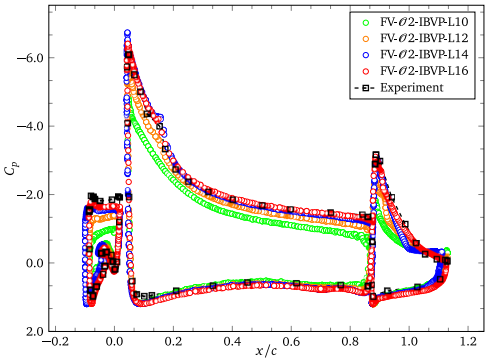

Fig. 22 Subsonic flow around the MDA30P30N multi-element airfoil. Grid convergence study in terms of the pressure coefficient C_p for the subsonic flow at angle of attack $\alpha = 8°$. Only results with wall model deactivated are shown

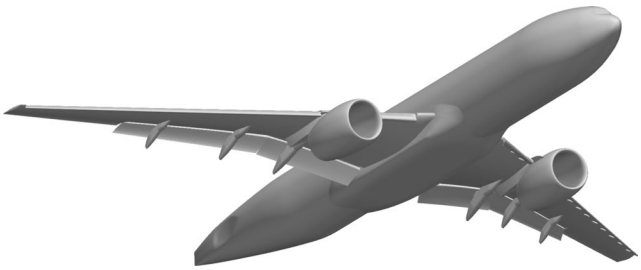

Fig. 23 Geometry used for the numerical simulations of the NASA high-lift CRM Wing-Body-Flap-Slat-Nacelle-Pylon

Workshop (DPW) and the High Lift Prediction Workshop (HLPW) to assess the accuracy of numerical methods in the prediction of aircraft forces and moments. The database of these workshops allows us to assess the robustness and effectiveness of state-of-the-art numerical programs and turbulence modeling techniques using Navier–Stokes solvers.

The CRM configuration was originally designed by Boeing and was further manufactured and tested by NASA [77]. The NASA CRM consists of a contemporary supercritical transonic wing and a fuselage that is representative of a wide-body commercial transport aircraft. Several CRM experiments have been carried out in several research facilities [78–80].

The computational domain used for this simulation is a box with dimensions $[-20, +20] \times [-20, +20] \times [0, +20]$. The domain is initially discretized with $n_x n_y n_z = 32{,}000$ hexahedral elements, where $n_x = 40$, $n_y = 40$, and $n_z = 20$. The mesh is then refined around the immersed geometry and in the wake region. The mesh around the immersed geometry is refined to the refinement level $l = 10$, and the wake region is resolved with a refined subregion with refinement level $l = 8$. In Table 7 are shown the characteristics of the meshes used for the mesh convergence study, ranging from a very coarse mesh to a medium mesh. The refinement level $l_{\max}$ listed in the table is the highest level of refinement around the immersed geometry. The geometry of the configuration is depicted in Fig. 23. The immersed geometry is located in the center of the computational domain. The triangulation of the surface geometry is made up of 74 331 triangles. This low-resolution tessellation will have a negative impact on the final numerical solution, as it could be observed in the computations of NACA0012 airfoil. In Fig. 24 is depicted the mesh around the immersed geometry.

The Reynolds number based on the mean aerodynamic chord (MAC = 7.005 m) is Re = 5.6 million, and the angle of attack is $\alpha = 8°$ with an incoming Mach number of $M_\infty = 0.2$. On the boundaries of the computational domain

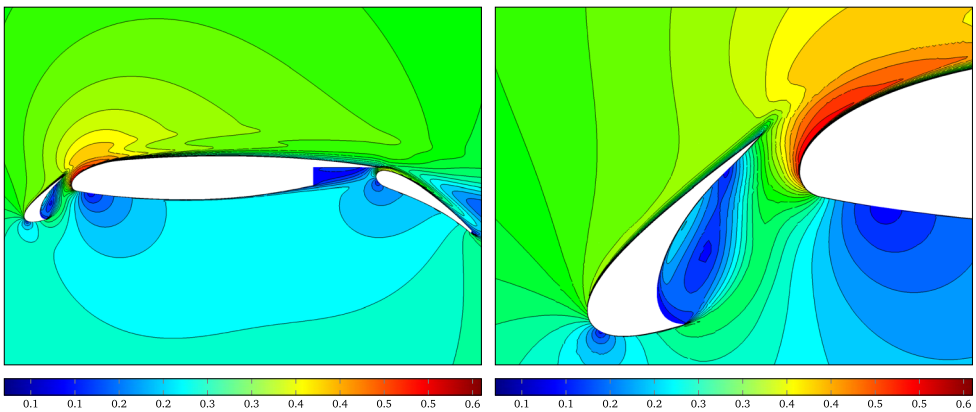

Fig. 21 Subsonic flow around the MDA30P30N multi-element airfoil. Contour plots of the velocity magnitude for the simulations with angle of attack $\alpha = 8°$. Computations were performed using the CODA solver based on an IBM Cartesian mesh with a level of refinement around the geometry $l_{\max} = 16$. The RANS equations were solved with the wall model deactivated

Table 7 Characteristics of the meshes for the flow around the NASA high-lift CRM Wing-Body-Flap-Slat-Nacelle-Pylon geometry

Name	$l_{\max}$	h_{wake}	h_{body}	*nElems*
Very coarse	8	3.91×10^{-3}	3.91×10^{-3}	5.827×10^{6}
Coarse	9	3.91×10^{-3}	1.95×10^{-3}	23.207×10^{6}
Medium	10	3.91×10^{-3}	9.77×10^{-4}	96.520×10^{6}

the following boundary conditions were set: the left face is set to inflow, the right face to outflow, the front and back faces were set to periodic, and the top and bottom to far field. The inflow pressure assumes the value of the stagnation pressure and the outflow pressure to the static pressure.

Contour plots of the velocity magnitude at different positions are shown in Fig. 25. The curves are qualitatively similar to those obtained in body-fitted simulations we have previously done with the solver CODA (not shown in this work). We can clearly observe flow separation on the surface of the fuselage (left) and on the suction side of the upstream side of the trailing edge of the flap.

Some final comments are necessary to mention at this point. In the computations of NACA0012, RAE2822 and MDA30P30N multi-element airfoils, we found that, in order to accurately capture the aerodynamics and obtain very good agreement with reference body-fitted simulations and experimental data, high-resolution surface representation of immersed geometries and very fine meshes around these are required. In a good IBM-Cartesian mesh, the element size close to the geometry is around $h_{\text{body}} = 1.53 \times 10^{-5}$ ($y^{+} \approx 100$), or even smaller. For the case of the NASA high-lift CRM, a mesh with those characteristics would have a very high cell number, typically over several billions. Cartesian meshes with up to 300 million elements have been generated in a fast manner with our preprocessing tool in 4 hours on a single core machine, but its generation requires around 2 terabytes of RAM. Besides, the simulations in CODA will need thousands of cores and several terabytes of RAM if such meshes are employed. With our limited computational resources, we were able only to simulate the NASA high-lift CRM on a medium-size mesh with 97 million cells and with element size close to the geometry $h_{\text{body}} = 9.77 \times 10^{-4}$. Taking into account all these previous considerations, in our computations of the NASA high-lift CRM on the medium-size mesh, this lack of resolution around the immersed body had as a consequence poor quality in the integral quantities like lift and drag coefficients, and the pressure and skin friction curves, this means that they are not in good agreement with experimental data, but this issue can be solved by using finer meshes. This is consistent with the results shown before for the MDA30P30N airfoil. We hope that all previous sections have already proved the validity of the implementation.

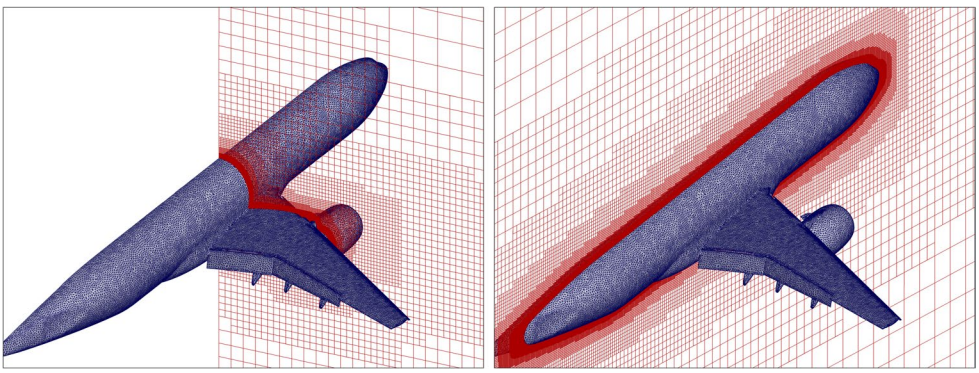

Fig. 24 Mesh refinement around the NASA high-lift CRM Wing-Body-Flap-Slat-Nacelle-Pylon geometry

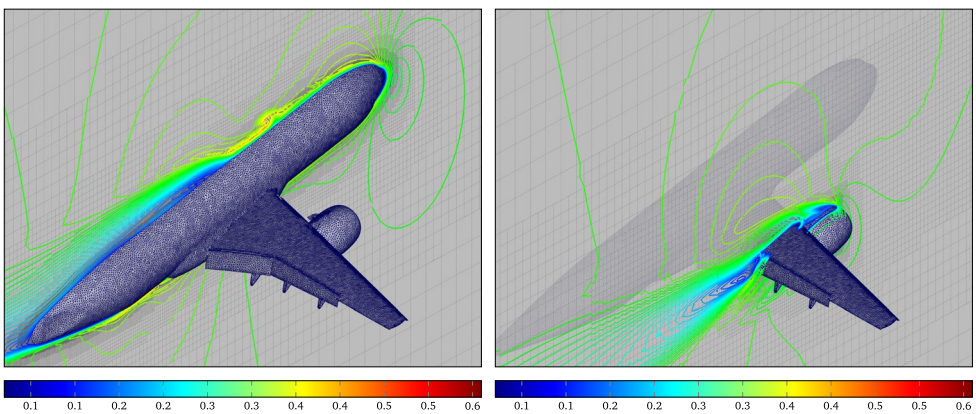

Fig. 25 Subsonic flow around the NASA high-lift CRM. Contour plots of the velocity magnitude at different positions for the simulations with angle of attack $\alpha = 8°$. Computations were performed using the CODA solver based on an IBM Cartesian mesh with a level of refinement around the geometry $l_{\max} = 10$. The RANS equations were solved with the wall model deactivated

This last section aims only to show qualitative results (not quantitative) to highlight the potential of the methodology for more general complex cases.

6 Conclusions

The immersed boundary volume penalization method for the Navier–Stokes and the RANS equations has been implemented in the CFD solver CODA. The immersed boundary method is compatible with the finite volume, modal discontinuous Galerkin and discontinuous Galerkin spectral element methods and with the explicit and implicit temporal schemes available in CODA. An efficient preprocessing tool for the construction of unstructured hexahedral meshes with adaptive mesh refinement around immersed geometries has been developed. The octree meshes generated with the tool are imported in CODA and used for simulating complex aerodynamics flow problems. Immersed boundary methods require a very simple meshing process: only the box dimension, initial number of elements of the Cartesian mesh, maximum level of refinement, and the geometry of the body are required as input. The automatic mesh generation process is able to automatically refine the mesh near the body. Besides, the immersed boundary technology can be used straightforward with an external predefined mesh, without any additional modification.

An important aspect regarding the volume penalization methodology is that the use of source terms increases the stiffness of the partial differential equations, and therefore, more iterations are required until convergence. This represents a fundamental disadvantage with respect to CFD solvers based on body-fitted meshes and also other flavors of immersed boundary methods. Mesh blanking in IBM Cartesian meshes aims to reduce the computation time when similar results are sought among CFD solvers based on IBM techniques and body-fitted meshes. We have explored the mesh blanking to assess the performance of the CODA solver with IBM. Computations done using blanked meshes have a superior efficiency compared to simulations using meshes without blanking. We have found that simulations using blanked meshes need up to two orders of magnitude fewer linear iterations than simulations with no blanked meshes in order to converge to a solution with similar residual in the density in computations performed with body-fitted meshes.

Several numerical computations were performed and discussed: subsonic flow past the NACA0012 airfoil, transonic flow past the RAE2822 airfoil, subsonic flow past the MDA30P30N multi-element airfoil and subsonic flow around the NASA high-lift CRM aircraft. The numerical computations for the airfoils are in good agreement with their corresponding experimental data for the pressure coefficient curve. The skin friction curves have an oscillatory behavior for low resolution meshes, but these oscillations tend to decrease once the geometry and mesh are refined. Drag and lift coefficients have values very close to the reference data, but our findings allow us to conclude that even finer meshes are required if an excellent agreement is sought. Computations with deactivation and activation of the wall model are quite similar, but computations are more robust with deactivated wall model. More research is necessary to improve the robustness of the method.

Even though the immersed boundary methodology works within the CFD solver CODA with very good results, a critical disadvantage of these techniques with respect to a body-fitted approach is that the mesh refinement around the immersed geometry leads to a very high number of elements when a high resolution of the boundary layer is desired. Anisotropic mesh refinement reduce the number of elements in comparison with isotropic mesh refinement, but the mesh size is still prohibitive.

As a consequence of these considerations, we will explore also alternative approaches to automatically generate meshes for IBM simulations. In particular, future work will focus on hybrid IBM approaches, in which only geometrical details (control surfaces, ice shapes, etc) will be immersed in a body-fitted, wall-resolved mesh of a clean aerodynamic surface. This approach would benefit from IBM advantages (several configurations of the geometrical details can be tested on the same body-fitted background mesh) without handling meshes of very big sizes.

Acknowledgements We thank the anonymous reviewers for their valuable and critical remarks that helped to improve the quality of the manuscript. Jonatan Núñez, David Huergo, Esteban Ferrer and Eusebio Valero acknowledge the funding received by the Grant NextSim/AEI/10.13039/501100011033 and H2020, GA-956104. This project has received funding from the Clean Aviation Joint Undertaking under the European Union's Horizon Europe research and innovation programme under Grant Agreement HERA (Hybrid-Electric Regional Architecture) No. 101102007. Views and opinions expressed are, however, those of the author(s) only and do not necessarily reflect those of the European Union or CAJU. Neither the European Union nor the granting authority can be held responsible for them. Esteban Ferrer and Eusebio Valero acknowledge the funding received by the Grant DeepCFD (Project No. PID2022-137899OB-I00) funded by MCIN/AEI/10.13039/501100011033 and by ERDF A way of making Europe. Esteban Ferrer would like to thank the support of the Comunidad de Madrid and Universidad Politécnica de Madrid for the Young Investigators award: APOYO-JOVENES-21-53NYUB-19-RRX1A0. Finally, all authors gratefully acknowledge the Universidad Politécnica de Madrid (www.upm.es) for providing computing resources on the Magerit Supercomputer. The authors also thankfully acknowledge the computer resources at MareNostrum and the technical support provided by the Barcelona Supercomputing Center (RES-IM-2022-3-0023).

Author contributions Jonatan Núñez: Conceptualization, Methodology, Software, Formal analysis and investigation, Writing. David Huergo: Conceptualization, Methodology, Formal analysis and investigation, Software. Diego Lodares: Conceptualization, Methodology, Formal analysis and investigation, Software. Suyash Shrestha:

Methodology, Software. Juan Guerra: Conceptualization, Methodology, Funding acquisition. Juan Florenciano: Conceptualization, Methodology, Funding acquisition. Esteban Ferrer: Conceptualization, Methodology, Funding acquisition, Supervision. Eusebio Valero: Conceptualization, Methodology, Funding acquisition, Supervision.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature. Not applicable.

Data availability Not applicable.

Code availability Not applicable.

Materials availability Not applicable

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Consent for publication Not applicable.

Ethics approval and consent to participate Not applicable.

References

1. Volpiani P, Chapelier J-B, Schwöppe A, Jägersküpper J, Champagneux S (2024) Aircraft simulations using the new CFD software from ONERA, DLR, and Airbus. J Aircr. https://doi.org/10.2514/1.C037506
2. Kroll N, Abu-Zurayk M, Dimitrov D, Franz T, Führer T, Gerhold T, Görtz S, Heinrich R, Ilic C, Jepsen J, Jägersküpper J, Kruse M, Krumbein A, Langer S, Liu D, Liepelt R, Reimer L, Ritter M, Schwöppe A, Scherer J, Spiering F, Thormann R, Togiti V, Vollmer D, Wendisch J-H (2016) DLR project Digital-X: towards virtual aircraft design and flight testing based on high-fidelity methods. CEAS Aeronaut J 7:3–27. https://doi.org/10.1007/s13272-015-0179-7
3. Jägersküpper J, Vollmer D (2022) On highly scalable 2-level-parallel unstructured CFD. In: Proceedings of the 8th European congress on computational methods in applied sciences and engineering. https://doi.org/10.23967/eccomas.2022.208
4. Peskin C (1972) Flow patterns around heart valves: a numerical method. J Comput Phys 10:252–271. https://doi.org/10.1016/0021-9991(72)90065-4
5. Iaccarino G, Verzicco R (2003) Immersed boundary technique for turbulent flow simulations. Appl Mech Rev 56(3):331–347. https://doi.org/10.1115/1.1563627
6. Shao J-Y, Shu C, Chew Y-T (2013) Development of an immersed boundary-phase field-lattice Boltzmann method for Neumann boundary condition to study contact line dynamics. J Comput Phys 234:8–32. https://doi.org/10.1016/j.jcp.2012.08.040
7. Wang Y, Shu C, Teo C-J, Wu J (2015) An immersed boundary-lattice Boltzmann flux solver and its applications to fluid–structure interaction problems. J Fluids Struct 54:440–465. https://doi.org/10.1016/j.jfluidstructs.2014.12.003
8. Ye T, Mittal R, Udaykumar H, Shyy W (1999) An accurate cartesian grid method for viscous incompressible flows with complex immersed boundaries. J Comput Phys 156:209–240. https://doi.org/10.1006/jcph.1999.6356
9. Udaykumar H, Mittal R, Rampunggoon P, Khanna A (2001) A sharp interface cartesian grid method for simulating flows with complex moving boundaries. J Comput Phys 174(1):345–380. https://doi.org/10.1006/jcph.2001.6916
10. Örley F, Pasquariello V, Hickel S, Adams N (2015) Cut-element based immersed boundary method for moving geometries in compressible liquid flows with cavitation. J Comput Phys 283:1–22. https://doi.org/10.1016/j.jcp.2014.11.028
11. Tseng Y-H, Ferziger J (2003) A ghost-cell immersed boundary method for flow in complex geometry. J Comput Phys 192(2):593–623. https://doi.org/10.1016/j.jcp.2003.07.024
12. Hu O, Zhao N, Liu JM (2013) A ghost cell method for turbulent compressible viscous flows on adaptive cartesian grids. Procedia Eng 67:241–249. https://doi.org/10.1016/j.proeng.2013.12.023
13. Xia J, Luo K, Fan J (2014) A ghost-cell based high-order immersed boundary method for inter-phase heat transfer simulation. Int J Heat Mass Transf 75:302–312. https://doi.org/10.1016/j.ijheatmasstransfer.2014.03.048
14. Khalili ME, Larsson M, Müller B (2018) High-order ghost-point immersed boundary method for viscous compressible flows based on summation-by-parts operators. Int J Numer Methods Fluids 89(7):256–282. https://doi.org/10.1002/fld.4696
15. Goldstein D, Handler R, Sirovich L (1993) Modeling a no-slip flow boundary with an external force field. J Comput Phys 105(2):354–366. https://doi.org/10.1006/jcph.1993.1081
16. Fadlun EA, Verzicco R, Orlandi P, Mohd-Yusof J (2000) Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations. J Comput Phys 161:35–60. https://doi.org/10.1006/jcph.2000.6484
17. Luo H, Dai H, Ferreira de Sousa P, Yin B (2012) On the numerical oscillation of the direct-forcing immersed-boundary method for moving boundaries. Comput Fluids 56:61–76. https://doi.org/10.1016/j.compfluid.2011.11.015
18. Boukharfane R, Ribeiro F, Bouali Z, Mura A (2018) A combined ghost-point-forcing / direct-forcing immersed boundary method (IBM) for compressible flow simulations. Comput Fluids 162:91–112. https://doi.org/10.1016/j.compfluid.2017.11.018
19. Ghias R, Mittal R, Dong H (2007) A sharp interface immersed boundary method for compressible viscous flows. J Comput Phys 225:528–553. https://doi.org/10.1016/j.jcp.2006.12.007
20. De Vanna F, Picano F, Benini E (2020) A sharp-interface immersed boundary method for moving objects in compressible viscous flows. Comput Fluids 201:104415. https://doi.org/10.1016/j.compfluid.2019.104415
21. Angot P, Bruneau C-H, Fabrie P (1999) A penalization method to take into account obstacles in incompressible viscous flows. Numer Math 81:497–520. https://doi.org/10.1007/s002110050401

22. Liu Q, Vasilyev O (2007) A brinkman penalization method for compressible flows in complex geometries. J Comput Phys 227(2):946–966. https://doi.org/10.1016/j.jcp.2007.07.037
23. Schneider K (2015) Immersed boundary methods for numerical simulation of confined fluid and plasma turbulence in complex geometries: a review. J Plasma Phys 81(6):435810601. https://doi.org/10.1017/S0022377815000598
24. Engels T, Kolomenskiy D, Schneider K, Sesterhenn J (2015) Numerical simulation of fluid-structure interaction with the volume penalization method. J Comput Phys 281:96–115. https://doi.org/10.1016/j.jcp.2014.10.005
25. Abalakin I, Zhdanova N, Kozubskaya T (2016) Immersed boundary method implemented for the simulation of an external flow on unstructured meshes. Math Models Comput Simul 8(3):219–230. https://doi.org/10.1134/S2070048216030029
26. Kou J, Joshi S, Hurtado-de-Mendoza A, Puri K, Hirsch C, Ferrer E (2022) Immersed boundary method for high-order flux reconstruction based on volume penalization. J Comput Phys 448:110721. https://doi.org/10.1016/j.jcp.2021.110721
27. Mittal R, Iaccarino G (2005) Immersed boundary methods. Annu Rev Fluid Mech 37:239–261. https://doi.org/10.1146/annurev.fluid.37.061903.175743
28. Sotiropoulos F, Yang X (2014) Immersed boundary methods for simulating fluid–structure interaction. Prog Aerosp Sci 65:1–21. https://doi.org/10.1016/j.paerosci.2013.09.003
29. Griffith B, Patankar N (2020) Immersed methods for fluid–structure interaction. Annu Rev Fluid Mech 52:421–448. https://doi.org/10.1146/annurev-fluid-010719-060228
30. Verzicco R (2023) Immersed boundary methods: historical perspective and future outlook. Annu Rev Fluid Mech 55:129–155. https://doi.org/10.1146/annurev-fluid-120720-022129
31. Capizzano F (2007) A compressible flow simulation system based on Cartesian grids with anisotropic refinements. In: Proceedings of the 45th AIAA Aerospace Sciences Meeting and Exhibit. American Institute of Aeronautics and Astronautics. https://doi.org/10.2514/6.2007-1450
32. Capizzano F (2011) Turbulent wall model for immersed boundary methods. AIAA J 49(11):2367–2381. https://doi.org/10.2514/1.J050466
33. Capizzano F (2018) Automatic generation of locally refined cartesian meshes: data management and algorithms. Int J Numer Methods Eng 113:789–813. https://doi.org/10.1002/nme.5636
34. Constant B, Péron S, Beaugendre H, Benoit C (2021) An improved immersed boundary method for turbulent flow simulations on Cartesian grids. J Comput Phys 435:110240. https://doi.org/10.1016/j.jcp.2021.110240
35. Capizzano F (2016) Coupling a wall diffusion model with an immersed boundary technique. AIAA J 54(2):728–734. https://doi.org/10.2514/1.J054197
36. Péron S, Renaud T, Mary I, Benoit C, Terracol M (2017) An immersed boundary method for preliminary design aerodynamic studies of complex configurations. In: Proceedings of the 23rd AIAA computational fluid dynamics conference. American Institute of Aeronautics and Astronautics. https://doi.org/10.2514/6.2017-3623
37. Renaud T, Benoit C, Peron S, Mary I, Alferez N (2019) Validation of an immersed boundary method for compressible flows. In: Proceedings of the AIAA Scitech 2019 Forum. https://doi.org/10.2514/6.2019-2179
38. Péron S, Benoit C, Renaud T, Mary I (2021) An immersed boundary method on Cartesian adaptive grids for the simulation of compressible flows around arbitrary geometries. Eng Comput 37(3):2419–2437. https://doi.org/10.1007/s00366-020-00950-y
39. Blazek J (2015) Computational fluid dynamics: principles and applications, 3rd edn. Elsevier Science, Amsterdam
40. Hesthaven J, Warburton T (2008) Nodal discontinuous Galerkin methods: algorithms, analysis. and applications. Springer, Heidelberg
41. Kopriva D (2009) Implementing spectral methods for partial differential equations. Springer, Berlin
42. Giraldo F (2020) An introduction to element-based Galerkin methods on tensor-product bases. Springer, Berlin
43. Basile F, Chapelier J-B, de la Llave Plata M, Laraufie R, Frey P (2022) Unstructured *h*- and *hp*-adaptive strategies for discontinuous Galerkin methods based on a posteriori error estimation for compressible flows. Comput Fluids 233:105245. https://doi.org/10.1016/j.compfluid.2021.105245
44. Schwöppe A, Diskin B (2013) In: Dillmann A, Heller G, Kreplin H-P, Nitsche W, Peltzer I (eds) Accuracy of the cell-centered grid metric in the DLR TAU-code. Springer, Berlin, pp 429–437. https://doi.org/10.1007/978-3-642-35680-3_51
45. Langer S, Schwöppe A, Leicht T (2024) In: Dillmann A, Heller G, Krämer E, Wagner C, Weiss J (eds) Comparison and unification of finite-volume discretization strategies for the unstructured node-centered and cell-centered grid metric in TAU and CODA. Springer, Cham, pp 262–272. https://doi.org/10.1007/978-3-031-40482-5_25
46. Mohnke J, Wagner M (2023) In: Cano J, Dikaiakos M, Papadopoulos G, Pericàs M, Sakellariou R (eds) A look at performance and scalability of the GPU accelerated sparse linear system solver Spliss. Springer, Cham, pp 637–648. https://doi.org/10.1007/978-3-031-39698-4_43
47. Kou J, Hurtado-de-Mendoza A, Joshi S, Le Clainche S, Ferrer E (2022) Eigensolution analysis of immersed boundary method based on volume penalization: applications to high-order schemes. J Comput Phys 449:110817. https://doi.org/10.1016/j.jcp.2021.110817
48. Llorente V, Kou J, Valero E, Ferrer E (2023) A modified equation analysis for immersed boundary methods based on volume penalization: applications to linear advection-diffusion and high-order discontinuous Galerkin schemes. Comput Fluids 257:105869. https://doi.org/10.1016/j.compfluid.2023.105869
49. Llorente V (2024) Theoretical considerations of the volume penalization immersed boundary method for turbulent flows. Phys Fluids 36(7):071702. https://doi.org/10.1063/5.0213290
50. Ferrer E, Rubio G, Ntoukas G, Laskowski W, Mariño O, Colombo S, Mateo-Gabín A, Marbona H, Lara F, Huergo D, Manzanero J, Rueda-Ramírez A, Kopriva D, Valero E (2023) HORSES3D: a high order discontinuous Galerkin solver for flow simulations and multiphysics applications. Comput Phys Commun 287:108700. https://doi.org/10.1016/j.cpc.2023.108700
51. Tamaki Y, Harada M, Imamura T (2017) Near-wall modification of Spalart–Allmaras turbulence model for immersed boundary method. AIAA J 55(9):3027–3039. https://doi.org/10.2514/1.J055824
52. Frère A, Wiart C, Hillewaert K, Chatelain P, Winckelmans G (2017) Application of wall-models to discontinuous Galerkin LES. Phys Fluids 29(8):085111. https://doi.org/10.1063/1.4998977
53. Geuzaine C, Remacle J-F (2009) Gmsh: a 3-D finite element mesh generator with built-in pre- and post-processing facilities. Int J Numer Methods Eng 79(11):1309–1331. https://doi.org/10.1002/nme.2579
54. Akenine-Möller T (2001) Fast 3D triangle-box overlap testing. J Graph Tools 6(1):29–33. https://doi.org/10.1080/10867651.2001.10487535
55. Ladson C, Hill A, Johnson W (1988) Pressure distributions from high Reynolds number transonic tests of an NACA 0012 airfoil in the Langley 0.3-meter transonic cryogenic tunnel. Techreport NASA TM-100526, NASA Langley Research Center, Hampton, Virginia

56. Ladson C, Hill A (1988) High Reynolds number transonic tests on an NACA 0012 airfoil in the Langley 0.3-meter transonic cryogenic tunnel. Techreport NASA TM-100527, NASA Langley Research Center, Hampton, Virginia
57. Ladson C (1988) Effects of independent variation of Mach and Reynolds numbers on the low-speed aerodynamic characteristics of the NACA 0012 airfoil section. Techreport NASA TM-4074, NASA Langley Research Center, Hampton, Virginia
58. Goza A, Liska S, Morley B, Colonius T (2016) Accurate computation of surface stresses and forces with immersed boundary methods. J Comput Phys 321:860–873. https://doi.org/10.1016/j.jcp.2016.06.014
59. Cook P, McDonald M, Firmin M (1979) Aerofoil RAE 2822—pressure distributions, and boundary layer and wake measurements. In: AGARD Advisory Report No. 138: experimental data base for computer program assessment. Advisory Group for Aerospace Research and Development
60. Haase W, Brandsma F, Elsholz E, Leschziner M, Schwamborn D (eds) (1993) EUROVAL—an European Initiative on Validation of CFD codes: results of the EC/BRITE-EURAM Project EUROVAL, 1990–1992. Notes on numerical fluid mechanics and multidisciplinary design, vol 42. Vieweg+Teubner Verlag, Wiesbaden. https://doi.org/10.1007/978-3-663-14131-0
61. Delanaye M, Aftosmis M, Berger M, Liu Y, Pulliam T (1999) Automatic hybrid-cartesian grid generation for high-Reynolds number flows around complex geometries. In: Proceedings of the 37th AIAA Aerospace Sciences Meeting and Exhibit. https://doi.org/10.2514/6.1999-777
62. Catalano P, Amato M (2003) An evaluation of RANS turbulence modelling for aerodynamic applications. Aerosp Sci Technol 7:493–509. https://doi.org/10.1016/S1270-9638(03)00061-0
63. Ménez L, Parnaudeau P, Béringhier M, Goncalves da Silva E (2023) Assessment of volume penalization and immersed boundary methods for compressible flows with various thermal boundary conditions. J Comput Phys 493:112465. https://doi.org/10.1016/j.jcp.2023.112465
64. Valarezo W, Dominik C, McGhee R, Goodman W, Paschal K (1991) Multi-element airfoil optimization for maximum lift at high Reynolds numbers. In: Proceedings of the 9th AIAA applied aerodynamics conference. https://doi.org/10.2514/6.1991-3332
65. Valarezo W (1992) High-lift testing at high Reynolds numbers. In: Proceedings of the 17th AIAA aerospace ground testing conference. https://doi.org/10.2514/6.1992-3986
66. Chin V, Peters D, Spaid F, McGhee R (1993) Flowfield measurements about a multi-element airfoil at high Reynolds numbers. In: Proceedings of the 23rd AIAA fluid dynamics, plasma dynamics, and lasers conference. https://doi.org/10.2514/6.1993-3137
67. Rogers S (1994) Progress in high-lift aerodynamic calculations. J Aircr 31(6):1244–1251. https://doi.org/10.2514/3.46642
68. Rogers S, Menter F, Durbin P, Mansour N (1994) A comparison of turbulence models in computing multi-element airfoil flows. In: Proceedings of the 32nd AIAA aerospace sciences meeting and exhibit. https://doi.org/10.2514/6.1994-291
69. Anderson W, Bonhaus D, McGhee R, Walker B (1995) Navier–Stokes computations and experimental comparisons for multielement airfoil configurations. J Aircr 32(6):1246–1253. https://doi.org/10.2514/3.46871
70. Klausmeyer S, Lin J (1997) Comparative results from a CFD challenge over a 2D three-element high-lift airfoil. Techreport NASA TM-112858, NASA Langley Research Center, Hampton, Virginia
71. Bertelrud A (1998) Transition on a three-element high lift configuration at high Reynolds numbers. In: Proceedings of the 36th AIAA aerospace sciences meeting and exhibit. https://doi.org/10.2514/6.1998-703
72. Rumsey C, Gatski T, Ying S, Bertelrud A (1998) Prediction of high-lift flows using turbulent closure models. AIAA J 36(5):765–774. https://doi.org/10.2514/2.435
73. Liou W, Liu F (1999) Computational modeling for the flow over a multi-element airfoil. In: Proceedings of the 17th AIAA applied aerodynamics conference. https://doi.org/10.2514/6.1999-3177
74. Spaid F (2000) High Reynolds number, multielement airfoil flowfield measurements. J Aircr 37(3):499–507. https://doi.org/10.2514/2.2626
75. Murayama M, Nakakita K, Yamamoto K, Ura H, Ito Y, Choudhari M (2014) Experimental study on slat noise from 30P30N three-element high-lift airfoil at JAXA Hard-Wall Lowspeed Wind Tunnel. In: Proceedings of the 20th AIAA/CEAS Aeroacoustics conference. https://doi.org/10.2514/6.2014-2080
76. Ashton N, Batten P, Cary A, Holst K (2024) Summary of the 4th high-lift prediction workshop hybrid RANS/LES Technology Focus Group. J Aircr 61(1):86–115. https://doi.org/10.2514/1.C037329
77. Vassberg J, Dehaan M, Rivers M, Wahls R (2008) Development of a common research model for applied CFD validation studies. In: Proceedings of the 26th AIAA applied aerodynamics conference. https://doi.org/10.2514/6.2008-6919
78. Rivers M, Dittberner A (2014) Experimental investigations of the NASA common research model. J Aircr 51(4):1183–1193. https://doi.org/10.2514/1.C032626
79. Ueno M, Kohzai M, Koga S, Kato H, Nakakita K, Sudani N, Nakamura Y (2015) Normalization of wind-tunnel data for NASA common research model. J Aircr 52(5):1535–1549. https://doi.org/10.2514/1.C032989
80. Cartieri A, Hue D, Chanzy Q, Atinault O (2018) Experimental investigations on common research model at ONERA-S1MA-Drag prediction workshop numerical results. J Aircr 55(4):1491–1508. https://doi.org/10.2514/1.C034414