Comparison of several interpolation methods to reconstruct field data in the vicinity of a finite element immersed boundary

Georis Billo, Michel Belliard, Pierre Sagaut

Comparison of several interpolation methods to reconstruct field data in the vicinity of a finite element immersed boundary

Georis Billo [a,*], Michel Belliard [a], Pierre Sagaut [b]

[a] French Alternative Energies and Atomic Energy Commission (CEA), DES, IRESNE, DER, SESI, LEMS, Cadarache, F-13118 Saint-Paul-Lez-Durance, France
[b] Aix-Marseille University, CNRS, Centrale Marseille, M2P2, Marseille, France

ARTICLE INFO

Keywords:
Navier-Stokes equations
Finite element method
Immersed boundary method
Data reconstruction
Directional and multi-directional interpolation

ABSTRACT

Thermal-hydraulics safety requirements for the second and third generation of nuclear reactors led to the development of innovative passive safety systems. In particular, new devices must be developed involving numerical simulations for turbulent two-phase flows around complex geometries. To reduce the time-consuming mesh generation phase when testing various geometries, we use a fictitious domain approach. More specifically, we choose the Penalized Direct Forcing Method to take into account inflow obstacles. Following a recent work, involving the resolution of the one-phase incompressible Navier-Stokes equations using a projection scheme and the Finite Element Method, this paper focuses on different techniques to recover data from the discrete immersed boundary and different ways to achieve order 2 in space via linear interpolation. Indeed, we investigate two data reconstruction approaches (one based on various weighted averaging, the other based on optimization) and compare their results for cylindrical and NACA0012 airfoil shapes: they provide similar accuracy but the weighting is much faster in terms of execution time. We also investigate three different interpolation types: unidirectional, multi-directional and a new hybrid between the two. The Taylor-Couette flow and the flow around a circular cylinder are used to carry out mesh convergence studies. Globally, order 2 in space is numerically assessed in both $\mathcal{L}^2$ and $\mathcal{L}^\infty$ norms for all the interpolation types, which is consistent with theoretical expectations – even if the space convergence order is a bit higher for the multi-directional approach. For the flow around a circular cylinder, the values of aerodynamic coefficients and Strouhal number are in good agreement with the literature, especially when using directional interpolation. Finally, an industrial case, representative of passive safety systems, is presented to assess the robustness and capability of the method. The simulations tend to show that, here again, the directional interpolation offers the best behavior when dealing with complex geometries and relatively coarse meshes.

1. Introduction

In engineering, Eulerian grid based numerical simulations are privileged tools to study fluid flows past or around complex geometries in order to optimize and assess new designs. In the case of moving, or deformable, inflow obstacles, the computation domain needs to be remeshed when the position, or shape, of the obstacle changes, which may lead to costly and time-consuming simulations. To overcome this issue, one can use the Immersed Boundary Method (IBM), introduced by C.S. PESKIN [1], in which the geometry of the obstacle is independent from the computational domain mesh – for a more extensive history of fictitious domain methods, the interested reader can refer to [2]. In this paper, we are interested in the Penalized Direct Forcing (PDF) [3,4,2]. The PDF inherits from both Direct Forcing [5] and penalty methods [6,7]. We use this method in the context of the new Research and Development (R&D) studies currently carried out by the CEA to match the increasing safety and performance requirements for the second and third generation of nuclear reactors. Those R&D studies include behavior characterization, design and statistical shape optimization of innovative passive safety systems (*i.e.* conceived to prevent or mitigate potential accidental or incidental situations without needing extra energy). Some of them, such as the flow limiter [8] and the advanced accumulator [9], are based on the principle of hydraulic diodes (*i.e.* reverse flow is strongly damped [10]) and composed of thin in-flow obstacles with complex geometries immersed in turbulent two-phase flows.

* Corresponding author.
E-mail addresses: georis.billo@cea.fr (G. Billo), michel.belliard@cea.fr (M. Belliard).

In a recent work [2], we presented a Finite Element version of the PDF method, inspired from [4], in the context of the incompressible laminar Navier-Stokes equations, using an implicit fractional step algorithm (or projection scheme) [11] and a directional linear interpolation technique of the velocity near the immersed boundaries. In this present paper, we focus on different techniques to recover data from the discrete immersed boundary and different ways to achieve order 2 in space via linear interpolation. In particular, we are motivated by the comparison of the data reconstruction and interpolation techniques proposed in [2] with the more global ones proposed in [4].

Generally speaking, data reconstruction on a boundary/interface or in its vicinity is a key issue in the scope of Fictitious Domain Methods. Hence, various techniques have been developed in order to reconstruct fields coming from a boundary on the computation domain (*i.e.* the computation of the velocity imposed by the IB in our case) or the contrary (reconstruction of the stress tensor on the envelop of an airfoil to compute aerodynamic forces, for instance [12]) such as: mollifier functions (used in the original IBM to approximate the Dirac delta functions [13,14]), extrapolation outside the computation domain (used in ghost cells techniques [15,16]), interpolation (widely used in all kind of fictitious domain methods [5,4]). In the case of infinitely thin obstacles, the eXtended (or Generalized) Finite Element Method (X-FEM), which is often used in the field of fracture mechanics [17,18], provides some advantages: it is capable to deal with discontinuous quantities (typically tangential velocities on each side of an infinitely thin obstacle with slip conditions) while preserving the standard finite element properties elsewhere.

Let us notice that extrapolation and interpolation techniques, aside from involving polynomials or spline functions, can rather be directional (1D) [5] or spatial (multi-D) [4]. Here, we propose three interpolation techniques (unidirectional, multi-directional and hybrid) and a comparison between them on several test cases. In any case, geometrical data coming from the immersed obstacles are also needed and this is why we also propose several methods to compute those geometrical data from a Lagrangian surface mesh, usually obtained via Computer Assisted Design (CAD).

In addition, in some fields of physics, geometrical reconstruction is not sufficient to take into account specific phenomena properly. In that case, information is added by the means of analytical or empirical laws. For instance, when dealing with turbulence modeling (which is of interest for nuclear safety systems design), numerous turbulent wall laws have been developed and used to compute data (velocity, velocity gradient, friction, *etc.*) on or in the vicinity of an immersed boundary/interface. Thus, our final purpose is to use turbulent wall laws (see [19] for instance) to interpolate the fluid velocity near the obstacle, but it will be treated in a forthcoming paper.

In this paper, we briefly recall the PDF method and the numerical methods involved in Section 2. Section 3.1 tackles the topic of recovering data from the discrete immersed boundary whereas Section 3.2 focuses on the different linear interpolation approaches. Finally, Section 4 provides numerical results with a comparison between the different geometrical data reconstruction techniques and the different interpolation methods.

2. Numerical methods

The numerical methods involved in this paper are directly taken from [2] and references within. A brief summary is given in the following section.

2.1. Governing equations

In this paper, we consider an incompressible fluid of unit density and neglect gravity effects, which means that the Navier-Stokes equations are reduced to the following form:

$$\begin{cases} \partial_t \mathbf{u} + \nabla \cdot (\mathbf{u} \otimes \mathbf{u} - \nu \nabla \mathbf{u}) + \nabla p = \mathbf{0} & \text{on } \Omega \\ \nabla \cdot \mathbf{u} = 0 & \text{on } \Omega \\ +\text{BC} & \text{on } \partial\Omega \\ +\text{IC} & \text{on } \Omega \end{cases} \tag{1}$$

with $\Omega \subseteq \mathbb{R}^d$ a d-dimensional open compact domain with a piecewise regular boundary denoted $\partial\Omega$ such as $\overline{\Omega} = \Omega \cup \partial\Omega$, $p : \Omega \times \mathbb{R}^+ \to \mathbb{R}$ the fluid pressure, $\mathbf{u} : \Omega \times \mathbb{R}^+ \to \mathbb{R}^d$ the fluid velocity and $\nu \in \mathbb{R}^+$ the fluid cinematic viscosity.

2.2. Time discretization

Let denote $\delta t \in \mathbb{R}^{+*}$ the adaptative time step and $N_T \in \mathbb{N}$ the number of time steps. Given the sequence associated to the discrete time steps $(t_n)_{n<N_T+1}$, we define $\mathbf{u}^n : \Omega \to \mathbb{R}^d$ (resp. $p^n : \Omega \to \mathbb{R}$) as the approximations of the velocity (resp. the pressure) at time t_n. Considering a semi-implicit scheme (implicit diffusive terms and linearized advective terms) coupled with a projection scheme [11], we obtain the following fractional-step algorithm:

1. Prediction: only time, inertia, viscous and source terms are considered, the pressure term is kept from the previous time step. An intermediate velocity, called predicted velocity and denoted $\mathbf{u}^*$, is computed.

$$\frac{1}{\delta t}\mathbf{u}^* + \nabla \cdot (\mathbf{u}^n \otimes \mathbf{u}^* - \nu \nabla \mathbf{u}^*) + \nabla p^n = \frac{1}{\delta t}\mathbf{u}^n \tag{2}$$

2. Projection: the pressure corrector field, denoted ϕ^{n+1}, is computed using the predicted velocity and mass balance equation.

$$\frac{1}{\delta t}\left(\mathbf{u}^{n+1} - \mathbf{u}^*\right) + \nabla \phi^{n+1} = \mathbf{0} \tag{3}$$

$$\nabla \cdot (3) \Rightarrow \Delta \phi^{n+1} = \frac{1}{\delta t} \nabla \cdot \mathbf{u}^* \tag{4}$$

3. Correction: the velocity is computed using the pressure corrector gradient.

$$\mathbf{u}^{n+1} = \mathbf{u}^* - \delta t \nabla \phi^{n+1} \quad \text{and} \quad p^{n+1} = p^n + \phi^{n+1} \tag{5}$$

2.3. Space discretization

The computation domain Ω is divided in $N_E \in \mathbb{N}$ hexahedral elements, each denoted K_e with $e \in [\![1, N_E]\!]$ and Ω_e the portion of Ω associated to the element K_e. Those elements are composed of nodes and the total number of nodes is denoted N_N. The mixed FEM is used, which means that the discrete unknowns of the problem are decomposed in two different FE basis. For the velocity, a $\mathbb{Q}_1$ basis (*i.e.* trilinear decomposition at nodes) is used while, for pressure, a $\mathbb{Q}_0$ basis (*i.e.* the discrete pressure field is constant by element) is used (for more detailed information about the Finite Element formulation used in this paper, the interested reader can refer to [20]). This pair of elements is known to be unstable and can induce pressure checkerboard patterns. Yet, we used it because it is rather simple to implement and because the methods presented in this paper (Penalized Direct Forcing Immersed Boundary Method, geometrical data reconstruction, directional and multi-directional interpolations) do not depend strongly on the chosen pair of elements. If needed, they could be adapted to another pair of elements with little effort and without changing the philosophy behind this approach.

2.4. Immersed boundary method

The chosen Immersed Boundary Method is called Penalized Direct Forcing and was initially developed by M. Belliard et al. in a Finite Difference framework [3,4]. In this paper, we consider the adaptation of this method to a Finite Element formulation, proposed in [2]. To make

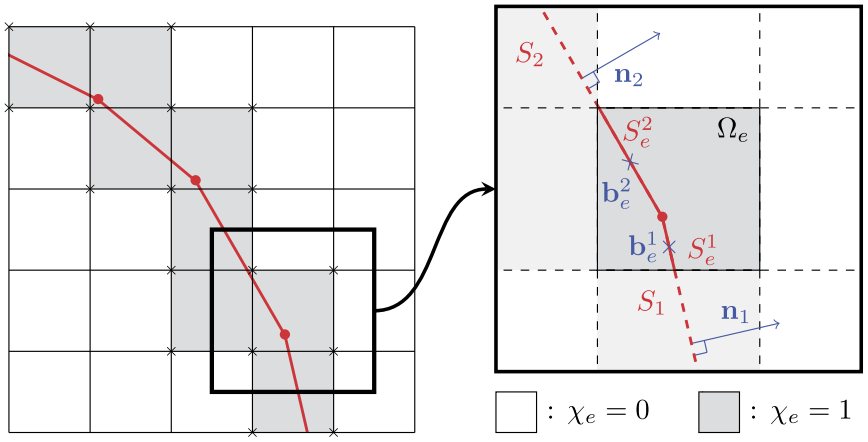

Fig. 1. Schematic representation of the computation grid and the collection of facets associated to the immersed boundary.

a brief reminder, the idea is to add a specific (penalized) Direct Forcing term to the momentum balance equation and to split it between the prediction and projection steps of the fractional-step algorithm (based on the work of [21] and references within):

$$\frac{1}{\delta t}\mathbf{u}^* + \mathbf{d}(\mathbf{u}^n, p^n, \mathbf{u}^*) = \frac{1}{\delta t}\mathbf{u}^n + \mathbf{f}_P^{n+1} \quad \text{with} \quad \mathbf{f}_P^{n+1} := \frac{\chi}{\eta \delta t}\left(\mathbf{u}_\Gamma^{n+1} - \mathbf{u}^*\right) \tag{6}$$

$$\frac{1}{\delta t}\left(\mathbf{u}^{n+1} - \mathbf{u}^*\right) + \nabla \phi^{n+1} = \mathbf{f}_C^{n+1} \quad \text{with} \quad \mathbf{f}_C^{n+1} := \frac{\chi}{\eta \delta t}\left(\mathbf{u}^* - \mathbf{u}^{n+1}\right) \tag{7}$$

where $\mathbf{d}(\mathbf{u}^n, p^{n+1}, \mathbf{u}^*) = \nabla \cdot (\mathbf{u}^n \otimes \mathbf{u}^* - \nu \nabla \mathbf{u}^*) + \nabla p^n$ is an operator used for purpose of clarity while $\mathbf{f}_P^{n+1}$ and $\mathbf{f}_C^{n+1}$ are the parts of the forcing term respectively added in equations (2) and (3) with $\eta \in \mathbb{R}^{+*}$ (such as $\eta \ll 1$) the penalty parameter, $\chi : \Omega \to \{0, 1\}$ the characteristic function of the immersed boundary Γ and $\mathbf{u}_\Gamma^{n+1}$ is the velocity imposed by Γ (or immersed Dirichlet boundary condition). By gathering terms together in the projection equation (7), we obtain:

$$\frac{1}{\delta t}\mathbf{u}^{n+1} + \frac{\eta}{\eta + \chi}\nabla \phi^{n+1} = \frac{1}{\delta t}\mathbf{u}^* \tag{8}$$

In terms of space discretization, we consider that the discrete characteristic function is equal to 1 in the elements crossed by Γ and 0 elsewhere (*i.e.* χ is decomposed in the $\mathbb{Q}_0$ FE basis). Thus, we can define χ_e and ξ_e such as:

$$\forall e \in [\![1, N_E]\!], \begin{cases} \chi_e = 1 & \text{if } \Omega_e \cap \Gamma \neq \emptyset \\ \chi_e = 0 & \text{else} \end{cases} \quad \text{and} \quad \xi_e = 1 + \frac{\chi_e}{\eta} \tag{9}$$

Regarding the fractional-step algorithm, let us denote $\mathbf{M}_e$ (resp. $\mathbf{D}_e$, $\mathbf{N}_e$ and $\mathbf{B}_e$) the lumped mass (resp. diffusive, advective and gradient-divergence) elemental matrix, λ_e^n (resp. λ_e^*) the components of the velocity (resp. predicted velocity) in the $\mathbb{Q}_1$ finite elements basis, ϕ_e^{n+1} the discrete pressure corrector in an element K_e (*i.e.* $\mathbb{Q}_0$) and, finally, λ_Γ^{n+1} the decomposition of the imposed velocity $\mathbf{u}_\Gamma^{n+1}$ in the $\mathbb{Q}_1$ finite elements basis. Then, the discrete projection scheme can be written as follows (*cf.* [2] for more details):

$$\left(\frac{1}{\delta t}\mathbf{M}_e \xi_e + \mathbf{D}_e \mu_e + \mathbf{N}_e\right)\lambda_e^* = +\mathbf{B}_e p_e^n + \frac{1}{\delta t}\mathbf{M}_e\left(\lambda_e^n + \frac{\chi_e}{\eta}\lambda_\Gamma^{n+1}\right) \tag{10}$$

$$\mathbf{B}_e^T \mathbf{M}_e^{-1}\frac{1}{\xi_e}\mathbf{B}_e \phi_e^{n+1} = -\frac{1}{\delta t}\mathbf{B}_e^T \lambda_e^* \tag{11}$$

$$\lambda_e^{n+1} = \lambda_e^* + \mathbf{M}_e^{-1}\frac{\delta t}{\xi_e}\mathbf{B}_e \phi_e^{n+1} \tag{12}$$

3. Imposed velocity interpolation

In [2], we presented two ways of computing the decomposition of $\mathbf{u}_\Gamma$ in the $\mathbb{Q}_1$ FE basis: one "staircase approximation" referred as direct assignment (first order accurate in space) and one involving linear directional interpolation (second order accurate in space). In this paper, we propose two other variants of linear interpolation and provide a comparison between all of them.

However, in order to achieve the linear interpolation of $\mathbf{u}_\Gamma$, additional geometrical data coming from the IB are needed. Several ways to obtain those data have been used and are described in section 3.1. From practical point of view, immersed boundaries are rather described by an equation when their geometry is simple – planes, circles, cylinders, *etc.* – and by a spline surface or a collection of facets – meshes coming from Computer Assisted Design (CAD) software for instance – when their geometry is complex. With respect to the aimed applications and CEA tools, only collection of plane facets – although exact equations are used on simple cases for purpose of verification and validation – are considered in the three investigated approaches.

3.1. Geometrical data reconstruction

The object of this section is to explain the different way used to recover useful geometrical data from the immersed boundary input data. Those input data, representative of the shape of the immersed obstacle, can come from various sources and take various forms. Without any claim to be exhaustive, we formally define two categories: analytical equations (simple shapes like circles, squares, *etc.*) and discrete values (measurements, experiments, Computer Aided Design, *etc.*).

Obviously, the second category is wider than the first one (discrete values can be generated from analytical equations) so, in order to remain as general as possible, we consider the approximate obstacle Γ_h (representative of Γ) as a union of discrete elements, here plane convex polygons called facets: $\Gamma_h = \bigcup_{i=1}^{N_S} S_i$

with N_S the total number of facets, $(S_i)_{i \in [\![1, N_S]\!]}$ the collection of facets and their respective normal vectors $(\mathbf{n}_i)_{i \in [\![1, N_S]\!]}$. Also, in order to make the following explanations clearer, let us define some notations: $J_e = \left\{ j \in [\![1, N_N]\!] \,\middle|\, \mathbf{x}_j \in \Omega_e \right\}$: the set of nodes included in element K_e, $E_j^0 = \left\{ e \in [\![1, N_E]\!] \,|\, j \in J_e \right\}$: the set of elements sharing the node j, $E_j^\chi = \left\{ e \in E_j^0 \,|\, \chi_e = 1 \right\}$: the set of element crossed by Γ_h sharing the node j and $J = \{ j \in [\![1, N_N]\!] \,|\, \exists e \in E_j^0 / \chi_e = 1 \}$: the set of node numbers for which a projection point need to be computed (points marked with a cross on Fig. 1).

Two approaches are considered here. The first one is the local weighting approach, introduced in [2] by the authors, that we confront to the second one, the global optimization approach, presented as a robust geometrical data reconstruction method in [4].

3.1.1. Weighting approach

The method based on quantity weighting and averaging is described in detail in [2]. Here, we only recall the main steps of the method. There

are four variants, depending of the weighting: arithmetic mean (variant 1), area weighting (variant 2), inverse distance weighting (variant 3), and area over distance weighting (variant 4). What ever is the considered variant, the geometrical computation is done in three steps.
First of all, we have to compute the intersection between hexahedral elements K_e and plane facets S_i (*cf.* Fig. 1). Doing this we get, in each intersected element K_e, a collection of facet's portions S_e^i, each characterized by an area $\mathcal{A}_e^i$, a normal $\mathbf{n}_i$ and barycenter $\mathbf{b}_e^i$.
Then, the second step is the determination of an equivalent facet by element, characterized by $\mathcal{A}_e$, $\mathbf{n}_e$ and $\mathbf{b}_e$. For this, in each intersected element, we sum the areas and we average the normals and the barycenters using an area weighting.

$$\forall e \in [\![1, N_E]\!]/\chi_e = 1, \mathcal{A}_e = \sum_{i=1}^{N_S} \mathcal{A}_e^i, \mathbf{n}_e = \frac{1}{\mathcal{A}_e} \sum_{i=1}^{N_S} \mathcal{A}_e^i \mathbf{n}_i, \mathbf{b}_e = \frac{1}{\mathcal{A}_e} \sum_{i=1}^{N_S} \mathcal{A}_e^i \mathbf{b}_e^i \tag{13}$$

Finally, the third step is the determination of immersed boundary projections $\mathbf{x}_j^p$ (*cf.* Fig. 2). In practice, we project each node $\mathbf{x}_j / j \in J$ on the equivalent facet of K_e: $\mathbf{p}_e^j := \mathbf{x}_j + l_e^j \mathbf{n}_e$, with l_e^j the oriented normal distance between the node and the equivalent facet. Then, $\mathbf{p}_e^j$ is assembled in a FE way to obtain: $\mathbf{x}_j^p = \frac{1}{\alpha_j} \sum_{e \in E_j^\chi} \alpha_e^j \mathbf{p}_e^j$ by average of the elementary projections of the neighboring elements using the selected variant for the weight α_e^j (arithmetic, area weighting, inverse distance weighting or area over distance weighting).

3.1.2. *Optimization approach*

In this approach, for a given node, we search for the point, located on Γ_h, which minimizes the distance between itself and the considered node – method inspired by C. Introïni et al. [4] and adapted to thin obstacles in this work. The optimization problem can be written as follows:

$$\forall j \in J, \quad \text{Find} \quad \mathbf{x}_j^p \in \Gamma_h \quad \text{such as} \quad (\mathbf{x}_j - \mathbf{x}_j^p)^2 = \inf_{\mathbf{y} \in \Gamma_h} (\mathbf{x}_j - \mathbf{y})^2 \tag{14}$$

To solve this problem for each $j \in J$, we use the algorithm proposed in [4]. First, the immersed boundary is partially reconstructed in the vicinity of $\mathbf{x}_j$ by collecting the facets of Γ_h which intersect at least one element sharing the node number j (the indexes of those facets form a set denoted I_j). Then, for each of those facets, we write the equation of the facet's plane:

$$\forall i \in I_j, \quad \mathbf{n}_i \cdot \mathbf{y} = \mathbf{n}_i \cdot \mathbf{a}_i \text{ with } \forall j \in J, \; I_j = \left\{ k \in [\![1, N_S]\!] \middle| \exists e \in E_j^0 \; / \; S_e^k \neq \emptyset \right\} \tag{15}$$

and $\mathbf{a}_i \in \mathbb{R}^d$ the coordinates of a point belonging to S_i. Gathering all the collected facets for a node j, the following linear system is obtained:

$$\mathbf{C}^j \mathbf{y} = \mathbf{r}^j \quad \text{with} \quad \forall \in j \in J, \forall i \in I_j, \forall k \in [\![1, d]\!], \quad C_{ik}^j = n_{i,k} \tag{16}$$

and:

$$\forall \in j \in J, \forall i \in I_j, r_i^j = \sum_{k=1}^{d} n_{i,k} a_{i,k} \tag{17}$$

where d is the space dimension, C_{ik}^j is the general term of $\mathbf{C}^j \in \mathbb{R}^{s \times d}$ with $s = \text{card}\left(I_j\right)$, $n_{i,k}$ the kth component of vector $\mathbf{n}_i$ and $a_{i,k}$ the kth component of vector $\mathbf{a}_i$.

Finally, we can consider that a point of coordinates $\mathbf{y}$ which verifies, at least approximately, the system (16) is located on Γ_h, or at least in its vicinity – *i.e.* solving this linear system is a way to impose the constraint of the minimization problem (14). If we formally denote $\mathcal{V}_{\Gamma_h}^j = \left\{ \mathbf{y} \in \mathbb{R}^d \middle| \mathbf{C}^j \mathbf{y} \approx \mathbf{r}^j \right\}$, the set of the points which approximately verify the system (16), the minimization problem (14) becomes:

$$\forall j \in J, \quad \text{Find} \quad \mathbf{x}_j^p \in \mathcal{V}_{\Gamma_h}^j \quad \text{such as} \quad (\mathbf{x}_j - \mathbf{x}_j^p)^2 = \inf_{\mathbf{y} \in \mathcal{V}_{\Gamma_h}^j} (\mathbf{x}_j - \mathbf{y})^2 \tag{18}$$

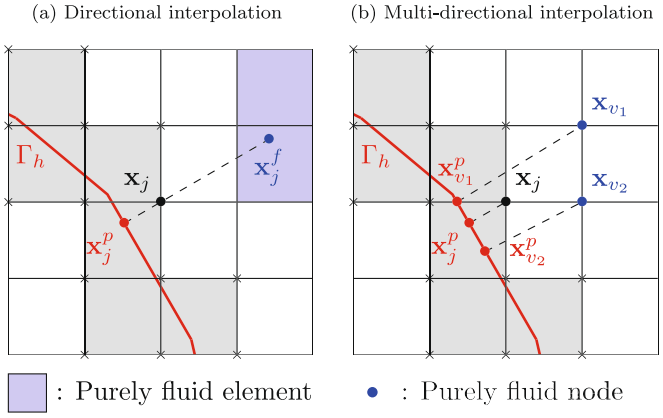

Fig. 2. Schematic drawings of the different interpolations techniques.

This problem is solved using an Uzawa algorithm for each node j, with $\epsilon \ll 1$ the shutoff parameter and Λ^m the Lagrange multiplier vector at iteration m:

1. Initialization ($m = 0$):
   - $\Lambda^0 = \mathbf{0}_{\mathbb{R}^s}$
2. Iterations ($m > 0$):
   - $\mathbf{y}^m = \mathbf{x}_j - \frac{1}{2} \left(\mathbf{C}^j\right)^T \Lambda^{m-1}$
   - $\Lambda^m = \Lambda^{m-1} + \frac{\mathbf{C}^j \mathbf{y}^m - \mathbf{r}^j}{\|\mathbf{C}^j\|_1 \|\mathbf{C}^j\|_\infty}$
3. Finalization:
   - if $(\mathbf{x}_j - \mathbf{y}^m)^2 < \epsilon$ then $\mathbf{x}_j^p = \mathbf{y}^m$

Another variant of this approach has also been developed. It relies on a different definition of I_j: instead of considering only the elements sharing the node j to collect the facets of Γ_h, we also consider the neighbors of those elements.

3.2. *Linear interpolation methods*

With those new pieces of information coming from the geometry of the immersed boundary Γ_h, interpolation methods can be used to compute the imposed velocity $\mathbf{u}_\Gamma^{n+1}$. In this document, we propose three techniques (detailed in the following sections): one purely directional, another one being multi-directional and a final one being a combination of the two. The first method was introduced by the authors in [2] taking advantage of the FE space discretization. The second one is inspired from [4] and was initially introduced in a context of a Finite Difference space discretization. The third one is an original approach.

From now on, the time index exponent notation will be omitted in this section for purpose of readability, keeping in mind that the imposed velocity is interpolated at each time step.

3.2.1. *Directional interpolation*

The directional interpolation variant is represented schematically in Fig. 2a and a full description can be found in [2]. Here, we briefly recall the principle of the method.

Given the coordinates $\mathbf{x}_j^p$ of the approximate projection of node j, we can reconstruct an outward normal vector $\mathbf{n}_j = (\mathbf{x}_j - \mathbf{x}_j^p)|\mathbf{x}_j - \mathbf{x}_j^p|^{-1}$ and then we find a point $\mathbf{x}_j^f$ in the prolongation of $\mathbf{n}_j$ (*cf.* Fig. 2a). The point $\mathbf{x}_j^f$ is chosen in the vicinity of $\mathbf{x}_j$ and as belonging to a "purely fluid" element (*i.e.* not belonging to elements crossed by the boundary). Finally, we can interpolate the velocity $\mathbf{u}_j$ at node j as follows:

$$\mathbf{u}_j = \mathbf{u}_j^p + \frac{\mathbf{u}_j^f - \mathbf{u}_j^p}{|\mathbf{x}_j^f - \mathbf{x}_j^p|} |\mathbf{x}_j - \mathbf{x}_j^p| \tag{19}$$

where $\mathbf{u}_j^f$ is the fluid velocity at $\mathbf{x}_j^f$, evaluated using the FE basis functions, and $\mathbf{u}_j^p$ is given by the IB condition.

3.2.2. Multi-directional interpolation

The idea of this approach, here adapted from [4] to a Finite Element framework (roughly schematized in Fig. 2b), can be summarized as follows:

1. For each purely fluid node surrounding the node $j \in J$, we compute an approximate derivative, in the direction normal to Γ_h, of the fluid velocity.
2. We compute the arithmetic mean of the obtained approximate derivatives.
3. We use this mean normal derivative to compute $\mathbf{u}_j$.

However, to be able to compute approximate normal derivative of the velocity at the nodes surrounding j, we need the projections of those nodes on Γ_h. Those purely fluid neighbors are denoted as follows:

$$\forall j \in [\![1, N_N]\!], V_j = \left\{ v \in [\![1, N_N]\!] \middle| \exists e \in E_j^0 / v \in J_e \text{ and } \forall f \in E_v^0, \chi_f = 0 \right\} \tag{20}$$

Regarding geometrical data reconstruction, the method to achieve the projection slightly differs between the weighting and optimization approaches:

In the weighting approach. First, new elemental projections are computed:

$$\forall e \in [\![1, N_E]\!] / \chi_e = 1, \forall j \in J_e, \forall v \in V_j, \quad \mathbf{p}_e^v = \mathbf{x}_v + l_e^v \mathbf{n}_e \tag{21}$$

with the definition of l_e^v being identical to the one presented in Section 3.1.1. Then they are assembled in the following way:

$$\forall j \in [\![1, N_N]\!] / \alpha_j \neq 0, \forall v \in V_j, \quad \mathbf{x}_v^p = \frac{1}{\alpha_v} \sum_{e \in E_j^\chi} \alpha_e^v \mathbf{p}_e^v \tag{22}$$

with:

$$\forall j \in [\![1, N_N]\!] / \alpha_j \neq 0, \forall v \in V_j, \quad \alpha_v = \sum_{e \in E_j^\chi} \alpha_e^v \tag{23}$$

In the optimization approach. For nodes of V_j, we must collect the facets that intersect neighbors or neighboring elements, which means:

$$\forall v \in \bigcup_{j \in J} V_j, \quad I_v = \left\{ k \in [\![1, N_S]\!] \middle| \exists e \in E_v^1 / S_e^k \neq \emptyset \right\} \tag{24}$$

Then the approximated minimization problem (18) is also solved for nodes v where $I_v \neq \emptyset$ using the methodology presented in Section 3.1.2.

Provided the projections of neighbors, the velocity imposed at node j is computed as follows:

$$\forall j \in J, \quad \mathbf{u}_j = \mathbf{u}_j^p + \gamma_j \sum_{v \in V_j} \frac{\mathbf{u}_v - \mathbf{u}_v^p}{|\mathbf{x}_v - \mathbf{x}_v^p|} |\mathbf{x}_j - \mathbf{x}_j^p| \tag{25}$$

with:

$$\forall j \in J, \quad \begin{cases} \gamma_j = \dfrac{1}{\text{card}(V_j)} & \text{if card}(V_j) > 0 \\ \gamma_j = 0 & \text{else} \end{cases} \tag{26}$$

3.2.3. Hybrid strategy

In future works, we will investigate turbulent immersed wall laws. As wall laws are usually directional, the directional approach of Section 3.2.1 is privileged. However, depending on the immersed boundary geometry and the space step resolution, finding purely fluid elements around the node j, to define a fluid point $\mathbf{x}_j^f$, is not always possible.

Table 1
Ranges of grid parameters for the geometrical test cases. The diameter of the cylinder $d = 2r$ and the chord length of the NACA0012 airfoil c are equal to 1.

Case	Length	N_E	Grid Step	N_S	Grid step
Cylinder	[21, 151]	[0.066, 0.63]	[25, 250]	[0.016, 0.13]	
NACA0012	[21, 201]	–	[25, 250]	–	

In this case, a direct assignment of the immersed boundary velocity on node j is usually done, decreasing the space order convergence. To overcome this issue, we propose a hybrid strategy consisting in applying the directional approach where possible and the multi-directional approach elsewhere. Doing this, we apply second order space discretizations for all the configurations. Nevertheless, the computational cost is slightly increased by the fact that we have to build the needed interpolation data, not only for the point $\mathbf{x}_j^f$, but also for the purely fluid nodes $\mathbf{x}_v$ and their projection on Γ_h (*cf.* Fig. 2)

4. Numerical results and discussion

This section is divided in two parts:

1. the first one focuses on testing the geometrical reconstruction methods without flow simulation (*cf.* Section 4.1),
2. the second one focuses on flow simulations considering a fixed geometrical reconstruction method (*cf.* Sections 4.2 and 4.3).

4.1. Comparison between the geometrical data reconstruction methods

First, the different geometrical methods are compared using a simple 2D case with regular boundary (computed in a 3D framework): the circular cylinder. Second, the same geometrical methods are compared using a more complex 2D case corresponding to the geometry of a NACA0012 airfoil. In this case, the trailing edge is no longer a regular boundary.

4.1.1. A regular test case: the circular cylinder

We consider a square domain of side 1.5 with a circular cylinder of radius $r = 0.5$ and center $\mathbf{c} = (0, 0)$ (*cf.* Fig. 3a). We compute some error indicators and observe their evolution when we refine the square hexahedral volume mesh (Cartesian grid) or the cylindrical surface mesh (*i.e.* increase the number of facets) in order to compare accuracy and convergence rate (summary of grid configurations in Table 1). We use three different error indicators:

1. The radius related error: $\forall j \in J, \quad e_j^r = \left| r_j^p - r \right|$ with $r_j^p = \left| \mathbf{x}_j^p - \mathbf{c} \right|$,
2. The distance related error: $\forall j \in J, \quad e_j^d = \left| \mathbf{x}_j^p - \mathbf{x}_j^e \right|$,
3. The normal vector related error: $\forall j \in J, \quad e_j^n = \left| \cos^{-1} \left| \mathbf{n}_j \cdot \mathbf{n}_j^e \right| \right|$ with $\mathbf{n}_j = \mathbf{x}_j - \mathbf{x}_j^p / \left| \mathbf{x}_j - \mathbf{x}_j^p \right|$ and $\mathbf{n}_j^e$ the exact normal vector,

and, for each of them, we compute an approximated $\mathcal{L}^2$ and $\mathcal{L}^\infty$ norms. Fig. 4a (resp. Fig. 4b) shows the evolution of those error indicators when the volume (resp. surface) mesh is refined for the weighting and optimization approaches. Let us recall that Weighting Ap. 1 (resp. 2, 3, 4) is arithmetic mean (resp. area, inverse distance and area over distance weighting) and that Optimization lv. 1 (resp. 2) collects facets intersecting neighboring elements (resp. neighboring elements and their neighbors) of a given node. As a whole, when considering convergence with respect to the Eulerian grid step, all approaches seem to provide a roughly quadratic order ($[2.34, 2.54]$ in $\mathcal{L}^2$ norm and [1.67,1.87] in $\mathcal{L}^\infty$ norm) for the radius and the distance related errors and linear ($[1.20, 1.42]$ in $\mathcal{L}^2$ norm and $[0.70, 0.87]$ in $\mathcal{L}^\infty$ norm) for the normal related error. Concerning convergence with respect to the Lagrangian, the conclusion for the weighting approach remains the same (except for the convergence of the distance indicator in $\mathcal{L}^\infty$ norm which becomes

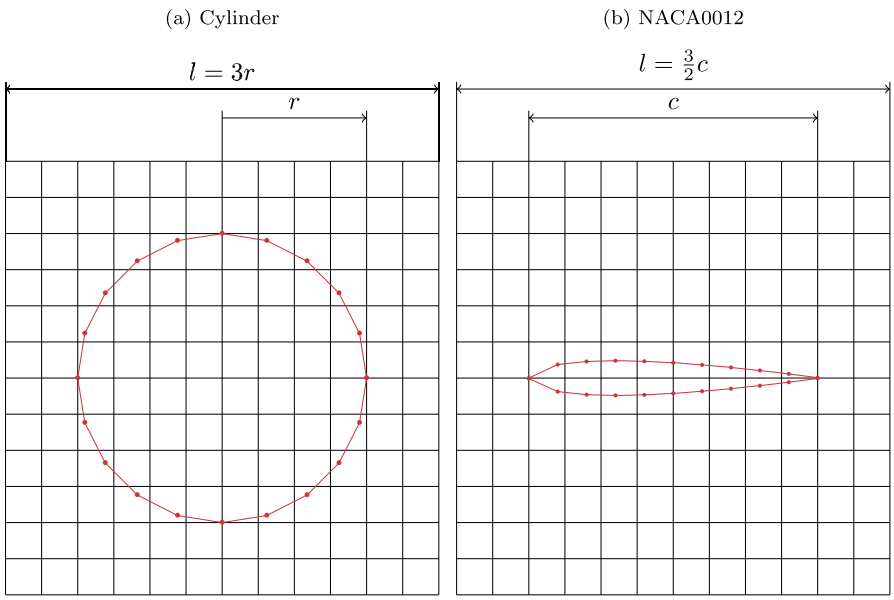

Fig. 3. Examples of Eulerian (black) and Lagrangian (red) meshes for the geometrical test cases with 144 elements/cells (*i.e.* $N_E = 144$) and 20 facets (*i.e.* $N_S = 20$) and $c = 2r = 1$.

roughly linear with value in $[0.53, 1.12]$). Optimization approaches systematically provide lower convergence rates.

It can also be noted that the optimization "level 1" systematically provides the largest error – probably because it takes into account data coming farther away from the considered point, making the approach formally non-local. On the other hand, when considering the weighting approach, the way of defining α_e^j (*cf.* section 3.1.1) seems to have little impact on the results except for the $\mathcal{L}^\infty$ norm of the normal vector related error when refining the surface mesh. Indeed, when $\alpha_e^j = 1$ or $\alpha_e^j = \mathcal{A}_e$ (*i.e.* Weigh. Ap. 1 and 2), we can see a jump at $N_S = 125$ which is softened if an inverse distance weighting is used (*i.e.* Weigh. Ap. 3 and 4). From Fig. 5b, we notice that a plateau minimal error is reached (or almost reached) for the finest Lagrangian space discretization. The level of this minimal error is driven by the Eulerian space step. It is not the case for the Eulerian space-step convergence study of Fig. 5a, where the minimal error driven by the Lagrangian space step is not reached. This could be explained by the fact that the minimum space step for the Lagrangian mesh (1.26×10^{-2}) is slightly lower than the one of the Eulerian mesh (6.62×10^{-2}) – factor 5. Finally, if we compare the weighting and optimization approaches, we can see that the optimization provides slightly larger errors but also a steadier convergence (less jagged), especially in $\mathcal{L}^\infty$ norm, as we can expect from this method reputed as a robust one.

Moreover, concerning computational resources, we monitored the execution time of the intersection pipeline – other CPU tasks can pollute execution time but the accuracy of the results is about 0.1 s, which is clearly sufficient for the comparison. The weighting approach is greatly faster: 0.7 to 21 s depending on the grids configurations against 41.7 to 496 with the optimization approach) – this is a key finding regarding our application of the proposed modeling (*i.e.* shape optimization in nuclear component design). Another interesting finding is that the execution time seems to be more dependent on the number of volume elements (*i.e.* grid step) when considering the weighting approach but more dependent on the number of facets when considering the optimization approach (in fact it depends on the number of collected facets per node). These tendencies were expected. On one side, the weighting approach is rather local and the number of operations is, to some extent, constant for one given volume. Therefore, the computation cost mainly depends on the total number of volumes (or meshes). On the other side, the optimization approach is global and the number of operations, for a given node, strongly depends on the collected-facet number in the adjacent volumes. Therefore, when decreasing the volume space step, the number of nodes to project is increasing, but the number of Uzawa operations is decreasing, leading to the observed behavior.

Table 2
Geometrical and physical parameters used for the laminar Taylor-Couette flow.

r_1 (m)	r_2 (m)	ω_1 (rad s^{-1})	ω_2 (rad s^{-1})	ν (m^2 s^{-1})	$Re = \lvert\omega\rvert r_1^2 \nu^{-1}$
5×10^{-1}	1	1	-1	2.5×10^{-1}	1

4.1.2. A non regular test case: the NACA0012 airfoil

In this case, we consider the 2D NACA0012 airfoil described in Fig. 3b for wich the trailing edge is no longer a regular boundary. A summary of grid configurations is presented in Table 1. The same error indicators are considered (except the radius one for obvious reasons) and their evolution with respect to the grid step (resp. number of facets) is show in Fig. 5a (resp. Fig. 5b). In that case, we can see that all approaches provide equivalent error values and equivalent convergence orders. Moreover, for all approaches, convergence is roughly linear in $\mathcal{L}^2$ norm ($[0.94, 0.97]$) with respect to the Eulerian grid step and really degraded when considering the Lagrangian mesh ($[0.17, 0.24]$) – with more than 75 facets, the error seems to reach a plateau, tending to show that the error linked to the Eulerian grid becomes predominant. Thus, all approaches fail to handle the trailing edge ($\mathcal{L}^\infty$ norm of error is close to 1 and located around the trailing edge). However, when we look at the execution times (0.6 to 23.9 s with the weighting approach and 53 to 179.4 s with the optimization approach) we can assess that the weighting approach is still largely faster for the same accuracy in the results. Therefore, in terms of performance and regarding our application, the weighting approach remains the most advantageous. Hence, in the following cases, only the weighting (approach 3) is considered to obtain geometrical data from the discrete immersed boundary.

4.2. Laminar Taylor-Couette flow

4.2.1. Description of the Taylor-Couette flow

Taylor-Couette flow refers to the flow between two infinitely long concentric circular cylinders rotating at different angular velocities. Fig. 6a gives a schematic view of the case configuration where $r_1 = 0.5$ m (resp. $r_2 = 1$ m) is the radius of the inner (resp. outer) cylinder, $\omega_1 = 1$ rad.s^{-1} (resp. $\omega_2 = -1$ rad.s^{-1}) is the angular velocity of

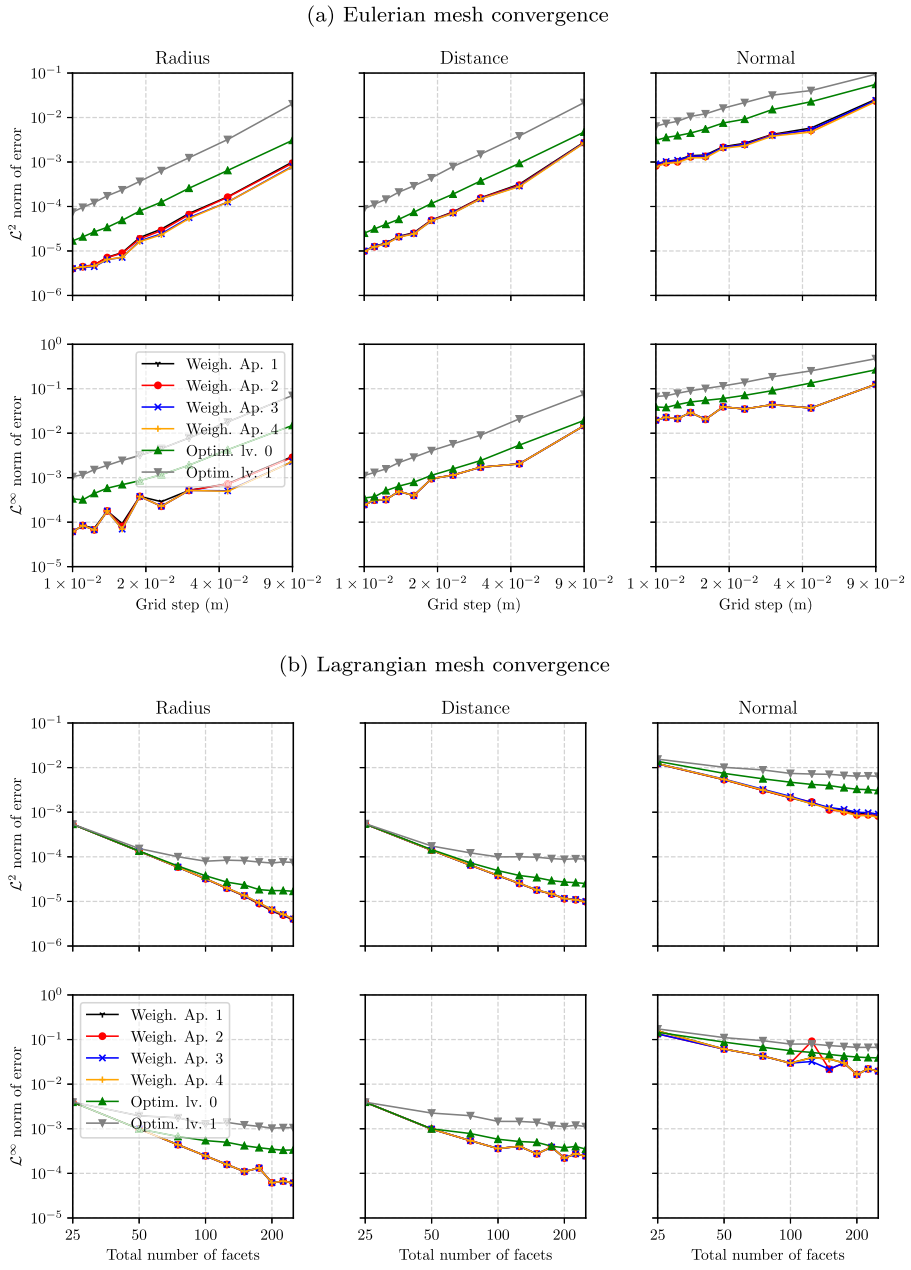

Fig. 4. Intersection of an Eulerian volume mesh and a cylindrical Lagrangian surface mesh: evolution of different error indicators with respect to the Eulerian (resp. Lagrangian) grid step at fixed Lagrangian (resp. Eulerian) grid step in log/log scale.

the inner (resp. outer) cylinder, Γ_1 (resp. Γ_2) is the immersed boundary corresponding to the inner (resp. outer) cylinder (they correspond to a no-slip condition, which means a Dirichlet BC with the Dirichlet fluid velocities given by the cylinders velocities), $l = 2r_2 + \frac{1}{4}$ is the side of the square domain and (e_r, e_θ) is the polar frame. An *ad hoc* value is imposed to the viscosity ($\nu = |\omega| r_1^2$) to guarantee that the Reynolds number ($Re = |\omega| r_1^2 \nu^{-1}$) is equal to 1 (*cf.* Table 2 for a summary of the different sets of parameters). The stability of this flow in ensured by a condition on the Taylor number: $Ta < Ta_c \approx 1.712$ (see [22] for value and definitions). The numerical application gives, in our case, $Ta = 1.5$, so the criterion is respected. This means that, using lubrication theory, we are able to compute a steady state solution in which the fluid velocity is purely azimuthal.

4.2.2. Mesh convergence study

The global behavior is in good agreement with the analytical solution (*cf.* Fig. 7a). The spatial convergence is numerically assessed in Fig. 8. The order of convergence, when using interpolation, is close to 2 (between 1.7 and 1.8 in $\mathcal{L}^2$ norm and about 1.5 in $\mathcal{L}^\infty$ norm), which is consistent with the theory. Moreover, the different interpolation techniques greatly enhance the results (a factor 10^{-1} applied on the $\mathcal{L}^2$ norm of the error). It can also be noted that the multi-directional interpolation seems more sensitive to the mesh refinement as it can produce higher local errors when the space resolution is low. For this test case, with well separated regular immersed boundaries, the hybrid approach provides the same results as the directional one.

4.3. Laminar flow around a circular cylinder

4.3.1. Description of the flow around a circular cylinder

The flow around a circular cylinder is a widely studied problem in the field of fluid dynamics. Fig. 6b gives the test case configuration where Γ is the immersed boundary corresponding to the surface of the cylinder, $r = 0.5$ m is the cylinder radius, $l = 120r$ is the side of the

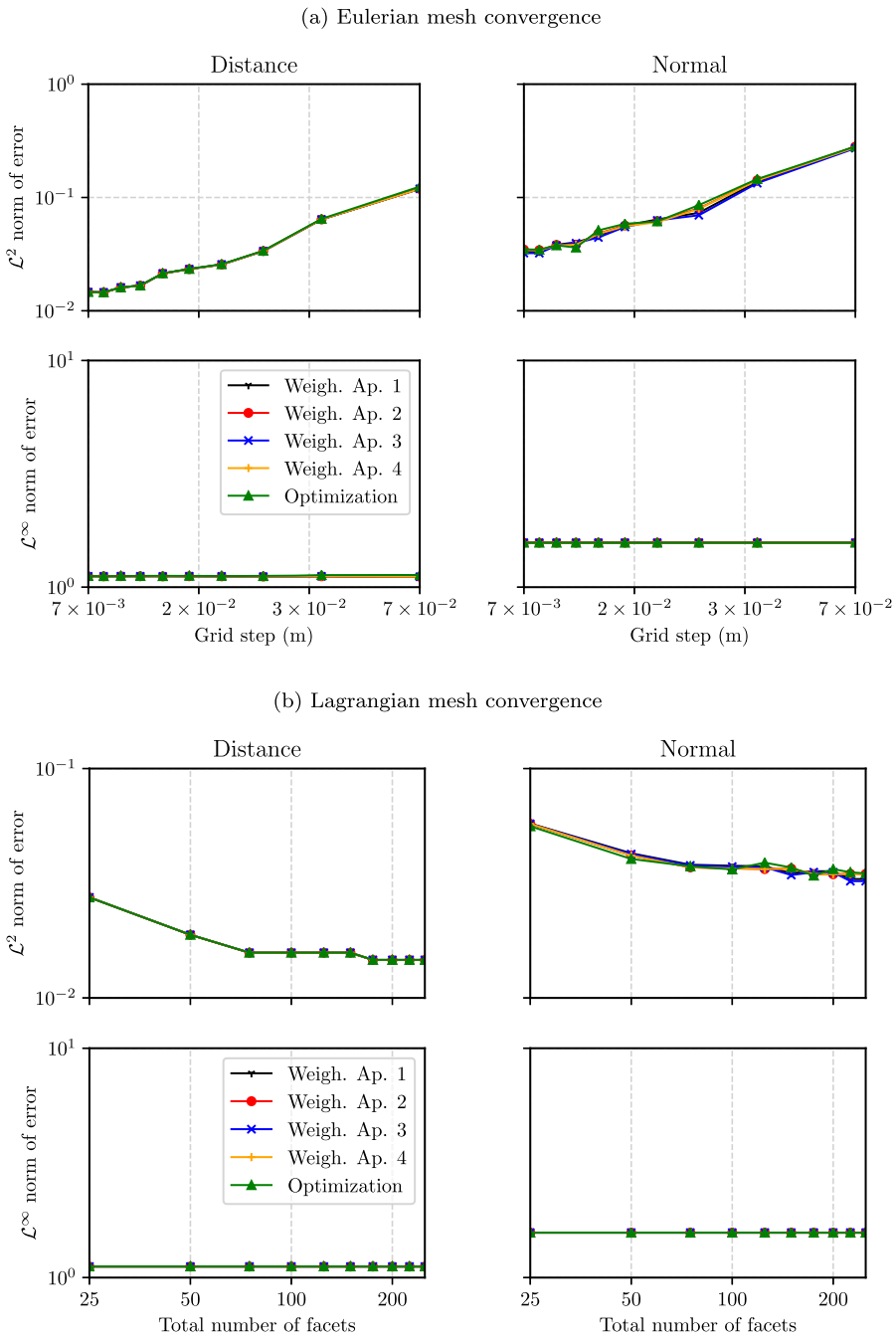

Fig. 5. Intersection of a hexahedral Eulerian volume mesh and a NACA 0012 airfoil Lagrangian surface mesh: evolution of different error indicators with respect to the Eulerian (resp. Lagrangian) grid step at fixed Lagrangian (resp. Eulerian) grid step in log/log scale.

square domain. It is large to avoid boundary effects. Here, $\mathbf{u}_\infty$ represents the uniform inlet velocity field (with $U_\infty = 1$ its norm) and $\omega \in \{0, 2\}$ is the dimensionless angular velocity of the cylinder. There is no analytical solution, but many experiments and simulations give macroscopic indicators, such as the drag and lift coefficients, as comparison elements in the static case (*i.e.* $\omega = 0$). Some experimental data are also available for the rotating case. In practice, similarly to the Taylor-Couette case, different values of the Reynolds number ($Re = 2U_\infty r\nu^{-1}$) are achieved by imposing *ad hoc* values to the viscosity ν (*cf.* Table 3 for a summary of the different sets of parameters).

Table 3
Geometrical and physical parameters used for the laminar flow around a cylinder.

r (m)	U_∞ (m s^{-1})	ω (rad s^{-1})	ν (m^2 s^{-1})	$Re = 2U_\infty r\nu^{-1}$
5×10^{-1}	1	0	5×10^{-2}	2×10^{1}
5×10^{-1}	1	2	5×10^{-2}	2×10^{1}
5×10^{-1}	1	0	10^{-2}	10^{2}
5×10^{-1}	1	2	10^{-2}	10^{2}

4.3.2. Mesh convergence study and aerodynamic coefficients

After checking that the global behavior of the flow was in good agreement with the literature (*cf.* Fig. 7b), the mesh convergence study was carried in both static and rotating configurations using the same parameters than the one presented in [2] – *i.e.* small domain ($20r \times 20r$) and finest simulation results considered as reference. The convergence curves, in $\mathcal{L}^2$ and $\mathcal{L}^\infty$ norm, are presented in Figs. 9 and 10 for both

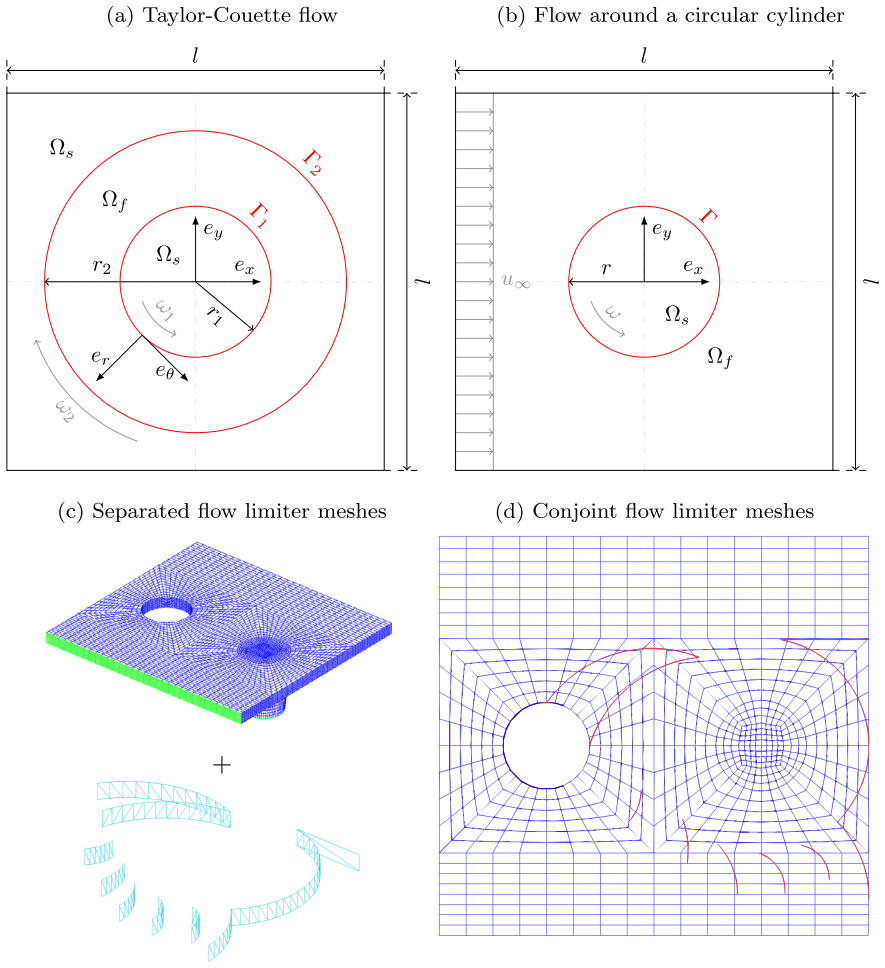

Fig. 6. Schematic representation of the computational domain for various cases and flow limiter meshes drawings (source: [2]).

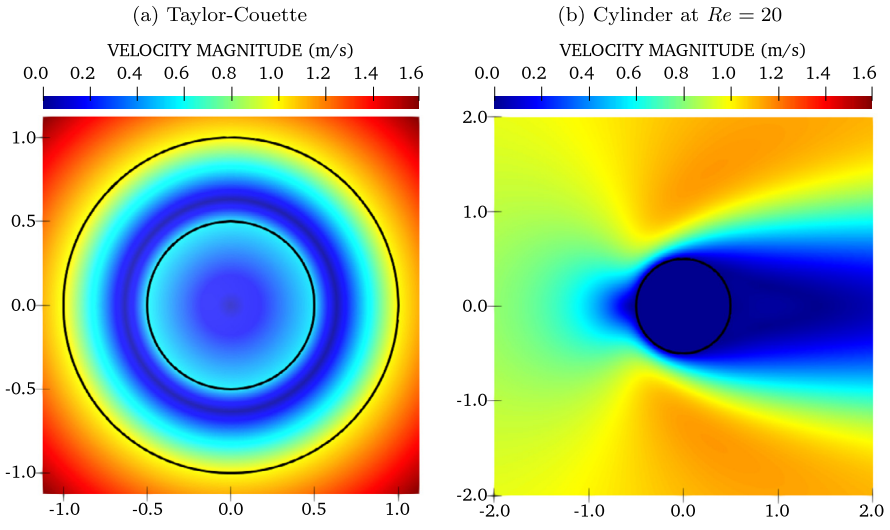

Fig. 7. Example of computed velocity magnitude fields for the flow around a circular cylinder at $Re = 20$ and the Taylor-Couette flow.

static and rotating cases. At $Re = 20$ (*cf.* Figs. 9a, 9b, 9c, 9d), the linear interpolation of the velocity in the vicinity of the obstacle reduces the error between the computed and reference solutions while increasing the rate of convergence. As expected, the order of convergence is close to 2 in $\mathcal{L}^2$ norm (in [1.7, 1.9]) when using linear interpolation. With direct assignment however, the order is slightly larger than 1 (1.3). As the physics is more complex and as there is no analytical solution available to compute the error, this deviation from the theory is expected. It is also noticeable in $\mathcal{L}^\infty$ norm: the value is in [1.3, 1.5] with linear interpolation and in [0.7, 0.9] with direct assignment. Nonetheless, what is important to note is that an approximative ratio of 2 in convergence order is preserved between direct assignment and linear interpolation.

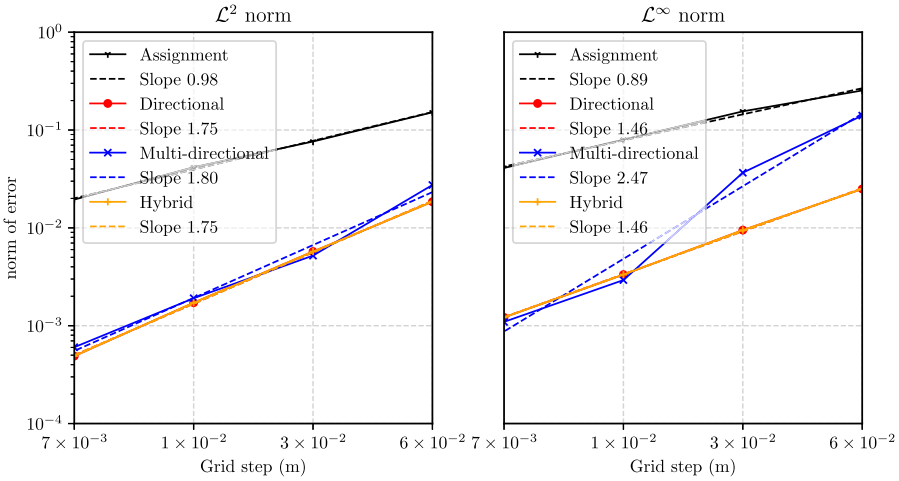

Fig. 8. Evolution of the relative $\mathcal{L}^2$ and $\mathcal{L}^\infty$ norms of error of the azimuthal velocity with respect to the grid step in the case of the Taylor-Couette flow.

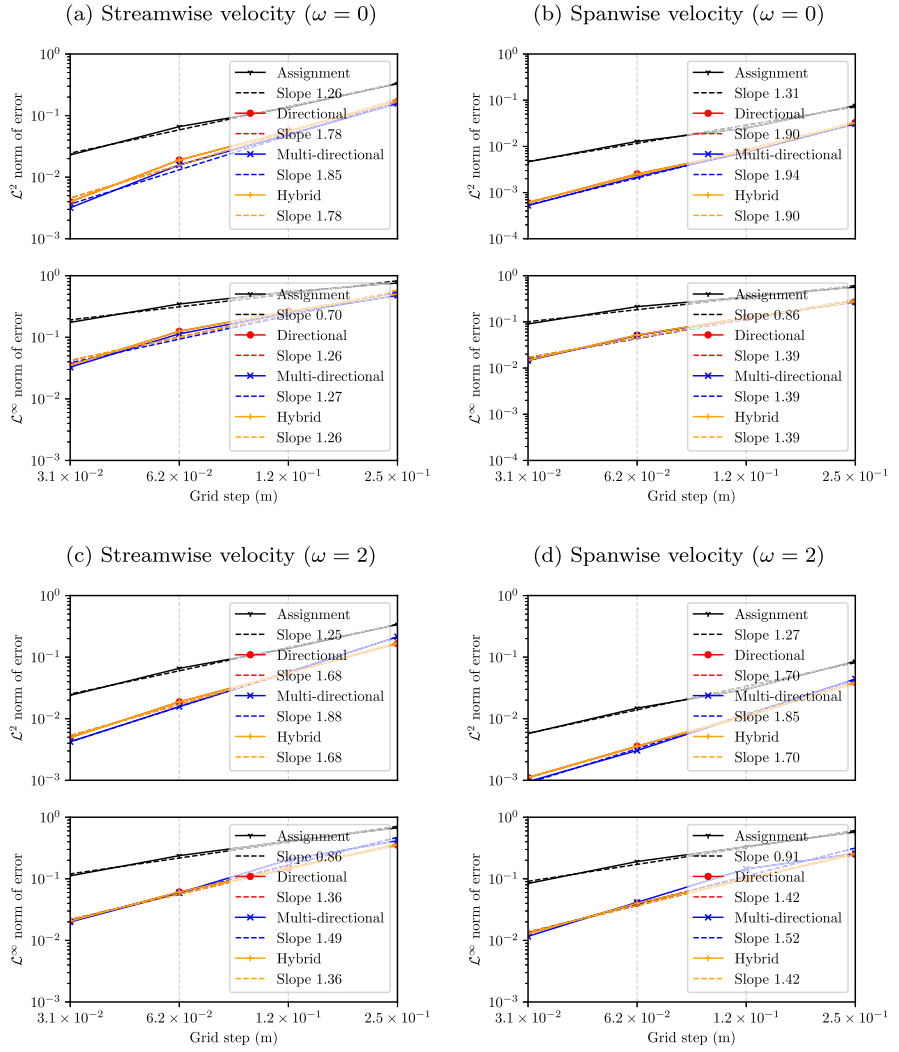

Fig. 9. Mesh convergence ($\mathcal{L}^2$ and $\mathcal{L}^\infty$) of the velocity for the steady flow around a circular cylinder ($Re = 20$) in both static and rotating configurations.

Furthermore, we can say that the multi-directional approach seems to provide slightly superior convergence rate but, overall, results are similar.

To study the mesh convergence at $Re = 100$ (unsteady flow regime), as we use an adaptative time step which depends on the grid step (so the solutions of unsteady cases desynchronize when the mesh changes), only the mean value of the drag coefficient is considered. The results obtained with the interpolation techniques are summarized in Fig. 10. The difference in convergence order between the linear interpolation ([1.5,1.7] in the static case and [1.7,2.1] in the rotating case) and the

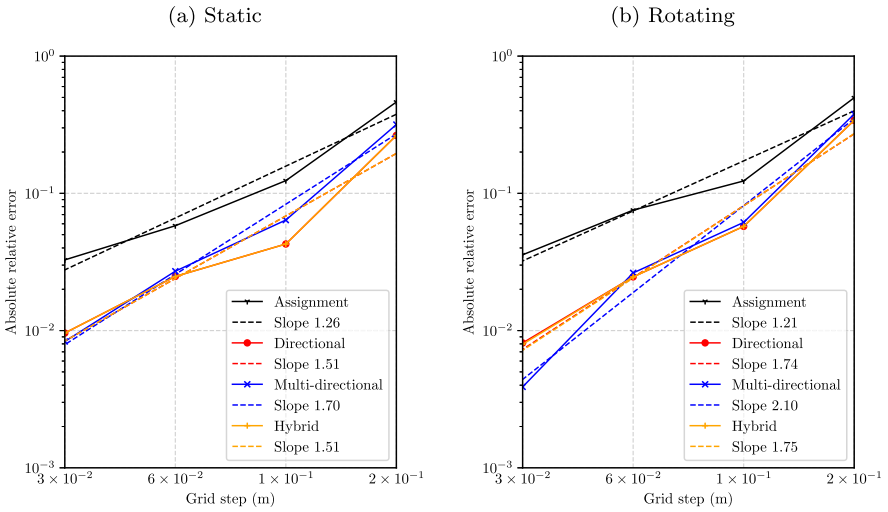

Fig. 10. Mesh convergence of the mean value of the drag coefficient for the unsteady flow around a circular cylinder ($Re = 100$).

Table 4
Aerodynamic coefficients computed for the steady laminar flow around a circular cylinder ($Re = 20$), static and rotating cases, using direct assignment (label "A") and linear interpolation techniques ("B": directional, "C": multi-directional, "D": hybrid). Reference value for the static case are taken from [4,23–27] whereas reference values for the rotating case are taken from [4,24,28–30].

	ω	Present work				References
		A	B	C	D	
C_d	0	2.099	2.054	2.053	2.054	2.02 – 2.09
C_d	2	1.911	1.880	1.879	1.880	1.85 – 2.000
L_w	0	1.047	0.918	0.919	0.918	0.900 – 094
C_l	2	2.877	2.779	2.780	2.779	2.617 – 3.032
$\alpha(°)$	2	56.41	55.92	55.95	55.92	53.66 – 57.68

direct assignment (about 1.2 in both cases) is noticeable. In both cases, the order of convergence is overestimated for the direct assignment with respect to the theory (*i.e.* slightly superior to 1). Concerning linear interpolation, the order of convergence is closer to 2 in the static case. It is also worth noting that the multi-directional technique seems to provide higher order of convergence (1.7 in the static case and 2.1 in the rotating case) than the directional and hybrid techniques even if, overall, the error values are very similar.

The laminar flow around a circular cylinder was also used to carry out a global quantity study. This time, we consider a much larger domain ($120r \times 120r$) to minimize potential boundary effects and we use a grid counting 1,638,400 elements (approximately 40 within the diameter of the cylinder). Depending on the configuration, different quantities have been monitored.

- Static cylinder at $Re = 20$: drag coefficient (C_d) and recirculation length (L_w) presented in Table 4.
- Rotating cylinder at $Re = 20$: drag coefficient (C_d), lift coefficient (C_l) and angle between the aerodynamic force and the horizontal axis (α) presented in Table 4.
- Static cylinder at $Re = 100$: mean drag coefficient and fluctuations (resp. $\overline{C_d}$ and C_d'), lift coefficient fluctuations (C_l') and Strouhal number St presented in Table 5.
- Rotating cylinder at $Re = 100$: mean drag coefficient and fluctuations (resp. $\overline{C_d}$ and C_d'), mean lift coefficient and fluctuations (resp. $\overline{C_l}$ and C_l') and Strouhal number St presented in Table 5.

Table 5
Aerodynamic coefficients related to the unsteady laminar flow around a circular cylinder ($Re = 100$) using direct assignment (label "A") and linear interpolation techniques ("B": directional, "C": multi-directional, "D": hybrid). Reference value for the static case are taken from [4,28,26,23,31–34] whereas reference values for the rotating case are taken from [4,28,24,35].

	ω	Present work				References
		A	B	C	D	
$\overline{C_d}$	0	1.363	1.325	1.321	1.325	1.317 – 1.376
$\overline{C_d}$	2	1.138	1.102	1.098	1.102	1.0979 – 1.1890
C_d'	0	0.012	0.011	0.011	0.011	0.009 – 0.012
C_d'	2	0.105	0.091	0.090	0.091	0.0986 – 0.1195
$\overline{C_l}$	2	2.641	2.562	2.560	2.562	2.4050 – 2.51
C_l'	0	0.343	0.295	0.294	0.295	0.227 – 0.349
C_l'	2	0.369	0.301	0.297	0.301	0.3603 – 0.4427
St	0	0.161	0.165	0.165	0.165	0.164 – 0.170
St	2	0.163	0.166	0.166	0.166	0.165 – 0.1732

As a whole, we can see that the obtained results are in very good agreement with the reference values, in both steady (*i.e.* $Re = 20$) and (*i.e.* $Re = 100$) unsteady cases. Moreover, the linear interpolation provides results that are systematically closer to the reference values (all the kinds of interpolation produce similar values in this case with regular geometry and a very fine mesh).

4.4. *Industrial case involving the flow limiter*

4.4.1. *Description of the flow limiter test case*

This case is representative of the flow into a hydraulic diode. As we can see in Figs. 6c and 6d, we have a volume mesh (which represents the fluid located in the downcomer of a Pressurized Water Reactor including the vessel inlet and vessel outlet) and a surface mesh (which comes from a CAD software and represents the shape of the flow limiter). The combination of the two, gives a volume mesh with embedded data about the immersed boundary (characteristic function, normal vector, *etc.*). Let us notice that, for this case, the geometry of the immersed boundary is not a regular one, it involves some salient corners.

The case in itself corresponds to a Loss Of Coolant Accident induced by a break on the vessel cold inlet. Thus, as the primary circuit is pressurized, the flow comes from the bottom of the core, with a flowrate of 5.2 $\mathrm{kg\,s^{-1}}$ (this estimation is coming from system scale computations), to the break (*i.e.* the vessel inlet). The outflow pressure is fixed at an *ad hoc* value of 50 bar. For this case, a Homogeneous Equilibrium Model

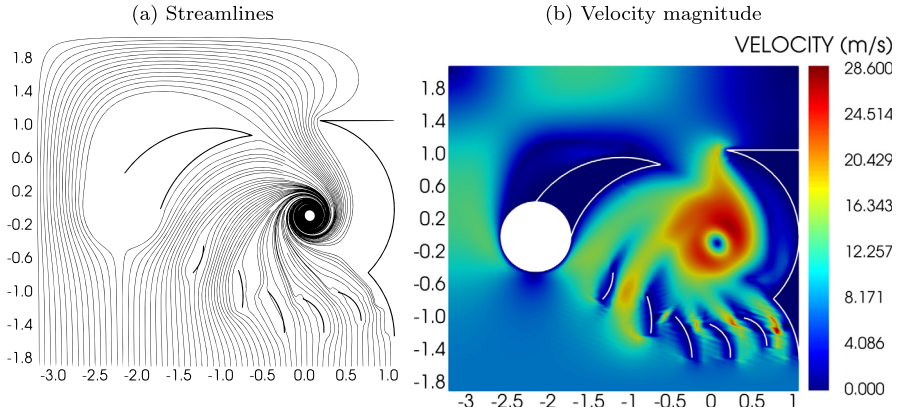

Fig. 11. Streamlines and velocity magnitude for the flow limiter case (converged state with 3,594,240 elements).

Table 6
Flow limiter case: values of the head loss coefficient K for different grids ($h = N^{-\frac{1}{3}}$) computed with the PDF method ("A": direct assignment, "B": directional interpolation, "C": multi-directional, "D": hybrid).

N	h (m)	K			
		A	B	C	D
6,080	5.48×10^{-2}	28.3	17.7	43.5	17.7
48,640	2.74×10^{-2}	13.8	14.6	15.8	14.4
164,160	1.83×10^{-2}	6.6	5.78	6.76	5.75
794,880	1.08×10^{-2}	5.7	5.45	5.65	5.48
3,594,240	6.53×10^{-3}	5.5	5.4	–	–

is used (with an inlet negative quality) as well as a Schlichting's scalar turbulence model with a characteristic length of turbulence $L_T = 0.3$ m. For a more detailed description of the case, the interested reader can refer to [8].

4.4.2. *Mesh convergence study and head loss coefficient*

We successfully ran simulations involving the 3D flow limiter [8], which demonstrates the generality and robustness of our method, as well as its capacity to deal with the aimed applications. First, the flow topology is respected as the creation of a vortex, supposed to delay core dewatering, occurs (*cf.* Figs. 11a and 11b). Second, we use the headloss coefficient K [8] to carry out a mesh convergence study over the different grids (with the solution computed on the finest grids considered as reference). The values of K are presented in Table 6. The results are consistent with the preliminary studies [8] and show that the directional interpolation tends to enhance the space convergence (*cf.* Fig. 12). Moreover, we can see that the values obtained with the directional interpolation are closer to the one obtained on the finest grid – this is encouraging for future turbulent wall law interpolation. What is also interesting to note is the threshold effect between 48,640 and 164,160 elements, clearly visible in Fig. 12. Indeed, with less than 164,160 elements, the space resolution is not sufficient to model the channel between the fins properly, leading to an overestimated head loss coefficient. Also, in this case, the results obtained with the hybrid interpolation are not exactly the same as the ones obtained with the directional interpolation (*cf.* Table 6) because the geometry is complex and non-regular at some locations. However, the hybrid approach does not show any significative improvement compared to the directional one.

In addition, some body-fitted simulations (with about 240,000 elements) have been carried out. The value of K obtained with the FE element discretization (resp. Volume Finite Element discretization) is 5.40 (resp. 4.61). Those values are very close to the one obtained with the PDF method but the meshes are much more difficult to build, which highlights the interest of our approach, especially in the scope of shape optimization.

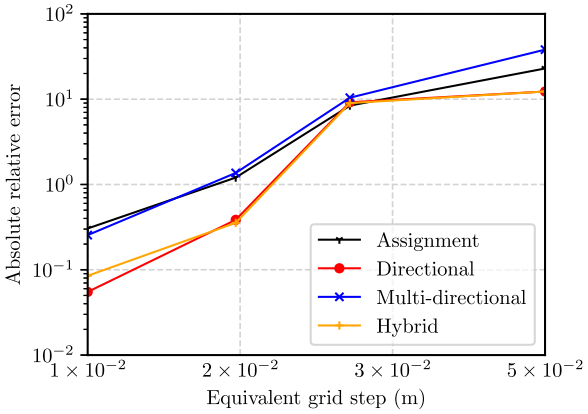

Fig. 12. Evolution of the head loss coefficient with respect to the grid step for the flow limiter case.

5. Conclusion and perspectives

In conclusion, we have introduced the context of this work, underlining the fact that the immersed boundary approach is more and more attractive for engineer studies, in particular considering the reduction of the meshing effort when the geometry is frequently modified. For instance, it is the case in fluid structure interaction or geometry optimization problems. Our work was focused on inflow thin complex obstacles with fluid on both sides, as we can find in the design of some innovative nuclear passive safety systems studied at CEA. Details about the original method developed by the authors to face this challenge and primary results were presented in [2]. This paper focuses specifically on different techniques to recover data from the discrete immersed boundary and different ways to achieve order 2 in space via linear interpolation.

The Penalized Direct Forcing method was briefly reminded, as well as the other numerical methods involved in the simulation tool (fractional step algorithm, FEM). However, additional data (such normal vector, normal projection, *etc.*) are needed to linearly interpolate field variables in the vicinity of the immersed obstacles in order to reach order 2 in space. Two different approaches, each including several variants, were considered: weighting and optimization. Results and performances for those two approaches were compared over analytical (cylinder) and quasi-industrial (NACA0012 airfoil) test cases. It turns out that the error values obtained are similar but the weighting approach offers much lower computation times.

Then, to reach second order in space, two different interpolation strategies were presented: one involving interpolation in the direction normal to the immersed obstacle and the other involving multi-directional interpolation. A hybrid strategy was proposed too.

Afterwards, all different interpolation techniques were tested over two cases. First, the laminar Taylor-Couette flow (for which the analytical solution is known) was used to carry out a mesh convergence study. It showed that all linear interpolation variant reduce the error while increasing the spatial order of convergence (reaching two for the velocity in $\mathcal{L}^2$ norm). Second, the laminar flow around a circular cylinder – declined in four configurations: steady regime with static cylinder, steady regime with rotating cylinder, unsteady regime with static cylinder and unsteady regime with rotating cylinder – was also used to carry out a mesh convergence study, even if no analytical solution is available (results obtained with finest grid considered as reference). All the linear interpolation techniques allow to reach order 2 in space. We also used experimental data and other simulations results as comparison for our method by the means of global quantities (mainly aerodynamic coefficients and Strouhal number). Although the results obtained with all interpolation techniques are similar, they still provide a great enhance when compared to the direct assignment variant (*i.e.* results are much closer to the reference values).

Finally, the capacity of our method to deal with the aimed application was demonstrated using a case representative of the flow limiter. Results are coherent with the preliminary study [8] but assessed a significative improvement in term of robustness. Also, this industrial test case tended to show that the directional interpolation approach provides the faster rate of convergence and lowest errors.

As perspectives, we are considering creating cases representative of other systems involving hydraulic diodes, for instance advanced accumulators [9]. Furthermore, we are currently developing a wall law (inspired from [19]) interpolation feature in order to deal with turbulence modeling. Finally, we could extend the method to Neumann boundary conditions or use the same linear interpolation techniques to deal with the pressure gradient in the prediction equation.

CRediT authorship contribution statement

Georis Billo: Conceptualization, Formal analysis, Methodology, Software, Writing – original draft. **Michel Belliard:** Conceptualization, Formal analysis, Methodology, Software, Writing – review & editing. **Pierre Sagaut:** Resources, Supervision, Writing – review & editing.

References

[1] C. Peskin, The immersed boundary method, Acta Numer. 11 (2002) 479–517.
[2] G. Billo, M. Belliard, P. Sagaut, A finite element penalized direct forcing immersed boundary method for infinitely thin obstacles in a dilatable flow, Comput. Math. Appl. 99 (2021) 292–304.
[3] M. Belliard, C. Fournier, Penalized direct forcing and projection schemes for Navier-Stokes, C. R. Acad. Sci. Paris, Ser. I 348 (2010) 1133–1136.
[4] C. Introïni, M. Belliard, C. Fournier, A second order penalized direct forcing for hybrid Cartesian/immersed boundary flow simulations, Comput. Fluids 90 (2014) 21–41.
[5] E. Fadlun, R. Verzicco, P. Orlandi, J. Mohd-Yusof, Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations, J. Comput. Phys. 161 (2000) 35–60.
[6] P. Angot, Analysis of singular perturbations on the Brinkman problem for fictitious domain of viscous flows, Math. Methods Appl. Sci. 22 (1999) 1395–1412.
[7] P. Angot, C. Bruneau, P. Fabrie, A penalization method to take into account obstacles in incompressible viscous flows, Numer. Math. 81 (1999) 497–520.
[8] M. Belliard, Numerical modeling of an in-vessel flow limiter using an immersed boundary approach, Nucl. Eng. Des. 330 (2018) 437–449.
[9] T. Shiraishi, Design of the advanced accumulator for the pressurized water reactor, Nucl. Eng. Des. 241 (2011) 3910–3924.
[10] D. Khabarova, A. Podzerko, E. Spiridonov, Experimental investigation of fluidic diodes, in: International Conference on Industrial Engineering, ICIE 2017, Proc. Eng. 206 (2017) 93–98.
[11] D.L. Brown, R. Cortez, M.L. Minion, Accurate projection methods for the incompressible Navier-Stokes equations, J. Comput. Phys. 168 (2001) 464–499.
[12] L. Manueco, P. Weiss, S. Deck, On the estimation of unsteady aerodynamic forces and wal spectral content with immersed boundary conditions, Comput. Fluids 201 (2020) 104471.
[13] C. Peskin, Flow Patterns around heart valves: a digital computer method for solving the equations of motion, Ph.D. thesis, Albert Einstein College of Medecine, 1972.
[14] A. Roma, C. Peskin, M. Berger, An adaptative version of the immersed boundary method, J. Comput. Phys. 153 (1999) 509–534.
[15] A. Baeza, P. Mulet, D. Zorío, High order boundary extrapolation technique for finite difference methods on complex domains with Cartesian meshes, J. Sci. Comput. 66 (2016) 761–791.
[16] C. Chi, A. Abdelsamie, D. Thévenin, A directional ghost-cell immersed boundary method for incompressible flows, J. Comput. Phys. 404 (2020) 593–623.
[17] G. Wagner, N. Moës, W. Liu, T. Belytschko, The extended finite element method for rigid particles in Stokes flow, Int. J. Numer. Methods Eng. 51 (2001) 293–313.
[18] A. Yazid, N. Abdelkader, H. Abdelmadjid, A state-of-the-art review of the X-FEM for computational fracture mechanics, Appl. Math. Model. 33 (2009) 4269–4282.
[19] S. Wilhelm, J. Jacob, P. Sagaut, An explicit power-law-based wall model for lattice Boltzmann method-Reynolds-averaged numerical simulations of the flow around airfoils, Phys. Fluids 30 (2018) 065111.
[20] M. Grandotto, P. Obry, Calculs des écoulements diphasiques dans les échangeurs par une méthode aux éléments finis, Rev. Eur. Éléments Finis 5 (1996) 53–74 (in French).
[21] R. Guy, D. Hertenstgine, On the accuracy of direct forcing immersed boundary methods with projection methods, J. Comput. Phys. 229 (2010) 2479–2496.
[22] E. Guyon, J. Hulin, L. Petit, Physical Hydrodynamics, Oxford University Press, 2001.
[23] J. Choi, R. Oberoi, J.R. Edwards, J.A. Rosati, An immersed boundary method for complex incompressible flows, J. Comput. Phys. 224 (2007) 757–784.
[24] M. Chung, Cartesian cut cell approach for simulating incompressible flows with rigid bodies of arbitrary shape, Comput. Fluids 35 (2006) 607–623.
[25] K. Taira, T. Colonius, The immersed boundary method: a projection approach, J. Comput. Phys. 225 (2007) 2118–2137.
[26] M. Linnick, H. Fasel, A high-order immersed interface method for simulating unsteady incompressible flows on irregular domains, J. Comput. Phys. 204 (2005) 157–192.
[27] B. Fornberg, A numerical study of steady viscous flow past a circular cylinder, J. Fluid Mech. 98 (1980) 819–855.
[28] D. Stojković, M. Breuer, F. Durst, Effect of high rotation rates on the laminar flow around a circular cylinder, Phys. Fluids 14 (2002) 3160–3178.
[29] D. Ingham, T. Tang, A numerical investigation into the steady flow past a rotating circular cylinder at low and intermediate Reynolds numbers, J. Comput. Phys. 87 (1990) 91–107.
[30] H. Badr, S. Dennis, P. Young, Steady and unsteady flow past a rotating circular cylinder at low Reynolds numbers, Comput. Fluids 17 (1989) 579–609.
[31] P. Chiu, R. Lin, T.W. Sheu, A differentially interpolated direct forcing immersed boundary method for predicting incompressible Navier-Stokes equations in time-varying complex geometries, J. Comput. Phys. 229 (2010) 4476–4500.
[32] Y. Cheny, O. Botella, The ls-stag method: a new immersed boundary/level-set method for the computation of incompressible viscous flows in complex moving geometries with good conservation properties, J. Comput. Phys. 229 (2010) 1043–1076.
[33] C. Ji, A. Munjiza, J. Williams, A novel iterative direct-forcing immersed boundary method and its finite volume applications, J. Comput. Phys. 231 (2012) 1797–1821.
[34] C. Norberg, Fluctuating lift on a circular cylinder: review and new measurements, J. Fluids Struct. 17 (2003) 57–96.
[35] S. Kang, H. Choi, S. Lee, Laminar flow past a rotating circular cylinder, Phys. Fluids 11 (1999) 3312–3321.