Reduction of the discretization stencil of direct forcing immersed boundary methods on rectangular cells: the ghost node shifting method.

Joris Picot[a], Stéphane Glockner[b]

[a] *Université de Bordeaux, I2M, UMR 5295, F-33400 Talence, France*
[b] *Bordeaux INP, I2M, UMR 5295, F-33400 Talence, France*

Abstract

We present an analytical study of discretization stencils for the Poisson problem and the incompressible Navier-Stokes problem when used with some direct forcing immersed boundary methods. This study uses, but is not limited to, second-order discretization and Ghost-Cell Finite-Difference methods. We show that the stencil size increases with the aspect ratio of rectangular cells, which is undesirable as it breaks assumptions of some linear system solvers. To circumvent this drawback, a modification of the Ghost-Cell Finite-Difference methods is proposed to reduce the size of the discretization stencil to the one observed for square cells, i. e. with an aspect ratio equal to one. Numerical results validate this proposed method in terms of accuracy and convergence, for the Poisson problem and both Dirichlet and Neumann boundary conditions. An improvement on error levels is also observed. In addition, we show that the application of the chosen Ghost-Cell Finite-Difference methods to the Navier-Stokes problem, discretized by a pressure-correction method, requires an additional interpolation step. This extra step is implemented and validated through well known test cases of the Navier-Stokes equations.

Keywords: Immersed boundary method, Direct forcing, Discretization stencil, Poisson problem, Incompressible Navier-Stokes, Boundary conditions

1. Introduction

The immersed boundary method has been proven a successful treatment of complex boundaries in numerical simulations. The difficult generation of body-fitted grids is replaced by a modification of the governing equations near the boundaries, thus benefits of structured grids can be exploited. In addition, the immersed boundary method is particularly attractive when dealing with moving boundaries as it avoids re-meshing. Many different variants have been developed since the original immersed boundary method [1] as they explored coupling with different problems, different regimes, etc. Now the best approach to choose depends on physics constraints one wants to model. The review of Mittal and Iaccarino [2] explains this in details. The Direct Forcing approach, adopted in this article, first discretizes governing equations into a linear system; then applies boundary conditions near immersed boundaries to complete the linear system. This approach is similar to ordinary boundary conditions, but the non-Cartesian immersed boundaries leads to more complex interpolations. Both finite-difference discretization and finite-volume discretization can be used to apply immersed boundary conditions. This article focuses on specific aspects of the Ghost-Cell Finite-Differences approach, in the versions proposed by Mittal et al. [3] and Coco and Russo [4], and applies them to the Poisson and Navier-Stokes problem, though this approach can be used to any boundary value problem.

Some linear system solvers constrain the matrix profile, i. e. cells allowed to be nonzero, to take advantage of properties that allow to achieve the highest performance in large parallel computations. For instance, geometric multi-grid algorithms SMG and PFMG of the *hypre* library can be used only with band matrices with 9/27 points stencils in 2D/3D for SMG [5, 6] and PFMG [7, 8]. More generic solvers and preconditioners such as the algebraic BoomerAMG of the *hypre* library are less efficient (see [9, 10, 11]) even if they are competitive. In addition, band matrices requires less memory than more generic sparse matrix format, such as Compressed Row Storage, because their topology do not need to be stored for each line of the matrix.

The Ghost-Cell Finite-Differences methods proposed by Mittal et al. [3] and Coco and Russo [4] do not fit in band matrices generated by the discretization of the equations, whose stencil is broken for rectangular cells. These methods are based on the "closest point of the boundary" (as explained in Sec. 2.2) which does not take into account the properties of the numerical grid. This article proposes a modification of these methods to adapt them on band matrices in the case of rectangular cells. This article is organized as follows: the two Ghost-Cell Finite-Differences methods [3, 4] are presented in Sect. 2 through the Poisson problem. Their impact on the discretization stencil is studied in Sect. 3, and the Ghost Node Shifting Method is defined in Sect. 4. Numerical simulations validate the proposed method in Sect. 5 for the Poisson problem with Dirichlet and Neumann boundary conditions. Finally, the Ghost-Cell Finite-Differences methods are introduced into the incompressible Navier-Stokes equations in Sect. 6 and some validations are presented in Sect. 7.

2. Immersed boundary methods with the Poisson problem

This section presents the Finite-Differences Ghost-Fluid immersed boundary methods applied to the Poisson problem, as found in the literature. For the sake of simplicity, methods are described in the 2-dimensional real space.

2.1. Problem description and discretization

The *computational domain* Ω is a rectangle of the 2-dimensional real space. It is split in two sub-domains by the *immersed boundary* Γ: one is the *inner part* Ω_i, while the other is the *outer part* Ω_o. The boundary Γ is oriented by a unit vector $\boldsymbol{n}$ from the outer domain to the inner domain. The *domain boundary* $\partial\Omega$ decomposes in four line segments $\partial\Omega_b$, $\partial\Omega_r$, $\partial\Omega_t$, $\partial\Omega_l$, where the letters b, r, t, l stand for bottom, right, top, left, respectively. These definitions are illustrated in Fig. 1a.

Poisson problem. We consider $u : \Omega_i \to \mathbb{R}$, a scalar field restricted to the inner domain, solution of the Poisson equation with a known *source field* $f : \Omega_i \to \mathbb{R}$:

$$\Delta u = f\,, \qquad \text{in } \Omega_i\,, \tag{1a}$$

$$u = D\,, \qquad \text{on } \partial\Omega_D \cup \Gamma_D\,, \tag{1b}$$

$$\partial_{\boldsymbol{n}} u = N\,, \qquad \text{on } \partial\Omega_N \cup \Gamma_N\,. \tag{1c}$$

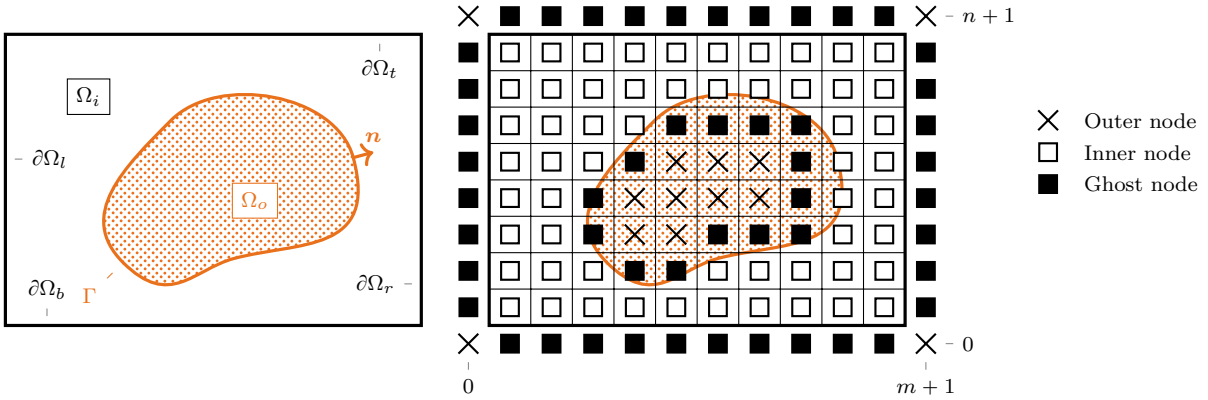
Here boundaries have been split into a "Dirichlet" part $\partial\Omega_D \cup \Gamma_D$ and a "Neumann" part $\partial\Omega_N \cup \Gamma_N$; boundary conditions $D : \Gamma \to \mathbb{R}$ and $N : \Gamma \to \mathbb{R}$ are known.

Numerical discretization. The computational domain is partitioned in *cells* into a Cartesian arrangement of size $m \times n$. The cell boundaries do not conform with Γ. The partitioning is *uniform,* i. e. the *cell sizes* h_x, h_y do not depend on the cell index; however, the cells can be either *square*: $h_x = h_y$, or *rectangular*: $h_x \neq h_y$. A row of cells is added across each domain boundary to handle the domain boundary conditions; hence the total grid size is $(m + 2) \times (n + 2)$. Finally, each cell of index (i, j) is associated to a *cell node* $\boldsymbol{X}_{i,j}$, with $0 \leqslant i \leqslant m + 1$ and $0 \leqslant j \leqslant n + 1$. Figure 1b shows the discretization of Ω.

We assign types to nodes to know on which side of the immersed boundary the node is, and to apply different equations on them. The *cell type* of the cell node $\boldsymbol{X}_{i,j}$ is

$$\begin{aligned}
&1.\ \text{if } \boldsymbol{X}_{i,j} \in \Omega_i, \text{ then } \boldsymbol{X}_{i,j} \text{ is an } \textit{inner node;}\\
&2.\ \text{else, if } \{\boldsymbol{X}_{i-1,j}, \boldsymbol{X}_{i+1,j}, \boldsymbol{X}_{i,j-1}, \boldsymbol{X}_{i,j+1}\} \cap \Omega_i \neq \varnothing, \text{ then } \boldsymbol{X}_{i,j} \text{ is a } \textit{ghost node;}\\
&3.\ \text{else, } \boldsymbol{X}_{i,j} \text{ is an } \textit{outer node.}
\end{aligned} \tag{2}$$

Figure 1b also illustrates how node types are defined. As explained below, the ghost node in Eq. (2) represents nodes that will need a special treatment to close the linear system. By extension: if $\boldsymbol{X}_{i,j}$ is an inner node (respectively, an ghost node, an outer node), then the cell at (i, j) is called an *inner cell*



(a) Example of computational domain Ω split in an inner part Ω_i and an outer part Ω_o by the immersed boundary Γ.

(b) Example of Cartesian grid of resolution $m \times n$ with domain boundary extensions build upon Example Fig. 1a. Different symbols are used to distinguish inner, ghost, and outer cell nodes.

Figure 1: Definitions and notations of the computational domain, the immersed boundary, the mesh, and node types.

(respectively, a *ghost cell*, an *outer cell*). Finally, we define the *set of inner cell indices* C_i, the *set of ghost cell indices* C_g, and the *set of outer cell indices* C_o, accordingly.

Equation (1a) discretizes at each inner node using standard, second-order finite differences

$$\frac{U_{i-1,j} - U_{i,j}}{h_x^2} + \frac{U_{i+1,j} - U_{i,j}}{h_x^2} + \frac{U_{i,j-1} - U_{i,j}}{h_y^2} + \frac{U_{i,j+1} - U_{i,j}}{h_y^2} = F_{i,j} + \mathcal{O}(h^2)\,, \qquad \forall (i,j) \in C_i\,, \tag{3}$$

where $h = \max\{h_x, h_y\}$; F and U are *discrete fields* of size of size $(m+2)\times(n+2)$ such as $F_{i,j} = f(\boldsymbol{X}_{i,j})$ and $U_{i,j} = u(\boldsymbol{X}_{i,j})$ at inner cells. Note that we choose $U_{i,j}$ to be the field value of the *exact* solution at $\boldsymbol{X}_{i,j}$ of the continuous equation Eq. (1); the truncation term $\mathcal{O}(h^2)$ in Eq. (3) accounts for the difference between finite differences and exact derivatives. We also write Eq. (3) in the matrix form $(LU)_{i,j} = F_{i,j} + \mathcal{O}(h^2), \forall (i,j) \in C_i$. Equation 3 is not closed as there are some $U_{i-1,j}$, $U_{i+1,j}$, $U_{i,j-1}$, or $U_{i,j+1}$ that do not correspond to inner nodes; these nodes correspond exactly to the ghost nodes, for which a value *must* be defined. For ghost node values behind the domain boundaries, we use standard boundary conditions, which gives on the left boundary $\partial\Omega_l$

$$\frac{U_{0,j} + U_{1,j}}{2} = D_l(\boldsymbol{X}_{\frac{1}{2},j}) + \mathcal{O}(h^2)\,, \qquad \forall j \in [\![1,n]\!] \cap \partial\Omega_D\,, \tag{4a}$$

$$\frac{U_{1,j} - U_{0,j}}{h} = N_l(\boldsymbol{X}_{\frac{1}{2},j}) + \mathcal{O}(h^2)\,, \qquad \forall j \in [\![1,n]\!] \cap \partial\Omega_N\,. \tag{4b}$$

For ghost node values behind the immersed boundary, similar boundary conditions can be derived from Eq. (1b-1c) that also takes into account the geometry of the immersed boundary, and is detailed in Sect. 2.2. Boundary conditions of all boundaries can be written in matrix form $(EU)_{i,j} = B_{i,j} + R_{i,j}, \forall (i,j) \in C_g$, where B contains boundary condition values at the boundary, such as $D_l(\boldsymbol{X}_{\frac{1}{2},j})$ in Eq. (4a). The truncation term R will be refined in Sect. 2.2. Finally, both the discretized Poisson equation Eq. (3) and boundary conditions are assembled into the closed linear system

$$\big((L+E)U\big)_{i,j} = F_{i,j} + B_{i,j} + \mathcal{O}(h^2) + R_{i,j}\,, \qquad \forall (i,j) \in C_{ig}\,, \tag{5}$$

where $C_{ig} = C_i \cup C_g$. To write Eq. (5), we consider that $(LU)_{i,j} = F_{i,j} = 0$ over C_g and $(EU)_{i,j} = B_{i,j} = 0$ over C_i. This methodology can be applied to other problems by adding the part $EU = B + R$ to the considered equation. An example is given in Eq. (52a) of this article with the momentum equation.

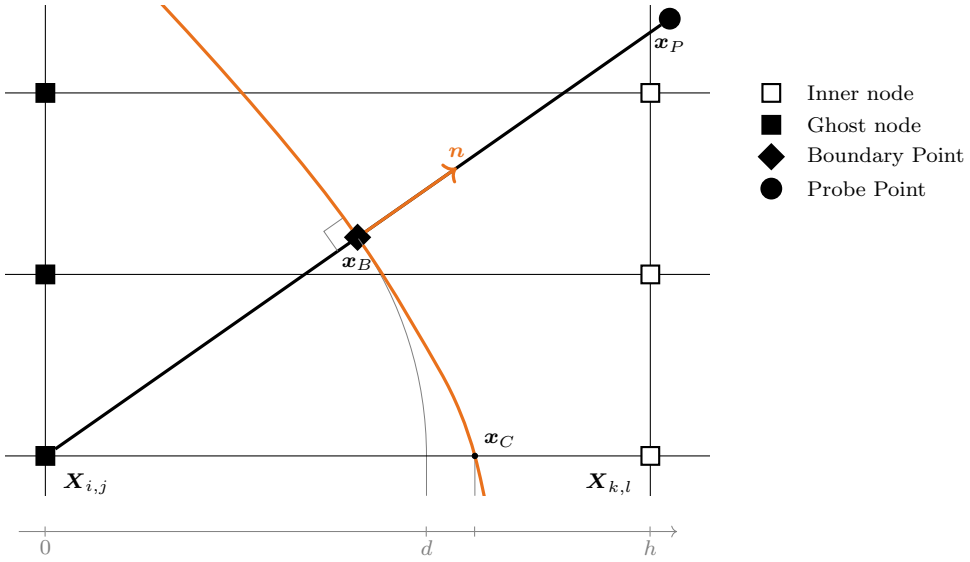


Figure 2: Sketch defining the boundary point $\boldsymbol{x}_B$ and the probe point $\boldsymbol{x}_P$ with respect to the ghost node $\boldsymbol{X}_{i,j}$. The boundary point is the closest point of Γ with respect to $\boldsymbol{X}_{i,j}$, thus $[\boldsymbol{X}_{i,j}, \boldsymbol{x}_B]$ is aligned with $\boldsymbol{n}$. As $\boldsymbol{X}_{k,l}$ is an inner node, $\boldsymbol{x}_C$ is always defined. The axis at the bottom illustrates that $|\boldsymbol{x}_B - \boldsymbol{X}_{i,j}| \leqslant |\boldsymbol{x}_C - \boldsymbol{X}_{i,j}| < |\boldsymbol{X}_{k,l} - \boldsymbol{X}_{i,j}|$.

Numerical simulations can be performed by dropping the truncation terms

$$\big((L+E)\hat{U}\big)_{i,j} = F_{i,j} + B_{i,j}\,, \qquad \forall (i,j) \in C_{ig}\,, \tag{6}$$

where $\hat{U}$ is the *numerical solution*. The discretization of the Poisson equation prevents ourselves from observing better than *second-order limiting behavior,* although this limitation is not due *a priori* by the immersed boundary method.

Implemented linear system arrays still have placeholders for outer node values due to the structured formulation, but their values should have no influence on the other node values. It is useful to set outer node values to a large value by setting both the diagonal and the right-hand side to, say, 10^{10}, as it helps bug detection in the implementation.

The numerical simulations presented in this document resort on the direct linear system solver *mumps* [12, 13] or the iterative linear system solvers of *hypre* [10, 14]. In the later case, we used GMRES as solver and SMG or BoomerAMG as preconditioner, unless specified otherwise.

2.2. Immersed boundary conditions

In this section, we present two different definitions of immersed boundary conditions that were introduced by Mittal et al. [3], the *linear method* in this paper, and by Coco and Russo [4], the *direct method*.

Let us consider a ghost node $\boldsymbol{X}_{i,j}$ next to some part of the immersed boundary Γ. We define the *boundary point* $\boldsymbol{x}_B$ as the closest point of Γ with respect to $\boldsymbol{X}_{i,j}$; the scalar d is the distance between $\boldsymbol{x}_B$ and the ghost node, and the unit vector $\boldsymbol{n}$ defined in Sect. 2.1 happens to be the direction of $\boldsymbol{x}_B$ from the ghost node. We also define the *probe point* $\boldsymbol{x}_P$ as the symmetric of $\boldsymbol{X}_{i,j}$ with respect to $\boldsymbol{x}_B$. Figure 2 illustrates these definitions.

The method used to obtain $\boldsymbol{x}_B$, $\boldsymbol{x}_P$, d, and $\boldsymbol{n}$ depend on how the immersed boundary is defined (by a parametrization, a level-set field, etc.). Immersed boundaries used in the numerical simulations presented in this document uses exact parametrizations or piecewise linear curves such as the variables can be evaluated up to machine epsilon.

2.2.1. Linear method [3]

The linear method builds boundary conditions using finite-differences along the *linear* segment $[\boldsymbol{X}_{i,j}, \boldsymbol{x}_P]$:

$$\frac{U_{i,j} + u_P}{2} = u_B + \mathcal{O}(d^2)\,, \tag{7a}$$

$$\frac{u_P - U_{i,j}}{2d} = \partial_{\boldsymbol{n}} u_B + \mathcal{O}(d^2)\,, \tag{7b}$$

where $u_P = u(\boldsymbol{x}_P)$, $u_B = u(\boldsymbol{x}_B)$, and $\partial_{\boldsymbol{n}} u_B = \partial_{\boldsymbol{n}} u(\boldsymbol{x}_B)$ are the *exact* field values and derivatives at the corresponding points. The right-hand side term $\mathcal{O}(d^2)$ is equivalent to $\mathcal{O}(h^2)$. Indeed, according to the definition of a ghost node (Eq. 2) there exists an adjacent inner node $\boldsymbol{X}_{k,l}$, so there is an intersection point $\boldsymbol{x}_C$ between Γ and the line segment $[\boldsymbol{X}_{i,j}, \boldsymbol{X}_{k,l}]$; see Fig. 2. And, as $\boldsymbol{x}_B$ is the closest point of Γ from $\boldsymbol{X}_{i,j}$,

$$d = \|\boldsymbol{x}_B - \boldsymbol{X}_{i,j}\|_2 \leqslant \|\boldsymbol{x}_C - \boldsymbol{X}_{i,j}\|_2 < h. \tag{8}$$

The left-hand side Eq. (7) must use only cell node values to fit in the linear system Eq. (5), so the value u_P is expanded as a p-th order interpolation of surrounding node values:

$$u_P = \sum_{(k,l)\in I_P} \rho_{k,l} U_{k,l} + \mathcal{O}(h^p)\,, \tag{9}$$

where *interpolation coefficients* $\rho_{k,l}$ and the *set of interpolation indices* I_P will be defined in Sect. 2.3. The boundary conditions Eq. (1b-1c) can thus be discretized as

$$\tfrac{1}{2} U_{i,j} + \sum_{(k,l)\in I_P} \tfrac{1}{2} \rho_{k,l} U_{k,l} = D(\boldsymbol{x}_B) + \mathcal{O}(h^2) + \mathcal{O}(h^p)\,, \qquad \forall (i,j) \in C_g \cap \partial\Omega_D\,, \tag{10a}$$

$$\sum_{(k,l)\in I_P} \tfrac{1}{2d} \rho_{k,l} U_{k,l} - \tfrac{1}{2d} U_{i,j} = N(\boldsymbol{x}_B) + \mathcal{O}(h^2) + \mathcal{O}(h^{p-1})\,, \qquad \forall (i,j) \in C_g \cap \partial\Omega_N\,. \tag{10b}$$

A numerical approximation of the Poisson problem can be build using the linear method Eq. (10) to discretize the Dirichlet and Neumann boundary conditions. A second-order limiting behavior is obtained when *second-order interpolations* are used with Dirichlet boundary conditions, and when *third-order interpolations* are used with Neumann boundary conditions.

2.2.2. Direct method [4]

In this method, the probe point $\boldsymbol{x}_P$ is not used and the field value at the boundary point u_B is *directly* expanded as a p-th order interpolation, which gives for the Dirichlet boundary condition

$$\sum_{(k,l)\in I_B} \beta_{k,l} U_{k,l} = u_B + \mathcal{O}(h^p)\,, \tag{11}$$

where interpolation coefficients $\beta_{k,l}$ and the set of interpolation indices I_B will also be defined in Sect. 2.3. For the Neumann boundary condition, the field gradient can be obtained through algebraic differentiation of the interpolation polynomial:

$$\nabla u_B = \sum_{(k,l)\in I_B} (\partial_x \beta_{k,l}, \partial_y \beta_{k,l}) U_{k,l} + \mathcal{O}(h^{p-1})\,, \tag{12}$$

where $\partial_x \beta_{k,l}$ and $\partial_y \beta_{k,l}$ are the interpolation coefficients of the polynomial gradient. Using $\partial_{\boldsymbol{n}} u_B = \nabla u_B \cdot \boldsymbol{n}$, the boundary conditions Eq. (1b-1c) can thus be discretized as

$$\sum_{(k,l)\in I_B} \beta_{k,l} U_{k,l} = D(\boldsymbol{x}_B) + \mathcal{O}(h^p)\,, \qquad \forall (i,j) \in C_g \cap \partial\Omega_D\,, \tag{13a}$$

$$\sum_{(k,l)\in I_B} (\partial_x \beta_{k,l} n_x + \partial_y \beta_{k,l} n_y) U_{k,l} = N(\boldsymbol{x}_B) + \mathcal{O}(h^{p-1})\,. \qquad \forall (i,j) \in C_g \cap \partial\Omega_N\,. \tag{13b}$$

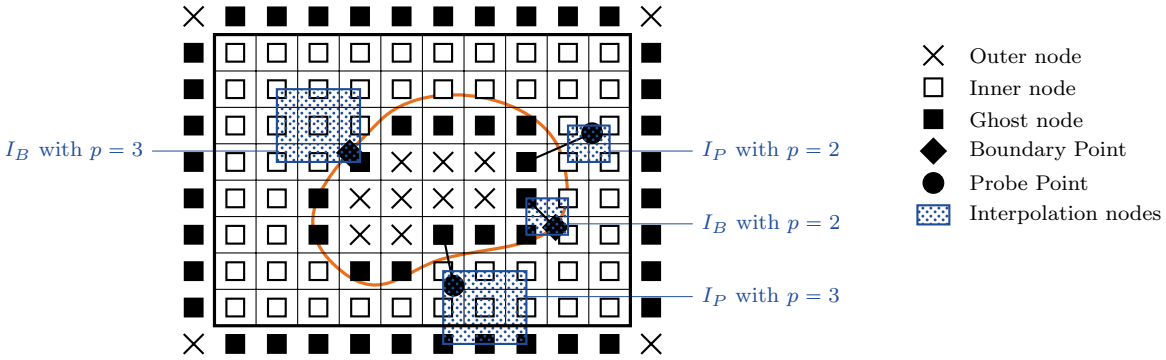


Figure 3: Examples of sets of interpolation nodes for the grid sketched in Fig. 1b. At four ghost nodes, the boundary or point is shown, and the corresponding set of interpolation nodes is shown with a blue dot-filled rectangle.

A numerical approximation of the Poisson problem can be build using the direct method Eq. (13) to discretize the Dirichlet and Neumann boundary conditions. A second-order limiting behavior is obtained when *second-order interpolation* are used with Dirichlet boundary conditions, and when *third-order interpolation* are used with Neumann boundary conditions.

2.3. Interpolation

Interpolations introduced in Eq. (9, 11) are the last part to define the immersed boundary methods. Any interpolation method can be used as long as two constraints are meet:

$$\begin{aligned} &1.\ \text{the set of interpolation nodes } \textit{must not} \text{ include any outer node,} \\ &2.\ \text{Equation (9, 11) } \textit{must} \text{ include the ghost node value } U_{i,j}. \end{aligned} \tag{14}$$

The first constraint ensures that the linear system Eq. (5) is closed because no values are defined on outer nodes. The second constraint ensures that $U_{i,j}$ is defined by Eq. (9, 11). We will see later that this is not always the case for the direct method on rectangular cells.

This article uses Lagrange polynomials over the Cartesian set of nodes $\{ \boldsymbol{X}_{k,l} = (X_k, Y_l)^\intercal \mid (k,l) \in I_\chi \}$, with $\chi = P$ for the probe point (Eq. 9) and $\chi = B$ for the boundary point (Eq. 11). The set of interpolation indices I_χ is be defined by the indices of the two opposite corners of the Cartesian set. The first index (i_c, j_c) corresponds to the node surrounding $\boldsymbol{x}_\chi$ in the opposite direction given by $\boldsymbol{n}$, and the second index is $p-1$ nodes away from (i_c, j_c) in the direction given by $\boldsymbol{n}$. Using $\boldsymbol{n}$ extends X_χ towards the inner domain to conform to the first constraint. The formal definition of I_χ is as follows, and Fig. 3 shows examples:

- when $\boldsymbol{n} = (n_x, n_y)^\intercal$ is oriented to the top right *quadrant*, i. e. $n_x \geqslant 0$ and $n_y \geqslant 0$, then

$$i_c = \max\{\, k \mid X_k < x_\chi \}\,, \quad j_c = \max\{\, l \mid Y_l < y_\chi \}\,, \quad I_\chi = [\![i_c, i_c + p - 1]\!] \times [\![j_c, j_c + p - 1]\!]\,; \tag{15a}$$

- when $\boldsymbol{n}$ is oriented to the top left quadrant, i. e. $n_x < 0$ and $n_y \geqslant 0$, then

$$i_c = \min\{\, k \mid x_\chi < X_k \}\,, \quad j_c = \max\{\, l \mid Y_l < y_\chi \}\,, \quad I_\chi = [\![i_c - p + 1, i_c]\!] \times [\![j_c, j_c + p - 1]\!]\,; \tag{15b}$$

- when $\boldsymbol{n}$ is oriented to the bottom left quadrant, i. e. $n_x < 0$ and $n_y < 0$, then

$$i_c = \min\{\, k \mid x_\chi < X_k \}\,, \quad j_c = \min\{\, l \mid y_\chi < Y_l \}\,, \quad I_\chi = [\![i_c - p + 1, i_c]\!] \times [\![j_c + p - 1, j_c]\!]\,; \tag{15c}$$

- and when $\boldsymbol{n}$ is oriented to the bottom right quadrant, i. e. $n_x \geqslant 0$ and $n_y < 0$, then

$$i_c = \max\{\, k \mid X_k < x_\chi \}\,, \quad j_c = \min\{\, l \mid y_\chi < Y_l \}\,, \quad I_\chi = [\![i_c, i_c + p - 1]\!] \times [\![j_c + p - 1, j_c]\!]\,. \tag{15d}$$

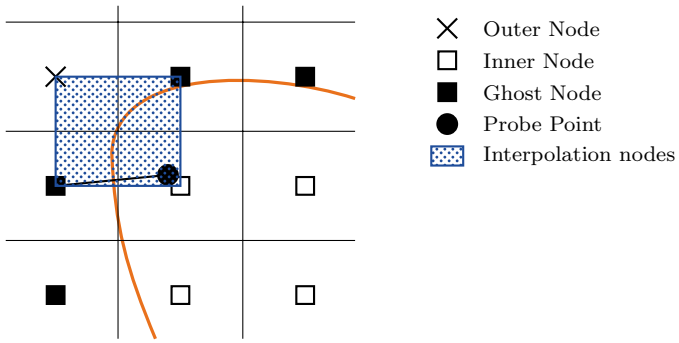


Figure 4: Example of failing interpolation regions for both the linear and direct methods. Notice that this can happen only if the immersed boundary is curved towards the inner domain over a region of two cells width, i. e. the boundary is *hollow*.

This definition and Fig. 3 shows that the region delimited by I_χ always contains $\boldsymbol{x}_\chi$, hence we always have interpolation and never extrapolation. Extending I_χ towards the direction given by $\boldsymbol{n}$ avoids I_χ to contain outer node indices. It is authorized to have other ghost nodes indices in I_χ as values at these nodes are defined by their corresponding immersed boundary condition. This results in couplings between the immersed boundary conditions that will be solved while solving the linear system Eq. (5).

However, even though Eq. (15) is designed to avoid outer nodes, one can still find a configuration where I_χ contains some outer nodes; see Fig 4 for examples. This occurs near parts of the immersed boundary that are *hollow,* i. e. Γ is curved towards the inner domain, over a range of two cells width; this issue is called the *hollow case problem.* Different solutions exist in the literature: Coco and Russo [4] stretches the discretization stencil to fit into the local inner domain and to reach enough inner nodes for the desired accuracy; Mousel [15] takes all nodes in a broader region in which there are more than enough inner nodes, and uses a least-squares algorithm to determine the interpolation coefficients. Using any of these methods increases the size of the discretization stencil as they use inner node values that are farther away from the ghost node. We do not use these methods in this article as we aim to keep the size of the discretization stencil tight. Instead, we resort on increasing the grid resolution as, for continuously differentiable immersed boundaries, the curvature with respect to the grid resolution decreases.

The second constraint is always meet for the linear method as $U_{i,j}$ is directly present in the expression Eq. 10. This is not the case for the direct method (Eq. 13), as $U_{i,j}$ is present through the interpolation only, which may not contain (i, j). Hopefully the ghost node correspond to the first corner node of I_B on square cells: $|x_B - X_i| < h = |X_{i+1} - X_i|$ and $|y_B - Y_j| < h = |Y_{j+1} - Y_j|$, so we have $(i, j) = (i_c, j_c)$, which ensures that the direct method always involves the ghost cell value. This is not true anymore for rectangular cells, see examples on Fig. 6 and Fig. 7 below.

3. Analysis of the discretization stencil

This section analyses the discretization stencil of the immersed boundary method introduced in Sect. 2 in the general case of rectangular cells.

3.1. Stencil size measurement

The matrix of the linear system that solves the Poisson problem with immersed boundaries Eq. (6) expands

$$(LU)_{i,j} = \sum_{(k,l)\in C_{ig}} L_{i,j}^{k,l} U_{k,l}\,, \qquad \forall (i,j) \in C_i\,, \tag{16}$$

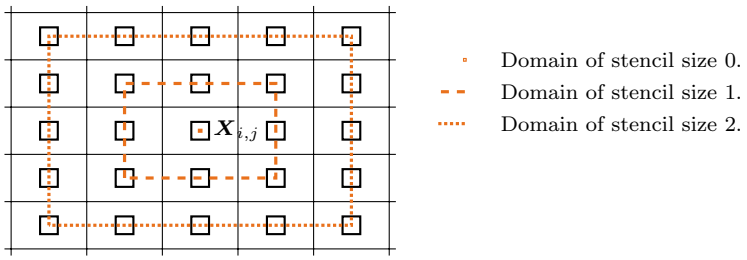


Figure 5: Domains of stencil sizes 0, 1, 2 around $\boldsymbol{X}_{i,j}$. The domain of stencil size 0 is reduced to a single point $\{\boldsymbol{X}_{i,j}\}$.

where $L_{i,j}^{k,l}$ are the scalar coefficients obtained from the discretization of the Poisson equation Eq. (3). Notice this expansion also applies for boundary conditions Eq. (10) or Eq. (13), i. e. over C_g, defining the scalar coefficients $E_{i,j}^{k,L}$. For an easier reading, the following does not write down expressions for E. The *stencil* of Eq. (16) is the set of indices $(k-i, l-j)$ whose coefficient $L_{i,j}^{k,l}$ is not zero. The *stencil size* $c_{i,j}$ of Eq. (16) is

$$c_{i,j} = \max\left\{ |k-i|, |l-j| \mid \forall (k,l) \in C_{ig}, L_{i,j}^{k,l} \neq 0 \right\}, \qquad \forall (i,j) \in C_i\,. \tag{17}$$

The expression "compact stencil" used in the literature means $c = 1$, and "non-compact stencil" means $c > 1$. From a metric perspective, the stencil size can be obtained with the *stencil norm* of $\boldsymbol{x} = (x, y)^\intercal$, defined as

$$\|\boldsymbol{x}\|_{\mathrm{c}} = \max\left\{ \frac{|x|}{h_x}, \frac{|y|}{h_y} \right\}. \tag{18}$$

The proof that $\boldsymbol{x} \mapsto \|\boldsymbol{x}\|_{\mathrm{c}}$ being a norm comes easily with the equivalence property

$$\frac{1}{h}\|\boldsymbol{x}\|_\infty \leqslant \|\boldsymbol{x}\|_{\mathrm{c}} \leqslant \frac{1}{\mu}\|\boldsymbol{x}\|_\infty\,, \tag{19}$$

where $h = \max\{h_x, h_y\}$ and $\mu = \min\{h_x, h_y\}$. Noticing the equalities $X_k - X_i = h_x(k-i)$ and $Y_l - Y_j = h_y(l-j)$, the stencil size can be equivalently defined as

$$c_{i,j} = \max\left\{ \|\boldsymbol{X}_{k,l} - \boldsymbol{X}_{i,j}\|_{\mathrm{c}} \mid \forall (k,l) \in C_i \cup C_g, L_{i,j}^{k,l} \neq 0 \right\}, \qquad \forall (i,j) \in C_{ig}\,, \tag{20}$$

that is the "grid" distance of the farthest node from $\boldsymbol{X}_{i,j}$ with $L_{i,j}^{k,l} \neq 0$. The region $\left\{ \boldsymbol{x} \mid \|\boldsymbol{x} - \boldsymbol{X}_{i,j}\|_{\mathrm{c}} \leqslant c_{i,j} \right\}$ is the *domain of stencil size* $c_{i,j}$. Figure 5 presents examples of such domains.

3.2. Stencil size for the Poisson problem

Let us review the stencil size for equations associated with each node types: at the inner nodes, the stencil size for the five-point stencil of the classical discretization of the Poisson equation Eq. (3) is 1; at the domain boundary conditions, Eq. (4) readily shows that the stencil size for these equations is also 1 for both Dirichlet and Neumann conditions. Finally, the stencil size for immersed boundary conditions Eq. (10) and Eq. (13) is determined by the farthest node included in I_χ, $\chi = P, B$, that is the *opposite corner node* defined in Sect. 2.3. Considering the ghost cell (i, j), we now determine the associated stencil size $c_{i,j}$:

- when $\boldsymbol{n}$ points to the top right quadrant, the opposite corner node is $(i_c + p - 1, j_c + p - 1)$ and

$$c_{i,j} = \|\boldsymbol{X}_{i_c+p-1,j_c+p-1} - \boldsymbol{X}_{i,j}\|_{\mathrm{c}} = \|\boldsymbol{X}_{i_c+1,j_c+1} - \boldsymbol{X}_{i,j}\|_{\mathrm{c}} + p - 2\,; \tag{21a}$$

- when $\boldsymbol{n}$ points to the top left quadrant, the opposite corner node is $(i_c - p + 1, j_c + p - 1)$ and

$$c_{i,j} = \|\boldsymbol{X}_{i_c-p+1,j_c+p-1} - \boldsymbol{X}_{i,j}\|_{\mathrm{c}} = \|\boldsymbol{X}_{i_c-1,j_c+1} - \boldsymbol{X}_{i,j}\|_{\mathrm{c}} + p - 2\,; \tag{21b}$$

- when $\boldsymbol{n}$ points to the bottom left quadrant, the opposite corner node is $(i_c - p + 1, j_c - p + 1)$ and

$$c_{i,j} = \|\boldsymbol{X}_{i_c-p+1,j_c-p+1} - \boldsymbol{X}_{i,j}\|_{\mathrm{c}} = \|\boldsymbol{X}_{i_c-1,j_c-1} - \boldsymbol{X}_{i,j}\|_{\mathrm{c}} + p - 2\,; \tag{21c}$$

- when $\boldsymbol{n}$ points to the bottom right quadrant, the opposite corner node is $(i_c + p - 1, j_c - p + 1)$ and

$$c_{i,j} = \|\boldsymbol{X}_{i_c+p-1,j_c-p+1} - \boldsymbol{X}_{i,j}\|_{\mathrm{c}} = \|\boldsymbol{X}_{i_c+1,j_c-1} - \boldsymbol{X}_{i,j}\|_{\mathrm{c}} + p - 2\,. \tag{21d}$$

The determination of $c_{i,j}$ is then deduced from the case $p = 2$ where I_χ contains just the four nodes surrounding $\boldsymbol{x}_\chi$;

- when $\boldsymbol{n}$ points to the right

$$x_\chi \leqslant X_{i_c+1} < x_\chi + h_x, \qquad \text{leading to} \qquad \frac{|X_{i_c+1} - X_i|}{h_x} = \left\lceil \frac{|x_\chi - X_i|}{h_x} \right\rceil, \tag{22a}$$

- when $\boldsymbol{n}$ points to the left

$$x_\chi - h_x \leqslant X_{i_c-1} < x_\chi, \qquad \text{leading to} \qquad \frac{|X_{i_c-1} - X_i|}{h_x} = \left\lceil \frac{|x_\chi - X_i|}{h_x} \right\rceil, \tag{22b}$$

- when $\boldsymbol{n}$ points to the top

$$y_\chi \leqslant Y_{j_c+1} < y_\chi + h_y, \qquad \text{leading to} \qquad \frac{|Y_{j_c+1} - Y_j|}{h_y} = \left\lceil \frac{|y_\chi - y_j|}{h_y} \right\rceil, \tag{22c}$$

- when $\boldsymbol{n}$ points to the bottom

$$y_\chi - h_y \leqslant Y_{j_c-1} < y_\chi, \qquad \text{leading to} \qquad \frac{|Y_{j_c-1} - Y_j|}{h_y} = \left\lceil \frac{|y_\chi - Y_j|}{h_y} \right\rceil, \tag{22d}$$

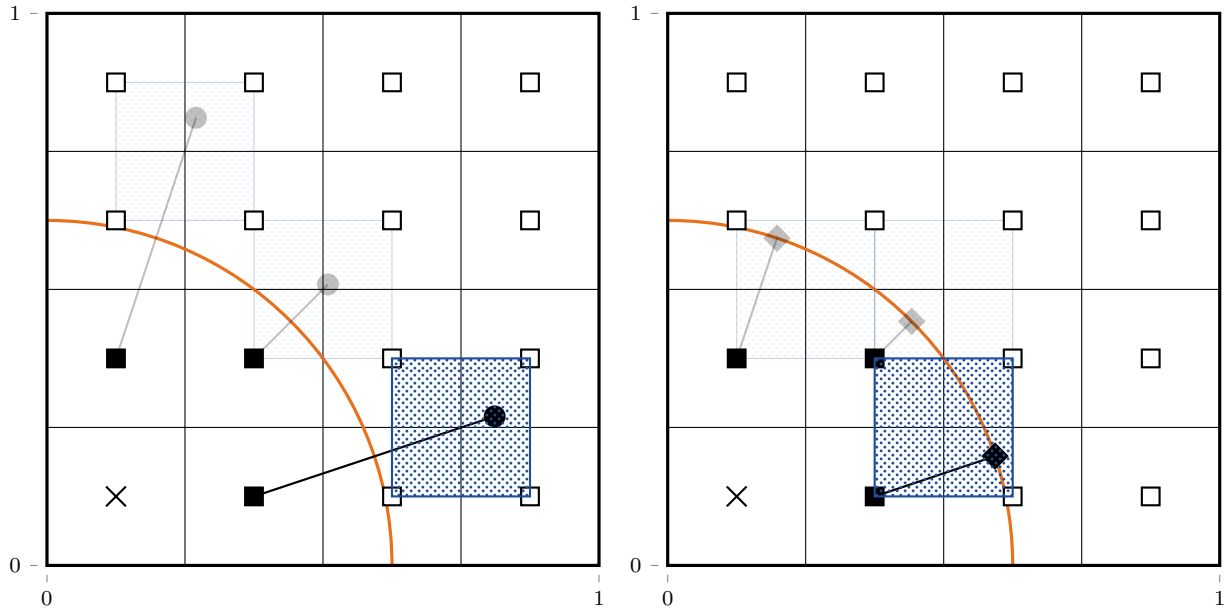
where $x \mapsto \lceil x \rceil$ is the *ceiling operator*. Therefore, we obtain for any direction of $\boldsymbol{n}$

$$c_{i,j} = \lceil \|\boldsymbol{x}_\chi - \boldsymbol{X}_{i,j}\|_{\mathrm{c}} \rceil + p - 2\,. \tag{23}$$

We can now deduce $c_{i,j}$ from the position of $\boldsymbol{x}_\chi$ with respect to $\boldsymbol{X}_{i,j}$. Equation (8) shows that $\|\boldsymbol{x}_\chi - \boldsymbol{X}_{i,j}\|_2$ is bounded by the cell size. We can use this property again to find a useful approximation of Eq. (23) with the help of the equivalence property Eq. (19)

$$\|\boldsymbol{x}_B - \boldsymbol{X}_{i,j}\|_{\mathrm{c}} \leqslant \frac{1}{\mu}\|\boldsymbol{x}_B - \boldsymbol{X}_{i,j}\|_\infty \leqslant \frac{1}{\mu}\|\boldsymbol{x}_B - \boldsymbol{X}_{i,j}\|_2 \leqslant \frac{h}{\mu}\,, \quad \|\boldsymbol{x}_P - \boldsymbol{X}_{i,j}\|_{\mathrm{c}} \leqslant 2\|\boldsymbol{x}_B - \boldsymbol{X}_{i,j}\|_{\mathrm{c}} \leqslant 2\frac{h}{\mu}\,. \tag{24}$$

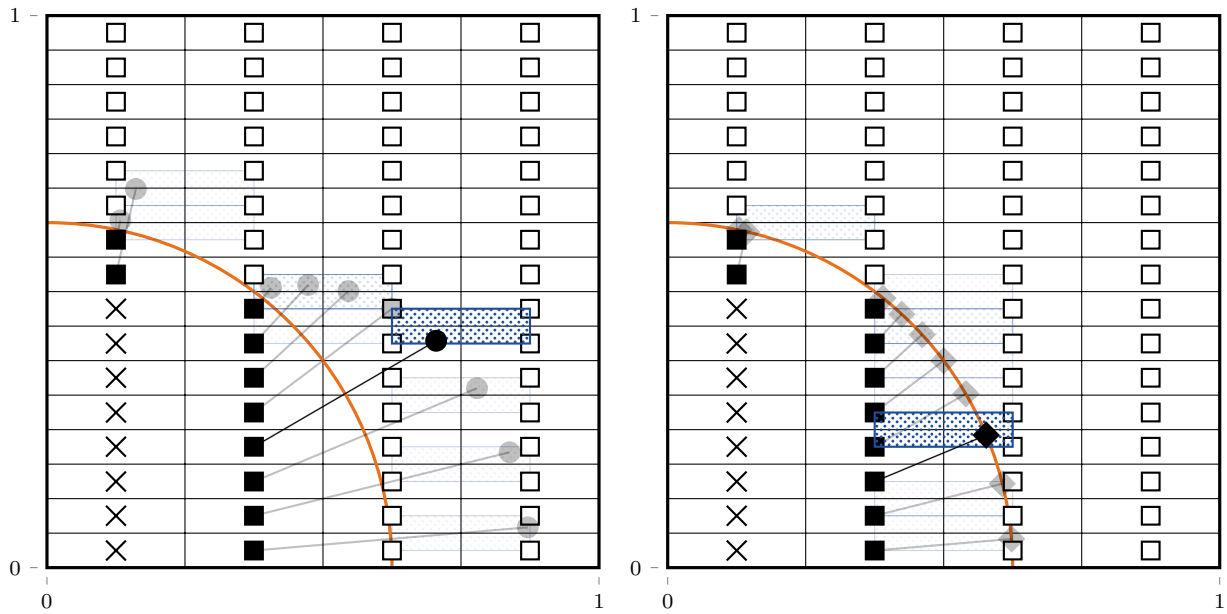
This equation suggests that the stencil size increases as the *cell size ratio* $a = h/\mu$ increases, and examples presented Fig. 6–7 show that it does increase. The figures present sets of interpolation nodes obtained with $p = 2$ and on very coarse grids around a circular immersed boundary. Figure 6 uses a 4×4 mesh of squares cells. The stencil size is 1 for the direct method and 2 for the linear method due to the greater distance reached by the probe points with respect to the ghost node. Figure 7 uses a 4×16 mesh of rectangular cells so $a = 4$. The highlighted ghost node has a stencil size of 2 for the direct method and 4 for the linear method. Indeed the stencil size has increased. More importantly for the direct method, (i, j) is no longer part of I_B and the second constraint of Eq. (14) is not verified anymore. Table 1 summarises minimum stencil sizes to obtain a second-order limiting behavior and for different cell size ratios and for Dirichlet and Neumann boundary conditions.



(a) Linear method.

(b) Direct method.

Figure 6: Examples of sets of interpolation nodes I_P and I_B on square cells. Notations of Fig. 3 applies. For the linear method, equation associated with the highlighted ghost node has a stencil of 2 as the probe point is farther than one cell width. For the direct method with the same discrete problem, equation associated with the highlighted ghost node has a stencil of 1 as the boundary point is closer that one cell width.



(a) Linear method.

(b) Direct method.

Figure 7: Examples of sets of interpolation nodes I_P and I_B on rectangular cells $a = 4$. Notations of Fig. 3 applies. For the linear method, equation associated with the highlighted ghost node has a stencil of 4, while for the direct method with the same discrete problem, equation associated with the highlighted ghost node has a stencil of 2. In the direct method, $U_{i,j}$ is not part of the set of interpolation nodes, and the boundary condition become ill posed.

Cell size ratio	*Method*	*Stencil size* *Dirichlet* $p = 2$	*Neumann* $p = 3$
1	Linear	2	3
	Direct	1	2
$a > 1$	Linear	$\lceil 2a \rceil$	$\lceil 2a \rceil + 1$
	Direct	n/a	n/a

Table 1: Stencil size for the Poisson problem with different immersed boundary conditions, immersed boundary methods, and different cell size ratios. Interpolation orders p are set to ensure second-order limiting behavior.

Conclusion. To reach second-order limiting behavior, this section showed that only the direct method is able to provide a stencil of size 1, i. e. a compact stencil, for the Dirichlet boundary condition. Moreover, it is not possible to expect a second-order limiting behavior when using Neumann boundary conditions on stencils of size 1 and for both methods; the stencil size must be at least equal to 2. On rectangular meshes, the direct method does not work anymore, and the stencil of the linear method increases as the cell size ratio a. Therefore band matrices become less suitable, and maybe inoperable, as more diagonals must be allocated as a increases. Next section proposes a method that reduces the stencil size on rectangular cells to the sizes observed on square cells.

4. The Ghost Node Shifting Method

This section defines the Ghost Node Shifting Method introduced in this paper. It modifies the linear and direct methods described in Sect. 2.2 to reduces the stencil size discussed in Sect. 3.

At high cell size ratio, the closest point of Γ from $\boldsymbol{X}_{i,j}$ may be several cells away, as in the example Fig 7b. But we know from Fig. 2 that there is at least one point, i. e. $\boldsymbol{x}_C$, which is less that one cell away from the ghost cell. The idea of the following procedure is to *shift* $\boldsymbol{X}_{i,j}$ towards $\boldsymbol{x}_C$ so that the new closest point of Γ will be closer to $\boldsymbol{x}_C$ and thus at fewer cells away.

4.1. Ghost point

We consider a ghost cell (i, j) and its inner cell neighbor (k, l) as in Sect. 2.2. We define the *ghost point* $\boldsymbol{x}_G$ as

$$\boldsymbol{x}_G = \boldsymbol{x}_C - s\boldsymbol{e}\,, \qquad s = \min\left\{\mu, \|\boldsymbol{x}_C - \boldsymbol{X}_{i,j}\|_2\right\}, \qquad \boldsymbol{e} = \begin{cases} (+1,0)^\intercal & \text{if } (k,l) = (i+1,j)\,, \\ (-1,0)^\intercal & \text{if } (k,l) = (i-1,j)\,, \\ (0,+1)^\intercal & \text{if } (k,l) = (i,j+1)\,, \\ (0,-1)^\intercal & \text{if } (k,l) = (i,j-1)\,, \end{cases} \tag{25}$$

recalling that $\mu = \min\{h_x, h_y\}$. The unit vector $\boldsymbol{e}$ gives the direction of $\boldsymbol{x}_C$ from $\boldsymbol{x}_G$ and the definition of s ensures that the distance between the two points is at most μ. Figure 8 illustrate the definition of $\boldsymbol{x}_G$ for $(k, l) = (i + 1, j)$. The situation from the point of view of $\boldsymbol{x}_G$ is similar to the case of a square cell of size μ, which fits in the cell (i, j). Now the quantities $\boldsymbol{x}_B$, $\boldsymbol{x}_P$, d, $\boldsymbol{n}$ are redefined as in Sect. 2.2, but using $\boldsymbol{x}_G$ instead of $\boldsymbol{X}_{i,j}$. The modified quantities are noted $\boldsymbol{x}_{B'}$, $\boldsymbol{x}_{P'}$, d', $\boldsymbol{n}'$ and Fig. 8 also shows these redefinitions.

The representation of the immersed boundary becomes important here. If Γ is represented by an exact parametrization or a piecewise linear curve, evaluations of d' and $\boldsymbol{n}'$ can be evaluated up to machine epsilon; if Γ is represented by a discretized level-set field, some sort of interpolation must be considered. In the numerical simulations presented in this document, d' and $\boldsymbol{n}'$ can be evaluated up to machine epsilon.

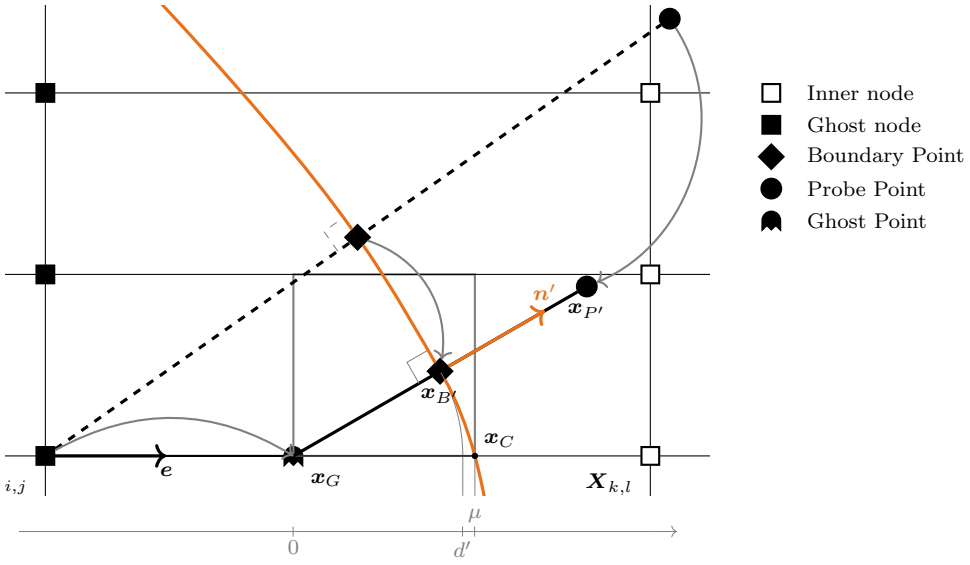


Figure 8: The Ghost Node Shifting Method applied to the example Fig. 2. The ghost point $\boldsymbol{x}_G$, initially placed at $\boldsymbol{X}_{i,j}$, is shifted towards $\boldsymbol{X}_{k,l}$. As a result, boundary and probe points computed from $\boldsymbol{x}_G$ are closer to the cell (i,j). The actual position of $\boldsymbol{x}_G$ is at the corner of the gray square of size μ.

4.2. Shifted linear and direct methods

The steps of the linear method Sect. 2.2.1 are done again replacing $U_{i,j}$ by $u_G = u(\boldsymbol{x}_G)$

$$\frac{u_G + u_{P'}}{2} = u_{B'} + \mathcal{O}(d'^2)\,, \tag{26a}$$

$$\frac{u_{P'} - u_G}{2d'} = \partial_{\boldsymbol{n}} u_{B'} + \mathcal{O}(d'^2)\,, \tag{26b}$$

As u_G is not a node value, another p-th order interpolation is introduced:

$$u_G = \sum_{(k,l)\in I_G} \gamma_{k,l} U_{k,l} + \mathcal{O}(h^p)\,, \tag{27}$$

where interpolation coefficients $\gamma_{k,l}$ and the set of interpolation indices I_G are computed as described in Sect. 2.3. The fully discretized equations are then

$$\sum_{(k,l)\in I_G} \frac{1}{2}\gamma_{k,l} U_{k,l} + \sum_{(k,l)\in I_{P'}} \frac{1}{2}\rho'_{k,l} U_{k,l} = D(\boldsymbol{x}_{B'}) + \mathcal{O}(h^2) + \mathcal{O}(h^p)\,, \qquad \forall (i,j) \in C_g\,, \tag{28a}$$

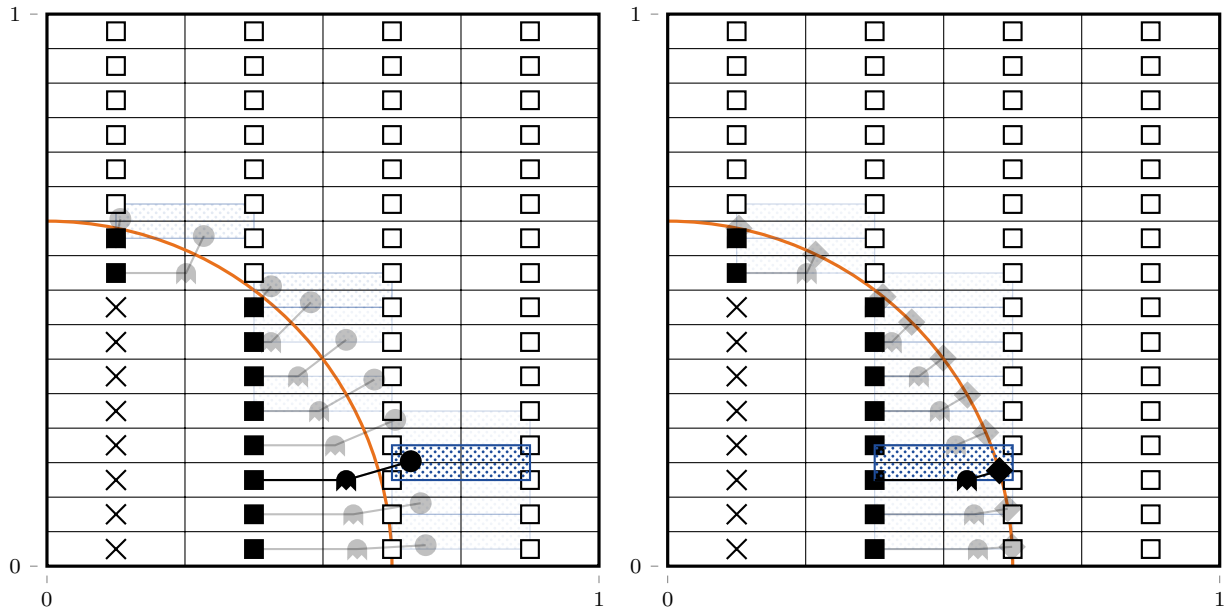
$$\sum_{(k,l)\in I_{P'}} \frac{1}{2d'}\rho'_{k,l} U_{k,l} - \sum_{(k,l)\in I_G} \frac{1}{2d'}\gamma_{k,l} U_{k,l} = N(\boldsymbol{x}_{B'}) + \mathcal{O}(h^2) + \mathcal{O}(h^{p-1})\,, \qquad \forall (i,j) \in C_g\,, \tag{28b}$$

where $\rho'_{k,l}$ are the interpolation coefficients for P'. Like the non-shifted Eq. (10), a numerical approximation of the Poisson problem can be built using the shifted linear method Eq. (28), and a second-order limiting behavior is obtained when second-order interpolations are used with Dirichlet boundary conditions, and when third-order interpolations are used with Neumann boundary conditions.

The direct method Sect. 2.2.2 can be applied seamlessly.

$$\sum_{(k,l)\in I_{B'}} \beta'_{k,l} U_{k,l} = D(\boldsymbol{x}_{B'}) + \mathcal{O}(h^p)\,, \qquad \forall (i,j) \in C_g \cap \partial\Omega_D\,, \tag{29a}$$

$$\sum_{(k,l)\in I_{B'}} (\partial_x \beta'_{k,l} n_x + \partial_y \beta'_{k,l} n_y) U_{k,l} = N(\boldsymbol{x}_{B'}) + \mathcal{O}(h^{p-1})\,, \qquad \forall (i,j) \in C_g \cap \partial\Omega_N\,, \tag{29b}$$



(a) Shifted linear method.

(b) Shifted direct method.

Figure 9: Examples of sets of interpolation nodes $I_{P'}$ and $I_{B'}$ on rectangular cells $a = 4$. Notations of Fig. 3 applies. Using the Ghost Node Shifting Method on the same discrete problem than Fig. 9, equation associated with the highlighted ghost node has a reduced stencil of 2 for the linear method, and 1 for the direct method. In the direct method, $U_{i,j}$ is now part of the set of interpolation nodes.

where $\beta'_{k,l}$ are the interpolation coefficients for B', and $\partial_x\beta'_{k,l}$ and $\partial_y\beta'_{k,l}$ are the interpolation coefficients of the polynomial gradient for B'. The effects of the Ghost Node Shifting Method affects the equations through $\boldsymbol{x}_{B'}$ and $\boldsymbol{n}'$ only. Like the non-shifted Eq. (13), a numerical approximation of the Poisson problem can be built using the shifted direct method Eq. (29), and a second-order limiting behavior is obtained when second-order interpolations are used with Dirichlet boundary conditions, and when third-order interpolations are used with Neumann boundary conditions.

4.3. Stencil size of the shifted methods

The discussion of Sect. 3.2 applies here, and Eq. (23) holds. In this section, we look for a new bounding of $\|\boldsymbol{x}_\chi - \boldsymbol{X}_{i,j}\|_c$ to provide a better approximation of Eq. (23). Let's start by developing the norm

$$\|\boldsymbol{x}_\chi - \boldsymbol{X}_{i,j}\|_{\mathrm{c}} \leqslant \max\left\{\frac{|x_\chi - x_G| + |x_G - x_i|}{h_x}, \frac{|y_\chi - y_G| + |y_G - y_j|}{h_y}\right\}. \tag{30}$$

The $\boldsymbol{x}_G - \boldsymbol{X}_{i,j}$ part depends on the location of the neighbor inner cell

$$\begin{cases} |x_G - x_i| < h_x - \mu\,, & |y_G - y_j| = 0\,, & \text{if } (k,l) = (i+1,j)\,, \\ |x_G - x_i| < h_x - \mu\,, & |y_G - y_j| = 0\,, & \text{if } (k,l) = (i-1,j)\,, \\ |x_G - x_i| = 0\,, & |y_G - y_j| < h_y - \mu\,, & \text{if } (k,l) = (i,j+1)\,, \\ |x_G - x_i| = 0\,, & |y_G - y_j| < h_y - \mu\,, & \text{if } (k,l) = (i,j-1)\,. \end{cases} \tag{31}$$

The $\boldsymbol{x}_\chi - \boldsymbol{x}_G$ part exploits its similarity with a square cell of size μ

$$\|\boldsymbol{x}_{B'} - \boldsymbol{x}_G\|_\infty \leqslant \|\boldsymbol{x}_{B'} - \boldsymbol{x}_G\|_2 \leqslant \|\boldsymbol{x}_C - \boldsymbol{x}_G\|_2 \leqslant \mu\,, \tag{32a}$$

and, by symmetry of the probe point

$$\|\boldsymbol{x}_{P'} - \boldsymbol{x}_G\|_\infty \leqslant 2\mu\,. \tag{32b}$$

The absolute values in Eq. (30) can then be replaced by these upper bounds. If $(k,l) = (i \pm 1, j)$:

$$\|\boldsymbol{x}_{P'} - \boldsymbol{X}_{i,j}\|_{\mathrm{c}} \leqslant \max\left\{\frac{h_x - \mu + 2\mu}{h_x}, \frac{2\mu}{h_y}\right\} \leqslant 2\,, \quad \|\boldsymbol{x}_{B'} - \boldsymbol{X}_{i,j}\|_{\mathrm{c}} \leqslant \max\left\{\frac{h_x - \mu + \mu}{h_x}, \frac{\mu}{h_y}\right\} \leqslant 1\,, \tag{33a}$$

and if $(k,l) = (i, j \pm 1)$:

$$\|\boldsymbol{x}_{P'} - \boldsymbol{X}_{i,j}\|_{\mathrm{c}} \leqslant \max\left\{\frac{2\mu}{h_x}\frac{h_y - \mu + 2\mu}{h_y}, \right\} \leqslant 2\,, \quad \|\boldsymbol{x}_{B'} - \boldsymbol{X}_{i,j}\|_{\mathrm{c}} \leqslant \max\left\{\frac{\mu}{h_x}\frac{h_y - \mu + \mu}{h_y}, \right\} \leqslant 1\,, \tag{33b}$$

Contrary to Eq. (24), these bounds do not depend on the cell size ratio. The example Fig (6-7) is completed by Fig. 9, which uses the shifted methods. The highlighted ghost node now has a stencil of 1 for the direct method and 2 for the linear method. Table 2 summarises minimum stencil sizes to obtain a second-order limiting behavior and for different cell size ratios.

		Stencil size	
Cell size ratio	*Method*	*Dirichlet* $p = 2$	*Neumann* $p = 3$
1	Linear	2	3
	Direct	1	2
$a > 1$	Linear	$\lceil 2a \rceil$	$\lceil 2a \rceil + 1$
	Direct	n/a	n/a
$a > 1$	Shifted Linear	2	3
	Shifted Direct	1	2

Table 2: Stencil size for the Poisson problem with different immersed boundary conditions, immersed boundary methods, and different cell size ratios. Interpolation orders p are set to ensure second-order limiting behavior.

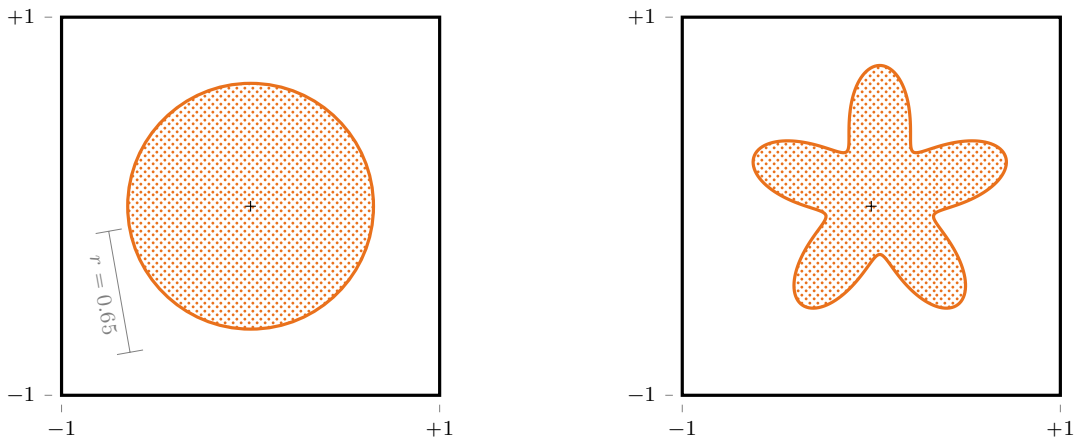
Conclusion. This section showed that it is possible to build immersed boundary conditions with a stencil size independent of the cell size ratio. The stencil size for rectangular cells is the same than the stencil size for square cells, which is the lowest size to be expected. With the Ghost Node Shifting Method, it is also possible to use the direct method on rectangular cells. This result is particularly useful to use band-matrix-limited linear system solvers such as in the *hypre* library. The next section present numerical results obtained for the Poisson problem.

5. Numerical simulations with the Poisson problem

This section discusses simulations of the Poisson problem solved with the Ghost Node Shifting Method and compare them with their classic counterparts.

Diagnostics. Simulations consists of convergence studies. As analytical solutions are available, the *error field* $\hat{U} - U$ can be computed at inner nodes, and $\mathcal{L}^2$ norm and $\mathcal{L}^\infty$ norm are used for comparison. These norms are

$$\mathcal{L}^2(\hat{U} - U) = \sqrt{\sum_{(i,j)\in C_i} |\hat{U}_{i,j} - U_{i,j}|^2 V_{i,j}}\,, \qquad \mathcal{L}^\infty(\hat{U} - U) = \max_{(i,j)\in C_i} |\hat{U}_{i,j} - U_{i,j}|\,, \tag{34}$$



(a) Circular immersed boundary.

(b) Flower-shaped immersed boundary.

Figure 10: Computational domain and immersed boundary used for numerical simulations of Sect. 5. The immersed boundary is shown by a thick colored line, and the outer domain is shaded by dots.

where $V_{i,j}$ is the volume of the inner portion of the cell (i, j). The numerical evaluation of $V_{i,j}$ is subject to errors due to the approximation in the representation of the immersed boundary, and it must be carefully done. We found that using cell-wise linear approximation were inaccurate and unreliable to evaluate the order of convergence. This happens because errors in $V_{i,j}$ computed this way were of the same order that $|\hat{U}_{i,j} - U_{i,j}|^2$. Instead we use a sub-sampling method as follows: first, take 400×400 sampling points evenly spread in the cell; then evaluate the ratio of sampling points included in Ω_i; finally approximate the cell volume by multiplying $h_x \times h_y$ to this ratio. With this method, errors in $V_{i,j}$ are below 10^{-6} for all runs.

Computational domain. The common computational domain for all simulations in this section is the square $\Omega = [-1, +1]^2$, with Dirichlet boundary conditions at the x-axis boundaries and Neumann boundary conditions at the y-axis boundaries. In the first set of simulations, the immersed boundary Γ is a circle of center $(0, 0)^\intercal$ and of radius 0.65; in the second set of simulations the immersed boundary Γ has a 5-branch flower shape defined by the parametrization

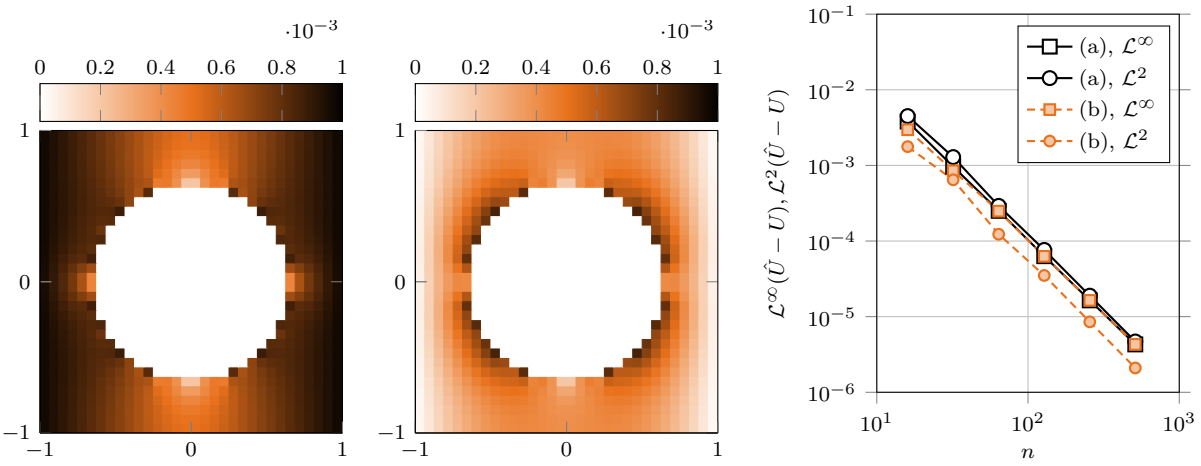
$$\vartheta \mapsto (0.02\sqrt{5}, 0.02\sqrt{5})^\intercal + \big(0.5 + 0.2\sin(5\vartheta)\big)\boldsymbol{e}_r\,, \qquad \forall \vartheta \in [0, 2\pi[\,. \tag{35}$$

This shape has also been used in [16, 17]. In both immersed boundaries, the inner part Ω_i is outside, as illustrated in Fig. 10.

Grids. We consider three mesh series with cell size ratio a of 1, 2.8, and 7.6. The first grid of each series has a cell size $m \times n$ of 16×16, 28×10, and 76×10, respectively. The next grid of each series is obtained by multiplying the preceding m and n by two in all directions. Thus convergence study can be performed on each of the grid series.

Numerical correction. Differences between numerical and exact solutions arise from the truncation terms $R_{i,j} = F_{i,j} - (LU)_{i,j}$ of the Poisson equation Eq. (3), the domain boundary conditions Eq. (4), and the immersed boundary conditions Eq. (10, 13, 28, 29) altogether. In the resulting error field $\hat{U} - U$, differences from all sources are fused together by diffusion, and numerical errors resulting from immersed boundary conditions alone cannot be readily seen. As an example, Fig. 11a shows the error field for the first numerical run presented in Sect. 5.1: the largest errors are both near the domain boundary and near the immersed boundary, which is coherent with the magnitude of the truncation terms at $\boldsymbol{x} = (1, 0)^\intercal$ and $\boldsymbol{x} = (0.65, 0)^\intercal$

$$r_{h_x,h_y}(1, 0) = -\frac{h_x^2}{4}, \qquad r_{h_x,h_y}(0.65, 0) = -\left((1-\theta)\theta^2 + (1-\theta)^2\theta\right)\frac{h_x^2}{2}, \quad \text{with } \theta \in [0, 1], \tag{36}$$



(a) Error field $|\hat{U} - U|$ computed without numerical correction.

(b) Error field $|\hat{U} - U|$ computed with numerical correction.

(c) Convergence study Fig. 12 of error field whose first grids correspond to (a, b).

Figure 11: (a, b) Effects of numerical correction on the error fields $|\hat{U} - U|$ for the problem Sect. 5.1 on the 16×16 grid and with the non-shifted direct method. (c) Effects of numerical correction on the convergence plot Fig. 12 with $a = 1$ and with the non-shifted direct method. A second-order limiting behavior is shown for the error field both with and without numerical correction, but the norm with the largest error is not the same: $\mathcal{L}^2$ without numerical correction and $\mathcal{L}^\infty$ with numerical correction.

where $r_{h_x,h_y}(\boldsymbol{x})$ is the truncation term taken at point $\boldsymbol{x}$. Equation (40) has been used for $r_{h_x,h_y}(1,0)$ and Eq. (41) has been used for $r_{h_x,h_y}(0.65,0)$.

As we are primarily interested on errors caused by the immersed boundary conditions, we decided to cancel error sources that are not related to immersed boundaries. To do this, we compute $R_{i,j}$ from the exact solution and we add it to the right-hand side of the system for all but immersed boundary condition. We call this the *numerical correction*. Figure 11b show effects on the error field when the numerical correction is applied: remaining errors are clearly located near immersed boundaries. Finally, Figure 11c compares convergence results in $\mathcal{L}^\infty$ and $\mathcal{L}^2$ norms for the same computations: while the second-order behavior is present in the two computations, errors are divided by $\frac{5}{2}$ in the $\mathcal{L}^2$ norm when numerical correction is applied.

5.1. Dirichlet immersed boundary condition for the circular interface

Analytical solution. A solution of the Poisson equation is given by

$$u(x,y) = (1+x)^2\,, \qquad f(x,y) = -2\,, \qquad \forall (x,y) \in \Omega_i\,, \tag{37}$$

provided consistent boundary conditions

$$u(-1,y) = 0\,, \qquad u(+1,y) = 4\,, \qquad \forall y \in [-1,+1]\,, \tag{38a}$$

$$\frac{\partial u}{\partial y}(x,-1) = 0\,, \qquad \frac{\partial u}{\partial y}(x,+1) = 0\,, \qquad \forall x \in [-1,+1]\,, \tag{38b}$$

$$u(x,y) = (1+x)^2\,, \qquad \forall (x,y) \in \Gamma\,. \tag{38c}$$

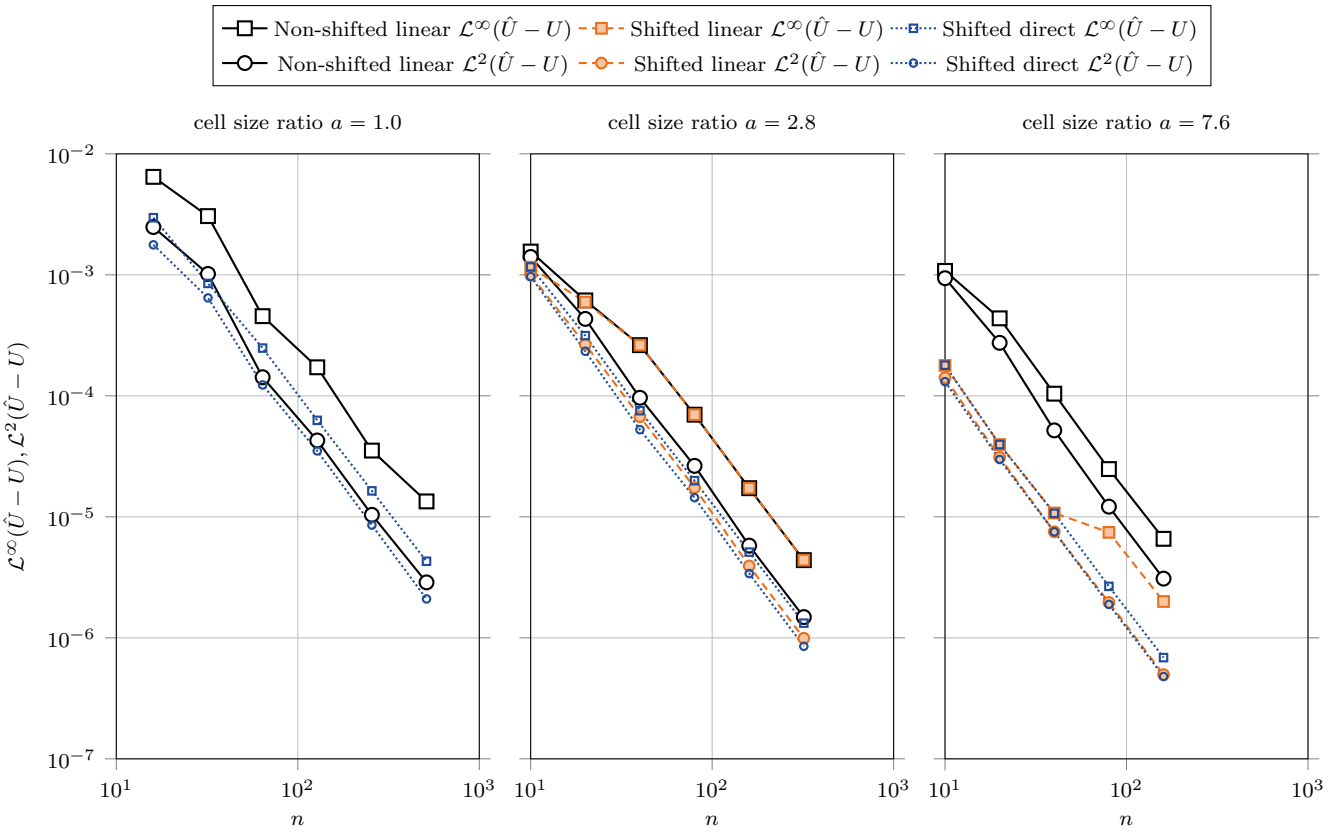


Figure 12: Norms of the solution error $\hat{U} - U$ of the Poisson problem for Dirichlet immersed boundary condition on the circular interface Sect. 5.1. Each plot represent a mesh series. Solutions have been computed with the non-shifted linear, shifted linear, and shifted direct methods, except for the shifted linear method on the mesh series $a = 1.0$ as it is equivalent to the non-shifted linear method. The non-shifted direct did not converge when $a > 1.0$ as expected in Sect 3.2 and in Fig. 7. The x-axis represents $1/h = 1/\max\{h_x, h_y\} = \min\{n_x, n_y\}$, which is always equal to n in this section.

The truncation terms for this particular solution are

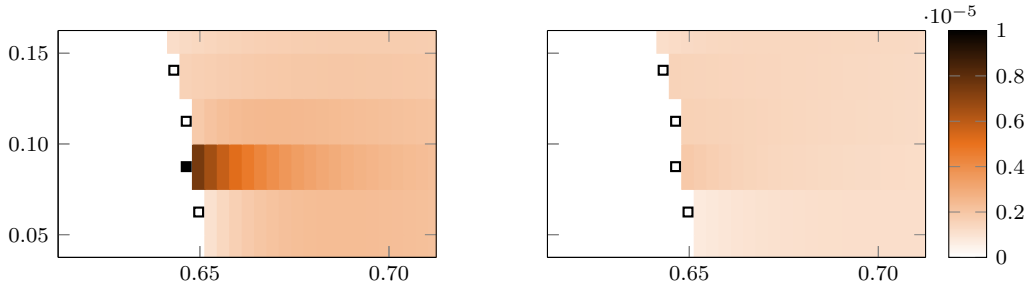
$$F_{i,j} - \frac{U_{i-1,j} - U_{i,j}}{h_x^2} - \frac{U_{i+1,j} - U_{i,j}}{h_x^2} - \frac{U_{i,j-1} - U_{i,j}}{h_y^2} - \frac{U_{i,j+1} - U_{i,j}}{h_y^2} = 0\,, \quad \forall (i,j) \in C_i\,, \tag{39}$$

$$\begin{aligned} 0 - \frac{U_{0,j} + U_{1,j}}{2} &= -\tfrac{1}{4}h_x^2\,, \quad 4 - \frac{U_{m,j} + U_{m+1,j}}{2} = -\tfrac{1}{4}h_x^2\,, \quad \forall j \in [\![1, n]\!]\,, \\ 0 - \frac{U_{i,1} - U_{i,0}}{h_y} &= 0\,, \qquad 0 - \frac{U_{i,n+1} - U_{i,n}}{h_y} = 0\,, \qquad \forall i \in [\![1, m]\!]\,. \end{aligned} \tag{40}$$

Therefore only the Dirichlet boundary conditions at the x-axis boundaries will be numerically corrected.

Results. Numerical solutions $\hat{U}$ are computed using the linear and the direct methods for the isotropic series, and the non-shifted linear (NL) method, the shifted linear (SL) method, and the shifted direct (SD) method for the two anisotropic series. Second-order interpolations ($p = 2$) are used to build immersed boundary conditions. Figure 12 shows $\mathcal{L}^2$ and $\mathcal{L}^\infty$ norms of the error field $\hat{U} - U$. As expected, a second-order limiting behavior is obtained with all methods.

The figure also shows that the SD method has less errors than the NL method. The $\mathcal{L}^2$ norm of the SL method is similar tho the $\mathcal{L}^2$ norm of the SD method, whereas the $\mathcal{L}^\infty$ norm of the SL method is as large as the NL method when $a = 2.8$, and changes when $a = 7.6$. Figure 13 shows the error field for the 608×80 grid, e. g. when the $\mathcal{L}^\infty$ norm of the SL method does not decreases as expected: large error occur



(a) Shifted linear method (b) Shifted direct method

Figure 13: Error fields $|\hat{U} - U|$ for the solution of Poisson problem with Dirichlet boundary conditions described in Sect. 5.1. Open symbols indicates ghost cell with a stencil size of 1, and closed symbols indicates ghost cells with a stencil size of 2.

around four nodes only. On the close up Fig. 13a, ghost cells are filled according to the compactness: white if $c_{i,j} = 1$, and black if $c_{i,j} = 2$. One can readily see that large errors occurs near nodes with $c_{i,j} = 2$. By comparison, Fig. 13b shows the corresponding results with the SD method: as $c_{i,j} = 1$ everywhere, large errors have disappeared. This feature calls for a finer analysis of the truncation error.

Taylor series of the truncation error. Let's consider a one-dimensional grid containing only three grid nodes, located at: $x_{-1} = -h$, $x_0 = 0$, and $x_{+1} = +h$. An immersed boundary crosses the grid at $x_B = x_{-1} + \theta h$, with $\theta \in [0, 1]$, as shown in the diagram Fig. 14a. Using Taylor series of a scalar field u expanded at x_B, the direct method writes

$$\hat{u}_B = (1-\theta)u_{-1} + \theta u_0 = u_B + \sum_{k=1}^{\infty} \left[(1-\theta)(-\theta)^k + \theta(1-\theta)^k \right] \frac{h^k}{k!} \frac{\partial^k u}{\partial x^k}\,, \qquad \text{if } 0 \leqslant \theta \leqslant 1, \tag{41}$$

and the linear method writes

$$\begin{aligned} \hat{u}_B &= (1-2\theta)u_{-1} + 2\theta u_0 \\ &= u_B + \sum_{k=1}^{\infty} \left[(1-\theta)(-\theta)^k + \theta(1-\theta)^k \right] \frac{h^k}{k!} \frac{\partial^k u}{\partial x^k}\,, \qquad \text{if } 0 \leqslant \theta \leqslant \frac{1}{2}, \end{aligned} \tag{42a}$$

$$\begin{aligned} \hat{u}_B &= (2-2\theta)u_0 + (2\theta-1)u_{+1} \\ &= u_B + \sum_{k=1}^{\infty} \left[(\theta-\tfrac{1}{2})(2-\theta)^k + (1-\theta)(1-\theta)^k + \tfrac{1}{2}(-\theta)^k \right] \frac{h^k}{k!} \frac{\partial^k u}{\partial x^k}\,, \qquad \text{if } \frac{1}{2} \leqslant \theta \leqslant 1. \end{aligned} \tag{42b}$$

Notice that the compactness of the linear method is 1 when $0 \leqslant \theta \leqslant \frac{1}{2}$ and 2 when $\frac{1}{2} \leqslant \theta \leqslant 1$. Coefficients of the first eight terms of the series are shown in Fig. 14a for the direct method, and in Fig. 14b for the linear method. Coefficients are identical when $0 \leqslant \theta \leqslant \frac{1}{2}$, whereas coefficients of the linear method have much higher values when $\frac{1}{2} \leqslant \theta \leqslant 1$. We therefore can expect higher error values from the linear method when the compactness is 2, which can reasonably explain the results of the two dimensional computation.

5.2. Neumann immersed boundary condition for the circular interface

Analytical solution. Another solution of the Poisson equation is given by the solution and source fields

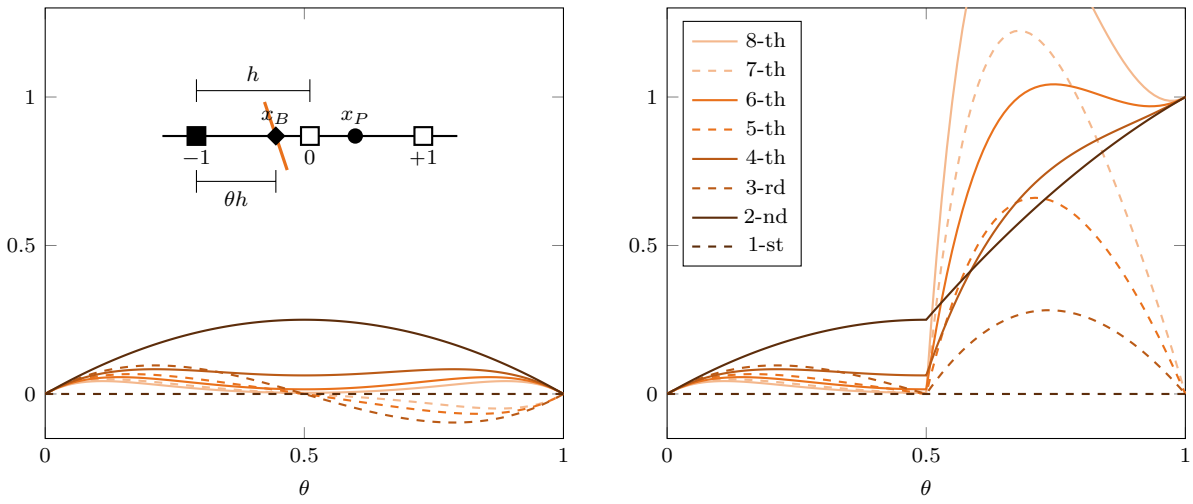
$$u(x,y) = (1+x)^3\,, \qquad f(x,y) = -6(1+x)\,, \qquad \forall (x,y) \in \Omega_i\,, \tag{43}$$

provided consistent computational domain boundary conditions

$$u(-1,y) = 0\,, \qquad u(+1,y) = 8\,, \qquad \forall y \in [-1,+1]\,, \tag{44a}$$

$$\frac{\partial u}{\partial y}(x,-1) = 0\,, \qquad \frac{\partial u}{\partial y}(x,+1) = 0\,, \qquad \forall x \in [-1,+1]\,, \tag{44b}$$

$$\partial_{\boldsymbol{n}} u(x,y) = 6x(1+x)^2\,, \qquad \forall (x,y) \in \Gamma\,. \tag{44c}$$



(a) Shifted direct method

(b) Shifted linear method

Figure 14: Coefficients of the Taylor's series of the truncation error.

The truncation terms for this particular solution are

$$F_{i,j} - \frac{U_{i-1,j} - U_{i,j}}{h_x^2} - \frac{U_{i+1,j} - U_{i,j}}{h_x^2} - \frac{U_{i,j-1} - U_{i,j}}{h_y^2} - \frac{U_{i,j+1} - U_{i,j}}{h_y^2} = 0\,, \quad \forall (i,j) \in C_i\,, \tag{45}$$

$$\begin{aligned} 0 - \frac{U_{0,j} + U_{1,j}}{2} &= 0\,, \quad 8 - \frac{U_{m,j} + U_{m+1,j}}{2} = -\tfrac{3}{2}h_x^2\,, \quad \forall j \in [\![1,n]\!]\,, \\ 0 - \frac{U_{i,1} - U_{i,0}}{h_y} &= 0, \quad 0 - \frac{U_{i,m+1} - U_{i,m}}{h_y} = 0\,, \qquad \forall i \in [\![1,m]\!]\,. \end{aligned} \tag{46}$$

Again, only the Dirichlet boundary conditions at $\partial\Omega_l$ and $\partial\Omega_r$ will be numerically corrected.

Results with second-order interpolation. Numerical solutions $\hat{U}$ are computed using the methods described in Sect. 5.1. Second-order interpolations ($p = 2$) are used to build immersed boundary conditions. Figure 15 shows $\mathcal{L}^2$ and $\mathcal{L}^\infty$ norms of the error field $\hat{U} - U$. As expected, a first-order limiting behavior is obtained with all methods. As with Dirichlet boundary conditions, the figure also shows that, for any given grid, the shifted linear methods SL and SD have less errors than the non-shifted linear method NL. The two shifted methods SL and SD are equivalent in terms of both $\mathcal{L}^2$ and $\mathcal{L}^\infty$ norms.

Results with third-order interpolation. The same computations are performed again, but third order interpolations ($p = 3$) are used to build immersed boundary conditions. Figure 16 shows $\mathcal{L}^2$ and $\mathcal{L}^\infty$ norms of the error field $\hat{U} - U$. As expected, a second-order limiting behavior is obtained with all methods. As with second-order interpolations, the figure also shows that, for any given grid, the shifted linear methods SL and SD have less errors than the non-shifted linear method NL. The two shifted methods SL and SD are equivalent in terms of both $\mathcal{L}^2$ and $\mathcal{L}^\infty$ norms.

5.3. Dirichlet immersed boundary condition for the flower-shaped interface

The parabolic analytical solution Eq. (37) with Dirichlet boundary conditions is also applied here. Numerical solutions $\hat{U}$ are computed using the shifted linear (SL) and the shifted direct (SD) methods for all series. As the anisotropy increases the low resolved grid become less and less adapted to the immersed boundary, and the hollow case problem discussed in Sect. 2.3 started to occur. This is shown in Fig. 17 where a row of four inner nodes are surrounded by ghost cells only, except on the left side.

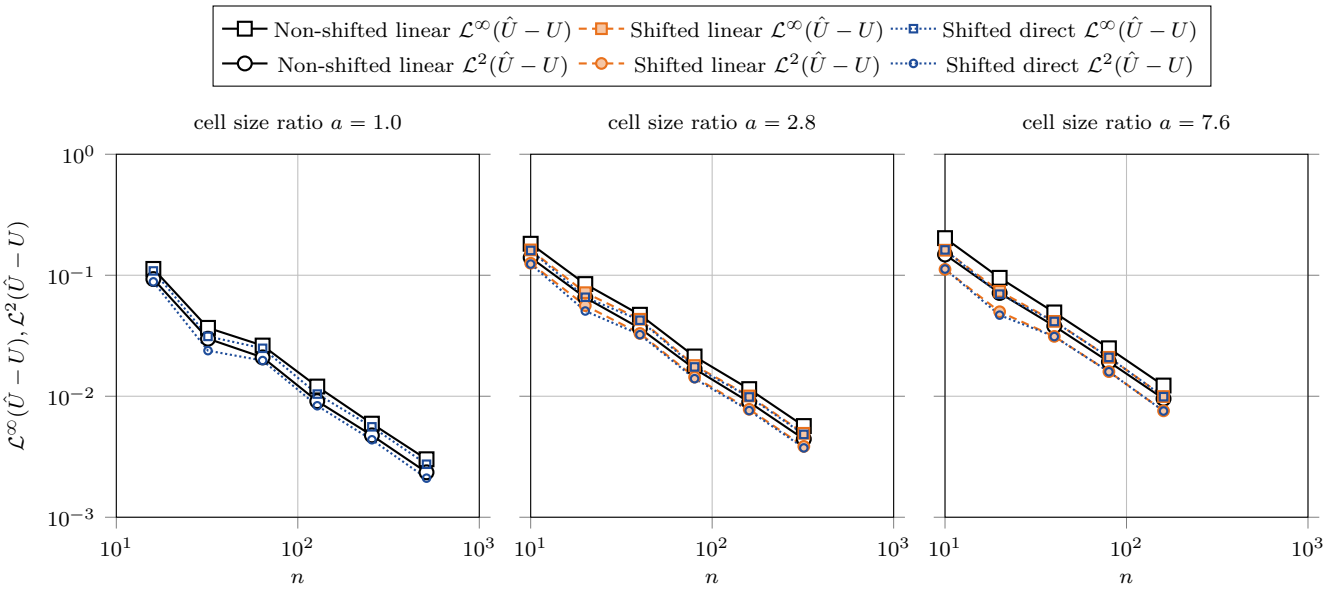


Figure 15: Norms of the solution error $\hat{U} - U$ of the Poisson problem for Neumann immersed boundary condition on the circular interface Sect. 5.2. Each plot represent a mesh series. Solutions have been computed with the non-shifted linear, shifted linear, and shifted direct methods, except for the shifted linear method on the mesh series $a = 1.0$ as it is equivalent to the non-shifted linear method. The x-axis represents $1/h = 1/\max\{h_x, h_y\} = \min\{n_x, n_y\}$, which is always equal to n in this section.

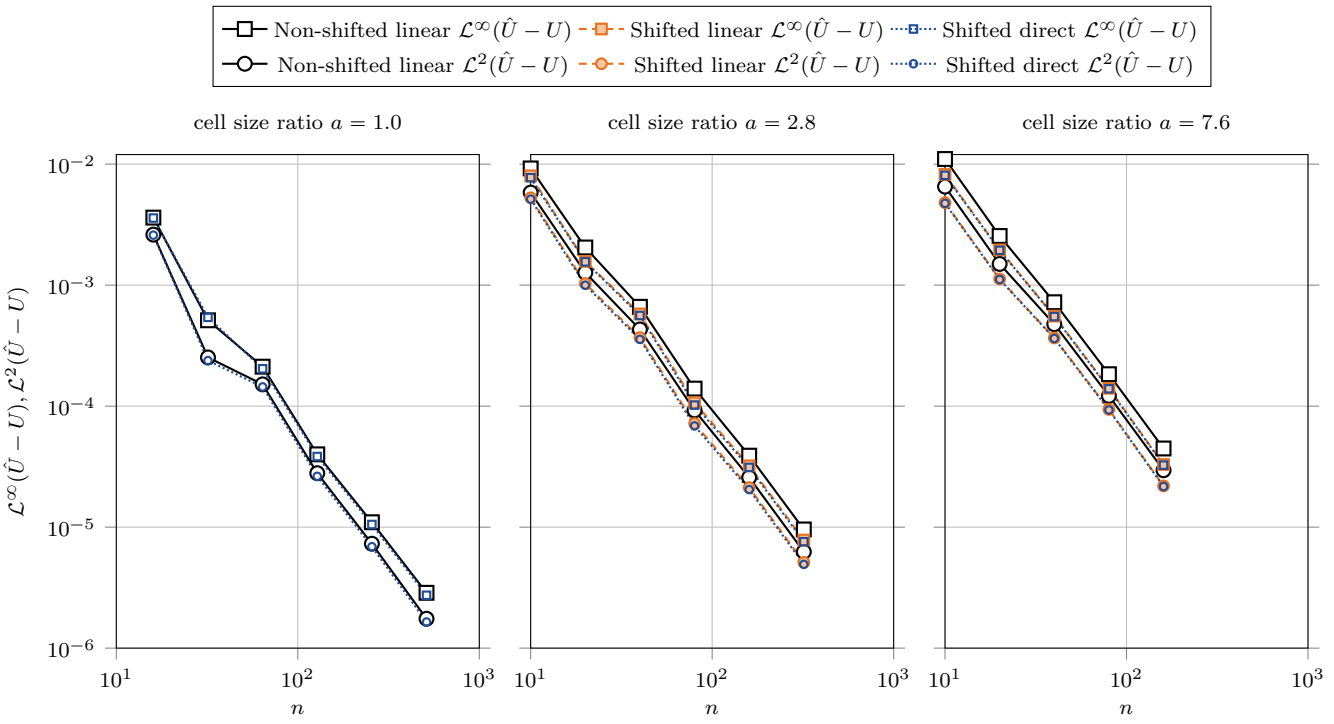


Figure 16: Norms $\mathcal{L}^2$ and $\mathcal{L}^\infty$ of the solution error $\hat{U} - U$ for the Poisson problem with Neumann immersed boundary conditions and third-order interpolation, as presented in Sect. 5.2. The non-shifted linear, shifted linear, and shifted direct methods are computed in each of the three mesh series. Although, the shifted linear method is not shown for the mesh series $a = 1$, as it is equivalent to the non-shifted linear method. The x-axis represents $1/h = 1/\max\{h_x, h_y\} = \min\{n_x, n_y\}$, which is always equal to n in this section.

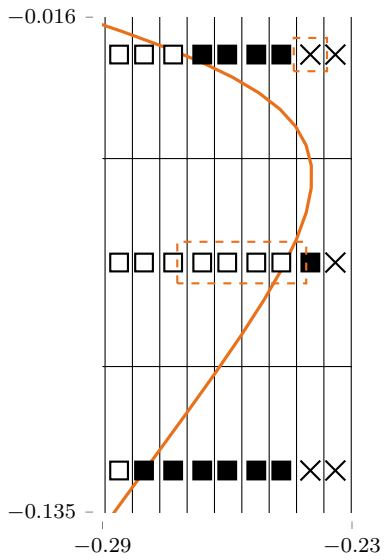


Figure 17: Close view of node types near a hollow part of the Flower-shaped immersed boundary with a gird of ratio $a = 7.6$. Outlined by a dashed box, a row of four inner cells are surrounded by ghost cells except on the left. Also outlined by a dashed boy, an outer cell whose value is used in interpolation due to the hollow case problem. Large error are expected in this region.

To complete this convergence study, we used another grid series with $a = 4.7$, and we increase the resolution for $a = 7.6$ to catch the limiting second-order convergence region. Figure 18 shows $\mathcal{L}^2$ and $\mathcal{L}^\infty$ norms of the error field $\hat{U} - U$. As expected, a second-order limiting behavior is obtained with all methods. Here again, error levels of the SD method are less or equal than error levels of the SL method. For sufficiently high resolution, a convergence is recovered for the grid series $a = 7.6$.

5.4. Solver comparison

Simulations presented in Sect. 5.3 and Sect 5.2 has been used to compare the performance of different linear system solvers. As the stencil size is 1 for the direct method, we use the GMRES solver with the SMG preconditioner from the *hypre* library (GMRES+SMG), and as the stencil size of the linear method is 2, we use the GMRES solver with the BoomerAMG preconditioner from the same library (GMRES+BoomerAMG). The allowed residual for all solvers is 10^{-10}. Also, the *mumps* solver has been used in both cases, for which we do not count time take by the symbolic factorization, which is quite long. Computations have been done in parallel using 16 processes on Intel® Xeon® CPU E5-4640 0 at 2.40 GHz, and solver details are given in Appendix B.

If we give a look first to the flower test case with Dirichlet boundary conditions in Tab. 3 , between the two iterative solvers, GMRES+BoomerAMG is faster than GMRES+SMG at low resolutions, but the former slows much more rapidly as the grid size increases than the latter. It confirms the interest we have to get the most compact scheme. It is also another advantage of the direct method which seems to be more precise than the linear method. Note that for the circular interface with Neumann immersed boundary conditions in Tab. 4, for the finest mesh and $a = 4.7$, both GMRES+SMG and GMRES+BoomerAMG fail or converge very slowly; instead we used *mumps*. The direct *mumps* solver is slower than their iterative counterparts, yet competitive; it is also less affected by the anisotropy a and is even faster than GMRES+SMG in one case.

6. Immersed boundary method with the incompressible Navier-Stokes problem

This section applies the shifted immersed boundary method to the incompressible Navier-Stokes problem with uniform density and viscosity. A pressure-correction method on a staggered grid is employed. It is

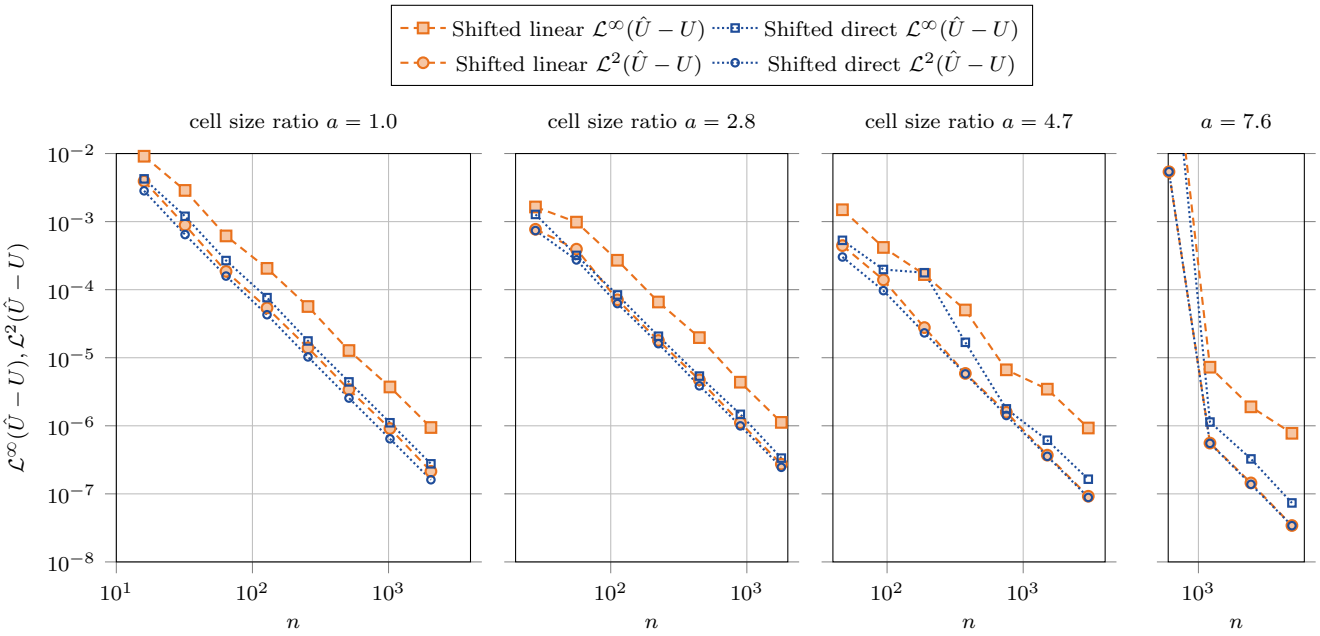


Figure 18: Norms $\mathcal{L}^2$ and $\mathcal{L}^\infty$ of the solution error $\hat{U} - U$ for the Poisson problem with Flower-shaped immersed boundary. The x-axis represents $1/h = 1/\max\{h_x, h_y\} = \min\{n_x, n_y\}$, which is always equal to n in this section.

		Direct method			*Linear method*		
		GMRES+SMG		*MUMPS*	*GMRES+BoomerAMG*		*MUMPS*
a	*Grid*	*Iter.*	*Time*/s	*Time*/s	*Iter.*	*Time*/s	*Time*/s
1	160 × 160	15	0.06	0.05	18	0.04	0.09
	320 × 320	19	0.13	0.14	19	0.12	0.31
	640 × 640	23	0.35	0.56	21	0.65	1.32
	1280 × 1280	24	1.35	2.71	19	4.01	9.93
7.6	608 × 80	18	0.08	0.07	23	0.07	0.12
	1216 × 160	17	0.16	0.22	25	0.34	0.53
	2432 × 320	21	0.54	0.88	28	1.7	3.5
	4864 × 640	29	6.25	4.2	31	24.6	19.7

Table 3: Comparison of the performance of linear system solvers to solve the Poisson problem with a Flower-shaped immersed boundary and Dirichlet boundary conditions Sect. 5.3 in terms of GMRES iterations (*Iter.* columns) and CPU time (*Time* columns). The tolerance of the GMRES solver is 10^{-10}. *Mumps* symbolic factorization is not taken into account in the CPU time. Computations have been done in parallel with 16 processes on Intel® Xeon® CPU E5-4640 0 at 2.40 GHz.

		Direct method			*Linear method*		
		GMRES+SMG		*MUMPS*	*GMRES+BoomerAMG*		*MUMPS*
a	*Grid*	*Iter.*	*Time*/s	*Time*/s	*Iter.*	*Time*/s	*Time*/s
2.8	448×160	39	0.06	0.11	43	0.1	0.2
	896×320	46	0.30	0.38	61	0.9	0.75
	1792×640	41	1.42	1.54	66	4.38	5.10
	3584×1280	78	14	8.7	126	38	37
4.7	376×80	62	0.05	0.06	246	0.23	0.06
	752×160	138	0.31	0.12	> 500	n/a	0.31
	1504×320	> 500	n/a	0.59	> 500	n/a	1.90
	3008×640	> 500	n/a	3.28	> 500	n/a	8.8

Table 4: Comparison of the performance of linear system solvers to solve the Poisson problem with a circular immersed boundary and Neumann boundary conditions Sect. 5.2 in terms of GMRES iterations (*Iter.* columns) and CPU time (*Time* columns). The tolerance of the GMRES solver is 10^{-10}. *Mumps* symbolic factorization is not taken into account in the CPU time. Computations have been done in parallel with 16 processes on Intel® Xeon® CPU E5-4640 0 at 2.40 GHz.

shown here that just applying the Ghost-Fluid Finite-Difference to the sub-problems is not enough, and, due to the nonlinear term, an additional extrapolation is necessary.

We consider the *velocity field* $\boldsymbol{u} = (u, v)^\intercal : \Omega_i \times [0, T] \to \mathbb{R}^2$ and the *kinematic pressure* $p : \Omega_i \times [0, T] \to \mathbb{R}$, both restricted to the inner domain, solution of the incompressible Navier-Stokes with uniform density, uniform viscosity, and *source field* $\boldsymbol{f} : \Omega_i \times [0, T] \to \mathbb{R}^2$:

$$\partial_t \boldsymbol{u} + \nabla\cdot(\boldsymbol{u} \otimes \boldsymbol{u}) = -\nabla p + \nu \Delta \boldsymbol{u} + \boldsymbol{f}\,, \qquad \text{in } \Omega_i\,, \tag{47a}$$

$$\nabla\cdot \boldsymbol{u} = 0\,, \qquad \text{in } \Omega_i\,, \tag{47b}$$

$$\boldsymbol{u} = \boldsymbol{D}\,, \qquad \text{on } \partial\Omega_D \cup \Gamma_D\,, \tag{47c}$$

$$\nabla \boldsymbol{u} \cdot \boldsymbol{n} = \boldsymbol{N}\,, \qquad \text{on } \partial\Omega_N \cup \Gamma_N\,, \tag{47d}$$

where ν is the *kinematic viscosity*, and $\boldsymbol{D}$ and $\boldsymbol{N}$ are known functions.

6.1. Time discretization and velocity-pressure coupling

The timeline $[0, T]$ is divided in constant *time steps* h_t. Equation (47) is applied at t^{n+1}, the time derivative $\partial_t \boldsymbol{u}$ is developed backwards, and the nonlinear term is linearized:

$$\frac{1}{h_t}\left(\boldsymbol{u}^{n+1} - \boldsymbol{u}^n\right) + \nabla\cdot(\boldsymbol{u}^{n+1} \otimes \boldsymbol{u}^n) = -\nabla p^{n+1} + \nu \Delta \boldsymbol{u}^{n+1} + \boldsymbol{f} + \mathcal{O}(h_t)\,, \qquad \text{in } \Omega_i\,, \tag{48a}$$

$$\nabla\cdot \boldsymbol{u}^{n+1} = 0\,, \qquad \text{in } \Omega_i\,, \tag{48b}$$

$$\boldsymbol{u}^{n+1} = \boldsymbol{D}\,, \qquad \text{on } \partial\Omega_D \cup \Gamma_D\,, \tag{48c}$$

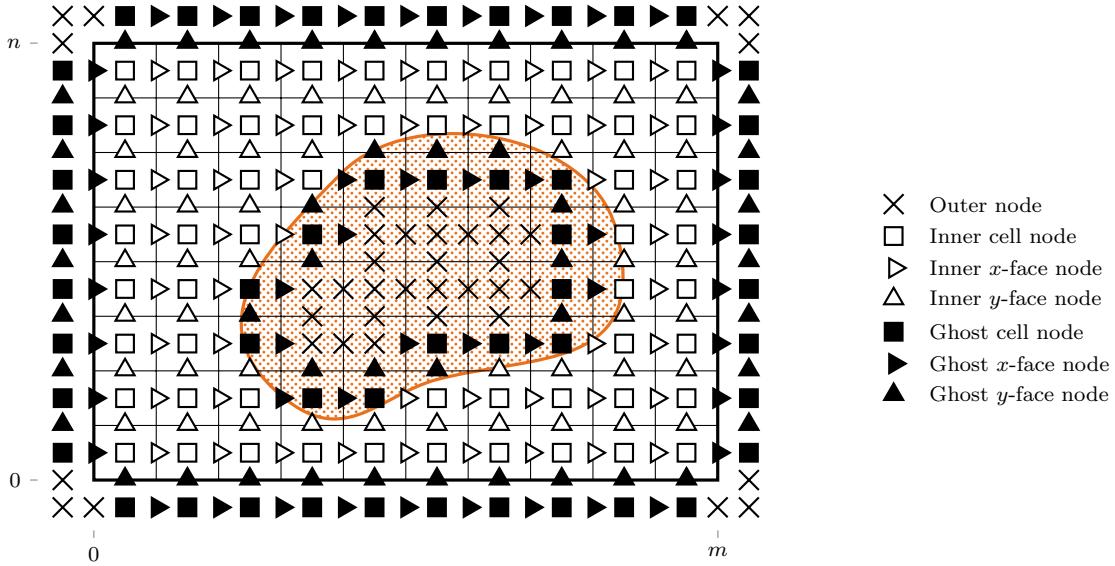
$$\nabla \boldsymbol{u}^{n+1} \cdot \boldsymbol{n} = \boldsymbol{N}\,, \qquad \text{on } \partial\Omega_N \cup \Gamma_N\,. \tag{48d}$$

In Eq. (48a), errors introduced by the terms $\frac{1}{h_t}\left(\boldsymbol{u}^{n+1} - \boldsymbol{u}^n\right)$ and $\nabla\cdot(\boldsymbol{u}^{n+1} \otimes \boldsymbol{u}^n)$ are collected in $\mathcal{O}(h_t)$; the pressure gradient term ∇p^{n+1} remains undefined. To solve the velocity-pressure coupling, we use the rotational incremental pressure-correction scheme [18, 19], in which Eq. (48) is divided in two subproblems: a prediction step that does not satisfies the solenoidal constraint (Eq. 48b), and a correction step. The prediction step consists in solving the advection-diffusion problem

$$\frac{1}{h_t}\left(\boldsymbol{u}^{*} - \boldsymbol{u}^n\right) + \nabla\cdot(\boldsymbol{u}^{*} \otimes \boldsymbol{u}^n) = -\nabla p^{n} + \nu \Delta \boldsymbol{u}^{*} + \boldsymbol{f} + \mathcal{O}(h_t)\,, \qquad \text{in } \Omega_i\,, \tag{49a}$$

$$\boldsymbol{u}^{*} = \boldsymbol{D}\,, \qquad \text{on } \partial\Omega_D \cup \Gamma_D\,, \tag{49b}$$

$$\nabla \boldsymbol{u}^{*} \cdot \boldsymbol{n} = \boldsymbol{N}\,, \qquad \text{on } \partial\Omega_N \cup \Gamma_N\,, \tag{49c}$$



(a) Sketch of a Cartesian grid with node types that can be used to discretize Fig. 1a.

Figure 19: Definitions and notations of the computational domain, the immersed boundary, the mesh, and node types.

in which the pressure p^{n+1} have been replaced by p^n to remove the velocity-pressure coupling. The solution $\boldsymbol{u}^*$ of Eq. (49) is a prediction of $\boldsymbol{u}^{n+1}$, for which $\nabla\cdot\boldsymbol{u}^*$ is not equal to zero, and $\boldsymbol{u}^{n+1}$ can be expressed from the difference between Eq. (48) and Eq. (49) multiplied by h_t

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^* - h_t\nabla\phi + \mathcal{O}(h_t)\,, \qquad \text{in } \Omega_i\,, \tag{50a}$$

$$\phi = p^{n+1} - p^n + \nu\,\nabla\cdot\boldsymbol{u}^*\,, \qquad \text{in } \Omega_i\,, \tag{50b}$$

where the decomposition of the Laplace operator $\Delta = \nabla(\nabla\cdot) - \nabla\times\nabla\times$ leads to the term $\nu\,\nabla\cdot\boldsymbol{u}^*$ in Eq. (50b), and the seemingly missing terms $-h_t\,\nabla\cdot\left((\boldsymbol{u}^{n+1}-\boldsymbol{u}^*)\otimes\boldsymbol{u}^n\right) + \nu h_t\,\nabla\times\nabla\times(\boldsymbol{u}^{n+1}-\boldsymbol{u}^*)$ are in fact collected in $\mathcal{O}(h_t)$. In Equation (50) the *pressure increment* ϕ is obtained by solving the Poisson problem

$$\nabla\cdot(h_t\nabla\phi) = \nabla\cdot\boldsymbol{u}^* + \mathcal{O}(h_t)\,, \qquad \text{in } \Omega_i\,, \tag{51a}$$

$$\nabla\phi\cdot\boldsymbol{n} = 0\,, \qquad \text{on } \partial\Omega_D\cup\Gamma_D\,, \tag{51b}$$

$$\phi = 0\,, \qquad \text{on } \partial\Omega_N\cup\Gamma_N\,. \tag{51c}$$

Equations (49-51) form the time-discretized Navier-Stokes equations used in this paper.

6.2. Spatial discretization

A staggered discretization of fields is used: to the cell nodes, x-face nodes $\boldsymbol{X}_{i+\frac{1}{2},j}$, $\forall(i,j)\in[\![0,m]\!]\times[\![0,n+1]\!]$, and y-face nodes $\boldsymbol{X}_{i,j+\frac{1}{2}}$, $\forall(i,j)\in[\![0,m+1]\!]\times[\![0,n]\!]$, are considered. Vector fields are discretized on face nodes: $\boldsymbol{U}=(U,V)^\intercal$ where $U_{i+\frac{1}{2},j}=u(\boldsymbol{X}_{i+\frac{1}{2},j})$ and $V_{i,j+\frac{1}{2}}=v(\boldsymbol{X}_{i,j+\frac{1}{2}})$, and $\boldsymbol{F}=(F,G)^\intercal$ where $F_{i+\frac{1}{2},j}=f(\boldsymbol{X}_{i+\frac{1}{2},j})$ and $G_{i,j+\frac{1}{2}}=g(\boldsymbol{X}_{i,j+\frac{1}{2}})$. Cell types are assigned using the method Eq. (2) defined in Sect. 2.1. It is applied to cell nodes, x-face nodes, and y-face nodes independently, as shown in Fig. 19. This way ensures that the discretized operators defined below are well-defined. Namely, two cell nodes that surround an inner face node are either inner or ghost, two face nodes that surround an inner cell node are either inner or ghost, and y-face nodes (resp. x-face nodes) that surround an x-face node (resp. y-face node) are either inner or ghost. The set of inner, ghost, outer x-face indices are C_i^x, C_g^x, C_o^x, respectively, and the set of inner, ghost, outer y-face indices are C_i^y, C_g^y, C_o^y.

Each term of Eq. (49—51) discretizes with second-order centered schemes as described in Appendix A; this results in

$$\Big(\frac{1}{h_t}\boldsymbol{I}_i + \boldsymbol{A}(\boldsymbol{U}^n) - \nu\boldsymbol{L} + \boldsymbol{E}\Big)\boldsymbol{U}^* = \frac{1}{h_t}\boldsymbol{U}^n - \boldsymbol{G}P^n + \boldsymbol{F} + \boldsymbol{B} + \mathcal{O}(h_t) + \mathcal{O}(h^2) + \mathcal{O}(h^p)\,, \quad \text{in } C_{ig}^{xy}\,, \tag{52a}$$

$$h_t(L+E)\Phi = D\boldsymbol{U}^* + B + \mathcal{O}(h_t) + \mathcal{O}(h^2) + \mathcal{O}(h^{q-1})\,, \quad \text{in } C_{ig}\,, \tag{52b}$$

$$\boldsymbol{U}^{n+1} = \boldsymbol{U}^* - h_t\boldsymbol{G}\Phi + \mathcal{O}(h_t) + \mathcal{O}(h^2)\,. \quad \text{in } C_i^{xy}\,, \tag{52c}$$

$$P^{n+1} = P^n + \Phi - \nu D\boldsymbol{U}^* + \mathcal{O}(h^2)\,, \quad \text{in } C_i\,, \tag{52d}$$

where $C_{ig}^{xy} = C_{ig}^x \cup C_{ig}^y$, and $\boldsymbol{I}_i$ is the identity matrix over the inner cells and the null matrix elsewhere. As in Sect. 2.1, we closed the two first equations using domain and immersed boundary conditions: $\boldsymbol{E}$ and $\boldsymbol{B} + \mathcal{O}(h^p)$ in Eq. (52a) and E and $B + \mathcal{O}(h^{q-1})$ in Eq. (52b). We recall that $(\boldsymbol{E}\boldsymbol{U}^*)_{i,j} = \boldsymbol{B}_{i,j} = 0$ over C_i^{xy}, $(E\Phi)_{i,j} = B_{i,j} = 0$ over C_i, and the other terms evaluates to zero over C_g^{xy} and C_g accordingly. Although it is not trivial that terms $\boldsymbol{A}(\boldsymbol{U}^n)$, $D\boldsymbol{U}^*$, $\boldsymbol{G}P^n$, and $\boldsymbol{G}\Phi$ are well defined over C_i^{xy}, so let's have a closer look to these terms.

According to interpolations and finite-differences involved in Eq. (A.2), $\boldsymbol{A}(\boldsymbol{U}^n)$ will be defined over C_i^{xy}, if $\boldsymbol{U}^n$ is defined over C_{ig}^{xy}. Therefore the whole iterative process Eq. (52) must output a $\boldsymbol{U}^{n+1}$ defined over C_{ig}^{xy}. To this end, the right-hand side of Eq. (52c) must be extended over the ghost nodes: $\boldsymbol{U}^*$ is already extended since it is a solution of Eq. (52a), although $\boldsymbol{G}\Phi$ cannot be defined over all ghost nodes (because is would require to define Φ over some outer nodes). An extrapolation of the velocity field must then be added:

$$(\boldsymbol{I}_i + \boldsymbol{E})\tilde{\boldsymbol{U}}^{n+1} = \boldsymbol{U}^{n+1} + \boldsymbol{B} + \mathcal{O}(h^p)\,, \quad \text{in } C_{ig}^{xy}\,, \tag{53}$$

where $\tilde{\boldsymbol{U}}^{n+1}$ is the new output of Eq. (52).

According to Eq. (A.3b), $\boldsymbol{G}P^n$ will be defined over C_i^{xy}, if P^n is defined over C_i plus ghost nodes next to an inner face node. Therefore the whole iterative process Eq. (52) must output a P^{n+1} defined over C_i plus ghost nodes next to an inner face node. To this end, the right-hand side of Eq. (52d) must be extended over the ghost nodes next to an inner face node: Φ is already defined over C_{ig} since it is a solution of Eq. (52b), $-\nu D\boldsymbol{U}^*$ is also defined over ghost nodes next to an inner face node since the opposite face node is necessarily a ghost node where $\boldsymbol{U}^*$ is defined and a finite-difference can be applied. As a consequence, no extrapolation is necessary for P^{n+1}.

To summarize, the fully discretized Navier-Stokes equations with immersed boundaries are

$$\Big(\frac{1}{h_t}\boldsymbol{I}_i + \boldsymbol{A}(\boldsymbol{U}^n) - \nu\boldsymbol{L} + \boldsymbol{E}\Big)\boldsymbol{U}^* = \frac{1}{h_t}\boldsymbol{U}^n - \boldsymbol{G}P^n + \boldsymbol{F} + \boldsymbol{B} + \mathcal{O}(h_t) + \mathcal{O}(h^2) + \mathcal{O}(h^p)\,, \quad \text{in } C_{ig}^{xy}\,, \tag{54a}$$

$$h_t(L+E)\Phi = D\boldsymbol{U}^* + B + \mathcal{O}(h_t) + \mathcal{O}(h^2) + \mathcal{O}(h^{q-1})\,, \quad \text{in } C_{ig}\,, \tag{54b}$$

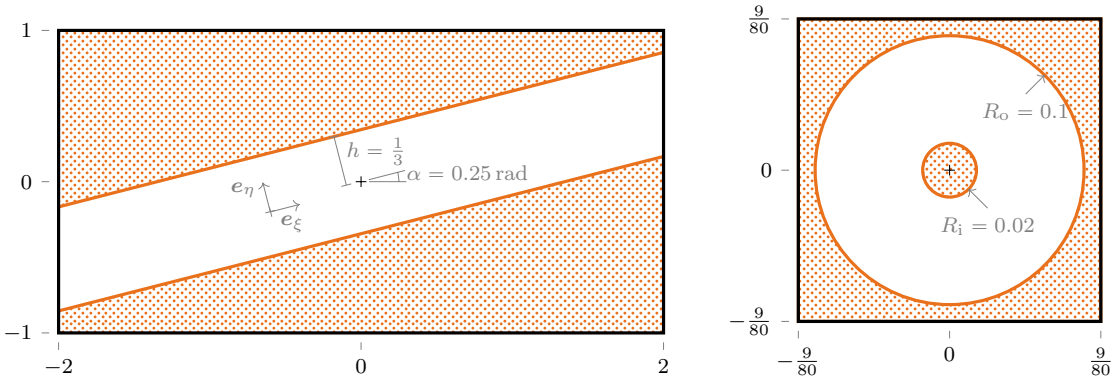
$$(\boldsymbol{I}_i + \boldsymbol{E})\boldsymbol{U}^{n+1} = \boldsymbol{U}^* - h_t\boldsymbol{G}\Phi + \boldsymbol{B} + \mathcal{O}(h_t) + \mathcal{O}(h^2) + \mathcal{O}(h^p)\,. \quad \text{in } C_{ig}^{xy}\,, \tag{54c}$$

$$P^{n+1} = P^n + \Phi - \nu D\boldsymbol{U}^* + \mathcal{O}(h^2)\,, \quad \text{in } C_i\,, \tag{54d}$$

with the precaution that $\boldsymbol{U}^0$ and P^0 have also been extrapolated over the ghost nodes. Immersed boundary conditions are applied three times: in Eq. (54a) and Eq. (54b), and an immersed boundary extrapolation is applied Eq. (54c).

7. Numerical simulations with the incompressible Navier-Stokes problem

This section presents simulations of four cases of the Navier-Stokes problem, and compares results obtained with both shifted direct and shifted linear methods. The size of the discretization stencil is the same for prediction and correction steps: $p = q = 2$, which equals to one for the direct method and two for the linear method.



(a) Poiseuille flow.

(b) Couette flow.

Figure 20: Computational domain and immersed boundary setups for the Poiseuille flow and the Couette flow. A thick black line shows the computational domain, a thick colored line shows the immersed boundary and colored dots shades the outer domain.

Linear system solvers. The same solvers are used than in Sect. 5, except for the prediction step for which a Jacobi preconditioner is enough. Stationarity tolerance is set to 10^{-8}. Details of the linear system solvers are given in Appendix B.

Diagnostics. For the two first cases, an analytical solution is available. Velocity and pressure error fields, $\hat{\boldsymbol{U}} - \boldsymbol{U}$ and $\hat{P} - P$, are computed at inner nodes, and their $\mathcal{L}^2$ and $\mathcal{L}^\infty$ norms are computed as in Eq. (34). Diagnostics for the two last cases will be presented in Sect. 7.3.

7.1. Poiseuille flow

Computational domain and grid. This computation takes place in the Cartesian domain $\Omega = [-2, +2] \times [-1, +1]$. The immersed boundary is a tilted channel of half-height $h = \frac{1}{3}$, and the angle between the x-axis and the axis of the channel is $\alpha = 0.25\,\mathrm{rad}$, as shown in Fig. 20a. Dirichlet boundary conditions are used at all boundaries, taking values of the analytical solution presented below. The same number of cells is used in both directions, so that the cell size ratio is always $a = 2$.

Analytical solution. Using the coordinate system $(\boldsymbol{e}_\xi, \boldsymbol{e}_\eta)$, aligned with the channel (see Fig. 20a), the classical solution of the Poiseuille flow is

$$\boldsymbol{u} = u_\xi \boldsymbol{e}_\xi\,, \qquad u_\xi(\eta) = u_m \left(1 - \frac{\eta^2}{h^2}\right)\,, \tag{55a}$$

$$p(\xi) = p_0 + \left(-\frac{\partial p}{\partial \xi}\right)\xi\,, \qquad \left(-\frac{\partial p}{\partial \xi}\right) = \frac{2\nu}{h^2} u_m\,, \tag{55b}$$

where u_m is the maximum velocity and p_0 is the pressure at the origin. The Reynolds number of the flow is $Re = 2hu_m/\nu$. In the following, $p_0 = 0$, $u_m = 1/(2h)$, $\nu = 1/Re$, and $Re = 20$. The average velocity and pressure of the flow are $\langle \boldsymbol{u} \rangle = \frac{2}{3} u_m \boldsymbol{e}_\xi$ and $\langle p \rangle = p_0 = 0$.

Results. Figure 21 shows convergence results in terms of velocity and pressure errors $\mathcal{L}^\infty(\hat{\boldsymbol{U}} - \boldsymbol{U})$, $\mathcal{L}^2(\hat{\boldsymbol{U}} - \boldsymbol{U})$, $\mathcal{L}^\infty(\hat{P} - P)$, and $\mathcal{L}^2(\hat{P} - P)$ using shifted direct and shifted linear methods. For both methods, velocity errors and pressure $\mathcal{L}^2$ error show a second-order limiting behavior, while pressure $\mathcal{L}^\infty$ error show a limiting behavior between 1.5 and 2. Error levels of the linear method are roughly twice as large as error levels of the direct method.

As we choose a consistent size of the discretization stencil ($p = q = 2$), the limiting behavior of Eq. (54b) is only of first order. This result is therefore better than expected. We suspect superconvergence due to the one-dimensional linear profile of the pressure field.

7.2. Circular Couette flow

Computational domain and grid. The computational domain is $\Omega = [-\frac{9}{80}, +\frac{9}{80}] \times [-\frac{9}{80}, +\frac{9}{80}]$, and the immersed boundary is two co-axial cylinders shown in Fig. 20b. The radius of the inner and outer cylinders are $R_\mathrm{i} = 0.02$ and $R_\mathrm{o} = 0.1$, respectively. Dirichlet boundary conditions are used at all boundaries, taking values of the analytical solution presented below. The number of cells in the x-axis is four times the number of cells in the y-axis, such that the cell size ratio is always $a = 4$.

Analytical solution. The cylinders are in rotation with angular velocities ω_i and ω_o, respectively. They induce motion to the fluid, and the classical solution of the Couette flow using the polar coordinates $(\boldsymbol{e}_r, \boldsymbol{e}_\theta)$ is

$$\boldsymbol{u} = u_\theta \boldsymbol{e}_\theta \,, \qquad u_\theta(r) = \frac{A}{2} r + \frac{B}{r} \,, \tag{56a}$$

$$p(r) = p_0 + \frac{A^2}{8} r^2 + AB \ln(r) - \frac{B^2}{2r^2} \,, \tag{56b}$$

where p_0 is an integration constant, and constants A and B are deduced from parameters:

$$A = 2\frac{\omega_\mathrm{o} R_\mathrm{o}^2 - \omega_\mathrm{i} R_\mathrm{i}^2}{R_\mathrm{o}^2 - R_\mathrm{i}^2} \,, \qquad B = (\omega_\mathrm{i} - \omega_\mathrm{o}) \frac{R_\mathrm{i}^2 R_\mathrm{o}^2}{R_\mathrm{o}^2 - R_\mathrm{i}^2} \,. \tag{57}$$

In the following $p_0 = 0$, $\omega_\mathrm{i} = 0.5\,\mathrm{rad}$, and $\omega_\mathrm{o} = 0\,\mathrm{rad}$.

Results. Figure 22 shows convergence results in terms of velocity and pressure errors $\mathcal{L}^\infty(\hat{\boldsymbol{U}} - \boldsymbol{U})$, $\mathcal{L}^2(\hat{\boldsymbol{U}} - \boldsymbol{U})$, $\mathcal{L}^\infty(\hat{P} - P)$, and $\mathcal{L}^2(\hat{P} - P)$. A minimum resolution of 64 cells in the low-resolved axis is required in order to properly represent the inner boundary. Velocity-related indicators show a second-order limiting behavior, whereas $\mathcal{L}^\infty(\hat{P} - P)$ show only a first-order limiting behavior and $\mathcal{L}^2(\hat{P} - P)$ show a in-between behavior. As with the Poiseuille flow, error levels of the linear method are roughly twice as large as error levels of the direct method. These results are again consistent with the chosen size of the discretization stencil.

7.3. Flows around circular and elliptic cylinders

The following numerical simulations validate the method on stationary flows around two cylinders of different shapes: circular and elliptic. The former have been extensively studied in the literature, while the latter demonstrates the interest of rectangular cells.

Computational domain and grid. The computational domain represents a tank of the 2-dimensional real space, whose dimensions are $\Omega = [-15, +30] \times [-15, +15]$, and the flow goes from left to right. Domain boundary conditions are: uniform inlet of velocity $(u_\infty, 0)^\intercal$ at the left boundary, slip at top and bottom boundaries, and Neumann at the right boundary. The cylinder is placed at the origin. A regular grid of ratio $a = 3.0$ is placed over a small *domain of interest,* of dimensions $[-1, +3] \times [-1.5, +1.5]$, while exponential grids are placed around. On the y-axis, we used that 512 nodes in the domain of interest and 256 nodes in each exponential region. On the x-axis, we used 1536 nodes in the domain of interest to have $a = 3.0$ and 768 nodes in each stretched region. Figure 23 shows the computational domain, the grid and the domain of interest.

The circular cylinder has diameter $d = 1$; the elliptic cylinder has major axis $d = 1$ and axis ratio 0.2; thus the characteristic length is d for both cylinders. The major axis of the elliptic cylinder is aligned with the x-axis of the coordinate frame $\mathfrak{R}' = (\boldsymbol{e}_{x'}, \boldsymbol{e}_{y'})$, which is rotated by α with respect to the original coordinate frame $\mathfrak{R} = (\boldsymbol{e}_x, \boldsymbol{e}_y)$ where the x-axis is aligned with the stream. The angle α is also the *angle of incidence;* it is set to -80° for the elliptical cylinder, and can be considered set to 0° for the circular cylinder. Figure 24 shows both cylinders with a description of their dimensions.

We choose the free-stream velocity $u_\infty = 1$ and, and we set the viscosity according to the desired value of the Reynolds number. For the elliptic cylinder, the Reynolds number is based on the long axis, thus $Re = du_\infty/\nu$ both cases. The Reynolds number is set to $Re = 40$ for the circular cylinder, and set to $Re = 20$ for the elliptic cylinder.

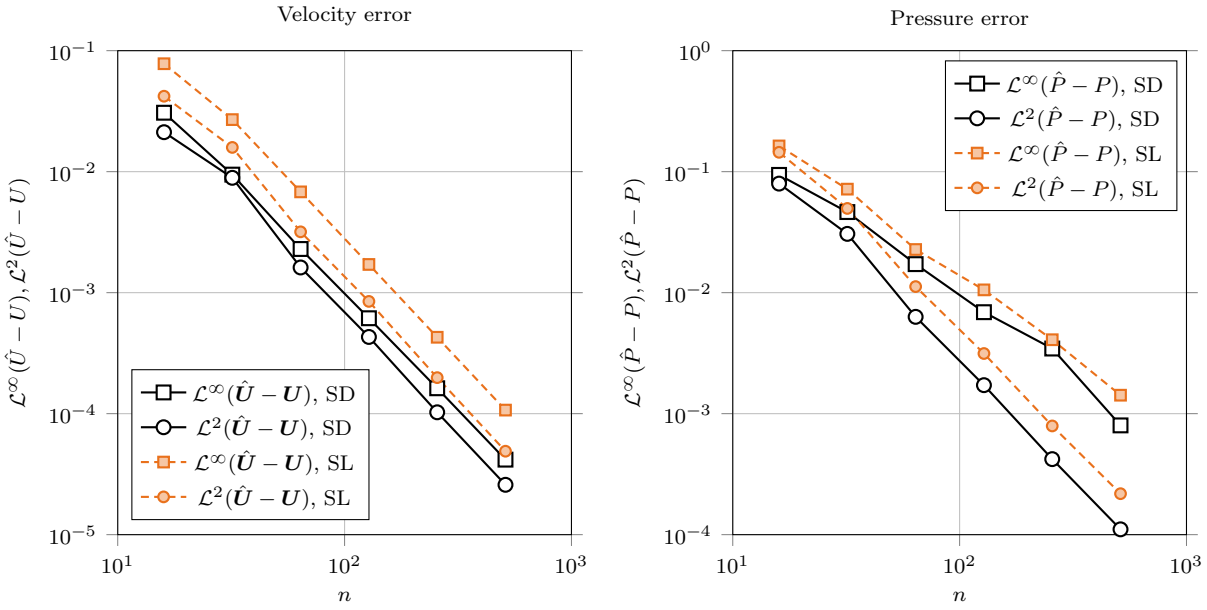


Figure 21: Norms $\mathcal{L}^2$ and $\mathcal{L}^\infty$ of the velocity error $\hat{\boldsymbol{U}}-\boldsymbol{U}$ and the pressure error $\hat{P}-P$ for the Poiseuille flow using both the shifted direct (SD) and shifted linear (SL) methods. The x-axis represents $1/h = 1/\max\{h_x, h_y\} = \min\{n_x, n_y\}$, which is always equal to n in this section.

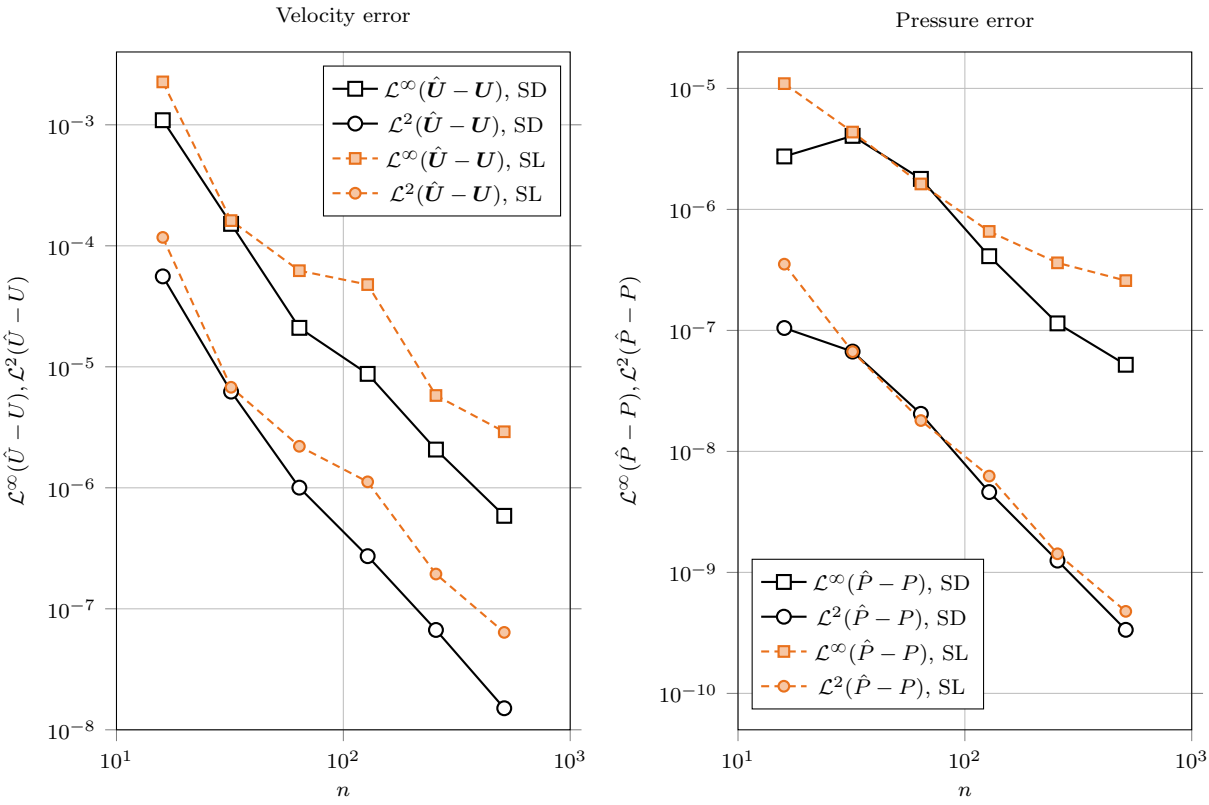


Figure 22: Norms $\mathcal{L}^2$ and $\mathcal{L}^\infty$ of the velocity error $\hat{\boldsymbol{U}}-\boldsymbol{U}$ and the pressure error $\hat{P}-P$ for the Couette flow using both the shifted direct (SD) and shifted linear (SL) methods. The x-axis represents $1/h = 1/\max\{h_x, h_y\} = \min\{n_x, n_y\}$, which is always equal to n in this section.

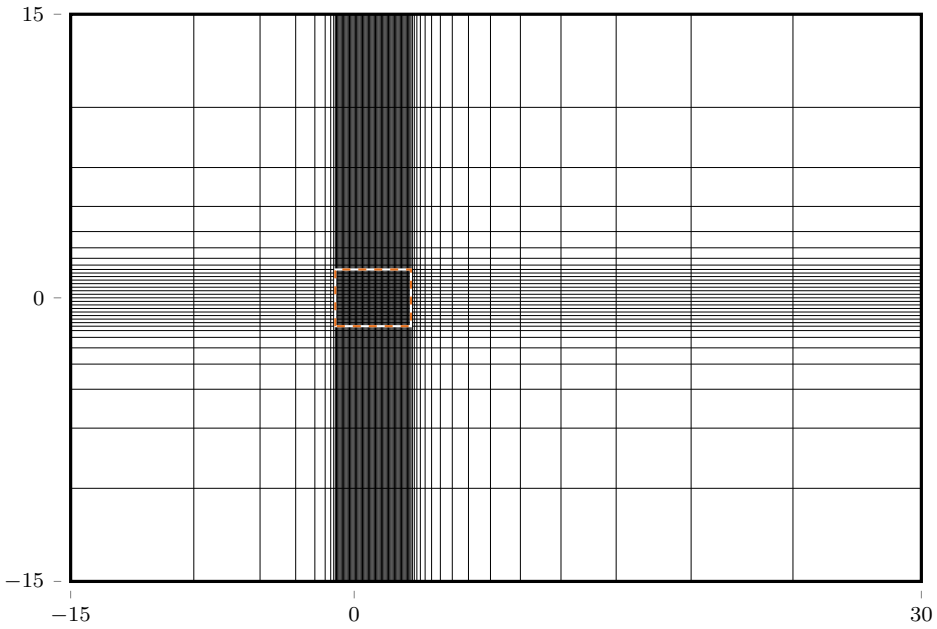
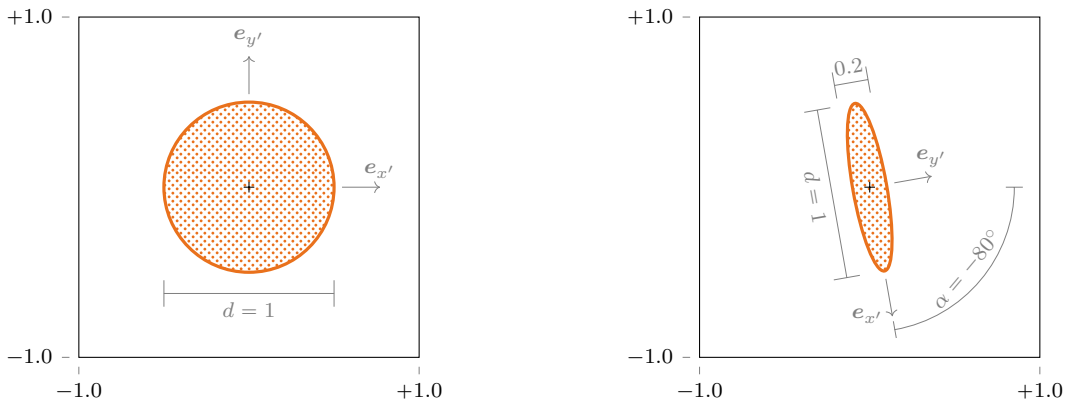


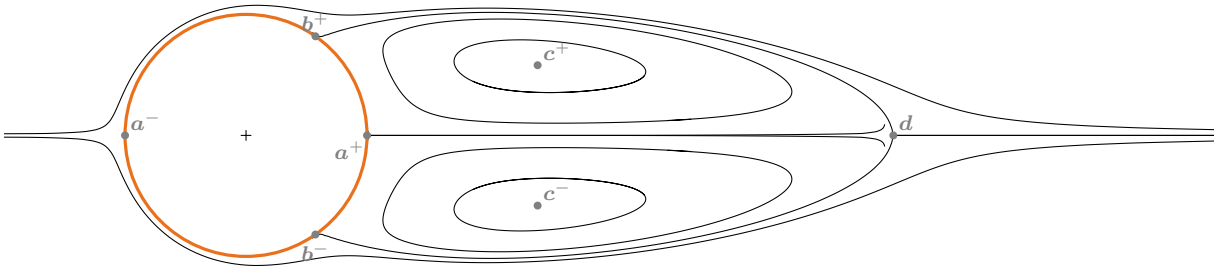
Figure 23: Computational domain and grid used to simulate the flow around the two cylinders, Sect. 7.3. Only one grid line out of 16 is drawn for the sake of readability. The regular sub-domain region is outlined in dashed colored lines.



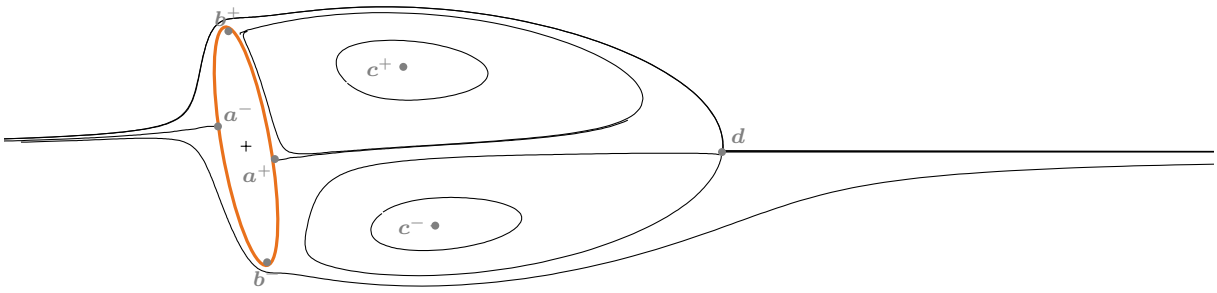
(a) Circular cylinder.

(b) Elliptic cylinder.

Figure 24: Dimensions of circular and elliptic cylinders. The boundary is shown by a thick colored line, and the outer domain is shaded by dots. Axes of the cylinder reference frame $\mathfrak{R}' = (\boldsymbol{e}_{x'}, \boldsymbol{e}_{y'})$ are also shown.



(a) Circular cylinder ($Re = 40$).



(b) Elliptic cylinder ($Re = 20$).

Figure 25: Points of interest $\boldsymbol{a}^-$, $\boldsymbol{a}^+$, $\boldsymbol{b}^-$, $\boldsymbol{b}^+$, $\boldsymbol{c}^-$, $\boldsymbol{c}^+$, and $\boldsymbol{d}$ of the steady wake flow. The streamlines are extracted from computations.

Diagnostics. As no analytical solution exists, we compare our results to experimental and numerical data found in the literature. First, we compare the position of seven different *points of interest:* four separation points on the surface of the obstacle $\boldsymbol{a}^-$, $\boldsymbol{a}^+$, $\boldsymbol{b}^-$, and $\boldsymbol{b}^+$; two vortex cores $\boldsymbol{c}^-$ and $\boldsymbol{c}^+$, and the steady point at the right of the wake vortices $\boldsymbol{d}$. Figure 25 shows these points for the two cylinders. Cartesian and polar coordinates of a point of interest is noted with the subscript $\boldsymbol{a} = (a_x, a_y)^\intercal$ for the former and $\boldsymbol{a} = (a_r, a_\theta)^\intercal$ for the latter. Next, we compute the drag and lift coefficients

$$C_\mathrm{D} = \frac{\mathcal{F}_x}{\frac{1}{2} d u_\infty^2}\,, \qquad\qquad C_\mathrm{L} = \frac{\mathcal{F}_y}{\frac{1}{2} d u_\infty^2}\,, \tag{58}$$

where $\mathcal{F}_x$ and $\mathcal{F}_y$ are the streamwise and the transverse components of the *volumic drag force* $\boldsymbol{\mathcal{F}}$. This force corresponds to the surface force of the fluid integrated over the cylinder Γ; it decomposes into pressure and viscous components

$$\boldsymbol{\mathcal{F}} = \boldsymbol{\mathcal{F}}_p + \boldsymbol{\mathcal{F}}_\nu = \int_\Gamma -p\boldsymbol{n}\mathrm{d}\ell + \int_\Gamma 2\nu\boldsymbol{\epsilon}\boldsymbol{n}\mathrm{d}\ell\,, \tag{59}$$

Data soucre	λ	ℓ/d	a/d	b/d	$\theta/^\circ$
Coutanceau and Bouard [21]*	0.024	2.04	0.73	0.58	—
	0	2.13	0.76	0.59	53.5
Linnick and Fasel [22]	0.023	2.28	0.72	0.60	53.6
Calhoun [23]	—	2.18	—	—	54.2
Le et al. [24]	—	2.22	—	—	53.6
Taira and Colonius [25]	1/60	2.30	0.73	0.60	53.7
Berthelsen and Faltinsen [26]	—	2.29	0.72	0.60	53.9
present (direct/linear method)	1/30	2.27	0.72	0.59	53.1

Table 5: Lengths of points of interest for the circular cylinder ($Re = 40$). This table follows the notation used in the literature: $\ell = d_x - a_x^+$, $a = c_x^+ - a_x^+$, $b = c_y^+ - c_y^-$, and θ is the angular coordinate of $\boldsymbol{b}^+$. The coefficient λ represent the size ratio between the cylinder and the tank height (i. e. the height of the computational domain). Both direct and linear method gives the same result up to presented precision.

$\boldsymbol{a}^-/d$	$\boldsymbol{a}^+/d$	$b_\theta^-/^\circ$	$b_\theta^+/^\circ$	$\boldsymbol{c}^-/d$	$\boldsymbol{c}^+/d$	$\boldsymbol{d}/d$
$\begin{pmatrix}-0.11\\0.08\end{pmatrix}$	$\begin{pmatrix}0.11\\-0.06\end{pmatrix}$	0.3	178.7	$\begin{pmatrix}0.78\\-0.33\end{pmatrix}$	$\begin{pmatrix}0.65\\0.33\end{pmatrix}$	$\begin{pmatrix}1.97\\-0.02\end{pmatrix}$

Table 6: Coordinates of points of interest for the elliptic cylinder ($Re = 20$). Both direct and linear method gives the same result up to presented precision.

where p is the kinematic pressure and $\boldsymbol{\epsilon} = \frac{1}{2}\nu(\nabla\boldsymbol{u} + \nabla\boldsymbol{u}^\intercal)$ is the strain rate tensor. Drag and lift coefficients can then be decomposed into pressure and viscous components.

$$C_{\mathrm{L}p} = \frac{\mathcal{F}_{p_y}}{\frac{1}{2}du_\infty^2}\,, \qquad C_{\mathrm{L}\nu} = \frac{\mathcal{F}_{\nu y}}{\frac{1}{2}du_\infty^2}\,, \qquad C_{\mathrm{D}p} = \frac{\mathcal{F}_{p_x}}{\frac{1}{2}du_\infty^2}\,, \qquad C_{\mathrm{D}\nu} = \frac{\mathcal{F}_{\nu x}}{\frac{1}{2}du_\infty^2}\,. \tag{60a}$$

The numerical evaluation of $\boldsymbol{\mathcal{F}}_p$ and $\boldsymbol{\mathcal{F}}_\nu$ approximates Γ by cell-wise line segments upon which p and $\boldsymbol{\epsilon}$ are considered uniform. Second-order extrapolations are used to compute p and $\boldsymbol{\epsilon}$ on the middle of the line segments, using nodes values from the inner domain only. Finally, we give a closer look at the normalized pressure profile π around the cylinder, defined by

$$C_p = \frac{p}{\frac{1}{2}du_\infty^2}\,. \tag{61}$$

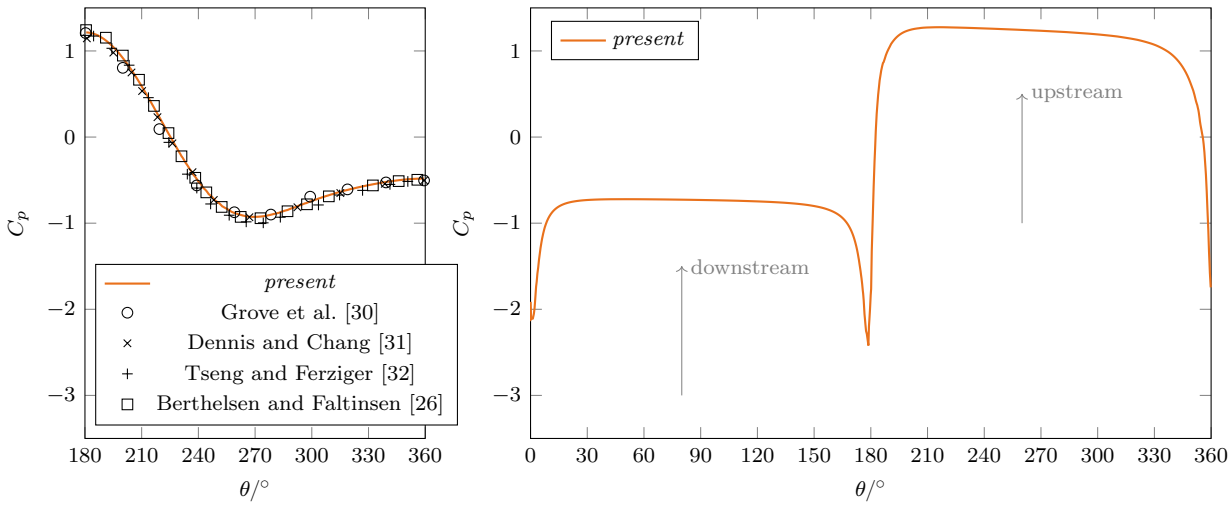
Results. Both simulations converge towards a steady state containing two recirculation regions. Both direct and linear methods gives very similar results. Coordinates of the points of interest are reported in Tab. 5 for the circular cylinder and in Tab. 6 for the elliptic cylinder. Good agreement with literature data is observed for the circular cylinder; Tab. 5 also shows data from the literature. Lengths for the elliptic cylinder are coherent with the streamline visualizations of Yoon et al. [20]. Drag coefficients are reported in Tab. 7 for the circular cylinder (lift coefficients are below 10^{-8}). The value is in the range of values found in the literature. Drag and Lift coefficients are reported in Tab. 8 for the elliptic cylinder, along with values obtained in the literature. Values are also in the range of values found in the literature. Most of the drag is due to the pressure drag. It is interesting to note that pressure and viscosity have opposite contribution to lift. Finally, profiles of normalized pressure C_p as a function of θ are shown on Fig. 26. The profile for the circular cylinder Fig. 26a follows closely the profiles found in the literature. The profile for the elliptic cylinder Fig. 26b has a maximum and minimum pressure than the circular cylinder. Although, very low pressure are seen at the tips of the ellipse ($\theta = 0^\circ, 180^\circ$).

Data source	$C_{\mathrm{D}p}$	$C_{\mathrm{D}\nu}$	C_{D}
Tritton [27]*			1.57
Linnick and Fasel [22]			1.54
Calhoun [23]			1.62
Le et al. [24]			1.56
Taira and Colonius [25]			1.54
Berthelsen and Faltinsen [26]			1.59
present (direct/linear method)	1.026	0.533	1.559

Table 7: Drag coefficients around the circular cylinder ($Re = 40$). Lifts coefficients have been checked below 10^{-8} for the present study. Both direct and linear method gives the same result up to presented precision.

Data source	$C_{\mathrm{D}p}$	$C_{\mathrm{D}\nu}$	C_{D}	$C_{\mathrm{L}p}$	$C_{\mathrm{L}\nu}$	C_{L}
D'alessio and Dennis [28]			2.116			0.256
Dennis and Young [29]			2.089			0.255
Yoon et al. [20]			2.102			0.252
present (direct/linear method)	1.864	0.266	2.130	0.303	−0.057	0.246

Table 8: Drag and lift coefficients around the elliptic cylinder inclined at 80° ($Re = 20$). Both direct and linear method gives the same result up to presented precision.



(a) Circular cylinder ($Re = 40$).

(b) Elliptic cylinder ($Re = 20$).

Figure 26: Normalized pressure profile C_p around the cylinder in the $\mathfrak{R}'$ coordinate frame.

8. Conclusions

We studied the size of the discretization stencil of the two Finite-Difference Ghost-Fluid Immersed Boundary methods defined in [3] and [4], that we called the linear and the direct method, respectively. We showed that the stencil size increases as cells become more and more rectangular, i. e. the aspect ratio of the cell increases, and we also showed that the direct method may become ill posed on rectangular cells. We proposed the Ghost Node Shifting Method to reduce the stencil size of any rectangular cell to the case of square cells, which also ensure the direct method to be well posed. We also showed that only the direct method is able to keep a stencil size of 1 while providing a second-order limiting behavior method with Dirichlet boundary conditions, and first-order limiting behavior method with Neumann boundary conditions.

We carried out numerical simulations demonstrating that shifted methods where able to achieve the desired limiting behavior, given an appropriate stencil size, see Tab. 2 for details. Simulations also showed that error levels of shifted methods are as least as low as error levels of original methods, and that the shifted direct method has less or the same error levels that the shifted linear method.

We also applied the shifted methods to the pressure-correction method to solve the incompressible Navier-Stokes problem with immersed boundaries. An emphasis has been given at each step of the pressure-correction method to produce a coherent solution with respect to the immersed boundaries: we showed that an additional extrapolation of the velocity field is necessary due to the nonlinear term. We were able to produce a method with a compact stencil, i. e. a stencil size is 1. Numerical simulation showed a second-order limiting behavior on the velocity and a first-to-second-order limiting behavior on the pressure. Finally we were able to perform numerical simulations of the flow around circular and elliptic cylinders with coherent results with respect to the literature.

Future works include applications of this method to other problems, studies of higher order discretization, in which the hollow case problem may become more cumbersome as interpolation stencil increases. An other work is therefore to find a robust handling of the hollow case problem. Finally, a customization of the linear system solver, such as performed by Coco and Russo [4] on multi-grid solvers, would improve the performances of the method on very large problems.

Acknowledgements

This study was carried out with financial support from the French National Research Agency (ANR) in the framework of the "Investments for the future" Programme IdEx Bordeaux (ANR-10-IDEX-03-02), Cluster of excellence CPU.

The authors wish to thank the Aquitaine Regional Council for the financial support towards a 432-processor cluster investment, located in the I2M laboratory.

Appendix A. Discrete operators for the incompressible Navier-Stokes problem

Let's give a closer look to the discretization of Eq. (49-51) on Ω_i. For the sake of readability, we use an intuitive notation of indexes, e. g. $\mathsf{C} = (i, j)$ (center), $\mathsf{L} = (i-1, j)$ (left), etc. Depending which nodes are considered the exact definition changes; it is given in Fig. A.27.

In the prediction step Eq. (49a), the diffusion term $-\nu\Delta\boldsymbol{u}^*$ discretizes as in Sect. 2 on the face nodes, and becomes the matrix product $-\nu\boldsymbol{L}\boldsymbol{U}^*$

$$(\boldsymbol{L}U^*)_\mathsf{C} = \frac{U^*_\mathsf{L} - U^*_\mathsf{C}}{h_x^2} + \frac{U^*_\mathsf{R} - U^*_\mathsf{C}}{h_x^2} + \frac{U^*_\mathsf{B} - U^*_\mathsf{C}}{h_y^2} + \frac{U^*_\mathsf{T} - U^*_\mathsf{C}}{h_y^2} + \mathcal{O}(h^2)\,, \qquad \forall \mathsf{C} \in C_i^x\,, \tag{A.1a}$$

$$(\boldsymbol{L}V^*)_\mathsf{C} = \frac{V^*_\mathsf{B} - V^*_\mathsf{C}}{h_y^2} + \frac{V^*_\mathsf{T} - V^*_\mathsf{C}}{h_y^2} + \frac{V^*_\mathsf{L} - V^*_\mathsf{C}}{h_x^2} + \frac{V^*_\mathsf{R} - V^*_\mathsf{C}}{h_x^2} + \mathcal{O}(h^2)\,, \qquad \forall \mathsf{C} \in C_i^y\,. \tag{A.1b}$$

Just as in Sect. 2, some $U^*_\mathsf{L}, U^*_\mathsf{R}, U^*_\mathsf{B}, U^*_\mathsf{T}$, and some $V^*_\mathsf{L}, V^*_\mathsf{R}, V^*_\mathsf{B}, V^*_\mathsf{T}$ correspond to ghost nodes, and a boundary condition treatment must be addressed. The advection term $\nabla\cdot(\boldsymbol{u}^*\otimes\boldsymbol{u}^n)$ discretizes into the matrix product

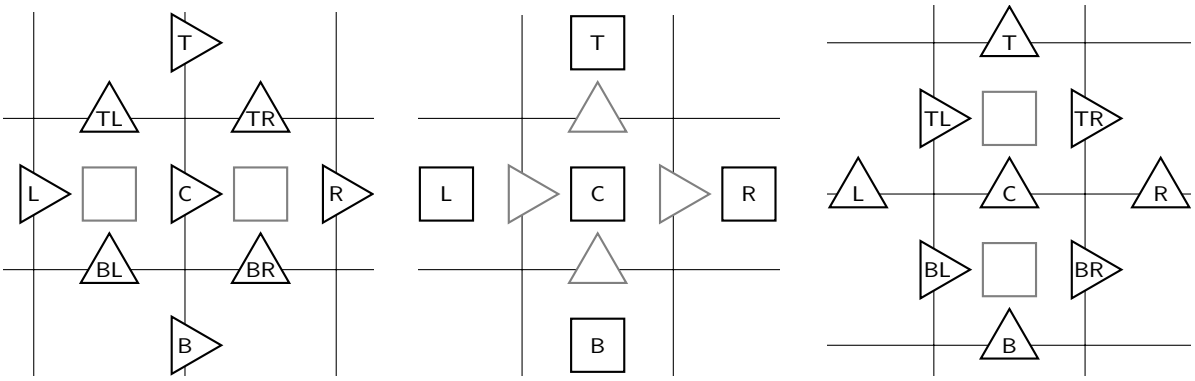


Figure A.27: Relative node position used in Eq. (A.2a, A.1a) (left), in Eq. (A.5) (center), and in Eq. (A.2b, A.1b) (right).

$\boldsymbol{A}(\boldsymbol{U}^n)\boldsymbol{U}^*$ combining second-order, centered interpolations and finite-differences

$$\begin{aligned}\big(\boldsymbol{A}(\boldsymbol{U}^n)\boldsymbol{U}^*\big)_{\mathsf{C}} = &-\frac{U^n_{\mathsf{L}}+U^n_{\mathsf{C}}}{4h_x}U^*_{\mathsf{L}} + \frac{U^n_{\mathsf{R}}+U^n_{\mathsf{C}}}{4h_x}U^*_{\mathsf{R}}\\ &+\left(-\frac{U^n_{\mathsf{L}}+U^n_{\mathsf{C}}}{4h_x}+\frac{U^n_{\mathsf{R}}+U^n_{\mathsf{C}}}{4h_x}-\frac{V^n_{\mathsf{BL}}+V^n_{\mathsf{BR}}}{4h_y}+\frac{V^n_{\mathsf{TL}}+V^n_{\mathsf{TR}}}{4h_y}\right)U^*_{\mathsf{C}} \qquad \forall \mathsf{C}\in C^x_i\,, \qquad \text{(A.2a)}\\ &+-\frac{V^n_{\mathsf{BL}}+V^n_{\mathsf{BR}}}{4h_y}U^*_{\mathsf{B}}+\frac{V^n_{\mathsf{TL}}+V^n_{\mathsf{TR}}}{4h_y}U^*_{\mathsf{T}}+\mathcal{O}(h^2)\,,\end{aligned}$$

$$\begin{aligned}\big(\boldsymbol{A}(\boldsymbol{U}^n)\boldsymbol{U}^*\big)_{\mathsf{C}} = &-\frac{U^n_{\mathsf{BL}}+U^n_{\mathsf{TL}}}{4h_x}V^*_{\mathsf{L}} + \frac{U^n_{\mathsf{BR}}+U^n_{\mathsf{TR}}}{4h_x}V^*_{\mathsf{R}}\\ &+\left(-\frac{U^n_{\mathsf{BL}}+U^n_{\mathsf{TL}}}{4h_x}+\frac{U^n_{\mathsf{BR}}+U^n_{\mathsf{TR}}}{4h_x}-\frac{V^n_{\mathsf{B}}+V^n_{\mathsf{C}}}{4h_y}+\frac{V^n_{\mathsf{T}}+V^n_{\mathsf{C}}}{4h_y}\right)V^*_{\mathsf{C}} \qquad \forall \mathsf{C}\in C^y_i\,. \qquad \text{(A.2b)}\\ &+-\frac{V^n_{\mathsf{B}}+V^n_{\mathsf{C}}}{4h_y}V^*_{\mathsf{B}}+\frac{V^n_{\mathsf{T}}+V^n_{\mathsf{C}}}{4h_y}V^*_{\mathsf{T}}+\mathcal{O}(h^2)\,,\end{aligned}$$

Here again some $U^*_{\mathsf{L}}, U^*_{\mathsf{R}}, U^*_{\mathsf{B}}, U^*_{\mathsf{T}}$, and some $V^*_{\mathsf{L}}, V^*_{\mathsf{R}}, V^*_{\mathsf{B}}, V^*_{\mathsf{T}}$ correspond to ghost nodes, but also some $U^n_{\mathsf{L}}, U^n_{\mathsf{R}}, U^n_{\mathsf{B}}, U^n_{\mathsf{T}}$, U^n_{BL}, U^n_{TL}, U^n_{BR}, U^n_{TR}, and some $V^n_{\mathsf{L}}, V^n_{\mathsf{R}}, V^n_{\mathsf{B}}, V^n_{\mathsf{T}}$, V^n_{BL}, V^n_{TL}, V^n_{BR}, V^n_{TR} does not correspond to inner nodes. Taking the assumption of Sect. 2.3 that the immersed boundary is not too hollow, none of theses values correspond to outer nodes and it is sufficient that $\boldsymbol{U}^n$ is being defined over C^{xy}_{ig}. The pressure gradient term $-\nabla p^n$ discretizes into the matrix product $-\boldsymbol{G}P^n$ using second-order centered finite-differences

$$(\boldsymbol{G}P^n)_{i+\frac{1}{2},j} = \frac{P^n_{i+1,j}-P^n_{i,j}}{h_x}+\mathcal{O}(h^2)\,, \qquad \forall (i,j)\in C^x_i\,, \qquad \text{(A.3a)}$$

$$(\boldsymbol{G}P^n)_{i,j+\frac{1}{2}} = \frac{P^n_{i,j+1}-P^n_{i,j}}{h_y}+\mathcal{O}(h^2)\,, \qquad \forall (i,j)\in C^y_i\,, \qquad \text{(A.3b)}$$

which requires that P^n is defined over C_i and some parts of C_g to be defined. More precisely, P^n must be defined on ghost cell nodes surrounded by a inner face node. In the correction step Eq. (51a), the right-hand side $\nabla\cdot\boldsymbol{u}^*$ discretizes into the matrix product $D\boldsymbol{U}^*$ using second-order, centered finite-differences

$$(D\boldsymbol{U}^*)_{i,j} = \frac{U^*_{i+\frac{1}{2},j}-U^*_{i-\frac{1}{2},j}}{h_x}+\frac{V^*_{i,j+\frac{1}{2}}-V^*_{i,j-\frac{1}{2}}}{h_y}+\mathcal{O}(h^2)\,. \qquad \forall (i,j)\in C_i\,, \qquad \text{(A.4)}$$

The right-hand side is always defined over C_i as $\boldsymbol{U}^*$ is solved over C^{xy}_{ig} (Eq. 52a). Finally the left-hand side $h_t\Delta\phi$ discretizes just as in Sect. 2 into the matrix product $h_t L\Phi+\mathcal{O}(h^2)$

$$(L\Phi)_{\mathsf{C}} = \frac{\Phi_{\mathsf{L}}-\Phi_{\mathsf{C}}}{h_x^2}+\frac{\Phi_{\mathsf{R}}-\Phi_{\mathsf{C}}}{h_x^2}+\frac{\Phi_{\mathsf{B}}-\Phi_{\mathsf{C}}}{h_y^2}+\frac{\Phi_{\mathsf{T}}-\Phi_{\mathsf{C}}}{h_y^2}+\mathcal{O}(h^2)\,, \qquad \forall (i,j)\in C_i\,. \qquad \text{(A.5)}$$

Here again some $\Phi_{\mathsf{L}}, \Phi_{\mathsf{R}}, \Phi_{\mathsf{B}}, \Phi_{\mathsf{T}}$ correspond to ghost nodes, and a boundary condition treatment must be addressed.

Appendix B. Detailed parameters of used linear system solvers

The following parameters has been used for the *hypre* preconditioners anytime they have been used.

SMG preconditioner (all cases)

```
      RelChange 1
      ZeroGuess set
    NumPreRelax 1
   NumPostRelax 1
```

BoomerAMG preconditioner (flower-shaped interface with Dirichlet boundary conditions)

```
StrongThreshold 0.25
    CoarsenType 6, Falgout coarsening
   AggNumLevels 1
      RelaxType 6, hybrid symmetric Gauss-Seidel or SSOR
     InterpType 0, classical modified interpolation
```

BoomerAMG preconditioner (circular interface with Neumann boundary conditions)

```
StrongThreshold 0.025
    CoarsenType 10, HMIS coarsening
   AggNumLevels 1
      RelaxType 6, hybrid symmetric Gauss-Seidel or SSOR
     InterpType 0, classical modified interpolation
```

References

[1] C. S. Peskin, Flow patterns around heart valves: A numerical method, Journal of Computational Physics 10 (2) (1972) 252–271, ISSN 0021-9991, doi:10.1016/0021-9991(72)90065-4.

[2] R. Mittal, G. Iaccarino, Immersed Boundary Methods, Annual Review of Fluid Mechanics 37 (1) (2005) 239–261, doi:10.1146/annurev.fluid.37.061903.175743.

[3] R. Mittal, H. Dong, M. Bozkurttas, F. M. Najjar, A. Vargas, A. von Loebbecke, A versatile sharp interface immersed boundary method for incompressible flows with complex boundaries, Journal of Computational Physics 227 (10) (2008) 4825–4852, ISSN 0021-9991, doi:10.1016/j.jcp.2008.01.028.

[4] A. Coco, G. Russo, Finite-difference ghost-point multigrid methods on Cartesian grids for elliptic problems in arbitrary domains, Journal of Computational Physics 241 (2013) 464–501, ISSN 0021-9991, doi:10.1016/j.jcp.2012.11.047.

[5] S. Schaffer, A Semicoarsening Multigrid Method for Elliptic Partial Differential Equations with Highly Discontinuous and Anisotropic Coefficients, SIAM Journal on Scientific Computing 20 (1) (2006) 228–242, doi:10.1137/S1064827595281587.

[6] P. N. Brown, R. D. Falgout, J. E. Jones, Semicoarsening Multigrid on Distributed Memory Machines, SIAM Journal on Scientific Computing 21 (5) (2000) 1823–1834, ISSN 1064-8275, doi:10.1137/S1064827598339141.

[7] S. F. Ashbf, R. D. Falgout, A Parallel Multigrid Preconditioned Conjugate Gradient Algorithm for Groundwater Flow Simulations,, Nuclear Science and Engineering 126 (1996) 145–159.

[8] R. D. Falgout, J. E. Jones, Multigrid on Massively Parallel Architectures, in: Multigrid Methods VI, Ghent, Belgium, 101–107, 1999.

[9] U. Langer, D. Pusch, Comparison of geometrical and algebraic multigrid preconditioners for data-sparse boundary element matrices, in: International Conference on Large-Scale Scientific Computing, Springer, 130–137, 2005.

[10] A. H. Baker, R. D. Falgout, T. V. Kolev, U. M. Yang, Scaling Hypre's Multigrid Solvers to 100,000 Cores, in: M. W. Berry, K. A. Gallivan, E. Gallopoulos, A. Grama, B. Philippe, Y. Saad, F. Saied (Eds.), High-Performance Scientific Computing, Springer London, ISBN 978-1-4471-2436-8 978-1-4471-2437-5, 261–279, dOI: 10.1007/978-1-4471-2437-5_13, 2012.

[11] E. H. Müller, R. Scheichl, Massively parallel solvers for elliptic partial differential equations in numerical weather and climate prediction: Scalability of Elliptic Solvers in NWP, Quarterly Journal of the Royal Meteorological Society 140 (685) (2014) 2608–2624, ISSN 00359009, doi:10.1002/qj.2327.

[12] P. R. Amestoy, I. S. Duff, J.-Y. L'Excellent, J. Koster, A Fully Asynchronous Multifrontal Solver Using Distributed Dynamic Scheduling, SIAM Journal on Matrix Analysis and Applications 23 (1) (2001) 15–41, ISSN 0895-4798, doi:10.1137/S0895479899358194.

[13] P. R. Amestoy, A. Guermouche, J.-Y. L'Excellent, S. Pralet, Hybrid scheduling for the parallel solution of linear systems, Parallel Computing 32 (2) (2006) 136–156, ISSN 0167-8191, doi:10.1016/j.parco.2005.07.004.
[14] R. D. Falgout, J. E. Jones, U. M. Yang, The Design and Implementation of hypre, a Library of Parallel High Performance Preconditioners, in: Numerical Solution of Partial Differential Equations on Parallel Computers, Springer, Berlin, Heidelberg, 267–294, dOI: 10.1007/3-540-31619-1_8, 2006.
[15] J. A. Mousel, A massively parallel adaptive sharp interface solver with application to mechanical heart valve simulations, Ph.D. thesis, University of Iowa, 2012.
[16] Z. Li, A Fast Iterative Algorithm for Elliptic Interface Problems, SIAM J. NUMER. ANAL 35 (1998) 230–254, doi:10.1137/S0036142995291329.
[17] F. Gibou, R. P. Fedkiw, L.-T. Cheng, M. Kang, A Second-Order-Accurate Symmetric Discretization of the Poisson Equation on Irregular Domains, Journal of Computational Physics 176 (1) (2002) 205–227, ISSN 0021-9991, doi:10.1006/jcph.2001.6977.
[18] J. L. Guermond, P. Minev, J. Shen, An overview of projection methods for incompressible flows, Computer Methods in Applied Mechanics and Engineering 195 (44–47) (2006) 6011–6045, ISSN 0045-7825, doi:10.1016/j.cma.2005.10.010.
[19] L. J. P. Timmermans, P. D. Minev, F. N. Van De Vosse, An Approximate projection scheme for incompressible flow using spectral elements, International Journal for Numerical Methods in Fluids 22 (1996) 673–688.
[20] H. S. Yoon, J. Yin, C. Choi, S. Balachandar, M. Y. Ha, Bifurcation of laminar flow around an elliptic cylinder at incidence for low Reynolds numbers, Progress in Computational Fluid Dynamics, an International Journal 16 (3) (2016) 163–178.
[21] M. Coutanceau, R. Bouard, Experimental determination of the main features of the viscous flow in the wake of a circular cylinder in uniform translation. Part 2. Unsteady flow, Journal of Fluid Mechanics 79 (2) (1977) 257–272, ISSN 1469-7645, 0022-1120, doi:10.1017/S0022112077000147.
[22] M. N. Linnick, H. F. Fasel, A high-order immersed interface method for simulating unsteady incompressible flows on irregular domains, Journal of Computational Physics 204 (1) (2005) 157–192, ISSN 0021-9991, doi:10.1016/j.jcp.2004.09.017.
[23] D. Calhoun, A Cartesian Grid Method for Solving the Two-Dimensional Streamfunction-Vorticity Equations in Irregular Regions, Journal of Computational Physics 176 (2) (2002) 231–275, ISSN 00219991, doi:10.1006/jcph.2001.6970.
[24] D. Le, B. Khoo, J. Peraire, An immersed interface method for viscous incompressible flows involving rigid and flexible boundaries, Journal of Computational Physics 220 (1) (2006) 109–138, ISSN 00219991, doi:10.1016/j.jcp.2006.05.004.
[25] K. Taira, T. Colonius, The immersed boundary method: A projection approach, Journal of Computational Physics 225 (2) (2007) 2118–2137, ISSN 00219991, doi:10.1016/j.jcp.2007.03.005.
[26] P. A. Berthelsen, O. M. Faltinsen, A local directional ghost cell approach for incompressible viscous flow problems with irregular boundaries, Journal of Computational Physics 227 (9) (2008) 4354–4397, ISSN 00219991, doi:10.1016/j.jcp.2007.12.022.
[27] D. J. Tritton, Experiments on the flow past a circular cylinder at low Reynolds numbers, Journal of Fluid Mechanics 6 (04) (1959) 547, ISSN 0022-1120, 1469-7645, doi:10.1017/S0022112059000829.
[28] S. J. D. D'alessio, S. C. R. Dennis, A vorticity model for viscous flow past a cylinder, Computers & fluids 23 (2) (1994) 279–293.
[29] S. C. R. Dennis, P. J. S. Young, Steady flow past an elliptic cylinder inclined to the stream, Journal of engineering mathematics 47 (2) (2003) 101–120.
[30] A. S. Grove, F. H. Shair, E. E. Petersen, An experimental investigation of the steady separated flow past a circular cylinder, Journal of Fluid Mechanics 19 (1) (1964) 60, ISSN 0022-1120, 1469-7645, doi:10.1017/S0022112064000544.
[31] S. C. R. Dennis, G.-Z. Chang, Numerical solutions for steady flow past a circular cylinder at Reynolds numbers up to 100, Journal of Fluid Mechanics 42 (03) (1970) 471, ISSN 0022-1120, 1469-7645, doi:10.1017/S0022112070001428.
[32] Y.-H. Tseng, J. H. Ferziger, A ghost-cell immersed boundary method for flow in complex geometry, Journal of Computational Physics 192 (2) (2003) 593–623, ISSN 0021-9991, doi:10.1016/j.jcp.2003.07.024.