A second-order cut-cell method for the numerical simulation of 2D flows past obstacles

François Bouchon, Thierry Dubois, Nicolas James

Laboratoire de Mathématiques, Université Blaise Pascal and CNRS (UMR 6620), Campus Universitaire des Cézeaux, 63 177 Aubière, France

Abstract

We present a new second-order method, based on the MAC scheme on cartesian grids, for the numerical simulation of two-dimensional incompressible flows past obstacles. In this approach, the solid boundary is embedded in the cartesian computational mesh. Discretizations of the viscous and convective terms are formulated in the context of finite volume methods ensuring local conservation properties of the scheme. Classical second-order centered schemes are applied in mesh cells which are *sufficiently* far from the obstacle. In the mesh cells cut by the obstacle, fisrt-order approximations are proposed. The resulting linear system is nonsymetric but the stencil remains local as in the classical MAC scheme on cartesian grids. The linear systems are solved by a direct method based on the capacitance matrix method. The time integration is achieved with a second-order projection scheme. While in cut-cells the scheme is locally first-order, a global second-order accuracy is recovered. This property is assessed by computing analytical solutions for a Taylor-Couette problem. The efficiency and robustness of the method is supported by numerical simulations of 2D flows past a circular cylinder at Reynolds number up to 9 500. Good agreement with experimental and published numerical results are obtained.

Keywords: Immersed boundary method, Cartesian grid method, Cut-cell method, Incompressible viscous flows, Complex geometry

1. Introduction

Numerical simulation of fluid mechanics problems is one of the most challenging scientific computing's field of research of the last decades. With the increasing power of computers in terms of memory available, speed and number of processors, more and more complex problems arising in industrial applications become accessible to numerical simulations. In complex geometries, the discretization of the Navier Stokes Equation (NSE) by finite element or finite volume methods on body-fitted grids allows to simulate flows at low to moderate Reynolds numbers. However, generating an efficient conformal mesh is a challenging problem when the geometry gets complex and this pre-processing step is very CPU-time consuming. Indeed, generating a body-fitted grid can be even more expansive than computing the solution itself. Eventhough matrices issued from finite element or finite volume discretizations are dense and sparse, the number of connected nodes is much larger than in finite difference methods on cartesian grids. Specific and more complex solvers have to be used resulting in a significant overhead in the computational effort required to obtain discrete solutions. An alternative to the body-fitted methods is proposed with the Immersed Boundary (IB) methods. The aim of IB methods is to handle complex geometric configurations without the use of body-fitted meshes. Simulations are performed on cartesian grids so that the efficiency and robustness of cartesian grid solvers are achieved. Therefore, significant advances in the application of numerical simulations to more and more complex industrial problems could be expected.

Since the late 90s, IB methods have been investigated in many published works and among the most investigated problems is the numerical simulations of two or three-dimensional flows past obstacles. Several technics have been developed to take into account the presence of fixed or mobile obstacles in the flow. IB methods can be classified in three categories : *momentum forcing methods*, *Cartesian grid methods* and *penalty methods*. IB methods have been first developed by Peskin (see [1], and [2] for a review) for fluid structure interaction problems with applications in Biology (blood flow). These methods use both Eulerian and Lagrangian variables, which are related via interaction equations. Discrete versions of dirac functions are used to describe the fluid structure interaction forcing, and to link the Eulerian and Lagrangian variables. In a similar approach, *momentum forcing method*, introduced by Mohd-Yusof [3], have been developped in the context of B-Splines or finite difference methods and applied to the numerical simulations of flows in complex geometries. Here, the challenge is to be able to compute the force applied by the obstacle on the fluid which would be such that the velocity field satisfies the boundary conditions on the immersed boundary (see [4], [5] and [6] for a review). *Momentum forcing methods* are easy to implement and efficient: spectacular numerical simulations of flows in complex geometries have been presented in [7] (flows in an full engine) and in [8] (heart valve dynamics). The coupling of *momentum forcing methods* with projection schemes introduces difficulties in imposing at the same time level the contuinity equation and the boundary conditions on the immersed interface (see [9] for example). The incompressibility of the flow may even be violated in the vicinity of the immersed boundary [10]. This may result in serious diffulties when computing boundary layers at high Reynolds number.

The *penalty methods* [11] have a mathematical background. The presence of a solid obstacle in the computational domain is modeled by adding a penalty term, depending on a *small* parameter, in the incompressible Navier-Stokes equations. The difference between the penalty solution and the solution of the Navier-Stokes equations can be bounded in Sobolev norms (energy norm for instance) in terms of this parameter. The *penalty method* does not depend on the choice of the discretization schemes used to approximate the equations. However, in this approach, the immersed interface is not directly and accurately represented in the flow simulation. Depending on the value of the parameter, a diffusion of the immersed boundary in the computational cells may prevent to accurately predict turbulent flows induced by very thin boundary layers.

The third class of IB methods, called *Cartesian grid methods* or *cut-cell methods*, focus on the discretization of the equations in the mesh cells cut by the immersed boundary (see [12, 13, 14, 15] for instance). Techniques relying on high-order interpolations have been developed in order to impose directly the boundary conditions to the velocity field on the immersed boundary. In a different approach, the method of merging cells (see [12], [14]) consists in gathering cells in the vicinity of the obstacle: a cut-cell is merged with one of their neighbours to form a new polygonal cell. The new merged cell has then more than four neighbours (in dimension 2). Thus, this method breaks the five-point structure of the underlying matrices. In a recent paper, Cheny and Botella (see [16]) proposed a new cut-cell method, called the LS-STAG method, based on the well-known second-order projection MAC scheme (see [17]). They aim to keep the five-point structure of the classical MAC scheme on cartesian grids and to preserve the global conservation properties of the continuous equations at the discrete level. In order to accurately represent the immersed boundary on the computational cartesian grid, they use the signed algebraic distance to the obstacle boundary, as in level-set methods [18]. The staggered arrangement of the unknowns for both the velocity field and the pressure is adapted to the geometry of the cut-cells. The LS-STAG method is highly efficient from a computational point of view as it is based on the MAC solver on cartesian grids which has been extensively and successfully used in numerical simulations of turbulent flows, both in the context of direct and Large-Eddy simulations. However, due to a low order approximation of the viscous and convective fluxes in cut-cells, the LS-STAG method is globally first-order accurate.

The numerical scheme proposed in this paper is a new *cut-cell method*. As in [16], the immersed boundary is geometrically represented by using the signed algebraic distance to the obstacle boundary. In fluid-cells, that is mesh cells which are far enough from the immersed boundary, classical centered, second-order finite volume schemes are used. In our approach, the location of the velocity component is, as in [16], adapted to the geometry of cut-cells. However, the discrete pressure is placed at the center of the cartesian cells for both fluid-cells and cut-cells. In the vicinity of the obstacle, second-order interpolations using boundary conditions on the solid boundaries are introduced to evaluate the convective fluxes. This results in a local first-order approximation of the nonlinear terms in cut-cells. A pointwise approximation of the viscous terms is used in cut-cells. When boundary conditions on the immersed boundary can be used, a five-point stencil scheme for the viscous term is employed. Otherwise, a six-point first-order approximation is introduced. The resulting linear system is close to the five-point structure symmetric system obtained on cartesian mesh with the MAC scheme. A direct solver, based on a capacitance matrix method, is proposed. The efficiency of the solver is similar to the cartesian grid solver obtained with the MAC scheme. The incompressibility of the discrete velocity field is enforced up to the computer accuracy. While first-order truncation errors are locally introduced in the scheme in the cut-cells, a second-order global accuracy is recovered. Note that a similar superconvergence result has been proved by Yamamoto in [19] in the context of elliptic equations.

The paper is organized as follows: in section 2, we present the problem together with the notation, and we describe the mesh used for the space discretization. In section 3, we give details for the space-discretization in the mesh cells cut by the obstacle and propose a fast solver adapted from the capacitance matrix method (see [20], [21]) to solve the linear systems. Section 4 is then devoted to numerical results for different configurations: one of them (the Taylor-Couette flow) allows to check the second-order accuracy of the method, some more challenging tests at high Reynolds numbers show the robustness of the scheme and comparisons with both experimental and numerical results found in literature are given.

2. The setting of the problem

2.1. Preliminaries and notations

We consider a two-dimensional flow past a solid obstacle $\Omega^S \subset \mathbb{R}^2$ which is governed by the incompressible Navier-Stokes equations

$$\frac{\partial \boldsymbol{u}}{\partial t} - \nu \Delta \boldsymbol{u} + \boldsymbol{\nabla}(\boldsymbol{u} \otimes \boldsymbol{u}) + \boldsymbol{\nabla} p = 0, \tag{1}$$

$$\boldsymbol{\nabla} \cdot \boldsymbol{u} = 0, \tag{2}$$

$$\boldsymbol{u}(\boldsymbol{x}, t = 0) = \boldsymbol{u}_0, \tag{3}$$

where $\boldsymbol{u}(\boldsymbol{x}, t) = (u, v)$ is the velocity field at location $\boldsymbol{x} = (x, y) \in \mathbb{R}^2$ at time $t > 0$, $\boldsymbol{u}_0$ is the initial condition and $\nu > 0$ is the kinematic viscosity. We assume that the flow fills a rectangular domain $\Omega = (0, L) \times (0, H)$ in which Ω^S is embedded : $\Omega^S \subset \Omega$ (see Figure 1). We denote by Ω^F the fluid domain in which the Navier-Stokes equations (1)-(3) are prescribed so that we have

$$\Omega = \Omega^F \cup \Omega^S \cup \Gamma^S$$

where $\Gamma^S = \partial \Omega^S$ is the solid boundary. In order to determine the location of each point in the computational domain with respect to the solid boundary Γ^S, we use the signed algebraic distance to Γ^S, which is given by

$$\begin{array}{rccc} d: & \Omega & \longrightarrow & \mathbb{R} \\ & (x, y) & \longmapsto & d(x, y) \end{array} \tag{4}$$

and which satisfies : $\Gamma^S = \{(x, y) \in \Omega; d(x, y) = 0\}$. Furthermore, we assume that

$$\begin{cases} d(x, y) > 0 & \text{if } (x, y) \in \Omega^S, \\ d(x, y) < 0 & \text{if } (x, y) \in \Omega^F. \end{cases} \tag{5}$$

Equations (1)-(3) are supplemented with boundary conditions on $\Gamma = \partial \Omega$ and on Γ^S. On the domain boundary Γ, they depend on the problem under consideration and will be detailed in Section 4. On the immersed boundary Γ^S, Dirichlet boundary conditions are imposed, namely

$$\boldsymbol{u}(\boldsymbol{x}) = \boldsymbol{g}(\boldsymbol{x}) \quad \text{for any } \boldsymbol{x} \in \Gamma^S, \tag{6}$$

where $\boldsymbol{g}$ is prescribed.

2.2. Time discretization : second-order projection method

The temporal discretization of (1)–(3) is achieved by using a second-order projection scheme. In a first step, momentum equations are advanced in time with a semi-implicit scheme decoupling the velocity and pressure unknowns. Then, the intermediate velocity is projected in order to obtain a free-divergence velocity field. Projection methods are efficient and are widely used for the numerical simulations of incompressible flows. In this paper, we use a second-order backward difference (BDF2) projection scheme, which is recalled hereafter.

Let $\delta t > 0$ stand for the time step and $t^k = k\,\delta t$ discrete time values. Let us consider that $(\boldsymbol{u}^j,\, P^j)$ are known for $j \leq k$. The computation of $(\boldsymbol{u}^{k+1},\, P^{k+1})$ consists in:

– Computing a predictor $\widetilde{\boldsymbol{u}}^{k+1}$ by solving:

$$\begin{aligned}\frac{3\widetilde{\boldsymbol{u}}^{k+1} - 4\boldsymbol{u}^k + \boldsymbol{u}^{k-1}}{2\delta t} - \frac{1}{Re}\,\Delta\widetilde{\boldsymbol{u}}^{k+1} + \boldsymbol{\nabla} P^k = \\ -\,2\,\boldsymbol{\nabla}\cdot(\boldsymbol{u}^k \otimes \boldsymbol{u}^k) + \boldsymbol{\nabla}\cdot(\boldsymbol{u}^{k-1} \otimes \boldsymbol{u}^{k-1})\end{aligned} \tag{7}$$

which is supplemented with Boundary conditions applied to $\widetilde{\boldsymbol{u}}^{k+1}$. As it was mentioned in Section 2.1, the choice of boundary conditions depend on the considered problem. They will be detailed in Section 4.

– Projecting to obtain a divergence free velocity $\boldsymbol{u}^{k+1}$:

$$\begin{aligned}&\frac{\boldsymbol{u}^{k+1} - \widetilde{\boldsymbol{u}}^{k+1}}{\delta t} + \frac{2}{3}\,\boldsymbol{\nabla}(P^{k+1} - P^k) = 0, \\ &\boldsymbol{\nabla}\cdot\boldsymbol{u}^{k+1} = 0, \quad (\boldsymbol{u}^{k+1} - \widetilde{\boldsymbol{u}}^{k+1})\cdot\boldsymbol{n} = 0 \quad \text{on } \Gamma.\end{aligned} \tag{8}$$

In the following and unless it is necessary, the superscript k denoting discrete times will be omitted.

2.3. The cartesian mesh

The rectangular computational domain $\Omega = (0, L) \times (0, H)$ is discretized by a cartesian mesh. Let N and M two integers, the sequences of points in each direction satisfy

$$\begin{aligned}0 = x_0 \leq \ldots \leq x_{i-1} \leq x_i \leq \ldots \leq x_N = L, \\ 0 = y_0 \leq \ldots \leq y_{j-1} \leq y_j \leq \ldots \leq y_M = H.\end{aligned}$$

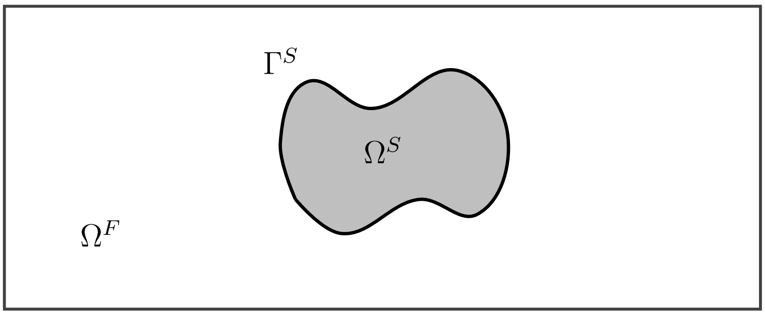

Figure 1: The solid body Ω^S with boundary Γ^S and the surrounding computational domain Ω^F in which the flow is to be simulated.

Mesh sizes are defined by : $\ell_i = x_i - x_{i-1}$ and $h_j = y_j - y_{j-1}$. In order to use computationally efficient algebraic solvers, one of the above sudivisions is assumed to be uniform. In the following, numerical simulations are performed in computational domains with $L \geq H$. Therefore, a uniform mesh is used in the vertical direction, namely

$$y_j = jh \quad \text{for} \quad j = 0, \ldots, M \text{ with } h = \frac{H}{M}.$$

We denote by $\mathcal{G}_{NM}$ the computational grid, that is

$$\mathcal{G}_{NM} = \Big\{(x_i, y_j),\, 0 \leq i \leq N,\, 0 \leq j \leq M\Big\}.$$

In order to simplify the notations we denote by $d_{ij} = d(x_i, y_j)$ the algebraic distance of the grid point $(x_i, y_j) \in \mathcal{G}_{NM}$ to the solid boundary Γ^S.

We define by $K_{ij} = (x_{i-1}, x_i) \times (y_{j-1}, y_j)$ the mesh cells in Ω such that

$$\overline{\Omega} \;=\; \cup_{i=1}^{N} \cup_{j=1}^{M} \; \overline{K}_{ij}.$$

The horizontal edge $\sigma^x_{i,j}$ of the mesh cell K_{ij} is defined by

$$\sigma^x_{i,j} \;=\; (x_{i-1}, x_i) \times \{y_j\}\,.$$

The vertical edge $\sigma^y_{i,j}$ is defined similarly.

The computational cells K_{ij}, that is K_{ij} such that $K_{ij} \cap \Omega^F \neq \emptyset$, can be classified in fluid-cells, namely cells which are totally filled by the fluid ($K_{ij} \cap \Omega^S = \emptyset$), and cut-cells, namely cells such that $K_{ij} \cap \Omega^S \neq \emptyset$.

2.4. The discrete representation of the immersed boundary

Let a cut-cell K_{ij} and assume that its horizontal edge $\sigma^x_{i,j}$ is cut by the obstacle, namely $\sigma^x_{i,j} \cap \Omega^S \neq \emptyset$ then we compute, as in [16], the ratio r^x_{ij} by

$$r^x_{ij} \;=\; \begin{cases} 0 & \text{if } d_{ij} \geq 0 \text{ and } d_{i-1,j} \geq 0, \\ \dfrac{d_{i-1,j}}{d_{i-1,j} - d_{ij}} & \text{if } d_{ij} \geq 0 \text{ and } d_{i-1,j} < 0, \\ \dfrac{d_{i,j}}{d_{ij} - d_{i-1,j}} & \text{if } d_{ij} < 0 \text{ and } d_{i-1,j} \geq 0, \\ 1 & \text{elsewhere.} \end{cases} \tag{9}$$

With the ratio r^x_{ij} we approximate the intersection of $\sigma^x_{i,j}$ and Ω^S by the point with coordinates :

$$\begin{cases} (x_{i-1} + r^x_{ij}\,\ell_i, y_j) & \text{if} \quad d_{i-1,j} < 0, \\ (x_i - r^x_{ij}\,\ell_i, y_j) & \text{if} \quad d_{i,j} < 0. \end{cases}$$

Similarly, we associate to the vertical edge $\sigma^y_{i,j}$ cut by the obstacle a ratio r^y_{ij} and we approximate the intersection of $\sigma^y_{i,j}$ and Ω^S by the point with coordinates $(x_i, y_{j-1} + r^y_{ij}\,h)$ or $(x_i, y_j - r^y_{ij}\,h)$ depending upon the sign of $d_{i,j-1}$ and $d_{i,j}$. By doing so, the obstacle boundary Γ^S is approximated by a piecewise-linear curve Γ^S_h (see Figure 2). This provides a discrete separation of the computational domain Ω in :

$$\Omega \;=\; \Omega^F_h \cup \Omega^S_h \cup \Gamma^S_h.$$

Let a computational cell K_{ij}, that is $K_{ij} \cap \Omega^F_h \neq \emptyset$, we denote by K^F_{ij} the part of the cell which is filled by the fluid, namely $K^F_{ij} = K_{ij} \cap \Omega^F_h$ and by $\sigma^{x,F}_{i,j}$ (resp. $\sigma^{y,F}_{i,j}$) the part of the edge $\sigma^x_{i,j}$ (resp. $\sigma^y_{i,j}$) which is in Ω^F_h, so that we have $\sigma^{x,F}_{i,j} = \sigma^x_{i,j}$ if K_{ij} is a fluid-cell and

$$\sigma^{x,F}_{i,j} \;=\; \begin{cases} (x_{i-1}, x_{i-1} + r^x_{ij}\,\ell_i) \times \{y_j\} & \text{if} \quad d_{i-1,j} < 0, \\ (x_i - r^x_{ij}\,\ell_i, x_i) \times \{y_j\} & \text{if} \quad d_{i,j} < 0, \end{cases}$$

if K_{ij} is a cut-cell.

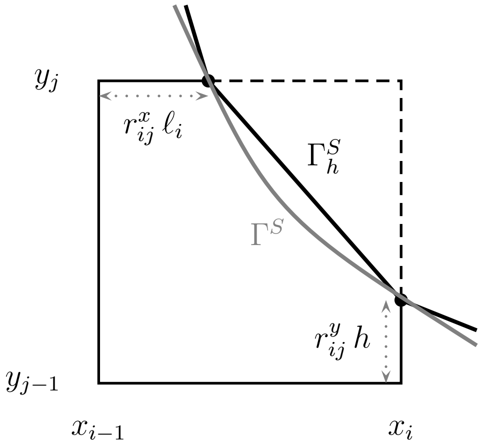

Figure 2: The discrete representation Γ^S_h of the immersed boundary Γ^S.

2.5. A staggered arrangement of the unknowns

As in the classical MAC scheme for cartesian grids (see [17]), the velocity unknowns are located at the midpoints of the cell edges. Let a computational cell K^F_{ij}, the associated velocity unknowns u_{ij} and v_{ij} are respectively located at the midpoints $\boldsymbol{\kappa}^{y,F}_{ij}$ of $\sigma^{y,F}_{ij}$ and $\boldsymbol{\kappa}^{x,F}_{ij}$ of $\sigma^{x,F}_{ij}$. Note that

$$\boldsymbol{\kappa}^{y,F}_{ij} = \begin{cases} (x_i, y_{j-1/2}) & \text{if } \ \sigma^{y,F}_{ij} \subset \Omega^F_h, \\ (x_i, y_{j-1} + \frac{r^y_{ij} h}{2}) & \text{if } \sigma^y_{ij} \cap \Omega^S_h \neq \emptyset \text{ and } d_{i,j-1} < 0, \\ (x_i, y_j - \frac{r^y_{ij} h}{2}) & \text{if } \sigma^y_{ij} \cap \Omega^S_h \neq \emptyset \text{ and } d_{i,j} < 0, \end{cases}$$

where $y_{j-1/2} = \frac{1}{2}(y_j + y_{j-1})$. Even if $K^F_{i,j}$ is a cut-cell, the discrete pressure value P_{ij} is always located at the center of the corresponding mesh cell $K_{i,j}$, that is at point of coordinates $(x_{i-1/2}, y_{j-1/2})$. This staggered arrangement of the unknowns is represented on Figure 3.

3. Description of the discrete operators

In this section, we describe the numerical discretization of the spatial partial derivatives present in the Navier-Stokes equations. Specific numerical treatments are required in cut-cells while, away from the obstacle, classical second-order centered finite volume schemes are used.

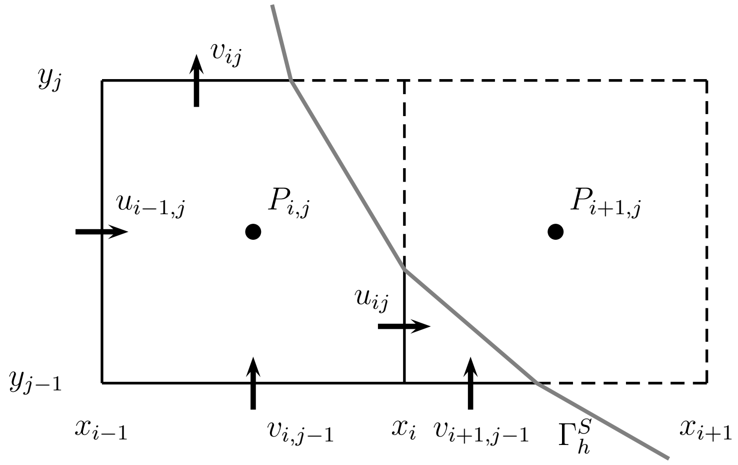

Figure 3: Location of the unknowns in the cut-cells K_{ij} (left) and $K_{i+1,j}$ (right).

3.1. The Laplace operator

In the classical MAC scheme on cartesian grid, the horizontal (resp. vertical) momentum equation is discretized by integration over the volume cell $K_{ij}^u = (x_{i-1/2}, x_{i+1/2}) \times (y_{j-1}, y_j)$ (resp. $K_{ij}^v = (x_{i-1}, x_i) \times (y_{j-1/2}, y_{j+1/2})$). In order to take into account the presence of the obstacle, integration is performed over the computational cells $K_{ij}^{u,F} = K_{ij}^u \cap \Omega_h^F$ and $K_{ij}^{v,F} = K_{ij}^v \cap \Omega_h^F$. In the following, we restrict the presentation to the equation of the horizontal velocity component. The vertical case is treated similarly.

Let a computational volume control $K_{ij}^{u,F}$, which is surrounding the velocity unknown u_{ij}, the approximation of $\int_{K_{i,j}^{u,F}} \Delta u \, d\mathbf{x}$ depends on the location of the mesh cell $K_{i,j}$ with respect to the obstacle.

If $K_{i,j}$ is sufficiently far from the obstacle so that both $K_{i,j}$ and $K_{i+1,j}$ are fluid cells, that is $(K_{i,j} \cup K_{i+1,j}) \subset \Omega_h^F$, and the faces $\sigma_{i,j-1}^y$ and $\sigma_{i,j+1}^y$ are not cut by the obstacle, then the following classical second-order approximation is used

$$\begin{aligned} \frac{1}{|K_{i,j}^{u,F}|} \int_{K_{i,j}^u} \Delta u \, d\mathbf{x} = & \ \frac{1}{|K_{i,j}^{u,F}|} \int_{\partial K_{i,j}^u} \frac{\partial u}{\partial n} \, ds \\ \approx & \ \frac{2}{(\ell_{i+1} + \ell_i)} \left(\frac{u_{i+1,j} - u_{i,j}}{\ell_{i+1}} - \frac{u_{i,j} - u_{i-1,j}}{\ell_i} \right) \\ & + \left(\frac{u_{i,j+1} - 2\, u_{i,j} + u_{i,j-1}}{h^2} \right). \end{aligned} \tag{10}$$

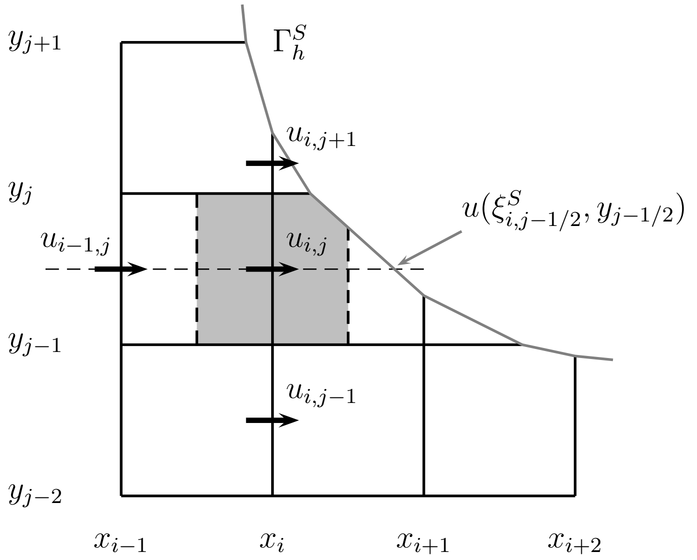

Figure 4: Five-point stencil, using the boundary condition on the obstacle Γ_h^S, for the first-order finite difference approximation of $\Delta u(\boldsymbol{\kappa}_{i,j}^{y,F})$. We recall that, in this case, $\boldsymbol{\kappa}_{i,j}^{y,F} = (x_i, y_{j-1/2})$.

In the other cases, that is either $K_{i,j}$, $K_{i+1,j}$, $\sigma^y_{i,j-1}$ or $\sigma^y_{i,j+1}$ are cut by the obstacle, we use a pointwise approximation, namely

$$\frac{1}{|K_{i,j}^{u,F}|} \int_{K_{i,j}^{u,F}} \Delta u \, d\mathbf{x} \approx \Delta u(\boldsymbol{\kappa}_{i,j}^{y,F}).$$

Obviously such approximation is first-order. When it is feasible, depending upon the location of $\boldsymbol{\kappa}_{i,j}^{y,F}$ with respect to the obstacle, the Dirichlet condition on Γ_h^S is used to write a five-point stencil, first-order finite difference approximation of $\Delta u(\boldsymbol{\kappa}_{i,j}^{y,F})$. Otherwise, a sixth point chosen in the neighbourhood of $\boldsymbol{\kappa}_{i,j}^{y,F}$, namely $\{\boldsymbol{\kappa}_{i\pm1,j\pm1}^{y,F}\}$, is added in the approximation scheme. In order to illustrate this approach, we detail hereafter two particular cases. All other cases are treated similarly. Let us first assume that $K_{i,j}$ is a fluid-cell and $K_{i+1,j}$ a cut-cell, such that $\sigma^y_{i+1,j}$ is cut by the obstacle and $r^y_{i+1,j} \leq 0.5$, as it is represented on Figure 4. Then, the discrete obstacle boundary Γ_h^S crosses the line $y = y_{j-1/2}$. Denoting by $\xi^S_{i,j-1/2}$ the horizontal coordinate of the intersection point, we note that $u(\xi^S_{i,j-1/2}, y_{j-1/2})$ is known as Dirichlet boundary conditions are imposed on the obstacle (see (6)). Therefore, $\{u_{i,j-1}, u_{i,j}, u_{i,j+1}\}$

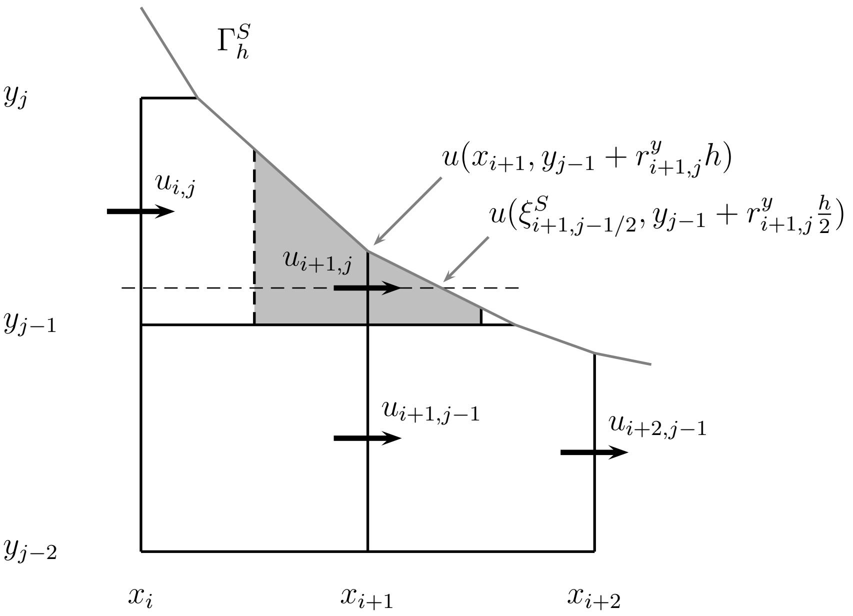

Figure 5: Six-point stencil, using the boundary condition on the obstacle Γ_h^S, for the first-order finite difference approximation of $\Delta u(\boldsymbol{\kappa}^{y,F}_{i+1,j})$. We recall that, in this case, $\boldsymbol{\kappa}^{y,F}_{i+1,j} = (x_{i+1}, y_{j-1} + r^y_{i+1,j}\frac{h}{2})$.

as well as $\{u_{i-1,j}, u_{i,j}, u(\xi^S_{i,j-1/2}, y_{j-1/2})\}$ are aligned so that, in the particular case represented on Figure 4, a first-order finite difference approximation of $\Delta u(\boldsymbol{\kappa}^{y,F}_{i,j})$ is easily obtained by writing

$$\begin{aligned}\Delta u(\boldsymbol{\kappa}^{y,F}_{i,j}) \;\approx\; & \frac{2}{(r^y_{i,j+1}+3)\frac{h}{2}}\left(\frac{u_{i,j+1}-u_{ij}}{(r^y_{i,j+1}+1)\frac{h}{2}} - \frac{u_{i,j}-u_{i,j-1}}{h}\right) \\ & + \frac{2}{(\xi^S_{i,j-1/2}-x_{i-1})}\left(\frac{u(\xi^S_{i,j-1/2}, y_{j-1/2}) - u_{i,j}}{(\xi^S_{i,j-1/2}-x_i)} - \frac{u_{i,j}-u_{i-1,j}}{\ell_i}\right).\end{aligned} \tag{11}$$

Remaining with the particular case depicted on Figure 4, let us now describe the approximation of $\Delta u(\boldsymbol{\kappa}^{y,F}_{i+1,j})$, which is associated with the velocity unknown $u_{i+1,j}$. In order to better represent the neighbourhood of $\boldsymbol{\kappa}^{y,F}_{i+1,j}$, Figure 4 is enlarged and centered on $\boldsymbol{\kappa}^{y,F}_{i+1,j}$ in order to obtain Figure 5. Boundary conditions at points $(x_{i+1}, y_{j-1} + r^y_{i+1,j}h)$ and $(\xi^S_{i+1,j-1/2}, y_{j-1} + r^y_{i+1,j}\frac{h}{2})$, which is the intersection point with the horizontal line $y = y_{j-1} + r^y_{i+1,j}\frac{h}{2}$ with the obstacle boundary Γ_h^S, can be used. Therefore, the five-point stencil $\{\boldsymbol{\kappa}^{y,F}_{i,j}, \boldsymbol{\kappa}^{y,F}_{i+1,j}, (\xi^S_{i+1,j-1/2}, y_{j-1} + r^y_{i+1,j}\frac{h}{2}), (x_{i+1}, y_{j-1} + r^y_{i+1,j}h), \kappa^{y,F}_{i+1,j-1}\}$ is available to write an approximation of $\Delta u(\boldsymbol{\kappa}^{y,F}_{i+1,j})$. However, as the points $\boldsymbol{\kappa}^{y,F}_{i,j}$, $\boldsymbol{\kappa}^{y,F}_{i+1,j}$ and $(\xi^S_{i+1,j-1/2}, y_{j-1} + r^y_{i+1,j}\frac{h}{2})$ are not aligned a sixth point has to be added in order to obtain a first-order scheme. This point is chosen among $\{\boldsymbol{\kappa}^{y,F}_{i,j\pm1}, \boldsymbol{\kappa}^{y,F}_{i+2,j-1}\}$. In order to preserve the symmetry of the flow in symmetric configurations (geometry, boundary conditions), a criterion must be imposed to define the sixth point. In our implementation, we choose among $\{\boldsymbol{\kappa}^{y,F}_{i-1,j\pm1}, \boldsymbol{\kappa}^{y,F}_{i+2,j-1}\}$ the closest point to Γ^S_h, which is $\boldsymbol{\kappa}^{y,F}_{i+2,j-1}$ in the situation depicted on Figure 5. Finally, we seek for a first-order finite difference approximation of $\Delta u(\boldsymbol{\kappa}^{y,F}_{i+1,j})$ of the form

$$\begin{aligned} a_0\, u(\boldsymbol{\kappa}^{y,F}_{i,j}) \;+\; a_1\, u(\boldsymbol{\kappa}^{y,F}_{i+1,j}) \;+\; a_2\, u(\xi^S_{i+1,j-1/2}, y_{j-1} + r^y_{i+1,j}\frac{h}{2}) \\ +\; a_3\, u(x_{i+1}, y_{j-1} + r^y_{i+1,j} h) \;+\; a_4\, u(\boldsymbol{\kappa}^{y,F}_{i+1,j-1}) \\ +\; a_5\, u(\boldsymbol{\kappa}^{y,F}_{i+2,j-1}) \;=\; \Delta u(\boldsymbol{\kappa}^{y,F}_{i+1,j}) \;+\; O(h, \ell_{i+1}, \ell_{i+2}). \end{aligned} \tag{12}$$

The coefficients $\{a_i, i = 0, \ldots, 5\}$ are computed by solving a sixth-order linear system derived from (12) in the same spirit as the Immersed Interface Techniques of Leveque and Li (see [22]). Therefore, for mesh cells close to the obstacle, a locally first-order error is done. We observe (see numerical results in Section 4) that this locally first-order truncation error near the obstacle does not affect the global second-order convergence rate, note that the same behaviour was observed in [22]. We mention that a similar superconvergence result has also been proved by Yamamoto in [19] in the context of elliptic equations.

3.2. The nonlinear (convective) terms

Let a computational volume control $K^{u,F}_{ij}$, which is surrounding the velocity unknown u_{ij}, the nonlinear term is discretized by first writing

$$\int_{K^{u,F}_{ij}} \boldsymbol{\nabla}(u\boldsymbol{u})\, d\boldsymbol{x} \;=\; \int_{\partial K^{u,F}_{ij}} u\, \boldsymbol{u}\cdot\boldsymbol{n}\, ds \;=\; \sum_{\sigma\in\partial K^{u,F}_{ij}} \int_\sigma u\, \boldsymbol{u}\cdot\boldsymbol{n}\, ds \tag{13}$$

where $\boldsymbol{n}$ denotes the unit normal vector to the boundary $\partial K^{u,F}_{ij}$, outward to $K^{u,F}_{ij}$. The computational cell $\partial K^{u,F}_{i,j}$ is a polygon (see Figure 6 for an example) which is bounded by, at most, six edges, namely we write

$$\begin{aligned} \partial K^{u,F}_{i,j} \;&=\; (\partial K^{u,F}_{i,j} \cap \Omega^F_h) \cup (\partial K^{u,F}_{i,j} \cap \Omega^S_h), \\ &=\; \sigma^{x,F}_{i+1/2,j-1} \cup \sigma^{x,F}_{i+1/2,j} \cup \sigma^{y,F}_{i-1/2,j} \cup \sigma^{y,F}_{i+1/2,j} \cup \sigma^{u,S}_{i,j} \cup \sigma^{u,S}_{i+1/2,j} \end{aligned} \tag{14}$$

where

$$\begin{aligned}
\sigma^{u,S}_{i,j} &= \partial K^{u,F}_{i,j} \cap \overline{K_{i,j}}^{F} \cap \Gamma^S_h, \\
\sigma^{u,S}_{i+1/2,j} &= \partial K^{u,F}_{i,j} \cap \overline{K^F_{i+1,j}} \cap \Gamma^S_h, \\
\sigma^{x,F}_{i+1/2,j} &= (x_{i-1/2}, x_{i+1/2}) \times \{y_j\} \cap \Omega^F_h, \\
\sigma^{y,F}_{i+1/2,j} &= \{x_{i+1/2}\} \times (y_{j-1}, y_j) \cap \Omega^F_h.
\end{aligned}$$

In (13), the numerical approximation of $\int_\sigma u\,\boldsymbol{u}\cdot\boldsymbol{n}\,ds$ is done by using the midpoint integration rule, which induces a third-order error. For an edge shared with the obstacle boundary, we write

$$\int_{\sigma^{u,S}_{i,j}} u\,\boldsymbol{u}\cdot\boldsymbol{n}\,ds \;\approx\; |\sigma^{u,S}_{i,j}|\,u(\boldsymbol{\kappa}^{u,S}_{i,j})\,\boldsymbol{u}(\boldsymbol{\kappa}^{u,S}_{i,j})\cdot\boldsymbol{n}(\boldsymbol{\kappa}^{u,S}_{i,j}) \tag{15}$$

where $\boldsymbol{\kappa}^{u,S}_{i,j}$ denotes the midpoint of $\sigma^{u,S}_{i,j}$. On the other edges, an interpolation is required in order to obtain an approximation of the velocity at the midpoint edges. A second-order interpolation is used and the interpolation formulae depend on how the cells are cut by the obstacle. Let us detail how these interpolations are performed in the particular case of Figure 6. Noting that $\boldsymbol{\kappa}^{y,F}_{i,j}$, $\boldsymbol{\kappa}^{y,F}_{i+1,j}$ and $\boldsymbol{\kappa}^{y,F}_{i+1/2,j}$ (the $\sigma^{y,F}_{i+1/2,j}$ midpoint) are aligned, we write

$$\int_{\sigma^{y,F}_{i+1/2,j}} u^2\,ds \;\approx\; (r^y_{i,j} + r^y_{i+1,j})\,\frac{h}{2}\,\left(\frac{u_{i,j}+u_{i+1,j}}{2}\right)^2.$$

Concerning the contribution of the edge $\sigma^{y,F}_{i-1/2,j}$, the points $\boldsymbol{\kappa}^{y,F}_{i-1,j}$, $\boldsymbol{\kappa}^{y,F}_{i,j}$ and $\boldsymbol{\kappa}^{y,F}_{i-1/2,j}$ are not aligned. In this case, we use a third point (here the boundary value at point $\boldsymbol{\kappa}^{u,S}_{i,j}$) in order to obtain a second-order interpolation for the horizontal velocity at point $\boldsymbol{\kappa}^{y,F}_{i-1/2,j}$, namely we write

$$-\int_{\sigma^{y,F}_{i-1/2,j}} u^2\,ds \;\approx\; -h\,\Big(a_0\,u_{i,j} + a_1\,u(\boldsymbol{\kappa}^{u,S}_{i,j}) + (1-a_0-a_1)\,u_{i-1,j}\Big)^2$$

where

$$\left\{\begin{aligned}
a_0 &= \tfrac{-r^y_{i,j}}{(r^x_{i,j}(r^y_{i,j}-1)-(r^y_{i,j}+1))}, \\
a_1 &= \tfrac{(r^y_{i,j}-1)}{(r^x_{i,j}(r^y_{i,j}-1)-(r^y_{i,j}+1))}.
\end{aligned}\right.$$

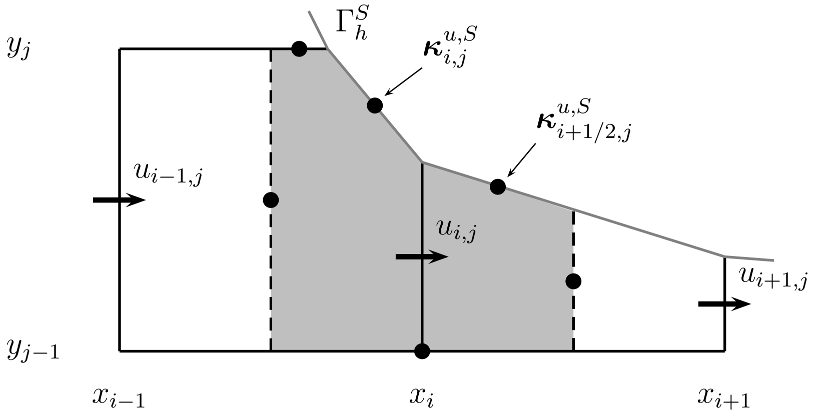

Figure 6: A polygonal volume control $K_{i,j}^{u,F}$ and the location of the points (black dot) where the convective fluxes are approximated.

The fluxes on the horizontal edges are treated similarly. Let us denote by $F_{i,j}^{\sigma}$ the discrete flux corresponding to an edge $\sigma \in \partial K_{i,j}^{u,F}$. We easily obtain the following result which controls the discretization error near the obstcale. Elsewhere, a second-order approximation is achieved.

Proposition 3.1. *Let $K_{i,j}^{u,F}$ be a cut-cell and assume that the velocity field $\boldsymbol{u}$ is sufficiently regular, then we have*

$$\frac{1}{|K_{i,j}^{u,F}|}\left|\int_{K_{i,j}^{u,F}} \boldsymbol{\nabla}(u\boldsymbol{u})\,d\boldsymbol{x} - \sum_{\sigma\in\partial K_{ij}^{u,F}} F_{i,j}^{\sigma}\right| \le \frac{c_{i,j}}{|K_{i,j}^{u,F}|}\max\left(h^3, (\ell_i+\ell_{i+1})^3\right) \quad (16)$$

where $c_{i,j}$ is a constant depending on high-order derivatives of $\boldsymbol{u}$.

Proof. By using (13), we have

$$\begin{aligned}\int_{K_{i,j}^{u,F}} \boldsymbol{\nabla}(u\boldsymbol{u})\,d\boldsymbol{x} - \sum_{\sigma\in\partial K_{ij}^{u,F}} F_{i,j}^{\sigma} &= \sum_{\sigma\in\partial K_{ij}^{u,F}}\left(\int_\sigma u\,\boldsymbol{u}\cdot\boldsymbol{n}\,ds \;-\; F_{i,j}^{\sigma}\right)\\ &= \sum_{\sigma\in\partial K_{ij}^{u,F}}\left(\int_\sigma u\,\boldsymbol{u}\cdot\boldsymbol{n}\,ds \;-\; |\sigma|\,u(\boldsymbol{\kappa}_\sigma)\,\boldsymbol{u}(\boldsymbol{\kappa}_\sigma)\cdot\boldsymbol{n}(\boldsymbol{\kappa}_\sigma)\right)\\ &+ \sum_{\sigma\in\partial K_{ij}^{u,F}}\left(|\sigma|\,u(\boldsymbol{\kappa}_\sigma)\,\boldsymbol{u}(\boldsymbol{\kappa}_\sigma)\cdot\boldsymbol{n}(\boldsymbol{\kappa}_\sigma) - F_{i,j}^{\sigma}\right)\end{aligned}$$

where $\boldsymbol{\kappa}_\sigma$ denotes the midpoint of σ. The first term above corresponds to the error due to the midpoint integration rule while the second one is an interpolation error. It follows easily

$$\Big|\int_{K_{i,j}^{u,F}} \boldsymbol{\nabla}(u\boldsymbol{u})\, d\boldsymbol{x} - \sum_{\sigma\in\partial K_{ij}^{u,F}} F_{i,j}^{\sigma}\Big| \leq \sum_{\sigma\in\partial K_{ij}^{u,F}} |\sigma| \left(c_{0,\sigma}|\sigma|^2 \,+\, c_{1,\sigma} \max\left(h^2, (\ell_i+\ell_{i+1})^2\right)\right)$$

which completes the proof. □

In (16), the right-hand side is first-order as long as the cut-cell $K_{i,j}^{u,F}$ is such that

$$\exists\, c'_{i,j} > 0 \quad \text{such that} \quad |K_{i,j}^{u,F}| \geq c'_{ij}\, h(\ell_i+\ell_{i+1}).$$

The above condition may be locally violated in the numerical simulations. Indeed, a cut-cell can be as small as we want. However, this phenomena should be a marginal one and was never encountered in the numerical simulations performed in this paper.

3.3. The pressure gradient

Let a computational volume control $K_{ij}^{u,F}$. The approximation of the mean pressure gradient over $K_{i,j}^{u,F}$ depends on the location of the mesh cell $K_{i,j}$ with respect to the obstacle. If $K_{i,j}$ is sufficiently far from the obstacle so that both $K_{i,j}$ and $K_{i+1,j}$ are fluid cells, that is $(K_{i,j}\cup K_{i+1,j}) \subset \Omega_h^F$, then the following classical second-order approximation is used

$$\frac{1}{|K_{i,j}^u|}\int_{K_{i,j}^u} \frac{\partial P}{\partial x}\, d\mathbf{x} = \frac{1}{|K_{i,j}^u|}\int_{\partial K_{i,j}^u} P\, n_x\, ds \approx\, 2\,\frac{(P_{i+1,j}-P_{i,j})}{(\ell_i+\ell_{i+1})}. \qquad (17)$$

If either $K_{i,j}$ or $K_{i+1,j}$ are cut-cell, the following pointwise approximation is applied

$$\frac{1}{|K_{i,j}^{u,F}|}\int_{K_{i,j}^{u,F}} \frac{\partial P}{\partial x}\, d\mathbf{x} \,\approx\, \frac{\partial P}{\partial x}(\boldsymbol{\kappa}_{i,j}^{y,F})$$

which is a first-order approximation. Combining the standard finite difference approximation with a second-order interpolation, we define the discrete horizontal component of the pressure gradient at point $\boldsymbol{\kappa}_{i,j}^{y,F}$ by

$$(G_xP)_{i,j} \;=\; \begin{cases} (1+r_{ij}^{y})\frac{P_{i+1,j}-P_{i,j}}{\ell_i+\ell_{i+1}} \;+\; (1-r_{ij}^{y})\frac{P_{i+1,j-1}-P_{i,j-1}}{\ell_i+\ell_{i+1}} \\ \qquad\qquad\qquad\qquad \text{if } d_{i,j-1}<0 \text{ and } d_{i,j}>0, \\ (1+r_{ij}^{y})\frac{P_{i+1,j}-P_{i,j}}{\ell_i+\ell_{i+1}} \;+\; (1-r_{ij}^{y})\frac{P_{i+1,j+1}-P_{i,j+1}}{\ell_i+\ell_{i+1}} \\ \qquad\qquad\qquad\qquad \text{if } d_{i,j-1}>0 \text{ and } d_{i,j}<0, \\ 2\frac{(P_{i+1,j}-P_{i,j})}{(\ell_i+\ell_{i+1})} \text{ elsewhere.} \end{cases} \tag{18}$$

The vertical component of the pressure gradient is discretized similarly at point $\boldsymbol{\kappa}_{i,j}^{x,F}$ and, its discrete approximation is denoted by $(G_yP)_{i,j}$.

3.4. The continuity equation

As in the classical MAC scheme on cartesian grids, the discrete divergence is obtained by integration over a computational cell K_{ij}^F, namely we write

$$\int_{K_{ij}^F} \operatorname{div}(\boldsymbol{u})\, d\boldsymbol{x} \;=\; \sum_{\sigma\in\partial K_{ij}^F} \int_\sigma \boldsymbol{u}\cdot\boldsymbol{n}\, ds, \tag{19}$$

where $\boldsymbol{n}$ denotes the unit normal vector to the boundary ∂K_{ij}^F, outward to K_{ij}^F. The computational cell $K_{i,j}^F$ is a polygon which is bounded by at most five edges, namely

$$\partial K_{i,j}^F \;=\; \sigma_{i,j-1}^{x,F} \cup \sigma_{i,j}^{x,F} \cup \sigma_{i-1,j}^{y,F} \cup \sigma_{i,j}^{y,F} \cup \sigma_{i,j}^{S} \tag{20}$$

where $\sigma_{i,j}^S \;=\; \partial K_{i,j}^F \cap \Gamma_h^S$. With the use of the midpoint integration rule in (19), we derive the discrete version of the continuity equation in the cell $K_{i,j}^F$, namely

$$(D\boldsymbol{u})_{i,j} \;+\; |\sigma_{i,j}^S|\, \boldsymbol{u}(\boldsymbol{\kappa}_{i,j}^S)\cdot\boldsymbol{n}(\boldsymbol{\kappa}_{i,j}^S) \;=\; 0. \tag{21}$$

where $(D\boldsymbol{u})_{i,j} \;=\; \ell_i(r_{i,j}^x v_{i,j} - r_{i,j-1}^x v_{i,j-1}) \;+\; h(r_{i,j}^y u_{i,j} - r_{i-1,j}^y u_{i-1,j})$.

3.5. The velocity correction step

In projection schemes, the intermediate velocity $\tilde{\boldsymbol{u}}$ is corrected with the help of the pressure gradient in order to obtain a free-divergence velocity field. For the horizontal velocity, the correction step is discretized by integration of (8) over each edges $\sigma_{ij}^{y,F}$, namely we write

$$\int_{\sigma_{ij}^{y,F}} \Big(u - \tilde{u} + \frac{2}{3}\delta t \frac{\partial}{\partial x}(\delta P^{k+1}) \Big)\, dy \;=\; 0 \tag{22}$$

where $\delta P^{k+1} = P^{k+1} - P^k$. The discrete version of (22) follows by writing

$$u_{ij} - \tilde{u}_{ij} + \frac{2}{3}\delta t (G_x \delta P^{k+1})_{i,j} \; = \; 0 \tag{23}$$

and similarly for the vertical velocity component

$$v_{ij} - \tilde{v}_{ij} + \frac{2}{3}\delta t (G_y \delta P^{k+1})_{i,j} \; = \; 0. \tag{24}$$

By reporting (23) and (24) in the definition of $(D\boldsymbol{u})_{ij}$ above, and using (21), we derive

$$\begin{aligned} \ell_i r^x_{i,j} \, (G_y \delta P^{k+1})_{i,j} - \ell_i r^x_{i,j-1} \, (G_y \delta P^{k+1})_{i,j-1} \, + \, h r^y_{i,j} \, (G_x \delta P^{k+1})_{i,j} \\ - h r^y_{i-1,j} \, (G_x \delta P^{k+1})_{i-1,j} \; = \; \frac{3}{2\delta t} (D\tilde{\boldsymbol{u}})_{i,j} \, + \, \frac{3}{2\delta t} \, |\sigma^S_{i,j}| \, \boldsymbol{u}(\boldsymbol{\kappa}^S_{i,j}) \cdot \boldsymbol{n}(\boldsymbol{\kappa}^S_{i,j}). \end{aligned} \tag{25}$$

which is a discrete Poisson equation satisfied by the pressure increment δP^{k+1} in a computational cut-cell $K^F_{i,j}$. Note that if $K^F_{i,j}$ is a fluid-cell, that is $K^F_{i,j} \cap \Omega^S_h = \emptyset$, equation (25) reduces to

$$\begin{aligned} \ell_i \frac{\delta P^{k+1}_{i,j+1} - \delta P^{k+1}_{i,j}}{h} \; - \; \ell_i \frac{\delta P^{k+1}_{i,j} - \delta P^{k+1}_{i,j-1}}{h} \; + \; 2h \frac{\delta P^{k+1}_{i+1,j} - \delta P^{k+1}_{i,j}}{\ell_i + \ell_{i+1}} \\ - \, 2h \frac{\delta P^{k+1}_{i,j} - \delta P^{k+1}_{i-1,j}}{\ell_i + \ell_{i-1}} \; = \; \frac{3}{2\delta t} \, \left(\ell_i (\tilde{v}_{i,j} - \tilde{v}_{i,j-1}) \, + \, h (\tilde{u}_{i,j} - \tilde{u}_{i-1,j}) \right). \end{aligned} \tag{26}$$

The linear system formed with (26) in fluid-cells and (25) in cut-cells is non-symmetric. The lost of symmetry is due to the presence of the obstacle. As for the classical MAC scheme, the resolution of this linear system ensures that the incompressiblity condition is enforced up to the computer accuracy.

3.6. A direct solver for the non-symmetric linear systems

In this section we describe the fast solver used to run the simulation. The algorithm is inspired from the capacitance matrix method (see [20] and [21]) and has been adapted for the case of non regular grids.

We detail the algorithm developed for the elliptic system (25) solved to compute the pseudo-pressure δP^{k+1}; similar methods for parabolic problems (see [23]) have been used for the velocity components equations derived from the temporal (see section 2.2) and spatial (see section 3.1) discretization.

For the sake of conciseness, we denote $\varphi = \delta P^{k+1}$ in the following.

The linear system (26) is equivalent with the following:

$$\begin{aligned}\frac{\varphi_{i,j}-\varphi_{i,j-1}}{h^2}+\frac{\varphi_{i,j}-\varphi_{i,j+1}}{h^2}&\\ +2\frac{\varphi_{i,j}-\varphi_{i-1,j}}{\ell_i(\ell_i+\ell_{i-1})}+2\frac{\varphi_{i,j}-\varphi_{i+1,j}}{\ell_i(\ell_{i+1}+\ell_i)}&=(rhs)_{i,j}\end{aligned} \tag{27}$$

for indices i,j such that $x_{i-1/2,j-1/2}\in\Omega_h^F$ are far from the obstacle.

***Remark* 1.** *For the points close to the obstacle, equations differ from* (27) *due to the presence of ratios $r_{ij}^x\neq 1$ or $r_{ij}^y\neq 1$ in* (18)*.*

Let us mention that the formulae also differ from (27) for points which are close to the boundary Γ, since one of the neighbour of i,j is out of the computational domain. For such points, one (at least) of the four terms $\dfrac{\varphi_{i,j}-\varphi_{i,j-1}}{h^2}$, $\dfrac{\varphi_{i,j}-\varphi_{i,j+1}}{h^2}$, $2\dfrac{\varphi_{i,j}-\varphi_{i-1,j}}{\ell_i(\ell_i+\ell_{i-1})}$, $2\dfrac{\varphi_{i,j}-\varphi_{i+1,j}}{\ell_i(\ell_{i+1}+\ell_i)}$ must be removed in equation (27).

***Remark* 2.** *The resulting linear system is singular, we then replace the first equation by $\varphi_{1,1}=0$ to insure existence and uniquess of the solution.*

We now complete this linear system with the following set of equations:

$$\frac{\varphi_{i,j}-\varphi_{i,j-1}}{h^2}+\frac{\varphi_{i,j}-\varphi_{i,j+1}}{h^2}+2\frac{\varphi_{i,j}-\varphi_{i-1,j}}{\ell_i(\ell_i+\ell_{i-1})}+2\frac{\varphi_{i,j}-\varphi_{i+1,j}}{\ell_i(\ell_{i+1}+\ell_i)}=0 \tag{28}$$

for subscripts i,j such that $K_{i,j}\subset\Omega^S$.

Equations described in remark 2, in equations (27) and (25) *(for cut-cells)*, and in equations (28) can be described by a linear system of K equations with K unknowns, with $K=N\times M$, where the unknowns inside/outside the obstacle are decoupled . Thus, the solution in the fluid (i.e. outside the obstacle) is the original solution of (25).

Let us consider the matrix $G\in\mathcal{M}_{K,K}(\mathbb{R})$ of the discretization of the same elliptic operator on the whole domain $(0,L)\times(0,H)$ with homogeneous Neumann boundary conditions on $\partial\left((0,L)\times(0,H)\right)$, with the first line of the system replaced by $\varphi_{1,1}=0$ as explained by Remark 2. This matrix G corresponds to the linear system which should be solved in the case where there were no obstacle in the domain. We denote by x the vector of $\mathbb{R}^K$ containing the values of $\varphi_{i,j}$ for $i=1,\cdots,N$ and $j=1,\cdots,M$. If the matrix formulation of this linear system described in remark 2, in equation (27), (25) and (28) is

$$\Xi x = z, \tag{29}$$

then Ξ and G are identical, except on the rows corresponding to the boundary condition on the obstacle. We denote n_1 the number of rows where $\Xi - G$ has nonvanishing coefficients.

Remark 3. *Since the boundary of the obstacle is a one-dimensional smooth curve in $\mathbb{R}^2$, we have $n_1 = \mathcal{O}(\Pi)$ with $\Pi = \max(M, N)$. Moreover, the number of nonvanishing entries on each of the rows of $\Xi - G$ is bounded.*

We now use these observations to propose a fast solver for (29). First, note that (29) is equivalent with the following system:

$$Gx = \tilde{z}, \tag{30}$$

$$\tilde{z} = z - Qy, \tag{31}$$

$$Qy = (\Xi - G)x, \tag{32}$$

where $y \in \mathbb{R}^{n_1}$ collects the possible nonvanishing values of $(\Xi - G)x$ and Q is a matrix of dimensions $K \times n_1$ with one nonvanishing coefficient (equal to one) on each column, which then satisfies the following properties:

$$Q^t Q = I_{n_1} \tag{33}$$

and

$$QQ^t(\Xi - G) = \Xi - G. \tag{34}$$

Now, (32) reads

$$y - Q^t(\Xi - G)x = 0. \tag{35}$$

Inserting (30) and (31) into (35) leads to

$$\left(I_{n_1} + Q^t(\Xi - G)G^{-1}Q\right) y = Q^t(\Xi - G)G^{-1}z. \tag{36}$$

It can be proved as in [23] that the matrix $(I_{n_1} + Q^t(\Xi - G)G^{-1}Q)$ is a nonsingular matrix of dimension n_1.

We now explain how one can efficiently compute x using (36), (31), and (30). First, we need to compute $I_{n_1} + Q^t(\Xi - G)G^{-1}Q$. Let D_1 stand for the matrix

$$D_1 = \frac{1}{h^2}\begin{pmatrix} 1 & -1 & 0 & \cdots & 0 \\ -1 & 2 & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & 2 & -1 \\ 0 & \cdots & 0 & -1 & 1 \end{pmatrix} \in \mathcal{M}_{M,M}(\mathbb{R}).$$

This matrix corresponds up to a coefficient -1 to the discretized Laplace operator with Neumann boundary conditions in one-space dimension. Let $V \in \mathcal{M}_{M,M}(\mathbb{R})$ denote the matrix made of an orthonormal basis of eigenvectors of D_1:

$$V_{ij} = \eta_j \cos\left(\frac{(j-1)(i-\frac{1}{2})\pi}{M}\right) \ \forall\ 1 \le i, j \le M, \tag{37}$$

where the positive number η_j is chosen such that each column V_j satisfies: $\|V_j\|_2 = 1$, namely:

$$\eta_1 = \sqrt{\frac{1}{M}}, \tag{38}$$

$$\eta_j = \sqrt{\frac{2}{M}} \ \ \forall\ 2 \le j \le M. \tag{39}$$

Then we have:

$$V^{-1} = V^t,$$
$$V^t D_1 V = \Lambda,$$

where the diagonal matrix $\Lambda = diag(\lambda_1, \cdots, \lambda_M) \in \mathcal{M}_{M,M}(\mathbb{R})$ collects the eigenvalues $\lambda_i = \frac{4}{h^2}\sin^2\left(\frac{(i-1)\pi}{M}\right)$ of D_1. Let $W \in \mathcal{M}_{K,K}(\mathbb{R})$ be the block diagonal matrix, with N diagonal blocks equal to V:

$$W = \left(\begin{array}{c|c|c|c|c} V & \mathbb{O} & \cdots & \cdots & \mathbb{O} \\ \hline \mathbb{O} & \ddots & & & \vdots \\ \hline \vdots & & \ddots & & \vdots \\ \hline \vdots & & & \ddots & \mathbb{O} \\ \hline \mathbb{O} & \cdots & \cdots & \mathbb{O} & V \end{array}\right) \in \mathcal{M}_{K,K}(\mathbb{R}).$$

Let $D_2 \in \mathcal{M}_{K,K}(\mathbb{R})$ be the matrix defined by:

$$D_2 = \left(\begin{array}{c|c|c|c|c} \tilde{\Lambda}_1 & -U_1 & \mathbb{O} & \cdots & \mathbb{O} \\ \hline -L_2 & \ddots & \ddots & & \vdots \\ \hline \mathbb{O} & \ddots & \ddots & \ddots & \mathbb{O} \\ \hline \vdots & & \ddots & \ddots & -U_{N-1} \\ \hline \mathbb{O} & \cdots & \mathbb{O} & -L_N & \tilde{\Lambda}_N \end{array}\right) \in \mathcal{M}_{K,K}(\mathbb{R})$$

with

$$\begin{aligned} L_i &= \frac{2}{\ell_i(\ell_i + \ell_{i-1})} I_M \in \mathcal{M}_M(\mathbb{R}), \\ U_i &= \frac{2}{\ell_i(\ell_i + \ell_{i+1})} I_M \in \mathcal{M}_M(\mathbb{R}), \\ \tilde{\Lambda}_1 &= U_1 + \Lambda = \frac{2}{\ell_1(\ell_1 + \ell_2)} I_M + \Lambda \in \mathcal{M}_M(\mathbb{R}), \\ \tilde{\Lambda}_N &= L_N + \Lambda = \frac{2}{\ell_N(\ell_{N-1} + \ell_N)} I_M + \Lambda \in \mathcal{M}_M(\mathbb{R}), \\ \forall 1 < i < N \quad \tilde{\Lambda}_i &= L_i + U_i + \Lambda = \frac{2(\ell_{i-1} + 2\ell_i + \ell_{i+1})}{\ell_i(\ell_i + \ell_{i-1})(\ell_i + \ell_{i+1})} I_M + \Lambda \in \mathcal{M}_M(\mathbb{R}). \end{aligned}$$

One can see that

$$W^t G W = \tilde{D}_2,$$

where the matricies D_2 and $\tilde{D}_2$ differ on the first row and on the first column *(due to the specific modification made on the matrix G to insure its non-singularity, see remark 2).*

Thus, we have

$$I_{n_1} + Q^t(\Xi - G)G^{-1}Q = I_{n_1} + Q^t(\Xi - G)W^t\tilde{D}_2^{-1}WQ.$$

We now show how one can compute this matrix in $\mathcal{O}(\Pi^3)$ floating point operations:

- The computation of $WQ \in \mathcal{M}_{K,n_1}(\mathbb{R})$ requires the computation of the eigenvectors of D_1 given by (37) (this gives $\mathcal{O}(M^2)$ floating point operations). Note that each column of WQ is a vector in $\mathbb{R}^K$ with M nonvanishing coefficients at most.

- The UL factorization of $\tilde{D}_2$ allows to solve linear systems associated with this matrix in $\mathcal{O}(K)$ floating point operations, since the profiles of the matrices U and L of this factorization are the same as the profile of the matrix $\tilde{D}_2$. Thus, for each column w of WQ, we need $\mathcal{O}(K)$ floating point operations to compute $\tilde{D}_2^{-1}w$; this gives $\mathcal{O}(n_1K)$ floating point operations for the computation of $\tilde{D}_2^{-1}WQ \in \mathcal{M}_{K,n_1}$.

- By Remark 3, we see that $(\Xi-G)W^t$ can be computed in $\mathcal{O}(K)$ floating point operations, and so the matrix $Q^t(\Xi - G)W^t \in \mathcal{M}_{n_1,K}$ can be computed in $\mathcal{O}(K)$ operations and has at most $\mathcal{O}(\Pi)$ nonvanishing coefficients on each of its rows.

- Finally, the product of $Q^t(\Xi - G)W^t$ by $\tilde{D}_2^{-1}WQ$ can be computed in $\mathcal{O}(n_1\Pi^2)$ floating point operations, and since this matrix is in $\mathcal{M}_{n_1,n_1}(\mathbb{R})$, its LU-factorization can also be computed in $\mathcal{O}(n_1^3)$ floating point operations.

We then need $\mathcal{O}(\Pi^3)$ floating point operations to compute the LU-factorization of the matrix on the left-hand side of (36). This preprocessing step is done once (for all) at the beginning of the code.

For each time iteration, the computation of the pseudo-pressure requires to compute the right-hand side of (36) and then to solve the linear system in (36) using the LU-factorization of $(I_{n_1} + Q^t(\Xi - G)G^{-1}Q)$ computed in the preprocessing step. The computation of $G^{-1}z$ requires $\mathcal{O}(NM \log M) = \mathcal{O}(K \log M)$ floating point operations using fast Fourier transforms (see [24], [25]), and the product by $Q^t(\Xi - G)$ requires $\mathcal{O}(n_1)$ operations (using again Remark 3). We then need $\mathcal{O}(K \log M)$ floating point operations to compute y in (36); it then takes another $\mathcal{O}(K \log M)$ operations to compute x in (30) (the computation of (31) being neglectible).

Similar strategies can be written to compute the predictor of the velocity components, since the discretized systems satisfied by this vectors are similar to the discretization of parabolic problems (see [23]).

Computing the solution at final time T then requires $\mathcal{O}(\Pi^3)$ floating point operations for the preprocessing step and $\mathcal{O}(\delta t^{-1}\Pi^2(\log M))$ floating point operations for $N = T\delta t^{-1}$ time-steps.

4. Numerical results

4.1. Taylor-Couette flow

As in [16], we run numerical tests on the Taylor-Couette flow between two concentric circular cylinders to study the convergence of the scheme.

Respective radii are $R_1 = 1$ and $R_2 = 4$. The larger cylinder does not move, the smaller rotates with an angular velocity ω_0 equal to 1. This flow is characterized by the Taylor number Ta defined by

$$\mathrm{Ta} = \frac{{\omega_0}^2(R_1 + R_2)(R_2 - R_1)^3}{2\nu^2}.$$

The fluid viscosity ν is adjusted so that $\mathrm{Ta} = 1000$. According to [26], the flow converges to a steady state for Ta smaller than the critical value $\mathrm{Ta}_c = 1712$. In this case, the steady state is given by:

$$\mathbf{u}(x, y) = \omega(r)\big(- (y - y_C)\ ,\ (x - x_C)\big)^t$$

with

$$r = \sqrt{(x - x_C)^2 + (y - y_C)^2}, \quad \omega(r) = K({R_2}^2 - r^2)/r^2$$

and the pressure is given by

$$p(r) = K^2\left(\frac{r^2}{2} - \frac{{R_2}^4}{2r^2} - {R_2}^2 log(r^2)\right),$$

where

$$K = \omega_0 {R_1}^2/({R_2}^2 - {R_1}^2).$$

The computational domain $\Omega = (-5, 5)^2$ is discretized with uniform meshes of size $h = 10/N$, $N \in \{48, 64, 96, 128, 192, 256, 384, 512\}$, and, for every simulation, the time-step δt is 2.5 10^{-3}.

The center of the cylinders is $C = (1.3\ 10^{-2}\ ,\ 2.3\ 10^{-2})^t$, it does not coincide with any grid point as in [16]. We have computed a discrete version of the L_∞-error for velocity and pressure in the whole fluid domain, even in cells cut by the cylinders, namely:

$$\max_{\substack{(i,j) s.t.\\ x_{i-1/2}, y_{j-1/2} \in \Omega^F}} (P_{i,j} - P(x_{i-1/2}, y_{j-1/2}))$$

for the pressure, and analoguously for the velocity components, including the shifted points close to the obstacle.

As shown on the Figure 7, the second-order convergence in space is observed.

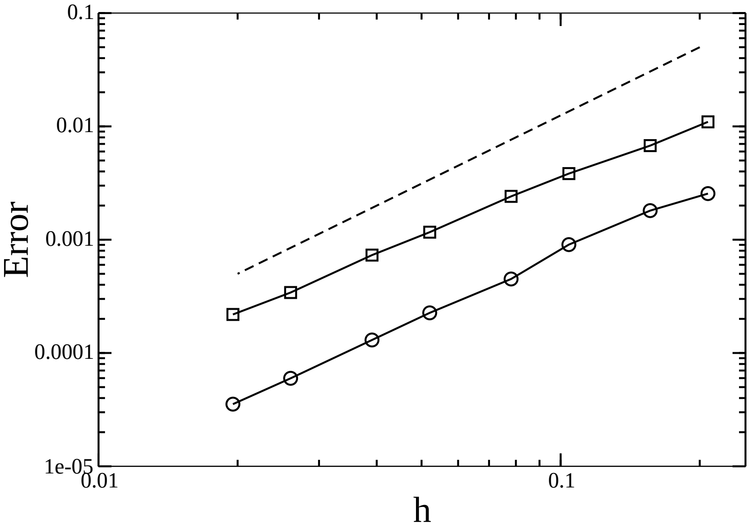

Figure 7: Taylor-Couette flow at $\mathrm{Ta} = 10^3$: error versus grid size for velocity (circles) and pressure (squares) fields. Dashed line represents second-order slope.

4.2. Flow past a circular cylinder

In this section, we present some numerical simulations of flows past a fixed circular cylinder for Reynolds number up to 9 500. The horizontal free stream velocity u_∞ and the cylinder diameter are equal to 1. Specific boundary conditions (b.c.) are used on the different part of $\partial\Omega^F$: uniform velocity profil $(u_\infty, 0)$ on inflow plan, convective outflow b.c. at the exit plan, slip b.c. ($\partial_y u = 0$, $v = 0$) on top and bottom of the domain, and no slip b.c. on the obstacle $\mathbf{u} = \mathbf{0}$. Several characteritic dimensions are defined for this flow, based on the stationnary state (see figure 8): the recirculation length l, the distance cylinder-vortex a, the distance between the two vortices b and the separation angle θ. Drag and lift coefficients (C_d , C_l) are computed

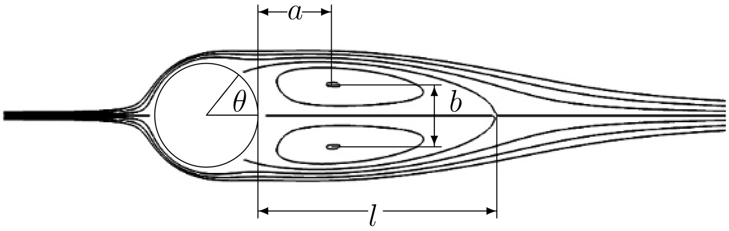

Figure 8: Characteristic dimensions of a stationary flow past a circular cylinder.

from the force applied by the fluid on the obstacle

$$F_{fluid/obst.} = \int_{\Gamma_S} \left(-p\mathbf{n} + \nu \frac{\partial \mathbf{u}}{\partial n} \right) \, d\sigma. \tag{40}$$

In the present situation, the computation of C_d and C_l reduces to $C_d = 2F_{fluid/obst.} \cdot (1,0)$ and $C_l = 2F_{fluid/obst.} \cdot (0,1)$ (see [26], [27] and [28]). The computation of (40) is achieved by using first-order approximations of p and $\frac{\partial \mathbf{u}}{\partial n}$ on each cut-cell.

4.2.1. Solution at Re = 40

Numerical results on square computational domains $\Omega = (-L, L)^2$ with $L \in \{5, 10, 15, 20, 25, 30, 35, 40\}$ show the influence of the boundary conditions on the development of the flow. The mesh size h ranges from $\approx 7.8\ 10^{-2}$ for the coarsest mesh to $h \approx 2.6\ 10^{-2}$ for the finest mesh. The number of grid points (resp. number of cut-cells) ranges from 128^2 to 3072^2 (resp. 52 and 156). The time-step is chosen so that the CFL number does not exceed 0.5 during the whole simulation. We observe in Table 1 that the drag coefficient is overestimated when the computational domain is not wide enough. For finest mesh and wider domains, the drag coefficient converges to the value of 1.50 which is selected for the comparison with the other results found in the literature. Results obtained with the present method with the finest grid ($L = 40$ and $h = 10/384$) present a good agreement with other numerical and experimental results reported in Table 2. Figure 9 shows velocity streamlines and pressure contours close to the cylinder. The pressure field is well resolved, there is no oscillation even close to the obstacle.

	$L=5$	$L=10$	$L=15$	$L=20$	$L=25$	$L=30$	$L=35$	$L=40$
$h=10/128$	1.734	1.568	1.524	1.505	1.494	1.487	1.482	1.479
$h=10/192$	1.730	1.565	1.522	1.503	1.492	1.485	1.481	1.477
$h=10/256$	1.748	1.582	1.538	1.519	1.508	1.502	1.497	1.493
$h=10/384$	1.758	1.591	1.547	1.528	1.517	1.510	1.505	**1.502**

Table 1: Grid convergence for the flow past a circular cylinder at Re = 40: drag coefficient as a function of h and L.

4.2.2. Solution at Re = 200

While the flow stays perfectly symmetric at Reynolds number 40, the symmetry of the flow is lost after a long time simulation at $Re = 200$ due

	C_d	θ	l	a	b
Bouard et al [29]	-	53.8	2.13	0.76	0.59
Dennis et al [30]	1.52	53.8	2.35	-	-
Ding et al [31]	1.58	52.8	2.32	-	-
Fornberg [32]	1.50	55.6	2.24	-	-
Linnick et al [33]	1.54	53.6	2.28	0.72	0.60
Taira et al [34]	1.55	54.1		0.73	0.60
Ye et al [12]	1.52	-	2.27	-	-
Present study	**1.50**	**53.4**	**2.26**	**0.71**	**0.60**

Table 2: Flow past a circular cylinder at $Re = 40$: comparison of characteristic dimensions of the flow and drag coefficient with literature. Dashes corresponds to unavailable results.

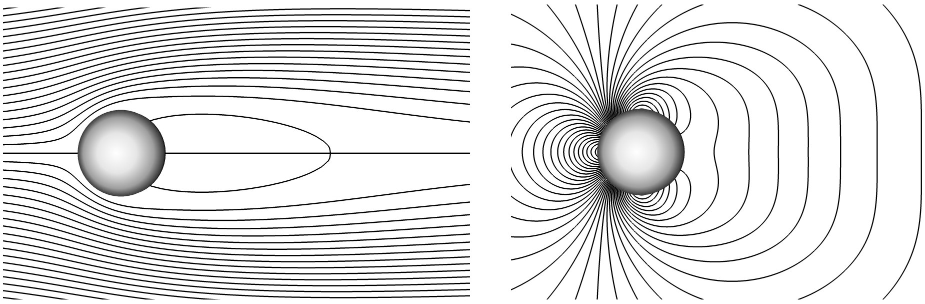

Figure 9: Flow past a circular cylinder at $Re = 40$: streamlines (left) and pressure contours (right).

to rounding errors inherent in computer calculations. The vortex shedding appears earlier if an artificial perturbation is used at the beginning of the simulation, but the numerical results obtained in both cases are the same for large times of simulation. As the flow reaches a periodic steady state, the drag and lift coefficients are computed as well as the Strouhal number St defined as the oscillation period of the lift coefficient. Table 3 contains results available in the literature obtained by other numerical methods or experiments. We found a good agreement with results obtained on the finest grid ($L = 40$ and $h = 10/384$). Figure 10 shows velocity streamlines and pressure contours close to the cylinder.

	C_d	C_l	St
Belov et al [35]	1.19 ± 0.042	0.64	0.193
Linnick et al [33]	1.34 ± 0.044	0.69	0.197
Liu et al [36]	1.31 ± 0.049	0.69	0.192
Rogers et al [37]	1.23 ± 0.05	0.65	0.185
Taira et al [34]	1.36 ± 0.043	0.69	0.197
Present study	**1.380 ± 0.0445**	**0.678**	**0.197**

Table 3: Flow past a circular cylinder at $Re = 200$: comparison of drag coefficient, lift coefficient and Strouhal number with literature.

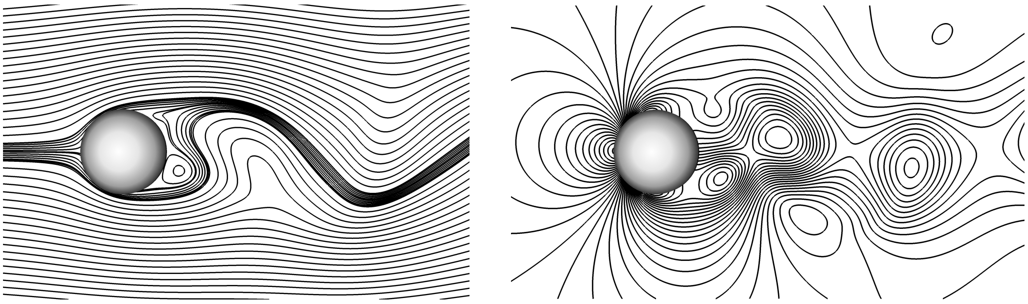

Figure 10: Flow past a circular cylinder at $Re = 200$: streamlines (left) and pressure contours (right) at time $t = 100$.

4.2.3. Solution at $Re = 9\,500$

The flow past a circular cylinder at Reynolds number 9 500 is simulated to show the robustness of the scheme. Many complex vortex strutures appear close to the obstacle (see figure 11). This challenging test has been investigated by many author, some comparisons can be found in the literature (see [27], [28], [38], [39], [40], [41]). In the Figure 11, velocity streamlines obtained with the present numerical method is compared with the experimental results given in [39]. For this numerical simulation, the computational domain is $\Omega = (-5, 5) \times (-2.5, 2.5)$. The grid is uniform in the vertical direction and non-uniform in the horizontal direction. Square cells of size $h = 5/3072$ are used in the vicinity of the obstacle. The value of the time step, satisfying a CFL stability condition, is 10^{-4}. This excellent agreement suggests the accuracy of the present scheme.

4.3. NACA aerofoil at Reynolds number Re $= 1\,000$

We have also studied the flow past a NACA aerofoil 0012 at incidence 34 degres for $\mathrm{Re} = 1\,000$ (see [41], [42]). A Karman vortex street develops behind the obstacle (see Figure 12): the flow is well resolved even near the sharp ending edge. For these numerical simulations, the computational domain is $\Omega = (-5, 5) \times (-2.5, 2.5)$ and the mesh size near the obstacle is $5/3072$. The value of the time step, satisfying a CFL stability condition, is 10^{-4}.

5. Concluding remarks and perspectives

We have presented a new cut-cell method for the simulation of flows past obstacles. The equations near the obstacle are discretized with first-order formulae, but the overall second-order accuracy in space is still observed on numerical tests. This new scheme is also validated on flows at moderate and high Reynolds numbers. Results found in literature are compared with the present method and a good agreement is observed.

This method can easily be adapted to moving obstacles problems by levelset methods (see [18]), since the signed distance function defined in (4) is a particular levelset function defining the boundary Γ^S. For this case of moving boundary problems, iterative solvers for the non symmetric linear systems should be preferred to the direct method described in Section 3.6, in order to avoid the “preprocessing step” in $\mathcal{O}(\Pi^3)$ floating operations which should be performed at each time-step to take into account the new position of the boundary.

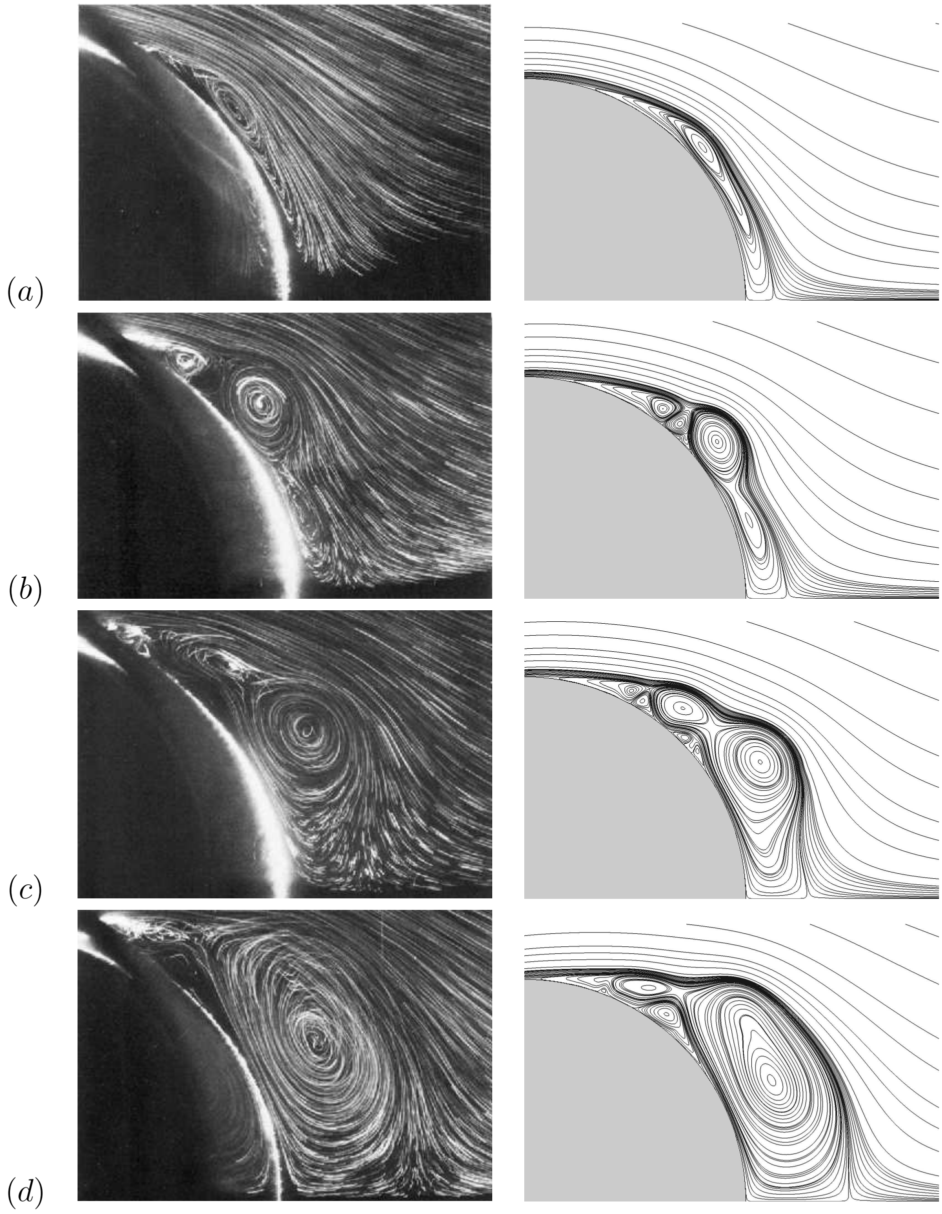

Figure 11: Flow past a circular cylinder at $Re = 9\,500$. Experimental results from Bouard and Coutanceau [39] (left) and streamlines obtained with the present numerical method (right). Comparison at time (*a*) $t = 0.75$, (*b*) $t = 1.0$, (*c*) $t = 1.25$, (*d*) $t = 1.5$.

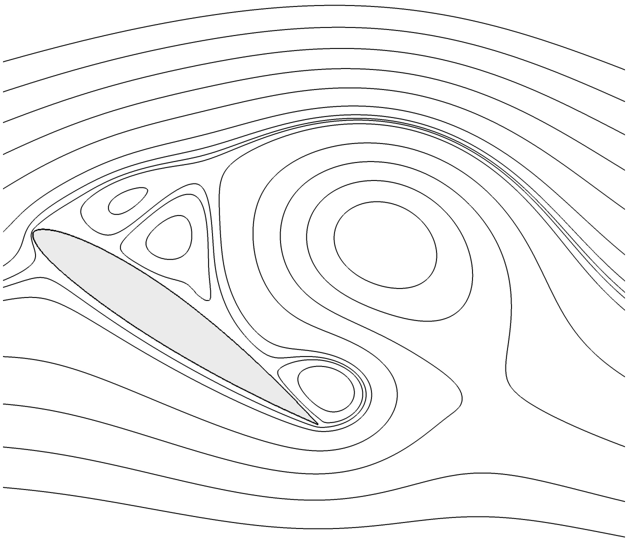

Figure 12: Flow past a NACA aerofoil at Re $= 1\,000$, adimensional time $t = 3.2$.

Acknowledgements

The numerical simulations presented in this paper were performed on the DELL cluster, with 24 bi-processors XEON (dual-core), of the Laboratoire de Mathmatiques (Universit Blaise Pascal and CNRS).

References

[1] C. S. Peskin, The fluid dynamics of heart valves: experimental, theoretical, and computational methods, Ann. Rev. Fluid Mech. 14 (1982) 235–259.

[2] C. S. Peskin, The immersed boundary method, Acta Numerica 11 (2002) 1–39.

[3] J. Mohd-Yusof, Combined immersed-boundary/B-Spline methods for simulations of flow in complex geometries, NASA Ames Research Center/Stanford University, 1997, pp. 317–327.

[4] E. M. Saiki, S. Biringen, Numerical simulation of a cylinder in uniform flow: application of a virtual boundary method, J. Comput. Phys. 123 (1996) 450–465.

[5] N. Zhang, Z. Zheng, An improved direct-forcing immersed boundary method for finite difference applications, J. Comput. Phys. 221 (2007) 250–268.

[6] R. Mittal, G. Iaccarino, Immersed boundary methods, Annual Review of Fluid Mechanics 37 (2005) 239–261.

[7] E. Fadlun, R. Verzicco, P. Orlandi, J. Mohd-Yusof, Combined immersed-boundary finite difference methods for three-dimensional complex flow simulations, J. Comput. Phys. 161 (2000) 35–60.

[8] M. de Tullio, A. Cristallo, E. Balaras, G. Pascazio, P. D. Palma, G. Iaccarino, M. Napolitano, R. Verzicco, Recent advances in the immersed boundary method, in: P. Wesseling, E. Oñate, J. Périaux (Eds.), ECCOMAS CFD, 2006.

[9] J. Kim, D. Kim, H. Choi, An immersed-boundary finite volume method for simulation of flow in complex geometries, J. Comput. Phys. 171 (2001) 132–150.

[10] F. Muldoon, S. Acharya, A divergence-free interpolation scheme for the immersed boundary method, Int. J. Numer. Method Fluid 56 (2008) 1845–1884.

[11] P. Angot, C. Bruneau, P. Fabrie, A penalization method to take into account obstacles in incompressible viscous flows, Numer. Math. 81 (1999) 497–520.

[12] T. Ye, R. Mittal, H. Udaykumar, W. Shyy, Numerical simulation of two-dimensional flows over a circular cylinder using the immersed boundary method, J. Comp. Phys. 156 (1999) 209–240.

[13] P. Tucker, Z. Pan, A cartesian cut-cell method for incompressible viscous flow, Appl. Math. Model. 24 (2000) 591–606.

[14] M.-H. Chung, Cartesian cut cell approach for simulating incompressible flows with rigid bodies of arbitrary shape, Computers and Fluids 35 (6) (2006) 607–623.

[15] R. Mittal, H. Dong, M. Bozkurttas, F. Najjar, A. Vargas, A. Loebbecke, A versatile sharp interface immersed boundary method for incompressible flows with complex boundaries, J. Comput. Phys. 227 (2008) 4825–4852.

[16] Y. Cheny, O. Botella, The ls-stag method: A new immersed boundary/level-set method for the computation of incompressible viscous flows in complex moving geometries with good conservation properties, J. Comp. phys. 229 (2010) 1043–1076.

[17] F. Harlow, J. Welch, Numerical calculation of time-dependent viscous incompressible fl ow of fluid with free surface, Phys. Fluids 12 (1965) 2182–2189.

[18] S. Osher, J. A. Sethian, Fronts propagating with curvature dependent speed: Algorithms based on hamilton-jacobi formulations, J. Comput. Phys. 79 (1) (1988) 12–49.

[19] N. Matsunaga, Y. Yamamoto, Superconvergence of the shortley-weller approximation for dirichlet problems, J. Comp. Appl. Math. 116 (2000) 263–273.

[20] B. Buzbee, F. Dorr, The direct solution of the biharmonic equation on rectangular regions and the poisson equation on irregular regions, SIAM J. Num. Anal. 11 (1974) 753–763.

[21] B. Buzbee, F. Dorr, J. George, G. Golub, The direct solution of the discrete poisson equation on irregular regions, SIAM J. Numer. Anal. 8 (1971) 722–736.

[22] R. J. LeVeque, Z. Li, The immersed interface method for elliptic equations with discontinuous coefficients and singular sources, SIAM J. Numer. Anal. 31 (1994) 1019–1044.

[23] F. Bouchon, G. H. Peichl, The immersed interface technique for parabolic problems with mixed boundary conditions, SIAM J. Num. Anal. 48 (2010) 2247–2266.

[24] R. Hockney, A fast direct solution of poisson's equation using fourier analysis, J. ACM 12 (1965) 95–113.

[25] P. Swarztrauber, The methods of cyclic reduction, fourier analysis and the facr algorithm for the discrete solution of poisson's equation on a rectangle, SIAM Rev. 19 (1977) 490–501.

[26] E. Guyon, J. Hulin, L. Petit, C. Mitescu, Physical Hydrodynamics, Oxford University Press, 2001.

[27] C.-C. Chang, R.-L. Chern, A numerical study of flow around an impulsively started circular cylinder by a deterministic vortex method, Journal of Fluid Mechanics 233 (1991) 243–263.

[28] X. D. Niu, Y. T. Chew, C. Shu, Simulation of flows around an impulsively started circular cylinder by taylor series expansion-and least squares-based lattice boltzmann method, J. Comput. Phys. 188 (2003) 176–193.

[29] R. Bouard, M. Coutanceau, Experimental determination of the main features of the viscous flow in the wake of a circular cylinder in uniform translation. part 1. steady flow, J. Fluid Mech. 79 (1977) 231–256.

[30] S. Dennis, G. Chang, Numerical solutions for steady flow past a circular cylinder at reynolds number up to 100., J. Fluid Mech. 42 (1970) 471–489.

[31] H. Ding, C. Shu, Q. Cai, Applications of stencil-adaptative finite difference method to incompressible viscous flows with curved boundary, Comp. Fluids 36 (2007) 786–793.

[32] B. Fornberg, A numerical study of steady viscous flow past a circular cylinder, J. Fluid Mech. 98 (1980) 819–855.

[33] M. Linnick, H. Fasel, A high-order immersed boundary interface method for simulating unsteady incompressible flows on irregular domains, J. Comp. Phys. 204 (2005) 157–192.

[34] K. Taira, T. Colonius, The immersed boundary method : a projection approach, J. Comp. Phys. 225 (2007) 2118–2137.

[35] A. Belov, L. Martinelli, A. Jameson, A new implicit algorithm with multigrid for unsteady incompressible flow calculations., AIAA Journal.

[36] C. Liu, X. Zheng, C. Sung, Preconditioned multigrid methods for unsteady incompressible flows., J. Comp. Phys. 139 (1998) 35–57.

[37] S. Rogers, D. Kwak, Upwind differencing scheme for the time-accurate incompressible navier-stokes equations, AIAA Journal 28 (1990) 254–262.

[38] T. K. Sengupta, R. Sengupta, Flow past an impulsively started circular cylinder at high reynolds number, Computational Mechanics 14 (1994) 298–310.

[39] R. Bouard, M. Coutenceau, The early stages of development of the wake behind an impulsively started cylinder for $40 < re < 10^4$, J. Fluid. Mech. 101 (1980) 583–607.

[40] P. Koumoutsakos, A. Leonard, High-resolutions of the flow around an impulsively started cylinder using vortex methods, J. Fluid Mech. 296 (1995) 1–38.

[41] A. George, L. Huang, W. Tang, Y. Wu, Numerical simulation of unsteady incompressible flow ($re < 9500$) on the curvilinear half-staggered mesh, SIAM J. Sci. Comp. 21 (6) (2000) 2331–2351.

[42] A. George, W. Tang, Y. Wu, Multilevel one-way dissection factorization, SIAM J. Matrix Anal. 22 (6) (2001) 752–771.