UCLouvain

École polytechnique de Louvain

Efficient and accurate simulations of three-dimensional flows past obstacles with vorticity-based penalization methods

Author: **Alexandre Youssef**
Supervisors: **Philippe Chatelain, Matthieu Duponcheel**
Reader: **Pierre-Antoine Absil**
Academic year 2022–2023
Master [120] in Mathematical Engineering

Acknowledgements

This work concludes my studies as a Mathematical Engineering student at the Université Catholique de Louvain. To begin with, I would like to thank my two thesis supervisors, the Professors Philippe Chatelain and Matthieu Duponcheel, for their availability throughout the year, their accurate comments on my work, their experience that helped me, and for providing me with the opportunity to work in an international research framework. This experience has undoubtedly enhanced my understanding of fluid mechanics and fostered my interest for this field of engineering.

I would like to extend my thanks to the Professor Pierre-Antoine Absil, for reviewing my work and for providing his valuable perspective from the field of Mathematical Engineering.

I also want to thank Pierre Balty for the endless discussions and meetings we had and for his continuous help during the year with specific and general questions. Many thanks to Gilles Poncelet for his expertise and for being available to assist me when needed. I am also grateful to Thomas Gillis for his patience in answering my many questions and for sharing his experience with penalization methods. Additionally, I would like to thank the Van Rees Lab team for their help on computational and technical details.

Finally, I wish to acknowledge my parents for their moral support and genuine interest in my work, as well as for their support during those five years at UCLouvain.

Contents

Introduction

Over the past decades, the importance of fluid mechanics in modern engineering practices has been steadily growing. Particularly, the analysis of flows past arbitrary shaped obstacles has become of significant interest, with applications ranging from academic research to concrete applications in aerodynamics or hydrodynamics. For instance, we can mention the importance of flow visualization, fluid-structure interaction or forces evaluation, that are crucial for our comprehension of the underlying physics. In addition to that, these insights have contributed to the development of innovative techniques for optimizing obstacle designs and improving performance in various industries, including aerospace and marine engineering.

To this end, numerical simulations have been extensively developed and have pushed further our understanding of fluid mechanics. This work focuses on the use of three-dimensional Brinkman penalization techniques. In brief, the objective is to model the flow around a body immersed in a viscous fluid by penalizing the no-slip boundary condition on its surface. The major advantage of this method is its ability to accommodate arbitrary bodies in a flow, making it suitable in complex cases. Indeed, such methods do not require the mesh to conform to the geometry of the object, so that structured Cartesian grids can be used and fast computations can be performed.

To be more specific, penalization techniques have been extensively studied in the so-called velocity-pressure formulation of the Navier-Stokes equations. However, in order to fit within a Lagrangian framework, we rather consider a vorticity-velocity formulation. While several studies have been conducted in 2D within this framework, 3D penalization techniques have not yet been thoroughly investigated

to fully comprehend their potential. Hence, the goal of this master's thesis is to build upon previous research and to explore new opportunities offered by Brinkman penalization techniques, such as three-dimensional iterative penalization or applications in aeronautics.

The *Murphy* framework

The focus of this work is on three-dimensional flows that are incompressible. The equations and the framework are implemented in C++, in a software called *Murphy*[1] [1] (standing for "*MUltiResolution multiPHYsics*"), which is jointly developed by UCLouvain and the Van Rees Lab at MIT.

Overall, the objective of this framework is to solve Partial Differential Equations (PDE), where a large range of different scales coexist, implying a large number of unknowns. *Murphy* works on two main aspects to be as general and efficient as possible: the algorithmic and the implementation. The final objective is to use structured collocated grids that are adaptively and automatically refined over time, to use as smartly as possible computational ressources. The code is still currently under development, following this optic as main thread.

In *Murphy*, the grid is split into three-dimensional blocks, but is not structured in an overlapping approach. It is rather in an octree based approach (tree-based, where each node has up to 8 children). This approach has a general distributed implementation in the library p4est [2]. Also, to benefit from massively parallel architectures, *Murphy* is based on the parallel computing library MPI. It is of particular interest for the ghost exchanges between the different computational subdomains. A GPU computing approach is also being developed, for the faster computations it provides for highly parallelizable tasks. Finally, *Murphy* uses external solvers and software, like for instance FLUPS[2] [3], that solves efficiently unbounded Poisson equations or also H3LPER[3], a tool for profiling, logging and parsing. The ultimate objective is to provide a general, optimal and flexible framework to solve optimally Partial Differential Equations.

In this work, we employ uniform Cartesian grids with a fully Eulerian framework, due to the ongoing development of a multiresolution Lagangian one. Nevertheless, this framework turns out to be sufficient for the Reynolds numbers and the cases

[1] `https://github.com/vanreeslab/murphy`
[2] `https://github.com/vortexlab-uclouvain/flups`
[3] `https://github.com/vanreeslab/h3lpr`

considered. Once finished, the code's flexibility and structure will easily accommodate those improvements, which will result in faster computational times. Those perspectives will naturally be pursued as a future direction of this work.

The three-dimensional incompressible Navier-Stokes equations

As a starting point, we consider the three-dimensional Navier-Stokes equations. Those come from the principles of conservation of mass, momentum and energy. Also, we consider the three following hypothesis:

1. The temperature equation is decoupled from the conservation of momentum equation. This means that the temperature field does not affect the velocity and pressure fields, and that the temperature equation can be solved independently.

2. The fluid considered is incompressible (constant density ρ of the flow in any fluid parcel). Under this hypothesis, the conservation of mass writes $\nabla \cdot \mathbf{u} = 0$, where $\mathbf{u}$ is the velocity field. This relates to the incompressibility condition.

3. Potential forces like gravity are negligible.

Under those assumptions, the Navier-Stokes equations in the velocity-pressure formulation simplify to

$$\nabla \cdot \mathbf{u} = 0 \qquad \text{(Conservation of mass)} \tag{1}$$

$$\underbrace{\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u}}_{=\frac{D\mathbf{u}}{Dt}} = -\nabla P + \nu \nabla^2 \mathbf{u} \qquad \text{(Conservation of momentum)} \tag{2}$$

where the operator $\frac{D(.)}{Dt}$ denotes the Lagrangian (or material) derivative, ν is the dynamic viscosity of the fluid and $\mathbf{P}$ is the reduced pressure. As a reminder, those two variables are defined as

$$\nu \triangleq \frac{\mu}{\rho} \qquad\qquad \mathbf{P} \triangleq \frac{\mathbf{p}}{\rho} \tag{3}$$

where μ is the kinematic viscosity and $\mathbf{p}$ is the regular pressure field. An adimensionalization of those equations eventually leads to a single parameter that governs the flow, the Reynolds number $Re \triangleq \frac{U_\infty L}{\nu}$, where L is a reference length. We note that $Re < 1$ leads to diffusion dominated flows, $1 < Re < 10^3$ leads to laminar flows and $Re > 10^3$ to turbulent flows, dominated by the advection-diffusion term.

In *Murphy* and in vorticity-based frameworks, the equations are rather implemented in another formulation, called the vorticity-velocity formulation. For this purpose, we define the three-dimensional vorticity field $\boldsymbol{\omega}$ as

$$\boldsymbol{\omega} \triangleq \nabla \times \mathbf{u} \tag{4}$$

By applying the curl operator $(\nabla \times \cdot)$ to Equation (2), we obtain the vorticity-velocity formulation of the momentum equation

$$\underbrace{\frac{\partial \boldsymbol{\omega}}{\partial t} + (\mathbf{u} \cdot \nabla)\boldsymbol{\omega}}_{=\frac{D\boldsymbol{\omega}}{Dt}} = \nu \nabla^2 \boldsymbol{\omega} + (\boldsymbol{\omega} \cdot \nabla)\mathbf{u} \tag{5}$$

$$\frac{\partial \boldsymbol{\omega}}{\partial t} = \underbrace{-\nabla \times (\mathbf{u} \times \boldsymbol{\omega})}_{(\boldsymbol{\omega}\cdot\nabla)\mathbf{u} - (\mathbf{u}\cdot\nabla)\boldsymbol{\omega}} + \nu \nabla^2 \boldsymbol{\omega} \tag{6}$$

$$\Rightarrow \frac{\partial \boldsymbol{\omega}}{\partial t} = \nabla \times ((\nabla \times \mathbf{u}) \times \mathbf{u}) + \nu \nabla^2 \boldsymbol{\omega} \tag{7}$$

where the last expression is obtained using Equation (4) and using the properties of the cross product. We note that on the contrary to the 2D formulation, the 3D one has a vortex stretching term $(\boldsymbol{\omega} \cdot \nabla)\mathbf{u}$. Also, we define the stream function $\boldsymbol{\Psi}$, that connects the vorticity and the velocity through a Helmholtz decomposition. If we consider a uniform free-stream velocity $\mathbf{u}_\infty$ as the only irrotational contribution [4], the Helmholtz decomposition simplifies to

$$\mathbf{u} = \nabla \times \boldsymbol{\Psi} + \mathbf{u}_\infty \tag{8}$$

Using this expression, we can rewrite the mass conservation Equation (1) as a Poisson equation solving for the stream function, which is given by

$$\nabla^2 \boldsymbol{\Psi} = -\boldsymbol{\omega} \tag{9}$$

Once the Poisson equation is solved for $\boldsymbol{\Psi}$, the velocity field $\mathbf{u}$ can be computed using Equation (8), that is by taking the curl of the stream function and shifting it by the free-stream velocity. This two steps computation is the approach done in *Murphy* for solving the Poisson problem. According to Gabbard [5], the use of the formulation presented in Equation (7) for solving the Navier-Stokes equations, combined with a stream function formulation for the Poisson problem, allows for the conservation of important quantities, such as energy and helicity.

Finally, we precise that only Direct Numerical Simulations (DNS) are considered in this work. As a reminder, DNS simulations can achieve very high spatial and

temporal resolutions, which allows to capture fine-scale turbulent structures and other important features of the flow. The disadvantage is that for higher Reynolds numbers, it becomes extremely expensive (the computational cost increases as Re^3 [6]), so that we are computationally limited on the Reynolds number choice. In those cases, Large Eddy Simulations (LES) are often used, since they use a filter to discard the small scales of the problem. We can also mention the Reynolds Averaged Navier-Stokes simulations (RANS) that use an averaged flow to alleviate the computational cost.

Outline

Concretely, this master's thesis is divided into two chapters. The first one covers the theoretical part of Brinkman penalization approaches and the second one the practical part, with computational results and analysis.

1. Firstly, we present the Brinkman penalization technique in the vorticity-velocity formulation of the Navier-Stokes equations. The concepts of continuous and discrete forcing approaches are briefly discussed. Then, we present three approaches that take into account the penalization term, increasing gradually in accuracy. We also cover the ways to compute the aerodynamic forces acting on objects.

    - As a starting point, we consider a direct explicit method, that consists in an explicit Euler discretization of the penalization term. Altough simple to understand and to implement, the too strong stability constraint on the penalization term will suggest to investigate other methods.
    - As a solution to this problem, we cover an implicit approach, based on an implicit Euler discretization of the velocity-based penalization term. The stability region generated by this scheme allows for an unconstrained choice of the so-called Lagrange multiplier λ that parametrize the obstacle. However, it is in practice observed that this method provides a poor accuracy of the solution and generates significant diffusion errors, especially for high Reynolds numbers.
    - Finally, we cover an iterative penalization approach, initially proposed by Hejlesen et al. [7], and further analyzed by Ramussen et al. [8]. As we will see, it provides a higher accuracy and captures properly the vortices generated by the objects. This method however needs to solve a Poisson problem at each iteration of the algorithm. There is thus a motivation in reducing the total number of iterations until convergence. Therefore, we briefly mention the work of Gillis et al. [9], that proposed

an iterative method based on Krylov recycled spaces. This approach is only covered theoretically in this work. The objective is to provide an overview of the improvement opportunities proposed by iterative penalization techniques.

2. Secondly, the validity of the different implementations is assessed and a comparison between the different methods is performed. Also, we consider a concrete application in the field of aeronautics to test a more complex object geometry. In this work, we consider flows past three types of obstacles:

   - Impulsively started flows past a circular cylinder, at a moderate Reynolds number of $Re = 550$ and a high Reynolds number of $Re = 9500$. Those choices are justified by the various experiments performed in the literature, and are a way to check for the validity of our implementations. A high Reynolds number of $Re = 9500$ will outline the advantages of iterative penalization methods. Different convergence studies and analysis are also performed on this obstacle.
   - A flow past a sphere at $Re = 300$. For this case, an impulsively started flow is considered, as well as a long time simulation were instabilities are artificially triggered with a perturbation of the uniform upstream flow. The frequency of the vortex sheddings as well as the lift and drag coefficients are compared with the literature. The main aim is to analyze the accuracy of the iterative penalization in a purely three-dimensional scenario and to compare it with the implicit penalization approach.
   - An application on flows past extruded airfoils at $Re = 1000$ and $Re = 5000$. This application is a first step towards penalization methods applied to the field of aeronautics. We compare a regularized (rounded trailing edge) Joukowski airfoil and a symmetric 4-digit NACA airfoil, both at an angle of attack $\alpha = 20°$. Those Reynolds numbers are for example typically encountered in the analysis and design of airfoils for micro air vehicles (MAVs) or other small-scale applications, where low Reynolds number flows are dominant. Finally, the second Reynolds number will outline the necessity to use a multi-resolution framework to reduce the computational time and increase the resolution of the boundary layer of the airfoils, especially at the leading and trailing edges where those are very thin.

The conclusions of this report outline the main outcomes of this study. The advantages of each penalization method are presented in a summary, and a trade-off between computational cost and accuracy is emphasized. Additionally, the main areas of improvement are identified as the final remarks of this report.

CHAPTER 1

Methodology

Contents

1.1 Introduction

In this first chapter, the Brinkman penalization technique is described and detailed. This method is used to embed arbitrarily shaped geometries in a fluid. We first consider the velocity-pressure formulation of the incompressible Navier-Stokes equations, and then use the corresponding vorticity-velocity formulation, that we consider for the rest of this work.

In a second step, we present three different algorithmic techniques for implementing the penalization term. The first technique is the most straightforward, implementing the penalization term explicitly. The second technique is an implicit approach based on the velocity penalization formulation, which allows for to discard the time step constraint imposed by the explicit method. Finally, we describe an iterative penalization approach introduced by Hejlesen et al. [7], which improves the accuracy of the solution.

Continuous forcing approach

The fundamental idea of the Brinkman penalization approach is to consider an additional term in the Navier-Stokes equations. The objective is to represent the force on the fluid exerted by the obstacle. Such approaches are called continuous forcing approaches (in opposition to discrete forcing approaches). There are two widespread techniques in this category: the Brinkman penalization approach and the Immersed Boundary (IB) approach.

The Brinkman penalization is based on a Lagrangian relaxation of the body constraint to impose the inner boundary condition [10]. Its main idea is to model solid obstacles as porous media with porosity, and viscous permeability approaching zero. The biggest advantage of this method is its straightforward implementation for any kind of geometry. This type of method also fits to Vortex Particle-Mesh (VPM) methods, which are widely used.

The IB method originally refers to the work proposed by Peskin [11] on fluid-structure interaction. In this method, the fluid is represented in an Eulerian coordinate system and the structure is represented in Lagrangian coordinates. As mentioned by Gillis [6], the method is based on discretizing the forcing term on the surface of the object in several grid points using regularized Dirac functions, hence mollifying the inner interface of the obstacle.

Discrete forcing approach

In contrast, discrete forcing approaches modify the discretization of the Navier-Stokes equations to incorporate the effects of an obstacle. This means that the modification is not directly applied to the Navier-Stokes equations, but rather to the discretized problem. Consequently, these methods are typically more complex than continuous approaches.

In this category, a first approach is the Ghost-Cell Method. In this method [12], the computational grid is extended, and cells neighboring the solid boundary are identified as ghost cells. Those cells have the particularity to satisfy conditions at the boundary. To deal with complex geometries, Ghost-Cell methods are often coupled with Level-Set methods [13].

The Boundary Element Method (BEM), also known as the panel method, is another technique within the same framework. It discretizes the object's boundary into panels and approximates the quantities on these panels using known-shaped distributions with unknown strengths. Those are determined by applying various boundary conditions. The BEM is similar to penalization methods as it computes the required vortex sheet to eliminate slip velocity at the interface.

Finally, there is the Immersed Interface Method (IIM) [14]. The IIM is based on Cartesian grids and sharply imposes stress jump conditions, enabling higher-order accuracy. The IIM introduces a set of fictitious forces that are used to model the interaction between the fluid and solid domains. These forces are derived from a discretization of the equations of motion and are imposed on the fluid equations at the interface between the two domains.

1.2 The Brinkman penalization

1.2.1 Velocity-pressure formulation

The objective of the Brinkman penalization technique is to force the velocity field $\mathbf{u}$ to be equal to the velocity of the obstacle $\mathbf{u}_b$ inside its region Ω_b [15]. It is thus natural to apply the constraint $\mathbf{u} = \mathbf{u}_b$ in Ω_b in the velocity-pressure formulation of the incompressible Navier-Stokes equations. Based on Equation (2), the penalized velocity-pressure formulation writes

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\nabla P + \nu \nabla^2 \mathbf{u} + \lambda \chi (\mathbf{u}_b - \mathbf{u}) \tag{1.1}$$

where we introduce χ as a mask function acting on the obstacle and λ as the so-called non-dimensional penalization parameter (or Lagrangian multiplier). In its simplest form, the mask χ is implemented through a Heaviside step function

$$\chi = \begin{cases} 1 & \text{if } \mathbf{x} \in \Omega_b \\ 0 & \text{if } \mathbf{x} \notin \Omega_b \end{cases} \tag{1.2}$$

We note that many sources (like Chatelain et al. [16]) consider a mollified mask function to smooth the boundary of the object to avoid instabilities because of the sharp boundary. In this work, we do not mollify explicitly this mask function. The reason is that we consider a high order Poisson solver (regularized 10^{th} order, introduced by Hejlesen et al. [17]), so that the velocity field is ensured to be sufficiently smooth in the region of the obstacle [7]. Considering a sharp mask function χ has the advantage that the losses of accuracy close to the boundary of the object are avoided.

If we consider a direct discretization of the penalization term (iterative methods have a different meaning for it), λ has a physical interpretation. As described by Mimeau et al. [18, 19], its value allows to distinguish different materials, and is correlated to the Brinkman equation. It is an extension of Darcy's law, which governs the flow of fluids through porous materials. The Brinkman equation incorporates the effects of viscous flow and the presence of a pressure gradient in the fluid. Within this framework, λ is defined as

$$\lambda \triangleq \frac{\mu \Phi L}{\rho k U_\infty} \tag{1.3}$$

with k the intrinsic permeability, μ the viscosity, Φ the porosity of the porous medium, L the height of the obstacle, ρ the fluid density and U_∞ the upstream velocity. We see that λ is inversely proportional to k. In a fluid, the permeability is infinitely high, so that $\lambda \to 0$. We thus get back to Equation (2). On the other side, fully solid obstacles have a permeability close to zero (the solid does not allow any fluid to flow through it), so that $\lambda \to \infty$. To guarantee accurate results, the value of λ has thus to be as high as possible. We analyze the influence of λ on the accuracy of the solution into more details in Section 2.4.1. For a concrete application of the Brinkman penalization approach to passive flow control using porous media, the reader is referred to [20].

1.2.2 Vorticity-velocity formulation

To use vorticity-based frameworks, we need to transform the penalization term in the vorticity-velocity formulation of the Navier-Stokes equations. Applying the

curl operator ($\nabla \times \cdot$) on Equation (1.1), we directly obtain

$$\frac{\partial \boldsymbol{\omega}}{\partial t} = \underbrace{\nabla \times ((\nabla \times \mathbf{u}) \times \mathbf{u})}_{\text{Advection + Stretching}} + \underbrace{\nu \nabla^2 \boldsymbol{\omega}}_{\text{Diffusion}} + \underbrace{\lambda \nabla \times [\chi(\mathbf{u}_b - \mathbf{u})]}_{\text{Penalization}} \tag{1.4}$$

On the other side, we note that the Poisson problem (9)

$$\nabla^2 \boldsymbol{\Psi} = -\boldsymbol{\omega} \quad \Leftrightarrow \quad \nabla^2 \mathbf{u} = -\nabla \times \boldsymbol{\omega} \tag{1.5}$$

does not change, except that $\boldsymbol{\omega}$ is now penalized because of Equation (1.4).

The next objective is to discretize accurately Equation (1.4) in space and in time. For the rest of this work, we consider a splitting approach to update the different fields over time, as initially done by Sharma et al. [21] in the velocity-pressure formulation. The difference is that we do not use a temporal integration on the velocity field, but exclusively on the vorticity field, as done by Gazzola et al. [22]. Given a penalized vorticity field $\boldsymbol{\omega}^n$ at time step n, we compute $\boldsymbol{\omega}^{n+1}$ at time step $n+1$ in two substeps:

1. Solve the classical Navier-Stokes Equation (7) and obtain the unpenalized vorticity field $\widetilde{\boldsymbol{\omega}}^{n+1}$ at time step $n+1$. In this work, we consider a third order RK3-TVD scheme as time integrator (see Gottlieb et al. [23, 24], technical details about this choice are provided in Section 2.2).

2. Penalize the vorticity field $\widetilde{\boldsymbol{\omega}}^{n+1}$ to obtain the desired penalized field $\boldsymbol{\omega}^{n+1}$. As mentioned above, three different approaches are considered, gradually increasing in terms efficiency and accuracy : (1) An explicit (direct) approach (2) An implicit (direct) approach and (3) An iterative approach proposed by Hejlesen et al. [7]. All approaches are based on a first order Euler time integration scheme, so are 1^{st} order accurate in time.

1.3 Explicit penalization technique

1.3.1 Discretization

The first way to implement the penalized vorticity field is by discretizing

$$\frac{\partial \boldsymbol{\omega}}{\partial t} = \lambda \nabla \times [\chi(\mathbf{u}_b - \mathbf{u})] \tag{1.6}$$

using a first order explicit Euler scheme, where the input and output vorticity fields are respectively $\widetilde{\boldsymbol{\omega}}^{n+1}$ and $\boldsymbol{\omega}^{n+1}$. Considering Δt as the time step between

iterations n and $n+1$, we directly obtain

$$\frac{\boldsymbol{\omega}^{n+1} - \widetilde{\boldsymbol{\omega}}^{n+1}}{\Delta t} = \lambda \nabla \times [\chi(\mathbf{u}_b^n - \mathbf{u}^n)] \tag{1.7}$$

which leads to an explicit update of the vorticity field

$$\boldsymbol{\omega}^{n+1} = \widetilde{\boldsymbol{\omega}}^{n+1} + \underbrace{\lambda \Delta t \, \nabla \times [\chi(\mathbf{u}_b^n - \mathbf{u}^n)]}_{\triangleq \delta\boldsymbol{\omega}} \tag{1.8}$$

We note that the penalization term acts as a vorticity correction field $\delta\boldsymbol{\omega}$ on the obstacle, depending only on the velocity field and the body velocity. Two choices are possible in practice: compute the Poisson equation before or after the penalization step, and thus use respectively $\widetilde{\mathbf{u}}^{n+1}$ or $\mathbf{u}^n$ to compute $\delta\boldsymbol{\omega}$. We choose in this case to compute Poisson after the penalization step as done by Gillis [25]. Also, this method is said to be "direct", meaning that the vorticity update is done in one single operation. This is in contrast with iterative methods that compute this vorticity correction iteratively.

1.3.2 Stability constraint

The drawback of this method is that it is subjected to a strong stability constraint on $\lambda \Delta t$ due to the direct explicit time integrator approach. In 2D, an analytical expression has been derived by Gillis [25], showing that the eigenvalues are shifted on the real axis with the penalization term. However, it becomes rapidly tedious for arbitrary 3D cases. On top of that, this constraint limits the choice of λ and affects the accuracy of the solution because of Equation (1.3).

As a consequence, this method is often only considered for two-dimensional behaving flows, like flows past a 2D cylinder. In this work, we consider a cylinder with finite thickness. Making it three-dimensional will not affect the stability constraint, provided that the flow remains purely planar. Considering (1) a uniform grid with a mesh size h, (2) a fixed obstacle ($\mathbf{u}_b = 0$) and (3) a mask function χ independent of time, the stability constraint is reported by Ramussen et al. [8] to be $\lambda \Delta t \leq 2$ in 2D. In practice, two choices are possible to satisfy this constraint:

1. Fix the value of λ and determine the corresponding time step Δt accordingly. If this time step is smaller than the convective and advective ones, the advantage is to maintain the permeability of the object constant over time.

2. Choose the value of λ adaptively, depending on the time step Δt determined through the diffusive and convective stability conditions, or fixed by the user. We consider this option to avoid restricting to too small time steps, and we hope those time steps to be sufficiently small to model a solid obstacle.

The explicit direct penalization algorithm is reported in Algorithm 1, where Ω denotes the whole computational domain. The force on the object computation is added to the algorithm and is explained in Section 1.3.3. Finally, we consider in this algorithm an arbitrarily oriented streamwise velocity U_∞ in the z-direction.

Algorithm 1 Explicit penalization method

Initialization

Set preliminary velocity field $\mathbf{u}^0 = (0, 0, U_\infty)$ (1.9)

Set preliminary vorticity field $\boldsymbol{\omega}^0 = (0, 0, 0)$ (1.10)

Update: Given $\boldsymbol{\omega}^n$, $\mathbf{u}^n$ at time t, step n

Stretching + Advection + Diffusion $\boldsymbol{\omega}^n \rightarrow \widetilde{\boldsymbol{\omega}}^{n+1}$ (1.11)

Vorticity penalization $\boldsymbol{\omega}^{n+1} = \widetilde{\boldsymbol{\omega}}^{n+1} + \lambda \Delta t \, \nabla \times [\chi(\mathbf{u}_b^n - \mathbf{u}^n)]$ (1.12)

Penalized Poisson equation $\boldsymbol{\omega}^{n+1} \rightarrow \mathbf{u}^{n+1}$ (1.13)

Force computation $\boldsymbol{F}^{n+1} = -\rho \int_\Omega \chi \lambda (\mathbf{u}_b^{n+1} - \mathbf{u}^{n+1}) \, d\Omega$ (1.14)

1.3.3 Force evaluation

An important way to check for the validity of the implementation is to evaluate the forces that act on the obstacle and compare them with the literature. The assessment of aerodynamic forces is also extremely important in aerodynamic engineering practices. For the case of an explicit penalization implementation, we can compute the force acting on an obstacle as [26]

$$\boldsymbol{F} = -\rho \int_\Omega \chi \lambda (\mathbf{u}_b - \mathbf{u}) \, d\Omega \quad \Leftrightarrow \quad \boldsymbol{F}^{n+1} = -\rho \int_\Omega \chi \lambda (\mathbf{u}_b^{n+1} - \mathbf{u}^{n+1}) \, d\Omega \tag{1.15}$$

where ρ is the density of the medium (supposed constant) and Ω is the computational domain. This formula is however only valid for the explicit direct penalization method. Indeed, continuous forcing approaches consider the additional term as a force contribution in the "continuous" world, but this interpretation is only valid in the "discrete" world if the shape of the discretized expression is conserved. Also, this formula forces to work with a penalized velocity field, which is not ideal in vorticity-based formulations.

In other more advanced methods, we do not seek to directly solve this equation, but rather to find a correction of vorticity that is consistent with the flow. Therefore, other ways to compute the forces are used. Nevertheless, Equation (1.15) remains useful to asses the validity of the alternative forces computation implementations, since it is straightforward to implement. We also note that the value of ρ vanishes with the adimensionalization of the force term.

1.4 Implicit penalization technique

1.4.1 Discretization

To model accurately a solid obstacle, we need to set λ as high as possible (since it controls how effective the solid boundary condition is). As a consequence, the stability constraint of the explicit approach involves to use extremely small time steps to have accurate results. As a solution, an implicit implementation of the penalization term is widely used in the literature (for instance by Mimeau et al. [18]). As above, we use a splitting approach between the classical Navier-Stokes equations and the penalization term. A first idea could be to use the vorticity-velocity penalization formulation (1.6) and discretize it implicitly. However, it would give us two correlated unknowns $\mathbf{u}^{n+1}$ and $\boldsymbol{\omega}^{n+1}$, making it unusable. Let us thus consider the penalized Equation (1.1) in the velocity-pressure formulation

$$\frac{\partial \mathbf{u}}{\partial t} = \lambda\chi(\mathbf{u}_b - \mathbf{u}) \tag{1.16}$$

If we discretize the time derivative with a first order implicit Euler scheme between time steps n and $n+1$, we successively obtain

$$\frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} = \lambda\chi(\mathbf{u}_b^{n+1} - \mathbf{u}^{n+1}) \tag{1.17}$$

$$\mathbf{u}^{n+1} = \frac{\mathbf{u}^n + \lambda\chi\Delta t\mathbf{u}_b^{n+1}}{1 + \lambda\chi\Delta t} \tag{1.18}$$

$$\mathbf{u}^{n+1} = \mathbf{u}^n - \left[\frac{\lambda\chi\Delta t(\mathbf{u}^n - \mathbf{u}_b^{n+1})}{1 + \lambda\chi\Delta t}\right] \tag{1.19}$$

By taking the curl of this expression, we obtain a conservative expression for the vorticity update

$$\boldsymbol{\omega}^{n+1} = \boldsymbol{\omega}^n - \nabla \times \left[\frac{\lambda\chi\Delta t(\mathbf{u}^n - \mathbf{u}_b^{n+1})}{1 + \lambda\chi\Delta t}\right] \tag{1.20}$$

In practice, since we consider a splitting approach, the penalization step transforms an unpenalized to a penalized field, so that we compute the following expression

$$\boldsymbol{\omega}^{n+1} = \widetilde{\boldsymbol{\omega}}^{n+1} \underbrace{-\nabla \times \left[\frac{\lambda \chi \Delta t (\widetilde{\mathbf{u}}^{n+1} - \mathbf{u}_b^{n+1})}{1 + \lambda \chi \Delta t} \right]}_{\triangleq \delta\boldsymbol{\omega}} \tag{1.21}$$

The implicit penalization algorithm is reported in Algorithm 2. We first note that the velocity field is not directly penalized in this algorithm and is implicitly realized through the resolution of the Poisson problem. Also, the implicit Euler scheme is unconditionally stable. The value of λ can thus be chosen arbitrarily high so that we can model solid objects. However, as explained by Ramussen et al. [8] and Gillis [6], penalizing the vorticity in this way has two major drawbacks, that we will observe in practice in Chapter 2:

1. Significant diffusion errors occur, especially for high Reynolds numbers producing complex vortices. Indeed, the perfect vorticity correction (obtained with an implicit integration in the vorticity-velocity formulation) is given by

$$\boldsymbol{\omega}^{n+1} - \widetilde{\boldsymbol{\omega}}^{n+1} = \delta\boldsymbol{\omega} = \lambda \Delta t \nabla \times [\chi(\mathbf{u}_b^{n+1} - \mathbf{u}^{n+1})] \tag{1.22}$$

   Subtracting the vorticity corrections of Equations (1.21) and (1.22) leads to a non-zero error term that generates substantial diffusion.

2. There is no direct correlation established between the vorticity and the velocity, so that the vortex sheet is not completely accurate and unsteady behaviours are not properly captured.

1.4.2 Force evaluation

As already mentioned, we can not compute the force anymore using Equation (1.15), since the penalization term is not directly added to the Navier-Stokes equations. Alternative solutions exists. For example, Mimeau et al. [18] use the change of momentum equation and reinject the implicit Euler scheme in it. Alternatively, Noca et al. propose a robust momentum formulation [27]. In this work, we rather consider widely used method based on the the temporal change in the vorticity moment of the flow (see Wu et al. [28]). Considering $\mathbf{u}_b = 0$, we compute

$$\boldsymbol{F} = -\rho \frac{d}{dt} \int_\Omega \boldsymbol{x} \times \boldsymbol{\omega} \, d\Omega \quad \Leftrightarrow \quad \boldsymbol{F}^{n+1} = -\frac{1}{\Delta t} \rho \int_\Omega \boldsymbol{x} \times (\underbrace{\boldsymbol{\omega}^{n+1} - \widetilde{\boldsymbol{\omega}}^{n+1}}_{\delta\boldsymbol{\omega}}) \, d\Omega \tag{1.29}$$

Algorithm 2 Implicit penalization method

Initialization

$$\text{Set preliminary velocity field} \qquad \mathbf{u}^0 = (0, 0, U_\infty) \tag{1.23}$$

$$\text{Set preliminary vorticity field} \qquad \boldsymbol{\omega}^0 = (0, 0, 0) \tag{1.24}$$

Update: Given $\boldsymbol{\omega}^n$, $\mathbf{u}^n$ at time t, step n

$$\text{Stretching + Advection + Diffusion} \qquad \boldsymbol{\omega}^n \rightarrow \widetilde{\boldsymbol{\omega}}^{n+1} \tag{1.25}$$

$$\text{Unpenalized Poisson equation} \qquad \widetilde{\boldsymbol{\omega}}^{n+1} \rightarrow \widetilde{\mathbf{u}}^{n+1} \tag{1.26}$$

$$\text{Vorticity penalization} \qquad \boldsymbol{\omega}^{n+1} = \widetilde{\boldsymbol{\omega}}^{n+1} - \nabla \times \left[\frac{\lambda \chi \Delta t (\widetilde{\mathbf{u}}^{n+1} - \mathbf{u}_b^{n+1})}{1 + \lambda \chi \Delta t} \right] \tag{1.27}$$

$$\text{Force computation} \qquad \boldsymbol{F}^{n+1} = -\rho \frac{d}{dt} \int_\Omega \boldsymbol{x} \times \boldsymbol{\omega}^{n+1} \, d\Omega \tag{1.28}$$

where a low order temporal discretization is performed. We can thus compute the force acting on the obstacle exclusively with the vorticity correction of the flow. However, as mentioned by Gazzola et al. [22], to use the latter formulation, it is essential for the computational domain to be sufficiently large to guarantee the preservation of vorticity generated by the object.

1.5 Iterative penalization technique

1.5.1 Concept

To address the drawbacks mentioned for the direct penalization methods, Hejlesen et al. [7] introduced an iterative Brinkman penalization approach. The core concept is to calculate the vorticity field using the penalization term and then use it to determine an updated velocity field by solving the corresponding Poisson equation. This iterative approach ensures that the kinematic consistency of the flow is satisfied and connects the vorticity and velocity between each other to give the correct vortex sheet.

At a given time step $n + 1$, the unpenalized velocity field $\widetilde{\mathbf{u}}^{n+1}$ is computed through the Poisson equation. In the penalized region, $\widetilde{\mathbf{u}}^{n+1}$ corresponds thus not to $\mathbf{u}_b$. The idea is to compute iteratively the vorticity correction $\delta\boldsymbol{\omega}$ generating

a velocity correction $\delta\mathbf{u}$ so that $\tilde{\mathbf{u}}^{n+1} + \delta\mathbf{u} = \mathbf{u}^{n+1} = \mathbf{u}_b$ inside the body Ω_b. The iterative penalization approach is described in Algorithm 3.

This algorithm incorporates a new parameter, denoted as α, which serves as a relaxation parameter to relax the increment of the penalization vorticity. As a result, the penalization parameter $\lambda = \frac{\alpha}{\Delta t}$, which was previously used as a measure of porosity, now functions as a relaxation factor because of the iterative aspect. According to Hejlesen et al. [7], the stability constraint within this framework is reported to be

$$0 < \alpha = \lambda \Delta t \leq 2 \tag{1.30}$$

Finally, as mentioned by Gillis [6], this method can be reinterpreted as solving a linear system with a Jacobi method. Indeed, we can reformulate the problem as

$$\mathbf{A}(\delta\boldsymbol{\omega}) = \nabla \times [\chi(\mathbf{u}_b - \tilde{\mathbf{u}}^{n+1})] \tag{1.31}$$

where the three-dimensional linear operator $\mathbf{A}$ is defined at iteration k as (using Equation (1.5))

$$\mathbf{A} : \delta\boldsymbol{\omega}_k \rightarrow \nabla \times [\chi \delta \mathbf{u}_k] = \nabla \times \left[\chi \left(\left(\nabla^2 \right)^{-1} (-\nabla \times (\delta\boldsymbol{\omega}_k)) \right) \right] \tag{1.32}$$

According to Saad [29, p. 115, Th. 4.1], the Jacobi method converges, provided that the eigenvalues of the iterative matrix behave nicely (spectral radius $\rho(\mathbf{A}) < 1$). In our case, the method was observed by various authors [4, 6, 7] to converge. Additionally, by reformulating this problem as a linear system, we notice that some more advanced algorithmic techniques can be used to reduce the number of iterations of the penalization algorithm. Also, using the solutions of the previous time steps is of particular interest. Those are the ideas underlying Section 1.5.4, that mix recycled spaces and a BiCGStab algorithm to compute the penalization increment.

1.5.2 Convergence criterion

For the stopping criterion, we have different choices. A first idea could be to base it on the residual velocity, which according to Hejlesen et al. [7] is not a good choice for impulsively started flows. A stopping criteria based on the increment penalization force (first moment integral) is then widely used. In this work, we alternatively consider the convergence criterion, based on the change of enstrophy $\mathcal{E}$ at iteration k

$$E^k = \frac{|\mathcal{E}^k - \mathcal{E}^{k-1}|}{\mathcal{E}^k} < \epsilon \tag{1.33}$$

where ϵ is the required tolerance and where the enstrophy of $\delta\boldsymbol{\omega}^k$ is defined as

$$\mathcal{E}^k \triangleq \int_\Omega \delta\boldsymbol{\omega}^k \cdot \delta\boldsymbol{\omega}^k \, d\Omega \tag{1.45}$$

Hejlesen noted that this criterion has the advantage to serve as a norm-based measure and is therefore appropriate as a convergence criterion. We analyze the optimal stopping criterion with a convergence analysis in Chapter 2. The main compromise for all iterative methods is to find the right balance between computational cost and accuracy.

Algorithm 3 Iterative penalization method Jacobi(α), from Hejlesen et al. [7]

Initialization

Set preliminary velocity field $\mathbf{u}^0 = (0, 0, U_\infty)$ (1.34)

Set preliminary vorticity field $\boldsymbol{\omega}^0 = (0, 0, 0)$ (1.35)

Update: Given $\boldsymbol{\omega^n}$, $\mathbf{u}^n$ at time t, step n

Stretching + Advection + Diffusion $\widetilde{\boldsymbol{\omega}}^n \rightarrow \widetilde{\boldsymbol{\omega}}^{n+1}$ (1.36)

Unpenalized Poisson equation $\widetilde{\boldsymbol{\omega}}^{n+1} \rightarrow \widetilde{\mathbf{u}}^{n+1}$ (1.37)

while $E^k \geq \epsilon$ and $k \leq N_{it}$ **do**

Update increment vorticity $\delta\boldsymbol{\omega}^k = \delta\boldsymbol{\omega}^{k-1} + \alpha\nabla\times(\chi(\mathbf{u}_b^{n+1} - \widetilde{\mathbf{u}}^{n+1} - \delta\mathbf{u}^{k-1}))$ (1.38)

Poisson equation on increment vorticity $\delta\boldsymbol{\omega}^k \rightarrow \delta\mathbf{u}^k$ (1.39)

Compute enstrophy $\mathcal{E}^k = \int_\Omega \delta\boldsymbol{\omega}^k \cdot \delta\boldsymbol{\omega}^k \, d\Omega$ (1.40)

Compute residual $E^k = \dfrac{|\mathcal{E}^k - \mathcal{E}^{k-1}|}{\mathcal{E}^k}$ (1.41)

end while

Compute total vorticity field $\boldsymbol{\omega}^{n+1} = \widetilde{\boldsymbol{\omega}}^{n+1} + \delta\boldsymbol{\omega}$ (1.42)

Penalized Poisson equation $\boldsymbol{\omega}^{n+1} \rightarrow \mathbf{u}^{n+1}$ (1.43)

Force computation $\boldsymbol{F}^{n+1} = -\rho\dfrac{d}{dt}\displaystyle\int_\Omega \boldsymbol{x} \times \delta\boldsymbol{\omega} \, d\Omega$ (1.44)

1.5.3 Force evaluation

What remains to be clarified is the way to compute the aerodynamic forces acting on the obstacle. As in Section 1.4.2, we can consider the momentum of vorticity expression on a fixed body. This is what is done for instance in the work Hejlesen et al. [7] and Spietz et al. [4]. In this framework, we thus compute at each time iteration

$$\mathbf{F} = -\rho \frac{d}{dt} \int_\Omega \boldsymbol{x} \times \delta\boldsymbol{\omega} \, d\Omega \qquad \Leftrightarrow \qquad \mathbf{F}^{n+1} = -\rho \frac{1}{\Delta t} \int_\Omega \boldsymbol{x} \times \delta\boldsymbol{\omega} \, d\Omega \tag{1.46}$$

We note that unlike for the implicit approach, we do not need to compute explicitly $\delta\boldsymbol{\omega}$ since it is already computed through the iterative algorithm.

1.5.4 Krylov-based iterative penalization

The iterative method proposed by Gillis et al. [9] aims to reduce the number of iterations required by the algorithm. In this section, we provide a brief theoretical overview of the so-called rBiCGStab(l) algorithm. Even though it is not yet implemented in practice in *Murphy*, the goal of this section is to have a general idea of the improvements of the classical Jacobi(α) penalization method. In brief, the rBiCGStab(l) algorithm is a recycling method that first computes the best solution within a known space of dimension l and then solves the remaining part of the system with the so-called BiCGStab algorithm.

Let us consider an arbitrary linear system $\mathbf{Ax} = \mathbf{b}$ like the one of Equation (1.31), with $\mathbf{A} \in \mathbb{R}^{n\times n}$. We consider its input and output subspaces $\mathbf{A} : \mathcal{U} \to \mathcal{C}$, and two arbitrary bases spanning those subspaces, $\mathbf{U}$ and $\mathbf{C}$ both in $\mathbb{R}^{n\times l}$. Considering an initial solution $\mathbf{x}_{-1}$ and its corresponding residual $\mathbf{r}_{-1} \triangleq \mathbf{Ax}_{-1} - \mathbf{b}$, we search for the best guesses $\mathbf{x}_0$ and $\mathbf{r}_0$. To do so, Krylov recycling methods consider an initial guess of the form $\mathbf{x}_0 = \mathbf{x}_{-1} + \mathbf{Uy}$, with $\mathbf{y} \in \mathbb{R}^l$, verifying [9]

$$\mathbf{x}_0 = \arg\min_{\mathbf{y}} ||\mathbf{r}_0|| \quad \text{with the constraints} \quad \mathbf{AU} = \mathbf{C} \quad \text{and} \quad \mathbf{C}^T\mathbf{C} = \mathbf{I} \tag{1.47}$$

According to Parks et al. [30], the optimal solution and its residual are given by

$$\mathbf{x}_0 = \mathbf{x}_{-1} + \mathbf{UC}^T\mathbf{r}_{-1} \qquad\qquad \mathbf{r}_0 = \mathbf{r}_{-1} - \mathbf{CC}^T\mathbf{r}_{-1} \tag{1.48}$$

Then, considering $\mathbf{x} \triangleq \mathbf{x}_0 + \boldsymbol{\zeta}$ and reminding that $\mathbf{r}_0 = \mathbf{b} - \mathbf{Ax}_0$, it remains to solve

$$\mathbf{A}\boldsymbol{\zeta} = \mathbf{r}_0 \tag{1.49}$$

that we do with well-know algorithms. Gillis et al. considered the biconjugate gradient stabilized method, commonly abbreviated as BiCGStab. This iterative

technique was developed by Van der Vorst [31] and is a variant of the biconjugate gradient method (BiCG). BiCGStab is known to have faster and smoother convergence when compared to the original BiCG as well as other variants such as the Conjugate Gradient Squared method (CGS). Also, this algorithm tends to converge faster than the Jacobi method for most types of linear systems. The Jacobi method converges slowly for large and sparse systems, especially when the coefficient matrix is ill-conditioned or contains dominant diagonal elements. In contrast, BiCGStab is more robust and can handle a wider range of system conditions. On top if that, a preconditionner is often used to accelerate the convergence of the iterative method.

During each iteration, BiCGStab employs a combination of two iterative techniques: Bi-Conjugate Gradient (Bi-CG) and Stabilized Iterative Refinement. The Bi-CG method is used to generate search directions, while the Stabilized Iterative Refinement technique improves the stability and convergence properties of the algorithm. At each iteration, BiCGStab calculates two vectors: the search direction and a second vector based on the residual. These vectors are used to update the solution iteratively. The algorithm dynamically adjusts the weights of these vectors to ensure convergence and stability, thereby enhancing the efficiency of the solution process.

According to Gillis et al. [9], the rBiCGStab(l) algorithm has been found to have two drawbacks. Firstly, in contrast to the Jacobi(α) method, it involves performing two matrix-vector products at each iteration, resulting in a higher computational cost per iteration. However, it is important to note that the rBiCGStab(l) algorithm typically requires fewer iterations to achieve convergence compared to the Jacobi method. Secondly, unlike methods such as GMRes, the residual does not exhibit monotonic convergence behavior. This non-monotonic behavior is a well-known characteristic of the algorithm and is attributed to the use of bi-orthogonalization techniques.

As a summary of this section, we briefly observed that a range of solvers could be used for solving iteratively penalization methods and the Jacobi method is one of them. The BiCGStab seems to be a better choice for the faster convergence properties it provides. Also, the solutions are not expected to change much from one time step to the other in penalization methods. Therefore, we can consider the l previous times steps solutions to have a more accurate initial guess for the iterative problem. Gillis et al. deduced $l = 2$ to be an optimal choice. This is the idea that lies behind Krylov recycling methods.

CHAPTER 2

Numerical results and comparisons

Contents

2.1 Introduction

In this chapter, we test and compare the described penalization techniques for different flows and obstacles. Our tests are performed with fully Eulerian DNS simulations on uniform Cartesian grids.

Firstly, impulsively started flows past a cylinder are analyzed. We compare the different penalization methods, their aerodynamic forces and analyze the convergence and precision of our simulations. We outline the advantages of using an iterative penalization method. Then, a purely three-dimensional case is tested with a flow past a sphere. An impulsively started flow and a long time simulation are considered. Finally, we analyze flows past extruded airfoils to cover more complex geometries. The flow behaviours of a regularized Joukowski airfoil and a NACA0012 airfoil are compared. For technical details about the technical implementations in *Murphy*, the reader is referred to Appendix B.

Throughout this chapter, we consider the following dimensionless variables

$$\mathbf{u}^* = \frac{\mathbf{u}}{U_\infty} \qquad \boldsymbol{\omega}^* = \frac{\boldsymbol{\omega} L}{U_\infty} \qquad t^* = \frac{t U_\infty}{L} \qquad Re = \frac{U_\infty L}{\nu} = \frac{\text{inertial forces}}{\text{viscous forces}} \tag{2.1}$$

where U_∞ denotes the free stream velocity (in the z-direction in this work), L is the characteristic length traveled by the fluid and $\nu = \frac{\mu}{\rho}$ is the kinematic viscosity of the fluid. We consider the following reference lengths : $L = D = 2R$ (diameter) for the cylinder and the sphere, and $L = c$ (chord length) for the airfoils. We note that in our case, the single degree of freedom of the flow is the Reynolds number.

2.2 Numerical setups

The first thing to do is to fix properly our computational setup. We consider flows with an arbitrary uniform upstream velocity $\mathbf{u}_\infty = (0, 0, 5)$. Also, all obstacles are fixed ($\mathbf{u}_b = 0$) for simplicity. Nevertheless, the technical implementation in *Murphy* takes into account body velocities and could be the object of further investigations (the general expressions considering body velocities are provided in Chapter 1).

The computational domains Ω are chosen depending on the purpose of the simulation (short time or long time) and on the object. We cover them with uniform Cartesian grids. We note that, unlike what is done by Mimeau et al. [20] and Cottet and Poncet [32] that considered fully periodic boundary conditions, we consider unbounded boundary conditions, such that the domains can be reduced

consequently. Considering the geometric center of the objects to be the origin, we set the following computational domains:

1. For the circular cylinder : $\Omega_{\text{yz}} = [-1.25D, 1.25D] \times [-1.25D, 3D]$, where D is the diameter of the cylinder. The domain is sufficiently large to capture the sort time vortices for $Re = 550$ and $Re = 9500$.

2. For the sphere : $\Omega_{xyz} = [-1.5D, 1.5D] \times [-1.5D, 1.5D] \times [-2D, 11D]$, where D is the diameter of the sphere. The streamwise dimension is chosen similarly to Mimeau et al. [18] and Ploumhans et al. [33].

3. For the airfoils : $\Omega_{\text{yz}} = [-c, c] \times [-c, 2.5c]$, where c is the chord length of the airfoil. The domain is big enough to capture the short time leading and trailing edges vortices produced by the airfoil.

Those domains and their corresponding mask functions χ are shown on Figure 2.1. Since the Joukowski airfoil is defined as a conformal mapping from a circle, its inverse Joukowski transform is also represented. Technical details regarding this transform are provided in Appendix A. Also, all "two-dimensional" like objects have been chosen with a fixed thickness of either 24 or 32 points in the x-direction. We choose such numbers because of the block-structured conception of *Murphy*, that forces to consider the same number of points per block in each direction.

Concerning the different chosen mesh sizes, it is important to use a sufficiently fine grids to capture properly the boundary layer around the obstacles. Since the non-dimensional boundary layer thickness δ^* depends on the object and on the Reynolds number considered, we precise it in the corresponding subsections.

We also fix properly the boundary conditions. For the cylinder and the airfoils, we fix a periodic boundary condition in the direction of the axis of the object. For the other directions, we set them as unbounded, except for the streamwise (z) direction, where we impose an unbounded-outflow boundary condition. The reason is that we want consistent values for our ghost points, that will model properly the flow behaviour outside of the domain. An unbounded boundary condition on the z+ face would kill the vorticity at the boundary and lead to a non-physical behaviour close to it. Let n and t denote respectively the normal and tangent directions of the z-plane. For the outflow condition, we set

$$\frac{\partial \mathbf{u}_n}{\partial n} = 0 \quad \text{(Even B.C)} \qquad \boldsymbol{\omega}_n = 0 \quad \text{(Odd B.C)} \tag{2.2}$$

$$\mathbf{u}_t = 0 \quad \text{(Odd B.C)} \qquad \frac{\partial \boldsymbol{\omega}_t}{\partial n} = 0 \quad \text{(Even B.C)} \tag{2.3}$$

Lastly, we precise the time and space discretizations of the different operators. For the time discretization, we consider an RK3-TVD scheme. According to Gottlieb et al. [23, Prop. 3.2], the optimal third-order Total Variation Diminushing (TVD) Runge-Kutta method is given by

$$\begin{cases} \mathbf{u}^{(1)} &= \mathbf{u}^n + \Delta t L(\mathbf{u}^n) \\ \mathbf{u}^{(2)} &= \frac{3}{4}\mathbf{u}^n + \frac{1}{4}\mathbf{u}^{(1)} + \frac{1}{4}\Delta t L(\mathbf{u}^{(1)}) \\ \mathbf{u}^{n+1} &= \frac{1}{3}\mathbf{u}^n + \frac{2}{3}\mathbf{u}^{(2)} + \frac{2}{3}\Delta t L(\mathbf{u}^{(2)}) \end{cases} \tag{2.4}$$

where the operator $L(\mathbf{u})$ denotes the right hand-side of the PDE and $\mathbf{u}$ is an arbitrary field, being the vorticity in our case. To justify this choice, we define the

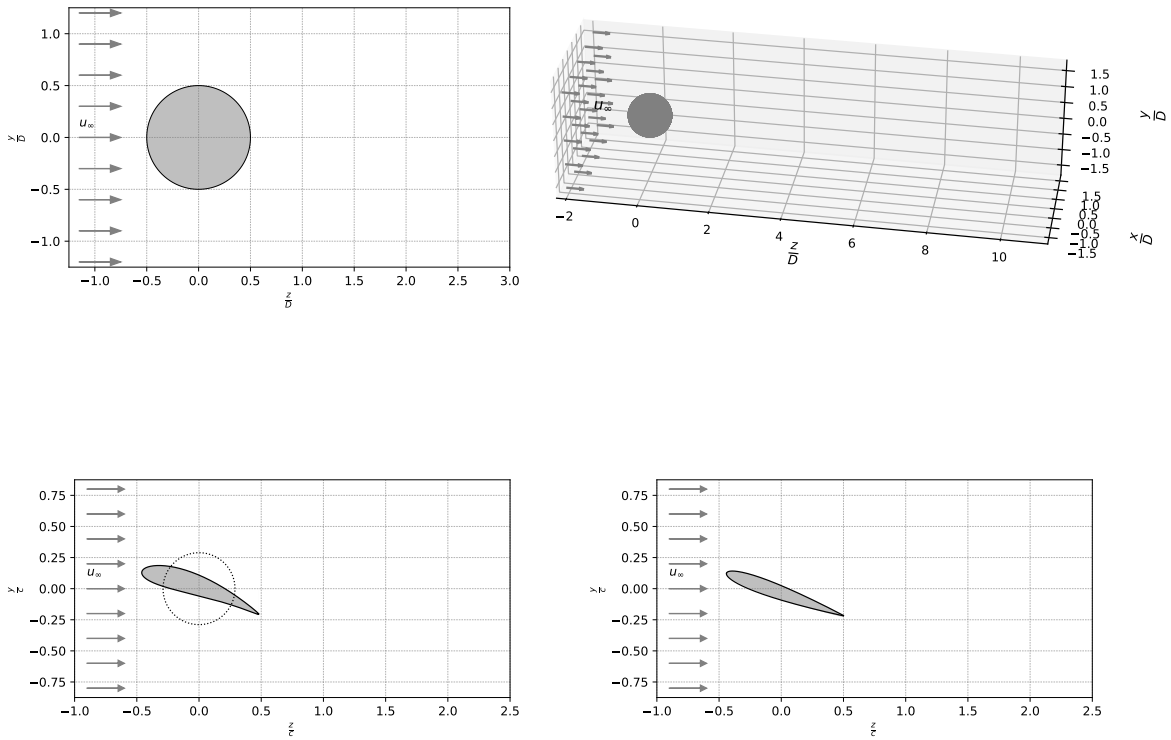

Figure 2.1: Computational domains and their corresponding mask functions χ. Upper left: circular cylinder case (slice). Upper right: sphere case. Lower left: Joukowski airfoil section and its corresponding inverse Joukowski transform. Lower right: NACA0012 airfoil section. Both airfoils here have an angle of attack (AoA) $\alpha = 20°$. The uniform upstream velocity is also represented.

total variation of the i^{th} component of a numerical solution as

$$TV(\mathbf{u}) \triangleq \sum_j |\mathbf{u}_{i,j+1} - \mathbf{u}_{i,j}| \tag{2.5}$$

A TVD scheme has the nice property that the total variation of the solution does not increase as we progress in time

$$TV(\mathbf{u}^{n+1}) \leq TV(\mathbf{u}^n) \qquad \forall n \in \mathbb{N} \tag{2.6}$$

where the "$\leq$" sign has to be understood component-wise. This ability to maintain the total variation of the flow is interesting, because it allows to preserve the sharpness of the discontinuities in the solution, while smoothing out other regions where the solution varies more slowly. On top of that, this scheme can capture sharp shock predictions and avoid spurious oscillations generated by a discontinuous field. This is of interest in our case since the mask function χ is a discontinuous Heaviside step function.

Regarding the spatial discretization of the different operators, we consider 2^{nd} order centered finite differences. As stated in Section 1.2.1, we adopt a regularized 10^{th} order solver for the Poisson equation, which is proposed by Hejlesen et al. [17]. This choice ensures the generation of adequately smooth fields using a Heaviside mask function, as mentioned by Spietz et al. [4].

2.3 Aerodynamic coefficients

We also define properly the dimensionless quantities related to the aerodynamic forces acting on the objects. Adimensionalizing the different computed forces is mandatory to compare dynamically similar flows and results between each other. The way those forces are computed is mentioned in Chapter 1. For a 3D flow with a streamwise velocity in the z-direction, the drag and lift coefficients are defined as

$$C_D = \frac{\mathbf{F} \cdot \mathbf{e}_z}{q_\infty \mathcal{A}} \qquad C_L = \frac{\mathbf{F} \cdot \mathbf{e}_x}{q_\infty \mathcal{A}} \qquad C_S = \frac{\mathbf{F} \cdot \mathbf{e}_y}{q_\infty \mathcal{A}} \tag{2.7}$$

where C_D is the drag coefficient and where the lift coefficient is divided in C_L (vertical lift) and C_S (side lift). Also, $q_\infty \triangleq \frac{1}{2}\rho U_\infty^2$ denotes the dynamic pressure and $\mathcal{A}$ is the reference area of the object. For the case of a 3D cylinder, $\mathcal{A} = HD$, where D is the diameter of the cylinder and H its height. For the sphere, we consider $\mathcal{A} = \pi\frac{D^2}{4}$. For an extruded airfoil in 3D, $\mathcal{A} = cH$, where c is the chord length and H the length of extrusion of the airfoil. A general description of the nomenclature of an airfoil is provided in Appendix A.

2.4 Impulsively started flow past a cylinder

We first consider impulsively started flows past a circular cylinder. Two Reynolds number are analyzed: $Re = 550$ and $Re = 9500$. The first one is a moderate Reynolds number and the second one is turbulent and characterized by complex vortex pairings in the wake. It is often considered as a highly transitional flow in the literature [20].

For a flow past a circular cylinder, an analytical solution has been derived by Barlev et al. [34] for short times ($t^* \lesssim 0.25$). It consists of the sum of the pressure drag and the skin friction drag on the cylinder

$$C_D = \underbrace{4(\frac{\pi}{Re\, t^*})^{1/2}}_{\text{skin friction drag}} + \underbrace{2\pi(9 - \frac{15}{\sqrt{\pi}})\frac{1}{Re}}_{\text{pressure drag}} \tag{2.8}$$

For larger times, we refer to already existing computational results. For the drag analysis, we refer to the work of Billuart et al. [35], Koumoutsakos et al. [36] and Hejlesen et al. [7]. For the vorticity contours analysis, we refer to the most accurate results, that are the ones of Billuart et al. Finally, we need to precise our uniform grid resolution denoted h, and our time step choice Δt. For a flow past a cylinder, the boundary layer thickness near the stagnation point of the flow is approximately given by [6, 20]

$$\delta \approx \frac{D}{\sqrt{Re}} \Leftrightarrow \delta^* \approx \frac{1}{\sqrt{Re}} \tag{2.9}$$

The mesh should be sufficiently fine to have enough points in the boundary layer. Also, the higher the Reynolds number, the finer the resolution must be to capture properly the boundary layer. The different parameters of the simulations are provided in Table 2.1. The sufficient number of cell layers in the boundary layer (denoted N_δ) justifies the mesh choice. The mesh resolution at $Re = 550$ is similar to Mimeau et al. [19], while the one at $Re = 9500$ is similar to Hejlesen et al. [7].

Re	$CFL_{\max}$	$R_{\max}$	$h^* = \frac{h}{D}$	$\delta^* = \frac{\delta}{D}$	N_δ
550	0.6	0.1	1/192	≈ 0.042	8.06
9500	0.6	0.1	1/576	≈ 0.010	5.76

Table 2.1: Flows past a circular cylinder. Choice of the grid space h^* based on the corresponding boundary layer thickness δ^* for the two Reynolds numbers considered.

Concerning the choice of the adaptive time step Δt^*, different time resolutions are provided in the literature. However, Lagrangian methods are often used to handle the convection term, such that there is no constraint on the CFL (Courant–Friedrichs–Lewy) number. Defining $u_{\max}$ as the maximal velocity magnitude of the flow, the maximal CFL number is defined as

$$CFL_{\max} = \frac{u_{\max}\Delta t}{h} \tag{2.10}$$

In this work, we consider a fixed $CFL_{\max} = 0.6$. From this expression, the advective time step can be determined. Also, a maximum Fourier condition due to the viscous term is imposed. We define the maximal Fourier number $R_{\max}$ as

$$R_{\max} = \frac{\nu\Delta t}{h^2} \tag{2.11}$$

In the case of third order Runge-Kutta methods, we usually set $R_{\max} = 0.1$. In practice, we observe in the case of penalization methods that the CFL number imposes a stronger restriction on the time step (hence the motivation to use Lagrangian methods).

2.4.1 Flow at Re=550

We first consider a flow at a moderate Reynolds number $Re = 550$. In this flow regime, periodic vortex sheddings appear after a sufficiently large time, but we only consider here $t^* \leq 5$, so that the flow is still planar, symmetric and stable, and our computational setup is coherent.

Concerning the different parameters used for the simulation, we set $\lambda\Delta t = 1$ for the explicit method and $\lambda = 10^8$ for the implicit method. For the iterative method, we use a relaxation parameter similar to the one used by Gillis [6], that is $\alpha = 1.9$ (over-relaxed Jacobi iterations). For the stopping criterion, we consider a relative stopping criterion based on the enstrophy $\mathcal{E}$ of the flow. We consider a default stopping criterion $\epsilon = 0.05$ (justified later on). Finally, we force the first time step to be $\Delta t^* = 10^{-6}$ to avoid instabilities due a too strong flow variation because of the sudden presence of the obstacle. The drag coefficients of the different methods are depicted on Figure 2.2 for short times and on Figure 2.3 for longer times.

We observe that our results compare favorably with the theoretical curve for short times. For $t^* = 0^+$, the drag seems to fit more quickly in the case of the iterative method, but all three results converge after $t^* \approx 0.05$. There is a slight shift with the theoretical curve for $t^* \geq 0.05$, that is also in observed in the results of Gillis [6]. We also note that the implicit method exhibits a slightly lower accuracy

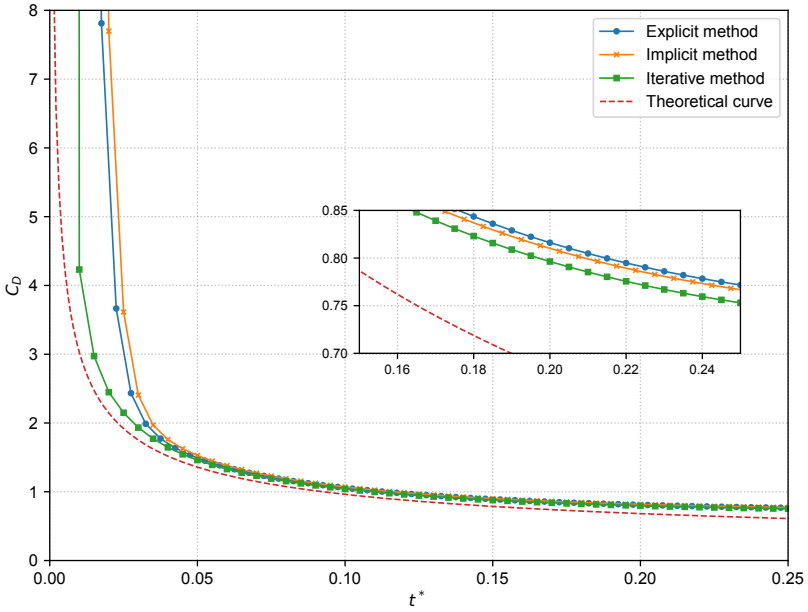

Figure 2.2: Impulsively started cylinder at $Re = 550$. Short time comparison with the theoretical curve of Barlev et al. [34]. The first time step is $\Delta t^* = 10^{-6}$.

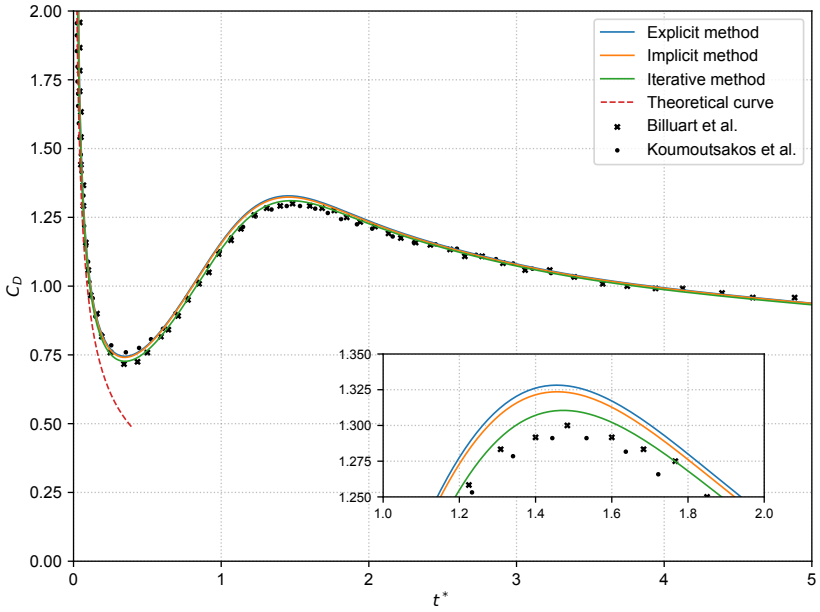

Figure 2.3: Impulsively started cylinder at $Re = 550$. Long time comparison with Koumoutsakos et al. [36] and Billuart et al. [35]. The first time step is $\Delta t^* = 10^{-6}$.

compared to the explicit method for $t^* \leq 0.05$, but it significantly improves beyond that threshold. Finally, the iterative method shows the highest accuracy among the three methods for short times, as it is closer to the theoretical curve.

For larger times, our results fit correctly the curves of Billuart et al. [35] and Koumoutsakos et al. [36]. The iterative method fits slightly better, especially at the two local extrema of the drag curve, as observed on the zoom on Figure 2.3. This overshoot at $t^* \approx 1.5$ of the direct methods is also observed by Mimeau et al. [19], that used an implicit approach. It is also worth mentioning that there is a clear difference in the value of λ of the two direct methods. Since the explicit method forces $\lambda \Delta t = 1$, the value of λ is in practice observed to be in $\mathcal{O}(10^3)$. We observe thus a slightly less accurate result for the explicit method, because the object is not fully solid. We will investigate it later in more details by doing a λ-convergence analysis. Finally, the drag computation using Equation (1.15) or (1.29) are observed to be equivalent for the explicit method, which confirms the validity of the implementation. The discrepancy is of approximately 1% at all times of the simulation, which is close to the one observed by Gazzola et al. [22].

Flow analysis

We can also analyze the different methods by comparing the vorticity contours at different times. The comparison is provided on Figure 2.4 for $t^* = 4$. The vorticity contours and the saturation are similar to Billuart et al. [35] to make consistent comparisons.

We observe that there is no significant difference between the three methods. The location and shape of the vortical structures that are generated on the cylinder boundary for the Hejlesen method and the direct methods at $t^* = 4$ seems to be the same. The explicit method has however slightly different contours than the two other methods. Also, the vorticity at the extremity of the sheddings of the reference result is not captured by the penalization methods, and the intensity of the heart of the vortical structure seems slightly smaller in the reference solution.

Finally, we observe in practice that there is a velocity residual remaining inside of the cylinder for the three methods. This residual can be further be analyzed by extracting velocity magnitude slices perpendicularly to the flow, and crossing the center of the cylinder. We choose to do it at the arbitrary time $t^* = 3$. The results are shown on Figures 2.5. We can observe that the iterative method has the lowest velocity residual in the body, followed by the implicit and the explicit methods. For instance, the velocity residual at the center of the cylinder is three times higher for the explicit method than for the iterative one. Also, small oscillations close to

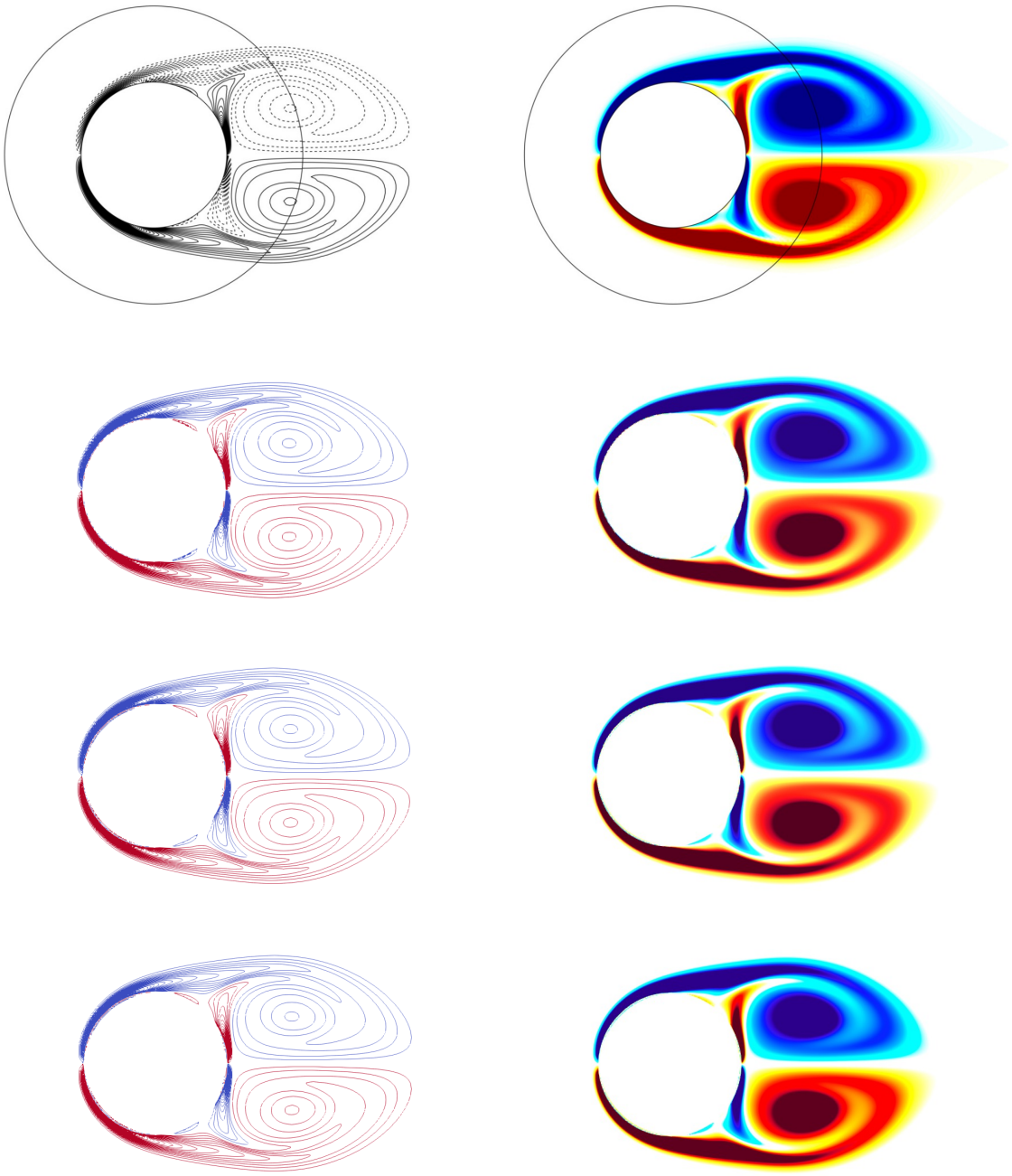

Figure 2.4: Impulsively-started cylinder at Re = 550: contours of dimensionless vorticity at $t^* \simeq 4$. Contour levels are by steps of 1.5. The color plot is saturated for $|\omega_x^*| > 6$. First row: Billuart et al. [35] at $t^* \simeq 4$. The circle represents the boundary between his Near-Wall (NW) region and his Vortex Particle-Mesh (VPM) region. Second row: Explicit approach with $\lambda \Delta t = 1$. Third row: Implicit approach with $\lambda = 10^8$. Fourth row: Iterative approach with $\alpha = 1.9$ and $\epsilon = 0.05$.

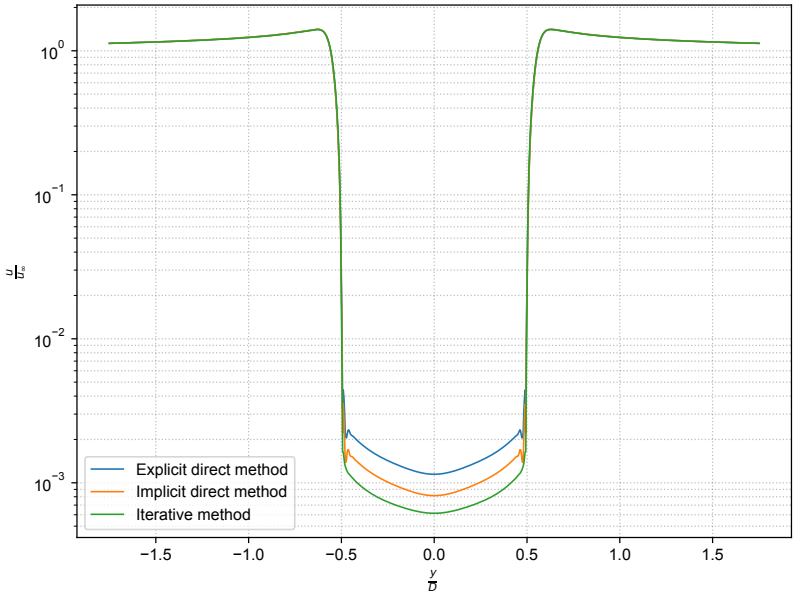

Figure 2.5: Mean velocity magnitude profiles u^* at $t^* \simeq 3$ in a section across the center of the cylinder at $\frac{x}{D} = 0$. The object is located between $\frac{y}{D} = -0.5$ and $\frac{y}{D} = 0.5$.

the boundary of the object are observed for the two direct methods, and are most likely due to the inconsistency of the vorticity-velocity pair. Overall, this velocity residual impacts the flow behaviour on a small scale, but impacts the drag results on a more significant scale, as seen on Figure 2.3 with the drag over/undershoot of the direct methods. There is thus a motivation in considering either the implicit method for its lower computational time, or the iterative one for its higher accuracy. We analyze the associated computational costs into more details in Section 2.7.

λ convergence study

In this section, we perform a convergence analysis on the parameter λ, as done by Mimeau et al. [19]. We check for the impact of this parameter upon the drag coefficient of the cylinder. We perform it on the implicit method. The accuracy of the explicit method can be directly found depending on the value of λ imposed by the stability constraint.

Firstly, we describe the methodology used for this study as well as the computed error. In this work, we only analyze the error using the drag coefficient, even though some convergence studies also exist with the enstrophy or with the velocity

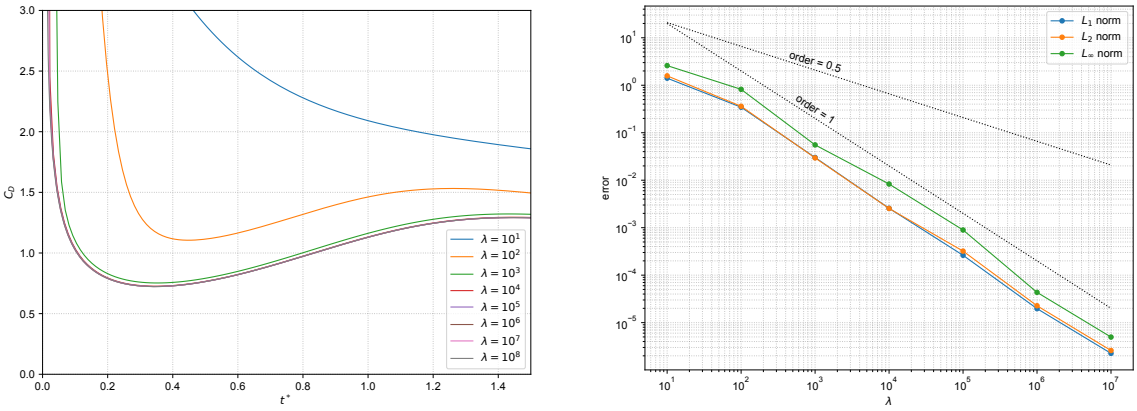

Figure 2.6: λ convergence study of the implicit direct method for an impulsively started flow past a cylinder at $Re = 550$. Left: drag coefficient C_D evolution. Right: Convergence analysis of λ. The reference case is set to $\lambda = 10^9$.

field directly. We evaluate the relative error E_{rel} of the L_1, L_2 and L_∞ norms of the drag coefficient with respect to a reference case

$$E_{\text{rel}} \triangleq \frac{||C_D(t^*) - C_{D\text{best resolved}}(t^*)||}{||C_{D\text{best resolved}}(t^*)||} \tag{2.12}$$

where the norm is either taken as the 1,2 or ∞ norm and where the reference case is in this framework taken with a sufficiently high λ (we choose $\lambda = 10^9$). The results and the convergence orders obtained are shown on Figure 2.6.

We observe that there is a strong impact of the λ parameter on the accuracy of the results. The error is very significant for $\lambda \leq 10^3$. The error seems to decrease as $\mathcal{O}(\lambda^{-1/2})$ for $\lambda \leq 10^2$ and as $\mathcal{O}(\lambda^{-1})$ for $\lambda \geq 10^2$. This is similar to the result of Mimeau et al. [19]. According to this result, there is a strong motivation in considering a high value of λ to provide accurate results, hence the motivation behind the implicit method. Since λ is physically correlated to the permeability of the medium, we consider here that a value of $\lambda \geq 10^4$ is sufficient to reach a solid state of the obstacle. For this reason, the explicit direct method will be set aside for the rest of this work, since it is redundant with the implicit case and has a too small value of λ, leading to less accurate results. This is indeed what we observed with the drag coefficient analysis and with the flow analysis.

Grid convergence study

In this section, we perform a grid convergence analysis for a flow past a circular cylinder at $Re = 550$. All three schemes are expected to provide the same

convergence rate (because the space discretizations are all of the same order), so we only provide the iterative one here ($\epsilon = 0.05$) as an example. The convergence analysis is performed on the short term drag coefficient (from $t^* = 0.3$ to $t^* = 0.4$) and the error norms are taken as in Equation (2.12). This time range is sufficiently far from the beginning of the simulation to provide accurate and reliable results. We consider a wide range of uniform grid steps and the reference case is set to $h^* \triangleq \frac{h}{D} = \frac{1}{384}$. The grid resolutions range from $h^* = \frac{1}{24}$ to $h^* = \frac{1}{192}$. We remind that all simulations use a $CFL_{\max} = 0.6$, so that the finest simulations use smaller time steps, on top of having more points. Also, we interpolate the drag from the fine mesh result to the coarse one, since the coarsest meshes have the least drag points because their time steps are bigger. The results are shown on Figure 2.7.

We observe that the order is around 2 for the three norms, which is in good agreement with the space discretization of order 2 we used. For the mesh choice, we keep in mind that a balance between computational cost and accuracy needs to be found. The choice of $h^* = \frac{1}{192}$ seems to be a good compromise here, since it produces a relatively small error and a reasonable time computational time (see Section 2.7). Finally, according to the grid convergence analysis of Spietz et al. [4], the rate of convergence does not appears to increase by applying fourth order finite difference schemes rather than second order schemes. It is observed that the convergence order is bounded because of the continuity of the vorticity field and of its derivatives at the boundary of the object. We leave this as a subject of further investigations.

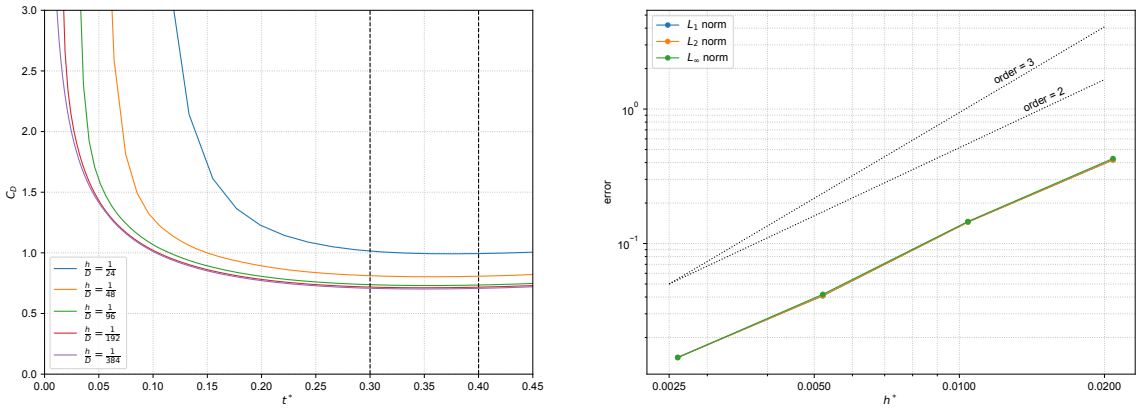

Figure 2.7: Grid-convergence study on the drag coefficient for flow past a circular cylinder at Re = 550 for the iterative method. Left: drag coefficient C_D evolution. The section on which the analysis is done is between the black dashed lines. Right: grid convergence analysis of the problem. The reference mesh is $h^* = \frac{1}{384}$.

2.4.2 Flow at Re=9500

We observed that all three methods provide a reliable solution of a impulsively started flow at a moderate Reynolds number, but there is a slightly less accurate result for direct methods, particularly for the explicit one. In this section, we consider a higher Reynolds number and check for the results. We still consider $\alpha = 1.9$ and $\epsilon = 0.05$. We remind that increasing the Reynolds number leads to finer mesh requirements as mentioned in Table 2.1. The resulting drag are depicted on Figure 2.8 for short times and on Figure 2.9 for longer times.

In the sort term, both methods are still accurate. The observations are similar to the ones at $Re = 550$. However, the non-iterative method produces an underestimation of the drag with the chosen time step for longer times ($t^* \geq 1$). More precisely, the oscillations for higher times seem to not be properly captured with the implicit method. This is observed by Hejlesen et al. [7] as well. Also, the results are closer form Hejlesen because the same approach and mesh resolution are used. On the other side, Billuart [35] considers a weak coupling between a near-wall Eulerian solver and a Vortex Particle-Mesh method. It allows for bigger time steps since it uses vortex methods, and a much higher resolution around the object, which is possible because a coupling framework is used.

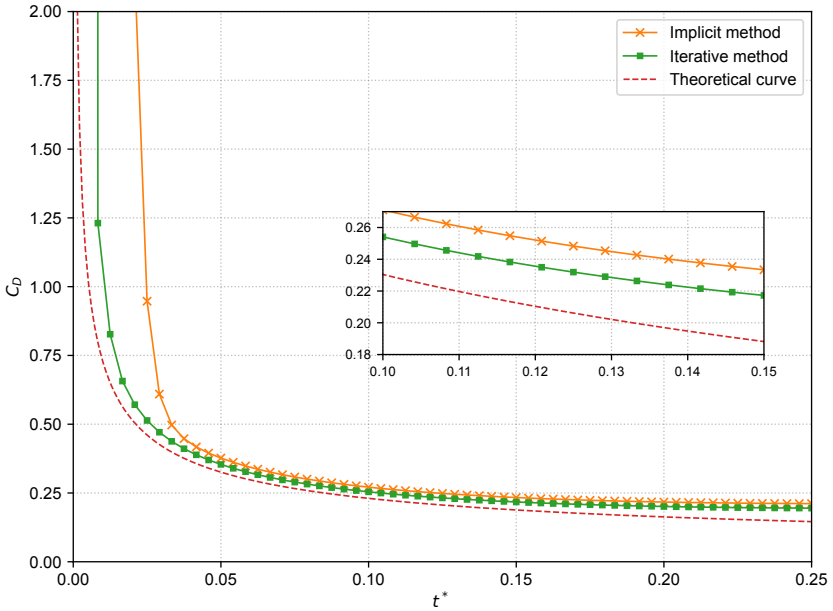

Figure 2.8: Impulsively started cylinder at $Re = 9500$. Short time comparison with the theoretical curve of Barlev et al. [34]. The first time step is $\Delta t^* = 10^{-6}$.

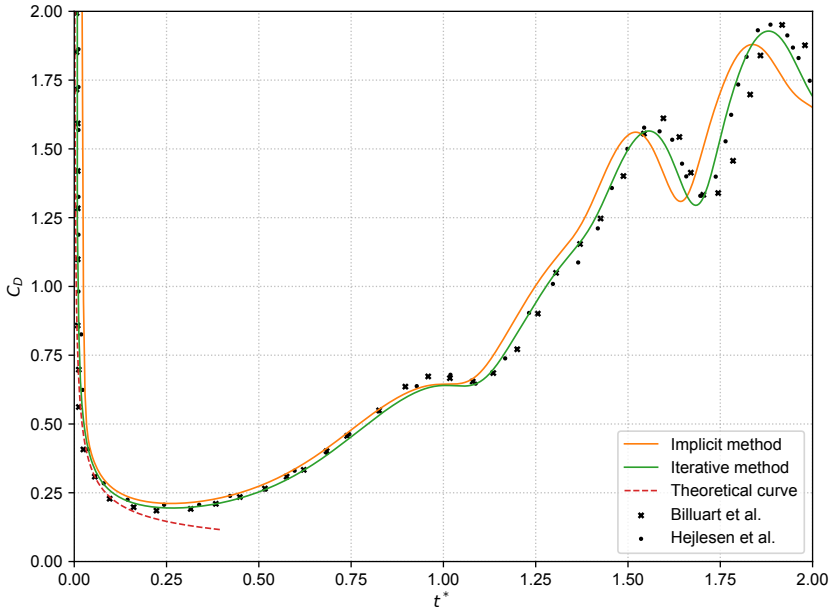

Figure 2.9: Impulsively started cylinder at $Re = 9500$. Long time comparison with Hejlesen et al. [7] and Billuart et al. [35]. The first time step is $\Delta t^* = 10^{-6}$.

Flow analysis

As for the case $Re = 550$, we also compare the vorticity contours with those found in the literature. We still consider Billuart et al. [35] as reference. The results are shown on Figure 2.10 for the implicit and the iterative methods.

The results are here different than for $Re = 550$. Overall, the two penalization results are not the same as the result of Billuart et al. However, the vortices are better oriented and located in the case of the iterative method and a clear difference is observed for the location and shape of the vortical structures that are generated on the cylinder boundary. The result is quite different than for a moderate Reynolds number because the turbulence becomes more dominant for $Re = 9500$, and the diffusion error introduced by a direct penalization (see Section 1.4.1) affects the flow dynamics. This can lead to inaccuracies and limitations in predicting the behavior of turbulent flows. On the other hand, by iteratively solving the penalization term, such methods can better capture the complex and evolving nature of turbulent flows, because the vortex sheet is correct. The iterative nature allows for a more accurate representation of the flow dynamics, including the formation and dissipation of turbulent structures. Finally, one could want to consider a finer mesh to get closer to the results of Billuart et al. [35]. However, it becomes then almost mandatory

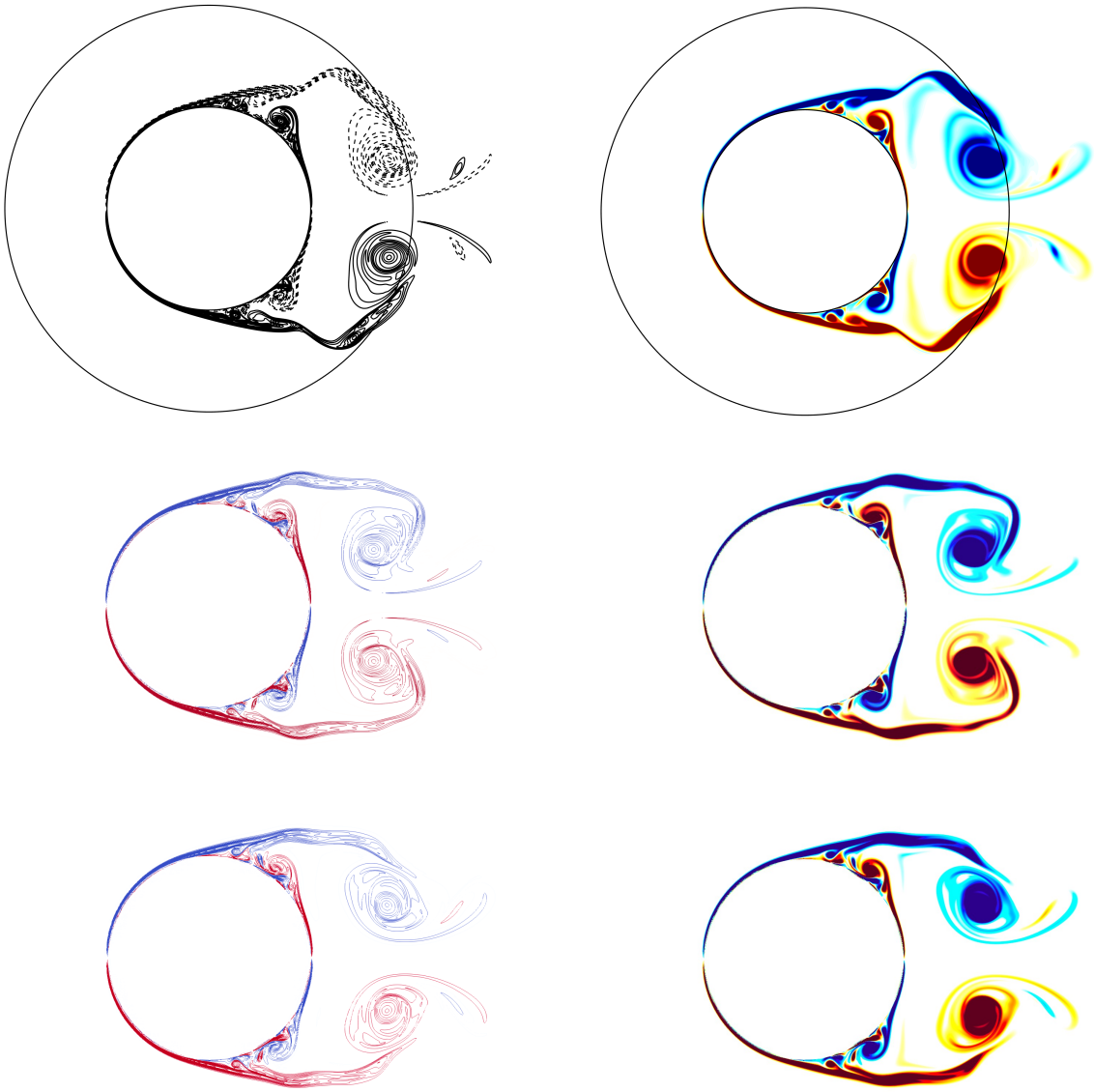

Figure 2.10: Impulsively-started cylinder at Re = 9500: contours of dimensionless vorticity at $t^* \simeq 3$. Contour levels are by steps of 6. The color plot is saturated for $|\omega_x^*| > 20$. First row: Billuart et al. [35] at $t^* \simeq 3$. The circle represents the boundary between his Near-Wall (NW) region and his Vortex Particle-Mesh (VPM) region. Second row: Implicit approach with $\lambda = 10^8$. Third row: Iterative approach with $\alpha = 1.9$ and $\epsilon = 0.05$.

to use a multiresolution and Lagrangian framework, especially if the cylinder has a non-zero thickness. Otherwise, the time step would be too small, and the computational time would be impossibly high. Those two considerations are the next important steps to improve the accuracy of our simulations and to consider even higher Reynolds numbers.

Iterative method convergence studies

In this section, we perform a convergence analysis of the different parameters involved in the iterative method. As mentioned above, λ is not anymore a parameter of the problem and is replaced by the relaxation parameter $\alpha \triangleq \lambda \Delta t < 2$.

We first analyse the influence of the stopping criterion ϵ for a fixed $\alpha = 1.9$. It is obviously expected that if ϵ is small, the solution takes more steps to converge but provides a better accuracy. The objective is to determine the optimal ϵ, that leads to a good balance between computational time and accuracy. The results are shown on Figure 2.11 for the beginning of the simulation. It is observed that the number of iterations approximately decreases as $\mathcal{O}(\epsilon^{-1})$ for this case. We note that this behaviour is case dependent, because number of iterations is correlated to the Jacobi method and to the eigenvalues of the problem, which vary depending on the linear system to solve. Here, considering $\epsilon = 0.05$ seems to be a good compromise, at least for short term simulations. It is also the value chosen by Spietz et al. [4]. Finally, we observe in practice that the number of iterations are different depending on the Poisson solver considered (regularized 10^{th} order from Hejlesen [17] in this

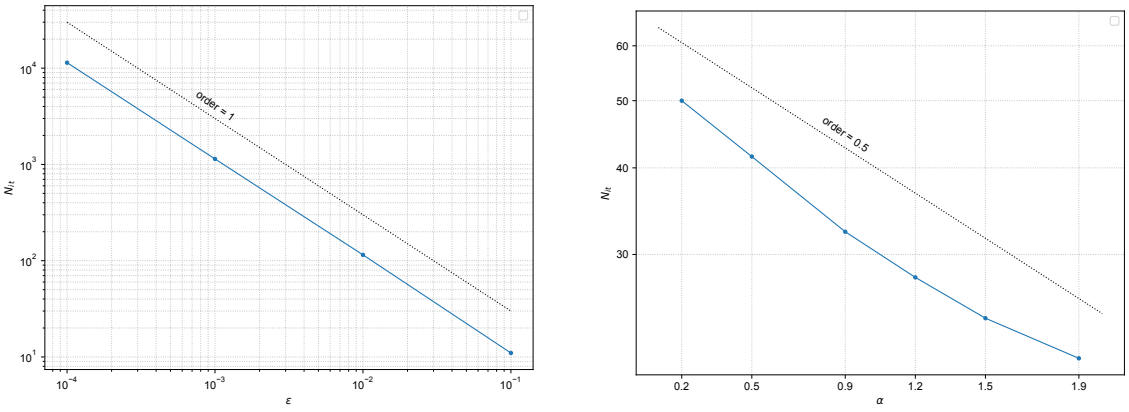

Figure 2.11: Impulsively started cylinder at $Re = 9500$, iterative method sensitivity analysis. Left : Number of iterations as a function of the stopping criterion ϵ, for $\alpha = 1.9$. Right = Number of iterations as a function of α, for $\epsilon = 0.05$.

work), and higher for higher order solvers ($\simeq 20$ iterations for $\epsilon = 0.05$ and $\alpha = 1.9$ in this work). Analyzing the effect of the Poisson solver on the number of iterations could be the object of further investigations.

For the relaxation criterion analysis, the number of iterations to reach the convergence criteria can be increased or decreased by choosing an under-relaxation ($0 < \alpha < 1$) or an over-relaxation ($1 < \alpha \leq 2$), respectively, but it does not affect the accuracy of the solution. In general, the choice between over-relaxed and under-relaxed Jacobi iteration depends on the specific problem being solved and on the behavior of the Jacobi method. In our case, it is more efficient to use α as high as possible to reduce the number of iteration, hence the choice $\alpha = 1.9$. In our case, the number of iteration decreases approximately in $\mathcal{O}(\alpha^{-0.5})$.

2.5 Flow past a sphere at Re=300

In this section, we consider a widely used benchmark in a purely three-dimensional case, that is a flow past a sphere at a moderate Reynolds number of $Re = 300$. Higher Reynolds number could naturally be considered, but we remind that since a uniform grid is used, the computational cost increases as Re^3. We consider a stationnary sphere like in the work of Gillis [6] and Mimeau et al. [18]. According to Ploumhans et al. [33], there exists different flow regimes for a flow past a sphere

- $Re \leq 200$: steady and axisymmetric flow.
- $210 \leq Re \leq 270$: steady and planar-symmetric flow.
- $270 \leq Re < 375$: unsteady, planar-symmetric and periodic flow.
- $375 < Re < 800$: unsteady, non-periodic and fully asymmetric flow.
- $Re > 800$: unsteady, non-periodic and asymmetric flow. A Kelvin-Helmholtz instability [37] appears in the shear layer.

2.5.1 Impulsively started flow

We first consider an impulsively started flow until $t^* = 1$. The first objective is to analyze the influence of the mesh size $h^* = \frac{h}{D}$ on the drag coefficient. We compare the results with the ones of Gillis [6], that used a 3D Immersed Interface Vortex Particle-Mesh method (IIVPM) for taking into account the obstacle. The latter work considered two forces computations, Noca's formula [27] and a divergence decomposition introduced by Wu et al. [38, p. 140] (based on the divergence of the vorticity). We remind that we only compute the drag using the implicit and iterative penalization methods. As already mentioned in Section 2.4, we do not

consider the direct explicit method, for its stability constraint and is less accurate results that the implicit method. The short term drag coefficients C_D are shown on Figure 2.12.

We can observe that both implicit and iterative methods provide similar results. The iterative method however seems slightly closer to the reference solution of Gillis that the implicit one, particularly for the coarsest mesh size. Also, the penalization methods are converging to the reference solution for $t^* \gtrsim 0.4$. Concerning the choice of the mesh size chosen for long time simulations at $Re = 300$, the coarsest mesh resolution gives a relative error (compared to the finest penalization resolution) $E_{\text{rel}} \approx 6.5\%$ at $t^* = 0.7$ and $E_{\text{rel}} \approx 5\%$ with the iterative one. However, the computational time is multiplied by almost 3 between $h^* = \frac{1}{48}$ and $h^* = \frac{1}{96}$. Therefore, we use an intermediate value that is a good trade-off between computational cost and accuracy. For this case, we consider $h^* = \frac{1}{64}$ and we analyze in more details the associated computational cost in Section 2.7. For long time simulations, this is the mesh choice considered by Gillis [6] and similar to Ploumhans et al. [33]. Mimeau et al. [18] considered however a coarser mesh resolution with $h^* = \frac{1}{50}$.

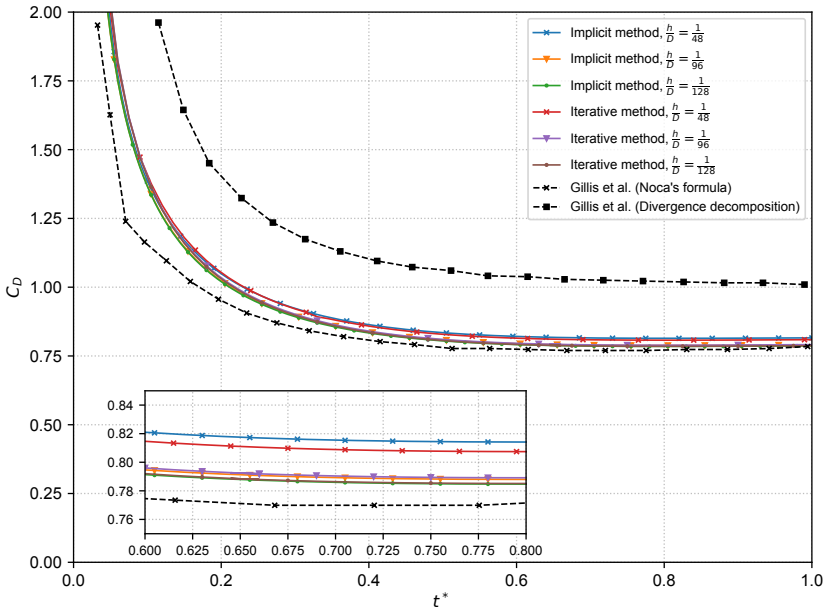

Figure 2.12: Impulsively started sphere at $Re = 300$. Comparison of the drag coefficients with Gillis [6]. The two reference solutions use $h^* = \frac{1}{128}$.

2.5.2 Long time simulation

In this section, we perform an analysis on a longer time. As mentionned, we consider a resolution of the mesh equivalent to the one used in Gillis et al. [6] and in Ploumhans et al. [33], that is $h^* = \frac{1}{64}$. To break the symmetry of the flow and trigger instabilities earlier, we consider a vertical perturbation of the uniform upstream flow. We perform the same one as in Mimeau et al. [19] and Ploumhans et al. [33], between $t^* = 3$ and $t^* = 4$

$$U_{\infty y}(t^*) = \sin\left[\pi(t^* - 3)\right] \tag{2.13}$$

The oscillations of the drag coefficient will allow us to extract the dominant frequencies of the system. In general cases, we determine those by computing a Fast Fourier Transform (FFT) on the drag coefficient. Here, we know that we only have one dominant frequency, so we can visually see the period of the oscillations. If we consider T^* to be the non-dimensional period of the oscillations, we define the Strouhal number as a non-dimensional frequency

$$St \triangleq \frac{fD}{U_\infty} = \frac{1}{T^*} \tag{2.14}$$

In the literature, the mean lift and drag coefficient $\overline{C_L}$ and $\overline{C_D}$ are often used as a comparison for an established flow. We compare our results with values obtained from experimental and computational approaches. Our implicit method is used with $\lambda = 10^8$ and the iterative method with $\alpha = 1.9$ and $\epsilon = 0.1$. We choose a smaller stopping criterion than for short time simulations since in practice, we already perform around 10 Poisson iterations per time step with this criterion on this test case. A summary of the results is shown in Table 2.2. The oscillations of the lift and drag coefficients are shown on Figures 2.13 and 2.14.

Author	$\overline{C_D}$	$\overline{C_L}$	St
Roos & Willmarth*[39]	0.629	-	-
Ploumhans et al. [33]	0.683	-0.061	0.135
Gillis [6]	0.681	-0.067	0.134
Mimeau et al. [18]	0.673	-0.066	0.133
Present work: Implicit method ($\lambda = 10^8$)	0.677	-0.071	0.134
Present work: Iterative method ($\epsilon = 0.1$)	0.682	-0.072	0.134

Table 2.2: Long time simulation of a flow past a sphere at Re=300. Comparison of the results with the literature. The sources marked with (*) are experimental results, the others are computational ones.

We see that the results are coherent with the literature. On one side, the Strouhal numbers are equivalent for both methods. On the other side, the lift and drag coefficients oscillate in a coherent range of values. We note that near the stagnation point of the flow, the resolution is poor. Indeed, if we consider equation (2.9) as a quality criterion and compute N_δ, we obtain a value of 3.7, which is relatively small compared to the values used for the cylinder. It certainly affects the accuracy of the forces computation and gives less accurate results. Also, as mentioned in Section 1.4.2, a loss of accuracy is expected in the case of long time simulations because we compute the force with the vorticity moment of the flow [22].

Comparing the two penalization methods, the iterative method yields slightly higher values compared to the implicit one. The implicit method is closer to the results of Mimeau et al. [18], since an implicit approach is also considered. There is however a slight discrepancy because of the different mesh resolutions considered, and the fact that Mimeau applied periodic boundary conditions and a vorticity absorption filter at the boundary to reset vorticity to zero. On the other side, the iterative method aligns more closely with the results of Gillis [6] and Ploumhans et al. [33]. This is because of accuracy improvement of iterative methods, that are already observed on Figure 2.12. However, the computational time is almost doubled, despite the relatively big stopping criterion $\epsilon = 0.1$ we use.

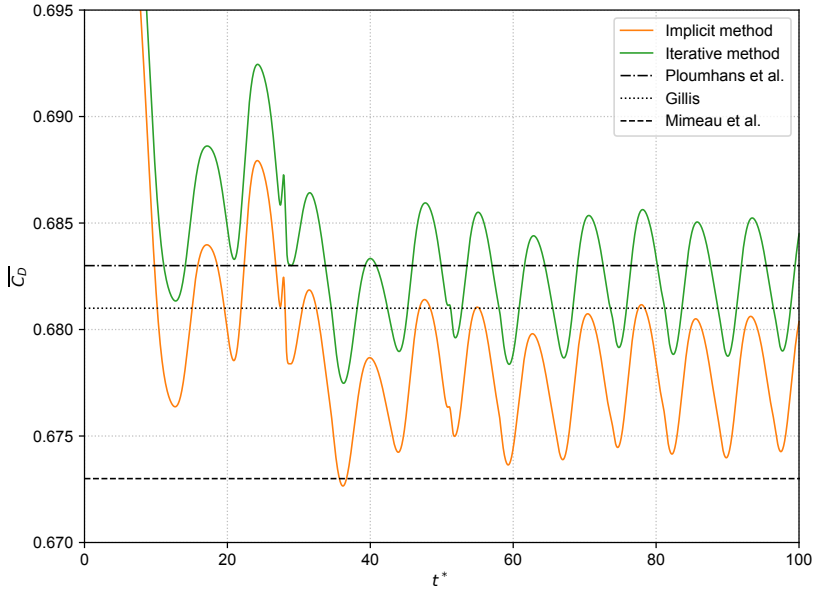

Figure 2.13: Flow past a sphere at $Re = 300$. Long term drag coefficient C_D using the iterative and the implicit method and comparison with $\overline{C_D}$ from the literature.

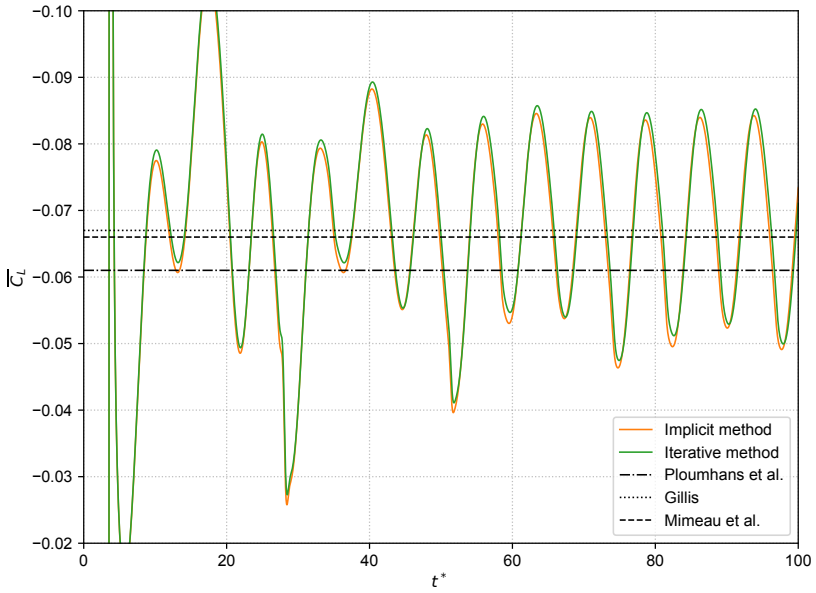

Figure 2.14: Flow past a sphere at $Re = 300$. Long term lift coefficient C_L using the iterative and the implicit method and comparison with $\overline{C_L}$ from the literature.

Finally, it is seen that small and predictable oscillations emerge at the outflow boundary during the perturbation phase, because of the vertical upstream velocity disturbance. However, these effects are observed to have no impact for the rest of the simulation, since the vortices are sufficiently far from the boundary during the perturbation.

Flow analysis

We also analyze the vorticity generated in the flow past a sphere. The corresponding results are depicted on Figures 2.15 and 2.16 at each quarter of period of a drag oscillation. We consider slices across the center of the sphere, and the plots are organized in rows. To ensure that the flow is fully established, we consider a period that is relatively far from the beginning the simulation (we consider the flow to be established for $t^* \gtrsim 50$ in our case).

Firstly, it is worth noticing that the vorticity component ω_x in the YZ plane exhibits an asymmetry. According to Gillis [6], this behaviour is typically associated to a non-zero lift coefficient. This confirms the presence of lift that we see on Figure 2.14. Also, this same component shows the highest vorticity close tho the boundary of the object. Secondly, we observe that the wake exhibits a perfect

symmetry in the XZ plane. This observation aligns with the description of the flow regime at $Re = 300$ from Ploumhans et al. [33], as presented in the introduction of this Section. Finally, we can see that the outflow boundary conditions models properly the flow at the z+ boundary.

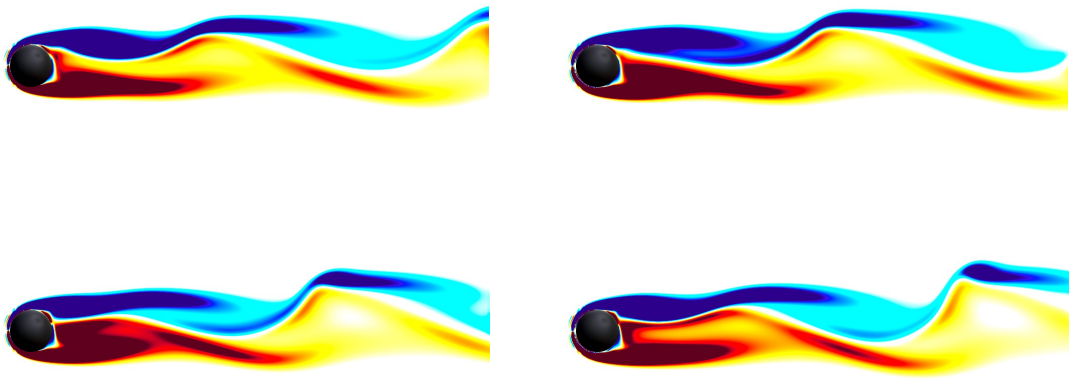

Figure 2.15: Flow past a sphere at $Re = 300$. Vorticity ω_x^* in the YZ plane for every quarter period of a shedding cycle, with the iterative method and $\epsilon = 0.1$. The color plot is saturated for $|\omega_x{}^*| \geq 6$.

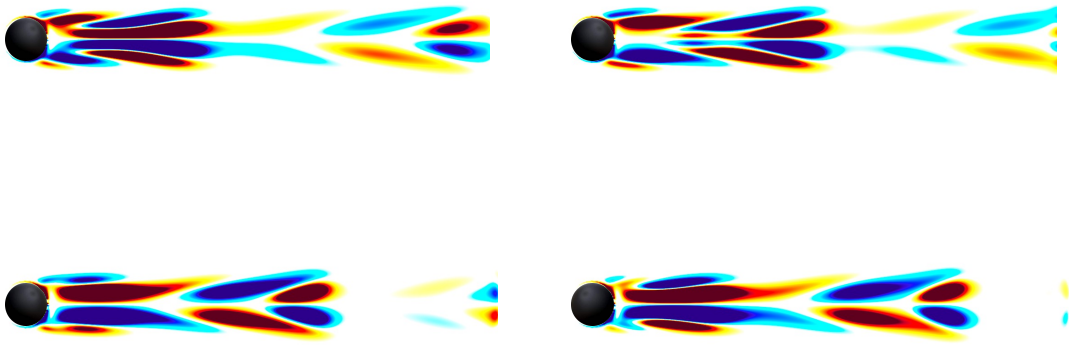

Figure 2.16: Flow past a sphere at $Re = 300$. Vorticity ω_z^* in the XZ plane for every quarter period of a shedding cycle, with the iterative method and $\epsilon = 0.1$. The color plot is saturated for $|\omega_z{}^*| \geq 1$.

2.6 Application: Impulsively started flows past airfoils

In this last test case, we consider airfoils extruded along the x-direction, with computational domains shown on Figure 2.1. To benefit from the three-dimensional framework, wing shapes with varying cross sections could also be considered and be the object of further investigations. This application case clearly constitutes in applications in aerodynamics, and is a first step towards penalization techniques applied to airfoils. The complex shape of airfoils makes penalization methods an excellent choice on structured grids, and the idea is to check whether penalization methods can accommodate more complex object configurations.

For our analysis, we consider two Reynolds numbers of $Re = 1000$ and $Re = 5000$, that are typically encountered in the analysis and design of airfoils for micro air vehicles (MAVs). We consider two airfoil shapes: a NACA0012 and a regularized Joukowski airfoil from Billuart et al. [35], with a radius of curvature at the trailing edge $r_t \approx \frac{c}{400}$. For technical details regarding the nomenclature of those airfoils, the reader is referred to Appendix A. The idea is to compare the flow behaviour around those two airfoils under a similar angle of attack ($\alpha = 20°$), at early stages of the simulation. The main difference between those is their camber, thickness and the fact that their trailing edge is curved (regularized Joukowski airfoil) or sharp (classical NACA airfoil). At $Re = 5000$, we compare the vorticity contours with Billuart et al. [35] for the regularized Joukowski airfoil flow.

2.6.1 Flow at Re=1000

Firstly, we consider a Reynolds number of $Re = 1000$. We only consider short time simulations, with $t^* \leq 6$ (the flow is still planar at this time) and the iterative method with $\epsilon = 0.05$. The analysis of the Strouhal number on longer times could naturally be the object of further investigations. Concerning the mesh resolution, we refer to the grid convergence study of Kurtulus [40] for $\alpha = 20°$ on a NACA0012 airfoil and consider a grid size of $h^* = \frac{h}{c} = \frac{1}{384}$. We fix the same mesh resolution for the case of the Joukowski airfoil, because the boundary layer is expected to be thicker since the leading and trailing edges have larger radii of curvature. We will check more precisely the thickness of the boundary layers at the leading edges with the flow analysis. Also, we still consider a constant $CFL_{\max} = 0.6$ along the simulations and a maximum Fourier number $R_{\max} = 0.1$.

Figure 2.17 shows the vortex dynamics of both airfoils at early stages of the flow. First, one can observe in both cases the startup vortex emerging from the

trailing edges at $t^* = 1$. The startup vortex is a phenomenon that occurs when an airfoil starts to generate lift, and is due to the effects of viscosity. We can see that it is slightly stronger for the Joukowski airfoil. One of the possible reasons is the rounded shape of the Joukowski trailing edge. Since it allows the high pressure fluid to bleed more easily into the upper surface of the airfoil, it could increase the intensity of the vortex. The other reason is that the startup vortex is slightly further for the NACA0012 and has thus attenuated a bit. At the same time $t^* = 1$, we can also see that the flow separation occurs closer to the leading edge for the NACA0012 airfoil, most probably because of its less pronounced suction side curvature there. Furthermore, a leading edge vortex is starting to develop on the suction side of the NACA0012, whereas the Joukowski airfoil flow still remains attached to the airfoil's surface and follows the curvature more smoothly.

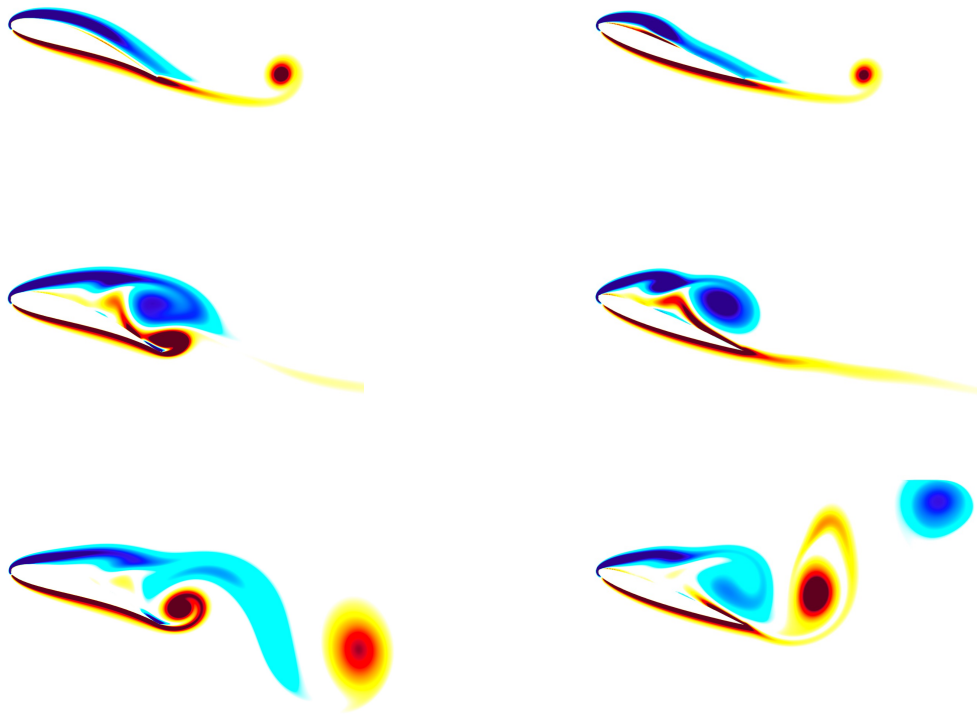

Figure 2.17: Dimensionless vorticity field for airfoils at $Re = 1000$ and $\alpha = 20°$. Left column: regularized Jukowski airfoil. Right column: NACA0012 airfoil. First row: $t^* \simeq 1$. Second Row: $t^* \simeq 3$. Third row: $t^* \simeq 6$. The plot is saturated for $|\omega_x^*| > 15$. The two airfoils are scaled with the same chord length. The iterative method is used with $\epsilon = 0.05$ and $\alpha = 1.9$.

At $t^* = 3$, we can observe the apparition of a large vortex at the suction side, which appears to form at different times depending on the airfoil shape. In the case of the NACA0012 airfoil, the leading edge vortex is distinguishable, whereas this vortex is interacting with an opposite sign vortex for the Joukowski airfoil. Indeed, the pressure side boundary layer of the Joukowski airfoil already rolls up into a positive vortex (trailing edge vortex), which has not yet clearly appeared for the NACA0012 airfoil at $t^* = 3$. At $t^* = 6$, a repetitive regime phase begins to establish, characterized by alternating positive and negative vortices. The regularized Joukowski airfoil seems to reach its regime phase earlier and to have a higher shedding frequency, since at $t^* = 6$, a second positive vortex is already being detached from the Joukowski airfoil. We also observe that those periodic vortex pairs do not have an opposite strength, as also mentioned and observed by Billuart et al. [35].

Concerning the number of points in the boundary layer N_δ, we observe in practice (looking at the vorticity contours with the mesh points on top) $N_\delta \approx 5.5$ at the leading edge of the NACA0012 airfoil and $N_\delta \approx 6$ for the Joukowski airfoil. It is expected to be lower for the NACA0012 because its leading edge curvature is smaller (thinner airfoil). The uniform mesh resolution seems thus appropriate for this Reynolds number and has the same order as the N_δ chosen for the cylinder case.

Finally, the different forces are shown on Figure 2.18 until $t^* = 6$. Firstly, for both airfoils, we can see that the drag coefficient is smaller than the lift produced, which is naturally expected for airfoils. Secondly, concerning the lift, the first

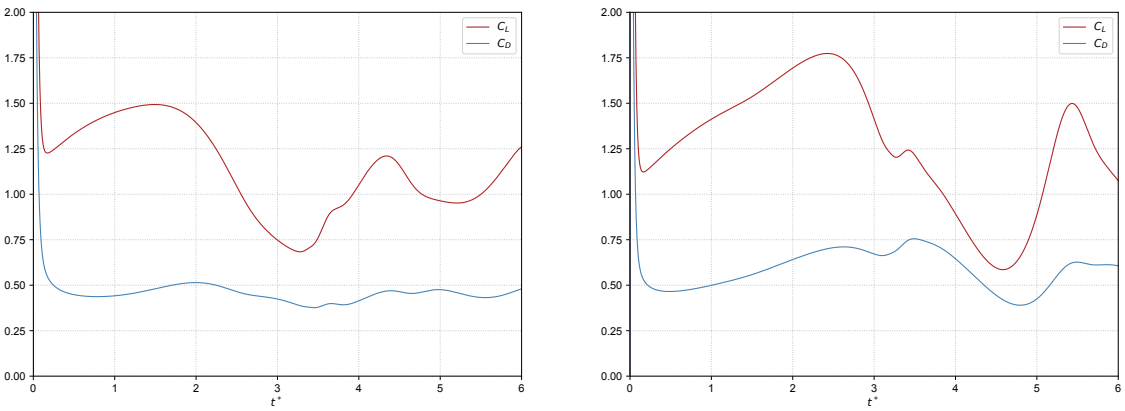

Figure 2.18: Flow past the airfoils at $Re = 1000$ and $\alpha = 20°$: evaluation of the forces. Left: regularized Joukowski airfoil with $r_t \approx \frac{1}{400}$. Right: NACA0012 airfoil.

peak value occurs earlier ($t^* \approx 1.5$ vs $t^* \approx 2.5$) and is less pronounced for the Joukowski airfoil. The lift oscillations are correlated to the leading and trailing edge vortices appearing one after another and detaching from the airfoil. Additionally, the vorticies appearing faster for the Joukowski airfoil are coherent with the force diagnostic. Indeed, the second lift peak appears at $t^* \approx 4.3$ for the Joukowski airfoil and at $t^* \approx 5.2$ for the NACA0012. Concerning the shedding frequency, we can only give hypothesis on such a short time period. We expect the Joukowski airfoil to have a higher shedding frequency because the vortices detach faster than for the NACA0012 airfoil. Longer simulations will help confirm this hypothesis and determine further properties of the flows. We leave it as a subject of future work.

2.6.2 Flow at Re=5000

This last section analyses a flow are $Re = 5000$ for the regularized Joukowski airfoil. The objective of this section is to compare our flow with the results of Billuart et al. [35]. The same outflow boundary conditions are considered. In this work, we consider the same mesh size as for $Re = 1000$, that is $\frac{h}{c} = \frac{1}{384}$. A comparison of the different vorticity fields of the regularized Joukowski airfoil is provided on Figure 2.19.

We see that the dynamics of the flow in the present study still bear resemblance to those in Billuart et al.'s work [35]. At $t^* = 1$, the results are almost identical and the startup vortex is at the same position. At $t^* = 3$, we can see that both airfoils experience a Kelvin-Helmoltz-like instability on the pressure sides [37], which was not observed for the case $Re = 1000$. As a reminder, this instability is characterized by a velocity shear in the fluid. For $Re = 5000$, the flow is more turbulent and exhibits stronger shear forces. This turbulence and higher shear triggers the Kelvin-Helmholtz instability at the trailing edge, leading to the formation of vortices. However, those vortices are not properly captured and are are too strong in this work's results. This is typical of an under resolution of the trailing edge, meaning that the rounded shaped is not perfectly captured. Also, the leading edge vortex is slightly different. At $t^* = 6$, the flow pattern is still approximately the same, but the amplitude and shape of the vortices becomes strongly different from the reference solution.

On top of the trailing edge under resolution, we can clearly observe on Figure 2.19 that the resolution at the leading edge is poor as well. If we have a closer look at the resolution at the stagnation point, we obtain $N_\delta \approx 3$, which is a clear sign of poor mesh resolution. This poor resolution affects the entire flow behaviour, and modifies more consequently the flow as time progresses. To have a better

resolution, using for instance a uniform resolution with $\frac{h}{c} \simeq \frac{1}{530}$ would of course be a better choice, giving a reliable flow with $N_\delta \geq 5$. However, we clearly see that the under resolution problems are located at the edges, and do not occur for the rest of the airfoil. Therefore, a multiresolution framework is of great interest and it is the reason why it is currently under development in *Murphy*. This framework was tested during this work, but was not fully optimal to consider it and to analyze it in this report. Ultimately, we want this multiresoltion framework to adapt automatically, because patched-based approaches require an "a priori" knowledge of the flow dynamics. For further details on the adaptive approach used in *Murphy*, the reader is referred to [1].

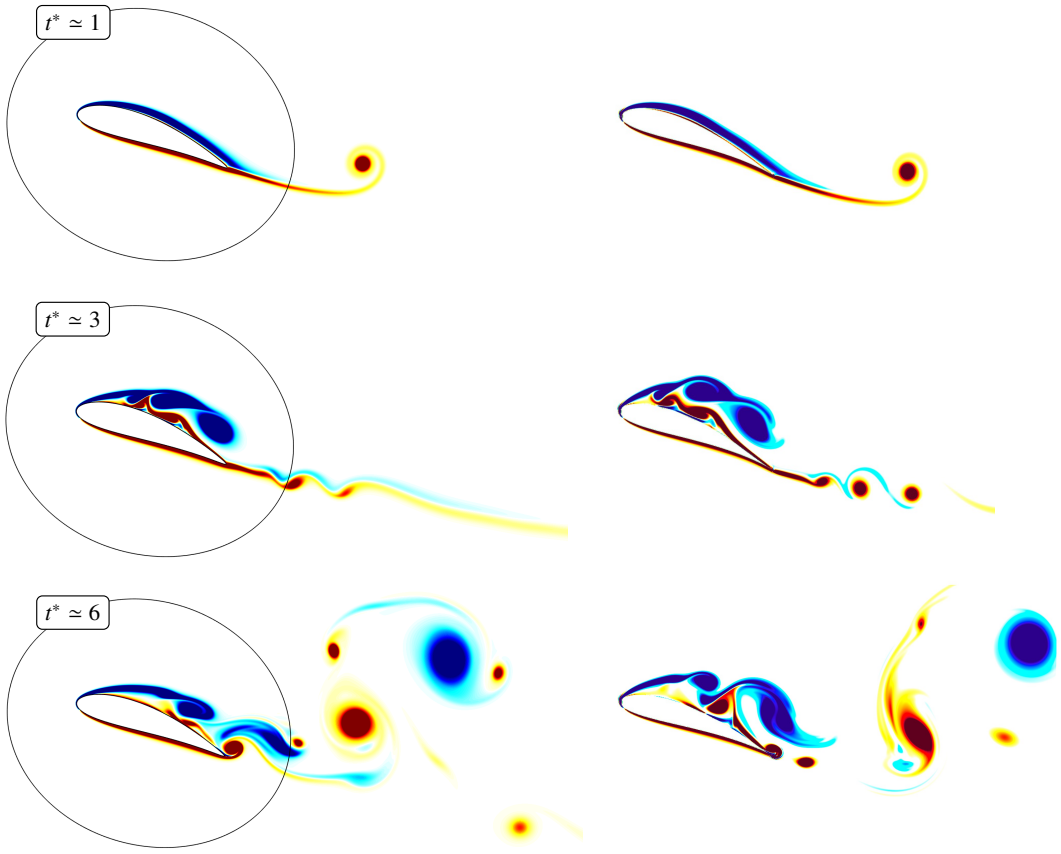

Figure 2.19: Dimensionless vorticity field for airfoils at $Re = 5000$ and $\alpha = 20°$. Left column : regularized Jukowski airfoil. Right column : NACA0012 airfoil. First row : $t^* \simeq 1$. Second Row : $t^* \simeq 3$. Third row : $t^* \simeq 6$. The plot is saturated for $|\omega_x^*| > 15$. The two airfoils are scaled with the same chord length. The iterative method is used with $\epsilon = 0.05$ and $\alpha = 1.9$.

2.7 Computational ressources

In this last section, we use a profiler to determine the repartition of operations for each of the penalization methods. The objective is to have a look the most expensive operations of the process and to compare the efficiency-accuracy trade-off between the different penalization methods proposed.

The simulations were performed on a cluster made of AMD Epyc Rome CPU processors, provided by the Consortium des Équipements de Calcul Intensif (CÉCI). Particularily, the cluster NIC5 has mostly been used. In this framework, the biggest configuration is 256 cores on a maximum running time of 2 days. Different configurations have been used, depending on the case considered in this work. Table 2.3 shows the parameter settings used for the different simulations. The different Δt^*_{avg} have been computed using Expression (2.1). We note that for the two airfoils, those are almost equivalent. Compared to the ressources used by Mimeau et al. [18], this work employs a higher number of cores, with for instance 4x more processors to run approximately the same sphere simulation. It outlines the clear efficiency improvements proposed by Lagrangian methods.

Also, Figures 2.20 and 2.21 give the percentage of the different stages implied in the resolution of the 3D penalized incompressible Navier-Stokes equations, in one simulation time step for the execution on 1 core, without dumping any field. We considered the case of the sphere at $Re = 300$. We can see that for all methods, the most expensive part is the Poisson problem. If we consider a direct penalization approach, the Poisson solver takes about half of the computational time. In Mimeau et al. [18], the most time consuming operation is the advection and the remeshing. In this work, we consider a uniform grid and Eulerian operations, so that the biggest operation is the Poisson problem. For the iterative methods, the majority of the computational time is spent in the iterative loop, and the proportion of time spent for solving the Poisson problem or doing the field operations remains

	Cylinder $Re = 550$	Cylinder $Re = 9500$	Sphere $Re = 300$	Airfoils $Re = 1000$
Δt^*_{avg}	8.37×10^{-4}	5.55×10^{-4}	5.75×10^{-3}	6.75×10^{-4}
t^*_{final}	5	3	100	6
N cores	128	256	256	256
CPU time	9h - 1.5d	1d - 2d	21h - 2d	13h - 1.7d

Table 2.3: Parameter settings and CPU time costs for the different simulations of this paper. • Direct methods • Iterative method

approximately the same for different values of ϵ. The proportion of time spent out of the iterative loop also naturally decreases with smaller values of ϵ, reaching an approximate asymptotic fraction of $\frac{2}{3}$ of the time spent in Poisson solver and the other $\frac{1}{3}$ spent for the rest of the iterative penalization operations.

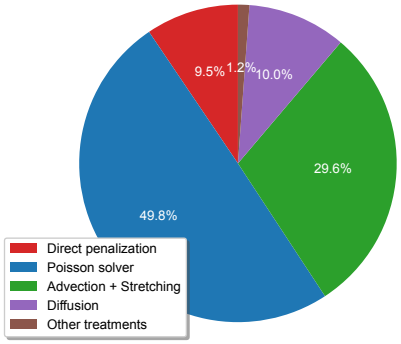

Figure 2.20: Percentage of the different stages implied in the resolution of the 3D penalized Navier-Stokes equations, in one simulation time step for the execution on 1 core, for the direct penalization methods.

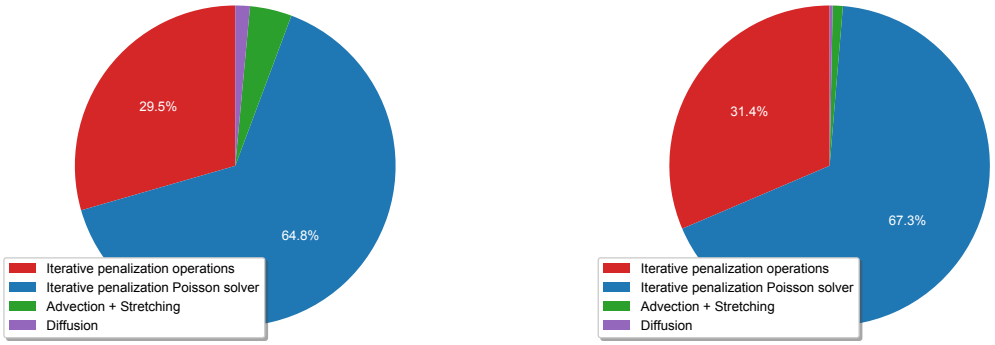

Figure 2.21: Percentage of the different stages implied in the resolution of the 3D penalized Navier-Stokes equations, in one simulation time step for the execution on 1 core, for the iterative penalization method. Left: $\epsilon = 0.1$. Right: $\epsilon = 0.05$.

Conclusion

In this work, we compared the treatment of the boundary of solid obstacles using three different Brinkman penalization techniques. These techniques employ continuous forcing approaches, which consider an additional term in the classical Navier-Stokes equations. The main advantage of such methods is their flexibility in implementing complex object geometries on structured grids. Also, we adopted a vorticity-based framework in which the penalization term acts as a correction of the unpenalized vorticity field

$$\boldsymbol{\omega}^{n+1} = \widetilde{\boldsymbol{\omega}}^{n+1} + \delta\boldsymbol{\omega}$$

As a first step, we employed an explicit direct penalization approach based on a first-order Euler discretization scheme. We observed a strong stability constraint on $\lambda\Delta t$ with this method. When considering a flow past a circular cylinder at $Re = 550$, we found that it resulted in a high velocity residual inside the body due to the small value of λ, which defines a porous object. This hypothesis was further confirmed by conducting a λ-convergence analysis. These observations prompted us to explore alternative approaches.

Then, we considered an implicit direct approach, based on an implicit Euler discretization of the velocity-based penalization formulation. This approach removed the stability constraint on the penalization term, so that arbitrarily high value of λ could be used. This led to more accurate results at moderate Reynolds numbers, but inaccuracies remained for the case of a flow past a cylinder at $Re = 9500$. The reason lies behind the substantial errors generated with this method, that leads to an inaccurate flow behaviour.

Lastly, we considered an iterative approach, initially introduced by Hejlesen et al. [7]. This method showed a higher accuracy, because the iterative aspect connects the velocity and the vorticity and gives the correct vortex sheet. We observed that this method could be equivalently formulated as solving a linear system using Jacobi iterations. Therefore, we mentioned that alternative solvers could be used such as the BiCGStab algorithm, that accelerates convergence. Additionally, we mentioned that recycling solutions from previous time steps were of great interest for penalization methods, because the solution slightly changes from one step to another. This idea forms the basis of recycled Krylov-based penalization methods.

In addition to testing flows past a cylinder, we conducted analyses on short and long time simulations of a flow past a sphere at $Re = 300$. The short time simulations served as validation for both the implicit and iterative methods in a purely three-dimensional case, and helped determine the appropriate mesh for longer simulations. By triggering instabilities, we observed periodic vortex sheddings and a frequency consistent with the literature results. Also, the amplitude of the drag coefficient slightly differed between the implicit and iterative methods, but still remained coherent.

As a more complex geometry application, we considered the case of airfoils and compared the short time flow behaviours of a sharp NACA0012 airfoil and a regularized Joukowski one at an angle of attack of 20°. At $Re = 1000$, the uniform grid and Eulerian framework were observed to be sufficient to obtain accurate and reliable results. A flow analysis helped us distinguish both flow behaviours and we saw that the drag and lift coefficients were behaving similarly for the two airfoils. At $Re = 5000$, the resolution was found to be very poor at specific locations, outlining the necessity of a multiresolution framework to capture the fine scales at the leading and trailing edges.

As a major conclusion of this work, we mention the balance between accuracy and computational cost. Clearly, iterative methods are of interest for the higher accuracy they propose. However, if we consider the classical iterative penalization algorithm, the implicit method has a net computational time advantage. For a flow past a cylinder at $Re = 550$ or past a sphere at $Re = 300$, the flow was observed to slightly differ, which outlines that in some cases, the implicit penalization is still a possible alternative. However, when the flow becomes more complex, an iterative approach becomes mandatory if we want to capture a coherent vortex sheet.

Finally, as an extension of this work, some improvements and analysis remain to be considered and will be a further step toward efficient penalization techniques.

1. A Lagrangian framework. We mentioned that the efficiency of our implementation could be drastically improved. The purely Eulerian framework restricts the choice of the time step and increases considerably the computational time. The use of Lagrangian techniques to treat the convective term of the Navier-Stokes equations is thus of prime interest. It will allow for longer simulation times and to increase even more the mesh resolution.

2. A multiresolution framework. This feature is still under development in *Murphy*. We observed that the majority of the domain was not having any vorticity and could have a coarser resolution. Working with an adaptive mesh that creates a coarse mesh at those locations and a fine one at the vicinity of the objects will definitely lead to an increase of the computational time. It is of particular interest at the stagnation points of objects, where the resolution is specifically low.

3. Efficiency improvements. The iterative method was shown to provide a better accuracy of the results, especially for higher Reynolds numbers. Reducing the number of steps until convergence is thus of interest, since one Poisson equation is solved at each step. Recycling the solutions of the previous iterations and considering a more efficient iterative algorithm will consequently decrease the number of iterations to converge.

4. Accuracy improvements. For example, a more robust force computation formula like Noca's formula [27] is of interest for its long-time simulation robustness and accuracy. Another idea could be to investigate temporal second order penalization methods, as introduced by Bergmann et al. [41].

5. More complex cases. For example, one could use multiple bodies and body velocities. We could also take advantage of the three-dimensional aspect of the cylinder and of the airfoils to obtain purely three-dimensional flows (either done by considering longer simulations or by artificially triggering instabilities).

This master's thesis explores the close relation between mathematical and mechanical engineering, highlighting the significant overlap between the two disciplines. The computational engineering field, with its emphasis on high-performance computing and numerical analysis tools, holds great importance for practical applications in mechanical engineering. This convergence continually gives rise to fresh opportunities, such as the conception of efficient tools like *Murphy* or the utilization of more advanced algorithms within penalization methods. By combining mathematics and mechanics, we discover endless possibilities and create groundbreaking innovations, leading to exciting opportunities for engineering excellence.

References

[1] T. Gillis and W.M. van Rees. MURPHY – a scalable multiresolution framework for scientific computing on 3D block-structured collocated grids. *Journal of Scientific Computing*, 51(6):1864–1872, 2022.

[2] C. Burstedde, L. Wilcox, and O. Ghattas. Scalable algorithms for parallel adaptive mesh refinement on forests of octrees. *SIAM Journal on Scientific Computing*, 33:1103–1133, 2011.

[3] D-G. Caprace, T. Gillis, and P. Chatelain. FLUPS: A Fourier-Based Library of unbounded poisson solvers. *SIAM Journal on Scientific Computing*, 43:31–60, 2021.

[4] H.J. Spietz, M.M. Hejlesen, and J.H. Walther. Iterative Brinkman penalization for simulation of impulsively started flow past a sphere and a circular disc. *Journal of Computational Physics*, 336:261–274, 2017.

[5] J. Gabbard. High-order energy- and helicity-preserving finite difference schemes for 3D vortex dynamics, MIT Van Rees Lab, 2021.

[6] T. Gillis. Accurate and efficient treatment of solid boundaries for the vortex particle-mesh method, 2019.

[7] M.M. Hejlesen, P. Koumoutsakos, A. Leonard, and J.H Walther. Iterative Brinkman penalization for remeshed vortex methods. *Journal of Computational Physics*, 280:547–562, 2015.

[8] J.T. Rasmussen, G-H. Cottet, and J.H. Walther. A multiresolution remeshed vortex-in-cell algorithm using patches. *Journal of Computational Physics*, 230:6742–6755, 2011.

[9] T. Gillis, G. Winckelmans, and P. Chatelain. An efficient iterative penalization method using recycled Krylov subspaces and its application to impulsively started flows. *Journal of Computational Physics*, 347:490–505, 2017.

[10] P. Angot, C. Bruneau, and P. Fabrie. A penalization method to take into account obstacles in incompressible viscous flows. *Numerische Mathematik*, 81:497–520, 1999.

[11] C.S. Peskin. Flow patterns around heart valves: A numerical method. *Journal of Computational Physics*, 10(2):252–271, 1972.

[12] R.P. Fedkiw, T. Aslam, B. Merriman, , and S. Osher. A non-oscillatory approach to interfaces in multimaterial flows (the ghost fluid method). *Journal of Computational Physics*, 152(2):457–492, 1999.

[13] T. Osher and R. Fedkiw. *Level set methods and dynamic implicit surfaces*. Springer-Verlag, New York, 2003.

[14] E.M. Kolahdouz, A. P. Singh, Brent, A. Craven, and B.E. Griffith. An immersed interface method for discrete surfaces. *Journal of Computational Physics*, 400:457–492, 2020.

[15] M.M. Hejlesen, P. Koumoutsakos, A. Leonard, and J.H. Walther. Iterative Brinkman penalization for remeshed vortex methods. *Journal of Computational Physics*, 280:547–562, 2015.

[16] P. Chatelain, A. Curioni, M. Bergdorf, D. Rossinelli, W. Andreoni, and P. Koumoutsakos. Billion vortex particle direct numerical simulations of aircraft wakes. *Journal of Computational Physics*, 197:1296–1304, 2009.

[17] M.M. Hejlesen, P. Chatelain J.T. Rasmussen, and J.H. Walther. A high order solver for the unbounded poisson equation. *Journal of Computational Physics*, pages 258–487, 2013.

[18] C. Mimeau, G-H. Cottet, and I. Mortazavi. Direct numerical simulations of three-dimensional flows past obstacles with a vortex penalization method. *Computers and Fluids*, 136:331–347, 2016.

[19] C. Mimeau, F. Gallizio, G-H Cottet, and I. Mortazavi. Vortex penalization method for bluff body flows. *International journal for numerical methods in fluids*, 79:55–83, 2015.

[20] C. Mimeau. Conception and implementation of a hybrid vortex penalization method for solid-fluid-porous media: application to the passive control of incompressible flows, 2015.

[21] N.Sharma and Patankar. A fast computation technique for the direct numerical simulation of rigid particulate flows. *Journal of Computational Physics*, 205:439–457, 2005.

[22] M. Gazzola, P. Chatelain, W. M. van Rees, and P. Koumoutsakos. Simulations of single and multiple swimmers with non-divergence free deforming geometries. *Journal of Computational Physics*, 230:7093–7114, 2011.

[23] S. Gottlieb and C. Shu. Total variation diminushing Runge-Kutta schemes. *Mathematics of Computation*, 67(221):73–85, 1998.

[24] S.Gottlieb, C. Shu, and E. Tadmor. Strong Stability-Preserving High-Order Time Discretization Methods. *SIAM review*, 43(1):89–112, 2001.

[25] T. Gillis. Investigation de techniques de pénalisation en formulation vorticité-vitesse des équations de Navier-Stokes, 2015.

[26] JP. Caltagirone. Sur l'intéraction fluide-milieu poreux: application au calcul des efforts exercés sur un obstacle par un fluide visqueux. *Comptes Rendus de l'Académie des Sciences Paris*, 318:571–577, 1994.

[27] F. Noca, D. Shiels, and D. Jeon. A comparison of methods for evaluating time-dependent fluid dynamic forces on bodies, using only velocity fields and their derivatives. *Journal of Fluids and Structures*, 72(5):551–578, 1999.

[28] J.C. Wu. Theory for aerodynamics force and moments in viscous flows. *American Institute of Aeronautics and Astronautics Journal*, 19(4):431–441, 1981.

[29] Y. Saad. *Iterative Methods for Sparse Linear Systems, Second Edition*. Society for Industrial and Applied Mathematics, 2003.

[30] M.L. Parks, E. de Sturler G. Mackey, D.D. Johson, and S. maiti. Recycling Krylov subspaces for sequences of linear systems. *SIAM Journal on Scientific Computing*, 28(5):1651—1674, 2006.

[31] H.A. Van der Vorst. BI-CGSTAB: A Fast and Smoothly Converging Variant of BI-CG for the Solution of Nonsymmetric Linear Systems. *SIAM Journal on Scientific computing*, 13(2):631—-644, 1992.

[32] G-H. Cottet and P. Poncet. Advances in direct numerical simulation of 3D wall-bounded flows by Vortex-In-Cell methods. *Journal of Computational Physics*, 193:136–158, 2002.

[33] P. Ploumhans, G. Winckelmans, J.K. Salmon, A. Leonard, and M.S. Warren. Vortex Methods for Direct Numerical Simulation of Three-Dimensional Bluff Body Flows: Application to the Sphere at Re = 300, 500, and 1000. *Journal of Computational Physics*, 178:427–463, 2002.

[34] M. Bar-Lev and H.T. Yang. Initial flow field over an impulsively started circular cylinder. *Journal of Fluid Mechanics*, 72:625–647, 1975.

[35] P. Billuart, M. Duponcheel, G. Winckelmans, and P. Chatelain. A weak coupling between a near-wall Eulerian solver and a Vortex Particle-Mesh method for the efficient simulation of 2d external flows. *Journal of Computational Physics*, 473:111726, 2023.

[36] P. Koumoutsakos and A. Leonard. High-resolution simulations of the flow around an impulsively started cylinder using vortex methods. *Journal of Fluid Mechanics*, 296:1–38, 1995.

[37] F. Gallaire. ME-466: Instability, 2022-2023. École Polytechnique Fédérale de Lausanne (EPFL).

[38] J.-Z. Wu, H.-Y. Ma, and M.-D. Zhou. *Vorticity and Vortex Dynamics*. Springer, 2006.

[39] F. W. Roos and W. Willmarth. Some Experimental Results on Sphere and Disk Drag. *American Institute of Aeronautics and Astronautics Journal*, 9:285–291, 1971.

[40] D.F. Kurtulus. On the Unsteady Behavior of the Flow Around NACA0012 Airfoil with Steady External Conditions at Re=1000. *International Journal of Micro Air Vehicles*, 7(3):301–326, 2017.

[41] M. Bergmann, J. Hovnanian, and A. Iollo. An accurate Cartesian method for incompressible flows with moving boundaries. *Communications in Computational Physics*, 15(5):1266–1290, 2014.

[42] A.L. Quintanilla. Development of a fast shape memory alloy based actuator for morphing airfoils, 2016.

[43] P. Chatelain, E. Deleersnijder, and G. Winckelmans. LMECA2322: Fluid mechanics II, 2021-2022. Université Catholique de Louvain.

[44] K. Mulleners. ME-445: Aerodynamics, 2022-2023. École Polytechnique Fédérale de Lausanne (EPFL).

[45] J. Moran. *An introduction to theoretical and computational aerodynamics*. Dover, 2003.

APPENDIX A

Background on airfoil theory

Airfoil nomenclature

We first remind the basic nomenclature of airfoils. An airfoil is the cross-sectional shape of a three dimensional wing. The forward section of the airfoil is named the leading edge and the rear is named the trailing edge. The upper part of the airfoil is named the suction side and the lower part is named the pressure side. The line linking the leading and trailing edge is named the chord, and the chord length is denoted c. An airfoil visualization is shown on Figure A.1.

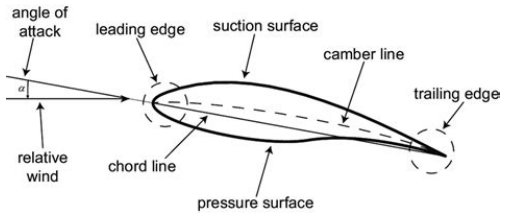

Figure A.1: Airfoil terminology, from [42]

Two parameters are important when designing an airfoil, the thickness and the camber. The thickness is often denoted t and the thickness to chord ratio $\frac{t}{c}$ is the non-dimensional parameter corresponding to it. The camber is a measure of the asymmetry between the upper and lower surface. Camber is generally introduced to an airfoil to increase its maximum lift coefficient, which in turn decreases the stall speed of the aircraft. The camber line is a line drawn equidistant between

the upper and lower surface at all points along the chord. Highly cambered airfoils produce more lift than lesser cambered airfoils, and an airfoil that has no camber is symmetrical upper and lower surface.

The last parameter that can be varied is the angle of attack (AoA or α) and is the angle between the body's reference line and the oncoming flow. For a symmetric airfoil, the only way to generate lift is to have a non-zero angle of attack.

The Joukowski airfoil

A conformal mapping function is an analytical function that preserves the local angle and whose derivative is non zero everywhere. If we consider a constant cross-section for our wing, we can switch to complex coordinates and create a conformal mapping between a circle and an airfoil. If the initial plane is set as the Z-plane and the mapped plane is set as the z-plane, we define the Joukowski transform [43] [44] as

$$z = [Z + \frac{b^2}{Z - Z_0}]e^{-i\pi\gamma} \quad \text{where} \quad Z_0 = \epsilon e^{-i\delta} \quad \text{with} \quad 0 < \epsilon < a \quad \text{and} \quad 0 \leq \delta \leq \frac{\pi}{2} \tag{A.1}$$

where a is the radius of the mapped circle and γ the angle of attack in radians. Also, ϵ and δ are two parameters that vary the shape of the airfoil. The extreme cases are $\epsilon = 0$, leading to a flat plate and $\delta = \frac{\pi}{2}$, leading to a zero thickness airfoil. For comparison purposes, we consider the same two parameters as in Billuart et al. [35], that is $\epsilon = 0.15a$ and $\delta = \frac{\pi}{4}$. Finally, the parameter b can be found by imposing a sharp trailing edge.

In the context of penalization methods, the task at hand is to determine whether a given point in the z-plane lies inside or outside the Joukowski airfoil. One way to do this is to apply the inverse Joukowski transform to the point, which will map it back onto the Z-plane. Then, determining whether the point is inside or outside of the circle is a straightforward. We also note that the classical Joukowski airfoil has a sharp trailing edge with a cusp. To address this, Billuart et al. [35] proposed using a regularized Joukowski airfoil, where the trailing edge is curved by considering a slightly larger initial radius $r > a$, and computing $Z = re^{i\theta}$, with $0 \leq \theta \leq 2\pi$. This approach results in a rounded trailing edge and regularizes the airfoil.

In this work, we consider the same radius of curvature at the trailing edge $r_t \approx \frac{c}{400}$. This can be me measured by placing a circle at the trailing edge and checking for its radius. This is shown on Figure A.2 on a Joukowski airfoil with $\alpha = 0°$. Rounding the trailing edge of an airfoil allows to modify the flow behaviour by displacing the trailing edge stagnation point, which is of interest for flow control and aerodynamic performances parametrization.

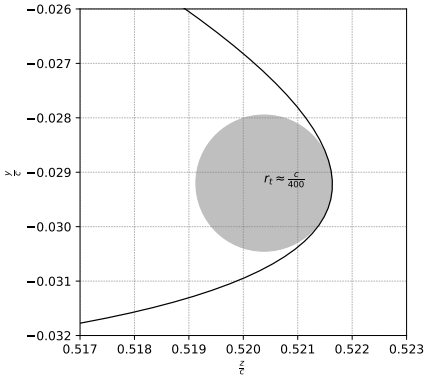

Figure A.2: Radius of curvature of the Regularized Joukowski airfoil

The 4-digit symmetric NACA airfoil

The second airfoil we analyze is the sharp symmetric 4-digit NACA airfoil. The two first digits are related to the camber. Since we consider a symmetric airfoil, those are set to 00. The last two digits correspond to the maximum thickness to chord ratio $\frac{t}{c}$. For instance, $\frac{t}{c} = 0.12$ for a NACA0012 airfoil. For such airfoils, an explicit polynomial formula exists for the upper and lower profiles [45]

$$y_{U,D} = \pm\frac{t}{0{,}2}\left[0{,}2969\sqrt{\frac{x}{c}} - 0.1260\left(\frac{x}{c}\right) - 0.3516\left(\frac{x}{c}\right)^2 + 0.2843\left(\frac{x}{c}\right)^3 - 0.1015\left(\frac{x}{c}\right)^4\right] \tag{A.2}$$

In practice, we also need to consider a variable angle of attack for this airfoil. To know whether an arbitrary point is inside or outside the airfoil with an angle of attack of α, we simply apply a rotation matrix to it to map it back to an $\alpha = 0°$ configuration of the airfoil

$$\begin{bmatrix} x_{\text{rot}} \\ y_{\text{rot}} \end{bmatrix} = \begin{bmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} \tag{A.3}$$

Since the airfoil is symmetric, the related mask function χ of penalization methods can be directly created.

APPENDIX B

Murphy: technical aspects

In this appendix, we describe the way the results and the simulations can be reproduced in *Murphy*. All the results were post-processed with Paraview and/or Python. Different parameters can be set as input to reproduce or recreate results. The different penalization clients are herited from the *"vortexdyn"* class. See `https://github.com/vanreeslab/murphy` (public repository) for further details. Also, the implementation details are provided in the GitHub repository documentation.

To run a simulation, different flags can be passed in the terminal and understood with the H3LPER library. To run a penalization algorithm, the code first has to be compiled and then can be run with the command *./murphy*, followed by the one of the following penalization client flags

Flag	Explanation
–penalization	Runs the explicit penalization algorithm The default value is set to $\lambda = \frac{1}{\Delta t}$
–pen-implicit	Runs the implicit penalization algorithm The default value is set to $\lambda = 10^8$
–pen-iterative	Runs the iterative penalization algorithm The default values are set to $\alpha = 1.9$ and $\epsilon = 0.05$

Table B.1: Penalization clients option

Once the type of penalization technique is chosen, one can define properly the object to be modeled. The different parameters to tune are provided in Table

B.2. Four objects are currently implemented, but others could be added. Each object correspond to a specific tag value listed here under. Also, one can change the boundary conditions to apply to our problem.

Flag	Explanation
–object	0: Cylinder case (default value) 1: Sphere case 2: NACA0012 airfoil ($\alpha = 20°$) 3: Regularized Joukowski airfoil ($\alpha = 20°$)
–radius	Radius of the cylinder/sphere Chord length of the airfoils
–center	Triplet fixing the center of the object
–uinf	Triplet fixing the upstream velocity $\mathbf{u}_\infty$
–reynolds	Reynolds number, based on $\mathbf{u}_\infty$ and the radius
–dom	Triplet fixing the domain dimensions Must be whole numbers
–per	Triplet fixing the boundary conditions in each direction Default boundaries are all periodic
–outflow	Enables the outflow boundary condition The +z face boundary condition is considered by default

Table B.2: Objects and domain parametrization options

Finally, one can tune the simulation and dumping parameters, as explain in Table B.3. By default, a 3D dumping of the fields is done, but it can be changed to 2D slices in the code if needed. The vorticity is always dumped by default. Also, the diagnostics are dumped within a single `.txt` file, while the fields are dumped at each iteration with one `.h5` file and one `.xmf` file. Those can be read with Paraview.

Flag	Explanation
–tfinal	Final time of the simulation Default value is 10
–iter-max	Maximal number of iterations of the simulation Default value is 100
–iter-dump	Frequency (in iterations) to dump the fields
–dump-vel	Enables the dumping of the velocity field.
–iter-diag	Frequency (in iterations) to dump the diagnostics
–diag-force	Enables the dumping of the forces diagnostics
–profile	Enables the profiler and writes the profiling in a `.txt` file

Table B.3: Simulation and dumping options

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain
Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | **www.uclouvain.be/epl**