Advanced immersed boundary methods for wind turbine wake modeling

Iason Tsetoglou

To cite this version:

Iason Tsetoglou. Advanced immersed boundary methods for wind turbine wake modeling. Fluid Dynamics [physics.flu-dyn]. Normandie Université, 2023. English. NNT : 2023NORMIR03 . tel-04186082

HAL Id: tel-04186082
https://theses.hal.science/tel-04186082v1

Submitted on 23 Aug 2023

THÈSE

Pour obtenir le diplôme de doctorat

Spécialité MECANIQUE DES FLUIDES, ENERGETIQUE, THERMIQUE, COMBUSTION, ACOUSTIQUE

Préparée au sein de l'INSA Rouen Normandie

Advanced immersed boundary methods for wind turbine wake modeling

Présentée et soutenue par
IASON TSETOGLOU

Thèse soutenue le 22/03/2023 devant le jury composé de		
M. STÉPHANE GLOCKNER	INGENIEUR DE RECHERCHE HDR, ENSCBP - I2M	Rapporteur
M. PIERRE BENARD	MAÎTRE DE CONFÉRENCES, Institut national des sciences appliquées de Rouen	Membre Co-encadrant
M. LUDOVIC CHATELLIER	MAITRE DE CONFERENCES DES UNIVERSITES HDR, Université de Poitiers - ISAE-ENSMA	Membre
M. FRANCK NICOUD	PROFESSEUR DES UNIVERSITÉS, Université de Montpellier	Membre
M. MARCELLO MELDI	PROFESSEUR DES UNIVERSITÉS, ENSAM Lille	Président du jury
M. JULIEN REVEILLON	PROFESSEUR DES UNIVERSITÉS, Université de Rouen Normandie	Directeur de thèse

Thèse dirigée par JULIEN REVEILLON (COMPLEXE DE RECHERCHE INTERPROFESSIONEL EN AEROTHERMOCHIMIE)

Abstract

Keywords: *Immersed Boundary Methods, Implicit Volume Penalty, Volume-of-Solid, Large-Eddy Simulations, Computational Fluid Dynamics, Finite Volume, Wind Turbines*

Many engineering applications involve fluid flows around moving complex solid objects. An example can be found in renewable energy production units such as wind turbines. Computational Fluid Dynamics (CFD) help us simulate numerically such flows. Immersed Boundary methods (IBM) have proven to be an attractive tool for handling arbitrarily large displacements of complex solid bodies on a fixed grid.

In this thesis, a novel Immersed Boundary method has been developed coupling a Lagrangian Volume-of-Solid (VOS) description of the immersed body and a robust implicit volume penalty IBM embedded in the low-Mach number projection method of the YALES2 code of the CORIA laboratory, in the framework of high-fidelity unsteady Large-Eddy Simulations (LES). This method is referred to as Volume-of-Solid Implicit Volume Penalty method (VOS-IVP) and can be broken down to three components: (i) Incorporating the solid volume fraction field into the Navier-Stokes equations, allowing to define a composite velocity field describing both fluid and solid domains at once; (ii) Expressing the penalty term in an implicit manner, imposing the solid velocity inside the solid region at the same time as the incompressibility constraint is satisfied; (iii) Representing the solid object as a set of Lagrangian particles carrying the local solid volume for a robust imposition of the solid movement.

The accuracy of the method has been assessed on several academic cases with both bluff and streamlined bodies and with low to moderate Reynolds number values. From the validation cases involving flow over a stationary solid, it is shown that the implicit form of the penalty term improved the accuracy of the velocity imposition at the immersed object compared to the usual penalty method. The predicted forces acting on the solid are in excellent agreement with values found in literature and with body-fitted reference simulations, where the relative errors remain under 10%. From the validation cases involving flow over moving bodies, the aerodynamic forces are well predicted but due to the sharp penalty mask used, they suffer from high frequency oscillations. These oscillations, however, subside very fast with decreasing mesh sizes. In all cases, the near wake behaviour is reproduced very well. The system of equations is proven to be fully mass conservative in time, independently of the mesh resolution.

The computational performances also make the method an attractive option. The costs are identical with body-fitted cases of stationary solids. In the simulation of a 2D vertical axis turbine subjected to laminar flow, a speed-up factor of 1.46 was observed compared to the computational time required by the body-fitted method.

To conclude, the VOS-IVP method shows very promising results with opportunities for further improvement, such as the integration of wall-models to the penalty term and the optimisation of the solid relocation process. Future work will focus on the method's use to more real-world configurations.

Résumé

Mots-clés : *Méthodes des Frontières Immergées, Pénalisation volumique implicite, Volume-of-Solid, Simulations Grandes Échelles, Mécanique des fluides numérique, Volumes finis, Éoliennes*

Plusieurs applications en ingénierie impliquent des écoulements des fluides autour des solides mobiles. Un tel exemple se trouve sur les unités de production d'énergie renouvelable comme les éoliennes. La mécanique des fluides numérique aide à simuler les écoulements fluides. Les méthodes des frontières immergées (IBM) sont un outil attractif pour traiter les grands déplacements des corps solides complexes sur des maillages fixes.

Pendant cette thèse, une nouvelle méthode des frontières immergées a été développée en couplant une approche Volume-of-Solid (VOS) Lagrangienne pour la description du corps immergé et une méthode de pénalisation volumique implicite avec le schéma de projection pour des écoulements à faible nombre de Mach du code YALES2 du laboratoire CORIA, dans le contexte des Simulations des Grandes Échelles (LES). Cette méthode s'appelle ≪ Volume-of-Solid Implicit Volume Penalty method ≫ (VOS-IVP) et se comporte de trois axes : (i) L'utilisation du champ de la fraction volumique locale du solide dans les équations Navier-Stokes, permettant la définition du champ de la vitesse composite décrivant le domaine fluide et le domaine solide en même temps; (ii) La formulation implicite du terme de pénalisation, ce qui permet l'imposition de la vitesse solide dans la région solide au même instant avec l'imposition de la contrainte d'incompressibilité; (iii) La représentation du corps solide comme une somme des particules Lagrangiennes contenant le volume du solide locale pour une imposition robuste du mouvement solide.

La précision de la méthode a été évaluée sur plusieurs cas académiques avec des corps à la fois bluff et streamlined et avec des valeurs de nombre de Reynolds faibles à modérées. A partir des cas de validation impliquant un écoulement autour un solide stationnaire, il est montré que la forme implicite du terme de pénalisation améliore la précision de l'imposition de la vitesse à l'objet immergé par rapport à la méthode de pénalisation usuelle. Les forces prédites agissant sur le solide sont en excellent accord avec les valeurs trouvées dans la littérature et avec les simulations body-fitted de référence, où les erreurs relatives restent inférieures à 10%. D'après les cas de validation impliquant un écoulement autour des corps en mouvement, les forces aérodynamiques sont bien prédites mais, en raison du masque de pénalisation aigu utilisé, elles souffrent d'oscillations à haute fréquence. Ces oscillations, cependant, s'atténuent très rapidement avec des tailles de mailles décroissantes. Dans tous les cas, le comportement du sillage proche est très bien reproduit. Il est prouvé que le système d'équations est entièrement conservatif en masse dans le temps, indépendamment de la résolution du maillage.

Les performances de calcul font également de cette méthode une option attrayante. Les coûts sont identiques à ceux des cas body-fitted de solides stationnaires. Dans la simulation d'une turbine à axe vertical 2D soumise à un écoulement laminaire, un facteur d'accélération de $1,46$ a été observé par rapport au temps de calcul requis par la méthode body-fitted.

En conclusion, la méthode VOS-IVP montre des résultats très prometteurs avec des possibilités d'améliorations supplémentaires, telles que l'intégration de modèles de parois au terme de

pénalisation et l'optimisation du processus de relocalisation des solides. Les travaux futurs se concentreront sur l'utilisation de la méthode pour des configurations plus réelles.

Acknowledgements

First of all, I would like to thank Marcello Meldi and Stéphane Glockner for having accepted to review the following manuscript and taking the time to offer constructive comments to improve it. I extend my gratitude to Franck Nicoud and Ludovic Chatellier for having accepted to be part of the jury for my PhD defense. I found the questions and the discussion that followed my presentation very interesting and fruitful, and I appreciate the interest that you showed for my work.

Now, I would like to thank the people that mentored and guided me through the journey of this thesis. Julien Réveillon, thank you for directing this thesis and always replying promptly to all my emails. Vincent Moureau, Ghislain Lartigue and Pierre Bénard; I still remember how embarrassed I was during our interview when I couldn't remember how to write correctly the Navier-Stokes equations... Despite that, you chose to invest in me and gave me the opportunity to not only carry out a PhD at CORIA, but to also be part of a fantastic team of people. Vincent, thank you for always providing an interesting insight with your expertise when facing difficulties in developing and improving the method. I also appreciate the fact that you always made a point to mention the value of my work and its usefulness for others. Ghislain, thank you for always having us covered when faced with numerical and informatics problems. I loved the short tutorials we had in the beginning every Thursday afternoon. I would also like to thank you for the patience and the long hours you sacrificed helping me when I was stuck on vortex-induced-vibrations during the summer of my first year. Finally, Pierre, I can't thank you enough for trusting in me that I would succeed and investing so much time and effort in guiding me all these years. Thank you for always re-setting my eyes on the goal when I lost focus and for lending an ear when I needed it. Without you as my supervisor this work wouldn't have been possible.

I would like to express my gratitude to Mélody Cailler. It was your and Vincent's initial work that sparked the topic of this thesis. Thank you for trusting me to work on your ideas and turn them into the central point of my work. Knowing that you can confidently rely on our method for your research gives me great satisfaction. It was a very fruitful collaboration and I will miss our bi-weekly meetings on Tuesdays. On the topic of the method's development, I also want to thank Pierre Bénez, fellow PhD candidate, with whom I felt a sense of camaraderie as the only two people working on IBM from the lab. It was not only a pleasure that my method was useful for you own doctoral work, but you were also actively contributing to the improvement of this work with your observations and analyses. Thank you for all the encouragement and I can't wait to see you pass your PhD defense.

Of course, I am thankful to the whole YALES2 team; previous PhD students that welcomed me to the team, like Francesco, Yann, Patricia and Romain, my fellows Pierre, Adrien and Serge that started the same time as me and the many other people that joined along the way, Etienne, Simone, François, Léa, Julien, Yacine, Antoine, Ibtissam. I would also like to thank Kévin for always being welcoming and for also helping me back in 2017 (when you were still doing your thesis) to decide to do a PhD. I would like to give special thanks to Félix, with whom I shared the office for two and a half years and who was always eager to help me and always made the office a pleasant place to be. Lastly, I am happy to be part of the YALES2 community, a group of remarkable people

from all over Europe, who continuously seek to innovate and push the boundaries of the research in numerical fluid flow modelling, and many of whom I have had the chance to meet and discuss with during the Extreme CFD Workshops organized every year.

Last but not least, I would like to thank my family and friends. I thank my mother and father for always supporting me in my endeavors. I thank my brother and my friends, especially Elias and Maria, for their help and encouragement during various hardships in these three years. I would also like to express my heartfelt gratitude to my grandparents for the constant support and love. In particular, I dedicate this thesis to my late grandfather, Iason, thanks to whom it was possible for me to pursue my studies in France.

And of course, a heartfelt thank you to my dear Mariana; your support was monumental during the last three years.

Contents

Chapter 1

General Introduction

Contents

1.1 Context

In recent years the average global temperature of the Earth has increased alarmingly as shown in Fig. 1.1. Climate scientists believe that one of the main factors in global warming is the dramatic increase of greenhouse gas emissions due to human activities. Fig. 1.2 reports data on the greenhouse gas production during the past century. The energy production sector constitutes a major contribution to these emissions as its majority is based on fossil fuel combustion which releases CO_2. These energy sources contributed significantly in the global economic growth after the industrial revolution but their impact on the global surface temperature can no longer be ignored. Under the 2015 Paris Agreement, nations collectively agreed to keep global warming under 2°C and pursue efforts to limit the increase even further to 1.5°C. To achieve this, the use of renewable energy sources is heavily promoted. Fig. 1.2 shows the renewable energy consumption during the past few decades by source. One can notice the significant rise of demand for wind energy. Indeed, wind energy is expected to grow significantly in the coming decades playing a major role in the energy transition strategy towards low-carbon electricity production systems in order to mitigate climate change.

Wind energy applications present a wide range of spatial and temporal flow scales, as described in Porté-Agel et al. [81] and illustrated in Fig. 1.3. When examining the aerodynamics around the

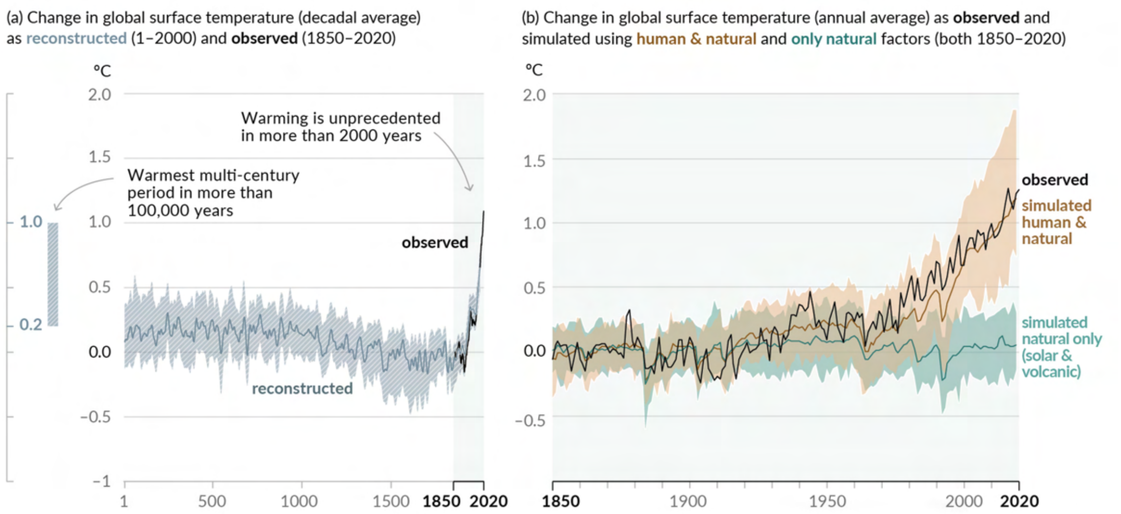
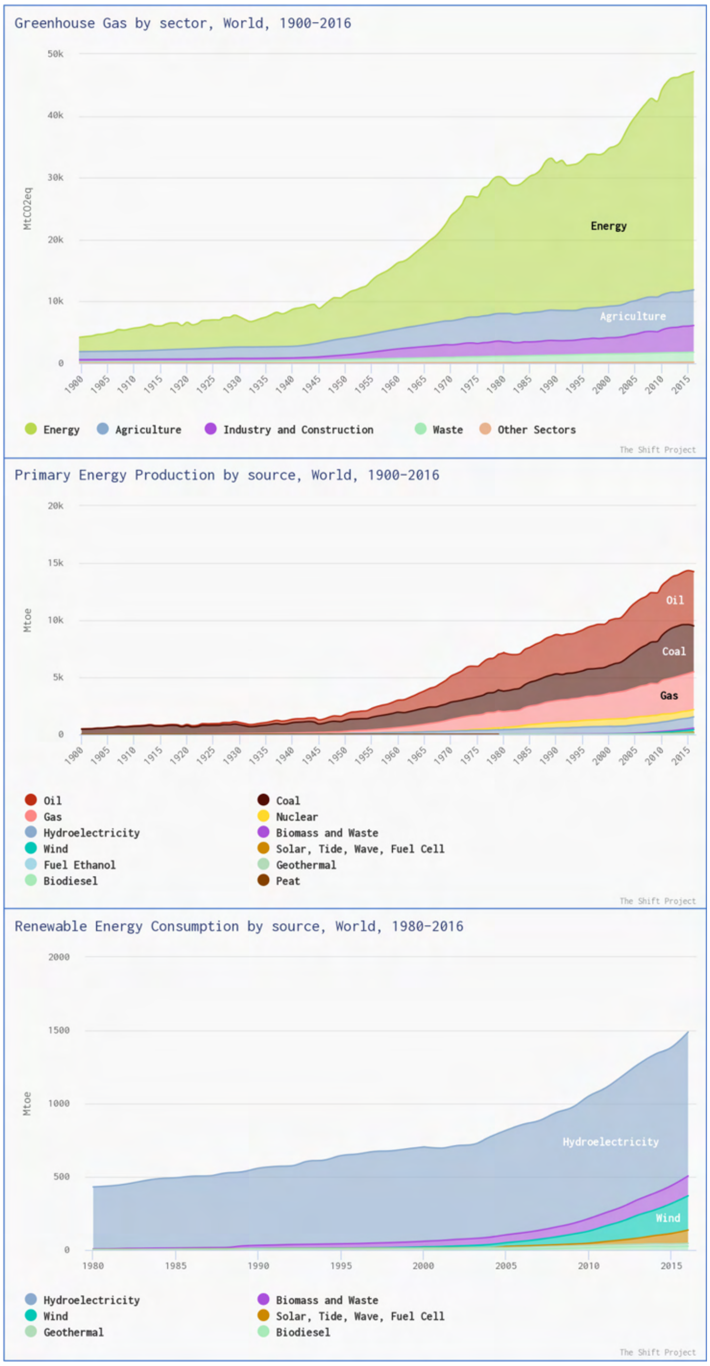


Figure 1.1: Changes in global surface temperature (°C) relative to 1850-1900, taken from [31].

rotor blades, the characteristic space-time scales are in the order of millimetres and milliseconds, respectively, but on a largest scale when looking at wind-farms and atmospheric effects, the scales can reach the order of tens of kilometres and multiple hours or even days. The multi-scale nature of the interaction between atmospheric turbulent flows and wind energy production units makes the study of wind energy harvesting particularly challenging. In recent years, experimental, analytical and numerical methods have been used in order to study the interaction of the atmospheric boundary layer flow with a wind turbine or multiple turbines in the case of wind farms. Field and wind-tunnel experiments provide valuable information on the structure and dynamics of the flow around wind turbines and wind farms. They also provide datasets for the validation of analytical and numerical models. Analytical models are derived from basic equations governing the conservation of flow properties and provide simple predictions of the average velocity deficit in the turbine wake. Their low computational cost makes them useful for wind farm control and layout optimisation over flat terrain. Finally, Computational Fluid Dynamics (CFD) provide numerical simulations of the complex turbulent flow around wind turbines. There exists a wide variety of techniques used for wind turbine wake prediction at varying degrees of accuracy. Accuracy, however, comes with higher computational costs. An illustration of the many numerical modelling approaches may be seen in Fig. 1.4.

In the context of high-fidelity numerical simulations there exist two main approaches to represent wind turbines [81]. The first way relies on resolving the rotor blades geometry and the boundary layer around them [7, 82]. This means taking into account the exact geometry of the blades in the computational mesh making this approach capable of predicting well the aerodynamic loads on the blades and the near wake behaviour. This comes with a high computational cost making it an unattractive option for studying the far wake or simulating multiple turbines in the context of wind farm simulations. The other approach, is to model the rotor blades using actuator-type methods [88, 25]. This allows to reduce the computational cost by losing accuracy in


Greenhouse Gas by sector, World, 1900-2016

Primary Energy Production by source, World, 1900-2016

Renewable Energy Consumption by source, World, 1980-2016

Figure 1.2: Greenhouse gas emissions by sector (top), energy production by source (middle) and renewable energy consumption by source (bottom), taken from: www.theshiftdataportal.org.

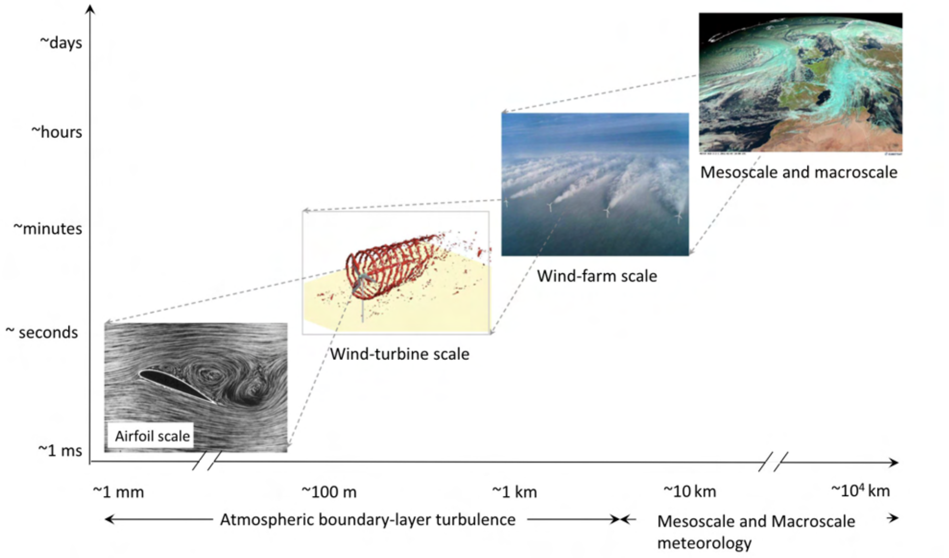


Figure 1.3: Schematic illustrating the wide range of flow scales relevant to wind energy: from the turbine blade scale to the meteorological mesoscale and macroscale, taken from [81].

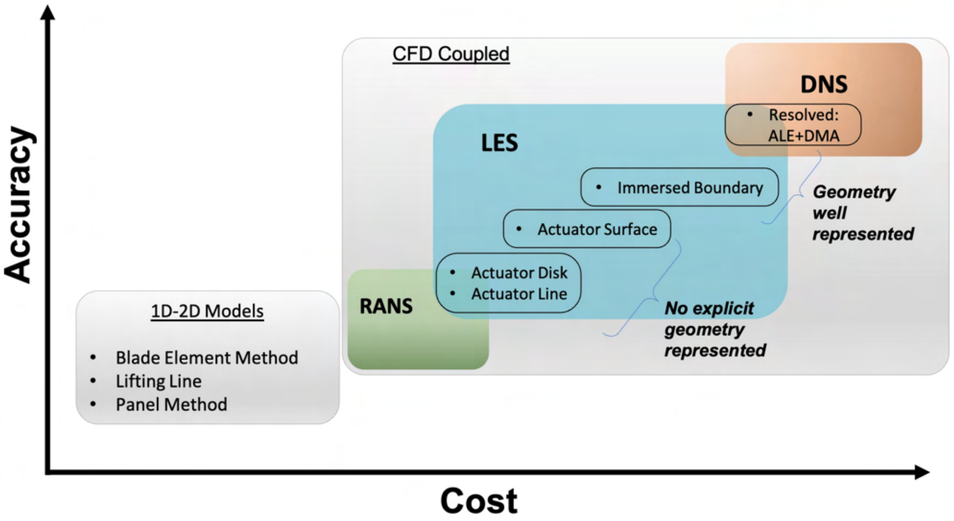


Figure 1.4: Schematic to illustrate the variety of methods used in wind turbine simulations. Computational cost tends to increase with higher fidelity techniques.

the near wake, making the simulation of turbines interacting with the atmospheric boundary layer or with other turbines in large computational domains more affordable.

1.2 State-of-the-art rotor modelling approaches

In CFD, computers are used to perform the necessary calculations to simulate the flow behaviour of a fluid in a particular configuration and the interaction of the fluid with surfaces. CFD simulations constitute an important way of gaining a better understanding of wind turbines and wind farm flows [81]. The differential equations that need to be solved in an incompressible approach to simulate the fluid flow are the Navier-Stokes equations:

$$\nabla \cdot \boldsymbol{u} = 0\,, \tag{1.1}$$

$$\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t} = -\frac{1}{\rho}\nabla P + \frac{1}{\rho}\nabla \cdot \boldsymbol{\tau} + \boldsymbol{f}_v\,, \tag{1.2}$$

where $\boldsymbol{u}$ is the fluid velocity, P the pressure, ρ the fluid density, $\boldsymbol{\tau} = \mu\left(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T\right)$ the viscous stress tensor, μ the fluid dynamic viscosity. The forcing term $\boldsymbol{f}_v$ represents volumetric body forces as momentum source terms depending on the physics involved in each case, such as the gravitational force.

1.2.1 Actuator Line Method (ALM)

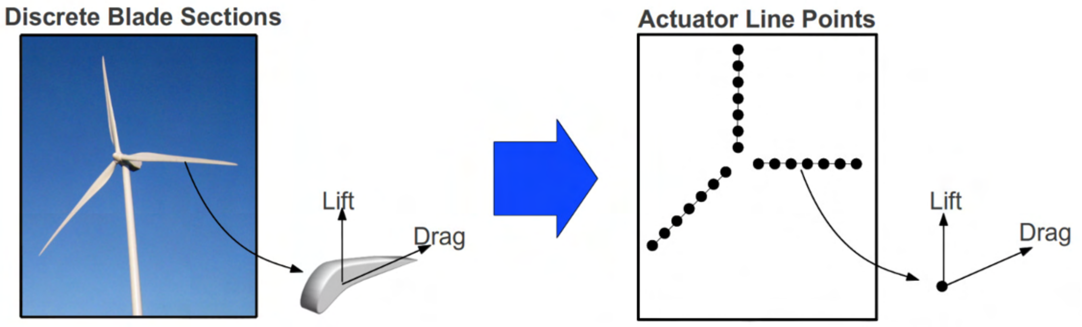


Figure 1.5: Schematic of a turbine blade being represented by an actuator line model [62].

The Actuator Line Method models the wind turbine blades as a series of discrete blade sections along the blade axis [77] as shown in Fig. 1.5. The force applied to the flow at each instance by the blade is calculated thanks to tabulated data of the airfoil lift and drag coefficients. This force is then applied to the flow as an added momentum source in the Navier-Stokes equations and is distributed along the lines representing the aerodynamic loads of the rotating blades [28]. Hence, the ALM does not fully resolve the geometry of the blades and their impact is prescribed.

The local force acting on each blade element is calculated thanks to the blade element theory [20] as:

$$\mathbf{f} = \frac{1}{2}\rho U_{\mathrm{rel}}^2 cw(\boldsymbol{C}_L(\alpha)\boldsymbol{e}_L + \boldsymbol{C}_D(\alpha)\boldsymbol{e}_D)\,, \tag{1.3}$$

where C_L and C_D are the lift and drag coefficients of the blade profile at the local angle of attack α with a local relative velocity to the element U_{rel}. c is the local chord of the local profile and w the actuator element width.

This force is then projected onto the Eulerian grid using a convolution:

$$\boldsymbol{f}_\varepsilon = \mathbf{f} \otimes \eta_\varepsilon\,, \tag{1.4}$$

where the mollification function η_ε is a Gaussian kernel:

$$\eta_\varepsilon(d) = \frac{1}{\varepsilon^3 \pi^{3/2}} \exp\left[-(d/\varepsilon)^2\right]\,, \tag{1.5}$$

with d being the distance between the grid node and the actuator element position and ε the mollification width. Finally, the forcing term $\boldsymbol{f}_\varepsilon$ is added to the momentum equation (Eq. 1.2) into $\boldsymbol{f}_v$.

As described by Houtin et al. [28] based on previous studies [90], a minimum grid-spacing of $\Delta x = c/10$ would suffice to provide a smooth distribution of the force on the grid obtained by the forces concentrated at each actuator element.

1.2.2 Arbitrary Lagrangian Eulerian method (ALE)

The Arbitrary Lagrangian Eulerian method (ALE) is a body-fitted approach where the fluid flow is resolved near the solid surface [12]. To account for the arbitrary large movement of a solid, deformable control volumes are used, whose nodes move as Lagrangian points to follow the surface. In the framework of moving control volumes, if the grid nodes move with a prescribed velocity $\mathbf{w}$, for a physical variable ϕ we have the following relation:

$$\int_{V(t)} \frac{\partial \phi}{\partial t} \mathrm{d}V = \frac{\mathrm{d}}{\mathrm{d}t} \int_{V(t)} \phi \mathrm{d}V - \int_{V(t)} \nabla \cdot \phi \mathbf{w} \mathrm{d}V\,. \tag{1.6}$$

The flow equations in this method need to describe the evolution of the material velocity $\mathbf{u}$. Hence the incompressible Navier-Stokes equations are rewritten as:

$$\nabla \cdot \mathbf{u} = 0\,, \tag{1.7}$$

$$\frac{\partial \mathbf{u}}{\partial t} + ((\mathbf{u} - \mathbf{w}) \cdot \nabla)\mathbf{u} = -\frac{1}{\rho}\nabla P + \frac{1}{\rho}\Delta \mathbf{u} + \boldsymbol{f}_v\,. \tag{1.8}$$

This method's development in the YALES2 code used in this work has been documented in detail in the PhD thesis of Fabbri [15].

As explained, the grid nodes near the geometry need to move according to the solid motion, but the nodes further away do not. This separated the computational domain into a moving zone and a stationary zone. Between them, there is a transition zone where the control volumes are deformed since some of their nodes need to move at different speeds. This distorts the mesh and deteriorates the mesh quality. To solve this problem, a new mesh is needed, adapted to the new position of the geometry and conserving a good mesh quality. For this, a Dynamic Mesh Adaptation (DMA) process can be used. An example of this process is illustrated in Fig. 1.6. The constant re-meshing needed for the ALE to work, however, adds a considerable amount of computational effort and cost to the simulation, making this method unattractive for industrial uses despite its accurate prediction capabilities.

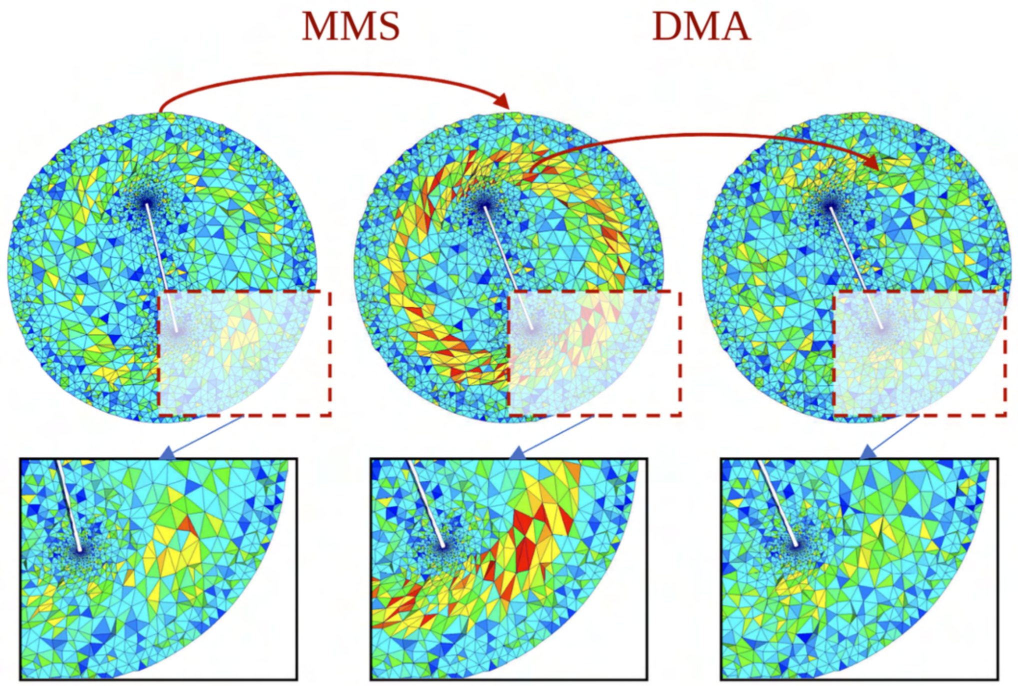


Figure 1.6: Case of a rotating plate in a cylindrical domain. The plate rotates and the Mesh Movement Solver (MMS) is used to rotate the nodes in the region near the plate. This creates the transition zone between the moving and stationary mesh zones where the cell sizes are stretched. Once a threshold amount of deformation is reached, the DMA is activated to generate a new mesh [15].

1.2.3 Comparison

The ALM requires prior knowledge of the aerodynamic coefficients of the blades to compute the local forces of the blades and the aerodynamic loads of the turbine blades are prescribed onto the fluid flow via a momentum source term. This technique does not represent the full geometry of the blades and fails to predict accurately the flow near the rotor. Its benefits lie on the use of a static computational grid with coarse mesh sizes relative to the blade length, which in turn makes the computations fairly cheap to run. On the other hand, the ALE method relies on fully resolving the solid geometry in motion to produce accurate prediction of the flow behaviour near the solid surface. To achieve this, the grid nodes near the surface need to move constantly alongside the solid geometry while node further away stay stationary. This stretches the mesh between the moving and the stationary nodes, deforming the computational cells. To remedy this, a dynamic mesh adaptation process is necessary to recreate the stretched mesh. This process, however, proves to be very computationally demanding.

1.3 An alternative approach: Immersed Boundary Method (IBM)

Immersed Boundary methods (IB or IBM) [78] represent an attractive alternative for simulating FSI problems involving complex geometries and arbitrarily large movements. The basic idea of the IBM is to carry out the simulation on a relatively simple mesh which extends inside of the region where the solid geometry would be and introduces a forcing term in the fluid's governing equations to simulate the interaction between the solid and the fluid (Fig. 1.7). Thus, one is able to impose boundary conditions on surfaces that are not aligned with the grid and to simulate arbitrarily large movements of complex solid boundaries on a fixed grid. The key point of these methods is how to treat the forcing term imposed at the immersed interface so that it does not degrade the precision and the conservation properties of the solver while accurately representing the fluid-solid interaction. Since the first introduction of the IBM, a wide variety of techniques have been developed to properly model the forcing term. A literature review on IBM will be presented in section 2.

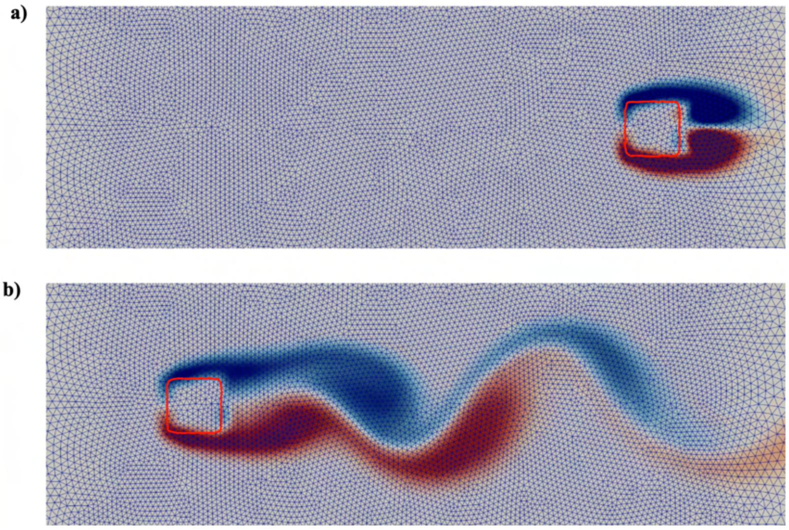


Figure 1.7: Vorticity field in case of a 2D square is impulsively moved on static mesh at two different time instances. The surface of the square, where the forcing term of the IBM is applied, is represented by a red line. The mesh is static during the simulation.

1.4 Thesis objective

The CORIA laboratory in Rouen houses their original CFD code, named YALES2, where the methods mentioned in sections 1.2.1 and 1.2.2 have been developed through the years and been used in simulating wind turbine flows, showcasing the strengths and weaknesses of both methods. The work of this thesis is motivated by the idea of using an alternative method that could combine the positive aspects of both the ALM and the body-fitted ALE approaches. This would result in the accurate prediction of the aerodynamic flow near the rotor of the wind turbine while avoiding the modelling associated with the ALM (tabulated aerodynamic coefficients, dynamic stall models, etc.). Hence, the aim of the present thesis is to develop an advanced Immersed Boundary Method for the simulation of wind turbine flows.

The method developed in this work serves to simulate the impact of a solid body, stationary or mobile, to the fluid flows dynamics by coupling two different approaches: the Volume-of-Solid approach and the volume penalty IBM. This method is referred to as Volume-of-Solid Implicit Volume Penalty method (VOS-IVP) and its development can be broken down to three tasks:

1. Coupling the VOS approach with an IBM
2. Modelling the implicit penalty forcing term
3. Representing the solid body in a Lagrangian manner

The development of the method will be explained in detail in section 4.

1.5 Manuscript content

This manuscript is composed of the following chapters:

- **Chapter 1: General introduction**

  The present chapter serves as a general introduction to the subject of this thesis and the challenges of the study of wind turbine rotors. The scientific context and motivation of the thesis is presented.

- **Chapter 2: Literature review on Immersed Boundary Methods**

  This chapter serves as an overview of the techniques included in the ‘Immersed Boundary Methods’ (IBM). What differentiates every technique is the way we model and compute the forcing term used to represent the influence of an immersed object to the fluid dynamics. Two main approaches exist: continuous forcing IBM and discrete forcing IBM. The most frequent methods derived from these approaches will be explained and finally a critical comparison between them will be made.

- **Chapter 3: Numerical modelling of turbulent flows**

  This chapter will introduce the reader to the principles of Computational Fluid Dynamics (CFD) and the basics of modelling turbulent flows. The governing equations of fluid dynamics will be presented first and then the notion of turbulence will be explained. The three basic approaches of handling turbulence will be presented: Direct Numerical Simulation, Reynolds-Averaged Navier-Stokes and Large-Eddy Simulation. The two most well-known methods of describing numerically interfaces between different phases will be briefly explained: Volume-of-Fluid method and Level-Set method. Finally, the main characteristics of the in-house CFD solver YALES2 that was used throughout this thesis will be presented.

- **Chapter 4: Volume-of-Solid Implicit Volume Penalty Method**

  In this chapter, the development of a novel IBM technique will be presented in detail. This method couples a Lagrangian Volume-of-Solid (equivalent of Volume-of-Fluid from the solid phase perspective) description of the body and a robust implicit volume penalty IBM technique. First, the derivation of the new set of governing equations due to the aforementioned coupling will be shown. The implicit volume penalty approach will then be examined in more detail. The chapter then proceeds to explain the procedure with which we represent the immersed solid object, how the motion of the object is prescribed and what are the constraints in the creation of the solid object. It ends with an algorithmic presentation of how the method is implemented into the YALES2 library.

- **Chapter 5: Validation**

  This chapter focuses on the validation of the VOS-IVP method through a series of well known academic configurations with both stationary and moving immersed bodies. The numerical results are compared against body-fitted simulations and reference data found in literature. The comparison includes the ability to accurately predict the aerodynamic loads on the body as well as the correct prediction of the wake behaviour. The accuracy of the method is also evaluated depending on the different parameters of the method, such as the grid spacing, the time-step and the penalty parameter. Finally, the computational performance of the method is also compared to body-fitted simulations.

- **Chapter 6: Application**

  Chapter 6 presents more applications of the VOS-IVP method, with more complex configurations, to further prove its ability to accurately predict the impact of the immersed solid to the fluid dynamics.

- **Chapter 7: Conclusion and perspectives**

  Finally, the last chapter will include concluding remarks on the method developed during the thesis and offer perspectives for future works and improvements.

Chapter 2

Literature review on Immersed Boundary Methods

Contents

2.1 Introduction

Immersed Boundary methods (IB or IBM) represent an attractive approach for simulating flow problems involving complex geometries and arbitrarily large movements. The term “immersed boundary method” was first used in reference to a method developed by Peskin (1972) to simulate cardiac mechanics and associated blood flow [78]. The novel feature of that method was that the entire simulation was carried out on a Cartesian grid without conforming to the heart’s geometry and that a procedure was formulated to impose the effects of the immersed boundary on the flow. The basic idea of the IBM is to carry out the simulation on a relatively simple mesh which extends inside of the region where the solid geometry would be and introduce a forcing term in the fluid's governing equations to simulate the interaction between the solid and the fluid. Thus, we are able to impose boundary conditions on surfaces that are not aligned with the grid and to simulate arbitrarily large movements of complex solid boundaries on a fixed grid. The key point of these methods is how to treat the forcing term imposed at the immersed interface so that it does not degrade the precision and the conservation properties of the solver while accurately representing the fluid-solid interaction. Since first introduced by Peskin for modelling blood flow through a beating heart, IBM has been extended to various applications in scientific and engineering fields (e.g. study of biological locomotion, as in Fig. 2.1, or the mixing process in a stirred-tank, as in Fig. 2.2).

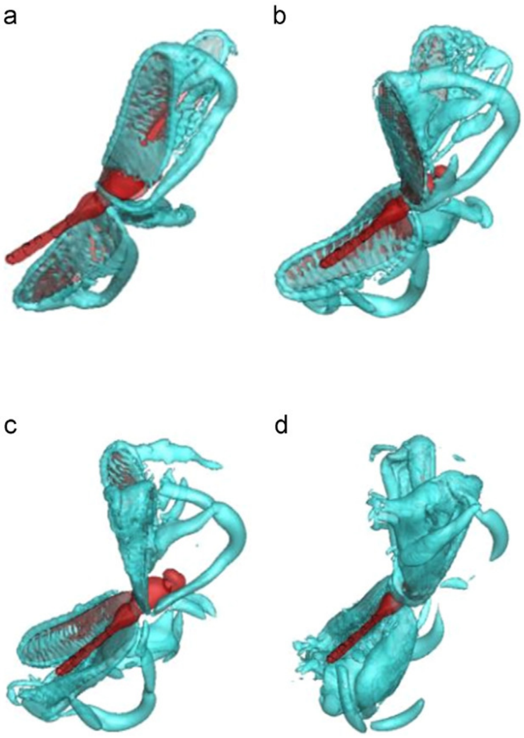


Figure 2.1: IBM used to visualise the wake structures of a dragonfly in take-off motion. Extracted from [13].

Numerous modifications and refinements have been proposed and a number of variants of this approach now exist focusing on the definition of the forcing term. In this section we will present the main governing equations of IB methods and the general terminology used. An overview of the different approaches of IB methods will also be presented based on previous reviews [66, 63, 91, 29, 42].

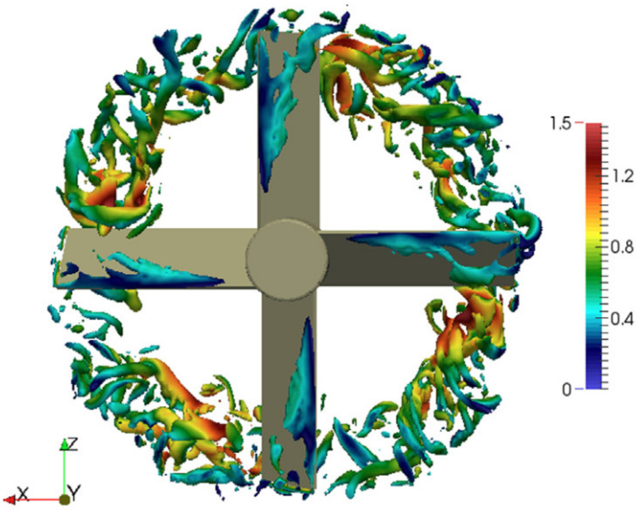


Figure 2.2: IBM used for mixing in a stirred-tank. Q-criterion iso-countours with $Q = 10^4$ coloured by the velocity magnitude. Extracted from [92].

2.2 Governing equations

The main advantage of the IBM is that body-conforming meshes are not necessary. Instead, the computational domain includes both fluid and solid domains, $\Omega = \Omega_f \cup \Omega_s$ (Fig. 2.3), the interface of which will be denoted as Γ. Thanks to the continuous mesh in the solid region, there is no need for re-meshing in the case of a moving immersed object, which makes it an attractive alternative for simulating moving bodies in fluid flow.

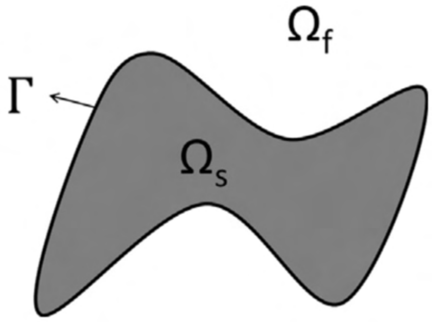


Figure 2.3: Schematic presenting the fluid and solid domains and their interface Γ. Extracted from [91].

To model the interaction between fluid and solid at the interface Γ, an extra volume force term $\boldsymbol{f}_\Gamma$ is added to the incompressible Navier-Stokes momentum equation as shown in Eq. 2.2. This term is evaluated in such a way as to obtain the desired solid velocity $\boldsymbol{u_s}(\boldsymbol{X_\Gamma}, t)$ on the immersed boundary. In the IBM context, the governing equations are expressed as:

$$\nabla \cdot \boldsymbol{u} = 0\,, \tag{2.1}$$

$$\frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot (\boldsymbol{u} \otimes \boldsymbol{u}) = -\frac{1}{\rho}\nabla P + \frac{1}{\rho}\nabla \cdot \boldsymbol{\tau} + \boldsymbol{f}_\Gamma\,, \tag{2.2}$$

where $\boldsymbol{u}$ is the fluid velocity, P the pressure, ρ the fluid density, $\boldsymbol{\tau} = \mu\left(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T\right)$ the viscous stress tensor, μ the fluid dynamic viscosity. The expression used for the forcing term $\boldsymbol{f}_\Gamma$ may vary.

Indeed, the imposition of boundary conditions on the IB is the key factor in developing an IB algorithm and is achieved through the additional forcing term in Eq. 2.2. This introduction of a forcing function can be implemented in two different ways, which leads to the fundamental dichotomy of IBM: continuous forcing approaches and discrete forcing approaches.

Continuous forcing approaches integrate the forcing term into the continuous Navier-Stokes equations for the whole domain Ω before the discretization happens. They include the first IBM developed by Peskin [78] and other methods derived from this like the one described by Goldstein et al. [21]. In discrete forcing approaches the governing equations are first discretized without taking into account the immersed boundary, then only in discretized cells near the interface Γ do we introduce a forcing term (Mohd-Yusof [67], Udaykumar et al. [96], Kim et al. [41], Fadlun et al. [16]). The forcing term is calculated in a way that compensates the difference between the predicted velocity and the desired imposed velocity at the IB.

The general advantage of continuous forcing is that the formulation is almost independent of the discretization scheme, which makes its integration easier to a pre-existing Navier-Stokes solver. The opposite is true for the discrete forcing methods since the forcing term is dependant on the spacial discretization. The advantage of the latter method is that its formulation allows for a more precise description of the immersed boundary compared to former method.

2.3 Continuous forcing

In this family of methods, elastic and rigid boundaries require different treatments which will be discussed separately. For elastic boundaries we will examine the “classic” IBM developed by Peskin [78], while for rigid boundaries we will examine the “feedback approach” proposed by Goldstein [21]. Finally, another widely used method, the penalisation method [2], will be presented.

2.3.1 Classic IBM

The classic IB method, originally proposed by Peskin for the coupled simulation of blood flow and muscle contraction in a beating heart, is best suited to fluid-structure simulations with elastic boundaries. The fluid flow is governed by the incompressible Navier-Stokes equations solved on a stationary Cartesian grid. The IB is represented through a set of elastic fibers whose location is tracked in a Lagrangian fashion by mass-less points moving with the local fluid velocity. All these points on the interface of coordinates $\boldsymbol{X}_\Gamma$ are governed by the Lagrangian equation:

$$\frac{\partial \boldsymbol{X}_\Gamma}{\partial t} = \boldsymbol{u}(\boldsymbol{X}_\Gamma, t)\,. \tag{2.3}$$

The stress and deformation of the elastic fibres are calculated based on the appropriate constitutive laws and can be represented through an operator describing the properties of the boundaries:

$$\boldsymbol{F}(\boldsymbol{X}_\Gamma, t) = L(\boldsymbol{X}_\Gamma, t)\,. \tag{2.4}$$

Such a law could be the well-known Hooke's law. The forces from the IB to the fluid are calculated by distributing the fibre stresses to the fluid using a forcing term in the momentum equation. Hence, due to the fibres, the forcing term at an arbitrary point $\boldsymbol{x} \in \Omega$:

$$\boldsymbol{f}(\boldsymbol{x}, t) = \int_\Gamma \boldsymbol{F}(\boldsymbol{X}, t)\delta(\boldsymbol{x} - \boldsymbol{X})\,\mathrm{d}\boldsymbol{X}\,, \tag{2.5}$$

where δ is the Dirac delta function. Since the location of the fibres rarely coincides with the nodes of the Cartesian grid, the forcing is distributed over a number of cells around the Lagrangian points' locations. The sharp delta function is then replaced by a smooth distribution function d, suitable for discrete meshes. Thus the previous equation is modified to:

$$\boldsymbol{f}(\boldsymbol{x}, t) = \int_\Gamma \boldsymbol{F}(\boldsymbol{X}, t)d(\boldsymbol{x} - \boldsymbol{X})\,\mathrm{d}\boldsymbol{X}\,. \tag{2.6}$$

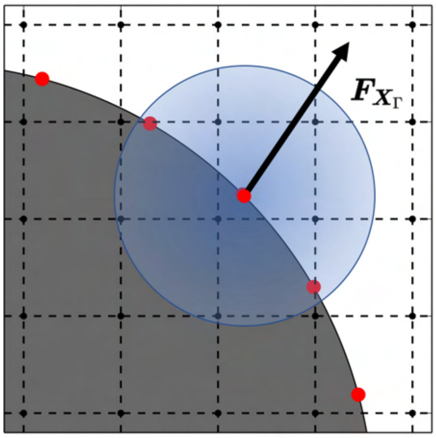


Figure 2.4: Force distribution $\boldsymbol{F}$ from a Lagrangian point (red) to the surrounding mesh nodes (black).

A schematic representation of the force distribution from one Lagrangian point to the fluid nodes can be seen in Fig. 2.4.

The fibre velocity in Eq. 2.3 at the Lagrangian points can be then computed from the ambient fluid velocity at the surrounding mesh nodes using the same distribution function.

$$\boldsymbol{u}_{\mathrm{IB}}(\boldsymbol{X}_\Gamma, t) = \int_\Omega \boldsymbol{u}(\boldsymbol{x}, t) d(|\boldsymbol{x} - \boldsymbol{X}|)\, \mathrm{d}\boldsymbol{x} \tag{2.7}$$

The key ingredient of this method is the choice of the distribution function d. Different functions have been employed through the years to better preserve the accuracy of the spacial scheme. Methods of this type have been successfully implemented in a variety of applications with elastic bodies such as cardiac mechanics, aquatic animal locomotion and bubble dynamics.

2.3.2 Feedback approach

As mentioned before, the previous method is well suited for elastic boundaries but it presents problems when applied in rigid boundary configurations because the constitutive laws for elastic bodies is not well posed for the rigid limit. One way to tackle this problem is to consider the body as elastic but extremely stiff. An other approach is to consider the Lagrangian points of the interface attached to an equilibrium location by a spring with a restoring force $\boldsymbol{F}$.

$$\boldsymbol{F}_\Gamma = -\kappa(\boldsymbol{X}_\Gamma - \boldsymbol{X}_\Gamma^{eq})\,, \tag{2.8}$$

where κ is a positive spring constant and $\boldsymbol{X}_\Gamma^{eq}$ is the equilibrium location of the interface.

If we want to accurately impose the boundary condition at the rigid IB, κ needs to take large values. This however makes the system of equations stiff and is subject to severe stability constraints. If lower values of κ are taken, it can lead to spurious elastic effects like excessive deviation from the equilibrium location [50].

This method can be viewed as a specific version of the method developed by Goldstein et al. [21]. This method seeks to model the effect of the rigid body on the surrounding flow through a forcing term with two components, one resembling a spring feedback force and one relative to damping:

$$\boldsymbol{f}(\boldsymbol{x}, t) = \underbrace{\alpha \int_0^t \left[\boldsymbol{u}\left(\boldsymbol{x}, t'\right)\right] - \boldsymbol{u}_0\left(\boldsymbol{x}, t'\right)\right] \mathrm{d}t'}_{\boldsymbol{f}_\alpha(\boldsymbol{x},t)} + \underbrace{\beta \left[\boldsymbol{u}(\boldsymbol{x}, t) - \boldsymbol{u}_0(\boldsymbol{x}, t)\right]}_{\boldsymbol{f}_\beta(\boldsymbol{x},t)}\,. \tag{2.9}$$

The constants α and β take negative values. The term $\boldsymbol{u}_0$ is the velocity of the IB, and can characterise mobile and deforming bodies thanks to its dependence on space and time.

The Eq. 2.9 should lead to $\boldsymbol{u} = \boldsymbol{u}_0$ at the IB. The first term $\boldsymbol{f}_\alpha(\boldsymbol{x}, t)$ seeks to minimise (even cancel) the difference between $\boldsymbol{u}$ and $\boldsymbol{u}_0$. The second term $\boldsymbol{f}_\beta(\boldsymbol{x}, t)$ can be interpreted as the resistance of the surface to take take a velocity value different than $\boldsymbol{u}_0$, similar to the spring restoring force previously mentioned.

To better understand this forcing term in an intuitive manner let us take only the first term in the LHS and the last term of the RHS of Eq. 2.2. Let us also replace the velocities by a target velocity field independent of time, $\boldsymbol{u'} = \boldsymbol{u} - \boldsymbol{u}_0$. This lead us to the following equation:

$$\frac{\partial^2 \boldsymbol{u}'}{\partial t^2} - \beta \frac{\partial \boldsymbol{u}'}{\partial t} - \alpha \boldsymbol{u}' = \boldsymbol{0}\,. \tag{2.10}$$

This equation behaves as a dampened oscillator or a mass-spring system whose parameters α and β characterise the frequencies of oscillation. This translates to the process: when $\boldsymbol{u}$ at the IB becomes different from the velocity $\boldsymbol{u}_0(\equiv \boldsymbol{u}_{\mathrm{IB}})$, the forcing term $\boldsymbol{f}$ activates to bring the velocity back to the reference $\boldsymbol{u}_0$.

To get correct results, the parameters need to be relatively constraining. Indeed, the oscillation frequency of the forcing term must be higher (in other words, the time of oscillation must be lower) than the dynamics of the physical phenomena. This can pose stability problems. The temporal integration of the forcing term is subject to severe restrictions, which leads to higher computational times. Goldstein et al. carried out a stability analysis and found that, when the forcing term is calculated explicitly with an Adams-Bashforth scheme, the time-step stability limit takes the form of:

$$\Delta t < \frac{-\beta - \sqrt{\beta^2 - 2\alpha k}}{\alpha}\,, \tag{2.11}$$

where k is a constant of order of magnitude of 1.

This stability study shows the importance of choosing well suited values for the parameters α, β. If these values are not sufficient to satisfy the forcing condition ($\boldsymbol{u} \approx \boldsymbol{u}_0$), the velocity field risks to deviate from the target velocity field. This method needs a compromise between the restrictive time step and the satisfaction of the desired boundary conditions. This restriction on the time-step makes it prohibitive for 3D LES applications.

2.3.3 Penalty method

The penalisation IB method is also based in the modification of the initial equations in order to maintain a target velocity field at the immersed boundary [2] similar to the feedback method. The solid domain is modelled as a porous medium immersed in the fluid. This porous characteristic is translated by the following forcing term:

$$\boldsymbol{f}(\boldsymbol{x}) = \frac{\chi_s}{\eta}[\boldsymbol{u}_{\mathrm{IB}}(\boldsymbol{x}) - \boldsymbol{u}(\boldsymbol{x})]\,. \tag{2.12}$$

Examining the force equation, the penalty term is proportional to the velocity of the fluid, and is applied at the grid points near the boundary. This extra term effectively creates a ‘repulsive force’ that pushes the fluid away from the boundary, preventing it from flowing through the immersed boundary. Comparing Eq. 2.12 to Eq. 2.9, the penalty method can be seen as a simplified version of the feedback approach utilising only the damping force.

Here, χ_s is the mask function with value equals to 1 inside the solid structure and 0 within the fluid. This mask can have either a smooth or a sharp profile. A smooth mask helps distribute smoothly the forces over the immersed boundary. One could use for example a smooth Heaviside function of a level-set function $\chi_s = \mathrm{H}(\Phi_s)$ like in [1]. Another example of a smooth mask is the following [91]:

$$\chi_s = \begin{cases} 0, & d < -\epsilon \\ \frac{1}{2}\left(1 + \frac{d}{\epsilon} + \frac{1}{\pi}\sin\left(\pi\frac{d}{\epsilon}\right)\right), & |d| \leq \epsilon \\ 1, & d > \epsilon \end{cases} \tag{2.13}$$

where, d is the signed distance from the fluid-structure interfaces and ϵ is the mollification length, which usually spans 4-5 grid nodes for numerical stability. On the other hand, sharper representations of the interface can be achieved by activating this mask only on the solid domain [27, 38, 92]. For this, the mask usually takes the binary values of 0 and 1 as follows:

$$\chi_s = \begin{cases} 0, & \boldsymbol{x} \in \Omega_f \\ 1, & \boldsymbol{x} \in \Omega_s \end{cases} \tag{2.14}$$

The parameter η, appearing in Eq. 2.12, is called the penalisation parameter and is specified for each simulation. This parameter dictates how 'fast' we impose the solid boundary velocity to the fluid as we approach the immersed boundary and acts as a permeability factor for the solid. This can also be an indication to the stiffness of the system. This parameter cannot be too large in order to avoid very stiff systems. Actually, the main inconvenience of this method lies in the obligation to conserve a time-step proportional to the permeability of the solid ($\Delta t \approx \eta$) to maintain numerical stability [38]. Thus, in the case of an immersed solid whose penalization/permeability parameter tends to zero, there is a compromise to be made between describing accurately the geometry ($\eta \rightarrow 0$) and keeping a reasonable time-step.

2.4 Discrete forcing

Compared to the previous family of IB methods, instead of including a forcing function to the equations of the whole computational domain, the discrete forcing approach seeks to add a term in the already discretised equation to compensate for the difference in the desired solid velocity from the predicted fluid velocity at the interface Γ. We will first examine the general formulation of Direct Forcing methods and then we will examine the Ghost-Cell method and the Cut-Cell method which are used for sharper representations of the solid-fluid interface.

2.4.1 Direct forcing

In direct forcing methods, the forces at the immersed boundaries are calculated based on the temporally discretised momentum equations. The forcing term acts as a velocity corrector for the nodes near the IB and is prescribed at each time step to establish the desired solid velocity. The equation describing this force is the following:

$$\frac{\boldsymbol{u}^{n+1} - \boldsymbol{u}^n}{\Delta t} = \mathrm{RHS} + \boldsymbol{f}, \tag{2.15}$$

where the RHS includes all the convective, viscous and pressure gradient terms.

If a background grid node coincides with the fluid–structure interface, then the body force at this node can be directly calculated by:

$$\boldsymbol{f} = \begin{cases} \frac{1}{\Delta t}[\boldsymbol{u}_s^{n+1} - \boldsymbol{u}^*], & \boldsymbol{x} = \boldsymbol{X}_\Gamma \\ 0, & \text{elsewhere} \end{cases} \tag{2.16}$$

Here $\boldsymbol{u}^* = \boldsymbol{u}^n + \Delta t \times \mathrm{RHS}$ is the predicted fluid velocity without considering the effects of the interface. Also, if the background node is located at the immersed boundary, $\boldsymbol{u}^{n+1}$ is known. Indeed, this approach only holds if the IB coincides with the grid. This, however, is generally not the case. Hence the algorithm used at each time step will be explained using the idea of Balaras et al. [3]:

Step 1: Firstly, we compute a prediction velocity field $\boldsymbol{u}^*$ with the discretized Navier-Stokes equations omitting the forcing term $\boldsymbol{f}$. This velocity, of course, will not satisfy the boundary condition on the IB.

Step 2: We compute the forcing term of this time step using Eq. 2.16. The value of the velocity $\boldsymbol{u}_s^{n+1}$, representing the desired prescribed local solid velocity, on the forcing points is computed using an interpolation procedure. The forcing points can be placed outside the body [3, 22]. They can also be placed inside for the use of ghost-cell method [65, 94].

Step 3: We then compute the velocity field $\boldsymbol{u}^*$ from the discretized Navier-Stokes equations taking into account the forcing term. This velocity will now satisfy the desired IB condition.

Step 4: Using the velocity field of $\boldsymbol{u}^*$ we compute the pressure using the Poisson equation.

Step 5: Correct the velocity field and compute $\boldsymbol{u}^{n+1}$.

In order to implement the direct forcing in the classic diffused interface method we can use the discrete delta functions to transfer the quantities of the background and boundary nodes to the fluid mesh. There exist two approaches to achieve this. The first one is called explicit direct forcing where the force distribution and the velocity interpolation is decoupled. The second method, called implicit direct forcing, is based on the coupling between the force distribution and the velocity interpolation.

The first method was proposed by Uhlmann [97] and the force at each point of the immersed boundary is directly calculated by:

$$\boldsymbol{f}(\boldsymbol{X}_\Gamma) = \frac{\boldsymbol{u}^{n+1}(\boldsymbol{X}_\Gamma) - \boldsymbol{u}^*(\boldsymbol{X}_\Gamma)}{\Delta t}\,. \tag{2.17}$$

Here, $\boldsymbol{u}^{n+1}(\boldsymbol{X}_\Gamma)$ is the velocity at Γ which can either be prescribed, or calculated through the coupling with a structure solver. Furthermore, $\boldsymbol{u}^*(\boldsymbol{X}_\Gamma)$ is the velocity interpolated from the surrounding background nodes. This interpolation is described in Eq. 2.7. However, we can notice that in this method the force is punctual and not distributed to the background nodes. The distribution of the force is important in order to mimic the effects of the boundary.

The implicit direct forcing method addresses this issue by using the distributed forces from all the boundary points. Numerically, we substitute Eq. 2.6 into Eq. 2.16 and then we interpolate the obtained equation to the immersed boundaries. This gives the following expression:

$$\sum_{\boldsymbol{X'}\in\Gamma}\left(\sum_{\boldsymbol{x}\in\Omega}\delta(\boldsymbol{x}-\boldsymbol{X'})\delta(\boldsymbol{x}-\boldsymbol{X}_\Gamma)\Delta\boldsymbol{x}\Delta\boldsymbol{X}\right)\boldsymbol{f}(\boldsymbol{X'}) = \frac{\boldsymbol{u}^{n+1}(\boldsymbol{X}_\Gamma) - \boldsymbol{u}^*(\boldsymbol{X}_\Gamma)}{\Delta t}\,. \tag{2.18}$$

The force distribution and velocity interpolation are coupled in the implicit direct forcing IB method, while they are decoupled in the explicit direct forcing IB method. One advantage of the implicit method is that the velocity boundary conditions at the interface are more accurately satisfied. However, it also requires more computational resources. Direct forcing methods can be coupled with local mesh refinement procedures in order to better represent the immersed boundary.

2.4.2 Ghost-cell method

In this method we first determine the cells (or nodes) inside the solid boundary, which we call solid cells, and outside which we call fluid cells. For stationary boundaries this process can be done just once at the beginning of the simulation, but for moving boundaries it has to be done at the beginning of every time-step. What we call as ghost-cells (GC) are the solid cells that have at least one neighbor cell in the fluid. Once the ghost-cells are determined, we have to find an appropriate formulation for these cells which will lead to the implicit satisfaction of the boundary condition on the IB near the ghost-cell.

To achieve this, we use an "image-point" (IP) of the ghost-cell inside the fluid. The position of the IP is found by extending a segment from the node of the GC such that it intersects normal to the IB and that the boundary intercept (BI) point is midway between the ghost-node and the image-point as shown in Fig. 2.5.

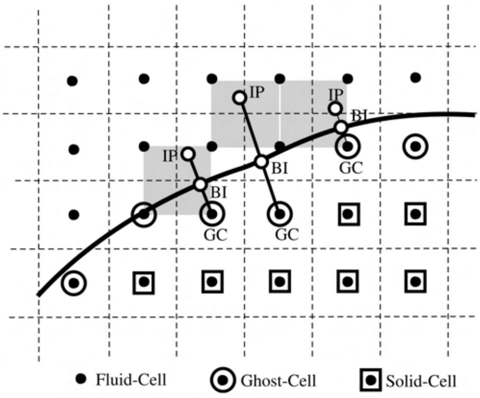


Figure 2.5: A 2D schematic representation of the identification of fluid-nodes, solid-nodes, ghost-nodes and the image-points (IP) and boundary intercepts (BI) for some ghost-nodes [65].

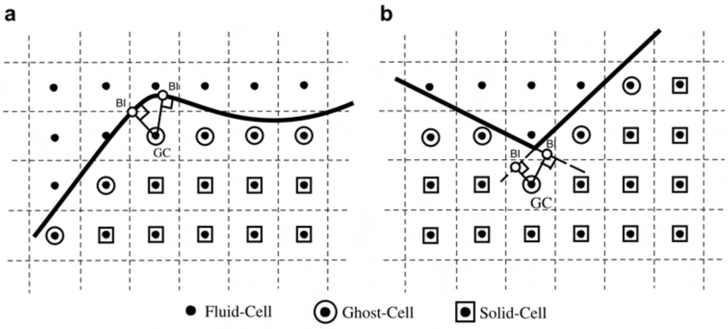


Figure 2.6: A 2D schematic representation of problematic BI identification. In (a) one ghost-node can have to corresponding BI points, while in (b) no BI can be determined from that ghost-node [65].

The identification of the BI may appear conceptually simple, but may present some complications during implementation, especially when dealing with highly complex geometries. In principle, the BI is a point on the immersed interface that has the minimum distance from the IB (in theory BI should be on the surface of the IB). In most cases, the IB is determined only by the normal-intercept from the ghost-node to the IB. However, even in 2D, we may encounter situations where the one-to-one determination of a BI from a ghost-node is not straightforward. Tthis can be seen in area with high curvature of the IB, as shown in Fig. 2.6). In 3D this problem becomes even more complicated. This problem should be addressed in order to avoid deteriorating the iterative convergence of the governing equations.

Once the boundary intercepts have been correctly identified we can proceed to treat the boundary condition on the IB. To impose the boundary condition on the IB, proper velocity vectors are assigned to the ghost-cells. The velocity vector on a ghost-node is interpolated with the values of the associated IP and BI. To do this, first we need to determine the velocity vector $\mathbf{v_{IP}}$ at the IP using the velocity vectors of the surrounding nodes.

Here, we examine the case of a 3D Cartesian mesh as shown in Fig. 2.7 and the equation demonstrations will be for a generic variable ϕ. Using a trilinear interpolation, the value of ϕ in the region between the eight nodes surrounding the image-point can be computed as:

$$\phi\left(x_1, x_2, x_3\right) = C_1 x_1 x_2 x_3 + C_2 x_1 x_2 + C_3 x_2 x_3 + C_4 x_1 x_3 + C_5 x_1 + C_6 x_2 + C_7 x_3 + C_8 \,. \tag{2.19}$$

The 8 unknown coefficients can be determined using the variable values of the 8 surrounding nodes:

$$\{C\} = [V]^{-1}\{\phi\}\,, \tag{2.20}$$

where

$$\{C\}^T = \{C_1, C_2, \ldots, C_8\}\ , \tag{2.21}$$

$$\{\phi\}^T = \{\phi_1, \phi_2, \ldots, \phi_8\}\ . \tag{2.22}$$

$[V]$ is the Vandermonde matrix corresponding to the trilinear interpolation scheme and has the form:

$$[V] = \begin{bmatrix} x_1x_2x_3|_1 & x_1x_2|_1 & x_1x_3|_1 & x_2x_3|_1 & x_1|_1 & x_2|_1 & x_3|_1 & 1 \\ x_1x_2x_3|_2 & x_1x_2|_2 & x_1x_3|_2 & x_2x_3|_2 & x_1|_2 & x_2|_2 & x_3|_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & & & & \\ x_1x_2x_3|_8 & x_1x_2|_8 & x_1x_3|_8 & x_2x_3|_8 & x_1|_8 & x_2|_8 & x_3|_8 & 1 \end{bmatrix}, \tag{2.23}$$

where the subscripts are identifiers of the eight surrounding nodes.

With Eq. 2.20 we calculate the coefficients and use them coupled with Eq. 2.19 to obtain the expression for the variable at the IP:

$$\phi_{\mathrm{IP}} = \sum_{i=1}^{8} \alpha_i \phi_i + \text{T.E.} \tag{2.24}$$

Here the coefficients α_i depend on $\{C\}$ and the coordinates of the IP. Since they are only dependant on the geometry, for stationary boundaries they can be computed and stored once in the beginning. Mittal et al. [65] show that the leading order of the truncation error is $\text{T.E.} = \text{O}\left(\Delta^2\right)$ where the grid spacing is $\text{O}\left(\Delta\right)$. For simplicity the truncation error will be omitted in the following equations.

Having computed the value of the variable at the IP and using the boundary condition imposed on the BI, we can compute the value of the variable of the ghost-cell by a linear approximation along the normal probe of length Δl extending from the GC to the IP.

More specifically, for a Dirichlet boundary condition we have that:

$$\phi_{\text{BI}} = \frac{1}{2}\left(\phi_{\text{IP}} + \phi_{\text{GC}}\right) = \frac{1}{2}\left(\sum_{i=1}^{8} \alpha_i \phi_i + \phi_{\text{GC}}\right) . \tag{2.25}$$

This can be then re-written with respect to the ghost-cell:

$$\phi_{\text{GC}} = 2\phi_{\text{BI}} - \sum_{i=1}^{8} \alpha_i \phi_i . \tag{2.26}$$

In the case of a Neumann boundary condition, as for the pressure, following a second-order central-difference scheme we obtain:

$$\left(\frac{\delta \phi}{\delta n}\right)_{\text{BI}} = \frac{\phi_{\text{IP}} - \phi_{\text{GC}}}{\Delta l} = \frac{1}{\Delta l}\left(\sum_{i=1}^{8} \alpha_i \phi_i - \phi_{\text{GC}}\right) , \tag{2.27}$$

$$\phi_{\text{GC}} = \sum_{i=1}^{8} \alpha_i \phi_i - \Delta l \left(\frac{\delta \phi}{\delta n}\right)_{\text{BI}} . \tag{2.28}$$

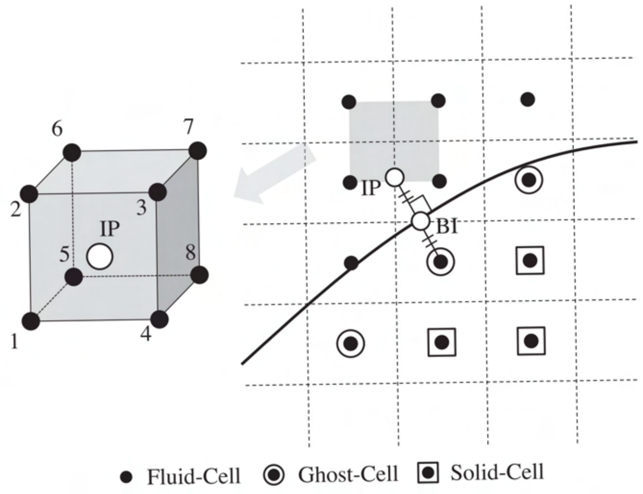


Figure 2.7: A schematic representation of the 8 point interpolation stencil for a 3D case in a Cartesian grid [51].

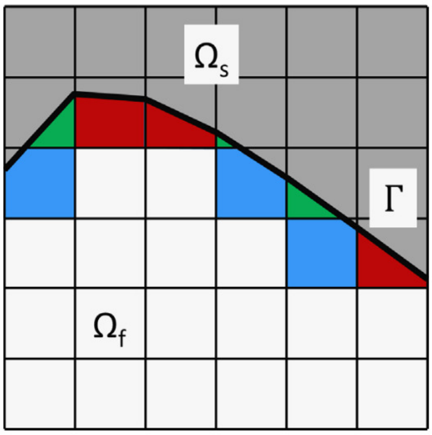


Figure 2.8: Schematic of reshaped cut-cells near the IB. Triangular (green), trapezoidal (red) and pentagonal (blue) cut-cells can be seen [91].

Using the method described for the velocity gives us the value of the velocity $\mathbf{v_{GC}}$ at the GC which acts as a forcing point. This is the velocity used in Eq. 2.16 and described at step 2 in the direct forcing algorithm presented earlier. In other words, we identify that:

$$\mathbf{v_{GC}} = \boldsymbol{u}_s^{n+1} \tag{2.29}$$

Other interpolation schemes, such as bi-linear or quadratic, can be used. A description of different approaches can be found in [63].

2.4.3 Cut-cell method

The next method is called cut-cell method, where the grid cells cut by the interface are reshaped according to the local geometry of the immersed boundary (Fig. 2.8). This way a boundary-conforming, albeit locally unstructured mesh is created. The fluxes across the faces of cut-cells are reconstructed from the surrounding regular cells and immersed boundaries.

This method is usually coupled with the level-set method to facilitate the tracking of the immersed boundary. Because of the high accuracy and the conservation properties near the interface, the cut-cell methods are suitable for turbulent flows. However, their use for turbulence is rare due to their disadvantages. One such drawback is the difficulty to extend the methods in 3D cases with complex geometries. Another important drawback is the very small time step that has to be used because of very small cut-cells that could arise near the interface.

2.5 Issues with IBM

Although significant progress has been accomplished over the preceding years in enhancing the performance of the IBM and promoting its applications in many areas, there are still several issues for further research [29, 42]. Some of the more prominent ones are the local violation of conservation laws near the IB, the spurious force oscillation caused by moving boundaries and the applicability of IBM in high Reynolds number flows.

2.5.1 Mass conservation

When using a finite-volume solver with a body conforming mesh, the characteristics of the integral form of the conservation equations translates to the discretised elementary volumes. The boundary conditions at the solid surface are directly introduced to the discretised equations and hence obey the conservative properties of the finite-volume approach.

In the case of IB methods, for example with the use of ghost-cells, the boundary condition is imposed on the immersed interface implicitly via an interpolation scheme using the variables of the fluid cells near the IB. This is done without taking into consideration the conservation principles, and thus, near the IB the conservation of the mass is not locally satisfied [63, 70].

Merlin [63] examined the local non-conservation of the mass using the following method. The flow around a cylinder in a canal was studied. To evaluate the mass error, the mass was integrated during two characteristic cycles (defined as the ratio of the cylinder's diameter to the fluid velocity in the middle of the canal). The local nature of the mass error became evident since increasing the domain of integration, the error decreased. More specifically, the mass was integrated between two planes upstream and downstream of the cylinder. The size of the domain was increased by distancing farther the planes from the cylinder in a way to increase the number of fluid cells. The configuration is shown in Fig. 2.9. The error was estimated by integrating the mass conservation equation as shown below:

$$\Delta M(t) = \frac{1}{M(t)} \left| M(t) - M(t-\tau) + \int_{t-\tau}^{t} \left(\dot{Q}_m \left(x_{max} \right) - \dot{Q}_m \left(x_{min} \right) \right) \mathrm{d}t \right| \tag{2.30}$$

where $M(t)$ is the mass included inside the integration domain at instant t and $\dot{Q}_m(x)$ the mass flow measured at the planes upstream and downstream of the cylinder, and τ the time-step.

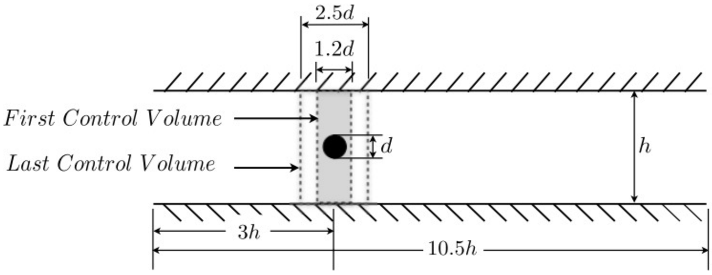


Figure 2.9: Geometry of the test case for the evaluation of the local mass error due to IBM [63].

The mass error was examined with respect to the fluid-mass volume ratio of the integration domain. To compare, the same mass computation was carried out in a case without a cylinder (i.e. without IBM). The results showed that by increasing the volume ratio of the fluid cells compared to the solid ones, the error decreased as shown in Fig. 2.10.

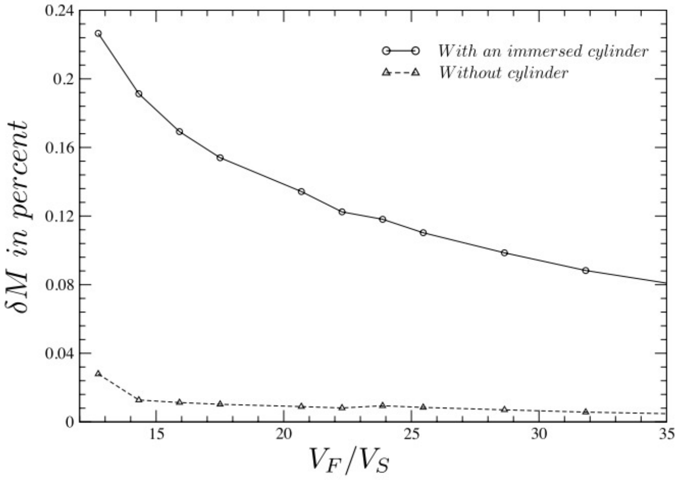


Figure 2.10: Evolution of mass error with the increase of domain of integration expressed as the ratio between the volume of the fluid and the volume occupied by the solid. Extracted from [63].

Of course, this error was tested on a direct forcing approach, hence the results are highly dependant on the spatial discretization scheme and the interpolation stencils used for the ghost-cells. However, Merlin provided a relatively trivial method to quantify the mass conservation error which can be implemented no matter the solver used.

A different way to measure the error in mass conservation would be to adopt a local approach, allowing us to obtain information at each computational cell around the location of the solid obstacle as shown in [11]. This method lets us examine if the no-slip condition at the solid surface is well-respected by computing the mass flow rate. In particular it suffices to examine the value of the velocity divergence at each cell, which theoretically, after the pressure correction step should be equal to zero at the no-slip boundary. The code YALES2 uses a node-centric mesh, so for an elementary volume Vi around a node 'i' we get:

$$\dot{M}_i = \rho_i \frac{\partial \boldsymbol{u_k}}{\partial \boldsymbol{x_k}} \, Vi \, . \tag{2.31}$$

2.5.2 Moving boundaries

For moving immersed boundaries, a common issue to consider is the treatment of what we call "fresh-cells" [66]. This term refers to cells of the Eulrian grid that at the previous time-step $t - \Delta t$ were solid-cells and at the current time-step t they become fluid-cells as shown in Fig. 2.11. The inverse, fluid-cells becoming solid-cells, can also cause problems.

To be more precise, in sharp-interface moving IBM (discrete forcing) spurious force oscillations arise due to the spatial discontinuity in pressure at the interface, the temporal discontinuity in the velocity of the fresh-cells or the violation of the mass conservation law as previously discussed [29]. Taking the example of a forcing point in a fluid-cell that becomes a solid-cell at time-step t, the next forcing point will become the node next to the interface. Although the velocity and pressure values can be interpolated at this new forcing point, the derivative values used

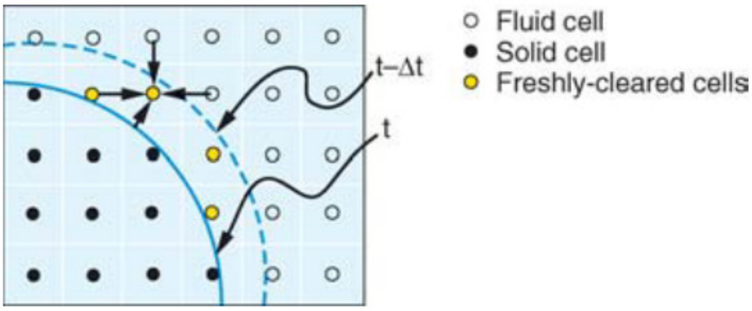


Figure 2.11: A 2D schematic representation of the fresh-cells created by the moving IB [66].

in Eq. 2.2 are not physical [56]. These nonphysical derivatives next to the solid surface introduce spurious pressure and vorticity distributions near the IB. A possible source of numerical oscillation in pressure is the nonphysical velocity gradient used in the pressure Poisson equation shown below:

$$\nabla^2(p^{n+\frac{1}{2}}) = \frac{1}{\Delta t}\nabla \cdot \boldsymbol{u}^* \,. \tag{2.32}$$

Various remedies have been developed through the years to overcome this issue. For flows around a single simple geometry a coordinate transformation strategy was proposed by Kim and Choi [39]. When dealing with fixed Eulerian grids, the field-extension technique was proposed to avoid the non-physical solutions caused by the sudden change of role of the fresh-cells [101]. Improvements for the pressure oscillations have been made by different methods. Such methods include adding a mass source/sink term in the continuity equation [51, 30, 52], using cut-cell approaches, and imposing Neumann boundary conditions on the IB for the pressure Poisson Eq. 2.32 [35].

Contrary to sharp-interface methods, with diffused-interface IB methods the spurious oscillations are naturally alleviated thanks to the smooth delta functions that distribute the forcing to the surroundings of the IB. The effect however is not completely eliminated but methods have been proposed involving a smoothing technique for the delta function so that it satisfies one-order higher moment condition of the derivatives of the delta function [102]. Also, penalisation methods (IBPM) generate smooth solutions due to the implicit treatment of the Lagrangian IB forces [29, 56].

2.5.3 High Reynolds number flows

IB methods have been implemented to flows of low Reynolds number values with great success. When it comes to higher Reynolds number flows, there are some challenges to overcome. For example, in the case of diffused-interface IBM, a small mesh spacing Δs is required between the Lagrangian points of the IB so that $\Delta s \leq 0.5\Delta x$ (Δx being the Eulerian fluid grid spacing). To resolve the boundary layer, it has been shown that near the IB, $\Delta x \propto 1/Re^n$, where $n = 1/2$ for laminar flows and $3/4 \leq n \leq 1$ for turbulent flows [29].

For direct numerical simulation (DNS) and wall resolved large eddy simulation (LES) of turbulent flows around complex mobile geometries, careful grid placement is needed to resolve the near-wall turbulence. Whether we use linear interpolation (for sharp-interface IBM) or smoothed delta function (for diffused-interface IBM), the first several fluid grid point should be located inside the viscous sub-layer of the wall bounded turbulence so that the interpolated velocity is consistent with the wall-law of turbulence. For high Reynolds number flows, a compromise can be made to combine the IBM with wall-models of LES. This way, a wall-layer model is adopted for the inner layer (viscous sub-layer, buffer region, and part of the logarithmic region) and the outer layer is resolved [29, 57]. Thus, the first fluid grid point can be located as far as the logarithmic region as the wall shear stress can be computed using a wall-law model. To incorporate this modelling approach to the IBM, the interpolated velocity at the forcing points is replaced by the velocity computed through the wall-law model (which is usually an iterative process). Except for the imposition of the velocity at the forcing point, the modification of the sub-grid scale turbulent viscosity ν_t can also be adopted to enforce the boundary conditions at the IB. If we look at the case of the ghost-cell method, the value of the turbulent viscosity is needed at all fluid-nodes and the ghost-nodes in order to correctly estimate the diffusive fluxes. For the GC, instead of using the explicit filtering operation used in the fluid domain, the ν_t value of the associated IP is used, which implies a Neumann boundary condition at the IB. Indeed, the use of Dirichlet condition for the imposition of zero turbulent viscosity at the IB would lead to a negative value of ν_t at the GC and would cause a back-scatter effect [63]. The value at the IP can be computed using an interpolation scheme similar to the strategy used for the velocity.

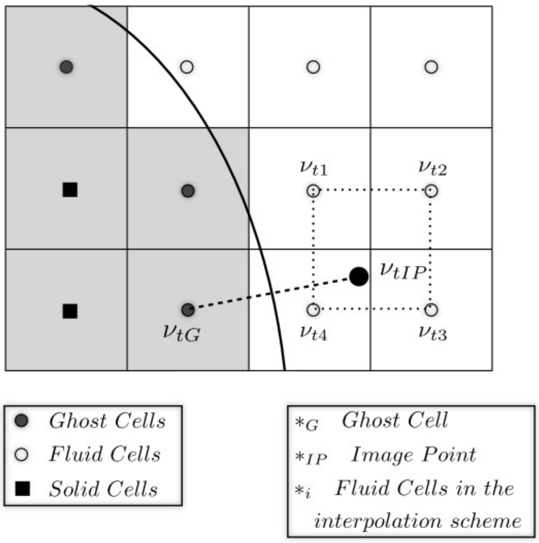


Figure 2.12: A 2D schematic representation of the interpolation stencil for the computation of the turbulent viscosity in a ghost-cell [63].

2.6 IBM in turbine flow applications

Renewable energy production units, such as wind turbines, involve geometrically complex bodies (rigid or flexible) in motion and in contact with fluids. Numerical simulations with deforming computational domains can be performed with either body-fitted meshes or non-conformal meshes. In the first approach, a widely used body-fitted method is the Arbitrary-Lagrangian-Eulerian (ALE) coupled with automatic mesh re-adaptation. However, in the case of rotating wind turbines, this proves computationally expensive and due to mesh degradation it may suffer from stability issues. State of the art computations in wind energy are based on the second approach where no re-meshing is required, using Large-Eddy simulations (LES) coupled with actuator line methods (ALM). This framework requires the geometry and the airfoil aerodynamic coefficients of the turbine's blades to model its influence onto the fluid. Although valid in a wide range of configurations, its specific assumptions makes it irrelevant in some cases like with yaw misalignment. Despite that, due to the main challenges of IB methods described in the previous section, the use of IBM in for wind-turbines is not yet that common.

One of the most common uses of IBM in turbine flows is its application for modelling complex terrain, or the hub and the supporting structures of the turbine. For example, Jafari et al. [32] used a direct forcing IBM for wind flow over a two-dimensional hill (Fig. 2.13) as well as a 3D case around the Bolund hill (a 12 m high coastal hill located in Denmark). The IB method which was applied to a RANS solver incorporated wall-functions and took into account the surface roughness of the boundary. The results were compared with experimental data and body-fitted simulations showing good agreement.

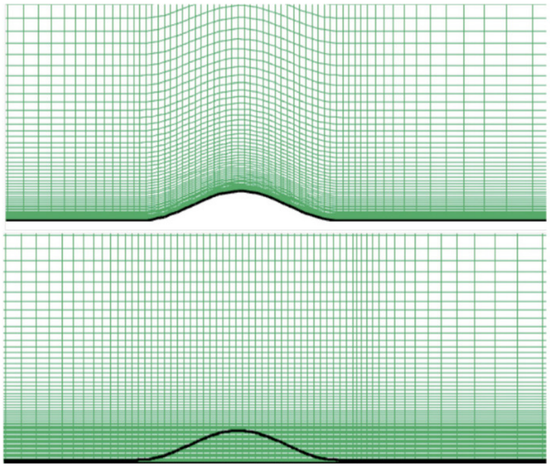


Figure 2.13: Representative body-conforming grid for the 2D hill and Cartesian grid used for IBM [32].

Another such application can be found in Liu and Hu [55] where Unsteady-RANS simulations were carried out for multiple horizontal axis tidal turbines (HATT). The hub and the supporting structures of the turbine were modeled using a sharp-interface IBM while the rotor blades were modelled using the ALM. Good agreement with experimental data was found for the predicted wake velocity and turbulence intensity (TI).

Jendoubi [34] has implemented IBM for the simulation of horizontal axis wind-turbines (HAWT) using a finite element solver over an unstructured grid coupled with cutting-edge approach. The cutting-edge strategy is similar to the cut-cell method and works by adding nodes to the mesh at the intersection points between the IB and the node-pairs of the initial mesh so that we have at each time-step a mesh that follows the surface of the body as shown in Fig. 2.14. In their method they impose at the new nodes (and the ones inside the solid region) Dirichlet conditions at the start of each iteration. In their test case of a simplified geometry resembling a four-blade wind-turbine results showed good agreement with conventional body-fitted methods.

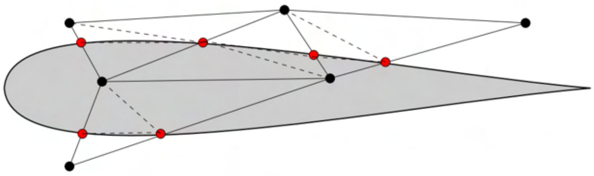


Figure 2.14: Black nodes represent the initial mesh and red nodes are the body-conforming nodes created by the cutting-edge method [34].

Ouro and Stoesser [75] carried out simulations for a Darrieus-type vertical axis tidal-turbines (VATT) (Fig 2.15) using the direct forcing approach introduced by Fadlun et al. [16] and later improved by Uhlmann [97]. They studied a 3-bladed turbine in a laminar flow regime and compared the results to highly-accurate body-fitted simulations. Torque and normal coefficients were in good agreement showing the stability and accuracy of the IBM in absence of turbulence. The results of this LES-IB method were then heavily assessed against experimental and two RANS-based model data. The predicted power production agrees well with the experimental data. Furthermore, the LES-IB outperformed the two RANS models ($k-\omega$ SST and $k-\omega$ SST LRE). The analysis of the evolution of the generated power and lift during a single revolution revealed a significant variation of these quantities depending on the phase of the revolution. Also, an important difference between LES and RANS is the good prediction of lift coefficient during dynamic stall, a phenomenon that RANS models struggle to reproduce in an accurate manner [75]. Finally, the visualisation of blade-vortex interaction for three different tip speed ratios (TSR) showed that the generation and transport of energetic large-scale structures depends strongly on the rotational speed of the turbine. With low TSR, the blades are subject to strong dynamic stall and large leading-edge flow separation creating vortices of approximately the size of the blade's chord length. On the contrary, with high TSR, there is light dynamic stall and the vortices are less significant. All these results show a promising future for the applicability of IB methods for turbine flows.

2.7 Concluding remarks

The continuous forcing approach is well suited for the study of flows with elastic IB, they have a sound physical basis and they are simple to implement. When it comes to rigid IB this approach

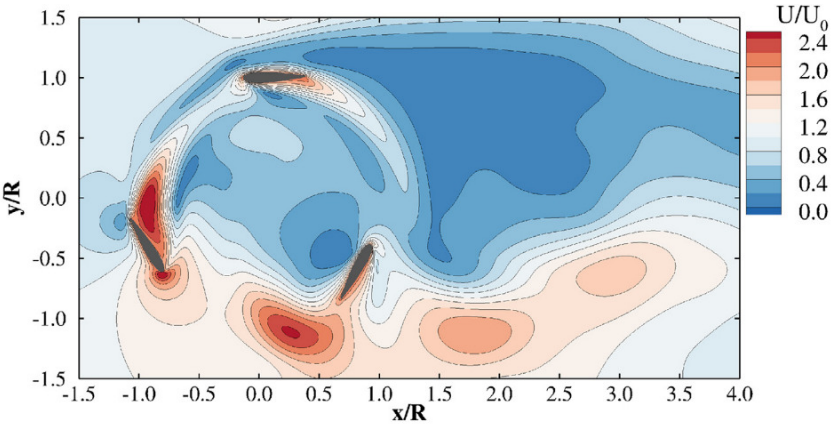


Figure 2.15: Stream-wise velocity contour for VATT in laminar flow. Extractd from [75].

sees some challenges due to the fact that the forcing terms used are generally not well behaved for the rigid limit. The parameters introduced for the treatment of rigid bodies introduce numerical accuracy and stability issues. The smooth distribution of the forcing term inherent to these approaches leads to a smeared representation of the IB which can be especially undesirable for high Reynolds number values. Finally, these methods require the resolution of the governing equations inside the IB. Thus, with increasing Reynolds numbers generally the cells inside the IB will also increase; this requirement of solving the equations inside the IB makes for an unattractive computational overhead.

The methods included in the discrete forcing approaches introduce the boundary condition of the IB directly into the discrete flow equations. Hence, the forcing term is intimately connected to the discretisation schemes chosen and the practical implementation is not straightforward like the continuous forcing approach. On the other hand, these discrete forcing methods enable a sharper representation of the IB which is desirable in simulation with higher Reynolds number values. Also, due to the lack of user defined parameters, these methods do not introduce extra stability constraints in the representation of solid bodies. IB methods in this category allow to use a larger time-step size than that allowed by continuous forcing immersed boundary methods and, with a proper treatment of discrete forcing, permit sharp representation of immersed boundaries. Finally, another advantage of these methods is the fact that the equations of the fluid nodes are decoupled from the ones for the solid nodes, which alleviates the need to solve the governing equations inside of the solid body. One drawback of discrete forcing is that the treatment of moving IB can be more delicate. Another drawback in contrast to the continuous forcing, is that they usually also require imposition of a pressure boundary condition on the IB [96].

A schematic diagram outlining the Immersed Boundary Methods described in this section is shown in Fig. 2.16.

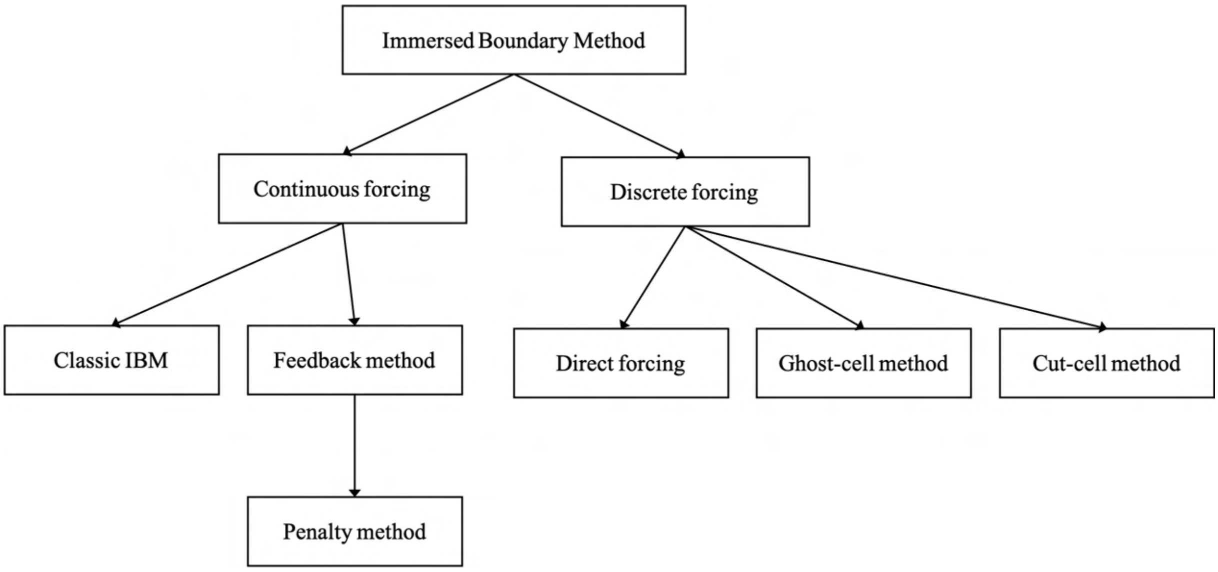


Figure 2.16: Diagram outlining the most well-known Immersed Boundary Methods.

Chapter 3

Numerical modelling of turbulent flows

Contents

3.1 Fluid flow governing equations

In continuum mechanics, the movement of fluid materials is described by the Navier-Stokes equations. These equations are rigorously derived by statistical mechanics and fundamental principles (conservation of molecule numbers, momentum, energy, etc). This derivation is not trivial [5].

The balance equations are translated to the macroscopic scale as the conservation equations of mass, momentum and energy. Using the Einstein index notation, the mass and momentum equations take the following forms:

- **Mass conservation**

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u_i}{\partial x_i} = 0\,, \tag{3.1}$$

where u_i is the component of the velocity vector in the direction i and ρ is the fluid density.

- **Momentum conservation**

$$\frac{\partial \rho u_j}{\partial t} + \frac{\partial \rho u_i u_j}{\partial x_i} = -\frac{\partial P}{\partial x_j} + \frac{\partial \tau_{ij}}{\partial x_i}\,, \tag{3.2}$$

where P is the pressure. For a Newtonian fluid, the viscous stress tensor τ_{ij} can be expressed in the following manner:

$$\tau_{ij} = \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) - \frac{2}{3}\mu\frac{\partial u_k}{\partial x_k}\delta_{ij}\,, \tag{3.3}$$

where μ is the dynamic viscosity of the fluid and δ_{ij} is the Kronecker symbol.

3.2 Methods for handling turbulence

3.2.1 Introduction to turbulence

Reynolds was the first to identify two possible states for fluid flows: laminar and turbulent. In flows characterised by lesser velocities, the small perturbations are dampened by the molecular viscosity preserving an orderly manner of flow. In this case the flow is said to be laminar. On the other hand, with increasing velocities, the viscosity is no longer sufficient to dissipate all the perturbations and they will in turn be amplified by various instability mechanisms of non-linear nature. In this case, the flow passes to a turbulent regime. An example of such an instability mechanism is the Kelvin-Helmholtz instability which occurs when there is velocity shear at the interface between two fluids or two regions of the same continuous fluid, producing vortical structures.

Turbulent flows involve a wide range of spatial and time scales, characterised by an unsteady nature and disordered flows, and are by definition three-dimensional, unsteady and chaotic phenomenon.

The majority of industrial applications involve turbulent flows, whose properties present various interesting advantages. Among the various uses of turbulence in the industrial domain, some interesting applications may be the reduction of drag force in zones where the flow is detached (for example in golf balls) or the triggering of mixing of a fluid with multiple components (such as in piston motors or gas turbines) to name a few.

To determine the nature of a fluid flow we use the Reynolds number. It is a number without units and it is calculated as the ratio of the inertial forces (which destabilise the flow and generate new scales of turbulence) over the viscous forces (which dissipate the turbulent energy and try to stabilise the flow):

$$Re = \frac{UL}{\nu}\,, \tag{3.4}$$

where U and L represent the characteristic speed and length of the flow and ν is the kinematic viscosity of the fluid. Small values characterise laminar flows while larger values turbulent flows. The transition between the two regimes is done around a value called critical Reynolds value, which depends on the geometry of the domain in which the fluid flows (for example, $Re_c = 3000$ for a cylindrical channel, or $Re_c = 5 \times 10^5$ for a flow over a plane plate).

The multitude of vortices created by turbulence can be defined either by their size or their energy levels. Indeed, in fluid mechanics, there is the phenomenon called energy cascade, where there is a transfer of energy from the larger and more energetic scales of motion to the smaller ones due to the non-linear nature of the flow equations. This principle of energy cascade was introduced by Richardson [85] and Kolmogorov [44].

We can distinguish between three different zones in the turbulent kinetic energy spectrum [79]:

- **Integral zone**: it contains the largest and most energetic structures, of low frequency, associated with the integral scale l_t defined as the macroscopic scale of the flow linked to the anisotropic large structures of the flow. The turbulent kinetic energy characterising this region is defined as:

  $$\mathrm{TKE} = \frac{1}{2} u_i' u_i' \,, \tag{3.5}$$

  where u_i' is the fluctuation of the characteristic velocity (in other words, the variation of the velocity u_i around its mean value $\bar{u}_i$) in the i direction.

  The length and velocities of the structures in the integral zone are comparable to the quantities defining the Reynolds number and are not affected by the viscous effects. Thus, a turbulent Reynolds number linked to the integral scales can be defined as:

  $$Re_t = \frac{u' l_t}{\nu} \,. \tag{3.6}$$

- **Inertial zone**: the larger vortical structures become unstable and break down to smaller ones via the cascade process. The energy is transferred from large structures to smaller ones by a $k^{-5/3}$ law.

- **Viscous dissipation zone**: this zone contains the smallest vortical structures with the highest frequencies. Here, all the turbulent kinetic energy of the fluid is dissipated in the form of heat. The turbulent structures are characterised by the Kolmogorov scale whose length and velocity are defined as:

  $$\eta = \left(\frac{\nu}{\epsilon}\right)^{\frac{1}{4}} \text{ and } u_\eta = (\nu\epsilon)^{\frac{1}{4}} \,, \tag{3.7}$$

  where ϵ is the average dissipation rate of the turbulent kinetic energy per unit mass. The characteristic Reynolds number of the Kolmogorov scale can be expressed as:

  $$Re_\eta = \frac{u_\eta' \eta}{\nu} \approx 1 \,. \tag{3.8}$$

  The smallest scales of the flow have solely the role of dissipating energy in an isotropic and universal manner, as the cascade has destroyed the initial anisotropy of the integral scales.

The ratio between the integral scale l_t and the Kolmogorov scale η leads to:

$$\frac{l_t}{\eta} = \mathcal{O}(Re_t^{3/4})\,. \tag{3.9}$$

This ratio shows the vast range of scales involved in turbulent flows, which increase with higher Reynolds number values. In other words, for a given computational domain, with higher Reynolds number values, we need to further diminish the size of the computational cells in order to resolve the whole spectrum of scales.

To numerically resolve the discretised Navier-Stokes equations, we can chose to resolve only part of the spectrum, that is the larger scales of turbulence, and use models to describe the smaller scales. Depending on the choice of which part of the spectrum to solve and which part to model, we can derive three principal approaches: Direct Numerical Simulation (DNS), Large-Eddy Simulation (LES) and Reynolds-Averaged Navier-Stokes simulations (RANS). Figure 3.1 illustrates which scales of the turbulence are solved and which are modelled depending on the method. In the following section, we'll describe in more detail each approach.

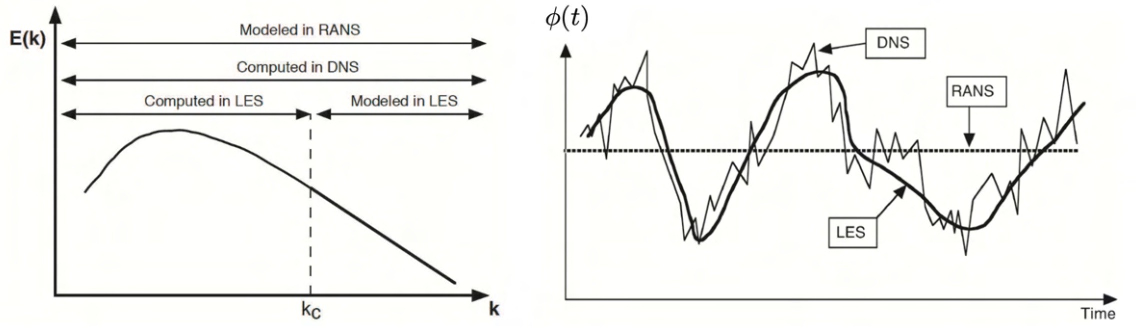


Figure 3.1: Illustration of the resolved and modelled parts of the turbulence for the DNS, LES and RANS approaches in the spectral space (left) and in the physical space (right). Extracted from [4].

3.2.2 DNS

In Direct Numerical Simulation (DNS), all the turbulent scales of the flow are resolved. The discretised Navier-Stokes equations (Eq. 3.1 and Eq. 3.2) are solved under the assumption that the cells are sufficiently small to capture correctly the continuous flow behaviour with all the turbulent scales. No modelling is required and the only errors are related to the discretised representation of the computational domain. The main inconvenience of this type of simulation is the very high computational cost to solve real problems. Despite of the progress of modern supercomputers, it is still impossible to solve all the scales of a high Reynolds number flow and this is why DNS is mainly used for academic test cases at lower Reynolds number flows.

3.2.3 RANS

The Reynolds-Averaged Navier-Stokes (RANS) equations are obtained by decomposing the flow variables into a statistical averaged value and a fluctuating value as follows:

$$\phi_i = \langle \phi_i \rangle + \phi_i' \,. \tag{3.10}$$

By using this decomposition for all the field variables, having a constant density and applying the averaging operator, the RANS equations can be expressed as:

$$\frac{\partial \rho \langle u_i \rangle}{\partial x_i} = 0 \,, \tag{3.11}$$

$$\frac{\partial \rho \langle u_j \rangle}{\partial t} + \frac{\partial \rho \langle u_i \rangle \langle u_j \rangle}{\partial x_i} = -\frac{\partial \langle P \rangle}{\partial x_j} + \frac{\partial \langle \tau_{ij} \rangle}{\partial x_i} - \frac{\partial \rho \langle u_i' u_j' \rangle}{\partial x_i} \,, \tag{3.12}$$

The additional term $-\rho \langle u_i' u_j' \rangle$ is called the Reynolds stress tensor. The turbulence modelling in RANS consists in formulating the Reynolds stress tensor in a way that the equations above can be solved numerically. In essence, RANS can only solve the steady part of the flow and thus the turbulence has to be entirely modelled.

3.2.4 LES

In the Large-Eddy Simulation approach, part of the turbulent spectrum, i.e. the larger eddies, are resolved while the smallest vortical structures are modelled. Essentially, the scale separation is achieved by applying a low-pass filter for the turbulence. This filtering consists of a convolution product in the physical space. Applying the filtering on a scalar $\phi(\mathbf{x}, t)$ gives:

$$\overline{\phi}(\mathbf{x}, t) = \int_{\mathbb{R}^3} \phi(\mathbf{x}', t) \, \mathcal{G}_\Delta(\mathbf{x}' - \mathbf{x}) \mathrm{d}\mathbf{x}' \,, \tag{3.13}$$

where $\overline{\phi}$ is the filtered quantity and $\mathcal{G}_\Delta$ is the filter associated to the cutoff scale length Δ. Homogeneous LES filters must satisfy a set of properties when applied to the Navier-Stokes equations [86]:

- **Conservation of constants**

$$\overline{\alpha} = \alpha \Rightarrow \int_{\mathbb{R}^3} \mathcal{G}_\Delta(\mathbf{x}) \mathrm{d}\mathbf{x} = 1 \,, \tag{3.14}$$

- **Linearity**

$$\overline{\phi + \psi} = \overline{\phi} + \overline{\psi} \,, \tag{3.15}$$

- **Commutation with derivatives**

$$\overline{\frac{\partial \phi}{\partial s}} = \frac{\partial \overline{\phi}}{\partial s} \,. \tag{3.16}$$

We can decompose a variable ϕ in a part that is representative of the turbulence scales larger than δ, noted as $\overline{\phi}$, and a part evolving in the turbulence scales smaller than Δ, noted as ϕ', thus giving the following expression:

$$\phi(\mathbf{x},t) = \overline{\phi}(\mathbf{x},t) + \phi'(\mathbf{x},t)\,. \tag{3.17}$$

In general, in case the fluid density is variable, we can introduce the notion of the Favre averaging which is a density weighted average expressed as:

$$\widetilde{\phi} = \frac{\overline{\rho\phi}}{\overline{\rho}}\,. \tag{3.18}$$

Applying this filtering to the discretised Navier-Stokes equations gives the filtered LES system of equations:

$$\frac{\partial\overline{\rho}}{\partial t} + \frac{\partial\overline{\rho}\widetilde{u}_i}{\partial x_i} = 0\,, \tag{3.19}$$

$$\frac{\partial\rho\widetilde{u}_j}{\partial t} + \frac{\partial\rho\widetilde{u}_i\widetilde{u}_j}{\partial x_i} = -\frac{\partial\overline{P}}{\partial x_j} + \frac{\partial\overline{\tau}_{ij}}{\partial x_i} - \frac{\partial}{\partial x_i}\left[\overline{\rho}\left(\widetilde{u'_i u'_j} - \widetilde{u}_i\widetilde{u}_j\right)\right]\,, \tag{3.20}$$

Similar to RANS, we need a closure model for the stress tensor $\tau'_{ij} = -\overline{\rho}\left(\widetilde{u'_i u'_j} - \widetilde{u}_i\widetilde{u}_j\right)$, named the sub-grid stress tensor. In LES, the turbulence modelling consists of proposing formulations for the sub-grid stress tensor based on the filtered variables of the above equations. Compared to RANS where the larger scale structures are dependant on the geometry and boundary conditions, in LES the sub-grid scale (SGS) turbulent structures are more suited for modelling due to their universal behaviour.

The sub-grid stress tensor needs to be described by a turbulence model in order to reproduce correctly the energy transfer from the resolved scales to the non-resolved ones. In this work, only the classical Boussinesq hypothesis [6] is used for the models, which suggests that the sub-grid stress tensor can be formulated like the viscous stress tensor by using a turbulent viscosity $\nu_t = \mu_t/\rho$ such that:

$$\tau'_{ij} = \mu_t\left(\frac{\partial\widetilde{u}_i}{\partial x_j} + \frac{\partial\widetilde{u}_j}{\partial x_i}\right) - \frac{2}{3}\mu_t\frac{\partial\widetilde{u}_k}{\partial x_k}\delta_{ij}\,. \tag{3.21}$$

The sub-grid scale turbulence models in LES are formulations of this turbulent viscosity ν_t. In the YALES2 library, a variety of turbulence models is available. Some of the most commonly used are the standard and dynamic Smagorinsky models and the WALE model.

3.2.4.1 Smagorinsky model

The classical Smagorinsky model [89] is based on the hypothesis of an equilibrium between the creation and the dissipation of kinetic energy at the filter scale Δ. The turbulence here is considered to be a purely dissipative phenomenon, and the formulation of ν_t is:

$$\nu_t = (C_S\Delta)^2\sqrt{2\widetilde{S}_{ij}\widetilde{S}_{ij}}\,, \tag{3.22}$$

where C_S is the Smagorinsky constant and $\widetilde{S}_{ij}$ is the filtered deformation tensor:

$$\widetilde{S}_{ij} = \frac{1}{2}\left(\frac{\partial \widetilde{u}_i}{\partial x_j} + \frac{\partial \widetilde{u}_j}{\partial x_i}\right) . \tag{3.23}$$

The choice of the value for the constant C_S depends on the configuration of interest, but the usual range is between 0.1 and 0.2. However, this model is known to be too dissipative and handle poorly the turbulence near walls.

Germano [19] and Lilly [54] suggested to modify the constant C_S locally and in time. The behaviour of the sub-grid scales is deduced by the smallest resolved scales. This requires a second filtering, noted as $\widehat{\bullet}$, of the resolved velocity with a filter scale $\Delta' > \Delta$. The constant is then determined using the two differently filtered velocity fields. This model is known as the dynamic Smagorinsky model. It is more costly to use, but gives better results for a wide range of applications.

3.2.4.2 WALE model

The Wall-Adapting Local Eddy-Viscosity (WALE) from Nicoud and Ducros [73] suggests to use a turbulent viscosity profile near the walls for better prediction of the laminar-turbulent transition. The formulation of the turbulent viscosity becomes:

$$\nu_t = (C_w \Delta)^2 \frac{(s^d_{ij} s^d_{ij})^{3/2}}{(\widetilde{S}_{ij}\widetilde{S}_{ij})^{5/2} + (s^d_{ij} s^d_{ij})^{5/4}} , \tag{3.24}$$

where C_w is a constant with recommended value of 0.5 and the tensor s^d_{ij} is written as:

$$s^d_{ij} = \frac{1}{2}\left(\widetilde{h}_{ij} + \widetilde{h}_{ji}\right) - \frac{1}{3}\widetilde{h}_{kk}\delta_{ij} , \tag{3.25}$$

with:

$$\widetilde{h}_{ij} = \widetilde{g}_{ik}\widetilde{g}_{kj} \ \text{ and } \ \widetilde{g}_{ij} = \frac{\partial \widetilde{u}_i}{\partial x_j} . \tag{3.26}$$

This formulation allows the model to consider rotation and strain rate with s^d_{ij}, thus all turbulent structures are considered for the energy dissipation. Also, the turbulent viscosity tends to zero in sheared flows which is the expected behaviour near the walls.

3.3 Interface description in multiphase flows

For the case of multiphase flows, most often involving liquid-gas flows, there is a need for numerical tools that allow the description of the evolution of the liquid-gas interface dynamics and topology in time and space. There is a wide variety of such methods belonging to several main classifications.

The first successful technique that was developed was the Marker-And-Cell (MAC) method [24] initially used for free-surface flows and later adapted for multifluid flows. This method consists of placing Lagrangian virtual particles as markers in the Eulerian cells and advecting them with the updated velocity. The cells that contained a marker were considered to contain fluid. Later, the use of Lagrangian markers to represent and advect an interface led to the Front-Track (FT) method [98, 80], where a chain of Lagrangian markers connected by segments (creating a front) was used to explicitly describe the interface position. These techniques, where a Lagrangian advection of the liquid-gas interface is used, belong to the interface-tracking class of methods.

On the other hand, a class of methods widely used for simulating material interface dynamics is the interface-capturing approaches, where Eulerian interface advection techniques are used. These methods use the transport of an implicit function which allows to identify the phases and describe the interface geometry. The two main methods of this class are the Volume-of-Fluid (VOF) [26] and the Level-Set (LS) [87] techniques. The former advects the fluid volume fraction to represent the interface while in the latter the interface is located at the zero-level of a regular signed-distance function.

In this work we are interested in two-phase flows where one phase is fluid and the other one is solid. While the movement and the form of the solid-fluid interface uses a Lagrangian framework, the formalism used in the Navier-Stokes equations to properly describe the interface and the influence of the solid phase to the fluid's dynamics corresponds to Eulerian framework. This is why in the following sub-section the VOF and LS methods are briefly presented in the context of liquid-gas interfaces.

3.3.1 Volume-of-Fluid method

The Volume-of-Fluid (VOF) method was proposed by [26]. The idea is to define a function C, also known as the color-function, whose value at a computational cell will determine if the cell contains fluid. In particular, C represents the fractional volume of the cell occupied by the fluid. This function is the spatial average over a control volume V of a phase indicator function H:

$$C(\mathbf{x},t) = \frac{1}{V}\int_V H(\mathbf{x},t)\,\mathrm{d}V\,, \tag{3.27}$$

where H is a Heaviside function defined as:

$$H(\mathbf{x},t) = \begin{cases} 1 & \text{if } \mathbf{x} \text{ is in liquid}\,, \\ 0 & \text{if } \mathbf{x} \text{ is in gas}\,. \end{cases} \tag{3.28}$$

The phase indicator function is conserved throughout the simulation so as to guaranty mass conservation, and its material derivative is equal to zero:

$$\frac{DH}{Dt} = \frac{\partial H}{\partial t} + \boldsymbol{u}\cdot\nabla H = \frac{\partial H}{\partial t} + \nabla\cdot(\boldsymbol{u}H) - H\nabla\cdot\boldsymbol{u} = 0\,, \tag{3.29}$$

Integrating Eq. 3.29 over a control volume V:

$$\int_V \frac{\partial H}{\partial t}\,\mathrm{d}V + \int_V \nabla\cdot(\boldsymbol{u}H)\,\mathrm{d}V - \int_V H\nabla\cdot\boldsymbol{u}\,\mathrm{d}V = 0\,, \tag{3.30}$$

Using Eq. 3.27 and applying the divergence theorem gives:

$$V\frac{\partial C}{\partial t} + \int_A \boldsymbol{u}H\cdot\mathrm{d}A - \int_V H\nabla\cdot\boldsymbol{u}\,\mathrm{d}V = 0\,, \tag{3.31}$$

In incompressible flows the last term is equal to zero but may still be used for numerical implementation [64]. By omitting it, the semi-discrete advection of C can be written as:

$$C^{n+1} = C^n + \frac{\Delta t}{V} \int_A \boldsymbol{u} H \cdot \mathrm{d}A \,, \tag{3.32}$$

where A is the surface of the control volume and $\boldsymbol{u}H$ the flux across the control surface. It is important to note that Eq. 3.32 is valid locally for a control volume equivalent to a computational cell, and if summed over all the computational grid cells (with correct boundary conditions) we get:

$$\sum_{\text{cells}} C^{n+1} = \sum_{\text{cells}} C^n \,. \tag{3.33}$$

This leads to the inherent volume conservation of the liquid phase in time when using the VOF method.

3.3.2 Level-Set method

The VOF method represents a robust method for tracking the liquid-gas interface while conserving the volume of the liquid, but it is inconvenient to use when it is necessary to compute the curvature of the interface or other smooth properties [37]. In response, Sethian [87] introduced the level-set method which represents the interface as the zero-level set (or zero value contour) of a higher-dimensional function. In practice it relies on the use of a smooth, continuous function indicating the signed distance to the interface Γ defined as:

$$\phi(\mathbf{x}, t) = \pm |\mathbf{x}(t) - \mathbf{x}_\Gamma(t)| \,, \tag{3.34}$$

where $\mathbf{x}_\Gamma(t)$ is the position of the interface at the time t. The sign of the distance is determined by the phase at each position in time:

$$\phi(\mathbf{x}, t) \begin{cases} > 0 & \text{if } \mathbf{x}(t) \text{ is in liquid} \,, \\ = 0 & \text{if } \mathbf{x}(t) \text{ is at the interface } \mathbf{x}_\Gamma \,, \\ < 0 & \text{if } \mathbf{x}(t) \text{ is in gas} \,. \end{cases} \tag{3.35}$$

This implies that the interface can be expressed as the zero-level contour of ϕ:

$$\Gamma(t) = \{\mathbf{x}(t) \in \mathbb{R}^3 \mid \phi(\mathbf{x}, t) = 0\} \,, \tag{3.36}$$

The interface between the two fluids follows a path given by the local flow velocity $\boldsymbol{u}$. Along this path in time, the level-set function must not change value. This leads to the evolution for ϕ expressed as:

$$\frac{D\phi}{Dt} = \frac{\partial \phi}{\partial t} + \boldsymbol{u} \cdot \nabla \phi = 0 \,, \tag{3.37}$$

Equation 3.37 is the level-set equation and represents the linear advection of ϕ in the same manner as any other passive scalar field.

Thanks to the smoothness of the level-set function, geometric quantities such as the interface normal vector $\mathbf{n}$ and the interface curvature κ can easily be estimated by the derivatives of the level-set function such that:

$$\mathbf{n} = \frac{\nabla\phi}{|\nabla\phi|} \quad \text{and} \quad \kappa = \nabla \cdot \frac{\nabla\phi}{|\nabla\phi|} \,. \tag{3.38}$$

3.4 YALES2 solver

For the present work and all the simulations carried out, the LES finite-volume flow solver YALES2 has been used [69]. It is specifically tailored for massively parallel computing on unstructured meshes (usually, but not limited to, triangle elements in 2D and tetrahedral elements in 3D). It consists of a node-centric framework using a dual mesh. The control volumes (CV), where the transport equations are integrated, are constructed around the computational grid nodes of the mesh. The physical variables, such as the velocity and the pressure for example, are also stored on the nodes representing the average value over the CV, while the fluxes are computed on the control volume's edges. An example of the control volume is presented in Fig. 3.2

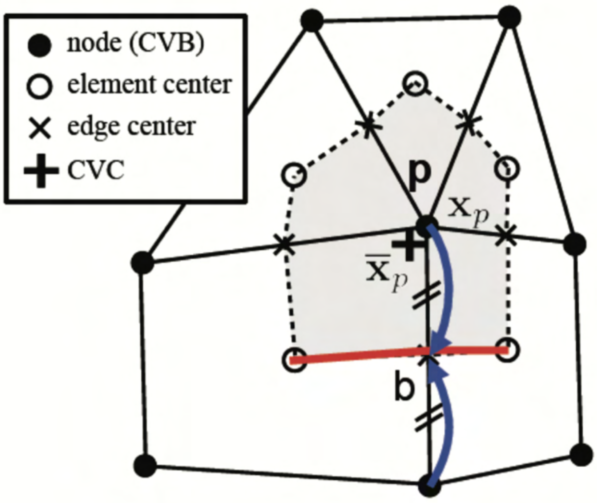


Figure 3.2: A YALES2 control volume based around a grid node. x_p is the position of the node and $\overline{x}_p$ is the position of the CV barycentre.

Several explicit time integration schemes are available in YALES2: Runge-Kutta (RK) of 2nd, 3rd and 4th order, Lax-Wendroff (LW), Crank-Nicholson (CN) and the more recent scheme proposed by Kraushaar [45], named TFV4A, which combines the RK and LW methods. The TFV4A is a 4th order accurate finite-volume method and will be used for all simulations in this work. For the spatial integration 2nd and 4th order schemes are available. The latter will be used in this work.

3.4.1 Operator splitting

To alleviate the constraint of the multiple temporal scales, the YALES2 solver uses an 'operator splitting' method. This method consists of solving each operator (convection, diffusion, etc.) separately with its proper characteristic time-step. The advantage of this method stems from the choice of the time-step used in each iteration taken as the largest characteristic time of the different physical phenomena. Usually, the time-step chosen is that of the convection. The diffusion takes multiple sub-steps inside the fluid iteration in order to respect its proper stability condition. However, the operator splitting is susceptible to accumulate numerical errors, known as splitting errors, over time. These errors become larger if the ratio between different characteristic time-steps becomes very large.

3.4.2 Incompressible flow solver

The YALES2 CFD platform offers a wide range of different solvers tailored for different physical problems. For aerodynamic studies at low-Mach number flows, the incompressible solver (ICS) is proposed.

3.4.2.1 Incompressible Navier-Stokes equations

The incompressibility hypothesis implies that the density of the fluid remains constant in time and space. This is incorporated into the Navier-Stokes equations used by the incompressible YALES2 solver and they read as follows:

$$\nabla \cdot \boldsymbol{u} = 0 \,, \tag{3.39}$$

$$\frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot (\boldsymbol{u} \otimes \boldsymbol{u}) = -\frac{1}{\rho}\nabla P + \frac{1}{\rho}\nabla \cdot \boldsymbol{\tau} + \boldsymbol{f} \,, \tag{3.40}$$

where $\boldsymbol{u}$ is the fluid velocity, P the pressure, ρ the constant fluid density, the viscous stress tensor is now written as $\boldsymbol{\tau} = \mu \left(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T\right)$ with μ the fluid dynamic viscosity and $\boldsymbol{f}$ other volumetric force terms depending on the physics (gravitational force for example). For simplicity, the latter forcing term will be omitted in the demonstration of the resolution method that follows.

3.4.2.2 Resolution method

The time advancement in YALES2 is based on a fractional-step projection method first proposed by Chorin [10] in 1968 and later modified by Kim and Moin [40]. The velocity is computed at time-steps $n, n+1$, etc while scalar quantities, the density and the pressure are computed at time-steps $n+1/2, n+3/2$, etc. This projection method, widely used for incompressible fluid flow problems, is based on the Helmholtz decomposition which states that a sufficiently smooth vector field can be expressed as the sum of an irrotational vector field and a solenoidal vector field. Applying this decomposition to the velocity field yields:

$$\boldsymbol{u} = \boldsymbol{u}_i + \boldsymbol{u}_s \,, \tag{3.41}$$

where $\boldsymbol{u}_s$ is the solenoidal part verifying $\nabla\cdot\boldsymbol{u}_s = 0$ and the $\boldsymbol{u}_i$ is the irrotational part verifying $\nabla\times\boldsymbol{u}_i = 0$. The irrotational vector field derives from a scalar potential and can be thus expressed as $\boldsymbol{u}_i = \nabla\varphi$. By applying the divergence operator to this equation gives:

$$\nabla\cdot\boldsymbol{u} = \nabla\cdot\boldsymbol{u}_i = \nabla^2\varphi\,. \tag{3.42}$$

Using this decomposition on the velocity field allows us to advance the momentum equation in two steps.

1. **Prediction step**

   A first estimation of the velocity at the time $n+1$ is obtained by advancing the momentum equation without the pressure gradient term as it only contributes to the irrotational velocity field. The prediction equation reads:

$$\frac{\boldsymbol{u}^* - \boldsymbol{u}^n}{\Delta t} = -\nabla\cdot(\boldsymbol{u}^*\otimes\boldsymbol{u}^n) + \frac{1}{\rho}\nabla\cdot\boldsymbol{\tau}^n\,. \tag{3.43}$$

2. **Correction step**

   Next, we use the pressure gradient to correct the predicted velocity by adding its irrotational component:

$$\frac{\boldsymbol{u}^{n+1} - \boldsymbol{u}^*}{\Delta t} = -\frac{1}{\rho}\nabla P^{n+1/2}\,. \tag{3.44}$$

   In order to compute $\boldsymbol{u}^{n+1}$ the value of the pressure at $n+1/2$ is needed. This value is determined by solving a Poisson equation for the pressure derived by taking the divergence of equation 3.44. At the same time we can impose the contraint of null divergence for the velocity $\boldsymbol{u}^{n+1}$. The pressure Poisson equation (PPE) reads:

$$\nabla\cdot\nabla P^{n+1/2} = \frac{\rho}{\Delta t}\nabla\cdot\boldsymbol{u}^* - \frac{\rho}{\Delta t}\cancelto{0}{\nabla\cdot\boldsymbol{u}^{n+1}}\,. \tag{3.45}$$

   Finally by inserting the correct pressure value in equation 3.44 we can compute the final velocity field:

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^* - \frac{\Delta t}{\rho}\nabla P^{n+1/2}\,. \tag{3.46}$$

In practice, the implementation of the prediction-correction method differs in YALES2.

1. **Prediction step**

   In the prediction step, instead of omitting completely the effect of the pressure, we keep the gradient of the pressure from the previous time $n-1/2$ [43]. This leads to a better estimation of $\boldsymbol{u}^*$, reducing the errors linked to the fractional time advancement.

$$\frac{\boldsymbol{u}^* - \boldsymbol{u}^n}{\Delta t} = -\nabla\cdot(\boldsymbol{u}^*\otimes\boldsymbol{u}^n) - \frac{1}{\rho}\nabla P^{n-1/2} + \frac{1}{\rho}\nabla\cdot\boldsymbol{\tau}^n\,. \tag{3.47}$$

2. **Correction step**

   In the correction step we have to subtract the previous pressure term, leading to the following correction equation:

$$\frac{\boldsymbol{u}^{n+1} - \boldsymbol{u}^*}{\Delta t} = -\frac{1}{\rho}\nabla P^{n+1/2} + \frac{1}{\rho}\nabla P^{n-1/2} \,. \tag{3.48}$$

   The PPE now reads:

$$\nabla \cdot \left(\nabla P^{n+1/2}\right) = \frac{\rho}{\Delta t}\nabla \cdot \boldsymbol{u}^* + \nabla P^{n-1/2} \,. \tag{3.49}$$

3.4.3 Solving the Poisson equation

As shown in the previous section for the pressure variation we need to solve the Poisson equation, an elliptic partial differential equation of the form:

$$\nabla \cdot (\alpha \nabla \varphi) = \mathrm{RHS} \,. \tag{3.50}$$

This invokes a system of linear equations, where the unknown is a vector containing the pressure values of all the nodes of the discretised computational domain. Its resolution needs the use of a linear solver and it may occupy a large part of the overall computational time needed for the simulation. Linear solvers are iterative processes where a large number of iterations may be needed for an estimation sufficiently close to the exact solution depending on the algorithm used as well as the characteristics of the laplacian operator. Furthermore, in the context of parallel computing, communications between the processors are necessary at each iteration of the linear solver increasing the portion of the total simulation time occupied by the linear solver. This may reach even 80% of the total time if there is no appropriate treatment of the linear solver [61]. Hence, the optimisation of the resolution of the Poisson equation is a very important factor to the cost of incompressible flow simulations. In YALES2, there exist various algorithms for inverting a linear system such as the Preconditioned Conjugate Gradient (PCG) [99], the Deflated PCG (DPCG) [72] and the BIGSTAB scheme [99].

3.4.4 Parallel computing

YALES2 has been developed with massively parallel computing in mind, in order to be suitable option for the simulations of recent years, where

In recent years, the size of the computational domain keeps getting bigger, the resolution in space keeps getting higher and more and more physical phenomena are taken into account. Consequently, the simulations need to be able to handle large meshes while using multiple processors in parallel to execute the computations. With these requirements in mind, YALES2 has been specifically tailored to be able to handle efficiently unstructured meshes with several billions of elements thanks to its massively parallel computing character.

To handle the large quantities of data associated with the computation, the most used method in CFD is the domain decomposition, where each processor is assigned to only one part of the computational domain. The interdependence of the sub-domains is taken care of by the communication between processors which exchange information at the interface of every computational element group thanks to MPI (Message Passing Interface) instructions. The decomposition of the mesh to sub-domains has to ensure that the distribution of the work load on all the processors is the most optimal possible. In the context of a Eulerian mesh, the evident solution would be to have each sub-domain contain the same number of control volumes. This decomposition is trivial for structured meshes but more difficult for unstructured ones. Since YALES2 mainly focuses on unstructured grids, the domain decomposition is handled by external libraries: METIS [36] or SCOTCH [9].

In YALES2 this approach is taken a step further to optimise the performances of computations involving a very large number of processors by using a double domain decomposition. In this case, the sub-domains created by the first decomposition of the mesh, and attributed to a processor, are further decomposed to multiple groups of computational cells (defined by the grey interfaces in Fig. 3.3). The size of these element groups is determined in a way so that the data contained in them suits the capacities of the cache memory of the processors. This double domain decomposition primarily benefits the optimisation of the Poisson solver's performances. In fact, these element groups serve as a coarse mesh used by the two-stage linear solvers like the DPCG. The solver now involves two types of communications: i) external communications between processors at the black interfaces in figure 3.3 handled by the MPI communications; ii) internal communications allowing exchanges between the element groups, at the grey interfaces in Fig. 3.3, found in a processor. The internal communications are not handled by MPI. In Fig. 3.4 a schematic showing the communication and exchange of data between element groups, communicators and boundary conditions.

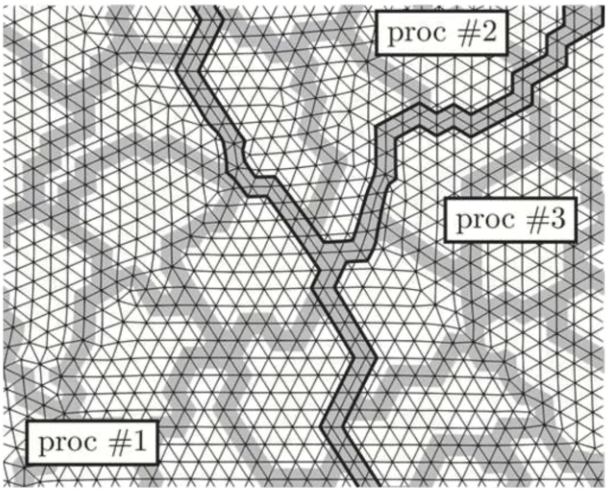


Figure 3.3: A representation of the double domain decomposition on a 2D triangular mesh.

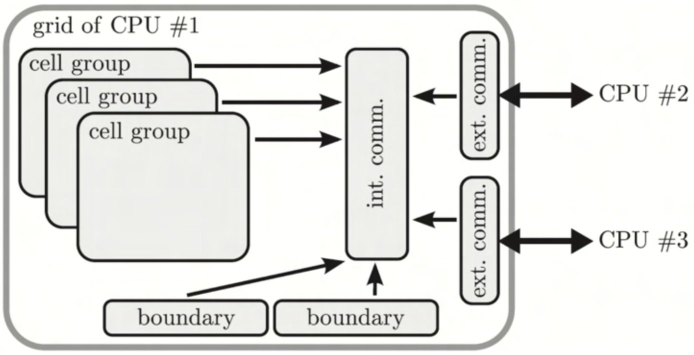


Figure 3.4: A schematic of the communications when using double domain decomposition.

Chapter 4

Volume-of-Solid Implicit Volume Penalty Method

Contents

4.1 Introduction

An essential part of the IBM is the procedure used to immerse the object into the computational mesh and the determination of the position of the nodes with respect to the object's surface in order to separate (tag) the computational cells into fluid cells (outside the object), solid cells (inside the object) and interface cells (partially inside) [95]. For the purpose of handling mobile interfaces on static meshes, two well-established numerical tools used are the Level-Set technique [87] and the Volume-of-Fluid (VOF) approach [26]. Usually utilised for tracking interfaces between fluids, these two techniques may be adapted for solid-fluid interfaces. In that case, the term Volume-of-Solid (VOS) can be used instead of VOF to indicate the presence of solid-fluid interface. Liu et al. [58] have used the VOS approach to represent the solid coupled with a direct forcing IB method, similar to the works of Ng [71]. In their work, the solid surface is represented by Lagrangian points and the cell type (fluid, solid or interface) is determined with a simple ray-tracing technique. The solid volume fraction field ϕ_s is then computed on the partially or fully immersed cells through a Gaussian quadrature integration. This volume fraction is then used to determine the value of the forcing term by correcting the velocity inside the solid region to a mean value of the predicted fluid velocity, obtained after the prediction step in a prediction-correction scheme, and the imposed solid velocity weighted by $(1-\phi_s)$ and ϕ_s respectively. Another feature of their method is the inclusion of the same forcing term to the pressure Poisson equation as a source term to impose the solid velocity at the same time as the divergence-free condition. Morente et al. [68] have used a VOF approach coupled with a penalty IBM for the simulation of bubbly flows where spherical bubbles are considered as moving penalised obstacles interacting with the fluid. A bubble is transported in a Lagrangian way and its interface is determined by the position of the bubble's mass centre and its radius. From the interface position a volume fraction field is defined separating the two phases which is then used as the penalty mask of the forcing term. However, instead of using the usual penalised momentum equation as proposed by Angot et al. [2], a two-fluid Eulerian multiphase frame is used where the momentum equations of both phases are coupled leading to a single equation. The IBM presented in this work bares similar elements to the aforementioned methods.

To expand on the method's main components, three tasks can be defined:

1. **Coupling the VOS approach with an IBM**

   Incorporating the solid volume fraction field from the VOS approach into the Navier-Stokes equations allowing to define a composite velocity field, computed as the mean of the solid and fluid velocities, weighted by their respective volume fractions. Using the composite velocity leads to a new system of equations capable of describing the evolution of both fluid and solid domains at once. These resemble the pure fluid equations but additional mass and momentum source terms appear to represent the solid movement.

2. **Modelling the implicit penalty forcing term**

   Expressing the penalty term in an implicit manner so that its contribution can be split in the prediction step and the correction step of a projection scheme. The contribution in the correction stage serves to impose the solid velocity in the solid region at the same time as the incompressibility constraint is satisfied. This leads to the challenge of solving a modified pressure Poisson equation.

3. **Representing the solid body in a Lagrangian manner**

   Representing the solid object as a set of Lagrangian particles containing an elementary quantity of solid volume. First, the immersed object is displaced by moving the Lagrangian particles according to the prescribed motion and then projecting the solid volume onto the Eulerian grid creates the local solid volume fraction field.

Taking into account the techniques used, this method will be referred to as Volume-of-Solid Implicit Volume Penalty method (VOS-IVP). In the following sections the method will be explained in detail.

4.2 Coupling the VOS approach with an IBM

4.2.1 Immersed Boundary Method for Large-Eddy Simulations

The main advantage of the IBM is that body-conforming meshes are not necessary. Instead, the computational domain Ω includes both fluid Ω_f and solid domains Ω_s, so $\Omega = \Omega_f \cup \Omega_s$ (as seen in Fig. 4.1). Thanks to the continuous mesh in the solid region, there is no need for re-meshing in the case of a moving immersed object, which makes it an attractive solution for simulating moving bodies in fluid flow.

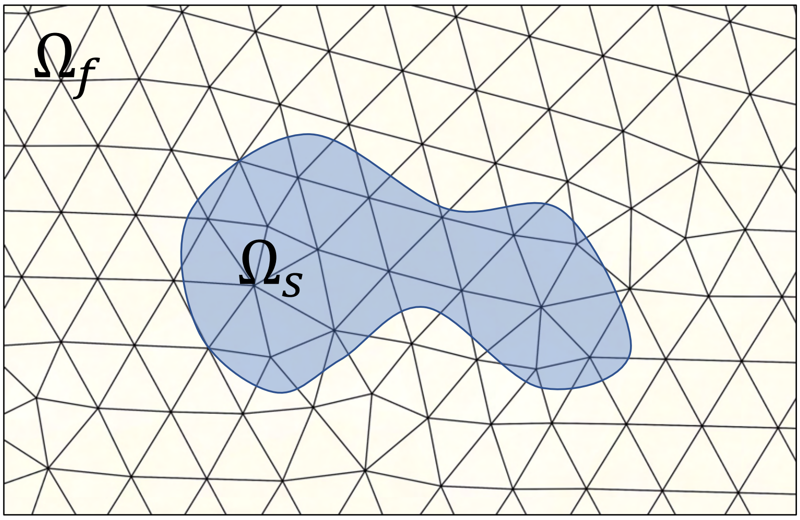


Figure 4.1: Mesh used with IBM including both fluid and solid domains.

Let us introduce the scalar field of the local solid volume fraction $\phi_s(\boldsymbol{x}, t)$ as the fraction of the volume occupied by the solid in a computational cell $V_{s,i}$ over the total cell volume V_i at time t, defined as:

$$\phi_{s,i}(t) = \frac{V_{s,i}(t)}{V_i}\,, \tag{4.1}$$

where it takes the following values:

$$\phi_s = \begin{cases} 1 & \text{in} \quad \Omega_s\,, \\ 0 & \text{in} \quad \Omega_f\,. \end{cases} \tag{4.2}$$

Equivalently we can define the local fluid volume fraction as $\phi_f = 1 - \phi_s$.

This study is carried out with Large-Eddy simulations (LES) in which the smallest vortical structures are not resolved but modelled. Hence the physical quantities pass through a filtering operation. The filtering operator, which consists of projecting a field on the LES grid, is expressed as $\widetilde{\bullet}$. The LES Navier-Stokes equations were previously shown (Eq. 3.19 and Eq. 3.20) in the case of a single fluid or phase occupy the whole computational domain. In our case, the domain is occupied by a fluid phase and a solid phase. Thus, the flow equations that describe the evolution of the fluid quantities in a control volume need to be modified to apply to the fraction of the control volume occupied by the fluid. This can be achieved by using the finite volume integration with a phase indicator. For the transport of the physical property Ψ of a phase k, with the local volume fraction as a phase indicator, we can write:

$$\int_{V_k} \frac{\partial}{\partial t}\left(\widetilde{\Psi}_k\right) \mathrm{d}V_k + \int_{V_k} \nabla \cdot \left(\widetilde{\Psi}_k \widetilde{\boldsymbol{u}}_k\right) \mathrm{d}V_k = 0\,, \tag{4.3}$$

$$\Rightarrow \int_{V} \frac{\partial}{\partial t}\left(\phi_k \widetilde{\Psi}_k\right) \mathrm{d}V + \int_{V} \nabla \cdot \left(\phi_k \widetilde{\Psi}_k \widetilde{\boldsymbol{u}}_k\right) \mathrm{d}V = 0\,. \tag{4.4}$$

In our context, this manipulation introduces the local fluid volume fraction ϕ_f in front of the fluid quantities while integrating in the whole domain (like in the case of one-phase problems). The modified LES flow equations for the fluid phase with a constant density and with the addition of the IBM term, read as:

$$\frac{\partial}{\partial t}(\phi_f) + \nabla \cdot (\phi_f \widetilde{\boldsymbol{u}}_f) = 0\,, \tag{4.5}$$

$$\frac{\partial}{\partial t}(\phi_f \widetilde{\boldsymbol{u}}_f) + \nabla \cdot (\phi_f \widetilde{\boldsymbol{u}}_f \otimes \widetilde{\boldsymbol{u}}_f) = -\frac{1}{\rho}\nabla \widetilde{P} + \frac{1}{\rho}\nabla \cdot (\phi_f \widetilde{\boldsymbol{\tau}}_f) + \mathbf{f}_{\mathrm{IB}}\,. \tag{4.6}$$

where $\widetilde{\boldsymbol{u}}$ is the fluid velocity, $\widetilde{P}$ the pressure and ρ the constant fluid density. The viscous stress tensor can be expressed as:

$$\widetilde{\boldsymbol{\tau}}_f = \mu_{\mathrm{eff}} \left(\nabla \widetilde{\boldsymbol{u}}_f + \nabla \widetilde{\boldsymbol{u}}_f^T - \frac{2}{3}(\nabla \cdot \widetilde{\boldsymbol{u}}_f)\boldsymbol{\mathcal{I}} \right)\,, \tag{4.7}$$

where $\boldsymbol{\mathcal{I}}$ is the identity tensor and μ_{eff} the effective fluid dynamic viscosity evaluated as the sum of the molecular and turbulent viscosities. The turbulent contribution in this study was obtained from the Dynamic Smagorinsky model [89, 19, 54]. The last term in Eq. 4.7 is equal to zero due to the incompressibility constraint $\nabla \cdot \widetilde{\boldsymbol{u}}_f = 0$.

Finally, the term $\mathbf{f}_{\mathrm{IB}}$ represents any additional volumetric momentum sources such as the forcing term of the IBM, the expression of which will be explored in section 4.3.1. Note that the LES-filtering notation will be dropped for the rest of this work.

The original VOS-IVP method was coupled with the incompressible solver of the YALES2 library [69] which solves the low-Mach number Navier-Stokes equations for turbulent flows on unstructured grids using a projection method for pressure-velocity coupling [10]. A central 4th-order numerical scheme is used for spatial discretisation and a 4th-order Runge-Kutta like scheme for the time integration. The Poisson equation is solved with a Deflated Preconditioned Conjugate Gradient (DPCG) solver [61].

4.2.2 Governing equations

In the previous section the flow equations were presented for the evolution of the fluid physical quantities such as the fluid velocity $\boldsymbol{u}_f$. We can also define a vector field $\boldsymbol{u}_s$ containing information about the local solid velocity. This allows the introduction of a new composite velocity field computed as the addition of the fluid and solid velocities weighted by the fluid and solid volume fraction, respectively, as follows:

$$\boldsymbol{u} = \phi_f \boldsymbol{u}_f + \phi_s \boldsymbol{u}_s \,. \tag{4.8}$$

Using the relation in Eq. 4.8 and the fact that $\phi_s + \phi_f = 1$, we can rearrange the Navier-Stokes equations so that they describe the evolution of the composite velocity $\boldsymbol{u}$. The composite velocity allows to describe the evolution of both the fluid and the solid quantities on the Eulerian mesh via a single momentum conservation equation and improves numerical stability of the solver, since it does not need specific treatment at the interface between the two phases. For a sharp representation of the solid volume fraction the convective cross terms that include the product $\phi_s\phi_f(\boldsymbol{u}_s - \boldsymbol{u}_f)$ can be neglected (shown in Appendix A) since $\phi_s\phi_f = 0$ away from the solid/fluid interface and $\boldsymbol{u}_f \approx \boldsymbol{u}_s \approx \boldsymbol{u}$ at the interface. This gives rise to a new conservative system of equations for both the fluid and the solid domains at once:

$$\nabla \cdot \boldsymbol{u} = \underbrace{\frac{\partial}{\partial t}(\phi_s) + \nabla \cdot (\phi_s \boldsymbol{u}_s)}_{\mathscr{Q}_s} \,, \tag{4.9}$$

$$\frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot (\boldsymbol{u} \otimes \boldsymbol{u}) = -\frac{1}{\rho}\nabla P + \frac{1}{\rho}\nabla \cdot \boldsymbol{\tau} + \underbrace{\frac{\partial}{\partial t}(\phi_s \boldsymbol{u}_s) + \nabla \cdot (\phi_s \boldsymbol{u}_s \otimes \boldsymbol{u}_s)}_{\mathscr{P}_s} + \mathbf{f}_{\mathrm{IB}} \,. \tag{4.10}$$

Detailed derivation of Eq. 4.9 and Eq. 4.10 is shown in Appendix A. These equations are similar to the pure fluid equations except for the additional source terms on the right-hand side representing the solid movement and/or deformation. The mass source, noted as $\mathscr{Q}_s$, represents the change of solid volume in space and time, and the momentum source $\mathscr{P}_s$ represents the solid acceleration projected on the Eulerian non-conforming grid. In the context of the usual one-fluid immersed boundary model, the term $\mathscr{P}_s$ would be equivalent to the momentum needed to move the fluid found at the interior of the immersed body according to the solid movement and will be further discussed in section 4.3.3.

4.2.3 Discretised mass and momentum source terms

In the VOS-IVP method, the incompressible Navier-Stokes equations can be discretised as follows:

$$\nabla \cdot \boldsymbol{u}^{n+1} = \mathscr{Q}^{n+1} \,, \tag{4.11}$$

$$\frac{\boldsymbol{u}^{n+1} - \boldsymbol{u}^n}{\Delta t} = -\nabla \cdot (\boldsymbol{u}^n \otimes \boldsymbol{u}^n) - \frac{1}{\rho}\nabla P^{n+1/2} + \frac{1}{\rho}\nabla \cdot \boldsymbol{\tau}^n + \mathscr{P}_s^{n+1} + \mathbf{f}_{\mathrm{IB}}^{n+1} \,, \tag{4.12}$$

where n denotes the iteration of the previous time-step and $n+1$ the current time-step.

The scalar quantities in the YALES2 solver are advanced from time-step $n-1/2$ to $n+1/2$ or from $n+1/2$ to $n+3/2$. The same happens to the solid particles, so after the re-localisation of the particles on the Eulerian mesh, the new solid volume fraction $\phi_s^{n+3/2}$ is computed according to the prescribed motion. Then it is computed at the time-step $n+1$ by:

$$\phi_s^{n+1} = \frac{1}{2}\left[\phi_s^{n+1/2} + \phi_s^{n+3/2}\right] . \tag{4.13}$$

The new solid velocity field is computed directly at $n+1$. This way, we can express the VOS mass and momentum sources at the time-step $n+1$ as:

$$\mathscr{Q}_s^{n+1} = \nabla\cdot\boldsymbol{u}^{n+1} = \left[\frac{\phi_s^{n+3/2} - \phi_s^{n+1/2}}{\Delta t}\right] + \nabla\cdot\left(\phi_s^{n+1}\boldsymbol{u}_s^{n+1}\right), \tag{4.14}$$

$$\mathscr{P}_s^{n+1} = \left[\frac{\phi_s^{n+3/2}\boldsymbol{u}_s^{n+1} - \phi_s^{n+1/2}\boldsymbol{u}_s^{n}}{\Delta t}\right] + \nabla\cdot\left(\phi_s^{n+1}\boldsymbol{u}_s^{n+1}\boldsymbol{u}_s^{n+1}\right) . \tag{4.15}$$

4.3 Modelling the penalty forcing term

4.3.1 Implicit volume penalisation and modified pressure Poisson equation

This new method utilises a volume penalty approach for the IB forcing term appearing in Eq. 4.10. The penalty term guaranties that the composite velocity $\boldsymbol{u}$ remains equal to the imposed solid velocity inside the solid region through a simple Dirichlet type boundary condition. At this time, no wall-law model has been implemented. The penalty forcing term reads:

$$\mathbf{f}_{\mathrm{IB}} = \frac{\chi_s}{\eta}\left(\boldsymbol{u}_s - \boldsymbol{u}\right) . \tag{4.16}$$

The penalty mask is a Heaviside function of the solid fraction at each new time-step:

$$\chi_s = \begin{cases} 1 & \text{if} \quad \phi_s > 0.5\,, \\ 0 & \text{else.} \end{cases} \tag{4.17}$$

The penalty parameter is set as a function of the time-step:

$$\eta = \alpha\Delta t\,, \tag{4.18}$$

where $0 < \alpha \leq 1$ is called the penalisation time-step ratio.

Usually this penalty forcing term is applied solely on the intermediate velocity $\boldsymbol{u}^*$ when a projection method is used [10], and the final velocity at time-step $n+1$ is then modified by the correction step with the new pressure field, so the boundary condition and the continuity constraint are not satisfied at the same time. In the present method, as the method's name suggest, the penalty term is expressed implicitly using the final unknown velocity field. This allows the forcing term to split into a contribution in the prediction step and a contribution in the correction step as shown below:

$$
\begin{aligned}
\mathbf{f}_{\mathrm{IB}}^{n+1} &= \frac{\chi_s}{\eta}\left(\boldsymbol{u}_s^{n+1}-\boldsymbol{u}^{n+1}\right)\\
&= \underbrace{\frac{\chi_s}{\eta}\left(\boldsymbol{u}_s^{n+1}-\boldsymbol{u}^{*}\right)}_{\mathbf{f}^*}+\underbrace{\frac{\chi_s}{\eta}\left(\boldsymbol{u}^{*}-\boldsymbol{u}^{n+1}\right)}_{\mathbf{f}^{\mathrm{corr}}}\ .
\end{aligned}
\tag{4.19}
$$

The YALES2 solver relies on a modified projection method based on the Helmholtz-Hodge decomposition [10] to advance the Navier-Stokes equations in time. Including the previous pressure gradient term in the computation of the intermediate velocity field $\boldsymbol{u}^*$ leads to a smaller error term in the prediction step making the correction step less computationally demanding [43]. The intermediate velocity field is computed taking into account the prediction penalty force $\mathbf{f}^*$:

$$
\begin{aligned}
\frac{\boldsymbol{u}^*-\boldsymbol{u}^n}{\Delta t} =& -\nabla\cdot\left(\boldsymbol{u}^*\otimes\boldsymbol{u}^n\right)-\frac{1}{\rho}\nabla P^{n-1/2}+\frac{1}{\rho}\nabla\cdot\boldsymbol{\tau}^n+\mathscr{P}_s^{n+1}\\
&+\frac{\chi_s}{\eta}\left(\boldsymbol{u}_s^{n+1}-\boldsymbol{u}^*\right)\\
\Rightarrow \boldsymbol{u}^* =& \frac{\boldsymbol{u}^n}{\gamma_s}-\frac{\Delta t}{\gamma_s}\left(\nabla\cdot\left(\boldsymbol{u}^*\otimes\boldsymbol{u}^n\right)+\rho^{-1}\nabla P^{n-1/2}-\rho^{-1}\nabla\cdot\boldsymbol{\tau}^n-\mathscr{P}_s^{n+1}\right)\\
&+\frac{\gamma_s-1}{\gamma_s}\boldsymbol{u}_s^{n+1}\,,
\end{aligned}
\tag{4.20}
$$

where the factor $\gamma_s=(1+\chi_s\Delta t/\eta)$ is named the penalty density factor, with values of $\gamma_s=1$ in the fluid and $\gamma_s>1$ inside the solid.

Before the correction step, the old pressure gradient needs to be subtracted leading to the new intermediate velocity $\boldsymbol{u}^{**}$:

$$
\boldsymbol{u}^{**}=\boldsymbol{u}^*-\left(-\frac{\Delta t}{\gamma_s\rho}\nabla P^{n-1/2}\right)\ . \tag{4.21}
$$

To find the irrotational part of the velocity field we correct the intermediate velocity with the addition of the new pressure term. The correction penalty term is also added:

$$
\frac{\boldsymbol{u}^{n+1}-\boldsymbol{u}^{**}}{\Delta t}=-\frac{1}{\rho}\nabla P^{n+1/2}+\frac{\chi_s}{\eta}\left(\boldsymbol{u}^{**}-\boldsymbol{u}^{n+1}\right)\ . \tag{4.22}
$$

By factorisation we can rearrange the previous equation to make the penalty density factor γ_s appear:

$$
\frac{\boldsymbol{u}^{n+1}-\boldsymbol{u}^{**}}{\Delta t}=-\frac{1}{\gamma_s\rho}\nabla P^{n+1/2}\,. \tag{4.23}
$$

The new pressure term however needs to be computed first. To achieve this, the operator of divergence is applied to Eq. 4.23 giving rise to a modified pressure Poisson equation:

$$\begin{aligned}\nabla \cdot \left(\frac{1}{\gamma_s \rho} \nabla P^{n+1/2}\right) &= \frac{\nabla \cdot \boldsymbol{u}^{**}}{\Delta t} - \frac{\nabla \cdot \boldsymbol{u}^{n+1}}{\Delta t} \\ &= \frac{\nabla \cdot \boldsymbol{u}^{**}}{\Delta t} - \frac{\mathscr{Q}_s^{n+1}}{\Delta t} \,.\end{aligned} \tag{4.24}$$

Thanks to Eq. 4.24 we manage to penalise the final velocity while satisfying the continuity constraint. We can also observe the appearance of the solid mass source term. Thanks to Eq. 4.9 we know that $\nabla \cdot \boldsymbol{u}^{n+1} = \mathscr{Q}_s^{n+1}$, which will guarantee a null divergence for the fluid velocity, $\nabla \cdot \boldsymbol{u}_f^{n+1} = 0$.

It is also interesting to note that γ_s acts as a modifier for the density, resulting in a higher density value inside the solid domain. In contrast to the variable density used in two-phase flows, which is based on the rule of mixtures, the modified density $\gamma_s \rho$ originates entirely from the penalty method, where only the fluid density ρ is defined. The implied solid density is directly proportional to the ratio $\alpha = \eta / \Delta t$.

4.3.2 Discretised pressure Poisson equation

In order to update the pressure to correct the predicted velocity, a Poisson equation needs to be solved. In this method, taking into account the implicit penalty term and the added mass sources, the modified PPE of Eq. 4.24 is obtained. The discretised form in time and space of this equation when integrated over the domain reads:

$$\begin{aligned}\sum_{ik} \frac{1}{\gamma_{ik} \rho} \frac{(P_k - P_i)}{\boldsymbol{\Delta x_{ik}}} \boldsymbol{dA_{ik}} =& \frac{1}{\Delta t}\left[\sum_{ik} \boldsymbol{u}_{ik}^{**} \cdot \boldsymbol{dA_{ik}}\right] \\ & - \frac{1}{\Delta t}\left[\sum_{ik} (\phi_s \boldsymbol{u}_s)_{ik}^{n+1} \cdot \boldsymbol{dA_{ik}} + \left[\frac{\phi_{s,i}^{n+\frac{3}{2}} - \phi_{s,i}^{n+\frac{1}{2}}}{\Delta t}\right] \Delta\Omega_i\right] \,,\end{aligned} \tag{4.25}$$

where ik indicates the pair index between two nodes i and k, $\Delta\Omega_i$ is the volume of the control volume around the node i, $\boldsymbol{dA_{ik}}$ is the surface of contact of the control volumes defined by i and k. The penalty density factor of the pair ik is computed as:

$$\frac{1}{\gamma_{ik}} = \frac{1}{2}\left[\frac{1}{\gamma_i} + \frac{1}{\gamma_k}\right] \,. \tag{4.26}$$

The modified PPE leads to a linear system of the form $Ax = B$. When expressed for the node i and the neighbour nodes k:

$$A_{ik} x_{ik} = \sum_{ik} \frac{1}{\gamma_{ik} \rho} \frac{1}{\boldsymbol{\Delta x_{ik}}} (P_k - P_i) \boldsymbol{dA_{ik}} \,, \tag{4.27}$$

$$B_{ik} = \frac{1}{\Delta t}\left[\sum_{ik} \boldsymbol{u}_{ik}^{**} \cdot \boldsymbol{dA_{ik}} - \sum_{ik} (\phi_s \boldsymbol{u}_s)_{ik}^{n+1} \cdot \boldsymbol{dA_{ik}} - \left[\frac{\phi_{s,i}^{n+\frac{3}{2}} - \phi_{s,i}^{n+\frac{1}{2}}}{\Delta t}\right] \Delta\Omega_i\right] \tag{4.28}$$

4.3.3 Estimation of the resistive force acting on the body

In the usual IBM framework, the resistive force $\boldsymbol{F}$ acting on the solid body from the fluid can be computed as the negative sum of the forcing terms applied. This is true in the case of stationary bodies. In the case of a mobile body, the sum of the forcing terms is equal to the total force $\boldsymbol{F}_{\mathrm{tot}}$ experienced by the solid's immersed surface; this includes the forces from the external fluid, $\boldsymbol{F}$, and the internal fluid, $\boldsymbol{F}_{\mathrm{in}}$, as explained by Suzuki et al. [93]. So, to compute the resistive force the following expression is used taking into account the internal force needed to move the fluid of inside the solid domain:

$$\boldsymbol{F} = \underbrace{-\int_{\Omega} \rho\, \mathbf{f}_{\mathrm{IB}}\, \mathrm{d}V}_{\boldsymbol{F}_{\mathrm{tot}}} + \underbrace{\int_{\Omega_s} \rho \frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t}\, \mathrm{d}V}_{\boldsymbol{F}_{\mathrm{in}}} . \tag{4.29}$$

In the VOS-IVP method, as seen in Eq. 4.10, we have the additional forcing term $\mathscr{P}_s$ acting alongside the IB forcing term. In other words the total force is:

$$\boldsymbol{F}_{\mathrm{tot}} = -\int_{\Omega} \rho\, \left(\mathbf{f}_{\mathrm{IB}} + \mathscr{P}_s\right)\, \mathrm{d}V . \tag{4.30}$$

Furthermore, the term $\mathscr{P}_s$ represents the force supplied to the fluid of the solid domain Ω_s so that it follows the prescribed motion of the solid body and is equivalent to the internal force:

$$\boldsymbol{F}_{\mathrm{in}} = \int_{\Omega} \rho\, \mathscr{P}_s\, \mathrm{d}V = \int_{\Omega_s} \rho \frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t}\, \mathrm{d}V . \tag{4.31}$$

By substituting the expressions of the total and internal forces from Eq. 4.30 and Eq. 4.31, respectively, into the Eq. 4.29 it is shown that the resistive force can be computed by simply integrating the IB forcing term over the volume of the computational domain:

$$\boldsymbol{F} = -\int_{\Omega} \rho\, \mathbf{f}_{\mathrm{IB}}\, \mathrm{d}V . \tag{4.32}$$

4.4 Representation of the solid body as a set of particles

4.4.1 Lagrangian framework

For the representation of the immersed body in the method developed in this work, a discretised volume mesh of the geometry is needed in the pre-processing stage, which for simplicity is called 'solid mesh'. This is an unstructured mesh consisting of triangles in 2D and tetrahedrons in 3D with a desired cell size. A set of Lagrangian particles is created by placing a particle at the centre of each cell E^s as shown in Fig. 4.2. In each particle p, the following data is stored:

- the volume V_p of the cell E^s they are placed in,
- the metric $\mathcal{M}_s$ (indicator of the local element size) of the cell E^s they are placed in,
- the coordinates of the nodes, $\mathbf{x}_n$, at the N vertices of the cell,

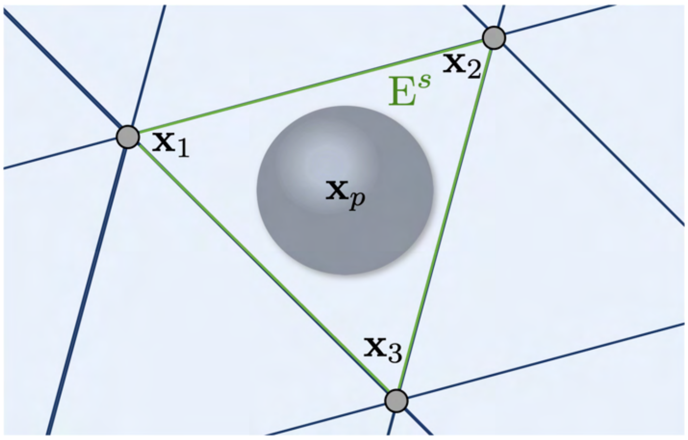


Figure 4.2: 2D schematic representation of a Lagrangian particle p placed in a solid element E^s.

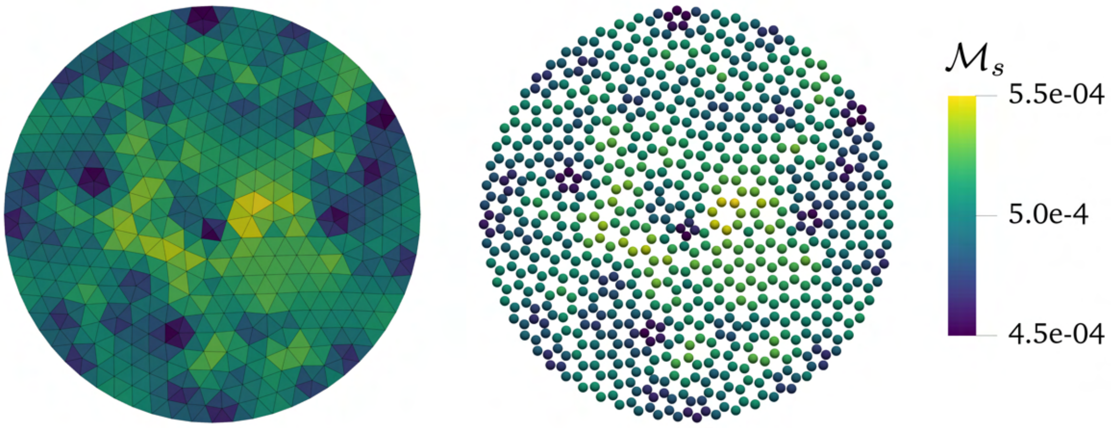


Figure 4.3: Pre-processing stage for solid body representation where a solid mesh is created and a Lagrangian particle is placed at each element storing the information of the elementary volume and the size of the element it represents.

- and the coordinates of the barycentre $\mathbf{x}_p$ of the cell where the particle is placed computed as:

$$\mathbf{x}_p = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n\,. \tag{4.33}$$

Fig. 4.3 shows an example of a discretised 2D cylinder, coloured by the metric, where the solid mesh on the left is replaced by the Lagrangian particles on the right.

This set of Lagrangian particles is then imported to the simulations in order to represent the solid volume and the solid movement in the Eulerian computational domain. After applying a

Lagrangian displacement of the particles, two new fields are computed on the Eulerian mesh. First, the volume contained by the particles is projected on the Eulerian mesh registering the local solid volume contained in each control volume. By dividing this quantity by the total volume of the cell, the local solid volume fraction field can be computed. Lastly, the solid velocity field is computed according to the prescribed motion of the solid and the solid volume fraction.

4.4.2 Lagrangian movement of solid particles

When the particles are imported at the beginning of the simulation their initial position $\mathbf{x}_p^0$ is saved. Based on this initial position a set of transformations can be prescribed to the solid particles under the assumption of a rigid solid body. The current solid movement types are: rotation, translation and oscillation. These operations can be applied at the same time by simply adding them. However, the rotation operation is always first.

4.4.2.1 Rotation

In the case of a prescribed rotary motion, the inputs required are the rotation axis $\boldsymbol{r} = \left(r_1\ r_2\ r_3 \right)$ as a unit vector, the coordinates of the rotation centre $\mathbf{x}^{\mathrm{R}}$ and the rotational speed ω $[\mathrm{rad.s}^{-1}]$. This way a rotation matrix $\boldsymbol{R}$ can be defined as:

$$\boldsymbol{R} = \begin{pmatrix} r_1 r_1(1-c)+c & r_1 r_2(1-c)-r_3 s & r_1 r_3(1-c)+r_2 s \\ r_2 r_1(1-c)+r_3 s & r_2 r_2(1-c)+c & r_2 r_3(1-c)-r_1 s \\ r_3 r_1(1-c)-r_2 s & r_3 r_2(1-c)+r_1 s & r_3 r_3(1-c)+c \end{pmatrix}, \tag{4.34}$$

where t [s] is the current physical time, $c = \cos(\omega t)$ and $s = \sin(\omega t)$.

Thus, at each time-step the new coordinates of any particle p can be computed as:

$$\mathbf{x}_p = \boldsymbol{R}(\mathbf{x}_p^0 - \mathbf{x}^{\mathrm{R}}) + \mathbf{x}^{\mathrm{R}}, \tag{4.35}$$

or in more detail:

$$\mathbf{x}_p = \begin{pmatrix} R_{11}(\mathrm{x}_{p,1}^0 - \mathrm{x}_1^{\mathrm{R}}) + R_{12}(\mathrm{x}_{p,2}^0 - \mathrm{x}_2^{\mathrm{R}}) + R_{13}(\mathrm{x}_{p,3}^0 - \mathrm{x}_3^{\mathrm{R}}) \\ R_{21}(\mathrm{x}_{p,1}^0 - \mathrm{x}_1^{\mathrm{R}}) + R_{22}(\mathrm{x}_{p,2}^0 - \mathrm{x}_2^{\mathrm{R}}) + R_{23}(\mathrm{x}_{p,3}^0 - \mathrm{x}_3^{\mathrm{R}}) \\ R_{31}(\mathrm{x}_{p,1}^0 - \mathrm{x}_1^{\mathrm{R}}) + R_{32}(\mathrm{x}_{p,2}^0 - \mathrm{x}_2^{\mathrm{R}}) + R_{33}(\mathrm{x}_{p,3}^0 - \mathrm{x}_3^{\mathrm{R}}) \end{pmatrix} + \begin{pmatrix} \mathrm{x}_1^{\mathrm{R}} \\ \mathrm{x}_2^{\mathrm{R}} \\ \mathrm{x}_3^{\mathrm{R}} \end{pmatrix}. \tag{4.36}$$

4.4.2.2 Translation

For a simple translation of the solid body at a constant speed the required inputs are the direction unit vector $\boldsymbol{r}_t$ and the constant movement speed v $[\mathrm{m.s}^{-1}]$. The new coordinates of a particle p at time t are computed as:

$$\mathbf{x}_p = \mathbf{x}_p^0 + (\mathrm{v}t)\boldsymbol{r}_t. \tag{4.37}$$

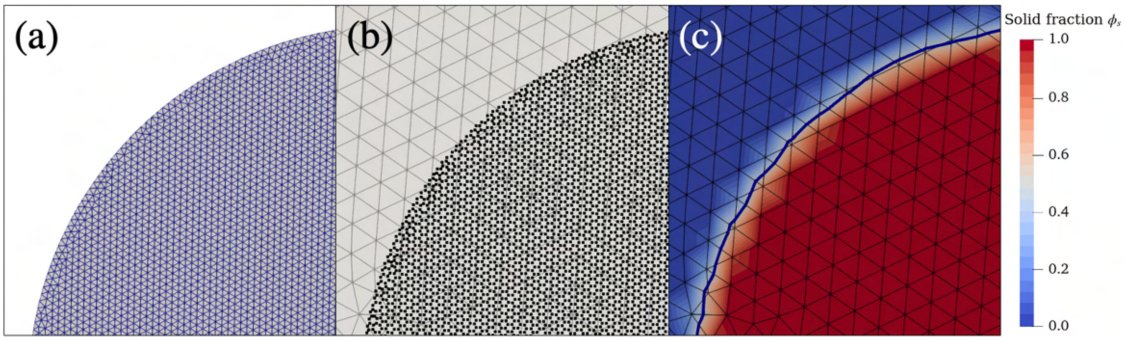


Figure 4.4: Pre-processing stage for solid body representation: (a) a solid mesh is created, (b) a Lagrangian particle is placed at each element and stores the information of the elementary volume and (c) the volume of the particles are interpolated onto the Eulerian mesh resulting in the computation of the solid volume fraction field ϕ_s.

4.4.2.3 Oscillation

For an oscillating solid body the required inputs are the oscillation axis as the unit vector $\boldsymbol{r}_o$, the oscillating amplitude A_o and the oscillating frequency f_o. The oscillation follows a sinusoidal evolution. The new coordinates of a particle p at time t are computed as:

$$\mathbf{x}_p = \mathbf{x}_p^0 + \mathrm{A}_o \sin(2\pi \mathrm{f}_o t)\boldsymbol{r}_o \,. \tag{4.38}$$

4.4.3 Projection of Lagrangian solid volume to Eulerian VOS field

During the computation stage, before the advection of the velocity at each iteration, the particles are relocated on the Eulerian mesh according to the prescribed solid motion. Then the volume carried by the particles is projected onto the Eulerian mesh resulting in the computation of the local quantity of solid volume, and by extension, the local solid and fluid fractions ϕ_s and ϕ_f, respectively, as illustrated by Fig. 4.4. The solid volume $V_{s,i}$ at grid node i is given by:

$$V_{s,i} = \sum_{p|\boldsymbol{x}_p \in E_i} \mathrm{V}_p W_{i,p} \,. \tag{4.39}$$

The subscript p denotes the properties of the p^{th} particle, E_i is the set of elements adjacent to the grid node i, and $W_{i,p}$ is the weight of the linear interpolation used. In our work, only triangular or tetrahedral elements are used in 2D and 3D cases, respectively, so the linear interpolation weights can be computed as:

$$W_{i,p} = \frac{|\boldsymbol{x}_p - \boldsymbol{x}_{fi}| \cdot \boldsymbol{S}_{fi}}{\sum_{i' \in \mathcal{N}(E_p)} |\boldsymbol{x}_p - \boldsymbol{x}_{fi'}| \cdot \boldsymbol{S}_{fi'}} \,, \tag{4.40}$$

where E_p is the element containing particle p, and $\mathcal{N}(E)$ is the set of nodes i' of the element E and $\boldsymbol{S}_{fi'}$ the vector area of the face fi' opposite to the node i. By taking the ratio of the local solid

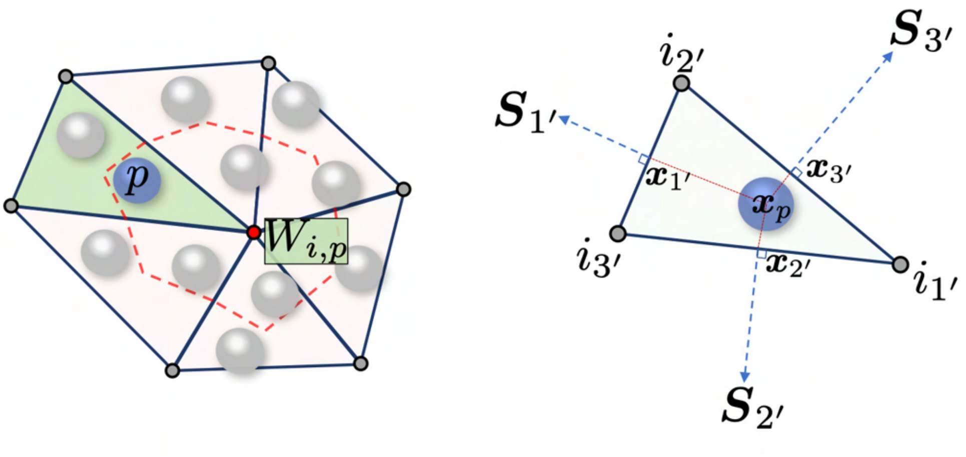


Figure 4.5: 2D representation of six elements around the node i (red). The control volume of the node i is delimited by the red lines. A solid particle p is located in the green coloured element. The interpolation weight of the particle p at the node i is marked as $W_{i,p}$ and can be computed using the surface vectors $\boldsymbol{S}_{1'}, \boldsymbol{S}_{2'}, \boldsymbol{S}_{3'}$, situated opposite of the element nodes $i_{1'}, i_{2'}, i_{3'}$, and the distance of the particle position x_p from the centres of the element edges $\boldsymbol{x}_{1'}, \boldsymbol{x}_{2'}, \boldsymbol{x}_{3'}$ as expressed in Eq. 4.40.

volume over the total cell volume we can compute the local solid volume fraction ϕ_s as shown in Eq. 4.1.

The benefits of the VOS representation of the immersed body is twofold. First, the penalty mask used, χ_s, is easily defined by a sharp Heaviside function of the solid volume fraction. Secondly, the rigid body movement is imposed on the Lagrangian particles and at each iteration the volume fraction is recomputed, hence the total volume of the solid in the Eulerian mesh is inherently conserved in time. Also, the added operations to represent the movement of the solid consist of Lagrangian displacement of the particles and their relocalisation on the processors in a parallel computing configuration. In terms of computational cost, these operations are less costly when compared to fully Eulerian approaches previously tested in YALES2 such as constructing a Level-Set function from an STL surface to represent the immersed surface and then displacing it. The last method would also need specific additional treatment to conserve the volume of the immersed solid.

4.4.4 Solid velocity field

In contrast to the Lagrangian displacement of the solid particles, the solid velocity field is computed directly onto the Eulerian mesh of the computational domain based on the solid volume fraction field at the current time-step and the prescribed motion of the rigid solid. As explained for the operations for displacing the particles, two or more types of movement may occur at the same time, and the final velocity of the solid object is a combination of the velocities prescribed by each of the movements. The velocity of a solid body k can be computed in space and time as:

$$\boldsymbol{u}_{s,k} = \omega_k \boldsymbol{r}_k \times (\boldsymbol{x} - \mathbf{x}_k^{\mathrm{R}}) + \mathrm{v}_k \boldsymbol{r}_{t,k} + \mathrm{A}_{o,k} 2\pi \mathrm{f}_{o,k} \cos(2\pi \mathrm{f}_{o,k} t) \boldsymbol{r}_{o,k}\,, \tag{4.41}$$

where the subscript 'k' has been added to all the quantities associated to the displacement operations acting on the specific solid k.

In many applications it is possible to have multiple immersed bodies in the same simulation undergoing different displacement operations. In some cases, such as in gearboxes, the surfaces of two or more solids can come very close to each other. Depending on the size of the computational cells, there may be multiple solids present in a cell with different solid velocities. To resolve this issue, we define the mean solid velocity field at each position in space (i.e. in each node) as the sum of the different solid velocities weighted by the local solid volume fraction of the solids:

$$\boldsymbol{u}_s = \frac{\sum_k \phi_{s,k} \boldsymbol{u}_{s,k}}{\sum_k \phi_{s,k}}\,. \tag{4.42}$$

This mean solid velocity is used in the definition of the composite velocity shown in Eq. 4.8 as it takes into account the existence of multiple solids in the same computational cell. This formulation also allows to set the solid velocity to zero in the fluid domain where $\phi_s = 0$.

4.4.5 Restrictions on the solid cell size

The choice for the characteristic cell size of the solid mesh $\mathcal{M}_s$ needs to be taken into consideration depending on the Eulerian fluid mesh cell size $\mathcal{M}_f$. From Fig. 4.5 it can be understood that if $\mathcal{M}_s$ is larger than $\mathcal{M}_f$, the solid particles will have more distance between them. This brings the risk of having some elements of the Eulerian mesh with no particles at all and some elements containing particles with a volume larger than the computational cell volume. In this case, the solid volume fraction field suffers from discontinuities in the form of 'holes' inside the solid region, with $\phi_s < 1$, accompanied by spots where the solid volume fraction overshoots, with $\phi_s > 1$. This is demonstrated in Fig. 4.6 where four different ratios of $\mathcal{M}_s/\mathcal{M}_f$ where tested. The fluid metric remained unchanged, $\mathcal{M}_f = 2.5$, while the solid metric varied from $\mathcal{M}_s = 5.5$ to $\mathcal{M}_s = 0.7$. For ratios larger than 1 the discontinuities are clearly visible and the overshoots may even reach values of $\phi_s = 1.93$, which would not be acceptable from a physical point of view. For ratios smaller than 1, the peaks and troughs are greatly diminished and the maximum value of ϕ_s is closer to the target value of unity inside the solid region. Thus the criterion for the solid mesh cell size that needs to be satisfied is the following:

$$\mathcal{M}_s/\mathcal{M}_f \leq 1\,. \tag{4.43}$$

4.5 Conservative & clipped solid volume fraction fields

The process of creating the solid volume fraction field by projecting the solid particle volume onto the mesh has been explained in section 4.4.3. It has also been explained that depending on the ratio of the cell-sizes of the solid and fluid meshes, and due to interpolation errors, the scalar field ϕ_s may not be perfectly uniform inside the solid region. Despite these deviations from the value

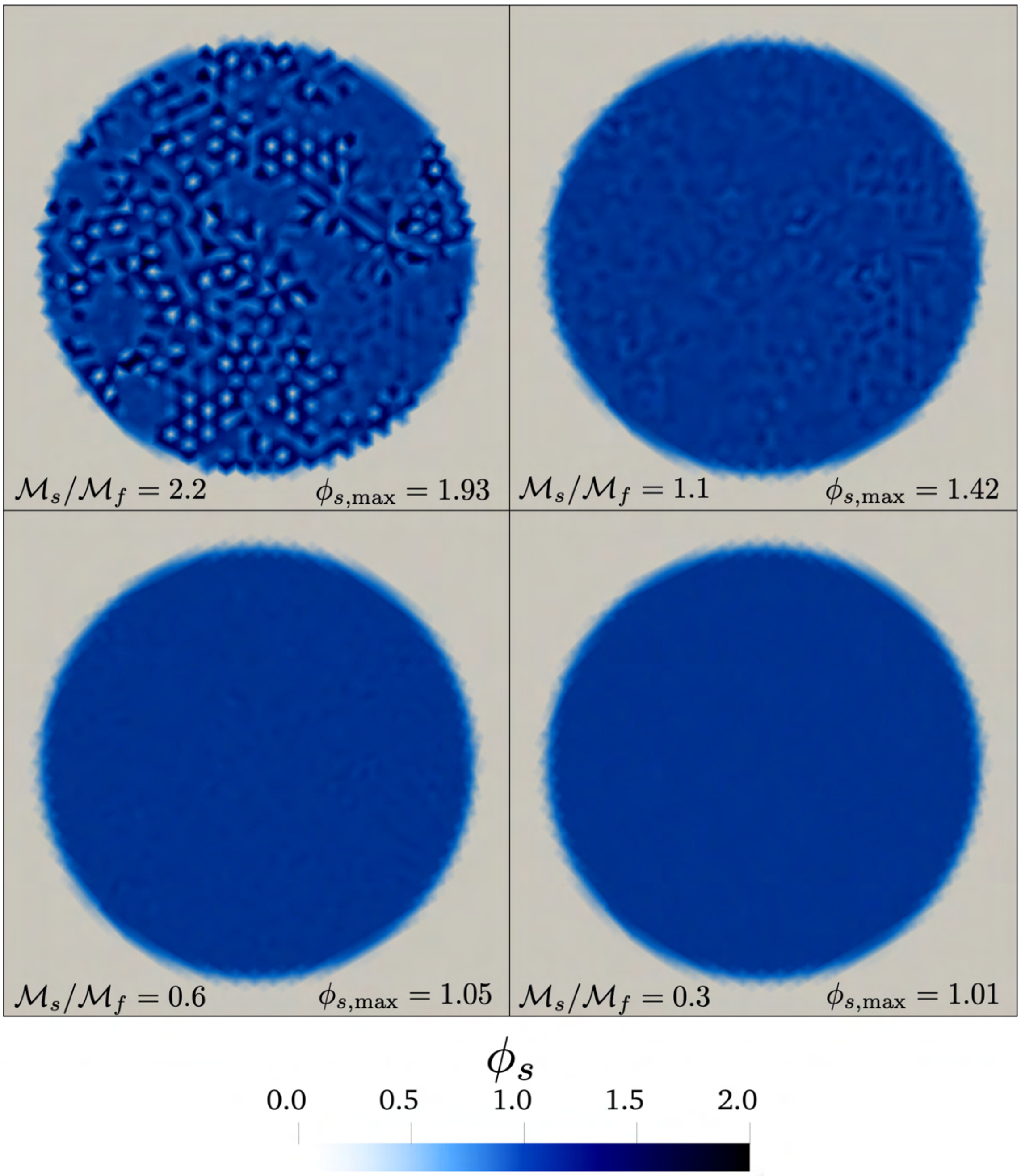


Figure 4.6: Solid volume fraction field and the maximum value for four different ratios of the solid mesh metric $\mathcal{M}_s$ to the fluid mesh metric $\mathcal{M}_f$.

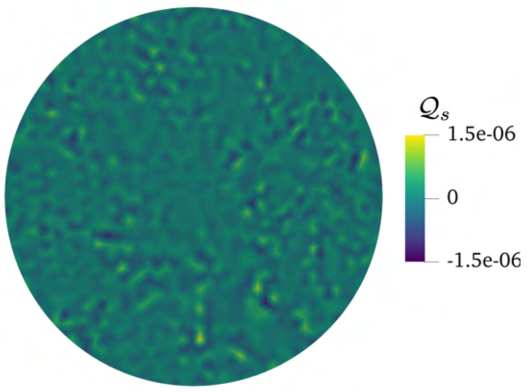


Figure 4.7: Variations of the mass source term $\mathscr{Q}_s$ $[1/s]$ around 0 inside the solid region in a case of a rotating 2D cylinder with $\mathcal{M}_s/\mathcal{M}_f = 0.3$ and $\Delta x = 0.025D$.

of 1, the integral of ϕ_s will always remain equal to the total volume of the initial solid mesh near machine precision. Hence we could call this the conservative solid volume fraction field.

In the case of a rigid immersed body in motion, after the prescribed movement of the solid particles, the solid volume fraction field will remain conservative in nature but will not be the same as the field of the previous iteration. This creates undesirable variations in time of the local solid volume fraction inside the solid region. This affects the source terms of the VOS-IVP method which include the derivative $\partial\phi_s/\partial t$ in their computation. To demonstrate this effect, a simulation was carried out on a 2D cylinder of diameter D rotating around its centre at a rotational speed $\omega = 15\ \mathrm{rad.s}^{-1}$. The fluid mesh cell size is set at $\Delta x = 0.025D$ and the solid mesh satisfies $\mathcal{M}_s/\mathcal{M}_f = 0.3$. This particular case was chosen due to the fact that despite the rotational movement of the immersed cylinder and the existence of momentum sources inside the solid region, the projected image of the cylinder, or in other words the solid volume fraction field, is supposed to remain in the same exact position from one iteration to the next. This would imply that the mass source term shown in Eq. 4.14 is zero, $\mathscr{Q}_s = 0$. In practice, Fig. 4.7 demonstrates the existence of non-zero values due to the small variations of the conservative volume fraction field ϕ_s.

The conservative nature of ϕ_s is essential in the computation of the mass source term $\mathscr{Q}_s$ for the numerical stability of the code when solving the PPE. However, this is not true for the source term $\mathscr{P}_s$ in the momentum equation (Eq. 4.10). In order to produce a more physical momentum source one would need that the values of ϕ_s remain constantly equal to 1 at the interior of the solid object. To achieve this, a new field is created by amplifying the higher values of ϕ_s and clipping them at the maximum value of 1. This clipped solid volume fraction field ϕ_s^{clip} is defined as:

$$\phi_s^{\mathrm{clip}} = \begin{cases} \min\left[\phi_s(1+6(\phi_s-0.7));1\right] & \text{if} \quad \phi_s \geq 0.7\,, \\ \phi_s & \text{if} \quad \phi_s < 0.7\,. \end{cases} \tag{4.44}$$

The relationship used has no particular physical meaning and was found empirically.

The difference between the conservative and the clipped solid volume fraction fields in the case of the rotating 2D cylinder can be seen in Fig. 4.8. The effects of the clipping is a completely

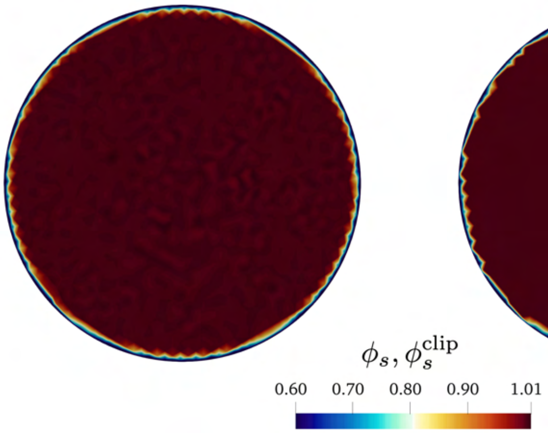


Figure 4.8: Comparison between the conservative solid volume fraction field ϕ_s on the left and the clipped solid volume fraction ϕ_s^{clip} on the right.

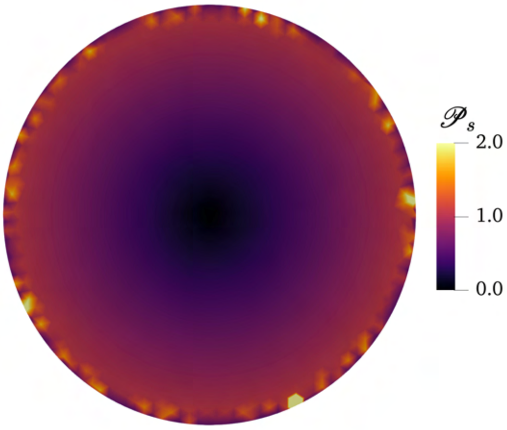


Figure 4.9: Variations of the momentum source term $\mathscr{P}_s$ $[m/s^2]$ inside the solid region in a case of a rotating 2D cylinder with $\mathcal{M}_s/\mathcal{M}_f = 0.3$ and $\Delta x = 0.025D$.

even field with $\phi_s = 1$ well inside the solid region but with a steeper slope for $\phi_s \geq 0.7$. These characteristics are made evident by examining the field of the momentum source term as shown in Fig. 4.9. The momentum source is equal to 0 at the centre of the cylinder and increases smoothly in value along the radius. At the solid surface the momentum source decreases again due to the decreasing value of ϕ_s. The only undesired artefacts is the very sharp profile of ϕ_s^{clip} between the values of 0.7 and 1. Indeed this sharp gradient generates discontinuous overshoots in these areas and is a source of spurious oscillations in the computation of the resistive forces between solid and fluid, as it will be discussed in section 5.4.

4.6 Numerical algorithm

To summarize, the simulations using the VOS-IVP method are carried out through the following procedure:

- Pre-processing stage:
    1. Generation of the meshes for the solid body and the computational domain.
    2. Creation of a set of Lagrangian particles whose volume and position is given by the discretized cells of the solid mesh.
- During the calculation:
    1. Computation of the new position of the solid particles and the solid velocity from the prescribed motion of the immersed body.
    2. Computation of the new solid volume fraction field and the penalty mask by interpolating the Lagrangian particle volume on the Eulerian mesh.
    3. Computation of the clipped solid volume fraction field for the momentum source term.
    4. Computation of the mass and momentum source terms based on the prescribed motion of the immersed body.
    5. Prediction of the intermediate composite velocity field.
    6. Computation of the new pressure gradient by solving a modified pressure Poisson equation.
    7. Correction of the composite velocity field with the updated pressure term.
    8. Computation of the resistive force by integrating the penalty forcing term over the whole domain.

The method is presented step-by-step in Algorithm 1.

Algorithm 1 VOS-IVP method algorithm

```
Create Lagrangian solid particles from initial solid mesh
Compute initial solid fraction $\phi_s^0$ by projecting particle volume $V_p$ on Eulerian grid
while $t < t_{\text{final}}$ do
    if Moving immersed body=True then
        Compute new $\boldsymbol{x}_p^{n+1}$ and $\boldsymbol{u}_s^{n+1}$ from prescribed motion
        Compute conservative solid fraction $\phi_s^{n+1}$ and clipped solid fraction $\phi_s^{\text{clip},n+1}$
        Compute penalty mask $\chi_s = \mathrm{H}(\phi_s^{n+1})$ and penalty density factor $\gamma_s$
        Compute mass and momentum sources: $\mathscr{Q}_s^{n+1}(\phi_s^{n+1}, \boldsymbol{u}_s^{n+1})$, $\mathscr{P}_s^{n+1}(\phi_s^{\text{clip},n+1}, \boldsymbol{u}_s^{n+1})$
    end if
    Advance the composite velocity:
    $\boldsymbol{u}^* = {\gamma_s}^{-1}\left[\left[\boldsymbol{u}^n - \Delta t\left(\nabla\cdot(\boldsymbol{u}^*\otimes\boldsymbol{u}^n) + \rho^{-1}\nabla P^{n-1/2} - \rho^{-1}\nabla\cdot\boldsymbol{\tau}^n - \mathscr{P}^{n+1}\right)\right] + (\gamma_s - 1)\boldsymbol{u}_s^{n+1}\right]$
    Remove previous pressure gradient:
    $\boldsymbol{u}^{**} = \boldsymbol{u}^* + \Delta t\nabla P^{n-1/2}/(\gamma_s\rho)$
    Solve modified pressure Poisson equation:
    $\nabla\cdot\left[\nabla P^{n+1/2}/(\gamma_s\rho)\right] = (\nabla\cdot\boldsymbol{u}^{**} - \mathscr{Q}_s^{n+1})/\Delta t$
    Correct the composite velocity with new pressure gradient:
    $\boldsymbol{u}^{n+1} = \boldsymbol{u}^{**} - \Delta t\nabla P^{n+1/2}/(\gamma_s\rho)$
    Post-processing
end while
```

Chapter 5

Validation

Contents

5.1 Introduction

In this section, several benchmark flow problems are solved using the VOS-IVP method to demonstrate its ability to obtain accurate results for different configurations. We examine the cases of flows around a cylinder of diameter D or an airfoil of chord c. The main quantities to compare between the numerical results with the reference data are the drag and lift coefficients (C_D and C_L) and the Strouhal number defined as:

$$C_D = \frac{2F_x}{\rho S U_\infty^2} \ , \ \ C_L = \frac{2F_y}{\rho S U_\infty^2} \ , \ \ St = \frac{f_s D}{U_\infty} \, . \tag{5.1}$$

F_x and F_y are the stream-wise and cross-flow total forces, respectively. U_∞ is the free-stream velocity, S is the cross-sectional area of the body and f_s the vortex shedding frequency in unsteady flows. Numerically, we will be interested in the mean value ($\langle\bullet\rangle$) and the mean fluctuation ($\bullet'$) of the variables. The shedding frequency f_s is computed through a Fast-Fourier Transform (FFT) analysis as the fundamental frequency of the lift's mean fluctuation.

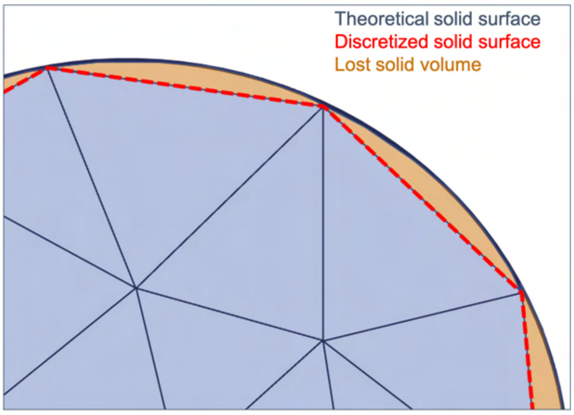


Figure 5.1: 2D schematic of the difference between a continuous surface and its discretised shape showing the lost volume of the discretised mesh.

5.2 Solid volume conservation

To demonstrate that the Lagrangian particles conserve the solid mass independently of the fluid mesh resolution and the movement of the solid, we solved the flow past a 2D cylinder of diameter D and volume V_{cyl}, oscillating with amplitude $y_{\max} = D$ and frequency $f = 2.2\,\mathrm{Hz}$. The theoretical value of the cylinder volume (equivalent to a circular surface in 2D) is computed as:

$$V_{cyl} = \frac{\pi D^2}{4}\,. \tag{5.2}$$

The volume of the discretised solid mesh of $\mathcal{M}_s = D/40$ differs from the theoretical value due to discretisation errors, as shown in Fig. 5.1, by 0.16%. During the simulation, as the solid is represented through a VOS approach, by integrating the solid volume fraction over the domain one should obtain the total volume of the immersed object. Let us define the error of the computed solid volume relative to the theoretical one as:

$$\varepsilon_s(t) = \frac{|V_{cyl} - \int_\Omega \phi_s(t)\,\mathrm{d}V|}{V_{cyl}}\,. \tag{5.3}$$

For the fluid mesh, two coarse grids were tested with $\Delta_x = D/5$ and $D/10$. Table 5.1 shows the mean and the r.m.s. values of the relative error after four oscillations of the cylinder. The mean value remains constant for both grids and its the same error as the one of the discretised mesh. This affirms the conservative nature of the solid volume fraction field. The r.m.s. value decreases with the finer mesh and can be attributed to interpolation errors. Despite that, both values are essentially near machine precision at an order of magnitude of 10^{-14} %.

The same oscillatory movement was imposed on a 2D square (a shape with no curves) of sides equal to the cylinder diameter D, on the grid of cell-size $\Delta_x = D/5$. The 2D volume of the square is equal to $V_{square} = D^2$. The mean and r.m.s. values of the volume relative error are 2.7×10^{-14} % and 5.9×10^{-14} %, respectively. Both values are essentially zero, further proving the fact that in the case of the cylinder, the errors originated from the discretisation of the curves in the solid mesh.

Table 5.1: Mean and r.m.s. values of the relative error in solid volume for two coarse meshes

$D/\Delta x$	$\langle \varepsilon_s \rangle$ [%]	ε_s' [%]
5	0.16	$2.11 \cdot 10^{-14}$
10	0.16	$1.38 \cdot 10^{-14}$

5.3 Laminar flow around a stationary cylinder

5.3.1 Flow at Reynolds number of 100

A well-researched benchmark problem for many years, the laminar unsteady flow past a two-dimensional stationary cylinder of diameter D is studied. The results of this problem are known to be sensitive to the size of the computational domain, particularly for relatively small values of Reynolds number. The computational domain, shown in Fig. 5.2, extends $15D$ upstream of the solid and $50D$ downstream. The top and bottom boundaries of the domain are placed $15D$ from the centre of the solid, sufficiently far to limit blockage effects, with slip-wall boundary conditions. The inlet velocity U_∞ is kept constant and the Reynolds number, computed as $Re = U_\infty D/\nu$, is imposed by changing the value of the kinematic viscosity. The cylinder is placed in a refined zone of dimensions $[-2D, 10D] \times [-2.5D, 2.5D]$ where the grid-spacing Δx corresponds to $D/\Delta x = 50$. The mesh is composed of 3.73×10^5 elements. The time-step of the simulations is determined by the CFL condition CFL $= 0.9$. The penalisation time-step ratio is $\alpha = 1$. The simulated physical time covers 1000 non-dimensional periods ($t^* = tU_\infty/D$) and all simulations run on 20 CPU cores. The VOS-IVP method is validated for $Re = 100$ against body-fitted (BF) simulations, also carried out with YALES2, and reference data obtained from numerical simulations from the literature [83, 76, 46].

Table 5.2: Mean drag coefficient $\langle C_D \rangle$, mean lift fluctuation C_L', Strouhal number St, wake closure length L_c and time-step Δt for the case of $Re = 100$.

Case	$\langle C_D \rangle$	C_L'	St	L_c	$\Delta t \times 10^{-4}$
Qu et al. [83]	1.317	0.222	0.165	1.41	–
Park et al. [76]	1.330	0.235	0.165	1.42	–
Kravchenko et al. [46]	1.320	0.222	0.164	1.45	–
BF	1.335	0.237	0.167	1.38	1.012
VOS-IVP	1.300	0.215	0.171	1.32	1.072

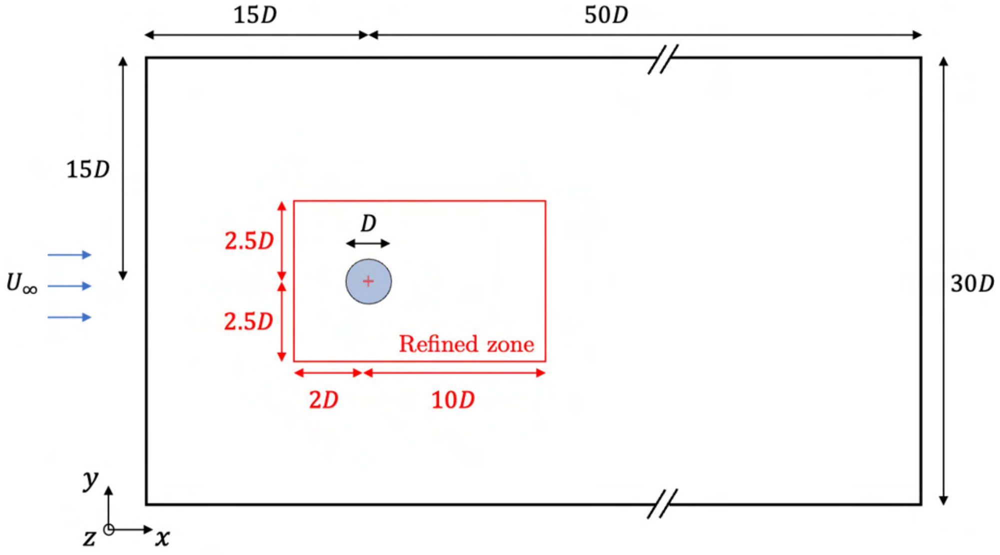


Figure 5.2: Sketch of the domain for the case of the stationary 2D cylinder.

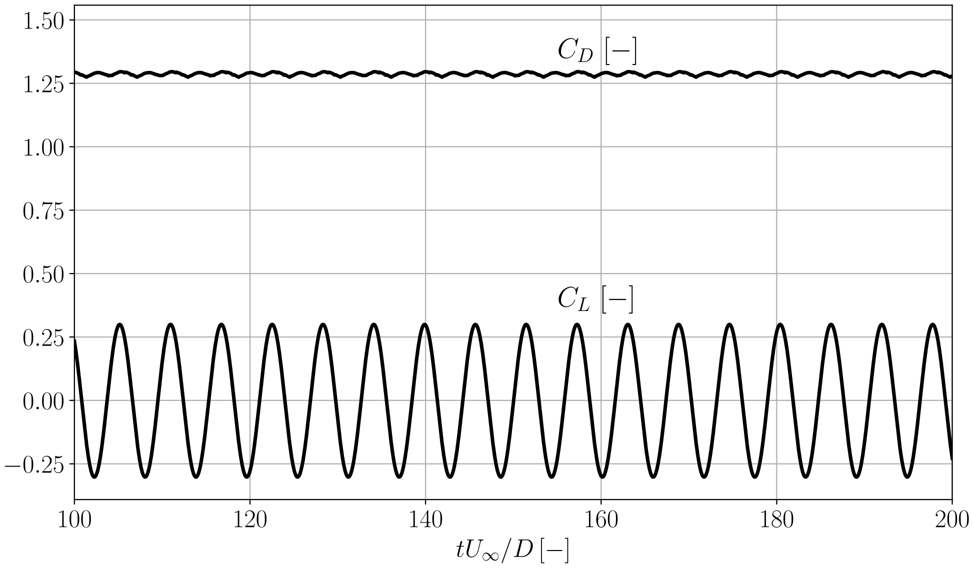


Figure 5.3: Time-series of the drag and lift coefficient predicted with the VOS-IVP method in the case of of $Re = 100$.

In Fig. 5.3 the time series of the drag and lift coefficients are shown and the numerical results of the VOS-IVP appear to be smooth. The frequency of the drag coefficient fluctuation is double

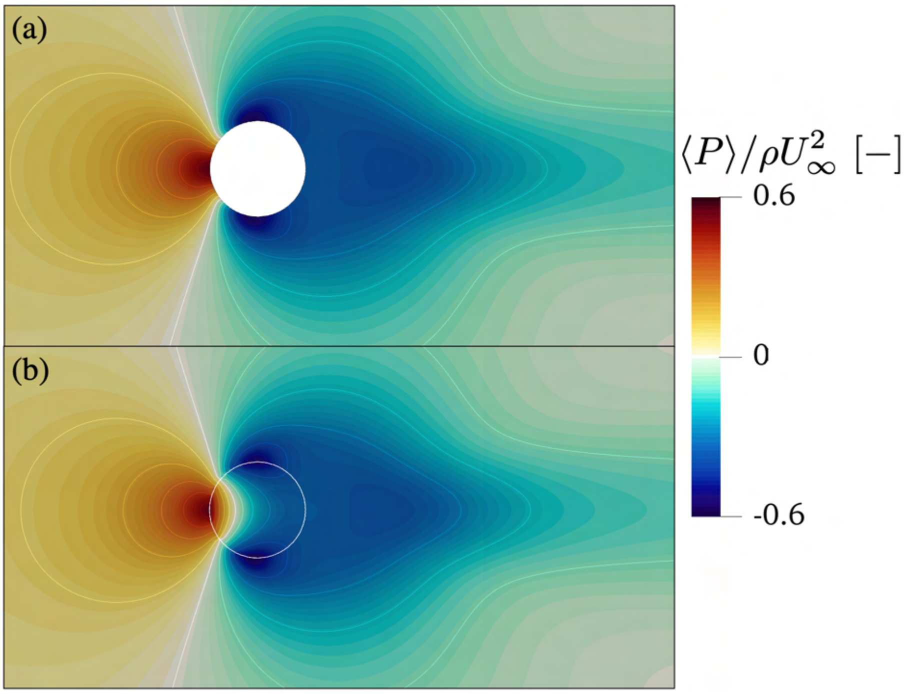


Figure 5.4: Time-averaged dynamic pressure fields of the (a) body-fitted method and (b) the VOS-IVP method for $Re = 100$.

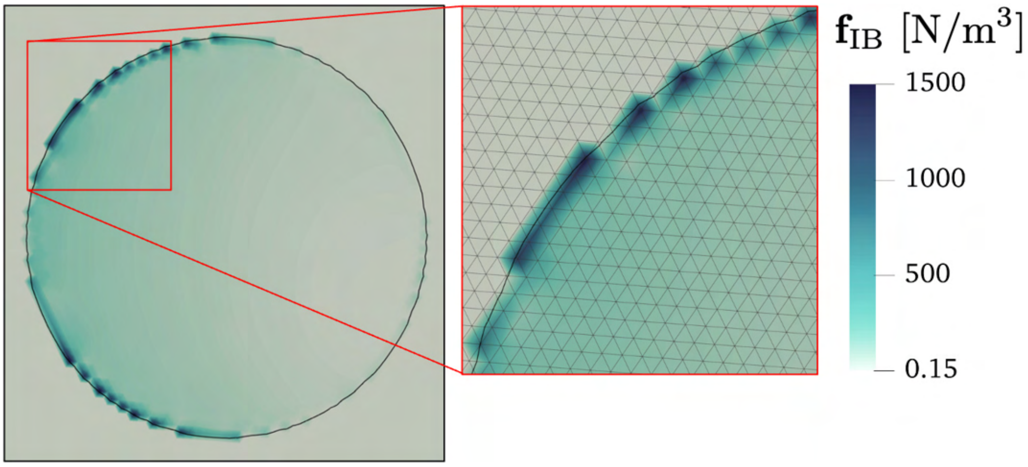


Figure 5.5: Local volumetric penalty forces applied in the solid region. A closer view near the fluid-solid interface is also provided with the grid visible.

the one of the lift fluctuation due to the contribution of the alternating upper and lower vortices to the drag force.

Table 5.2 shows that the mean drag coefficient, the mean fluctuation of lift coefficient, and the Strouhal number match very well with the body-fitted case and the reference data. The wake closure length, i.e. the distance along the wake centre line from the cylinder to the point of zero velocity, is very close to the body-fitted case, but overall underestimated in our simulations compared to the literature. This can be seen in Fig. 5.4 where the time-averaged fields of the pressure are shown for the body-fitted method and the VOS-IVP method. The fields match very well with slight variations near the cylinder. The two zones of low pressure at the upper and lower parts of the cylinder appear smaller in the VOS-IVP results. This shows that the fluid does not decelerate as much when passing the cylinder as in the body-fitted case. The difference stems from the sharp penalty mask used. As seen in Fig. 5.5 the first points where a volumetric forcing term is applied are well within the solid region. Effectively, the cross-sectional surface of the body seen by the fluid is smaller than the real one. This fact also explains the lower values in the aerodynamic quantities for the present method.

The velocity profiles in Fig. 5.6 show the mean and r.m.s. values of the stream-wise and cross-flow velocity components at three different positions ($x = [1D, 2D, 5D]$) for the body-fitted and the VOS-IVP methods. The profiles match very well between the two methods but we can still notice marginally higher velocity values in both directions for the VOS-IVP method, further supporting the previous observations from the mean pressure fields. Fig. 5.7 shows the mean stream-wise velocity along the centre line of the wake. There is a noticeable difference in the region between $x = 1D$ and $x = 3D$ where the wake recovery is slightly faster in the VOS-IVP method. The wake closure length L_c was determined as the point of vanishing $\langle \boldsymbol{u} \rangle$, where there is a change from negative to positive values, along the wake centre line downstream of the solid's surface. The VOS-IVP measures the wake closure at $L_c = 1.32D$ downstream of the object, 4% shorter compared than the body-fitted case.

5.3.2 Mesh dependency study and influence of the implicit penalty term

Table 5.3: Mean drag coefficient $\langle C_D \rangle$, mean lift fluctuation C_L' and Strouhal number St for different mesh sizes in the case of $Re = 100$.

Mesh	$\langle C_D \rangle$	C_L'	St
$D/\Delta x = 10$	1.314	0.217	0.178
$D/\Delta x = 25$	1.297	0.221	0.177
$D/\Delta x = 50$	1.300	0.215	0.171

A mesh dependency study is conducted on the case of section 5.3.1 by coarsening the computational grid. Three mesh sizes were tested: $D/\Delta x = [10, 25, 50]$. The purpose of this study is to examine the influence of the mesh size to i) the aerodynamic coefficients, ii) the penalty force distribution inside the solid and iii) the velocity profile inside the solid when using the VOS-IVP method.

Table 5.3 shows the mean drag coefficient, the r.m.s. of the lift coefficient and the Strouhal number for different mesh sizes. The aerodynamic coefficients do not show any particular trends

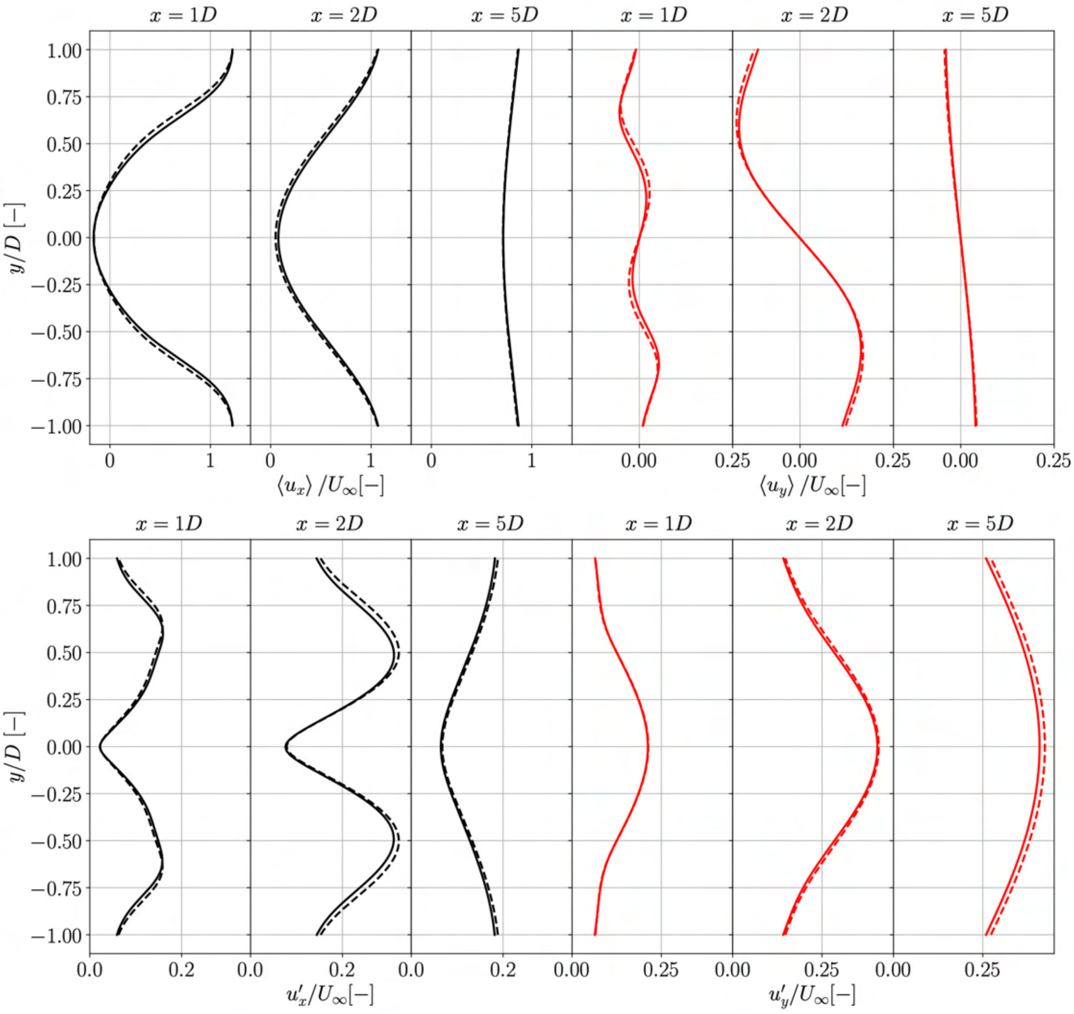


Figure 5.6: Vertical profiles of the mean (top) and r.ms. (bottom) velocity components in stream-wise (black) and cross-flow (red) directions at three different positions, $x = [1D, 2D, 5D]$, for $Re = 100$. Dashed lines: body-fitted case; solid lines: VOS-IVP.

when changing the mesh size, their values remain unchanged. This can be attributed to the relatively simple form of the bluff body. In the case of the Strouhal number, its value decreases with a smaller grid spacing approaching the values found in literature (Table 5.2).

The mesh size influences strongly the accurate imposition of the prescribed solid velocity $\boldsymbol{u}_s = 0$ inside the solid volume due to the change in distribution of the penalty force which serves to bring the fluid at rest. The top of Fig. 5.8 shows the profile of the mean stream-wise component of the penalty force $\langle \mathbf{f}_{\text{IB},x} \rangle$ along a horizontal line passing by the cylinder centre ($y = 0$). As a reminder, the free stream flows from left to right. For the finest grid, the peak force value is inside the solid volume near the left solid-fluid interface at $x/D = -0.5$. As the grid coarsens, the

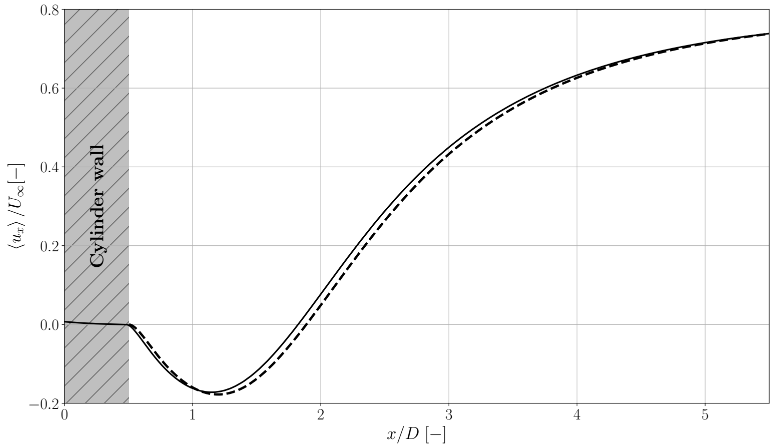


Figure 5.7: Mean stream-wise velocity normalised by the free flow velocity along the centre line of the wake, for $Re = 100$. Dashed line: body-fitted case; solid line: VOS-IVP.

peak force value weakens in magnitude and is applied further inside the solid region. The profile also loses its initial sharpness but further inside the solid, $x/D > -0.2$, all the profiles converge towards the same values. The direct impact of the change in force distribution with the different mesh sizes can be seen at the bottom of Fig. 5.8, which shows the profile of the mean stream-wise component of the composite velocity. It is evident that with coarser grids, the solid velocity is not well imposed. The penalty force is not sufficient to decelerate the fluid fast enough and fairly high positive values of $\langle \boldsymbol{u}_x \rangle$ persist inside the solid. With refining the mesh, the velocity values drop significantly approaching the target value.

Let us now examine the influence of the implicit penalty term of the IVP compared to the forcing of the usual volume penalty method (VP). The same simulations were carried out but with the correction contribution of the penalty force deactivated. Their respective velocity profiles can also be seen in Fig. 5.8. It is evident that the simple penalty method is not as effective as the implicit penalty method developed in this work and for the same mesh size the positive velocity values are higher. For $D/\Delta x = 10$, the velocity fails to reach the 0 value. It is interesting to look at the penalty force profiles for $D/\Delta x = 50$. The VP method gives a sharper profile and the force is applied closer to the solid surface, but this influences only the intermediate velocity field $\boldsymbol{u}^*$. In the correction step the velocity will be modified due to the new pressure gradient in order to satisfy only the incompressibility constraint. This results in smoothing the velocity gradient at the solid surface and thus the higher velocity values. On the contrary, in the IVP method the penalty force comprises a contribution in the prediction step and in the correction step. The correction contribution counteracts with the pressure gradient and results in a smoother penalty force profile. However, the boundary condition of the velocity is imposed in the immersed volume at the same time as the incompressibility constraint, resulting in velocity values closer to the imposed solid velocity.

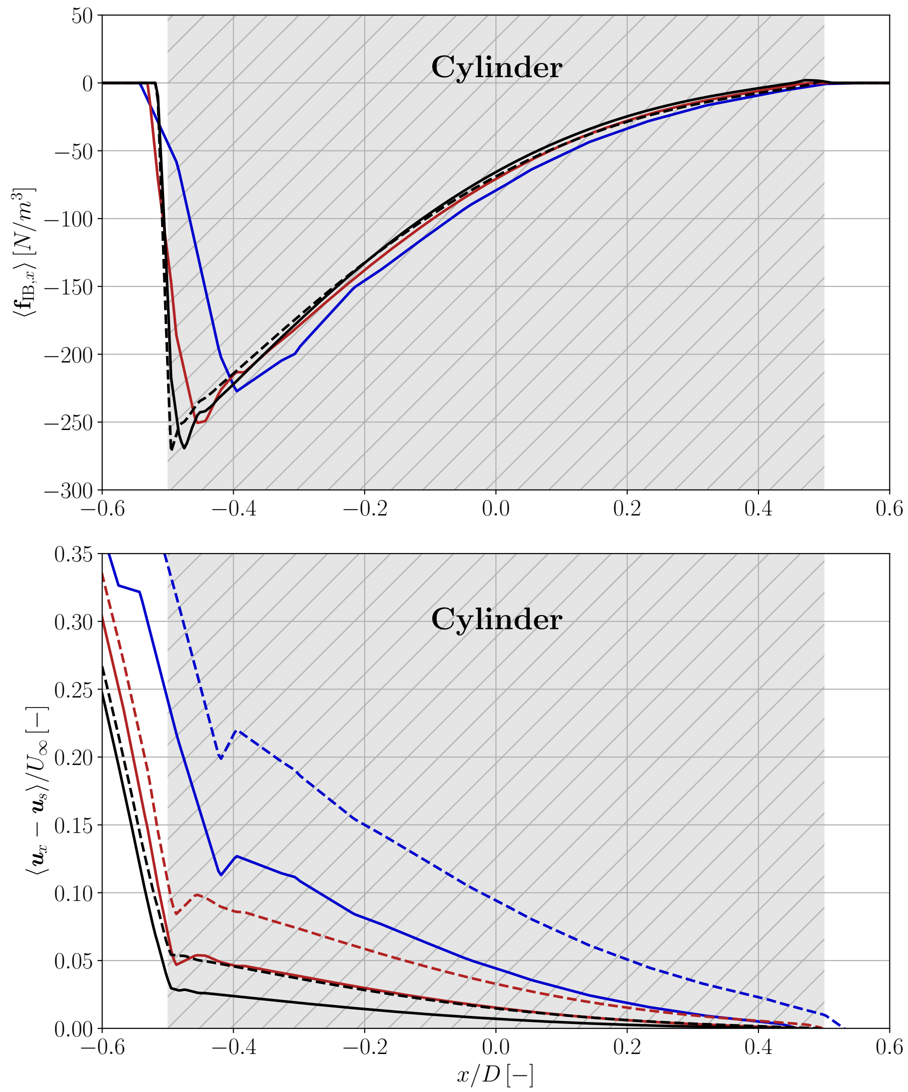


Figure 5.8: Top figure shows the profile of the mean stream-wise component of the penalty force along a horizontal line passing by the cylinder centre. Bottom figure shows the mean stream-wise component of the composite velocity along the same line. Bold lines: IVP; dashed lines: VP. Blue colour: $D/\Delta x = 10$; red colour: $D/\Delta x = 25$; black colour: $D/\Delta x = 50$.

Fig. 5.9 shows, for both VP and IVP methods, the evolution of the mean stream-wise velocity value at the first node inside the solid domain for the different mesh sizes. There appears to be a linear relationship between them for both methods: $\langle \boldsymbol{u}_x \rangle \propto \Delta x$. Here again, the error committed by the IVP approach is lower than the VP approach.

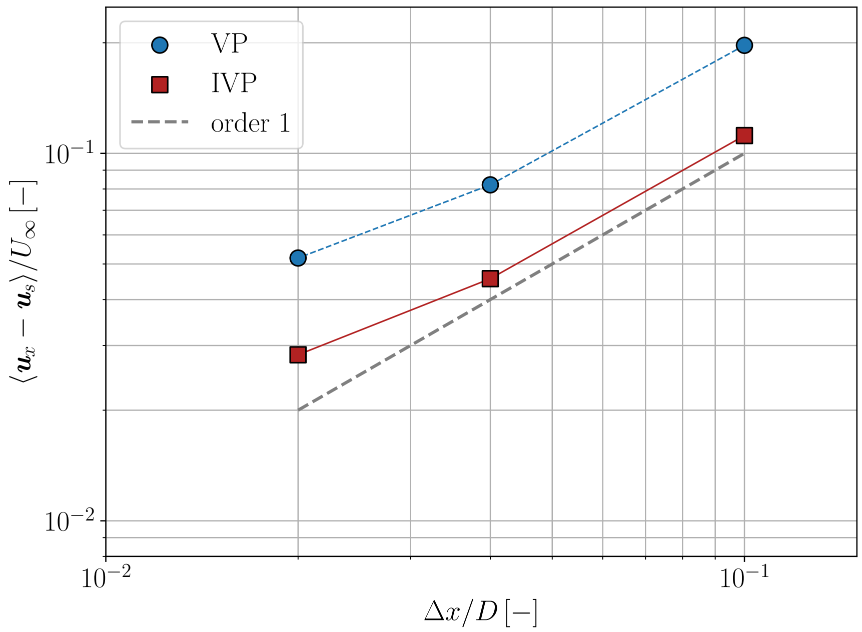


Figure 5.9: Evolution of the mean stream-wise velocity value at the first node inside the solid domain for the different mesh sizes.

5.3.3 Influence of time-step and penalty parameter

A parametric study has been conducted to evaluate the influence of the time-step Δt and the penalty parameter η on the force estimation and the imposition of the solid velocity inside the immersed body. Using the same mesh as before, and keeping the same Reynolds number of $Re = 100$, 200 non-dimensional periods were simulated. The time-step was determined by the CFL condition and the penalty parameter by the penalisation time-step ratio $\alpha = \eta / \Delta t$. Four values of each parameter were tested: $\text{CFL} = [0.09, 0.18, 0.45, 0.9]$ and $\alpha = [0.1, 0.2, 0.5, 1]$. The main tools of comparison are the relative error of the mean drag coefficient compared to the previously mentioned body-fitted case and the L_∞ norm of the velocity magnitude in the region were the penalty term is applied. These quantities are computed as:

$$\varepsilon_{\langle C_D \rangle} = \frac{\left| \langle C_D \rangle_{\text{VOS-IVP}} - \langle C_D \rangle_{\text{BF}} \right|}{\langle C_D \rangle_{\text{BF}}} \, , \tag{5.4}$$

$$\epsilon_\infty = \max \left(\chi_s \phi_s ||\boldsymbol{u}||_2 \right) \, . \tag{5.5}$$

Fig. 5.10 shows the maximum velocity error in the solid region depending on the CFL and α. The error diminishes with smaller time-step values and with smaller penalty parameter values. Furthermore, with smaller time-steps, the order of convergence of ϵ_∞ with respect to η increases from $O(\eta^{1/2})$ to almost $O(\eta^1)$. This result agrees with the findings of Angot et al. [2] who have

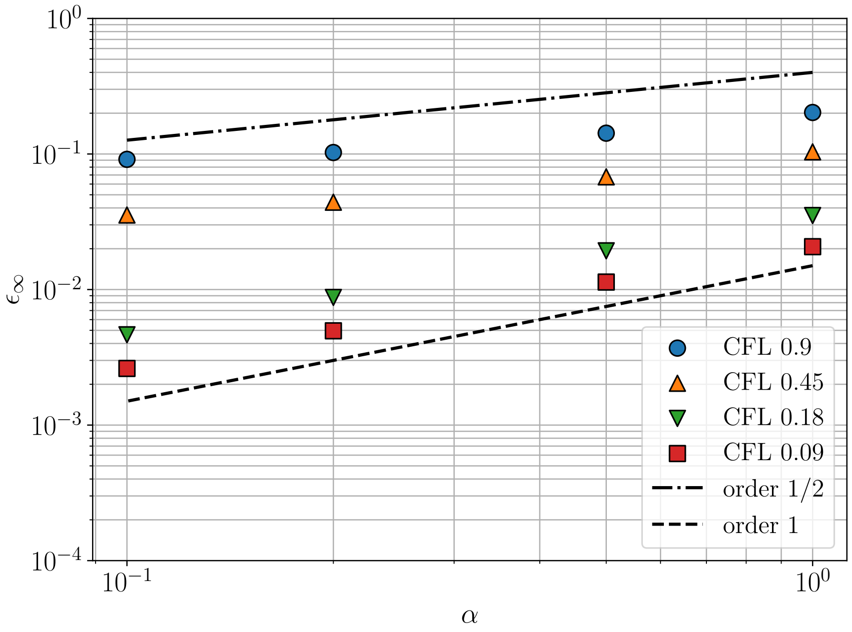


Figure 5.10: Evolution of maximum velocity error inside the solid with the penalisation time-step ratio for different CFL values.

rigorously shown using asymptotic analysis that the solution of the penalized velocity converges to the exact Navier-Stokes equations at the immersed boundaries as $\eta \to 0$ with a global error of $O(\eta^{3/4})$.

When examining the value of $\varepsilon_{\langle C_D \rangle}$ for the different combinations of CFL and α values, there is small variation in the error with the minimum and maximum values of 2.6% and 3.6%, respectively, and no clear tendency can be observed. With higher time-steps, a value of $\alpha = 1$ gives a closer estimation to the body-fitted result. But for the smaller values of Δt, smaller values of α are preferred. From a computational point of view, decreasing the value of α with a constant CFL doesn't have an important influence on the computational cost in the case of stationary solids. However, reducing the CFL limit and forcing smaller time-steps the computational cost increases considerably. The average cost for CFL $= 0.9$ was 20 hCPU while for CFL $= 0.09$ the average cost was 80 hCPU. Hence, there's a compromise to be made between the accurate velocity imposition and the computational cost.

5.3.4 Flow at Reynolds numbers of 50 to 200

Finally, the evolution of the aerodynamic quantities and Strouhal number are examined for a range of low Reynolds numbers, $Re = [50 - 200]$. The same numerical set-up was used as previously described, with CFL $= 0.9$ and $\alpha = 1$. Fig. 5.11 shows the evolution of the mean drag coefficient, the mean fluctuation of the lift coefficient and the Strouhal number with the increase in Reynolds

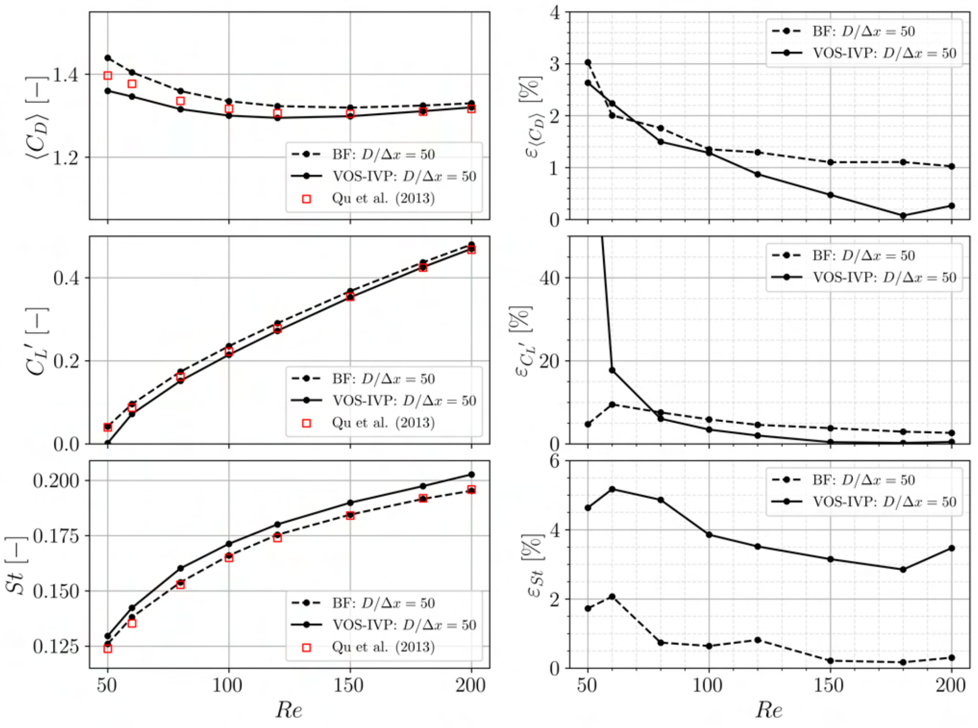


Figure 5.11: Aerodynamic quantities and Strouhal evolution (left) and relative errors compared to Qu et al. [83] (right) for different Reynolds numbers. Squares: reference data from Qu et al. [83]; dashed lines: body-fitted case; solid lines: VOS-IVP.

number. A great agreement can be seen between the predicted results of the VOS-IVP method and both the body-fitted and reference values. This can be further supported by examining the relative errors of the present method and the body-fitted case with respect to the data of Qu et al. [83]. The error in the mean drag coefficient remains under 3% showing an excellent agreement with the reference data. The absolute error compared to the reference remains almost the same for the C_L' values and the relative error remains lower than 10% except for the very first point, at $Re = 50$, where we have $C_L' = 0.001$. In fact, due to the sharp penalty mask the cross sectional area of the body seen by the flow is slightly smaller. This leads to effectively reducing the diameter based Reynolds number of the flow, approaching the critical value $Re_c = 47$ where the laminar shedding regime starts [74, 47]. The differences in Strouhal number stay under 6% with respect to the reference data, demonstrating the ability of the method to reproduce correctly the physical phenomenon of vortex shedding.

5.4 Oscillating cylinder in a quiescent fluid

To validate the present method for a moving solid body problem, the case of an oscillating cylinder in a fluid at rest was examined. This case involves the simple harmonic motion in the x-direction of a 2D cylinder with diameter $D = 1\mathrm{m}$ placed at the centre of a square domain as shown in Fig. 5.12, where periodic boundary conditions were applied. The cylinder oscillation leads to the development of boundary layers on the upper and lower sides, which separate from the body generating two counter-rotating vortices. When the cylinder starts to move in the opposite direction the vortex generation stops and the body splits the previously created vortex pair.

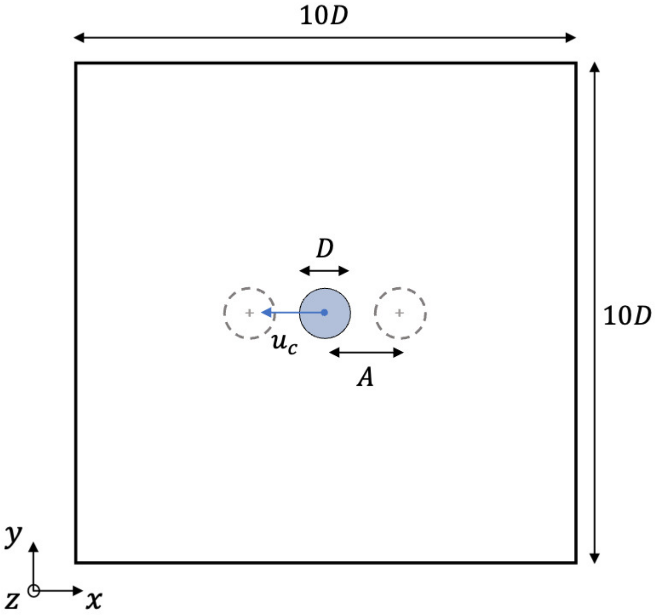


Figure 5.12: Sketch of the domain for the case of the in-line oscillating cylinder in a quiescent fluid.

The periodic motion of the cylinder is described by the following equations:

$$x_\mathrm{c}(t) = -A\sin(2\pi f t)\,, \tag{5.6}$$

$$u_\mathrm{c}(t) = -2\pi f A\cos(2\pi f t)\,, \tag{5.7}$$

$$v_\mathrm{c}(t) = 0\,. \tag{5.8}$$

where x_c, u_c and v_c are the position, the horizontal velocity and the vertical velocity of the cylinder centre, respectively. The frequency of the oscillation is expressed as f and the amplitude of oscillation A. The maximum velocity of the oscillation is defined as $U_\mathrm{max} = 2\pi f A$. The Reynolds number characterising this flow problem is calculated from this velocity value and the cylinder diameter. Another useful non-dimensional quantity is the Keulegan-Carpenter (KC) number, describing the relative importance of the drag forces over the inertia forces for a bluff body in an oscillatory flow, defined as:

$$KC = \frac{U_{\max}}{fD} \,. \tag{5.9}$$

To compare our results with the experimental data we match the two key numbers to the values of $Re = 100$ and $KC = 5$. The maximum velocity is set to $U_{\max} = 1\ \mathrm{m/s}$ with the frequency of oscillation set to $f = 0.2$ Hz and the amplitude to $A = 5/2\pi\ \mathrm{m}$. The simulations ran for $T = 200\ s$ (corresponding to 40 oscillation periods) with a time-step of $\Delta t = 0.0025\ s$. The present method was tested on four unstructured meshes where $D/\Delta x = [10, 25, 50, 100]$, Δx being the grid-spacing near the solid body, and the body-fitted case was conducted on a mesh of $D/\Delta x = 100$. The penalisation time-step ratio was set to $\alpha = 1$.

Our computational results are compared to the experimental data of Dutsch et al. [14] and resolved body-fitted simulations with imposed moving reference frame. As shown in Table 5.4, the mean fluctuation of the in-line force is very well predicted with the present method even for the coarser grid resolution. Its relative error compared to the body-fitted case remains under 9%. Similarly the fundamental frequency of the force's fluctuation is well predicted for all the grid resolutions and its value is $f_{\text{fund}} = 0.2$ Hz, matching the frequency of the body's periodic motion. Fig. 5.13 shows the evolution of the in-line force over one oscillation period compared to the results of Dutsch et al. [14] and the body-fitted simulation. The predicted force is in excellent agreement with the reference data for a grid resolution of $D/\Delta x = 25$ or higher. It is important to note, however, that due to the sharp penalisation mask used in the present method, high frequency (HF) noise is observed on the predicted forces for the case of a moving immersed body, a known problem with IB methods as reported in literature [48, 66]. These HF oscillations originate from the solid momentum source term $\mathscr{P}$, as seen in Fig. 5.14, whose local value at the solid-fluid interface may spike when that interface traverses a grid node and changes from a solid node to a fluid one and vice versa. That is why at the phase angles of 90° and 270° where the solid slows down approaching the maximum displacement, the HF noise disappears. Fig. 5.15 shows the frequency spectra of the in-line force for all simulations carried out. One can see that with the reduction of grid-spacing, the HF noise rapidly decreases in strength and starts at a higher frequency.

In Fig. 5.16 the velocity profiles along the y-direction at three different locations, those being $x = [-0.6D, 0.0D, 0.6D]$, for the phase-angle of 180° are shown compared to the reference data. At that phase, the cylinder passes from its initial position with a positive velocity in the x-direction. The present method reproduces well the velocity field around the solid since both the x-component and y-component of the velocity are in very good agreement with the experimental data. The velocity profile at $x = 0$ shows the smooth transition from the velocity of the fluid to the correct solid velocity inside the body ($-0.5 < y/D < 0.5$) imposed thanks to solid momentum source term $\mathscr{P}$. We can also observe that with a grid-spacing of $D/\Delta x = 25$ or higher, the profiles are identical to those of the body-fitted simulation.

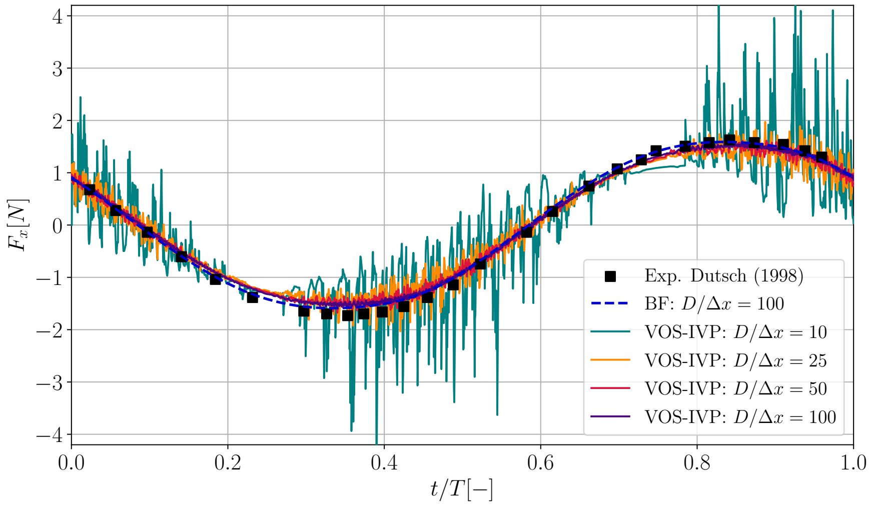


Figure 5.13: In-line force acting on the body during one oscillation period.

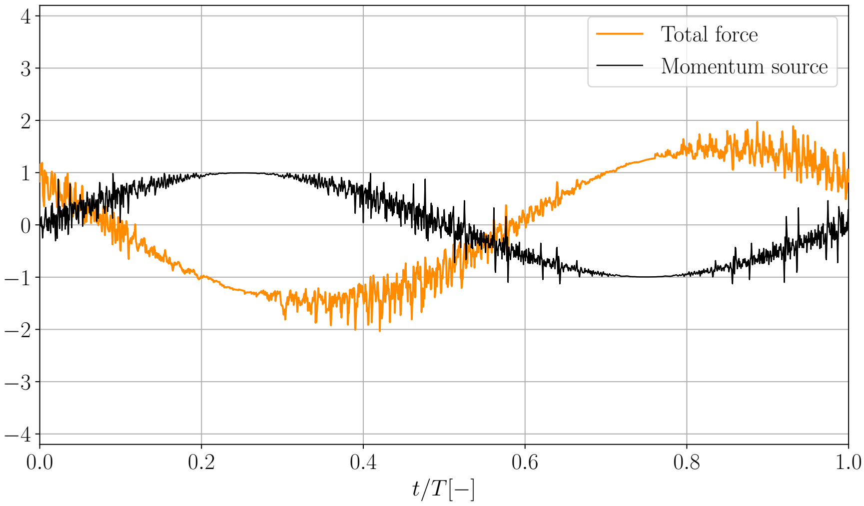


Figure 5.14: In-line force acting on the body (orange) and solid momentum source (black) over one period for $D/\Delta x = 25$.

Table 5.4: Mean fluctuation of the in-line force over 20 periods, relative error of the in-line force compared to the body-fitted case and fundamental frequency of the predicted force for all the simulations.

Case	$F'_x[N]$	$\varepsilon(F'_x)[\%]$	f_{fund}[Hz]
VOS-IVP $D/\Delta x = 10$	1.24	8.77	0.200 ± 0.005
VOS-IVP $D/\Delta x = 25$	1.07	6.14	0.200 ± 0.005
VOS-IVP $D/\Delta x = 50$	1.08	5.26	0.200 ± 0.005
VOS-IVP $D/\Delta x = 100$	1.11	2.63	0.200 ± 0.005
BF $D/\Delta x = 100$	1.14	–	0.200 ± 0.005

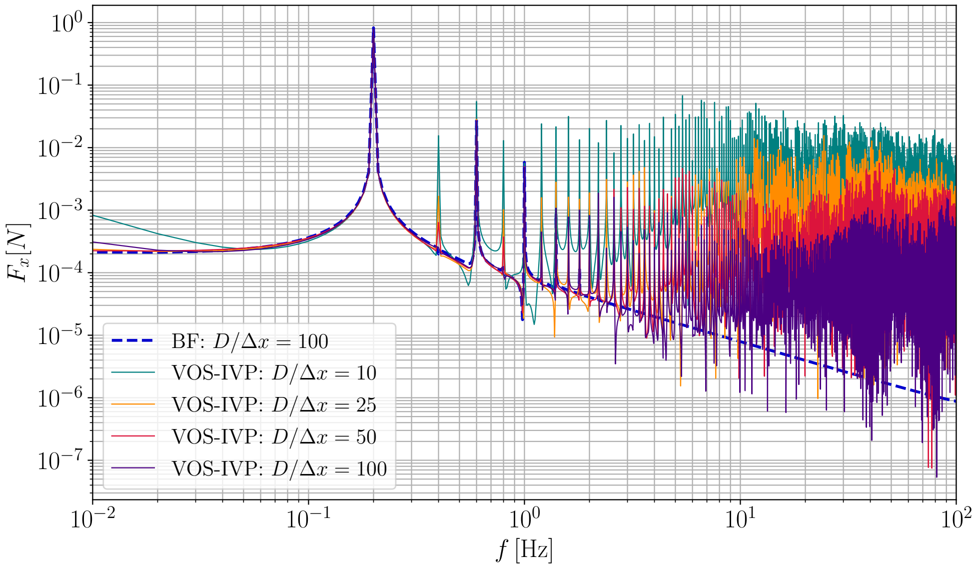


Figure 5.15: Frequency spectra of the in-line force.

5.5 Computational performance

The computational performance of the VOS-IVP method is compared to the performance of the body-fitted method used in the YALES2 solver for the cases of a stationary immersed body and a mobile immersed body as seen in sections 5.3 and 5.4, respectively. The main parameters for comparison are the reduced computational time RCT, representing the computational time spent per iteration per control volume (μs$/iter/node$), and the total computational cost of the simulation $hCPU$ in hours, computed as the wall-clock time of the simulation multiplied by the number of CPU cores used. In both cases the CPUs used are the dual-processor Intel Skylake 8168 (2.7 GHz)

Table 5.5 shows the performances for the 2D case of fluid flow around a stationary cylinder

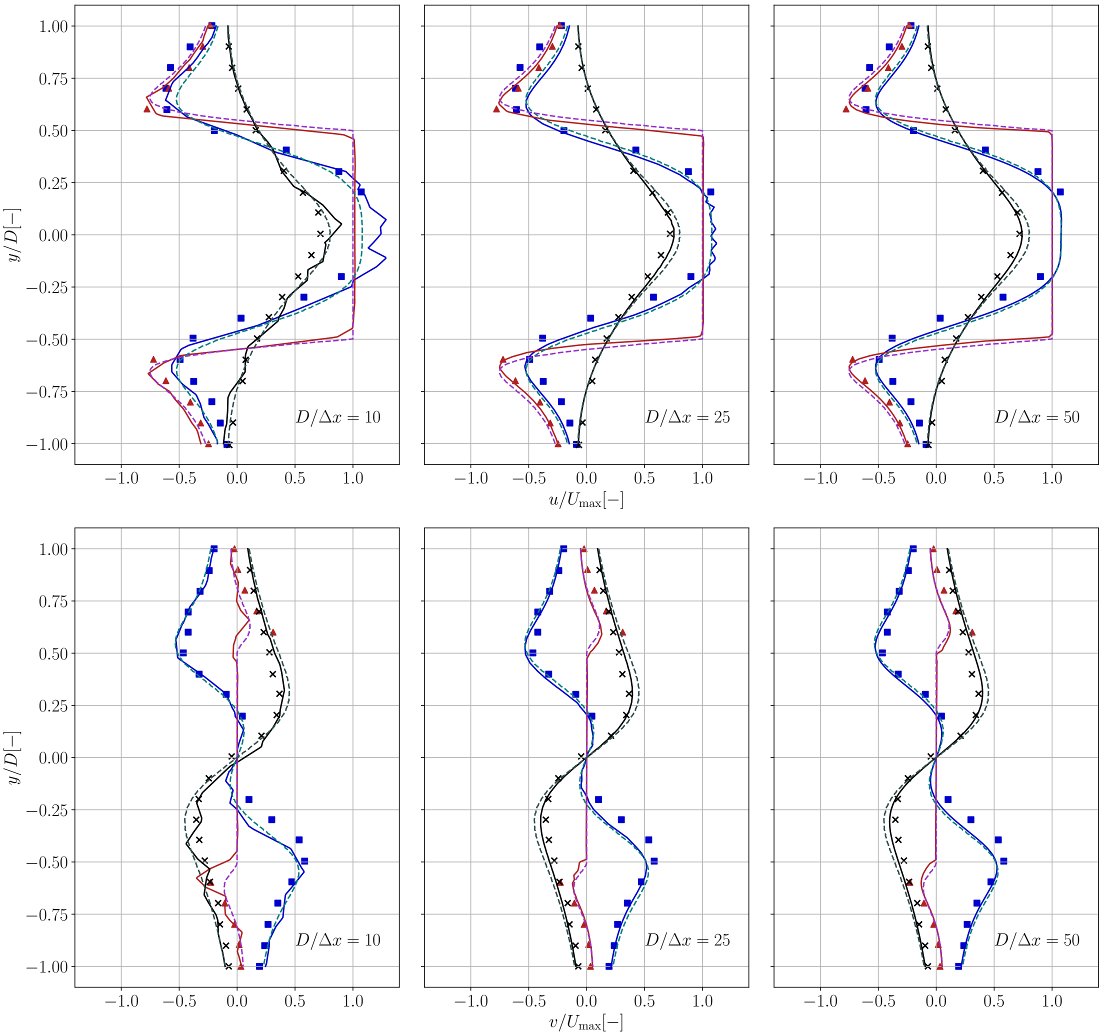


Figure 5.16: Velocity profiles at different locations when the phase angle is 180°. Solid lines represent VOS-IVP results, dashed lines BF results and symbols experimental data [14]. Blue shades correspond to $x = -0.6D$, red shades to $x = 0.0D$ and black shades to $x = 0.6D$.

at $Re = 100$. Keeping the same grid-spacing of $D/\Delta x = 100$ between the two methods, the VOS-IVP case has a higher number of computational cells due to the mesh inside the solid region. In both cases, 20 CPU cores were used. The reduced computational cost per iteration is slightly higher in the VOS-IVP method mainly due to the supplementary calculation of the solid volume fraction field, which costs 0.5 $\mu s/iter/node$. In contrast, the total CPU cost of the method is slightly lower than the one of the body-fitted case. This is attributed to higher time-step values and less iterations needed to simulate 10 seconds of physical time. Indeed, the VOS-IVP needed 93.2×10^3 iterations, while the body-fitted needed 98.4×10^3 iterations.

Table 5.5: Performances for the case of fluid flow at $Re = 100$ around a stationary cylinder: number of computational cells, number of CPU cores used, average time-step in seconds, reduced computational time and total computational cost $hCPU$.

Case	$\mathrm{N}_{\mathrm{elements}}$	$\mathrm{N}_{\mathrm{cores}}$	$\Delta t \times 10^{-4}$	RCT	$hCPU$
VOS-IVP	373×10^3	20	1.072	8.2	39.7
BF	371×10^3	20	1.012	7.9	40.1

Table 5.6 shows the performances for the 2D case of an oscillating cylinder in quiescent fluid at $Re = 100$ and $KC = 5$. Once again, with the same grid-spacing of $D/\Delta x = 100$ between the two methods, the VOS-IVP case has a higher number of computational cells. In both cases, 8 CPU cores were used. The reduced computational cost per iteration is double in the VOS-IVP method and this translates to double the total computational cost compared to the body-fitted method. First of all, there is the extra cost of $4.3\ \mu s/iter/node$ associated with the solid particles for their relocation and the computation of the solid fraction field at each iteration. A more significant cost appears in the solution of the pressure Poisson equation, where in the case of the VOS-IVP with a moving immersed body the Laplacian operator is variable due to the penalty density factor. This makes solving the PPE 3.6 times more expensive than the body-fitted case.

Table 5.6: Performances for the case of oscillating cylinder in fluid at rest: number of computational cells, number of CPU cores used, reduced computational time, reduced computational time for the particle movement and re-localization, reduced computational time for the pressure Poisson equation and total computational cost $hCPU$.

Case	$\mathrm{N}_{\mathrm{elements}}$	$\mathrm{N}_{\mathrm{cores}}$	RCT	RCT(particles)	RCT(PPE)	$hCPU$
VOS-IVP	602×10^3	8	24.5	4.3	11.5	164
BF	584×10^3	8	11.8	–	3.2	76.7

Chapter 6

Application

Contents

6.1 Introduction

In the previous section the VOS-IVP method was shown to be a reliable approach to model an immersed solid object in fluid flow, whether the object is moving or not. The predicted aerodynamic forces matched well with the reference data and the wake downstream of the solid was well reproduced compared to body-fitted simulations. The influence of the penalty parameter, the time-step and the grid size was also examined. In this section, more applications of the VOS-IVP method will be shown for different configurations to further demonstrate the method's ability to predict the influence of a solid object to the fluid dynamics.

6.2 Oscillating cylinder in cross-flow at Re=185

The method is applied in the case of a flow over a 2D cylinder with diameter D with prescribed oscillatory motion in the cross-flow direction. The computation domain is the same as the one described in section 5.3.1 with a grid spacing of $D/\Delta x = 50$ and CFL $= 0.5$. The simulated time was 200 non-dimensional periods. The periodic motion of the cylinder is described by the displacement of the cylinder centre given by:

Table 6.1: Strouhal values for a fixed cylinder case at $Re = 185$.

$Re = 185$	$St[-]$
VOS-IVP	0.203
Numerical results Guilmineau & Queuty [23]	0.195
Numerical results Lu & Dalton [60]	0.195
‘Universal’ Strouhal Williamson [100]	0.193

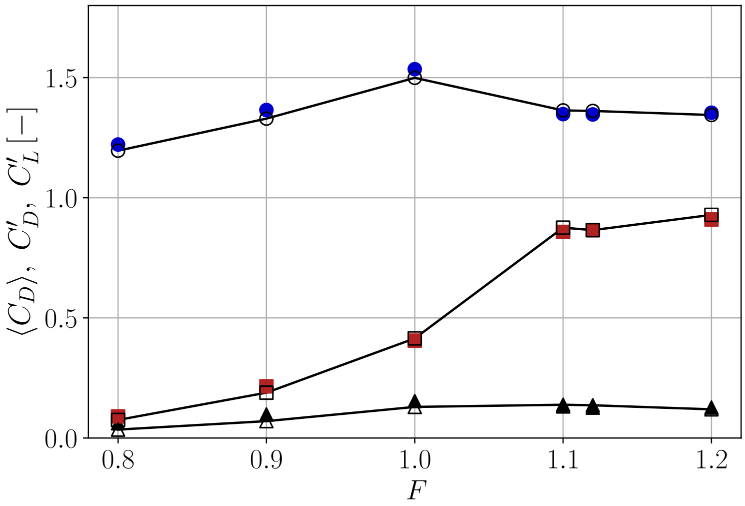


Figure 6.1: Evolution of the force coefficients for $Re = 185$, $A/D = 0.2$ with values of $F = [0.8 - 1.2]$. Symbols: •: $\langle C_D \rangle$, ▲: C'_D, ■: C'_L, ○: $\langle C_{D,\mathrm{ref}} \rangle$, △: $C'_{D,\mathrm{ref}}$, □: $C'_{L,\mathrm{ref}}$.

$$x_c(t) = 0, \tag{6.1}$$

$$y_c(t) = A \sin(2\pi f t)\ , \tag{6.2}$$

where $A = 0.2D$ is the oscillation amplitude and f the oscillation frequency. The Reynolds number based on the cylinder diameter, the free-stream velocity U_∞ and the kinematic viscosity ν, is set to $Re = 185$.

First, simulations on a stationary cylinder were conducted to find the natural vortex shedding frequency f_o at a flow of $Re = 185$. The corresponding Strouhal number measured $St = f_o D/U_\infty = 0.203$, which is higher than the value found in literature, as shown in Table 6.1. This result is consistent with the previous observations that the Strouhal number is slightly overestimated with VOS-IVP if the grid resolution is not high enough. This predicted value was used to define the oscillation frequency of the cylinder.

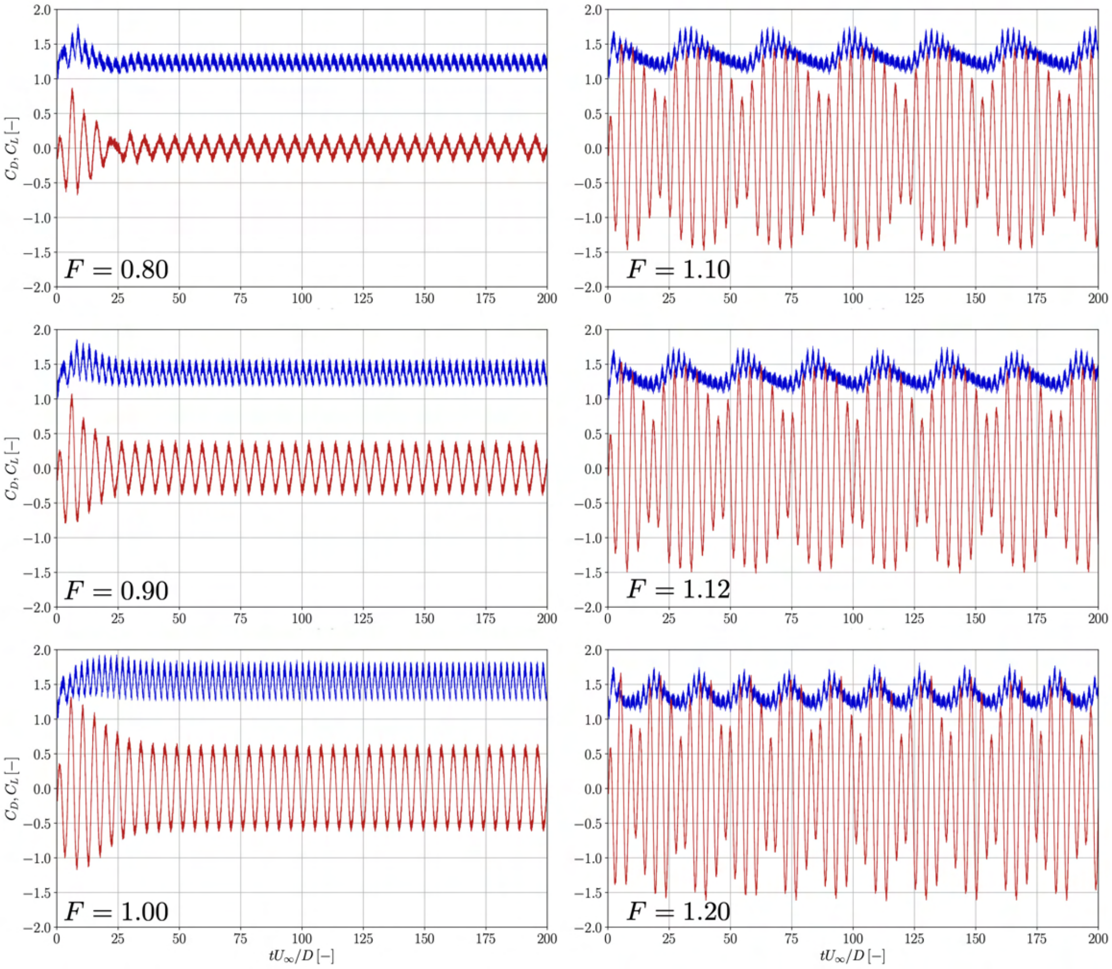


Figure 6.2: Drag (blue) and lift (red) coefficient time-series for $Re = 185$, $A/D = 0.2$ and values of $F = [0.8 - 1.2]$.

The ratio between the oscillation frequency and the natural vortex shedding frequency is expressed as $F = f/f_o$. Simulations were carried out for a range of frequency ratios, $F = [0.8{-}1.2]$, and the results were compared to the reference data from Guilmineau & Queuty [23]. Fig. 6.1 shows the mean and r.m.s. values of the drag coefficient and the r.m.s. values of the lift coefficient depending on the frequency ratio F. A perfect agreement between the predicted values and the reference data can be seen for all quantities. Taking a closer look at the time-series of the aerodynamic coefficients in Fig. 6.2, their forms agree perfectly with the ones reported in literature. After the vortex shedding regime is established, the signals follow a fairly regular behaviour. However, for values of F higher than 1 we can see a beating frequency on the signals. This results from the impact of the cylinder oscillation frequency being higher than the natural vortex shedding frequency of the system.

6.3 Airfoil NACA 0012 at Re=1000 for different angles of attack

Previous cases examined flows involving bluff solid bodies. Two dimensional simulations of flows around a streamline body (the NACA 0012 airfoil) is now examined. The airfoil of chord length c was placed at different angles of attack, $\mathrm{AoA} = [0° - 20°]$, in flows of chord-based Reynolds numbers $Re_c = 10^3$. The size of the computational domain is $20c$ in cross-flow direction, $10c$ upstream of the quarter-length ($c/4$) of airfoil and $30c$ downstream as shown in Fig. 6.3. An unstructured mesh of 2.24×10^5 computational elements is used and the grid resolution near the airfoil is set to $\Delta x = 0.01c$, which translates to 100 computational cells along the chord and 12 cells along the width of the airfoil. This corresponds to a maximum of $\mathrm{y}^+ = 2$. The cases ran for $t^* = tU_\infty/c = 500$ of simulated time using 20 CPU cores.

Fig. 6.4 shows the mean aerodynamic coefficients versus the angle of attack of the airfoil for the VOS-IVP method compared to the results of the body-fitted method. The r.m.s. values are also shown as shaded areas. The results of the methods agree well with each other. Fig. 6.5 and Fig. 6.6 show the instantaneous normalised vorticity $\omega^* = \frac{1}{2}\nabla \times \boldsymbol{u}c/U_\infty$ and the normalised mean velocity $\langle \boldsymbol{u} \rangle/U_\infty$ fields at $t^* = 500$ for three angles of attack, $AoA = [0°, 10°, 20°]$, demonstrating the transition from a steady flow regime to an unsteady one with increasing values of angles of attack. At $AoA = 0°$, the flow is attached to the airfoil from both upper and lower sides, which leads to no deviation of the flow and hence a zero value for the lift. At $4°$, two counter-rotating vortices are observed to form at the trailing edge (T.E.) at the upper side. At $8°$, alternating vortices are observed to start shedding from the airfoil entering the unsteady regime. On the upper side a large clock-wise vortex covers the airfoil surface as the angle of attack increases. As the angle increases from $2°$ to $8°$ the separation point moves from the trailing edge to the half-chord point of the upper side. From $8°$ to $15°$ the separation point further approaches the leading edge (L.E.). For higher angles of attack, a new counter-clockwise vortex is formed near the T.E. at around the 0.75-chord point. This new vortex causes the flow separated at the L.E. of the airfoil to reattach a bit further behind the half-chord point. At $AoA = 20°$, a new counter-clockwise bubble forms as observed in Fig. 6.5 creating a new separation point and the flow reattaches to the upper surface thanks to the T.E. clock-wise vortex. These flow characteristics match perfectly to the ones observed in an extensive study of the same airfoil at the same Reynolds number flow conducted by Kurtulus [49] and Liu et al. [59].

The results for the aerodynamic coefficients are compared to body-fitted reference simulations of similar grid resolution. For the force estimation, one can see an overall good agreement with the body-fitted case of similar grid-spacing ($\Delta x = 0.01c$) near the wall. The lift also compares well with the numerical results of Kurtulus [49] where a grid-spacing of $\Delta x = 0.0015c$ was used. The average relative errors of the forces compared to the body-fitted results are 4.2% and 5.5% for the drag and lift coefficients respectively.

6.4 Turbulent fluid flow over a stationary cylinder

The turbulent flow past a three-dimensional stationary cylinder of diameter D is now examined. The computational domain used for this case is shown in Fig. 6.7. It extends $15D$ upstream of the solid and $30D$ downstream. The top and bottom boundaries of the domain are placed $15D$ from

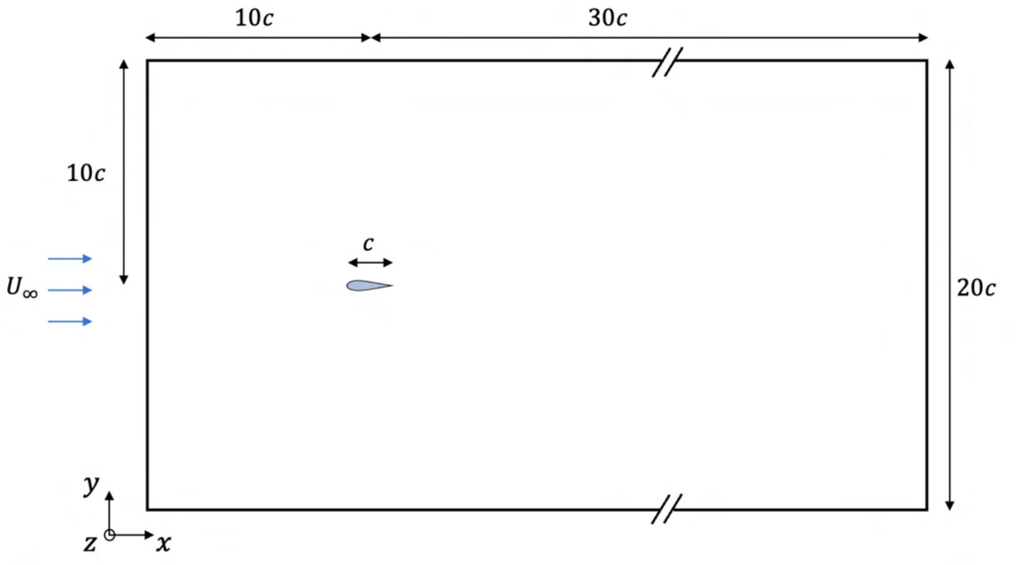


Figure 6.3: Sketch of the domain for the case of the stationary 2D NACA 0012 airfoil.

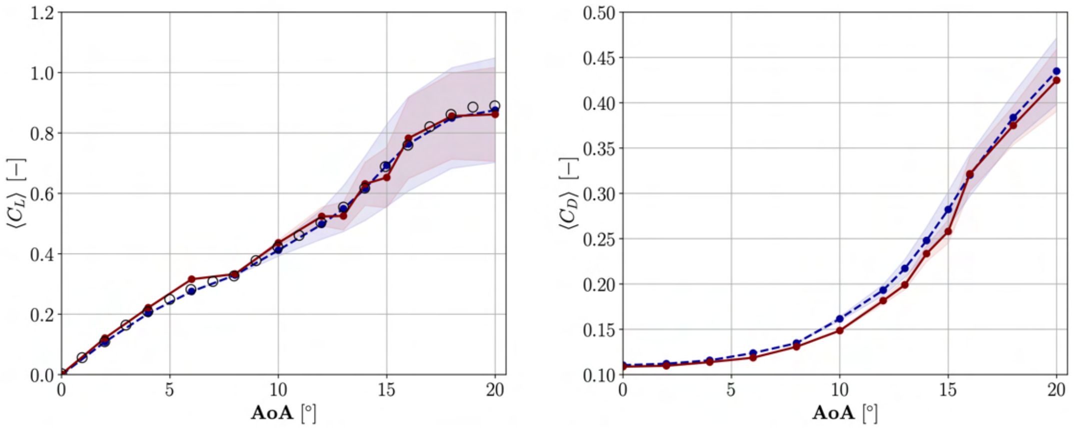


Figure 6.4: Evolution of the aerodynamic coefficients with the angle of attack of the 2D stationary NACA 0012. The red line represents VOS-IVP results, the blue line represents BF results and symbols represent reference data [49].

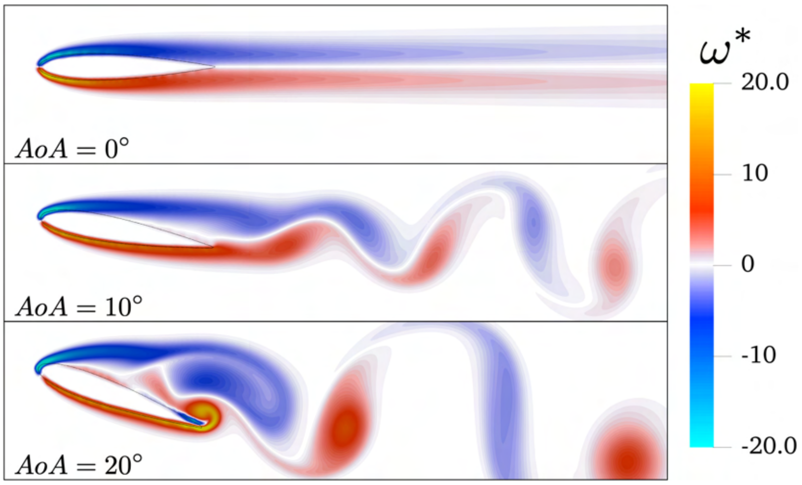


Figure 6.5: Instantaneous non-dimensional vorticity fields for different angles of attack $AoA = [0°, 10°, 20°]$ at $t^* = 500$.

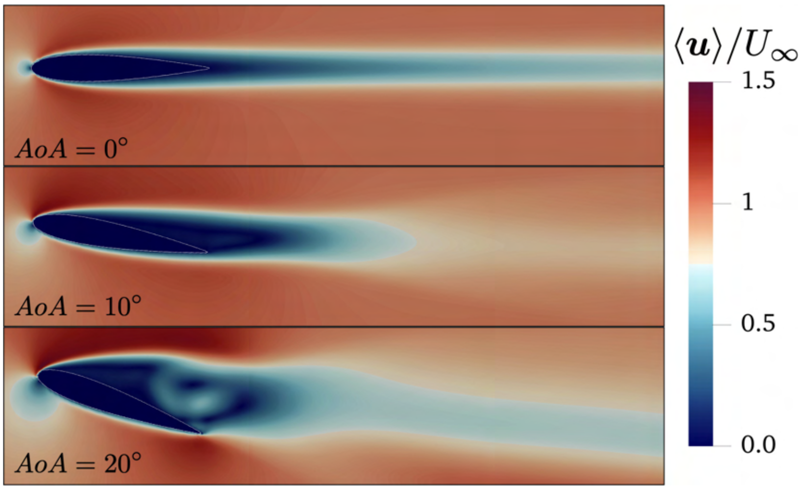


Figure 6.6: Mean non-dimensional velocity fields for different angles of attack $AoA = [0°, 10°, 20°]$ at $t^* = 500$ (statistics were accumulated for 450 non-dimensional periods).

the centre of the solid. The domain extends in the z-direction for a span length of $2D$ and the front and back boundaries along the z-axis are placed at $z = D$ and $z = -D$ respectively, with periodic boundary conditions. The effect of the span-wise length to the numerical study of a flow around a 3D cylinder has been studied by Lei et al. [53]. It was shown that sufficiently accurate results can be achieved with span-wise lengths of two cylinder diameters and above, where the numerical results agree very well with experimental data. For this reason, the span-wise length of two cylinder diameters $L_z = 2D$ was adopted for this case. Three grid resolutions were tested with this method and compared to a reference body-fitted simulation as shown in Table 6.2. This

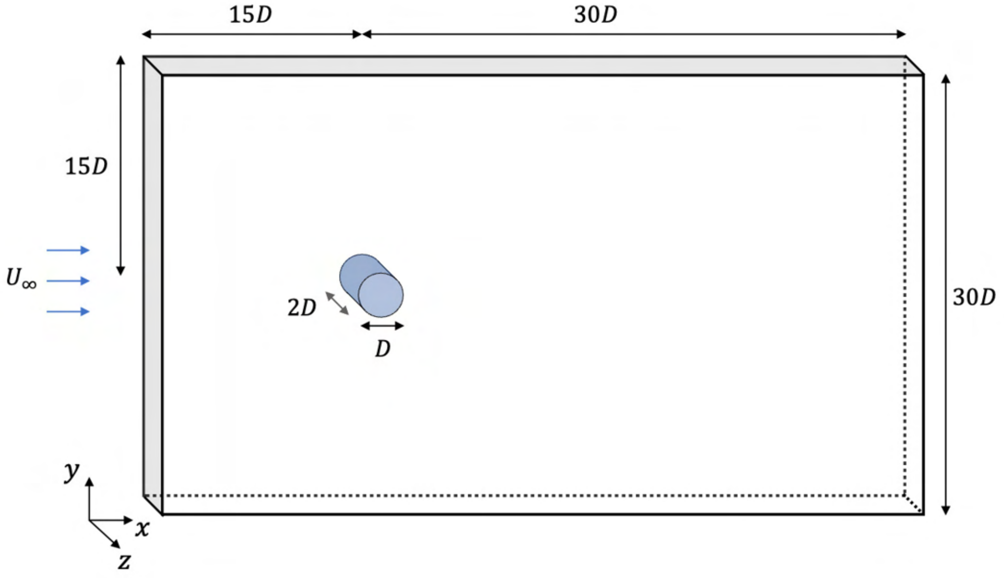


Figure 6.7: Sketch of the domain for the case of the stationary 3D cylinder of span equal to two cylinder diameters.

Table 6.2: Number of elements in mesh, number of CPU cores used, reduced computational cost per iteration RCT $[\mu\mathrm{s/iter/nodes}]$, total computational cost $hCPU$, the maximum dimensionless distance y^+ of an equivalent BF case, mean drag coefficient, mean fluctuation of lift coefficient and Strouhal number.

Case	$\mathrm{N_{elements}}$	$\mathrm{N_{cores}}$	RCT	$khCPU$	y^+	$\langle C_D \rangle$	C'_L	St
VOS-IVP $D/\Delta x = 25$	1.75×10^6	25	45.8	0.12	6.5	1.00	0.173	0.221
VOS-IVP $D/\Delta x = 50$	3.75×10^6	40	37.0	0.37	4.3	0.97	0.152	0.228
VOS-IVP $D/\Delta x = 100$	5.30×10^6	50	39.7	1.05	2.5	0.98	0.153	0.217
BF $D/\Delta x = 100$	4.60×10^6	50	34.5	0.81	2.5	1.01	0.152	0.214

grid-spacing remained sufficiently small so that the first 'fluid' nodes are found in the viscous sublayer, essential for the penalty forcing to give accurate results. The time-step of the simulations was determined by the CFL condition set to 0.9. The penalisation time-step ratio was set to $\alpha = 0.1$. The simulated physical time covered 500 non-dimensional periods $t^* = tU_\infty/D$.

The numerical results presented in Table 6.2 show a very good estimation of forces using the VOS-IVP method compared to the BF case. The mean drag coefficient is well predicted for all grid resolutions, while the mean fluctuation of the lift coefficient is over-estimated only for the coarsest grid. The Strouhal number is the most sensitive to grid resolution. Its relative error compared to the BF case drops from 6% to 1% passing from the medium to the fine mesh resolution.

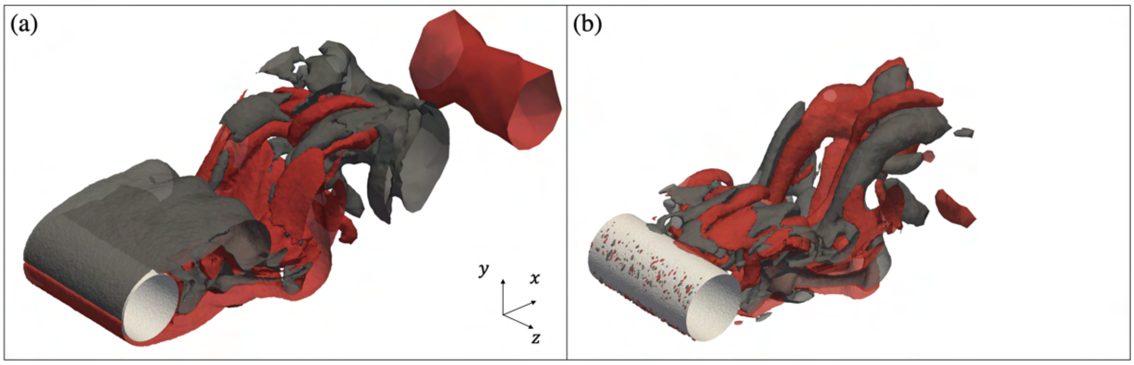


Figure 6.8: Vorticity contours around the cylinder at $Re = 1000$ and with $D/\Delta x = 50$. The left side shows the z-component of the instantaneous non-dimensional vorticity contours at iso-values of $\omega_z^* = \pm 0.35$. The right side shows the x-component contours at iso-values of $\omega_x^* = \pm 0.60$. Contours of positive vorticity values are represented in red and negative values in grey. The cylinder is represented by a contour at the iso-value of $\phi_s = 0.5$ in light grey. Snapshots taken at $t^* = 490$.

Fig. 6.8 shows the contours of the z- and x-components of the non-dimensional vorticity $\omega^* = \frac{1}{2}\nabla \times \boldsymbol{u}D/U_\infty$ downstream of the cylinder. Contrary to the regular alternating patterns of positive and negative vorticity along the z-axis in a 2D configuration, i.e. the out of plane scalar vorticity, we can see that near the cylinder the vorticity along the z-axis changes signs. This is due to the existence of vortical structures in the stream-wise direction demonstrating the three-dimensional unsteady behaviour of the wake near the cylinder. Further downstream, once the vortices become detached, they recover a more regular shape along the z-axis and the x-component of the vorticity subsides.

6.5 Vertical Axis Turbine (VAT) under laminar flow

A more complex case to test the capabilities of the VOS-IVP method is the study of the unsteady incompressible fluid flow through a vertical axis turbine. More specifically, a two-dimensional study is conducted on a three-bladed Darrieus type turbine subjected to laminar flow. The results of the VOS-IVP method, as well as the computational performances, are compared to body-fitted simulations using the ALE method on YALES2 and other numerical studies found in the literature using either rotating sliding body-fitted meshes [18, 84] or direct IBM forcing [75].

6.5.1 Case description

The case setup in this work is the same as the one defined in [18]. The turbine consists of three NACA 0015 airfoils of chord length $c = 1$ m as blades. They are placed at a radius of $R = 2c$ from the rotation centre and the radius connects to the airfoil at the quarter-length ($c/4$) of the chord. The blades are equally spaced from each other in the radial direction (at 120° angles).

Table 6.3: Operating conditions of VAT simulation.

Blade profile	NACA 0015
$\mathrm{N}_{\mathrm{blades}}$	3
Blade chord c $[\mathrm{m}]$	1
Rotor radius R	$2c$
Free-stram velocity U_∞ $[\mathrm{m.s}^{-1}]$	0.5
Chord-based Reynolds Re_c	100
Rotation speed ω $[\mathrm{rad.s}^{-1}]$	0.5
TSR $\lambda = \omega R / U_\infty$	2

The operating conditions are presented in Table 6.3. The free-stream velocity is set to $U_\infty = 0.5\mathrm{m/s}$ and the prescribed rotational speed of the blades is $\omega = 0.5$ rad/s. Hence, the tip-speed ratio, computed as $\lambda = \omega R / U_\infty$, is 2. The chord-based Reynolds number is $Re_c = 100$. The computational domain is shown in Fig. 6.9. It extends $5R$ upstream from the rotor centre, $12.5R$ downstream and $5R$ in each cross-flow direction. The domain is divided into 3 regions of different element sizes. The element size at the blades and the interior of the rotor is $\Delta x = 0.01c$, while in the near wake region $\Delta x = 0.02c$ and for the rest $\Delta x = c$. The mesh contains 0.44×10^6 elements. The simulations are driven by a CFL condition, $\mathrm{CFL} = 0.9$. The penalty time-step ratio is $\alpha = 0.1$. Each blade is represented by 15×10^3 Lagrangian particles, as shown in Fig. 6.10. The particles are obtained from a 2D solid mesh of the NACA 0015 airfoil with a cell-size of $\mathcal{M}_s = 0.004c$. The physical time simulated covers 10 rotor revolutions τ ($T_{\max} = 10\tau$). The body-fitted simulation follows the same setup.

A 2D graphic presenting the rotor movement can be seen in Fig. 6.11. The rotor moves in the counter clock-wise direction around the z-axis at the rotational speed ω. The angle swept by the blades in time t is $\theta = \omega t$. The rotation matrix defining the rotor motion is computed as:

$$\boldsymbol{R} = \begin{pmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} . \tag{6.3}$$

Thus, the position of a blade b at a given instant is computed as:

$$\boldsymbol{X}_b = \boldsymbol{R}\boldsymbol{X}_b^0 , \tag{6.4}$$

where $\boldsymbol{X}_b$ and $\boldsymbol{X}_b^0$ are the current and initial positions, respectively, of the blade.

As discussed in section 4.3.3, the resistive force on a blade is computed as the negative sum of the penalty forces in the volume V_b occupied by the blade:

$$\boldsymbol{F} = -\int_{V_b} \rho \, \mathbf{f}_{\mathrm{IB}} \, \mathrm{d}V , \tag{6.5}$$

where the components of the force follow the x and y directions, i.e. $\boldsymbol{F} = (F_x, F_y)$, and ρ is the fluid density. Due to the assumption of a two-dimensional flow, the turbine is considered infinitely

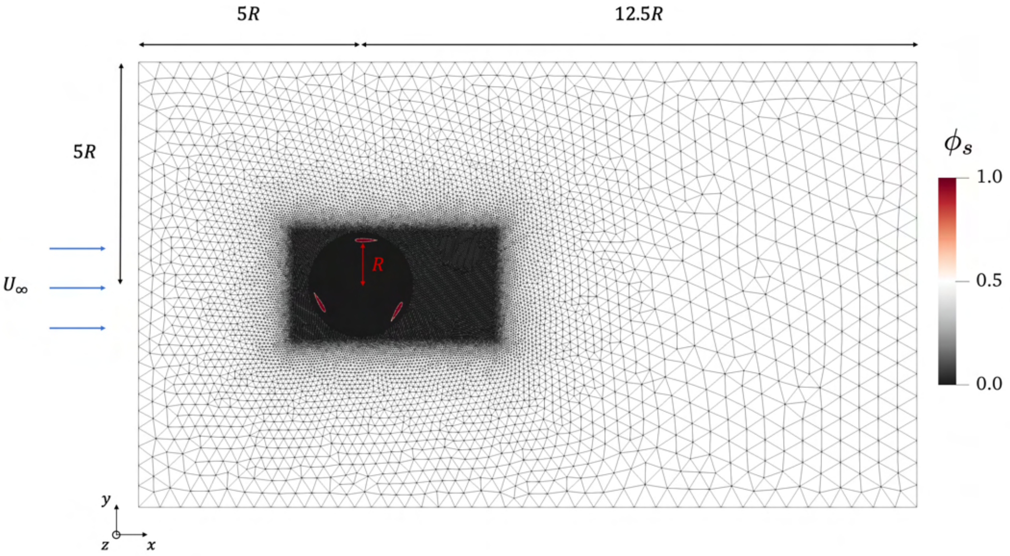


Figure 6.9: Dimensions of the computational domain relative to the rotor radius. The mesh is coloured by the value of the solid volume fraction at the nodes making the blades visible.

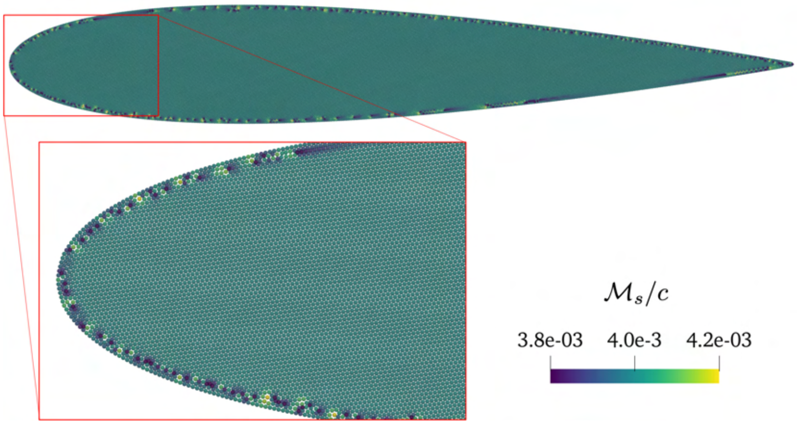


Figure 6.10: Solid particle set of a NACA 0015 airfoil profile. The particles are coloured by their solid mesh size compared to the airfoil chord.

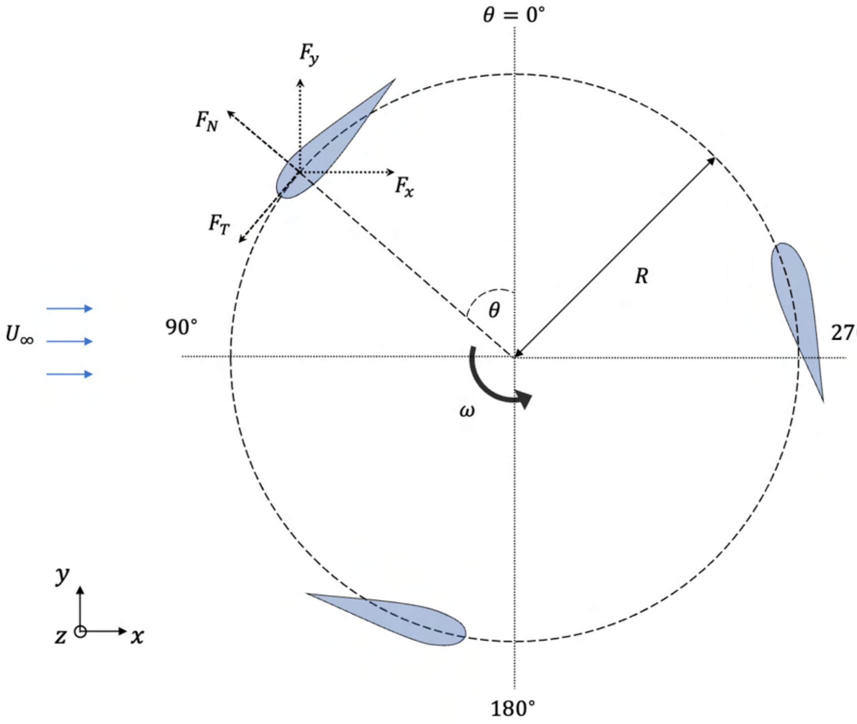


Figure 6.11: A 2D schematic of the turbine geometry and the main parameters for the rotor movement and the computation of the forces.

long and the force in the z direction is ignored.

However, the quantities of interest are the torque, F_T, and the normal forces, F_N, acting on the blades, i.e. the resistive forces in the parallel and perpendicular directions with respect to the blade movement. These can be computed directly from the force estimation in Eq. 6.5 and the position angle θ of the blades. This leads to the expression:

$$\begin{pmatrix} F_T \\ F_N \end{pmatrix} = \begin{pmatrix} -\cos(\theta) & -\sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix} \begin{pmatrix} F_x \\ F_y \end{pmatrix} . \tag{6.6}$$

The torque and normal coefficients can then be calculated as:

$$C_T = \frac{2F_T}{\rho c U_\infty^2} , \quad C_N = \frac{2F_N}{\rho c U_\infty^2} . \tag{6.7}$$

From the torque, one can also compute the power coefficient of the VAT, C_P, as the ratio between the power generated by a turbine blade and the available power in the fluid:

$$C_P = \frac{2\omega R F_T}{\rho A U_\infty^3} \tag{6.8}$$

where $A = 2RH$ is the turbine's projected area, with $H = 1$ due to the 2D assumption.

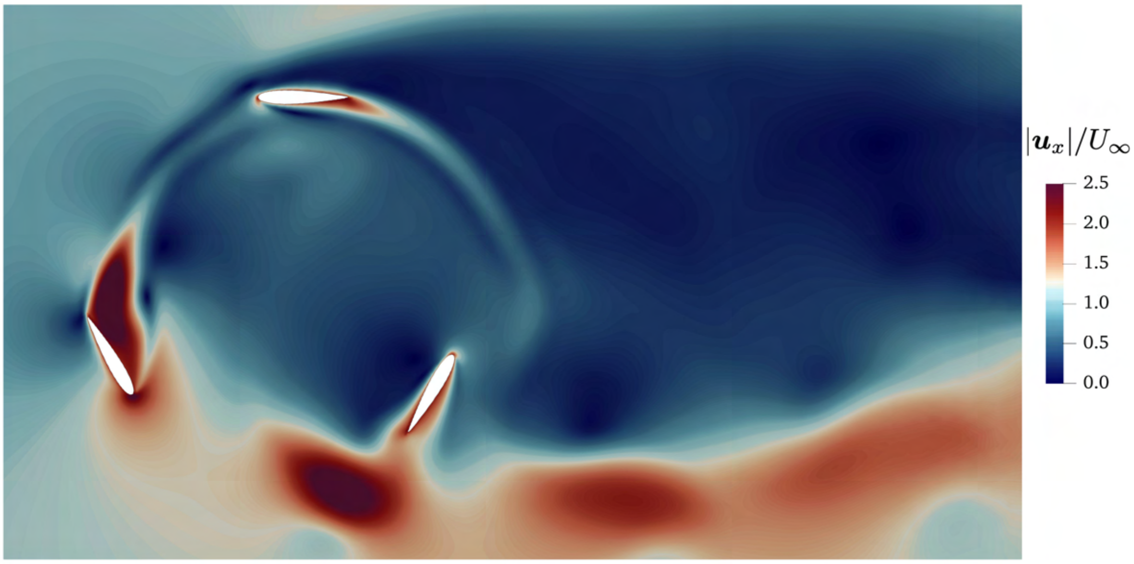


Figure 6.12: Streamwise velocity magnitude at $\theta = 720°$.

6.5.2 Results

For this section, the results presented concern the last two revolutions of the simulation, $(8\tau - 10\tau)$. Fig. 6.12 shows the streamwise velocity magnitude field when the turbine is at $\theta = 720°$, i.e. the end of the tenth revolution. The velocity contours bear very close resemblance to the ones found by Ramirez et al. [84] and Ouro and Stoesser [75] (shown in Fig. 2.15). One can notice classical features of vertical axis turbines such as the high velocity wakes behind the lower part of the turbine and the velocity deficit at the largest part of the wake. Furthermore, one can observe the vastly different near-field aerodynamics around the blades at different angles θ. Upstroke $(270° < \theta < 90°)$, the airfoils encounter much lower velocity values compared to the downstroke region $(90° < \theta < 270°)$. Finally, the velocity field is smooth, even near the moving immersed boundaries.

Figure 6.13 shows the time-history of the coefficients of the torque, the normal force and the power from the blade whose initial position was at $\theta = 0°$, during the last two revolutions. The VOS-IVP coefficient predictions are compared against the ALE results and the results of Ramirez et al. [84]. Concerning the torque coefficient C_T, the VOS-IVP and ALE predictions are in very good agreement. Although, both methods give higher torque values than the ones found in [84], especially during the upstroke movement. The normal force coefficient C_N, appears to follow better the values found in [84]. It also follows the same trend as the ALE approach but presents a continuous underestimation. Finally, the power coefficient C_P is also shown between the two YALES2 methods. The power coefficient value is systematically higher in the VOS-IVP method than the ALE approach.

Usually, the important quantity in VAT studies is the torque, thanks to which the turbine power can be found. The time-average values of the torque and power coefficients are computed as:

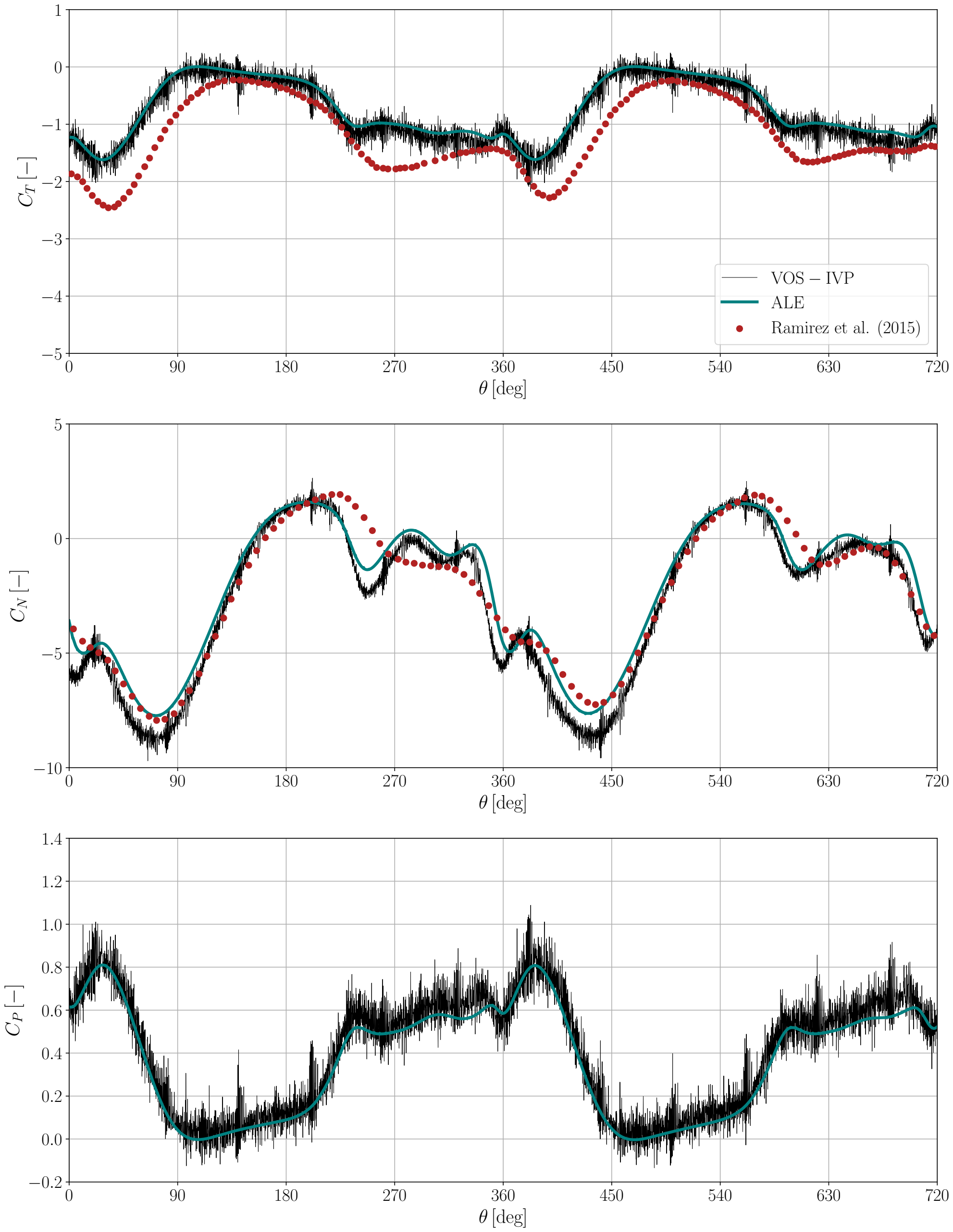


Figure 6.13: Aerodynamic coefficients of the turbine under laminar flow between the present and blade resolved reference data [84]: Torque coefficient C_T (top), normal coefficient C_N (middle) and power coefficient C_P (bottom).

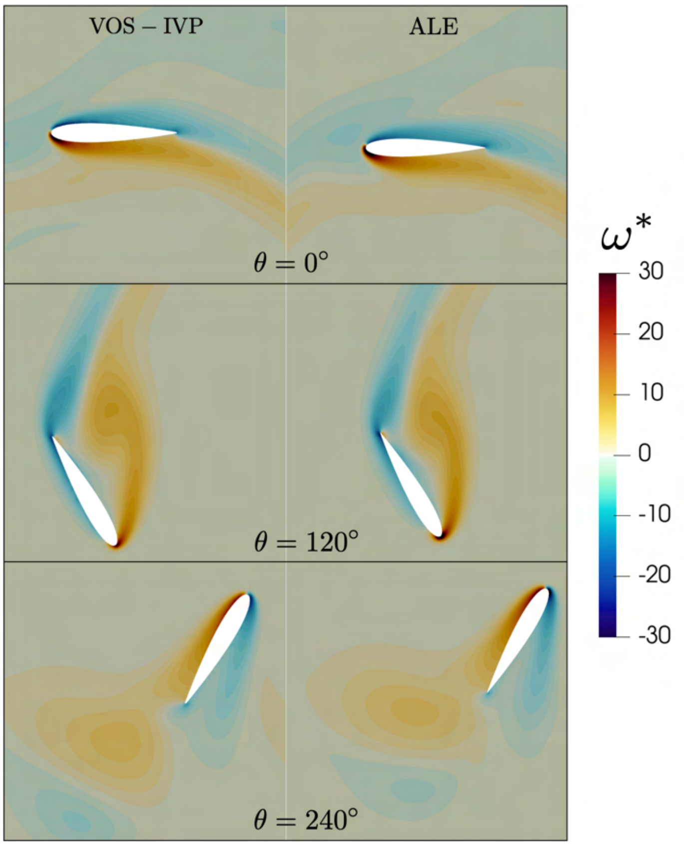


Figure 6.14: Non-dimensional vorticity fields around the blades at positions $\theta = [0°, 120°, 240°]$ for the VOS-IVP case on the left and the ALE case on the right.

$$\langle C_i \rangle = \frac{1}{2\tau} \int_{8\tau}^{10\tau} C_i \, \mathrm{d}t \,, \quad \text{where}\, i = (T, P) \,. \tag{6.9}$$

The average torque coefficient $\langle C_T \rangle$ predicted by the VOS-IVP method is -0.814, while the ALE approach gives -0.776, showing a 5% difference with respect to the body-fitted case. The average power coefficients $\langle C_P \rangle$ take the values 0.407 and 0.368 for the VOS-IVP and ALE cases, respectively. This results in a 10% difference between the two methods.

Overall, the predicted aerodynamic quantities show very good agreement with the body-fitted method, despite the fact that the VOS-IVP time-series suffer from high frequency noise, as this artefact was established in section 5.4.

6.5.3 Computational performance

The numerical details concerning the computational performance of both YALES2 simulations are presented in Table 6.4. In both cases, the simulations are driven by CFL $= 0.9$ and the cell-size at the airfoil surfaces is $\Delta x = 0.01c$, but the time-step is higher in the VOS-IVP method. This results in fewer iterations to cover 10 rotor revolutions compared to the body-fitted case. A possible explanation of this difference is the fact that in the body-fitted method we observe slightly higher velocities at the leading edge of the airfoils, with stronger local vortices as shown in Fig. 6.14, leading to bigger restrictions in the global time-step.

To quantify the computational cost of the simulations we need to multiply the total time of the simulation, WCT, by the number of CPU cores used, $\mathrm{N_{cores}} = 32$. The computational cost in CPU hours per one rotor revolution in the VOS-IVP case is $5.82\,\mathrm{hCPU/revolution}$ and in the ALE case it reaches $8.50\,\mathrm{hCPU/revolution}$. We achieve a speed-up factor of 1.46 with our method compared to the body-fitted simulation. This means that the VOS-IVP method costed 30% less than the ALE method.

Table 6.4: Computational performances of VAT simulations.

	VOS-IVP	ALE
Δt [ms]	4.92	3.48
$\mathrm{N_{iterations}}$	27.1×10^3	32.8×10^3
WCT [s]	6.55×10^3	9.56×10^3
$\mathrm{N_{cores}}$	32	32
hCPU	58.2	85.0
hCPU/revolution	5.82	8.50
speed − up	1.46	1.00

Table 6.5: Reduced computational times of VAT simulations.

RCT [μs/iter/nodes]	VOS-IVP	ALE
Total	35.1	37.5
Update of grid variables	8	3.1
Advection	1.3	15.1
Pressure correction	18	4.3
Post-processing	3.3	0.7
VOS-IVP pre-processing	1.7	–
Relocate solid particles	7.8	–
Mesh adaptation	–	13.9

Table 6.5 shows in more detail the reduced computational times (RCT) of the different processes used in both approaches. The three most costly processes in VOS-IVP are the pressure correction stage, 51% of the cost, the update of the data/variables on the grid nodes, 23%, and the relocation of the solid particles according to the prescribed rotor motion, 22%. For the ALE case the three most costly processes are the velocity advection, 40%, the mesh adaptation, 37%, and the pressure correction, 11%.

The pressure correction step included the process of solving the elliptic pressure poisson differential equation (PPE). In both cases, the Deflated PCG algorithm was used with a convergence criterion of 10^{-7}. However, in the VOS-IVP case, due to the implicit penalty term, we have a variable coefficient in front of the density, known as the penalty density factor, as explained in section 4.3.2. This increases the computational effort needed to solve the PPE, explaining the $18\,\mu\mathrm{s/iter/nodes}$ RCT compared to the $4.3\,\mu\mathrm{s/iter/nodes}$ RCT in the ALE case. The data update on the grid costs more in the VOS-IVP method due to the increased number of variables that need to be stored for the penalty parameters and the computation of the mass and momentum source terms, as shown in section 4.2.3. Comparing the advection step, the ALE seems to cost more due to the treatment of the moving mesh nodes, as described in section 1.2.2. The last differences between the methods are the cost of the particle relocation present in the VOS-IVP case, $7.8\,\mu\mathrm{s/iter/nodes}$, and the mesh adaptation cost in the ALE case, $13.9\,\mu\mathrm{s/iter/nodes}$.

To conclude, in this two-dimensional flow around a moving complex geometry, the costs associated with the VOS-IVP processes (PPE and solid particle relocation) seem to be smaller than the costs of the ALE processes (mesh adaptation and mesh movement). The results show a great speed-up when using our method. This is particularly promising for future three-dimensional studies where the mesh adaptation cost in the ALE method increases exponentially.

Chapter 7

Conclusion and perspectives

Contents

7.1 Conclusion

The aim of this work is to develop a novel Immersed Boundary Method as a tool to model fluid flows around complex solid geometries in arbitrarily large movement or rotation, such as the rotor of a wind turbine. It represents an alternative to the affordable yet low-fidelity Actuator Line Method and the costly but accurate Arbitrary Lagrangian-Eulerian method.

The method developed is called Volume-of-Solid Implicit Volume Penalty (VOS-IVP) IBM. As the name suggests, it couples a Lagrangian Volume-of-Solid (VOS) description of the body and a robust implicit volume penalty (IVP) IBM [2] embedded in the low-Mach number projection method of the YALES2 code of the CORIA laboratory [69]. As explained in section 4 there are three mechanisms at play:

1. **Coupling the VOS approach with an IBM**

   Incorporating the solid volume fraction field from the VOS approach into the Navier-Stokes equations allowing to define a composite velocity field, computed as the mean of the solid and fluid velocities, weighted by their respective volume fractions. Using the composite velocity leads to a new system of equations capable of describing the evolution of both fluid and solid domains at once. These resemble the pure fluid equations but additional mass and momentum source terms appear to represent the solid movement.

2. **Modelling the implicit penalty forcing term**

   Expressing the penalty term in an implicit manner so that its contribution can be split in the prediction step and the correction step of a projection scheme. The contribution in the correction stage serves to impose the solid velocity in the solid region at the same time as the incompressibility constraint is satisfied. This leads to the challenge of solving a modified pressure Poisson equation.

3. **Representing the solid body in a Lagrangian manner**

   Representing the solid object as a set of Lagrangian particles containing an elementary quantity of solid volume. First, the immersed object is displaced by moving the Lagrangian particles according to the prescribed motion and then projecting the solid volume onto the Eulerian grid creates the local solid volume fraction field.

Chapter 5 focused on the validation of the VOS-IVP method through a series of well known academic configurations with both stationary and moving immersed bodies. The numerical results are compared against body-fitted simulations and reference data found in literature. The conservative nature of the solid volume fraction was validated proving the choice of representing the solid volume in a Lagrangian way through the use of mass-less particles an appropriate one. The system of equations is proven to be fully mass conservative in time, independently of the mesh resolution.

From the validation cases involving flow over a stationary cylinder, it is shown that the implicit form of the penalty term improved the accuracy of the velocity imposition at the immersed object compared to the usual penalty method. The predicted forces acting on the solid are in excellent agreement with the reference values (with the relative errors remaining under 10%). The natural vortex shedding frequency is systematically overestimated, by 4% to 8%, but its value tends to fall with decreasing mesh sizes. The near wake behaviour is reproduced very well. Both the velocity and pressure fields were compared to body-fitted simulations showing near perfect agreement between them.

From the validation cases involving flow over a moving cylinder we observe that the VOS momentum sources help to better impose the velocity inside the solid region. The aerodynamic forces are also well predicted but due to the sharp penalty mask used, they suffer from high frequency oscillations. These oscillations, however, subside very fast with decreasing mesh sizes. The fluid flow behaviour near the immersed solid is reproduced very well even for very coarse grids. From this, we can conclude that in the case of moving geometries, good near wake prediction can be achieved with coarser grids but an accurate evaluation of the resistive forces acting on the body needs a finer mesh.

In chapter 6 more applications of the VOS-IVP method were shown, with more complex configurations, to further prove its ability to accurately predict the impact of the immersed solid to the fluid dynamics. Flows around bluff and streamlined bodies and with low to moderate Reynolds number values were examined. Overall, both the predicted aerodynamic loads and the near wake of the flow matched very well literature values and body-fitted results.

Finally, the method was applied on a two-dimensional study on a three-bladed vertical axis turbine (VAT) subjected to laminar flow. The results were compared to body-fitted simulations using the ALE method in YALES2. The velocity and vorticity fields compare very well to reference data found in the literature. The predicted aerodynamic quantities also show very good agreement with the body-fitted method. The high frequency noise plagues the signals of the computed forces, but without having a significant impact to the global forces.

The computational performance of the method was also examined and compared against body-fitted simulations. In the case of stationary immersed solids, and with comparable cell sizes, the VOS-IVP method shows comparable costs with the body-fitted cases. In the study of the VAT, the performance of the VOS-IVP method was compared in detail to the ALE method, which relies heavily on the regular reconstruction of the mesh (mesh adaptations). For the same physical time simulated, our method achieves a speed-up factor of 1.46 compared to the ALE case, which is particularly promising for future three-dimensional studies where the mesh adaptation cost increases exponentially.

To conclude, the VOS-IVP method shows very promising results. The accurate force prediction, the good near wake flow description and the computational performances of the VOS-IVP method make it an attractive option for simulating moving solid geometries. There are also opportunities for further development and improvements that will be discussed in the following section. Future work should focus on more cases with moving immersed bodies and with high Reynolds number flows in order to assess the the method's ability to yield accurate results in an efficient manner for more real-world configurations found in wind energy applications.

7.2 Perspectives

7.2.1 Improvements on the method

Despite the promising results shown in this work there are still aspects to improve:

- The process of relocating the particles is the primary source of computational costs. Each solid is represented by a set of Lagrangian particles and at the start of the simulation each particle is placed on the grid. To achieve this, the coordinates of each particle are checked and the algorithm loops on the element groups allocated to the different processors in order to place the particle inside the correct element. This needs to be repeated at each iteration with the new coordinates of the particles due to the prescribed solid motion. The need of communicating information between the processors for each particle at each iteration makes this process particularly expensive. Hence, future work on optimising this process would greatly benefit the method's computational performances.

- The penalisation method can be further improved by adding wall models. Modelling the flow dynamics near solid surfaces in high Reynolds number flows presents an important challenge in CFD, where in order to fully resolve the boundary layer very small mesh sizes are needed. This makes the simulations much more computationally demanding. Wall models serve to model the flow dynamics near the solid boundary, thus allowing a larger mesh size to be used. These models have also been adapted for the IBM context [8, 57, 17]. In the current version of the VOS-IVP, most of the quantities are represented or calculated in a volume-based form that inherently comes from the VOS method, avoiding any extra computational cost associated to calculating distances. Such calculations would be needed for the computation of the surface normals needed to model the near wall fluid dynamics.

Therefore, the implementation of a wall-model to the penalty term would present an interesting choice where we add the extra cost of distance computations for the trade-off of decreasing the overall cost by using coarser grids. A performance study would be necessary to determine the benefits of the method.

7.2.2 Application of the VOS-IVP in other studies

The method developed in this thesis was originally intended for the simulation of flows around wind turbine rotors, but the nature of the method makes it a useful tool for a wide variety of fluid flows involving one or more solid geometries in motion:

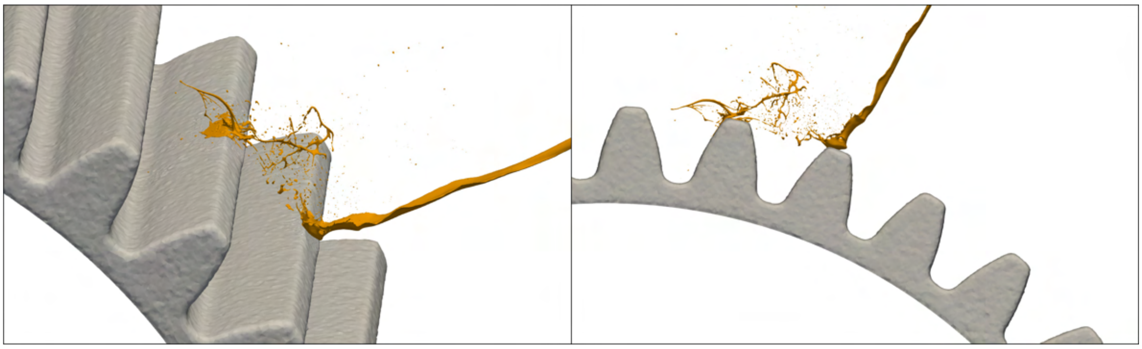


Figure 7.1: VOS-IVP method used in the study of spur gear lubrication. Image provided by Cailler M.

- At the current state of the method, the strategy of replacing the solid geometry with a set of Lagrangian particles has been presented for the case of a rigid body. Since the particles replace the elemental volumes of an initial discretised mesh of the body, each particle holds a portion of the solid volume which is then interpolated onto the Eulerian grid for the computation of the local solid fraction field. In the case of a rigid immersed body, the particles remain at the same distance between each other, and the volume that each particle represents remains unchanged. In the case of a deformable immersed object, the solid particles would move with respect to each other: they would approach in regions of volume contraction and they would distance themselves in regions of volume expansion. Hence, the initial solid mesh element represented by the Lagrangian particle would be deformed and the information of the solid volume held by the particle should be updated to reflect that deformation. In other words, the method could be extended to deformable geometries by the addition of process to compute the new local solid volume of a particle based on the change of its position relative to the neighbouring particles. This approach could easily be implemented for relatively small or regulated deformations of the immersed body, as in the study of aeroelasticity or in the case of some biological flows. This method would prove weak for immersed bodies undergoing arbitrarily large deformations. From a computational point of view, the VOS approach and the extra mass source term in the continuity equation of the method, are readily capable of handling variations in the volume of the solid object.

- During this thesis there were regular exchanges with engineers from the French industrial group Safran. Notably, Mélody Cailler from Safran Tech, Digital Sciences and Technologies Department. There was an active collaboration in the development of the VOS-IVP method and in particular its implementation and validation on complex industrial 3D flow configurations. Most notably, the method was used in predictive simulations of lubrication flows in gearboxes (Fig. 7.1). Such configurations present many numerical challenges due to the multiple complex parts in motion, the dynamic liquid/gas interface, turbulence, etc. The VOS-IVP was used to model the impact of the gear movement to the fluid domain and it was coupled with the Conservative Level-Set method [33] of YALES2 for the accurate description of the liquid/gas topology and Dynamic Mesh Adaptation to keep a refined mesh near the liquid/gas interface. Significant effort was also put into the optimisation of the solid particle relocation algorithm in the case of multiple solid objects in motion. Finally, the use of Lagrangian solid particles was further developed to take automatically into account the mesh criterion (cf. section 4.4.5) in case the refined liquid/gas interface approached the solid surface and the relation $\mathcal{M}_s \leq \mathcal{M}_f$ was no longer valid.

Appendix A

Derivation of the VOS-IVP governing equations

In section 4.2.1 the initial mass and momentum conservation equations were shown for the fluid phase and in section 4.2.2 the final equations of the VOS-IVP method, describing both fluid and solid phases through the use of the composite velocity, were shown. In this section the process of passing from the former set of equations to the latter shall be shown step by step.

The two relations used to derive the final set of equations are:

$$\phi_s + \phi_f = 1\,, \tag{A.1}$$

$$\boldsymbol{u} = \phi_s \boldsymbol{u}_s + \phi_f \boldsymbol{u}_f\,, \tag{A.2}$$

where, ϕ_s and ϕ_f are the solid and fluid volume fractions, respectively, $\boldsymbol{u}_s$ and $\boldsymbol{u}_f$ are the solid and fluid velocities, respectively, and $\boldsymbol{u}$ is the composite velocity.

- **Mass conservation equation**

  The mass conservation equation was derived by simple substitutions of the fluid quantities:

$$\begin{aligned} &\frac{\partial}{\partial t}(\phi_f) + \nabla \cdot (\phi_f \boldsymbol{u}_f) = 0 \\ \Rightarrow\ &\frac{\partial}{\partial t}(1-\phi_s) + \nabla \cdot (\boldsymbol{u} - \phi_s \boldsymbol{u}_s) = 0 \\ &\Rightarrow \nabla \cdot \boldsymbol{u} = \frac{\partial}{\partial t}(\phi_s) + \nabla \cdot (\phi_s \boldsymbol{u}_s)\ . \end{aligned} \tag{A.3}$$

- **Momentum conservation equation**

  The initial fluid momentum equation is:

$$\frac{\partial}{\partial t}\underbrace{(\phi_f \boldsymbol{u}_f)}_{\text{I}} + \nabla \cdot \underbrace{(\phi_f \boldsymbol{u}_f \otimes \boldsymbol{u}_f)}_{\text{II}} = -\frac{1}{\rho}\nabla P + \frac{1}{\rho}\nabla \cdot \underbrace{(\phi_f \boldsymbol{\tau}_f)}_{\text{III}} + \mathbf{f}_{\text{IB}}\,. \tag{A.4}$$

Let us examine term by term how the composite velocity appears. The time derivative term changes as follows:

$$\mathrm{I}: \frac{\partial}{\partial t}(\phi_f \boldsymbol{u}_f) = \frac{\partial \boldsymbol{u}}{\partial t} - \frac{\partial}{\partial t}(\phi_s \boldsymbol{u}_s) \ . \tag{A.5}$$

For the convective term, the following operations take place:

$$\begin{aligned}\mathrm{II}: \phi_f \boldsymbol{u}_f \otimes \boldsymbol{u}_f &= \boldsymbol{u} \otimes \boldsymbol{u}_f - \phi_s \boldsymbol{u}_s \otimes \boldsymbol{u}_f \\ &= \boldsymbol{u} \otimes (\phi_f + \phi_s)\boldsymbol{u}_f - \phi_s \boldsymbol{u}_s \otimes \boldsymbol{u}_f \\ &= \boldsymbol{u} \otimes \phi_f \boldsymbol{u}_f + \boldsymbol{u} \otimes \phi_s \boldsymbol{u}_f - \phi_s \boldsymbol{u}_s \otimes \boldsymbol{u}_f \\ &= \boldsymbol{u} \otimes \boldsymbol{u} - \boldsymbol{u} \otimes \phi_s \boldsymbol{u}_s + \boldsymbol{u} \otimes \phi_s \boldsymbol{u}_f - \phi_s \boldsymbol{u}_s \otimes \boldsymbol{u}_f \\ &= \boldsymbol{u} \otimes \boldsymbol{u} - \phi_s \underbrace{(\boldsymbol{u} \otimes \boldsymbol{u}_s - \boldsymbol{u} \otimes \boldsymbol{u}_f + \boldsymbol{u}_s \otimes \boldsymbol{u}_f)}_{\mathrm{IV}} \ .\end{aligned} \tag{A.6}$$

Developing further the term IV we get:

$$\begin{aligned}&\mathrm{IV}: \boldsymbol{u} \otimes \boldsymbol{u}_s - \boldsymbol{u} \otimes \boldsymbol{u}_f + \boldsymbol{u}_s \otimes \boldsymbol{u}_f \\ &\quad = \phi_f \boldsymbol{u}_f \otimes \boldsymbol{u}_s + \phi_s \boldsymbol{u}_s \otimes \boldsymbol{u}_s - \boldsymbol{u} \otimes \boldsymbol{u}_f + \boldsymbol{u}_s \otimes \boldsymbol{u}_f \\ &\quad = \phi_f \boldsymbol{u}_f \otimes \boldsymbol{u}_s + \phi_s \boldsymbol{u}_s \otimes \boldsymbol{u}_s - \phi_f \boldsymbol{u}_f \otimes \boldsymbol{u}_f - \phi_s \boldsymbol{u}_s \otimes \boldsymbol{u}_f + \boldsymbol{u}_s \otimes \boldsymbol{u}_f \\ &\quad = \phi_f \boldsymbol{u}_f \otimes \boldsymbol{u}_s + \phi_s \boldsymbol{u}_s \otimes \boldsymbol{u}_s - \phi_f \boldsymbol{u}_f \otimes \boldsymbol{u}_f - (1-\phi_f) \boldsymbol{u}_s \otimes \boldsymbol{u}_f + \boldsymbol{u}_s \otimes \boldsymbol{u}_f \\ &\quad = 2\phi_f \boldsymbol{u}_f \otimes \boldsymbol{u}_s + (1-\phi_f) \boldsymbol{u}_s \otimes \boldsymbol{u}_s - \phi_f \boldsymbol{u}_f \otimes \boldsymbol{u}_f \\ &\quad = \boldsymbol{u}_s \otimes \boldsymbol{u}_s + \phi_f \left(2\boldsymbol{u}_f \otimes \boldsymbol{u}_s - \boldsymbol{u}_s \otimes \boldsymbol{u}_s - \boldsymbol{u}_f \otimes \boldsymbol{u}_f\right) \\ &\quad = \boldsymbol{u}_s \otimes \boldsymbol{u}_s - \phi_f \left(\boldsymbol{u}_s - \boldsymbol{u}_f\right) \otimes \left(\boldsymbol{u}_s - \boldsymbol{u}_f\right) \ .\end{aligned} \tag{A.7}$$

Injecting this formula back to term II gives:

$$\mathrm{II}: \phi_f \boldsymbol{u}_f \otimes \boldsymbol{u}_f = \boldsymbol{u} \otimes \boldsymbol{u} - \phi_s \boldsymbol{u}_s \otimes \boldsymbol{u}_s + \phi_s \phi_f \left(\boldsymbol{u}_s - \boldsymbol{u}_f\right) \otimes \left(\boldsymbol{u}_s - \boldsymbol{u}_f\right) \ . \tag{A.8}$$

The last term can be neglected for a sharp representation of the solid volume fraction. The product $\phi_s \phi_f (\boldsymbol{u}_s - \boldsymbol{u}_f)$ can be neglected since $\phi_s \phi_f = 0$ away from the solid/fluid interface and $\boldsymbol{u}_f \approx \boldsymbol{u}_s \approx \boldsymbol{u}$ at the interface. Hence, the convective term in the momentum equation can be expressed as:

$$\nabla \cdot (\phi_f \boldsymbol{u}_f \otimes \boldsymbol{u}_f) = \nabla \cdot (\boldsymbol{u} \otimes \boldsymbol{u}) - \nabla \cdot (\phi_s \boldsymbol{u}_s \otimes \boldsymbol{u}_s) \, . \tag{A.9}$$

For the diffusive term we first define a composite dynamic viscosity in the same manner as the composite velocity:

$$\mu = \phi_f \mu_f + \phi_s \mu_s \ , \tag{A.10}$$

but we set the solid viscosity to zero to avoid shear flows inside the solid domain and so that only the pressure term would counteract the forcing terms of the VOS-IVP method. So, the composite viscosity μ used in the solver is equal to $\phi_f \mu_f$.

Term III can be re-arranged in the following manner:

$$\begin{aligned} \text{III} : \phi_f \boldsymbol{\tau}_f &= \mu \left(\nabla(\phi_f \boldsymbol{u}_f) + \nabla(\phi_f \boldsymbol{u}_f)^T \right) \\ &= \mu \left(\nabla(\boldsymbol{u} - \phi_s \boldsymbol{u}_s) + \nabla(\boldsymbol{u} - \phi_s \boldsymbol{u}_s)^T \right) \ , \end{aligned} \tag{A.11}$$

where since the viscosity is a multiple of the fluid volume fraction, the cross terms including the product $\mu\phi_s$ can be neglected. Thus, a new viscous stress tensor $\boldsymbol{\tau}$ can be computed from the composite fields, where:

$$\text{III} : \phi_f \boldsymbol{\tau}_f = \mu \left(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T \right) = \boldsymbol{\tau} \, . \tag{A.12}$$

Finally, replacing all the terms containing the fluid velocity with those containing the composite velocity gives the final form of the momentum conservation equation describing both phases at once:

$$\frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot (\boldsymbol{u} \otimes \boldsymbol{u}) = -\frac{1}{\rho} \nabla P + \frac{1}{\rho} \nabla \cdot \boldsymbol{\tau} + \frac{\partial}{\partial t} (\phi_s \boldsymbol{u}_s) + \nabla \cdot (\phi_s \boldsymbol{u}_s \otimes \boldsymbol{u}_s) + \mathbf{f}_{\text{IB}} \, . \tag{A.13}$$

List of Figures

List of Tables

References

[1] R. Abgrall, H. Beaugendre, and C. Dobrzynski. “An immersed boundary method using unstructured anisotropic mesh adaptation combined with level-sets and penalization techniques”. In: *Journal of Computational Physics* 257 (PA 2014), pp. 83–101.

[2] Philippe Angot, Charles Henri Bruneau, and Pierre Fabrie. “A penalization method to take into account obstacles in incompressible viscous flows”. In: *Numerische Mathematik* 81 (4 1999), pp. 497–520.

[3] Elias Balaras. “Modeling complex boundaries using an external force field on fixed Cartesian grids in large-eddy simulations”. In: *Computers and Fluids* 33 (3 2004), pp. 375–404.

[4] Pierre Benard. “Analyse et amélioration d’une chambre de combustion centimétrique par simulations aux grandes échelles”. 2015.

[5] R. Byron Bird, E. Warren Stewart, and Edwin N. Lightfoot. *Transport Phenomena*. 2nd. Wiley, 2009. ISBN: 978-0470115398.

[6] Joseph Boussinesq. “Théorie de l’écoulement tourbillonnant”. In: *Mém. prés. Acad. Sci.* 23 (1877), pp. 23–46.

[7] Marco Raciti Castelli, Alessandro Englaro, and Ernesto Benini. “The Darrieus wind turbine: Proposal for a new performance prediction model based on CFD”. In: *Energy* 36 (8 2011), pp. 4919–4934.

[8] Stuart Chester, Charles Meneveau, and Marc B. Parlange. “Modeling turbulent flow over fractal trees with renormalized numerical simulation”. In: *Journal of Computational Physics* 225 (1 2007), pp. 427–448.

[9] C Chevalier and F Pellegrini. “PT-Scotch: A tool for efficient parallel graph ordering”. In: *Parallel Computing* 34 (6-8 2008), pp. 318–331.

[10] Alexandre Joel Chorin. “Numerical solution of the Navier-Stokes equations”. In: *Mathematics of Computation* 22 (104 1968), pp. 745–745.

[11] E. Constant et al. “An immersed boundary method in OpenFOAM: Verification and validation”. In: *Computers and Fluids* 157 (2017), pp. 55–72.

[12] Jean Donea et al. *Arbitrary Lagrangian-Eulerian Methods*. 2004.

[13] Haibo Dong et al. “An Integrated Analysis of a Dragonfly in Free Flight”. In: *40th AIAA fluid dynamics conference and exhibit, AIAA2010-4390* ().

[14] H. Dutsch et al. “Low-Reynolds-number flow around an oscillating circular cylinder at low Keulegan–Carpenter numbers”. In: *Journal of Fluid Mechanics* 360 (1998), pp. 249–271.

[15] Thomas Fabbri. “Development of a high fidelity fluid-structure interaction solver: towards flexible foils simulation”. Université Grenoble Alpes, 2022.

[16] E. A. Fadlun et al. “Combined Immersed-Boundary Finite-Difference Methods for Three-Dimensional Complex Flow Simulations”. In: *Journal of Computational Physics* 161 (1 2000), pp. 35–60.

[17] Ahmad Fakhari. “A new wall model for large eddy simulation of separated flows”. In: *Fluids* 4 (4 2019).

[18] Esteban Ferrer and Richard H.J. Willden. “A high order Discontinuous Galerkin - Fourier incompressible 3D Navier-Stokes solver with rotating sliding meshes”. In: *Journal of Computational Physics* 231 (21 2012), pp. 7037–7056.

[19] Massimo Germano et al. “A dynamic subgrid-scale eddy viscosity model”. In: *Physics of Fluids A : Fluid Dynamics* 3 (7 1991), pp. 1760–1765.

[20] H. Glauert. *Airplane Propellers*. 1935.

[21] D. Goldstein, R. Handler, and L. Sirovich. *Modeling a No-Slip Flow Boundary with an External Force Field*. 1993.

[22] Alejandro Gronskis and Guillermo Artana. “A simple and efficient direct forcing immersed boundary method combined with a high order compact scheme for simulating flows with moving rigid boundaries”. In: *Computers and Fluids* (2016).

[23] E Guilmineau and P Queutey. “A numerical simulation of vortex shedding from an oscillating circular cylinder”. In: *Journal of Fluids and Structures* (6 2002), pp. 773–794.

[24] Francis H. Harlow and J. Eddie Welch. “Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface”. In: *Physics of Fluids* 8 (12 1965), pp. 2182–2189.

[25] Seyed Hossein Hezaveh et al. “Simulation and wake analysis of a single vertical axis wind turbine”. In: *Wind Energy* 20 (4 2017), pp. 713–730.

[26] C.W Hirt and B.D Nichols. “Volume of fluid (VOF) method for the dynamics of free boundaries”. In: *Journal of Computational Physics* 39 (1 1981), pp. 201–225.

[27] Pierre Horgue, Marc Prat, and Michel Quintard. “A penalization technique applied to the ”Volume-Of-Fluid” method: Wettability condition on immersed boundaries”. In: *Computers and Fluids* (2014).

[28] Félix Houtin-Mongrolle et al. “Actuator line method applied to grid turbulence generation for large-Eddy simulations”. In: *Journal of Turbulence* (2020).

[29] Wei Xi Huang and Fang Bao Tian. “Recent trends and progresses in the immersed boundary method”. In: *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* 0 (0 2019), pp. 1–20.

[30] Wei-Xi Huang and Hyung Jin Sung. “Improvement of mass source/sink for an immersed boundary method”. In: *International Journal for Numerical Methods in Fluids* 53 (11 2007), pp. 1659–1671.

[31] IPCC. *Climate Change 2021: The Physical Science Basis. Summary for Policymakers*. 2021.

[32] S. Jafari, N. Chokani, and R. S. Abhari. “An immersed boundary method for simulation of wind flow over complex terrain”. In: *Journal of Solar Energy Engineering, Transactions of the ASME* 134 (1 2012), pp. 1–12.

[33] Romain Janodet et al. “A massively parallel accurate conservative level set algorithm for simulating turbulent atomization on adaptive unstructured grids”. In: *Journal of Computational Physics* 458 (2022), p. 111075.

[34] Aymen Jendoubi. “Immersed boundary method for flow simulation around horizontal wind turbine”. In: *Journal of Physics: Conference Series* 1276 (1 2019), pp. 0-13.

[35] Seung Jeon and Haecheon Choi. “Boundary treatment for the unsteady surface velocity in an immersed boundary method”. In: *Journal of Mechanical Science and Technology* 23 (9 2009), pp. 2502–2506.

[36] George Karypis and Vipin Kumar. *METIS * Unstructured Graph Partitioning and Sparse Matrix Ordering System Version 2.0*. 1995.

[37] Nikolaos D. Katopodes. *Level Set Method*. 2019.

[38] Nicholas K.R. Kevlahan and Jean Michel Ghidaglia. “Computation of turbulent flow past an array of cylinders using a spectral method with Brinkman penalization”. In: *European Journal of Mechanics, B/Fluids* 20 (3 2001), pp. 333–350.

[39] Dokyun Kim and Haecheon Choi. “Immersed boundary method for flow around an arbitrarily moving body”. In: *Journal of Computational Physics* 212 (2 2006), pp. 662–680.

[40] J Kim and P Moin. *Application of a Fractional-Step Method to Incompressible Navier-Stokes Equations*. 1985, pp. 308–323.

[41] Jungwoo Kim, Dongjoo Kim, and Haecheon Choi. “An Immersed-Boundary Finite-Volume Method for Simulations of Flow in Complex Geometries”. In: *Journal of Computational Physics* 171 (1 2001), pp. 132–150.

[42] Woojin Kim and Haecheon Choi. “Immersed boundary methods for fluid-structure interaction: A review”. In: *International Journal of Heat and Fluid Flow* 75 (July 2018 2019), pp. 301–309.

[43] Sergiu Klainerman and Andrew Majda. “Compressible and incompressible fluids”. In: *Communications on Pure and Applied Mathematics* 35 (5 1982), pp. 629–651.

[44] A N Kolmogorov. *The Local Structure of Turbulence in Incompressible Viscous Fluid for Very Large Reynolds Numbers*. 1991, pp. 9–13.

[45] Matthias Kraushaar. “Low-mach number approaches to large-eddy simulation of turbulent flows in aero-engines”. In: (2012).

[46] Arthur G Kravchenko, Parviz Moin, and Karim Shariff. *B-Spline Method and Zonal Grids forSimulations of Complex Turbulent Flows*. 1999, pp. 757–789.

[47] Bhaskar Kumar and Sanjay Mittal. “Effect of blockage on critical parameters for row past a circular cylinder”. In: *International Journal for Numerical Methods in Fluids* 50 (2006), pp. 987–1001.

[48] Manish Kumar and Somnath Roy. “A sharp interface immersed boundary method for moving geometries with mass conservation and smooth pressure variation”. In: *Computers and Fluids* (2016).

[49] Dilek Funda Kurtulus. *On the Unsteady Behavior of the Flow Around NACA 0012 Airfoil with Steady External Conditions at Re=1000*. 2015.

[50] Ming Chih Lai and Charles S. Peskin. “An immersed boundary method with formal second-order accuracy and reduced numerical viscosity”. In: *Journal of Computational Physics* 160 (2 2000), pp. 705–719.

[51] Jinmo Lee and Donghyun You. “An implicit ghost-cell immersed boundary method for simulations of moving body problems with control of spurious force oscillations”. In: *Journal of Computational Physics* (2013).

[52] Jongho Lee et al. “Sources of spurious force oscillations from an immersed boundary method for moving-body problems”. In: *Journal of Computational Physics* 230 (7 2011), pp. 2677–2695.

[53] C Lei, L Cheng, and K Kavanagh. “Spanwise length effects on three-dimensional modelling of flow over a circular cylinder”. In: *Computer Methods in Applied Mechanics and Engineering* 190 (22-23 2001), pp. 2909–2923.

[54] D K Lilly. “A proposed modification of the Germano subgrid-scale closure method”. In: *Phys. Fluids A* 4 (3 1992).

[55] Cheng Liu and Changhong Hu. “An actuator line - immersed boundary method for simulation of multiple tidal turbines”. In: *Renewable Energy* 136 (2019), pp. 473–490.

[56] Cheng Liu and Changhong Hu. “An efficient immersed boundary treatment for complex moving object”. In: *Journal of Computational Physics* 274 (2014), pp. 654–680.

[57] Luoqin Liu and Richard J.A.M. Stevens. “Wall modeled immersed boundary method for high Reynolds number flow over complex terrain”. In: *Computers and Fluids* 208 (2020).

[58] Rex Kuan-Shuo Liu, Khai-Ching Ng, and Tony Wen-Hann Sheu. “A volume of solid implicit forcing immersed boundary method for solving incompressible Navier-Stokes equations in complex domain”. In: *Computers and Fluids* 218 (2021), p. 104856.

[59] Yan Liu et al. “Numerical bifurcation analysis of static stall of airfoil and dynamic stall under unsteady perturbation”. In: *Communications in Nonlinear Science and Numerical Simulation* 17 (8 2012), pp. 3427–3434.

[60] X. Y. Lu and C. Dalton. “Calculation of the timing of vortex formation from an oscillating cylinder”. In: *Journal of Fluids and Structures* 10 (1996), pp. 527–541.

[61] Mathias Malandain, Nicolas Maheu, and Vincent Moureau. "Optimization of the deflated Conjugate Gradient algorithm for the solving of elliptic equations on massively parallel machines". In: *Journal of Computational Physics* 238 (2013), pp. 32–47.

[62] L a Martinez Tossas and S Leonardi. "Wind Turbine Modeling for Computational Fluid Dynamics December 2010 — December 2012 Wind Turbine Modeling for Computational Fluid Dynamics". In: (July 2013), p. 57.

[63] Cindy Merlin. "Simulation numérique de la combustion turbulente : Méthode de frontières immergées pour les écoulements compressibles , application à la combustion en aval d ' une cavité". In: *PhD Thesis, Institut National des Sciences Appliquées de Rouen* (2011).

[64] S Mirjalili, S S Jain A N, and D M S Dodd. *Interface-capturing methods for two-phase flows: An overview and recent developments*. 2017.

[65] R. Mittal et al. "A versatile sharp interface immersed boundary method for incompressible flows with complex boundaries". In: *Journal of Computational Physics* (2008).

[66] Rajat Mittal and Gianluca Iaccarino. "Immersed boundary methods". In: *Annual Review of Fluid Mechanics* 37 (2005), pp. 239–261.

[67] J. Mohd-Yusof. "Combined immersed boundary/B-spline methods for simulation of flow in complex geometries". In: *Center for Turbulence Research Annual Research Briefs* (1997), pp. 317–328.

[68] Antoine Morente, Jérôme Laviéville, and Dominique Legendre. "A penalization method for the simulation of bubbly flows". In: *Journal of Computational Physics* 374 (2018), pp. 563–590.

[69] Vincent Moureau, Pascale Domingo, and Luc Vervisch. *Design of a massively parallel CFD code for complex geometries*. 2011.

[70] Frank Muldoon and Sumanta Acharya. "Mass conservation in the immersed boundary method". In: *Proceedings of 2005 ASME Fluids Engineering Division Summer Meeting, FEDSM2005* 2005 (2005), pp. 821–829.

[71] K. C. Ng. "A collocated finite volume embedding method for simulation of flow past stationary and moving body". In: *Computers and Fluids* 38 (2 2009), pp. 347–357.

[72] R. A. Nicolaides. "Deflation of Conjugate Gradients With Applications To Boundary Value Problems." In: *SIAM Journal on Numerical Analysis* 24 (2 1987), pp. 355–365.

[73] F Nicoud and F Ducros. *Subgrid-Scale Stress Modelling Based on the Square of the Velocity Gradient Tensor*. 1999, pp. 183–200.

[74] C. Norberg. "An experimental investigation of the flow around a circular cylinder: Influence of aspect ratio". In: *Journal of Fluid Mechanics* 258 (1994), pp. 287–316.

[75] Pablo Ouro and Thorsten Stoesser. "An immersed boundary-based large-eddy simulation approach to predict the performance of vertical axis tidal turbines". In: *Computers and Fluids* 152 (2017), pp. 74–87.

[76] Jeongyoung Park, Kiyoung Kwon, and Haecheon Choi. *Numerical Solutions of Flow Past a Circular Cylinder at Reynolds Numbers up to 160*. 1998, pp. 1200–1205.

[77] Yulia Peet et al. “Actuator line aerodynamics model with spectral elements”. In: American Institute of Aeronautics and Astronautics Inc., 2013. ISBN: 9781624101816.

[78] Charles S. Peskin. “Flow patterns around heart valves: A numerical method”. In: *Journal of Computational Physics* 10 (2 1972), pp. 252–271.

[79] Stephen B Pope. *Turbulent Flows*. Cambridge university press, 2000.

[80] S Popinet et al. “A front-tracking algorithm for accurate representation of surface tension A front-tracking algorithm for accurate representation of surface ten-sion A front tracking algorithm for the accurate representation of surface tension”. In: *International Journal for Numerical Methods in Fluids* 30 (6 1999), pp. 775–793.

[81] Fernando Porté-Agel, Majid Bastankhah, and Sina Shamsoddin. *Wind-Turbine and Wind-Farm Flows: A Review*. Vol. 174. Springer Netherlands, 2019, pp. 1–59.

[82] Antonio Posa et al. “Wake structure of a single vertical axis wind turbine”. In: *International Journal of Heat and Fluid Flow* 61 (2016), pp. 75–84.

[83] Lixia Qu et al. “Quantitative numerical analysis of flow past a circular cylinder at Reynolds number between 50 and 200”. In: *Journal of Fluids and Structures* 39 (28 2013), pp. 347–370.

[84] Luis Ramírez et al. “New high-resolution-preserving sliding mesh techniques for higher-order finite volume schemes”. In: *Computers and Fluids* 118 (2015), pp. 114–130.

[85] LF Richardson. “Weather prediction by numerical process”. In: (1922).

[86] Pierre Sagaut. *Large Eddy Simulation for Incompressible Flows. An Introduction*. Vol. 12. 2001, pp. 1745–1746. ISBN: 9783540263449.

[87] J. A. Sethian and Peter Smereka. “Level set methods for fluid interfaces”. In: *Annual Review of Fluid Mechanics* 35 (2003), pp. 341–372.

[88] Wen Zhong Shen, Jian Hui Zhang, and Jens Nørkær Sørensen. “The actuator surface model: A new navier-stokes based model for rotor computations”. In: *Journal of Solar Energy Engineering, Transactions of the ASME* 131 (1 2009), pp. 0110021–0110029.

[89] Joseph Smagorinsky. “General circulation experiments with the primitive equations I. the basic experiment”. In: *Monthly weather review* 91 (3 1963), pp. 99–164.

[90] Jens Nørkær Sørensen and Wen Zhong Shen. “Numerical modeling of wind turbine wakes”. In: *Journal of Fluids Engineering, Transactions of the ASME* 124 (2 2002), pp. 393–399.

[91] Fotis Sotiropoulos and Xiaolei Yang. “Immersed boundary methods for simulating fluid-structure interaction”. In: *Progress in Aerospace Sciences* 65 (2014), pp. 1–21.

[92] M. Specklin and Y. Delauré. “A sharp immersed boundary method based on penalization and its application to moving boundaries and turbulent rotating flows”. In: *European Journal of Mechanics, B/Fluids* 70 (2018), pp. 130–147.

[93] Kosuke Suzuki and Takaji Inamuro. “Effect of internal mass in the simulation of a moving body by the immersed boundary method”. In: *Computers and Fluids* 49 (1 2011), pp. 173–187.

[94] Yu Heng Tseng and Joel H. Ferziger. “A ghost-cell immersed boundary method for flow in complex geometry”. In: *Journal of Computational Physics* 192 (2 2003), pp. 593–623.

[95] M D De Tullio et al. *Immersed Boundary Technique for Large-Eddy-Simulation*. von Karman Institute for Fluid Dynamics, 2022.

[96] H. S. Udaykumar et al. “A sharp interface Cartesian grid method for simulating flows with complex moving boundaries”. In: *Journal of Computational Physics* 174 (1 2001), pp. 345–380.

[97] Markus Uhlmann. “An immersed boundary method with direct forcing for the simulation of particulate flows”. In: *Journal of Computational Physics* 209 (2 2005), pp. 448–476.

[98] S. O. Unverdi and Gretar Tryggvason. “A front-tracking method for viscous, incompressible, multi- fluid flows”. In: *Journal of Computational Physics* 100 (1992), pp. 25–37.

[99] Henk A Van Der Vorst. *Parallel Iterative Solution Methods for Linear Systems arising from Discretized PDE’s*.

[100] C. H.K. Williamson. “Defining a universal and continuous Strouhal-Reynolds number relationship for the laminar vortex shedding of a circular cylinder”. In: *Physics of Fluids* 31 (10 1988), pp. 2742–2744.

[101] Jianming Yang and Elias Balaras. “An embedded-boundary formulation for large-eddy simulation of turbulent flows interacting with moving boundaries”. In: *Journal of Computational Physics* 215 (1 2006), pp. 12–40.

[102] Jianming Yang and Frederick Stern. “Sharp interface immersed-boundary/level-set method for wave-body interactions”. In: *Journal of Computational Physics* (2009).

Méthodes des Frontières Immergées Avancées pour la modélisation des sillages des éoliennes

Mots-clés : *Méthodes des Frontières Immergées, Pénalisation volumique implicite, Volume-of-Solid, Simulations Grandes Échelles, Mécanique des fluides numérique, Volumes finis, Éoliennes*

Résumé : Plusieurs applications en ingénierie impliquent des écoulements des fluides autour des solides mobiles. Un tel exemple se trouve sur les unités de production d’énergie renouvelable comme les éoliennes. La mécanique des fluides numérique aide à simuler les écoulements fluides. Les méthodes des frontières immergées (IBM) sont un outil attractif pour traiter les grands déplacements des corps solides complexes sur des maillages fixes.

Pendant cette thèse, une nouvelle méthode des frontières immergées a été développée en couplant une approche Volume-of-Solid (VOS) Lagrangienne pour la description du corps immergé et une méthode de pénalisation volumique implicite avec le schéma de projection pour des écoulements à faible nombre de Mach du code YALES2 du laboratoire CORIA, dans le contexte des Simulations des Grandes Échelles (LES). Cette méthode s’appelle « Volume-of-Solid Implicit Volume Penalty method » (VOS-IVP) et se comporte de trois axes : (i) L’utilisation du champ de la fraction volumique locale du solide dans les équations Navier-Stokes, permettant la définition du champ de la vitesse composite décrivant le domaine fluide et le domaine solide en même temps ; (ii) La formulation implicite du terme de pénalisation, ce qui permet l’imposition de la vitesse solide dans la région solide au même instant avec l’imposition de la contrainte d’incompressibilité ; (iii) La représentation du corps solide comme une somme des particules Lagrangiennes contenant le volume du solide locale pour une imposition robuste du mouvement solide.

La précision de la méthode a été évaluée sur plusieurs cas académiques avec des corps à la fois bluff et streamlined et avec des valeurs de nombre de Reynolds faibles à modérées. A partir des cas de validation impliquant un écoulement autour un solide stationnaire, il est montré que la forme implicite du terme de pénalisation améliore la précision de l'imposition de la vitesse à l'objet immergé par rapport à la méthode de pénalisation usuelle. Les forces prédites agissant sur le solide sont en excellent accord avec les valeurs trouvées dans la littérature et avec les simulations body-fitted de référence, où les erreurs relatives restent inférieures à 10%. D'après les cas de validation impliquant un écoulement autour des corps en mouvement, les forces aérodynamiques sont bien prédites mais, en raison du masque de pénalisation aigu utilisé, elles souffrent d'oscillations à haute fréquence. Ces oscillations, cependant, s'atténuent très rapidement avec des tailles de mailles décroissantes. Dans tous les cas, le comportement du sillage proche est très bien reproduit. Il est prouvé que le système d'équations est entièrement conservatif en masse dans le temps, indépendamment de la résolution du maillage. Les performances de calcul font également de cette méthode une option attrayante. Les coûts sont identiques à ceux des cas body-fitted de solides stationnaires. Dans la simulation d'une turbine à axe vertical 2D soumise à un écoulement laminaire, un facteur d'accélération de 1,46 a été observé par rapport au temps de calcul requis par la méthode body-fitted.

En conclusion, la méthode VOS-IVP montre des résultats très prometteurs avec des possibilités d'améliorations supplémentaires, telles que l'intégration de modèles de parois au terme de pénalisation et l'optimisation du processus de relocalisation des solides. Les travaux futurs se concentreront sur l'utilisation de la méthode pour des configurations plus réelles.

Advanced Immersed Boundary Methods for Wind Turbine Wake Modelling

Keywords: *Immersed Boundary Methods, Implicit Volume Penalty, Volume-of-Solid, Large-Eddy Simulations, Computational Fluid Dynamics, Finite Volume, Wind Turbines*

Abstract: Many engineering applications involve fluid flows around moving complex solid objects. An example can be found in renewable energy production units such as wind turbines. Computational Fluid Dynamics (CFD) help us simulate numerically such flows. Immersed Boundary methods (IBM) have proven to be an attractive tool for handling arbitrarily large displacements of complex solid bodies on a fixed grid.

In this thesis, a novel Immersed Boundary method has been developed coupling a Lagrangian Volume-of-Solid (VOS) description of the immersed body and a robust implicit volume penalty IBM embedded in the low-Mach number projection method of the YALES2 code of the CORIA laboratory, in the framework of high-fidelity unsteady Large-Eddy Simulations (LES). This method is referred to as Volume-of-Solid Implicit Volume Penalty method (VOS-IVP) and can be broken down to three components: (i) Incorporating the solid volume fraction field into the Navier-Stokes equations, allowing to define a composite velocity field describing both fluid and solid domains at once; (ii) Expressing the penalty term in an implicit manner, imposing the solid velocity inside the solid region at the same time as the incompressibility constraint is satisfied; (iii) Representing the solid object as a set of Lagrangian particles carrying the local solid volume for a robust imposition of the solid movement.

The accuracy of the method has been assessed on several academic cases with both bluff and streamlined bodies and with low to moderate Reynolds number values. From the validation cases involving flow over a stationary solid, it is shown that the implicit form of the penalty term improved the accuracy of the velocity imposition at the immersed object compared to the usual penalty method. The predicted forces acting on the solid are in excellent agreement with values found in literature and with body-fitted reference simulations, where the relative errors remain under 10%. From the validation cases involving flow over moving bodies, the aerodynamic forces are well predicted but due to the sharp penalty mask used, they suffer from high frequency oscillations. These oscillations, however, subside very fast with decreasing mesh sizes. In all cases, the near wake behaviour is reproduced very well. The system of equations is proven to be fully mass conservative in time, independently of the mesh resolution.

The computational performances also make the method an attractive option. The costs are identical with body-fitted cases of stationary solids. In the simulation of a 2D vertical axis turbine subjected to laminar flow, a speed-up factor of 1.46 was observed compared to the computational time required by the body-fitted method.

To conclude, the VOS-IVP method shows very promising results with opportunities for further improvement, such as the integration of wall-models to the penalty term and the optimisation of the solid relocation process. Future work will focus on the method's use to more real-world configurations.