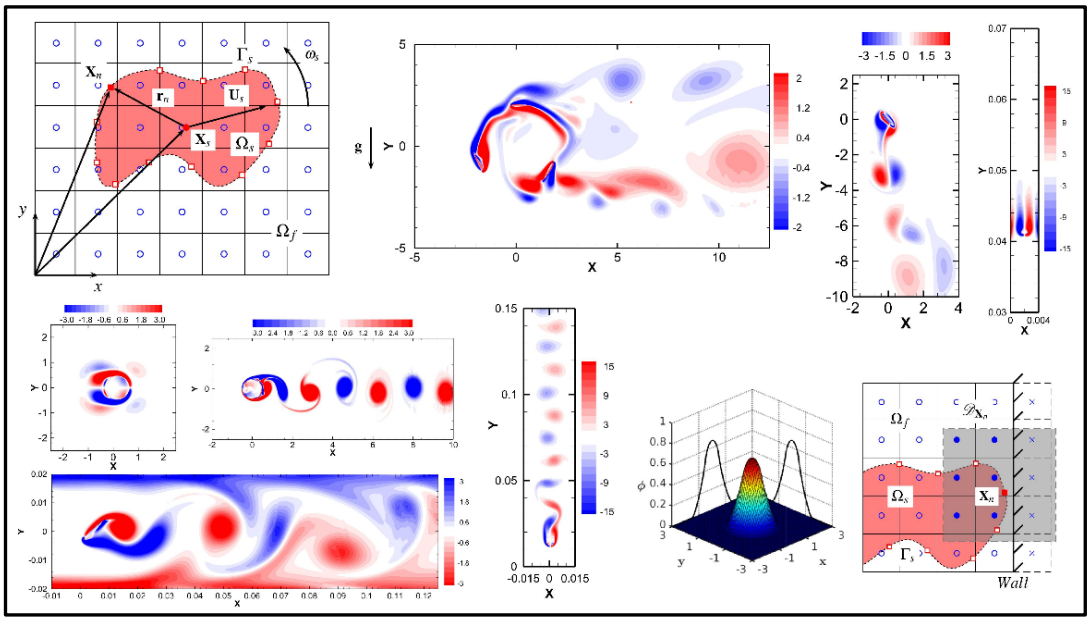
Immersed boundary method with improved implicit direct-forcing for fluid-structure interaction problems

Elias Farah

Par **Elias FARAH**

Méthode de frontière immergée avec forçage direct implicite amélioré pour les problèmes d'interaction fluide-structure

Thèse présentée en cotutelle pour l'obtention du grade de Docteur de l'UTC



Soutenue le 8 avril 2025
Spécialité : Mécanique Numérique : Unité de recherche en Mécanique - Laboratoire Roberval (FRE UTC - CNRS 2012)
D2870

THÈSE DE DOCTORAT

Pour obtenir le grade de Docteur de

L'UNIVERSITÉ DE TECHNOLOGIE DE COMPIEGNE (UTC)

&

CRANFIELD UNIVERSITY (CU)

Méthode de frontière immergée avec forçage direct implicite amélioré pour les problèmes d'interaction fluide-structure

Par **Elias FARAH**

Spécialité : Mécanique Numérique

Laboratoire Roberval

Unité de recherche en mécanique, énergie et électricité

Soutenue le 8 avril 2025 devant le jury composé de :

M. A. RASSINEUX, Professeur des universités, Université de Technologie de Compiègne	(Président)
M. A. OUAHSINE, Professeur des universités, Université de Technologie de Compiègne	(Directeur de thèse)
M. P. VERDIN, Maître de Conférences, Cranfield University, Royaume-Uni	(Directeur de thèse)
M. S. GUILLOU, Professeur des universités, ESIX Normandie	(Rapporteur)
Mme É. LACAZEDIEU, Professeure des universités, INSA Hauts-de-France	(Rapporteure)
M. P. DRUAULT, Maître de Conférences à Sorbonne Université, CNRS	(Examinateur)
M. L. HUANG, Maître de Conférences, Cranfield University, Royaume-Uni	(Examinateur)
M. T. KANIT, Enseignant-chercheur, Faculté des Sciences et Technologies de l'Université de Lille	(Examinateur)

Invités :

M. A. IBRAHIMBEGOVIC

M. S. CAI

M. B. KAOUI

M. W. OTTEN

UNIVERSITÉ DE TECHNOLOGIE DE COMPIEGNE (UTC)

&

CRANFIELD UNIVERSITY (CU)

IMMERSED BOUNDARY METHOD WITH IMPROVED IMPLICIT DIRECT-FORCING FOR FLUID–STRUCTURE INTERACTION PROBLEMS

By **Elias FARAH**

Spécialité : Mécanique Numérique

Roberval Laboratory
Research Unit in Mechanics, Energy, and Electricity - Computational Mechanics Team (UTC)
&
Energy and Sustainability Department (CU)

Thesis defended on April 8, 2025

M. A. RASSINEUX, Professeur des universités, Université de Technologie de Compiègne	(Président)
M. A. OUAHSINE, Professeur des universités, Université de Technologie de Compiègne	(Directeur de thèse)
M. P. VERDIN, Maître de Conférences, Cranfield University, Royaume-Uni	(Directeur de thèse)
M. S. GUILLOU, Professeur des universités, ESIX Normandie	(Rapporteur)
Mme É. LACAZEDIEU, Professeure des universités, INSA Hauts-de-France	(Rapporteure)
M. P. DRUAULT, Maître de Conférences à Sorbonne Université, CNRS	(Examinateur)
M. L. HUANG, Maître de Conférences, Cranfield University, Royaume-Uni	(Examinateur)
M. T. KANIT, Enseignant-chercheur, Faculté des Sciences et Technologies de l'Université de Lille	(Examinateur)

Invités :

M. A. IBRAHIMBEGOVIC
M. S. CAI
M. B. KAOUI
M. W. OTTEN

Abstract

An improved implicit direct-forcing immersed boundary method (DF-IBM) is proposed for simulating interactions between incompressible fluid flows and complex rigid structures undergoing arbitrary free motion, commonly referred to as fluid–rigid body interaction problems. The proposed approach harnesses the pressure implicit with splitting of operators (PISO) algorithm to efficiently handle the dual constraints of the fluid–solid system in a segregated manner. Consequently, the divergence-free condition is maintained throughout the Eulerian domain, while the kinematic no-slip velocity boundary condition is exactly enforced on the immersed boundary, also termed as the fluid–structure interface. A new pressure Poisson equation (PPE) is derived, incorporating the boundary force directly where the no-slip condition is satisfied. This approach avoids altering the coefficient matrix of the PPE, which could otherwise introduce convergence issues, enabling the use of fast iterative PPE solvers without modifications. The improvement involves integrating Lagrangian weight methods, having better reciprocity over the IBM-related linear operators, within the implicit formulation. An additional force initialization scheme is introduced to accelerate the convergence of the no-slip boundary condition, thereby improving the algorithm's performance.

The Navier-Stokes equations are coupled with the rigid body dynamics, described by the Newton-Euler equations, within the improved DF-IBM framework. Both explicit and implicit coupling algorithms are developed to address weakly and strongly coupled fluid–rigid body interaction problems, respectively, under a partitioned approach. Stability and convergence issues, particularly stemming from critical solid–fluid density ratios and/or the rigid body approximation of the internal mass effects (IME) in rotational dynamics, are mitigated using a fixed relaxation technique for the rigid body kinematics. For implicit coupling, a fixed-point strategy is employed, complemented by the relaxation technique used for the IME to ensure robustness. Additionally, the proposed coupling algorithms leverage the DF-IBM formulation and the predictor-corrector strategy of the PISO solution algorithm, by excluding the momentum predictor step and the time-intensive corrector loops from the implicit iterations.

The proposed method is validated through various stationary, prescribed, and freely moving immersed boundary cases, with results compared against experimental and numerical data from the literature. The method demonstrates robustness, accuracy, and efficiency in handling the complex dynamics of fluid–rigid body interactions across a range of challenging scenarios. The suggested improvements integrate seamlessly into existing incompressible fluid solvers with minimal adjustments to the original system of equations, highlighting their ease of implementation. Finally, the present work is implemented within the cell-centred finite volume approach inside the open-source `C++` toolbox OpenFOAM environment, version 7.0 of the OpenFOAM Foundation variant.

Keywords

Immersed boundary method; direct-forcing; Lagrangian weight; rigid body dynamics, fluid–rigid body interaction; incompressible flow; finite volume method.

Acknowledgements

First and foremost, I wish to express my sincere gratitude and appreciation to my PhD supervisors, Prof. Abdellatif Ouahsine from UTC and Dr. Patrick G. Verdin from CU, for their invaluable mentorship, unwavering support, and insightful guidance throughout this challenging but rewarding endeavour. Their expertise and dedication have been a source of inspiration and have significantly shaped both my research and my growth as a scholar.

I extend my sincere gratitude to the jury members for their time and effort in evaluating my work. In particular, I would like to thank the reporters, Prof. Sylvain Guillou and Prof. Elisabeth Lacazedieu, for their insightful reviews. I am also deeply appreciative of the examiners, Dr. Luofeng Huang, Prof. Alain Rassineux, Prof. Toufik Kanit, and Dr. Philippe Druault, for their valuable feedback and contributions.

I would also like to acknowledge Prof. Adnan Ibrahimbegovic and Prof. Abdelouahed Laksimi for their enriching discussions and constructive feedback during the annual review meetings at UTC, as well as Dr. Tamás I. Józsa and Dr. Heather Smith during the annual review meetings at CU. Their input has been instrumental in refining my work.

A special acknowledgment goes to Dr. Badr Kaoui for his generous support and practical advice, which have been immensely helpful in navigating the complexities of my thesis.

I also extend my heartfelt gratitude to Dr. Tom-Robin Teschner from CU for introducing me to this PhD opportunity and supporting me throughout the application process. I am deeply thankful for his belief in my potential and his encouragement at the very beginning of this journey.

I am also deeply appreciative of the faculty and laboratory members at both UTC and CU who contributed to making my experience at these institutions intellectually stimulating and personally fulfilling. Their collaboration, kindness, and support have left a lasting impression on me.

Last but not least, I am profoundly thankful to my family, my partner, and my friends for their endless encouragement, patience, and love. Their unwavering belief in me has been my anchor and a constant source of motivation throughout this long journey.

Table of Contents

List of Figures

List of Tables

List of Algorithms

Abbreviations

ALE	Arbitrary Lagrangian–Eulerian
BDF2	Backward Differentiation Formula of order 2
BiCG	Bi-Conjugate Gradient
BiCGStab	Bi-Conjugate Gradient Stabilised
CBC	Convection Boundedness Criterion
CD	Central Differencing
CFD	Computational Fluid Dynamics
CFL	Courant–Friedrichs–Lewy
CG	Conjugate Gradient
CPU	Central Processing Unit
CSD	Computational Structural Dynamics
DF-IBM	Direct-Forcing Immersed Boundary Method
DIC	Diagonal-based Incomplete Cholesky
DILU	Diagonal-based Incomplete LU factorisation
DOF	Degrees-Of-Freedom
FDIC	Fast Diagonal-based Incomplete Cholesky
FDM	Finite Difference Method
FEM	Finite Element Method
FSI	Fluid–Structure Interaction
FVM	Finite Volume Method
GAMG	Geometric Agglomerated Algebraic Multi-Grid
IBM	Immersed Boundary Method
IBPM	Immersed Boundary Projection Method
IIM	Immersed Interface Method

IME	Internal Mass Effect
LBM	Lattice Boltzmann Method
LES	Large Eddy Simulation
MG	Multi-Grid
MIBM	Moving Immersed Boundary Method
MLS	Moving-Least-Squares
mPPE	Modified Pressure Poisson Equation
NVD	Normalised Variable Diagram
OpenFOAM	Open Field Operation And Manipulation
PDE	Partial Differential Equation
PISO	Pressure Implicit with Splitting of Operators
PPE	Pressure Poisson Equation
RANS	Reynolds-Averaged Navier-Stokes
RBD	Rigid Body Dynamics
RBF	Radial Basis Function
RKPM	Reproducing Kernel Particle Method
TVD	Total Variation Diminishing
UD	Upwind Differencing
VAT	Vertical Axis Turbine

Symbols, Units, and Names

Latin Symbols	Unit	Name
$[A]$	$-$	Coefficient matrix
A	m^2	Area
A_0	m	Translational oscillation amplitude
A_f	m^2	Face f surface area
$\mathbf{a}_s$	m/s^2	Solid linear acceleration vector about centre of mass
a, b, l	m	Wake dimensions
a_N	$-$	Non-diagonal terms coefficient
a_P	$-$	Diagonal terms coefficient
C	$-$	Sum of squares condition constant
C_D	$-$	Drag force coefficient
C_L	$-$	Lift force coefficient
C_N	$-$	Normal force coefficient
C_P	$-$	Power coefficient
C_T	$-$	Tangential force coefficient
c	m	Chord length
D	m	Diameter
$\mathbf{d}$	m	Vector connecting owner cell P and neighbouring cell N
E	$-$	Error
$\mathbf{F}_b$	N	Buoyancy force
$\mathbf{F}_c$	N	Collision force
$\mathbf{F}_{ext}$	N	External force vector
$\mathbf{F}_g$	N	Gravity force

$\mathbf{F}_h$	N	Hydrodynamic force
$\mathbf{F}_{IB}$, $\mathbf{F}_n$	m/s^2	Lagrangian boundary force vector
F_D	N	Drag force
F_L	N	Lift force
F_N	N	Normal force
F_T	N	Tangential force
$\mathbf{f}$	m/s^2	Eulerian boundary force vector
$\mathbf{f}_{ext}$	N/m^3	External body force vector
f	$1/s$	Oscillation characteristic frequency
f_s	$1/s$	Natural vortex shedding frequency
f_w	$-$	Linear interpolation weighing factor
$\mathbf{g}$	m/s^2	Earth's gravitational acceleration
H	$-$	Heaviside step function
h	m	Mesh spacing
$\mathbf{I}$	$-$	Identity matrix
$\mathbf{I}_s$	$kg \cdot m^2$	Solid mass moment of inertia tensor
I_s	$kg \cdot m^2$	Solid mass moment of inertia around z-axis
K	$-$	Second moment condition constant
$\boldsymbol{k}$	m	Non-orthogonal vector part of $\mathbf{S}_f$
k	$-$	User-specified factor in OpenFOAM for Gamma scheme
KC	$-$	Keulegan-Carpenter number
$\mathbf{L}_{int}$	$kg \cdot m/s$	Linear momentum of internal Lagrangian markers
L_1, L_2, L_∞	$-$	Error norms
m_s	kg	Solid mass
N_{Lag}	$-$	Number of Lagrangian markers
N_x, N_y	$-$	Number of Eulerian grid cells in x and y-directions

$\mathbf{n}$	$-$	Outward-pointing unit normal vector
$\mathbf{n}_f$	$-$	Outward-pointing unit normal vector to the face f
$\mathbf{P}_{int}$	$kg \cdot m^2/s$	Angular momentum of internal Lagrangian markers
P	W	Power
p	m^2/s^2	Kinematic pressure
p_0	m^2/s^2	Initial kinematic pressure field
p_s	Pa	Fluid static pressure
$\mathbf{R}$	$-$	2D rotational matrix
$\mathbf{R}_1$, $\mathbf{R}_2$	$-$	Residuals
$[R]$	$-$	Source terms vector
R	m	Radius
$\mathbf{r}_{i,j}$	m	Relative position vector of $\mathbf{x}_{i,j}$ with respect to $\mathbf{X}_s$
$\mathbf{r}_n$	m	Relative position vector of $\mathbf{X}_n$ with respect to $\mathbf{X}_s$
r_s	$-$	Kernel function support radius
r_x, r_y	$-$	Non-dimensional distance between $\mathbf{X}_n$ and $\mathbf{x}_{i,j}$
Re	$-$	Reynolds number
$\mathbf{S}$	m^2	Outward-pointing surface element vector
$\mathbf{S}_f$	m^2	Outward-pointing face area normal vector to the face f
S	m^2	Surface element vector magnitude
Δs	m	Lagrangian markers spacing
Δs_{int}	m	Internal Lagrangian markers spacing
Str	$-$	Strouhal number
$\mathbf{T}_{ext}$	$N \cdot m$	External torque vector
$\mathbf{T}_h$	$N \cdot m$	Hydrodynamic torque
T	s	Oscillation period
t	s	Time

t_{rev}	s	Time per revolution
Δt	s	Simulation Time-step
$\mathbf{U}$	m/s	Predicted Lagrangian boundary velocity vector
$\mathbf{U}_{IB}$, $\mathbf{U}_n$	m/s	Lagrangian boundary velocity vector
$\mathbf{U}_s$	m/s	Solid linear velocity vector about centre of mass
$\mathbf{u}$	m/s	Velocity vector
$\mathbf{u}^*$	m/s	Predicted Eulerian velocity vector
$\tilde{\mathbf{u}}$	m/s	Updated Eulerian velocity vector
$\hat{\mathbf{u}}$	m/s	Explicit corrected Eulerian velocity vector
u_0, v_0	m/s	Initial velocity field
u_∞, v_∞	m/s	Freestream velocity
u_n	m/s	Advection velocity
u_{wall}	m/s	Moving wall velocity
V	m^3	Volume
V_P	m^3	Volume of owner cell P
V_s	m^3	Solid volume
V_t	m/s	V-component of the terminal linear velocity
ΔV_{int}	m^3	Internal Lagrangian markers weight
$\Delta V_{i,j}$	m^3	Eulerian weight in 3D
W_n	m^3	Lagrangian weight in 3D
$\mathbf{X}_{int}$	m	Internal Lagrangian markers position vector
$\mathbf{X}_n$	m	Solid position vector of Lagrangian marker n
$\mathbf{X}_s$	m	Solid centre of mass position vector
$\mathbf{X}'_n$	m	Imaginary solid position vector of Lagrangian marker n
$\mathbf{X}'_s$	m	Imaginary solid centre of mass position vector
$\Delta\mathbf{X}$	m	Change in translational motion

X, Y, Z	m	Cartesian coordinates in Lagrangian domain
$\mathbf{x}$	m	Fluid position vector
$\mathbf{x}_{i,j}$	m	Fluid position vector of Eulerian grid point (i, j)
$\Delta x, \Delta y$	m	Eulerian grid spacing
x, y, z	m	Cartesian coordinates in Eulerian domain

Greek Symbols	Unit	Name
α	rad	Angle of attack
α_r	$-$	Relaxation parameter
$\boldsymbol{\alpha}_s$	rad/s^2	Solid angular acceleration vector about centre of mass
α_s	rad/s^2	Solid angular acceleration around z-axis about centre of mass
β_m	$-$	User-specified factor for Gamma scheme
γ	$-$	Blending factor for Gamma differencing scheme
$\boldsymbol{\Delta}$	m	Orthogonal vector part of $\mathbf{S}_f$
δ_h	$1/m^3$	Interpolation function in 3D
ε_h	$-$	Correction factor for Lagrangian support domain
ε_w	$-$	Collision model constant stiffness parameter
ϵ_{IBM}	$-$	Implicit IBM iterations tolerance
ϵ_{FSI}	$-$	Implicit FSI iterations tolerance
$\boldsymbol{\varepsilon}$	$1/s$	Fluid strain-rate (or deformation-rate) tensor
θ, θ_s	$^\circ$	Solid rotation angle
θ_0	rad	Rotational oscillation amplitude
θ_{ws}	$^\circ$	Wake separation angle
$\Delta\theta$	rad	Change in rotational motion
λ	$-$	Tip speed ratio
μ, μ_f	$Pa \cdot s$	Fluid dynamic viscosity
ν, ν_f	m^2/s	Fluid kinematic viscosity
ξ	$-$	Collision model safe zone threshold
ρ, ρ_f	kg/m^3	Fluid density
ρ_s	kg/m^3	Solid density
$\boldsymbol{\sigma}$, $\boldsymbol{\sigma}_f$	Pa	Fluid Cauchy (or total) stress tensor

$\boldsymbol{\sigma}_{dev}$	Pa	Fluid deviatoric (or viscous) stress tensor
$\boldsymbol{\sigma}_{iso}$	Pa	Fluid isotropic (or hydrostatic) stress tensor
τ	Pa	Fluid viscous (or shear-rate) stress tensor
Φ	$-$	Signed-distance function
ϕ_f, ϕ_P, ϕ_N	m^3/s	Velocity flux field
ϕ_i $(i \in \mathbb{N})$	$-$	Kernel function
$\tilde{\phi}_P$	$-$	Normalised velocity flux field
φ, φ_0	rad	Phase angle difference
Ψ	$-$	Any transported quantity (scalar or vector field)
$\boldsymbol{\omega}_s$	rad/s	Solid angular velocity vector about centre of mass
ω, ω_s	rad/s	Solid angular velocity around z-axis about centre of mass
ω_t	rad/s	Terminal angular velocity around z-axis about centre of mass

Operators	**Name**
∂	Partial derivative
∇	Differential operator
$\mathbf{H}$	Linear operator for the pressure Poisson equation
$\mathscr{I}$	Velocity interpolation operator
$\mathscr{S}$	Force spreading operator

Domains	**Name**
Γ_s	Immersed boundary Lagrangian domain
Ω_f	Fluid Eulerian domain
Ω_s	Solid rigid body domain
$\mathscr{D}_{\mathbf{X}_n}$	Lagrangian support domain

Journal Publications

Published

E. Farah, A. Ouahsine, and P. G. Verdin, **"An improved implicit direct-forcing immersed boundary method (DF-IBM) around arbitrarily moving rigid structures"**, *Physics of Fluids*, Vol. 36, No. 10, pp. 103618, 2024. *DOI:* 10.1063/5.0231218 – Editor's Pick (EP).

Z. Huang, A. Ouahsine, E. Farah, P. Du, and X. Zhao, **"Coupled computational fluid dynamics-discrete element method for simulating the interactions between ship-induced waves and riprap on restricted waterway banks"**, *Physics of Fluids*, Vol. 36, No. 10, pp. 107140, 2024. *DOI:* 10.1063/5.0230654.

Accepted

E. Farah, A. Ouahsine, and P. G. Verdin, **"Fast Implicit Direct-Forcing Immersed Boundary Method (FIDF-IBM)"**, *Engineering Analysis with Boundary Elements*.

Under Review

E. Farah, A. Ouahsine, P. G. Verdin and B. Kaoui, **"Stable fluid-rigid body interaction algorithm using the direct-forcing immersed boundary method (DF-IBM)"**, *Journal of Computational Physics*.

Chapter 1

Introduction

Contents

This chapter serves as an introduction to the present research, outlining the motivation behind the study, followed by a clear presentation of the thesis aim and objectives. The chapter concludes with an overview of the structure of the thesis.

1.1 Background and Motivation

Fluid–structure interaction (FSI) refers to the study of a coupled multiphysics system composed of two distinct physical entities: a moving rigid or deformable structure and a surrounding fluid flow. This phenomenon plays a crucial role in a variety of scientific and technological fields, including, but not limited to, particle sedimentation [1], aerodynamics (aeroelasticity) [2], biofluid and biomechanics [3], hydrodynamics [4], and many more. At its core, FSI involves the intricate interaction between a solid structure and its surrounding fluid, with each influencing the other. For example, the motion and/or deformation of solid structures are driven by the forces exerted by the surrounding fluid, while, conversely, the movement and/or deformation of the structure impacts the fluid dynamics. This reciprocal relationship leads to a bi-directional coupling that is inherently challenging to solve.

Analytical solutions are typically unattainable for most FSI problems, and laboratory experiments are often constrained in scope and prohibitively expensive. To overcome these challenges and gain insight into the fundamental multiphysics behind the interactions between fluids and solids, numerical simulations offer a valuable alternative. These simulations allow for a deeper understanding of the underlying phenomena, while significantly reducing the need for costly and time-consuming physical experiments. In recent decades, the rise of powerful computational tools and resources has made numerical simulation an increasingly valuable approach for investigating the fundamental physics of FSI.

Consequently, the development of numerical techniques for simulating FSI has attracted considerable attention across various scientific communities. FSI simulations have become a key focus of research, with continuous efforts to improve performance and develop new techniques capable of handling a wide range of complexities, ranging from mesh adaptation challenges to the coupling of diverse computational codes, from laminar and highly turbulent flows to large/small structural displacement, and from simple to highly intricate geometries.

In this context, the present research intends to advance a novel formulation for solving general-purpose FSI problems, with a particular focus on the interaction between incompressible fluid flows and complex solid bodies undergoing large, free motion. The research seeks to address the complexities associated with FSI simulations, providing more efficient and practical solutions for both academic and industrial applications. A key aspect of this work is the adoption of a body non-conformal mesh method, which avoids the computational challenges associated with heavy mesh generation during pre-processing and the meshing issues that arise during FSI solving, especially when large structural motions occur.

1.2 Aim and Objectives

The aim of this research is to reduce the time required for the pre-processing and processing stages associated with traditional numerical simulations by adopting and enhancing the immersed boundary method (IBM), an existing body non-conformal approach. The IBM enables complex rigid structures to move arbitrarily and freely within an incompressible viscous fluid without requiring the generation of complex body-fitted meshes. By eliminating the need for sophisticated dynamic and/or re-meshing techniques, which are known to be computationally intensive, the IBM simplifies mesh generation to a trivial task requiring only a few seconds of computational time. This research seeks to streamline simulation workflows and improve the overall efficiency of the numerical simulation process.

The objectives to achieve the aforementioned aim are given as follows:

- Develop an accurate and efficient implicit direct-forcing IBM (DF-IBM) solver by improving the implicit iterations to enforce accurately the kinematic conditions on the immersed boundary.
- Integrate the DF-IBM into the 2D incompressible Navier-Stokes fluid solver within the open-source Computational Fluid Dynamics (CFD) code OpenFOAM, by leveraging the Pressure Implicit with Splitting of Operators (PISO) algorithm to satisfy the newly coupled fluid–solid dual system constraints.
- Implement the Newton-Euler equations of motion to compute the kinetics and kinematics of a solid rigid body undergoing free motion.
- Couple the incompressible Navier-Stokes equations with the rigid body dynamics under the DF-IBM framework to study weakly and strongly coupled fluid–rigid body interaction problems.

1.3 Thesis Structure

This thesis work is composed of the following chapters:

- **Chapter 1** serves as an introduction to this research, presenting the motivation behind the study along with its aims and objectives.
- **Chapter 2** provides a concise overview of the literature on two mesh approaches, namely the body conformal and body non-conformal techniques, used for simulating

fluid–structure interaction problems. Additionally, it highlights the developments and advancements of the immersed boundary method in detail.

- **Chapter 3** derives the governing equations for incompressible fluid flow and provides an in-depth explanation of the cell-centred finite volume discretisation applied to all terms, as implemented in OpenFOAM. The chapter closes with a detailed derivation of the PISO algorithm steps for managing pressure–velocity coupling, alongside an overview of the system of linear equation solvers available in OpenFOAM.

- **Chapter 4** describes the theoretical foundation and computational framework of DF-IBM. This chapter proposes an efficient implicit variant of the DF-IBM, which sequentially enforces the dual constraints of the formed fluid–solid system for an incompressible flow by leveraging the operator-splitting strategy of the PISO algorithm.

- **Chapter 5** presents numerical results validating the proposed algorithm against experimental and numerical data from the literature. The chapter evaluates the accuracy, performance, and sensitivity of the method. Moreover, stationary and prescribed motion cases are analysed to demonstrate the method's effectiveness in handling complex dynamics.

- **Chapter 6** introduces the rigid body dynamics equations of motion, with a particular focus on the internal mass effect arising from hydrodynamic forces and torques. It also provides the formulation resulting from the coupling of fluid equations with rigid body dynamics within the DF-IBM framework. Furthermore, the chapter proposes efficient and stable partitioned explicit and implicit coupling algorithms, that leverage the predictor-corrector strategy of the PISO algorithm and build upon the implicit DF-IBM methodology presented in Chapter 4.

- **Chapter 7** presents numerical results for fluid–rigid body interaction problems, showcasing the method's ability to address challenges arising from the internal mass effect (IME) in cases involving neutrally buoyant, non-buoyant, and buoyant solids. Besides, it demonstrates the method's effectiveness in overcoming difficulties associated with the rigid body approximation of the IME in rotational dynamics.

- **Chapter 8** summarises the key conclusions of the present research and outlines potential extensions and recommendations for future work.

Chapter 2

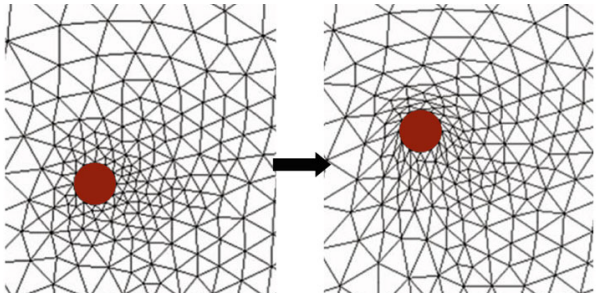
Literature Review

Contents

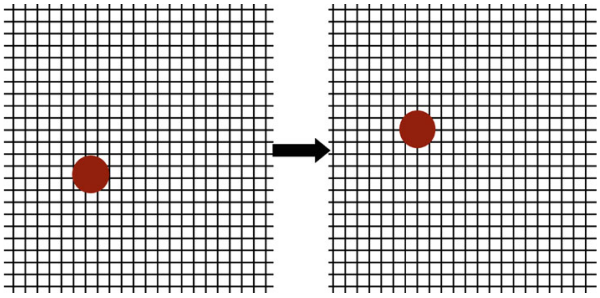
This chapter presents a comprehensive literature review of bodyconformal and body non-conformal mesh methods for simulating fluid–structure interaction (FSI) problems. It provides a general overview of the Arbitrary Lagrangian–Eulerian (ALE) and overset (Chimera) methods within the body-fitted framework, highlighting their respective advantages and limitations. Particular emphasis is placed on the immersed boundary method (IBM), a prominent non-conformal approach. The review explores the fundamental principles of IBM, its advancements, applications, and key challenges (limitations), with a specific focus on the direct-forcing IBM (DF-IBM) variant. Additionally, various FSI strategies within the IBM framework are examined, providing a detailed discussion of their underlying principles and potential improvements.

2.1 Body Conformal Approach

Complex fluid–structure interaction (FSI) problems are typically simulated using conventional body conforming mesh techniques, where the fluid–structure interface is described by physical boundary conditions treated as part of the solution. These simulations require computational meshes that conform to both the fluid and structural domains to accurately capture motion and deformation, particularly in cases involving non-rigid, deformable structures. The interaction is governed by the two-way coupling between the fluid and structure. However, body-conforming methods, also known as body-fitted mesh methods, become computationally expensive for complex geometries when body motion and deformation are present. This is due to the need for frequent re-meshing or mesh updates as the solution progresses, as illustrated in Fig. (2.1a). This necessitates the development of sophisticated techniques and strategies to avoid critical mesh distortion and strong deformation while preserving the discretisation accuracy [5–9].



(a) Body conformal mesh method, at t^n (left) and t^{n+1} (right)



(b) Body non-conformal mesh method, at t^n (left) and t^{n+1} (right)

Figure 2.1: *Difference between body conformal and body non-conformal mesh methods, taken from [10].*

FSI methods based on the body conformal meshes typically involve three key components: (1) fluid dynamics, modelled using Computational Fluid Dynamics (CFD) solvers; (2)

structural dynamics, addressed through Computational Structural Dynamics (CSD) solvers; and (3) mesh movement to accommodate the interface dynamics. Among the most widely recognised and effective approaches for capturing fluid–structure interactions are the Arbitrary Lagrangian–Eulerian (ALE) method [11] followed by the multi-block structured grids known as overset (or Chimera) grids [12, 13].

2.1.1 Arbitrary Lagrangian–Eulerian (ALE)

This method combines the strengths of both the Lagrangian and Eulerian formulations. In the Lagrangian approach, the reference frame moves with the solid particles, facilitating accurate tracking of the fluid–solid interface and simplifying the imposition of boundary conditions, resulting in a sharply defined interface. In contrast, the Eulerian approach uses a fixed reference frame in the fluid space, allowing fluid quantities to flow through it and enabling effective handling of significant mesh distortions. This mixed formulation creates a system where the structure moves relative to the fluid mesh (Eulerian), with its external shape governed by the imposed boundary conditions (Lagrangian). The moving mesh is explicitly integrated into the fluid governing equations, particularly within the convective term, by subtracting the mesh velocity from the fluid velocity [11]. This approach enables the mesh to move dynamically and adaptively within the fluid domain, aligning with the motion of the structural boundary.

ALE methods have been shown to perform well in cases where the structure's rotations, translations, or deformations remain within specific bounds [14, 15]. However, when these bounds are exceeded, such as in cases of large motion, the moving mesh can result in poorly shaped mesh elements, ultimately causing the ALE method to fail. For example, see Fig. (2.2). A common solution to this issue is coupling the ALE method with a re-meshing procedure [16, 17], which involves data mapping strategies to transfer mesh information between the old and new meshes [18].

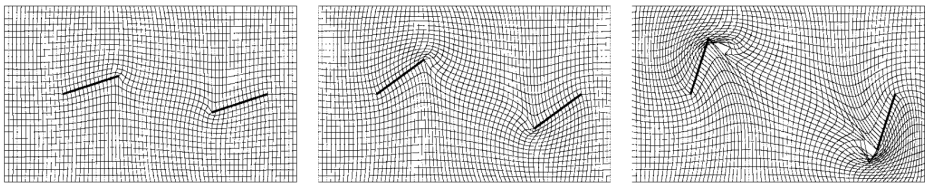


Figure 2.2: *ALE mesh with large distortions of a thin structure in motion, taken from [19].*

2.1.2 Overset (Chimera) Method

The fundamental idea of the overset, or Chimera, approach is to overlap multiple structured grid blocks within a unified computational framework. A curvilinear, body conformal structured grid is used to handle complex geometrical shapes, which is overlaid on a fixed Cartesian background grid, either uniform or non-uniform. The process of constructing a Chimera grid can be summarised in the following steps:

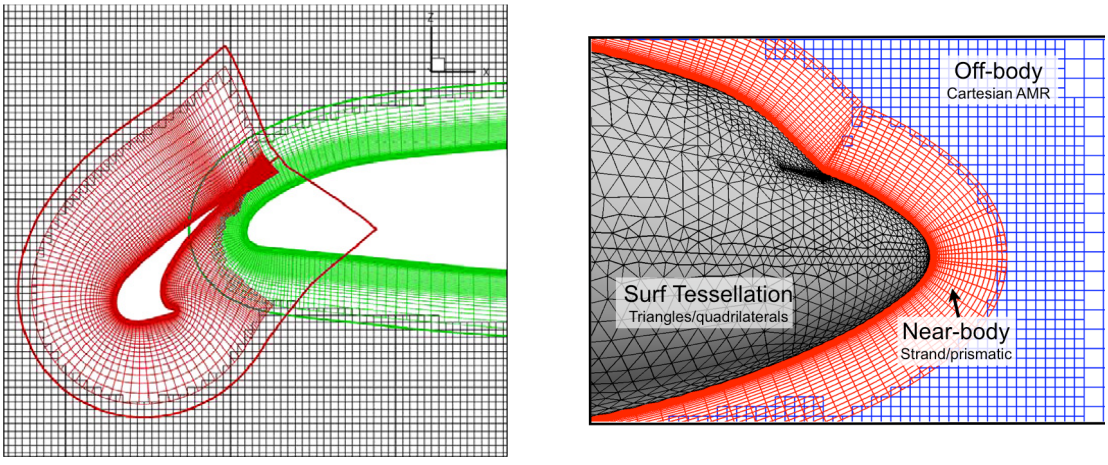
1. Generate a fixed Cartesian background mesh: this step is straightforward and relatively easy to implement.

2. Build a curvilinear, body-fitted mesh: this step is designed for the complex geometry under consideration and can be challenging and time-intensive, depending on the complexity of the geometry.

3. Overlap the meshes: align the different grids at their respective physical locations within a single unified framework.

4. Cut a hole in the background grid: this step removes the portions of the background grid that correspond to the body-fitted mesh of the complex geometry.

The final result of the Chimera grid applied to complex geometrical shapes, such as a dual-element airfoil or an airplane fuselage cone, is demonstrated in Fig. (2.3).



(a) Dual-element airfoil *(b) Airplane fuselage nose cone*

Figure 2.3: *Multi-block structured girds using the overset (Chimera) grid over different complex geometries, taken from [20] (left) and [21] (right).*

The solution process for a Chimera grid involves the following steps:

1. Solve the governing equations on the background mesh.
2. Interpolate the solution to the block boundary of the body-fitted mesh.
3. Solve the governing equations on the complex geometry's body-fitted mesh.
4. Interpolate the updated solution back to the hole boundary of the background mesh.
5. Repeat the above steps iteratively until a converged solution is achieved.

This approach was subsequently extended to accommodate unstructured grids [22, 23] and further developed to handle multiple bodies in contact [24]. The overset method enables the simulation of interactions between complex fluid flows and large, moving rigid bodies, eliminating the need for dynamically deforming and/or re-meshing of the background grid at each time-step. Instead, grid blocks around moving bodies are simply displaced, with new holes cut in the main background grid. This method helps alleviate the difficulties of mesh generation during the pre-processing phase and mitigates the limitations of traditional mesh generation algorithms in adapting to complex geometries.

Despite these advancements, the primary challenge in implementing the Chimera method lies in handling data interpolation between overlapping blocks, managing boundary conditions at the block and hole boundaries, and addressing the placement of these boundaries. Additionally, the amount of overlap and the quality of the grid in the overlapping region play crucial roles. If these steps are not executed properly, accuracy may degrade, leading to numerical errors and potential divergence [25, 26].

2.1.3 Concluding Remarks

Applying the body-fitted approach, such as the ALE method or the overset method, to moving boundary problems necessitates managing dynamic mesh motion or performing mesh updates (re-meshing) at each time increment. This process also requires specialised techniques to transfer solution data onto the updated grid or between overlapping grids. Such procedures demand significant user expertise and intervention to manage the meshing process effectively. Additionally, these methods are known to increase the computational cost of FSI simulations and are prone to errors, potentially introducing numerical inaccuracies into the desired solution. The challenges become even more pronounced with large structural motions, making the body-fitted approach less favourable compared to body non-conformal methods like the immersed boundary method (IBM). The latter employs a stationary, non-deforming grid, which greatly simplifies mesh handling. Despite substantial advancements in body-fitted

methods for moving boundaries [18, 27, 28], body non-conformal approaches remain an attractive alternative for simulations involving body motion due to their inherent simple mesh generation and robustness.

2.2 Body Non-Conformal Approach

Unlike the previous approach, body non-conforming meshes handle the boundary/interface condition as a constraint imposed on the flow governing equations, allowing non-conformity with the solid structure. This enables the fluid and solid governing equations to be solved independently, each on its own computational grid, without the need for mesh motion or re-meshing at each time increment, as illustrated in Fig. (2.1b). This method serves as an alternative to body-fitted techniques, as it facilitates the usage of simple Cartesian uniform or non-uniform meshes.

This approach notably simplifies the heavy mesh generation procedure, by eliminating mesh deformation and distortions, and the re-meshing procedures in case of moving structures. Paving the way for a new era of non-conforming meshing techniques, the most widely used example in recent years is the IBM [29]. Fig. (2.4) reflects the scientific community's growing interest in the applicability and effectiveness of the IBM in recent years. This increasing attention has led to significant amount of research in this field aimed at advancing the method, overcoming its limitations, and broadening its scope of applications.

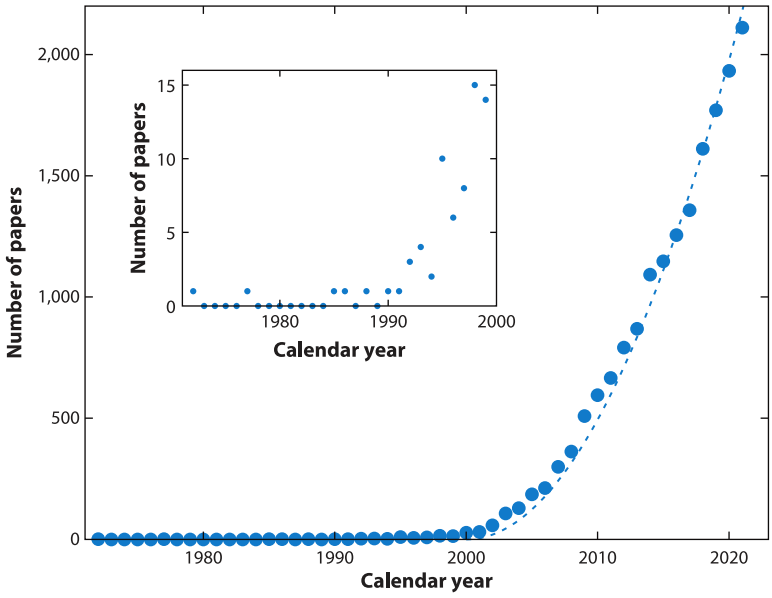


Figure 2.4: *Number of scientific publications with the immersed boundary method in its title, taken from [29].*

2.2.1 Immersed Boundary Method (IBM)

Developed by Peskin in 1972 [30, 31], the IBM was first introduced to simulate the FSI between blood flow patterns inside the human heart with flexible valves, which was later extended by Peskin in 1977 [32] to include the muscular heart wall.

The fundamental idea behind the IBM is to use two separate domains (or grids). The first is fixed and non-deformable, labelled as the Eulerian domain, whose nodes do not conform to the solid geometry, and where the fluid Navier-Stokes equations are solved. The second, labelled as the Lagrangian domain, represents the solid immersed boundary using a discrete set of points, named Lagrangian markers, which are used to track the information at the embedded structure border. These two domain representations eliminate the meshing-related issues discussed earlier when a fluid and a solid structure are coupled using the ALE method in the body conforming realm [15, 33]. As a result, separate interaction equations were created to transfer the data between both domains, named interpolation and spreading operators. The novelty of IBM lies in the incorporation of a fictitious body force, named boundary force hereupon, in the governing equations as a source term to compensate for the real effects of the solid immersed boundary. Although multiple steps are required to compute the boundary forces and execute the interaction equations for the data transfer between the two domains, the IBM offers a computational advantage over body-fitted methods. This is because the overall numerical approach involves fewer operations and less communication per grid point as reported in [29]. In addition, for moving and deforming bodies, the IBM becomes the more practical and efficient choice compared to the body-fitted approach, as it eliminates the need for mesh deformation and re-meshing of the fluid domain

Such imposition of the boundary force involves altering the fluid flow in the vicinity of the solid boundary. Subsequently, the governing equations can be discretised using the traditional Finite Difference Method (FDM), Finite Volume Method (FVM), Finite Element Method (FEM), Lattice Boltzmann Method (LBM), etc. without forcing any coordinate system transformation and complex discretisation operators.

2.2.1.1 Classification

The IBM can be classified according to the fluid–structure interface representation, i.e., either diffused-interface, where the interface characterised by singular functions along the surface is smeared through a regularised discrete Dirac delta function over the neighbouring Eulerian grid points, or sharp-interface where a sharp representation of fluid–structure interface is obtained since no regularisation of the forcing term is present. The class of family that

belongs to the diffused-interface methods include the classical IBM of Peskin [30, 31, 34–36], the feedback forcing IBM [37, 38], the direct-forcing IBM (DF-IBM) [39–41], the fictitious domain method [42], and many more. In contrast, sharp-interface methods feature the Cut-Cell method [43–46], Ghost-Cell method [47–49], immersed interface method (IIM) [50–53]. An overview of the methods outlined earlier is illustrated in the diagram in Fig. (2.5).

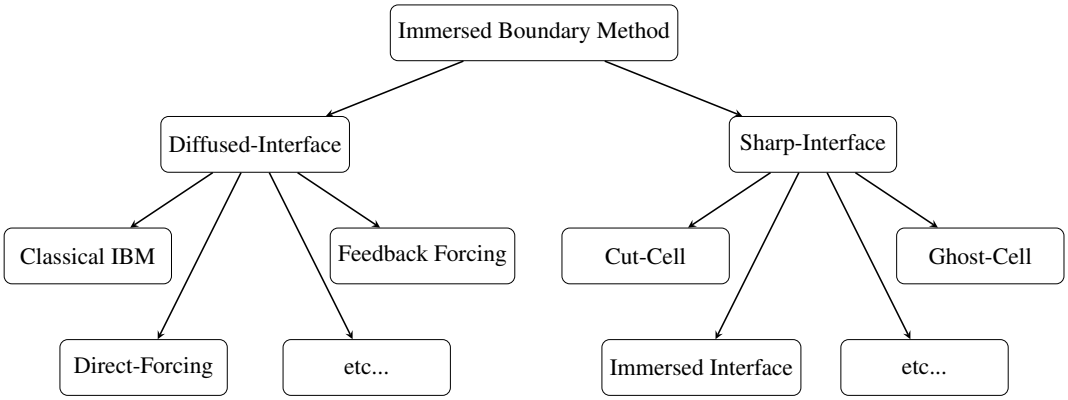


Figure 2.5: *Overview of IBM classification based on fluid–structure interface representation.*

2.2.1.2 Applications and Limitations

Based on how the boundary force is treated, a vast number of IBM variants have emerged that are suited for specific application areas. Several review papers can be found in the literature [29, 54–58], which discuss thoroughly and extensively all variants of the IBM and their possible area of applications along with a comprehensive analysis of the current developments in the IBM field.

The application of the IBM began with simulations of biological blood flows [30, 31], and it quickly expanded to a broad range of engineering fields and domains. This demonstrates the method's versatility, accuracy, and fidelity in numerical simulations. For example, IBM has been used to study the hydrodynamics of schooling fish [59], where it analysed the swimming performance of multiple bodies with prescribed motions subjected to incoming flow. The body non-conformal approach, namely the IBM, effectively simulated the multi-bodies dynamics and deformation in this complex flow configuration. In the field of unsteady aerodynamics, IBM has been applied to study 3D flying birds [60, 61], providing insights into their flight dynamics. The method has also been used to investigate FSI problems, such as the impact of a spherical rigid pendulum on a rubber membrane in a viscous fluid [62], where IBM accurately captured the contact dynamics of deformable bodies. IBM has successfully simulated transitional and turbulent wall-bounded flows over both simple [63] and com-

plex [64] surface geometries. Additionally, high-speed compressible flows have also been a focus of IBM research [65, 66]. Most recently, the IBM applicability has been extended to the simulation of vertical axis turbines [67], further highlighting its broad applicability and adaptability.

Despite its clear advantages, the IBM has several limitations that are strongly problem-dependent and require careful consideration when selecting the most appropriate IBM algorithm or variant, and whether additional modelling is necessary. One of the most prominent challenges in the IBM realm arises in turbulence-dominated flows with high Reynolds numbers. This limitation is primarily due to difficulties in resolving the thin wall shear layers, which cannot leverage anisotropic grid refinement near solid boundaries in the wall-normal direction. To address this, numerous studies have proposed solutions such as wall-models and local grid refinement techniques. However, these approaches are not universally effective and may fail in certain scenarios. Conversely, body-fitted methods stand out in such situations, as boundary conditions are straightforward to impose, and grid refinement can be effectively concentrated in the wall-normal direction near the boundary, enabling accurate resolution of steep gradients and flow separation without refining unnecessary regions of the mesh. A key drawback of the IBM lies in the absence of wall-normal and wall-parallel cell refinement, which inhibits accurate grid resolution within near-wall layers. Instead, mesh refinement in IBM occurs isotropically throughout the domain, leading to over-refinement in regions/areas where it is not required. This inefficiency becomes increasingly problematic as the Reynolds number rises, resulting in thinner boundary layers that demand finer resolution. An empirical correlation presented in [29] highlights the disparity in computational requirements between body-fitted and body non-conformal methods mainly the IBM. For body-fitted methods, the number of grid points required to resolve the boundary layer scales as $N_{BC} \sim Re^{1.8}\log(Re)$, while for the IBM, it scales as $N_{IBM} \sim Re^{2.7}$. Plotting these correlations on a $log - log$ scale, Fig. (2.6) demonstrates the inadequacy of the IBM for highly turbulent flows. Efforts to extend the IBM's applicability to high Reynolds number flows have included local grid refinement techniques [65, 68, 69], though these remain far from achieving wall-resolved accuracy. Conversely, wall-modelled solutions have shown significant improvements, particularly in Reynolds-Averaged Navier-Stokes (RANS) turbulence models [70, 71] and Large Eddy Simulation (LES) turbulence models [72–74].

Another limitation of the IBM in incompressible flows is the difficulty in enforcing exactly the no-slip velocity boundary condition. This issue can lead to flow slippage and penetration into the immersed boundary. This challenge will be explored in detail in the

following section of this chapter, particularly in the context of the DF-IBM variant, along with potential remedies to address this problem. An additional significant drawback, is the occurrence of non-physical spurious force oscillations in the hydrodynamic forces, especially in scenarios involving moving immersed boundaries. This behavior, commonly encountered in the IBM, will also be discussed in the context of the DF-IBM, where solutions aimed at mitigating these spurious oscillations will be presented.

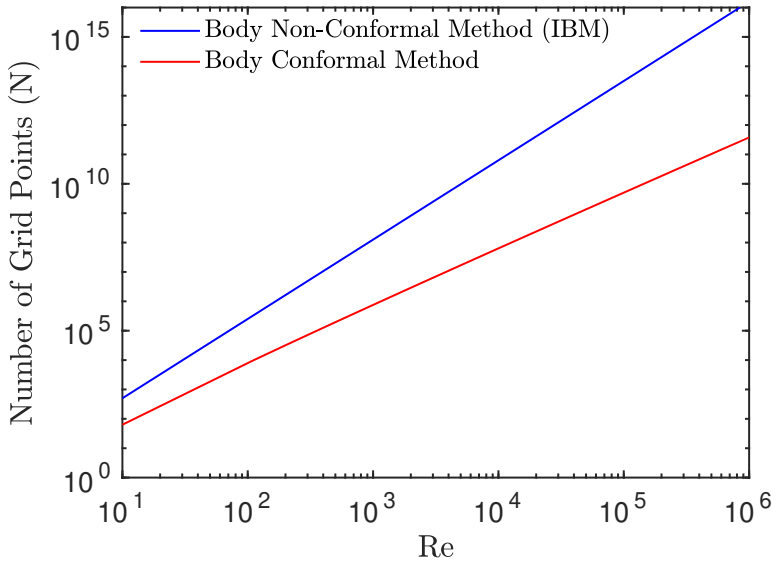


Figure 2.6: *Relationship between the number of grid points and the Reynolds number for body conformal and body non-conformal (IBM) methods.*

2.2.1.3 Direct-Forcing IBM (DF-IBM) Variant

The DF-IBM was initially proposed by [75, 76] and considered part of sharp-interface methods following an appropriate interpolation approach that avoids the usage of any force regularisation. It was later converted to the family of diffused-interface IBM by Uhlmann [39] by applying the discrete Dirac delta functions to cover the data mapping between the two grids.

In the present work, the DF-IBM of Uhlmann [39] is adopted, due to its widespread among researchers, stability, versatility, and ease of implementation in any fluid solver, owing to its minimal requirement to integrate it in the discretised form of the Navier-Stokes equations. The role of boundary force in the DF-IBM is to impose the desired velocity boundary condition on the immersed boundary.

The originality of the DF-IBM lies in eliminating the constitutive relation restriction imposed by the stiffness coefficient of Hooke's law found in the classical IBM [31], primarily designed for elastically deformable boundaries. Hence, a resulting stiff system is obtained

allowing the simulation of rigid non-deformable immersed boundaries. By removing the assumptions about the system's dynamics, the kinematics are enforced directly on the immersed boundary, i.e., the no-slip velocity boundary constraint. By doing so, the no-slip velocity boundary error at the fluid–solid interface is penalised, leading to flow penetration inside the solid immersed boundary. This occurs because the no-slip boundary condition is enforced through the boundary force term calculation using a predicted, non-divergence-free velocity field, rather than the final corrected flow field, which is unknown during the DF-IBM steps.

The approach of Uhlmann [39], termed as the explicit DF-IBM, failed remarkably to impose the no-slip boundary condition on the immersed boundary as reported in [41, 77, 78]. Therefore a multi-direct forcing approach was introduced, dubbed here multi-DF-IBM. The idea is to perform multiple direct-forcing loops to improve the no-slip boundary condition. The impact of this approach on the no-slip condition will be investigated in detail in Chapter 4. It was agreed in [41, 78] that three loops are enough to reduce the error on the immersed boundary while maintaining a good computational efficiency of the algorithm. This approach is also regarded as explicit, since a prescribed number of direct-forcing loops are implemented in the algorithm. If the exact imposition of the no-slip boundary constraint is of interest, implicit IBMs are constructed for such a purpose. Iterations of the whole IBM system are required until reaching a certain tolerance specified by the user, which approximates the boundary force computations by using a closer value to the final velocity field. A fully implicit IBM was developed in [79] for the projection method, referred to as the immersed boundary projection method (IBPM). The main idea behind this method is to couple the velocity field to both the pressure field and boundary force to form a set of Lagrange multipliers to enforce the divergence-free and no-slip boundary constraints simultaneously. A new form for the pressure Poisson equation (PPE) is obtained with the pressure and boundary force considered as unknowns, solved simultaneously in a single projection step. However, it was revealed by Ji et al. [80], that the IBPM suffers from convergence problems, when a Lagrangian marker is situated close enough to an Eulerian grid point, contributing to a larger condition number of the coefficient matrix of the new PPE, due to the singular property of the interpolation and spreading operators. Hence, to avoid such convergence problems, Ji et al. [80] proposed to iterate each constraint alone. This approach ensures that the coefficient matrix is never modified, with the no-slip boundary constraint being iterated inside the continuity constraint, which is taken as the outer loop. This results in a drastic increase in the computational effort. Although expensive, this approach is nevertheless

interesting as it provides simplicity and ease of implementation in any existing fluid solver code. A moving implicit IBM [81], referred to as MIBM, was later developed to maintain an acceptable computational efficiency compared to the explicit IBM variants and keep the accuracy of implicit IBM. The key idea of MIBM is that an additional equation related to the boundary force is derived and labelled as the moving force equation. Analogously to the IBPM approach [79], the boundary force in the MIBM is also regarded as a Lagrange multiplier to enforce the no-slip boundary condition, where the moving force equation is derived directly from the exact no-slip boundary condition, producing a smaller coefficient matrix that is well-conditioned compared to the one obtained in the IBPM. This method showed good results and improved performance. However, its increased implementation complexity and increased computational memory requirements to store a new coefficient matrix for the moving force equation, especially when the immersed boundary geometry involves a much higher number of Lagrangian markers or multiple bodies, make it less appealing. The efficiency and performance of several of the aforementioned explicit and implicit approaches are thoroughly examined in the results provided in Chapter 5.

Another approach to improve the imposition of the no-slip boundary constraint, is by improving the mathematical reciprocity of the IBM-related interpolation and spreading operators, which is translated directly to the computation of the Lagrangian weight used in the force spreading operator. It was proved in [82–84] that assigning the Lagrangian weight equal to the Eulerian weight, as originally suggested by Uhlmann in the explicit DF-IBM [39], would violate the reciprocal property of these operators, leading to a no-slip boundary error and fluid penetration inside the solid domain. To counter this problem, an appropriate method was proposed to determine the Lagrangian weight, see Pinelli et al. [85], for instance. These authors presented a mathematical method to calculate the appropriate Lagrangian weight by solving an additional system of linear equations that improves the reciprocity of the interpolation and spreading equations. Another study [83] introduced an analytical correction on the Lagrangian weight without the need to solve a linear system, which was later improved [84] to account for the immersed boundary curvature. The methodology behind each method is outlined in Chapter 4. In contrast, the influence of all proposed methods for computing the Lagrangian weight on the no-slip constraint is explored in the numerical results presented in Chapter 5.

The DF-IBM is known to suffer from non-physical spurious force oscillations in hydrodynamic computations involving moving immersed boundaries [84]. Numerous studies have been conducted to identify the causes of these numerical oscillations and propose mitigation

strategies [61, 86–88]. Uhlmann [39] initially addressed this issue by suggesting the use of a regularised discrete Dirac delta function with a larger support width. While this approach reduced spurious force oscillations, it also resulted in higher mean drag values. Later, Yang et al. [86] observed that when the derivative of the discrete delta function fails to meet certain moment conditions, non-physical oscillations dominate the solution. To resolve this problem, they developed a smoothing technique for discrete Dirac delta functions based on Uhlmann's concept. This technique led to the construction of various smoothed versions of commonly used discrete delta functions, which effectively reduced spurious force oscillations and significantly improved hydrodynamic force predictions. The definition of the moment conditions, along with several discrete delta functions and their smoothed counterparts, are reviewed in detail in Chapter 4. Another study [87] identified two primary sources of these oscillations. The first source is related to spatial discontinuities in the pressure field across the immersed boundary, while the second stems from temporal discontinuities in the velocity field. These discontinuities arise because, as the immersed boundary moves across the fixed Eulerian grid, grid cells dynamically change their status as the simulation advances, transitioning from internal grid points to interface markers, then from interface markers to external grid points, and vice versa. This dynamic status change invalidates the time history of such cells when computing explicit terms in the fluid Navier-Stokes equations. These unwanted oscillations degrade solution quality and can lead to over-predictions of aerodynamic coefficients. Their findings also revealed that reducing the fluid grid spacing or increasing the time-step could notably reduce the force oscillations, with grid spacing having a more dominant effect. This insight suggests that finer grids can better dampen these oscillations. To further counter these challenges, Yang and Balaras [89] proposed a "field-extension" approach. This method involves extrapolating velocity and pressure fields from adjacent fluid grid points for the dynamically changing cells. By doing so, the explicit terms in the fluid equations are computed more accurately, even as the immersed boundary motion alters the cell status.

Finally, as the DF-IBM belongs to the family of diffused-interface methods, the boundary condition at the interface (i.e., the immersed boundary) is not precisely enforced at the exact location of the boundary. Instead, it is distributed or smeared across several fluid grid points near the boundary due to the use of a regularised discrete Dirac delta function, resulting in the formation of a thin shell. This smearing can reduce the formal order of accuracy of the DF-IBM [35], particularly for non-smooth fields. Uhlmann [39] observed that second-order accuracy was preserved only in cases involving smooth fields, such as the Taylor–Green vortices problem. This issue will be further examined in Chapter 5 for the Taylor–Green

vortices problem. To reduce the thickness of the diffused interface and restore its "sharp" nature, especially for high Reynolds number turbulent flows, various strategies have been explored. The simplest approach involves using a compact discrete delta function [90] or decreasing the fluid grid spacing. Alternatively, the force can be spread exclusively to the interior of the immersed boundary [80, 91]. This approach successfully reduced the thickness of the diffused layer by a factor of two [80], achieving second-order accuracy as reported in [91]. However, it was discovered by [92] that this single-sided spreading formulation violated the torque identity, leading to inaccuracies in hydrodynamic torque computations for torque-driven flows. Another method to reduce the interface thickness is the "Lagrangian grid retraction" approach, introduced by Breugem [78]. In this technique, the interface is retracted inward by a fraction of the fluid grid spacing toward the interior of the immersed boundary. This adjustment offsets the diffused layer toward the interior without altering its overall thickness value. An optimal retraction distance of $0.3 \sim 0.4h$ was identified to achieve second-order accuracy [78] and was subsequently recommended in [92]. However, it was also found that this retraction distance did not fully recover the actual velocity gradient within the diffused-interface. In contrast, the method of spreading the boundary force exclusively to the interior of the immersed boundary was found to better preserve the sharpness of the interface and the accuracy of the velocity gradient. This finding suggests that single-sided spreading of the boundary force within the internal part of the immersed boundary may be preferable to the grid retraction approach for achieving sharper interfaces and improved accuracy.

2.2.1.4 Fluid–Structure Modelling Using DF-IBM

FSI problems in the context of the DF-IBM typically involve two main types of interactions. The first type concerns the interaction between fluid dynamics and a prescribed solid structure kinematics. The second type pertains to the dynamic interaction of the solid structure in response to fluid forces. In the first category, the position, velocity, and acceleration of the structure are known a priori, resulting in a prescribed or forced motion. In contrast, the second category considers the motion of the solid structure to be free, induced by the fluid forces interacting with the solid boundary.

The primary goal of any FSI strategy is to ensure interface matching between the kinematic and dynamic properties of the fluid and structural domains. In the body conforming realm, the interface is inherently aligned with the meshes of both domains, facilitating a matched interface (as seen in ALE and overset methods). On the other hand, for body non-conformal methods like the IBM, a kinematic interface matching must be enforced in the

numerical method through forces applied at the interface. This approach introduces an additional kinematic constraint on the fluid–solid system, particularly for incompressible fluid flow, which already has the incompressibility constraint that should be enforced implicitly by the PPE.

Significant research has been conducted on fluid–structure coupling within the IBM framework, with numerous studies examining various methods of coupling through the partitioned strategy [54–57]. As a consequence, each domain is solved using its dedicated solver, providing modularity and flexibility that allows independent development of each solver. These methods typically employ either weakly coupled or strongly coupled algorithms. In a weakly coupled approach, the interface matching for both kinematic and dynamic conditions is not enforced accurately, while in a strongly coupled approach, these conditions are enforced simultaneously with high accuracy. Each algorithm offers trade-offs between computational simplicity and accuracy. The weakly coupled method favours efficiency but compromises interface matching accuracy and stability. In contrast, the strongly coupled method provides superior accuracy and stability but at a higher computational cost due to the implicit iterations involved. The different coupling algorithms described above, which fall under the partitioned approach, will be covered in depth in Chapter 6.

Several numerical challenges arise when addressing FSI problems within the IBM framework [93–95]. These issues are primarily related, to the added mass effect, termed internal mass effect in the IBM context. Instabilities and convergence problems are stemmed if the internal mass effect is not carefully handled. Uhlmann [39] proposed an explicit coupling strategy between the fluid's incompressible equations and the rigid body dynamics of the solid structure under the DF-IBM framework. However, the rigid body approximation, which assumes that the internal fluid inside the immersed boundary follows the rigid body motion, can cause stability issues. These problems are influenced by the ratio of solid to fluid density. For matching densities, a singularity arises, and for a density ratio roughly below a value of 1.2, the algorithm becomes unstable. Several studies have attempted to mitigate this singularity problem and reduce the critical density ratio value. An improved method was proposed in [41], which eliminates the singularity and lowers the critical density ratio to a value of 0.3. This approach has been widely adopted as in [78, 95].

All these methods are implemented within a weakly coupled framework. A comparison between weak and strong coupling algorithms for different FSI problems, presented in [93], revealed that the weak coupling fails when the solid body's mass falls below a certain threshold. Similarly, the strong coupling method faces the same issue, though a relaxation

scheme was proposed to address this, to the detriment of increased computational effort. An explicit treatment for the internal mass effect was introduced in [96] to surmount the density ratio limitations. Furthermore, a novel approach proposed in [97] resulted in a more stable algorithm, capable of handling a broader range of rotational Reynolds numbers compared to the explicit treatment in [96]. This improvement was particularly significant as it was observed that the rigid body approximation of the internal mass effect had a more pronounced impact on rotational dynamics than on translational motion. The various IME treatments will be thoroughly analysed in Chapter 6, while the impact of the IME on the fluid–rigid body coupling is fully highlighted in the numerical results demonstrated in Chapter 7.

FSI problems involving the DF-IBM continue to be an area of active development, with ongoing efforts to create algorithms that balance numerical stability, convergence, modularity, and computational efficiency. The internal mass effect plays a crucial role in achieving stability and convergence when its influence is significant. Implicit coupling algorithms show promise in this regard, though they require further refinement and optimisation to strike the right balance between computational efficiency and stability.

2.3 Summary

This chapter reviewed methodologies for simulating FSI problems, focusing on both body conformal and body non-conformal approaches. The body-conformal approach, illustrated by the ALE and overset methods, relies on computational meshes that align with the fluid–structure interface. While these methods allow for precise boundary representation, they face notable challenges.

In contrast, body non-conformal methods, such as the IBM, offer a more flexible and computationally efficient framework and have gained significant popularity over the past two decades. By decoupling the fluid and solid grids, non-conformal approaches eliminate the need for dynamic mesh updates, simplifying the simulation of moving boundaries. The IBM introduces a two-grid system comprising Eulerian and Lagrangian domains, facilitating the effective coupling of fluid and structural dynamics through interpolation and spreading operators. Numerous IBM variants have since been developed to address specific engineering and technological challenges. However, significant challenges remain, the most prominent being the difficulty of accurately resolving boundary layers in turbulent flows at high Reynolds numbers. This chapter highlighted several studies that aim to address these challenges.

Special attention was given to the DF-IBM due to its widespread adoption, stability, and ease of integration into fluid solvers. Variants such as the explicit DF-IBM, multi-

DF-IBM, and implicit IBM approaches; including the IBPM and MIBM, were discussed. Strategies to improve the reciprocity of interpolation and spreading operators were also reviewed. Challenges within the DF-IBM framework, such as spurious force oscillations and the diffused-interface of the immersed boundary, were addressed, along with proposed solutions to mitigate these issues.

The chapter concluded with an overview of fluid–structure coupling within the DF-IBM framework, encompassing both prescribed and fluid-induced (free) motions. These scenarios require kinematic and dynamic matching at the fluid–solid interface. Weak and strong coupling strategies were discussed, highlighting trade-offs between computational efficiency and accuracy. Particular emphasis was placed on the added mass effect, which is especially critical at low solid–fluid density ratios. The challenges posed by the added mass effect, including its impact on the stability and convergence of coupling algorithms, were examined. Additionally, various studies were highlighted that propose methods to mitigate singularities and overcome density ratio limitations. These advancements enhance stability for both rotational and translational dynamics, further strengthening the robustness of IBM for FSI simulations.

Chapter 3

Fluid–Solid Solver in OpenFOAM

Contents

The following chapter provides an in-depth overview of the current fluid–solid solver used inside the Open Field Operation And Manipulation (OpenFOAM) software package. The fluid governing equations for an isotropic Newtonian incompressible fluid are first derived and then discretised in space and time using the cell-centred finite volume approach used by OpenFOAM. To model the interaction of the fluid in the Eulerian domain with a solid immersed boundary in the Lagrangian domain, a boundary force term is incorporated as a source term within the fluid equations. In addition, details of the modifications required to handle the new pressure–velocity coupling via the Pressure Implicit with Splitting of Operators (PISO) solution algorithm are provided.

3.1 Fluid Governing Equations

The Eulerian fluid flow under the continuum hypothesis is governed by a set of conservation equations, namely the mass conservation equation also called the continuity equation, and the momentum conservation equations also called the Navier-Stokes momentum equations. The momentum conservation is derived from the second law of Newton for fluid motion. The internal (surface) and external (body/volume) forces acting on a fluid particle are summarised in Fig. (3.1). The fluid governing equations can be expressed as:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0, \tag{3.1a}$$

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u}) = \nabla \cdot \boldsymbol{\sigma} + \rho \mathbf{f}_{ext}, \tag{3.1b}$$

where ρ is the fluid density, $\mathbf{u}(\mathbf{x}_{i,j}, t)$ is the fluid Eulerian velocity vector, $\boldsymbol{\sigma}$ is the fluid Cauchy stress tensor (or total stress tensor), and $\mathbf{f}_{ext}$ represents any additional external body force.

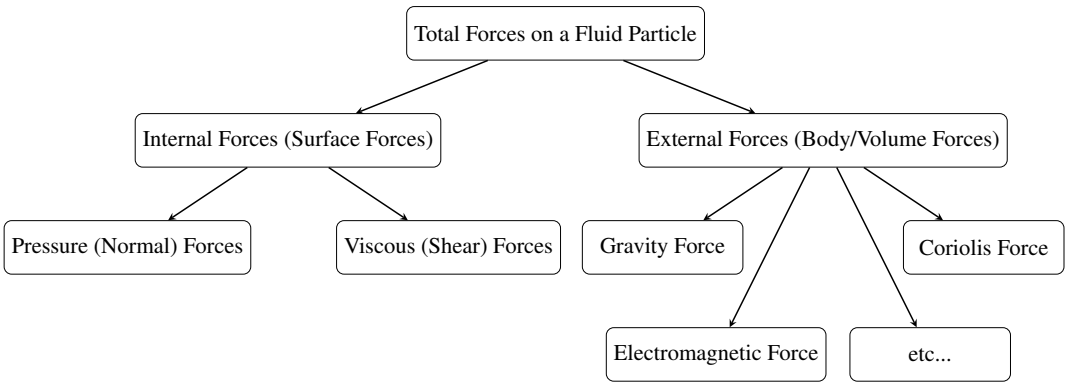


Figure 3.1: *Types of forces acting on a fluid particle.*

The total stress stress tensor can be expressed as the summation of both hydrostatic (or isotropic) stress tensor and viscous (or deviatoric) stress tensor:

$$\boldsymbol{\sigma} = -p_s \mathbf{I} + \tau, \tag{3.2}$$

where p_s is the static pressure, the negative sign indicates a compressive stress, $\mathbf{I}$ is the identity matrix, and τ is the viscous stress tensor also called shear-rate tensor, it accounts for the shear and normal stresses due to fluid motion and viscosity.

According to the Stokes hypothesis for Newtonian fluids, the shear-rate tensor τ is linearly

related to the strain-rate (or deformation-rate) tensor $\varepsilon = \frac{1}{2}\left(\nabla\mathbf{u} + (\nabla\mathbf{u})^{\mathrm{T}}\right)$, resulting in the following constitutive relation for an isotropic fluid:

$$\begin{aligned}\tau &= 2\mu\varepsilon - \frac{2}{3}\mu(\nabla\cdot\mathbf{u})\mathbf{I} \\ &= \mu\left(\nabla\mathbf{u} + (\nabla\mathbf{u})^{\mathrm{T}}\right) - \frac{2}{3}\mu(\nabla\cdot\mathbf{u})\mathbf{I},\end{aligned} \tag{3.3}$$

where μ is the fluid dynamic viscosity.

Replacing Eq. (3.3) in Eq. (3.2), yield:

$$\boldsymbol{\sigma} = -p_s\mathbf{I} + \mu\left(\nabla\mathbf{u} + (\nabla\mathbf{u})^{\mathrm{T}}\right) - \frac{2}{3}\mu(\nabla\cdot\mathbf{u})\mathbf{I}, \tag{3.4}$$

and taking the divergence of the total stress tensor found in Eq. (3.4) with the aid of tensor calculus identities, the following result is obtained:

$$\begin{aligned}\nabla\cdot\boldsymbol{\sigma} &= -\nabla p_s + \nabla\cdot\tau \\ &= -\nabla p_s + \nabla\cdot(\mu\nabla\mathbf{u}) + \nabla\Big(\mu(\nabla\cdot\mathbf{u})\Big) - \nabla\cdot\left(\frac{2}{3}\mu(\nabla\cdot\mathbf{u})\mathbf{I}\right).\end{aligned} \tag{3.5}$$

Next, assuming that the fluid is incompressible of constant density ρ, the mass conversation Eq. (3.1a) reduces to the form:

$$\nabla\cdot\mathbf{u} = 0. \tag{3.6}$$

This condition leads to the first fluid–solid system constraint, the divergence-free velocity field inside the fluid Eulerian domain.

Finally, using the incompressibility constraint defined by Eq. (3.6) on Eq. (3.5), and replacing inside the momentum conservation Eq. (3.1b), the dimensional incompressible fluid Navier-Stokes momentum equations in the conservative form with no external forces is obtained:

$$\frac{\partial\mathbf{u}}{\partial t} + \nabla\cdot(\mathbf{u}\otimes\mathbf{u}) - \nabla\cdot(\nu\nabla\mathbf{u}) = -\nabla p, \tag{3.7}$$

where $\nu = \dfrac{\mu}{\rho}$ is the effective kinematic viscosity and $p = \dfrac{p_s}{\rho}$ is the kinematic pressure (as referred to in OpenFOAM). Since laminar flow problems are being considered, the effective viscosity is simply the laminar molecular kinematic viscosity ν, which is a fluid property. The tensor product symbol $\otimes$ will be dropped for the remainder of this thesis, for convenience.

To account for the presence of the solid immersed boundary inside the fluid domain, the

boundary force **f** is added as a source term to the Navier-Stokes momentum Eq. (3.7), yielding the following fluid governing equations that will be used for the remainder of this thesis:

$$\nabla \cdot \mathbf{u} = 0, \tag{3.8a}$$

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{u}\mathbf{u}) - \nabla \cdot (\nu \nabla \mathbf{u}) = -\nabla p + \mathbf{f}, \tag{3.8b}$$

the respective unit of every term in Eq. (3.8b) is force per unit volume per unit density $\frac{m}{s^2}$. The first term of Eq. (3.8b) is the unsteady transient term, the second term is the convective term, the third term is the viscous diffusion term, and the last term is the pressure gradient term. Eq. (3.8b) is a second-order nonlinear partial differential equation (PDE), as the diffusion term includes a second-order derivative of the velocity field **u** in space and the nonlinearity is present in the convection term where the velocity is being transported by itself. For good discretisation accuracy, in this research, a second-order accuracy in space and time is aimed for and will be described in detail in the following. To avoid this problem of nonlinearity, the convection term is linearised in OpenFOAM [98]. The linearisation will be explained in detail in what follows.

From here on, all cell-centred values will have the following subscript $(.)_P$ or $(.)_N$ depending on whether the owner or the neighbour control volume is being used, while face-centred values will have the subscript $(.)_f$.

3.2 Cell-Centred Finite Volume Method (FVM) Discretisation

The fluid solver is described in the OpenFOAM environment. OpenFOAM is based on the collocated grid arrangement, therefore fluid dynamics quantities (pressure, velocity, etc.) are stored at cell-centres. However, OpenFOAM uses the FVM for the discretisation, therefore some quantities (such as the velocity) need to be obtained at other locations in space to obtain the flux-based terms. Hence interpolation is performed for the velocities from cell-centres to cell face-centres to obtain the flux-based velocity field, where the calculations of the latter are managed in a staggered fashion. Once a divergence-free face volumetric (incompressible) flux field is reached, it is reconstructed to get an approximate divergence-free velocity field at the cell-centre.

To avoid the checkerboard (even-odd) pressure oscillation problem on a collocated grid, OpenFOAM implements the Rhie-Chow momentum interpolation (or correction) method.

However, this implementation is implicit and somewhat obscured in the code, potentially deviating from the original theoretical formulation described by Rhie and Chow [99], as will be discussed later. In simple terms, the Rhie-Chow interpolation introduces a correction term to the velocity components that take into account the pressure gradients. The correction term is added to the velocity components at the cell faces and is based on a linear interpolation of the pressure values from adjacent neighbouring cells. This approach ensures that the pressure field is smoothed out and that the pressure gradients are properly accounted for in the momentum equations.

An essential theorem in the FVM is the mean value theorem for any transported quantity Ψ over a control volume centred in P, with the centroid value Ψ_P equal to the mean value of Ψ inside the control volume and stored at its centre, such that:

$$\int_{V_P} \Psi(\mathbf{x})\, dV = \Psi_P V_P, \tag{3.9}$$

where V_P is the volume of the cell-centred in P, assuming that V_P does not change in time.

Another important theorem is the Gauss theorem, also known as the divergence theorem, which plays a critical role in the discretisation procedure within OpenFOAM, enabling the reduction of volume integrals to surface integrals. As mentioned earlier, this integral transformation requires interpolation to convert cell-centred, volume-based quantities into flux-based quantities at cell face-centres. This step is necessary because flux information at cell faces is initially unavailable; only volume-based variables are computed at the control volume centroids due to OpenFOAM's collocated grid arrangement. Consequently, interpolation becomes essential for estimating flux quantities at the cell faces.

To address this, the relevant quantity are assumed to vary linearly over each face f of the control volume centred at P. This assumption allows us to approximate the value at the face centroid, using its mean value as a representative flux across the face. Multiplying the face-centred value by the face area corresponds to applying the midpoint rule approximation to the surface integral, yielding a second-order accurate result. This approximation is illustrated by the following transformations, where a control volume is known to be bounded by a series of flat faces, the Gauss theorem first transforms a volume into a surface integral and then the midpoint rule transforms the surface integral into a discrete sum over all faces of the control volume V_P:

$$\int_{V_P} \nabla \cdot \Psi \, dV = \oint_{\partial V_P} \Psi \cdot d\mathbf{S} = \sum_f \Psi_f \cdot \mathbf{S}_f, \tag{3.10}$$

where Ψ is taken here as a vector field.

$$\int_{V_P} \nabla\Psi \, dV = \oint_{\partial V_P} \Psi \, d\mathbf{S} = \sum_f \Psi_f \mathbf{S}_f, \tag{3.11}$$

where Ψ can be either a scalar or a vector field.

The volume integral $\int_{V_P}$ is computed over the cell-centred in P with the volume equal to V_P, whereas the surface integral $\oint_{\partial V_P}$ is computed over the closed boundary (or surface) of the cell-centred in P with a surface area equal to ∂V_P. The infinitesimal surface element dS over ∂V_P is oriented by an outward-pointing unit normal vector $\mathbf{n}$ to the surface, hence $d\mathbf{S} = dS\mathbf{n}$.

Now combining Eqs. (3.9)–(3.11), yields:

$$\nabla \cdot \Psi_P = \frac{1}{V_P} \sum_f \Psi_f \cdot \mathbf{S}_f, \tag{3.12}$$

where Ψ is taken here as a vector field.

$$\nabla \Psi_P = \frac{1}{V_P} \sum_f \Psi_f \mathbf{S}_f, \tag{3.13}$$

where Ψ can be either a scalar or a vector field, $\mathbf{S}_f$ is the outward-pointing face area normal vector between the current control volume P and its neighbouring control volume N. It is equal to $\mathbf{S}_f = A_f \mathbf{n}_f$, with A_f equal to the surface area of the face f and $\mathbf{n}_f$ is the outward-pointing unit normal vector to the face f.

3.2.1 Space Discretisation

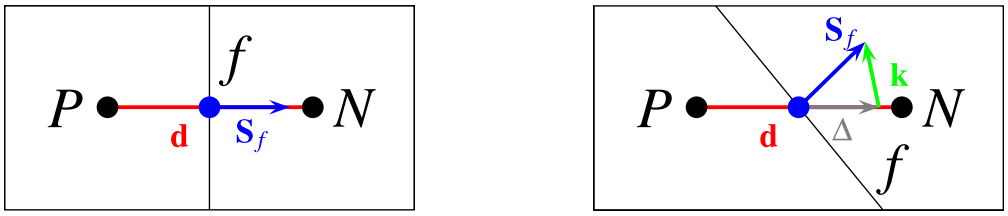
Each term of the Navier-Stokes Eq. (3.8b) discretised in space is treated separately in what follows, with the face-centre f values of all explicit quantities computed from their owner P and neighbouring N cell-centres values, as shown in Fig. (3.2). This computation employs a second-order accurate linear interpolation, also called central differencing (CD) interpolation scheme and implemented in OpenFOAM under the name `linear` scheme. The interpolation is applied whenever a face-centre value is required in the discretisation procedure:

$$\phi_f^n = \overline{(\phi_P^n)}_f = f_w \phi_P^n + (1 - f_w)\, \phi_N^n. \tag{3.14}$$

The interpolation weighting factor f_w, is defined as the ratio of the distances fN and PN:

$$f_w = \frac{fN}{PN} = \frac{|\mathbf{x}_f - \mathbf{x}_N|}{|\mathbf{d}|}, \tag{3.15}$$

where $\mathbf{x}_f$ and $\mathbf{x}_N$ are the location of the face-centre f and neighbouring cell centroid N respectively, and $|\mathbf{d}|$ is the magnitude of the vector connecting P and N, i.e., the distance between P and N.



(a) Orthogonal cell face *(b) Non-orthogonal cell face*

Figure 3.2: *Schematic of the face area normal vector* $\mathbf{S}_f$ *on different meshes.*

For Cartesian uniform meshes where $\Delta x = \Delta y = h = |\mathbf{d}|$ as considered here, a special case arises since the face f is situated midway between the owner P and neighbouring N control volumes V_P and V_N. The approximation reduces to an arithmetic average:

$$f_w = \frac{fN}{PN} = \frac{d/2}{d} = \frac{h/2}{h} = \frac{1}{2}, \tag{3.16}$$

yielding,

$$\phi_f^n = \overline{(\phi_P^n)}_f = \frac{\phi_P^n + \phi_N^n}{2}. \tag{3.17}$$

3.2.1.1 Convection Term

The convective term in Eq. (3.8b) is discretised using Eq. (3.12) as follows:

$$\nabla \cdot (\mathbf{u}_P \mathbf{u}_P) = \frac{1}{V_P} \sum_f (\mathbf{uu})_f \cdot \mathbf{S}_f. \tag{3.18}$$

The linearisation of the convection term implies that an existing velocity flux field ϕ_f^n that satisfies the continuity equation (or the divergence-free condition) will be used to calculate the convection term:

$$\sum_f (\mathbf{uu})_f \cdot \mathbf{S}_f = \sum_f \left(\mathbf{u}_f^n \cdot \mathbf{S}_f\right) \mathbf{u}_f^{n+1} = \sum_f \phi_f^n \mathbf{u}_f^{n+1}, \tag{3.19}$$

where $\phi_f^n = \mathbf{u}_f^n \cdot \mathbf{S}_f$ represents the face volumetric flux across the cell face. The flux ϕ_f^n depends on the face value of $\mathbf{u}_P^n$, with the condition that the velocity field from which the

fluxes are derived must be such that the continuity equation is conserved:

$$\int_{V_P} \nabla \cdot \mathbf{u}_P^n \, dV = \sum_f \mathbf{u}_f^n \cdot \mathbf{S}_f = \sum_f \phi_f^n = 0. \tag{3.20}$$

The idea is that the flux will be calculated using known velocities $\mathbf{u}_f^n$ and thus the convection term will only contain an explicit flux variable ϕ_f^n and one implicit term $\mathbf{u}_f^{n+1}$, the velocity being solved. This treatment linearises the convection term.

The implicit face variable $\mathbf{u}_f^{n+1}$, is calculated from the values at the cell-centres from the neighbouring control volumes (i.e., $\mathbf{u}_P^{n+1}$, $\mathbf{u}_N^{n+1}$, $\mathbf{u}_S^{n+1}$, ...), which is obtained using the convection differencing scheme also called the convection interpolation scheme. Their role is to determine the value of the transported quantity on the face f of the control volume V_P from the values at the cell-centres. Different interpolation schemes exist in OpenFOAM, ranging from centred schemes, upwind-biased convection schemes, and Normalised Variable Diagram (NVD) / Total Variation Diminishing (TVD) schemes to construct the implicit face term.

The convection interpolation scheme used is one of the most popular schemes in the OpenFOAM library, named the Gamma differencing scheme [100], and implemented under the name `Gamma` scheme. This scheme blends the first-order unconditionally bounded upwind differencing (UD) interpolation scheme, implemented in OpenFOAM as the `upwind` scheme, and expressed as:

$$\mathbf{u}_f^{n+1} = \begin{cases} \mathbf{u}_P^{n+1}, & \text{for } \phi_f^n \geq 0, \\ \mathbf{u}_N^{n+1}, & \text{for } \phi_f^n < 0, \end{cases} \tag{3.21}$$

with the second-order accurate CD interpolation scheme, equivalent to the `linear` scheme shown in Eq. (3.14), and written as:

$$\mathbf{u}_f^{n+1} = \overline{\left(\mathbf{u}_P^{n+1}\right)}_f = f_w \mathbf{u}_P^{n+1} + (1 - f_w)\, \mathbf{u}_N^{n+1}, \tag{3.22}$$

in order to satisfy the Convection Boundedness Criterion (CBC), which corresponds to the region on an NVD diagram with bounded behavior [100]. The scheme works as follows in terms of unnormalised variables:

$$\phi_f = \begin{cases} \big(1 - \gamma(1 - f_w)\big)\phi_P + \gamma(1 - f_w)\, \phi_N, & \text{if } \ 0 < \tilde{\phi}_P < \beta_m, \\ f_w \phi_P + (1 - f_w)\, \phi_N, & \text{if } \ \beta_m \leq \tilde{\phi}_P < 1, \\ \phi_P, & \text{if } \ \tilde{\phi}_P \leq 0 \text{ or } \ \tilde{\phi}_P \geq 1, \end{cases} \tag{3.23}$$

with $\gamma = \dfrac{\tilde{\phi}_P}{\beta_m}$ being the blending factor and β_m a user-specified factor implemented in OpenFOAM under a scalar variable k $(= 2\beta_m)$ with $0 \leq k \leq 1$ $(0 \leq \beta_m \leq 0.5)$. The variable k regulates the transition between the unconditionally bounded `upwind` scheme and the bounded region of the `linear` scheme in OpenFOAM. For $k = 0$ $(\beta_m = 0)$, an abrupt change is done from UD to the bounded region of the CD scheme as shown in [100]. for $\tilde{\phi}_P \leq 0$ or $\tilde{\phi}_P \geq 1$ the Gamma differencing scheme remains as the UD scheme whereas for $\tilde{\phi}_P > 0$ the scheme switches to the CD scheme; this sort of switching can cause instability in the calculation, often preventing convergence. For good resolution, it is suggested that the value of k should ideally be restricted such as $0.2 \leq k \leq 1$ $(0.1 \leq \beta_m \leq 0.5)$.

The normalised variable $\tilde{\phi}_P$ is defined as:

$$\tilde{\phi}_P = 1 - \frac{\phi_N - \phi_P}{2\,(\nabla\phi)_P \cdot \mathbf{d}}, \tag{3.24}$$

where the term $(\nabla\phi)_P$ is approximated using the `linear` interpolation scheme.

For Cartesian uniform meshes as in this case, the `Gamma` scheme reduces to the following:

$$\phi_f = \begin{cases} \left(1 - \dfrac{\gamma}{2}\right)\phi_P + \dfrac{\gamma}{2}\phi_N, & \text{if } \ 0 < \tilde{\phi}_P < \beta_m, \\ \dfrac{\phi_P + \phi_N}{2}, & \text{if } \ \beta_m \leq \tilde{\phi}_P < 1, \\ \phi_P, & \text{if } \ \tilde{\phi}_P \leq 0 \text{ or } \ \tilde{\phi}_P \geq 1. \end{cases} \tag{3.25}$$

3.2.1.2 Diffusion Term

The diffusion term in Eq. (3.8b) is discretised using Eq. (3.12) as follows:

$$\nabla \cdot (\nu \nabla \mathbf{u}_P) = \frac{1}{V_P} \sum_f \nu_f \, (\nabla \mathbf{u})_f \cdot \mathbf{S}_f. \tag{3.26}$$

Due to mesh uniformity and constant molecular viscosity, the face viscosity ν_f is the same as the laminar molecular kinematic viscosity ν defined earlier. The term $(\nabla \mathbf{u})_f \cdot \mathbf{S}_f$ is calculated in OpenFOAM as:

$$(\nabla \mathbf{u})_f \cdot \mathbf{S}_f = \underbrace{(\nabla \mathbf{u})_f^{n+1} \cdot \boldsymbol{\Delta}}_{\substack{\text{implicit orthogonal} \\ \text{contribution}}} + \underbrace{(\nabla \mathbf{u})_f^{n} \cdot \mathbf{k}}_{\substack{\text{explicit non-orthogonal} \\ \text{correction}}}, \tag{3.27}$$

where the two vectors $\boldsymbol{\Delta}$ and $\mathbf{k}$ introduced, have to satisfy the following summation as shown

in Fig. (3.2b):

$$\mathbf{S}_f = \boldsymbol{\Delta} + \mathbf{k}, \tag{3.28}$$

with the vector $\boldsymbol{\Delta}$ being parallel to $\mathbf{d}$. This approach enables the application of a second-order method to the orthogonal contribution, restricting the less accurate method solely to the non-orthogonal component, which cannot be addressed otherwise. For Cartesian uniform meshes as in the present case, the mesh is completely and purely orthogonal, thus no non-orthogonal corrections are present, so $\mathbf{k} = 0$, leaving $\mathbf{S}_f = \boldsymbol{\Delta}$ only.

The face gradient term $(\nabla \mathbf{u})_f \cdot \mathbf{S}_f$ can be calculated for a purely orthogonal mesh from the two values of the owner cell P and neighbouring cell N around the face f as:

$$(\nabla \mathbf{u})_f^{n+1} \cdot \mathbf{S}_f = \frac{\mathbf{u}_N^{n+1} - \mathbf{u}_P^{n+1}}{|\mathbf{d}|} |\mathbf{S}_f|, \tag{3.29}$$

with $|\mathbf{d}| = h$ for Cartesian uniform meshes, and $|\mathbf{S}_f| = A_f$, being the magnitude of the face area normal vector, where for Cartesian uniform meshes $A_f = h$ for 2D topology, the face gradient is simply written in 2D as:

$$(\nabla \mathbf{u})_f^{n+1} \cdot \mathbf{S}_f = \left(\mathbf{u}_N^{n+1} - \mathbf{u}_P^{n+1}\right). \tag{3.30}$$

The above equation states that the face gradient is computed by using a CD approximation of the first-order derivative in the direction of the vector $\mathbf{d}$. This method is second-order accurate, but can only be used on orthogonal meshes.

3.2.1.3 Pressure Gradient Term

The cell-centred pressure gradient term present in the Navier-Stokes Eq. (3.8b), will be initially treated explicitly $(\nabla p)_P^n$ in the predictor step of the PISO solution algorithm. In the spirit of the Rhie-Chow procedure [99], the pressure gradient will remain firstly undiscretised to produce the linear algebraic equation resulting from the discretised form of Eq. (3.8b).

Once the linear algebraic equation that represents the finite volume discretised form of the Navier-Stokes Eq. (3.8b) is established, the cell-centred pressure gradient will be computed with the help of the Gauss theorem Eq. (3.13), with the face values obtained from linear interpolation as in Eq. (3.14):

$$(\nabla p)_P^n = \frac{1}{V_P} \sum_f p_f^n \mathbf{S}_f = \frac{1}{V_P} \sum_f \overline{(p_P^n)}_f \mathbf{S}_f = \frac{1}{V_P} \sum_f \left(f_w p_P^n + (1 - f_w)\, p_N^n\right) \mathbf{S}_f. \tag{3.31}$$

For Cartesian uniform meshes as considered here, the cell-centred pressure gradient is expressed as:

$$(\nabla p)_P^n = \frac{1}{V_P} \sum_f \left(\frac{p_P^n + p_N^n}{2} \right) \mathbf{S}_f. \tag{3.32}$$

Another pressure gradient term will appear in the pressure Poisson equation (PPE) in the corrector step of the PISO solution algorithm, that will be discussed later in this chapter. This term is treated implicitly and is similar to the diffusion term shown in Eq. (3.29). It is called the face normal pressure gradient and has the following form:

$$(\nabla p)_f^{n+1} \cdot \mathbf{S}_f = \frac{p_N^{n+1} - p_P^{n+1}}{|\mathbf{d}|} |\mathbf{S}_f|. \tag{3.33}$$

For Cartesian uniform meshes as in this case, the face normal pressure gradient is expressed in 2D as:

$$(\nabla p)_f^{n+1} \cdot \mathbf{S}_f = \left(p_N^{n+1} - p_P^{n+1} \right). \tag{3.34}$$

3.2.1.4 Boundary Force Term

The cell-centred boundary force term in Eq. (3.8b) is treated implicitly $\mathbf{f}_P^{n+1}$ and computed using the current implicit DF-IBM algorithm to enforce the second fluid–solid system constraint on the solid Lagrangian domain; this will be discussed in depth in Chapter 4.

Other face values of the boundary force term appear inside the PPE in the corrector step of the PISO algorithm. These face values are treated using the linear interpolation Eq. (3.14). For Cartesian uniform meshes, this reduces to:

$$\mathbf{f}_f^{n+1} = \overline{\left(\mathbf{f}_P^{n+1}\right)}_f = \frac{\mathbf{f}_P^{n+1} + \mathbf{f}_N^{n+1}}{2}. \tag{3.35}$$

3.2.2 Time Discretisation - Unsteady Term

For the solution's overall (space and time) accuracy to be second-order, all the individual terms in space and time have to be second-order. The temporal term in Eq. (3.8b) is therefore chosen to be discretised using a fully implicit three-level second-order scheme, also called BDF2 scheme (Backward Differentiation Formula of order 2) in the literature and implemented in OpenFOAM under the `backward` scheme. It is described through the following equation:

$$\left(\frac{\partial \mathbf{u}_P}{\partial t} \right)^{n+1} = \frac{\frac{3}{2}\mathbf{u}_P^{n+1} - 2\mathbf{u}_P^n + \frac{1}{2}\mathbf{u}_P^{n-1}}{\Delta t}. \tag{3.36}$$

For the first time-step, since only one old time level is available ($\mathbf{u}_P^n$), then the first-order implicit backward Euler is used as the temporal discretisation scheme:

$$\left(\frac{\partial \mathbf{u}_P}{\partial t}\right)^{t=\Delta t} = \frac{\mathbf{u}_P^{n+1} - \mathbf{u}_P^n}{\Delta t}. \tag{3.37}$$

After that, it switches back to the chosen BDF2 scheme.

3.3 Linear Algebraic Discretised Fluid Equation

Summing up all discretised terms of Eq. (3.8b) with their respective explicit/implicit treatment form the algebraic equation, also called the discrete momentum equation, where the implicit pressure gradient term is kept undiscretised as explained earlier:

$$\left(\frac{\partial \mathbf{u}_P}{\partial t}\right)^{n+1} + \frac{1}{V_P}\sum_f \phi_f^n \mathbf{u}_f^{n+1} - \frac{1}{V_P}\sum_f \nu \left(\nabla \mathbf{u}\right)_f^{n+1} \cdot \mathbf{S}_f = -\left(\nabla p\right)_P^{n+1} + \mathbf{f}_P^{n+1}. \tag{3.38}$$

Now, transforming face-centred values to cell-centred values by replacing the convection and diffusion terms in Eq. (3.38) with their correct discretisation schemes, for simplicity, in the convection term, the interpolation scheme for the implicit part was chosen for $0 < \tilde{\phi}_P < \beta_m$ as shown in Eq. (3.25):

$$\begin{aligned}\frac{\frac{3}{2}\mathbf{u}_P^{n+1} - 2\mathbf{u}_P^n + \frac{1}{2}\mathbf{u}_P^{n-1}}{\Delta t} + \frac{1}{V_P}\sum_f \phi_f^n \left(\left(1-\frac{\gamma}{2}\right)\mathbf{u}_P^{n+1} + \frac{\gamma}{2}\mathbf{u}_N^{n+1}\right) \\ - \frac{1}{V_P}\sum_f \nu\left(\mathbf{u}_N^{n+1} - \mathbf{u}_P^{n+1}\right) = -\left(\nabla p\right)_P^{n+1} + \mathbf{f}_P^{n+1}.\end{aligned} \tag{3.39}$$

Rearranging the above equation, the simplified linear algebraic equation is obtained:

$$a_P \mathbf{u}_P^{n+1} + \sum_N a_N \mathbf{u}_N^{n+1} = \mathbf{RHS}\left(\mathbf{u}_P^n, \mathbf{u}_P^{n-1}\right) - \left(\nabla p\right)_P^{n+1} + \mathbf{f}_P^{n+1}, \tag{3.40}$$

where

$$a_P = \frac{3}{2\Delta t} + \frac{1}{V_P}\sum_f \phi_f^n \left(1 - \frac{\gamma}{2}\right) + \frac{1}{V_P}\sum_f \nu, \tag{3.41a}$$

$$a_N = \frac{1}{V_P}\phi_f^n \frac{\gamma}{2} - \frac{1}{V_P}\nu, \tag{3.41b}$$

$$\mathbf{RHS}\left(\mathbf{u}_P^n, \mathbf{u}_P^{n-1}\right) = \frac{2}{\Delta t}\mathbf{u}_P^n - \frac{1}{2\Delta t}\mathbf{u}_P^{n-1}. \tag{3.41c}$$

3.4 Pressure–Velocity ($p-\mathbf{U}$) Coupling

In the unsteady incompressible Navier-Stokes equations, the pressure and velocity are dependent flow variables. They are coupled together through the continuity equation. As a result, solving the Navier-Stokes equations requires simultaneously determining both the pressure and velocity fields. The role of the pressure is to make sure the velocity is divergence-free.

There are two main families of methods for solving this coupled system of equations: fully coupled methods and segregated methods. The former is preferred in the computations of compressible flows while the latter has been widely employed in incompressible flow computations. In the segregated approach (also called decoupled approach), the pressure and the velocity fields are decoupled numerically and then solved separately or consecutively, unlike a coupled solution technique.

The segregated approach split the equations into two parts: one for the velocity and one for the pressure. The velocity equation is solved first, then the pressure equation is solved using the velocity solution. There are several segregated pressure–velocity decoupling algorithms including non-iterative methods such as the MAC method [101] being an explicit transient algorithm. Implicit algorithms for unsteady flows can be found in the literature, namely Chorin first-order projection method (also called fractional step method, time splitting method) [102], as well as improved variants of the projection method with second-order accuracy [103–107], etc. Other implicit solution algorithm methods exist in the literature, such as the SIMPLE-type methods, including SIMPLE [108], SIMPLER [109], SIMPLEC [110], PISO [111], and PIMPLE (used in OpenFOAM environment) which combines the PISO and SIMPLE (SIMPLEC) algorithms. They are all based on the fundamental idea of using a predictor-corrector procedure and are classified as iterative methods since a converged solution is sought, except PISO, which falls in the non-iterative category.

The PISO solution algorithm is adopted in the present work for pressure–velocity decoupling. This algorithm can be found in OpenFOAM framework, under the name `pisoFoam`, therefore the built-in solver is modified to incorporate the solid immersed boundary using DF-IBM.

3.4.1 PISO Solution Algorithm

The PISO algorithm short for Pressure Implicit with Splitting of Operators algorithm, belongs to the group of segregated solution procedures, due to its non-iterative nature, the algorithm splits the operators into one implicit momentum predictor step at the beginning of the time-step using a finite volume discretisation of the Navier-Stokes by treating an estimated pressure

field as known, followed by multiple explicit momentum corrector steps, usually 2 to 3 [111]. This approach enforces a divergence-free velocity field by the solution of the pressure equation using the projection method. The idea behind multiple velocity correctors is to fully resolve the pressure–velocity coupling in a single time-step.

The implicit predictor step is represented by Eq. (3.40) without the inclusion of the boundary force term, and with the pressure solution from the previous time level. The equation for the predicted velocity field is re-written in the following form:

$$a_P \mathbf{u}_P^* + \sum_N a_N \mathbf{u}_N^* = \mathbf{RHS}\left(\mathbf{u}_P^n, \mathbf{u}_P^{n-1}\right) - (\nabla p)_P^n, \tag{3.42}$$

where $\mathbf{u}^*$ is the predicted velocity field and the explicit pressure gradient term is treated using Eq. (3.32).

The multiple explicit corrector steps are derived here. Going back to Eq. (3.40), the neighbouring velocities term is taken to the right-hand side of the equation, and the pressure gradient term is treated now implicitly as an unknown:

$$a_P \mathbf{u}_P^{n+1} = \mathbf{H}\left(\mathbf{u}_N^{n+1}, \mathbf{u}_P^n, \mathbf{u}_P^{n-1}\right) - (\nabla p)_P^{n+1} + \mathbf{f}_P^{n+1}, \tag{3.43}$$

where the linear operator $\mathbf{H}$ is described by:

$$\mathbf{H}\left(\mathbf{u}_N^{n+1}, \mathbf{u}_P^n, \mathbf{u}_P^{n-1}\right) = -\sum_N a_N \mathbf{u}_N^{n+1} + \mathbf{RHS}\left(\mathbf{u}_P^n, \mathbf{u}_P^{n-1}\right). \tag{3.44}$$

The implicit velocity in the $\mathbf{H}$ operator is replaced by the predicted velocity field $\mathbf{u}^*$ to produce the explicit behavior of the corrector step. From this formulation, a new cell-centred velocity can be defined:

$$\mathbf{u}_P^{n+1} = \frac{\mathbf{H}\left(\mathbf{u}_N^*, \mathbf{u}_P^n, \mathbf{u}_P^{n-1}\right)}{a_P} - \frac{1}{a_P}(\nabla p)_P^{n+1} + \frac{1}{a_P}\mathbf{f}_P^{n+1}. \tag{3.45}$$

The cell-face velocity $\mathbf{u}_f^{n+1}$ needed in the discretised continuity equation can be expressed by linearly interpolating the cell-centre values of $\mathbf{u}_P^{n+1}$ to face-centre values using Eq. (3.14):

$$\mathbf{u}_f^{n+1} = \overline{\left(\mathbf{u}_P^{n+1}\right)}_f = \left(\frac{\mathbf{H}\left(\mathbf{u}_N^*, \mathbf{u}_P^n, \mathbf{u}_P^{n-1}\right)}{a_P}\right)_f - \left(\frac{1}{a_P}\right)_f (\nabla p)_f^{n+1} + \left(\frac{1}{a_P}\right)_f \mathbf{f}_f^{n+1}. \tag{3.46}$$

Taking the divergence of Eq. (3.45), and enforcing the divergence-free condition on the

new time level results in the continuity equation:

$$\nabla \cdot \mathbf{u}_P^{n+1} = \sum_f \mathbf{u}_f^{n+1} \cdot \mathbf{S}_f = \sum_f \phi_f^{n+1} = 0, \tag{3.47}$$

where $\phi_f^{n+1} = \mathbf{u}_f^{n+1} \cdot \mathbf{S}_f$ is the divergence-free volumetric flux across the cell-face f.

Now to obtain the discrete pressure equation also known as the PPE, the previous equation for the velocity Eq. (3.45) is substituted in the continuity Eq. (3.47):

$$\nabla \cdot \left(\frac{1}{a_P} (\nabla p)_P^{n+1} \right) = \nabla \cdot \left(\frac{\mathbf{H}\left(\mathbf{u}_N^*, \mathbf{u}_P^n, \mathbf{u}_P^{n-1}\right)}{a_P} \right) + \nabla \cdot \left(\frac{1}{a_P} \mathbf{f}_P^{n+1} \right), \tag{3.48a}$$

and for the face values format,

$$\sum_f \left(\left(\frac{1}{a_P} \right)_f (\nabla p)_f^{n+1} \right) \cdot \mathbf{S}_f = \sum_f \left(\frac{\mathbf{H}\left(\mathbf{u}_N^*, \mathbf{u}_P^n, \mathbf{u}_P^{n-1}\right)}{a_P} \right)_f \cdot \mathbf{S}_f + \sum_f \left(\left(\frac{1}{a_P} \right)_f \mathbf{f}_f^{n+1} \right)_f \cdot \mathbf{S}_f, \tag{3.48b}$$

the face normal pressure gradient is treated here using Eq. (3.34).

Solving the modified PPE (mPPE) expressed above by Eq. (3.48b) that differs from the standard PPE by the addition of the last term related to the boundary force, the solution of the pressure equation p^{n+1} is obtained. It contains both the physical pressure field, which originates from the flow field, as well as the correction part, which is responsible for enforcing the divergence-free condition presented here as the first fluid–solid system constraint, and compensating for the error in the initial pressure field.

The face volumetric flux of the face velocity is corrected using the new pressure solution and the boundary force, knowing that the mPPE is derived from the divergence-free condition, the new cell-face volumetric flux also satisfies the divergence-free condition:

$$\phi_f^{n+1} = \mathbf{u}_f^{n+1} \cdot \mathbf{S}_f = \left(\left(\frac{\mathbf{H}\left(\mathbf{u}_N^*, \mathbf{u}_P^n, \mathbf{u}_P^{n-1}\right)}{a_P} \right)_f - \left(\frac{1}{a_P} \right)_f (\nabla p)_f^{n+1} + \left(\frac{1}{a_P} \right)_f \mathbf{f}_f^{n+1} \right) \cdot \mathbf{S}_f, \tag{3.49}$$

as for the Rhie-Chow correction, instead of introducing an artificial pressure term as proposed in [99], the same is achieved here by introducing the face values of the $\mathbf{H}$ operator by linear interpolation using Eq. (3.14).

Finally, the cell-centred velocity is corrected explicitly to satisfy the divergence-free con-

dition as well:

$$\mathbf{u}_P^{n+1} = \frac{\mathbf{H}\left(\mathbf{u}_N^*, \mathbf{u}_P^n, \mathbf{u}_P^{n-1}\right)}{a_P} - \frac{1}{a_P}\left(\nabla p\right)_P^{n+1} + \frac{1}{a_P}\mathbf{f}_P^{n+1}. \tag{3.50}$$

3.5 System of Linear Algebraic Equations Solvers

After discretisation of the Navier-Stokes momentum Eq. (3.42) and the mPPE Eq. (3.48a), a linear algebraic equation similar to Eq. (3.40) is assembled for every control volume in the fluid Eulerian domain, where a system of linear algebraic equations is built, representing the whole discrete fluid domain and expressed in the matrix form as:

$$[A][\Psi] = [R], \tag{3.51}$$

where $[A]$ is a sparse matrix called the coefficient matrix resulting from the discretisation. It is split into diagonal terms a_P for each computational cell-centre P, and non-diagonal terms $a_N \sum_N \Psi_N$ for its surrounding neighbouring cells N:

$$[A][\Psi] = [A_P][\Psi_P] + [A_N^\Psi] = a_P\Psi_P + \sum_N a_N\Psi_N. \tag{3.52}$$

The vector $[\Psi]$ represents the cell-centred unknown variables Ψ_P at time level $n+1$, which can be pressure or velocity, depending on which system is being solved. The vector $[R]$ contains the source terms and their corresponding coefficients, i.e., the known variables at time level n.

OpenFOAM possesses a big library of iterative solvers, with several solvers that belong to the family of Krylov subspace methods such as the Conjugate Gradient (CG) for symmetric matrices, Bi-Conjugate Gradient (BiCG) for non-symmetric matrices, and another efficient variant compared to the previously mentioned solvers for both symmetric and non-symmetric matrices called the Bi-Conjugate Gradient Stabilised (BiCGStab) method. Another family of iterative solvers, is the Multi-Grid (MG) solvers, which have the Geometric Agglomerated Algebraic Multi-Grid (GAMG) variant implemented inside OpenFOAM.

The convergence rate of the above mentioned solvers can be improved when the matrix is preconditioned, i.e., by decreasing the condition number through matrix preconditioning. OpenFOAM implements several preconditioners, such as the Diagonal-based Incomplete LU factorisation (DILU) preconditioner for non-symmetric matrices, the Diagonal-based Incomplete Cholesky preconditioner (DIC) for symmetric matrices or a Faster version of

the DIC preconditioner (FDIC), and the diagonal (or Jacobi) preconditioner compatible with any matrix type. The GAMG solver mentioned earlier can also serve as a preconditioner, and it has been noted in [81] that, for transient problems, it can achieve higher efficiency and better performance when applied as a preconditioner for a Krylov subspace method.

The GAMG method, when used either as a solver or preconditioner, requires a smoother algorithm to remove high-frequency errors through the multi-grid levels. The Gauss-Seidel smoother can be found in OpenFOAM for both symmetric and non-symmetric matrices, whereas a symmetrical variant called sym-Gauss-Seidel also exists, or another symmetrical variant that combines DIC and Gauss-Seidel, and another DILU-Gauss-Seidel blended variant for non-symmetrical matrices.

The non-exhaustive list of OpenFOAM iterative solvers with their corresponding preconditioners and smoothers are listed in Table 3.1.

Table 3.1: *OpenFOAM iterative solution solvers, preconditioners, and smoothers.*

Iterative Solvers		Symmetric	Non-Symmetric	Preconditioners	Smoothers	Symmetric	Non-Symmetric
Krylov Subspace	CG	✓	–	none		–	–
				DIC		✓	–
	BiCG	–	✓	FDIC		✓	–
				DILU		–	✓
	BiCGStab	✓	✓	Diagonal		✓	✓
				GAMG		✓	✓
Multi-Grid (MG)	GAMG	✓	✓		Gauss-Seidel	✓	✓
					sym-Gauss-Seidel	✓	–
					DIC-Gauss-Seidel	✓	–
					DILU-Gauss-Seidel	–	✓

In the present work, the pressure equation, referred to as the mPPE, leads to a symmetrical coefficient matrix where it was found that the CG iterative solver in conjunction with the GAMG as a preconditioner and the Gauss-Seidel smoother is superior to all remaining solvers and preconditioners combinations in terms of computational efficiency. Instead, the DILU preconditioner with the BiCGStab solver showed to be more computationally efficient for the momentum equation that resulted in a non-symmetrical coefficient matrix due to the convective term linearisation.

It is important to note that the pressure equation is the main bottleneck in the numerical simulation process. Solving the pressure equation is considered the most time-consuming aspect of the entire simulation due to its high condition number[1].

[1]High condition number is an indicator of the convergence rate of the solution method that depends on the distribution of eigenvalues of the coefficient matrix $[A]$.

3.6 Summary

The fluid–solid solver employed in this research was implemented within the OpenFOAM environment. The Eulerian formulation, governed by the incompressible Navier-Stokes momentum equations, was modified in OpenFOAM to enable the coupling with the solid Lagrangian formulation. This coupling was achieved by introducing a boundary force term as a source term in the fluid governing equations to mimic the influence of an immersed solid boundary inside the fluid domain. The resulting fluid–solid system imposed two main constraints: one on the fluid's Eulerian domain, i.e., a divergence-free fluid velocity field, and another on the solid's Lagrangian domain, which will be addressed in detail in Chapter 4 of this thesis. Each term in the resulting system is discretised using a cell-centred finite volume approach with a second-order discretisation scheme, achieving overall second-order accuracy in both space and time. Given OpenFOAM's collocated grid arrangement, the Rhie-Chow momentum correction technique was applied to prevent checkerboard pressure oscillations.

A comprehensive overview is provided for the pressure–velocity coupling problem along with the strategies found in literature to address this coupled system. In this research, the PISO algorithm's predictor-corrector strategy was selected as the decoupling method, where a thorough derivation of its predictor and corrector steps was demonstrated. Additionally, a modified PPE was formulated by incorporating the divergence of the boundary force term. Finally, the iterative solvers available in OpenFOAM for solving the system of linear algebraic equations arising from the momentum and pressure equations are introduced.

Chapter 4

Direct-Forcing Immersed Boundary Method (DF-IBM)

Contents

In this chapter, the mathematical foundations underpinning the direct-forcing immersed boundary method (DF-IBM) are highlighted to effectively simulate flows around arbitrarily moving complex and rigid structures. To address the data mapping between the Eulerian and Lagrangian domains, this chapter introduces the IBM-related linear operators and several interpolation functions along with the construction of the Lagrangian support domain, to ensure accurate data mapping. This chapter also explains the roots of the second fluid–solid system constraint, i.e., the no-slip velocity boundary condition as encountered within the DF-IBM framework, and its direct relation to the accurate computation of the Lagrangian weights. Finally, an improved implicit DF-IBM algorithm is proposed, where a unified set of improvements is sought that enhances the overall performance and efficiency of the method.

4.1 DF-IBM Governing Equations

The IBM is a general-purpose numerical approach for simulating FSI through a body non-conformal mesh. The formulation derived involves a mixture of variables from the underlying fluid Eulerian domain Ω_f and solid Lagrangian domain Γ_s, where the interior of the immersed boundary Ω_s is treated as part of the fluid Eulerian domain as illustrated in Fig. (4.1), allowing the fluid flow to develop inside Γ_s. The general direct-forcing immersed boundary method (DF-IBM) governing equations of the fluid–solid system are derived as follows:

$$\nabla \cdot \mathbf{u} = 0, \qquad \text{in } \Omega_f, \tag{4.1a}$$

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{u}\mathbf{u}) - \nabla \cdot (\nu \nabla \mathbf{u}) = -\nabla p + \mathbf{f}, \qquad \text{in } \Omega_f, \tag{4.1b}$$

$$\mathbf{U}_{IB}(\mathbf{X}_n, t) = \mathscr{I}[\mathbf{u}(\mathbf{x}_{i,j}, t)], \qquad \text{from } \Omega_f \text{ to } \Gamma_s, \tag{4.1c}$$

$$\mathbf{f}(\mathbf{x}_{i,j}, t) = \mathscr{S}[\mathbf{F}_{IB}(\mathbf{X}_n, t)], \qquad \text{from } \Gamma_s \text{ to } \Omega_f, \tag{4.1d}$$

where,

- $\mathbf{x}_{i,j}$ is the fluid position of an Eulerian grid point (i, j),
- $\mathbf{X}_n$ is the boundary position of a Lagrangian marker n,
- $\mathbf{u}(\mathbf{x}_{i,j}, t)$ is the fluid velocity of an Eulerian grid point,
- $\mathbf{U}_{IB}(\mathbf{X}_n, t)$ is the desired solid immersed boundary velocity at each Lagrangian marker,
- $\mathbf{f}(\mathbf{x}_{i,j}, t)$ is the boundary force exerted on the Eulerian grid points,
- $\mathbf{F}_{IB}(\mathbf{X}_n, t)$ is the singular boundary force at each Lagrangian marker,
- $\mathscr{I}$ and $\mathscr{S}$ are two linear operators that manage the data mapping between Eulerian and Lagrangian grids.

Note that Eqs. (4.1a) and (4.1b) are the fluid governing equations, as explained in Chapter 3. They are completely expressed in the Eulerian domain, whereas Eqs. (4.1c) and (4.1d) are the IBM-related linear operators, which are used to map the variables from the fluid Eulerian grid to the solid Lagrangian grid using Eq. (4.1c) and in the opposite direction using Eq. (4.1d).

The operation carried out in Eq. (4.1c) serves to transfer the local fluid velocity $\mathbf{u}$ to the solid immersed boundary $\mathbf{U}_{IB}$. It is named the velocity interpolation linear operator. While

the operation performed in Eq. (4.1d) provides a transfer of the forces exerted by the solid immersed boundary $\mathbf{F}_{IB}$ onto the fluid $\mathbf{f}$, and is named force spreading linear operator. The obtained Eulerian force $\mathbf{f}$ is inserted as a source term inside the momentum conservation Eq. (4.1b) to account for the presence of a solid immersed boundary inside the fluid Eulerian domain. The spreading is considered as the reciprocal of the interpolation.

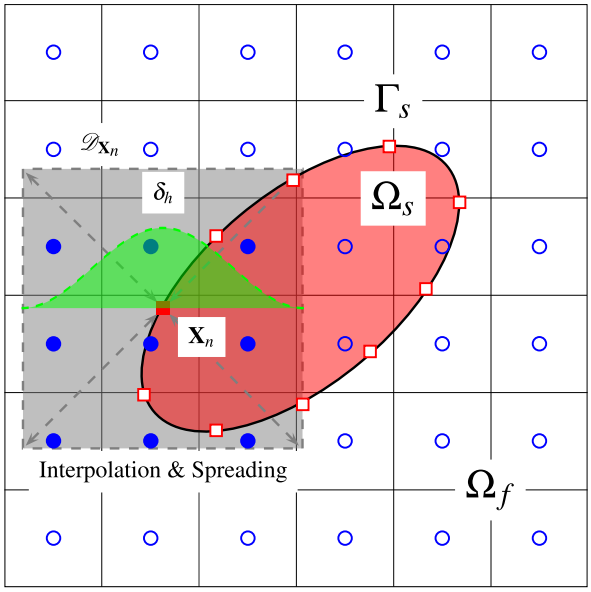


Figure 4.1: *Schematic of the interpolation and spreading operators in the DF-IBM with the aid of a smoothed approximation of the Dirac delta function δ_h. The Lagrangian marker $\mathbf{X}_n$ where the operators are taking place is represented by "■", the Eulerian grid points that fall inside the Lagrangian support domain $\mathscr{D}_{\mathbf{X}_n}$ are represented by "●", and all remaining Lagrangian markers and Eulerian grid points are illustrated by "□" and "○", respectively.*

4.2 IBM-Related Linear Operators

The data transfer between the Eulerian and Lagrangian grids is conducted in a two-way procedure using two separate linear operators $\mathscr{I}$ and $\mathscr{S}$ through a smoothed approximation of the Dirac delta function δ_h also known as the distribution function, as shown in Fig. 4.1. The former takes an Eulerian input and generates a Lagrangian output, while the latter does the contrary.

a) Velocity interpolation operator:

$$\mathbf{U}(\mathbf{X}_n,t) = \mathscr{I}[\mathbf{u}(\mathbf{x}_{i,j},t)] = \sum_{\mathbf{x}_{i,j}\in\mathscr{D}_{\mathbf{X}_n}} \mathbf{u}(\mathbf{x}_{i,j},t)\delta_h(\mathbf{x}_{i,j}-\mathbf{X}_n)\Delta V_{i,j}. \tag{4.2}$$

This operator represents the interpolation of the Eulerian velocity field back to each Lagrangian marker $\mathbf{X}_n$. For Cartesian uniform meshes, the Eulerian weight is $\Delta V_{i,j} = h^2$ (for 2D topology), with h being the distance between two Eulerian grid points such as $h = \Delta x = \Delta y$, and $\left(\mathbf{x}_{i,j} \in \mathscr{D}_{\mathbf{X}_n}\right)$ represents the collection of all neighbouring Eulerian grid points that fall inside the support domain $\mathscr{D}_{\mathbf{X}_n}$ of the Lagrangian marker $\mathbf{X}_n$ shown by the dashed square in Fig. 4.1.

b) Force spreading operator:

$$\mathbf{f}(\mathbf{x}_{i,j},t) = \mathscr{S}[\mathbf{F}_{IB}(\mathbf{X}_n,t)] = \sum_{\mathbf{X}_n \in \mathscr{D}_{\mathbf{x}_{i,j}}} \mathbf{F}_{IB}(\mathbf{X}_n,t)\delta_h(\mathbf{x}_{i,j} - \mathbf{X}_n)W_n. \tag{4.3}$$

The computed Lagrangian force $\mathbf{F}_{IB}$ is spread back to the neighbouring Eulerian grid points in the support domain of the Lagrangian marker $\mathbf{X}_n$. The Lagrangian weight W_n is the weight associated to the solid immersed boundary marker, and the methods used to compute it are detailed in Section 4.4.2. $(\mathbf{X}_n \in \mathscr{D}_{\mathbf{x}_{i,j}})$ refers to a loop over all the Lagrangian markers whose support domain $\mathscr{D}_{\mathbf{X}_n}$ contains the same Eulerian grid point $\mathbf{x}_{i,j}$.

In addition, the immersed boundary is represented by a set of Lagrangian markers which are uniformly distributed over the boundary, with the arc length Δs, i.e., the distance between two consecutive Lagrangian markers approximately equal to the Eulerian grid spacing, $\Delta s/h \approx 1$.

4.3 Eulerian–Lagrangian Mapping

4.3.1 Interpolation Function δ_h

Numerous techniques are present to manage the data mapping from the Eulerian to Lagrangian grids and vice versa. One can categorise them into two main groups [112], (a) The first group is based on the local reconstruction schemes of the solution that are dependent on the desired immersed boundary values, for example, these schemes can be found in DF-IBM variant employed in [75, 76, 113], or in the ghost-cell IBM approach exploited in [47, 49], and (b) the second group is based on the spreading of the boundary force $\mathbf{F}_{IB}$ onto the neighbouring fluid grid points of the solid immersed boundary using, for instance, using a kernel function such as the discrete smoothed Dirac delta function as in [31, 32, 34, 35, 90,

114], the reproducing kernel particle method (RKPM) found in [85, 115], the radial basis function (RBF) used in [116] and the moving-least-squares method (MLS) as in [117].

In the present work, the conventional Dirac delta function was adopted in its discrete form as the interpolation function. For Cartesian uniform meshes, the 2D approximation of the discrete Dirac delta function is given by the tensor product of 1D single-variable continuous functions:

$$\delta_h(\mathbf{x}_{i,j} - \mathbf{X}_n) = \delta_h(x_{i,j} - X_n)\delta_h(y_{i,j} - Y_n), \tag{4.4}$$

with

$$\delta_h(x_{i,j} - X_n) = \frac{1}{h}\phi(r_x) \quad \& \quad \delta_h(y_{i,j} - Y_n) = \frac{1}{h}\phi(r_y), \tag{4.5}$$

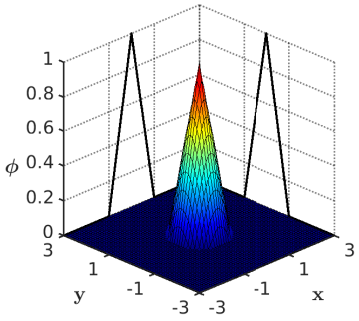
and

$$(r_x, r_y) = \left(\frac{x_{i,j} - X_n}{h}, \frac{y_{i,j} - Y_n}{h}\right), \tag{4.6}$$

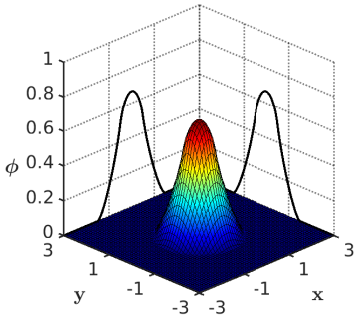
where ϕ is the non-dimensional continuous function named kernel function and (r_x, r_y) denotes a non-dimensional distance between a Lagrangian marker and an Eulerian grid point. The kernel function scales to the mesh size h, i.e., the fluid grid spacing inside the Lagrangian support domain $\mathscr{D}_{\mathbf{X}_n}$ centred at each Lagrangian marker $\mathbf{X}_n = (X_n, Y_n)$. The Lagrangian support domain encloses all neighbouring Eulerian grid points $\mathbf{x}_{i,j} = (x_{i,j}, y_{i,j})$ used for interpolating and spreading the velocity and boundary force, respectively.

The continuous kernel function ϕ can take various forms. In the present numerical investigation, the 2-point-width hat function ϕ_1 found in [114] was used as the kernel function, as well as the 3-point-width function ϕ_2 of [90], the 4-point-width function ϕ_3 [34], and the 4-point-width cosine function ϕ_4 of Peskin [32]. Additionally, the smoothed version of all previously mentioned functions ϕ_5, ϕ_6, ϕ_7, and ϕ_8 found in [86], were employed since they suppress the non-physical force oscillations when a moving immersed boundary is simulated due to its high moment condition [86]. The study also includes a new 5-point-width function ϕ_9 and a new 6-point-width function ϕ_{10} developed in [118, 119]. The mathematical formulations of all mentioned kernel functions can be found in Appendix A.

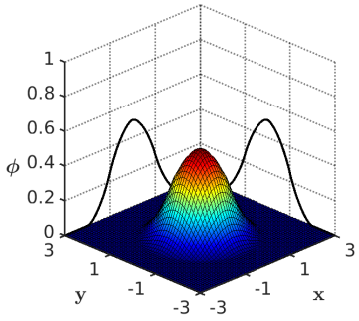
The former kernel functions are displayed in one-dimensional and two-dimensional forms in Fig. (4.2). Those functions are not chosen arbitrarily, there exist three important criteria for constructing an ideal kernel function [118, 119]: (1) grid translational invariance, (2) interpolation accuracy, and (3) computational efficiency. Therefore, a certain set of properties and conditions needs to be satisfied firsthand so that ϕ is uniquely determined. The two essential properties that need to be satisfied by any kernel function ϕ in all dimensions are the following:



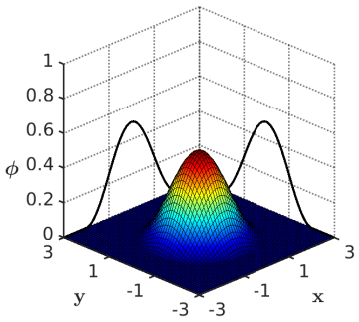
(a) ϕ_1



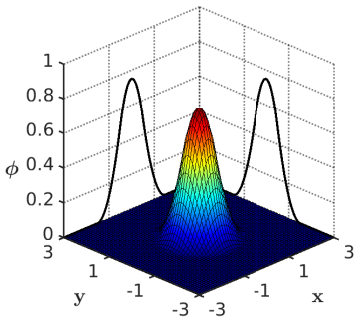
(b) ϕ_2



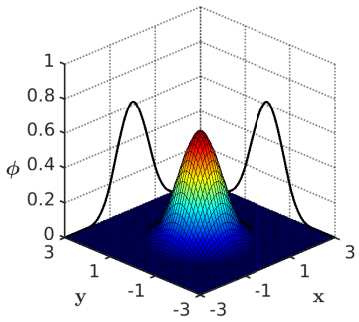
(c) ϕ_3



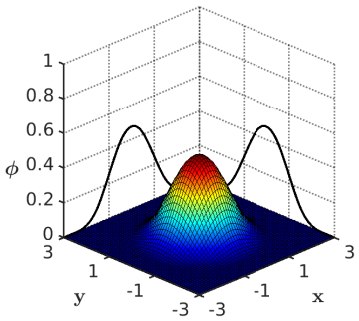
(d) ϕ_4



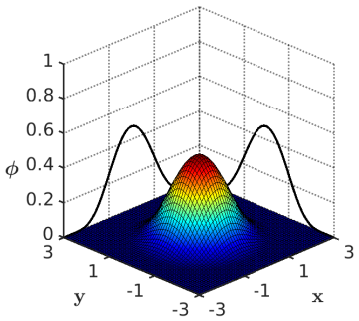
(e) ϕ_5



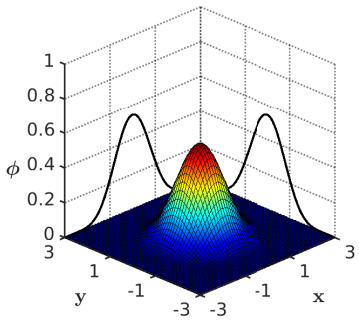
(f) ϕ_6



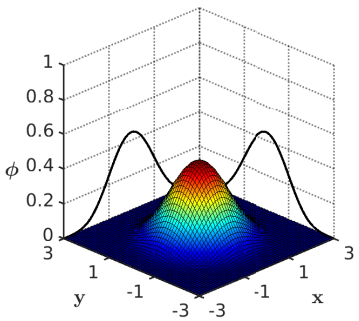
(g) ϕ_7



(h) ϕ_8



(i) ϕ_9



(j) ϕ_{10}

Figure 4.2: *One-dimensional and two-dimensional plots of all continuous kernel functions* ϕ.

(i) $\phi\left(\dfrac{x_{i,j}-X_n}{h}\right)$ is continuous for all real numbers $r_x = \dfrac{x_{i,j}-X_n}{h}$,

(ii) $\phi\left(\dfrac{x_{i,j}-X_n}{h}\right)$ have a compact support, i.e., $\phi\left(\dfrac{x_{i,j}-X_n}{h}\right) = 0$, for $|r_x| = \dfrac{|x_{i,j}-X_n|}{h} \geq r_s$,

where r_s is the support radius of the kernel function ϕ selected. Note that condition (i) is used to make sure the interpolated velocity and spread forces vary continuously as the Lagrangian markers move through the Eulerian grid and no sudden jumps are present in the field. Condition (ii) however, is proposed for computational efficiency, so a bounded support is created to prevent each Lagrangian marker from interacting with all fluid grid points. Now, the remaining conditions that are necessary but not essential by the kernel function ϕ are called the moment conditions and they are listed as the following:

(iii) Zeroth moment condition:

$$\sum_{\mathbf{x}_{i,j}\in\mathscr{D}_{\mathbf{X}_n}} \phi\left(\frac{x_{i,j}-X_n}{h}\right) = 1, \tag{4.7}$$

(iv) First moment condition:

$$\sum_{\mathbf{x}_{i,j}\in\mathscr{D}_{\mathbf{X}_n}} \left(x_{i,j}-X_n\right)\phi\left(\frac{x_{i,j}-X_n}{h}\right) = 0, \tag{4.8}$$

(iii) & (iv) Zeroth & First moment conditions combined:

$$\sum_{\mathbf{x}_{i,j}\in\mathscr{D}_{\mathbf{X}_n}} x_{i,j}\phi\left(\frac{x_{i,j}-X_n}{h}\right) = X_n, \tag{4.9}$$

(v) Second moment condition:

$$\sum_{\mathbf{x}_{i,j}\in\mathscr{D}_{\mathbf{X}_n}} \left(x_{i,j}-X_n\right)^2\phi\left(\frac{x_{i,j}-X_n}{h}\right) = K, \text{ where } K \text{ is a constant}, \tag{4.10}$$

(vi) Third moment condition:

$$\sum_{\mathbf{x}_{i,j}\in\mathscr{D}_{\mathbf{X}_n}} \left(x_{i,j}-X_n\right)^3\phi\left(\frac{x_{i,j}-X_n}{h}\right) = 0, \tag{4.11}$$

(vii) Even-odd condition:

$$\sum_{\mathbf{x}_{i,j}\in\mathscr{D}_{\mathbf{X}_n},\ even} \phi\left(\frac{x_{i,j}-X_n}{h}\right) = \sum_{\mathbf{x}_{i,j}\in\mathscr{D}_{\mathbf{X}_n},\ odd} \phi\left(\frac{x_{i,j}-X_n}{h}\right) = \frac{1}{2}, \tag{4.12}$$

(viii) Sum of squares condition:

$$\sum_{\mathbf{x}_{i,j}\in\mathscr{D}_{\mathbf{X}_n}} \left(\phi\left(\frac{x_{i,j}-X_n}{h}\right)\right)^2 = C, \text{ where } C \text{ is a constant.} \tag{4.13}$$

In a nutshell:

- The zeroth moment condition (iii), when satisfied, implies that the mass density and total force on the Lagrangian and Eulerian grids are equivalent, which guarantees that a constant function is interpolated exactly, though smoothed functions are interpolated with a first-order of accuracy.
- The first moment condition (iv) indicates that the total torque is identical on both grids.
- Conditions (iii) and (iv) combined exhibit a second-order interpolation accuracy when smoothed functions are interpolated using δ_h, although linear functions are interpolated exactly.
- The second moment condition (v) and the third moment condition (vi) are required when higher-order interpolation accuracy is necessary.
- Condition (v) ensures that quadratic functions are interpolated exactly by the corresponding interpolation function δ_h, while smoothed functions achieve third-order accuracy.
- In contrast, condition (vi) guarantees exact interpolation of cubic functions, and a fourth-order interpolation accuracy is attained for smoothed functions.
- Condition (vii) is considered the strong form of condition (iii), therefore, if the former condition is satisfied the zeroth moment condition is met. When using a collocated grid arrangement for the fluid grid points, where the pressure and the velocity are located at the same position in the fluid grid cell, can lead to even-odd decoupling between the pressure and velocity since the P and u are not fully decoupled. These even-odd oscillations are due because of the discretisation error on this specific grid arrangement that

produces checkerboard patterns in the converged solution. Whereas using a staggered grid arrangement for the fluid grid points, where the velocity and pressure fields are decoupled having the pressure located at the centre of the fluid cell and the velocity components located at the fluid cell faces, can avoid the even-odd decoupling between P and u, and prevent non-physical oscillations on the fluid grid points and instability in the sought solution. The even-odd condition hints that the separate sums over $\mathbf{x}_{i,j}$ even and over $\mathbf{x}_{i,j}$ odd add up to $\frac{1}{2}$ each, to circumvent the checkerboard type instability when using a collocated grid arrangement for the Eulerian grid.

- The final condition (viii), is a weaker version of the exact grid translational invariance [118, 119]. This exact grid translational invariance condition indicates that the IBM results would stay identical regardless of the Lagrangian marker's position if shifted for fixed amounts with respect to the Eulerian grid points. In other words, the interaction between the two grids is independent of the position of the solid immersed boundary relative to the fluid grid. Moreover, the sum of squares condition (viii) denotes that the two-way process, involving the IBM-related linear operators, is independent of the Lagrangian marker's position relative to the fluid grid. This condition also implies that the mapping is constant; when a force is spread from a Lagrangian marker to the Eulerian grid and then interpolated back from the Eulerian grid to the Lagrangian marker, its value remains unchanged. This results in a reciprocal relationship between the linear operators.

All mentioned kernel functions ϕ and their corresponding properties and conditions are presented in Table 4.1. Since conditions (iii) and (iv) together ensure that a smoothed function will be interpolated with second-order accuracy. Consequently, ϕ_1, ϕ_2, ϕ_3, and ϕ_8 are second-order accurate, but ϕ_4 gives only a first-order interpolation since it only satisfies the zeroth moment condition. Note that, ϕ_5, ϕ_6, and ϕ_7 ensure a third-order of interpolation accuracy, while a fourth-order accuracy is manifested by ϕ_9 and ϕ_{10}. Only the kernel functions ϕ_2, ϕ_3, ϕ_4, ϕ_9, and ϕ_{10} satisfies the sum of squares condition (viii) where each value of C is mentioned in the table. The last column in Table 4.1, stands for the regularity $\mathscr{C}^n$ of the continuous kernel function ϕ, where n here represents the order of continuous derivatives of the ϕ function. It is clear that, ϕ_1 is of $\mathscr{C}^0$ continuity, which means it does not have any continuous derivative. On the other hand, at least a first-order continuous derivative for the remaining kernel functions exists, bearing in mind that the smoothed versions of the kernel function ϕ possess one higher-order derivative compared to their counterpart. In [118, 119],

it was stated that for certain applications, a higher regularity is considered as a nice feature of the kernel function.

Table 4.1: *Properties and moment conditions for all continuous kernel function ϕ.*

	Continuous $\forall r$ (i)	Support radius r_s (ii)	Zeroth Moment (iii)	First Moment (iv)	Second Moment (v)	Third Moment (vi)	Even-Odd Condition (vii)	Sum of Squares Condition (viii)	Order of Accuracy	Regularity $\mathscr{C}^n$
ϕ_1	✓	1	✓	✓	×	×	×	×	2^{nd}	$\mathscr{C}^0$
ϕ_2	✓	1.5	✓	✓	×	×	×	✓$(C=\frac{1}{2})$	2^{nd}	$\mathscr{C}^1$
ϕ_3	✓	2	✓	✓	×	×	✓	✓$(C=\frac{3}{8})$	2^{nd}	$\mathscr{C}^1$
ϕ_4	✓	2	✓	×	×	×	✓	✓$(C=\frac{3}{8})$	1^{st}	$\mathscr{C}^1$
ϕ_5	✓	1.5	✓	✓	✓$(K=\frac{1}{4})$	×	×	×	3^{rd}	$\mathscr{C}^1$
ϕ_6	✓	2	✓	✓	✓$(K=\frac{7}{12}+\frac{-\sqrt{3}\pi}{27})$	×	✓	×	3^{rd}	$\mathscr{C}^2$
ϕ_7	✓	2.5	✓	✓	✓$(K=1-\frac{\pi}{8})$	×	×	×	3^{rd}	$\mathscr{C}^2$
ϕ_8	✓	2.5	✓	✓	×	×	×	×	2^{nd}	$\mathscr{C}^2$
ϕ_9	✓	2.5	✓	✓	✓$(K=\frac{38}{60}-\frac{\sqrt{69}}{60})$	✓	×	✓$(C=0.39254)$	4^{th}	$\mathscr{C}^3$
ϕ_{10}	✓	3	✓	✓	✓$(K=\frac{59}{60}-\frac{\sqrt{29}}{20})$	✓	✓	✓$(C=0.32577)$	4^{th}	$\mathscr{C}^3$

4.3.2 Lagrangian Support Domain $\mathscr{D}_{\mathbf{X}_n}$

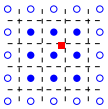
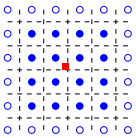
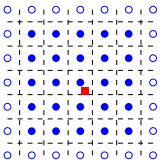
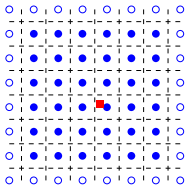
A 2D Lagrangian support domain $\mathscr{D}_{\mathbf{X}_n}$ centred at every Lagrangian marker $\mathbf{X}_n$ of the immersed boundary Γ_s needs to be created for the IBM-related operators to take place. In a Cartesian uniform mesh, if the 4-point-width function ϕ_3 is considered for instance, the support width is set equal to $(2r_s)$, i.e., 4; therefore the dimensions of $\mathscr{D}_{\mathbf{X}_n}$ for the kernel function ϕ_3 is equal to 4 mesh widths $(=4h)$ in each spatial direction, to ensure at least 4 Eulerian grid points are placed inside $\mathscr{D}_{\mathbf{X}_n}$ in every dimension in space, leading to a support domain with 16 Eulerian grid points in total. The characteristics of every kernel function's Lagrangian support domain $\mathscr{D}_{\mathbf{X}_n}$ are presented in Table 4.2 for 2D applications.

The following algorithm generates the square-shaped Lagrangian support domain $\mathscr{D}_{\mathbf{X}_n}$, centred at each Lagrangian marker $\mathbf{X}_n$ to compute the Dirac delta function δ_h. This approach is inspired by the method developed in [85] for non-uniform meshes:

1. First, the Eulerian grid point $\mathbf{x}_{i,j}^{proxi} = (x_{i,j}^{proxi}, y_{i,j}^{proxi})$ in close proximity to the Lagrangian point $\mathbf{X}_n = (X_n, Y_n)$ is chosen.

2. For every $\mathbf{x}_{i,j}^{proxi}$ point, its neighbouring points in the West, East, North, and South

Table 4.2: *Lagrangian support domain $\mathscr{D}_{\mathbf{X}_n}$ characteristics for every kernel function ϕ.*

	Support width ($2r_s$)	Minimum number of Eulerian grid points in 2D that $\in \mathscr{D}_{\mathbf{X}_n}$	Configuration in 2D†
ϕ_1	2	4	
ϕ_2, ϕ_5	3	9	
ϕ_3, ϕ_4, ϕ_6	4	16	
ϕ_7, ϕ_8, ϕ_9	5	25	
ϕ_{10}	6	36	

† "○" Eulerian grid points situated outside $\mathscr{D}_{\mathbf{X}_n}$,
"●" Eulerian grid points that falls inside $\mathscr{D}_{\mathbf{X}_n}$,
"■" Lagrangian marker centre of $\mathscr{D}_{\mathbf{X}_n}$.

directions are sought as follows:

$$\begin{Bmatrix} - & \mathbf{x}^{N}_{i+1,j} & - \\ \mathbf{x}^{W}_{i,j-1} & \mathbf{x}^{proxi}_{i,j} & \mathbf{x}^{E}_{i,j+1} \\ - & \mathbf{x}^{S}_{i-1,j} & - \end{Bmatrix}$$

3. Add these Eulerian grid points to $\mathscr{D}_{\mathbf{X}_n}$ surrounding the Lagrangian point $\mathbf{X}_n$ as illustrated in Fig. (4.3a), where the set $\mathscr{N}_{\mathbf{X}_n} = \{\mathbf{x}^{proxi}_{i,j}, \mathbf{x}^{W}_{i+1,j}, \mathbf{x}^{E}_{i-1,j}, \mathbf{x}^{N}_{i,j+1}, \mathbf{x}^{S}_{i,j-1}\}$ is formed.

4. Next, the dimensions of the rectangular support domain are automatically adapted to the selected kernel function since they depend on its support radius r_s, having the dimensions evaluated as $[2r_sh, 2r_sh]$. Hence, for the 4-point-width function ϕ_4 the dimensions are $[4h, 4h]$, as shown in Fig. (4.3b).

5. Finally, the set of remaining neighbouring Eulerian grid points $\mathscr{P}_{\mathbf{X}_n}$ that fall inside

this support domain are added using the following criteria that is based on the support radius r_s of the selected kernel function ϕ:

$$\mathscr{P}_{\mathbf{X}_n} = \left\{ |\mathbf{x}_{i,j}^{neigh} - \mathbf{X}_n| < r_s h + \varepsilon_h \right\}. \tag{4.14}$$

To ensure that neighbouring Eulerian grid points do not touch the boundaries of the Lagrangian support domain, a correction factor ε_h is added. The final set of Eulerian grid points within the Lagrangian support domain is represented by $\mathscr{D}_{\mathbf{X}_n} = \{\mathscr{N}_{\mathbf{X}_n} \cup \mathscr{P}_{\mathbf{X}_n}\}$. The condition specified for the 4-point-width function ϕ_3 is illustrated in Fig. (4.3c), where 16 Eulerian grid points are grouped within $\mathscr{D}_{\mathbf{X}_n}$ and centred at $\mathbf{X}_n$.

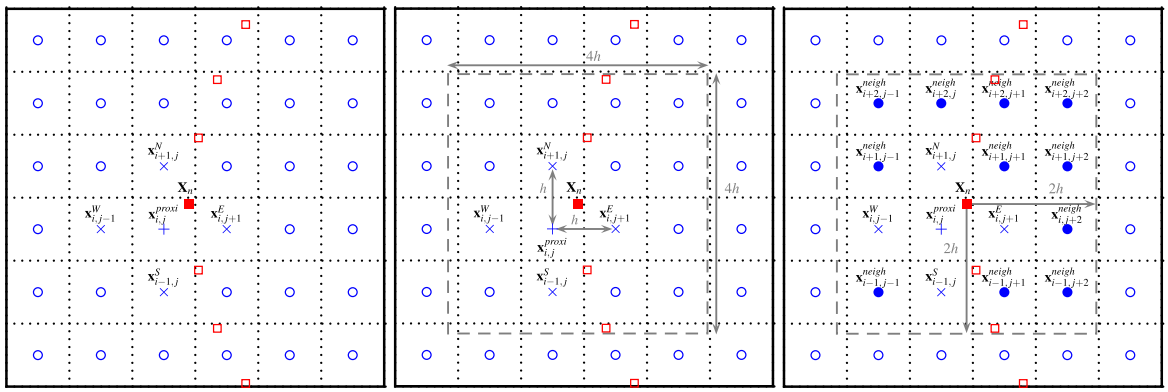


Figure 4.3: *Schematic of the Lagrangian support domain $\mathscr{D}_{\mathbf{X}_n}$ for the 4-point-width function ϕ_3 (with $r_s = 2$), where its centre is represented by the Lagrangian marker $\mathbf{X}_n$ "■", the Eulerian grid point $\mathbf{x}^{proxi}$ in close proximity to $\mathbf{X}_n$ is represented by "+", its neighbouring points from West, East, North, and South directions $\mathbf{x}^{W,E,N,S}$ are represented by "×", and the remaining neighbouring Eulerian grid points $\mathbf{x}^{neigh}$ that fall inside the computed dimensions $[4h, 4h]$ of $\mathscr{D}_{\mathbf{X}_n}$ are represented by "•".*

4.3.3 Eulerian–Lagrangian Identities

The spreading operator is considered as the adjoint of the interpolation, which leads to two important identities regardless of the form of the interpolation function δ_h [34]. The first identity states that the total linear momentum is the same whether computed in the Lagrangian or Eulerian domain. The second identity declares that the total power is also equivalent on both domains regardless of the interpolation function used.

However, other identities exist, such as mass density, total force, and total torque identities, but they express absolute dependence on the interpolation function used [34]. Put

differently, they depend on a certain set of properties that should be met by the interpolation function δ_h used in the calculations. The two principal properties that need to be imposed on the interpolation function are the zeroth moment Eq. (4.7) and the first moment Eq. (4.8) conditions. Mass density identity is met once δ_h satisfies the zeroth moment condition only, which also indicates conservation of mass due to its independence on time. Similarly, the total force identity is also met once δ_h satisfies the zeroth moment Eq. (4.7) condition only:

$$\sum_{\mathbf{x}_{i,j}\in\mathscr{D}_{\mathbf{X}_n}} \mathbf{f}(\mathbf{x}_{i,j},t)\Delta V_{i,j} = \sum_{\mathbf{X}_n\in\mathscr{D}_{\mathbf{x}_{i,j}}} \mathbf{F}_{IB}(\mathbf{X}_n,t)W_n. \tag{4.15}$$

Finally, the total torque identity is established once δ_h satisfies both zeroth moment Eq. (4.7) and first moment Eq. (4.8) conditions simultaneously:

$$\sum_{\mathbf{x}_{i,j}\in\mathscr{D}_{\mathbf{X}_n}} \mathbf{x}_{i,j}(t)\times\mathbf{f}(\mathbf{x}_{i,j},t)\Delta V_{i,j} = \sum_{\mathbf{X}_n\in\mathscr{D}_{\mathbf{x}_{i,j}}} \mathbf{X}_n(t)\times\mathbf{F}_{IB}(\mathbf{X}_n,t)W_n. \tag{4.16}$$

4.4 DF-IBM Algorithms: A Review

Different DF-IBM algorithms can be found in the literature, implemented within a variety of numerical techniques for solving PDEs ranging from FDM [39, 76, 81], FVM [85, 120, 121], FEM [115, 122, 123], and LBM [82, 84, 124]. These DF-IBM algorithms are inserted inside the incompressible Navier-Stokes equations where the pressure–velocity ($p-\mathbf{U}$) coupling is handled by using either the projection method (fractional step method, time splitting method) and its variants as in [39, 81, 85], the SIMPLE algorithm [125], and the PISO algorithm [126, 127]. Regardless of the numerical technique and $p-\mathbf{U}$ decoupling algorithm used, the DF-IBM steps present in every approach are immutable. The generic steps are ordered as follows:

1. Eulerian Velocity Interpolation Step:

$$\mathbf{U}(\mathbf{X}_n,t) = \mathscr{I}[\mathbf{u}^*(\mathbf{x}_{i,j},t)]. \tag{4.17}$$

   Here, a predicted Eulerian velocity field $\mathbf{u}^*$ is interpolated, which ignores the presence of a solid immersed boundary inside the fluid region, to obtain the Lagrangian velocity $\mathbf{U}$ on the Lagrangian marker $\mathbf{X}_n$.

2. Direct-Forcing Step:

$$\mathbf{F}_{IB}(\mathbf{X}_n,t) = \frac{\mathbf{U}_{IB}(\mathbf{X}_n,t)-\mathbf{U}(\mathbf{X}_n,t)}{\Delta t}, \tag{4.18}$$

where $\mathbf{U}_{IB}$ represents the desired (or target) Lagrangian velocity of the immersed boundary and is directly imposed at the Lagrangian marker $\mathbf{X}_n$, which can be determined either using a prescribed equation of motion or from the coupling with a solid rigid body solver. For stationary immersed bodies, $\mathbf{U}_{IB}$ is set equal to zero.
In this step, the singular Lagrangian boundary force $\mathbf{F}_{IB}$ is calculated based on the second law of Newton for rigid bodies, by taking the difference between the desired velocity and the Lagrangian velocity obtained from the previous interpolation step, for each Lagrangian marker $\mathbf{X}_n$ and dividing by the simulation time-step.

3. Lagrangian Force Spreading Step:

$$\mathbf{f}(\mathbf{x}_{i,j},t) = \mathscr{S}[\mathbf{F}_{IB}(\mathbf{X}_n,t)]. \tag{4.19}$$

The Lagrangian boundary force $\mathbf{F}_{IB}$ is spread from each Lagrangian marker $\mathbf{X}_n$ onto their corresponding neighbouring Eulerian grid points to obtain the Eulerian boundary force $\mathbf{f}$.

4. Update Eulerian Velocity Step:
This step is used to update and correct the predicted velocity field $\tilde{\mathbf{u}}$ to reckon the presence of the solid immersed boundary inside the fluid region. The equation used to update the Eulerian velocity is not generic, but solver dependent. The explicit correction using the implicit Euler scheme (also known as the Backward Euler method) for time discretisation yields the following correction:

$$\tilde{\mathbf{u}}(\mathbf{x}_{i,j},t) = \mathbf{u}^*(\mathbf{x}_{i,j},t) + \Delta t\mathbf{f}(\mathbf{x}_{i,j},t). \tag{4.20}$$

4.4.1 Immersed Boundary No-Slip Velocity Boundary Condition

The spreading and the interpolation operators in the DF-IBM are reciprocal, i.e., inversely related. For instance, if a vector field $\mathbf{u}$ is interpolated from an Eulerian to a Lagrangian domain $\mathscr{I}[\mathbf{u}]$, then the obtained vector field in the Lagrangian domain is spread back to the Eulerian domain $\mathscr{S}[\mathscr{I}[\mathbf{u}]]$, the same vector field $\mathbf{u}$ is obtained. Once this reciprocity is violated, a no-slip velocity boundary error is generated, allowing the fluid to penetrate the solid domain.

This boundary error problem can be manifested by using the first explicit DF-IBM approach proposed by Uhlmann [39], and employing the concept of multi-DF-IBM found

in [41, 77, 78] will help understand the roots of this error and how it can be resolved.

The error will be explained in a simplified context, assuming a Lagrangian domain represented by one marker ($\mathbf{X}_1$) and surrounded by two equidistant Eulerian grid points ($\mathbf{x}_1, \mathbf{x}_2$), and assuming that $\delta_1 = 0.6$ and $\delta_2 = 0.4$ while keeping in mind the zeroth moment condition. If one proceeds with the explicit DF-IBM, i.e., one iteration as in [39] and assumes that the Lagrangian weight is almost equal to the Eulerian weight, as proposed by Uhlmann, with $W_n = \Delta V_{i,j} = 1$ for simplicity, the following equations for the interpolation, direct-forcing, spreading, and updated Eulerian velocity steps for a single time-step can be obtained

$$\mathbf{U}(\mathbf{X}_1, t) = \mathbf{u}^*(\mathbf{x}_1, t)\delta_1 + \mathbf{u}^*(\mathbf{x}_2, t)\delta_2,$$

$$\mathbf{F}_{IB}(\mathbf{X}_1, t) = \frac{\mathbf{U}_{IB}(\mathbf{X}_1, t) - \mathbf{U}(\mathbf{X}_1, t)}{\Delta t},$$

$$\mathbf{f}(\mathbf{x}_1, t) = \mathbf{F}_{IB}(\mathbf{X}_1, t)\delta_1 \;\text{ and }\; \mathbf{f}(\mathbf{x}_2, t) = \mathbf{F}_{IB}(\mathbf{X}_1, t)\delta_2,$$

$$\tilde{\mathbf{u}}(\mathbf{x}_1, t) = \mathbf{u}^*(\mathbf{x}_1, t) + \Delta t\mathbf{f}(\mathbf{x}_1, t) \;\text{ and }\; \tilde{\mathbf{u}}(\mathbf{x}_2, t) = \mathbf{u}^*(\mathbf{x}_2, t) + \Delta t\mathbf{f}(\mathbf{x}_2, t).$$

The no-slip velocity boundary condition is labelled as the second fluid–solid system constraint. This error disappears once $\mathbf{U}_{IB}(\mathbf{X}_1, t) = \mathscr{I}[\tilde{\mathbf{u}}]$, which means that the no-slip velocity boundary condition is exactly enforced on the Lagrangian marker $\mathbf{X}_1$. However, this is not the case since

$$\mathscr{I}[\tilde{\mathbf{u}}] = \mathbf{U}_{IB}(\mathbf{X}_1, t)\left(\delta_1{}^2 + \delta_2{}^2\right) + \mathbf{U}(\mathbf{X}_1, t)\left(1 - \left(\delta_1{}^2 + \delta_2{}^2\right)\right), \tag{4.21}$$

with $(\delta_1 + \delta_2) = 1$, according to the zeroth moment condition Eq. (4.7), while $(\delta_1{}^2 + \delta_2{}^2) = C \neq 1$ according to the sum of squares condition expressed by Eq. (4.13) described earlier.

By introducing multiple DF-IBM iterations, a family of explicit or implicit IBMs is obtained. The distinction arises from whether a fixed number of iterations is prescribed or a tolerance between two successive DF-IBM iterations is defined. This approach ensures a better satisfaction of the second fluid–solid system constraint, namely the no-slip velocity boundary condition. After m iterations, the interpolated Eulerian velocity field is obtained as

follows:

$$\mathscr{I}[\tilde{\mathbf{u}}^{(m)}] = \mathbf{U}_{IB}(\mathbf{X}_1,t)\ \mathscr{A} + \mathbf{U}^{(0)}(\mathbf{X}_1,t)\ \mathscr{B}, \tag{4.22}$$

where $\mathscr{A} = \sum_{i=0}^{m-1} C(1-C)^i$, and $\mathscr{B} = (1-C)^m$.

The superscript (m) represents the number of iterations, having $\tilde{\mathbf{u}}^{(0)} = \mathbf{u}^*$ and $\mathbf{U}^{(0)}(\mathbf{X}_1,t) = \mathscr{I}[\mathbf{u}^*]$. The ideal case is to have the term $\mathscr{A} = 1$ and the term $\mathscr{B} = 0$, to satisfy the no-slip velocity boundary condition $\mathbf{U}_{IB}(\mathbf{X}_1,t) = \mathscr{I}[\tilde{\mathbf{u}}^{(m)}] = \mathbf{U}^{(m)}(\mathbf{X}_1,t)$ and impose the reciprocity on the interpolation and spreading operations. Table 4.3 reveals how the terms $\mathscr{A}$ and $\mathscr{B}$ behave when the multi-DF-IBM is implemented. Clearly, the term $\mathscr{A} \to 1$ and the term $\mathscr{B} \to 0$ when the number of iterations m increases.

Table 4.3: *Multi-DF-IBM effect on the interpolated Eulerian velocity after m iterations.*

m	**Term** $\mathscr{A}$	**Term** $\mathscr{B}$
0	0.0	1.0
1	0.52	0.48
2	0.7696	0.2304
20	0.99999957	0.00000043
∞	1.0	0.0

The no-slip velocity boundary error is directly related to the Lagrangian weight W_n, used in the force spreading operator. According to Uhlmann [39], the Lagrangian weight was first set equal to the Eulerian weight, which is equal in a 2D Cartesian uniform mesh, to h^2. This method was found to be inaccurate [84], due to the Lagrangian representation of the immersed boundary having a dimensional issue since it is described only through its boundary and not the whole volume, so one dimension is missing in the Lagrangian weight used in the force spreading operator. Hence, the Eulerian and Lagrangian weights belong to two distinct discretisation spaces as mentioned in [84] and therefore should not be equal. Different methods to calibrate the Lagrangian weight computations can be found in the literature, which improve the mathematical reciprocity of the interpolation/spreading operators. The methods used in this research are described in the following section.

In addition to the DF-IBM iterations, another approach to reduce the no-slip velocity boundary error is by decreasing the simulation time-step, as stated in [41, 78, 83].

4.4.2 Lagrangian Weight Computation Methods

The Lagrangian weight was primarily set equal to the Eulerian weight, as adopted by the majority of researchers [39, 41, 81, 128]

$$W_n = \Delta V_{i,j}. \tag{4.23}$$

Another method was proposed by Pinelli et al. [85] to enforce the reciprocity on the interpolation and spreading operators, where the Lagrangian weight was computed by solving the following system of linear equations,

$$\mathbf{A}\mathbf{W}_n = \mathbf{1}, \tag{4.24}$$

where the vectors $\mathbf{W}_n = [W_1, ..., W_{N_{Lag}}]^{\mathrm{T}}$ and $\mathbf{1} = [1, ..., 1]^{\mathrm{T}}$ have the dimension of N_{Lag}, with N_{Lag} being the total number of Lagrangian markers. $\mathbf{A}$ is the matrix defined by

$$\mathbf{A} = A_{m,n} = \sum_{\mathbf{x}_{i,j} \in \mathscr{D}_{\mathbf{X}_m}} \delta_h(\mathbf{x}_{i,j} - \mathbf{X}_m)\delta_h(\mathbf{x}_{i,j} - \mathbf{X}_n)h^2.$$

The system is solved using the Krylov subspace solver the BiCGStab. The advantage of this method is that it can be used on uniform or non-uniform Eulerian grids, in addition to reducing the no-slip velocity boundary error [84]. However, the iterative solving of the system of linear equations increases the computational cost, which is directly proportional to the total number of Lagrangian markers, N_{Lag}.

Furthermore, an analytical correction for the Lagrangian weight was proposed in [82, 83], based on two main assumptions. First, the curvature of the immersed boundary is significantly small, so the local distance can be represented by a straight wall. Second, the variation of the Lagrangian force along the immersed boundary is considered small, which means that no discontinuities are taken into consideration. The correction performed on the Lagrangian weight was deduced from Eq. (4.21), i.e., W_n is divided by the constant C obtained from the sum of squares Eq. (4.13) condition imposed by the kernel function used, meaning that this correction is kernel function dependent since the sum of squares condition is not satisfied by all kernel functions, hence can be applied for specific set of kernel functions as shown in Table 4.1. If Eq. (4.21) is re-calculated using the described correction, the no-slip velocity boundary condition $\mathbf{U}_{IB}(\mathbf{X}_1, t) = \mathscr{I}[\tilde{\mathbf{u}}]$ is satisfied exactly for this particular case. For a general case where multiple Lagrangian markers exist enclosed each by multiple Eulerian grid points, the condition is not purely satisfied. However, it was shown in [82, 83] that it

could still eliminate some of the error generated by the non-reciprocity of the DF-IBM linear operators. The aforementioned correction was later improved to include the curvature of the immersed boundary by Cheylan et al. [84]:

$$W_n = \frac{1}{\sum_{\mathbf{x}_{i,j}\in\mathscr{D}_{\mathbf{X}_m}} \sum_{\mathbf{X}_n\in\mathscr{D}_{\mathbf{x}_{i,j}}} \delta_h(\mathbf{x}_{i,j}-\mathbf{X}_n)\delta_h(\mathbf{x}_{i,j}-\mathbf{X}_m)h^2}. \tag{4.25}$$

The first sum is performed on each Lagrangian marker $\mathbf{X}_n$ whose support contains the same Eulerian grid points $\mathbf{x}_{i,j}$, while the second sum is performed on all Eulerian grid points $\mathbf{x}_{i,j}$ that fall inside the support domain of $\mathbf{X}_m$. This method can be used independently of the selected kernel function and its computational cost is considerably lower compared to the method developed by Pinelli et al. [85]. The difference between Eq. (4.24) and Eq. (4.25) for the no-slip velocity boundary error was shown to be minimal in [84].

4.5 Present DF-IBM Algorithm

An efficient implicit DF-IBM variant that reduces the computational effort of previous implicit IBMs, such as the IBPM of Taira and Colonius [79] and the IBM algorithm of Ji et al. [80] is proposed. This approach fulfils sequentially in the solution process the fluid–solid system's dual constraints formed for an incompressible flow problem, using the PISO algorithm as the fluid solver.

Modifications to the original PISO algorithm are presented here, for the velocity field to satisfy the two constraints problem at the current time-step, within the predictor-corrector framework, formed by:

(1) Divergence-free condition on Eulerian grid points, $\nabla\cdot\mathbf{u}^{n+1}=0$ in Ω_f,

(2) No-slip velocity boundary condition on the Lagrangian markers, ${\mathbf{U}_{IB}}^{n+1}=\mathscr{I}[\tilde{\mathbf{u}}]$ on Γ_s.

In addition, a stability criterion needs to be fulfilled for the PISO algorithm, stating that the Courant–Friedrichs–Lewy (CFL) number has to be limited to less than one. The two constraints are decoupled from the Navier-Stokes momentum equations through the operator-splitting of the PISO algorithm. The following procedure is adopted at each time-step, with $\mathbf{u}$, p, and $\mathbf{f}$ solved sequentially, and up until reaching convergence, where the superscript n and $n+1$ represent the past (known) and current time (unknown) levels, respectively,

1. Momentum Predictor:
   An initial implicit prediction of the velocity field $\mathbf{u}^*$ in the Eulerian grid is calculated

using Eq. (3.42), without the addition of the boundary force term, while the pressure field p from previous time-step t^n is used for the calculation:

$$a_P \mathbf{u}_P^* + \sum_N a_N \mathbf{u}_N^* = \mathbf{RHS}\left(\mathbf{u}^n, \mathbf{u}^{n-1}\right) - (\nabla p)^n . \tag{4.26}$$

It is noted that the predicted velocity does not satisfy the divergence-free condition, $\nabla \cdot \mathbf{u}^* \neq 0$.

2. DF-IBM Iterations:
Inner iterations start for $m = 1$, having $\tilde{\mathbf{u}}^{(0)}(\mathbf{x}_{i,j}, t^{n+1}) = \mathbf{u}^*$ and $\mathbf{F}_{IB}{}^{(0)}(\mathbf{X}_n, t^{n+1}) = \mathbf{F}_{IB}(\mathbf{X}_n, t^n)$. The iterations procedure is carried out using the following steps:

   (a) Eulerian Velocity Interpolation:

$$\mathbf{U}^{(m-1)}(\mathbf{X}_n, t^{n+1}) = \mathscr{I}[\tilde{\mathbf{u}}^{(m-1)}(\mathbf{x}_{i,j}, t^{n+1})], \tag{4.27}$$

   (b) Direct-Forcing:

$$\mathbf{F}_{IB}{}^{(m)}(\mathbf{X}_n, t^{n+1}) = \frac{\mathbf{U}_{IB}(\mathbf{X}_n, t^{n+1}) - \mathbf{U}^{(m-1)}(\mathbf{X}_n, t^{n+1})}{\Delta t} + \mathbf{F}_{IB}{}^{(m-1)}(\mathbf{X}_n, t^{n+1}), \tag{4.28}$$

   (c) Lagrangian Force Spreading:

$$\mathbf{f}^{(m)}(\mathbf{x}_{i,j}, t^{n+1}) = \mathscr{S}[\mathbf{F}_{IB}{}^{(m)}(\mathbf{X}_n, t^{n+1})], \tag{4.29}$$

   (d) Update Eulerian Velocity:

$$a_P \tilde{\mathbf{u}}_P^{(m)} + \sum_N a_N \tilde{\mathbf{u}}_N^{(m)} = \mathbf{RHS}\left(\mathbf{u}^n, \mathbf{u}^{n-1}\right) - (\nabla p)^n + \mathbf{f}^{(m)}, \tag{4.30}$$

   (e) The no-slip velocity boundary condition requirement:

$$\mathbf{U}_{IB}(\mathbf{X}_n, t^{n+1}) = \mathscr{I}[\tilde{\mathbf{u}}^{(m)}(\mathbf{x}_{i,j}, t^{n+1})] = \mathbf{U}^{(m)}(\mathbf{X}_n, t^{n+1}). \tag{4.31}$$

   A convergence criterion is set for two consecutive DF-IBM iterations to check whether the updated predicted velocity field satisfies the no-slip velocity bound-

ary condition on the Lagrangian markers:

$$\left\|E_{no-slip}\right\|_2 = \left\|E_{\tilde{\mathbf{u}}^{(m)}}\right\|_2 - \left\|E_{\tilde{\mathbf{u}}^{(m-1)}}\right\|_2 < \epsilon_{IBM}, \tag{4.32}$$

with

$$\left\|E_{\tilde{\mathbf{u}}^{(m)}}\right\|_2 = \left\|\mathbf{U}_{IB}(\mathbf{X}_n, t^{n+1}) - \mathbf{U}^{(m)}(\mathbf{X}_n, t^{n+1})\right\|_2,$$

and

$$\left\|E_{\tilde{\mathbf{u}}^{(m-1)}}\right\|_2 = \left\|\mathbf{U}_{IB}(\mathbf{X}_n, t^{n+1}) - \mathbf{U}^{(m-1)}(\mathbf{X}_n, t^{n+1})\right\|_2,$$

where ϵ_{IBM} is a user-specified tolerance, based on whether an accurate no-slip velocity boundary condition is required.

(f) If the defined convergence criterion is reached in *Step 2 (e)*, then:

$$\mathbf{f}^{n+1} = \mathbf{f}^{(m)}(\mathbf{x}_{i,j}, t^{n+1}) \quad \text{and} \quad \tilde{\mathbf{u}} = \tilde{\mathbf{u}}^{(m)}(\mathbf{x}_{i,j}, t^{n+1}), \tag{4.33}$$

otherwise, return to *Step 2 (a)*, and let $m = m + 1$.
It can be noted, that at this stage, the updated predicted velocity still does not satisfy the divergence-free condition, $\nabla \cdot \tilde{\mathbf{u}} \neq 0$, while the no-slip velocity boundary condition is fulfilled $\mathbf{U}^{(m)}(\mathbf{X}_n, t^{n+1}) = \mathscr{I}[\tilde{\mathbf{u}}^{(m)}(\mathbf{x}_{i,j}, t^{n+1})] = \mathbf{U}_{IB}(\mathbf{X}_n, t^{n+1})$.

3. PISO Corrector Loops:
Inner corrector loops start for $m = 1$, where $1 \leq m \leq m_f$, with $m_f = 3$ is the fixed number of loops. For $m = 1$, $\hat{\mathbf{u}}^{(0)} = \tilde{\mathbf{u}}$:

(a) To enforce the divergence-free constraint, the mPPE represented by Eq. (3.48a) is used here to compute the new pressure field:

$$\nabla \cdot \left(\frac{1}{a_P} (\nabla p)^{(m)} \right) = \nabla \cdot \left(\frac{\mathbf{H}\left(\hat{\mathbf{u}}^{(m-1)}, \mathbf{u}^n, \mathbf{u}^{n-1}\right)}{a_P} \right) + \nabla \cdot \left(\frac{1}{a_P} \mathbf{f}^{n+1} \right), \tag{4.34}$$

(b) The velocity field is then corrected explicitly from the former pressure solution

$p^{(m)}$ using Eq. (3.50):

$$\hat{\mathbf{u}}^{(m)} = \frac{\mathbf{H}\left(\hat{\mathbf{u}}^{(m-1)}, \mathbf{u}^{n}, \mathbf{u}^{n-1}\right)}{a_P} - \frac{1}{a_P}\left(\nabla p\right)^{(m)} + \frac{1}{a_P}\mathbf{f}^{n+1}, \tag{4.35}$$

(c) At $m = m_f$, the velocity and pressure fields are updated at time-step t^{n+1}:

$$\mathbf{u}^{n+1} = \hat{\mathbf{u}}^{(m_f)}, \tag{4.36}$$

$$p^{n+1} = p^{(m_f)}. \tag{4.37}$$

This new velocity field satisfies the divergence-free constraint $\nabla \cdot \mathbf{u}^{n+1} = 0$.

The current modified PISO algorithm with the DF-IBM for a single simulation time-step is summarised in the flowchart described in Fig. 4.4, as well as in Algorithm 1.

The proposed algorithm is based on the original PISO method of Issa [111], which has the velocity, pressure, and boundary force decoupled, following the operator-splitting strategy of the PISO algorithm, to enforce the no-slip velocity boundary condition on the Lagrangian domain and the divergence-free condition on the Eulerian domain in a segregated manner. Although the no-slip velocity boundary constraint is satisfied before the corrector loops of the PISO algorithm, the Lagrangian interpolated velocity remains primarily the same even if computed after the corrector loops, in accordance with previous findings [41, 81]. Similar to the work of Ji et al. [80], the mPPE differs from the PPE found in IPBM [79] by having the boundary force already satisfying the no-slip velocity boundary condition. In the present case, the divergence-free constraint is only fulfilled by the mPPE allowing the use of fast iterative solvers such as GAMG or Krylov subspace CG solvers without any modification. This also avoids changing the symmetric coefficient matrix and undermining the condition number of the PPE, which leads to convergence problems similar to those encountered by the IPBM [79].

An additional attractive aspect of this current algorithm is the rapid process to achieve a no-slip velocity boundary condition on the Lagrangian domain. This is done by replacing Eq. (4.23) of Uhlmann [39] for computing the Lagrangian weight with Eq. (4.24) from Pinelli et al.[85] for stationary immersed boundary problems and Eq. (4.25) found in [84] for moving immersed boundaries. In the latter case, since the Lagrangian weight is re-computed at each time level, to avoid increasing the computational effort by solving the linear system in Eq. (4.24) at each time-step, the analytical Lagrangian weight computation expressed

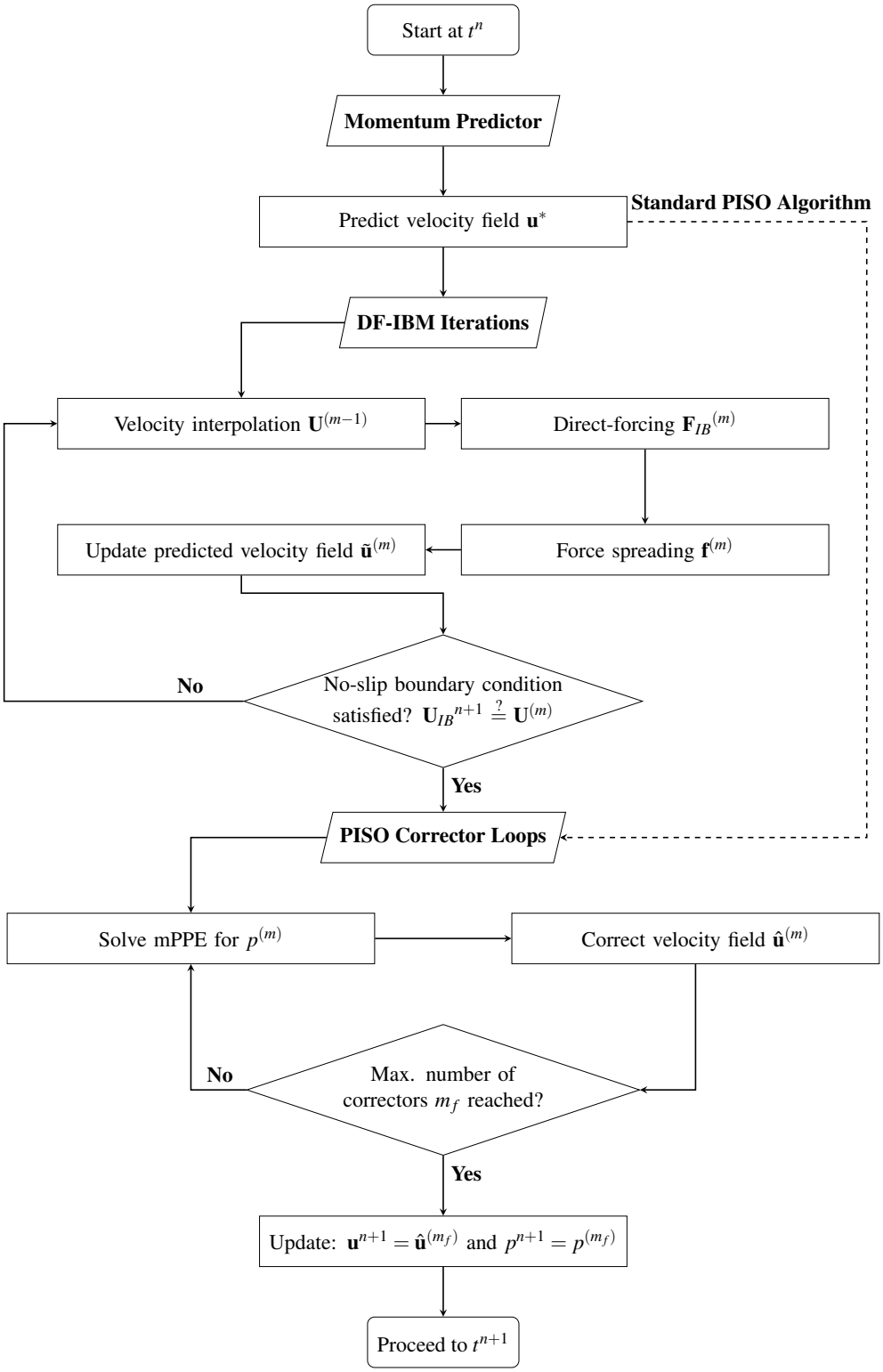


Figure 4.4: *Flowchart of the modified PISO algorithm with the current DF-IBM for one simulation time-step.*

Algorithm 1: Present DF-IBM algorithm.

Given p^n and $\mathbf{u}^n$;
Predict initial Eulerian velocity $\mathbf{u}^*$ using Eq. (4.26);
Initialise: $m = 1$, $\tilde{\mathbf{u}}^{(0)} = \mathbf{u}^*$ and $\mathbf{F}_{IB}{}^{(0)} = \mathbf{F}_{IB}{}^n$;
while $m \leq m_{max}$ **do**
    Interpolate Eulerian velocity to obtain $\mathbf{U}^{(m-1)}$ using Eq. (4.27);
    Compute singular Lagrangian force $\mathbf{F}_{IB}{}^{(m)}$ using Eq. (4.28);
    Spread Lagrangian force to obtain $\mathbf{f}^{(m)}$ using Eq. (4.29);
    Update the Eulerian predicted velocity $\tilde{\mathbf{u}}^{(m)}$ using Eq. (4.30);
    if $\|E_{no-slip}\|_2 < \epsilon_{IBM}$ **then**
        $\mathbf{f}^{n+1} = \mathbf{f}^{(m)}$ and $\tilde{\mathbf{u}} = \tilde{\mathbf{u}}^{(m)}$;
        break;
    else
        $m = m + 1$;
    end
end
Initialise: $m_f = 3$ and $\hat{\mathbf{u}}^{(0)} = \tilde{\mathbf{u}}$;
for $m = 1$ *to* m_f **do**
    Solve mPPE for $p^{(m)}$ using Eq. (4.34);
    Correct Eulerian velocity $\hat{\mathbf{u}}^{(m)}$ using Eq. (4.35);
    if $m = m_f$ **then**
        $\mathbf{u}^{n+1} = \hat{\mathbf{u}}^{(m_f)}$ and $p^{n+1} = p^{(m_f)}$;
    end
end

by Eq. (4.25) is used instead. It was determined through a trial and error process that an additional correction of $3/2$ to Eqs. (4.24) and (4.25) for the Lagrangian weight computation, i.e., by simply multiplying the Lagrangian weight by this factor, was notably speeding up the no-slip velocity boundary convergence while keeping the forces on the immersed boundary intact. Furthermore, the Lagrangian boundary force initialisation method for the term $\mathbf{F}_{IB}{}^{(m-1)}$ used in Eq. (4.28) was also affecting the efficiency of the no-slip velocity boundary convergence. It is therefore proposed here to initialise $\mathbf{F}_{IB}{}^{(0)}$ at the beginning of a new time level by the value from the previous time level, i.e., $\mathbf{F}_{IB}{}^{(0)} = \mathbf{F}_{IB}{}^n$ instead of initialising by zero. Finally, a convergence criterion for the no-slip velocity boundary condition was set for two consecutive DF-IBM iterations, since it was reported in [77], that after multiple iterations the error convergence rate becomes very minimal, hence iterating even further will not affect the desired solution. This significantly reduces the number of iterations per time-step.

4.6 Summary

This chapter presents the general mathematical formulation of the DF-IBM for any fluid–solid system. It encompasses mass and momentum conservation equations in the fluid Eulerian domain, coupled to the solid Lagrangian domain via the addition of a source term in the Navier-Stokes equations referred to as the boundary force term, along with two additional sets of equations known as the IBM-related linear operators. These operators consist of a velocity interpolation operator and a force spreading operator, which manage data mapping between the Lagrangian and Eulerian domains through an interpolation function; the discrete Dirac delta function. A detailed explanation of the no-slip velocity boundary condition, labelled the second fluid–solid system constraint, was provided. This constraint is directly related to the reciprocity between the interpolation and spreading operators; any violation of this reciprocity results in a no-slip velocity boundary error, allowing fluid flow to penetrate the solid immersed boundary.

A unified DF-IBM algorithm is proposed, combining implicit IBM techniques with calibrated Lagrangian weight computations to enforce the second fluid–solid constraint accurately and efficiently on the Lagrangian domain. The improved DF-IBM algorithm is implemented within the predictor-corrector framework of the PISO solution algorithm in OpenFOAM. The dual constraint problem of the fluid–solid system is sequentially enforced at each time-step throughout the solution process. This is achieved by decoupling the pressure, velocity, and boundary force from the Navier-Stokes equations using the operator-splitting strategy of the PISO algorithm. A new modified PPE was developed, having the boundary force already satisfying the second fluid–solid system constraint, preserving a symmetric coefficient matrix, and avoiding any increase in its condition number, compared to the standard PPE. This modification allows the use of fast iterative solvers without additional adjustments. An additional Lagrangian force initialisation scheme is proposed to further boost the algorithm's performance. No additional restriction is imposed on the time-step, except for the stability criterion of the CFL number established by the PISO algorithm (CFL < 1). The overall algorithm comprises three major steps: momentum predictor, DF-IBM iterations, and PISO corrector loops.

The proposed DF-IBM variant is distinguished by its ease of understanding and implementation. The current algorithm can be readily integrated into any incompressible fluid solver that utilises the PISO algorithm or other decoupling algorithms, without significantly increasing computational memory requirements. Only minimal modifications to the original system equations are necessary, specifically in the momentum equations and the PPE.

Chapter 5

Numerical Results: Stationary and Prescribed Motion

Contents

The aim of this chapter is threefold: first, to assess the accuracy, performance, and sensitivity of the proposed method through a series of numerical studies; second, to validate the proposed method by comparing it to experimental and numerical data found in the literature; and third, to demonstrate the capability of this method to handle both circular and non-circular solid immersed boundaries including stationary and prescribed motion, in both translational and rotational dynamics.

5.1 Present Method Analysis

5.1.1 Accuracy Study Using Taylor–Green Decaying Vortices

To validate the temporal and spatial accuracy of the proposed discretisation in the present method, the unsteady Taylor–Green decaying vortices problem is considered in a two-dimensional form. The exact analytical solution of this test problem is given by:

$$u(x,\ y,\ t) = -u_0 \cos\left(\frac{\pi}{L}x\right) \sin\left(\frac{\pi}{L}y\right) e^{-2\left(\frac{\pi}{L}\right)^2 \nu t}, \tag{5.1a}$$

$$v(x,\ y,\ t) = u_0 \sin\left(\frac{\pi}{L}x\right) \cos\left(\frac{\pi}{L}y\right) e^{-2\left(\frac{\pi}{L}\right)^2 \nu t}, \tag{5.1b}$$

$$p(x,\ y,\ t) = p_0 - \frac{{u_0}^2}{4}\left[\cos\left(\frac{2\pi}{L}x\right) + \cos\left(\frac{2\pi}{L}y\right)\right] e^{-4\left(\frac{\pi}{L}\right)^2 \nu t}. \tag{5.1c}$$

In this analysis, the simulation is conducted in a quadrilateral domain $\Omega_f = [-\pi,\ \pi]\ m \times [-\pi,\ \pi]\ m$, with a circular cylinder of unitary radius immersed inside the squared domain and located at its centre $(0,\ 0)$, as shown in Fig. (5.1).

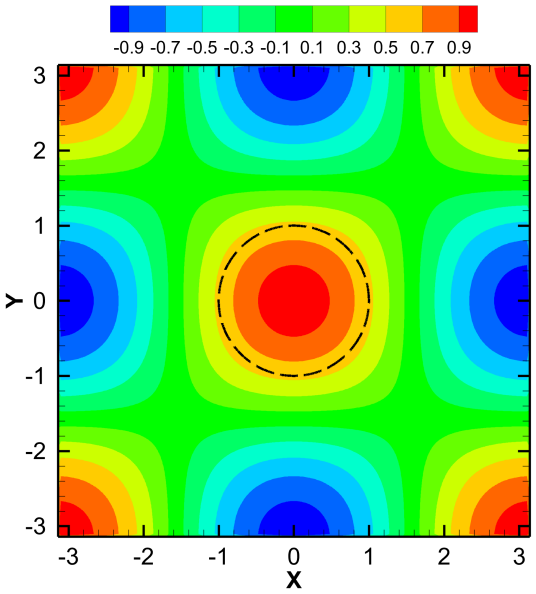


Figure 5.1: *Eulerian domain, vorticity contours, and the cylinder immersed boundary at* $t = 0.34$ *s.*

The initial boundary conditions at $t = 0$ and the time-dependent boundary conditions for the whole Eulerian domain for the velocity and pressure are imposed by Eqs. (5.1a)–(5.1c), and the time-dependent boundary condition, i.e., the desired Lagrangian velocities

at the cylinder immersed boundary, are imposed by Eqs. (5.1a) and (5.1b). The simulation is advanced to a final time of 0.34 s, close to the half-life of the decaying vortices. Periodic boundary conditions are imposed on the Eulerian domain's outer boundaries, p_0 is set to zero, and the $Re = u_0 L/\nu$ is set to π, where $u_0 = 1\ m/s$ is the initial maximum velocity, $L = \pi\ m$ is the vortex size, and $\nu = 1\ m^2/s$ is the kinematic viscosity. The 3-point-width function ϕ_2 is selected.

For the temporal accuracy study, simulations are carried out for different values of the time-step $\Delta t = 0.02, 0.01, 0.005, 0.0025, 0.00125\ s$, with the numerical results compared to a refined solution obtained for a very small time-step of $\Delta t = 0.0001\ s$. A fixed Eulerian grid of 320×320 is set for all simulations to cancel out the error due to the spatial discretisation. The errors of the u-component of the velocity in the x-direction are computed using L_1, L_2, and L_∞ norms as well as the error convergence rate represented by Eqs. (B.1)–(B.4), respectively found in Appendix B. Table 5.1 shows that the convergence rate of the temporal error of the current method is about 2 for all norms, as expected, meaning that the error is decreased by a factor of 4 for all norms when the time-step is divided by 2. The same results are shown in a log-log plot in Fig. (5.2a), and it is worth noting that the same behavior is observed for the v-component of the velocity confirming the second-order temporal accuracy of the present method.

Table 5.1: *Temporal error convergence analysis for the u-component of the velocity of the current solver.*

$\Delta t\ (s)$	$\|E_u\|_1$	Order	$\|E_u\|_2$	Order	$\|E_u\|_\infty$	Order
0.02	2.01×10^{-4}		2.48×10^{-4}		4.92×10^{-4}	
		2.03		2.03		2.02
0.01	4.93×10^{-5}		6.06×10^{-5}		1.20×10^{-4}	
		2.03		2.02		2.00
0.005	1.20×10^{-5}		1.48×10^{-5}		3.01×10^{-5}	
		2.00		2.00		1.90
0.0025	3.00×10^{-6}		3.70×10^{-6}		8.06×10^{-6}	
		1.96		1.90		1.71
0.00125	7.68×10^{-7}		9.92×10^{-7}		2.45×10^{-6}	

For the spatial accuracy study, two separate analyses are carried out, one without the immersed boundary, and another with the inclusion of the immersed boundary since it is known that the diffused DF-IBM deteriorates the overall spatial accuracy of the method due to the discrete Dirac delta function. The analytical solution is compared to numerical solutions for different Eulerian grid sizes $N_x \times N_y = 20 \times 20$, 40×40, 80×80, 160×160, and 320×320. A fine time-step $\Delta t = 0.0001\ s$ is used to ensure that the temporal error is negligible compared to the spatial error. The first analysis without the immersed cylinder is carried out, and the results obtained for the convergence rate of L_1, L_2, and L_∞ norms for the error of the

u-component of the velocity are written in Table 5.2. A second-order accuracy is recorded for the spatial discretisation without the immersed boundary for all norms as expected since the second-order accurate central differencing interpolation scheme is used for the convection term in this case.

Table 5.2: *Spatial error convergence analysis for the u-component of the velocity of the current solver without an immersed boundary.*

$N_x \times N_y$	$\|E_u\|_1$	Order	$\|E_u\|_2$	Order	$\|E_u\|_\infty$	Order
20×20	1.12×10^{-3}		1.39×10^{-3}		2.80×10^{-3}	
		1.98		1.98		2.00
40×40	2.84×10^{-4}		3.50×10^{-4}		7.00×10^{-4}	
		1.99		1.99		1.98
80×80	7.16×10^{-5}		8.83×10^{-5}		1.76×10^{-4}	
		1.99		1.99		1.99
160×160	1.79×10^{-5}		2.21×10^{-5}		4.43×10^{-5}	
		2.00		2.00		1.98
320×320	4.48×10^{-6}		5.53×10^{-6}		1.11×10^{-5}	

The second analysis includes the immersed cylinder where the number of Lagrangian markers representing Γ_s are $N_{Lag} = 20$, 40, 80, 160, and 320 for all Eulerian grid sizes, respectively. The results are presented in Table 5.3, where a similar second-order error convergence rate is detected, which confirms the accuracy of the discrete Dirac delta function in the case of smooth fields as reported in [39]. Fig. (5.2b) shows the error of the u-component of the velocity function of the grid size in a log-log plot for both cases with and without the immersed boundary. It is evident that the immersed boundary introduces error in the original discretisation, but it maintains the same order of accuracy for smooth fields as in the case of the Taylor–Green problem.

Table 5.3: *Spatial error convergence analysis for the u-component of the velocity of the current solver with an immersed boundary.*

$N_x \times N_y$	$\|E_u\|_1$	Order	$\|E_u\|_2$	Order	$\|E_u\|_\infty$	Order
20×20	2.51×10^{-3}		3.71×10^{-3}		1.18×10^{-2}	
		2.01		2.01		1.95
40×40	6.23×10^{-4}		9.18×10^{-4}		3.05×10^{-3}	
		2.02		2.03		1.98
80×80	1.52×10^{-4}		2.24×10^{-4}		7.71×10^{-4}	
		2.01		2.01		1.88
160×160	3.77×10^{-5}		5.54×10^{-5}		2.08×10^{-4}	
		2.00		2.00		1.94
320×320	9.39×10^{-6}		1.37×10^{-5}		5.43×10^{-5}	

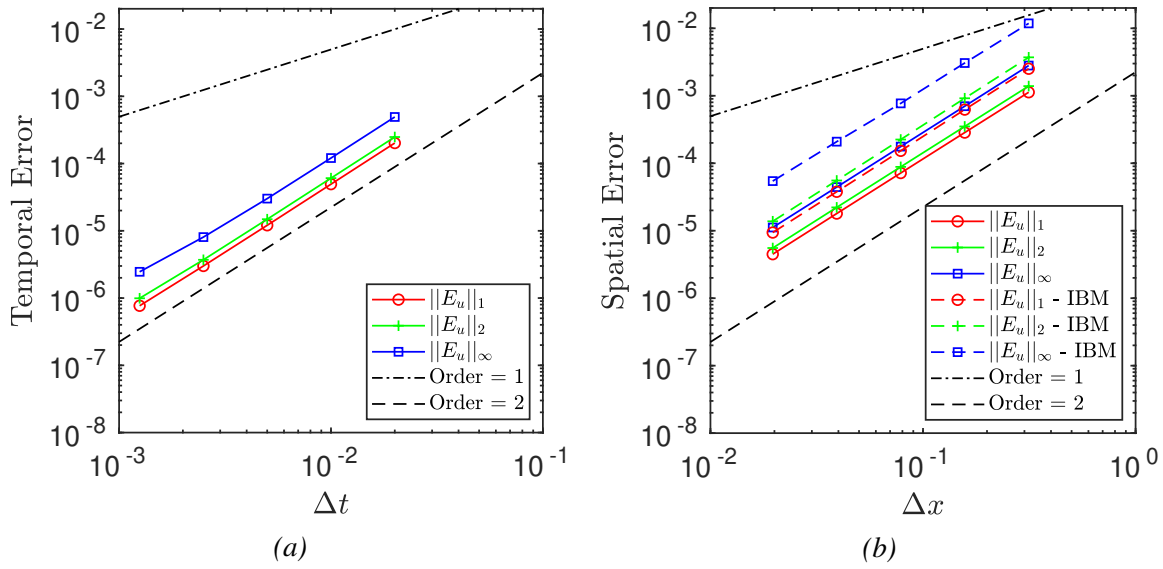


Figure 5.2: *Temporal (a) and spatial (b) accuracy study of the Taylor–Green decaying vortices for the present solver.*

5.1.2 Performance Study of Different DF-IBM Strategies

The performance and efficiency of different DF-IBM strategies are assessed quantitatively and qualitatively against the present strategy where the PISO decoupling algorithm is dropped out for this study, to examine the different DF-IBM sub-routine performance alone for one single execution to satisfy the second fluid–solid system constraint on the Lagrangian domain. The performance is analysed through the following test as in [81]:

1. Initialise velocity field in x-direction:

$$u_0(x,\ y) = e^x \cos(y) - 2 \quad 0 \leq (x,\ y) \leq 1 \quad \text{in } \Omega_f,$$

2. Find f such that

$$U_{IB}(X,\ Y) = \mathscr{I}[\tilde{u}^{(m)}(x,\ y)] = \mathscr{I}[u_0(x,\ y) + \Delta t f^{(m)}] \quad \text{on } \Gamma_s.$$

The following test is conducted over a circular cylinder of radius 0.2 m with its centre located at $(0.52,\ 0.54)$ in an Eulerian squared domain $\Omega_f = [0,\ 1]\ m \times [0,\ 1]\ m$. The cylinder is considered stationary; therefore, $U_{IB} = 0$. The domain is covered by 64×64 Eulerian grid points in the streamwise (x) and transverse (y) directions respectively, with $h = 0.015625\ m$ and the cylinder boundary Γ_s represented by 80 Lagrangian markers where $\Delta s \approx 0.015708\ m$,

Δt is fixed to 1.0 s, and ϕ_2 is the kernel function used. Slip boundary conditions are applied to all domain boundaries.

Because the desired Lagrangian velocity U_{IB} is set to zero for all Lagrangian markers, the no-slip boundary error on the immersed boundary is defined as the computed non-zero velocity at Lagrangian markers. For this analysis only, a stricter convergence criterion is set unlike the one specified in Eq. (4.32) for two successive DF-IBM iterations, which reads

$$\begin{aligned} \left\|E_{no-slip}\right\|_2 &= \left\|E_{\tilde{u}^{(m)}}\right\|_2 \\ &= \left\|U_{IB}(X_n, Y_n) - \mathscr{I}[\tilde{u}^{(m)}(x_{i,j}, y_{i,j})]\right\|_2 < \epsilon_{IBM}, \end{aligned} \tag{5.2}$$

since the updated no-slip boundary error is of interest through the newly updated velocity field and not the error between two successive DF-IBM iterations.

Table 5.4 compares the number of iterations to reach the no-slip boundary constraint, the CPU time to satisfy this constraint, and the no-slip boundary error using L_1, L_2, and L_∞ norms represented by Eqs. (B.1)–(B.4). The tolerance is set to 1×10^{-15} for the implicit DF-IBM strategies using the convergence criterion defined by Eq. (5.2).

Table 5.4: *Performance comparison of different DF-IBM variants in terms of iterations, CPU time, and no-slip boundary error.*

DF-IBM Variant	Explicit/Implicit Strategy	Lagrangian Weight Method	Iterations (m)	CPU Time (s)	$\left\|E_{no-slip}\right\|_1$	$\left\|E_{no-slip}\right\|_2$	$\left\|E_{no-slip}\right\|_\infty$
Uhlmann [39]	Explicit	Eq. (4.23)	1	1.20×10^{-3}	2.77×10^{-1}	3.02×10^{-1}	4.36×10^{-1}
Pinelli et al. [85]	Explicit	Eq. (4.24)	1	1.21×10^{-3}	4.88×10^{-4}	5.58×10^{-4}	1.11×10^{-3}
Cheylan et al. [84]	Explicit	Eq. (4.25)	1	1.17×10^{-3}	2.83×10^{-3}	4.63×10^{-3}	1.65×10^{-2}
Multi-DF-IBM [41, 77, 78]	Explicit	Eq. (4.23)	3	1.70×10^{-3}	6.91×10^{-2}	7.55×10^{-2}	1.17×10^{-1}
Ji et al. [80]	Implicit	Eq. (4.23)	807	1.10×10^{-1}	6.03×10^{-16}	9.98×10^{-16}	2.87×10^{-15}
Present	Implicit	Eq. (4.24)	415	6.48×10^{-2}	5.57×10^{-16}	9.78×10^{-16}	3.19×10^{-15}
Present	Implicit	Eq. (4.25)	402	6.16×10^{-2}	5.52×10^{-16}	9.59×10^{-16}	3.25×10^{-15}
Present	Implicit	Eq. (4.24) + Correction	273	4.46×10^{-2}	5.23×10^{-16}	9.18×10^{-16}	3.10×10^{-15}
Present	Implicit	Eq. (4.25) + Correction	264	4.26×10^{-2}	5.40×10^{-16}	9.36×10^{-16}	3.20×10^{-15}

The explicit DF-IBM of Uhlmann [39], Pinelli et al. [85], and Cheylan et al. [84], all required almost the same CPU time since only one iteration is needed due to their explicit nature, while keeping in mind that the time needed to compute the Lagrangian weight is not included. The Lagrangian weight method using Eq. (4.24) reduced the error for the L_2 norm with respect to Uhlmann's result by a factor of 541 while using Eq. (4.25) reduced the error by a factor of 65. This behavior is expected since these two methods to compute the Lagrangian weight improve the reciprocity of the interpolation and spreading operators of the DF-IBM. The multi-DF-IBM of Luo et al. [77], Kempe and Fröhlich [41], and Breugem [78] apply three iterations in their algorithms using the Lagrangian weight method suggested by

Uhlmann [39] i.e., Eq. (4.23). An error reduction by a factor of 4 is recorded for the L_2 norm. Nevertheless, all these errors obtained using the explicit approach are still considered large.

The implicit strategy proposed by Ji et al. [80] required a total of 807 iterations to drive the error to the prescribed tolerance, where the CPU time increased almost by a factor of 91 compared to Uhlmann's variant. On the other hand, the present implicit DF-IBM variant required 415 and 402 iterations when using Eqs. (4.24) and (4.25) for the computation of the Lagrangian weight respectively, having the CPU time increased by a factor of 54 and 51 compared to Uhlmann's. The analytical correction proposed here increased the CPU time only by a factor of 37 and 35 when using Eqs. (4.24) and (4.25) for the Lagrangian weight respectively, showing a decrease of almost 60% in CPU time with the present implicit DF-IBM when using Lagrangian weight methods that enhance the reciprocity of the linear operators along with the correction proposed, compared to the implicit variant of Ji et al. [80].

It is worth noting that defining the Lagrangian immersed boundary Γ_s using an odd number of Lagrangian markers drastically increased the number of iterations, see [81]. If 81 Lagrangian markers are used to represent the circular cylinder, a total of 3051 iterations would be needed for the implicit method of Ji et al. [80], whereas for the present DF-IBM 1413 iterations would be required to reach the imposed tolerance using Eq. (4.24) for the Lagrangian weight when no correction is included, and 938 iterations when the correction is applied, as shown in Fig. (5.3). Therefore, it is always recommended to use an even number for the immersed boundary Lagrangian representation.

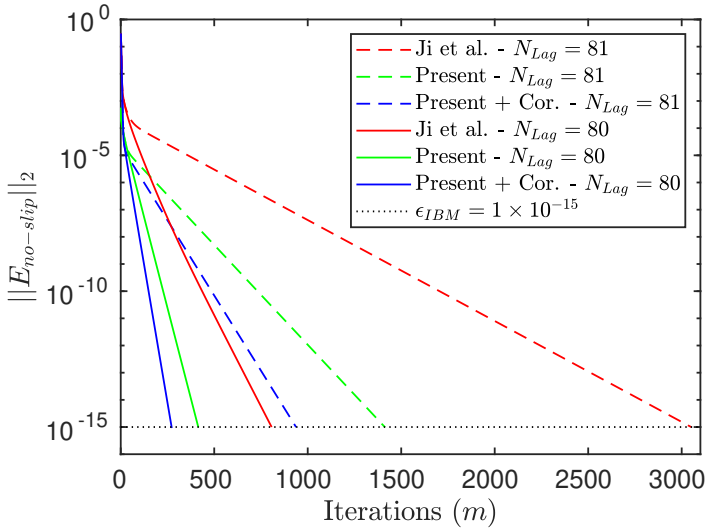
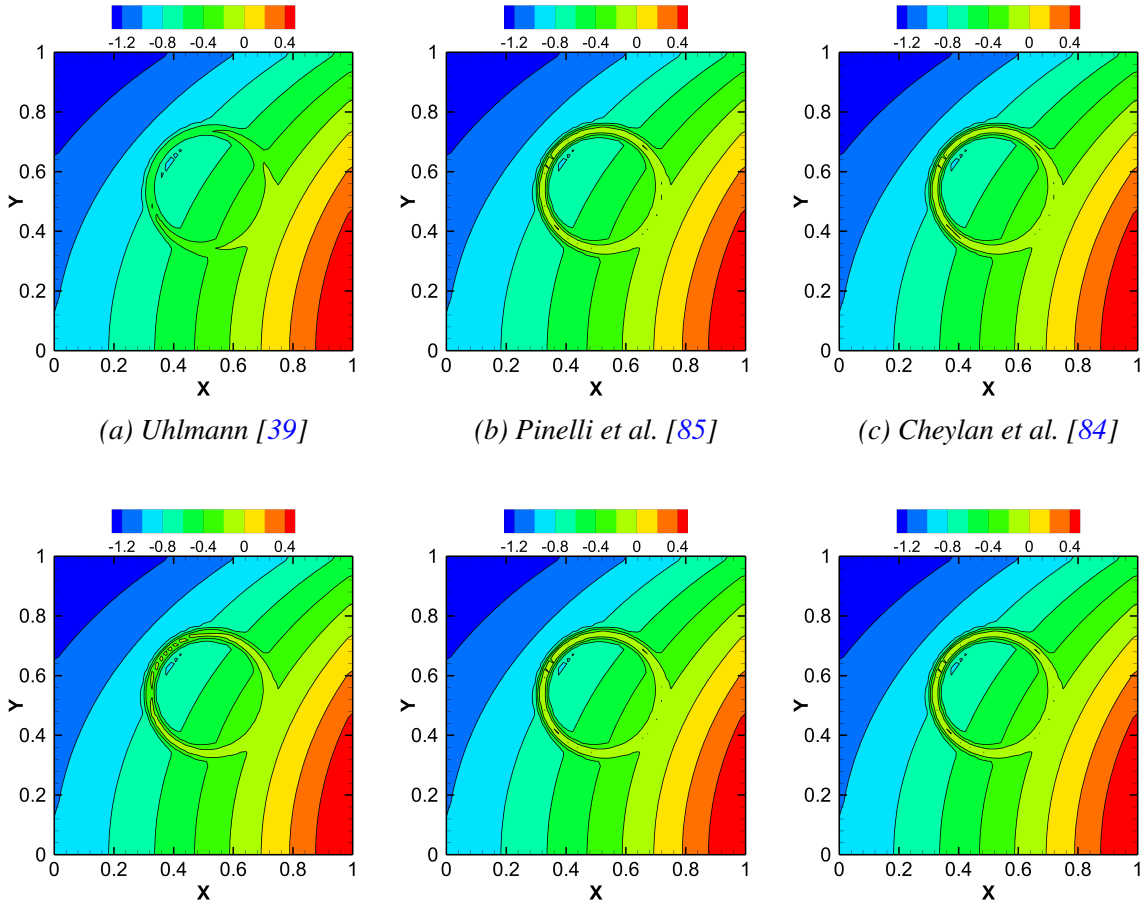


Figure 5.3: *No-slip boundary error against the number of iterations for odd and even number of Lagrangian markers. The results of the present DF-IBM with Eq. (4.24) for the Lagrangian weight method are used for comparison.*

All these results are reflected qualitatively in the velocity contours over the Eulerian domain in Fig. (5.4). Uhlmann's method [39] does not impose accurately the desired Lagrangian velocity on the immersed boundary $(U_{IB} = 0)$, as shown in Fig. (5.4a). A better improvement is noticed for Pinelli et al. [85] and Cheylan et al. [84], as illustrated, respectively, in Figs. (5.4b) and (5.4c), when compared to the multi-DF-IBM variant adopted in [41, 77, 78] as shown in Fig. (5.4d).

Finally, the results of Ji et al. [80] displayed in Fig. (5.4e) and the results for any of the cases shown in Table 5.4 for the present method revealed in Fig. (5.4f) are identical and almost indistinguishable from Figs. (5.4b) and (5.4c) from a qualitative point of view, contrary to the quantitative analysis shown in Table 5.4.



(a) Uhlmann [39] *(b) Pinelli et al. [85]* *(c) Cheylan et al. [84]*

(d) Multi-DF-IBM [41, 77, 78] *(e) Ji et al. [80]* *(f) Present*

Figure 5.4: *Velocity contours for different DF-IBM strategies.*

5.1.3 Sensitivity Study of No-Slip Boundary Error and Flow Penetration

The following investigation differs from the previous one since the latter was used to test the DF-IBM strategies alone for a stationary circular cylinder. Conversely, the forthcoming investigation exploits the proposed DF-IBM inside the PISO algorithm for transient flows to study the no-slip boundary error sensitivity over different parameters and demonstrate the overall improvements made. Thus, this comparative study considers the benchmark problem of the unsteady flow over a stationary circular cylinder.

In this investigation, a circular cylinder of diameter $D = 1\ m$ located at the centre $(0,\ 0)$ of an Eulerian squared domain $\Omega_f = [-2.5,\ 2.5]\ m \times [-2.5,\ 2.5]\ m$ is considered, covered by 250×250 Eulerian grid points, with $h = 0.02\ m$. The cylinder is stationary, i.e., $\mathbf{U}_{IB}$ is set to 0 in all directions, and is formed by 156 Lagrangian markers uniformly distributed over Γ_s, with $\Delta s \approx 0.020138\ m$. $Re = u_\infty D/\nu$ is set to 100, with the inlet freestream velocity $u_\infty = 1\ m/s$ and the kinematic viscosity ν equal to $0.01\ m^2/s$. For the velocity, a Dirichlet boundary condition $(u_\infty = 1\ m/s,\ v_\infty = 0)$ is set at the inflow boundary, and a non-reflective advective boundary condition $(\partial \mathbf{u}/\partial t + u_n \partial \mathbf{u}/\partial \mathbf{n} = 0)$ is applied at the outflow boundary with u_n being the advection velocity, i.e., the component of the velocity normal to the boundary. Slip boundary conditions $(\partial u/\partial y,\ v = 0)$ are set to the lateral sides of the domain. For the pressure, Neumann boundary conditions $(\partial p/\partial \mathbf{n} = 0)$ are applied on the inlet and lateral sides of the Eulerian domain, while a zero pressure $(p = 0)$ is specified at the outflow boundary, as shown in Fig. (5.11a). The time-step is set to $\Delta t = 0.004\ s$ leading to an initial $CFL_{ini} = 0.2$ for all simulations. The time is advanced until the non-dimensional time $\frac{tu_\infty}{D}$ reaches a value of 10. The tolerance for Eq. (4.32) for two successive DF-IBM iterations is set to 1×10^{-8} and ϕ_2 is selected as the kernel function.

Lagrangian Force Initialisation Effect

First, the Lagrangian force $\mathbf{F}_{IB}{}^{(m-1)}$ found in Eq. (4.28) of the current DF-IBM algorithm should be initialised at the start of the iterations, i.e., $\mathbf{F}_{IB}{}^{(0)}$ for $m = 1$. This initialisation process can be handled using two different schemes. The first is done by static initialisation, i.e., by setting the variable $\mathbf{F}_{IB}{}^{(0)} = 0$ at the beginning of the DF-IBM iterations when $m = 1$, implying that this variable acts as a constant throughout the whole simulation run time. The other approach is achieved through dynamic initialisation of the variable, which means that the variable at the current time level is assigned a value from the previous time level during the simulation run time, $\mathbf{F}_{IB}{}^{(0)} = \mathbf{F}_{IB}{}^{n}$, ergo the value of this variable is altered at every

time-step.

Fig. (5.5a) shows the no-slip boundary error in terms of L_2 norm, as computed in Eq. (4.32) against the non-dimensional time. Fig. (5.5b) shows the number of iterations to reach the tolerance for two successive DF-IBM iterations against the non-dimensional time. It appears clearly that the dynamic initialisation scheme speeds up the calculations to reach the prescribed tolerance, by reducing the number of DF-IBM iterations from 414 for the static initialisation case to 108 for the dynamic initialisation case at $\frac{tu_\infty}{D} = 10$. In addition, the no-slip error is reduced by one order of magnitude from 3.81×10^{-6} for the static case to 4.74×10^{-7} for the dynamic case. Moreover, an asymptotic behavior for both the number of iterations and the no-slip boundary error against the non-dimensional time can be observed, where a slight change occurred from $\frac{tu_\infty}{D} = 1$, with an error of 3.82×10^{-6} corresponding to 483 iterations, to $\frac{tu_\infty}{D} = 10$, with a recorded error of 3.81×10^{-6} and 414 iterations for the static case.

Similarly for the dynamic case, the error went from 4.87×10^{-7} to 4.74×10^{-7} and the number of iterations decreased from 114 at $\frac{tu_\infty}{D} = 1$ to 108 at $\frac{tu_\infty}{D} = 10$. The dynamic approach will be used in the remaining part of this investigation. It is also worth mentioning that the aerodynamic forces are identical for both initialisation schemes.

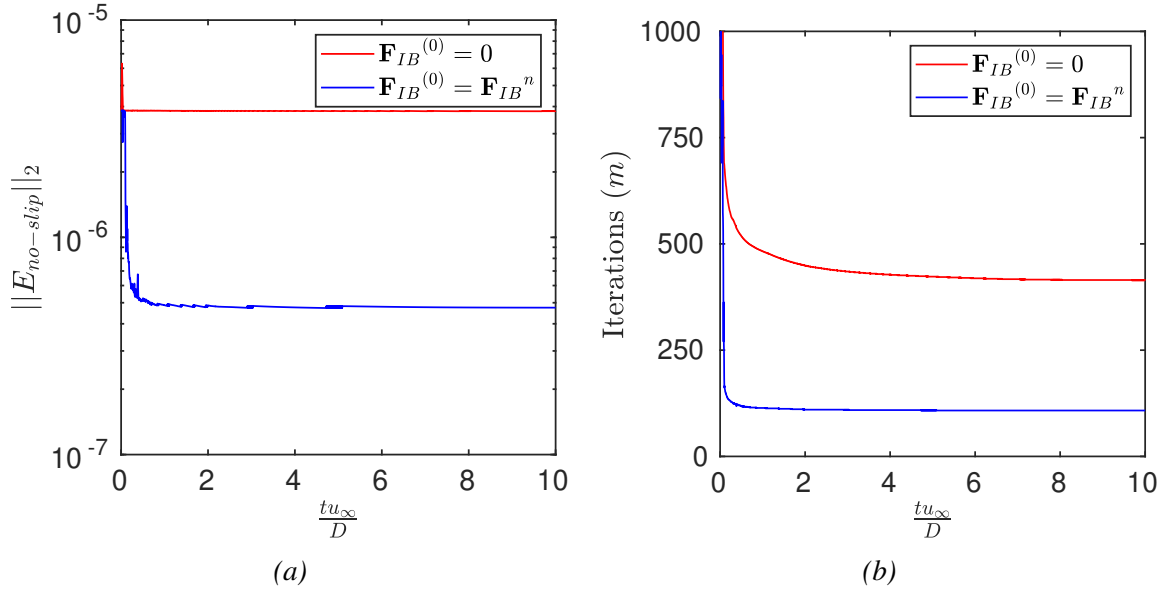


Figure 5.5: *Effect of the Lagrangian force initialisation approach on the no-slip boundary error (a) and the number of implicit iterations (b).*

Lagrangian Weight Method and Lagrangian Weight Correction Effect

The correlation between the Lagrangian weight method, the Lagrangian weight correction, and the no-slip boundary error is illustrated in Fig. (5.6). The no-slip boundary error in the L_2 norm following Eq. (4.32) is not heavily affected by the different methods to compute the Lagrangian weight and the inclusion of the proposed correction, since a minor decrease is observed for the cases using Eqs. (4.24) and (4.25) compared to Eq. (4.23). The number of iterations at $\frac{tu_\infty}{D} = 10$ decreased from 114 iterations in the case of W_n being computed according to Uhlmann [39] using Eq. (4.23) to approximately 86 iterations when using Eqs. (4.24) and (4.25). This number is further decreased to roughly 70 iterations when the analytical correction is applied. The difference between the results of Eqs. (4.24) and (4.25) is barely distinguishable when no correction is used, similar to the case where the correction is applied. This decrease of almost 40% of the number of iterations per time-step to reach the prescribed tolerance on two successive iterations by employing the proposed correction on the Lagrangian weight Eqs. (4.24) and (4.25) compared to Eq. (4.23) will reflect a large decrease in the CPU time to reach the simulation end time.

Similarly to the previous investigation, an asymptotic behavior is observed for both the no-slip error and the number of iterations, and the aerodynamic forces are also identical for all Lagrangian weight methods, with or without correction.

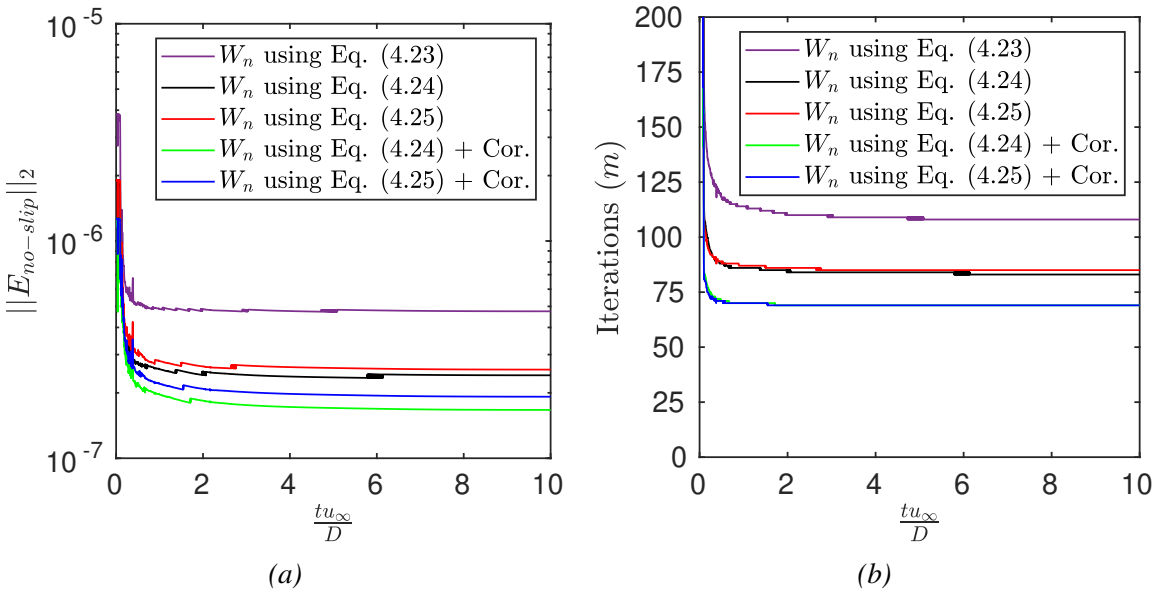


Figure 5.6: *Effect of the Lagrangian weight method and the Lagrangian weight correction on the no-slip boundary error (a) and the number of implicit iterations (b).*

Time-Step Effect

A final investigation on the effect of the simulation time-step, i.e., the initial CFL number, on the no-slip boundary error and the number of iterations is presented in Fig. (5.7). Different values of the time-step are selected, $\Delta t = 0.01$, 0.008, 0.006, 0.004, and 0.002 s, leading to an initial Courant number of $CFL_{ini} = 0.5$, 0.4, 0.3, 0.2, and 0.1, respectively. The no-slip boundary error and the number of iterations are reduced with decreasing values of the CFL number, in agreement with previous findings in the literature [41, 83]. Unfortunately, manipulating the time-step to reduce the no-slip boundary error is not considered the best approach, since this will drastically increase the CPU time to reach the simulation end time, once Δt is set to a very small value.

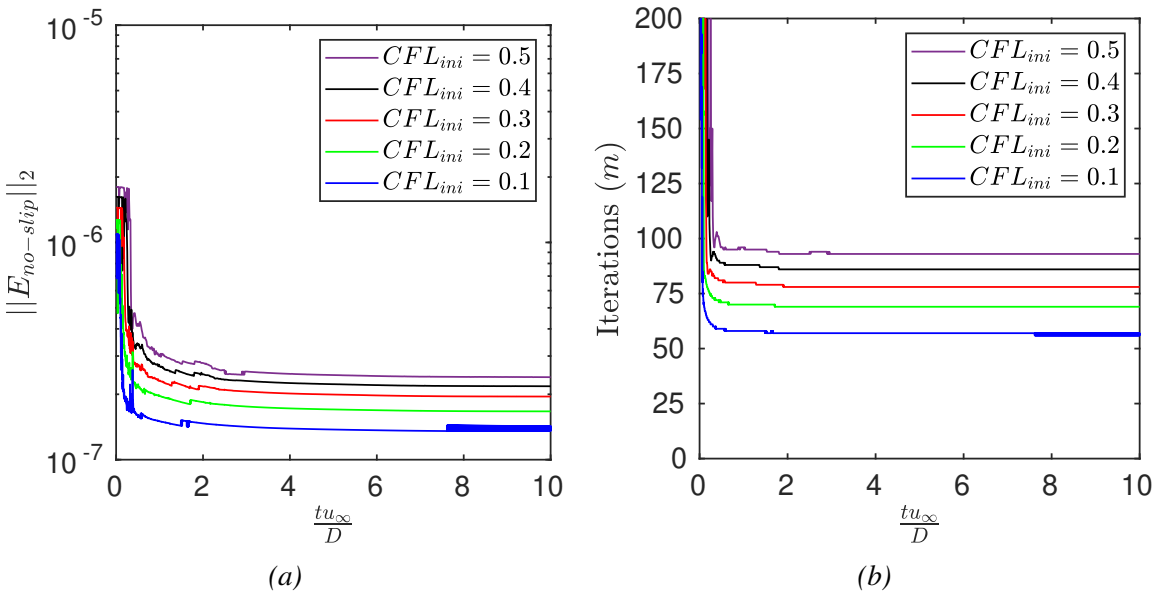


Figure 5.7: *Effect of the simulation time-step on the no-slip boundary error (a) and the number of implicit iterations (b).*

5.2 Stationary Cases

5.2.1 Lid-Driven Cavity With an Embedded Cylinder at Re = 1000

The flow developed inside a square lid-driven cavity with an embedded cylinder placed at the centre is considered. The cavity length is taken as $L = 1\ m$ and the cylinder diameter is set equal to $0.4L$. The boundary conditions correspond to the classical lid-driven cavity problem, with a Dirichlet boundary condition specified at the top $(u_\infty = 1\ m/s,\ v_\infty = 0)$ and a wall (no-slip) boundary condition specified for the remaining sides $(u = 0,\ v = 0)$ as shown in Fig. (5.8). The kinematic viscosity ν is set equal to $0.001\ m^2/s$, leading to a Reynolds number $Re = u_\infty L/\nu$ based on the cavity length equal to 1000. A steady-state solution governs the flow as the simulation time is advanced, with a time-step chosen equal to $\Delta t = 0.001\ s$. A uniform Eulerian mesh covered by 256×256 with $h = 3.90625 \times 10^{-3}\ m$ is used similar to the work of [125]. The cylinder being stationary is formed by 320 Lagrangian markers. The 3-point-width kernel function ϕ_2 is selected as the kernel function.

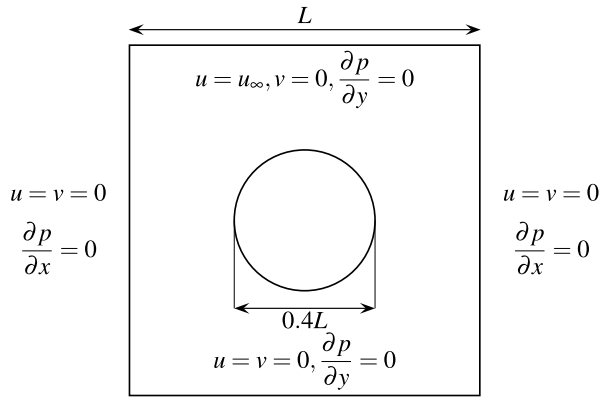
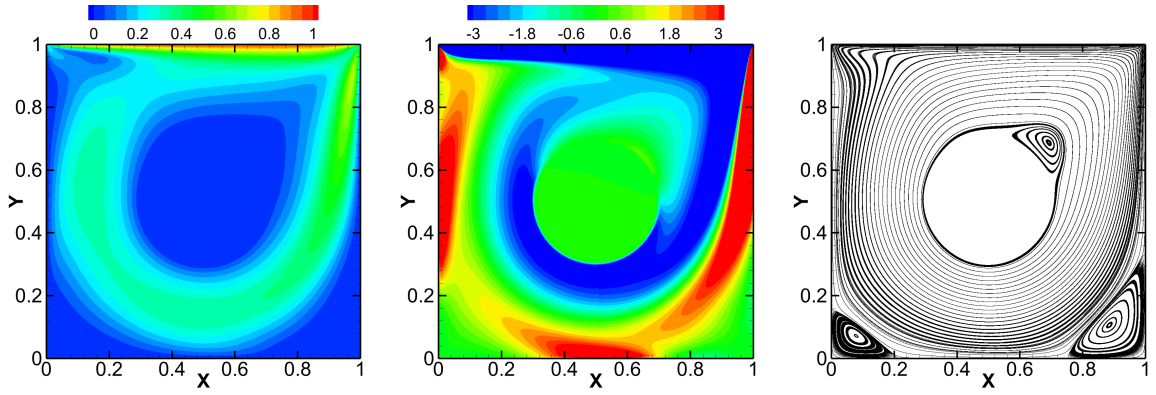


Figure 5.8: *Computational domain and boundary conditions for lid-driven cavity flow with an embedded cylinder.*

The contours of velocity magnitude and vorticity are shown in Figs. (5.9a) and (5.9b) respectively. The results are very similar to the previous works of [81, 125]. The streamlines are illustrated in Fig. (5.9c). It is evident that three vortices emerged in the flow, located at the bottom left and right corners, and the third one located on the top right of the embedded cylinder. The location of the centre of each vortex is compared in Table 5.5 with the data reported in [81]. A very good agreement is obtained.

A quantitative comparison for the velocity profiles is conducted with previous IBM data presented in [81, 125], for the velocity components $(u,\ v)$ in the vertical $(x = 0.5)$ and horizontal $(y = 0.5)$ centrelines of the cavity. The data obtained is shown in Fig. (5.10).

Excellent agreement with previous studies is again obtained.

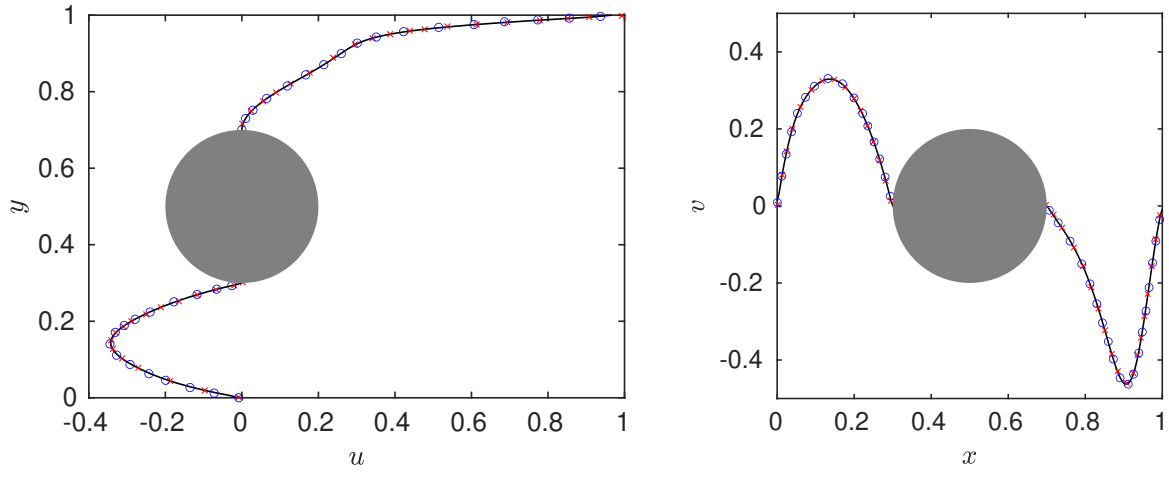


(a) *(b)* *(c)*

Figure 5.9: *Velocity magnitude contours (a), vorticity contours (b), and streamlines (c) for the lid-driven cavity with the embedded cylinder at* $Re = 1000$.

Table 5.5: *Comparison of the vortices centre location for the lid-driven cavity with embedded cylinder case for* $Re = 1000$. *The vortices centre* $(x_1,\ y_1)$, $(x_2,\ y_2)$, *and* $(x_3,\ y_3)$ *correspond to the bottom left corner, bottom right corner, and top right of the cylinder respectively.*

	$(x_1,\ y_1)$	$(x_2,\ y_2)$	$(x_3,\ y_3)$
Cai et al. (2017) [81]	$(0.0789, 0.0720)$	$(0.8852, 0.1063)$	$(0.6942, 0.6881)$
Present	$(0.0787, 0.0717)$	$(0.8847, 0.1062)$	$(0.6921, 0.6873)$



(a) *(b)*

Figure 5.10: *Velocity u-component along* $x = 0.5$ *(a) and velocity v-component along* $y = 0.5$ *(b) for the lid-driven cavity with the embedded cylinder at* $Re = 1000$. *Solid lines (—) correspond to the present results, (∘) symbols correspond to the data reported in [125], and (×) symbols correspond to the data reported in [81].*

5.2.2 Flow Over a Stationary Cylinder at Re = 40, 100, 300, and 1000

The current algorithm is validated against well-established experimental and numerical results for the benchmark problem of the steady and unsteady flows over a stationary circular cylinder, over a wide range of Reynolds numbers $Re = u_\infty D/\nu$, with u_∞ the freestream inlet velocity, D the cylinder diameter, and ν the kinematic viscosity. The cylinder diameter and inlet velocity are fixed to 1, while the kinematic viscosity is varied to obtain the simulated Re numbers, i.e., 40, 100, 300, and 1000. Two Eulerian squared domains are used for the current simulations, $\Omega_{f_1} = 40D \times 40D = [-15,\ 25] \times [-20,\ 20]$ and an enlarged one $\Omega_{f_2} = 60D \times 60D = [-20,\ 40] \times [-30,\ 30]$, to test the finite lateral domain boundaries confinement effect. The challenge is to reduce the computational time, especially when Re is set equal to 1000. Considering that only the region in the vicinity of the immersed boundary needs to be surrounded by Cartesian uniform cells, for the IBM-related operators, a good strategy would be to avoid meshing the whole computational domain with Cartesian uniform cells, and create only a region enclosing the cylinder's immersed boundary with a resolution equal to the Lagrangian markers spacing Δs. This enables the reduction of the mesh size in the far-field regions of the Eulerian domain. Hence, a refined grid for every Re number is adopted in the region $[-5,\ 5] \times [-5,\ 5]$, enclosing the cylinder. The Eulerian grid resolution is shown in Table 5.6.

Table 5.6: *Eulerian and Lagrangian grids resolution for the flow over a stationary cylinder at different Re.*

Re	$h\ (m)$	$\Delta s\ (m)$	$\Delta s/h$	N_{Lag}	$\Delta t\ (s)$
40	0.03	0.030207	1.006920	104	0.006
100	0.03	0.030207	1.006920	104	0.006
300	0.02	0.020138	1.006920	156	0.004
1000	0.01	0.010069	1.006920	312	0.002

The same boundary conditions as for the previous problem are used here as shown in Fig. (5.11a). The cylinder centre is located at $(0,0)$ where the numbers of uniformly distributed Lagrangian markers forming Γ_s for every Re are shown in Table 5.6 with $\Delta s/h \approx 1$ for all conducted simulations. A time-step is selected according to Table 5.6 leading to a maximum Courant number $CFL_{max} \approx 0.911$. The 3-point-width kernel function ϕ_2 is selected for the data mapping between the two grids.

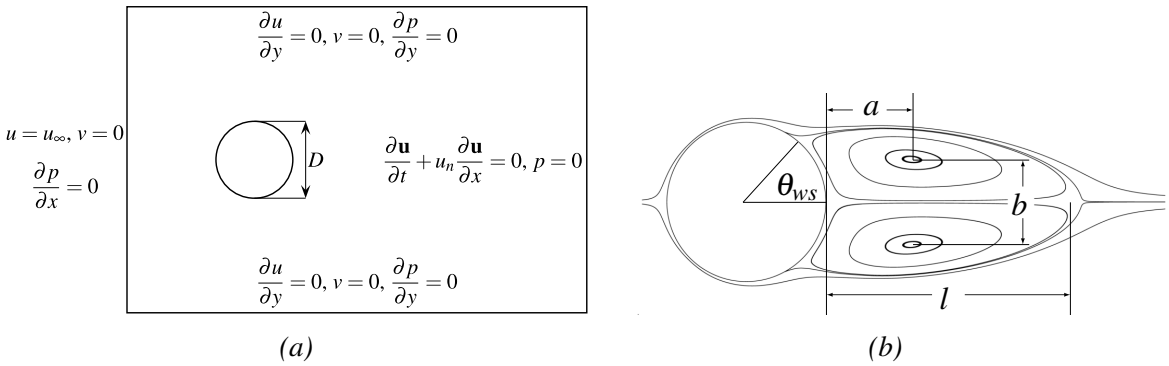


Figure 5.11: *Computational domain and boundary conditions (a) and wake nomenclature (b) for flow over a stationary cylinder.*

The instantaneous non-dimensional force coefficients and Strouhal number are defined as:

$$\text{Drag coefficient: } C_D = \frac{F_D}{1/2\rho D H {u_\infty}^2}, \tag{5.3a}$$

$$\text{Lift coefficient: } C_L = \frac{F_L}{1/2\rho D H {u_\infty}^2}, \tag{5.3b}$$

$$\text{Strouhal number: Str} = \frac{f_s D}{u_\infty}, \tag{5.3c}$$

where the fluid density ρ is set to 1 kg/m^3, f_s is the natural vortex shedding frequency expressed in $(1/s)$, and (F_D, F_L) are the respective drag and lift forces exerted on the cylinder's immersed boundary by the fluid expressed in Newtons, keeping in mind that the third dimension H is taken equal to 1 since a 2D case is considered. They are computed by summing the singular Lagrangian forces located on each Lagrangian marker:

$$\begin{pmatrix} F_D \\ F_L \end{pmatrix} = -\rho \sum_{n=1}^{N_{Lag}} \mathbf{F}_{IB}(\mathbf{X}_n, t) W_n. \tag{5.4}$$

For $Re = 40$, the flow pattern is expected to be symmetric and steady, where a recirculation region is formed in the wake of the cylinder, as described in Fig. (5.11b), with the following dimensions in (m): l, the wake streamwise length, a, the horizontal distance from the cylinder boundary to the wake vortex centre, b the vertical distance between the centre of the two wake vortices, and θ_{ws} $(^\circ)$ the separation angle measured from the x-axis. The obtained pressure contours, vorticity contours, and streamlines for the steady flow at one time-instant are displayed in Fig. (5.12). It can be noted that the flow profile closely matches

those documented in the literature. In addition, the symmetric wake dimensions are compared with previous experimental and numerical results, see Table 5.7. As can be seen, the findings are consistent with the literature. The drag and lift coefficients are plotted in Fig. (5.16a), and the steady-state drag value is also compared with values extracted from the literature in Table 5.7. It is worth mentioning that by using a larger domain, i.e., by increasing the lateral dimensions of the computational domain to minimise the confinement effect of the lateral boundaries, a reduction of roughly 1.2% is recorded on the drag coefficient value.

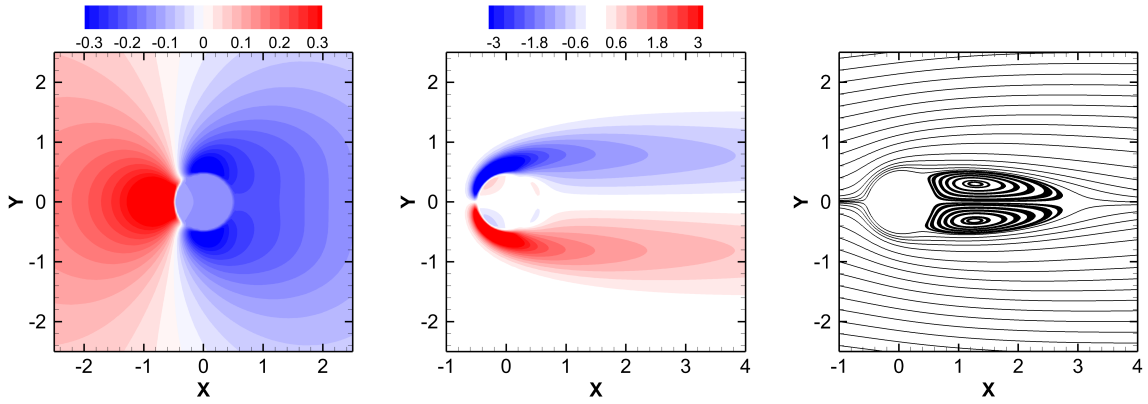


Figure 5.12: *Instantaneous pressure contours (left), vorticity contours (middle), and streamlines (right) for the flow over a stationary cylinder at* $Re = 40$.

Table 5.7: *Comparison of the wake dimensions and the mean drag coefficient for the flow over a stationary cylinder at* $Re = 40$.

References	l/D	a/D	b/D	$\theta_{ws}(^\circ)$	$\overline{C_D}$
Tritton [129]†	–	–	–	–	1.585
Coutanceau & Bouard [130]†	2.13	0.76	0.59	53.5	–
Russell & Wang [131]	2.29	–	–	53.1	1.600
Linnick & Fasel [132]	2.23	0.71	0.59	53.4	1.610
Xu & Wang [133]	2.21	–	–	53.5	1.660
Taira & Colonius [79]	2.33	0.75	0.60	54.1	1.550
Wang & Zhang [128]	2.36	0.72	0.60	53.8	1.540
Cai et al. [81]	2.36	0.75	0.62	52.1	1.560
Present ($\Omega_{f_1} = 40D \times 40D$)	2.28	0.78	0.61	52.6	1.568
Present ($\Omega_{f_2} = 60D \times 60D$)	2.24	0.78	0.61	52.0	1.550

† Experimental results.

At $Re = 100$, the flow pattern becomes unsteady and exhibits the famous von Kármán vortex street, also known as periodic vortex shedding. The pressure and vorticity contours are shown in Fig. (5.13). The vortex shedding leads to time-varying fluctuations of the

drag and lift coefficients, and they can be well observed in Fig. (5.16b). Different force quantities obtained for the current simulations including the mean drag coefficient, the drag and lift fluctuations, and the Strouhal number, are compared in Table 5.8 against different experimental and numerical results reported in the literature for this Re number. These quantities are computed after a periodic state has been achieved by the vortex shedding. A close agreement can be found when compared with previous results. Similar to the steady-state smaller $Re = 40$, a reduction of almost 1.7% is obtained on the mean drag coefficient by enlarging the Eulerian domain lateral dimensions, along with a decrease of 3% for the Strouhal number.

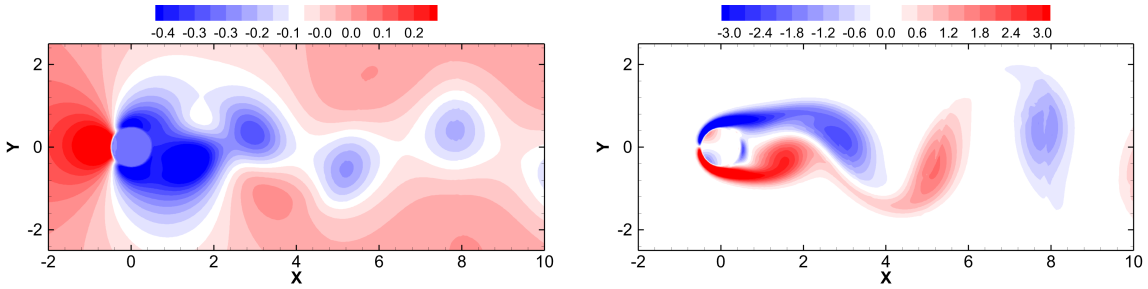


Figure 5.13: *Instantaneous pressure contours (left) and vorticity contours (right) for the flow over a stationary cylinder at* $Re = 100$.

Table 5.8: *Comparison of the mean drag coefficient, the drag and lift fluctuations amplitude, and the Strouhal number for the flow over a stationary cylinder at* $Re = 100$.

References	$\overline{C_D}$	C'_D	C'_L	Str
Williamson [134]†	–	–	–	0.166
Braza et al. [135]	1.364	±0.015	±0.250	0.160
Lai & Peskin [35]	1.447	–	±0.329	0.165
Uhlmann [39]	1.453	±0.011	±0.339	0.169
Yang et al. [86]	1.393	–	±0.335	0.165
Ji et al. [80]	1.402	±0.010	±0.349	0.167
Mimeau et al. [136]	1.400	±0.010	±0.320	0.165
Cai et al. [81]	1.366	±0.010	±0.342	0.160
Cheylan et al. [84]	1.408	–	±0.330	0.166
Present ($\Omega_{f_1} = 40D \times 40D$)	1.411	±0.012	±0.371	0.165
Present ($\Omega_{f_2} = 60D \times 60D$)	1.387	±0.011	±0.359	0.160

† Experimental results.

The periodic von Kármán vortex shedding is still present at $Re = 300$, see Fig. (5.14), along with the pressure contours. At this Re and higher values, the numerical results are

expected to deviate from experimental values, due to the intrinsic three-dimensional effects of the flow around the cylinder [137] leading to inaccurate prediction of the force coefficients for two-dimensional numerical simulations. The time evolution of the drag and lift coefficients caused by the vortex shedding is plotted in Fig. (5.16c). Table 5.9 compares the corresponding mean drag coefficient, the rms of the lift fluctuations, and the Strouhal number for the current solver against various two-dimensional simulation results extracted from the literature. The mean value of the drag coefficient for all six references is equal to 1.384, leading to a discrepancy of about 8.45% and 7.22% with the present computed result for Ω_{f_1} and Ω_{f_2}, respectively. The mean Strouhal number for all mentioned references is approximately equal to 0.211, resulting in a difference of only 0.47% with the presently calculated Strouhal number, which remained intact for different dimensions of the domain.

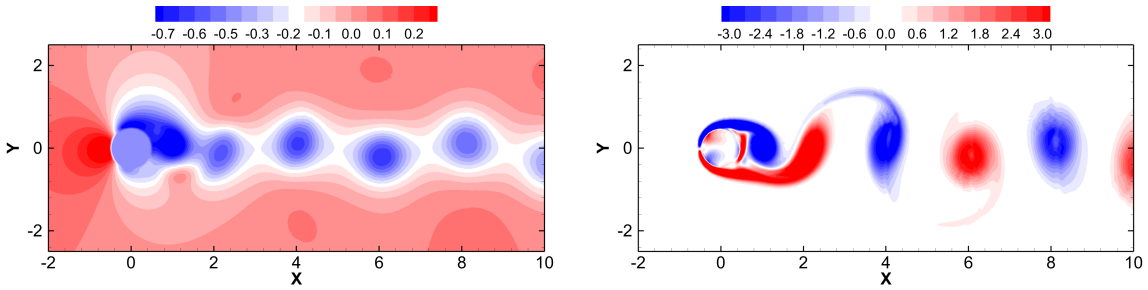


Figure 5.14: *Instantaneous pressure contours (left) and vorticity contours (right) for the flow over a stationary cylinder at* $Re = 300$.

Table 5.9: *Comparison of the mean drag coefficient, the root mean square of the lift fluctuations, and the Strouhal number for the flow over a stationary cylinder at* $Re = 300$.

References	$\overline{C_D}$	$(C_L)_{rms}$	Str
Henderson [138]	1.370	–	–
Mittal & Balachandar [137]	1.381	0.652	0.213
Persillon & Braza [139]	1.405	0.526	0.209
Ye et al. [44]	1.380	–	0.210
Mittal et al. [49]	1.360	–	0.210
Apte et al. [140]	1.410	–	0.212
Present ($\Omega_{f_1} = 40D \times 40D$)	1.501	0.691	0.210
Present ($\Omega_{f_2} = 60D \times 60D$)	1.484	0.680	0.210

At $Re = 1000$, the flow regime becomes dominant by convection effects, leading to a decrease in the boundary layer thickness, expressed as in [136], by $\delta \approx D/\sqrt{Re} \approx 0.0316\ m$. Hence, to marginally capture the thin boundary layer, three Eulerian grid cells are considered inside the boundary layer for this case, leading to an Eulerian grid resolution of $h = 0.01\ m$ in

Table 5.6, similarly to previous research [49, 140]. Despite the boundary layer being partially resolved, the flow characteristics are well depicted, see Fig. (5.15) for the instantaneous pressure and vorticity fields. The time evolution of the drag and lift coefficients are plotted in Fig. (5.16d). The two-dimensional mean drag coefficient, the drag and lift fluctuations, and the Strouhal number are compared against the numerical results reported in the literature in Table 5.10. A difference of 5.26% is recorded for the drag coefficient when compared with the average value of all literature values, whereas a 4.84% deviation is noted for the Strouhal number.

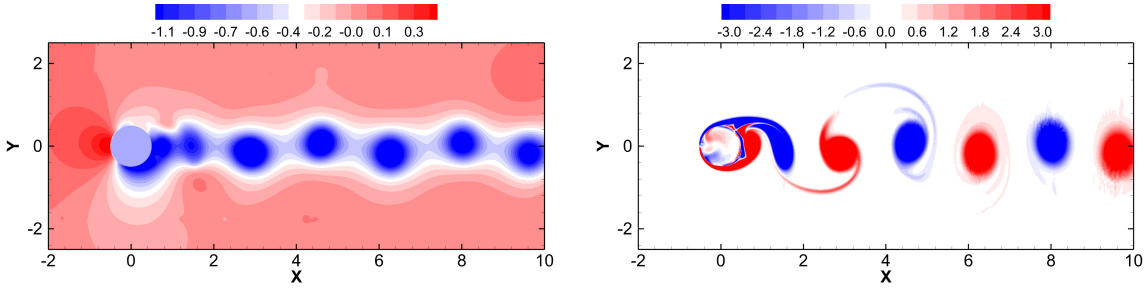
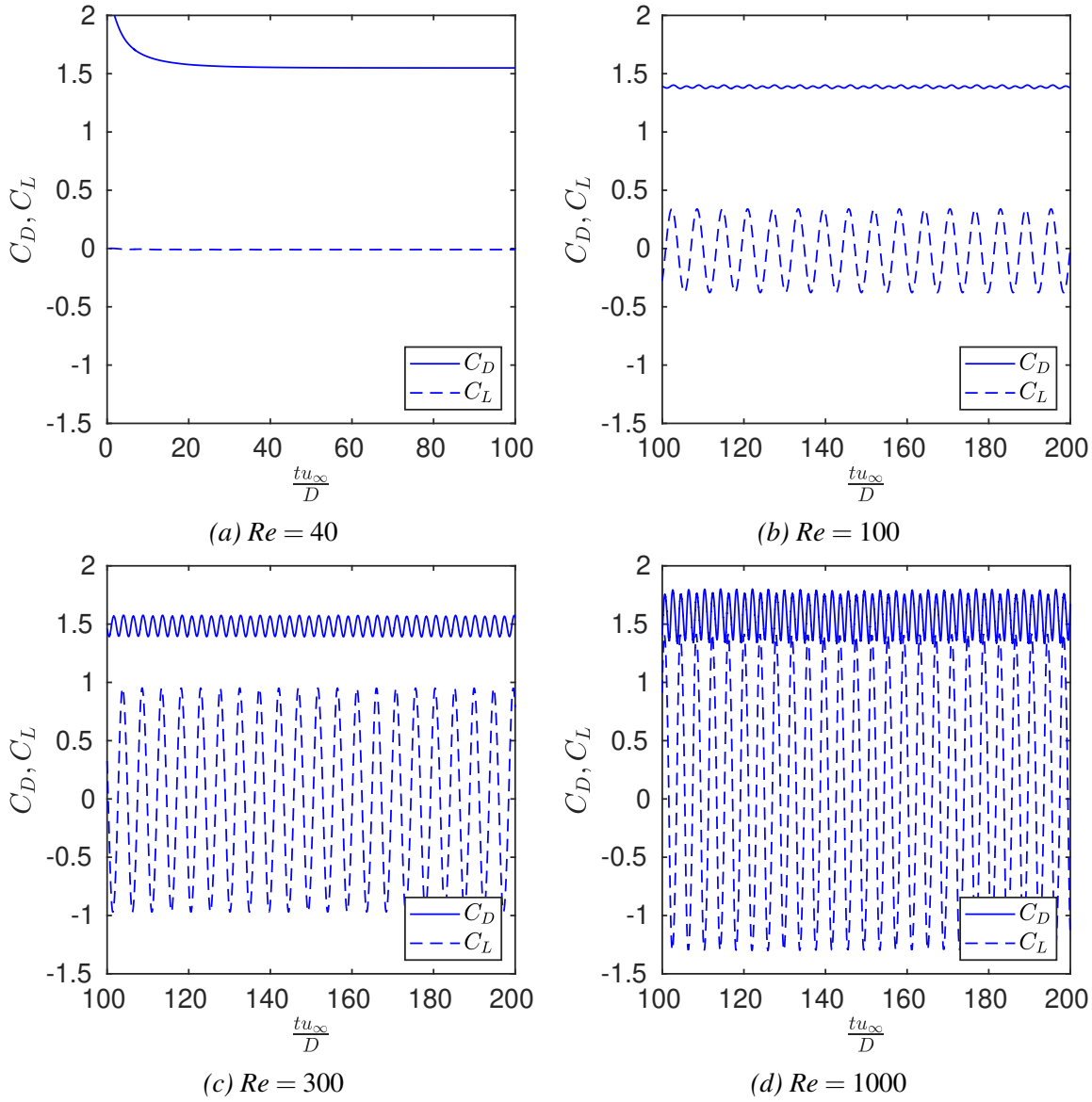


Figure 5.15: *Instantaneous pressure contours (left) and vorticity contours (right) for the flow over a stationary cylinder at* $Re = 1000$.

Table 5.10: *Comparison of the mean drag coefficient, the drag and lift fluctuations amplitude, and the Strouhal number for the flow over a stationary cylinder at* $Re = 1000$.

References	$\overline{C_D}$	C'_D	C'_L	Str
Henderson [138]	1.510	–	–	–
Mittal & Kumar [141]	1.480	$\pm$0.210	$\pm$1.360	0.250
Mittal et al. [49]	1.450	–	–	–
Apte et al. [140]	1.500	–	–	0.238
Mimeau et al. [136]	1.510	$\pm$0.230	$\pm$1.540	0.245
Cai et al. [81]	1.550	$\pm$0.220	$\pm$1.460	0.240
Present ($\Omega_{f_1} = 40D \times 40D$)	1.579	$\pm$0.227	$\pm$1.353	0.255

Overall, the present solver has demonstrated strong agreement with experimental results and other numerical simulations, further validating and confirming its accuracy across a relatively broad range of Reynolds numbers.



(a) $Re = 40$ *(b)* $Re = 100$

(c) $Re = 300$ *(d)* $Re = 1000$

Figure 5.16: *Temporal evolution of the drag C_D and lift C_L coefficients for the flow over a stationary cylinder at different Re.*

5.3 Prescribed Motions

5.3.1 Flow Over an In-Line Oscillating Cylinder in a Quiescent Fluid at Re = 100

The flow produced by the periodic streamwise oscillation of a circular cylinder is now simulated using the present method. In this test case, the fluid is considered stagnant, and the in-line translational motion of the cylinder is prescribed by the following equation:

$$x(t) = -A_0 \sin(2\pi f t), \tag{5.5}$$

where $x\ (m)$ is the location of the cylinder's centre in the longitudinal direction, $A_0\ (m)$ is the amplitude of the oscillation, and $f\ (1/s)$ is the characteristic frequency.

This flow is characterised by the Reynolds number defined as $Re = U_{max}D/\nu$, and the Keulegan-Carpenter number $KC = U_{max}/fD$, where $U_{max} = 2\pi A_0 f$ is the maximum velocity of the cylinder during oscillation, D is the cylinder diameter, and ν is the kinematic viscosity. The computations are therefore performed at $Re = 100$ and $KC = 5$ for this test case, with $D = 1\ m$, $\nu = 0.01\ m^2/s$, $U_{max} = 1\ m/s$, $A_0 = 2.5/\pi\ m$, and $f = 0.2\ 1/s$. Experimental and numerical results obtained by Dütsch et al. [142] are used for comparison.

An Eulerian squared domain is used, with dimensions $\Omega_f = 40D \times 30D = [-20,\ 20] \times [-15,\ 15]$. To reduce the computational time similarly to the previous problem, the region $[-5,\ 5] \times [-5,\ 5]$ near the cylinder boundary that covers its minimum and maximum displacement is created with an Eulerian grid resolution of $h = 0.025\ m$. Neumann boundary conditions are applied to both pressure ($\partial p/\partial \mathbf{n} = 0$) and velocity ($\partial \mathbf{u}/\partial \mathbf{n} = 0$) for all domain boundaries, as shown in Fig. (5.17). The cylinder is located initially at the centre of Ω_f, i.e., $(0,\ 0)$, and Γ_s is formed by 124 Lagrangian markers having the arc length between two consecutive markers $\Delta s \approx 0.025335\ m$. The desired Lagrangian velocity of the cylinder boundary $\mathbf{U}_{IB}$ is enforced at each time-step by taking the derivative of Eq. (5.5) with respect to time. A constant time-step of $\Delta t = 0.005D/U_{max}$ is used, similarly to [143], and the kernel function ϕ_9 is selected for the interpolation and spreading operators.

Fig. (5.18) demonstrates the pressure and vorticity contours at four different phase angles, $\varphi = 0°$, $96°$, $192°$, and $288°$ (having $\varphi = t/T \times 360°$). Good agreement is observed qualitatively with the numerical results obtained by Dütsch et al. [142], and a similar flow structure for the two counter-rotating vortices of equal strength is illustrated. Another quantitative comparison is carried out with the experimental results extracted by Dütsch et al. [142] for the velocity profiles for both u and v-components, at three different phase angles $\varphi = 180°$,

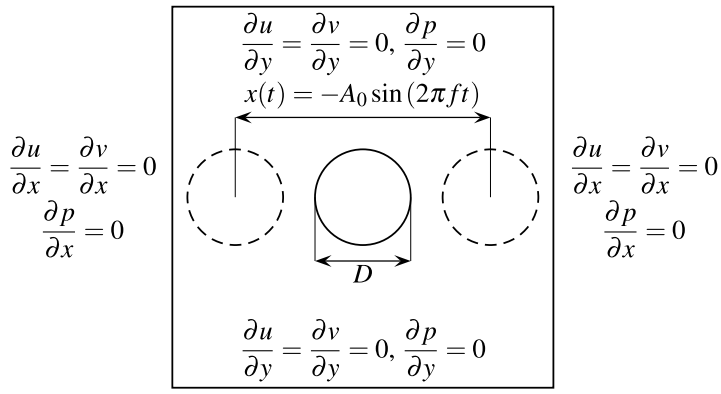


Figure 5.17: *Computational domain and boundary conditions for flow over an in-line oscillating cylinder.*

210°, and 330°, for four different streamwise cross sections $x = -0.6D$, $0D$, $0.6D$, and $1.2D$. Fig. (5.19) shows the results obtained by the current approach in comparison with the experimental results. It can be concluded that the computed results for the velocity profile are consistent and align well with the experimental data. This example demonstrates that the present solver can properly capture the dynamics of the flow field as well as the velocity distributions for moving immersed boundaries.

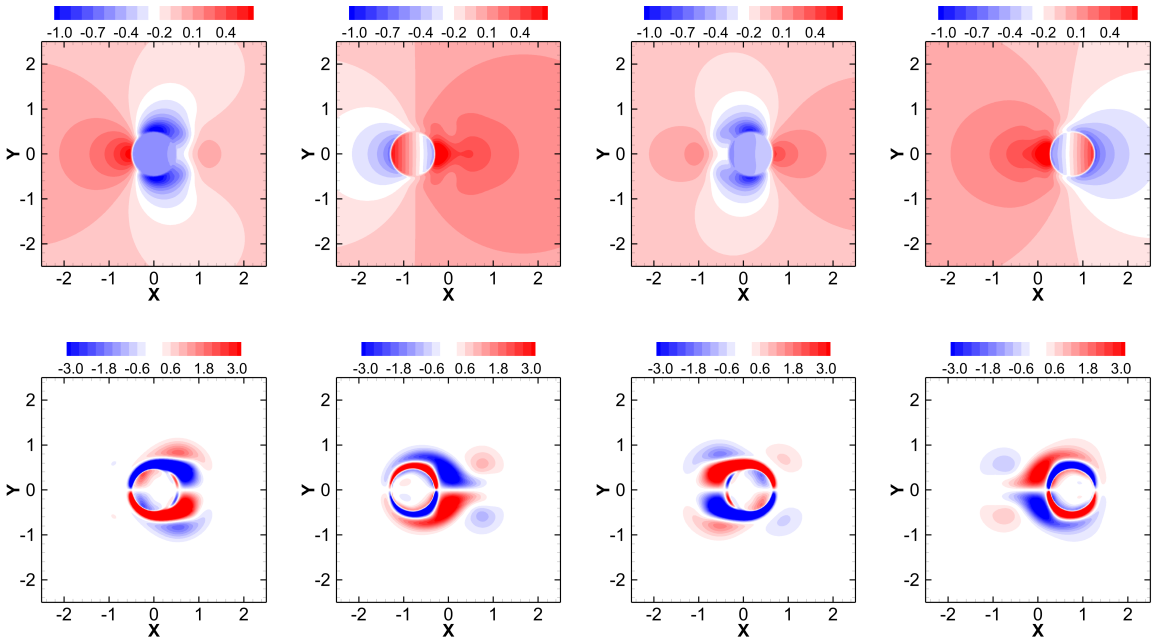
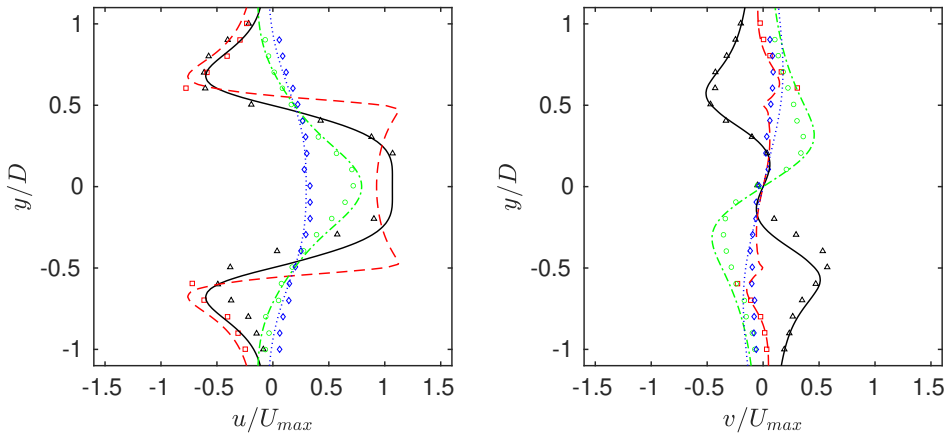
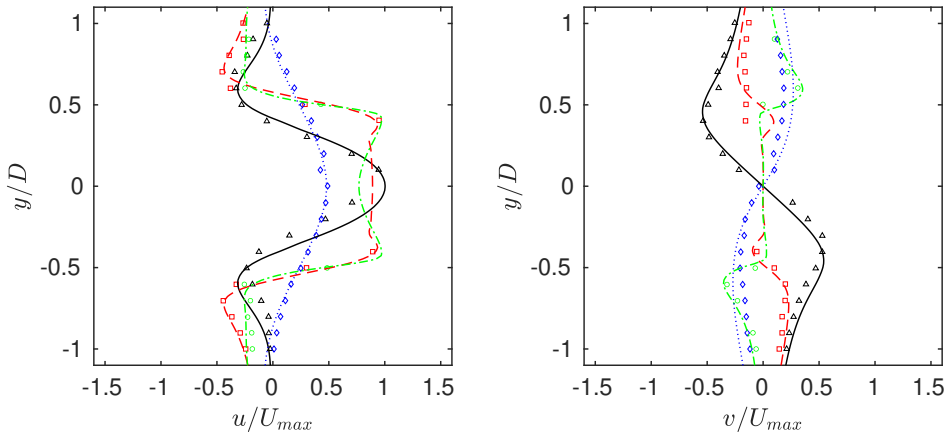


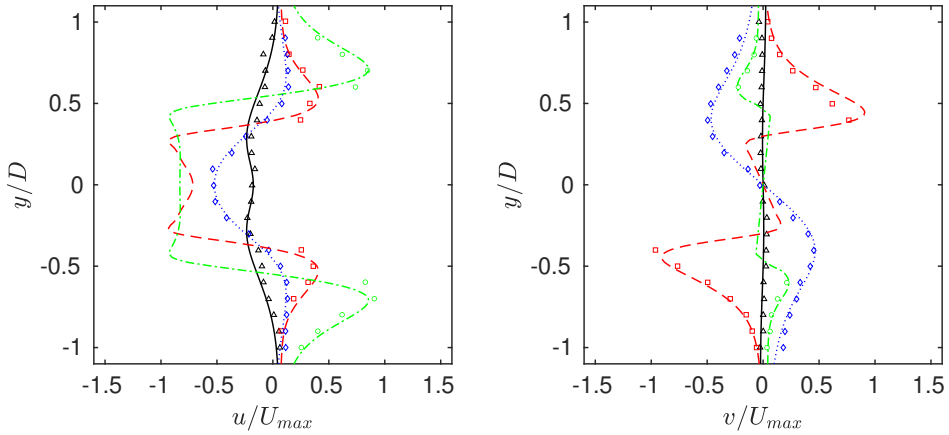
Figure 5.18: *Instantaneous pressure contours (top) and vorticity contours (bottom) for different phase angle $\varphi = 0°$, 96°, 192°, and 288° (from left to right) for the flow over an in-line oscillating cylinder in a quiescent fluid at $Re = 100$ and $KC = 5$.*



(a) $\varphi = 180°$



(b) $\varphi = 210°$



(c) $\varphi = 330°$

Figure 5.19: *Non-dimensional velocity profiles for the u (left) and v (right) components at 3 different phase angles φ and at 4 different streamwise cross-sections. The current results are marked with lines while the experimental results of Dütsch et al. [142] are marked with symbols. At* $x = -0.6D$ *(—— and* $\triangle$*),* $x = 0D$ *(- - - and* $\square$*),* $x = 0.6D$ *(-·-· and* $\circ$*), and* $x = 1.2D$ *(····· and* $\Diamond$*).*

5.3.2 Flow Over a Transversely Oscillating Cylinder in a Freestream at Re = 185

Another cylinder oscillating case is presented, where the cylinder is now oscillating in the crosswise direction of a freestream going from the left to the right boundaries of the Eulerian domain. The crosswise translational motion of the cylinder is prescribed by the following equation:

$$y(t) = A_0 \sin(2\pi f t), \tag{5.6}$$

with $y\ (m)$ the lateral position of the cylinder centroid, $A_0\ (m)$ the amplitude of the oscillation, and $f\ (1/s)$ the characteristic frequency of the oscillation.

The flow Reynolds number is set to $Re = u_\infty D/\nu = 185$, with $u_\infty = 1\ m/s$ the uniform freestream velocity, $D = 1\ m$ the cylinder diameter, and $\nu = 5.4054 \times 10^{-3}\ m^2/s$ the fluid kinematic viscosity. The amplitude is fixed to $A_0 = 0.2D$, and the frequency is varied between $0.8 \leq f/f_s \leq 1.2$, with $f_s = 0.195\ 1/s$ the natural vortex shedding frequency for the stationary cylinder case at $Re = 185$, taken from [144]. Those parameters are considered to compare the results with those extracted from the literature [89, 145].

The dimensions of the Eulerian domain are defined as $\Omega_f = 60D \times 60D = [-20,\ 40] \times [-30,\ 30]$, with a grid resolution $h = 0.02\ m$ in the refined region $[-5,\ 5] \times [-5,\ 5]$, which covers the displacement of the moving cylinder boundary. The boundary conditions are similar to the flow over a stationary cylinder case, see Fig. (5.20).

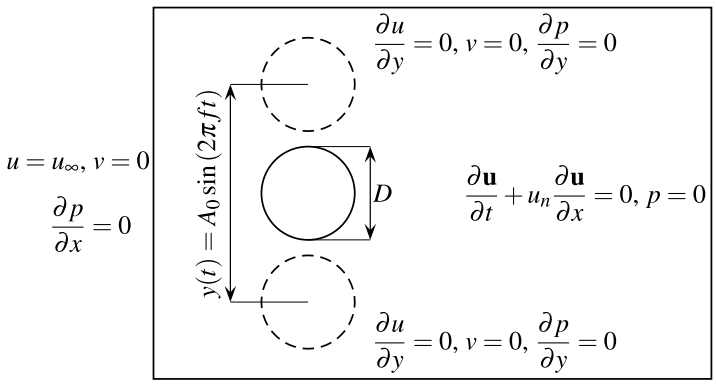
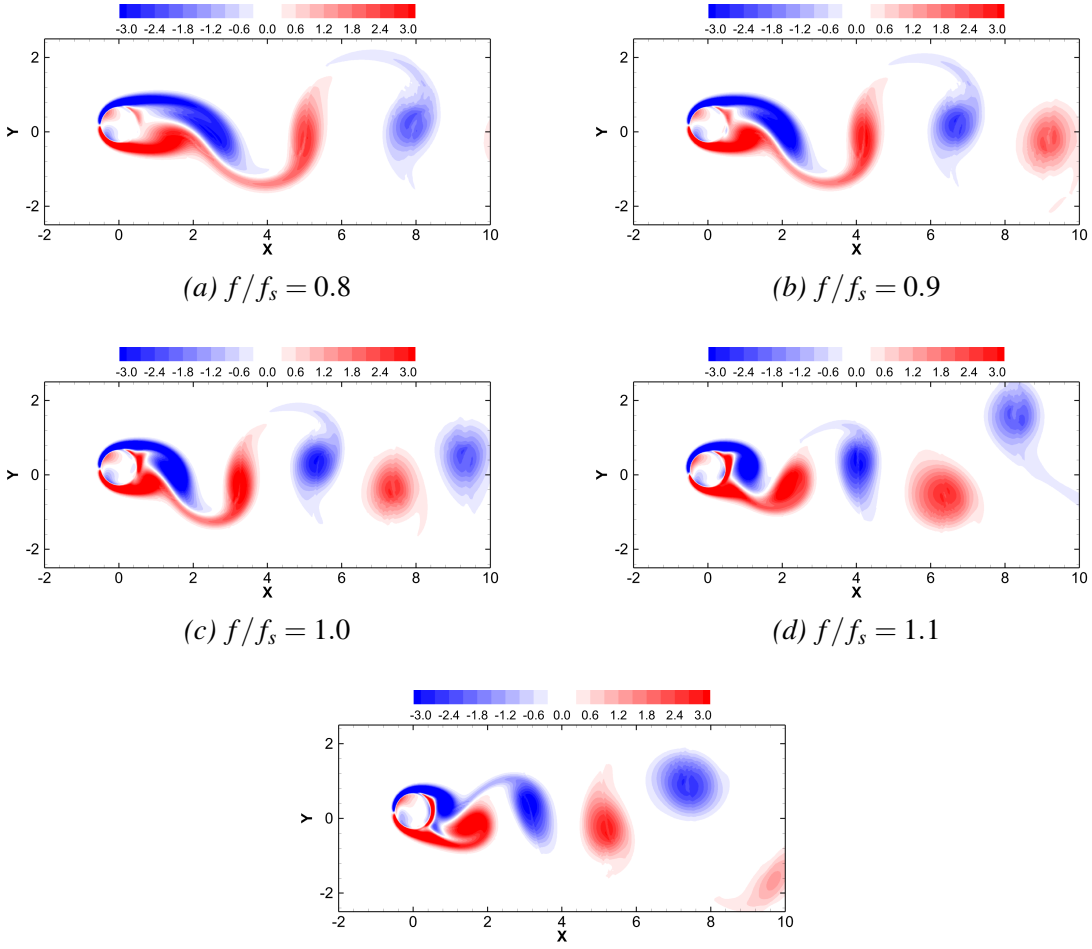


Figure 5.20: *Computational domain and boundary conditions for flow over a transversely oscillating cylinder.*

The cylinder is located at $t = 0$ at the origin of Ω_f, i.e., $(0,\ 0)$, and Γ_s is represented by 156 Lagrangian markers, with a Lagrangian spacing Δs computed as in Table 5.6. A time-step of $\Delta t = 0.004\ s$ is adopted for all simulations. Similarly to the cylinder in-line

oscillation problem, the transient desired Lagrangian velocity $\mathbf{U}_{IB}$ is enforced at each time level by taking the derivative of Eq. (5.6), and the 5-point-width ϕ_9 kernel function is used to reduce spurious force oscillations.

Fig. (5.21) illustrates the instantaneous vorticity contours when the cylinder is located at the highest positive position. The overall vorticity pattern obtained with the present method when the frequency f/f_s varies, is in agreement with the observations reported in [89, 145]. The vortex shedding is found to be closer to the wake of the cylinder with increasing f/f_s.



(a) $f/f_s = 0.8$ (b) $f/f_s = 0.9$

(c) $f/f_s = 1.0$ (d) $f/f_s = 1.1$

(e) $f/f_s = 1.2$

Figure 5.21: *Instantaneous vorticity contours at different frequencies f/f_s for the flow over a transversely oscillating cylinder in a freestream at Re* $= 185$.

A quantitative analysis is also carried out for the mean drag coefficient and the rms values of the drag and lift fluctuations. The findings are presented in Fig. (5.22). The variation of the mean drag coefficient with increasing f/f_s is in fair agreement with the literature data of [89, 145]. However, results are slightly higher here, with an average difference of

9.8% when compared with [145] and 3.8% when compared with [89] for all frequency ratios combined. Similarly, excellent agreement is found for the rms values of the lift fluctuations for higher frequencies between $1.1 \leq f/f_s \leq 1.2$. For lower frequencies, the computed values are also slightly higher when compared with results in [89, 145]. The rms values of the drag fluctuations also agree well with the results reported in the literature.

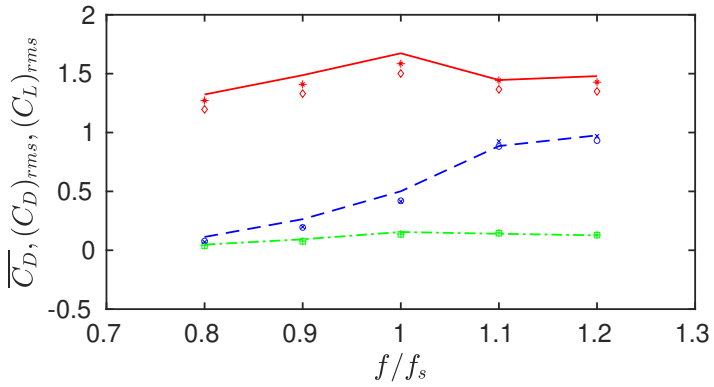


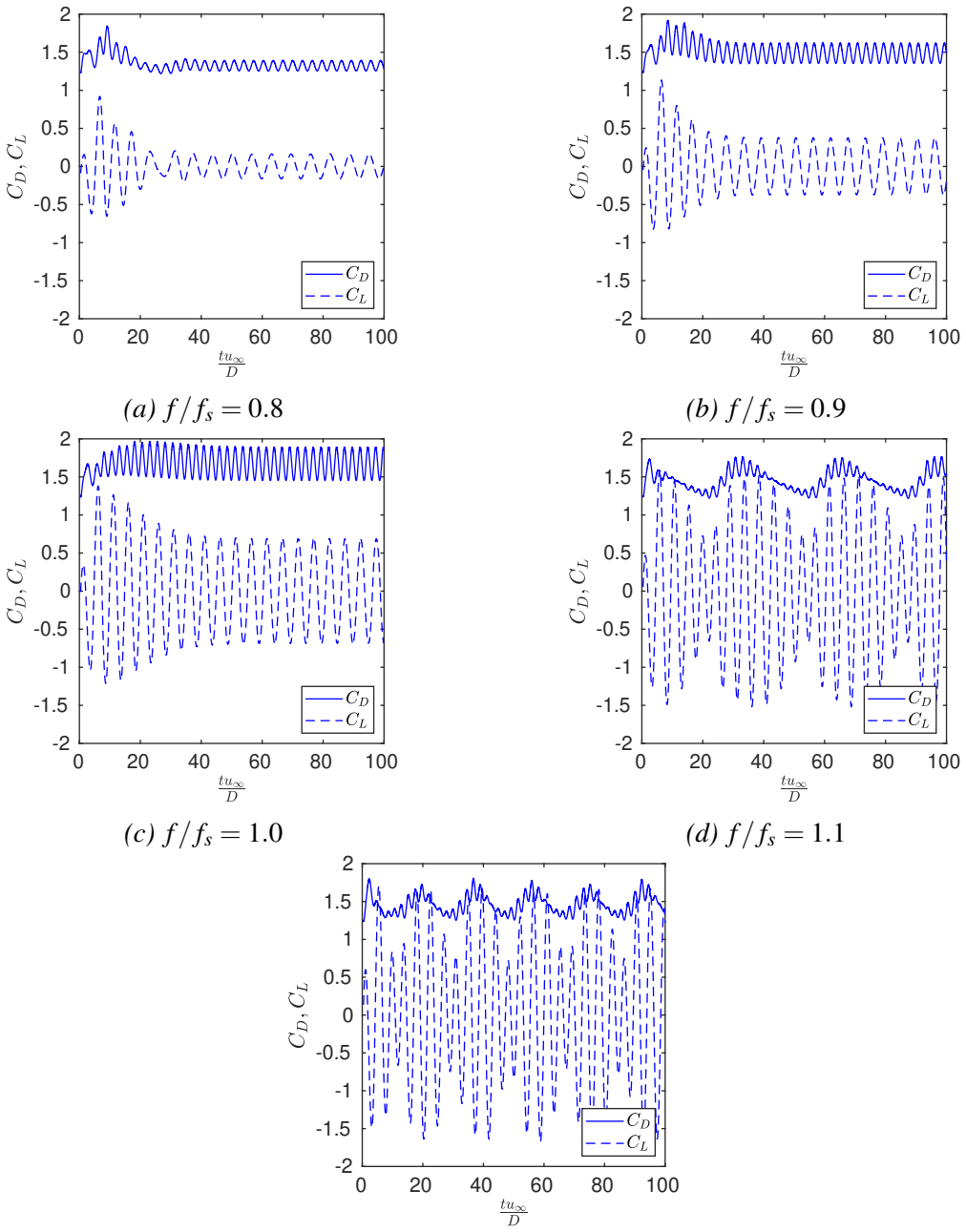
Figure 5.22: *Comparison of the force coefficients at different frequencies f/f_s for the flow over a transversely oscillating cylinder in a freestream at $Re = 185$. The present results are marked with lines, the results corresponding to [145] are marked with ($\Diamond$; $\square$; $\circ$), and the results corresponding to [89] are marked with ($*$; $+$; $\times$). $\overline{C_D}$ (—, $\Diamond$, and $*$); $(C_D)_{rms}$ (-·-·, $\square$, and $+$); $(C_L)_{rms}$ (- - -, $\circ$, and $\times$).*

Fig. (5.23) illustrates the instantaneous temporal evolution of the drag and lift coefficients for all values of the frequencies f/f_s. The challenge was to compute correctly the drag and lift forces exerted on the oscillating cylinder. Due to the nature of the immersed boundary method, a flow is developed inside the immersed object since the interior is treated as part of the fluid Eulerian domain. This internal fluid does not affect the flow outside the immersed object, whereas the forces on the immersed boundary are influenced by this internal flow development, i.e., by the internal mass. Once the internal fluid is moved (or accelerated), the internal mass effect (also called added-mass effect) should be added to the force equations to compensate for the accelerating fluid inside the immersed boundary [146]. A more comprehensive discussion of the internal mass effect is presented in Chapter 6 in the context of fluid–rigid body coupling. Therefore, unlike the computation of drag and lift forces for flow over a stationary cylinder, as expressed in Eq. (5.4), the forces for moving (accelerating) solid immersed boundaries are computed using the following approach:

$$\begin{pmatrix} F_D \\ F_L \end{pmatrix} = -\rho \sum_{n=1}^{N_{Lag}} \mathbf{F}_{IB}(\mathbf{X}_n, t) W_n + \rho V \frac{d\mathbf{U}_{IB}(\mathbf{X}_n, t)}{dt}, \tag{5.7}$$

where ρ is the fluid density and V is the volume occupied by the solid immersed boundary.

The patterns observed in the force coefficient plots across this wide spectrum of frequencies are similar to those reported in [145], where a body conformal method was employed, as well as to the results obtained using a body non-conforming method in [89].



(a) $f/f_s = 0.8$ *(b)* $f/f_s = 0.9$

(c) $f/f_s = 1.0$ *(d)* $f/f_s = 1.1$

(e) $f/f_s = 1.2$

Figure 5.23: *Temporal evolution of the drag C_D and lift C_L coefficients at different frequencies f/f_s for the flow over a transversely oscillating cylinder in a freestream at $Re = 185$.*

5.3.3 Flow Over a Hovering Elliptical Wing at Re = 157

The flow produced by a non-circular object in a fluid at rest is simulated, where the immersed boundary is described by an elliptical wing that mimics the shape of an insect's wing, and the hovering of the wing is subject to both translational and rotational prescribed motions governed by the following equations:

$$x(t) = \frac{A_0}{2}\left[cos\left(\frac{2\pi t}{T}\right)+1\right]cos(\beta), \tag{5.8}$$

$$y(t) = \frac{A_0}{2}\left[cos\left(\frac{2\pi t}{T}\right)+1\right]sin(\beta), \tag{5.9}$$

$$\alpha(t) = \frac{\theta_0}{2}\left[1-sin\left(\frac{2\pi t}{T}+\varphi_0\right)\right], \tag{5.10}$$

where $(x,\ y)\ (m)$ are the time-dependent ellipse centroid due to motion and $\alpha\ (rad)$ is the time-dependent angle of attack of the wing. A_0 and θ_0 designate the translational and rotational amplitudes, respectively, β is the stroke angle, φ_0 is the phase difference between the translational and rotational movements, and T is the hovering period. The configuration of the moving wing is described in Fig. (5.24), with c the major axis (or chord length) and b the minor axis of the ellipse. The flow is characterised by the maximum velocity of translational motion along the hovering path computed as $U_{max} = \pi A_0/T$. The hovering period is defined by $T = \pi A_0/c$ and the Reynolds number is $Re = U_{max}c/\nu$. The simulation is performed at $Re = 157$, with $c = 1\ m$, $c/b = 4$, $A_0 = 2.5\ m$, $\theta_0 = \pi/2$, $\beta = \pi/3$, $\varphi_0 = 0$, $U_{max} = 1\ m/s$, and $T = 2.5\pi\ s$, to compare with previous numerical studies from the literature [81, 143, 147].

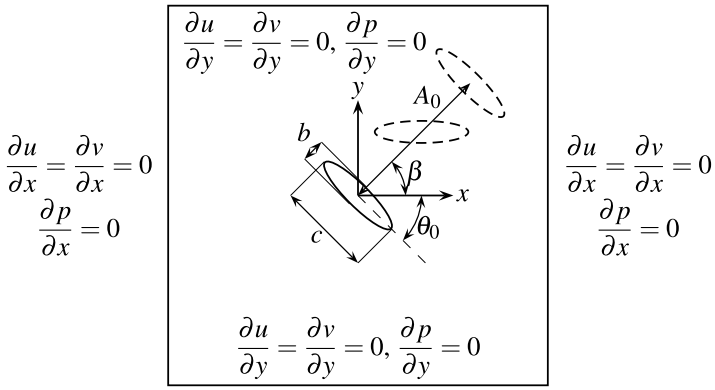
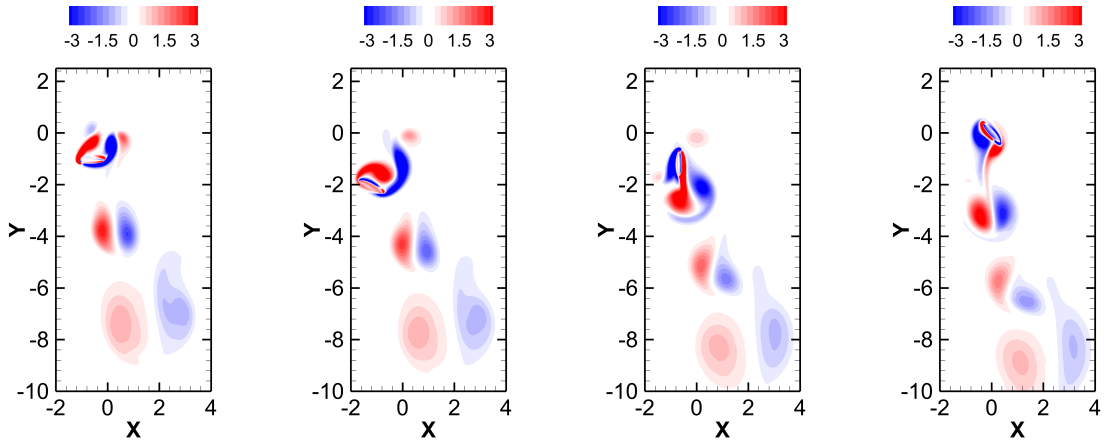


Figure 5.24: *Computational domain and boundary conditions for flow over a hovering elliptical wing.*

A smaller squared Eulerian domain is used compared to the recommended one in the works of [81, 143], such that $\Omega_f = 20c \times 20c = [-10,\ 10] \times [-10,\ 10]$, with an Eulerian grid

resolution $h = 0.02\ m$ in the region $[-5,\ 5] \times [-5,\ 5]$ around the wing minimum and maximum displacement. The domain boundary conditions are similar to the in-line oscillations of a cylinder problem, see Fig. (5.24). The ellipse is located at the beginning of the simulation at $(0,\ 0)$ and Γ_s is formed by 106 Lagrangian markers with $\Delta s \approx 0.020232\ m$. The desired Lagrangian velocity of the ellipse boundary $\mathbf{U}_{IB}$ is enforced by summing up the translational and rotational velocities derived from Eqs. (5.8)–(5.10). A constant time-step, $\Delta t = 0.004\ s$, is used for this simulation, and ϕ_9 is used as the kernel function to reduce the nonphysical spurious force oscillations.

Fig. (5.25) shows the instantaneous vorticity field produced by the hovering elliptical wing at four different instances of a single flapping period, i.e., $t/T = 0.25$, 0.44, 0.74, and 0.99. Those correspond to the wing downstroke, where a counter-rotating pair of vortices is generated. These vortices are then shed to generate lift at the half-cycle of the period. Afterward, the wing upstroke is initiated to push a vortex dipole away from the wing. At the end of the period, the vortex dipole is driven further downward of the computational domain. These vorticity contours are very similar to those available in the literature, see [81, 143, 147] for instance.

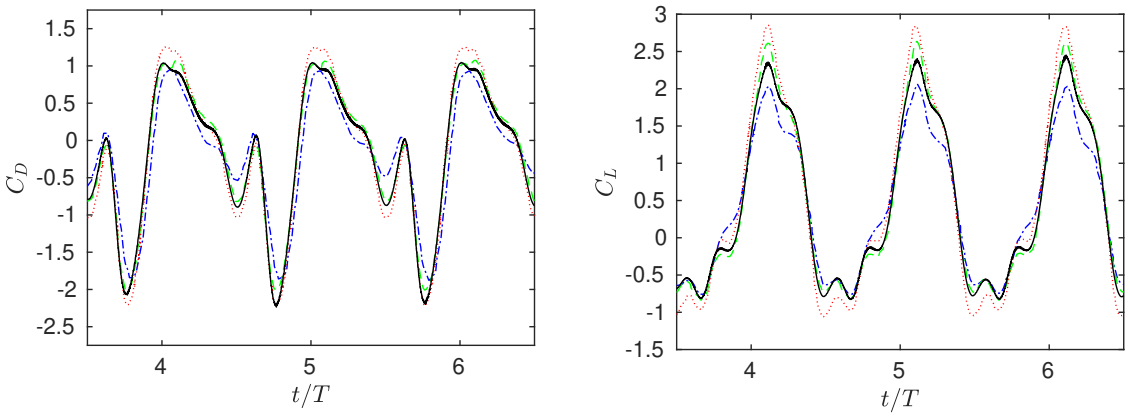


(a) $t/T = 0.25$ *(b)* $t/T = 0.44$ *(c)* $t/T = 0.74$ *(d)* $t/T = 0.99$

Figure 5.25: *Instantaneous vorticity contours at four different period instances* t/T *for the flow over a hovering elliptical wing at* $Re = 157$.

A quantitative validation of the forces generated by the hovering wing is carried out, by comparing the time history of the instantaneous drag and lift coefficients in Fig. (5.26), with previous studies computed using a body conformal mesh method [147] and body non-conforming immersed boundary methods [81, 143]. The internal mass effect is also considered in this case, similarly to the previous problem. The data obtained with the present

DF-IBM agree well with other results from the literature, even when performed on a smaller Eulerian domain.



(a) *(b)*

Figure 5.26: *Temporal evolution of the drag C_D (a) and lift C_L (b) coefficients for the flow over a hovering elliptical wing at $Re = 157$: (——) present results, (·····) moving immersed boundary method (MIBM) [81], (- - -) immersed boundary method [143], (-·-·) body conformal mesh method [147].*

This problem demonstrates that the present method can capture the flow physics even for non-circular objects, while maintaining a good overall accuracy when compared to body-fitted techniques.

5.3.4 Laminar Flow Over a Forced Rotating 2D Vertical Axis Turbine (VAT) at Re = 50 and 100

Finally, the unsteady and complex flow through a vertical axis turbine (VAT) with a prescribed rotation operating in the laminar flow regime is analysed. This study focuses on a three-bladed H-Darrieus type turbine, with each blade designed based on the NACA0015 airfoil profile. Previous numerical investigations have explored similar flows using various advanced methods, including sliding/rotating body-fitted (ALE) techniques coupled with high-order FVM solver [148], high-order discontinuous Galerkin solver [149, 150], and, more recently, a body non-conformal IBM variant known as the volume of solid implicit volume penalty (VOS-IVP) method [67].

The computational setup closely follows that of previous studies [67, 148] to facilitate comparison. The NACA0015 airfoils, each with a chord length of $c = 1\ m$, are positioned at a radial distance of $R = 2c$. The latter is defined as the rotor radius, from the turbine's fixed centre of rotation located at the origin of the fluid domain, $(0,\ 0)$. The radius R intersects the airfoil at the quarter-chord point, $c/4$. The three blades are evenly spaced around the radial direction, with a separating radial angle of 120° between them, as illustrated in Fig. (5.27).

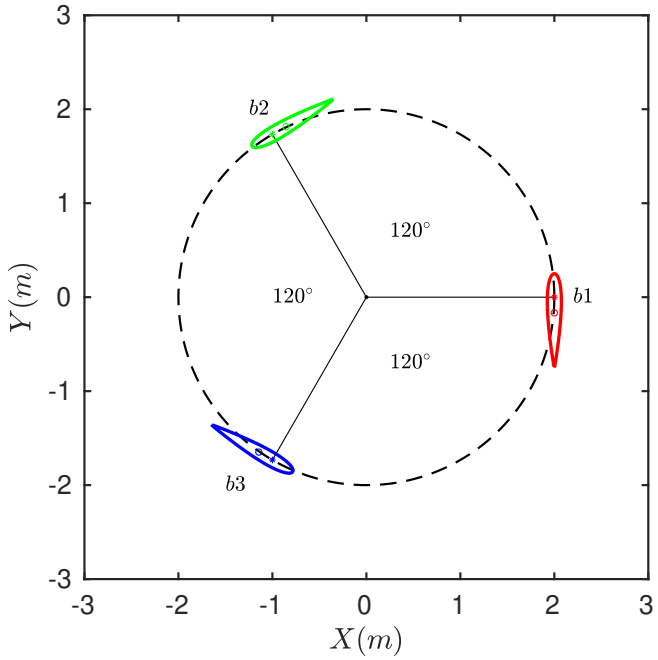


Figure 5.27: *Schematic of the VAT geometrical configuration. The centre of rotation of the VAT is represented by "•", the airfoil quarter-chord point is represented by "∗", and the airfoil centre of mass is represented by "○". b1, b2, and b3 represent blade numbers 1, 2, and 3 of the VAT, respectively.*

The freestream velocity is set to $u_\infty = 0.5\ m/s$, and the turbine operates at a prescribed angular velocity of $\omega = 0.5\ rad/s$. This configuration results in a tip-speed ratio, defined as $\lambda = R\omega/u_\infty$, equal to 2. The flow Reynolds number, based on the airfoil chord length, $Re_c = u_\infty c/\nu$, along with corresponding references for comparison, is provided in Table 5.11. Additionally, the kernel function ϕ_9 is employed to mitigate spurious force oscillations in the simulation.

Table 5.11: *VAT flow Reynolds number based on the airfoil chord length Re_c, kinematic viscosity ν, and the corresponding reference and numerical approach used for comparison.*

Re_c	$\nu\ (m^2/s)$	Reference	Approach
50	0.01	[148]	ALE
100	0.005	[67]	IBM (VOS-IVP)

A fluid Eulerian domain with dimensions $\Omega_f = 17.5R \times 10R = [-5R,\ 12.5R] \times [-5R,\ 5R] = [-10,\ 25] \times [-10,\ 10]$ is used. Within this domain, a refined grid resolution of $h = 0.01\ m$ is applied to the region $[-2.5,\ 5] \times [-3,\ 3]$. The boundary conditions, illustrated in Fig. (5.28), are selected accordingly. Each NACA0015 airfoil is formed using 200 uniformly distributed Lagrangian markers along its surface Γ_s, corresponding to $\Delta s \approx 0.0102814\ m$. This distribution allocates 99 markers to both the upper and lower surfaces, with 2 additional markers positioned at the leading and trailing edges of the airfoil. A time-step of $\Delta t = 0.004\ s$ is chosen to ensure that $CFL < 1$ is maintained throughout the simulation. The computations are carried out for 10 full rotor revolutions, with each revolution requiring $t_{rev} = 2\pi/\omega = 4\pi\ s$. The angle swept by the blades at a given time t is determined using $\theta = \omega t$.

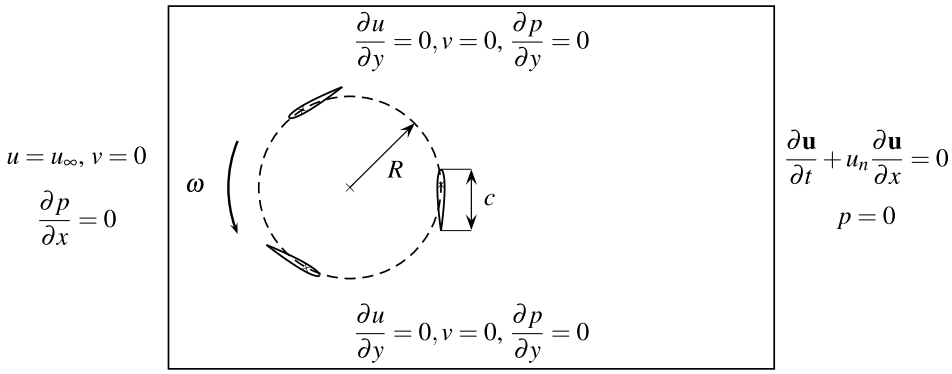


Figure 5.28: *Computational domain and boundary conditions for flow over a 2D VAT in laminar regime.*

The key quantities of interest are the tangential force coefficient (C_T) and the normal

force coefficient (C_N) acting on each blade, defined as:

$$C_T = \frac{F_T}{1/2\rho c H {u_\infty}^2} \quad \text{and} \quad C_N = \frac{F_N}{1/2\rho c H {u_\infty}^2}, \tag{5.11}$$

where H represents the height of the VAT. In this 2D representation of the VAT, H is set to $1\ m$.

The forces $(F_T,\ F_N)$ measured in (N), correspond to the resistive forces in the tangential and normal directions relative to the radial movement of the blades. Since the blades rotate at a constant angular velocity, the internal mass effect vanishes. Consequently, the forces $(F_x,\ F_y)$ in (N), acting on a single blade in the x- and y-directions, are computed using Eq. (5.4). These components are then used to calculate the tangential and normal forces as described in [67, 148]:

$$\begin{pmatrix} F_T \\ F_N \end{pmatrix} = \begin{bmatrix} -\cos(\theta) & -\sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \begin{pmatrix} F_x \\ F_y \end{pmatrix}. \tag{5.12}$$

Special attention must be given to the angle θ when calculating the tangential and normal forces, as the radial angle of 120° separating the blades must be taken into account relative to the specific blade being considered for the computation. Using the tangential force coefficient for a single blade, the power coefficient (C_P) for that blade of the VAT can be calculated. The power coefficient, C_P, is defined as the ratio between the power generated by a single blade and the total power extracted from the fluid, expressed as:

$$C_P = \frac{P}{1/2\rho A {u_\infty}^3}, \tag{5.13}$$

where A represents the swept area of the VAT, given by $A = 2RH\ m^2$, and P is the power generated by the VAT, defined as $P = |F_T| R\omega$ and measured in $(W = N \cdot m/s)$.

The computed aerodynamic force coefficients and power coefficients at $Re_c = 50$ going from the second to the seventh revolutions $(360^\circ \leq \theta \leq 2520^\circ)$ are compared with the results from [148], which were obtained using a body conformal technique (ALE), as shown in Fig. (5.29). The results demonstrate good agreement between the predicted coefficients using the DF-IBM algorithm and those obtained via the ALE method. Notably, the aerodynamic force and power coefficients exhibit a roughly periodic pattern per revolution, with identical profiles observed for all three blades. The only distinction lies in the phase shift between the blades, resulting from the 120° radial angle separating them.

A comparison is also conducted at $Re_c = 100$ to highlight the reduced spurious force oscillations achieved with the present method compared to another IBM variant from the literature, namely the VOS-IVP approach [67]. The data reported in [67] exhibited significant spurious force oscillations, a known challenge associated with the IBM framework. A digitised form of the data was extracted from [67]. However, it should be noted that the digitalised data provided here do not fully capture all the force fluctuations displayed in the original paper, the original signal being highly jagged. For accuracy and reference, the exact values of the aerodynamic force and power coefficients reported in [67] are included in Appendix C. The results obtained using the current method are presented in Fig. (5.30) for the last 2 revolutions ($2880^\circ \leq \theta \leq 3600^\circ$). The present method reduces the noise fluctuations compared to the reference data.

The velocity and vorticity fields at the end of the tenth revolution ($t = 10t_{rev}$, corresponding to $\theta = 3600^\circ$) are shown in Figs. (5.31) and (5.32), for $Re_c = 50$ and 100, respectively. A distinct high-velocity wake region is observed on the downstroke side of the VAT, in contrast to the low-velocity wake region located on the upstroke side.

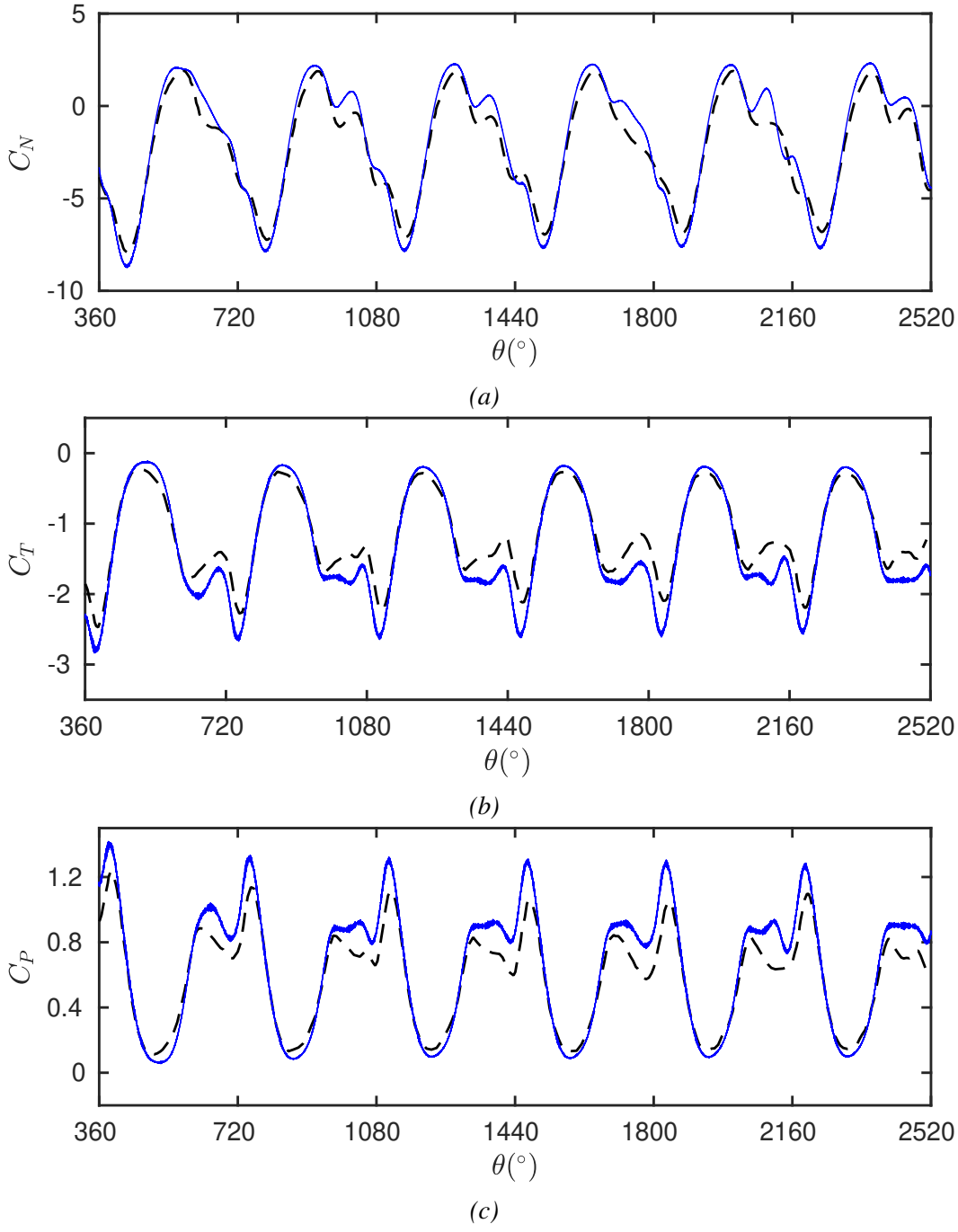


Figure 5.29: *Normal force C_N (a), tangential force C_T (b), and power C_P (c) coefficients for the flow over a forced rotating 2D VAT at $Re_c = 50$ from the second to the seventh revolutions, $t = [2t_{rev},\ 7t_{rev}]$ $(360^\circ \leq \theta \leq 2520^\circ)$: (——) present results, (- - -) ALE [148].*

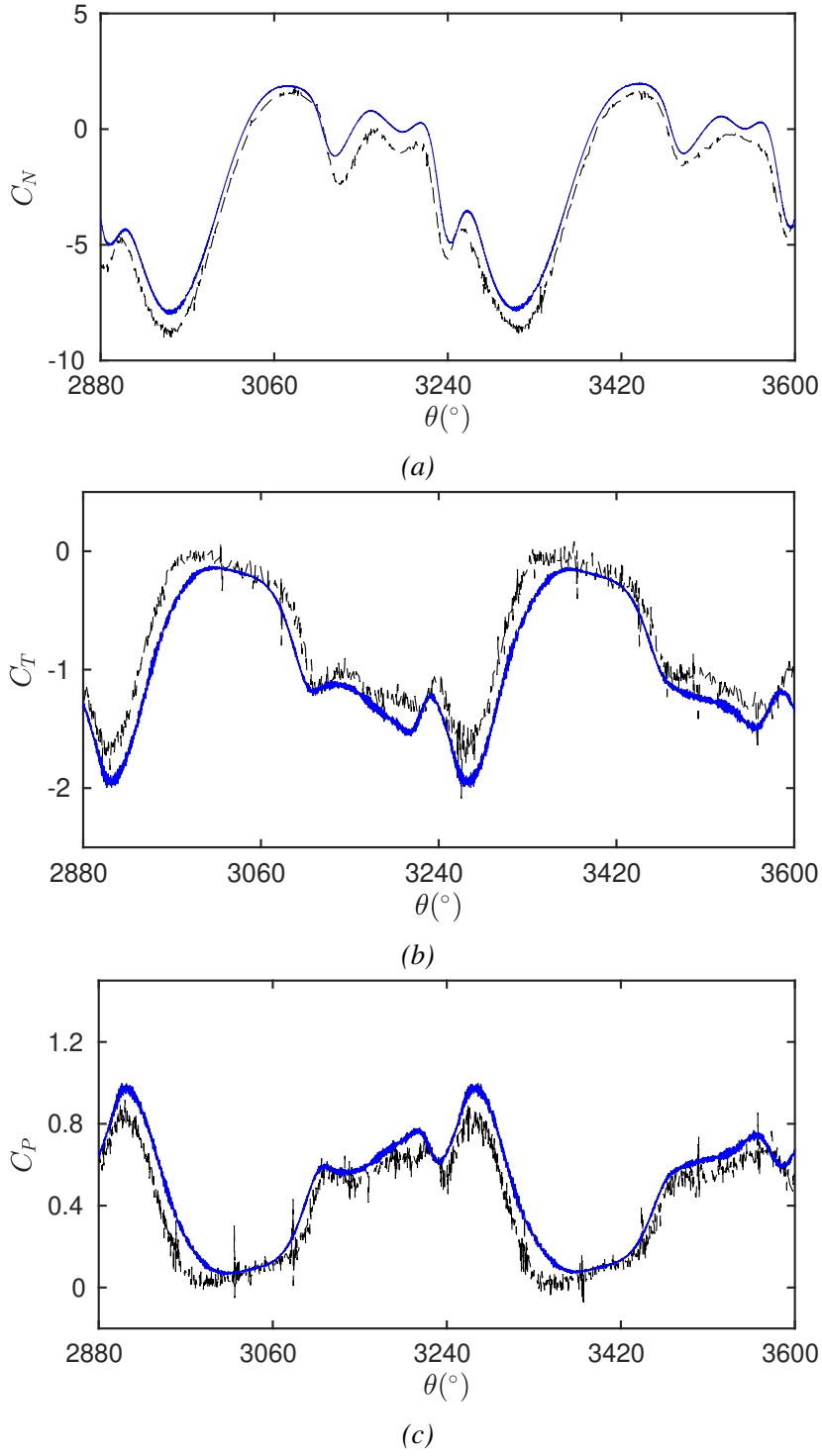
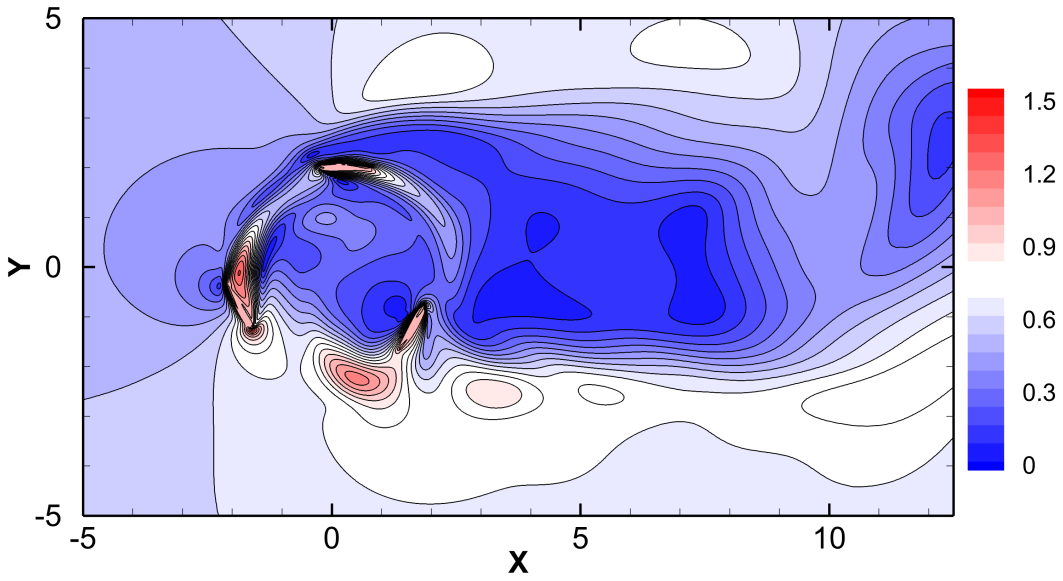
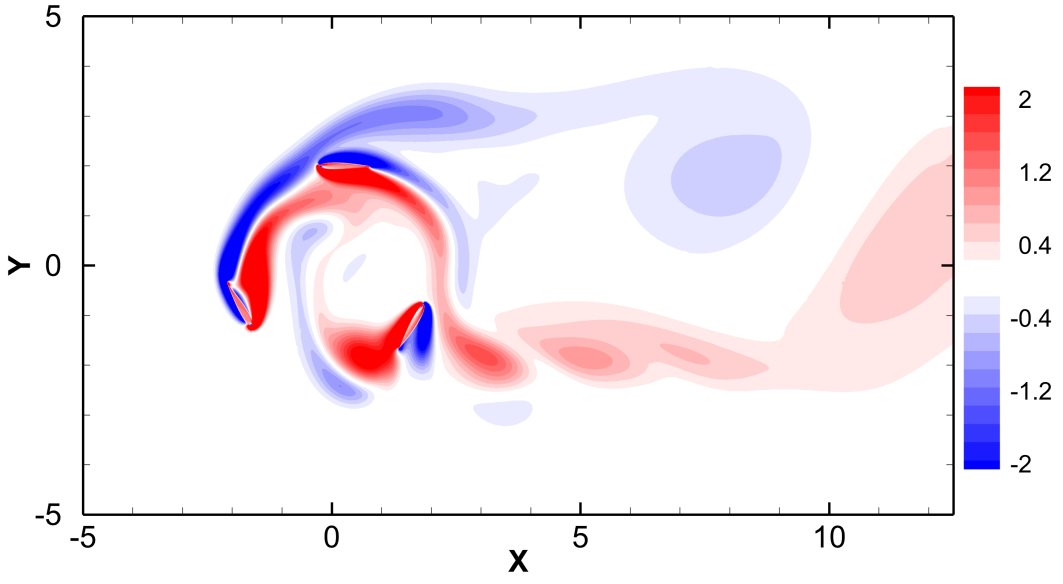


Figure 5.30: *Normal force C_N (a), tangential force C_T (b), and power C_P (c) coefficients for the flow over a forced rotating 2D VAT at $Re_c = 100$ for the ninth and tenth revolutions, $t = [9t_{rev},\ 10t_{rev}]$ $(2880^\circ \leq \theta \leq 3600^\circ)$: (——) present results, (- - -) IBM (VOS-IVP) [67]. The exact data extracted from [67] with their corresponding spurious force oscillation peaks are shown in Appendix C.*

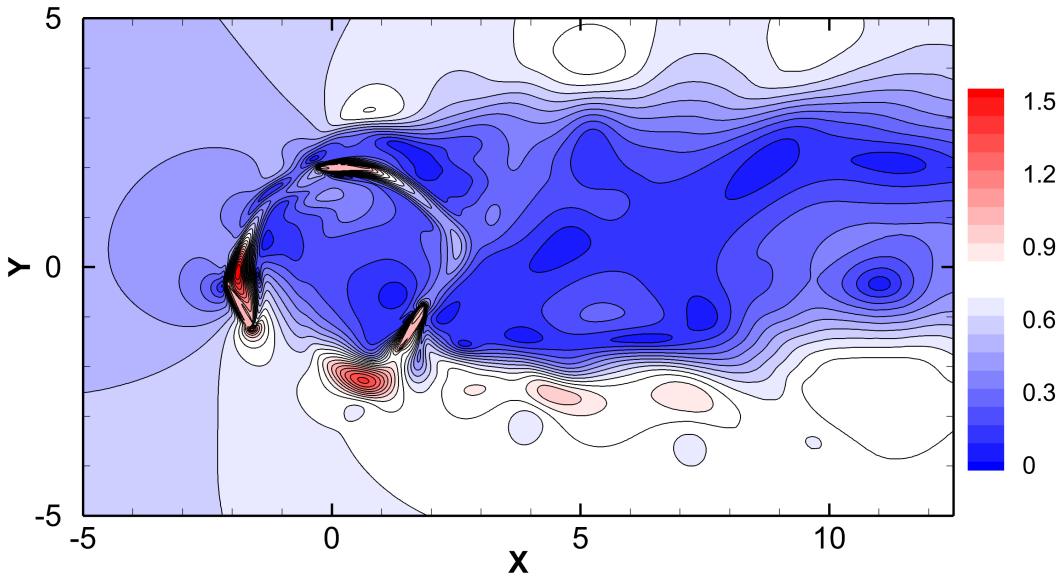


(a)

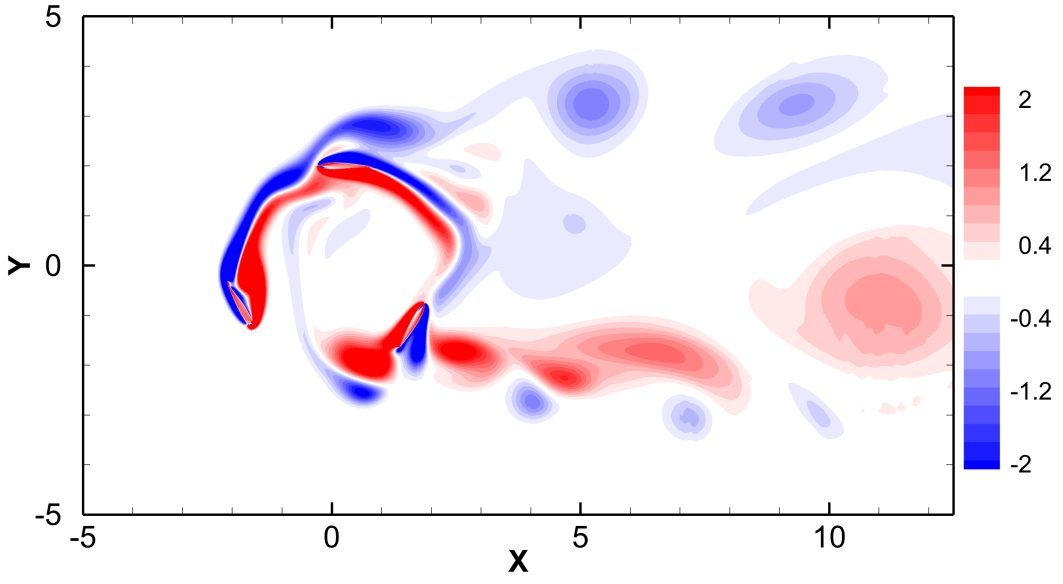


(b)

Figure 5.31: *Velocity (a) and vorticity (b) contours for the flow over a forced rotating 2D VAT at the end of the tenth revolution* $t = 10t_{rev}$*, i.e.,* $\theta = 3600^{\circ}$ *for* $Re_c = 50$*.*



(a)



(b)

Figure 5.32: *Velocity (a) and vorticity (b) contours for the flow over a forced rotating 2D VAT at the end of the tenth revolution* $t = 10t_{rev}$*, i.e.,* $\theta = 3600^{\circ}$ *for* $Re_c = 100$*.*

5.4 Summary

The proposed implicit DF-IBM has been numerically validated in this chapter. An in-depth analysis of the present development in terms of accuracy showed that the method maintains a second-order accuracy when a second-order kernel function is used to match the second-order discretisation of all the terms in the Navier-Stokes equations. Though, this accuracy is maintained only for smooth fields, as demonstrated in the Taylor–Green decaying vortices problem.

However, a second-order solution is not guaranteed for non-smooth fields due to the discrete Dirac delta function. A comprehensive performance and efficiency analysis revealed that the current implicit DF-IBM enforces exactly the second fluid–solid system constraint on the Lagrangian domain with significantly less computational effort compared to the implicit IBM of Ji et al. [80]. Specifically, it reduces the number of implicit iterations by nearly 60%, with only a modest increase in computational time by a factor of 35 compared to the explicit IBM of Uhlmann [39]. Whereas Ji et al.'s implicit variant, revealed an increase by a factor of 91 compared to Uhlmann. An important finding is that using an even number of Lagrangian markers on the immersed boundary rather than an odd number greatly reduces the number of implicit iterations, compared to using an odd number. The sensitivity of the second fluid–solid system constraint to the Lagrangian force initialisation scheme, the Lagrangian weight, the Lagrangian weight analytical correction, and the simulation time-step are also demonstrated.

Furthermore, the current method's capability to handle both circular and non-circular solid immersed boundaries was confirmed. Its effectiveness in capturing flow physics and dynamics for stationary and prescribed motions (both translational and rotational) across various Reynolds numbers was validated through comparisons with experimental data. The results compared favourably with other IBM variants and body conformal methods from the literature, demonstrating excellent agreement across all simulated cases.

Chapter 6

Rigid Body Dynamics (RBD) Coupling

Contents

In the previous chapters, stationary and prescribed motions of solid immersed boundaries were explored. Free and flow-induced motions are now investigated. Accordingly, this chapter introduces the two-dimensional rigid body dynamics (RBD) equations, where the Newton-Euler equations govern the motion of the rigid body, forming a 3-degrees-of-freedom system for general planar motion. Additionally, the internal mass effect (IME), arising from the hydrodynamic force and torque, is thoroughly discussed. This chapter also outlines the mathematical formulation resulting from the coupling of the fluid Navier-Stokes equations with

the solid RBD within the DF-IBM framework. Moreover, different coupling approaches for fluid–rigid body interactions are examined, with particular attention given to the partitioned approach. Efficient explicit and implicit coupling algorithms are presented to address both weak and strong fluid–rigid body interactions while effectively overcoming the challenges associated with the IME.

6.1 Solid Governing Equations

In FSI problems involving a rigid body and a fluid, fluid variables expressed in the Eulerian domain are represented using the subscript f. Solid variables, described in the Lagrangian domain and computed about the rigid body's centre of mass, are denoted with the subscript s. Meanwhile, Lagrangian variables computed at the fluid–solid interface, i.e., the rigid immersed boundary, are represented with the subscript n, instead of IB as in previous chapters, where n corresponds to the Lagrangian marker under consideration.

The general motion of a rigid body is governed by the rigid body dynamics (RBD) equations, also known as, the Newton-Euler equations of motion. They describe a rigid body's combined translational (Newton) and rotational (Euler) dynamics. For a general 3D motion, the system is governed by three equations for the translational motion in every direction of the reference frame (x,y,z) and three equations for the rotational part about every orthogonal axis of the chosen coordinate system (pitch, yaw, and roll). Together, these form a system with a 6-degrees-of-freedom (6-DOF) system. The RBD equations give the relationship between the motion of the centre of mass of a rigid body and the sum of forces and torques acting on that rigid body.

The first three equations are derived from Newton's second law of motion, also known as the translational momentum balance. The remaining three equations are based on Euler's law of motion, also referred to as the rotational momentum balance. These equations are expressed in vector form as:

$$\sum \mathbf{F}_{ext} = m_s \mathbf{a}_s = m_s \frac{d\mathbf{U}_s}{dt}, \tag{6.1a}$$

$$\sum \mathbf{T}_{ext} = \mathbf{I}_s \boldsymbol{\alpha}_s + \boldsymbol{\omega}_s \times \mathbf{I}_s \boldsymbol{\omega}_s = \mathbf{I}_s \frac{d\boldsymbol{\omega}_s}{dt} + \boldsymbol{\omega}_s \times \mathbf{I}_s \boldsymbol{\omega}_s, \tag{6.1b}$$

where,

- $\mathbf{F}_{ext}$, is the external forces vector exerted on the solid rigid body, about its centre of mass,
- m_s, is the solid rigid body mass, with $\rho_s = \dfrac{m_s}{V_s}$ the rigid body density and V_s the rigid body Volume,
- $\mathbf{a}_s = (a_{sx}, a_{sy}, a_{sz})$, is the solid rigid body linear acceleration vector, about its centre of mass,
- $\mathbf{U}_s = (U_s, V_s, W_s)$, is the solid rigid body linear velocity vector, about its centre of mass.

- $\mathbf{T}_{ext}$, is the external torques vector exerted on the solid rigid body, about its centre of mass,
- $\mathbf{I}_s$, is the solid rigid body mass moment of inertia second-rank tensor,
- $\boldsymbol{\alpha}_s = (\alpha_{sx}, \alpha_{sy}, \alpha_{sz})$, is the solid rigid body angular acceleration vector, about its centre of mass,
- $\boldsymbol{\omega}_s = (\omega_{sx}, \omega_{sy}, \omega_{sz})$, is the solid rigid body angular velocity vector, about its centre of mass.

The term of the right-hand side of Eq. (6.1a), $m_s \dfrac{d\mathbf{U}_s}{dt}$, is viewed as the time rate of change of the linear momentum $(m_s\mathbf{U}_s)$ of the solid rigid body. The first term appearing on the right-hand side of Eq. (6.1b), $\mathbf{I}_s \dfrac{d\boldsymbol{\omega}_s}{dt}$, is considered as the time rate of change of the angular momentum $\mathbf{I}_s\boldsymbol{\omega}_s$ of the solid rigid body, whereas the second term, $\boldsymbol{\omega}_s \times \mathbf{I}_s\boldsymbol{\omega}_s$, accounts for the changes in the angular momentum due to the rotation of the solid rigid body, also known as the gyroscopic term.

However, for two-dimensional cases, i.e., for a general planar motion the system reduces to a 3-DOF system only, two of which are related to the translation inside a single plane of motion in order to describe the trajectory of the rigid body, for instance, $(x-y)$ plane, and a third one for the rotation about an axis perpendicular to the plane itself in order to depict the orientation of the rigid body, for example, the z-axis; in case of the $(x-y)$ plane.

6.1.1 2D Solid Kinetics

The Newton-Euler equations for a solid rigid body undergoing a general planar motion in the $(x-y)$ plane are given by:

$$\sum \mathbf{F}_{ext} = m_s\mathbf{a}_s = m_s \frac{d\mathbf{U}_s}{dt}, \tag{6.2}$$

$$\sum \mathbf{T}_{ext} = I_s\alpha_s = I_s \frac{d\omega_s}{dt}, \tag{6.3}$$

where the linear acceleration reduces in 2D to $\mathbf{a}_s = (a_{sx}, a_{sy})$, as well as the linear velocity $\mathbf{U}_s = (U_s, V_s)$. All rotational variables reduce in a planar motion to a scalar value about the z-axis; in the case of the $(x-y)$ plane motion, i.e., the angular acceleration $\alpha_s = \alpha_{sz}$, the angular velocity $\omega_s = \omega_{sz}$, and the mass moment of inertia $I_s = I_{sz}$, having $I_s = \rho_s \int_{V_s} \mathbf{r}_n^2 \, dV$, with $V_s = \int_{\Omega_s} dV$ being the solid rigid body volume, ρ_s the solid rigid body density, and $\mathbf{r}_n = \mathbf{X}_n - \mathbf{X}_s$ the relative position vector of any solid Lagrangian marker $\mathbf{X}_n$ on the rigid body's immersed surface Γ_s with respect to the rigid body centre of mass $\mathbf{X}_s$.

It can be clearly seen, that the gyroscopic term disappears since the cross product of the angular velocity and the angular momentum reduces to zero when working with general planar motion.

Plugging the known forces, torques, and accelerations (or velocities) into the above equations three unknowns can be solved. If more than three unknowns exist in the equations, it might be necessary to go back to kinematics to relate quantities such as linear acceleration and angular acceleration.

6.1.2 2D Solid Kinematics

The absolute velocity $\mathbf{U}_n$ of a Lagrangian marker $\mathbf{X}_n$ undergoing a general planar motion, is equal to the combined translational velocity of the centre of mass $\mathbf{X}_s$ of the solid rigid body and the rotational velocity of that same Lagrangian marker about the centre of mass of the rigid body, yielding:

$$\begin{aligned}\mathbf{U}_n(\mathbf{X}_n,t) &= \mathbf{U}_n^{trans}(\mathbf{X}_n,t) + \mathbf{U}_n^{rot}(\mathbf{X}_n,t)\\ &= \mathbf{U}_s(t) + \omega_s(t) \times \mathbf{r}_n(\mathbf{X}_n,t)\\ &= \mathbf{U}_s(t) + \omega_s(t) \times (\mathbf{X}_n(t) - \mathbf{X}_s(t))\,. \end{aligned} \tag{6.4}$$

The absolute linear positions of a Lagrangian marker $\mathbf{X}_n$ and the centre of mass $\mathbf{X}_s$ undergoing translational motion is computed as:

$$\frac{d\mathbf{X}_n}{dt} = \mathbf{U}_s \quad \text{and} \quad \frac{d\mathbf{X}_s}{dt} = \mathbf{U}_s. \tag{6.5}$$

The overall angular position (orientation) also called the rotation angle of the rigid body θ_s undergoing rotational motion about its centre of mass is computed as:

$$\frac{d\theta_s}{dt} = \omega_s. \tag{6.6}$$

6.2 External Forces and Torques

A rigid body undergoing a general planar motion is influenced by external forces and torques that determine its position, velocity, and acceleration.

1. **Gravity Force:**

   The gravitational force is a downward force exerted on the solid rigid body by the

earth's gravitational acceleration. It is expressed by:

$$\mathbf{F}_g = m_s \mathbf{g} = \rho_s V_s \mathbf{g}, \tag{6.7}$$

where m_s, ρ_s, and V_s are the solid rigid body mass, density, and volume respectively, and $\mathbf{g} = (0,\ -9.8067,\ 0)\ m/s^2$ is the earth's gravitational acceleration.

2. **Buoyancy Force:**
The buoyancy force is an upward force exerted by the fluid on the fully submerged solid rigid body. The magnitude of this buoyancy force is determined by Archimedes' principle, which states: "*The buoyant force acting on an object immersed in a fluid is equal to the weight of the fluid displaced by the object*", therefore the force is computed using:

$$\mathbf{F}_b = -\rho_f V_s \mathbf{g}, \tag{6.8}$$

where ρ_f is the fluid density and the negative sign is added to indicate that the buoyancy force is acting in the opposite direction of the earth's gravitational acceleration.

3. **Hydrodynamic Force and Torque:**
Due to the interaction between the viscous fluid and the surface of the solid rigid body in motion Γ_s, which is termed from now on the fluid–solid interface, a force and a torque, dubbed in the literature as the hydrodynamic force and torque, are exerted on the solid by the fluid stress tensor, resulting in the drag and lift forces as well as the torque on the solid rigid body, having the following forms:

$$\mathbf{F}_h = \rho_f \oint_{\Gamma_s} \boldsymbol{\sigma}_f \cdot \mathbf{n}\, dS, \tag{6.9a}$$

$$\mathbf{T}_h = \rho_f \oint_{\Gamma_s} \mathbf{r}_{i,j} \times (\boldsymbol{\sigma}_f \cdot \mathbf{n})\, dS, \tag{6.9b}$$

where $\mathbf{r}_{i,j} = \mathbf{x}_{i,j} - \mathbf{X}_s$ is the relative position vector of any fluid Eulerian grid point $\mathbf{x}_{i,j}$ with respect to the solid rigid body centre of mass $\mathbf{X}_s$, $\mathbf{n}$ is the outward-pointing unit normal vector on the immersed boundary surface Γ_s, dS is an infinitesimal surface element over Γ_s, and $\boldsymbol{\sigma}_f$ is the fluid Cauchy stress tensor computed similarly as in Eq. (3.4) in addition to enforcing the incompressibility constraint and dividing by the fluid density ρ_f:

$$\boldsymbol{\sigma}_f = -p\mathbf{I} + \nu_f \left(\nabla \mathbf{u}_f + (\nabla \mathbf{u}_f)^{\mathrm{T}}\right), \tag{6.10}$$

where ν_f is the fluid kinematic viscosity.

A direct evaluation of the surface integrals over the fluid stress tensor on the solid rigid body immersed boundary Γ_s shown in Eqs. (6.9a) and (6.9b) are considered a cumbersome task and numerically exigent. To overcome these challenges, Uhlmann [151], proposed an accurate and efficient method to solve these surface integrals using the linear and angular momentum balance using the specific volume forces already computed in the DF-IBM algorithm.

The Cauchy principle states [152], through the momentum balance over the immersed boundary surface Γ_s fully enclosing the solid rigid body volume Ω_s, that the rate of change of the linear and angular momentum over Ω_s, is equal to the sum of the forces and torques applied on the volume Ω_s and the surface Γ_s of that rigid body subjected to motion:

$$\rho_f \frac{d}{dt} \int_{\Omega_s} \mathbf{u}_f \, dV = \rho_f \int_{\Omega_s} \mathbf{f} \, dV + \rho_f \oint_{\Gamma_s} \boldsymbol{\sigma}_f \cdot \mathbf{n} \, dS, \tag{6.11a}$$

$$\rho_f \frac{d}{dt} \int_{\Omega_s} \mathbf{r}_{i,j} \times \mathbf{u}_f \, dV = \rho_f \int_{\Omega_s} \mathbf{r}_{i,j} \times \mathbf{f} \, dV + \rho_f \oint_{\Gamma_s} \mathbf{r}_{i,j} \times (\boldsymbol{\sigma}_f \cdot \mathbf{n}) \, dS, \tag{6.11b}$$

where $\mathbf{f}$ is the boundary force, i.e., the body force applied on the solid rigid body undergoing a general planar motion.

Therefore, the hydrodynamic force and torque acting upon a solid body fully immersed in a fluid are computed by replacing Eqs. (6.11a) and (6.11b) in Eqs. (6.9a) and (6.9b):

$$\mathbf{F}_h = -\rho_f \int_{\Omega_s} \mathbf{f} \, dV + \rho_f \frac{d}{dt} \int_{\Omega_s} \mathbf{u}_f \, dV, \tag{6.12a}$$

$$\mathbf{T}_h = -\rho_f \int_{\Omega_s} \mathbf{r}_{i,j} \times \mathbf{f} \, dV + \rho_f \frac{d}{dt} \int_{\Omega_s} \mathbf{r}_{i,j} \times \mathbf{u}_f \, dV, \tag{6.12b}$$

a physical explanation of each term in Eqs. (6.12a) and (6.12b) are given below in the IBM framework.

The interior of the rigid body is treated as part of the fluid Eulerian domain. Ergo the fluid on the interior of the rigid body is called the internal fluid, while the fluid on the exterior of the rigid body is called the external fluid. An appropriate boundary force $\mathbf{f}$ is applied to enforce the second fluid–solid system constraint, i.e., the no-slip velocity boundary condition on the immersed boundary Γ_s of the solid rigid body Ω_s, to prevent the exterior fluid from penetrating the rigid body.

The first term on the right-hand side of Eqs. (6.12a) and (6.12b), is evaluated over grid

points from both the external and internal fluids to the rigid body, which is denoted here as $\mathbf{F}_{h,tot}$ and $\mathbf{T}_{h,tot}$; the subscript $(.)_{tot}$ stands for total. This implies that the internal fluid (or mass) has basically moved. While this internal flow does not affect the flow outside Γ_s, the moving or accelerating internal fluid has a major influence on the hydrodynamic force and torque exerted on Γ_s. This influence of internal mass is referred to as the internal mass effect, denoted here by $\mathbf{F}_{h,int}$ and $\mathbf{T}_{h,int}$; the subscript $(.)_{int}$ stands for internal, and is equal to the second term on the right-hand side of Eqs. (6.12a) and (6.12b). Since only the external contribution of the fluid acting on the rigid body is of interest, i.e., the hydrodynamic force $\mathbf{F}_h$ and torque $\mathbf{T}_h$ acting on Γ_s from the external fluid, the total force and torque are compensated with the internal force and torque used to move the internal fluid [39, 97, 153]; in other words:

$$\mathbf{F}_h = \mathbf{F}_{h,tot} + \mathbf{F}_{h,int}, \tag{6.13a}$$

$$\mathbf{T}_h = \mathbf{T}_{h,tot} + \mathbf{T}_{h,int}, \tag{6.13b}$$

with,

$$\mathbf{F}_{h,tot} = -\rho_f \int_{\Omega_s} \mathbf{f}\, dV, \tag{6.14a}$$

$$\mathbf{F}_{h,int} = \rho_f \frac{d}{dt} \int_{\Omega_s} \mathbf{u}_f\, dV, \tag{6.14b}$$

$$\mathbf{T}_{h,tot} = -\rho_f \int_{\Omega_s} \mathbf{r}_{i,j} \times \mathbf{f}\, dV, \tag{6.14c}$$

$$\mathbf{T}_{h,int} = \rho_f \frac{d}{dt} \int_{\Omega_s} \mathbf{r}_{i,j} \times \mathbf{u}_f\, dV. \tag{6.14d}$$

In the DF-IBM framework, the total force and torque shown in Eqs. (6.14a) and (6.14c) are computed in the discrete form:

$$\mathbf{F}_{h,tot} = -\rho_f \int_{\Omega_s} \mathbf{f}\, dV = -\rho_f \sum_{\forall \mathbf{x}_{i,j} \in \Omega_f} \mathbf{f}(\mathbf{x}_{i,j}, t) \Delta V_{i,j}, \tag{6.15a}$$

$$\mathbf{T}_{h,tot} = -\rho_f \int_{\Omega_s} \mathbf{r}_{i,j} \times \mathbf{f}\, dV = -\rho_f \sum_{\forall \mathbf{x}_{i,j} \in \Omega_f} \mathbf{r}_{i,j}(\mathbf{x}_{i,j}, t) \times \mathbf{f}(\mathbf{x}_{i,j}, t) \Delta V_{i,j}. \tag{6.15b}$$

By resorting back to the Eulerian–Lagrangian identities discussed in Section 4.3, the total force and total torque identities, as expressed in Eqs. (4.15) and (4.16), respectively, indicate that the total force and the total torque are the same whether evaluated

in the Eulerian or Lagrangian domain. As a consequence Eqs. (6.15a) and (6.15b) can be re-written as:

$$\mathbf{F}_{h,tot} = -\rho_f \sum_{\forall \mathbf{x}_{i,j} \in \Omega_f} \mathbf{f}(\mathbf{x}_{i,j},t)\Delta V_{i,j} = -\rho_f \sum_{\forall \mathbf{X}_n \in \Gamma_s} \mathbf{F}_n(\mathbf{X}_n,t)W_n, \tag{6.16a}$$

$$\mathbf{T}_{h,tot} = -\rho_f \sum_{\forall \mathbf{x}_{i,j} \in \Omega_f} \mathbf{r}_{i,j}(\mathbf{x}_{i,j},t) \times \mathbf{f}(\mathbf{x}_{i,j},t)\Delta V_{i,j} = -\rho_f \sum_{\forall \mathbf{X}_n \in \Gamma_s} \mathbf{r}_n(\mathbf{X}_n,t) \times \mathbf{F}_n(\mathbf{X}_n,t)W_n. \tag{6.16b}$$

The internal mass effect terms shown in Eqs. (6.14b) and (6.14d) is discussed in greater detail in the following section.

The internal mass effect or simply IME, widely recognised in the literature as the added mass effect, refers here to the influence of the internal fluid contained within a solid rigid body. In the context of the immersed boundary method, the term "*internal mass*" is preferred over "*added mass*" because it specifically accounts for the contribution of the internal fluid (mass) enclosed within the immersed boundary of the rigid body.

Subsequently, the rate of change of the linear and angular momentum of the internal fluid occupying the solid rigid body, designated here as the internal force and torque, shown by Eqs. (6.14b) and (6.14d), respectively, need to be computed.

It was proved theoretically by Uhlmann [151], that the rate of change of the linear momentum of the internal fluid is exactly equal to the rate of change of the linear momentum of the rigid body, despite actual internal flow development. Therefore Eq. (6.14b) is re-written as:

$$\mathbf{F}_{h,int} = \rho_f \frac{d}{dt} \int_{\Omega_s} \mathbf{u}_f \, dV = \rho_f \frac{m_s}{\rho_s} \frac{d\mathbf{U}_s}{dt} = \rho_f V_s \frac{d\mathbf{U}_s}{dt}. \tag{6.17}$$

This equality signifies that the force generated by the internal fluid within the volume of the rigid body Ω_s, whose boundary Γ_s is undergoing a rigid body motion, is unaffected by the specific nature of the motion inside the rigid body. Whether the internal fluid follows the rigid body motion or an arbitrary motion, Eq. (6.17) always holds.

However, the same principle does not apply analogously to the rate of change of angular momentum, unless the internal fluid follows the rigid body motion, as demonstrated by Uhlmann [151]. Hence, it becomes necessary to impose an additional approximation by assuming that the internal fluid is forced to follow the rigid body motion, i.e., by

setting:

$$\mathbf{u}_f(\mathbf{x}_{i,j},t) = \mathbf{U}_s(t) + \omega_s(t) \times \mathbf{r}_{i,j}(\mathbf{x}_{i,j},t), \quad \forall \mathbf{x}_{i,j} \in \Omega_s, \tag{6.18}$$

therefore Eq. (6.14d) is reformulated as:

$$\mathbf{T}_{h,int} = \rho_f \frac{d}{dt} \int_{\Omega_s} \mathbf{r}_{i,j} \times \mathbf{u}_f \, dV \approx \rho_f \frac{I_s}{\rho_s} \frac{d\omega_s}{dt}. \tag{6.19}$$

This approximation forces the torque generated by the internal fluid, enclosed by Ω_s, to follow the rigid body motion regardless of the true nature of the internal fluid motion.

4. **Collision Force:**

   Two critical issues arise in the immersed boundary framework when a solid rigid body in motion approaches a wall within the fluid Eulerian domain, that requires special precautions when dealing with it. The first problem is related to the rigid body penetrating the domain's wall, which renders the simulation not realistic, and the second problem primarily stems from the violation of the zeroth moment condition Eq. (4.8) also known as the partition of unity [41], inherent to the kernel function used. The second problem is illustrated in Fig. (6.1). It is evident that the Lagrangian support domain $\mathscr{D}_{\mathbf{X}_n}$ only grouped six Eulerian grid nodes that lie inside Ω_f, and three additional grid nodes, called ghost nodes, that fall outside of Ω_f. These additional ghost nodes are actually not recognised by the computational domain, implying that the IBM-related linear operators are executed solely between the six Eulerian grid nodes situated inside $\mathscr{D}_{\mathbf{X}_n}$ and the Lagrangian marker $\mathbf{X}_n$. This incomplete interaction leads to a violation of the zeroth moment condition Eq. (4.8), property of the kernel function, causing its value to fall below one (or unity). This loss of partition of unity will introduce errors per time-step. Consequently, decreasing the time-step will only exacerbate the accumulation of these errors over time.

   One easy approach to mitigate these issues is by using specialised models designed to prevent a solid rigid body from colliding with a wall of the fluid Eulerian domain, when the distance or the gap between the rigid body and the wall exceeds a certain threshold referred to as the safe zone. These models were originally created for solid particle-particle collisions [154], but the same model can be readily adapted to handle solid rigid body-wall collisions as well.

   In the latter case, the approach involves positioning an imaginary solid rigid body on the opposite side of the wall, i.e., outside the fluid domain, where the imaginary Lagrangian

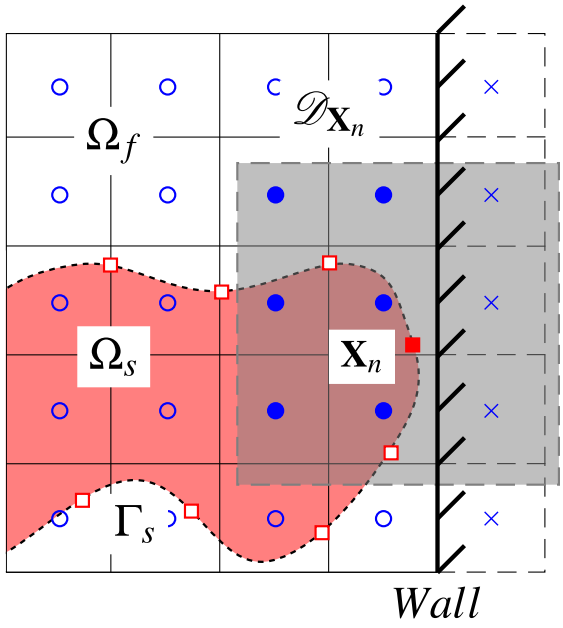


Figure 6.1: *Schematic of the solid rigid body-wall collision in the DF-IBM framework. The Lagrangian marker $\mathbf{X}_n$ where the linear operators are taking place is represented by "■", the Eulerian grid points that fall inside the Lagrangian support domain $\mathscr{D}_{\mathbf{X}_n}$ are represented by "●", the ghost Eulerian grid nodes that fall outside the Eulerian domain are shown in ×, and all remaining Lagrangian markers and Eulerian grid points are illustrated by "□" and "○", respectively.*

marker $\mathbf{X}_n'$ lies exactly on the domain's wall. The system is then modelled as a collision between the real solid rigid body and the imaginary rigid body, effectively treating it as a particle-particle collision model.

In this research, the simplest collision model adopted here is the repulsive potential model (RPM) proposed by Glowinski et al. [155, 156]. This collision model generates an artificial repulsive force that operates only in a short distance range. Therefore, this repulsive force is activated once the distance between the colliding rigid body and the wall is below the pre-defined safe zone threshold ξ. Preventing, on one hand, the rigid body from penetrating the domain's wall, and on the other hand, violating the partition of unity by keeping a distance from the wall, labelled here as the safe zone, allowing for the interpolation and spreading operators to take place without any setbacks. In other words, the rigid body and the wall do not collide physically, while in the numerical sense, an artificial force that models this near-to-collision process is added inside the Newton-Euler equations of motion. For that reason, the value of the safe zone threshold ξ should be based on the support radius r_s of the kernel function appointed to the interpolation function δ_h. Table 4.1 lists all r_s values for all kernel

functions adopted in this research. A good estimation of the value of ξ yields:

$$\xi = (r_s + 1)h, \tag{6.20}$$

with $h = \Delta x = \Delta y$ is the Eulerian mesh resolution; keeping in mind that Cartesian uniform meshes are considered here. The $+1$ was added as a safety factor to guarantee the satisfaction of the partition of unity.

The artificial repulsive force is computed as follows:

$$\mathbf{F}_c = \begin{cases} 0, & d_{s-s'} > 2d_{n-s} + \xi, \\ \dfrac{1}{\varepsilon_w}(\mathbf{X}_s - \mathbf{X}'_s)(2d_{n-s} + \xi - d_{s-s'})^2, & d_{s-s'} \leq 2d_{n-s} + \xi, \end{cases} \tag{6.21}$$

where,

- $\mathbf{X}_s$, is the centre of mass of the solid rigid body,
- $\mathbf{X}'_s$, is the centre of mass of the imaginary solid rigid body on the outer side of the wall,
- $d_{s-s'} = \|\mathbf{X}_s - \mathbf{X}'_s\|$, is the distance between the centre of masses of the solid rigid body and its imaginary counterpart,
- $d_{n-s} = \|\mathbf{X}_n - \mathbf{X}_s\| = \|\mathbf{r}_n\|$, is the distance between the centre of mass and a Lagrangian marker $\mathbf{X}_n$ on the solid rigid body. For circular rigid bodies, this distance is equal to the radius of the circular body,
- ξ, is the safe zone threshold and it's computed according to Eq. (6.20),
- ε_w, is a model constant called the stiffness parameter. A value of $\varepsilon_w = \dfrac{h^2}{2}$ according to [157] was found to produce satisfactory results in the current simulations.

6.3 Navier-Stokes and Rigid Body Dynamics Coupling: A Mathematical Formulation via the DF-IBM

In a general fluid–rigid body interaction problem, an arbitrarily shaped rigid body occupies the domain Ω_s, described in the Lagrangian framework, and undergoes general planar motion within a viscous incompressible fluid, Ω_f, represented in the Eulerian framework. Within the context of the DF-IBM, the rigid body is modelled through its rigid immersed boundary, Γ_s,

with the fluid occupying the interior of the rigid body. The interaction between the fluid and the rigid body occurs at the fluid–solid interface, which coincides with the immersed boundary, $\Gamma_s = \Omega_f \cap \Omega_s$. The fluid motion is governed by the Navier-Stokes equations, while the rigid body follows the RBD equations, expressed through the Newton-Euler equations of motion. In the DF-IBM framework, to account for the presence of the rigid immersed boundary Γ_s within Ω_f, a boundary force $\mathbf{f}$ is introduced into the fluid equations as a source term. Consequently, the fluid equations are solved on a fixed Cartesian mesh, independent of the rigid body's arbitrary motion. A schematic illustrating the physical domain of the fluid–rigid body system is provided in Fig. (6.2), with the whole system subjected to earth's gravitational acceleration.

The mathematical formulation of the fluid–rigid body system couples the Navier-Stokes equations for the fluid with the Newton-Euler equations of motion for the rigid body under the DF-IBM framework:

$$\nabla \cdot \mathbf{u}_f = 0, \qquad \text{in } \Omega_f, \tag{6.22a}$$

$$\frac{\partial \mathbf{u}_f}{\partial t} + \nabla \cdot (\mathbf{u}_f \mathbf{u}_f) - \nabla \cdot (\nu_f \nabla \mathbf{u}_f) = -\nabla p + \mathbf{f}, \qquad \text{in } \Omega_f, \tag{6.22b}$$

$$\mathbf{u}_f(\mathbf{x}_{i,j}, t) = \mathbf{U}_s(t) + \boldsymbol{\omega}_s(t) \times \mathbf{r}_n(\mathbf{X}_n, t), \qquad \text{on } \Gamma_s, \tag{6.22c}$$

$$\mathbf{U}_n(\mathbf{X}_n, t) = \mathscr{I}[\mathbf{u}_f(\mathbf{x}_{i,j}, t)], \qquad \text{from } \Omega_f \text{ to } \Gamma_s, \tag{6.22d}$$

$$\mathbf{f}(\mathbf{x}_{i,j}, t) = \mathscr{S}[\mathbf{F}_n(\mathbf{X}_n, t)], \qquad \text{from } \Gamma_s \text{ to } \Omega_f, \tag{6.22e}$$

$$m_s \frac{d\mathbf{U}_s}{dt} = \mathbf{F}_g + \mathbf{F}_b + \mathbf{F}_h + \mathbf{F}_c, \qquad \text{on } \Omega_s, \tag{6.22f}$$

$$I_s \frac{d\boldsymbol{\omega}_s}{dt} = \mathbf{T}_h, \qquad \text{on } \Omega_s, \tag{6.22g}$$

where Eqs. (6.22a) and (6.22b) are the fluid governing equations, Eq. (6.22c) is the new fluid–solid interface no-slip velocity boundary condition, Eqs. (6.22d) and (6.22e) are the IBM-related linear operators, and Eqs. (6.22f) and (6.22g) are the rigid body governing equations of motion.

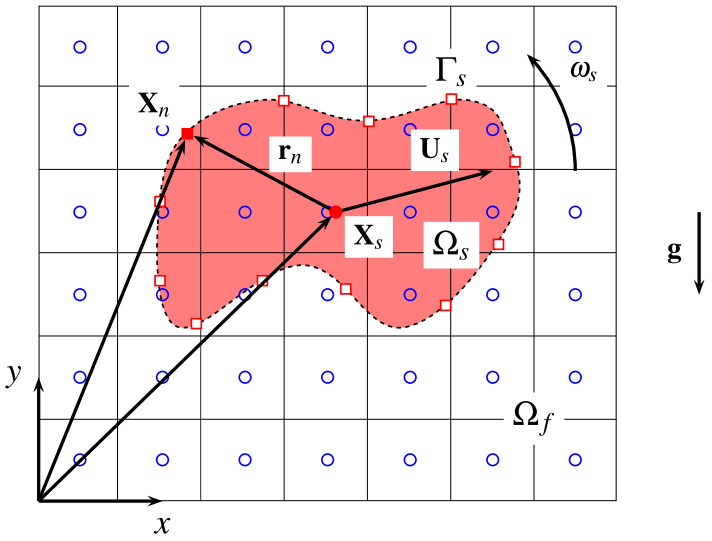


Figure 6.2: *Schematic of the fluid–rigid body domain used in DF-IBM framework to simulate FSI problems.* $\mathbf{U}_s$ *and* ω_s *are the linear and angular velocities of the rigid body applied at its centre of mass* $\mathbf{X}_s$ *shown by "•".* $\mathbf{r}_n$ *is the relative position vector a Lagrangian marker* $\mathbf{X}_n$ *shown by "■" situated on* Γ_s *with respect to* $\mathbf{X}_s$*. The remaining Lagrangian markers and Eulerian grid points are illustrated by "□" and "○", respectively.* $\mathbf{g}$ *is the earth's gravitational acceleration acting downward.*

6.4 Numerical Discretisation

6.4.1 Solid Governing Equations Discretisation

The fluid governing equations, Eqs. (6.22a) and (6.22b), are discretised as in Chapter 3. The solid governing equations, Eqs. (6.22f) and (6.22g), are advanced in time using the first-order implicit backward Euler. Hence, the linear and angular accelerations are discretised as follows:

$$m_s \frac{d\mathbf{U}_s}{dt} = \rho_s V_s \frac{\mathbf{U}_s{}^{n+1} - \mathbf{U}_s{}^{n}}{\Delta t}, \tag{6.23a}$$

$$I_s \frac{d\omega_s}{dt} = I_s \frac{\omega_s{}^{n+1} - \omega_s{}^{n}}{\Delta t}. \tag{6.23b}$$

Adding all external forces and torque to the discretised Newton-Euler Eqs. (6.22f) and (6.22g), one obtains:

$$\rho_s V_s \frac{\mathbf{U}_s{}^{n+1} - \mathbf{U}_s{}^{n}}{\Delta t} = \mathbf{F}_g + \mathbf{F}_b + \mathbf{F}_h + \mathbf{F}_c, \tag{6.24a}$$

$$I_s \frac{\omega_s{}^{n+1} - \omega_s{}^{n}}{\Delta t} = \mathbf{T}_h. \tag{6.24b}$$

Now, substituting each external force and torque with their respective equality shown by Eq. (6.7), Eq. (6.8), Eq. (6.16a), Eq. (6.16b), Eq. (6.17), Eq. (6.19), while treating the interface variables implicitly as unknowns to be solved for, the system yields the final discretised form:

$$\rho_s V_s \frac{\mathbf{U}_s{}^{n+1} - \mathbf{U}_s{}^{n}}{\Delta t} = \rho_s V_s \mathbf{g} - \rho_f V_s \mathbf{g} - \rho_f \sum_{\forall \mathbf{X}_n \in \Gamma_s} \mathbf{F}_n{}^{n+1}(\mathbf{X}_n{}^{n+1}) W_n + \rho_f V_s \frac{d\mathbf{U}_s}{dt} + \mathbf{F}_c{}^{n+1}(\mathbf{X}_n{}^{n+1}), \quad (6.25a)$$

$$I_s \frac{\omega_s{}^{n+1} - \omega_s{}^{n}}{\Delta t} = -\rho_f \sum_{\forall \mathbf{X}_n \in \Gamma_s} \mathbf{r}_n{}^{n+1}(\mathbf{X}_n{}^{n+1}) \times \mathbf{F}_n{}^{n+1}(\mathbf{X}_n{}^{n+1}) W_n + \rho_f \frac{I_s}{\rho_s} \frac{d\omega_s}{dt}. \quad (6.25b)$$

The collision force term $\mathbf{F}_c$ is kept without using the equality shown in Eq. (6.21) for simplicity. Various approaches for handling the IME discretisation within the Newton-Euler equations are discussed in the following sections.

6.4.1.1 Internal Mass Effect (IME) Discretisation

1. **Without IME:**
   The first scheme is set by ignoring the presence of the IME in the Newton-Euler equations, i.e.,

   $$\mathbf{F}_{h,int} = 0, \quad (6.26a)$$

   $$\mathbf{T}_{h,int} = 0. \quad (6.26b)$$

   This scheme has been found to have no significant impact on the solution for low solid Reynolds numbers ($Re_{solid} < 1$) or low solid rotational Reynolds numbers ($Re^{rot}_{solid} < 1$), with the latter defined based on the maximum angular velocity experienced by the rigid body. However, neglecting the IME leads to highly inaccurate results when either Re_{solid} or $Re^{rot}_{solid} \gg 1$ [97].

2. **Implicit IME:**
   The second scheme was first proposed by Uhlmann [39] in the context of the DF-IBM and has been widely adopted for its simplicity of implementation, where the IME is discretised implicitly using the first-order backward Euler scheme:

   $$\mathbf{F}_{h,int} = \rho_f V_s \frac{\mathbf{U}_s{}^{n+1} - \mathbf{U}_s{}^{n}}{\Delta t}, \quad (6.27a)$$

$$\mathbf{T}_{h,int} = \rho_f \frac{I_s}{\rho_s} \frac{{\omega_s}^{n+1} - {\omega_s}^n}{\Delta t}, \tag{6.27b}$$

adding these terms to Eqs. (6.25a) and (6.25b) leads to:

$$\mathbf{U}_s{}^{n+1} = \Delta t \left(\mathbf{g} - \frac{\rho_f}{(\rho_s - \rho_f)V_s} \sum_{\forall \mathbf{X}_n \in \Gamma_s} \mathbf{F}_n{}^{n+1}(\mathbf{X}_n{}^{n+1}) W_n + \frac{1}{(\rho_s - \rho_f)V_s} \mathbf{F}_c{}^{n+1}(\mathbf{X}_n{}^{n+1}) \right) + \mathbf{U}_s{}^n, \tag{6.28a}$$

$${\omega_s}^{n+1} = \Delta t \left(\frac{\rho_f \rho_s}{(\rho_s - \rho_f) I_s} \sum_{\forall \mathbf{X}_n \in \Gamma_s} \mathbf{r}_n{}^{n+1}(\mathbf{X}_n{}^{n+1}) \times \mathbf{F}_n{}^{n+1}(\mathbf{X}_n{}^{n+1}) W_n \right) + {\omega_s}^n. \tag{6.28b}$$

It is evident that this implicit treatment introduces a singularity problem in the obtained Newton-Euler Eqs. (6.28a) and (6.28b) when $\rho_s/\rho_f = 1$, corresponding to a neutrally buoyant rigid body in the fluid domain. Consequently, the stability of the FSI problem is constrained by this density ratio limitation. However, Uhlmann [39] reported that stable computations were achieved for 2D rigid bodies having $\rho_s/\rho_f \gtrapprox 1.05$. Though higher limits on the density ratio were recorded for stable computations [41, 97], with these limits depending on the specific problem, the numerical method employed, and the mesh resolution used. Another drawback of this scheme is that it produces inaccurate results only when $Re^{rot}_{solid} \gg 10$ [97].

3. **Explicit IME:**
   To counter this density ratio limitation, an explicit scheme was proposed by Feng and Michaelides [96], where the first-order forward Euler scheme was used to discretise the IME:

$$\mathbf{F}_{h,int} = \rho_f V_s \frac{\mathbf{U}_s{}^n - \mathbf{U}_s{}^{n-1}}{\Delta t}, \tag{6.29a}$$

$$\mathbf{T}_{h,int} = \rho_f \frac{I_s}{\rho_s} \frac{{\omega_s}^n - {\omega_s}^{n-1}}{\Delta t}. \tag{6.29b}$$

Replacing in Eqs. (6.25a) and (6.25b) yields the following Newton-Euler equations:

$$\mathbf{U}_s{}^{n+1} = \Delta t \left(\frac{\rho_s - \rho_f}{\rho_s} \mathbf{g} - \frac{\rho_f}{\rho_s V_s} \sum_{\forall \mathbf{X}_n \in \Gamma_s} \mathbf{F}_n{}^{n+1}(\mathbf{X}_n{}^{n+1}) W_n + \frac{1}{\rho_s V_s} \mathbf{F}_c{}^{n+1}(\mathbf{X}_n{}^{n+1}) \right) + \frac{\rho_s + \rho_f}{\rho_s} \mathbf{U}_s{}^n - \frac{\rho_f}{\rho_s} \mathbf{U}_s{}^{n-1}, \tag{6.30a}$$

$$\omega_s{}^{n+1} = \Delta t \left(-\frac{\rho_f}{I_s} \sum_{\forall \mathbf{X}_n \in \Gamma_s} \mathbf{r}_n{}^{n+1}(\mathbf{X}_n{}^{n+1}) \times \mathbf{F}_n{}^{n+1}(\mathbf{X}_n{}^{n+1}) W_n \right) + \frac{\rho_s + \rho_f}{\rho_s} \omega_s{}^n - \frac{\rho_f}{\rho_s} \omega_s{}^{n-1}. \quad (6.30b)$$

This explicit treatment overcomes the limitation on the density ratio, allowing the simulation of buoyant rigid bodies, specifically for $\rho_s/\rho_f < 1$, hence for solid rigid bodies lighter than the surrounding fluid. Furthermore, as noted in [97], this scheme enables stable computations and provides reasonably accurate results. However, it only exhibits phase discrepancies in the torque coefficient when the solid rotational Reynolds number exceeds a value of $Re^{rot}_{solid} > 10$.

4. **Lagrangian Markers Approximation:**
Another approach to discretise the IME was introduced in [97], termed the Lagrangian markers approximation. Following the IME definition, i.e., Eqs. (6.14b) and (6.14d), the rate of change of the linear and angular momentum of the internal fluid (or mass) is computed over internal Lagrangian markers that move together with the solid rigid body. In other words, all fluid internal grid points are replaced with internal Lagrangian markers when computing the rate of change of the linear and angular momentum of the internal fluid. The internal Lagrangian markers are initially distributed uniformly with an internal spacing equal to the Eulerian grid spacing, i.e., $\Delta s_{int} \approx h$.

To do so, first, the velocities of the internal Lagrangian markers should be computed using the velocity interpolation operator represented by Eq. (4.2):

$$\mathbf{U}(\mathbf{X}_{int}, t) = \sum_{\mathbf{x}_{i,j} \in \mathscr{D}_{\mathbf{X}_{int}}} \mathbf{u}_f(\mathbf{x}_{i,j}, t) \delta_h(\mathbf{x}_{i,j} - \mathbf{X}_{int}) \Delta V_{i,j}, \quad \forall \mathbf{X}_{int} \in \Omega_s. \quad (6.31)$$

Then, the linear and angular momentum of the internal Lagrangian markers are computed respectively as:

$$\mathbf{P}_{int}(t) = \rho_f \sum_{\forall \mathbf{X}_{int} \in \Omega_s} \mathbf{U}(\mathbf{X}_{int}, t) \Delta V_{int}, \quad (6.32a)$$

$$\mathbf{L}_{int}(t) = \rho_f \sum_{\forall \mathbf{X}_{int} \in \Omega_s} (\mathbf{X}_{int}(t) - \mathbf{X}_s(t)) \times \mathbf{U}(\mathbf{X}_{int}, t) \Delta V_{int}, \quad (6.32b)$$

where ΔV_{int} is the internal volume occupied by the internal Lagrangian marker and is set equal to $\Delta V_{i,j}$, since $\Delta s_{int} \approx h$. Finally, the rate of change of the linear and angular

momentum is computed explicitly using the first-order forward Euler scheme:

$$\mathbf{F}_{h,int} \approx \frac{\mathbf{P}_{int}(t^n) - \mathbf{P}_{int}(t^{n-1})}{\Delta t}, \tag{6.33a}$$

$$\mathbf{T}_{h,int} \approx \frac{\mathbf{L}_{int}(t^n) - \mathbf{L}_{int}(t^{n-1})}{\Delta t}. \tag{6.33b}$$

This approach has been demonstrated [97] to produce accurate results even for solid rotational Reynolds numbers exceeding $Re^{rot}_{solid} > 10$, while also enabling stable computations without imposing restrictions on the density ratio. Nevertheless, this approach significantly increases the computational effort of the algorithm, especially when the number of internal Lagrangian markers becomes large. Additionally, for rigid bodies with complex shapes, the process of uniformly distributing internal Lagrangian markers within the solid becomes a challenging and time-consuming task.

5. **Volume Fraction Method:**
A different method to alleviate the density ratio limitation was presented by Kempe and Fröhlich [41], named the volume fraction method. This method evaluates numerically the volume integrals shown in the rate of change of the linear and angular momentum Eqs. (6.14b) and (6.14d), by employing the second-order midpoint quadrature rule, reading:

$$\rho_f \int_{\Omega_s} \mathbf{u}_f \, dV = \rho_f \sum_{\mathbf{x}_{i,j} \in \mathscr{D}_{\mathbf{X}_n}} \mathbf{u}_f(\mathbf{x}_{i,j}, t) \, \Delta V_{i,j} \, \alpha_{i,j}, \tag{6.34a}$$

$$\rho_f \int_{\Omega_s} \mathbf{r}_{i,j} \times \mathbf{u}_f \, dV = \rho_f \sum_{\mathbf{x}_{i,j} \in \mathscr{D}_{\mathbf{X}_n}} \mathbf{r}_{i,j}(\mathbf{x}_{i,j}, t) \times \mathbf{u}_f(\mathbf{x}_{i,j}, t) \, \Delta V_{i,j} \, \alpha_{i,j}, \tag{6.34b}$$

where $\alpha_{i,j}$ is the volume fraction defined as the ratio (or fraction) of the solid rigid body volume occupying an Eulerian grid cell volume:

$$\alpha_{i,j} = \frac{V_{n \in (i,j)}}{\Delta V_{i,j}}, \tag{6.35}$$

with $\Delta V_{i,j} = h^2$ is the Eulerian grid cell volume and $V_{n \in (i,j)}$ represents the intersecting volume of the solid Lagrangian marker $\mathbf{X}_n$ with the Eulerian grid cell volume. To efficiently approximate the volume fraction $\alpha_{i,j}$, Kempe and Fröhlich [41] employed a signed-distance function Φ and a level-set function H defined at the fluid–solid inter-

face Γ_s. In 2D, this approach yields the following expression:

$$\alpha_{i,j} = \frac{\sum_{m=1}^{4} -\Phi_m H(-\Phi_m)}{\sum_{m=1}^{4} |\Phi_m|}, \tag{6.36}$$

where H is the Heaviside step function:

$$H(\Phi) = \begin{cases} 0, & \Phi \geq 0, \\ 1, & \Phi < 0. \end{cases} \tag{6.37}$$

The summation of Eq. (6.36) is performed over all corners m of the fluid grid cell, having $m = 4$ corners in 2D. The term Φ_m is assumed positive outside of Γ_s and negative when situated inside of Γ_s. Finally, the rate of change of the linear and angular momentum is computed explicitly using the first-order forward Euler scheme:

$$\mathbf{F}_{h,int} \approx \frac{\left[\rho_f \int_{\Omega_s} \mathbf{u}_f \, dV\right]^n - \left[\rho_f \int_{\Omega_s} \mathbf{u}_f \, dV\right]^{n-1}}{\Delta t}, \tag{6.38a}$$

$$\mathbf{T}_{h,int} \approx \frac{\left[\rho_f \int_{\Omega_s} \mathbf{r}_{i,j} \times \mathbf{u}_f \, dV\right]^n - \left[\rho_f \int_{\Omega_s} \mathbf{r}_{i,j} \times \mathbf{u}_f \, dV\right]^{n-1}}{\Delta t}. \tag{6.38b}$$

This method has been shown to produce accurate and stable results for neutrally buoyant as well as for buoyant (light) rigid bodies, with density ratios as low as $\rho_s/\rho_f \approx 0.3$ [41]. However, the increased complexity and computational cost associated with computing the volume fraction as the number of Lagrangian markers gets large enough make it less attractive compared to the explicit IME method proposed by Feng and Michaelides [96].

6.4.2 Solid Position Discretisation

Once the linear and angular velocities are obtained by solving the Newton-Euler equations of motion, the linear translation and angular rotation of the solid rigid body must be computed to update the overall position and orientation of the rigid body. Two reference frames are used to update the new position of the solid rigid body, the first is the absolute reference frame $(.)|_{abs}$, having its origin located at any point in the Eulerian space, usually the origin of the Cartesian coordinate system. The second, is the relative reference frame $(.)|_{rel}$, having the centre of mass of the solid rigid body as its origin. The former is used to treat the translational dynamics while the latter is used for the rotational one.

The solid position update will be divided into three cases depending on the rigid body motion: (1) translational motion only, (2) rotational motion only, and (3) combined translation and rotational motions. For clarity, all variables expressed in the forms $(.)_n{}^{n+1}$ and $(.)_s{}^{n+1}$ are equivalent to $(.)_n$ (t^{n+1}) and $(.)_s$ (t^{n+1}), respectively.

1. **Translational motion:**
   In the case of pure translational dynamics, once the linear velocity of the rigid body $\mathbf{U}_s$ is updated at the new time level $n+1$, the centre of mass $\mathbf{X}_s$ and any Lagrangian marker $\mathbf{X}_n$ situated on Γ_s are translated using Eq. (6.5). The absolute linear position equation is discretised in time as:

$$\frac{\mathbf{X}_n{}^{n+1} - \mathbf{X}_n{}^{n}}{\Delta t} = \mathbf{U}_s{}^{n+1} \quad \text{and} \quad \frac{\mathbf{X}_s{}^{n+1} - \mathbf{X}_s{}^{n}}{\Delta t} = \mathbf{U}_s{}^{n+1}. \tag{6.39}$$

   The difference between the old and new positions for both the centre of mass and any Lagrangian marker is equal due to the body's rigidity. This difference is denoted as $\Delta\mathbf{X} = \mathbf{X}_n{}^{n+1} - \mathbf{X}_n{}^{n} = \mathbf{X}_s{}^{n+1} - \mathbf{X}_s{}^{n}$. After the body's pure translational dynamics, the updated position at the new time level $n+1$ expressed in the absolute reference frame is:

$$\mathbf{X}_n{}^{n+1}|_{abs} = \mathbf{X}_n{}^{n}|_{abs} + \Delta\mathbf{X} \quad \text{and} \quad \mathbf{X}_s{}^{n+1}|_{abs} = \mathbf{X}_s{}^{n}|_{abs} + \Delta\mathbf{X}, \tag{6.40}$$

   where $\Delta\mathbf{X} = \mathbf{U}_s{}^{n+1}\Delta t$.

2. **Rotational motion:**
   For the pure rotational case, once the angular velocity ω_s is obtained at the time level $n+1$, the position of any Lagrangian marker $\mathbf{X}_n$ describing Γ_s is rotated according to Eq. (6.6). Notice that, the centre of mass $\mathbf{X}_s$ is kept fixed for pure rotational motion. The body orientation angle is discretised in time as:

$$\frac{\theta_s{}^{n+1} - \theta_s{}^{n}}{\Delta t} = \omega_s{}^{n+1}. \tag{6.41}$$

   The variation of the orientation of the rigid body θ_s between two consecutive time-steps is expressed as $\Delta\theta = \theta_s{}^{n+1} - \theta_s{}^{n}$. After the body undergoes pure rotational motion, the updated orientation angle, $\theta_s{}^{n+1}$, about the solid rigid body centre of mass $\mathbf{X}_s$ is computed as:

$$\theta_s{}^{n+1} = \theta_s{}^{n} + \Delta\theta, \tag{6.42}$$

   where $\Delta\theta = \omega_s{}^{n+1}\Delta t$.

In terms of relative linear position due to rotation, the effect of the rotation angle ${\theta_s}^{n+1}$ is transformed into a linear translational vector to update the new position of the Lagrangian marker $\mathbf{X}_n$, through the following steps:

(1) First, transform all Lagrangian markers $\mathbf{X}_n$ into the relative reference frame fixed at the solid rigid body's centre of mass $\mathbf{X}_s$:

$$\mathbf{X}_n{}^n|_{rel} = \mathbf{X}_n{}^n|_{abs} - \mathbf{X}_s|_{abs}. \tag{6.43}$$

(2) Second, rotate all Lagrangian markers $\mathbf{X}_n$ in the relative reference frame about its origin, i.e., about the centre of mass $\mathbf{X}_s$:

$$\mathbf{X}_n{}^{n+1}|_{rel} = \mathbf{R}(\Delta\theta)\mathbf{X}_n{}^n|_{rel}, \tag{6.44}$$

where $\mathbf{R}$ is the 2D rotational matrix, defined as:

$$\mathbf{R}(\Delta\theta) = \begin{bmatrix} \cos(\Delta\theta) & -\sin(\Delta\theta) \\ \sin(\Delta\theta) & \cos(\Delta\theta) \end{bmatrix}. \tag{6.45}$$

(3) Third, convert all Lagrangian markers $\mathbf{X}_n$ back to the absolute reference frame:

$$\mathbf{X}_n{}^{n+1}|_{abs} = \mathbf{X}_n{}^{n+1}|_{rel} + \mathbf{X}_s|_{abs}. \tag{6.46}$$

3. **Translational and rotational motions:**
This scenario represents a combination of pure translational and pure rotational motions. Therefore, the strategy employed involves first rotating the rigid body within the relative reference frame centred at the solid rigid body centre of mass, followed by translating the rigid body in the absolute reference frame. Note that in this case, the translational motion of the rigid body does not alter the relative position vector of the Lagrangian markers; only the rotational motion influences its value.

Additionally, careful attention must be given to updating the new position of the centre of mass due to translational motion, which is used to compute the relative Lagrangian markers' position. For that reason, the rotation is executed first using the centre of mass from the old time level n, then followed by translation for both the Lagrangian markers and the centre of mass. The steps for updating the positions of the Lagrangian markers, $\mathbf{X}_n$, are outlined below:

(1) First, transform all Lagrangian markers $\mathbf{X}_n$ into the relative reference frame:

$$\mathbf{X}_n{}^n|_{rel} = \mathbf{X}_n{}^n|_{abs} - \mathbf{X}_s{}^n|_{abs}. \tag{6.47}$$

(2) Second, rotate all $\mathbf{X}_n$ in the relative reference frame about $\mathbf{X}_s$:

$$\mathbf{X}_n{}^{n+1}|_{rel}^{rot} = \mathbf{R}(\Delta\theta)\mathbf{X}_n{}^n|_{rel}. \tag{6.48}$$

(3) Third, revert all $\mathbf{X}_n$ back to the absolute reference frame:

$$\mathbf{X}_n{}^{n+1}|_{abs}^{rot} = \mathbf{X}_n{}^{n+1}|_{rel}^{rot} + \mathbf{X}_s{}^n|_{abs}. \tag{6.49}$$

(4) Fourth, the updated position of all $\mathbf{X}_n$ due to the translational motion contribution only is computed in the absolute reference frame as:

$$\mathbf{X}_n{}^{n+1}|_{abs}^{trans} = \mathbf{X}_n{}^n|_{abs} + \Delta\mathbf{X}. \tag{6.50}$$

(5) Fifth, update the global position of all $\mathbf{X}_n$ due to the combined translational and rotational motions in the absolute reference frame:

$$\mathbf{X}_n{}^{n+1}|_{abs} = \mathbf{X}_n{}^{n+1}|_{abs}^{rot} + \mathbf{X}_n{}^{n+1}|_{abs}^{trans}. \tag{6.51}$$

Then, the solid centre of mass position, $\mathbf{X}_s$, is updated due to translation:

$$\mathbf{X}_s{}^{n+1}|_{abs} = \mathbf{X}_s{}^n|_{abs} + \Delta\mathbf{X}. \tag{6.52}$$

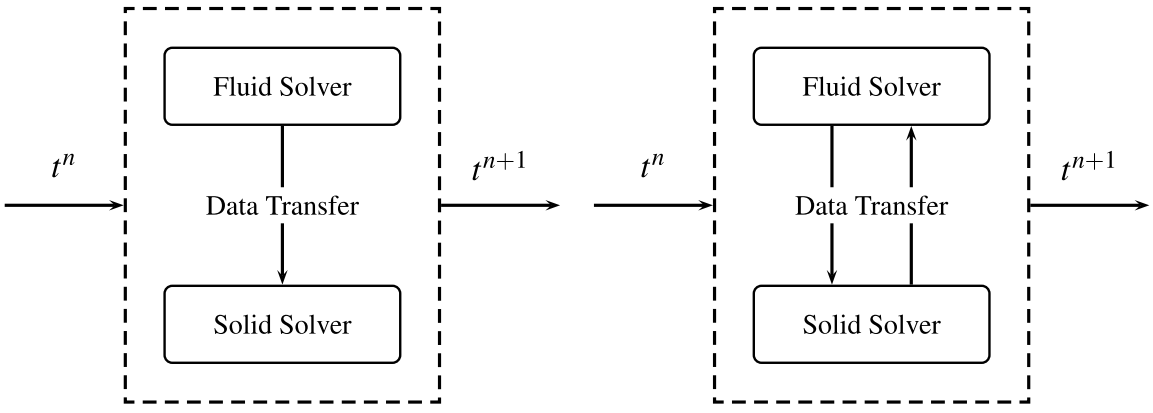
6.5 Fluid–Rigid Body Coupling

A multiphysics problem requires a comprehensive understanding of the coupling between multiple physical domains (also called subproblems). In the context of fluid–rigid body interaction, the physical domains are divided between the fluid dynamics and rigid body dynamics subproblems.

The FSI coupling between a fluid and a rigid body presents a significant multiphysics challenge. It requires accounting for the hydrodynamic forces and torques exerted by the fluid on the rigid body, while simultaneously recognising that the motion of the rigid body modifies the boundary conditions for the fluid flow. This interplay (or interaction) between the fluid

and rigid body creates a dynamic feedback loop complicating the analysis and solution of the system.

Each physical domain or subproblem is addressed using its own dedicated solver: the fluid dynamics subproblem is solved with a fluid solver, while the rigid body dynamics subproblem is handled by a solid solver. In FSI problems two primary approaches classify how data transfer, also referred to as information exchange, is conducted between the two solvers, based on the direction and strength of the interaction. The two approaches are illustrated in Fig. (6.3). The one-way FSI approach is adopted when the influence of one subproblem is weak compared to the other, consequently, the reaction of the other subproblem is neglected. This leads to a weak or loose coupling. In this approach, as depicted in Fig. (6.3a), the data transfer is performed in a uni-directional manner from the fluid solver to the solid solver at each time-step. This approach is popular for being computationally efficient for weakly coupled systems. On the other hand, the two-way FSI approach is employed when the influence between the two subproblems is strong and cannot be neglected. This approach of the data transfer is suitable for strong or tight coupling. In strongly coupled systems, the interaction is considered bi-directional, i.e., a feedback loop is added to the solid solver in order to exchange simultaneously the information between the two solvers at each simulation time-step, as depicted in Fig. (6.3b). The two-way approach is known to deliver accurate results for strongly coupled systems where the one-way approach may fail. However, this increased accuracy comes at the cost of significant computational effort.



(a) One-way: uni-directional, weak (loose) coupling

(b) Two-way: bi-directional, strong (tight) coupling

Figure 6.3: *Different FSI approaches for the data transfer.*

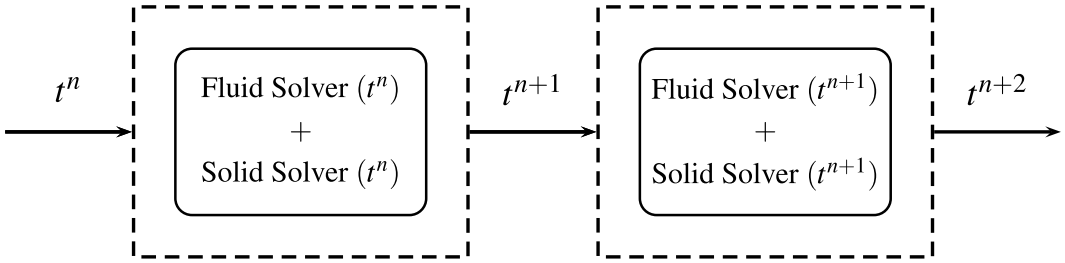
6.5.1 Monolithic vs. Partitioned Coupling Methods

The numerical strategies for solving a multiphysics FSI problem are broadly categorised into two main types of coupling methods [158]: (1) the monolithic coupling methods and (2) the partitioned coupling methods.

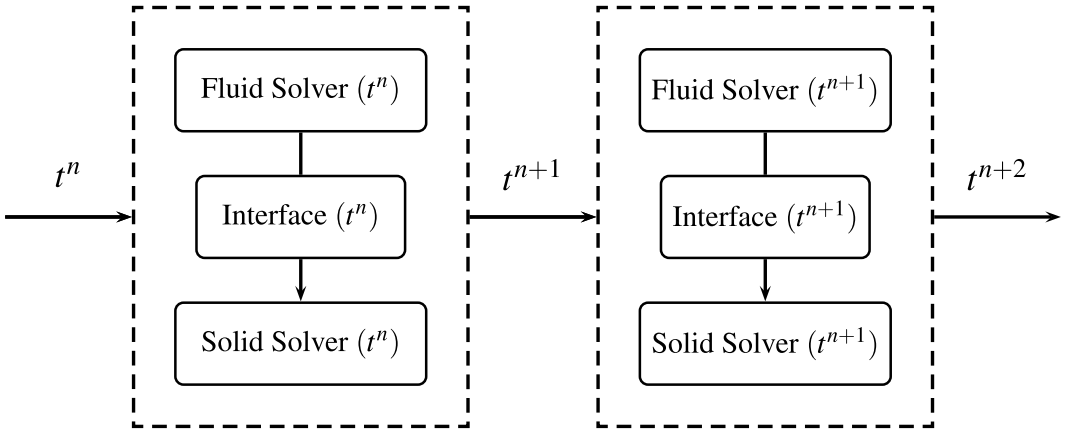
In monolithic coupling methods, the two subproblems are mathematically coupled into a unified framework and discretised in both space and time using the same numerical techniques. This approach produces a large, single system of equations representing the entire problem, which requires a unified algorithm for its solution, as depicted in Fig. (6.4a). The interface conditions, where data exchange occurs, are implicitly enforced, resulting in a two-way strongly coupled system by design. Monolithic methods are renowned for their highly efficient, stable, and accurate solvers. However, this comes at the expense of significant computational development resources and specialised expertise needed to design, implement, and maintain such a unified algorithm. This approach is not considered in the current work, as the objective is to utilise an existing solver; specifically OpenFOAM, in a general context where the solid solver is implemented in a separate library and coupled with the Navier-Stokes fluid solver already integrated within the OpenFOAM environment.

In partitioned coupling methods, the two subproblems are treated independently, each having its own solver, that can be implemented separately in the corresponding software. This strategy allows for flexibility in choosing different discretisation methods in space for each subproblem, such as the finite volume method for the fluid and the finite element method for the solid. Additionally, it supports the use of distinct discretisation techniques for the time domain and even different time-steps for each subproblem. This flexibility is particularly advantageous when addressing the contrasting time scales between the fluid and solid dynamics. For example, accurately resolving fluid flows, especially in turbulent regimes, often requires much smaller time-steps than those needed to capture the motion of the solid. Partitioned coupling efficiently handles this disparity by synchronising the behavior of the two subproblems at selected time-steps, enabling accurate and computationally efficient simulations.

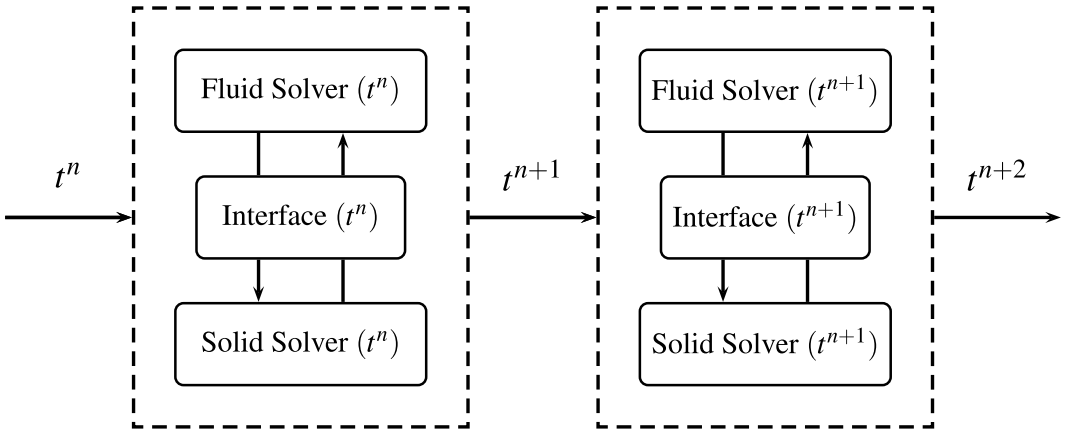
The challenge with the partitioned coupling method lies in handling the interface conditions where information is exchanged between the two subproblems. Since the subproblems are solved independently, an interface condition must be formulated to coordinate the transfer of data between the two ends. This process introduces additional unknown variables at the interface, increasing the storage requirements for the simulation. The efficiency of a partitioned coupling algorithm is significantly influenced by the size of the interface, making it a key factor in the method's performance. An extensive amount of research can be found



(a) Monolithic method: implicit, two-way strongly (tighly) coupled system



(b) Partitioned method: explicit, one-way weakly (loosely) coupled system



(c) Partitioned method: implicit, two-way strongly (tighly) coupled system

Figure 6.4: *Different FSI coupling methods.*

on the development and optimisation of coupling algorithms for the partitioned approach [2, 159–165]. The coupling algorithms can be classified into two groups based on the data transfer mechanism adopted: (1) the explicit coupling associated with the one-way data transfer

approach for weakly (or loosely) coupled systems, and (2) the implicit coupling related to the two-way approach for the data transfer found for strongly (or tightly) coupled systems, as illustrated in Figs. (6.4b) and (6.4c), respectively.

For the explicit coupling, also named staggered coupling, the subproblems are solved sequentially, at each time increment without the need for a feedback loop between the subproblems, leading to a single solution per subproblem per time-step (one-way approach). As shown in Figs. (6.4b), the fluid is solved at t^{n+1} with all variables taken from t^n, afterward the interface condition is computed using values obtained from the fluid solver, and finally the solid is solved from the updated interface condition. The method is renowned for being computationally efficient compared to its counterpart method. However, the interface condition can never be accurately satisfied in the explicit coupling due to spurious numerical energy created at the fluid–solid interface.

For implicit coupling, also referred to as iterative coupling, the subproblems are solved iteratively within each time-step until a convergence criterion is met. This iterative process ensures accurate enforcement of the interface condition. As illustrated in Fig. (6.4c), the steps outlined for explicit coupling are repeated at each time increment until the residual between two consecutive iterations falls below a prescribed threshold (two-way approach). Implicit coupling can be implemented using either a fixed-point strategy or a Newton-based (or quasi-Newton) strategy [158]. The fixed-point strategy is widely recognised for its simplicity but tends to converge slowly due to the large number of iterations required. In contrast, the Newton-based strategy generally converges in fewer iterations but incurs a higher computational cost per iteration, as it involves constructing and evaluating the Jacobian matrix associated with the fluid–solid coupled equations. The fixed-point strategy is often preferred because its convergence rate can be significantly improved using relaxation techniques, also known as acceleration techniques, which enhance the efficiency and stability of the fixed-point iterations [166, 167]. Among these techniques, the fixed relaxation is considered the simplest one, which applies a constant relaxation parameter throughout the iterations. On the other hand, dynamic relaxation methods exist, that provide a more flexible approach by adjusting the relaxation parameter at each iteration, offering improved performance by adapting to the changing conditions of the iterative process. Notable examples include the Aitken relaxation [168], which uses the residuals from the last two iterations to improve the current solution, and the steepest descent relaxation [169], which computes an optimal iteration dependent relaxation parameter to minimise a specific cost function. These relaxation techniques have demonstrated substantial improvements in terms of convergence rate

and computational stabilisation [158, 166, 167, 170]. It is worth noting that a fixed-point strategy implemented without the use of any relaxation techniques is commonly referred to as a simple fixed-point strategy.

6.5.2 Internal Mass Effect (IME) Challenges

Fluid–structure interaction problems involving incompressible viscous fluids are particularly challenging due to the IME, which becomes prominent when the fluid density closely matches the solid density [171–174], i.e., for $\rho_s/\rho_f \approx 1$. The IME represents the inertia added to the system from the fluid's acceleration around a moving body and manifests as numerical difficulties in the FSI realm, especially in incompressible flows.

Explicit coupling methods, typically used for weakly coupled systems, encounter numerical instabilities in the presence of strong IME because they fail to enforce the interface conditions accurately [171, 172, 174]. This failure leads to spurious numerical power at the fluid–solid interface, resulting in an unstable coupling [172]. In contrast, implicit coupling methods, which are designed to preserve energy balance, are more stable for strongly coupled systems. They effectively suppress the spurious power on the fluid–solid interface by enforcing the interface condition more accurately. Nevertheless, implicit methods can encounter convergence issues during the iterative process under significant IME [174]. Fig. (6.5) highlights the difficulties associated with partitioned coupling for FSI problems in the presence of IME.

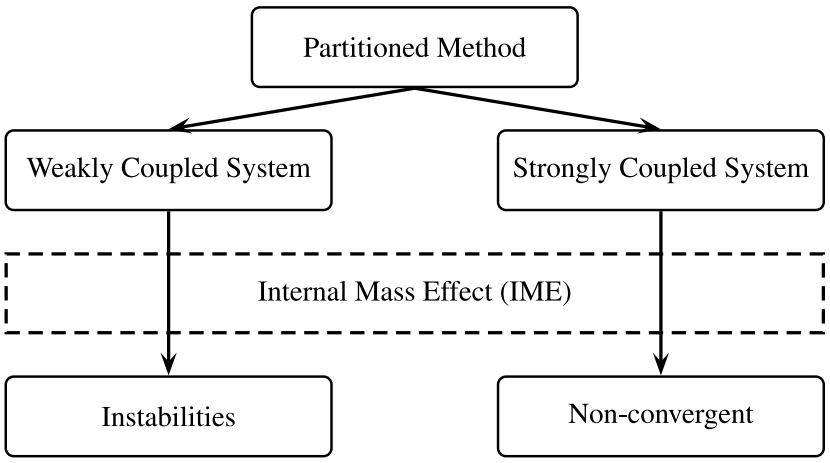


Figure 6.5: *Numerical challenges induced by the IME in the FSI partitioned coupling methods.*

These challenges are particularly critical in biomechanics and biomedical applications, such as blood flow simulations, where the IME is unavoidable due to the similar densities

of the fluid and solid. Furthermore, reducing the time-step fails to resolve these instabilities [172], highlighting the fundamental nature of the issue. In contrast, for compressible flows where the IME is less significant, explicit coupling methods are more stable, as seen in aeroelastic applications where interface conditions are less stringent [172, 174].

For fluid–rigid body interaction and in the context of the IBM, it was found that these difficulties are further exacerbated by the rigid body approximation for the IME when the rigid body is undergoing rotational dynamics, which forces the internal fluid (or mass) to follow the rigid body motion of the immersed boundary, regardless of the true induced motion of the internal flow field. This behavior, depicted by Eq. (6.19), highlights a key source of instability when simulating such interactions.

6.5.3 Present Coupling Algorithm in the DF-IBM Framework

In this section, two efficient coupling algorithms are proposed under the partitioned method category: (1) an explicit coupling algorithm designed for weakly coupled systems and (2) an implicit coupling algorithm for strongly coupled systems.

For both algorithms, the IME is treated explicitly using Eqs. (6.29a) and (6.29b). By doing so, the density ratio limitation encountered in the Newton-Euler equations of motion is eliminated. Theoretically speaking, this allows the simulation of neutrally buoyant solids $\rho_s/\rho_f = 1$, non-buoyant (dense) solids $\rho_s/\rho_f > 1$, as well as buoyant (light) solids $\rho_s/\rho_f < 1$.

In the IBM framework, the fluid domain is represented on a fixed Eulerian mesh that remains fixed and does not adapt to the motion of the solid body. The solid motion is expressed in the Lagrangian domain, and its influence on the fluid is incorporated by introducing a boundary force term into the fluid's governing equations. This decoupled representation allows for greater flexibility for the fluid–rigid body coupling, when implemented inside the predictor-corrector strategy of the PISO solution algorithm. The proposed coupling methods leverage the present implicit DF-IBM Algorithm 1 already developed in Chapter 4, comprised of three new major steps, listed as: (1) momentum predictor, (2) fluid–rigid body coupling, and (3) PISO corrector loops, to improve the overall efficiency of the current FSI coupling methods.

The two subproblems and the interface condition formulated for the fluid–rigid body coupling partition method, are defined as follows:

Subproblem 1:

- Defined as the fluid solver.
- Solved on the Eulerian domain Ω_f (fixed and non-deformable).
- Fluid governing equations: Eqs. (6.22a) and (6.22b).

- System unkowns: $(\mathbf{u}_f{}^{n+1}, p^{n+1})$.

Interface:

- Defined for the interface condition via the DF-IBM.
- Solved on the Lagrangian domain Γ_s (moving), referred to as the immersed boundary.
- Fluid–solid interface no-slip velocity boundary condition: Eq. (6.22c).
- Interaction (IBM-related linear operators) equations: Eqs. (6.22d) and (6.22e).
- System unkowns: $(\mathbf{X}_n{}^{n+1}, \mathbf{U}_n{}^{n+1}, \mathbf{F}_n{}^{n+1})$.

Subproblem 2:

- Defined as the solid rigid body solver.
- Solved on the Lagrangian domain Ω_s (moving).
- Solid rigid body governing equations: Eqs. (6.22f) and (6.22g).
- System unkowns: $(\mathbf{X}_s{}^{n+1}, \mathbf{U}_s{}^{n+1}, \omega_s{}^{n+1})$.

The two subproblems are joined together through the fluid–solid interface Γ_s condition. The fluid unknowns are stored at the fluid cell centroid, the solid unknowns are stored at the solid rigid body centre of mass, and the fluid–solid interface variables are stored at each Lagrangian marker.

Analysing the subproblems and the interface condition formulated in this work, it becomes evident that the fluid is coupled to the solid rigid body through the boundary force term, $\mathbf{F}_n$, which acts at the fluid–solid interface. This boundary force term is partially derived from the neighbouring fluid Eulerian cells and partially from the variables associated with the solid rigid body (refer to direct-forcing step Eq. (4.18)). This approach contrasts with the traditional coupling found in body conforming mesh methods, such as the ALE method, where the interface coupling is achieved through the direct interaction of the pressure field, p, across the matched interface.

This analysis reveals two critical findings regarding the FSI implicit coupling method. First, it shows that the momentum predictor step and the PISO corrector loops can be excluded from the FSI iterations adopted inside the PISO predictor-corrector algorithm for strongly coupled systems. The momentum prediction step can be omitted since the fluid governing equations are solved on a fixed Eulerian mesh where the boundary force is not yet incorporated. Additionally, the PISO corrector loops involve solving the PPE, notoriously the most computationally demanding step in the entire solution process, this suggested exclusion significantly enhances the efficiency of the FSI implicit coupling compared to conventional implicit coupling methods, like those used in the ALE formulation.

Second, given that the Lagrangian domain is substantially smaller compared to the Eulerian domain, updating the Lagrangian mesh and its associated variables at each time-step is therefore computationally affordable and faster once compared to the mesh and variables heavy updates required in the ALE formulation. Together, these advantages mitigate the computational burden encountered in the traditional body conforming methods and highlight the efficiency of the proposed approach within the DF-IBM framework for FSI problems.

In a similar fashion to Algorithm 1, the fluid–rigid body system dual constraints problem formed for an incompressible viscous flow interacting with a solid rigid body is satisfied sequentially in the solution process of the PISO predictor-corrector algorithm:

(1) Divergence-free condition on the fluid domain: $\nabla \cdot \mathbf{u}_f{}^{n+1} = 0$ in Ω_f,

(2) Fluid–solid interface no-slip velocity boundary condition based on Eq. (6.22c) and Eq. (6.22d) in the DF-IBM framework: $\mathbf{U}_n{}^{n+1} = \mathscr{I}[\tilde{\mathbf{u}}_f]$ on Γ_s, having $\mathbf{U}_n = \mathbf{U}_s + \omega_s \times \mathbf{r}_n$.

The second fluid–solid interface condition has two physical meanings, first to enforce the DF-IBM no-slip velocity boundary condition, and second, to ensure that the fluid–solid interface is accurately following the solid rigid body motion. The interface condition can be either, weakly coupled as in the case of explicit coupling, hence the condition is expressed as:

$$\begin{aligned} \mathbf{U}_n{}^{n+1} &= \mathbf{U}_s{}^{n} + \omega_s{}^{n} \times \mathbf{r}_n{}^{n} \\ &= \mathscr{I}[\tilde{\mathbf{u}}_f], \end{aligned} \tag{6.53}$$

where the RBD variables are used from the previous time level t^n, or the same condition can be exactly and accurately enforced in the case of implicit coupling for the strongly coupled schemes, therefore:

$$\begin{aligned} \mathbf{U}_n{}^{n+1} &= \mathbf{U}_s{}^{n+1} + \omega_s{}^{n+1} \times \mathbf{r}_n{}^{n+1} \\ &= \mathscr{I}[\tilde{\mathbf{u}}_f], \end{aligned} \tag{6.54}$$

with the rigid body variables taken from the current time-step t^{n+1}.

The following sections detail the implementation of the explicit and implicit coupling algorithms for fluid–rigid body interactions within the PISO solution algorithm. The developed algorithms are considered as an extension of the previously proposed implicit DF-IBM Algorithm 1, to enforce the interface condition in weakly coupled and strongly coupled

systems, respectively. These implementations are tailored to the unique requirements of the DF-IBM framework, ensuring robust and efficient fluid–rigid body coupling.

6.5.3.1 Explicit Coupling Algorithm for Weakly Coupled Systems

In weakly coupled schemes, the fluid–solid interface condition is not exactly satisfied at each time-step. Each subproblem is solved once per time-step.

Based on Algorithm 1 and according to previous findings, the momentum predictor and PISO corrector loops remain unchanged. To emphasise on the new developments, these steps are omitted from the following algorithm. In this implementation, the DF-IBM iterations step found in Algorithm 1 is replaced with the fluid–rigid body coupling step. The following FSI procedure is performed at each time-step, where the superscript n and $n+1$ denote the past and current time levels, respectively:

1. Momentum Predictor:
   ... (Refer to Algorithm 1)

2. Fluid–Rigid Body Explicit Coupling:

   (a) Build the Lagrangian support domain $\mathscr{D}_{\mathbf{X}_n}$ for all Lagrangian markers from time level t^n, i.e., for all ${\mathbf{X}_n}^n$.

   (b) Determine the desired Lagrangian velocity on the fluid–solid interface, Γ_s, using the rigid body motion computed at time-step t^n, as described for the weakly coupled system in Eq. (6.53):

$$\begin{aligned} {\mathbf{U}_n}^{n+1}\left({\mathbf{X}_n}^n\right) &= {\mathbf{U}_s}^n + {\omega_s}^n \times {\mathbf{r}_n}^n\left({\mathbf{X}_n}^n\right) \\ &= {\mathbf{U}_s}^n + {\omega_s}^n \times \left({\mathbf{X}_n}^n - {\mathbf{X}_s}^n\right). \end{aligned} \tag{6.55}$$

   (c) Inner implicit DF-IBM iterations start for $m = 1$, having $\tilde{\mathbf{u}}_f^{(0)} = {\mathbf{u}_f}^*$ and ${\mathbf{F}_n}^{(0)}\left({\mathbf{X}_n}^n\right) = {\mathbf{F}_n}^n\left({\mathbf{X}_n}^n\right)$. The iterative procedure is executed through the following steps:

      (i) Eulerian Velocity Interpolation:

$$\mathbf{U}^{(m-1)}\left({\mathbf{X}_n}^n\right) = \mathscr{I}[\tilde{\mathbf{u}}_f^{(m-1)}], \tag{6.56}$$

(ii) Direct-Forcing:

$$\mathbf{F}_n{}^{(m)}\left(\mathbf{X}_n{}^n\right)=\frac{\mathbf{U}_n{}^{n+1}\left(\mathbf{X}_n{}^n\right)-\mathbf{U}^{(m-1)}\left(\mathbf{X}_n{}^n\right)}{\Delta t}+\mathbf{F}_n{}^{(m-1)}\left(\mathbf{X}_n{}^n\right), \tag{6.57}$$

(iii) Lagrangian Force Spreading:

$$\mathbf{f}^{(m)}=\mathscr{S}[\mathbf{F}_n{}^{(m)}\left(\mathbf{X}_n{}^n\right)], \tag{6.58}$$

(iv) Update Eulerian Velocity:

$$a_P\tilde{\mathbf{u}}_{f_P}^{(m)}+\sum_N a_N\tilde{\mathbf{u}}_{f_N}^{(m)}=\mathbf{RHS}\left(\mathbf{u}_f{}^n,\mathbf{u}_f{}^{n-1}\right)-(\nabla p)^n+\mathbf{f}^{(m)}, \tag{6.59}$$

(v) Fluid-solid interface no-slip velocity boundary condition:

$$\left\|E_{no-slip}\right\|_2=\left\|E_{\tilde{\mathbf{u}}_f{}^{(m)}}\right\|_2-\left\|E_{\tilde{\mathbf{u}}_f{}^{(m-1)}}\right\|_2<\epsilon_{IBM}, \tag{6.60}$$

with

$$\left\|E_{\tilde{\mathbf{u}}_f{}^{(m)}}\right\|_2=\left\|\mathbf{U}_n{}^{n+1}\left(\mathbf{X}_n{}^n\right)-\mathbf{U}^{(m)}\left(\mathbf{X}_n{}^n\right)\right\|_2,$$

and

$$\left\|E_{\tilde{\mathbf{u}}_f{}^{(m-1)}}\right\|_2=\left\|\mathbf{U}_n{}^{n+1}\left(\mathbf{X}_n{}^n\right)-\mathbf{U}^{(m-1)}\left(\mathbf{X}_n{}^n\right)\right\|_2,$$

where ϵ_{IBM} is a user-specified tolerance for the implicit DF-IBM iterations.

(vi) If the defined convergence criterion is reached in *Step 2 (v)*, then

$$\mathbf{f}^{n+1}=\mathbf{f}^{(m)},\ \tilde{\mathbf{u}}_f=\tilde{\mathbf{u}}_f^{(m)},\ \text{and}\ \mathbf{F}_n{}^{n+1}\left(\mathbf{X}_n{}^n\right)=\mathbf{F}_n{}^{(m)}\left(\mathbf{X}_n{}^n\right), \tag{6.61}$$

otherwise, return to *Step 2 (i)*, and let $m=m+1$.
It can be noted that after this step, the DF-IBM no-slip velocity boundary condition is fulfilled, $\mathbf{U}^{(m)}\left(\mathbf{X}_n{}^n\right)=\mathscr{I}[\tilde{\mathbf{u}}_f^{(m)}]=\mathbf{U}_n{}^{n+1}\left(\mathbf{X}_n{}^n\right)$.

(d) Solve the Newton-Euler equations of motion for the linear and angular velocities

of the solid rigid body in Ω_s, based on Eqs. (6.30a) and (6.30b):

$$\mathbf{U}_s{}^{n+1,*} = \Delta t \left(\frac{\rho_s - \rho_f}{\rho_s}\mathbf{g} - \frac{\rho_f}{\rho_s V_s} \sum_{\forall \mathbf{X}_n \in \Gamma_s} \mathbf{F}_n{}^{n+1}\left(\mathbf{X}_n{}^n\right) W_n + \frac{1}{\rho_s V_s}\mathbf{F}_c{}^{n+1}\left(\mathbf{X}_n{}^n\right) \right) + \frac{\rho_s + \rho_f}{\rho_s}\mathbf{U}_s{}^n - \frac{\rho_f}{\rho_s}\mathbf{U}_s{}^{n-1}, \quad (6.62a)$$

$$\omega_s{}^{n+1,*} = \Delta t \left(-\frac{\rho_f}{I_s} \sum_{\forall \mathbf{X}_n \in \Gamma_s} \mathbf{r}_n{}^n\left(\mathbf{X}_n{}^n\right) \times \mathbf{F}_n{}^{n+1}\left(\mathbf{X}_n{}^n\right) W_n \right) + \frac{\rho_s + \rho_f}{\rho_s}\omega_s{}^n - \frac{\rho_f}{\rho_s}\omega_s{}^{n-1}. \quad (6.62b)$$

(e) Relaxation of the linear and angular velocities, using fixed relaxation technique:

$$\mathbf{U}_s{}^{n+1} = \alpha_r \mathbf{U}_s{}^{n+1,*} + (1-\alpha_r)\mathbf{U}_s{}^n, \quad (6.63a)$$

$$\omega_s{}^{n+1} = \alpha_r \omega_s{}^{n+1,*} + (1-\alpha_r)\omega_s{}^n, \quad (6.63b)$$

where α_r is the fixed relaxation parameter, with $0 < \alpha_r \leq 1$. For $\alpha_r = 1$, no relaxation occurs, i.e., $(.)_s{}^{n+1} = (.)_s{}^{n+1,*}$.

(f) Update the solid rigid body position using Eqs. (6.47)–(6.51) for all Lagrangian markers $\mathbf{X}_n{}^{n+1}$ on Γ_s and Eq. (6.52) for its centre of mass $\mathbf{X}_s{}^{n+1}$ in Ω_s, in case of combined translational and rotational dynamics.

3. PISO Corrector Loops:
   ... (Refer to Algorithm 1)

Notably, due to the explicit coupling, the position of the solid rigid body is not known in advance at t^{n+1}. Consequently, the velocity interpolation and force spreading operators are performed on the Lagrangian markers from the previous time-step, t^n, i.e., using $\mathbf{X}_n{}^n$. As a result, the discrete Dirac delta function is also based on $\mathbf{X}_n{}^n$, introducing in theory, a time-lagged operators that reduce the overall time accuracy of the presented method to first-order, as in [95, 175]. In addition, the collision force $\mathbf{F}_c$ and the position vector $\mathbf{r}_n$, are all based on the Lagrangian markers and the solid rigid body centre of mass at the old time level, i.e., $\mathbf{X}_n{}^n$ and $\mathbf{X}_s{}^n$.

To mitigate the instabilities experienced with explicit coupling algorithms, particularly

in cases involving neutrally buoyant $\rho_s/\rho_f \approx 1$ and buoyant $\rho_s/\rho_f < 1$ solids, the fixed relaxation technique was found to be effective in stabilising computations by relaxing high velocity fluctuation rates. Furthermore, to address additional difficulties introduced by the rigid body approximation for the IME in rotational dynamics, the same relaxation technique can be applied to the angular velocity when fluid–rigid body interactions induce rotational motion. The fluid–rigid body explicit coupling algorithm implemented within the PISO predictor-corrector strategy is summarised in Algorithm 2.

Algorithm 2: Present FSI explicit coupling algorithm.

Given: fluid $(\mathbf{u}_f{}^n, p^n)$, interface $(\mathbf{X}_n{}^n, \mathbf{U}_n{}^n, \mathbf{F}_n{}^n)$, and solid $(\mathbf{X}_s{}^n, \mathbf{U}_s{}^n, \omega_s{}^n)$;
(Fluid): Predict initial Eulerian velocity $\mathbf{u}_f{}^*$ using Eq. (4.26);
(Interface): Build Lagrangian support domain $\mathscr{D}_{\mathbf{X}_n}, \forall\, \mathbf{X}_n{}^n$;
(Interface): Compute desired Lagrangian velocity $\mathbf{U}_n{}^{n+1}$ using Eq. (6.55);
Initialise: $m = 1$, $\tilde{\mathbf{u}}_f{}^{(0)} = \mathbf{u}_f{}^*$, and $\mathbf{F}_n{}^{(0)} = \mathbf{F}_n{}^n$;
while $m \leq m_{max}$ **do**
    (Interface): Interpolate Eulerian velocity to obtain $\mathbf{U}^{(m-1)}$ using Eq. (6.56);
    (Interface): Compute singular Lagrangian force $\mathbf{F}_n{}^{(m)}$ using Eq. (6.57);
    (Interface): Spread Lagrangian force to obtain $\mathbf{f}^{(m)}$ using Eq. (6.58);
    (Interface): Update the predicted Eulerian velocity $\tilde{\mathbf{u}}_f{}^{(m)}$ using Eq. (6.59);
    if $\|E_{no-slip}\|_2 < \epsilon_{IBM}$ **then**
        $\mathbf{f}^{n+1} = \mathbf{f}^{(m)}$, $\tilde{\mathbf{u}}_f = \tilde{\mathbf{u}}_f{}^{(m)}$, and $\mathbf{F}_n{}^{n+1} = \mathbf{F}_n{}^{(m)}$;
        break;
    else
        $m = m + 1$;
    end
end
(Solid): Compute Newton-Euler equations of motion to obtain $\mathbf{U}_s{}^{n+1}$ and $\omega_s{}^{n+1}$ using Eqs. (6.62a) and (6.62b);
(Solid): Relax $\mathbf{U}_s{}^{n+1}$ and $\omega_s{}^{n+1}$ when necessary using Eqs. (6.63a) and (6.63b);
(Solid): Update $\mathbf{X}_s{}^{n+1}$ using Eq. (6.52);
(Interface): Update $\mathbf{X}_n{}^{n+1}$ using Eqs. (6.47)–(6.51);
Initialise: $m_{max} = 3$ and $\hat{\mathbf{u}}_f{}^{(0)} = \tilde{\mathbf{u}}_f$;
for $m = 1$ *to* m_{max} **do**
    (Fluid): Solve mPPE for $p^{(m)}$ using Eq. (4.34);
    (Fluid): Correct Eulerian velocity $\hat{\mathbf{u}}_f{}^{(m)}$ using Eq. (4.35);
    if $m = m_{max}$ **then**
        $\mathbf{u}_f{}^{n+1} = \hat{\mathbf{u}}_f{}^{(m_{max})}$ and $p^{n+1} = p^{(m_{max})}$;
    end
end

6.5.3.2 Implicit Coupling Algorithm for Strongly Coupled Systems

In strongly coupled schemes, the fluid–solid interface condition is enforced with high accuracy at each time-step. This requires an iterative process between the fluid and rigid body solvers during each simulation time-step.

The proposed implicit coupling algorithm differs from Algorithm 2 by operating in a strongly coupled manner, i.e., by enforcing the interface condition accurately. Consequently, an iterative procedure is employed to satisfy the interface condition described by Eq. (6.54). Analogously to Algorithm 2, the FSI implicit coupling procedure for each time-step is outlined as follows:

1. Momentum Predictor:
   ... (Refer to Algorithm 1)

2. Fluid–Rigid Body Implicit Coupling:
   Outer implicit FSI iterations start for $k = 1$, having $\mathbf{X}_n{}^{(0)} = \mathbf{X}_n{}^n$, $\mathbf{X}_s{}^{(0)} = \mathbf{X}_s{}^n$, $\mathbf{U}_s{}^{(0)} = \mathbf{U}_s{}^n$, and $\omega_s{}^{(0)} = \omega_s{}^n$. The implicit coupling of the fluid–rigid body interaction is carried out using the following steps:

   (a) Build the Lagrangian support domain $\mathscr{D}_{\mathbf{X}_n}$ for all Lagrangian markers $\mathbf{X}_n{}^{(k-1)}$.

   (b) Determine the desired Lagrangian velocity on the fluid–solid interface, Γ_s, using the rigid body motion computed at time-step t^{n+1}, as described for the strongly coupled system in Eq. (6.54):

$$\begin{aligned}\mathbf{U}_n{}^{(k-1)}\big(\mathbf{X}_n{}^{(k-1)}\big) &= \mathbf{U}_s{}^{(k-1)} + \omega_s{}^{(k-1)} \times \mathbf{r}_n{}^{(k-1)}\big(\mathbf{X}_n{}^{(k-1)}\big) \\ &= \mathbf{U}_s{}^{(k-1)} + \omega_s{}^{(k-1)} \times \big(\mathbf{X}_n{}^{(k-1)} - \mathbf{X}_s{}^{(k-1)}\big).\end{aligned} \tag{6.64}$$

   (c) Inner implicit DF-IBM iterations start for $m = 1$, having $\tilde{\mathbf{u}}_f{}^{(k=0,m=0)} = \mathbf{u}_f{}^*$ and $\mathbf{F}_n{}^{(k,m=0)}\big(\mathbf{X}_n{}^{(k-1)}\big) = \mathbf{F}_n{}^n\big(\mathbf{X}_n{}^n\big)$. The iterative procedure is executed through the following steps:

      (i) Eulerian Velocity Interpolation:

$$\mathbf{U}^{(k,m-1)}\big(\mathbf{X}_n{}^{(k-1)}\big) = \mathscr{I}[\tilde{\mathbf{u}}_f{}^{(k,m-1)}], \tag{6.65}$$

(ii) Direct-Forcing:

$$\mathbf{F}_n{}^{(k,m)}\left(\mathbf{X}_n{}^{(k-1)}\right)=\frac{\mathbf{U}_n{}^{(k-1)}\left(\mathbf{X}_n{}^{(k-1)}\right)-\mathbf{U}^{(k,m-1)}\left(\mathbf{X}_n{}^{(k-1)}\right)}{\Delta t} + \mathbf{F}_n{}^{(k,m-1)}\left(\mathbf{X}_n{}^{(k-1)}\right), \quad (6.66)$$

(iii) Lagrangian Force Spreading:

$$\mathbf{f}^{(k,m)}=\mathscr{S}[\mathbf{F}_n{}^{(k,m)}\left(\mathbf{X}_n{}^{(k-1)}\right)], \quad (6.67)$$

(iv) Update Eulerian Velocity:

$$a_P\tilde{\mathbf{u}}_{f_P}^{(k,m)}+\sum_N a_N\tilde{\mathbf{u}}_{f_N}^{(k,m)}=\mathbf{RHS}\left(\mathbf{u}_f{}^n,\mathbf{u}_f{}^{n-1}\right)-(\nabla p)^n+\mathbf{f}^{(k,m)}, \quad (6.68)$$

(v) Fluid-solid interface no-slip velocity boundary condition:

$$\left\|E_{no-slip}\right\|_2=\left\|E_{\tilde{\mathbf{u}}_f{}^{(k,m)}}\right\|_2-\left\|E_{\tilde{\mathbf{u}}_f{}^{(k,m-1)}}\right\|_2<\epsilon_{IBM}, \quad (6.69)$$

with

$$\left\|E_{\tilde{\mathbf{u}}_f{}^{(k,m)}}\right\|_2=\left\|\mathbf{U}_n{}^{(k-1)}\left(\mathbf{X}_n{}^{(k-1)}\right)-\mathbf{U}^{(k,m)}\left(\mathbf{X}_n{}^{(k-1)}\right)\right\|_2,$$

and

$$\left\|E_{\tilde{\mathbf{u}}_f{}^{(k,m-1)}}\right\|_2=\left\|\mathbf{U}_n{}^{(k-1)}\left(\mathbf{X}_n{}^{(k-1)}\right)-\mathbf{U}^{(k,m-1)}\left(\mathbf{X}_n{}^{(k-1)}\right)\right\|_2,$$

where ϵ_{IBM} is a user-specified tolerance for the implicit DF-IBM iterations.

(vi) If the defined convergence criterion is reached in *Step 2 (v)*, then

$$\mathbf{f}^{(k)}=\mathbf{f}^{(k,m)},\ \tilde{\mathbf{u}}_f{}^{(k)}=\tilde{\mathbf{u}}_f{}^{(k,m)},\ \text{and}\ \mathbf{F}_n{}^{(k)}\left(\mathbf{X}_n{}^{(k-1)}\right)=\mathbf{F}_n{}^{(k,m)}\left(\mathbf{X}_n{}^{(k-1)}\right), \quad (6.70)$$

otherwise, return to *Step 2 (i)*, and let $m=m+1$.

It can be noted that after this step, the DF-IBM no-slip velocity boundary condition is fulfilled, $\mathbf{U}^{(k,m)}\left(\mathbf{X}_n{}^{(k-1)}\right)=\mathscr{I}[\tilde{\mathbf{u}}_f{}^{(k,m)}]=\mathbf{U}_n{}^{(k-1)}\left(\mathbf{X}_n{}^{(k-1)}\right)$.

(d) Solve the Newton-Euler equations of motion for the linear and angular velocities

of the solid rigid body in Ω_s, based on Eqs. (6.30a) and (6.30b):

$$\mathbf{U}_s{}^{(k),*} = \Delta t \left(\frac{\rho_s - \rho_f}{\rho_s}\mathbf{g} - \frac{\rho_f}{\rho_s V_s} \sum_{\forall \mathbf{X}_n \in \Gamma_s} \mathbf{F}_n{}^{(k)}\left(\mathbf{X}_n{}^{(k-1)}\right) W_n + \frac{1}{\rho_s V_s}\mathbf{F}_c{}^{(k)}\left(\mathbf{X}_n{}^{(k-1)}\right) \right) + \frac{\rho_s + \rho_f}{\rho_s}\mathbf{U}_s{}^n - \frac{\rho_f}{\rho_s}\mathbf{U}_s{}^{n-1}, \quad (6.71a)$$

$$\omega_s{}^{(k),*} = \Delta t \left(-\frac{\rho_f}{I_s} \sum_{\forall \mathbf{X}_n \in \Gamma_s} \mathbf{r}_n{}^{(k-1)}\left(\mathbf{X}_n{}^{(k-1)}\right) \times \mathbf{F}_n{}^{(k)}\left(\mathbf{X}_n{}^{(k-1)}\right) W_n \right) + \frac{\rho_s + \rho_f}{\rho_s}\omega_s{}^n - \frac{\rho_f}{\rho_s}\omega_s{}^{n-1}. \quad (6.71b)$$

(e) Relaxation of the linear and angular velocities, using fixed relaxation technique:

$$\mathbf{U}_s{}^{(k)} = \alpha_r \mathbf{U}_s{}^{(k),*} + (1-\alpha_r)\mathbf{U}_s{}^n, \quad (6.72a)$$

$$\omega_s{}^{(k)} = \alpha_r \omega_s{}^{(k),*} + (1-\alpha_r)\omega_s{}^n, \quad (6.72b)$$

where α_r is the fixed relaxation parameter, with $0 < \alpha_r \leq 1$. For $\alpha_r = 1$, no relaxation occurs, i.e., $(.)_s{}^{(k)} = (.)_s{}^{(k),*}$.

(f) Update the solid rigid body position using Eqs. (6.47)–(6.51) for all Lagrangian markers $\mathbf{X}_n{}^{(k)}$ on Γ_s and Eq. (6.52) for its centre of mass $\mathbf{X}_s{}^{(k)}$ in Ω_s, in case of combined translational and rotational dynamics.

(g) Check if the residuals of the implicit coupling fall below a certain threshold to exit the fluid–rigid body implicit coupling step:

$$\mathbf{R}_1 < \epsilon_{FSI} \quad \text{and} \quad \mathbf{R}_2 < \epsilon_{FSI}, \quad (6.73)$$

where ϵ_{FSI} is a user-specified tolerance for the FSI iterations. The residuals $\mathbf{R}_1$ and $\mathbf{R}_2$ are defined for the linear and angular velocities of the rigid body, respectively:

$$\mathbf{R}_1 = \frac{\left\| \mathbf{U}_s{}^{(k)} - \mathbf{U}_s{}^{(k-1)} \right\|}{\left\| \mathbf{U}_s{}^{(k)} \right\|}, \quad (6.74a)$$

$$\mathbf{R_2} = \frac{\left\| \omega_s{}^{(k)} - \omega_s{}^{(k-1)} \right\|}{\left\| \omega_s{}^{(k)} \right\|}. \quad (6.74b)$$

If the convergence criteria are met, then:

$$\mathbf{X}_s{}^{n+1} = \mathbf{X}_s{}^{(k)},\ \mathbf{U}_s{}^{n+1} = \mathbf{U}_s{}^{(k)},\ \text{and}\ \boldsymbol{\omega}_s{}^{n+1} = \boldsymbol{\omega}_s{}^{(k)}, \tag{6.75a}$$

$$\mathbf{X}_n{}^{n+1} = \mathbf{X}_n{}^{(k)},\ \mathbf{U}_n{}^{n+1} = \mathbf{U}_s{}^{n+1} + \boldsymbol{\omega}_s{}^{n+1} \times \mathbf{r}_n{}^{n+1},\ \text{and}\ \mathbf{F}_n{}^{n+1} = \mathbf{F}_n{}^{(k)}. \tag{6.75b}$$

Otherwise, return to *Step 2 (a)* and let $k = k + 1$.

3. PISO Corrector Loops:
   ... (Refer to Algorithm 1)

From a theoretical standpoint, the error introduced by the time-lagged operators diminishes owing to the fluid–rigid body implicit coupling found in the present FSI algorithm. Due to the strong coupling, the fluid–solid interface is accurately enforced using the values of the solid rigid body from the new time level t^{n+1}. Furthermore, in the final iteration of the FSI coupling, the collision force $\mathbf{F}_c$ and the position vector $\mathbf{r}_n$ are calculated based on the Lagrangian markers and the rigid body centre of mass from the new time level, i.e., $\mathbf{X}_n{}^{n+1}$ and $\mathbf{X}_s{}^{n+1}$. This approach ensures greater accuracy compared to the explicit coupling method.

Similarly to the explicit coupling approach, the present implicit algorithm employs the fixed relaxation technique to address two primary challenges leading to a non-convergence of the implicit FSI iterations: (1) the non-convergence issues encountered with neutrally buoyant $\rho_s/\rho_f \approx 1$ and buoyant $\rho_s/\rho_f < 1$ solids, and (2) the convergence difficulties arising from the rigid body approximation of the IME in rotational dynamics.

Finally, since the Lagrangian domain is significantly smaller than the Eulerian domain, ergo updating and building the Lagrangian support domain during each FSI iteration is considered way less costly from a computational point of view compared to the ALE mesh update. The fluid–rigid body efficient implicit coupling algorithm implemented within the PISO predictor-corrector strategy is summarised in Algorithm 3.

Algorithm 3: Present FSI implicit coupling algorithm.

Given: fluid $(\mathbf{u}_f{}^n, p^n)$, interface $(\mathbf{X}_n{}^n, \mathbf{U}_n{}^n, \mathbf{F}_n{}^n)$, and solid $(\mathbf{X}_s{}^n, \mathbf{U}_s{}^n, \boldsymbol{\omega}_s{}^n)$;
(Fluid): Predict initial Eulerian velocity $\mathbf{u}_f{}^*$ using Eq. (4.26);
Initialise: $k = 1$, $\mathbf{X}_n{}^{(0)} = \mathbf{X}_n{}^n$, $\mathbf{X}_s{}^{(0)} = \mathbf{X}_s{}^n$, $\mathbf{U}_s{}^{(0)} = \mathbf{U}_s{}^n$, and $\boldsymbol{\omega}_s{}^{(0)} = \boldsymbol{\omega}_s{}^n$;
while $k \leq k_{max}$ **do**
    (Interface): Build Lagrangian support domain $\mathscr{D}_{\mathbf{X}_n}$, $\forall\, \mathbf{X}_n{}^{(k-1)}$;
    (Interface): Compute desired Lagrangian velocity $\mathbf{U}_n{}^{(k-1)}$ using Eq. (6.64);
    Initialise: $m = 1$, $\tilde{\mathbf{u}}_f{}^{(k=0,m=0)} = \mathbf{u}_f{}^*$, and $\mathbf{F}_n{}^{(k,m=0)} = \mathbf{F}_n{}^n$;
    while $m \leq m_{max}$ **do**
        (Interface): Interpolate Eulerian velocity to obtain $\mathbf{U}^{(k,m-1)}$ using Eq. (6.65);
        (Interface): Compute singular Lagrangian force $\mathbf{F}_n{}^{(k,m)}$ using Eq. (6.66);
        (Interface): Spread Lagrangian force to obtain $\mathbf{f}^{(k,m)}$ using Eq. (6.67);
        (Interface): Update the predicted Eulerian velocity $\tilde{\mathbf{u}}_f{}^{(k,m)}$ using Eq. (6.68);
        if $\|E_{no-slip}\|_2 < \epsilon_{IBM}$ **then**
            $\mathbf{f}^{(k)} = \mathbf{f}^{(k,m)}$, $\tilde{\mathbf{u}}_f{}^{(k)} = \tilde{\mathbf{u}}_f{}^{(k,m)}$, and $\mathbf{F}_n{}^{(k)} = \mathbf{F}_n{}^{(k,m)}$;
            break;
        else
            $m = m + 1$;
        end
    end
    (Solid): Compute Newton-Euler equations of motion to obtain $\mathbf{U}_s{}^{(k)}$ and $\boldsymbol{\omega}_s{}^{(k)}$ using Eqs. (6.71a) and (6.71b);
    (Solid): Relax $\mathbf{U}_s{}^{(k)}$ and $\boldsymbol{\omega}_s{}^{(k)}$ when necessary using Eqs. (6.72a) and (6.72b);
    (Solid): Update $\mathbf{X}_s{}^{(k)}$ using Eq. (6.52);
    (Interface): Update $\mathbf{X}_n{}^{(k)}$ using Eqs. (6.47)–(6.51);
    if $\|\mathbf{U}_s{}^{(k)} - \mathbf{U}_s{}^{(k-1)}\| / \|\mathbf{U}_s{}^{(k)}\| < \epsilon_{FSI}$ ***and*** $\|\boldsymbol{\omega}_s{}^{(k)} - \boldsymbol{\omega}_s{}^{(k-1)}\| / \|\boldsymbol{\omega}_s{}^{(k)}\| < \epsilon_{FSI}$ **then**
        $\mathbf{X}_s{}^{n+1} = \mathbf{X}_s{}^{(k)}$, $\mathbf{U}_s{}^{n+1} = \mathbf{U}_s{}^{(k)}$, $\boldsymbol{\omega}_s{}^{n+1} = \boldsymbol{\omega}_s{}^{(k)}$, $\mathbf{X}_n{}^{n+1} = \mathbf{X}_n{}^{(k)}$, $\mathbf{U}_n{}^{n+1} = \mathbf{U}_s{}^{n+1} + \boldsymbol{\omega}_s{}^{n+1} \times \mathbf{r}_n{}^{n+1}$, and $\mathbf{F}_n{}^{n+1} = \mathbf{F}_n{}^{(k)}$;
        break;
    else
        $k = k + 1$;
    end
end
Initialise: $m_{max} = 3$ and $\hat{\mathbf{u}}_f{}^{(0)} = \tilde{\mathbf{u}}_f{}^{(k)}$;
for $m = 1$ *to* m_{max} **do**
    (Fluid): Solve mPPE for $p^{(m)}$ using Eq. (4.34);
    (Fluid): Correct Eulerian velocity $\hat{\mathbf{u}}_f{}^{(m)}$ using Eq. (4.35);
    if $m = m_{max}$ **then**
        $\mathbf{u}_f{}^{n+1} = \hat{\mathbf{u}}_f{}^{(m_{max})}$ and $p^{n+1} = p^{(m_{max})}$;
    end
end

6.6 Summary

This chapter was devoted to the numerical modelling of fluid–rigid body interactions within the DF-IBM framework. The solid governing equations were introduced based on the RBD Newton-Euler equations of motion. The general mathematical formulation for the coupling of the Navier-Stokes and RBD within the DF-IBM framework is presented. This formulation encompasses the fluid governing equations, solid governing equations, fluid–solid interface condition, and IBM-related operators. The discretisation procedure for this coupled system was also examined in this chapter. An explicit treatment was chosen for handling the IME due to its simplicity of integration and its ability to circumvent the limitations on the solid–fluid density ratio that arise when the implicit IME scheme is used with the Newton-Euler equations.

A partitioned coupling approach is favoured for the fluid–rigid body coupling.This approach provides modularity and flexibility, allowing independent development and optimisation of each solver. In this chapter, two distinct FSI coupling algorithms are proposed: an explicit algorithm with weak coupling of the interface condition and an implicit algorithm with strong coupling of the interface condition.

For the implicit algorithm, the fixed-point strategy is employed due to its simplicity, with fixed relaxation adopted as the acceleration technique. This algorithm was found to give sufficient results and stabilised (converged) computations in cases where the IME introduced challenges, particularly when the densities of the fluid and rigid bodies are identical and for buoyant solids. The problems encountered with the rigid body approximation of the IME in the case of rotational dynamics were simultaneously solved as a consequence of the fixed relaxation technique. Similarly, in the explicit algorithm, instabilities arising from neutrally buoyant and buoyant solids were mitigated, and difficulties associated with the IME during rotational motion were effectively alleviated using the same relaxation technique.

The proposed implicit coupling algorithm is characterised by its efficiency compared to traditional algorithms used in the ALE formulation. This approach leverages the PISO predictor-corrector strategy and the implicit DF-IBM algorithm developed in Chapter 4, by ruling out the momentum predictor step and the PISO corrector loops, known as the most time-consuming step in the PISO solution algorithm, from the fluid–rigid body implicit coupling iterations. Finally, due to the substantial size difference between the Eulerian and Lagrangian domains, the Lagrangian mesh updates required by the implicit coupling algorithm are much less computationally demanding compared to the Eulerian mesh updates in the ALE formulation, further enhancing its overall efficiency.

Chapter 7

Numerical Results: Free Motion

Contents

This chapter presents a numerical validation of the proposed fluid–rigid body coupling algorithms. Various coupling scenarios are analysed, ranging from pure translational or rotational dynamics to complex coupled arbitrary motions. The effectiveness of the method is demonstrated using rigid bodies with varying density ratios, including light, dense, and neutrally buoyant bodies, tested on both simple and complex geometries. Additionally, the sensitivity of the coupling to simulation time-steps and mesh resolution is thoroughly examined. The chapter also explores the impact of the IME on the coupling process, highlighting the role and influence of the relaxation parameter on the solution and algorithm stability.

7.1 Sedimentation of a 2D Circular Disk in an Incompressible Viscous Fluid Inside a Confined Channel

The fluid–rigid body interaction of a sedimenting 2D circular disk in an incompressible, viscous, quiescent fluid is first examined to test the coupling during pure translational motion for non-buoyant rigid bodies. The circular disk is placed inside a confined channel with the following dimensions $\Omega_f = [0,\ 0.02] \times [0,\ 0.06]\ m^2 = 0.02 \times 0.06\ m^2$, corresponding to the fluid Eulerian domain. The disk, with a diameter $D = 2.5 \times 10^{-3}\ m$ and density $\rho_s = 1250\ kg/m^3$, is initially at rest, with its centre of mass located at $(X_s,\ Y_s) = (0.01\ m,\ 0.04\ m)$. The disk is allowed to fall freely due to gravity, and its velocity is determined by its interaction with the viscous fluid, which has a density $\rho_f = 1000\ kg/m^3$ and kinematic viscosity $\nu_f = 10^{-5}\ m^2/s$. The obtained density ratio for the sedimentation problem is $\rho_s/\rho_f = 1.25$. Once the disk reaches the bottom of the fluid channel, the wall collision model is activated automatically, using $\xi = 2.5h$ and $\varepsilon_w = h^2/2$. No-slip boundary conditions for the velocity $(u = v = 0)$ and Neumann boundary conditions for the pressure $(\partial p/\partial \mathbf{n} = 0)$ are applied on all sides of the computational domain, as illustrated in Fig. (7.1). The 3-point-width ϕ_2 is selected as the kernel function for this and all subsequent problems.

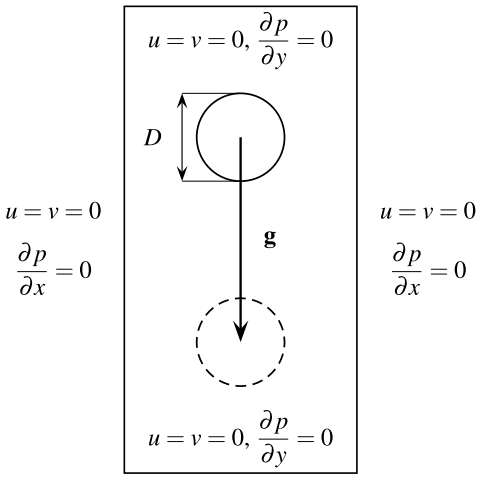


Figure 7.1: *Computational domain and boundary conditions for the sedimenting 2D circular disk inside a confined channel.*

A computational setup similar to the work of Wan and Turek [157] was chosen to allow numerical comparison with the computed results. Simulations are conducted on two different uniform mesh sizes: a coarse mesh with $h = 1/4800\ m$ and a fine mesh with $h = 1/9600\ m$, leading to $(N_x,\ N_y) = (100,\ 300)$ and $(N_x,\ N_y) = (200,\ 600)$ Eulerian grid cells, respectively.

Additionally, two different time-steps are tested: $\Delta t = 0.001\ s$ and $\Delta t = 0.0005\ s$. The simulation is advanced until reaching $t = 1.0\ s$ of simulated time. For the coarse mesh, the disk in the Lagrangian domain is formed by a total of $N_{Lag} = 38$ markers, uniformly distributed along Γ_s with $\Delta s \approx 2.0668 \times 10^{-4}\ m$, while for the fine mesh, 78 Lagrangian markers are distributed with $\Delta s \approx 1.0069 \times 10^{-4}\ m$.

The first investigation focuses on highlighting the differences between explicit and implicit fluid–rigid body coupling and examining the influence of explicit IME on the computed results. A tolerance of $\epsilon_{FSI} = 10^{-5}$ was set for the implicit FSI iterations for this and all subsequent problems. The disk is expected to reach a constant velocity, known as the terminal velocity V_t, where the net force becomes zero. The solid disk parameters of interest include the y-coordinate of the disk's centre of mass Y_s in (m), the vertical component of the disk's translational velocity V_s in (m/s), the solid Reynolds number based on the disk's translational velocity, $Re_{solid} = D\sqrt{U_s{}^2 + V_s{}^2}/\nu_f$, and the disk's translational kinetic energy, defined as $E_T = \rho_s V_s(U_s{}^2 + V_s{}^2)/2$ in (J). Fig. (7.2) presents these four quantities as functions of the simulated time t. For this analysis, the coarse mesh with $h = 1/4800\ m$ and a time-step of $\Delta t = 0.001\ s$ is used for all FSI simulations. Clearly, the sedimenting disk achieves the terminal velocity, $V_t \approx 0.054\ m/s$ corresponding to a $Re_{solid} \approx 17$, after approximately $t \approx 0.3\ s$, before hitting the bottom of the channel at $t \approx 0.8\ s$. Both explicit and implicit coupling approaches yield identical results for all plotted quantities, regardless of the IME presence. However, excluding the IME shifts all quantities slightly to the right as the sedimentation was delayed, aligning with findings in [97], which state that for $Re_{solid} > 1$, ignoring the IME leads to inaccuracies. Once the IME is included, the results are found to be in excellent agreement with the numerical data reported by Wan and Turek [157], demonstrating the importance of the IME in fluid–rigid body interactions. After the disk collides with the bottom wall of the channel, the results deviate from Wan and Turek's findings. However, the present results are more realistic since the disk is expected to rebound slightly upward with a positive velocity after settling at the bottom with a velocity equal to zero, unlike Wan and Turek's data, which showed a downward velocity after collision, which is a physically incorrect outcome.

The second investigation assesses the sensitivity of the coupling to different mesh sizes and time-steps. For this analysis, implicit coupling with explicit IME is used for all simulations. Fig. (7.3) illustrates the four quantities of interest for all h and Δt. It can be concluded that reducing the time-step has minimal influence on the results, whereas refining the mesh size slightly over-predicts the quantities once the sedimenting disk reaches its terminal velocity. Despite this over-prediction, the results remain in good agreement with Wan and

Turek [157], further validating the implementation of translational dynamics in the present work.

The instantaneous vorticity contours generated by the sedimenting disk are shown in Fig. (7.4) for five simulation times: $t = 0.2,\ 0.4,\ 0.6,\ 0.8,$ and $1.0\ s$. The vorticity field dissipates as observed at $t = 1.0\ s$, after the disk reaches the channel bottom approximately at $t \approx 0.8\ s$.

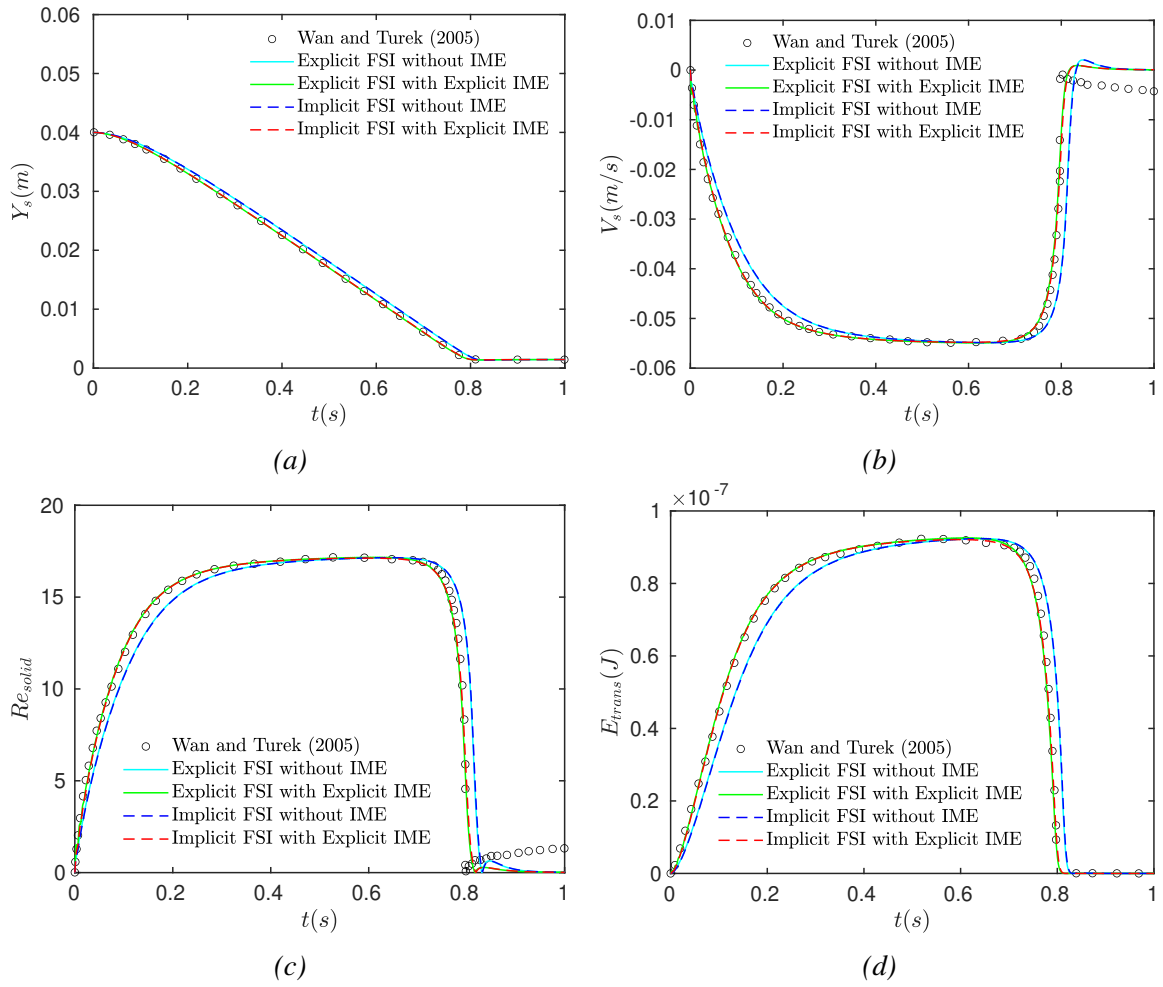


Figure 7.2: *Time history of the sedimenting 2D circular disk for different FSI coupling strategies (explicit/implicit) and IME (without/explicit) using $h = 1/4800\ m$ and $\Delta t = 0.001\ s$. (a) y-coordinate of the disk centre of mass position Y_s; (b) v-component of the disk velocity V_s; (c) solid Reynolds number of the disk Re_{solid}; and (d) translational kinetic energy of the disk E_{trans}.*

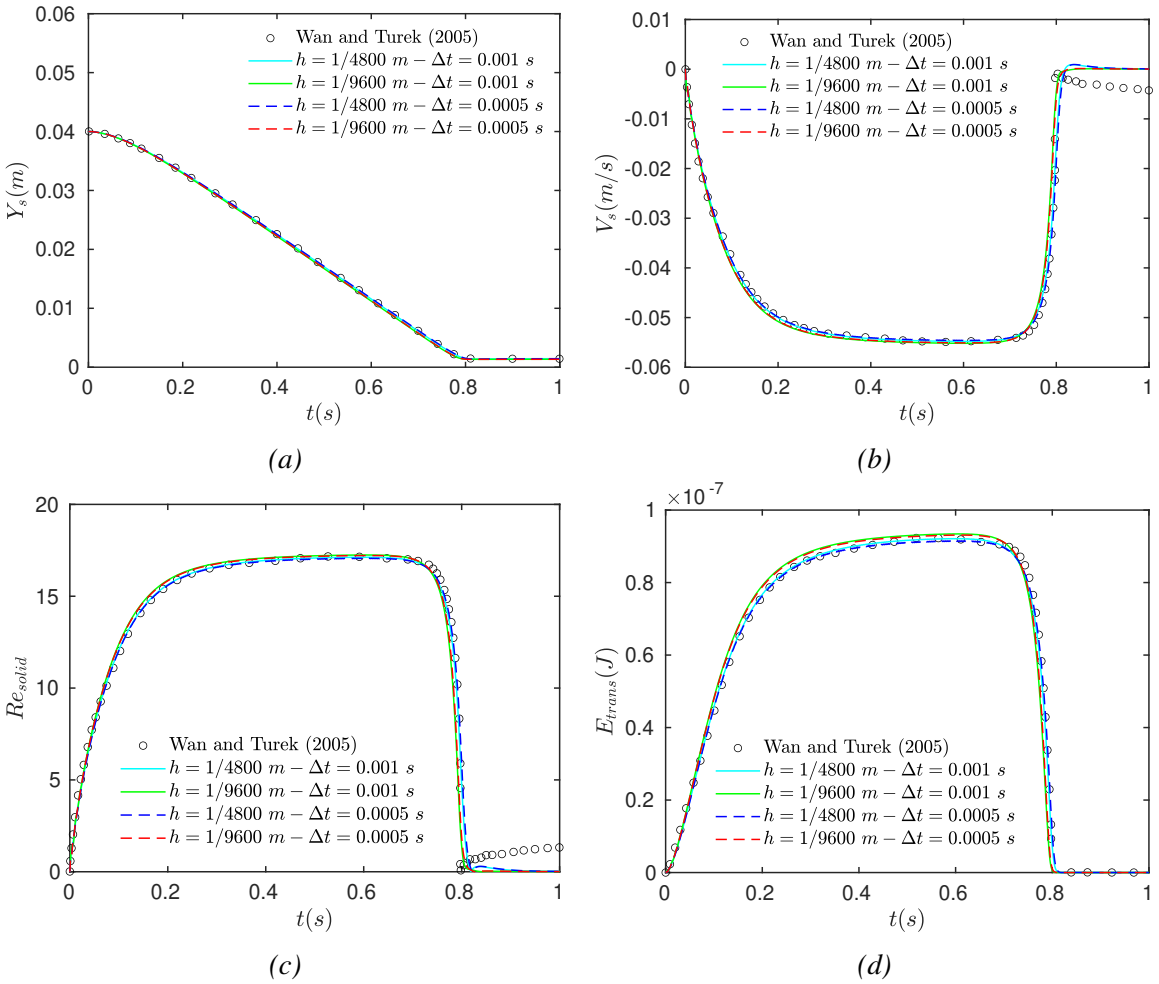
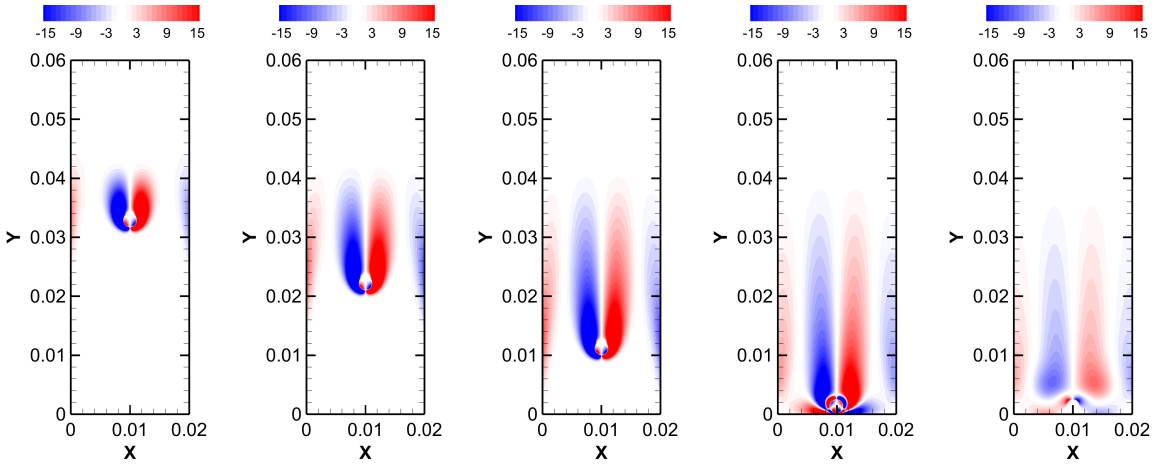


Figure 7.3: *Time history of the sedimenting 2D circular disk for different mesh sizes* ($h = 1/4800\ m,\ h = 1/9600\ m$) *and time-steps* ($\Delta t = 0.001\ s,\ \Delta t = 0.0005\ s$) *using the implicit FSI coupling and explicit IME. (a) y-coordinate of the disk centre of mass position* Y_s*; (b) v-component of the disk velocity* V_s*; (c) solid Reynolds number of the disk* Re_{solid}*; and (d) translational kinetic energy of the disk* E_{trans}*.*



(a) $t = 0.2\ s$ *(b)* $t = 0.4\ s$ *(c)* $t = 0.6\ s$ *(d)* $t = 0.8\ s$ *(e)* $t = 1.0\ s$

Figure 7.4: *Instantaneous vorticity contours at different time t instances of the sedimenting 2D circular disk.*

7.2 Free Rotation Due to a Linear Shear Flow of a Neutrally Buoyant 2D Circular Disk

The free rotation of a 2D circular disk subjected to a linear shear flow is examined to evaluate the coupling during pure rotational motion of neutrally buoyant rigid bodies. In this problem, the circular disk is placed inside a fluid domain defined as $\Omega_f = [-0.03,\ 0.03] \times [-0.02,\ 0.02]\ m^2 = 0.06 \times 0.04\ m^2$. The domain is discretised uniformly with a resolution of $h = 1/9600\ m$, resulting in $(N_x,\ N_y) = (600,\ 400)$ Eulerian grid cells in the x- and y-directions, respectively. The disk is positioned at the centre of the computational domain, with its centre of mass located at $(X_s,\ Y_s) = (0\ m,\ 0\ m)$ at $t = 0$. The fluid density is set to $\rho_f = 1000\ kg/m^3$, and the kinematic viscosity is $\nu_f = 10^{-6}\ m^2/s$. Three disk diameters are considered for this problem, $D = 0.004,\ 0.008,$ and $0.02\ m$. The solid density is chosen to be equal to the fluid density, ensuring a neutrally buoyant disk, $\rho_s = \rho_f = 1000\ kg/m^3$. The volume, mass moment of inertia, number of Lagrangian markers, and marker spacing for each disk diameter are summarised in Table 7.1. The mass moment of inertia is calculated using the following formula, $I_s = (1/32)\rho_s \pi D^4$.

Table 7.1: *Circular disk volume V_s, mass moment of inertia I_s, number of Lagrangian markers N_{Lag}, and Lagrangian spacing Δs, for all values of the disk diameter D.*

	$D = 0.004\ m$	$D = 0.008\ m$	$D = 0.02\ m$
$V_s\ (m^3)$	1.256637×10^{-5}	5.026548×10^{-5}	3.141592×10^{-4}
$I_s\ (kg \cdot m^2)$	2.513274×10^{-8}	4.021238×10^{-7}	1.570796×10^{-5}
N_{Lag}	124	250	628
$\Delta s\ (m)$	0.00010134	0.00010053	0.00010005

The boundary conditions of the fluid domain are presented in Fig. (7.5). The top and bottom walls move at constant velocities in opposite directions, with $u_{wall} = \pm 2 \times 10^{-4}\ m/s$, enabling the development of a linear shear flow between them. A Neumann boundary condition is imposed for the pressure on the moving walls, expressed as $(\partial p/\partial \mathbf{n} = 0)$. On the remaining boundaries, periodic boundary conditions are applied for both velocity and pressure. To ensure numerical stability, a CFL value below one is maintained throughout all simulations. The simulations are run until the non-dimensional time, $\frac{tu_{wall}}{D}$, reaches a value of 600.

Since the centre of mass of the disk is located at the centre of the computational domain, the circular disk will exhibit no translational motion. Instead, only rotational dynamics are expected about its centre of mass, with an angular velocity given by $\omega_s = \omega_z/2$, where ω_z

represents the vorticity in the third dimension (z-direction). For a linear shear flow without any solid immersed in the fluid domain, the flow is expected to reach a uniform state, characterised by a constant vorticity field value $\omega_z = -\gamma$, where γ denotes the shear rate. The shear rate for this flow is computed as $\gamma = 2u_{wall}/H = 0.01\ 1/s$, where H is the distance between the top and bottom walls ($H = 0.04\ m$). As the disk diameter approaches zero $D \to 0$, the angular velocity should asymptotically reach $\omega_s = \omega_z/2 = -0.005\ rad/s$, as if the solid was absent from the flow. However, as the disk diameter increases, the angular velocity is expected to decrease due to the disruption of the linear shear flow caused by the presence of the disk.

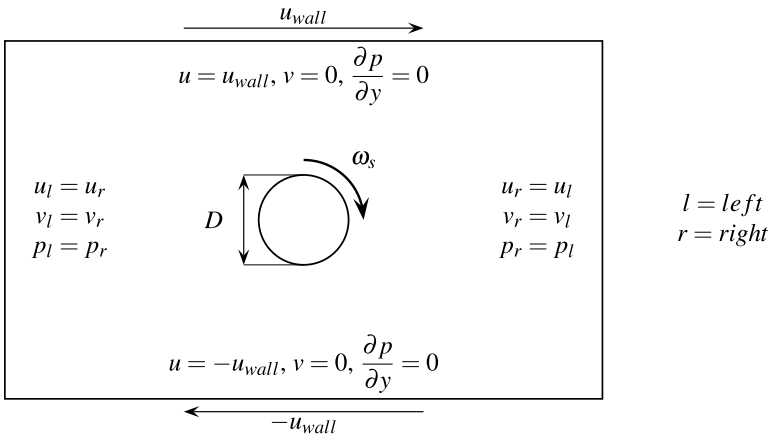


Figure 7.5: *Computational domain and boundary conditions for the free rotating 2D circular disk due to a linear shear flow.*

The same fluid and solid Reynolds numbers as those in Wan and Turek [157] and Cai [176] are used in the current simulations to enable comparison with data from the literature. These values are $Re_{fluid} = 2u_{wall}H/\nu_f = 16$ and $Re^{rot}_{solid} = \omega_s D^2/2\nu_f = 0.04,\ 0.16,$ and 1, corresponding to all disk diameters, respectively.

The first examination focuses on testing different coupling algorithms and various IME treatments. Both the exclusion of the IME and the explicit treatment, with and without relaxation, are tested. The results for the disk with a diameter $D = 0.004\ m$ are shown in Fig. (7.6). All simulations were able to achieve a steady-state solution as the shear flow developed. The results from explicit coupling converged to the same value, as did all solutions from implicit coupling. A very tiny difference in the steady-state angular velocity is observed between the different coupling strategies.

Instabilities arising from the rigid body approximation in the rotational dynamics are clearly observed when the explicit IME is used without relaxation. However, the simulation

was able to recover and reach convergence. These instabilities are resolved once a relaxation parameter is introduced, as indicated in Fig. (7.6). No significant difference is distinguished between the results obtained without the inclusion of the IME and those with explicit IME with relaxation, whether using explicit or implicit coupling. This is consistent with the findings in [97], which state that for $Re^{rot}_{solid} < 1$, both the exclusion and explicit treatment yield similar results, as the IME is relatively small in this case. The minor oscillations that appeared with the explicit IME during the steady-state portion of the solution are mitigated by using the relaxation parameter.

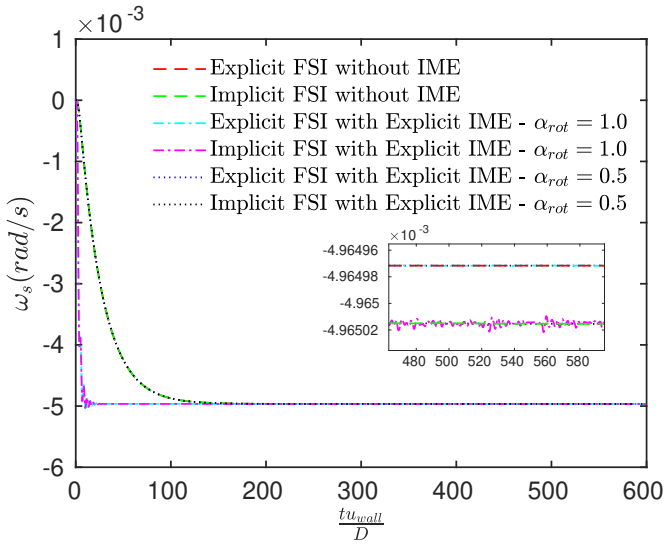


Figure 7.6: *Non-dimensional time history of the disk angular velocity ω_s for different FSI coupling strategies (explicit/implicit) and IME (without/explicit/explicit with relaxation) for a circular disk with diameter $D = 0.004$ m.*

The next step in this problem is to validate the implementation of rotational dynamics for different disk diameters. Fig. (7.7) reveals the evolution of the angular velocity ω_s of the circular disk subjected to a linear shear flow. As expected, the steady-state angular velocity decreases with increasing disk diameter. Additionally, the time required to reach the steady-state solution is also directly proportional to the disk diameter.

The steady-state angular velocity, also referred to as the terminal angular velocity ω_t, is compared with data from the literature in Table 7.2. Close agreement is observed with the literature for all disk diameters. Finally, the steady-state streamlines, velocity magnitude, pressure, and vorticity contours for the free rotation of a circular disk due to a linear shear flow are displayed in Fig. (7.8) for all disk diameters.

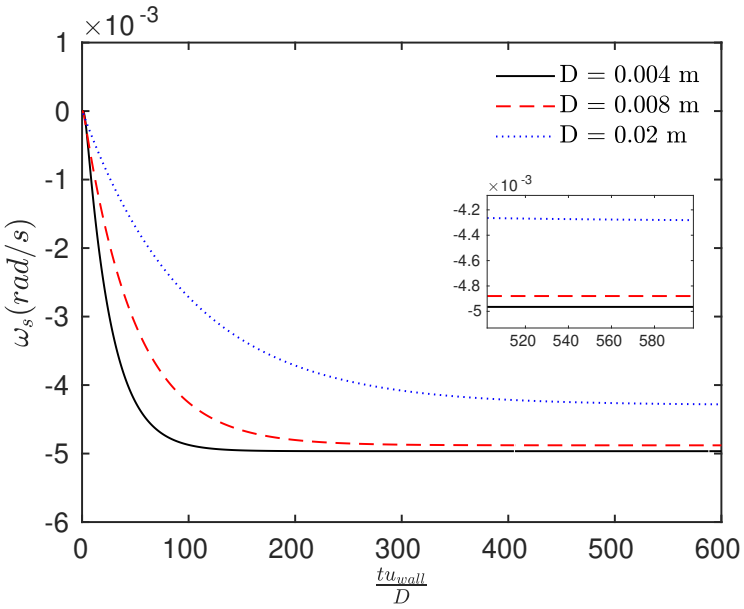


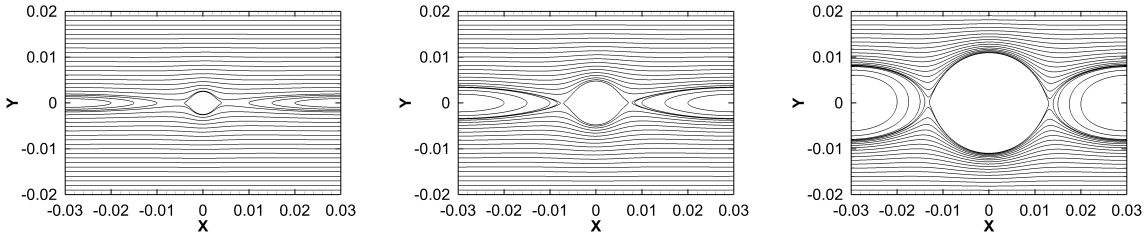
Figure 7.7: *Non-dimensional time history of the disk angular velocity ω_s for different circular disk diameters $D = 0.004,\ 0.008,$ and 0.02 m, using the implicit FSI coupling and the explicit IME with a relaxation parameter $\alpha_r = 0.5$.*

Table 7.2: *Comparison of the terminal angular velocity ω_t of the circular disk for different disk diameters.*

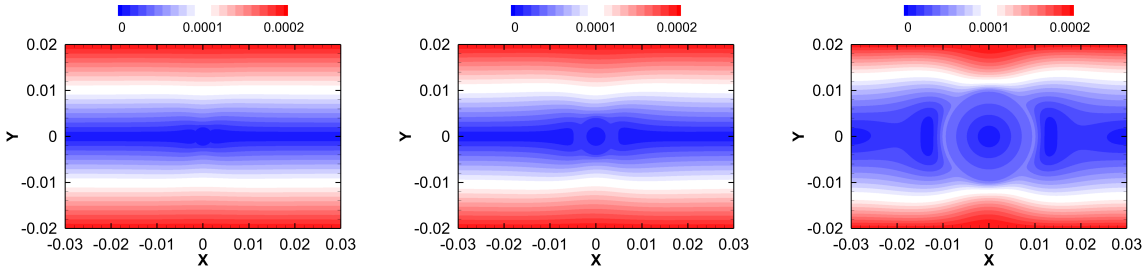
References	$D = 0.004\ m$	$D = 0.008\ m$	$D = 0.02\ m$
Wan and Turek [157]	-0.0049584	-0.0048697	-0.0043148
Cai [176]	-0.0049818	-0.0049167	-0.0043878
Present	-0.0049650	-0.0048795	-0.0042814

7.3 Freely Rotating 2D NACA0012 Airfoil Subjected to a Freestream and Fixed at its Centre of Mass

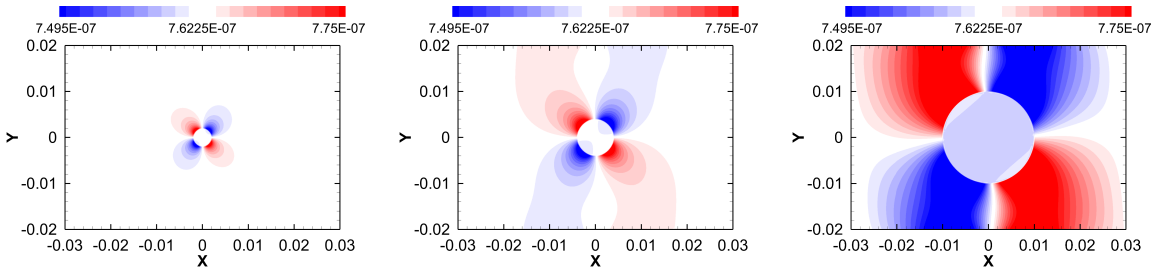
In this case study, a freely rotating 2D NACA0012 airfoil subjected to a freestream is considered. The symmetrical airfoil is positioned within a fluid Eulerian domain, $\Omega_f = [-0.04,\ 0.16] \times [-0.02,\ 0.02]\ m^2 = 0.2 \times 0.04\ m^2$. To reduce computational time, a refined region around the rotating airfoil is introduced. The refined region, $[-0.01,\ 0.05] \times [-0.015,\ 0.015]$, is discretised with two Eulerian grid resolutions, $h = 1/4800\ m$ and $h = 1/9600\ m$, to analyse the influence of the mesh size on the fluid–rigid body coupling. These resolutions result in $(N_x,\ N_y) = (300,\ 150)$ and $(N_x,\ N_y) = (600,\ 300)$ Eulerian grid cells within the refined region. Furthermore, to evaluate the influence of the time-step on the coupling, two time-steps, $\Delta t = 0.002\ s$ and $\Delta t = 0.001\ s$, are considered. The



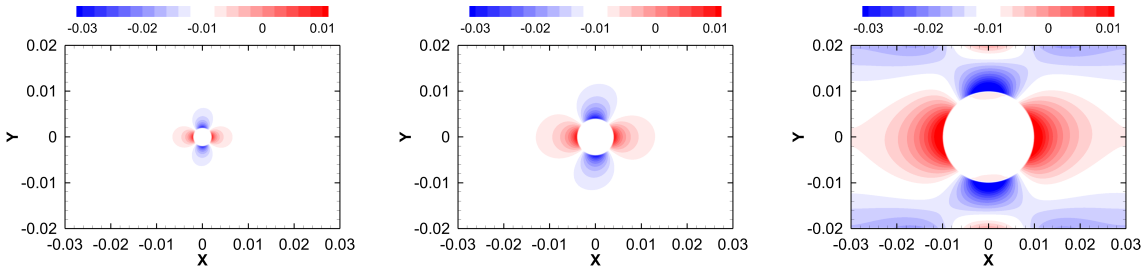
(a) Streamlines



(b) Velocity magnitude



(c) Pressure



(d) Vorticity

Figure 7.8: *Steady-state streamlines (a), velocity magnitude (b), pressure (c), and vorticity contours (d) for the free rotation of a neutrally buoyant circular disk due to a linear shear flow for different disk diameters. For $D = 0.004$ m, left column; $D = 0.008$ m, middle column; and $D = 0.02$ m, right column.*

simulation is advanced until $t = 22\ s$ using the implicit coupling and the explicit IME. It is noteworthy that the simulation diverged when the IME was omitted. The airfoil has a chord length of $c = 1.0089304 \times 10^{-2}\ m$ and is fixed at its centre of mass, located at $(X_s,\ Y_s) = (4.20515 \times 10^{-3}\ m,\ 0\ m)$. The airfoil density is set to $\rho_s = 1100\ kg/m^3$, with a volume of $V_s \approx 8.222127 \times 10^{-6}\ m^3$ and a mass moment of inertia of $I_s \approx 5.071302 \times 10^{-8}\ kg \cdot m^2$. The initial angle of attack, θ_s, and angular velocity, ω_s, are both set to zero. The boundary conditions for this problem are illustrated in Fig. (7.9). A Dirichlet boundary condition, $(u_\infty = 0.01\ m/s,\ v_\infty = 0)$, is applied at the inflow and outflow boundaries for the velocity, while a no-slip boundary condition, $(u = v = 0)$, is enforced on the lateral domain boundaries. For the pressure, a Neumann boundary condition, $(\partial p/\partial \mathbf{n} = 0)$, is applied to all domain boundaries.

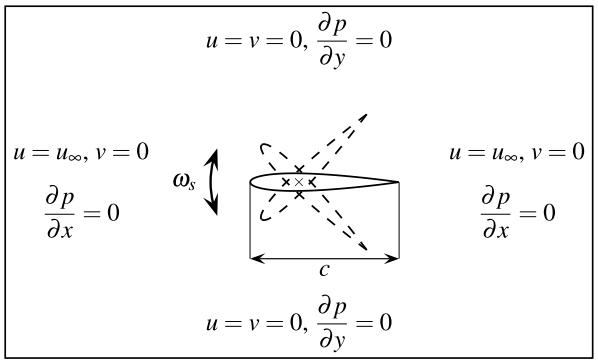
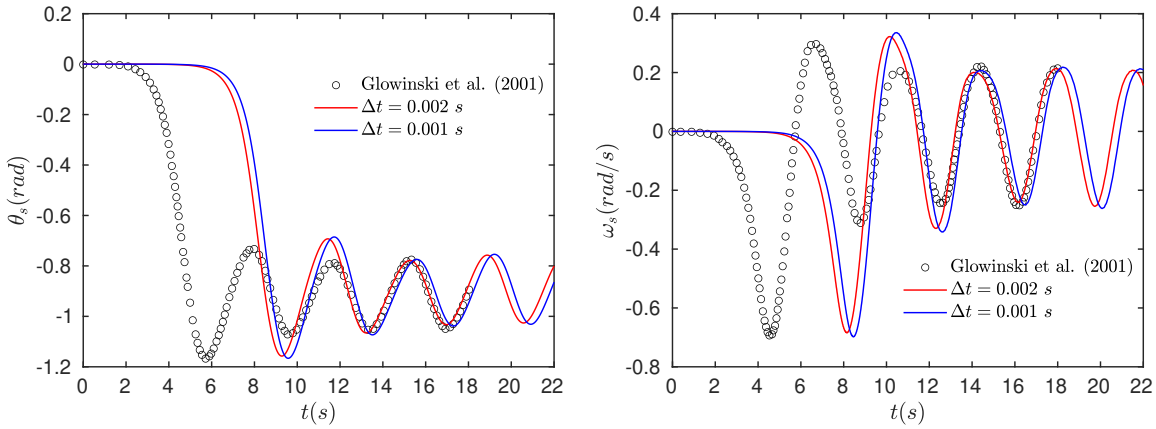


Figure 7.9: *Computational domain and boundary conditions for the freely rotating 2D NACA0012 airfoil subjected to a freestream and fixed centre of mass.*

The fluid density is set to $\rho_f = 1000\,kg/m^3$, with a kinematic viscosity of $\nu_f = 10^{-6}\,m^2/s$, resulting in a fluid Reynolds number of $Re_{fluid} = u_\infty c/\nu_f = 100.89$, based on the airfoil chord length, consistent with the values used by Glowinski et al. [177]. The density ratio for this problem is $\rho_s/\rho_f = 1.1$. For the coarse grid, the airfoil is represented by $N_{Lag} = 100$ Lagrangian markers, uniformly distributed along Γ_s, with a spacing of $\Delta s \approx 2.0557 \times 10^{-4}\ m$. On the fine grid, $N_{Lag} = 200$ markers are evenly distributed over Γ_s, with a spacing of $\Delta s \approx 1.0284 \times 10^{-4}\ m$. Since the airfoil is fixed at its centre of mass and subjected to a freestream, it is allowed to rotate solely around $\mathbf{X}_s$ due to its interaction with the fluid flow (hydrodynamic torque). To stabilise the simulation, a relaxation parameter for the Euler equation is set to $\alpha_r = 0.5$, as the simulation failed to converge when no relaxation was applied $(\alpha_r = 1.0)$. The objective of this simulation is to investigate the influence of the explicit treatment of the IME on the coupling solution.

The first study involves varying the time-step Δt to assess its impact on the computed

solution using the finer grid resolution, $h = 1/9600\ m$. Fig. (7.10) presents the time evolution of the angle of attack θ_s and the angular velocity ω_s of the rotating airfoil.



(a) *(b)*

Figure 7.10: *Time history of the angle of attack θ_s (a) and angular velocity ω_s (b) at two different time-steps for the freely rotating 2D NACA0012 airfoil for $h = 1/9600\ m$.*

A phase shift is observed in the computed results compared to those obtained by Glowinski et al. [177]. In the present results, the airfoil's rotation begins at approximately $t \approx 6.0\ s$, whereas for Glowinski et al., the rotation was initiated at $t \approx 3.0\ s$. This phase discrepancy aligns with findings from [97], which state that an explicit IME treatment introduces a phase shift in rotational dynamics quantities when the solid rotational Reynolds number Re^{rot}_{solid} exceeds 10. For the current FSI scenario, Re^{rot}_{solid} is calculated based on the maximum angular velocity, $\max\{|\omega_s|\} = 0.697\ rad/s$, yielding $Re^{rot}_{solid} = c^2 \max\{|\omega_s|\}/2\nu_f = 35.47$.

However, when the plots are compared with data from the literature, identical patterns emerge, with the maxima and minima of all oscillations captured correctly. This suggests that, despite the delayed interactions caused by the specific IME treatment, the essential flow physics are still effectively captured. From Fig. (7.10), it can be concluded that refining the time-step further delayed the rotational dynamics of the airfoil while preserving nearly identical oscillation peaks.

Next, to evaluate the effect of the time-step on the implicit coupling, Fig. (7.11) is presented. The number of FSI iterations decreases with a smaller time-step, as revealed in Fig. (7.11a). This reduction in implicit FSI iterations directly translates to a decrease in the CPU time required per iteration in this case, as demonstrated in Fig. (7.11b). The total CPU time for the simulations shows a relatively small difference, with a 10% increase observed for the smaller Δt compared to the larger Δt, despite requiring twice the number of time-steps

for the refined Δt. This is attributed to the lower computational cost per iteration due to the reduced number of FSI iterations when the time-step is refined.

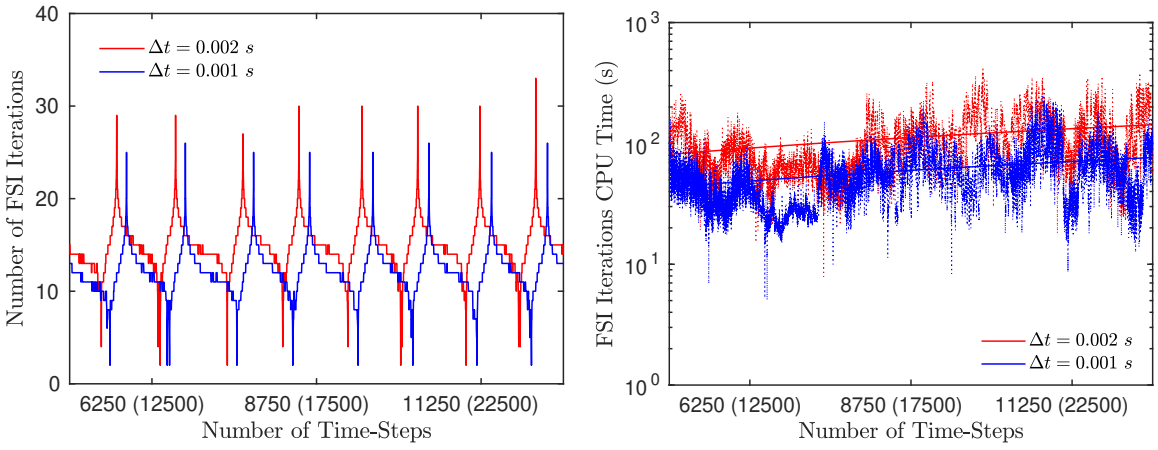
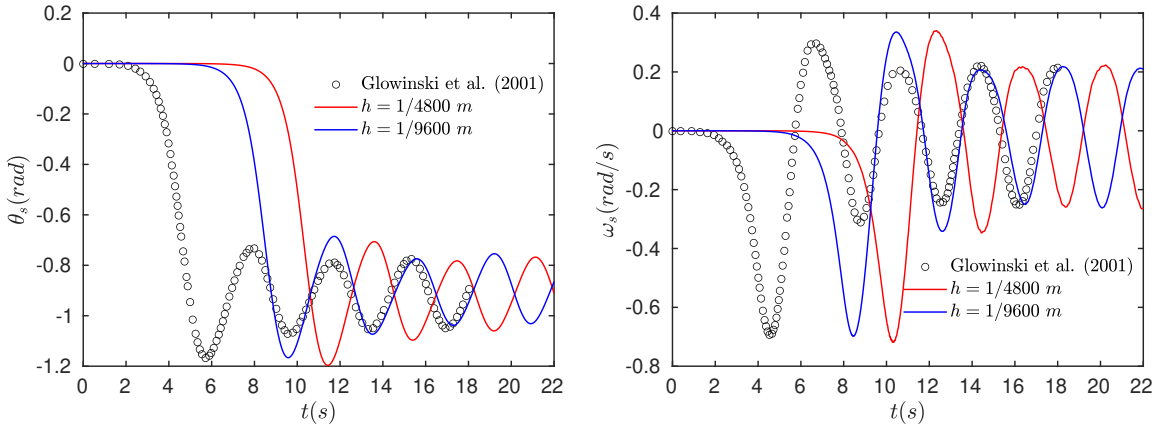


Figure 7.11: *Number of implicit FSI iterations (a) and CPU time per FSI iteration (b) at two different time-steps for the freely rotating 2D NACA0012 airfoil for $h = 1/9600$ m. The number of time-steps corresponding to $\Delta t = 0.001$ s are shown in brackets; the rest corresponds to $\Delta t = 0.002$ s. Dashed lines correspond to the actual CPU time; solid lines correspond to a first-degree (linear) polynomial fit of the CPU time data.*

The sensitivity of the FSI solution to mesh resolution is also examined for this problem. A time-step of $\Delta t = 0.001\ s$ was selected for this comparison. The results for θ_s and ω_s of the rotating NACA airfoil are presented in Fig. (7.12). A similar conclusion to the time-step analysis can be drawn for mesh resolution. Although the phase difference significantly increases when the mesh resolution is doubled (i.e., for the coarser grid), compared to the smaller phase difference observed when the time-step was halved. Nevertheless, both simulations recorded the same oscillation peaks, aligning with the data reported by Glowinski et al. [177].

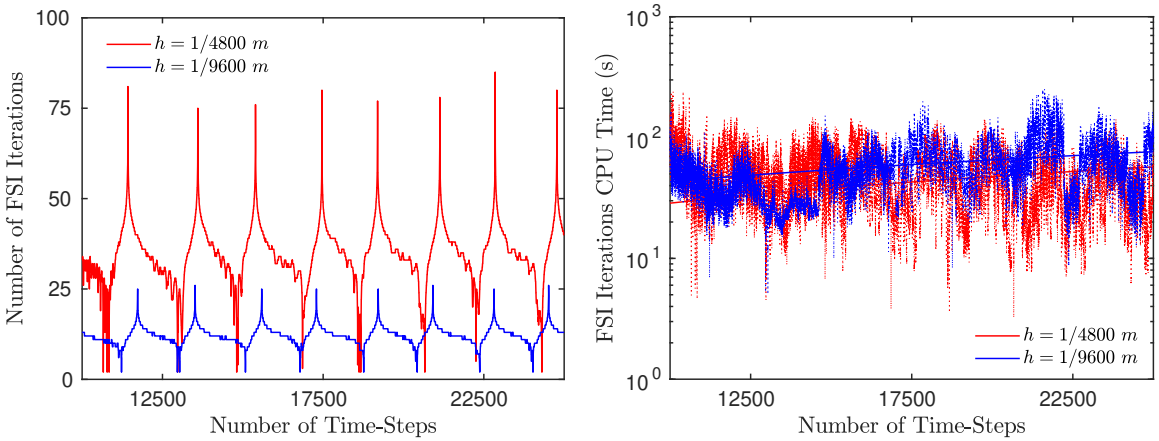
The effect of mesh resolution on the implicit coupling is depicted in Fig. (7.13). A notable increase in the number of implicit iterations is observed, with the number of iterations tripling when the mesh resolution is doubled, as seen in Fig. (7.13a). Conversely, the CPU time required per iteration is higher for the finer grid compared to the coarser one, as illustrated in Fig. (7.13b).

This suggests that, despite the reduced number of iterations, the size of the grid matrix is the dominant factor affecting the FSI computational cost, rather than the number of iterations. In fact, a smaller number of iterations does not necessarily result in lower computational costs. As a result, the total simulation CPU time for the fine mesh was found to increase by nearly 60% compared to the coarse mesh.



(a) *(b)*

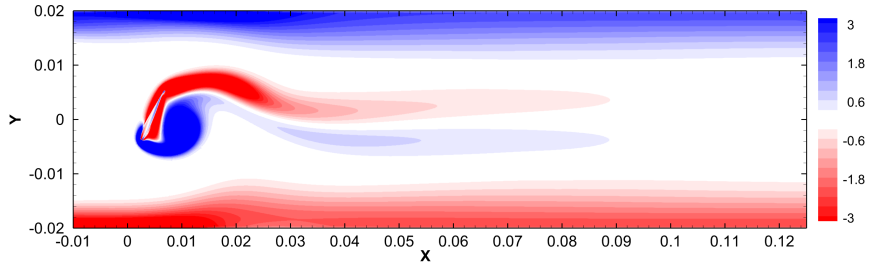
Figure 7.12: *Time history of the angle of attack θ_s (a) and angular velocity ω_s (b) at two different mesh sizes for the freely rotating 2D NACA0012 airfoil for $\Delta t = 0.001$ s.*



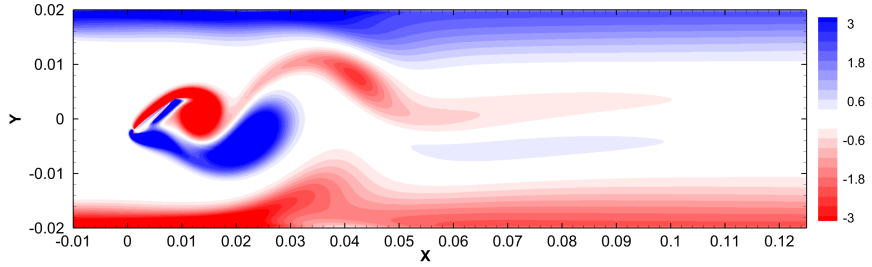
(a) *(b)*

Figure 7.13: *Number of implicit FSI iterations (a) and CPU time per FSI iteration (b) at two different mesh sizes for the freely rotating 2D NACA0012 airfoil for $\Delta t = 0.001$ s. Dashed lines correspond to the actual CPU time; solid lines correspond to a first-degree (linear) polynomial fit of the CPU time data.*

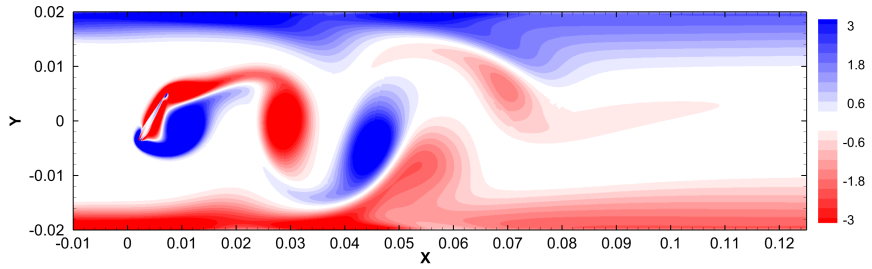
Lastly, the instantaneous vorticity contours are presented in Fig. (7.14) at various time instances corresponding to the maxima and minima of the angle of attack oscillations, specifically at $t = 10,\ 12,\ 14,\ 16,$ and $18\ s$.



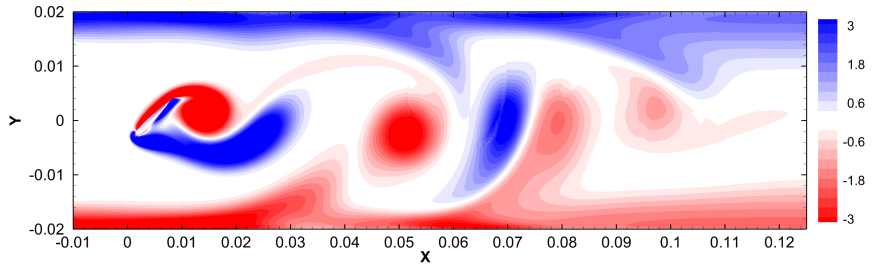
(a) $t = 10\ s$



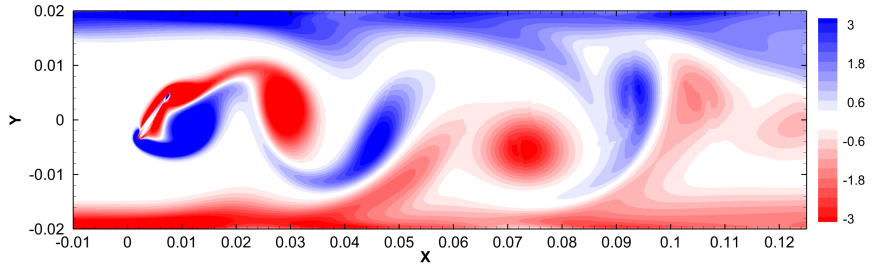
(b) $t = 12\ s$



(c) $t = 14\ s$



(d) $t = 16\ s$



(e) $t = 18\ s$

Figure 7.14: *Instantaneous vorticity contours at different time t instances for the freely rotating 2D NACA0012 airfoil.*

7.4 Sedimentation of a 2D Ellipse in an Incompressible Viscous Fluid Inside a Confined Channel

The arbitrary free motion of a rigid body, including coupled translational and rotational motions, is investigated. The interaction of a sedimenting ellipse, driven by gravity, within an incompressible viscous fluid confined to a bounded domain is considered. A comparison is conducted with the results obtained by Xia et al. [178], who employed the conventional ALE method and a FEM solver. To facilitate a meaningful comparison with the literature, four non-dimensional parameters are replicated:

- Density ratio: $\gamma = \dfrac{\rho_s}{\rho_f} = 1.1$, non-buoyant solid.
- Ellipse aspect ratio: $\alpha = \dfrac{a}{b} = 2$, where a and b are the major and minor axes of the ellipse, respectively (see Fig. (7.15)).
- Domain blockage ratio: $\beta = \dfrac{L}{a} = 4$, where L is the channel width (see Fig. (7.15)).
- Solid Reynolds number: $Re_{solid} = \dfrac{V_t a}{\nu_f} = 12.96$, where V_t is the vertical v-component of the ellipse's terminal velocity. It is determined as $V_t = 1.296 \times 10^{-2}\ m/s$ by averaging the vertical velocity values extracted from [178] over the time interval $t = 1.0\ s$ to $2.0\ s$, which corresponds to the constant terminal velocity regime.

The 2D ellipse is placed inside a confined channel with a narrow width L and a height of $17.5L$. In this simulation, the width is set to $L = 0.004m$. This results in a fluid domain with the following dimensions, $\Omega_f = [0,\ L] \times [0,\ 17.5L]\ m^2 = [0,\ 0.004] \times [0,\ 0.07]\ m^2 = 0.004 \times 0.07\ m^2$, with a uniform grid resolution of $h = 2 \times 10^{-5}\ m$. This setup yields $(N_x,\ N_y) = (200,\ 3500)$ grid points in the x- and y-directions. A time-step of $\Delta t = 1 \times 10^{-4}\ s$ is used, and the simulation is advanced until $t = 2.0\ s$. The boundary conditions are analogous to those of the 2D circular disk sedimentation problem, as illustrated in Fig. (7.15). The quiescent fluid has a density of $\rho_f = 1000\ kg/m^3$ and a kinematic viscosity of $\nu_f = 10^{-6}\ m^2/s$. For the ellipse, the major and minor axes are $a = 1 \times 10^{-3}\ m$ and $b = 5 \times 10^{-4}\ m$, respectively. Initially, at $t = 0$, the centre of mass of the rigid body is located at $(X_s,\ Y_s) = (0.5L\ m,\ 15L\ m) = (0.002\ m,\ 0.06\ m)$, with an orientation angle $\theta_s = \pi/4$. The solid density is set to $\rho_s = 1100\ kg/m^3$ to satisfy the density ratio imposed by Xia et al. [178]. Knowing that $a = 2b$, the volume of the ellipse is calculated as $V_s = (1/2)\pi b^2 = 3.926990 \times 10^{-7}\ m^3$, while the mass moment of inertia is computed as $I_s = (5/32)\rho\pi b^4 = 3.374757 \times 10^{-11}\ kg \cdot m^2$. The

ellipse boundary Γ_s is discretised using $N_{Lag} = 120$ uniformly distributed markers, with a spacing of $\Delta s \approx 2.018426 \times 10^{-5}\ m$. Based on the results obtained in the previous problems, the implicit coupling algorithm is employed in combination with the explicit IME scheme, using a relaxation parameter of $\alpha_r = 0.5$ for the Euler equation of motion.

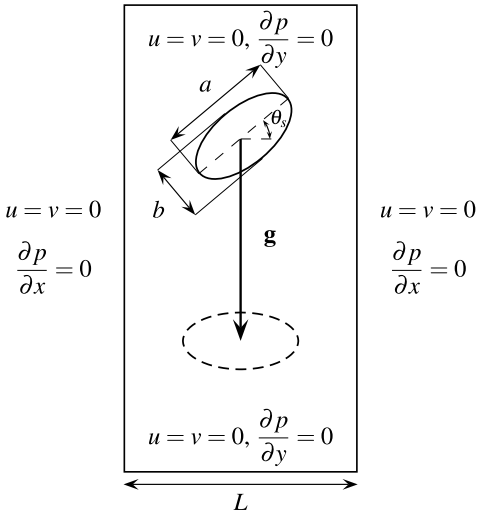


Figure 7.15: *Computational domain and boundary conditions for the sedimenting 2D ellipse inside a confined channel.*

Fig. (7.16) illustrates the non-dimensional trajectory of the ellipse's centre of mass $(X_s,\ Y_s)$ and its non-dimensional orientation θ_s during the sedimentation process. Due to the initial orientation of the ellipse, the symmetry of the flow is disrupted compared to the circular disk case, causing the ellipse to rotate and deviate from its initial horizontal position for $t > 0$. Over time, the ellipse progressively returns to its initial horizontal position, aligning its major axis perpendicular to the direction of the gravitational force. The simulation results show excellent agreement with the data extracted from the work of Xia et al. [178]. The maximum angular velocity recorded in the present simulation is, $\max\{|\omega_s|\} = 4.582\ rad/s$, leading to a solid rotational Reynolds number based on the ellipse's major axis of $Re^{rot}_{solid} = a^2 \max\{|\omega_s|\}/2\nu_f = 2.291$. This value satisfies the constraints imposed by the IME explicit scheme on Re^{rot}_{solid}.

To compare the current Re_{solid} with that obtained by Xia et al., the terminal velocity of the ellipse must be calculated. Fig. (7.17) shows the vertical v-component of the ellipse's velocity. The mean value of the vertical velocity between $t = 1.0\ s$ and $2.0\ s$ is $V_t = 1.306 \times 10^{-2}\ m/s$. Consequently, the computed solid Reynolds number for the present simulation is $Re_{solid} = 13.06$, which is in close agreement with the value reported by Xia et al. [178].

Furthermore, the variation of the vertical velocity aligns well with the data taken from the literature.

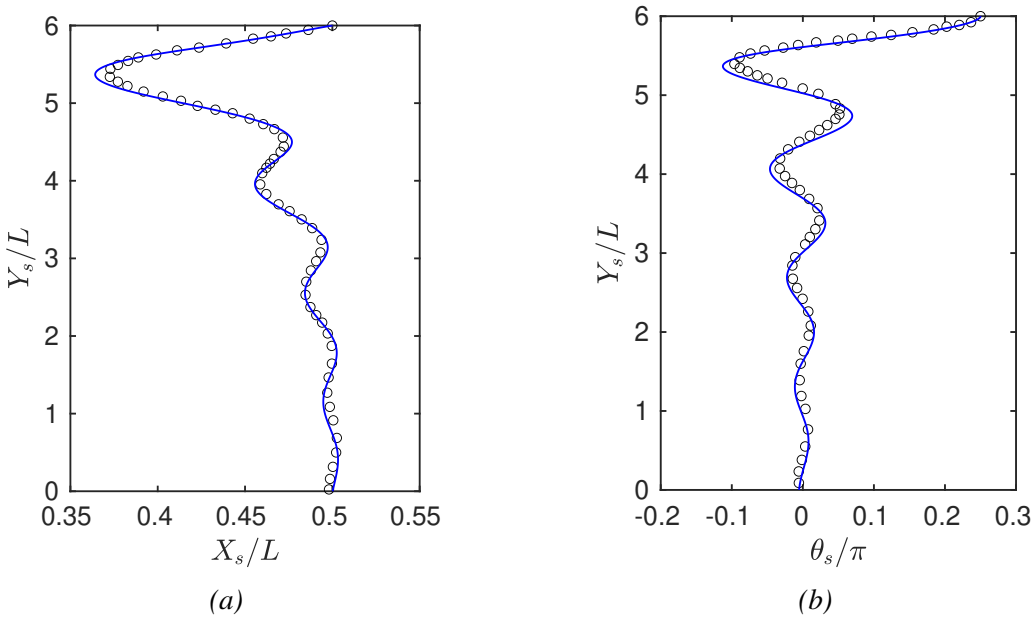


Figure 7.16: *Trajectory of the centre of mass (X_s, Y_s) (a) and orientation θ_s (b) of the 2D sedimenting ellipse in a confined channel. The current results are marked with solid lines (——) and the numerical results extracted from the work of Xia et al. [178] are marked with symbols (○).*

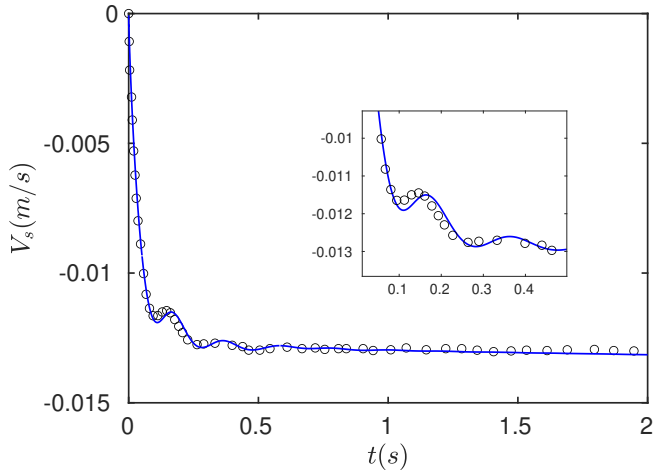
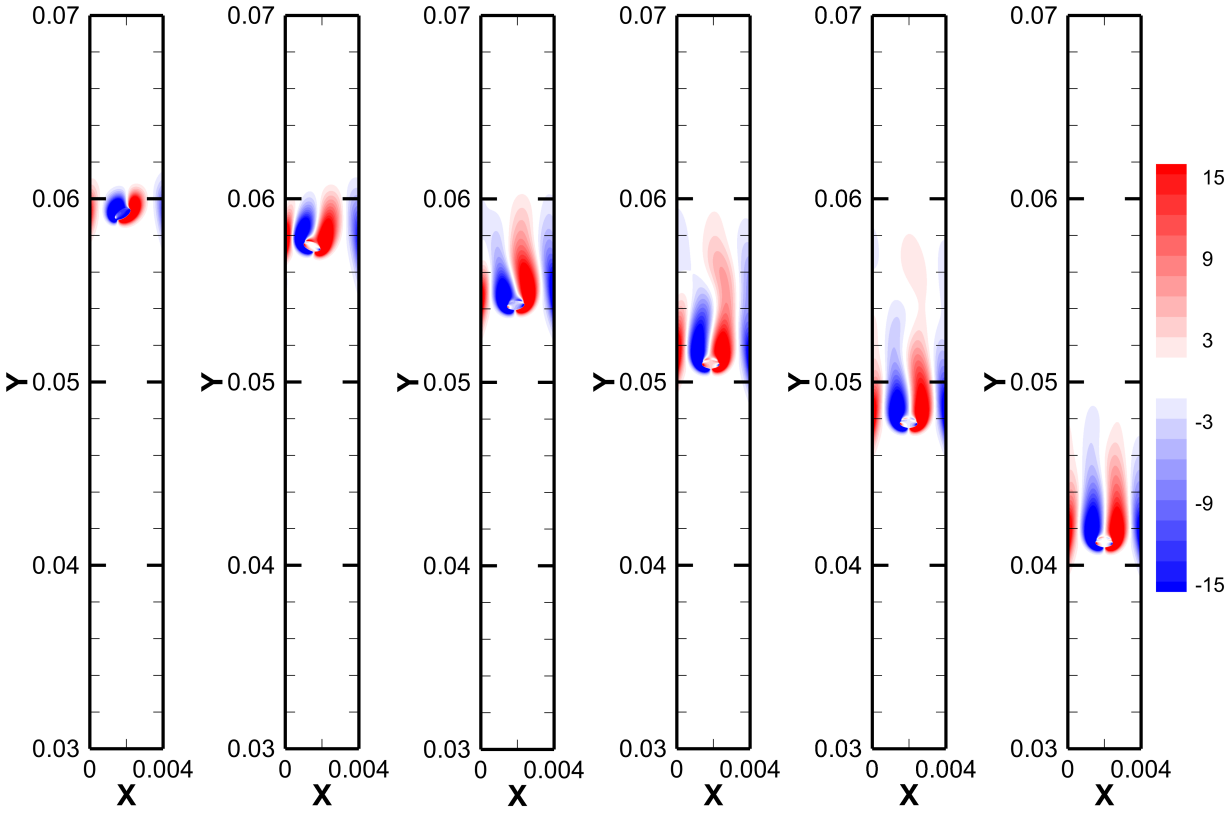


Figure 7.17: *Vertical v-component of the velocity V_s of the 2D sedimenting ellipse in a confined channel. The current results are marked with solid lines (——) and the numerical results extracted from the work of Xia et al. [178] are marked with symbols (○).*

Snapshots of the vorticity fields at various time instances, $t = 0.1$, 0.25, 0.5, 0.75, 1.0, and 1.5 s, are highlighted in Fig. (7.18). The results demonstrate the capability of the present work to accurately capture the complex flow physics arising from the interaction between the sedimenting ellipse and the initially quiescent flow field.



(a) $t = 0.1$ *s* *(b)* $t = 0.25$ *s* *(c)* $t = 0.5$ *s* *(d)* $t = 0.75$ *s* *(e)* $t = 1.0$ *s* *(f)* $t = 1.5$ *s*

Figure 7.18: *Instantaneous vorticity contours at different time t instances of the sedimenting 2D ellipse.*

7.5 Freely Falling (Dense) and Rising (Light) of a 2D Circular Disk in an Open Domain

In this problem, cases where the solid–fluid density ratio is close to unity, both from the upper and lower bounds, are investigated. Therefore, the free fall of a dense circular disk ($\rho_s/\rho_f > 1$) and the free rise of a lighter circular disk ($\rho_s/\rho_f < 1$) are simulated in a stagnant fluid. Despite the apparent simplicity, this problem is considered challenging due to the critical density ratios chosen, which are known to destabilise the numerical coupling algorithm. The computed results are compared with reference data [179], which utilised the ALE method and a FEM solver to address the problem numerically. For consistency, the following non-dimensional parameters are adhered to during the analysis:

- Falling density ratio: $\dfrac{\rho_s}{\rho_f} = 1.01$.
- Rising density ratio: $\dfrac{\rho_s}{\rho_f} = 0.99$.
- Galileo number: $Ga = \dfrac{\sqrt{\left|\dfrac{\rho_s}{\rho_f} - 1\right| gD^3}}{\nu_f} = 138$.
- Solid Reynolds number: $Re_{solid} = \dfrac{V_t D}{\nu_f} = 156$, where V_t is the vertical v-component of the terminal velocity.

To model the effect of an open (infinite) domain, an extensively extended finite domain is employed. The domain dimensions are defined as a function of the circular disk diameter D, having $\Omega_f = [-5D,\ 5D] \times [-70D,\ 70D]\ m^2$, which corresponds to $[-0.025,\ 0.025] \times [-0.35,\ 0.35]\ m^2 = 0.05 \times 0.7\ m^2$ for a disk diameter $D = 0.005\ m$. A uniform mesh size of $h = 2 \times 10^{-4}\ m$ is used, resulting in a grid resolution of $(N_x,\ N_y) = (250,\ 3500)$. The simulations are advanced with a time-step $\Delta t = 2 \times 10^{-4}\ s$ until the non-dimensional time, $\dfrac{tV_t}{D}$, reaches 100. The incompressible viscous fluid has the following properties, $\rho_f = 996\ kg/m^3$ and $\nu_f = 8.02 \times 10^{-7}\ m^2/s$. The circular disk is discretised into $N_{Lag} = 78$ Lagrangian markers with a spacing of $\Delta s \approx 2.01384 \times 10^{-4}\ m$. Figs. (7.19a) and (7.19b) illustrate the computational domain and boundary conditions for the freely falling and rising cases, respectively. The initial position, volume, and density for both scenarios are detailed in Table 7.3. The implicit coupling algorithm is used with the explicit IME scheme. Due to the critical

density ratios simulated in this problem, a relaxation parameter of $\alpha_r = 0.5$ was used for the Newton equation of motion, as the moving disk undergoes only translational motion.

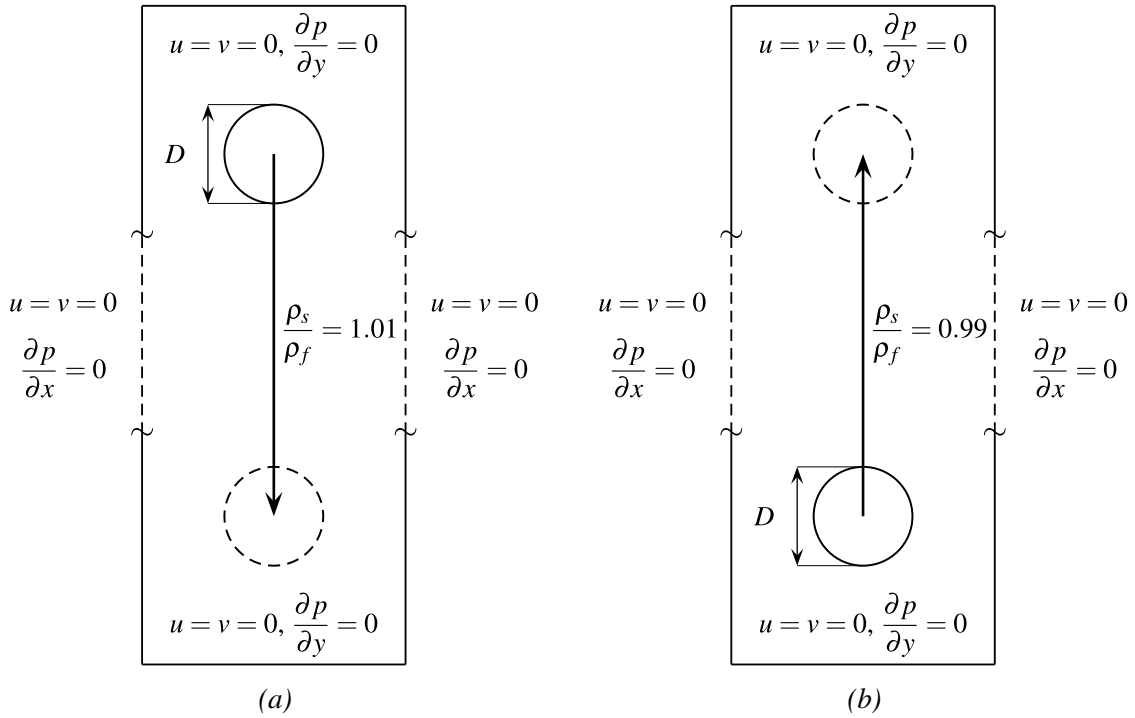


Figure 7.19: *Computational domain and boundary conditions for the freely falling (a) and freely rising (b) 2D circular disk in an open domain.*

Table 7.3: *Circular disk initial position $(X_s,\ Y_s)$, volume V_s, and density ρ_s, for the freely falling and rising cases.*

	Falling	Rising
$(X_s,\ Y_s)\ (m)$	$(0,\ 65D) = (0,\ 0.325)$	$(0,\ -65D) = (0,\ -0.325)$
$V_s\ (m^3)$	1.963495×10^{-5}	1.963495×10^{-5}
$\rho_s\ (kg/m^3)$	1005.96	986.04

The non-dimensional vertical v-component and horizontal u-component of the velocity for the freely rising circular disk are shown in Fig. (7.20), respectively. The results are in good agreement with the data shown in [179]. The horizontal contribution of the disk velocity is less than 10% than its vertical contribution. For $\frac{tV_t}{D} > 50$, the disk attains a periodic motion, with an average vertical terminal velocity $V_t = 2.494 \times 10^{-2}\ m/s$ for the falling case, compared to $V_t = 2.501 \times 10^{-2}\ m/s$ reported in [179].

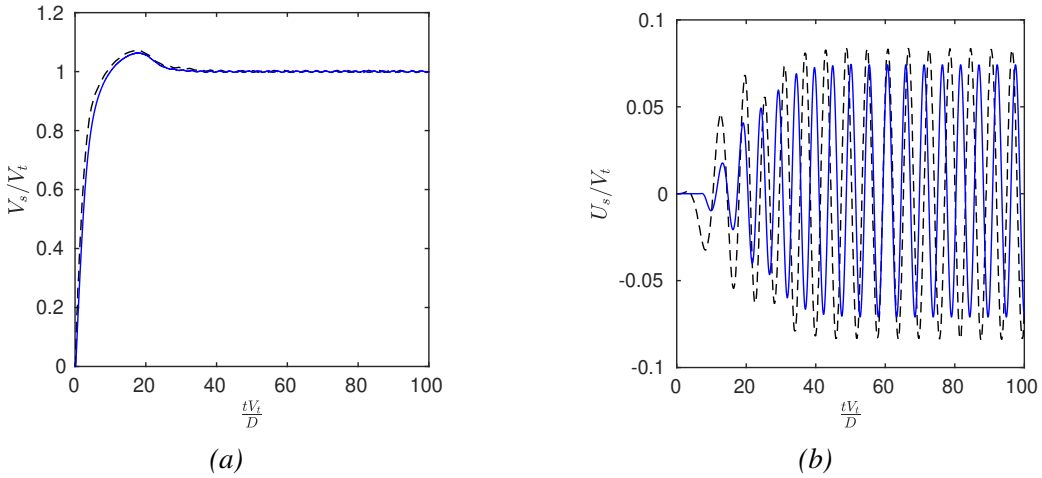


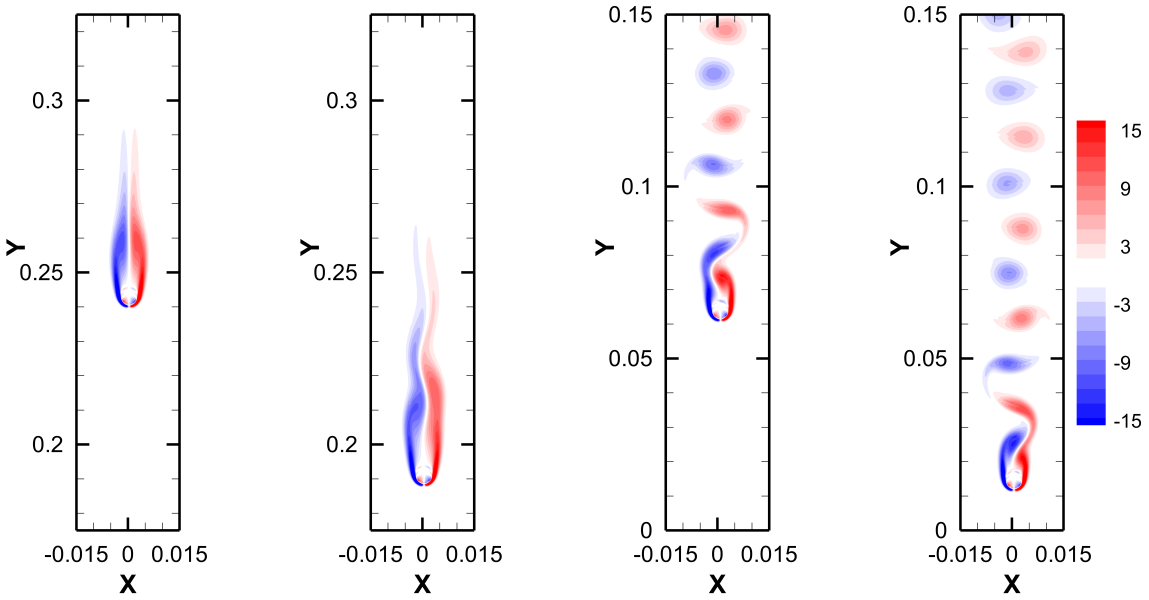
Figure 7.20: *Non-dimensional vertical v-component (a) and horizontal u-component (b) of the circular disk velocity for the freely rising circular disk for $\rho_s/\rho_f = 0.99$ at $Re = 156$ and $Ga = 138$. The current results are marked with solid lines (——) and the numerical data extracted from [179] are marked with dashed lines (- - -).*

A comparison of the force coefficients and Strouhal number is provided in Table 7.4. The mean drag, lift coefficient amplitude, and Strouhal number are computed using the terminal velocity V_t. The results show satisfactory agreement with values reported in the literature.

Table 7.4: *Comparison of the mean drag, lift coefficient amplitude, and Strouhal number for the freely falling and rising circular disk.*

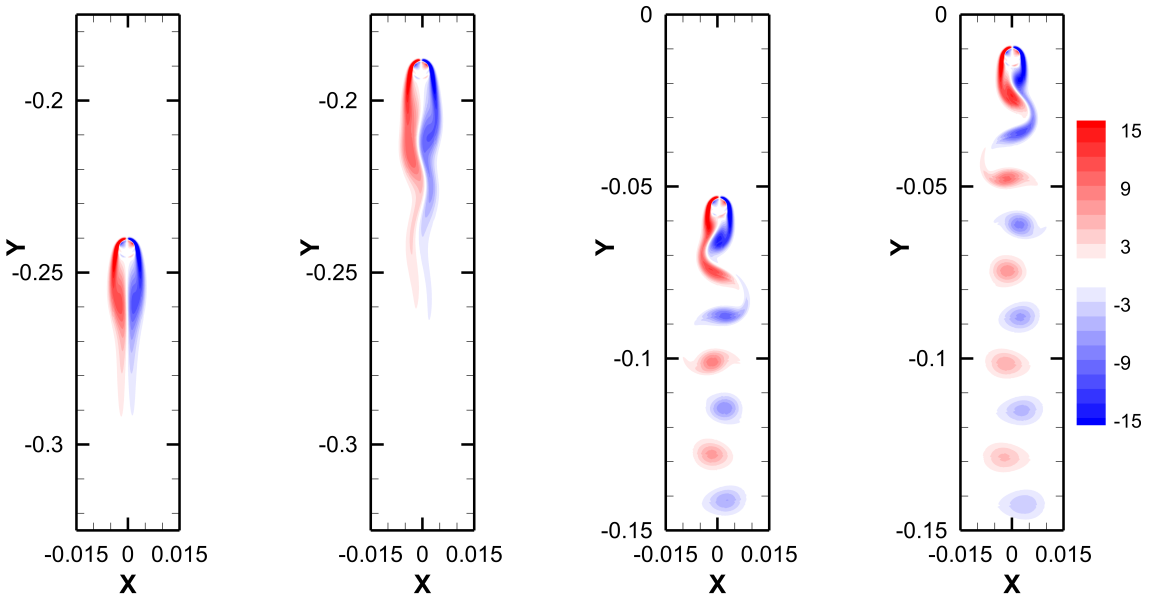
References	$\rho_s/\rho_f = 1.01$ – Falling			$\rho_s/\rho_f = 0.99$ – Rising		
	$\overline{C_D}$	$\max\|C_L\|$	Str	$\overline{C_D}$	$\max\|C_L\|$	Str
Namkoong et al. [179]	1.23	0.15	0.16840	–	–	0.16870
Lācis et al. [95]	1.29	0.14	0.17185	1.29	0.14	0.17188
Cai [176]	1.35	0.10	0.18900	1.35	0.10	0.18900
Present	1.24	0.10	0.16252	1.24	0.10	0.16473

Instantaneous vorticity contours for the freely falling and rising circular disk are shown in Figs. (7.21) and (7.22), respectively. The flow fields exhibit strong similarities due to the similar Reynolds numbers and closely matched density ratios in both cases. Initially, the flow remains symmetric, with a symmetric vortex pair forming downstream of the disk around $\frac{tV_t}{D} = 20$. However, symmetry begins to break at approximately $\frac{tV_t}{D} = 30$, and by $\frac{tV_t}{D} = 55$ and 65, the wake becomes fully asymmetric, leading to unsteady periodic vortex shedding phenomena. The latter is what caused the horizontal oscillations of the disk.



(a) $tV_t/D = 20$ (b) $tV_t/D = 30$ (c) $tV_t/D = 55$ (d) $tV_t/D = 65$

Figure 7.21: *Instantaneous vorticity contours at different time instances for the freely falling circular disk at* $\rho_s/\rho_f = 1.01$.



(a) $tV_t/D = 20$ (b) $tV_t/D = 30$ (c) $tV_t/D = 55$ (d) $tV_t/D = 65$

Figure 7.22: *Instantaneous vorticity contours at different time instances for the freely rising circular disk at* $\rho_s/\rho_f = 0.99$.

7.6 Freely Rotating 2D Vertical Axis Turbine (VAT) Subjected to a Freestream Under Laminar Regime

Finally, the following problem aims to simulate the VAT behavior under laminar flow using an alternative approach compared to the one presented in Section 5.3.4. Unlike the conventional method, which enforces a constant angular velocity on the VAT rotor, this method allows the VAT to rotate freely under the influence of a freestream velocity; referred to in the literature as a freestream-induced rotation. This rotation results from the aerodynamic torque acting on the VAT rotor over time and is modeled using the Euler equations, which is able to simulate the interaction between the freestream and the VAT blades.

The primary goal of this approach is to predict the self-starting characteristics of an H-Darrieus type VAT. Understanding these dynamics is crucial for addressing self-starting challenges associated with these types of VAT, particularly in low freestream velocity environments.

The three-bladed H-Darrieus type VAT employed in Section 5.3.4 is used. Each blade follows the NACA0015 airfoil profile, as displayed in Fig. (5.27). The computational setup remains identical to that in Section 5.3.4, unless stated otherwise. Likewise, the boundary conditions are the same as those in Fig. (5.28). The fluid density is fixed at $\rho_f = 1000\ kg/m^3$ with a kinematic viscosity of $\nu_f = 0.01\ m^2/s$. The VAT density is set to $\rho_s = 1100\ kg/m^3$, resulting in a density ratio of $\rho_s/\rho_f = 1.1$. The airfoil chord length is fixed at $c = 1\ m$, and the total volume occupied by the airfoils is computed as $V_s = 3V_{airfoil} \approx 0.303562\ m^3$. The mass moment of inertia is approximately $I_s \approx 1.3636 \times 10^3\ kg \cdot m^2$. The implicit coupling algorithm with the explicit IME scheme is selected, and a relaxation parameter of $\alpha_r = 0.5$ is applied to the Euler equation of motion, as the VAT undergoes only rotational motion.

First, the freestream velocity used in Section 5.3.4, $u_\infty = 0.5\ m/s$, is simulated. This corresponds to a Reynolds number based on the chord length i.e. $Re_c = 50$. The angular rotation θ_s and angular velocity ω_s of the VAT are shown in Fig. (7.23). The results clearly indicate that the VAT is unable to complete a full rotor revolution, even after $t = 300\ s$ of simulation time.

Experimental data from [180] have shown that vertical axis wind turbines with a NACA0015 blade profile require a minimum freestream wind velocity of 6.0 m/s to initiate rotation and generate torque. This finding was later validated numerically in [181], where the RANS turbulence models, such as the $k-\varepsilon$ Realizable and $k-\omega$ SST, accurately predicted the vertical axis wind turbine performance in comparison to the experimental results of [180].

Consequently, these critical velocity thresholds should be reassessed for fluids with densities higher than that of air, such as the one considered in this study. Nonetheless, this provides a general estimate of the starting velocity range for such VATs.

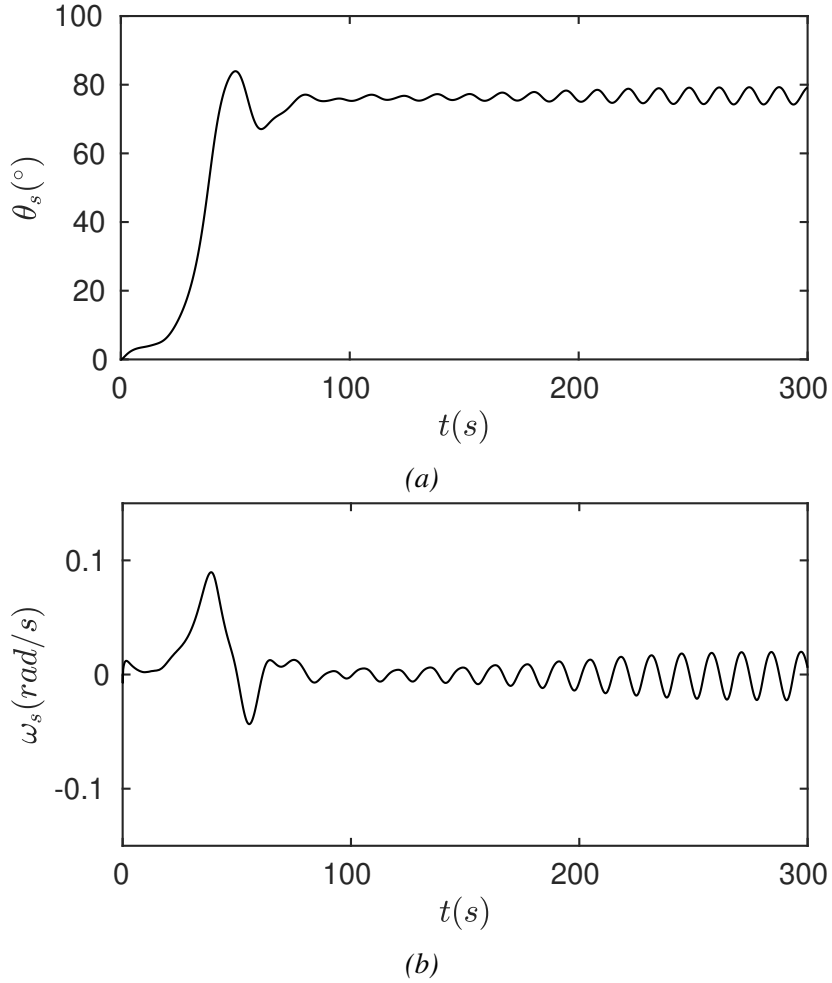
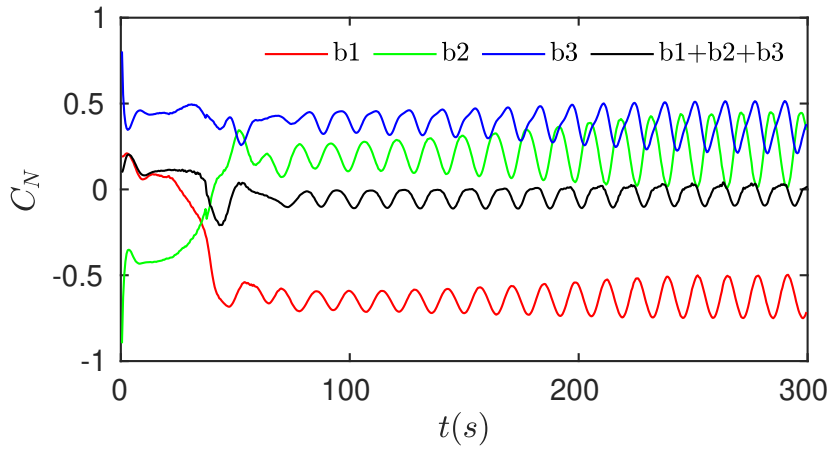
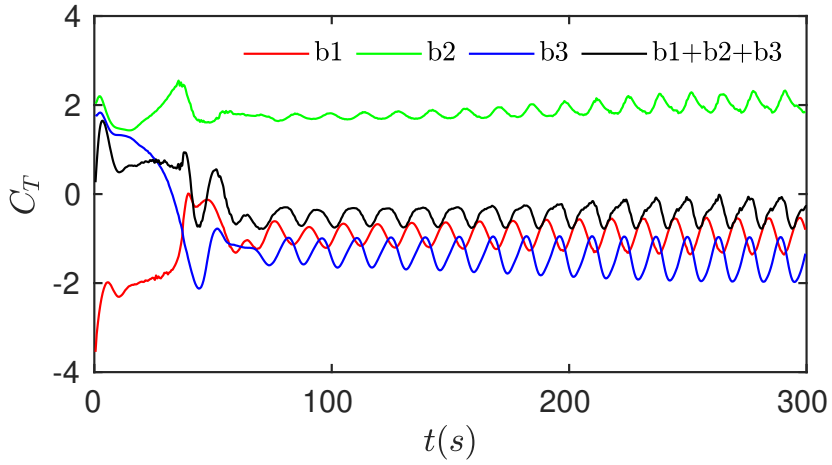


Figure 7.23: *Time history of the angular rotation θ_s (a) and angular velocity ω_S (b) for the freely rotating 2D VAT subjected to a freestream at $Re_c = 50$.*

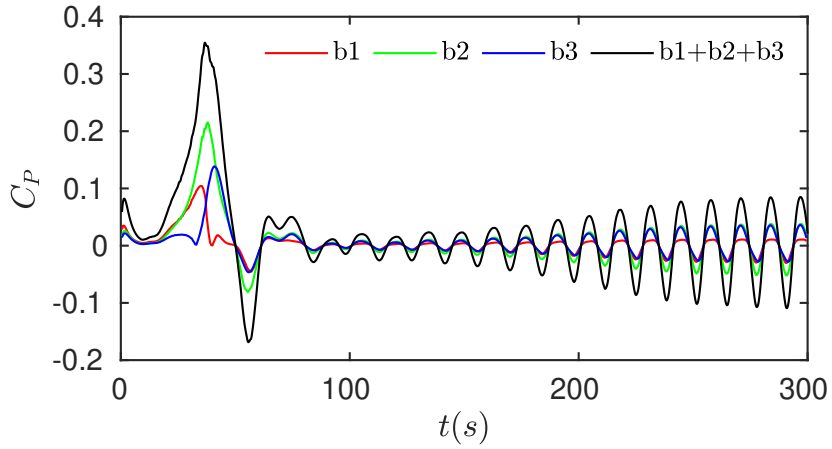
It is worth noting that, in the case of freestream-induced rotation, the tangential and normal force coefficients given in Eq. (5.11) are not identical for all blades, unlike the prescribed constant angular velocity case presented in Section 5.3.4. This is evident in Fig. (7.24), where each blade, denoted as $b1$, $b2$, and $b3$, exhibits distinct force and power curves due to variations in angular rotation and angular velocity over time, as shown in Fig. (7.23). The total force contribution from all blades combined, represented as $(b1+b2+b3)$, is also plotted in Fig. (7.24). The power coefficient fluctuates between positive and negative values due to variations in the VAT's angular velocity, with an overall mean of approximately $C_P \approx 0$.



(a)



(b)



(c)

Figure 7.24: *Normal force C_N (a), tangential force C_T (b), and power C_P (c) coefficients for the freely rotating 2D VAT subjected to a freestream at $Re_c = 50$.*

A second investigation is conducted, where the freestream velocity is increased to $u_\infty = 5.0\,m/s$, which remains below the minimum freestream requirement of $6.0\,m/s$ reported in [180]. This choice is due to the laminar assumption in the present method, as no turbulence model has been implemented. The corresponding Reynolds number based on the chord length is $Re_c = 500$. At this Re_c, the simulation remains challenging, as the flow around the VAT at higher freestream velocities is predominantly influenced by turbulent effects. However, the results are considered satisfactory in demonstrating the small improvement in VAT performance as the freestream velocity increases. A comparison of the angular rotation θ_s and angular velocity ω_s for $u_\infty = 0.5$ and $5.0\,m/s$ is demonstrated in Fig. (7.25).

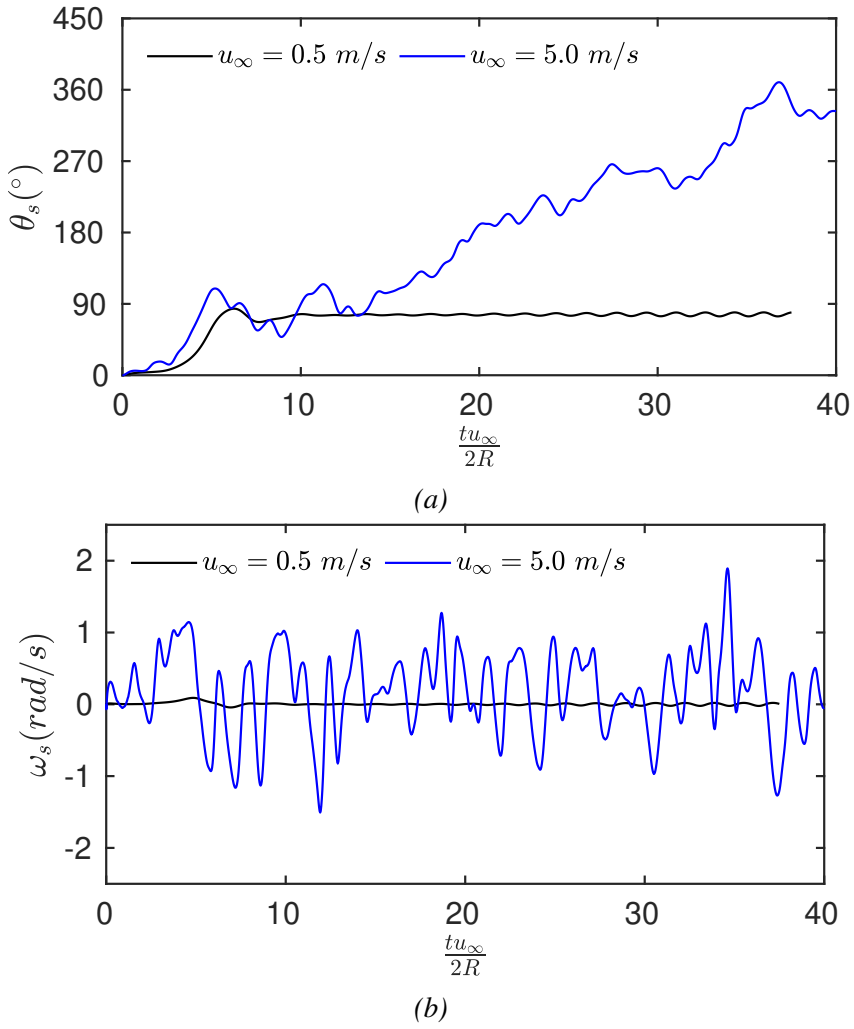


Figure 7.25: *Time history of the angular rotation θ_s (a) and angular velocity ω_S (b) for the freely rotating 2D VAT subjected to two different freestream velocities $u_\infty = 0.5$ and $5.0\,m/s$.*

The time in Fig. (7.25) is non-dimensionalised as $\frac{tu_\infty}{2R}$ to facilitate comparison between

both cases. At $u_\infty = 5.0\ m/s$, the VAT completes a full rotor revolution at approximately $\frac{tu_\infty}{2R} = 36$, whereas at the lower freestream velocity, the VAT exhibits a plateau behavior without completing a full revolution.

The tangential force and power coefficients for both cases are presented in Fig. (7.26). A higher tangential force is observed for $u_\infty = 5.0\ m/s$, leading to an increase in the torque generated by the VAT. Consequently, this results in a higher C_P coefficient, as shown in Fig. (7.26).

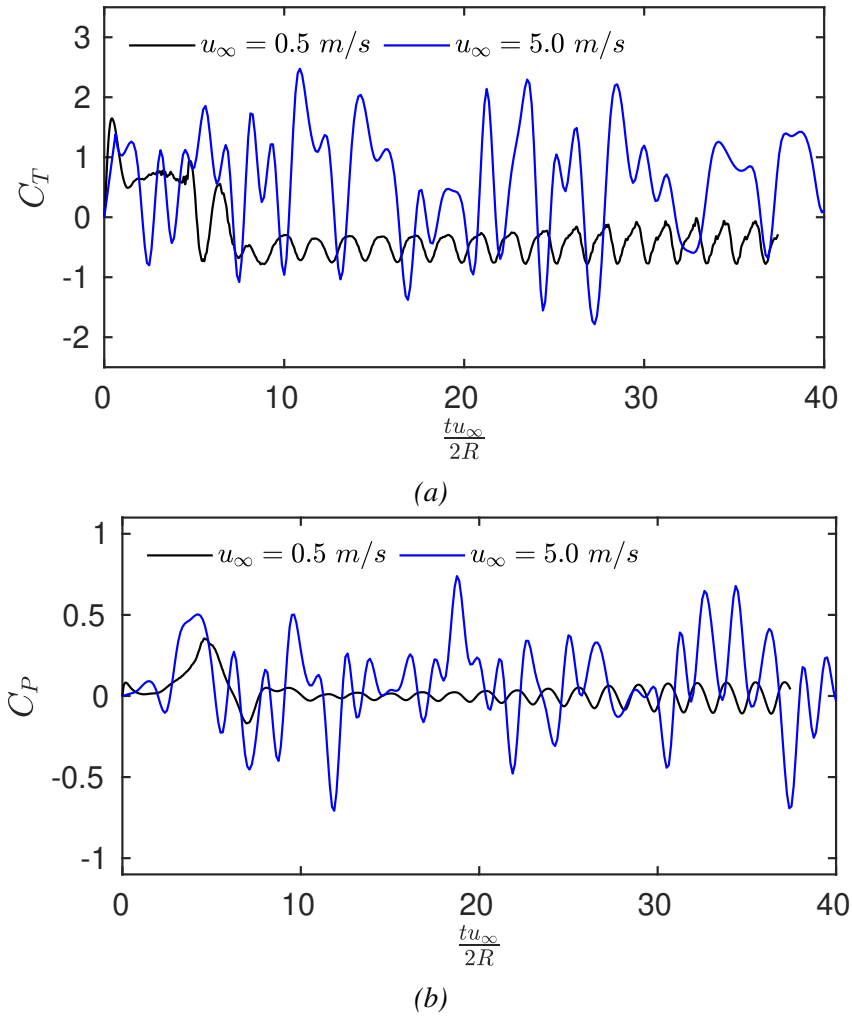
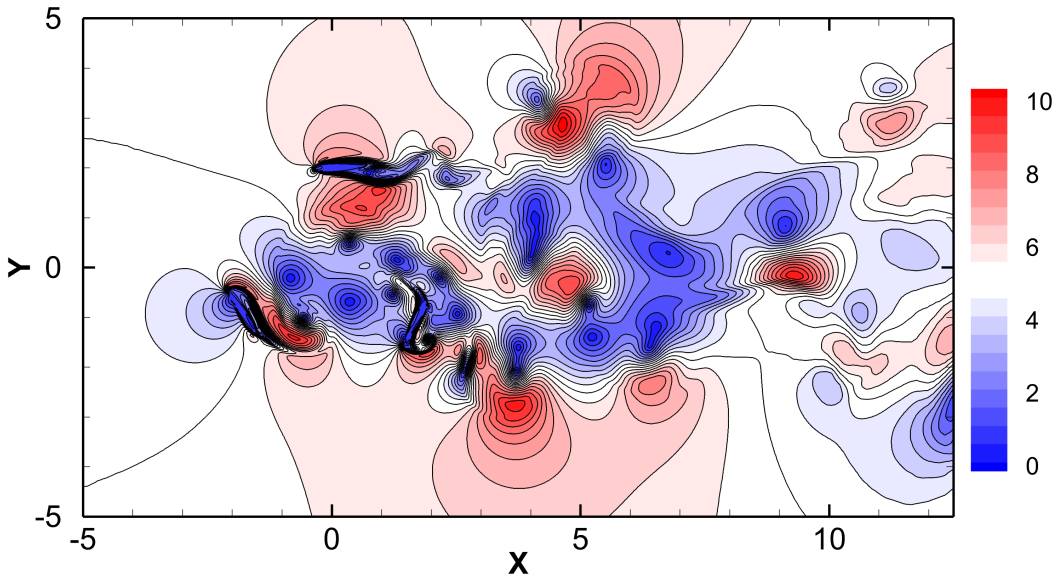
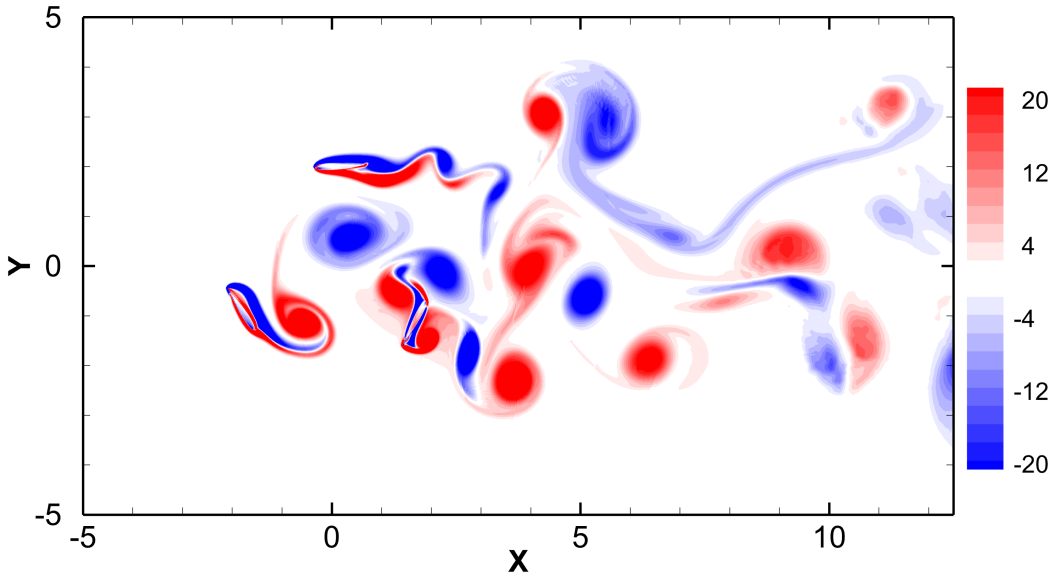


Figure 7.26: *The sum of* $(b1+b2+b3)$ *for the tangential force* C_T *(b) and power* C_P *(c) coefficients for the freely rotating 2D VAT subjected to two different freestream velocities* $u_\infty = 0.5$ *and* $5.0\ m/s$.

The velocity and vorticity fields at the end of the first revolution are shown in Fig. (7.27). A high-velocity region of approximately $10\ m/s$ develops on the downstroke side of the VAT, while a stagnant region forms in its wake.



(a)



(b)

Figure 7.27: *Velocity (a) and vorticity (b) contours for the freely rotating 2D VAT at the end of the first revolution, i.e. for* $\theta = 360°$*, for* $Re_c = 500$*.*

A comprehensive study [182] explored various methods to enhance the aerodynamic performance and efficiency of VATs. Among these, optimising the airfoil geometry; such as varying thickness, camber, and profile shape, can improve the self-starting capability and overall performance of the VAT. Another approach involves adjusting the airfoil pitch angle,

either by selecting an optimised fixed-pitch angle or implementing a variable-pitch mechanism that dynamically adjusts based on the incident angle of attack. A recent study [183] introduced a novel J-shaped blade profile, which reportedly increases the VAT's starting torque by nearly 135% compared to the NACA0015 profile. This CFD-optimised design is expected to reduce the minimum freestream velocity required for self-starting rotation, thereby shortening the self-starting time in low-speed environments. However, further research is needed to fully assess the effectiveness of the current approach in handling complex fluid–structure interactions, especially given the current constraint to the laminar regime, which imposes significant limitations on such intricate flows.

7.7 Summary

The proposed explicit and implicit fluid–rigid body coupling algorithms were numerically validated in this chapter through a series of test cases. Validation began with simulations of pure translational and pure rotational motions, followed by coupled and complex arbitrary free motion.The current development effectively handled a wide range of density ratios, from non-buoyant to buoyant rigid bodies, including the critical case of neutrally buoyant and lighter bodies, for which stable and converged computations were achieved.

The study encompassed various geometries, including circular disks, ellipses, and NACA airfoils, demonstrating the versatility of the approach. The significance of incorporating the IME was highlighted, particularly for cases where the solid Reynolds number exceeded a value of one. Instabilities arising from the rigid body approximation of the IME during rotational dynamics were identified and addressed. A fixed relaxation technique proved to be an effective solution to mitigate these instabilities, enabling convergence of the implicit coupling algorithm for specific challenging cases.

It was observed that the explicit treatment of the IME introduced a time shift in the rotational dynamics quantities, particularly noticeable when the solid rotational Reynolds number exceeded 10. Despite this phase shift, the essential flow physics and rigid body dynamics were preserved. The sensitivity analyses of the implicit coupling to time-step size and mesh resolution revealed that, by reducing the time-step or refining the mesh, the convergence rate of the implicit iterations is improved. Notably, a coarser grid amplified the phase shift in the rotational quantities when the solid rotational Reynolds number is above 10, in comparison to finer grids. Additionally, the impact of the time-steps on the phase shift is less significant than the effect of mesh resolution.

Overall, the proposed coupling algorithms demonstrated excellent agreement with refer-

ence data from the literature across all test cases, validating their capability and robustness for a broad range of challenging fluid–rigid body interaction problems.

Chapter 8

Conclusions and Future Work

Contents

In this concluding chapter, a summary of the research conducted will be presented, with an emphasis on the key findings. Additionally, the chapter will outline several recommendations and suggest potential extensions to the current developments.

8.1 Conclusions

In the context of the immersed boundary method (IBM), the fluid governing equations formulated in an Eulerian framework are coupled with the solid immersed boundary represented using a Lagrangian formulation. This coupling is accomplished by introducing a boundary force as a source term in the Navier-Stokes equations. This force effectively mimics the presence of the solid immersed boundary within the fluid domain. For incompressible flow problems, the coupled fluid–solid system imposes dual constraints that must be enforced separately within their respective formulations. The first constraint ensures a divergence-free velocity field in the Eulerian domain, while the second enforces the kinematic no-slip velocity boundary condition on the Lagrangian domain. The current implementation employs the cell-centred finite volume approach, achieving second-order accuracy in both space and time. To address the collocated grid arrangement used in OpenFOAM and prevent checkerboard pressure oscillations, the Rhie-Chow momentum interpolation technique was applied (see Chapter 3).

This work proposes a unified direct-forcing immersed boundary method (DF-IBM) algorithm that combines implicit IBM techniques with a calibrated Lagrangian weight, which enhances the reciprocity of the IBM-related velocity interpolation and force spreading operators. This enables the accurate and efficient enforcement of the second fluid–solid constraint. The pressure, velocity, and boundary force are decoupled from the fluid governing equations using the operator-splitting strategy of the PISO solution algorithm, implemented within the OpenFOAM environment. The improved DF-IBM algorithm leverages the predictor-corrector framework of the PISO algorithm to sequentially enforce the dual constraint problem of the fluid–solid system at each time-step during the solution process. This formulation leads to a modified pressure Poisson equation (PPE), characterised by the incorporation of the boundary force that already satisfies the second fluid–solid constraint on the immersed boundary. Consequently, the remaining role of the PPE is to ensure that the Eulerian velocity field maintains a divergence-free condition. This approach preserves a symmetric pressure coefficient matrix and avoids increasing its condition number, making it comparable to the standard PPE. This modification enables the use of fast iterative solvers for the pressure equation, such as the Conjugate Gradient (CG) solver in conjunction with the Geometric Agglomerated Algebraic Multi-Grid (GAMG) as preconditioner, without requiring additional adjustments in OpenFOAM. To accelerate the convergence of the implicit iterations and satisfy the kinematic constraint, a Lagrangian force initialization scheme is introduced. No additional restrictions are imposed on the time-step, other than the stability criterion of the

CFL number, which must satisfy CFL < 1 (see Chapter 4).

The overall order of accuracy was proven to be second-order in time (see Fig. (5.2a)) and second-order in space (see Fig. (5.2b)), though the latter is only guaranteed for smooth fields. Furthermore, the algorithm's overall performance and efficiency were shown to be superior to previous DF-IBM algorithms (see Table 5.4). It was also demonstrated that using an even number of Lagrangian markers for the immersed boundary representation consistently accelerates the convergence rate of the algorithm (see Fig. (5.3)). Additionally, the current algorithm exhibited excellent capabilities in capturing the essential flow physics and features for both stationary and prescribed motion cases when compared with experimental and numerical data extracted from the literature (see Chapter 5).

To simulate complex, arbitrary free motions, the present algorithm is coupled with the rigid body dynamics (RBD) equations of motion, commonly referred to as the Newton-Euler equations. This work presents the numerical modelling of fluid–rigid body interactions within the DF-IBM framework, along with the general mathematical formulation resulting from such coupling. Several challenges arise in this type of FSI problem that can impact the stability and convergence of the coupling algorithm. These challenges include limitations imposed by the solid–fluid density ratio and the internal mass effect (IME) rigid body approximation. To address these issues, a partitioned coupling approach was adopted, ensuring modularity and flexibility for both the fluid and rigid body solvers. Explicit (one-way) and implicit (two-way) coupling algorithms were proposed to achieve weak and strong coupling of the interface conditions, respectively. For the implicit algorithm, a fixed-point strategy was employed due to its simplicity. The fixed relaxation technique proved effective in mitigating difficulties related to the IME in cases involving rotational dynamics for both explicit and implicit algorithms. It also addressed challenges stemming from the limitations of the solid–fluid density ratio for neutrally buoyant and non-buoyant rigid bodies. The IME was discretised using an explicit scheme, which successfully circumvented the critical value of the density ratio. Meanwhile, the implicit coupling algorithm was distinguished by its efficiency compared to ALE methods. It leveraged the predictor-corrector strategy and the developed DF-IBM algorithm by eliminating the momentum predictor step and the computationally expensive PISO corrector loops from the implicit coupling iterations. Additionally, the reduced size difference between the Eulerian and Lagrangian domains meant that the Lagrangian mesh updates required for the implicit coupling iterations were significantly less computationally demanding than the Eulerian mesh updates needed in ALE formulations. This further advocates for the efficiency and practicality of the proposed method (see Chapter 7).

Finally, both algorithms proved effective in handling the complex dynamics (translational and/or rotational) of fluid–rigid body interactions across a range of challenging scenarios and intricate geometries. The present development achieved converged and stable computations over a broad span of density ratios. The results highlighted a significant dependence on the inclusion of the IME in the computations. The proposed fixed relaxation technique effectively addressed the instabilities and convergence issues caused by the IME in rotational dynamics. Although a phase shift was observed in the computed quantities during rotational motion when the solid rotational Reynolds number exceeded 10, the core flow physics and rigid body dynamics were accurately captured. It was further concluded that reducing the time-step or refining the mesh resolution improved the convergence rate of the implicit coupling iterations. However, this improvement did not correspond directly to a reduction in the total simulation execution time. Overall, the proposed coupling algorithms exhibited excellent agreement with reference data from the literature (see Chapter 7).

8.2 Future Work

While the current development has yielded promising results, there remains significant potential for enhancement and further exploration, particularly in various numerical aspects. These opportunities for improvement and potential extensions are outlined below:

1. An extension of the present work to three-dimensional (3D) flows is recommended, as the three-dimensional effects of fluid motion on structures are crucial for accurately studying fluid–structure interaction (FSI) problems. These effects play a pivotal role in capturing the complex physics and realistic behavior inherent to such systems. Unlike two-dimensional (2D) approximations, 3D modelling provides a more comprehensive and realistic representation of physical phenomena. Many practical FSI applications are inherently three-dimensional, thus simplifying these problems to 2D risks overlooking critical flow features. Moreover, 3D modelling excels at capturing complex flow structures, such as turbulent eddies and flow separation, in all spatial directions. Additionally, extending to 3D enhances the method's applicability across a wide range of systems, from microscale devices, such as micro-electromechanical systems (MEMS), to macroscale systems, including full-scale 3D wind turbines. While 2D models are computationally efficient and useful for preliminary analyses, they fail to fully encapsulate the depth and intricacy of physical interactions seen in 3D flows. Thus, transitioning to 3D modelling is essential for achieving high-fidelity simulations and enabling the reliable design and optimization of fluid–structure systems.

2. Turbulence models are essential in CFD for accurately simulating and analyzing turbulent flows, which are defined by chaotic and unsteady fluid motion. Within the context of the IBM, turbulent flows are gaining increasing attention, highlighting the growing need for the development of accurate and reliable wall-models. A key limitation of the IBM lies in its inability to precisely cluster fluid grid nodes near the immersed boundary to resolve the thin boundary layer, particularly in high Reynolds number flows. To overcome this limitation and expand the scope of the current work to include turbulent-dominated flows at high Reynolds numbers, the implementation of accurate wall models is recommended. These models could be incorporated either within the framework of RANS turbulence models, as demonstrated in [70, 71], or LES turbulence models, as in [72–74, 184, 185]. Alternatively, hybrid RANS/LES wall models, such as those explored in [186], could offer a balanced approach between computational efficiency and accuracy. Furthermore, this suggestion aligns with the importance of transitioning to three-dimensional modelling. Turbulence is inherently three-dimensional, and modelling turbulent or transitional flows in a 2D framework introduces significant inaccuracies, as many turbulence-related phenomena depend on the interaction of flow structures across all three dimensions.

3. To broaden the applicability of the present method to flexible, deforming solid structures, it is proposed to couple the current development with a FEM solver. Such coupling strategies have been successfully implemented in recent studies, including [187–190]. This integration allows the IBM to address complex biomechanics phenomena, such as the intricate interaction between blood flow and the aortic valve [188, 190], a critical aspect of cardiovascular health. Furthermore, the extended method can also encompass applications directly related to the field of ocean engineering and naval architecture [189], providing a powerful tool for modelling the dynamic behavior of flexible marine structures. This includes floating offshore platforms, ship hulls, and other deformable structures operating in challenging marine environments, where accurate fluid–structure interaction simulations are essential for design and optimization.

4. The undesirable diffused immersed boundary or interface inherent in the DF-IBM can be reduced by adopting a single-sided force spreading operator, as proposed in [92]. In this approach, the spreading operator is applied exclusively to the internal portion of the fluid located inside the immersed boundary. This preserves the sharpness of the interface and ensures accurate capture of velocity gradients, which is particularly

critical for turbulent flow simulations; bringing attention back to the considerations discussed in point number two. However, further research is necessary to ensure that the torque identity imposed by the IBM is preserved, as failure to do so may result in inaccuracies in hydrodynamic torque computations for torque-driven flows. This issue was highlighted in [92] and requires careful consideration to maintain the reliability of the method in such flows. An additional, often overlooked advantage of the single-sided force spreading approach is its reduced computational cost. By restricting the force spreading to internal fluid grid points only, the CPU time required for the communication process between each Lagrangian marker and its neighbouring fluid grid points is reduced by approximately half, compared to the traditional two-sided spreading operator.

5. Another recommendation is to adopt the "field-extension" approach presented in [89] if the spurious force oscillations produced by the DF-IBM are deemed excessive. These unwanted and unphysical oscillations have been shown to over-predict aerodynamic coefficients. By extrapolating the velocity and pressure fields from adjacent fluid grid points to the dynamically changing cells; those transitioning between fluid grid points and solid markers, and vice versa, the time history, i.e., the explicit terms in the fluid equations can be computed with greater accuracy. This approach effectively eliminates spatial and temporal discontinuities in the pressure and velocity fields, respectively, as the immersed boundary moves across the fluid grid, leading to more physically consistent results.

6. Finally, to preserve the second-order spatial accuracy of the method, even in the presence of non-smooth fields with sharp gradients and discontinuities that can locally degrade the solution accuracy, it is recommended to adopt the moving-least-squares (MLS) reconstruction method. This approach would replace the commonly used regularised discrete Dirac delta function for constructing the mapping function between the Lagrangian and Eulerian grids. The MLS method was first introduced by Vanella and Balaras [117] and later improved by [191], by eliminating its associated numerical instabilities and significantly reducing computational cost. Additionally, a single-sided MLS technique was proposed in [192], which combines the second-order spatial accuracy of the MLS method with a single-sided force spreading operator. This method further reduces the diffused boundary thickness and preserves its sharp interface, as highlighted in point number four.

Bibliography

[1] T. Nakamura and S. C. Yim. "A Nonlinear Three-Dimensional Coupled Fluid-Sediment Interaction Model for Large Seabed Deformation". In: *Journal of Offshore Mechanics and Arctic Engineering* 133.3 (2011), p. 031103. DOI: 10.1115/1.4002733.

[2] R. Kamakotim and W. Shyy. "Fluid-structure interaction for aeroelastic applications". In: *Progress in Aerospace Sciences* 40.8 (2004), pp. 535–558. DOI: 10.1016/j.paerosci.2005.01.001.

[3] S. C. Vigmostad, H. S. Udaykumar, J. Lu, and K. B. Chandran. "Fluid-structure interaction methods in biological flows with special emphasis on heart valve dynamics". In: *International Journal for Numerical Methods in Biomedical Engineering* 26.3-4 (2010), pp. 435–470. DOI: 10.1002/cnm.1340.

[4] S. E. Hirdaris, N. J. White, N. Angoshtari, M. C. Johnson, Y. Lee, and N. Bakkers. "Wave loads and flexible fluid-structure interactions: current developments and future directions". In: *Ships and Offshore Structures* 5.4 (2010), pp. 307–325. DOI: 10.1080/17445301003626263.

[5] J.F. Thompson, B.K. Soni, and N.P. Weatherill. *Handbook of Grid Generation*. CRC Press, 1998.

[6] J. Hermansson and P. Hansbo. "A variable diffusion method for mesh smoothing". In: *Communications in Numerical Methods in Engineering* 19.11 (2003), pp. 897–908. DOI: 10.1002/cnm.639.

[7] T. E. Tezduyar, S. Sathe, R. Keedy, and K. Stein. "Space-time finite element techniques for computation of fluid-structure interactions". In: *Computer Methods in Applied Mechanics and Engineering* 195.17 (2006), pp. 2002–2027. DOI: 10.1016/j.cma.2004.09.014.

[8] T. Nakata and H. Liu. "A fluid-structure interaction model of insect flight with flexible wings". In: *Journal of Computational Physics* 231.4 (2012), pp. 1822–1847. DOI: 10.1016/j.jcp.2011.11.005.

[9] S. Roy, A. De, and E. Balaras. *Immersed Boundary Method: Development and Applications*. Computational Methods in Engineering & the Sciences. Springer Nature Singapore, 2020.

[10] X. Wang, C. Wang, and L.T. Zhang. "Semi-implicit formulation of the immersed finite element method". In: *Computational Mechanics* 49 (2012), pp. 421–430. DOI: 10.1007/s00466-011-0652-z.

[11] J. Donea, A. Huerta, J.-Ph. Ponthot, and A. Rodríguez-Ferran. "Arbitrary Lagrangian-Eulerian Methods". In: *Encyclopedia of Computational Mechanics*. John Wiley & Sons, Ltd, 2004. Chap. 14. DOI: 10.1002/0470091355.ecm009.

[12] J. L. Steger, F. C. Dougherty, and J. A. Benek. "A Chimera Grid Scheme. [multiple overset body-conforming mesh system for finite difference adaptation to complex aircraft configurations]". In: *Advances in Grid Generation: Presented at Applied Mechanics, Bioengineering, and Fluids Engineering Conference*. Houston, Texas, 1983.

[13] J. A. Benek, P. G. Buning, and J. L. Steger. "A 3-D chimera grid embedding technique". In: *7th Computational Physics Conference*. Cincinnati, OH: AIAA, 1985. DOI: 10.2514/6.1985-1523.

[14] S. Feghali. "Novel monolithic stabilized finite element method for fluid-structure interaction". PhD thesis. École nationale supérieure des mines de Paris, 2012.

[15] M. Souli, A. Ouahsine, and L. Lewin. "ALE formulation for fluid-structure interaction problems". In: *Computer Methods in Applied Mechanics and Engineering* 190 (2000), pp. 659–675. DOI: 10.1016/S0045-7825(99)00432-6.

[16] P. H. Saksono, W. G. Dettmer, and D. Perić. "An adaptive remeshing strategy for flows with moving boundaries and fluid-structure interaction". In: *International Journal for Numerical Methods in Engineering* 71.9 (2007), pp. 1009–1050. DOI: 10.1002/nme.1971.

[17] C. M. Murea. "Arbitrary Lagrangian Eulerian approximation with remeshing for Navier-Stokes equations". In: *International Journal for Numerical Methods in Biomedical Engineering* 26.11 (2010), pp. 1435–1448. DOI: 10.1002/cnm.1223.

[18] T. Tezduyar. "Finite Element Methods for Flow Problems with Moving Boundaries and Interfaces". In: *Archives of Computational Methods in Engineering* 8.2 (2001), pp. 83–130. DOI: 10.1007/BF02897870.

[19] A. Zilian and A. Legay. "The enriched space-time finite element method (EST) for simultaneous solution of fluid-structure interaction". In: *International Journal for Numerical Methods in Engineering* 75.3 (2008), pp. 305–334. DOI: 10.1002/nme.2258.

[20] B. Roget and J. Sitaraman. “Robust and efficient overset grid assembly for partitioned unstructured meshes”. In: *Journal of Computational Physics* 260 (2014), pp. 1–24. DOI: 10.1016/j.jcp.2013.12.021.

[21] A. Wissink, A. Katz, W. Chan, and R. Meakin. “Validation of the Strand Grid Approach”. In: *19th AIAA Computational Fluid Dynamics*. San Antonio, Texas: AIAA, 2012. DOI: 10.2514/6.2009-3792.

[22] K. W. Cho, J. H. Kwon, and S. Lee. “Development of a Fully Systemized Chimera Methodology for Steady/Unsteady Problems”. In: *Journal of Aircraft* 36.6 (1999), pp. 973–980. DOI: 10.2514/2.2538.

[23] K. Nakahashi, F. Togashi, and D. Sharov. “Intergrid-Boundary Definition Method for Overset Unstructured Grid Approach”. In: *AIAA Journal* 38.11 (2000), pp. 2077–2084. DOI: 10.2514/2.869.

[24] F. Togashi, Y. Ito, K. Nakahashi, and S. Obayashi. “Extensions of Overset Unstructured Grids to Multiple Bodies in Contact”. In: *Journal of Aircraft* 43.1 (2006), pp. 52–57. DOI: 10.2514/1.540.

[25] T. Shih. “Overset Grids: Fundamental and Practical Issues”. In: *20th AIAA Applied Aerodynamics Conference*. St. Louis, Missouri: AIAA, 2002. DOI: 10.2514/6.2002-3259.

[26] S. Völkner, J. Brunswig, and T. Rung. “Analysis of non-conservative interpolation techniques in overset grid finite-volume methods”. In: *Computers & Fluids* 148 (2017), pp. 39–55. DOI: 10.1016/j.compfluid.2017.02.010.

[27] R. Löhner, C. Yang, J. D. Baum, H. Luo, D. Pelessone, and C. M. Charman. “The numerical simulation of strongly unsteady flow with hundreds of moving bodies”. In: *International Journal for Numerical Methods in Fluids* 31.1 (1999), pp. 113–120. DOI: 10.1002/(SICI)1097-0363(19990915)31:1<113::AID-FLD958>3.0.CO;2-Q.

[28] R. Ramamurti and W. Sandberg. “Simulation of Flow About Flapping Airfoils Using Finite Element Incompressible Flow Solver”. In: *AIAA Journal* 39.2 (2001), pp. 253–260. DOI: 10.2514/2.1320.

[29] R. Verzicco. “Immersed Boundary Methods: Historical Perspective and Future Outlook”. In: *Annual Review of Fluid Mechanics* 55 (2023), pp. 129–155. DOI: 10.1146/annurev-fluid-120720-022129.

[30] C. S. Peskin. “Flow patterns around heart valves: A digital computer method for solving the equations of motion”. PhD thesis. Albert Einstein College of Medicine, Yeshiva University, 1972a.

[31] C. S. Peskin. “Flow patterns around heart valves: A numerical method”. In: *Journal of Computational Physics* 10.2 (1972b), pp. 252–271. DOI: 10.1016/0021-9991(72)90065-4.

[32] C. S. Peskin. “Numerical analysis of blood flow in the heart”. In: *Journal of Computational Physics* 25.3 (1977), pp. 220–252. DOI: 10.1016/0021-9991(77)90100-0.

[33] D. Boffi and L. Gastaldi. “Stability and geometric conservation laws for ALE formulations”. In: *Computer Methods in Applied Mechanics and Engineering* 193.42 (2004), pp. 4717–4739. DOI: 10.1016/j.cma.2004.02.020.

[34] C. S. Peskin. “The immersed boundary method”. In: *Acta Numerica* 11 (2002), pp. 479–517. DOI: 10.1017/S0962492902000077.

[35] M.-C. Lai and C. 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. DOI: 10.1006/jcph.2000.6483.

[36] B. E. Griffith and C. S. Peskin. “On the order of accuracy of the immersed boundary method: Higher order convergence rates for sufficiently smooth problems”. In: *Journal of Computational Physics* 208.1 (2005), pp. 75–105. DOI: 10.1016/j.jcp.2005.02.011.

[37] D. Goldstein, R. Handler, and L. Sirovich. “Modeling a No-Slip Flow Boundary with an External Force Field”. In: *Journal of Computational Physics* 105.2 (1993), pp. 354–366. DOI: 10.1006/jcph.1993.1081.

[38] E. M. Saiki and S. Biringen. “Numerical Simulation of a Cylinder in Uniform Flow: Application of a Virtual Boundary Method”. In: *Journal of Computational Physics* 123.2 (1996), pp. 450–465. DOI: 10.1006/jcph.1996.0036.

[39] M. 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. DOI: 10.1016/j.jcp.2005.03.017.

[40] S.-W. Su, M.-C. Lai, and C.-A. Lin. "An immersed boundary technique for simulating complex flows with rigid boundary". In: *Computers & Fluids* 36.2 (2007), pp. 313–324. DOI: 10.1016/j.compfluid.2005.09.004.

[41] T. Kempe and J. Fröhlich. "An improved immersed boundary method with direct forcing for the simulation of particle laden flows". In: *Journal of Computational Physics* 231.9 (2012), pp. 3663–3684. DOI: 10.1016/j.jcp.2012.01.021.

[42] R. Glowinski, T.-W. Pan, and J. Periaux. "A fictitious domain method for external incompressible viscous flow modeled by Navier-Stokes equations". In: *Computer Methods in Applied Mechanics and Engineering* 112.1 (1994), pp. 133–148. DOI: 10.1016/0045-7825(94)90022-1.

[43] H. S. Udaykumar, W. Shyy, and M. M. Rao. "Elafint: a mixed Eulerian-Lagrangian method for fluid flows with complex and moving boundaries". In: *International Journal for Numerical Methods in Fluids* 22.8 (1996), pp. 691–712. DOI: 10.1002/(SICI)1097-0363(19960430)22:8<691::AID-FLD371>3.0.CO;2-U.

[44] T. Ye, R. Mittal, H. S. Udaykumar, and W. Shyy. "An Accurate Cartesian Grid Method for Viscous Incompressible Flows with Complex Immersed Boundaries". In: *Journal of Computational Physics* 156.2 (1999), pp. 209–240. DOI: 10.1006/jcph.1999.6356.

[45] H. S. Udaykumar, R. Mittal, P. Rampunggoon, and A. Khanna. "A Sharp Interface Cartesian Grid Method for Simulating Flows with Complex Moving Boundaries". In: *Journal of Computational Physics* 174.1 (2001), pp. 345–380. DOI: 10.1006/jcph.2001.6916.

[46] L. Schneiders, D. Hartmann, M. Meinke, and W. Schröder. "An accurate moving boundary formulation in cut-cell methods". In: *Journal of Computational Physics* 235 (2013), pp. 786–809. DOI: 10.1016/j.jcp.2012.09.038.

[47] Y.-H. Tseng and J. H. Ferziger. "A ghost-cell immersed boundary method for flow in complex geometry". In: *Journal of Computational Physics* 192.2 (2003), pp. 593–623. DOI: 10.1016/j.jcp.2003.07.024.

[48] R. Ghias, R. Mittal, and H. Dong. "A sharp interface immersed boundary method for compressible viscous flows". In: *Journal of Computational Physics* 225.1 (2007), pp. 528–553. DOI: 10.1016/j.jcp.2006.12.007.

[49] R. Mittal, H. Dong, M. Bozkurttas, F. M. Najjar, A. Vargas, and A. Von Loebbecke. "A versatile sharp interface immersed boundary method for incompressible flows with complex boundaries". In: *Journal of Computational Physics* 227.10 (2008), pp. 4825–4852. DOI: 10.1016/j.jcp.2008.01.028.

[50] R. J. LeVeque and Z. Li. "The Immersed Interface Method for Elliptic Equations with Discontinuous Coefficients and Singular Sources". In: *SIAM Journal on Numerical Analysis* 31.4 (1994), pp. 1019–1044. DOI: 10.1137/0731054.

[51] Z. Li and M.-C. Lai. "The Immersed Interface Method for the Navier-Stokes Equations with Singular Forces". In: *Journal of Computational Physics* 171.2 (2001), pp. 822–842. DOI: 10.1006/jcph.2001.6813.

[52] L. Lee and R. J. LeVeque. "An Immersed Interface Method for Incompressible Navier–Stokes Equations". In: *SIAM Journal on Scientific Computing* 25.3 (2003), pp. 832–856. DOI: 10.1137/S1064827502414060.

[53] M. Meyer, A. Devesa, S. Hickel, X. Y. Hu, and N. A. Adams. "A conservative immersed interface method for Large-Eddy Simulation of incompressible flows". In: *Journal of Computational Physics* 229.18 (2010), pp. 6300–6317. DOI: 10.1016/j.jcp.2010.04.040.

[54] F. Sotiropoulos and X. Yang. "Immersed boundary methods for simulating fluid-structure interaction". In: *Progress in Aerospace Sciences* 65 (2014), pp. 1–21. DOI: 10.1016/j.paerosci.2013.09.003.

[55] W. Kim and H. Choi. "Immersed boundary methods for fluid-structure interaction: A review". In: *International Journal of Heat and Fluid Flow* 75 (2019), pp. 301–309. DOI: 10.1016/j.ijheatfluidflow.2019.01.010.

[56] W.-X. Huang and F.-B. Tian. "Recent trends and progress in the immersed boundary method". In: *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* 233.23–24 (2019), pp. 7617–7636. DOI: 10.1177/0954406219842606.

[57] B. E. Griffith and N. A. Patankar. "Immersed Methods for Fluid-Structure Interaction". In: *Annual Review of Fluid Mechanics* 52 (2020), pp. 421–448. DOI: 10.1146/annurev-fluid-010719-060228.

[58] R. Mittal and J. H. Seo. "Origin and evolution of immersed boundary methods in computational fluid dynamics". In: *Phys. Rev. Fluids* 8.10 (2023), p. 100501. DOI: 10.1103/PhysRevFluids.8.100501.

[59] J.-H. Seo and R. Mittal. "Improved swimming performance in schooling fish via leading-edge vortex enhancement". In: *Bioinspiration & Biomimetics* 17.6 (2022), p. 066020. DOI: 10.1088/1748-3190/ac9bb4.

[60] J. Song, H. Luo, and T. L. Hedrick. "Three-dimensional flow and lift characteristics of a hovering ruby-throated hummingbird". In: *Journal of The Royal Society Interface* 11.98 (2014), p. 20140541. DOI: 10.1098/rsif.2014.0541.

[61] H. Luo, H. Dai, P. Ferreira de Sousa, and B. Yin. "On the numerical oscillation of the direct-forcing immersed-boundary method for moving boundaries". In: *Computers & Fluids* 56 (2012), pp. 61–76. DOI: 10.1016/j.compfluid.2011.11.015.

[62] R. Verzicco and G. Querzoli. "On the collision of a rigid sphere with a deformable membrane in a viscous fluid". In: *Journal of Fluid Mechanics* 914 (2021), A19. DOI: 10.1017/jfm.2020.939.

[63] A. Scotti, M. Gasser i. Rubinat, and E. Balaras. "LES of Pulsating Turbulent Flows over Smooth and Wavy Boundaries". In: *Turbulence and Interactions*. Berlin, Heidelberg: Springer Nature, 2010, pp. 25–36. DOI: 10.1007/978-3-642-14139-3_4.

[64] M. Aghaei-Jouybari, J.-H. Seo, J. Yuan, R. Mittal, and C. Meneveau. "Contributions to pressure drag in rough-wall turbulent flows: Insights from force partitioning". In: *Phys. Rev. Fluids* 7.8 (2022), p. 084602. DOI: 10.1103/PhysRevFluids.7.084602.

[65] M. D. de Tullio, P. De Palma, G. Iaccarino, G. Pascazio, and M. Napolitano. "An immersed boundary method for compressible flows using local grid refinement". In: *Journal of Computational Physics* 225.2 (2007), pp. 2098–2117. DOI: 10.1016/j.jcp.2007.03.008.

[66] M. Al-Marouf and R. Samtaney. "A versatile embedded boundary adaptive mesh method for compressible flow in complex geometry". In: *Journal of Computational Physics* 337 (2017), pp. 339–378. DOI: 10.1016/j.jcp.2017.02.044.

[67] I. Tsetoglou, M. Cailler, P. Bénard, G. Lartigue, V. Moureau, and J. Réveillon. "A volume-of-solid implicit volume penalty method for moving-body flows". In: *International Journal for Numerical Methods in Fluids* 97.2 (2024), pp. 117–150. DOI: 10.1002/fld.5334.

[68] M. Vanella, P. Rabenold, and E. Balaras. "A direct-forcing embedded-boundary method with adaptive mesh refinement for fluid-structure interaction problems". In: *Journal of Computational Physics* 229.18 (2010), pp. 6427–6449. DOI: 10.1016/j.jcp.2010.05.003.

[69] S. Sambasivan, A. Kapahi, and H. S. Udaykumar. "Simulation of high speed impact, penetration and fragmentation problems on locally refined Cartesian grids". In: *Journal of Computational Physics* 235 (2013), pp. 334–370. DOI: 10.1016/j.jcp.2012.10.031.

[70] F. Capizzano. "Turbulent Wall Model for Immersed Boundary Methods". In: *AIAA Journal* 49.11 (2011), pp. 2367–2381. DOI: 10.2514/1.J050466.

[71] S.-G. Cai, J. Degrigny, J.-F. Boussuge, and P. Sagaut. "Coupling of turbulence wall models and immersed boundaries on Cartesian grids". In: *Journal of Computational Physics* 429 (2021), p. 109995. DOI: 10.1016/j.jcp.2020.109995.

[72] A. Cristallo and R. Verzicco. "Combined Immersed Boundary/Large-Eddy-Simulations of Incompressible Three Dimensional Complex Flows". In: *Flow, Turbulence and Combustion* 77 (2006), pp. 3–26. DOI: 10.1007/s10494-006-9034-6.

[73] N. S. Dhamankar, G. A. Blaisdell, and A. S. Lyrintzis. "Implementation of a Wall-Modeled Sharp Immersed Boundary Method in a High-Order Large Eddy Simulation Tool for Jet Aeroacoustics". In: *54th AIAA Aerospace Sciences Meeting*. San Diego, California, 2016. DOI: 10.2514/6.2016-0257.

[74] S.-G. Cai, J. Jacob, and P. Sagaut. "Immersed boundary based near-wall modeling for large eddy simulation of turbulent wall-bounded flow". In: *Computers & Fluids* 259 (2023), p. 105893. DOI: 10.1016/j.compfluid.2023.105893.

[75] J. Mohd-Yusof. "Combined Immersed Boundary/B-Spline Methods for Simulations of Flows in Complex Geometries". In: *Annual Research Briefs, Center for Turbulence Research* (1997), pp. 317–328.

[76] E. A. Fadlun, R. Verzicco, P. Orlandi, and J. Mohd-Yusof. "Combined Immersed-Boundary Finite-Difference Methods for Three-Dimensional Complex Flow Simulations". In: *Journal of Computational Physics* 161.1 (2000), pp. 35–60. DOI: 10.1006/jcph.2000.6484.

[77] K. Luo, Z. Wang, J. Fan, and K. Cen. "Full-scale solutions to particle-laden flows: Multidirect forcing and immersed boundary method". In: *Phys. Rev. E* 76.6 (2007), p. 066709. DOI: 10.1103/PhysRevE.76.066709.

[78] W.-P. Breugem. "A second-order accurate immersed boundary method for fully resolved simulations of particle-laden flows". In: *Journal of Computational Physics* 231.13 (2012), pp. 4469–4498. DOI: 10.1016/j.jcp.2012.02.026.

[79] K. Taira and T Colonius. "The immersed boundary method: A projection approach". In: *Journal of Computational Physics* 225.2 (2007), pp. 2118–2137. DOI: 10.1016/j.jcp.2007.03.005.

[80] C. Ji, A. Munjiza, and J. J. R. Williams. "A novel iterative direct-forcing immersed boundary method and its finite volume applications". In: *Journal of Computational Physics* 231.4 (2012), pp. 1797–1821. DOI: 10.1016/j.jcp.2011.11.010.

[81] S.-G. Cai, A. Ouahsine, J. Favier, and Y. Hoarau. "Moving immersed boundary method". In: *International Journal for Numerical Methods in Fluids* 85.5 (2017), pp. 288–323. DOI: 10.1002/fld.4382.

[82] S. Gsell, U. D'Ortona, and J. Favier. "Explicit and viscosity-independent immersed-boundary scheme for the lattice Boltzmann method". In: *Phys. Rev. E* 100.3 (2019), p. 033306. DOI: 10.1103/PhysRevE.100.033306.

[83] S. Gsell and J. Favier. "Direct-forcing immersed-boundary method: A simple correction preventing boundary slip error". In: *Journal of Computational Physics* 435 (2021), p. 110265. DOI: 10.1016/j.jcp.2021.110265.

[84] I. Cheylan, T. Fringand, J. Jacob, and J. Favier. "Analysis of the immersed boundary method for turbulent fluid-structure interaction with Lattice Boltzmann method". In: *Journal of Computational Physics* 492 (2023), p. 112418. DOI: 10.1016/j.jcp.2023.112418.

[85] A. Pinelli, I.Z. Naqavi, U. Piomelli, and J. Favier. "Immersed-boundary methods for general finite-difference and finite-volume Navier-Stokes solvers". In: *Journal of Computational Physics* 229.24 (2010), pp. 9073–9091. DOI: 10.1016/j.jcp.2010.08.021.

[86] X. Yang, X. Zhang, Z. Li, and G.-W. He. "A smoothing technique for discrete delta functions with application to immersed boundary method in moving boundary simulations". In: *Journal of Computational Physics* 228.20 (2009), pp. 7821–7836. DOI: 10.1016/j.jcp.2009.07.023.

[87] J. Lee, J. Kim, H. Choi, and K.-S. Yang. "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. DOI: 10.1016/j.jcp.2011.01.004.

[88] M. Giahi and D. Bergstrom. "A critical assessment of the immersed boundary method for modeling flow around fixed and moving bodies". In: *Computers & Fluids* 226 (2023), p. 105841. DOI: 10.1016/j.compfluid.2023.105841.

[89] J. Yang and E. Balaras. "An embedded-boundary formulation for large-eddy simulation of turbulent flows interacting with moving boundaries". In: *Journal of Computational Physics* 215 (2006), pp. 12–40. DOI: 10.1016/j.jcp.2005.10.035.

[90] A. M. Roma, C. S. Peskin, and M. J. Berger. "An Adaptive Version of the Immersed Boundary Method". In: *Journal of Computational Physics* 153.2 (1999), pp. 509–534. DOI: 10.1006/jcph.1999.6293.

[91] N. Zhang and Z. C. Zheng. "An improved direct-forcing immersed-boundary method for finite difference applications". In: *Journal of Computational Physics* 221.1 (2007), pp. 250–268. DOI: 10.1016/j.jcp.2006.06.012.

[92] C. Peng and L.-P. Wang. "Force-amplified, single-sided diffused-interface immersed boundary kernel for correct local velocity gradient computation and accurate no-slip boundary enforcement". In: *Phys. Rev. E* 101.5 (2020), p. 053305. DOI: 10.1103/PhysRevE.101.053305.

[93] I. Borazjani, L. Ge, and F. Sotiropoulos. "Curvilinear immersed boundary method for simulating fluid structure interaction with complex 3D rigid bodies". In: *Journal of Computational Physics* 227.16 (2008), pp. 7587–7620. DOI: 10.1016/j.jcp.2008.04.028.

[94] C. Wang and J. D. Eldredge. "Strongly coupled dynamics of fluids and rigid-body systems with the immersed boundary projection method". In: *Journal of Computational Physics* 295 (2015), pp. 87–113. DOI: 10.1016/j.jcp.2015.04.005.

[95] U. Lācis, K. Taira, and S. Bagheri. "A stable fluid-structure-interaction solver for low-density rigid bodies using the immersed boundary projection method". In: *Journal of Computational Physics* 305 (2016), pp. 300–318. DOI: 10.1016/j.jcp.2015.10.041.

[96] Z.-G. Feng and E. E. Michaelides. "Robust treatment of no-slip boundary condition and velocity updating for the lattice-Boltzmann simulation of particulate flows". In: *Computers & Fluids* 38.2 (2009), pp. 370–381. DOI: 10.1016/j.compfluid.2008.04.013.

[97] K. Suzuki and T. Inamuro. "Effect of internal mass in the simulation of a moving body by the immersed boundary method". In: *Computers & Fluids* 49.1 (2011), pp. 173–187. DOI: 10.1016/j.compfluid.2011.05.011.

[98] H. G. Weller, G. Tabor, H. Jasak, and C. Fureby. "A tensorial approach to computational continuum mechanics using object-oriented techniques". In: *Computer in Physics* 12.6 (1998), pp. 620–631. DOI: 10.1063/1.168744.

[99] C. M. Rhie and W. L. Chow. "Numerical study of the turbulent flow past an airfoil with trailing edge separation". In: *AIAA Journal* 21.11 (1983), pp. 1525–1532. DOI: 10.2514/3.8284.

[100] H. Jasak, H.G. Weller, and A.D. Gosman. "High resolution NVD differencing scheme for arbitrarily unstructured meshes". In: *International Journal for Numerical Methods in Fluids* 31.2 (1999), pp. 431–449. DOI: 10.1002/(SICI)1097-0363(19990930)31:2<431::AID-FLD884>3.0.CO;2-T.

[101] F. H. Harlow and J. E. Welch. "Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with Free Surface". In: *The Physics of Fluids* 8.12 (1965), pp. 2182–2189. DOI: 10.1063/1.1761178.

[102] A. J. Chorin. "Numerical Solution of Navier-Stokes Equations". In: *Mathematics of Computations* 22.104 (1968), pp. 745–762.

[103] J. Kim and P. Moin. "Application of a fractional-step method to incompressible Navier-Stokes equations". In: *Journal of Computational Physics* 59.2 (1985), pp. 308–323. DOI: 10.1016/0021-9991(85)90148-2.

[104] J. Van Kan. “A Second-Order Accurate Pressure-Correction Scheme for Viscous Incompressible Flow”. In: *SIAM Journal on Scientific and Statistical Computing* 7.3 (1986), pp. 870–891. DOI: 10.1137/0907059.

[105] J. B. Bell, P. Colella, and H. M. Glaz. “A second-order projection method for the incompressible navier-stokes equations”. In: *Journal of Computational Physics* 85.2 (1989), pp. 257–283. DOI: 10.1016/0021-9991(89)90151-4.

[106] M. M. Rai and P. Moin. “Direct simulations of turbulent flow using finite-difference schemes”. In: *Journal of Computational Physics* 96.1 (1991), pp. 15–53. DOI: 10.1016/0021-9991(91)90264-L.

[107] M.-J. Ni and S. Komori. “General Second-Order Projection Formulas for Unsteady Flows”. In: *AIAA Journal* 40.7 (2002), pp. 1464–1467. DOI: 10.2514/2.1810.

[108] S. V. Patankar and D. B. Spalding. “A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows”. In: *International Journal of Heat and Mass Transfer* 15.10 (1972), pp. 1787–1806. DOI: 10.1016/0017-9310(72)90054-3.

[109] S. V. Patankar. “A Calculation Procedure for Two-Dimensional Elliptic Situations”. In: *Numerical Heat Transfer* 4.4 (1981), pp. 409–425. DOI: 10.1080/01495728108961801.

[110] J. P. Van Doormaal and G. D. Raithby. “Enhancements of the SIMPLE method for predicting incompressible fluid flows”. In: *Numerical Heat Transfer* 7.2 (1984), pp. 147–163. DOI: 10.1080/01495728408961817.

[111] R. I. Issa. “Solution of the implicitly discretised fluid flow equations by operator-splitting”. In: *Journal of Computational Physics* 62.1 (1985), pp. 40–65. DOI: 10.1016/0021-9991(86)90099-9.

[112] G. Iaccarino and R. Verzicco. “Immersed boundary technique for turbulent flow simulations”. In: *Applied Mechanics Reviews* 56.3 (2003), pp. 331–347. DOI: 10.1115/1.1563627.

[113] R. Verzicco, J. Mohd-Yusof, P. Orlandi, and D. C. Haworth. “Large Eddy Simulation in Complex Geometric Configurations Using Boundary Body Forces”. In: *AIAA Journal* 38.3 (2000), pp. 427–433. DOI: 10.2514/2.1001.

[114] R. P. Beyer and R. J. LeVeque. “Analysis of a One-Dimensional Model for the Immersed Boundary Method”. In: *SIAM Journal on Numerical Analysis* 29.2 (1992), pp. 332–364. DOI: 10.1137/0729022.

[115] X. Wang and W. K. Liu. “Extended immersed boundary method using FEM and RKPM”. In: *Computer Methods in Applied Mechanics and Engineering* 193.12 (2004), pp. 1305–1321. DOI: 10.1016/j.cma.2003.12.024.

[116] F. Toja-Silva, J. Favier, and A. Pinelli. “Radial basis function (RBF)-based interpolation and spreading for the immersed boundary method”. In: *Computers & Fluids* 105 (2014), pp. 66–75. DOI: 10.1016/j.compfluid.2014.09.026.

[117] M. Vanella and E. Balaras. “A moving-least-squares reconstruction for embedded-boundary formulations”. In: *Journal of Computational Physics* 228.18 (2009), pp. 6617–6628. DOI: 10.1016/j.jcp.2009.06.003.

[118] Y. Bao, A. D. Kaiser, J. Kaye, and C. S. Peskin. “Gaussian-Like Immersed-Boundary Kernels with Three Continuous Derivatives and Improved Translational Invariance”. In: *arXiv:1505.07529* (2020). DOI: 10.48550/arXiv.1505.07529v4.

[119] Y. Bao, J. Kaye, and C. S. Peskin. “A Gaussian-like immersed-boundary kernel with three continuous derivatives and improved translational invariance”. In: *Journal of Computational Physics* 316 (2016), pp. 139–144. DOI: 10.1016/j.jcp.2016.04.024.

[120] J. Kim, D. Kim, and H. 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. DOI: 10.1006/jcph.2001.6778.

[121] E. Constant. “Développement d’un solveur de frontières immergées dans OpenFOAM: vers le contrôle des vibrations induites par vortex dans le sillage d’un cylindre”. PhD thesis. Aix-Marseille University, Laboratoire de Mécanique, Modélisation et Procédés Propres (M2P2), 2017.

[122] F. Ilinca and Hétu J.-F. “Numerical simulation of fluid-solid interaction using an immersed boundary finite element method”. In: *Computers & Fluids* 59 (2012), pp. 31–43. DOI: 10.1016/j.compfluid.2012.02.008.

[123] B. E. Griffith and X. Luo. “Hybrid finite difference/finite element immersed boundary method”. In: *International Journal for Numerical Methods in Biomedical Engineering* 33.12 (2017), e2888. DOI: 10.1002/cnm.2888.

[124] J. Favier, A. Revell, and A. Pinelli. "A Lattice Boltzmann-Immersed Boundary method to simulate the fluid interaction with moving and slender flexible objects". In: *Journal of Computational Physics* 261 (2014), pp. 145–161. DOI: 10.1016/j.jcp.2013.12.052.

[125] K. Goncharuk, O. Oshri, and Y. Feldman. "The immersed boundary method: A SIMPLE approach". In: *Journal of Computational Physics* 487 (2023), p. 112148. DOI: 10.1016/j.jcp.2023.112148.

[126] E. Constant, J. Favier, M. Meldi, P. Meliga, and E. Serre. "An immersed boundary method in OpenFOAM: Verification and validation". In: *Computers & Fluids* 157 (2017), pp. 55–72. DOI: 10.1016/j.compfluid.2017.08.001.

[127] M. Askarishahi. "Immersed-boundary/soft-sphere method for particle-particle-fluid interaction in a viscous flow: An OpenFOAM solver". In: *Advanced Powder Technology* 34.11 (2023), p. 104204. DOI: 10.1016/j.apt.2023.104204.

[128] S. Wang and X. Zhang. "An immersed boundary method based on discrete stream function formulation for two- and three-dimensional incompressible flows". In: *Journal of Computational Physics* 230.9 (2011), pp. 3479–3499. DOI: 10.1016/j.jcp.2011.01.045.

[129] D. J. Tritton. "Experiments on the flow past a circular cylinder at low Reynolds numbers". In: *Journal of Fluid Mechanics* 6.4 (1959), pp. 547–567. DOI: 10.1017/S0022112059000829.

[130] M. Coutanceau and R. Bouard. "Experimental determination of the main features of the viscous flow in the wake of a circular cylinder in uniform translation. Part 1. Steady flow". In: *Journal of Fluid Mechanics* 79.2 (1977), pp. 231–256. DOI: 10.1017/S0022112077000135.

[131] D. Russell and Z. J. Wang. "A Cartesian grid method for modeling multiple moving objects in 2D incompressible viscous flow". In: *Journal of Computational Physics* 191 (2003), pp. 177–205. DOI: 10.1016/S0021-9991(03)00310-3.

[132] M. Linnick and H. Fasel. "A high-order immersed interface method for simulating unsteady incompressible flows on irregular domains". In: *Journal of Computational Physics* 204 (2005), pp. 157–192. DOI: 10.1016/j.jcp.2004.09.017.

[133] S. Xu and Z. J. Wang. “An immersed interface method for simulating the interaction of a fluid with moving boundaries”. In: *Journal of Computational Physics* 216 (2006), pp. 454–493. DOI: 10.1016/j.jcp.2005.12.016.

[134] C. H. K. Williamson. “Oblique and parallel modes of vortex shedding in the wake of a circular cylinder at low Reynolds numbers”. In: *Journal of Fluid Mechanics* 206 (1989), pp. 579–627. DOI: 10.1017/S0022112089002429.

[135] M. Braza, P. Chassaing, and H. Ha Minh. “Numerical study and physical analysis of the pressure and velocity fields in the near wake of a circular cylinder”. In: *Journal of Fluid Mechanics* 165 (1986), pp. 79–130. DOI: 10.1017/S0022112086003014.

[136] Chloe Mimeau, Federico Gallizio, Georges-Henri Cottet, and Iraj Mortazavi. “Vortex penalization method for bluff body flows”. In: *International Journal for Numerical Methods in Fluids* 79.2 (2015), pp. 55–83. DOI: 10.1002/fld.4038.

[137] R. Mittal and S. Balachandar. “On the inclusion of three-dimensional effects in simulations of two-dimensional bluff-body wake flows”. In: *Proceedings of the 1997 ASME Fluids Engineering Division Summer Meeting* (1997).

[138] R. D. Henderson. “Details of the drag curve near the onset of vortex shedding”. In: *Physics of Fluids* 7.9 (1995), pp. 2102–2104. DOI: 10.1063/1.868459.

[139] H. Persillon and M. Braza. “Physical analysis of the transition to turbulence in the wake of a circular cylinder by three-dimensional Navier-Stokes simulation”. In: *Journal of Fluid Mechanics* 365 (1998), pp. 23–88. DOI: 10.1017/S0022112098001116.

[140] S. Apte, M. Martin, and N. Patankar. “A numerical method for fully resolved simulation (FRS) of rigid particle-flow interactions in complex flows”. In: *Journal of Computational Physics* 228 (2009), pp. 2712–2738. DOI: 10.1016/j.jcp.2008.11.034.

[141] S. Mittal and V. Kumar. “Flow-induced vibrations of a light circular cylinder at Reynolds numbers 10^3 to 10^4”. In: *Journal of Sound and Vibration* 245 (2001), pp. 923–946. DOI: 10.1006/jsvi.2001.3612.

[142] H. Dütsch, F. Durst, S. Becker, and H. Lienhart. "Low-Reynolds-number flow around an oscillating circular cylinder at low Keulegan-Carpenter numbers". In: *Journal of Fluid Mechanics* 360 (1998), pp. 249–271. DOI: 10.1017/S002211209800860X.

[143] J. Yang and F. Stern. "A simple and efficient direct forcing immersed boundary framework for fluid-structure interactions". In: *Journal of Computational Physics* 231.15 (2012), pp. 5029–5061. DOI: 10.1016/j.jcp.2012.04.012.

[144] X.-Y. Lu and C. Dalton. "Calculation of the timing of vortex formation from an oscillating cylinder". In: *Journal of Fluids and Structures* 10.5 (1996), pp. 527–541. DOI: 10.1006/jfls.1996.0035.

[145] E. Guilmineau and P. Queutey. "A Numerical Simulation of vortex shedding from an oscillating circular cylinder". In: *Journal of Fluids and Structures* 16.6 (2002), pp. 773–794. DOI: 10.1006/jfls.2002.0449.

[146] Y. Wang, C. Shu, C. J. Teo, and J. Wu. "An immersed boundary-lattice Boltzmann flux solver and its applications to fluid-structure interaction problems". In: *Journal of Fluids and Structures* 54 (2015), pp. 440–465. DOI: 10.1016/j.jfluidstructs.2014.12.003.

[147] Z. J. Wang. "Two Dimensional Mechanism for Insect Hovering". In: *Phys. Rev. Lett.* 85.10 (2000), pp. 2216–2219. DOI: 10.1103/PhysRevLett.85.2216.

[148] L. Ramírez, C. Foulquié, X. Nogueira, S. Khelladi, J.-C. Chassaing, and I. Colominas. "New high-resolution-preserving sliding mesh techniques for higher-order finite volume schemes". In: *Computers & Fluids* 118 (2015), pp. 114–130. DOI: 10.1016/j.compfluid.2015.06.008.

[149] E. Ferrer and R. 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. DOI: 10.1016/j.jcp.2012.04.039.

[150] E. Ferrer and R. H. J. Willden. "Blade-wake interactions in cross-flow turbines". In: *International Journal of Marine Energy* 11 (2015), pp. 71–83. DOI: 10.1016/j.ijome.2015.06.001.

[151] M. Uhlmann. *First Experiments with the Simulation of Particulate Flows*. Tech. rep. 1020. Informes técnicos Ciemat, 2003.

[152] R. Aris. *Vectors, Tensors and the Basic Equations of Fluid Mechanics*. New York: Dover Publications, 1962.

[153] A. ten Cate, C. H. Nieuwstad, J. J. Derksen, and H. E. A. Van den Akker. "Particle imaging velocimetry experiments and lattice-Boltzmann simulations on a single sphere settling under gravity". In: *Physics of Fluids* 14.11 (2002), pp. 4012–4025. DOI: 10.1063/1.1512918.

[154] T. Kempe and J. Fröhlich. "Collision modelling for the interface-resolved simulation of spherical particles in viscous fluids". In: *Journal of Fluid Mechanics* 709 (2012), pp. 445–489. DOI: 10.1017/jfm.2012.343.

[155] R. Glowinski, T.-W. Pan, T.I. Hesla, and D.D. Joseph. "A distributed Lagrange multiplier/fictitious domain method for particulate flows". In: *International Journal of Multiphase Flow* 25.5 (1999), pp. 755–794. DOI: 10.1016/S0301-9322(98)00048-2.

[156] P. Singh, D.D. Joseph, T.I. Hesla, R. Glowinski, and T.-W. Pan. "A distributed Lagrange multiplier/fictitious domain method for viscoelastic particulate flows". In: *Journal of Non-Newtonian Fluid Mechanics* 91 (2000), pp. 165–188. DOI: 10.1016/S0377-0257(99)00104-4.

[157] D. Wan and S. Turek. "Direct numerical simulation of particulate flow via multigrid FEM techniques and the fictitious boundary method". In: *International Journal for Numerical Methods in Fluids* 51.5 (2006), pp. 531–566. DOI: 10.1002/fld.1129.

[158] C. Kassiotis. "Nonlinear fluid-structure interaction: a partitioned approach and its application through component technology". PhD thesis. Université Paris-Est, 2009.

[159] E. H. Dowell and K. C. Hall. "Modeling of fluid-structure interaction". In: *Annual Review of Fluid Mechanics* 33 (2001), pp. 445–490. DOI: 10.1146/annurev.fluid.33.1.445.

[160] G. Hou, J. Wang, and A. Layton. "Numerical Methods for Fluid-Structure Interaction - A Review". In: *Communications in Computational Physics* 12.2 (2012), pp. 337–377. DOI: 10.4208/cicp.291210.290411s.

[161] H. G. Matthies and J. Steindorf. "Partitioned but strongly coupled iteration schemes for nonlinear fluid-structure interaction". In: *Computers & Structures* 80.27 (2002), pp. 1991–1999. DOI: 10.1016/S0045-7949(02)00259-6.

[162] H. G. Matthies and J. Steindorf. “Partitioned strong coupling algorithms for fluid-structure interaction”. In: *Computers & Structures* 81.8 (2003), pp. 805–812. DOI: 10.1016/S0045-7949(02)00409-1.

[163] H. G. Matthies, R. Niekamp, and J. Steindorf. “Algorithms for strong coupling procedures”. In: *Computer Methods in Applied Mechanics and Engineering* 195.17 (2006), pp. 2028–2049. DOI: 10.1016/j.cma.2004.11.032.

[164] C. Kassiotis, A. Ibrahimbegovic, R. Niekamp, and H. Matthies. “Nonlinear fluid-structure interaction problem. Part I: Implicit partitioned algorithm, nonlinear stability proof and validation examples”. In: *Computational Mechanics* 47 (2011), pp. 305–323. DOI: 10.1007/s00466-010-0545-6.

[165] C. Kassiotis, A. Ibrahimbegovic, R. Niekamp, and H. Matthies. “Nonlinear fluid-structure interaction problem. Part II: Space discretization, implementation aspects, nested parallelization and application examples”. In: *Computational Mechanics* 47 (2011), pp. 335–357. DOI: 10.1007/s00466-010-0544-7.

[166] U. Küttler and A. W. Wall. “Fixed-point fluid-structure interaction solvers with dynamic relaxation”. In: *Computational Mechanics* 43 (2008), pp. 61–72. DOI: 10.1007/s00466-008-0255-5.

[167] M. M. Joosten, W. G. Dettmer, and D. Perić. “Analysis of the block Gauss-Seidel solution procedure for a strongly coupled model problem with reference to fluid-structure interaction”. In: *International Journal for Numerical Methods in Engineering* 78.7 (2009), pp. 757–778. DOI: 10.1002/nme.2503.

[168] B. M. Irons and R. C. Tuck. “A version of the Aitken accelerator for computer iteration”. In: *International Journal for Numerical Methods in Engineering* 1.3 (1969), pp. 275–277. DOI: 10.1002/nme.1620010306.

[169] L. D. Marini and A. Quarteroni. “A relaxation procedure for domain decomposition methods using finite elements”. In: *Numerische Mathematik* 55 (1989), pp. 575–598. DOI: 10.1007/BF01398917.

[170] P. Le Tallec and J. Mouro. “Fluid structure interaction with large structural displacements”. In: *Computer Methods in Applied Mechanics and Engineering* 190.24 (2001), pp. 3039–3067. DOI: 10.1016/S0045-7825(00)00381-9.

[171] P. Causin, Gerbeau J. F., and F. Nobile. "Added-mass effect in the design of partitioned algorithms for fluid-structure problems". In: *Computer Methods in Applied Mechanics and Engineering* 194.42 (2005), pp. 4506–4527. DOI: 10.1016/j.cma.2004.12.005.

[172] M. A. Fernández, J.-F. Gerbeau, and C. Grandmont. "A projection semi-implicit scheme for the coupling of an elastic structure with an incompressible fluid". In: *International Journal for Numerical Methods in Engineering* 69.4 (2007), pp. 794–821. DOI: 10.1002/nme.1792.

[173] S. Badia, A. Quaini, and A. Quarteroni. "Modular vs. non-modular preconditioners for fluid-structure systems with large added-mass effect". In: *Computer Methods in Applied Mechanics and Engineering* 197.49 (2008), pp. 4216–4232. DOI: 10.1016/j.cma.2008.04.018.

[174] S. R. Idelsohn, F. Del Pin, R. Rossi, and E. Oñate. "Fluid-structure interaction problems with strong added-mass effect". In: *International Journal for Numerical Methods in Engineering* 80.10 (2009), pp. 1261–1294. DOI: 10.1002/nme.2659.

[175] K. C. Ong, Y. Seol, and M.-C. Lai. "An immersed boundary projection method for solving the fluid-rigid body interaction problems". In: *Journal of Computational Physics* 466 (2022), p. 111367. DOI: 10.1016/j.jcp.2022.111367.

[176] S.-G. Cai. "Computational fluid-structure interaction with the moving immersed boundary method". PhD thesis. Compiègne, France: Université de Technologie de Compiègne, 2016.

[177] R. Glowinski, T.W. Pan, T.I. Hesla, D.D. Joseph, and J. Périaux. "A Fictitious Domain Approach to the Direct Numerical Simulation of Incompressible Viscous Flow past Moving Rigid Bodies: Application to Particulate Flow". In: *Journal of Computational Physics* 169.2 (2001), pp. 363–426. DOI: 10.1006/jcph.2000.6542.

[178] Z. Xia, K. W. Connington, S. Rapaka, P. Yue, J. J. Feng, and S. Chen. "Flow patterns in the sedimentation of an elliptical particle". In: *Journal of Fluid Mechanics* 625 (2009), pp. 249–272. DOI: 10.1017/S0022112008005521.

[179] K. Namkoong, J. Y. Yoo, and H. G. Choi. "Numerical analysis of two-dimensional motion of a freely falling circular cylinder in an infinite fluid". In: *Journal of Fluid Mechanics* 604 (2008), pp. 33–53. DOI: 10.1017/S0022112008001304.

[180] R. Bravo, S. Tullis, and S. Ziada. “Performance Testing of a Small Vertical-Axis Wind Turbine”. In: *Proceedings of the 21st Canadian Congress of Applied Mechanics* (2007).

[181] László Daróczy, Gábor Janiga, Klaus Petrasch, Michael Webner, and Dominique Thévenin. “Comparative analysis of turbulence models for the aerodynamic simulation of H-Darrieus rotors”. In: *Energy* 90 (2015), pp. 680–690. DOI: 10.1016/j.energy.2015.07.102.

[182] B. Hand, G. Kelly, and A. Cashman. “Aerodynamic design and performance parameters of a lift-type vertical axis wind turbine: A comprehensive review”. In: *Renewable and Sustainable Energy Reviews* 139 (2021), p. 110699. DOI: 10.1016/j.rser.2020.110699.

[183] A. García Auyanet, R. E. Santoso, H. Mohan, S. S. Rathore, D. Chakraborty, and P. G. Verdin. “CFD-Based J-Shaped Blade Design Improvement for Vertical Axis Wind Turbines”. In: *Sustainability* 14.22 (2022), p. 15343. DOI: 10.3390/su142215343.

[184] S.-G. Cai, S. Mozaffari, J. Jacob, and P. Sagaut. “Application of immersed boundary based turbulence wall modeling to the Ahmed body aerodynamics”. In: *Physics of Fluids* 34.9 (2022), p. 095106. DOI: 10.1063/5.0098232.

[185] L. Liu and R. J. A. M. Stevens. “Wall modeled immersed boundary method for high Reynolds number flow over complex terrain”. In: *Computers & Fluids* 208 (2020), p. 104604. DOI: 10.1016/j.compfluid.2020.104604.

[186] S. Mozaffari, S.-G. Cai, J. Jacob, and P. Sagaut. “Lattice Boltzmann $k-\omega$ SST based hybrid RANS/LES simulations of turbulent flows”. In: *Journal of Computational Physics* 514 (2024), p. 113269. DOI: 10.1016/j.jcp.2024.113269.

[187] Z. Li and Julien Favier. “A non-staggered coupling of finite element and lattice Boltzmann methods via an immersed boundary scheme for fluid-structure interaction”. In: *Computers & Fluids* 143 (2017), pp. 90–102. DOI: 10.1016/j.compfluid.2016.11.008.

[188] Z. Li, G. Oger, and D. Le Touzé. “A partitioned framework for coupling LBM and FEM through an implicit IBM allowing non-conforming time-steps: Application to fluid-structure interaction in biomechanics”. In: *Journal of Computational Physics* 449 (2022), p. 110786. DOI: 10.1016/j.jcp.2021.110786.

[189] Y. Di, L. Zhao, and J. Mao. “A hybrid FEM-IBM-level set algorithm for water entry of deformable body”. In: *Ocean Engineering* 306 (2024), p. 118007. DOI: 10.1016/j.oceaneng.2024.118007.

[190] T. Fringand, I. Cheylan, M. Lenoir, L. Mace, and J. Favier. “A stable and explicit fluid-structure interaction solver based on lattice-Boltzmann and immersed boundary methods”. In: *Computer Methods in Applied Mechanics and Engineering* 421 (2024), p. 116777. DOI: 10.1016/j.cma.2024.116777.

[191] D. Li, A. Wei, K. Luo, and J. Fan. “An improved moving-least-squares reconstruction for immersed boundary method”. In: *International Journal for Numerical Methods in Engineering* 104.8 (2015), pp. 789–804. DOI: 10.1002/nme.4949.

[192] R. Bale, A. P. S. Bhalla, B. E. Griffith, and M. Tsubokura. “A one-sided direct forcing immersed boundary method using moving least squares”. In: *Journal of Computational Physics* 440 (2021), p. 110359. DOI: 10.1016/j.jcp.2021.110359.

Appendix A

Kernel Functions ϕ

2-point-width hat function (Beyer & LeVeque, 1992) [114]

$$\phi_1(r) = \begin{cases} 1-|r| & \text{for } |r|<1, \\ \\ 0 & \text{for } |r|\geq 1. \end{cases} \tag{A.1}$$

3-point-width function (Roma et al., 1999) [90]

$$\phi_2(r) = \begin{cases} \frac{1}{3}\left(1+\sqrt{1-3r^2}\right) & \text{for } |r|<0.5, \\ \\ \frac{1}{6}\left(5-3|r|-\sqrt{1-3\left(1-|r|\right)^2}\right) & \text{for } 0.5\leq |r|<1.5, \\ \\ 0 & \text{for } |r|\geq 1.5. \end{cases} \tag{A.2}$$

4-point-width function (Lai & Peskin, 2000) [34]

$$\phi_3(r) = \begin{cases} \frac{1}{8}\left(3-2|r|+\sqrt{1+4|r|-4r^2}\right) & \text{for } |r|<1, \\ \\ \frac{1}{8}\left(5-2|r|-\sqrt{-7+12|r|-4r^2}\right) & \text{for } 1\leq |r|<2, \\ \\ 0 & \text{for } |r|\geq 2. \end{cases} \tag{A.3}$$

4-point-width cosine function (Peskin, 1977) [32]

$$\phi_4(r) = \begin{cases} \frac{1}{4}\left(1+\cos\left(\frac{\pi r}{2}\right)\right) & \text{for } |r|<2, \\ \\ 0 & \text{for } |r|\geq 2. \end{cases} \tag{A.4}$$

Smoothed 2-point-width hat function (Yang et al., 2009) [86]

$$\phi_5(r) = \begin{cases} \frac{3}{4} - r^2 & \text{for } |r| < 0.5, \\ \frac{9}{8} - \frac{3}{2}|r| + \frac{1}{2}r^2 & \text{for } 0.5 \le |r| < 1.5, \\ 0 & \text{for } |r| \ge 1.5. \end{cases} \tag{A.5}$$

Smoothed 3-point-width function (Yang et al., 2009) [86]

$$\phi_6(r) = \begin{cases} \frac{17}{48} + \frac{\sqrt{3}\pi}{108} + \frac{1}{4}|r| - \frac{1}{4}r^2 + \frac{1-2|r|}{16}\sqrt{1+12|r|-12r^2} \\ \quad - \frac{\sqrt{3}}{12}\arcsin\left(\frac{\sqrt{3}}{2}\left(2|r|-1\right)\right) & \text{for } |r| < 1, \\ \frac{55}{48} - \frac{\sqrt{3}\pi}{108} - \frac{13}{12}|r| + \frac{1}{4}r^2 + \frac{2|r|-3}{48}\sqrt{-23+36|r|-12r^2} \\ \quad + \frac{\sqrt{3}}{36}\arcsin\left(\frac{\sqrt{3}}{2}\left(2|r|-3\right)\right) & \text{for } 1 \le |r| < 2, \\ 0 & \text{for } |r| \ge 2. \end{cases} \tag{A.6}$$

Smoothed 4-point-width function (Yang et al., 2009) [86]

$$\phi_7(r) = \begin{cases} \frac{3}{8} + \frac{\pi}{32} - \frac{1}{4}r^2 & \text{for } |r| < 0.5, \\ \frac{1}{4} + \frac{1-|r|}{8}\sqrt{-2+8|r|-4r^2} \\ \quad - \frac{1}{8}\arcsin\left(\sqrt{2}\left(|r|-1\right)\right) & \text{for } 0.5 \le |r| < 1.5, \\ \frac{17}{16} - \frac{\pi}{64} - \frac{3}{4}|r| + \frac{1}{8}r^2 + \frac{|r|-2}{16}\sqrt{-14+16|r|-4r^2} \\ \quad + \frac{1}{16}\arcsin\left(\sqrt{2}\left(|r|-2\right)\right) & \text{for } 1.5 \le |r| < 2.5, \\ 0 & \text{for } |r| \ge 2.5. \end{cases} \tag{A.7}$$

Smoothed 4-point-width cosine function (Yang et al., 2009) [86]

$$\phi_8(r) = \begin{cases} \frac{1}{4\pi}\left(\pi + 2\sin\left(\frac{\pi}{4}(2r+1)\right) - 2\sin\left(\frac{\pi}{4}(2r-1)\right)\right) & \text{for } |r| < 1.5, \\ -\frac{1}{8\pi}\left(-5\pi + 2\pi|r| + 4\sin\left(\frac{\pi}{4}(2|r|-1)\right)\right) & \text{for } 1.5 \le |r| < 2.5, \\ 0 & \text{for } |r| \ge 2.5. \end{cases} \tag{A.8}$$

New 5-point-width function (Bao et al., 2020) [118]

$$r^* = \begin{cases} |r| & \text{for } |r| < 0.5, \\ |r| - 1 & \text{for } 0.5 \le |r| < 1.5, \\ |r| - 2 & \text{for } 1.5 \le |r| < 2.5, \\ 0 & \text{for } |r| \ge 2.5. \end{cases} \qquad \text{where } r^* \in \left[-\frac{1}{2}; \frac{1}{2}\right]$$

$$K = \frac{1}{60}\left(38 - \sqrt{69}\right),$$

$$\beta(r) = 3123 - 6840K + 3600K^2 + 25680Kr^{*2} - 12600K^2r^{*2} - 8400Kr^{*4},$$

$$\gamma(r) = -12440r^{*2} + 8080r^{*4} - 1400r^{*6},$$

$$\Phi(r) = \frac{1}{280}\left(136 - 40K - 40r^{*2} + \sqrt{2\beta(r) + 2\gamma(r)}\right).$$

$$\phi_9(r) = \begin{cases} \Phi(r) & \text{for } |r| < 0.5, \\ \frac{1}{6}\left(-4\Phi(r) + 4 - K - 4r^* + 3Kr^* - r^{*2} + r^{*3}\right) & \text{for } 0.5 \le |r| < 1.5, \\ \frac{1}{12}\left(2\Phi(r) - 2 + 2K + r^* - 3Kr^* + 2r^{*2} - r^{*3}\right) & \text{for } 1.5 \le |r| < 2.5, \\ 0 & \text{for } |r| \ge 2.5. \end{cases} \tag{A.9}$$

New 6-point-width function (Bao et al., 2016) [119]

$$\lceil r \rceil = \min\{n \in \mathbb{Z} \mid n \ge r\},$$

$$r^* = r - \lceil r \rceil + 1, \qquad \text{where } r^* \in [0;1]$$

$$K = \frac{59}{60} - \frac{\sqrt{29}}{20},$$
$$\beta(r) = \frac{9}{4} - \frac{3}{2}K + \frac{22}{3}r^* - 7Kr^* - \frac{3}{2}r^{*2} - \frac{7}{3}r^{*3},$$
$$\gamma(r) = \frac{1}{288}\left(161r^{*2} - 354Kr^{*2} + 180K^2r^{*2} - 109r^{*4} + 120Kr^{*4} + 20r^{*6}\right),$$
$$\Phi(r) = \frac{1}{56}\left(-\beta(r) + \sqrt{\beta^2(r) - 112\gamma(r)}\right).$$

$$\phi_{10}(r) = \begin{cases} \Phi(r) & \text{for } -3 < r \le -2, \\ -3\Phi(r) - \frac{1}{16} + \frac{1}{8}K - \frac{1}{12}r^* + \frac{1}{4}Kr^* + \frac{1}{8}r^{*2} + \frac{1}{12}r^{*3} & \text{for } -2 < r \le -1, \\ 2\Phi(r) + \frac{1}{4} + \frac{2}{3}r^* - \frac{1}{2}Kr^* - \frac{1}{6}r^{*3} & \text{for } -1 < r \le 0, \\ 2\Phi(r) + \frac{5}{8} - \frac{1}{4}K - \frac{1}{4}r^{*2} & \text{for } 0 < r \le 1, \\ -3\Phi(r) + \frac{1}{4} - \frac{2}{3}r^* + \frac{1}{2}Kr^* + \frac{1}{6}r^{*3} & \text{for } 1 < r \le 2, \\ \Phi(r) - \frac{1}{16} + \frac{1}{8}K + \frac{1}{12}r^* - \frac{1}{4}Kr^* + \frac{1}{8}r^{*2} - \frac{1}{12}r^{*3} & \text{for } 2 < r < 3, \\ 0 & \text{for } |r| \ge 3. \end{cases} \tag{A.10}$$

Appendix B

Error Norms and Order of Accuracy

The L_1, L_2, and L_∞ norms of the error for any variable x are defined as:

- The L_1 norm:

$$\|E_x\|_1 = \frac{1}{N_x N_y} \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \left| x_{i,j} - \hat{x}_{i,j} \right|, \tag{B.1}$$

- The L_2 norm:

$$\|E_x\|_2 = \sqrt{\frac{1}{N_x N_y} \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \left| x_{i,j} - \hat{x}_{i,j} \right|^2}, \tag{B.2}$$

- The L_∞ norm:

$$\|E_x\|_\infty = \max_{\substack{1 \le i \le N_x \\ 1 \le j \le N_y}} \left| x_{i,j} - \hat{x}_{i,j} \right|, \tag{B.3}$$

  where $x_{i,j}$ is the value obtained numerically, and $\hat{x}_{i,j}$ is the analytical or refined solution when no exact solution exists. N_x and N_y are the total number of Eulerian grid points in the x and y-directions, respectively.

The order of accuracy of the spatial discretisation, i.e., the convergence rate between numerical solutions obtained with refined grids is:

$$\text{Order} = \log_2 \left(\frac{\|E_x\|^{(h)}}{\|E_x\|^{(h/2)}} \right), \tag{B.4}$$

where $\|E_x\|^{(h)}$ and $\|E_x\|^{(h/2)}$ are the error norms on a grid size h and a refined one $h/2$. The same formula is used to compute the order of accuracy for the temporal discretisation for refined time-steps.

Appendix C

Vertical Axis Turbine (VAT) Reference Data

The exact data extracted from [67] with their corresponding spurious force oscillation peaks for the flow over a 2D VAT at $Re = 100$ are displayed in Fig. (C.1).

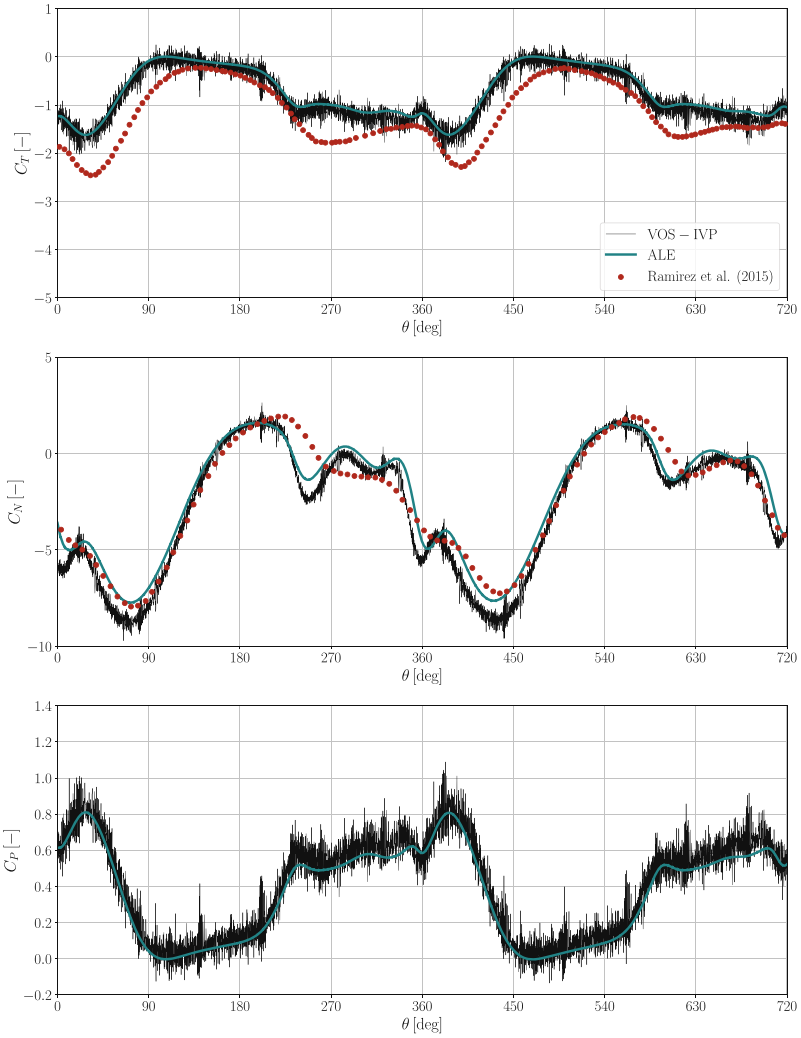


Figure C.1: *Tangential force C_T (top), normal force C_T (middle), and power C_P (bottom) coefficients for the flow over 2D VAT at $Re = 100$, taken from [67].*

Résumé en Français

L'interaction fluide–structure (IFS) désigne l'étude d'un système multiphysique couplé composé de deux entités physiques distinctes : une structure rigide ou déformable en mouvement et un écoulement de fluide environnant. Ce phénomène joue un rôle crucial dans divers domaines scientifiques et technologiques, notamment, mais sans s'y limiter, la sédimentation de particules [1], l'aérodynamique (aéroélasticité) [2], la biofluide et la biomécanique [3], l'hydrodynamique [4], et bien d'autres. Au cœur de ce concept, l'IFS implique une interaction complexe entre une structure solide et le fluide qui l'entoure, chacun influençant l'autre. Par exemple, le mouvement et/ou la déformation des structures solides sont entraînés par les forces exercées par le fluide environnant, tandis qu'inversement, le mouvement et/ou la déformation de la structure impactent la dynamique du fluide. Cette relation réciproque entraîne un couplage bidirectionnel intrinsèquement difficile à résoudre.

Les solutions analytiques sont généralement inaccessibles pour la plupart des problèmes IFS, et les expériences en laboratoire sont souvent limitées dans le champ d'application et coûteuses. Pour surmonter ces défis et approfondir la compréhension des interactions multiphysiques fondamentales entre fluides et solides, les simulations numériques offrent une alternative précieuse. Ces simulations permettent de mieux comprendre les phénomènes sous-jacents tout en réduisant considérablement le besoin d'expériences physiques coûteuses et chronophages. Au cours des dernières décennies, l'émergence d'outils et de ressources informatiques puissants a rendu la simulation numérique une approche de plus en plus précieuse pour l'étude des bases physiques de l'IFS.

En conséquence, le développement de techniques numériques pour simuler l'IFS a suscité une attention considérable au sein de diverses communautés scientifiques. Les simulations d'IFS sont devenues un axe clé de recherche, avec des efforts continus pour améliorer les performances et développer de nouvelles techniques capables de gérer une large gamme de complexités allant de l'adaptation de maillage au couplage de codes de calcul divers, des écoulements laminaires aux flux hautement turbulents, des petits aux grands déplacements structuraux, et des géométries simples aux géométries très complexes.

Dans ce contexte, la présente recherche vise à développer une formulation novatrice pour résoudre des problèmes IFS à usage général, en mettant l'accent sur l'interaction entre des écoulements de fluide incompressibles et des corps solides complexes en mouvement libre et de grande amplitude. Cette recherche vise à relever les défis associés aux simulations IFS, en fournissant des solutions plus efficaces et pratiques tant pour les applications académiques

qu'industrielles. Un aspect clé de ce travail est l'adoption d'une méthode de maillage non conforme au corps, qui évite les défis informatiques associés à une génération de maillage complexe pendant le prétraitement, ainsi que les problèmes de maillage qui surviennent lors de la résolution IFS, en particulier lorsque de grands mouvements structurels se produisent.

L'objectif de cette recherche est de réduire le temps nécessaire pour les étapes de prétraitement et de traitement associées aux simulations numériques traditionnelles en adoptant et en améliorant une méthode existante non conforme au corps; la méthode des frontières immergées (IBM). Cette approche permet à des structures rigides complexes de se déplacer librement et de manière arbitraire au sein d'un fluide visqueux incompressible sans nécessiter la génération de maillages adaptés au corps. En éliminant le besoin de techniques dynamiques et/ou de remeshing sophistiquées, qui sont connues pour être intensives en calcul, l'IBM simplifie la génération de maillage à une tâche triviale nécessitant seulement quelques secondes de temps de calcul. Cette recherche vise à rationaliser les flux de travail de simulation et à améliorer l'efficacité globale du processus de simulation numérique.

Les objectifs pour atteindre le but mentionné ci-dessus sont les suivants :

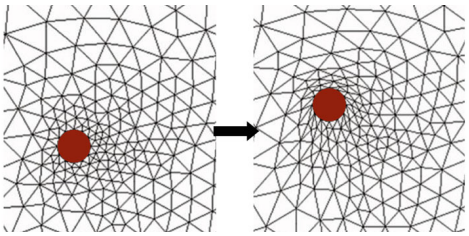
- Développer un solveur IBM par forçage direct implicite (DF-IBM) précis et efficace en améliorant les itérations implicites pour appliquer avec précision les conditions cinématiques sur la frontière immergée.

- Intégrer le DF-IBM dans le solveur de fluide 2D incompressible Navier-Stokes au sein du code open-source de dynamique des fluides numérique (CFD) OpenFOAM, en s'appuyant sur l'algorithme "Pressure Implicit with Splitting of Operators" (PISO) pour satisfaire les nouvelles contraintes du système fluide–solide couplé.

- Implémenter les équations de mouvement de Newton-Euler pour calculer les cinétiques et cinématiques d'un corps rigide en mouvement libre.

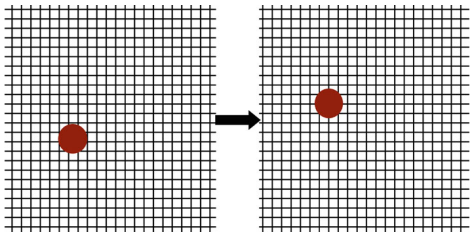
- Coupler les équations de Navier-Stokes incompressibles avec la dynamique des corps rigides dans le cadre DF-IBM pour étudier les problèmes d'interaction fluide–corps rigide faiblement et fortement couplés.

Ce travail de thèse a passé en revue les méthodologies de simulation des problèmes d'interaction fluide–structure (IFS), en se concentrant sur les approches conformes et non conformes au corps. L'approche conforme au corps (voir la figure ci-dessous), illustrée par les méthodes ALE (Arbitrary Lagrangian–Eulerian) et Chimère (Overset), repose sur des maillages qui s'alignent avec l'interface fluide–structure.



Bien que ces méthodes permettent une représentation précise des frontières, elles présentent des défis notables. La méthode ALE rencontre des difficultés à gérer de grands mouvements structurels en raison de la distorsion du maillage causée par les mouvements dynamiques de celui-ci et les coûts informatiques liés au remeshing. De même, la méthode Chimère, bien qu'offrant une flexibilité dans la gestion des grilles chevauchantes, elle rencontre des problèmes d'interpolation des données entre les blocs et dépend toujours des stratégies de maillage conformes au corps, ce qui complique la modélisation des géométries complexes. Malgré ces avancées, la complexité et les limites des méthodes conformes au corps soulignent le besoin d'approches alternatives.

En revanche, les méthodes non conformes au corps (voir la figure ci-dessous), telle que la méthode des frontières immergées (IBM), offrent un cadre plus flexible et plus efficace sur le plan computationnel et ont gagné en popularité au cours des deux dernières décennies.



En découplant les grilles du fluide et du solide, les approches non conformes éliminent le besoin de mises à jour dynamiques des maillages, simplifiant ainsi la simulation des frontières mobiles. L'IBM, initialement développée par Peskin, a introduit un système à deux grilles comprenant des domaines Eulériens et Lagrangiens, facilitant le couplage efficace de la dynamique des fluides et des structures grâce à des opérateurs d'interpolation et de diffusion. De nombreuses variantes de l'IBM ont depuis été développées pour répondre à des défis spécifiques en ingénierie et technologie. Cependant, des défis importants subsistent, le plus notable étant la difficulté à résoudre avec précision les couches limites dans les flux turbulents à nombre de Reynolds élevé. Cette thèse a mis en lumière plusieurs études visant à relever ces défis.

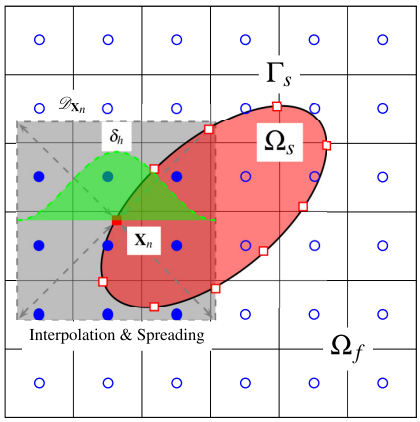
Une attention particulière a été accordée à l'IBM par forçage direct (DF-IBM) en raison

de son adoption généralisée, de sa stabilité et de sa facilité d'intégration dans les solveurs de fluides. Des variantes telles que le DF-IBM explicite, le multi-DF-IBM et les approches implicites de l'IBM, y compris l'IBPM et le MIBM, ont été discutées. Les stratégies visant à améliorer la réciprocité des opérateurs d'interpolation et de diffusion ont également été examinées. Ces avancées démontrent la polyvalence de l'IBM et sa capacité à surmonter les limites inhérentes aux méthodes conformes au corps, en faisant un choix robuste pour simuler des scénarios IFS complexes. Les défis du cadre DF-IBM, tels que les oscillations de forces parasites et l'interface diffusée de la frontière immergée, ont été abordés, ainsi que des solutions proposées pour atténuer ces problèmes.

Cette thèse se conclue par une vue d'ensemble du couplage fluide–structure dans le cadre DF-IBM, couvrant à la fois les mouvements prescrits et les mouvements libres du fluide. Ces scénarios nécessitent une correspondance cinématique et dynamique à l'interface fluide–solide. Des stratégies de couplage faible et fort ont été discutées, mettant en évidence les compromis entre efficacité computationnelle et précision.

Une attention particulière a été également portée à l'effet de masse ajoutée, qui est particulièrement critique pour des rapports de densité solide-fluide faibles. Les défis posés par l'effet de masse ajoutée, notamment son impact sur la stabilité et la convergence des algorithmes de couplage, ont été examinés. En outre, diverses études proposant des méthodes pour atténuer les singularités et surmonter les limitations liées aux rapports de densité ont été mises en avant. Ces avancées renforcent la stabilité pour les dynamiques rotationnelles et translationnelles, consolidant davantage la robustesse de l'IBM pour les simulations d'IFS.

Le solveur fluide–solide utilisé dans cette recherche a été implémenté dans l'environnement OpenFOAM. Dans le contexte de la méthode de frontière immergée (IBM), les équations gouvernantes du fluide formulées dans un cadre Eulérien sont couplées avec la frontière immergée du solide représentée par une formulation Lagrangienne (voir la figure ci-dessous).



Ce couplage est réalisé en introduisant une force de frontière comme terme source dans les équations de Navier-Stokes. Cette force mime efficacement la présence de la frontière immergée solide dans le domaine du fluide. Pour les problèmes de flux incompressibles, le système fluide–solide couplé impose une double contrainte qui doit être appliquée séparément dans leurs formulations respectives. La première contrainte garantit un champ de vitesse sans divergence dans le domaine Eulérien, tandis que la seconde impose la condition cinématique de vitesse sans glissement dans le domaine Lagrangien. Chaque terme du système résultant est discrétisé à l'aide d'une méthode des volumes finis centrée sur les cellules avec un schéma de discrétisation d'ordre deux, assurant une précision globale d'ordre deux en espace et en temps. Étant donné la disposition des grilles colocalisées dans OpenFOAM, la technique de correction de Rhie-Chow a été appliquée pour éviter les oscillations "checkerboard" de la pression.

Une vue d'ensemble détaillée est fournie sur le problème de couplage pression-vitesse, ainsi que sur les stratégies présentes dans la littérature pour aborder ce système couplé. Dans cette recherche, la stratégie prédicteur-correcteur de l'algorithme PISO a été sélectionnée comme méthode de découplage, et une dérivation approfondie de ses étapes de prédiction et de correction a été démontrée. De plus, une équation de pression modifiée a été formulée en incorporant la divergence du terme de force frontière. Enfin, les solveurs itératifs disponibles dans OpenFOAM pour résoudre le système d'équations algébriques linéaires découlant des équations de quantité de mouvement et de pression sont présentés.

La formulation mathématique générale de la méthode DF-IBM inclut les équations de conservation de la masse et de la quantité de mouvement dans le domaine Eulérien du fluide, couplées au domaine Lagrangien du solide par l'ajout d'un terme source dans les équations de Navier-Stokes, appelé terme de force frontière, ainsi que deux ensembles supplémentaires d'équations connus sous le nom d'opérateurs linéaires liés à l'IBM. Ces opérateurs comprennent un opérateur d'interpolation de vitesse et un opérateur de diffusion de force, qui gèrent le transfert de données entre les domaines Lagrangien et Eulérien à l'aide d'une fonction d'interpolation : la fonction delta de Dirac discrète. Cette fonction d'interpolation est construite à partir de fonctions continues 1D, appelées fonctions noyaux (Kernel functions), dont dix ont été introduites, chacune conçue pour satisfaire des propriétés spécifiques et des conditions de moment. Le processus de transfert bidirectionnel des données se déroule dans le domaine de support Lagrangien, et un algorithme a été introduit pour construire ce domaine de support.

Les opérateurs linéaires liés à l'IBM conduisent à deux identités fondamentales

Eulériennes–Lagrangiennes : les identités de force et de couple, garantissant que les valeurs totales de force ou de couple soient identiques, qu'elles soient calculées dans le domaine Eulérien ou Lagrangien. Les étapes générales de l'algorithme DF-IBM, ainsi que les approches numériques et les algorithmes de découplage pression-vitesse mis en œuvre dans le cadre du DF-IBM ont été examinées.

Une explication détaillée de la condition limite de vitesse sans glissement, qualifiée de deuxième contrainte du système fluide–solide, a été fournie. Cette contrainte repose directement sur la réciprocité entre les opérateurs d'interpolation et de diffusion ; toute violation de cette réciprocité entraîne une erreur de frontière de vitesse sans glissement, permettant au fluide de pénétrer dans la frontière solide immergée. Les causes sous-jacentes de cette erreur de frontière ont été illustrées à travers la variante initiale explicite du DF-IBM proposée par Uhlmann [39]. Les approches pour atténuer cette erreur ont été examinées, notamment le concept de multi-DF-IBM développé par [41, 77, 78], qui a conduit à diverses variantes explicites et implicites d'IBM. Une stratégie alternative de réduction des erreurs consiste à calibrer le poids Lagrangien utilisé dans l'opérateur de diffusion de force en adoptant les méthodes de la littérature [82, 83, 85, 124], qui améliorent la réciprocité des opérateurs linéaires liés à l'IBM.

Ce travail propose un algorithme unifié de méthode de frontière immergée à forçage direct (DF-IBM) qui combine des techniques IBM implicites avec un poids Lagrangien calibré, ce qui améliore la réciprocité des opérateurs d'interpolation de vitesse et de propagation de force liés à l'IBM. Cela permet d'appliquer de manière précise et efficace la deuxième contrainte fluide–solide. La pression, la vitesse et la force de frontière sont découplées des équations gouvernantes du fluide en utilisant la stratégie de séparation des opérateurs de l'algorithme de solution PISO, implémentée dans l'environnement OpenFOAM. L'algorithme DF-IBM amélioré bénéficie du cadre prédicteur-correcteur de l'algorithme PISO pour appliquer séquentiellement le problème de la double contrainte du système fluide–solide à chaque pas de temps pendant le processus de solution. Cette formulation conduit à une équation de pression modifiée, caractérisée par l'incorporation de la force de frontière qui satisfait déjà la deuxième contrainte fluide–solide sur la frontière immergée. Par conséquence, le rôle restant de l'équation de pression est de garantir que le champ de vitesse Eulérien maintienne une condition sans divergence. Cette approche préserve une matrice de coefficients de pression symétriques et évite d'augmenter son nombre de condition, ce qui la rend comparable à l'équation de pression standard. Cette modification permet l'utilisation de solveurs itératifs rapides pour l'équation de pression, tel que le solveur de CG (Conjugate Gradient) en com-

binaison avec le préconditionneur GAMG (Geometric Agglomerated Algebraic Multi-Grid), sans nécessiter d'ajustements supplémentaires dans OpenFOAM. Un schéma supplémentaire d'initialisation de force Lagrangienne est proposé pour améliorer davantage les performances de l'algorithme. Aucune restriction supplémentaire n'est imposée au pas de temps, hormis le critère de stabilité du nombre CFL établi par l'algorithme PISO ($CFL < 1$). L'algorithme global comprend trois étapes principales : le prédicteur de quantité de mouvement, les itérations DF-IBM et les boucles correctrices PISO.

Le DF-IBM implicite proposé a été validé numériquement. Une analyse approfondie du développement actuel en termes de précision a montré que cette méthode maintient une précision de second ordre en temps (voir Fig. (5.2a)) et de second ordre en espace (voir Fig. (5.2b) lorsqu'une fonction noyau d'ordre deux est utilisée pour correspondre à la discrétisation d'ordre deux de tous les termes dans les équations de Navier-Stokes. Cependant, cet ordre de précision n'est valable que pour les champs lisses, comme démontré dans le problème des tourbillons déclinants de Taylor–Green. Néanmoins, une solution d'ordre deux n'est pas garantie pour les champs non lisses en raison de la fonction delta de Dirac discrète.

La performance et l'efficacité globale de l'algorithme se sont révélées supérieures à celles des algorithmes DF-IBM précédents (voir le tableau 5.4). Par exemple, la méthode développé applique exactement la deuxième contrainte du système fluide–solide dans le domaine Lagrangien avec un effort computationnel significativement moindre par rapport à l'IBM implicite de Ji et al. [80]. Plus précisément, il réduit le nombre d'itérations implicites de près de 60%, avec seulement une modeste augmentation du temps de calcul par un facteur de 35 par rapport à l'IBM explicite d'Uhlmann [39], contrairement à la variante implicite de Ji et al., qui a montré une augmentation par un facteur de 91. Il s'est avéré que l'utilisation d'un nombre pair de marqueurs Lagrangiens pour la représentation de la frontière immergée, plutôt que d'un nombre impair, réduit considérablement le nombre d'itérations implicites ce qui accélère systématiquement le taux de convergence de l'algorithme (voir Fig. (5.3)). La sensibilité de la deuxième contrainte du système fluide–solide au schéma d'initialisation de force Lagrangienne, au poids Lagrangien, à la correction analytique du poids Lagrangien et au pas de temps de simulation est également démontrée.

De plus, la capacité de la méthode actuelle à traiter à la fois des frontières solides immergées circulaires et non circulaires a été confirmée. Son efficacité à capturer la physique et la dynamique des écoulements pour des mouvements stationnaires et prescrits (translationnels et rotationnels) à travers différents nombres de Reynolds a été validée par des comparaisons avec des données expérimentales. Les résultats se sont avérés favorables par rapport

à d'autres variantes de l'IBM et aux méthodes conformes au corps issues de la littérature, montrant une excellente concordance dans tous les cas simulés.

La modélisation numérique des interactions fluide–corps rigide dans le cadre du DF-IBM a été abordée. Les équations générales gouvernant le corps solide en trois dimensions ont été introduites initialement sur la base des équations du mouvement de Newton-Euler pour la dynamique des corps rigides. Une forme réduite applicable au mouvement 2D a ensuite été présentée, ce qui a donné un système à 3 degrés de liberté capturant les dynamiques translatoires et rotatoires arbitraires sous l'influence de forces et de couples externes. Les équations cinématiques de ce système ont été énumérées pour déterminer la position, l'orientation et la vitesse absolue des marqueurs Lagrangiens situés à la surface du corps rigide, c'est-à-dire la frontière immergée rigide. Une vue d'ensemble complète des forces et des couples externes agissant sur le corps rigide a été fournie. Une attention particulière a été accordée à la force et au couple hydrodynamiques résultant de l'interaction fluide–corps rigide à leur interface commune.

Étant donné que le calcul direct de ces forces hydrodynamiques est numériquement exigeant, le principe de Cauchy [152] a été utilisé comme une alternative efficace, suivant l'approche proposée par Uhlmann [151]. Cette méthode utilise les forces de volume déjà calculées dans le cadre du DF-IBM pour évaluer de manière précise et efficace ces forces hydrodynamiques. De plus, l'effet de masse ajoutée a été examiné. Ce phénomène provient de la force et du couple hydrodynamiques associés au fluide interne enfermé à l'intérieur de la frontière immergée du corps rigide en raison de la formulation IBM. La force et le couple générés par le fluide interne sont équivalents au taux de moment linéaire et angulaire du corps rigide, assurant ainsi la cohérence avec la formulation IBM [151].

La formulation mathématique générale pour le couplage des équations de Navier-Stokes et de la dynamique des corps rigides dans le cadre du DF-IBM est présentée. Cette formulation inclut les équations gouvernant le fluide, les équations gouvernant le solide, la condition d'interface fluide–solide, ainsi que les opérateurs liés à l'IBM. La procédure de discrétisation de ce système couplé est détaillée, y compris cinq schémas de discrétisation différents pour l'effet de masse ajoutée et le processus de mise à jour de la position et de l'orientation du solide à chaque incrément de temps. Parmi ces schémas, l'approche explicite a été choisie. Elle permet de traiter l'effet de masse ajoutée en raison de sa simplicité d'intégration et de sa capacité à contourner les limitations sur le ratio de densité solide–fluide qui apparaissent lorsque le schéma implicite est utilisé avec les équations de Newton-Euler.

Le couplage fluide–corps rigide implique l'interaction entre deux sous-problèmes dis-

tincts : l'un gouverne la dynamique du fluide et l'autre gouverne la dynamique du corps rigide. Une approche de couplage partitionné est privilégiée, où chaque sous-problème est résolu à l'aide d'un solveur dédié. Cette approche offre modularité et flexibilité, permettant le développement et l'optimisation indépendants de chaque solveur. La condition d'interface, où les données sont échangées entre les solveurs fluide et solide, peut être traitée en utilisant deux stratégies : unidirectionnelle ou bidirectionnelle. Deux algorithmes distincts de couplage FSI ont été proposés : un algorithme explicite avec un couplage faible de la condition d'interface et un algorithme implicite avec un couplage fort de la condition d'interface.

Pour l'algorithme implicite, la stratégie du point fixe est utilisée en raison de sa simplicité, avec une relaxation fixe adoptée comme technique d'accélération. Cet algorithme a permis d'obtenir des résultats suffisants et des calculs stabilisés (convergents) dans les cas où l'effet de masse ajoutée introduisait des difficultés, notamment lorsque les densités du fluide et des corps rigides sont identiques et pour les solides flottants. Les problèmes rencontrés avec l'approximation du corps rigide de l'effet de masse ajoutée dans le cas de dynamiques de rotation ont été simultanément résolus grâce à la technique de relaxation fixe. De même, dans l'algorithme explicite, les instabilités dues aux solides flottants et neutres ont été atténuées, et les difficultés associées à l'effet de masse ajoutée lors du mouvement de rotation ont été efficacement allégées à l'aide de la même technique de relaxation.

L'algorithme de couplage implicite proposé se distingue par son efficacité par rapport aux algorithmes traditionnels utilisés dans la formulation ALE. Cette approche bénéficie de la stratégie prédicteur-correcteur PISO et de l'algorithme DF-IBM implicite développé, en excluant l'étape prédicteur de quantité de mouvement et les boucles correctrices PISO des itérations de couplage implicite fluide–corps rigide. A savoir, les boucles correctrices de PISO sont considérées l'étape la plus chronophage de l'algorithme de solution PISO. Enfin, en raison de la différence substantielle de taille entre les domaines Eulériens et Lagrangiens, les mises à jour de maillage Lagrangien nécessaires dans l'algorithme de couplage implicite sont nettement moins exigeantes en termes de calcul que les mises à jour de maillage Eulérien requises dans la formulation ALE, ce qui améliore encore son efficacité globale.

Les algorithmes de couplage fluide–corps rigide explicite et implicite proposés ont été validés numériquement à travers une série de cas tests. La validation a commencé par des simulations de mouvements purement translationnels et purement rotationnels, suivis de mouvements libres arbitraires couplés et complexes résultant des interactions entre un fluide visqueux incompressible et un corps rigide. Le développement actuel a efficacement pris en charge une large gamme de ratios de densité, allant des corps rigides non flottants aux corps

flottants, y compris le cas critique des corps neutres en flottabilité, pour lesquels des calculs stables et convergents ont été obtenus.

L'étude a couvert diverses géométries, y compris des disques circulaires, des ellipses et des profils aérodynamiques NACA, démontrant la polyvalence de l'approche. L'importance d'intégrer l'effet de masse ajoutée a été mise en évidence, en particulier pour les cas où le nombre de Reynolds du solide dépassait une valeur de un. Les instabilités résultant de l'approximation du corps rigide de l'effet de masse ajoutée pendant les dynamiques de rotation ont été identifiées et traitées. Une technique de relaxation fixe s'est avérée être une solution efficace pour atténuer ces instabilités, permettant ainsi la convergence de l'algorithme de couplage implicite pour des cas spécifiques difficiles.

Il a été observé que le traitement explicite de l'effet de masse ajoutée introduisait un décalage temporel dans les quantités de dynamique rotationnelle, particulièrement visible lorsque le nombre de Reynolds de rotation du solide dépassait 10. Malgré ce décalage de phase, la physique fondamentale des écoulements et la dynamique du corps rigide ont été préservées. Les analyses de sensibilité du couplage implicite à la taille du pas de temps et à la résolution du maillage ont révélé qu'en réduisant le pas de temps ou en affinant le maillage, le taux de convergence des itérations implicites s'améliore. Notamment, une grille plus grossière amplifiait le décalage de phase dans les quantités de rotation lorsque le nombre de Reynolds de rotation du solide dépasse 10, comparé à des grilles plus fines. De plus, l'impact du changement des pas de temps est plus faible sur le décalage de phase que l'impact du changement du maillage.

Dans l'ensemble, les algorithmes de couplage proposés ont montré un excellent accord avec les données de référence provenant de la littérature pour tous les cas tests, validant leur capacité et leur robustesse pour une large gamme de problèmes d'interaction fluide–corps rigide complexes.