Technische Universität München

Professur für Hydromechanik

Implementation of an Immersed Boundary Method for a fourth-order Finite Volume Scheme

Master thesis
2019

Khaled Boulbrachene

Submitted to the Faculty of Civil, Geo and Environmental Engineering in partial fulfillment of the requirements for the degree of

Master of Science

at the Technical University of Munich.

Supervisor:	Lukas Unglehrt, M.Sc.	Professur für Hydromechanik
Prüfer der Arbeit:	Univ.-Prof. Dr.-Ing. habil. Michael Manhart	

Acknowledgment

I would like to use this chance to express my sincere gratitude to my supervisor, Lukas Unglehrt, for his invaluable assistance and mentoring throughout this work. He was always willing to share his expertise, inspiring ideas and insightful suggestions to guide me through the obstacles in my research. I have truly enjoyed the moments discussing numerical methods on the whiteboard or by the desk in his office. This work would not have been possible without his continuous support. I would also like to convey my special regards to Prof. Dr.-Ing. Michael Manhart for giving me the opportunity to pursue my Master thesis at the chair of Hydromechanics at the Technical University of Munich.

Some special words of gratitude go to my friends, Yaseen Hago and Sulaiman Fadlallah who have always been a major source of moral support, joy and happiness. My thanks extend to my colleagues from the Computational Mechanics Masters program, especially, Mahmoud Zidan, Mahmoud Ammar and Rahul Krishna for making my life in Munich a wonderful experience.

There are no words that could possibly express an appreciation to my parents, Dr. Messaoud Boulbrachene and M.D. Fouzia Saadni. Their love, encouragement and never ending support shaped me and provided me strength to achieve my goals and come this far.

Abstract

First, a second-order and fourth-order discretization schemes for the incompressible Navier-Stokes equations on staggered grids are implemented. Two spatial dimensions are considered and the discretizations are carried out for Cartesian uniform and non-uniform grids. Implementations are then verified and the order of accuracy of the adopted spatial and temporal discretization schemes is validated. Next, an Immersed Boundary method (IBM) with a direct forcing strategy is integrated with both flow solvers to simulate fluid-solid interaction problems. The continuous IB method employs a smoothed approximation of the Dirac delta function with a specific order of accuracy to smear (regularize) the immersed boundary forces over the adjacent fluid cells. The accuracy of the solver then becomes dependent on the order at which the coupling quantities are interpolated. In the literature, there exist different second-order regularized delta-function variants which are well-suited to use with the second-order finite volume solver. However, an appropriate fourth-order regularized delta function is not available and its detailed derivation is presented. At last, several numerical tests with smooth and non-smooth velocity fields are conducted to observe the rate of convergence achieved of both IBM solvers.

Contents

List of Figures

Nomenclature

$\alpha_1, \alpha_2, \alpha_3$	Filter coefficients for a fourth-order convected velocity
$\alpha_4, \alpha_5, \alpha_6$	Filter coefficients for a fourth-order first derivative
$\beta_1,\ \beta_2,\ \beta_3$	Coefficients of the cell-centered deconvolution for momentum cell
$\beta_4,\ \beta_5,\ \beta_6$	Coefficients of the cell-centered deconvolution for pressure cell
δ_h	Regularized delta function
Δt	Time step
$\Delta x, \Delta y, \Delta z$	Grid spacing in x, y, z directions
$\Delta x_{is}, \Delta y_{js}$	Staggered grid spacing in $x,\ y$ directions
$\lambda_{i,1-4}$	Coefficients of the divergence-free convective velocity aligned with the momentum cell
$\mathbf{C},\ \mathbf{F}$	Convective and diffusive coefficients matrices
$\mathbf{D}$	Discrete approximation operator of the discrete divergence
$\mathbf{D}_2$	Second-order approximation of the discrete divergence
$\mathbf{D}_4$	Fourth-order approximation of the discrete divergence
$\mathbf{F}$	Lagrangian force
$\mathbf{f}$	Eulerian force
$\mathbf{F}^c,\ \mathbf{F}^p,\ \mathbf{F}^d$	Sum of convective, pressure and diffusive fluxes
$\mathbf{G}$	Discrete approximation operator of the discrete gradient
$\mathbf{G}_2$	Second-order approximation of the discrete gradient
$\mathbf{G}_4$	Fourth-order approximation of the discrete gradient
$\mathbf{L}_2$	Discrete Laplace operator of the second-order projection method
$\mathbf{L}_4$	Discrete Laplace operator of the fourth-order projection method

$\mathbf{RHS}$	Discrete sum of the convective, pressure and diffusive fluxes at Lagrangian positions
$\mathbf{rhs}$	Discrete sum of the convective, pressure and diffusive fluxes at Eulerian positions
$\mathbf{u}$	Velocity vector
$\mathbf{U}*$	Intermediate velocity vector at Lagrangian positions
$\mathbf{u}^*$	Intermediate velocity vector at Eulerian points
$\mathbf{X}$	Lagrangian force point coordinates
$\mathbf{x}$	Eulerian grid point coordinates
$\mathsf{u},\ \mathsf{v}$	Convective velocity in $x,\ y$ directions
ν	Kinematic viscosity
Ω	Computational domain
$\Omega_i,\ \Omega S_{is},\ \Omega S_{js}$	Pressure, u-momentum and v-momentum cells control volumes
$\phi(r)$	Regularized delta function kernel
ϕ^{xy}	Cell-averaged quantity
ϕ^y, ϕ^x	Vertical edge and horizontal edge averaged quantity
ρ	Density
$\theta_{is,1-4}$	Coefficients of the divergence-free convective velocity normal to the momentum cell
$\tilde{M}$	Modified discrete moment condition
h	mesh width
M	Discrete moment condition
N_L	Number of Lagrangian force points on the immersed body
q	Number of moment conditions
s'_x, s'_y	Refined grid constant stretching factors along x and y-coordinates
s_x, s_y	Constant stretching factors along x and y-coordinates
T	Computational time
u, v	Convected velocity in $x,\ y$ directions
x_{is}, y_{js}	Staggered coordinates in $x,\ y$ directions

1. Introduction

1.1. Motivation

Fluid flows can be described by Partial Differential Equations (PDE) which are quite complex in nature. In few particular cases only, these equations can be simplified and solved analytically. Therefore, Computational Fluid Dynamics (CFD) is used for the purpose of generating approximate solutions to fluid flows by employing numerical schemes replacing the system of partial differential equations by a system of algebraic equations which can then be solved on a computer. This process includes subdividing the domain of interest in space and time, hence providing solutions at discrete locations in space and time. Direct Numerical Simulation (DNS) serves as an accurate tool for solving laminar as well as turbulent fluid flows. However, the method becomes extremely expensive in the latter case as the whole range of spatial and temporal scales must be resolved. Mainly, two factors affect the quality of the numerical solution obtained, namely, grid resolution and the accuracy of the numerical scheme used. The use of higher-order schemes shows its great advantage in three-dimensional simulations. While its cost is linearly proportional to second-order scheme, a cubic reduction of the number of grid points is granted [Hokpunna, 2009]. In this work, we choose to use the Finite Volume Method (FVM) as a discretization technique for the governing equations. This method gained its great popularity due to its built-in local as well as global conservation properties.

In engineering applications of CFD, it is very common to face fluid-solid interaction problems. These problems arise due to the existence of an interface between any two materials with different physical properties. A broad range of examples include fluid flow around an automotive engine, blood flow inside the beating heart or flows around flapping wings. The analysis of unsteady fluid flows evolving inside or around bodies of often complex geometries poses a great challenge to numerical simulations. Solving such problems with a technique employing a pre-defined time invariant grid can be an attractive option, as the generation of the difficult time consuming boundary fitted mesh can be avoided. An example of such technique is the immersed boundary method. Studying the order of accuracy of this method has become an active field of research recently. However, this needs further analytical study and numerical testing for higher-order schemes.

2. Background

This chapter defines the Navier-Stokes equations we intend to solve numerically for incompressible flows of a Newtonian fluid by the finite volume method. We then introduce the indicial notation followed in this work for staggered grid systems. Next, we present the projection method adopted to ensure incompressibility as we advance the solution in time. Then, the time integration scheme employed in this work is presented by means of a pseudo-code and we finally close this chapter by a theoretical introduction to the Immersed Boundary Method.

2.1. Navier-Stokes equations

Navier-Stokes equations are the governing equations of Fluid Dynamics. Theoretically, these equations arise from the conservation of mass and the conservation of momentum. In the scope of this work, we confine ourselves to viscous incompressible fluids which can be described by the incompressible Navier-Stokes equations:

$$\left.\begin{aligned} \frac{\partial \mathbf{u}}{\partial t} &= (\mathbf{u}\cdot\nabla)\,\mathbf{u} + \nu\Delta\mathbf{u} - \tfrac{1}{\rho}\nabla p + \mathbf{f} \\ \nabla\cdot\mathbf{u} &= 0 \end{aligned}\right\} \quad \text{in } \Omega\times[0,\ T]. \tag{2.1}$$

Here, the variables $\mathbf{u} = \mathbf{u}(\mathbf{x},t)$ and $p = p(\mathbf{x},t)$ denote the velocity vector and the pressure of the fluid at some point $\mathbf{x}$ in the domain Ω at a time t. The kinematic viscosity ν and the density ρ of the fluid occupying the domain are constants. For two-dimensional flows, we can write $\mathbf{u} = (u,\ v) = (u(x,y,t),\ v(x,y,t))$ and $p = p(x,y,t)$. These equations relate the rate of local time change of the velocity of a fluid to the summation of forces acting upon it. The first term on the right hand side is known as the convective term and it represents the inertia forces acting on the fluid. The second term is known as the diffusive term and it is responsible for the dissipation of energy in a fluid flow. The third term which is called the pressure term accounts for the pressure forces developing in the case of a present pressure gradient. The last term $\mathbf{f} = (f_x(x,y,t),\ f_y(x,y,t))$ represents the density of the body forces acting upon the fluid.

More details related to the physical background and the derivation of these equations can be found in [Batchelor, 2000]. This set of non-linear partial differential equations can sometimes be simplified to obtain the exact solution describing the flow profile. However, in the majority of the practical cases, the analytical solution becomes intractable and the solution must be approximated numerically.

In this work, we wish to use the finite volume method (FVM) to evaluate the partial differential equation (2.1) in the form of algebraic equations. The initial step in the finite volume discretization process is to integrate the governing equations over the finite volumes (Ω) into which the domain has been partitioned. The integral form of the conservation law (2.1) reads:

$$\frac{\partial}{\partial t}\int_\Omega u_i\ dV = -\int_\Omega \frac{\partial u_i u_j}{\partial x_i}\ dV - \int_\Omega \frac{1}{\rho}\frac{\partial p}{\partial x_i}\ dV + \int_\Omega \nu\frac{\partial^2 u_i}{\partial x_j^2}\ dV,$$

$$\int_\Omega \frac{\partial u_i}{\partial x_i}\ dV = 0.$$

Gauss theorem is then employed to convert volume integrals to surface integrals over the boundaries of the finite volume (Γ):

$$\frac{\partial}{\partial t}\int_\Omega u_i\ dV = -\underbrace{\int_\Gamma u_i u_j n_j\ dA}_{\text{Convective flux } F^c} - \underbrace{\int_\Gamma \frac{1}{\rho} p n_i\ dA}_{\text{Pressure flux } F^p} + \underbrace{\int_\Gamma \nu\frac{\partial u_i}{\partial x_j} n_j\ dA}_{\text{Diffusive flux } F^d}, \tag{2.2}$$

$$\int_\Gamma u_i\ dA = 0. \tag{2.3}$$

In the two-dimensional context, equation (2.2) states that the rate of change of a cell-averaged quantity u_i within the area Ω is equal to the net flux across its boundary Γ. In order to solve these equations numerically, one has to approximate the area and boundary integrals as well as the fluxes defined in equations (2.2) and (2.3).

2.2. Indicial notation and staggered grid generation

When we discretize the Navier-Stokes equations on Cartesian grids, we have the freedom to arrange the primary variables on the grid. Colocated and staggered [Harlow and Welch, 1965] arrangements represent the main two choices of primary variables arrangement on the computational grid. In colocated grids, all unknown dependent variables ($u,\ v$ and p) are stored at one position sharing the same control volume as shown in Fig 2.1a. An immediate consequence of this arrangement is a minimized number of coefficients to be computed when the governing equations are discretized and a simpler implementation in complicated solution domains. In contrast to colocated arrangement, staggered grids have the primary variables situated at different locations each having its own control volume as shown in Fig 2.1b. Staggered grids are known for their biggest advantage of the strong coupling between the pressure and velocities, which in turn eliminates the odd-even decoupling leading to the checkerboard pressure problem. Colocated grids checkerboard problem is not restricted to low order discretization schemes only and can be observed on higher order schemes when an even number of cells is used [Pereira et al., 2001]. In order to avoid additional treatment of pressure in colocated grid setting, staggered grids were used in this work. While on uniform two-dimensional Cartesian grids pressure cell edges lie midway between the cell centroids, on non-uniform grids only one of these two properties can be satisfied. Here, the former one was chosen and one can now define the pressure points to be $(x_i,\ y_j)$ with $i =\ 1,\ \ldots,\ N_x$ and $j =\ 1,\ \ldots,\ N_y$. On the other hand, the staggered u and v-velocity points are defined as $(x_{is},\ y_j)$ and $(x_i,\ y_{js})$, respectively, where $x_{is} = \frac{1}{2}\left(x_i + x_{i+1}\right)$ and $y_{js} = \frac{1}{2}\left(y_i + y_{i+1}\right)$. This grid definition can be shown in Fig 2.2. The control volume of the pressure, u and v-momentum cells can now be defined as $\Omega_{i,j} = [x_{is-1},\ x_{is}] \times [y_{js-1},\ y_{js}]$,

$\Omega S_{is,j} = [x_i,\ x_{i+1}] \times [y_{js-1},\ y_{js}]$, and $\Omega S_{i,js} = [x_{is-1},\ x_{is}] \times [y_j,\ y_{j+1}]$, respectively.

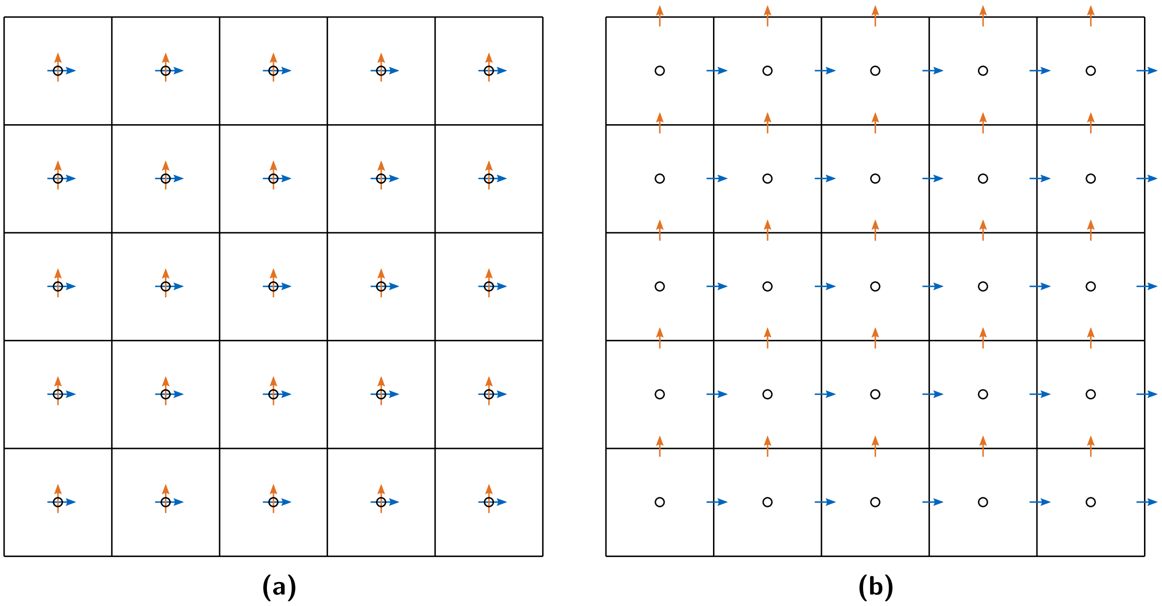


Figure 2.1: (a) Colocated grid setting. (b) Staggered grid setting.

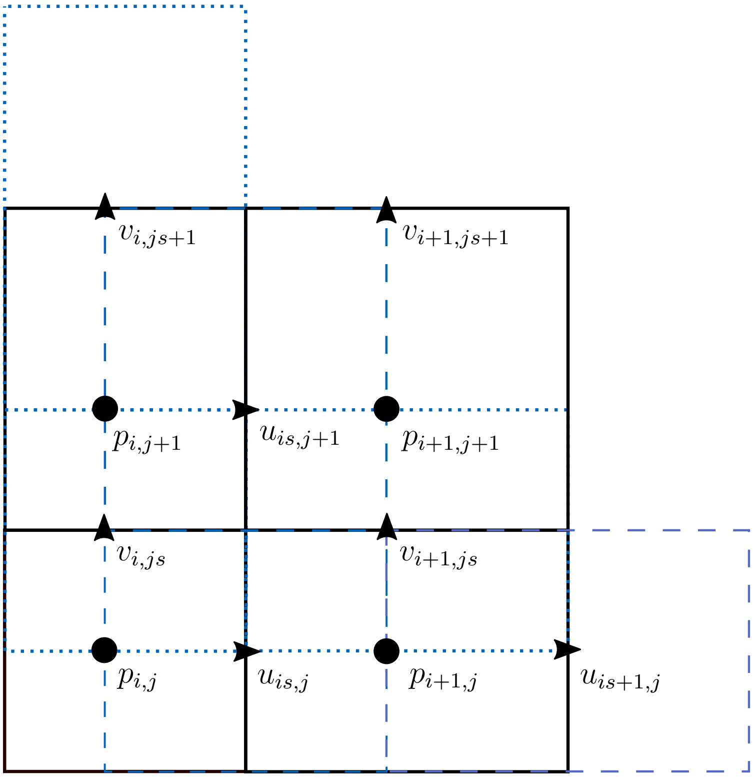


Figure 2.2: Arrangement of variables on a nonuniform staggered grids.

In this work, discretizations of the governing equations are carried out for uniform as well as non-uniform two-dimensional Cartesian grids with constant stretching factors in each dimension. Defining the length of i^{th} u-momentum cell to be $\Delta x_{is} = x_{i+1} - x_i$, the stretching factor reads:

$$s_x = \frac{\Delta x_{is}}{\Delta x_{is-1}}, \tag{2.4}$$

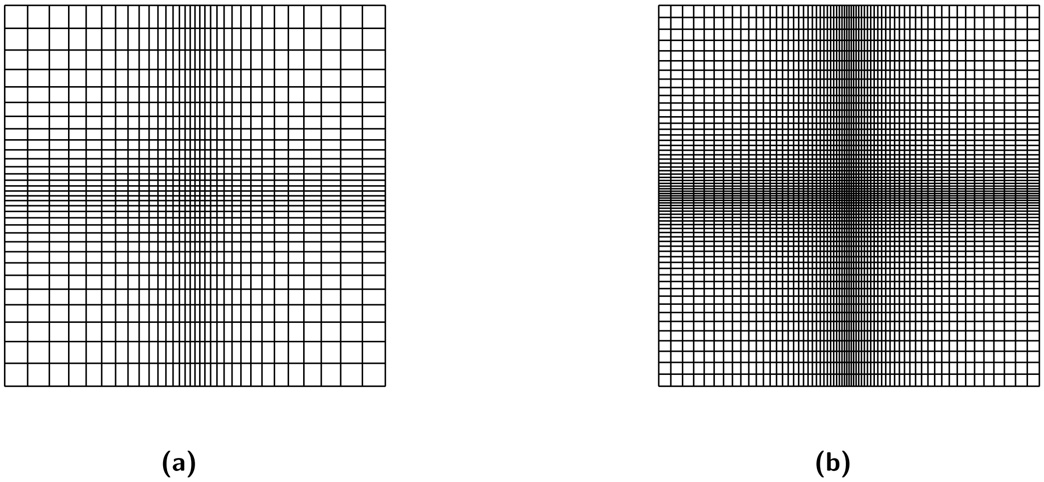
with the immediate consequence of using a constant stretch factor:

$$\Delta x_{is} = s_x \ \Delta x_{is-1} = s_x^2 \ \Delta x_{is-2} = s_x^k \ \Delta x_{is-k}.$$

Now, we can define the length of the i^{th} pressure cell ($\Delta x_i = x_{is} - x_{is-1}$) as:

$$\Delta x_i = \frac{1}{2}(\Delta x_{is-1} + \Delta x_{is}) = \frac{\Delta x_{is-1}}{2} \ (1 + s_x). \tag{2.5}$$

The use of non-uniform grids is often considered to be better than uniform grids for the same number of cells as it becomes advantageous when we seek to capture sharp gradients and nonlinearities in some spatial locations with minimal numerical error and with a cheaper system to be solved. For simplicity, this work considers only domains with periodic boundary conditions, hence the periodicity of the computational grid. Fig 2.3 shows an example of a periodic non-uniform grid in two spatial dimensions.



(a)

(b)

Figure 2.3: Successive stretched grid refinement with $s_x = s_y = 0.8$. (a) 32 cells. (b) 64 cells.

When we refine non-uniform grids, we choose to insert new grid points so that a constant stretch factor is maintained throughout the fine grid. Referring to Fig 2.4b we can write the stretch factor of the refined grid as:

$$s'_x = \frac{\Delta x'_{is,j}}{\Delta x'_{is-1,j}} = \frac{\Delta x'_{is+1,j}}{\Delta x'_{is,j}} = \frac{\Delta x'_{is+2,j}}{\Delta x'_{is+1,j}}.$$

We can show the relation between the stretching factor of the coarse and the fine grids by the following arithmetic:

$$\begin{aligned}
\frac{\Delta x'_{is+1,j} + \Delta x'_{is+2,j}}{\Delta x'_{is-1,j} + \Delta x'_{is,j}} = s_x \longrightarrow \ & \frac{\Delta x'_{is+1,j}(1 + {\scriptstyle \Delta x'_{is+2,j}/\Delta x'_{is+1,j}})}{\Delta x'_{is-1,j}(1 + {\scriptstyle \Delta x'_{is,j}/\Delta x'_{is-1,j}})} = s_x, \\
\frac{s'_x \Delta x'_{is,j}(1 + s'_x)}{\Delta x'_{is-1,j}(1 + s'_x)} = s_x \longrightarrow \ & (s'_x)^2 = s_x,
\end{aligned}$$

Hence,

$$s'_x = \sqrt{s_x} \tag{2.6}$$

From this relation, it can be said that the grid stretching factor always approaches the value 1 when the grid is refined.

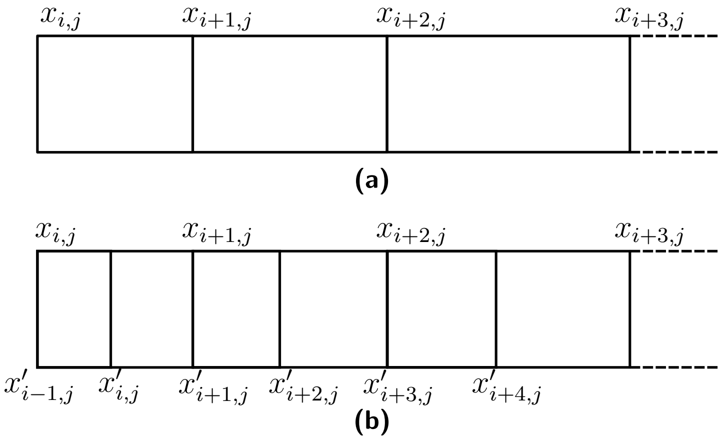


Figure 2.4: Non-uniform grid. (a) Initial grid. (b) Refined grid.

2.3. Conservation of momentum and conservation of mass

As was mentioned in section 2.1, the Navier-Stokes equations describe the conservation of momentum per unit mass (velocity) in each spatial dimension. Hence on two-dimensional staggered grids, we conserve the cell-averaged velocities $u^{xy}_{is,j}$, $v^{xy}_{i,js}$ for the x and y momentum, respectively. The cell-averaged values of an x-staggered variable (u-velocity) and a y-staggered variable (v-velocity) are defined as:

$$u^{xy}_{is,j} = \frac{1}{\Omega S_{is,j}} \int_{y_{js-1}}^{y_{js}} \int_{x_i}^{x_{i+1}} u(x,y)\ dxdy, \tag{2.7}$$

$$v^{xy}_{i,js} = \frac{1}{\Omega S_{i,js}} \int_{y_j}^{y_{j+1}} \int_{x_{is-1}}^{x_{is}} v(x,y)\ dxdy, \tag{2.8}$$

and that of a collocated variable (pressure) reads:

$$p^{xy}_{i,j} = \frac{1}{\Omega_{i,j}} \int_{y_{js-1}}^{y_{js}} \int_{x_{is-1}}^{x_{is}} p(x,y)\ dxdy. \tag{2.9}$$

In a similar way, the edge-averaged value of any variable can be defined by reducing the double integration to a single one. For example, $u^{y}_{i,j}$ is the edge-averaged u-velocity along the left boundary of a u-momentum cell.

Now, from the integral form of the NS equations (2.2), we can write the semi-discretitized u-momentum equation as:

$$\frac{\partial u^{xy}_{is,j}}{\partial t} = \frac{1}{\Omega S_{is,j}} \left(-F^{c}_{is,j} - F^{p}_{is,j} + F^{d}_{is,j} \right), \tag{2.10}$$

where $F^{c}_{is,j}$, $F^{p}_{is,j}$ and $F^{d}_{is,j}$ are the net convective, pressure and diffusive fluxes upon a u-cell, respectively. These fluxes are defined as:

$$F^{c}_{is,j} = \left([\mathsf{u}u]^{y}_{i+1,j} - [\mathsf{u}u]^{y}_{i,j} \right) \Delta y_i + \left([\mathsf{v}u]^{x}_{is,js} - [\mathsf{v}u]^{x}_{is,js-1} \right) \Delta x_{is}, \tag{2.11}$$

$$F^{p}_{is,j} = \left(p^{y}_{i+1,j} - p^{y}_{i,j} \right) \Delta y_j + \left(p^{x}_{is,js} - p^{x}_{is,js-1} \right) \Delta x_{is}, \tag{2.12}$$

$$F^{d}_{is,j} = \left(\left.\frac{\partial u}{\partial x}\right|^{y}_{i+1,j} - \left.\frac{\partial u}{\partial x}\right|^{y}_{i,j} \right) \Delta y_i + \left(\left.\frac{\partial u}{\partial y}\right|^{x}_{is,js} - \left.\frac{\partial u}{\partial y}\right|^{x}_{is,js-1} \right) \Delta x_{is}. \tag{2.13}$$

Attention should be paid to the distinction made between the convected velocity u and the convective velocity u which has to be conservative (i.e the divergence of this velocity over momentum cells has to vanish).

The continuity equation accompanying the Navier-Stokes equations describes the conservation of mass. On staggered grids, this is enforced by taking the divergence over the pressure cells as:

$$\nabla \cdot \mathbf{u} = \left.\frac{\partial u}{\partial x}\right|^{y} + \left.\frac{\partial v}{\partial y}\right|^{x} = \frac{1}{\Delta x_i} \left(u^{y}_{is,j} - u^{y}_{is-1,j} \right) + \frac{1}{\Delta y_j} \left(v^{x}_{i,js} - u^{x}_{i,js-1} \right) \tag{2.14}$$

All of the discrete equations introduced up to this point are exact and their second-order and fourth-order approximations are the concern of chapter 3.

2.4. Projection Method

When dealing with unsteady flow cases, a time integration scheme has to be implemented to advance the solution in time. However, this alone cannot guarantee the conservation of mass in the updated field, and therefore, the Fractional Time Step Method (FTSM) using the projection method was employed in this work [Chorin, 1967]. The projection method consists of three steps: a) Computing an intermediate velocity which does not satisfy the incompressibility constraint, b) Solving the pressure Poisson equation, and c) Projecting the intermediate velocity to the space of divergence free vector fields. To show how this method works, we consider the simple case of Explicit Euler for time advancement. Letting $\mathbf{u}^n$, $\mathbf{u}^*$ and $\mathbf{u}^{n+1}$ to be the current, intermediate and the updated velocity fields, respectively, then one can write:

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \Delta t \left(\mathbf{rhs}^n + \mathbf{P}^{n+1}\right), \tag{2.15}$$

$$\mathbf{u}^{*} = \mathbf{u}^n + \Delta t \left(\mathbf{rhs}^n + \mathbf{P}^{n}\right), \tag{2.16}$$

where $\mathbf{rhs}^n$ regroups the convective and diffusive terms at the current time, ($\mathbf{P}^{n+1}$, $\mathbf{P}^n$) are the current and the updated pressure terms, respectively. Subtracting (2.16) from (2.15) yields:

$$\begin{aligned} \mathbf{u}^{n+1} - \mathbf{u}^* &= \Delta t\ \mathbf{dP^{n+1}}, \\ \text{with, } \mathbf{dP}^{n+1} = \mathbf{P}^{n+1} - \mathbf{P}^n &= -\frac{1}{\rho}\nabla dp^{n+1}. \end{aligned} \tag{2.17}$$

In contrast to the intermediate velocity field, the updated velocity field must satisfy the continuity equation. Taking the divergence of (2.17) yields the pressure Poisson equation:

$$\nabla \cdot \nabla dp^{n+1} = \frac{\rho}{\Delta t}\nabla \cdot \mathbf{u}^*. \tag{2.18}$$

The pressure computed in (2.18) is then used to project the intermediate velocity and obtain a divergence-free velocity field. From (2.17) we can write:

$$\begin{aligned} \mathbf{u}^{n+1} &= \mathbf{u}^* - \frac{\Delta t}{\rho}\nabla dp^{n+1}, \\ p^{n+1} &= p^n + dp^{n+1}. \end{aligned} \tag{2.19}$$

Introducing the discrete divergence $\mathbf{D}$ and gradient $\mathbf{G}$ operators to (2.18) results in its discrete form:

$$\mathbf{DG}p = \frac{\rho}{\Delta t}\mathbf{Du}^*,$$

where the matrix product $\mathbf{DG}$ is known as the discrete Laplacain. While the second-order divergence and gradient operators are to be used for the second-order finite volume implementation, the fourth-order operators are employed for its fourth-order counterpart. In this work, we tend to match the order of the discrete gradient operator used to evaluate the pressure gradient over the momentum cells together with that of the formal order Laplacian, such that consistent projection methods are utilized.

2.5. Time integration scheme

In this work, the explicit third-order low storage Runge-Kutta (RK3) scheme [Williamson, 1980] was used to advance the solution in time. Algorithm 1 from [Peller, 2010] shows the three RK sub-steps along with the projection method. This time integration scheme is third-order accurate for the velocity and first-order accurate for the pressure.

Algorithm 1: Low storage RK3 with the projection method

```
initialize u, v and p fields
$n \leftarrow 1$
while $n <$ number of time steps do

    First RK sub-step:
    $\mathbf{rhs}^1 \leftarrow \mathbf{F}^c(\mathbf{u}^n) + \mathbf{F}^d(\mathbf{u}^n) + \mathbf{F}^p(p^n)$
    $\mathbf{u}^n \leftarrow \mathbf{u}^n + \frac{1}{3}\,\Delta t\,\mathbf{rhs}^1$                // Intermediate velocity
    $\Delta dp \leftarrow \frac{\rho}{\frac{1}{3}\,\Delta t}\,\nabla \cdot \mathbf{u}^n$                // Correct pressure
    $\mathbf{u}^n \leftarrow \mathbf{u}^n - \frac{\frac{1}{3}\,\Delta t}{\rho}\,\nabla dp$                // Project fields
    $p^n \leftarrow p^n + \nabla dp$

    Second RK sub-step:
    $\mathbf{rhs}^2 \leftarrow \mathbf{F}^c(\mathbf{u}^n) + \mathbf{F}^d(\mathbf{u}^n) + \mathbf{F}^p(p^n)$
    $\mathbf{rhs}^2 \leftarrow \mathbf{rhs}^2 - \frac{5}{9}\,\mathbf{rhs}^1$
    $\mathbf{u}^n \leftarrow \mathbf{u}^n + \frac{15}{16}\,\Delta t\,\mathbf{rhs}^2$                // Intermediate velocity
    $\Delta dp \leftarrow \frac{\rho}{\frac{3}{4}\,\Delta t}\,\nabla \cdot \mathbf{u}^n$                // Correct pressure
    $\mathbf{u}^n \leftarrow \mathbf{u}^n - \frac{\frac{3}{4}\,\Delta t}{\rho}\,\nabla dp$                // Project fields
    $p^n \leftarrow p^n + \nabla dp$

    Third RK sub-step:
    $\mathbf{rhs}^1 \leftarrow \mathbf{F}^c(\mathbf{u}^n) + \mathbf{F}^d(\mathbf{u}^n) + \mathbf{F}^p(p^n)$
    $\mathbf{rhs}^1 \leftarrow \mathbf{rhs}^1 - \frac{153}{128}\,\mathbf{rhs}^2$
    $\mathbf{u}^n \leftarrow \mathbf{u}^n + \frac{8}{15}\,\Delta t\,\mathbf{rhs}^1$                // Intermediate velocity
    $\Delta dp \leftarrow \frac{\rho}{\Delta t}\,\nabla \cdot \mathbf{u}^n$                // Correct pressure
    $\mathbf{u}^{n+1} \leftarrow \mathbf{u}^n - \frac{\Delta t}{\rho}\,\nabla dp$                // Project fields
    $p^{n+1} \leftarrow p^n + \nabla dp$

    $n \leftarrow n + 1$
```

2.6. The Immersed Boundary Method

To simulate a wall bounded fluid flow or particle suspension, one has to solve the Navier-Stokes (NS) equations in the fluid domain with no-slip boundary conditions imposed at the interface between the fluid domain and the solid object. Among the schemes that have been applied to moving interface problems, the most common one is the conventional approach of using a body-conforming mesh as shown in Fig 2.5a to directly enforce the boundary conditions. This strategy is followed by the Arbitrary Lagrangian-Eulerian (ALE) method [Donea et al., 2004], which combines the advantages of both Lagrangian description following an individual parcel as it moves through space and Eulerian description focusing on fixed locations in the space. Although adaptive local mesh refinement techniques have been in continuous development, this method requires coordinate transformation and complex re-meshing depending on solid particles displacements as time evolves, leading to a substantial computational cost. To avoid repeated re-meshing, another alternative is to mesh the whole computational domain consisting of the fluid and the solid with a fixed structured Cartesian grid. Such an embedded domain method is called the Immersed Boundary method (hereinafter also referred to as IB-method).

The Immersed Boundary method was first introduced by Peskin in 1972 [Peskin, 1972] where it was used to simulate cardiac mechanics and associated blood flow. Fig 2.5b shows an example of an IBM mesh. As the suspended solids are interpreted as a part of the fluid domain, the grid is not necessarily conforming to the body geometry. In order to incorporate the impact of the solid phase on the flow, one has to manipulate the governing equations in an appropriate way. Since the original IB-method was introduced, numerous techniques were adapted and developed based on the IB principle. According to [Mittal and Iaccarino, 2005], depending on the approach the boundary conditions are incorporated to the governing equations, these techniques can be divided into two main categories, namely, the discrete forcing approach and the continuous forcing approach.

In the discrete forcing method, the governing equations are first discretized with no considerations given to the immersed boundary, then its influence is brought up by manipulating the discretized system of equations in such a way to enforce the no-slip condition along the fluid-solid interface. This method can be seen similar to the technique of applying a Dirichlet boundary conditions (homogeneous or non-homogeneous) to the outer boundary of the computational domain Ω with the difference that this modifications can also be applied to a set of an interior cells. Examples of this method are the Ghost cell finite difference approach and the Cut cell finite volume approach discussed thoroughly in [Mittal and Iaccarino, 2005].

On the other hand, the continuous forcing method, such that of [Peskin, 1972] and the virtual boundary method of [Goldstein et al., 1993], integrates the boundary conditions by adequately formulating a forcing term f and adding it to the right hand side of the continuous governing equations. In contrast to its discrete counterpart, the continuous forcing approach does not allow the prescription of velocity values at particular cells of the finite volume mesh. Therefore, it is found to be less accurate than the discrete approach in preserving the sharpness of the immersed boundary. However, three major advantages can be stated for this method of boundary conditions incorporation. From a physical point of view, associating the presence of a solid body by means of a forcing term can be thought of as using a free body diagram to describe the interaction between the fluid and solid phases. The second advantage of the continuous approach is its flexibility to include additional properties of the immersed solid, such as its material properties. This feature enables one to simulate the interaction between elastic objects and a fluid. Furthermore, continuous forcing methods are more practicable for moving immersed boundaries due to their simple formation.

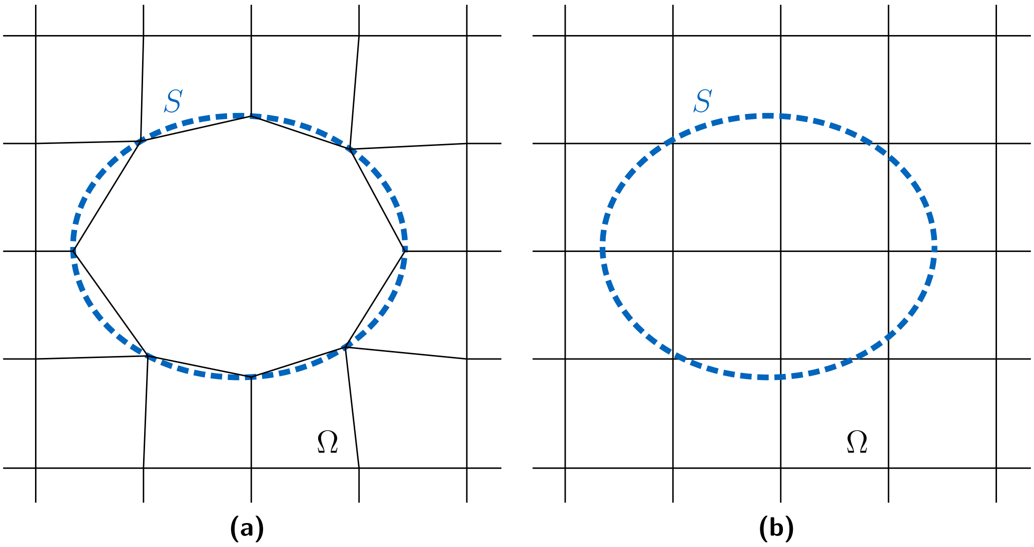


Figure 2.5: (a) Body conforming mesh. (b) IB-method Cartesian uniform mesh.

After these general considerations, it is noteworthy to mention that the continuous forcing immersed boundary method proposed by [Uhlmann, 2005] was adopted in this work. The reasons behind this choice is the weak dependency of the method on the underlying discretization scheme, its ability to easily treat moving immersed boundaries and its simplicity to be implemented in a computer code.

3. Finite Volume Discretization of Navier-Stokes on Staggered Grids

In this chapter, the second-order and the fourth-order finite volume discretizations of the Navier-Stokes equations are presented. The approximations addressed here are for both the uniform and non-uniform staggered grid systems. In addition, a derivation of the discrete Laplacian operator with a formal order of accuracy is demonstrated for each case. Finally, we close this chapter by validating the implemented schemes numerically.

3.1. Second-order discretizations on uniform and non-uniform grids

In the second-order context, the cell-averaged and the edge-averaged values are approximated by the cell-centered and edge-centered values, respectively. Fluxes through the boundaries of a finite volume are approximated using neighboring cell-centered values. In the following, we derive the discretization for uniform and non-uniform staggered grids. We restrict the discretization presented to the u-momentum equation as the analogous set of the v-component is straightforward.

3.1.1. Semi-Discretized equation on uniform grids

Fig 3.1 shows the cell-centered values needed for the evaluation of u-momentum cell fluxes. Replacing the surface integral by a summation of flux terms over the faces of the cell [Ferziger and Peric, 2001, p. 71], then the convective, diffusive and pressure fluxes read:

$$\begin{aligned}
F^c_{is,j} &= (u_{i+1,j}\ u_{i+1,j} - u_{i,j}\ u_{i,j})\,\Delta y + (v_{is,js}\ u_{is,js} - v_{is,js-1}\ u_{is,js-1})\,\Delta x,\\
F^p_{is,j} &= \frac{1}{\rho}(p_{i+1,j} - p_{i,j})\Delta y,\\
F^d_{is,j} &= \nu\left[\left(\frac{\partial u}{\partial x}\Big|_{i+1,j} - \frac{\partial u}{\partial x}\Big|_{i,j}\right)\Delta y + \left(\frac{\partial u}{\partial y}\Big|_{is,js} - \frac{\partial u}{\partial y}\Big|_{is,js-1}\right)\Delta x\right],
\end{aligned}$$

with the second-order approximation of the edge-centered values:

$$\begin{aligned}
u_{i+1,j} &= \frac{1}{2}(u_{is+1,j} + u_{is,j}), & u_{is,js} &= \frac{1}{2}(u_{is,j+1} + u_{is,j}),\\
v_{is,js} &= \frac{1}{2}(v_{i,js} + v_{i+1,js}), & \frac{\partial u}{\partial x}\Big|_{i+1,j} &= \frac{u_{is+1,j} - u_{is,j}}{\Delta x},
\end{aligned}$$

$$\left.\frac{\partial u}{\partial y}\right|_{is,js} = \frac{u_{is,j+1} - u_{is,j}}{\Delta y}.$$

Hence the semi-discretized form of equation (2.2) for the u-velocity component reads:

$$\frac{\partial u}{\partial t} = \frac{1}{4}\left[\frac{(u_{is,j} + u_{is-1,j})^2 - (u_{is+1,j} + u_{is,j})^2}{\Delta x}\right.$$
$$\left.+\frac{(v_{i,js-1} + v_{i+1,js-1})(u_{is,j} + u_{is,j-1}) - (v_{i,js} + v_{i+1,js})(u_{is,j+1} + u_{is,j})}{\Delta y}\right] + \frac{1}{\rho}\left[\frac{p_{i,j} - p_{i+1,j}}{\Delta x}\right]$$
$$+ \nu\left[\frac{u_{is+1,j} - 2u_{is,j} + u_{is-1,j}}{\Delta x^2} + \frac{u_{is,j+1} - 2u_{is,j} + u_{is,j-1}}{\Delta y^2}\right].$$

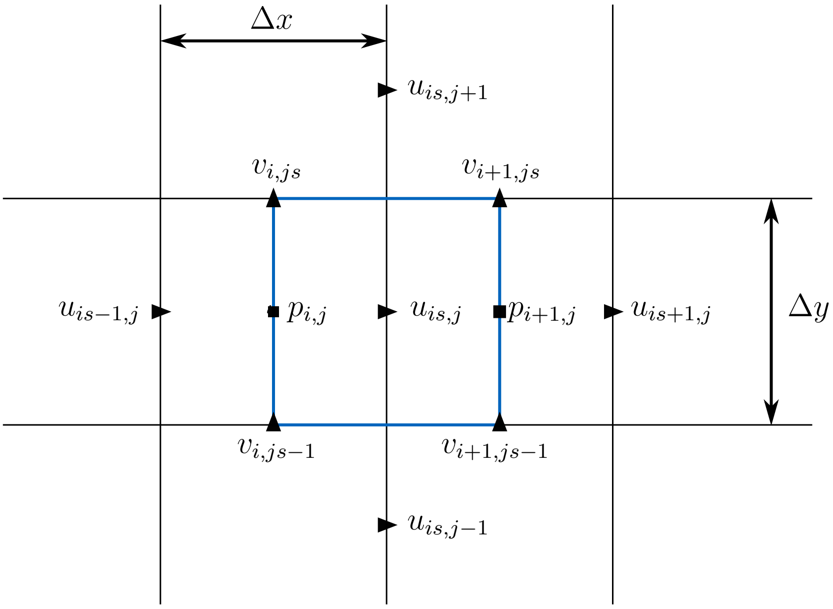


Figure 3.1: Stencil required for the computation of u-cell fluxes.

3.1.2. Semi-Discretized equation on non-uniform grids

The discretization adopted in this section was described by [Verstappen and Veldman, 2003]. In their work, they showed the importance of using a symmetry-preserving discretization to conserve the total mass, momentum and kinetic energy. This section will briefly introduce the idea behind this concept, followed by the discretization afterwards.

A symmetry-preserving discretization is known in the literature as mimetic discretization, and it denotes that the symmetry properties of the underlying differential operators are preserved in their discrete counterparts. While the convective operator is approximated by a skew-symmetric discrete operator, the diffusive operator is approximated by a symmetric positive-definite one. We let $\mathbf{\Omega}$ denote a positive-definite diagonal matrix representing the size of momentum cells control volumes and we define the convective, diffusive and the divergence coefficient matrices as $\mathbf{C}(\mathbf{u_h})$, $\mathbf{F}$ and $\mathbf{D}$, respectively. The conservation property and stability then can be shown by first writing the semi-discretized NS equations in a matrix-vector notation as:

$$\mathbf{\Omega}\frac{d\mathbf{u}}{dt} + \mathbf{C}(\mathbf{u})\mathbf{u} + \mathbf{F}\mathbf{u} - \mathbf{D}^{\mathsf{T}}\mathbf{p} = \mathbf{0}, \qquad \mathbf{D}\mathbf{u} = \mathbf{0},$$

where $^{\mathsf{T}}$ denotes the transpose of a matrix. Writing the evolution of the kinetic energy $(\mathbf{u}^{\mathsf{T}}\mathbf{\Omega}\mathbf{u})$ as:

$$\begin{aligned}\frac{d\left(\mathbf{u}^{\mathsf{T}}\mathbf{\Omega}\mathbf{u}\right)}{dt} &= -\mathbf{u}^{\mathsf{T}}\left(\mathbf{C}(\mathbf{u}) + \mathbf{C}^{\mathsf{T}}(\mathbf{u})\right)\mathbf{u} - \mathbf{u}^{\mathsf{T}}(\mathbf{F} + \mathbf{F}^{\mathsf{T}})\mathbf{u} + \mathbf{u}^{\mathsf{T}}(\mathbf{D}^{\mathsf{T}}\mathbf{p}) + (\mathbf{D}^{\mathsf{T}}\mathbf{p})^{\mathsf{T}}\mathbf{u} \\ &= -\mathbf{u}^{\mathsf{T}}(\mathbf{F} + \mathbf{F}^{\mathsf{T}})\mathbf{u} \leq 0,\end{aligned}$$

where the convective term cancels out as $\mathbf{C}$ is skew-symmetric ($\mathbf{C}^{\mathsf{T}} = -\mathbf{C}$), and the pressure term cancels out because the gradient matrix is equal to the negative of the transpose of the divergence matrix. Hence under this condition, the energy of the discrete solution is conserved if the flow is inviscid, and decreases in time if dissipation is present.
In order to comply with the proposition stated above, the convected velocities at momentum cell edges have to be interpolated with weights of $\frac{1}{2}$:

$$u_{i+1,j} = \frac{1}{2}(u_{is+1,j} + u_{is,j}).$$

In contrast, the convective velocities (u) are linearly interpolated to momentum cell edges by means of a second-order linear interpolation:

$$\mathsf{u}_{i+1,j} = \omega_{i+1,j}\ u_{is+1,j} + (1 - \omega_{i+1,j})\ u_{is,j}, \qquad \text{with } \omega_{i+1,j} = \frac{x_{i+1,j} - x_{is,j}}{x_{is+1,j} - x_{is,j}}.$$

This linear interpolation can be shown to give a divergence-free field for the convective velocity at momentum cells. The diffusive term is discretized with a symmetric positive-definite diffusive operator:

$$\left.\frac{\partial u}{\partial x}\right|_{i+1,j} = \frac{u_{is+1,j} - u_{is,j}}{x_{is+1,j} - x_{is,j}}.$$

In spite of the local truncation error of this approximation being only first-order, [Manteuffel and White, 1986] showed that the local truncation error is not decisive and proved that this approximation delivers second-order accurate solution on non-uniform grids. We shall present this behavior following the detailed explanation in [Ferziger and Peric, 2001, p. 61]. Referring to the non-uniform grid in Fig 2.4 , we write the Taylor expansion of $u_{is+1,j}$ and $u_{is,j}$ around $x_{i+1,j}$:

$$u_{is+1,j} = u_{i+1,j} + (x_{is+1,j} - x_{i+1,j})\ u'_{i+1,j} + \frac{(x_{is+1,j} - x_{i+1,j})^2}{2}\ u''_{i+1,j} + H, \tag{3.1}$$

$$u_{is,j} = u_{i+1,j} - (x_{i+1,j} - x_{is,j})\ u'_{i+1,j} + \frac{(x_{i+1,j} - x_{is,j})^2}{2}\ u''_{i+1,j} + H, \tag{3.2}$$

where H denotes higher-order terms. Subtracting (3.2) from (3.1) and solving for the first derivative we obtain:

$$u'_{i+1,j} = \frac{u_{is+1,j} - u_{is,j}}{x_{is+1,j} - x_{is,j}} - \frac{(x_{is+1,j} - x_{i+1,j})^2 - (x_{i+1,j} - x_{is,j})^2}{2(x_{is+1,j} - x_{is})}\ u''_{i+1,j} + H.$$

Employing some of the grid definitions introduced in section 2.2, we can simplify the leading truncation error term to:

$$\epsilon = -\frac{\Delta x_{is+1,j}^2 - \Delta x_{is,j}^2}{4(\Delta x_{is+1,j} + \Delta x_{is,j})}\, u''_{i+1,j} = -\frac{\Delta x_{is+1,j} - \Delta x_{is,j}}{4}\, u''_{i+1,j} = \frac{\Delta x_{is,j}(1-s_x)}{4}\, u''_{i+1,j}.$$

To see what happens when the grid is refined, we take the ratio of the leading truncation error of a coarse and a refined grid (with primed parameters) at the point $x_{i+1,j}$:

$$r = \frac{\epsilon}{\epsilon'} = \frac{\Delta x_{is,j}\ (1-s_x)}{\Delta x'_{is,j}\ (1-s'_x)}. \tag{3.3}$$

From 2.4b, we write $\Delta x_{is,j}$ in terms of $\Delta x'_{is,j}$ as:

$$\Delta x_{is,j} = \Delta x'_{is} + \Delta x'_{is-1} = (s'_x + 1)\Delta x'_{is-1}.$$

Substituting this relation into equation (3.3) we obtain:

$$r = \frac{(1+s'_x)^2}{s'_{x'}}. \tag{3.4}$$

When the grid is uniform (i.e $s'_x = 1$), this ratio has the value 4 which implies a second-order rate of convergence. However, in in expanding grids (i.e $s'_x > 1$) or contracting grids (i.e $s'_x < 1$) the ratio becomes strictly greater than 4. This implies that a rate of convergence higher than 2 is achieved. Since that the stretching factor tends to the value 1 as the grid is refined (see section 2.2), error will converge at a second-order asymptotically.

3.1.3. Formal second-order discrete Laplacian operator

Solving the pressure Poisson equation (2.18) in the projection method is an essential step to enforce the continuity equation on the velocity field. This step is known to be the most consuming part for the computations of incompressible flows. Among the different alternatives available to solve this equations (direct solver or an iterative solver), a direct solver based on the LU factorization was used to solve the Poisson equation in its matrix-vector notation:

$$\mathbf{Lp} = \frac{\rho}{\Delta t}\mathbf{Du}^*, \tag{3.5}$$

where $\mathbf{L}$ is the Laplacian square matrix of size $[N_xN_y \times N_xN_y]$, $\mathbf{p}$ is the pressure in each computational cell organized into a column vector of size $[N_xN_y \times 1]$ and $\mathbf{Du}^*$ is a $[N_xN_y \times 1]$ vector containing the divergence of the predicted velocity field over each respective computational pressure cell. As was already mentioned in 2.4, the discrete Laplace operator is defined as the inner product of a discrete divergence and a discrete gradient operators, i.e $\mathbf{L} = \mathbf{DG}$. In the context of staggered grid system, the gradient operator is applied over a momentum computational cell whereas the divergence is applied over pressure computational cells. Having this in mind and employing the second-order divergence and gradient operators, the formal second-order discrete Laplace operator $\mathbf{L_2}$ applied to pressure on uniform grids reads:

$$\begin{aligned}\mathbf{L_2}p &= D_{2x}G_{2x}p + D_{2y}G_{2y}p\\ &= \frac{p_{i-1,j} - 2p_{i,j} + p_{i+1,j}}{\Delta x^2} + \frac{p_{i,j-1} - 2p_{i,j} + p_{i,j+1}}{\Delta y^2}.\end{aligned}$$

For the purpose of illustration, we break-up the application of the operators on non-uniform grids to:

1. Pressure gradient over $u_{is,j}$ momentum-cell:

$$(G_{2x}p)_{is,j} = \frac{p_{i+1,j} - p_{i,j}}{x_{i+1} - x_i}.$$

2. Pressure gradient over $u_{is-1,j}$ momentum-cell:

$$(G_{2x}p)_{is-1,j} = \frac{p_{i,j} - p_{i-1,j}}{x_i - x_{i-1}}.$$

3. Divergence of the gradients 1 and 2 over $p_{i,j}$ pressure cell:

$$D_{2x}G_{2x}p = \frac{(G_{2x}p)_{is,j} - (G_{2x}p)_{is-1,j}}{x_{is,j} - x_{is-1,j}} = \frac{1}{x_{is,j} - x_{is-1,j}}\left[\frac{p_{i+1,j} - p_{i,j}}{x_{i+1} - x_i} - \frac{p_{i,j} - p_{i-1,j}}{x_i - x_{i-1}}\right].$$

Given the stretched grid definition introduced in section 2.2, we can simplify the above expression to:

$$D_{2x}G_{2x}p = \frac{1}{\Delta x_{is-1}\Delta x_i}p_{i-1,j} - \frac{2}{\Delta x_{is}\Delta x_{is-1}}p_{i,j} + \frac{1}{\Delta x_{is}\Delta x_i}p_{i+1,j}.$$

Hence, writing $\mathbf{L_2}$p for non-uniform grids becomes straightforward. Now one can construct the Laplacian matrix by letting the coefficients of $p_{i-1,j},\ p_{i+1,j},\ p_{i,j-1},\ p_{i,j+1}$ and $p_{i,j}$ be $a_w,\ a_e,\ a_s,\ a_n,$ and a_c, respectively.

The linear system of equations below shows equation (3.5) for a computational domain of size $N_x = N_y = 3$ with periodic boundary conditions applied to both spatial dimensions.

$$\underbrace{\begin{bmatrix} a_c & a_e & a_w & a_n & 0 & 0 & a_s & 0 & 0\\ a_w & a_c & a_e & 0 & a_n & 0 & 0 & a_s & 0\\ a_e & a_w & a_c & 0 & 0 & a_n & 0 & 0 & a_s\\ a_s & 0 & 0 & a_c & a_e & a_w & a_n & 0 & 0\\ 0 & a_s & 0 & a_w & a_c & a_e & 0 & a_n & 0\\ 0 & 0 & a_s & a_e & a_w & a_c & 0 & 0 & a_n\\ a_n & 0 & 0 & a_s & 0 & 0 & a_c & a_e & a_w\\ 0 & a_n & 0 & 0 & a_s & 0 & a_w & a_c & a_e\\ 0 & 0 & a_n & 0 & 0 & a_s & a_e & a_w & a_c \end{bmatrix}}_{\mathbf{L}} \underbrace{\begin{bmatrix} p_{1,1}\\ p_{2,1}\\ p_{3,1}\\ p_{1,2}\\ p_{2,2}\\ p_{3,2}\\ p_{1,3}\\ p_{2,3}\\ p_{3,3} \end{bmatrix}}_{\mathbf{p}} = \frac{\rho}{\Delta t} \underbrace{\begin{bmatrix} (Du^*)_{1,1}\\ (Du^*)_{2,1}\\ (Du^*)_{3,1}\\ (Du^*)_{1,2}\\ (Du^*)_{2,2}\\ (Du^*)_{3,2}\\ (Du^*)_{1,3}\\ (Du^*)_{2,3}\\ (Du^*)_{3,3} \end{bmatrix}}_{\mathbf{Du^*}}$$

3.2. Fourth-order discretizations on uniform and non-uniform grids

This section treats the fourth-order finite volume discretization of the Navier-Stokes equations. It was mentioned in section 2.1 that the finite volume method describes the change of a cell-averaged quantity by the net fluxes on the boundaries enclosing that cell. In section 3.1, we approximated the cell-averaged and edge-averaged values by interchanging them with the pointwise centered values. This approximation was permissible as its local truncation error is second-order. However, this approximation is not allowed in higher-order context and an explicit distinction should be made between these quantities.

3.2.1. Cell-centered deconvolution of velocity

Cell-centered deconvolution is known to be the approximation of an edge-average value at the center of a cell from its neighboring cell-average values. Cell-centered deconvolution of a momentum cell finds the edge-average u and v velocities at the boundaries of a pressure cell. These quantities are used to evaluate the divergence of the the velocity field (conservation of mass). The fourth-order cell centered deconvolution of a u-momentum cell can be written as:

$$u^y_{is,j} = \beta_1 u^{xy}_{is-1,j} + \beta_2 u^{xy}_{is,j} + \beta_3 u^{xy}_{is+1,j}. \tag{3.6}$$

The coefficients β_1, β_2 and β_3 can be derived from the Taylor series expansion of each term on the right hand side around $x_{is,j}$. The derivation below is shown for a uniform grid configuration where in this case, $k = -1, 0, 1$.

$$\begin{aligned}
u^{xy}_{is+k,j} &= \frac{1}{\Delta x}\int_{x_{i+k\Delta x}}^{x_i+(k+1)\Delta x} u^y \; dx \\
&\approx \frac{1}{\Delta x}\int_{x_{i+k\Delta x}}^{x_i+(k+1)\Delta x} u^y_{is,j} + (x-x_{is})\frac{du}{dx}\Big|^y_{is,j} + \frac{(x-x_{is})^2}{2!}\frac{d^2u}{dx^2}\Big|^y_{is,j} \; dx \\
&\approx \frac{1}{\Delta x}\left[u^y_{is,j}x + \frac{(x-x_{is})^2}{2}\frac{du}{dx}\Big|^y_{is,j} + \frac{(x-x_{is})^3}{6}\frac{d^2u}{dx^2}\Big|^y_{is,j}\right]\Bigg|^{x_i+(k+1)\Delta x}_{x_{i+k\Delta x}} .
\end{aligned}$$

Upon the substitution of limits of integration for $u_{is-1,j}$, $u_{is,j}$ and $u_{is+1,j}$ and matching the orders of the derivatives with left hand side of (3.6), the coefficients are $\beta_1 = \beta_3 = -\frac{1}{24}$ and $\beta_2 = \frac{13}{12}$. The same derivation could be applied to non-uniform grids taking into consideration the different grid spacing of momentum cells (see Appendix A.1), employing the consistency criterion the coefficients are found as:

$$\beta_1 = -\frac{\Delta x_{is}^2}{4(\Delta x_{is-1} + \Delta x_{is})(\Delta x_{is-1} + \Delta x_{is} + \Delta x_{is+1})}, \qquad \beta_2 = 1 - (\beta_1 + \beta_3),$$

$$\beta_3 = -\frac{\Delta x_{is}^2}{4(\Delta x_{is} + \Delta x_{is+1})(\Delta x_{is-1} + \Delta x_{is} + \Delta x_{is+1})}.$$

These coefficients are similar to those found in [Hokpunna, 2009].

3.2.2. Cell-centered deconvolution of pressure

In order to write a fourth-order approximation of the pressure gradient over a momentum-cell, it is essential to first approximate the edge-average pressure at its boundaries. Similar to the cell-centered deconvolution above, the fourth-order approximation of the edge-average pressure at the left boundary of a u-momentum cell reads:

$$p^{y}_{i,j} = \beta_4 p^{xy}_{i-1,j} + \beta_5 p^{xy}_{i,j} + \beta_6 p^{xy}_{i+1,j} \tag{3.7}$$

On uniform grids, these coefficients are identical to the ones derived in (3.6). However, on non-uniform grids, even though a stencil of the same width is used, pressure cell-averaged values are not situated on the centers of the cells (edges halfway between grid points). Therefore, one has to use the following coefficients (see Appendix A.2):

$$\begin{aligned}
\beta_4 &= \frac{2\Delta x_{is}^2 + \Delta x_{is}\Delta x_{is+1} - \Delta x_{is-1}3\Delta x_{is} - \Delta x_{is-1}\Delta x_{is+1}}{(\Delta x_{is} + 2\Delta x_{is-1} + \Delta x_{is-2})A_1}, \\
\beta_6 &= \frac{2\Delta x_{is-1}^2 + \Delta x_{is-1}\Delta x_{is-2} - \Delta x_{is-1}3\Delta x_{is} - \Delta x_{is}\Delta x_{is-2}}{(\Delta x_{is-1} + 2\Delta x_{is} + \Delta x_{is+1})A_1}, \\
\beta_5 &= 1 - (\beta_4 + \beta_5), \\
A_1 &= \Delta x_{is-2} + 2\Delta x_{is-1} + 2\Delta x_{is} + \Delta x_{is+1}
\end{aligned}$$

Here, β_6 differs from the coefficient derived in [Hokpunna, 2009] with a minus sign. Other than this difference, all other coefficients are identical.

3.2.3. Filtering for the computation of convected velocities and first derivatives on uniform grids

The evaluation of the edge-average convected (transported) velocities and first derivatives which are both located at the interface between momentum cells is required for the computation of the convective and diffusive fluxes, respectively. These edge-averaged quantities are located at non-staggered positions and a pre-processed adjacent cell-averaged velocities are used for their approximation.
The concept of improving the order of approximation followed in this section was presented by [Schwertfirm et al., 2008]. The main idea is to introduce a pre-processing step to the second order mid-point interpolation rule and the second order central scheme for the first derivative to enhance their spatial resolution. Using this approach, one can reduce the stencil length required for each individual stencil operation and hence the number of ghost cells layers needed. What follows is a brief description of the method.

Lets consider an operator A_{exact}, which if applied to a 1D velocity field, delivers $u(x)_{op,exact} = A_{exact}u(x)$. In this context, the operator A_{exact} could be the interpolation or the first derivative of the field variable $u(x)$. On the other hand, consider an approximation of this operator to be A_{app} which returns the function $u(x)_{op,app} = A_{app}u(x)$ and that these two operators can be linked by a filter G as $A_{app} = G * A_{exact}$. Consequently, one can retrieve $u_{op,exact}$ from $u_{op,app}$ in a pre-processing step if the inverse G^{-1} exists and commutes with A_{app} as shown:

$$u_{op,exact} = G^{-1}u_{op,app} = G^{-1}A_{app}u(x) = A_{app}G^{-1}u(x).$$

Now, $G^{-1}u(x)$ can be performed as a pre-processing step to A_{app}. Due to the fact that constructing and inverting the filter G in physical space is difficult to accomplish, it is more convenient to approximate the implied filter $G' \approx G$ by the local explicit moving averages formula:

$$\bar{u}_{is}^{xy} = \alpha_1 u_{is-1}^{xy} + \alpha_2 u_{is}^{xy} + \alpha_3 u_{is+1}^{xy} = G' * u_{is}^{xy}, \tag{3.8}$$

where α_1, α_2 and α_3 denote the filter coefficients and their derivation will be explained later in this section. By letting Q be an approximate of the filter's inverse ($Q \approx G'^{-1}$), [Schwertfirm et al., 2008] applied the approximate deconvolution relation given by [Stolz and Adams, 1999] :

$$Q(G')_M = \sum_{m=0}^{M}(I - G')^m, \tag{3.9}$$

where I is the identity, and $M + 1$ is the number of terms taken in the deconvolution. Given the introduced approximations, $Q(G')u_{is}^{xy}$ becomes the pre-processing step of the field variable $u(x)$ before the operator A_{app} is applied. In addition, it was reported in [Schwertfirm et al., 2008] that it is sufficient to use a two-term deconvolution (M=1) in (3.9) to achieve a fourth order spatial accuracy on a uniform grid. Hence, we can write the pre-processed field as:

$$\begin{aligned} Q_1(G')u_{is}^{xy} &= \sum_{m=0}^{1}(I - G')^m * u_{is}^{xy} = u_{is}^{xy} + (1 - G') * u_{is}^{xy} \\ &= -\alpha_1 u_{is-1}^{xy} + (2 - \alpha_2)u_{is}^{xy} - \alpha_3 u_{is+1}^{xy}. \end{aligned} \tag{3.10}$$

Depending on the nature of the approximated operator A_{app} (whether it is second-order midpoint rule or central scheme), the filter coefficients are derived by matching terms with the respective explicit fourth-order formula.

Fourth-order convected velocity

An approximation of the edge-average convected velocity at the left boundary of a u-momentum cell can be written by applying the second order interpolation mid-point rule to the pre-processed cell-averaged u-velocity field as:

$$u_{i,j}^{y} = \frac{1}{2}(Q_1 u_{is,j}^{xy} + Q_1 u_{is-1,j}^{xy}),$$

with,

$$\begin{aligned} Q_1 u_{is,j}^{xy} &= -\alpha_1 u_{is-1,j}^{xy} + (2 - \alpha_2)u_{is,j}^{xy} - \alpha_3 u_{is+1,j}^{xy}, \\ Q_1 u_{is-1,j}^{xy} &= -\alpha_1 u_{is-2,j}^{xy} + (2 - \alpha_2)u_{is-1,j}^{xy} - \alpha_3 u_{is,j}^{xy}, \end{aligned}$$

hence,

$$u_{i,j}^{y} = \frac{1}{2}\left[-\alpha_1 u_{is-2,j}^{xy} + (-\alpha_1 + 2 - \alpha_2)u_{is-1,j}^{xy} + (2 - \alpha_2 - \alpha_3)u_{is,j}^{xy} - \alpha_3 u_{is+1,j}^{xy}\right]. \tag{3.11}$$

In order to derive the filter coefficients, one has to match the terms of equation (3.11) with the explicit fourth-order scheme given by:

$$u^{y}_{i,j} = -\frac{1}{12}u^{xy}_{is-2,j} + \frac{7}{12}u^{xy}_{is-1,j} + \frac{7}{12}u^{xy}_{is,j} - \frac{1}{12}u^{xy}_{is+1,j}. \tag{3.12}$$

The coefficients of equation (3.12) were derived by taking the Taylor expansion around $x_{i,j}$ for each term on the right-hand-side, matching the derivatives with the left-hand-side and solving a linear system of equations (see Appendix A.3). As a result, the filter's coefficients were found to be $\alpha_3 = \alpha_1 = \frac{1}{6}$ and $\alpha_2 = \frac{4}{6}$.

Fourth-order first derivative

The fourth-order edge-averaged approximation of the first derivative of the velocity at the boundary of a momentum cell is obtained by applying a filter with a different set of coefficients to the cell-averaged velocity field and evaluating the derivative using the second order central scheme:

$$\left.\frac{\partial u}{\partial x}\right|^{y}_{i,j} = \frac{1}{\Delta x}\left(Q_1 u^{xy}_{is,j} - Q_1 u^{xy}_{is-1,j}\right),$$

hence,

$$\left.\frac{\partial u}{\partial x}\right|^{y}_{i,j} = \frac{1}{\Delta x}\left[\alpha_4 u^{xy}_{is-2,j} + \left(-\alpha_4 - 2 + \alpha_5\right)u^{xy}_{is-1,j} + \left(2 - \alpha_5 + \alpha_6\right)u^{xy}_{is,j} - \alpha_6 u^{xy}_{is+1,j}\right]. \tag{3.13}$$

Similarly, the filter coefficients are found by matching the terms of (3.13) to the explicit fourth-order scheme:

$$\left.\frac{\partial u}{\partial x}\right|^{y}_{i,j} = \frac{1}{12}u^{xy}_{is-2,j} - \frac{5}{4}u^{xy}_{is-1,j} + \frac{5}{4}u^{xy}_{is,j} - \frac{1}{12}u^{xy}_{is+1,j}. \tag{3.14}$$

Following the same procedure explained in the derivation of (3.12) coefficients, the coefficients of (3.14) were derived (see Appendix A.4). Eventually, matching the terms between (3.13) and (3.14) yields the filter coefficients $\alpha_4 = \alpha_6 = \frac{1}{12}$ and $\alpha_5 = \frac{5}{6}$.

3.2.4. Divergence free convective velocities

The property of the convective velocities being divergence-free appears to be of great significance to satisfy the condition of conservation of energy [Verstappen and Veldman, 2003]. In their work, they avoided any direct fourth-order approximation of the convective velocities. Instead, a Richardson extrapolation was used to obtain a fourth order convective flux. Even though mass and energy conservation are guaranteed on momentum cells, yet this method requires 7-Δx stencil in each direction for one momentum cell lowering its spatial resolution properties. On the other hand, [Hokpunna, 2009] proposed a novel fourth-order divergence free interpolation based on a linear combination of the already divergence free cell-average values. A fourth-order Lagrange interpolation is used when the convective velocity is aligned with the momentum cell (e.g. approximation of u for u-momentum cell) as follows:

$$\mathsf{u}^{y}_{i,j} = \lambda_{i,1}u^{y}_{is-2,j} + \lambda_{i,2}u^{y}_{is-1,j} + \lambda_{i,3}u^{y}_{is,j} + \lambda_{i,4}u^{y}_{is+1,j}. \tag{3.15}$$

When the convective velocity is normal to the momentum cell (e.g. approximation of v for u-momentum cell):

$$\mathsf{v}^x_{is,j} = \theta_{is,1} v^x_{i-1,j} + \theta_{is,2} v^x_{i,j} + \theta_{is,3} v^x_{i+1,j} + \theta_{is,4} v^x_{i+2,j}. \tag{3.16}$$

For uniform grid system, the coefficients $(\lambda_{1-4}, \theta_{1-4})$ are identical and equal to those of a fourth-order Lagrange interpolation $\left[-\frac{1}{16}, \frac{9}{16}, \frac{9}{16}, -\frac{1}{16}\right]$. However, for non-uniform grids the λ_{1-4} coefficients were derived as (see Appendix A.5):

$$\begin{aligned}
\lambda_{i,1} &= -\frac{\Delta x_{is} \Delta x_{is-1} \left(2\Delta x_{is} + \Delta x_{is+1}\right)}{\left(\Delta x_{is-1} + \Delta x_{is-2}\right) AB}, \\
\lambda_{i,2} &= \frac{\Delta x_{is} \left(2\Delta x_{is} + \Delta x_{is+1}\right)\left(2\Delta x_{is-1} + \Delta x_{is-2}\right)}{\left(\Delta x_{is} + \Delta x_{is-1}\right)\left(\Delta x_{is-1} + \Delta x_{is-2}\right) C}, \\
\lambda_{i,3} &= \frac{\Delta x_{is-1} \left(2\Delta x_{is} + \Delta x_{is+1}\right)\left(2\Delta x_{is-1} + \Delta x_{is-2}\right)}{\left(\Delta x_{is} + \Delta x_{is-1}\right)\left(\Delta x_{is} + \Delta x_{is+1}\right) B}, \\
\lambda_{i,4} &= -\frac{\Delta x_{is} \Delta x_{is-1} \left(2\Delta x_{is-1} + \Delta x_{is-2}\right)}{\left(\Delta x_{is} + \Delta x_{is+1}\right) AC}, \\
A &= 2\Delta x_{is} + 2\Delta x_{is-1} + \Delta x_{is+1} + \Delta x_{is-2}, \\
B &= \Delta x_{is} + 2\Delta x_{is-1} + \Delta x_{is-2}, \\
C &= 2\Delta x_{is} + \Delta x_{is-1} + \Delta x_{is+1},
\end{aligned}$$

and θ_{1-4} coefficients are defined as [Hokpunna, 2009]:

$$\theta_{is,l} = \left(\sum_{j=1}^{4} \lambda_{i+1,j} - \sum_{j=l+1}^{4} \lambda_{i,j} \right) \frac{\Delta x_{i+l-2}}{\Delta x_{is}}.$$

3.2.5. Convective fluxes with nonlinear correction

The Navier-Stokes equations (2.1) are nonlinear partial differential equations, and the convective term represents the source of nonlinearity. This term is known to be the main contributor to the turbulence phenomenon these equations model, where energy is transported nonlinearly from large to smaller eddies until they reach the dissipation scale. In finite volume methods, the approximation $[\mathbf{u}_j u_i] = [\mathbf{u}_j]\,[u_i]$ is second-order accurate and, therefore, to obtain higher orders, non-linear correction terms are to be added. [Hokpunna, 2009] proposed the use of the already computed edge-average convected and convective velocities as they deliver better resolution properties. Below is the fourth-order edge-average convective flux on the right boundary of a u-momentum cell [Hokpunna, 2009]:

$$[\mathsf{u}u]^y_{i+1,j} = \mathsf{u}^y_{i+1,j} u^y_{i+1,j} + \frac{1}{48}\left(u^y_{i+1,j+1} - u^y_{i+1,j-1}\right)^2. \tag{3.17}$$

Three different equations analogous to (3.17) can be obtained to evaluate the convective fluxes at the left boundary of a u-momentum cell and the top/bottom boundaries of a v-momentum cell with special care given to indices. On the other hand, the following expression is written for the convective flux at the top edge of a u-momentum cell and could be used to write those at the bottom boundary of a u-momentum cell and right/left boundaries of a v-momentum cell [Hokpunna, 2009]:

$$[\mathsf{v}u]^{x}_{is,js} = \mathsf{v}^{x}_{is,js} u^{x}_{is,js} + \frac{1}{24}\left(u^{y}_{is+1,js} - u^{y}_{is-1,js}\right)\left(v^{xy}_{i+1,js} - v^{xy}_{i,js}\right). \tag{3.18}$$

3.2.6. Formal fourth-order discrete Laplacian operator

Keeping in mind that the discrete divergence operator is applied over pressure-cells whereas the discrete gradient operator is applied over momentum-cells. Employing (3.6) and (3.7), the discrete fourth-order x-component of the Laplacian operator can be written for uniform grids as (see Appendix A.6 for derivation) [Hokpunna, 2009]:

$$\left.\frac{\partial^2 p}{\partial x^2}\right|^{xy}_{i,j} = \frac{p^{xy}_{i\pm 3,j} - 54p^{xy}_{i\pm 2,j} + 783p^{xy}_{i\pm 1,j} - 1460p^{xy}_{i,j}}{576\;\Delta x^2}. \tag{3.19}$$

On non-uniform grids, this component could be derived by considering the non-uniform set of coefficients in (3.6) and (3.7) when writing the discrete divergence and gradient operators:

$$D_{4x} = \frac{\mathrm{eq}(3.7)_{\text{at }(is,j)} - \mathrm{eq}(3.7)_{\text{at }(is-1,j)}}{x_{is,j} - x_{is-1}} \quad \text{and} \quad G_{4x} = \frac{\mathrm{eq}(3.6)_{\text{at }(i+1,j)} - \mathrm{eq}(3.6)_{\text{at }(i,j)}}{x_{i+1} - x_{i}}.$$

3.3. Validation and Numerical Test Cases

When validation is applied to a flow solver, the intention of this process is to justify that the code is capable of properly solving the governing mathematical equations with the desired order of accuracy built in the numerical approximations. Therefore, a verified code has very high chances of being free of implementation errors as well as approximations errors. To test the order of spatial and time convergence of the discretization schemes introduced in the previous chapters, a well known benchmark test case is used .

3.3.1. Classical and Convected Taylor-Green vortex test cases

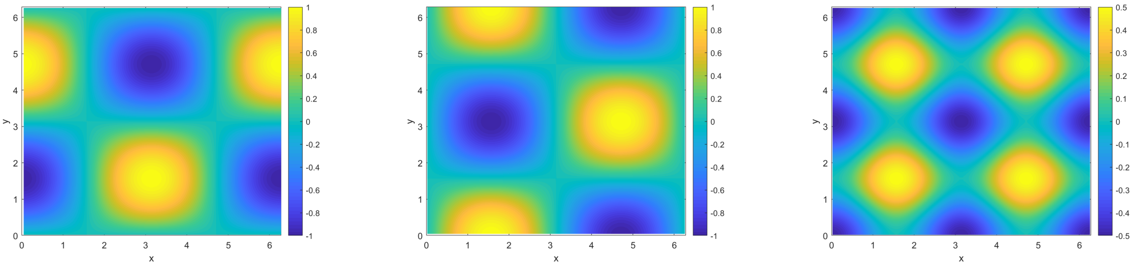
Taylor-Green vortex flow is an unsteady flow describing the decay of a two-dimensional periodic vortex in a square domain. This flow has an exact closed solution of the incompressible Navier-Stokes equations, the thing which made it serve as a benchmark problem for testing the order of accuracy of various discretization schemes. The Taylor-Green vortex (TGV) solution is given by:

$$u(x,y,t,\nu) = c_1 - cos(x - c_1 t) sin(y) e^{-2\nu t}, \tag{3.20}$$

$$v(x,y,t,\nu) = c_2 + sin(x - c_1 t) cos(y) e^{-2\nu t}, \tag{3.21}$$

$$p(x,y,t,\nu) = -\frac{1}{4}\left(cos(2(x - c_1 t)) + cos(2(y - c_2 t))\right) e^{-4\nu t}. \tag{3.22}$$

The computational domain is set to $(x,y) \in [0,2\pi]^2$ and the flow is initialized using equations (3.20)-(3.22) with $t = 0$ as shown Fig.3.2. It is worth mentioning that when testing the fourth-order implementation, the fields are initialized with and compared against averaged quantities along each cell.



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

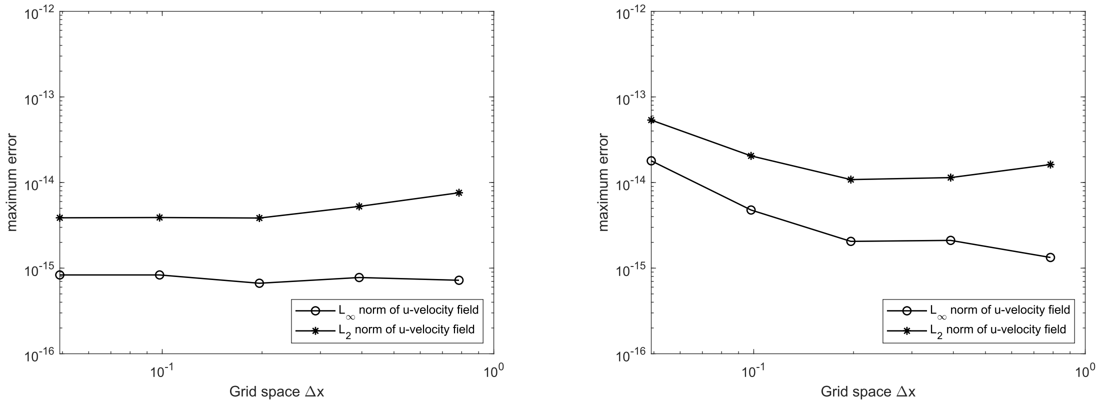
Figure 3.2: Fields at $t = 0s$ (a) u-velocity field (b) v-velocity field (c) pressure field

Spatial convergence:
In the classical TGV (with $c_1 = c_2 = 0$, $Re = 100$), only viscous diffusion occurs and therefore the only accuracy demonstrated will be of the diffusive term. This argument can be justified when the flow is invscid, where the flow turns to be steady as the convective and pressure terms will cancel out. This can be shown by comparing the maximum error of a velocity field (here u-velocity) computed for the invscid TGV by second and fourth-order schemes. Fig 3.3 shows that both schemes deliver errors nearly at machine precision, hence the classical TGV is unable to evaluate the accuracy of the convective and pressure terms

and only restricted to that of the diffusive term. Essentially, this represents the reason behind introducing the convective velocity (c_1, c_2) to the TGV solution, where the inviscid convected TGV (with $\nu = c_2 = 0,\ c_1 = 1$) is used to test the convergence of the convective term. To facilitate the convergence study, the measures of error used in this section are the L_2 and the L_∞ norms of error defined as:

$$\|e_u\|_2 = \sqrt{\Delta x \Delta y \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \left(u_{i,j} - u_{i,j}^{exact}\right)^2}, \tag{3.23}$$

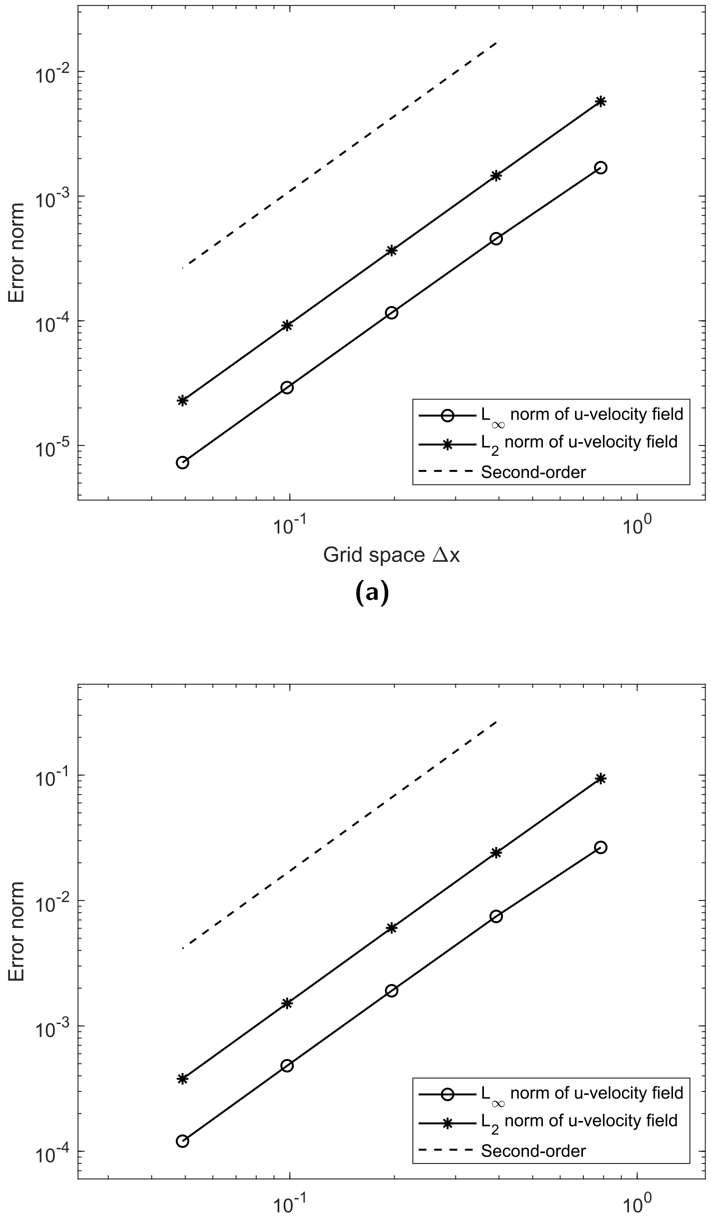
$$\|e_u\|_\infty = max|u_{i,j} - u_{i,j}^{exact}|. \tag{3.24}$$



(a) **(b)**

Figure 3.3: Maximum error of inviscid TGV in (a) Second-order scheme (b) fourth-order scheme.

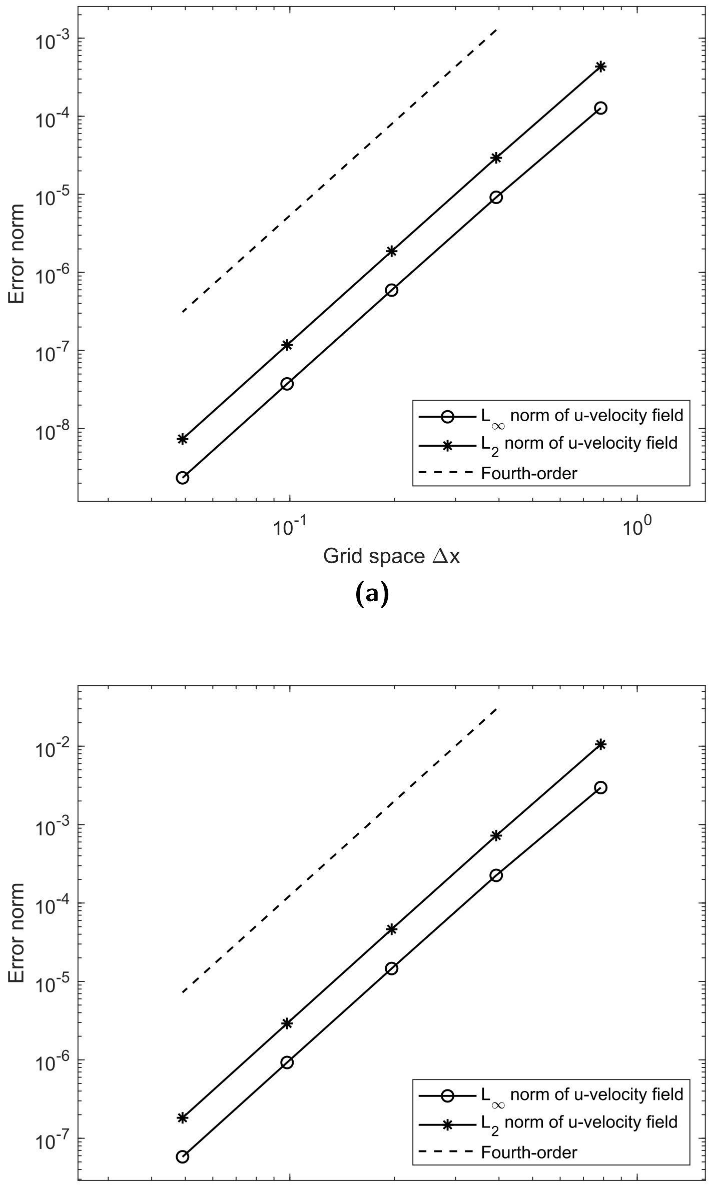
Proceeding to the results of the spatial convergence study, setting the viscosity to $\nu = \frac{2\pi}{100}$, the characteristic length L equal to the size of the computational domain (2π) and the velocity equal to the maximum velocity recorded in the field, the Reynolds number is then $Re = 100$. The solution is marched in time to $t = 0.3$ with a time step of $\Delta t = 0.001$ and the norms of the error are recorded for each grid refinement step. Fig 3.4 shows the convergence rates of the second-order scheme and Fig 3.5 shows those of the fourth-order scheme. Both schemes converges at the intended rate of convergence with higher error magnitudes recorded for the convected TGV case. Finally, the order of convergence of the non-uniform grid implementation is shown in Fig 3.6.



(a)

(b)

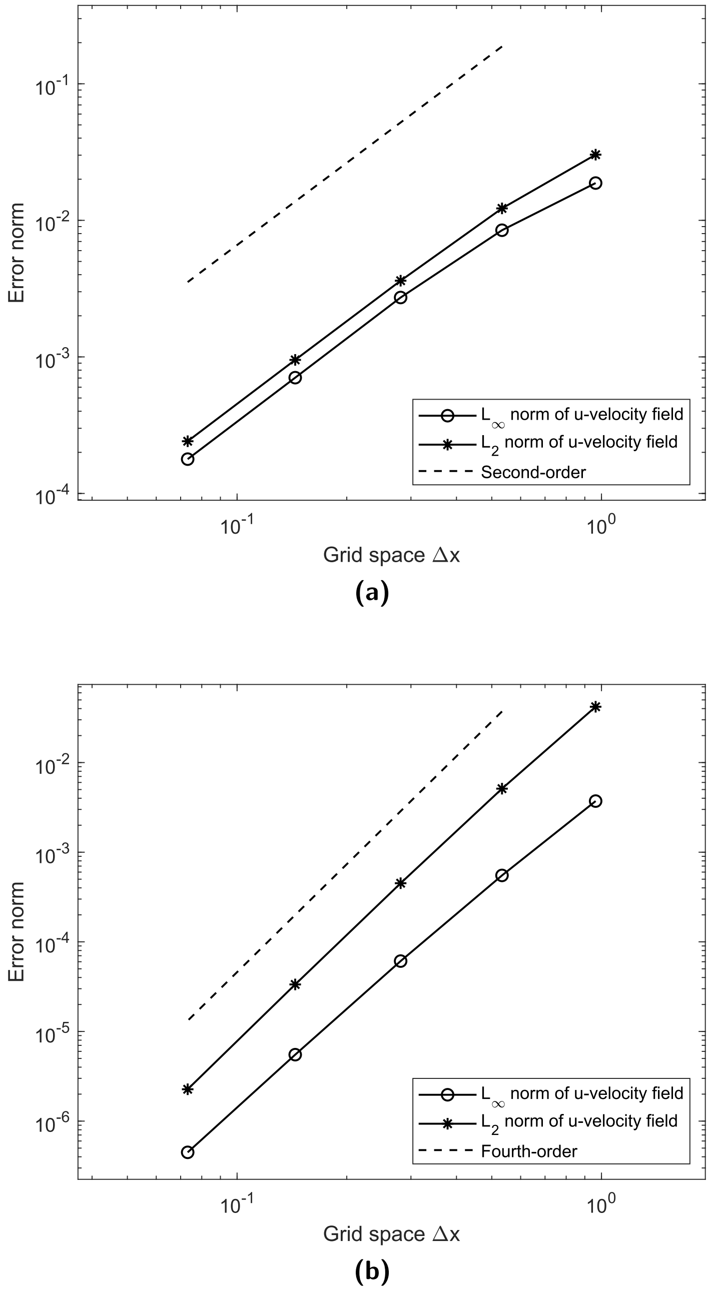
Figure 3.4: Convergence rates of the second-order scheme for (a) Classic TGV (b) Convected TGV.



(a)

(b)

Figure 3.5: Convergence rates of the fourth-order scheme for (a) Classic TGV (b) Convected TGV.



(a)

(b)

Figure 3.6: Convergence rates of schemes at non-uniform staggered grids. (a) second-order (b) fourth-order.

Temporal convergence:
As already was mentioned in 2.4, the time integration scheme used to march the solution in time is the third-order low storage explicit Runge-Kutta (RK3) scheme of [Williamson, 1980]. This scheme is third-order accurate for the velocity and first-order accurate for the pressure. The difference between the the numerical and the exact analytical solution represents the error which is essentially composed of two components, namely, the temporal error and the spatial error. Depending on the size of the time step, it might occur that the latter exceeds the former significantly and the error becomes independent of the time step size. Therefore, it is more convenient to eliminate the spatial error when verifying the convergence of the implemented time integration schemes. One way to do that is to take ratios of differences between the numerical solution computed for a successively halved time step. This can be demonstrated by letting $\tilde{u}_{\Delta t}$ be the numerical solutions evaluated by means of some time step Δt and a time integration scheme of p-order of accuracy, then we can write:

$$\begin{aligned}
\tilde{u}_{\Delta t} &= u_{exact} + C\Delta t^p, \\
\tilde{u}_{\Delta t/2} &= u_{exact} + C\left(\frac{\Delta t}{2}\right)^p, \\
\tilde{u}_{\Delta t/4} &= u_{exact} + C\left(\frac{\Delta t}{4}\right)^p,
\end{aligned}$$

where C is some constant. Now, taking the ratios of differences we get:

$$\frac{\tilde{u}_{\Delta t} - \tilde{u}_{\Delta t/2}}{\tilde{u}_{\Delta t/2} - \tilde{u}_{\Delta t/4}} = \frac{C\Delta t^p - C\left(\frac{\Delta t}{2}\right)^p}{C\left(\frac{\Delta t}{2}\right)^p - C\left(\frac{\Delta t}{4}\right)^p} = \frac{1-2^{-p}}{2^{-p}-2^{-2p}} = 2^p.$$

Then once these ratios are plotted in a logarithmic scale, the rate of convergence is obtained as shown in Fig 3.7.

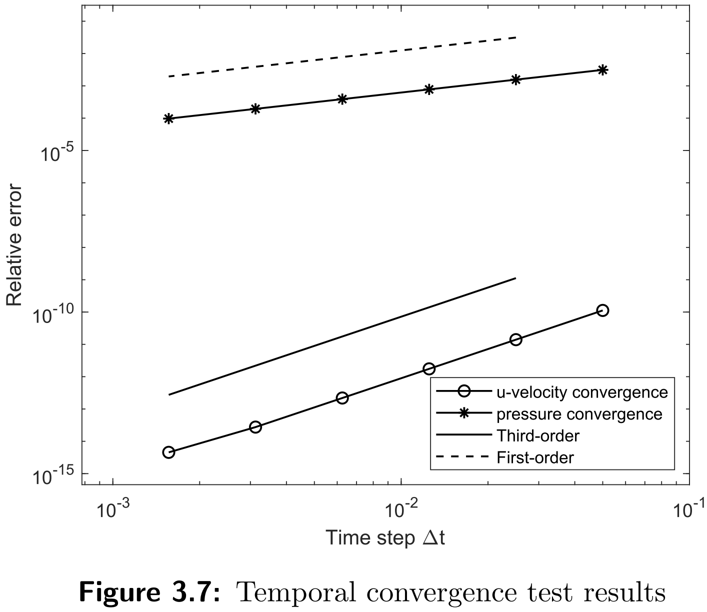


Figure 3.7: Temporal convergence test results

4. The Immersed Boundary Method

This chapter is devoted for introducing the Immersed Boundary Method with the direct continuous forcing approach followed. First the mathematical formulation along the with essential definitions of variables is presented. Then, the equations describing the fluid-solid coupling are presented. Next, a derivation of a new fourth-order interpolation function is explained with a mathematical proof. Finally, numerical test cases are conducted to test the accuracy of the implemented IB solver.

This chapter contains frequently-used new terms which require to be introduced. This can be done best by means of the schematic shown in Fig 4.1 along with the following description. We separately discretize the computational domain Ω with a uniform fixed Cartesian grid g_h, where $\mathbf{x}_{i,j}$ are the computational Eulerian nodes. Moreover, we define the immersed boundary embedded into the fluid by the continuous curve Γ and discretize it with a set of uniformly distributed points N_L. These points are so called Lagrangian force points and their coordinates are to be used in the interpolation and spreading steps discussed later. The coordinates of the force points are denoted by:

$$\mathbf{X}_l \in \Gamma \qquad \forall \quad 1 \leq l \leq N_L,$$

and from now on, in order to differentiate between the Eulerian quantities evaluated at the Eulerian points ($\mathbf{x}$) and their Lagrangian counterparts evaluated at the Lagrangian points ($\mathbf{X}$), we shall use upper case letters for the latter one.

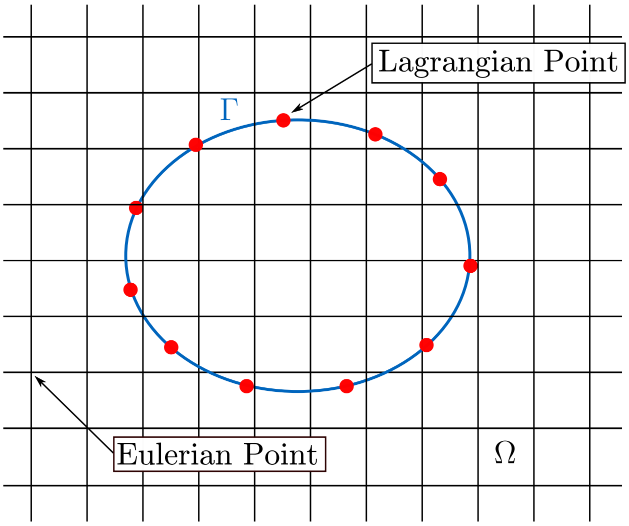


Figure 4.1: Eulerian and Lagrangian locations.

4.1. Mathematical formulation

Although the immersed boundary method is not confined to viscous incompressible fluids, in the scope of this work we will only consider fluids that can be described by the incompressible NS equations. For convenience, we rewrite the NS equations below:

$$\left.\begin{aligned} \frac{\partial \mathbf{u}}{\partial t} &= (\mathbf{u}\cdot\nabla)\,\mathbf{u} + \nu\Delta\mathbf{u} - \tfrac{1}{\rho}\nabla p + f \\ \nabla\cdot\mathbf{u} &= 0 \end{aligned}\right\} \qquad \mathbf{x}\in\Omega,$$

where f is a volume source term that will be used as described in the Continuous forcing approach, i.e to represent the influence of the solid object on the surrounding fluid. These equations will be enforced throughout the whole computational domain Ω including the area occupied by the immersed boundary. The strategy followed here for the formulation of the fluid-solid interaction force was presented by [Uhlmann, 2005]. In his work, he combined a direct and explicit forcing method with Peskin's regularized delta function approach known for its capability of performing smooth transfer of quantities between the Lagrangian and Eulerian positions, while at the same time avoiding strong restrictions of the time step. What follows next is a description of volume forces formulation.

For the purpose of presenting the method adopted, let the time discretization of the NS equations be given by:

$$\frac{\mathbf{u}^{n+1}-\mathbf{u}^{n}}{\Delta t} = \mathbf{rhs}^{n+1/2} + \mathbf{f}^{n+1/2},$$

where $\mathbf{rhs}^{n+1/2}$ is a short-hand for the collection of convective, diffusive and pressure terms. One way to find the force $\mathbf{f}^{n+1/2}$ arising from directly imposing the desired value of the velocity $\mathbf{u}_d$ on the boundary is to simply write [Fadlun et al., 2000]:

$$\mathbf{f}^{n+1/2} = \frac{\mathbf{u}_d - \mathbf{u}^{n}}{\Delta t} - \mathbf{rhs}^{n+1/2} \tag{4.1}$$

Due to the fact that the Eulerian and Lagrangain points do not usually coincide, [Fadlun et al., 2000] suggested three interpolation procedures with an ascending order of accuracy when this direct forcing approach is used. The first was to assign the force in (4.1) to all Eulerian points closest to the immersed boundary as if they were coincident to it. This resulted in a diffused interface rather than a sharp one. The second interpolation procedure was an improved version of the first where volume fraction weighting was considered to scale the forcing applied to the momentum-cells closest to the boundary. The last proposition was to linearly interpolate the velocity of the immersed boundary $\mathbf{u}_d$ to the Eulerian points closest to the boundary to obtain $\bar{\mathbf{u}}_d$ and substitute it in (4.1) instead of $\mathbf{u}_d$. However, [Uhlmann, 2005] reported that the use of this direct forcing approach might cause serious oscillations of the hydrodynamical forces due to inadequate smoothing. Therefore, we follow the direct forcing scheme proposed by [Uhlmann, 2005] where the force term is evaluated at the Lagrangian force points $\mathbf{X}_l$:

$$\mathbf{F}^{n+1/2} = \frac{\mathbf{U}_d - \mathbf{U}^{n}}{\Delta t} - \mathbf{RHS}^{n+1/2} \qquad \forall \quad \mathbf{X}_l, \tag{4.2}$$

with $\mathbf{U}_d$ being the velocity at the fluid-solid interface. Although this velocity could be let free to be determined by Newton's equations for some specific applications, it is however explicitly prescribed by the rigid body motion of the solid object in the test cases considered in this work. This velocity is composed of linear and rotational parts as:

$$\mathbf{U}_d\left(\mathbf{X}_l\right) = \mathbf{U}_c + \mathbf{W}_c \times \left(\mathbf{X}_l - \mathbf{X}_c\right), \tag{4.3}$$

where $\mathbf{U}_c$, $\mathbf{W}_c$ and $\mathbf{X}_c$ are the linear velocity, angular velocity and the coordinates of the object's centroid, respectively.
In (4.2), $\mathbf{U}^n$ and $\mathbf{RHS}^{n+1/2}$ can both be combined into a single unknown, namely, the intermediate velocity at a Lagrangian location without adding the force term $\mathbf{F}$:

$$\mathbf{U}^* = \mathbf{U}^n + \mathbf{RHS}^{n+1/2}\Delta t \quad \longrightarrow \quad \mathbf{U}^n = \mathbf{U}^* - \mathbf{RHS}^{n+1/2}\Delta t. \tag{4.4}$$

Substituting (4.4) into (4.2) yields the following new expression of the Lagrangian force:

$$\mathbf{F}^{n+1/2} = \frac{\mathbf{U}_d - \mathbf{U}^*}{\Delta t} \qquad \forall \quad \mathbf{X}_l. \tag{4.5}$$

To close the required coupling terms , it is worth mentioning that an Eulerian analogue of (4.4) exists in the projection method 2.4 used in this work. Up to this point, we can conclude the mathematical formulation of the problem defining the Eulerian and Lagrangian variables, and we can proceed with the next task of presenting the interaction equations of the numerical scheme.

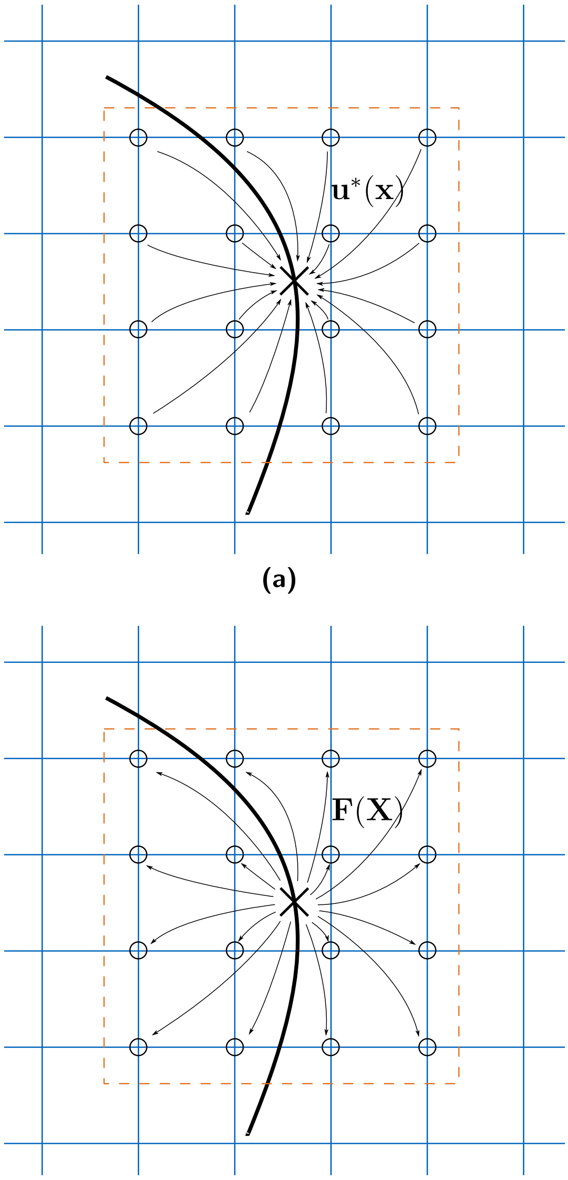
Interaction equations

In order to build a proper communication between the fluid and the solid object, one has to interpolate the coupling quantities ($\mathbf{u}^*, \mathbf{U}^*, \mathbf{F}$ and $\mathbf{f}$) back and forth between these two phases. The method used to achieve this task is the original IB method introduced by [Peskin, 1972], where the idea was built on the use of a bounded continuous approximation δ_h of a Dirac delta function δ to link the quantities between the two frames of reference. Restricting ourselves to the two-dimensional spatial implementation of this work, the interaction equations read [Uhlmann, 2005]:

$$\mathbf{U}^*(\mathbf{X}_l) = \sum_{\mathbf{x}\in g_h} \mathbf{u}^*(\mathbf{x})\delta_h\left(\mathbf{x} - \mathbf{X}_l\right)h^2 \qquad \forall\, 1 \leq l \leq N_L, \tag{4.6}$$

$$\mathbf{f}(\mathbf{x}) = \sum_{l=1}^{N_L} \mathbf{F}\left(\mathbf{X}_l\right)\delta_h\left(\mathbf{x} - \mathbf{X}_l\right)\Delta A_l \qquad \forall\, \mathbf{x} \in g_h. \tag{4.7}$$

The operation performed by equation (4.6) takes an Eulerian input $\mathbf{u}^*(\mathbf{x})$ and generates a Lagrangian output $\mathbf{U}^*(\mathbf{X}_l)$. We shall call this the interpolation operation, owing to the fact that it computes the Lagrangian quantity as a weighted average of the neighbouring Eulerian quantities as shown in Fig 4.2a. Alternatively, equation (4.7) takes a Lagrangian input $\mathbf{F}$ and generates an Eulerian output $\mathbf{f}$. This operation is called the spreading operation, where the force is spread out from a Lagrangian force point to the Eulerian points surrounding it as shown in Fig 4.2b. A detailed presentation of the explicit formulation of the coupling force within each RK sub-step is shown in Algorithm 2.



(a)

(b)

Figure 4.2: (a) Interpolation of Eulerian velocities. (b) Spreading of a Lagrangian force.

Algorithm 2: RK3 time integration with IBM

```
initialize u, v and p fields
$n \leftarrow 1$
while $n <$ number of time steps do

    First RK sub-step:
    $\mathbf{RHS}^1 \leftarrow \mathbf{F}^c(\mathbf{u}^n) + \mathbf{F}^d(\mathbf{u}^n) + \mathbf{F}^p(p^n)$
    $\mathbf{u}^n \leftarrow \mathbf{u}^n + \frac{1}{3}\,\Delta t\ \mathbf{RHS}^1$                                    // Intermediate velocity 1
    $\mathbf{U}(\mathbf{X}_l) \leftarrow \sum_{\mathbf{x}\in g_h} \mathbf{u}^n(\mathbf{x})\delta_h\left(\mathbf{x}-\mathbf{X}_l\right)h^2 \quad \forall\ N_L$          // Interpolate velocity
    $\mathbf{F}(\mathbf{X}_l) \leftarrow \rho\ \frac{\mathbf{U}^d-\mathbf{U}}{\frac{1}{3}\ \Delta t}$                                          // Evaluate forces
    $\mathbf{f}(\mathbf{x}) \leftarrow \sum_{l=1}^{N_L} \mathbf{F}\left(\mathbf{X}_l\right)\delta_h\left(\mathbf{x}-\mathbf{X}_l\right)\Delta A_l \quad \forall\ \mathbf{x}$                 // Spread forces
    $\mathbf{u}^n \leftarrow \mathbf{u}^n + \frac{\frac{1}{3}\ \Delta t}{\rho}\ \mathbf{f}(\mathbf{x})$                                // Intermediate velocity 2
    $\Delta dp \leftarrow \frac{\rho}{\frac{1}{3}\ \Delta t}\ \nabla\cdot\mathbf{u}^n$                                          // Correct pressure
    $\mathbf{u}^n \leftarrow \mathbf{u}^n - \frac{\frac{1}{3}\ \Delta t}{\rho}\ \nabla dp$                                     // Project fields
    $p^n \leftarrow p^n + \nabla dp$

    Second RK sub-step:
    $\mathbf{RHS}^2 \leftarrow \mathbf{F}^c(\mathbf{u}^n) + \mathbf{F}^d(\mathbf{u}^n) + \mathbf{F}^p(p^n)$
    $\mathbf{RHS}^2 \leftarrow \mathbf{RHS}^2 - \frac{5}{9}\ \mathbf{RHS}^1$
    $\mathbf{u}^n \leftarrow \mathbf{u}^n + \frac{15}{16}\ \Delta t\ \mathbf{RHS}^2$                                   // Intermediate velocity 1
    $\mathbf{U}(\mathbf{X}_l) \leftarrow \sum_{\mathbf{x}\in g_h} \mathbf{u}^n(\mathbf{x})\delta_h\left(\mathbf{x}-\mathbf{X}_l\right)h^2 \quad \forall\ N_L$          // Interpolate velocity
    $\mathbf{F}(\mathbf{X}_l) \leftarrow \rho\ \frac{\mathbf{U}^d-\mathbf{U}}{\frac{3}{4}\ \Delta t}$                                          // Evaluate forces
    $\mathbf{f}(\mathbf{x}) \leftarrow \sum_{l=1}^{N_L} \mathbf{F}\left(\mathbf{X}_l\right)\delta_h\left(\mathbf{x}-\mathbf{X}_l\right)\Delta A_l \quad \forall\ \mathbf{x}$                 // Spread forces
    $\mathbf{u}^n \leftarrow \mathbf{u}^n + \frac{\frac{3}{4}\ \Delta t}{\rho}\ \mathbf{f}(\mathbf{x})$                                // Intermediate velocity 2
    $\Delta dp \leftarrow \frac{\rho}{\frac{3}{4}\ \Delta t}\ \nabla\cdot\mathbf{u}^n$                                          // Correct pressure
    $\mathbf{u}^n \leftarrow \mathbf{u}^n - \frac{\frac{3}{4}\ \Delta t}{\rho}\ \nabla dp$                                     // Project fields
    $p^n \leftarrow p^n + \nabla dp$

    Third RK sub-step:
    $\mathbf{RHS}^1 \leftarrow \mathbf{F}^c(\mathbf{u}^n) + \mathbf{F}^d(\mathbf{u}^n) + \mathbf{F}^p(p^n)$
    $\mathbf{RHS}^1 \leftarrow \mathbf{RHS}^1 - \frac{153}{128}\ \mathbf{RHS}^2$
    $\mathbf{u}^n \leftarrow \mathbf{u}^n + \frac{8}{15}\ \Delta t\ \mathbf{RHS}^1$                                    // Intermediate velocity 1
    $\mathbf{U}(\mathbf{X}_l) \leftarrow \sum_{\mathbf{x}\in g_h} \mathbf{u}^n(\mathbf{x})\delta_h\left(\mathbf{x}-\mathbf{X}_l\right)h^2 \quad \forall\ N_L$          // Interpolate velocity
    $\mathbf{F}(\mathbf{X}_l) \leftarrow \rho\ \frac{\mathbf{U}^d-\mathbf{U}}{\Delta t}$                                              // Evaluate forces
    $\mathbf{f}(\mathbf{x}) \leftarrow \sum_{l=1}^{N_L} \mathbf{F}\left(\mathbf{X}_l\right)\delta_h\left(\mathbf{x}-\mathbf{X}_l\right)\Delta A_l \quad \forall\ \mathbf{x}$                 // Spread forces
    $\mathbf{u}^n \leftarrow \mathbf{u}^n + \frac{\Delta t}{\rho}\ \mathbf{f}(\mathbf{x})$                                     // Intermediate velocity 2
    $\Delta dp \leftarrow \frac{\rho}{\Delta t}\ \nabla\cdot\mathbf{u}^n$                                               // Correct pressure
    $\mathbf{u}^{n+1} \leftarrow \mathbf{u}^n - \frac{\Delta t}{\rho}\ \nabla dp$                                      // Project fields
    $p^{n+1} \leftarrow p^n + \nabla dp$

    $n \leftarrow n+1$
```

As the regularized delta function δ_ϵ (that will soon make its appearance) plays a prominent role in the interaction equations (4.6) and (4.7), its construction should be carried out carefully as it has a considerable influence on the solution behavior and accuracy. There exist different regularized delta functions in the literature such as the cosine shape delta by [Peskin, 1972]:

$$\phi(r) = \begin{cases} \frac{1}{4}\left(1 + cos(\frac{\pi r}{2})\right), & \text{if } |r| \leq 2 \\ 0, & \text{otherwise,} \end{cases} \tag{4.8}$$

the three-point delta by [Roma et al., 1999]:

$$\phi(r) = \begin{cases} \frac{1}{3}\left(1 + \sqrt{-3r^2 + 1}\right), & \text{if } |r| \leq \frac{1}{2} \\ \frac{1}{6}\left(5 - 3r - \sqrt{-3(1 - r^2 + 1)}\right), & \text{if } \frac{1}{2} \leq |r| \leq \frac{3}{2} \\ 0, & \text{otherwise,} \end{cases} \tag{4.9}$$

and the four-point delta by [Peskin, 2002]:

$$\phi(r) = \begin{cases} \frac{1}{8}\left(3 - 2r + \sqrt{1 + 4r - 4r^2}\right), & \text{if } |r| \leq 1 \\ \frac{1}{8}\left(5 - 2r - \sqrt{-7 + 12r - 4r^2}\right), & \text{if } 1 \leq |r| \leq 2 \\ 0, & \text{otherwise,} \end{cases} \tag{4.10}$$

depicted in Fig 4.3. These functions were derived by satisfying a set of discrete compatibility conditions which will be introduced in the next section. Both kernels (4.9) and (4.10) have a second-order accuracy when used with smooth function fields, hence serve as a good choice to use in cooperation with the second-order finite volume scheme in section 3.1. The remaining task is to seek a regularized delta function capable of interpolating fields at a fourth-order accuracy and to be used with the higher order implemented scheme.

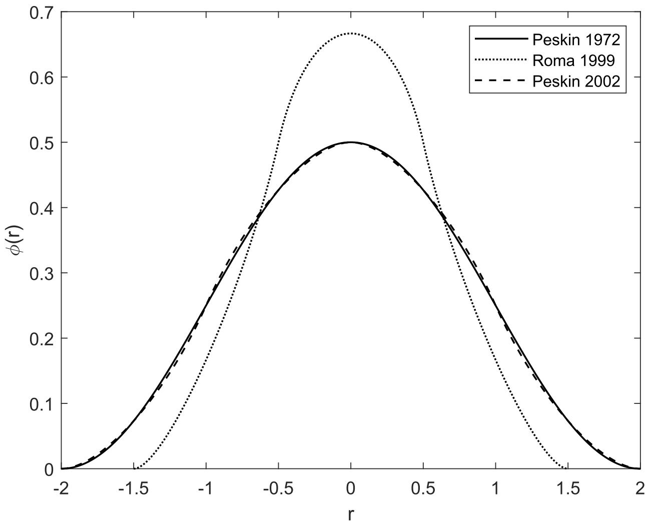


Figure 4.3: Regularized delta functions from the literature

4.2. Derivation of fourth-order regularized delta-function

This section is devoted for the derivation of a regularized delta function with a fourth-order accuracy inspired by [Tornberg and Engquist, 2004]. In their work, they showed that one can design a higher order regularized delta function by satisfying higher discrete moment conditions. Letting δ_ϵ be a one-dimensional regularization of the Dirac delta function δ with a compact support $[-\epsilon, \epsilon]$, and q being the number of moments satisfied, the discrete moment conditions can be written as:

$$M_r\left(\delta_\epsilon, X, h\right) = h \sum_{j=-\infty}^{\infty} \delta_\epsilon\left(x_j - X\right)\left(x_j - X\right)^r = \begin{cases} 1 & \text{if } r = 0 \\ 0 & \text{if } 1 \leq r \leq q \end{cases} \tag{4.11}$$

The zeroth moment (M_0) ensures that the integral over δ_ϵ is unity, and yields the exact interpolation of constant functions.
In order to show the dependency between the numerical order of accuracy and the number of moments a regularized delta satisfies, one can write the error as [Tornberg and Engquist, 2004]:

$$E = \left| h \sum_{j=-\infty}^{\infty} \delta_\epsilon\left(x_j - X\right) \overline{f(x_j)} - f(X) \right| \leq Ch^q. \tag{4.12}$$

In order to be consistent with the fourth-order finite volume scheme implemented in this work, it is worth mentioning that the interpolated values $\overline{f(x_j)}$ are cell-averaged quantities and not cell-center point values. On uniform grids, a proof of (4.12) can be carried out by taking the Taylor series expansion of $\overline{f(x_j)}$ around X:

$$h \sum_{j=-\infty}^{\infty} \delta_\epsilon\left(x_j - X\right) \overline{f(x_j)} = h \sum_{j=-\infty}^{\infty} \delta_\epsilon\left(x_j - X\right) \frac{1}{h} \int_{x_j - \frac{h}{2}}^{x_j + \frac{h}{2}} f(x)\, dx, \tag{4.13}$$

with,

$$\frac{1}{h} \int_{x_j - \frac{h}{2}}^{x_j + \frac{h}{2}} f(x)\, dx = \frac{1}{h} \int_{x_j - \frac{h}{2}}^{x_j + \frac{h}{2}} f(X) + (x - X) \frac{df}{dx}\Big|_X + \frac{(x-X)^2}{2!} \frac{d^2 f}{dx^2}\Big|_X + \\ \ldots + \frac{(x-X)^{q-1}}{(q-1)!} \frac{d^{(q-1)} f}{dx^{(q-1)}}\Big|_X + O(h^q)\, dx. \tag{4.14}$$

Choosing $q = 4$ and evaluating the integral up to the $(q-1)^{th}$ derivative and substituting that into (4.13) gives:

$$h \sum_{j=-\infty}^{\infty} \delta_\epsilon\left(x_j - X\right) \overline{f(x_j)} = h \sum_{j=-\infty}^{\infty} \delta_\epsilon\left(x_j - X\right) \Bigg[\underbrace{1}_{\xi_{0,i}} f(X) + \underbrace{(x_j - X)}_{\xi_{1,i}} \frac{df}{dx}\Big|_X + \\ \underbrace{\left(\frac{1}{2}(x_j - X)^2 + \frac{h^2}{24}\right)}_{\xi_{2,i}} \frac{d^2 f}{dx^2}\Big|_X + \underbrace{\left(\frac{1}{6}(x_j - X)^3 + \frac{h^2}{24}(x_j - X)\right)}_{\xi_{3,i}} \frac{d^3 f}{dx^3}\Big|_X \Bigg] + O(h^4). \tag{4.15}$$

Using the definition of discrete moment conditions (4.11), equation (4.15) can be simplified to:

$$h\sum_{j=-\infty}^{\infty}\delta_\epsilon\left(x_j-X\right)\overline{f(x_j)} = M_0(\delta_\epsilon, X, h)f(X) \;+\; M_1(\delta_\epsilon, X, h)\frac{df}{dx}\Big|_X + \widetilde{M_2}(\delta_\epsilon, X, h)\frac{d^2f}{dx^2}\Big|_X \;+\; \widetilde{M_3}(\delta_\epsilon, X, h)\frac{d^3f}{dx^3}\Big|_X + O(h^4). \quad (4.16)$$

Where the moments with tilde are the modified moment conditions originating from the fact that the quantities interpolated are cell-averaged quantities. One can now see that satisfying four moment conditions results in a regularized delta with a fourth-order interpolating order of accuracy. The satisfaction of the first, second and third moment conditions enables the the regularized delta to exactly interpolate linear, quadratic and cubic smooth functions, respectively.

Regarding the minimal compact support required to achieve the desired order of accuracy, [Tornberg and Engquist, 2004] proofed that for δ_ϵ to be of q^{th}-order, its compact support 2ϵ has to be greater than or equal to qh. Seeking an overall fourth-order accuracy as well as minimal computational effort, the compact support was chosen as $2\epsilon = 4h$. This implies that exactly q Eulerian points are in the support of δ_ϵ.

The remaining concern is the unique evaluation of the fourth-order δ_ϵ at these Eulerian points. Defining the Eulerian points as $x_j = x_1, ..., x_4$ to be within the $2h$ neighborhood of a Lagrangian force point X, and the regularized delta to be:

$$\delta_\epsilon = \frac{1}{h}\phi(r_j = \frac{x_j - X}{h}),$$

where ϕ is a continuous function. One can find $\phi(r)$ by solving the following linear system of equations:

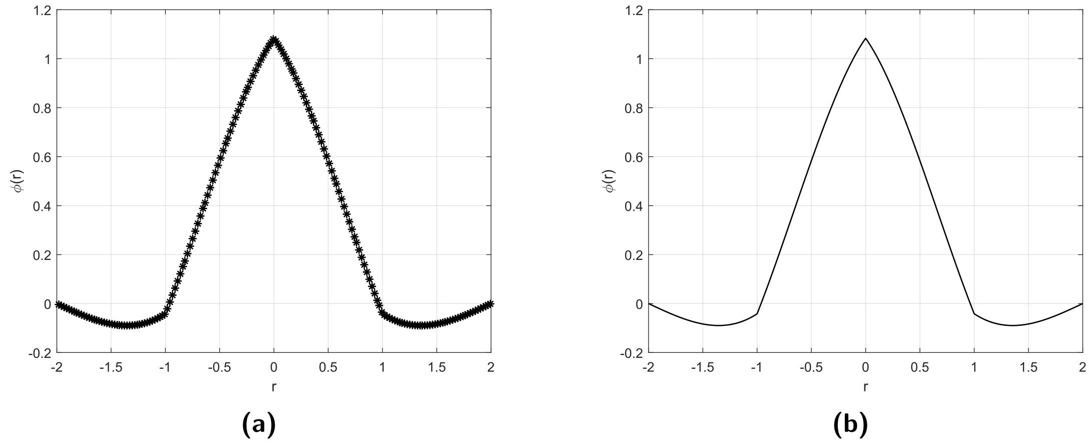
$$\mathbf{\Xi\Phi} = \mathbf{c}, \quad (4.17)$$

where each term is written as:

$$\underbrace{\begin{bmatrix} 1 & 1 & 1 & 1 \\ \xi_{1,1} & \xi_{1,2} & \xi_{1,3} & \xi_{1,4} \\ \xi_{2,1} & \xi_{2,2} & \xi_{2,3} & \xi_{2,4} \\ \xi_{3,1} & \xi_{3,2} & \xi_{3,3} & \xi_{3,4} \end{bmatrix}}_{\mathbf{\Xi}} \underbrace{\begin{bmatrix} \phi(r_1) \\ \phi(r_2) \\ \phi(r_3) \\ \phi(r_4) \end{bmatrix}}_{\mathbf{\Phi}} = \underbrace{\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}}_{\mathbf{c}}.$$

The linear system of equations (4.17) gives access to all points of the kernel $\phi(r)$ and one could fit a curve to derive its expression. It was observed in figure 3.2a that ϕ is symmetric ($\phi(-r) = \phi(r)$) and has a discontinuous derivative at $r = 1$. Based on these two observations and upon the evaluation of derivatives and y-intercepts, $\phi(r)$ shown in figure 3.2b was found to be a cubic piece-wise function written as:

$$\phi(r) = \begin{cases} \frac{13}{12} - \frac{5}{8}r - r^2 + \frac{1}{2}r^3, & \text{if } |r| \le 1 \\ \frac{11}{12} - \frac{43}{24}r + r^2 - \frac{1}{6}r^3, & \text{if } 1 \le |r| \le 2 \\ 0, & \text{otherwise} \end{cases} \quad (4.18)$$



(a) **(b)**

Figure 4.4: (a) Multiple evaluations of the linear system of equations (4.17). (b) Derived regularized delta function with modified moment conditions.

We can now summarize the set of conditions satisfied by the regularized delta function and their influence on its properties:

1. $\boxed{\phi(r) \text{ is continuous}}$: this ensures that the coefficients of interpolation and spreading vary continuously, thus no jumps in velocities or forces can take place.

2. $\boxed{\phi(r) = 0 \ \forall \ |r| \geq 2}$: i.e the delta function must be of a finite support. This condition is introduced to keep the computations of the IBM reasonable, whereas the use of unbounded functions will entail a prohibitive computational cost since each Lagrangian force point will interact with Eulerian grid points.

3. $\boxed{\sum_{i=-\infty}^{\infty} \phi(r-i) = 1 \ \forall \ r}$: this condition guarantees the exact interpolation of constant functions with the physical implication that the total force added to the fluid is not amplified or damped spuriously in the transfer step. This can be shown by:

$$\sum_{\mathbf{x}\in g_h} \mathbf{f}(\mathbf{x}) h^2 = \sum_{\mathbf{x}\in g_h} \sum_{l=1}^{N_L} \mathbf{F}\left(\mathbf{X}_l\right) \delta_h \left(\mathbf{x} - \mathbf{X}_l\right) \Delta A_l h^2,$$

$$\sum_{\mathbf{x}\in g_h} \mathbf{f}(\mathbf{x}) h^2 = \sum_{l=1}^{N_L} \mathbf{F}\left(\mathbf{X}_l\right) \Delta A_l \cancelto{1}{\sum_{\mathbf{x}\in g_h} \delta_h \left(\mathbf{x} - \mathbf{X}_l\right) h^2} = \sum_{l=1}^{N_L} \mathbf{F}\left(\mathbf{X}_l\right) \Delta A_l$$

4. $\boxed{\sum_{i=-\infty}^{\infty} (r-i)\phi(r-i) = 0 \ \forall \ r}$: this condition ensures the exact interpolation of linear functions, with the interpretation of conserving the torque during the transfer step. A similar proof of that in condition 3 can be showed by:

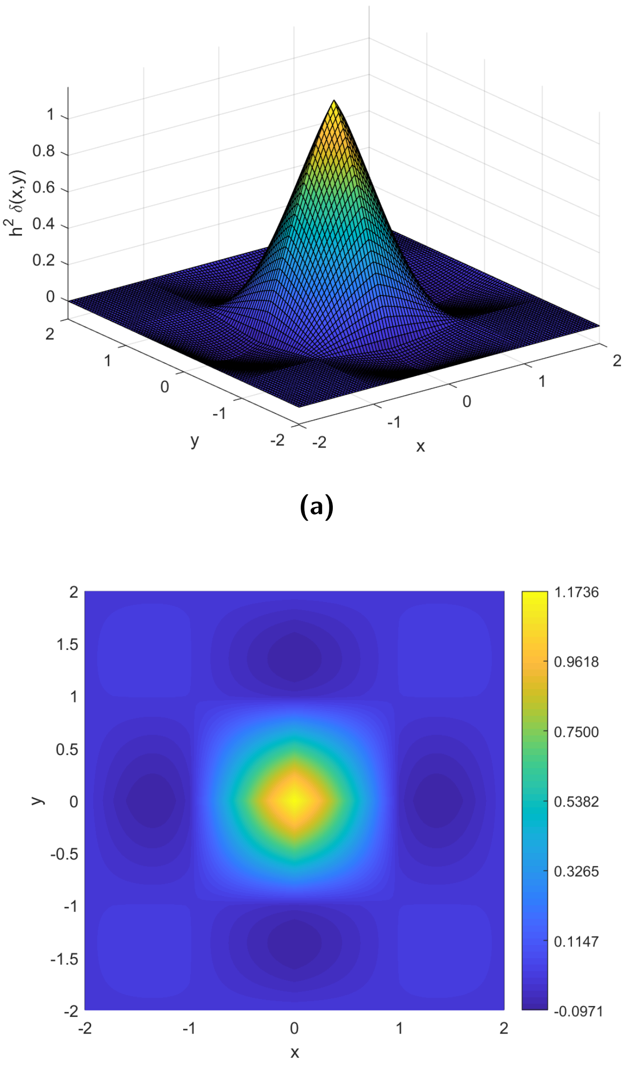
$$\sum_{\mathbf{x}\in g_h} \mathbf{x}\times\mathbf{f}(\mathbf{x})h^2 = \sum_{\mathbf{x}\in g_h}\sum_{l=1}^{N_L} \mathbf{x}\times\mathbf{F}\left(\mathbf{X}_l\right)\delta_h\left(\mathbf{x}-\mathbf{X}_l\right)\Delta A_l h^2,$$

$$\sum_{\mathbf{x}\in g_h} \mathbf{x}\times\mathbf{f}(\mathbf{x})h^2 = \sum_{l=1}^{N_L}\cancelto{\mathbf{X}}{\left(\sum_{\mathbf{x}\in g_h}\mathbf{x}\delta_h\left(\mathbf{x}-\mathbf{X}_l\right)h^2\right)}\times\mathbf{F}\left(\mathbf{X}_l\right)\Delta A_l = \sum_{l=1}^{N_L}\mathbf{X}\times\mathbf{F}\left(\mathbf{X}_l\right)\Delta A_l$$

5. $\boxed{\widetilde{M_2}(\phi)=0\ \forall\ r}$: which guarantees that quadratic functions are interpolated exactly.

6. $\boxed{\widetilde{M_3}(\phi)=0\ \forall\ r}$: this condition ensures the exact interpolation of cubic functions.

Fig 4.5 shows the derived delta function in its two-dimensional shape.



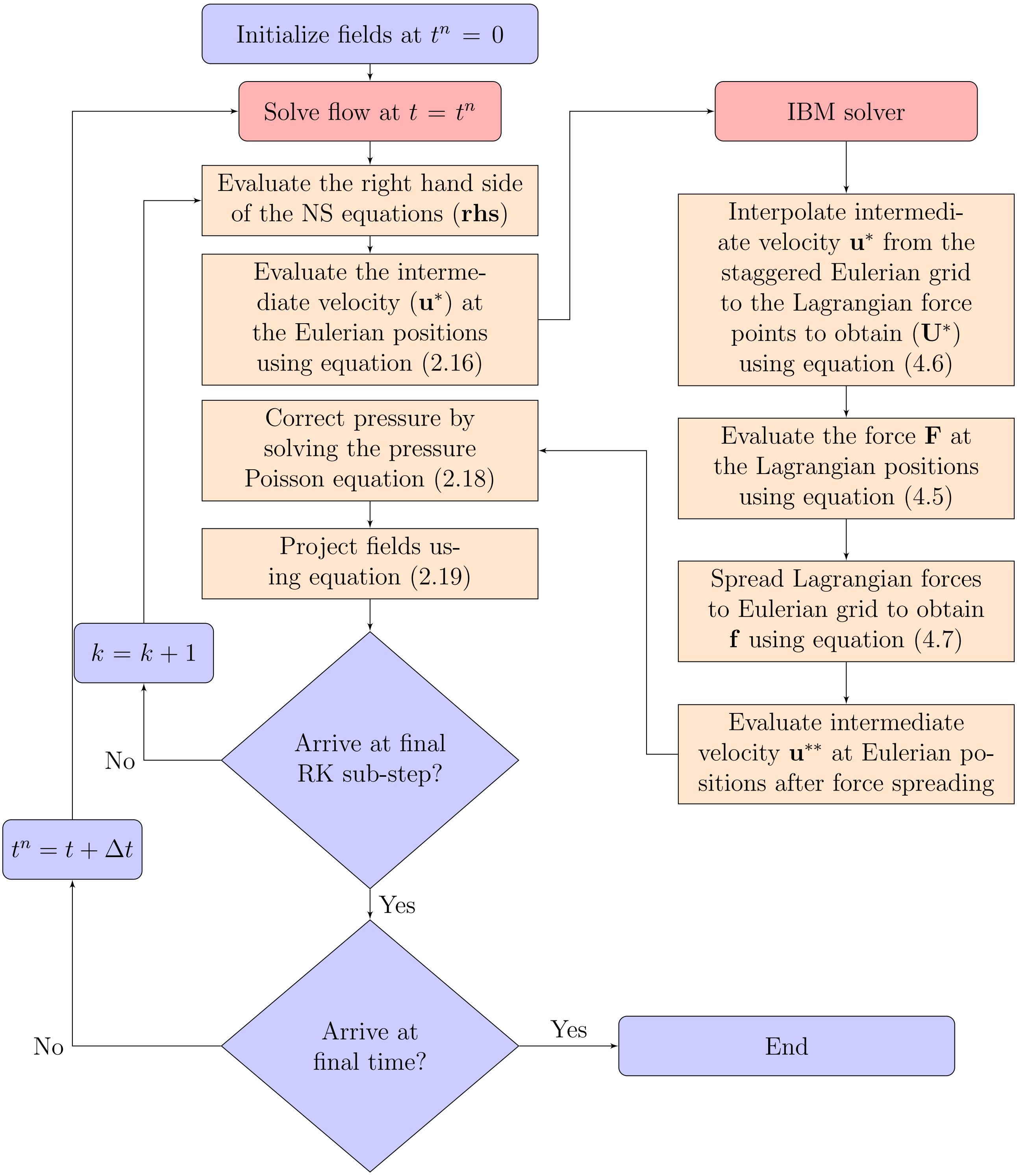
(a)

(b)

Figure 4.5: Two-dimensional Regularized delta function.

4.3. Integration of IB solver with the fourth order uniform FV code

The flow chart depicted below demonstrates the point where the IB solver is merged into flow solver. In addition, the flow of the numerical steps is sorted for each solver along with the associated equations.


Initialize fields at $t^n = 0$
Solve flow at $t = t^n$
Evaluate the right hand side of the NS equations (**rhs**)
Evaluate the intermediate velocity ($\mathbf{u}^*$) at the Eulerian positions using equation (2.16)
Correct pressure by solving the pressure Poisson equation (2.18)
Project fields using equation (2.19)
$k = k + 1$
No
Arrive at final RK sub-step?
Yes
$t^n = t + \Delta t$
No
Arrive at final time?
Yes
End
IBM solver
Interpolate intermediate velocity $\mathbf{u}^*$ from the staggered Eulerian grid to the Lagrangian force points to obtain ($\mathbf{U}^*$) using equation (4.6)
Evaluate the force $\mathbf{F}$ at the Lagrangian positions using equation (4.5)
Spread Lagrangian forces to Eulerian grid to obtain $\mathbf{f}$ using equation (4.7)
Evaluate intermediate velocity $\mathbf{u}^{**}$ at Eulerian positions after force spreading

4.4. Numerical Tests

In the numerical tests considered in this work, the embedded boundary has a prescribed velocity (4.3). In addition, only two-dimensional flows are examined and this can be, of course, extended to three spatial dimensions.

4.4.1. Taylor-Green vortex with an embedded circle

The Taylor-Green vortex flow was introduced in 3.3.1. In addition to its use in validating flow solvers, this smooth flow serves as a good test for the order of accuracy of the proposed regularized delta function (4.18). In order to do that, a circular sub-domain of radius $r = 2$ is embedded within the square computational domain $\Omega = [0,\ 2\pi]^2$. Fig 4.6 shows a schematic of the test setup.

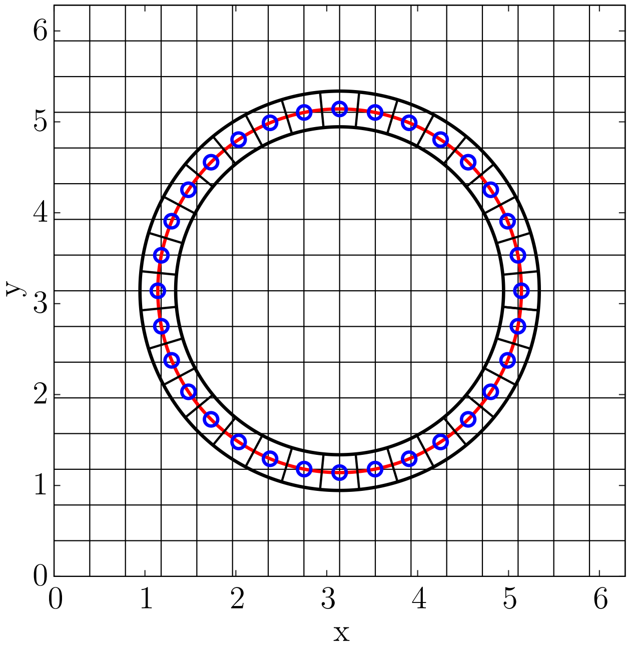


Figure 4.6: Embedded circle in the square computational domain.

Geometrical definitions:
In practice, the number of the Lagrangian force points N_L evenly distributed along the immersed boundary is chosen such that the area occupied by a Lagrangian element is equivalent to that of a finite volume of the Eulerian grid (i.e $\Delta A_l = \Delta x\ \Delta y$). Further increase of N_L does not lead to any significant improvement of the results [Uhlmann, 2005]. Like [Uhlmann, 2004] and [Uhlmann, 2005], the Lagrangian elements of the embedded circle (with radius r_c) are equi-partitioned sectors of an annulus with inner and outer radii $(r_{in},\ r_{out})$ having the Lagrangian points located at their centers. With the radial length of the element being equal to the mesh size:

$$\Delta x = r_{out} - r_{in},$$

and the arc-length at the actual radius of the immersed boundary r_c:

$$s = \frac{2\pi}{N_L} r_c,$$

the area of a Lagrangian element is therefore:

$$\Delta A_l = \Delta x \; s = \frac{2\pi r_c \Delta x}{N_L}.$$

Taking $\Delta A_l = \Delta x \Delta y$ and $N_x = N_y$, the number of Lagrangian points is hence:

$$N_L \approx \frac{2\pi}{r_c}.$$

Spatial convergence:
We set the characteristic length of the problem to $D = 4$ and tune the kinematic viscosity to $\nu = 0.04$ to result in a Reynolds number of $Re = 100$. The exact solution (3.20)-(3.22) initializes the fields and provides the time-dependent desired velocity $\mathbf{U}_d$ at the Lagrangian force points $\mathbf{X}_l$ of the immersed circle. The maximum error of the velocity inside the immersed boundary is recorded for successive grid refinements at the end of the simulation time $t = 0.1$ where the equations are advanced with a time step of $\Delta t = 0.0005$. Fig 4.7 confirms the accuracy of the interpolation of the derived delta function as it demonstrates the fourth-order rate of convergence achieved.

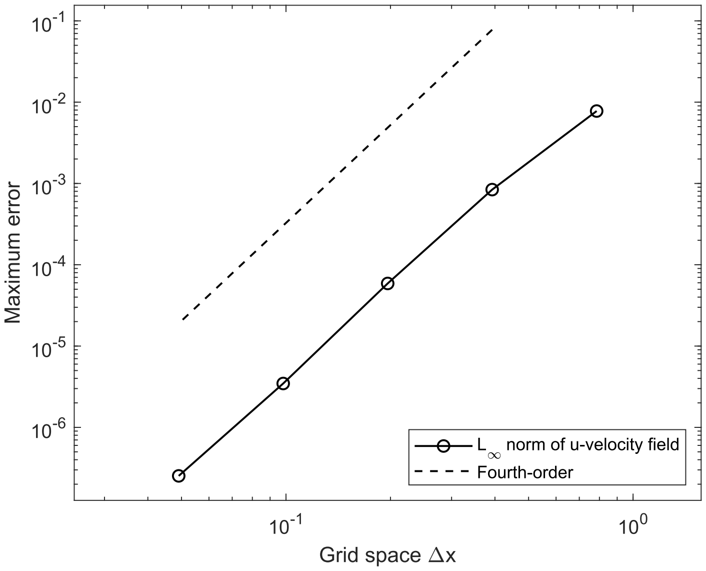


Figure 4.7: Rate of convergence of the L_∞ of the error.

A spatial distribution of error of the u-velocity field is depicted in Fig 4.8 for $N_x = N_y = 128$. Although it is clear that the larger error committed throughout the computational domain is in the layer surrounding the immersed circle, its magnitude is still very small. To verify the satisfaction of the no-slip condition at the Lagrangian force points, we interpolate the velocity field from the Eulerian points to each Lagrangian point at the end of the simulation and compare these values against the analytical solution. Fig 4.9 shows the convergence of the maximum error. Although the error is continuously decreasing, the rate of convergence is not the expected fourth-order. A probable cause of this result is the projection step following the IBM solver.

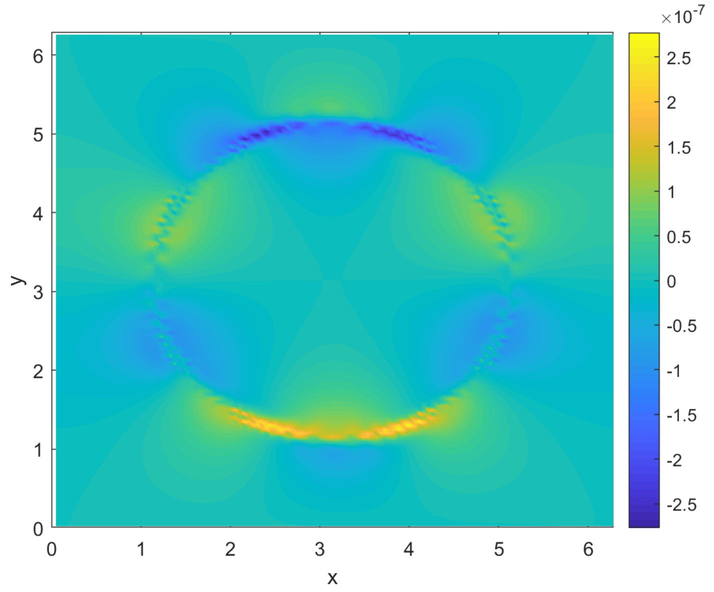


Figure 4.8: Spatial distribution of the error throughout the computational domain.

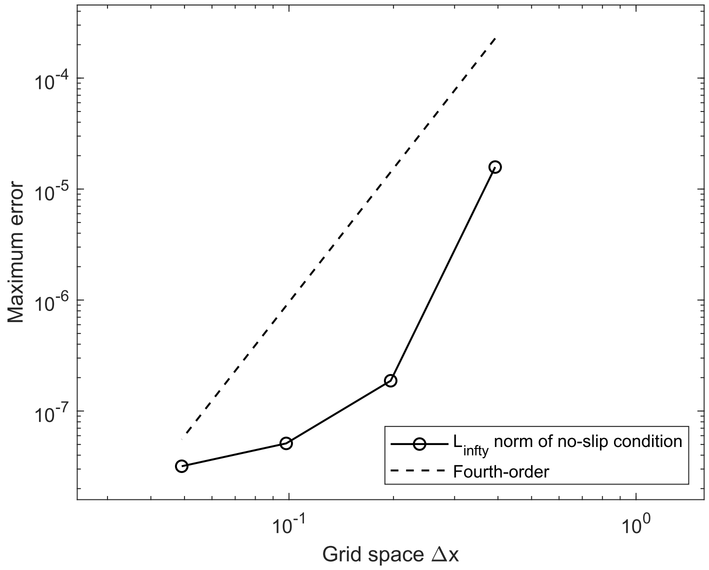


Figure 4.9: Rate of convergence of no-slip condition L_∞ norm.

We proceed to evaluate another important result, namely, the dependency of the error upon the position of the immersed boundary with respect to the Eulerian grid. This property can be tested by simulating the setup for several times with a fixed grid spacing (here with $h =$

0.05) and applying horizontal shifts to the center of the embedded circle at the beginning of each simulation. A total of 50 numerical simulations were conducted, in each simulation the x-coordinate of the circle was updated as $x_{circle} = x_{circle} + \frac{\Delta x}{50}$. Fig 4.10 depicts the maximum error with respect to fractions of mesh width where it can be clearly seen that error is not sensitive to the relative position of the immersed boundary.

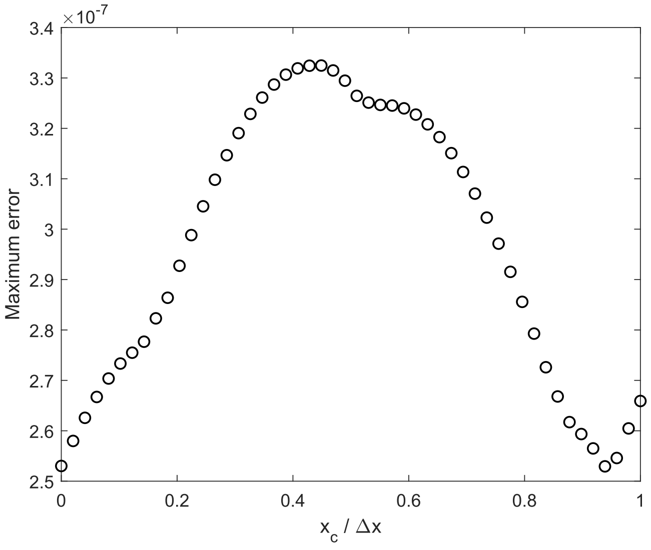


Figure 4.10: Translation invariance of the derived regularized delta function

Similar results were also evaluated for the second-order finite volume implementation with the regularized delta function of [Peskin, 2002]. The order of accuracy for the maximum norm of the error recorded inside the embedded circle and that of the no-slip condition is depicted in Fig 4.11a and Fig 4.11b, respectively.

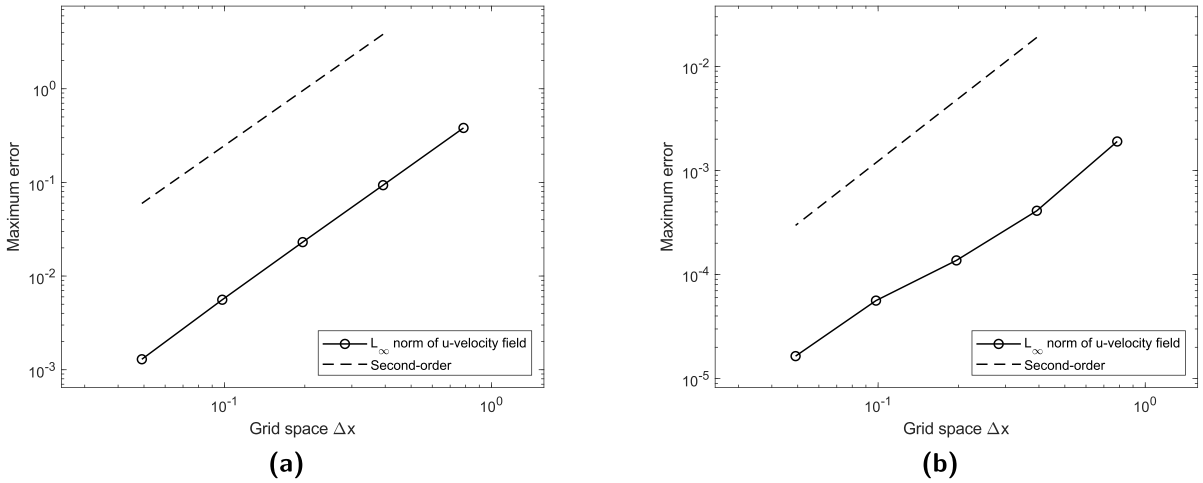
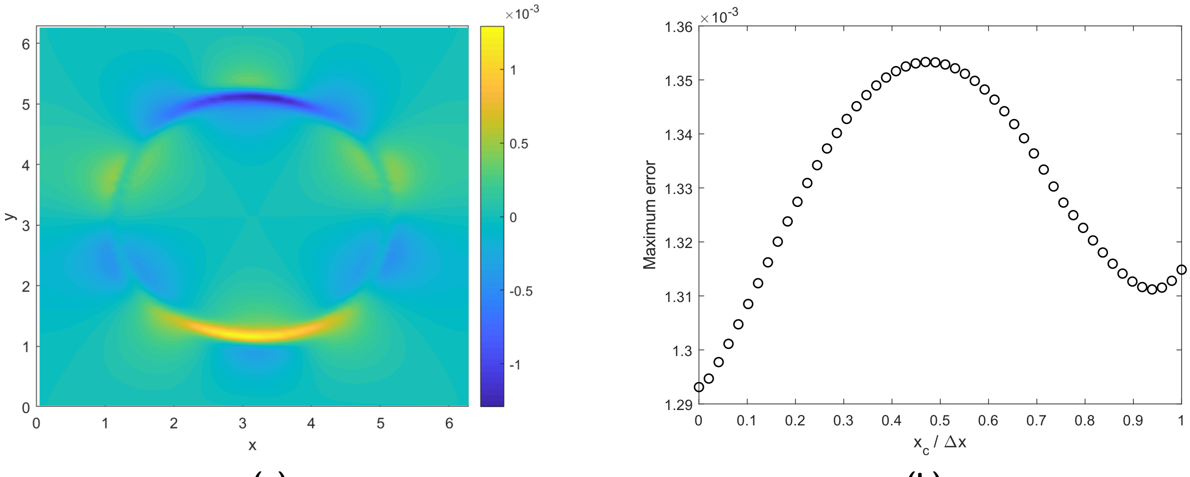


Figure 4.11: (a) Embedded circle in the square computational domain. (b) Rate of convergence of no-slip condition L_∞ norm.

Fig 4.12a shows the spatial distribution of the error and Fig 4.12b confirms the low sensitivity of results to the relative position of the immersed boundary.



(a) **(b)**

Figure 4.12: Using the regularized delta by [Peskin, 2002]. (a) The spatial distribution of the error throughout the computational domain. (b) Translation invariance.

Temporal convergence:

A temporal convergence study has been conducted to check the influence of the immersed boundary on the accuracy of the time integration scheme. Fig 4.13 depicts the rate of convergence of the u-velocity and pressure fields. It can be observed that the first-order convergence of the pressure is maintained. However, the third-order convergence of velocity is lost and an alternating rate of ≈ 1 is obtained instead. The investigation of this effect was outside the scope of this work.

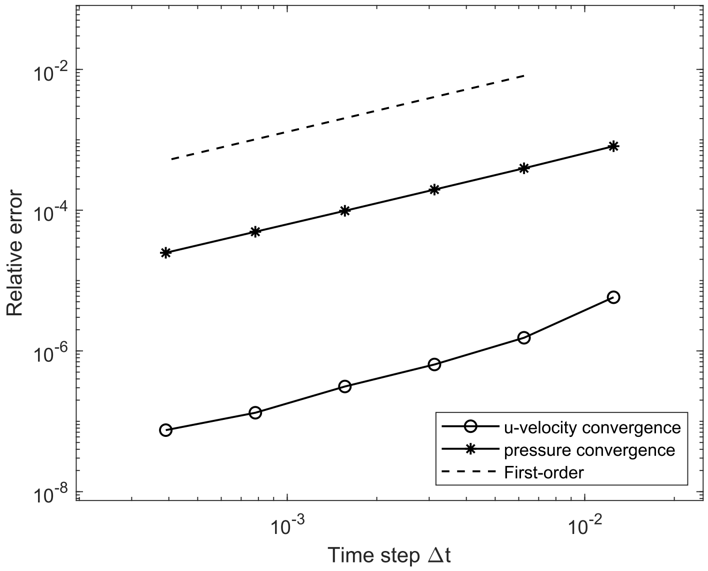


Figure 4.13: IBM temporal convergence test results.

4.4.2. Oscillating channel flow

This test case is taken from [Schlichting et al., 2017] and the objective is to verify the convergence of the method in non-smooth velocity fields. An oscillating channel flow is an unsteady internal flow occurring when the fluid is acted on by a periodic pressure gradient. This pulsatile flow can be realized by blood flow in vascular system, or the periodic movement of a piston against a fluid in a channel. Assuming an infinitely long channel along the x-axis, the flow becomes independent of the coordinate x and hence any velocity derivative with respect to x will vanish. We let D be the diameter of the channel and given the no-slip boundary conditions $u(y = \pm\ \frac{D}{2}) = v(y = \pm\ \frac{D}{2}) = 0$, the NS equations are simplified to the linear equation:

$$\frac{\partial u}{\partial t} = -\frac{1}{\rho}\frac{\partial p}{\partial x} + \nu\frac{\partial^2 u}{\partial y^2}. \tag{4.19}$$

We impose a harmonic pressure gradient to the u-momentum equation:

$$-\frac{1}{\rho}\frac{\partial p}{\partial x} = sin(nt) = -i\ e^{i(nt)},$$

where i is the imaginary unit. The solution of the differential equation (4.19) in a complex notation is then [Schlichting et al., 2017]:

$$u(y,t) = -\frac{1}{n}e^{i(nt)}\left[1 - \frac{cosh(y\sqrt{{}^{in}/_{\nu}})}{cosh(({}^{h}/_{2})\sqrt{{}^{in}/_{\nu}})}\right], \tag{4.20}$$

where only the real part has the physical significance. The pulsatile flow velocity profile changes its shape based on the frequency of the imposed pressure gradient. A dimensionless expression relating the frequency n to the viscous effects is given by Womersley number:

$$Wr = \sqrt{\frac{nL^2}{\nu}}, \tag{4.21}$$

where L is the characteristic length scale taken as the radius of the channel. In the setup of this case, we define the boundaries of the channel by means of two immersed boundaries as shown in Fig 4.14. We initialize the domain confined by the immersed boundaries with the exact solution (4.20) by setting the kinematic viscosity to $\nu = 0.2$, the frequency $n = 20$ and the diameter of the channel $D = 1$. Given the aforementioned parameters and the characteristic length $L = \frac{D}{2} = 0.5$, the Womersley number is evaluated $Wr = 5$.

We perform simulations for varying Δy with a fine time step of $\Delta t = 0.00005$ and compare the results to the exact solution at $t = \frac{1}{8}T \approx 0.04$, where T is the time period of the periodic pressure gradient defined as $T = \frac{2\pi}{n}$.

Convergence study results for the maximum error inside the channel are shown for the second and fourth-order implementations in Fig 4.15. While the rate of convergence of the former is reduced from 2 to 1, that of the latter went down from 4 to $\approx$ 1.3 with a lower order of magnitude recorded for all convergence steps. One can clearly see the drop in the order

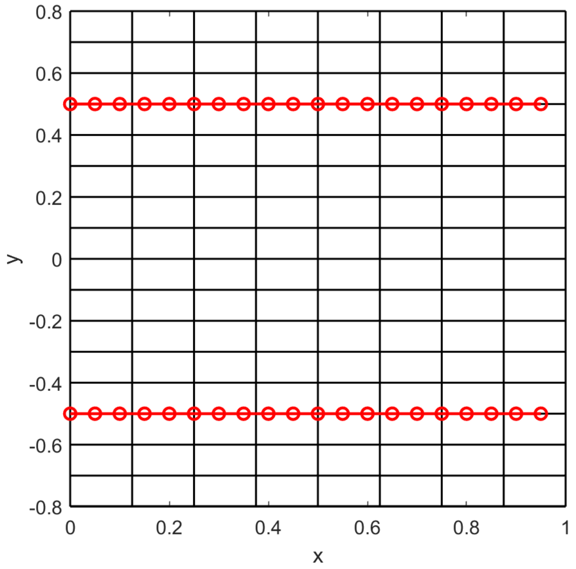


Figure 4.14: Immersed boundaries representing the boundaries of the channel.

of convergence caused when using the regularized delta functions with non-smooth velocity fields. In this problem, the velocity field has a kink at the immersed boundaries and it is only C^0 continuous. An interpretation of this behavior is regarded to the assumptions made when the regularized delta is derived. The existence of higher order derivatives at the Lagrangian points was an obligation for the derivation in section 4.2 to hold. Therefore, the delta functions can no longer maintain the order of accuracy they were designed to achieve.

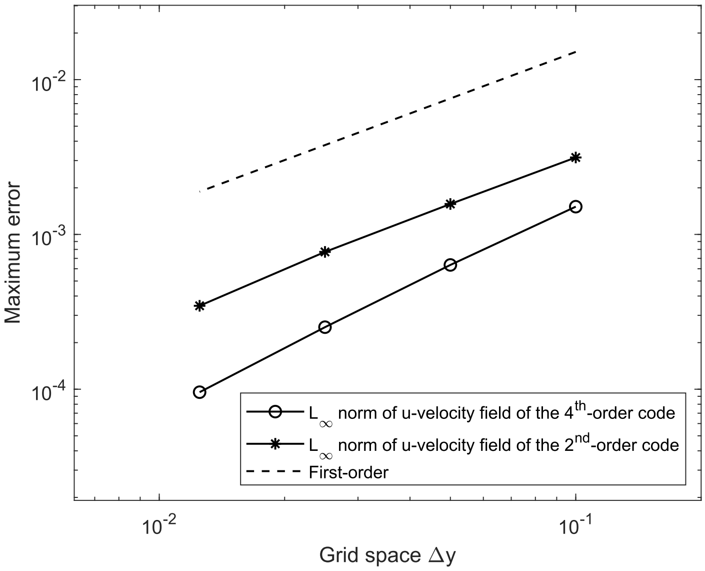
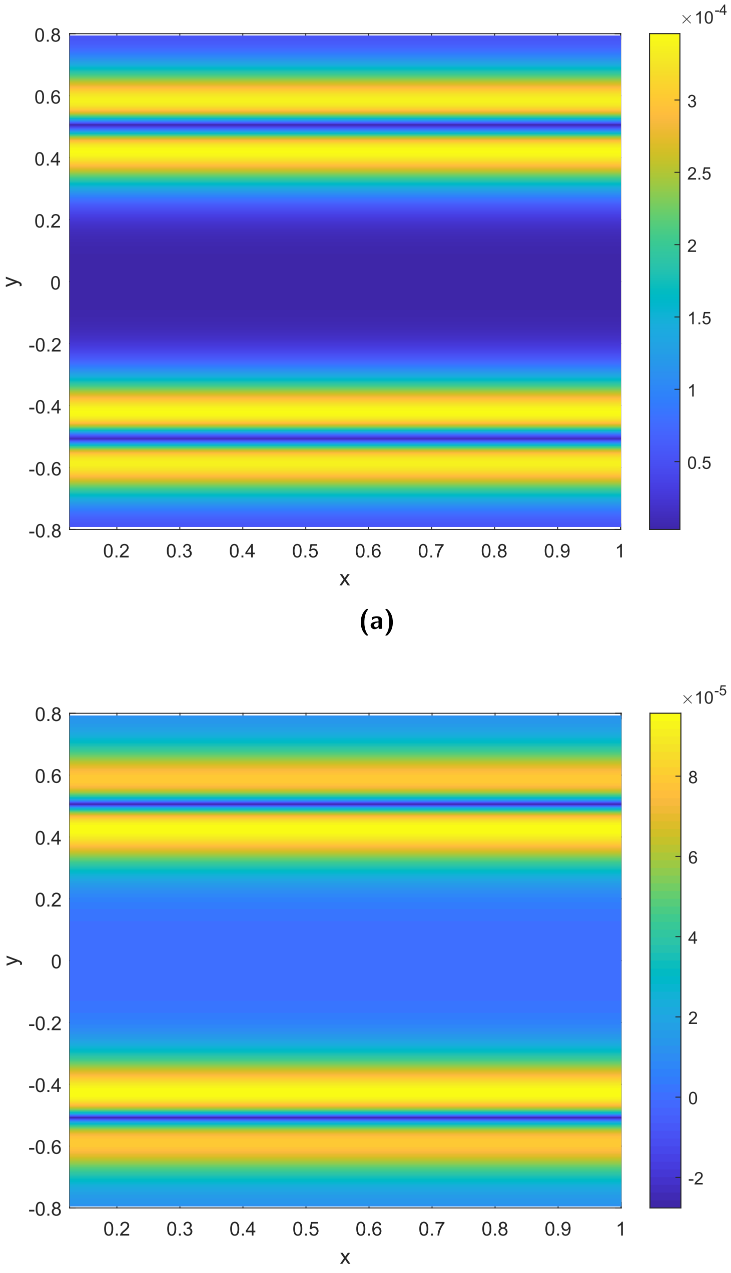


Figure 4.15: Rate of convergence of the maximum error for the second and fourth-order implementations.

Fig 4.16 depicts the spatial distribution of error throughout the computational domain of both implementations. As expected, high values of error are committed in the neighborhood where the no-slip condition is imposed.



(a)

(b)

Figure 4.16: Spatial distribution of error for: (a) Second-order code, (b) Fourth-order code.

4.4.3. Stokes first problem

The solution of Stokes first problem (also known as Rayleigh problem) defines the velocity profile of a semi-infinite layer of an initially quiescent incompressible fluid caused by an impulsive movement of an infinitely long stationary plate beneath it (Fig 4.17).

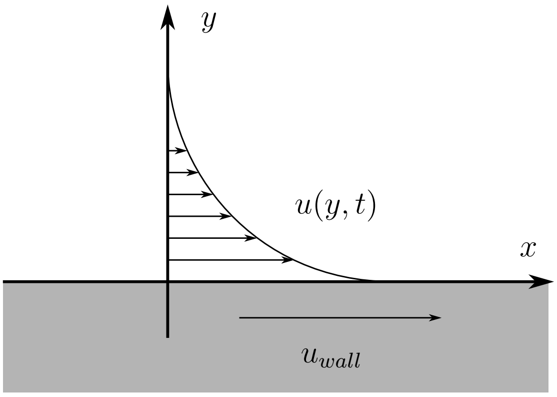


Figure 4.17: Stokes first problem

The simplifications associated with this problem are:

$$\frac{\partial u}{\partial x} = \frac{\partial p}{\partial x} = v = 0,$$

and the u-momentum equation reduces to:

$$\frac{\partial u}{\partial t} = \nu \frac{\partial^2 u}{\partial y^2}.$$

The boundary and the initial conditions are defined as:

$$u(y, t = 0) = 0, \quad u(y = 0, t > 0) = U_0, \quad u(y \to \infty, t) = 0,$$

and the exact solution of the problem reads:

$$u(y,t) = U_0 \ \mathrm{erfc}(\eta), \tag{4.22}$$

where erfc is the complimentary error function and η is the dimensionless similarity variable. To simulate this problem, we place an immersed boundary in the middle of the computational domain as shown Fig 4.18 and assign the velocity $U_o = 0.5$ to it. We set the kinematic viscosity to $\nu = 1$ and initialize the problem with a zero-velocity field as the fluid is initially at rest. We perform multiple simulations with an increasing spatial resolution and evaluate the error with respect to the exact solution (4.22) at time $t = 0.05$ with a time step of $\Delta t = 0.00005$. The influence of delta function regularization on the rate of convergence is depicted in Fig 4.19. A rate of convergence of ≈ 0.8 is measured when using the delta function by [Peskin, 2002] and a first order convergence is recorded for the derived delta function.

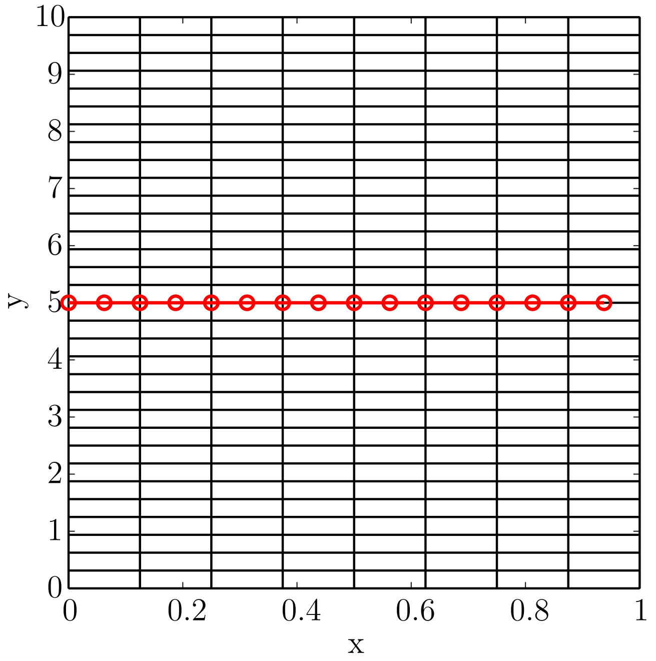


Figure 4.18: Immersed boundary representing the moving wall

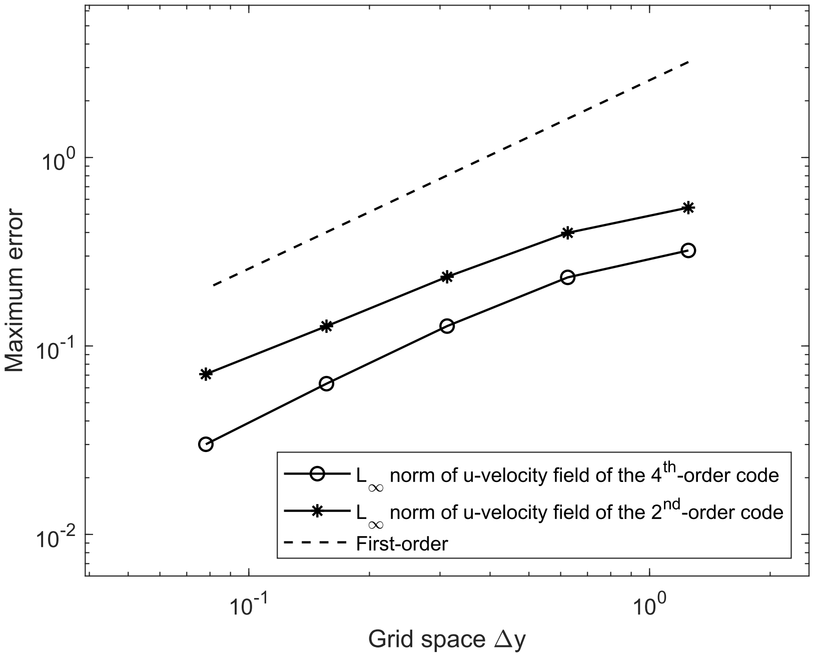
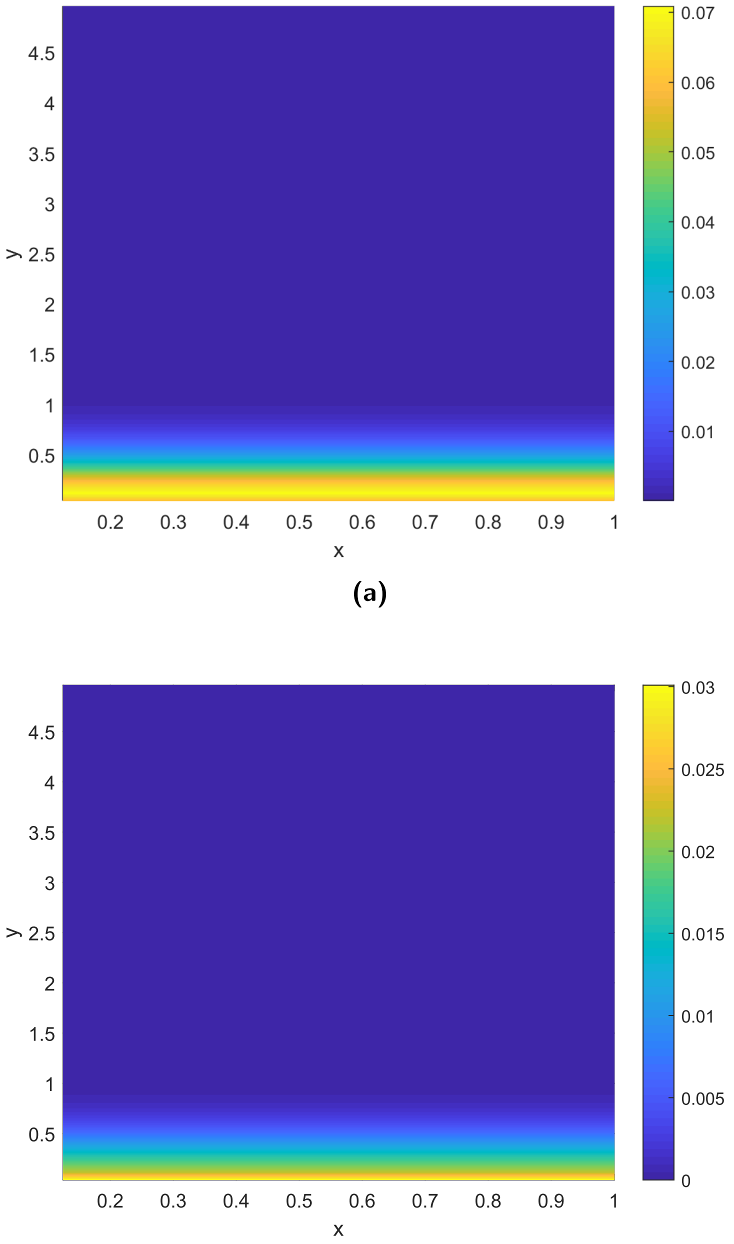


Figure 4.19: Rate of convergence of the maximum error for the second and fourth-order implementations.

Fig 4.20 depicts the spatial distribution of error for this test case. Higher values are, of course, recorded in the vicinity of the moving immersed boundary.



(a)

(b)

Figure 4.20: Spatial distribution of error for: (a) Second-order code, (b) Fourth-order code.

5. Conclusion and Outlook

We summarize the main points covered in this thesis along with the concluding remarks as follows. First, general theoretical concepts behind this work were introduced in detail. These concepts included the Navier-Stokes equations we intended to solve, the indicial notation providing important grid related definitions, definition of flux terms in the semi-discretized equations, time advancement scheme with the associated projection method and a brief introduction to the immersed boundary method.

Next, second-order and fourth-order finite volume discretizations of the Navier-Stokes equations (NSE) tailored specially for staggered grids were presented. For each case, discretizations were performed on uniform as well as non-uniform structured grids. We initially stated the simple second-order discretization scheme of the convective, diffusive and pressure fluxes on uniform grids. For non-uniform grids, we performed a spatial discretization method which preserves the symmetries of the convective (skew-symmetric) and diffusive (symmetric positive-definite) operators [Verstappen and Veldman, 2003]. Some approximations of terms (e.g first derivative) had only a first-order local truncation error. However, it was shown that the concept does not strive for a minimal local truncation error and carries out discretizations on physical grounds and still achieves the second-order rate of convergence. This interesting concept results a semi-discrete representation of conservation of mass, momentum and energy (in the absence of viscous effects).

Then, fourth-order discretization concepts mainly discussed by [Hokpunna, 2009] were presented. First, an explicit distinction was made between cell-averaged, edge-averaged and point-wise centered quantities, as these values are only allowed to interchange in the second-order context. Various numerical techniques approximating each term in the NS equations were described. Cell-centered deconvolution approximating an edge-averaged quantity at cell centroid was used on: 1) momentum cells to compute mass fluxes across pressure cells and 2) pressure cells to find pressure gradients across momentum cells. Instead of using fourth-order compact schemes of [Kobayashi, 1999], we opted to use a pre-processing step prior to applying the second-order operators to evaluate the convected velocities and their first derivatives at the boundaries of the momentum cells [Schwertfirm et al., 2008]. This pre-processing step improves resolution characteristics of numerical operations and increases their order of truncation error. In addition to being easy to implement, the applied filter functions work on small, compact stencils, so no grid-boundaries problems arise. To interpolate the convective velocities to momentum cell boundaries in a divergence-free manner (necessary condition for energy conservation), the novel approach developed by [Hokpunna, 2009] was used. Fourth-order approximation of convective fluxes is then ensured by the addition of a non-linear correction term. Finally, we closed the presentation of the second and fourth-order schemes by a derivation of a formal second-order and fourth-order discrete Laplace operators. Within the framework of consistent projection method, we were obliged to solve the latter one to achieve a fourth-order rate of convergence. Numerical tests were performed for classical and convected Taylor-Green vortex flow to check the accuracy of each solver. Flow fields of the second-order solver converge at second-order for both uniform and non-uniform grids, where its fourth-order counterpart converge at fourth-order. Temporal convergence study of both solvers proves the third-order convergence of velocity and the first-order convergence of pressure. Of course, a room for improvement exists for the above-mentioned fluid solvers. The main improvement would be the proper implementation of boundary closure for Dirichlet and Neumann boundary conditions. Then, one can be more flexible with the choice of test casesser.

Upon the validation of the implemented schemes, we extended our scope to cover the immersed boundary method with a direct continuous approach by [Uhlmann, 2005]. We presented the mathematical formulation of the problem with a clear distinction made between quantities evaluated at Eulerian (fluid) positions and others evaluated at Lagrangian (solid) positions. As in the original continuous IB method by [Peskin, 1972], an approximation of the Dirac delta function is used to link the two types of variables. As the main theme of this thesis was the spatial accuracy of the numerical schemes, we stated some regularized delta functions from the literature (such that of [Roma et al., 1999] and [Peskin, 2002]) capable of interpolating quantities at second-order. To derive a fourth-order regularized delta function, we presented a theorem relating the order of truncation error of a delta function to the number of discrete moment conditions it satisfies [Tornberg and Engquist, 2004]. We adapted the proof of this theorem to fit the use of cell-averaged velocities. This resulted in a set of four modified moment conditions that needed to be satisfied. Upon the appropriate selection of delta function support, a unique expression could be derived. Finally, second and fourth order immersed boundary solvers were properly integrated with the respective above-mentioned fluid solvers.

Numerical testing stage of the immersed boundary implementation confirmed the fourth-order accuracy of the derived regularized delta function. However, this rate of convergence was only restricted to cases where the interpolated function fields were smooth at the immersed boundary. It was observed that the fourth-order of convergence drops to first-order when there was a kink of the velocity field at the immersed boundary. At first sight, this seemed like a flaw in theory or in the implementation. Yet, once a closer look was given to the derivation of the delta function, we realized that we have made an assumption in order for this derivation to hold. The assumption was in the Taylor expansion in equation (4.14), where we assumed that all derivatives exist up to the third order. In the case of non-smooth fields, the velocity field has a C^0 continuity across the immersed boundary, hence we can't achieve a fourth-order rate of convergence with the derived delta function. We suggest a remedy for this problem by adding a set of one-sided discrete moment conditions described in [Beyer and Leveque, 1992] to the rows of the linear system of equations (4.17) and seek a new delta function accordingly. The IB solver presented in this work was limited to uniform Cartesian grids. A possible suggestion for future work would also be to extend its capabilities to account for non-uniform grid systems while maintaining the order of the underlying spatial discretizations. The reader is encouraged to read the work of [Pinelli et al., 2010] treating this matter with a second-order rate of convergence. This extension would significantly improve the accuracy of results obtained by this method, and of course, at a lower computational cost.

Bibliography

[Batchelor, 2000] Batchelor, G. K. (2000). *An Introduction to Fluid Dynamics*. Cambridge Mathematical Library. Cambridge University Press.

[Battista et al., 2018] Battista, N. A., Strickland, W. C., Barrett, A., and Miller, L. A. (2018). IB2d Reloaded: A more powerful Python and MATLAB implementation of the immersed boundary method. *Mathematical Methods in the Applied Sciences*, 41(18):8455–8480.

[Beyer and Leveque, 1992] Beyer, R. P. and Leveque, R. J. (1992). Analysis of a one-dimensional model for the immersed boundary method. *SIAM Journal on Numerical Analysis*, 29(2):332–364.

[Chorin, 1967] Chorin, A. J. (1967). The numerical solution of the navier-stokes equations for an incompressible fluid. *Bull. Amer. Math. Soc.*, 73(6):928–931.

[Donea et al., 2004] Donea, J., Huerta, A., Ponthot, J.-P., and Rodríguez-Ferran, A. (2004). *Arbitrary Lagrangian–Eulerian Methods*, chapter 14. American Cancer Society.

[Fadlun et al., 2000] Fadlun, E., Verzicco, R., Orlandi, P., and Mohd-Yusof, J. (2000). Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations. *Journal of Computational Physics*, 161(1):35 – 60.

[Ferziger and Peric, 2001] Ferziger, J. and Peric, M. (2001). *Computational Methods for Fluid Dynamics*. Springer Berlin Heidelberg.

[Goldstein et al., 1993] Goldstein, D., Handler, R., and Sirovich, L. (1993). Modeling a no-slip flow boundary with an external force field. *Journal of Computational Physics*, 105(2):354 – 366.

[Harlow and Welch, 1965] Harlow, F. H. and Welch, J. E. (1965). Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with Free Surface. *Physics of Fluids*, 8(12):2182–2189.

[Hokpunna, 2009] Hokpunna, A. (2009). *Compact fourth-order scheme for numerical simulations of Navier-Stokes Equations*. PhD thesis, Technical University of Munich, Munich.

[Kobayashi, 1999] Kobayashi, M. H. (1999). On a class of padé finite volume methods. *Journal of Computational Physics*, 156(1):137 – 180.

[Lai and Peskin, 2000] Lai, M.-C. and Peskin, C. S. (2000). An immersed boundary method with formal second-order accuracy and reduced numerical viscosity. *Journal of Computational Physics*, 160(2):705 – 719.

[Manteuffel and White, 1986] Manteuffel, T. A. and White, A. B. (1986). The numerical solution of second-order boundary value problems on nonuniform meshes. *Mathematics of Computation*, 47(176):511–535.

[Mittal and Iaccarino, 2005] Mittal, R. and Iaccarino, G. (2005). Immersed boundary methods. *Annual Review of Fluid Mechanics*, 37(1):239–261.

[Peller, 2010] Peller, N. (2010). *Numerische Simulation turbulenter Strömungen mit Immersed Boundaries.* Dissertation, Technische Universität München, München.

[Pereira et al., 2001] Pereira, J., Kobayashi, M., and Pereira, J. (2001). A fourth-order-accurate finite volume compact method for the incompressible navier–stokes solutions. *Journal of Computational Physics*, 167(1):217 – 243.

[Peskin, 1972] Peskin, C. (1972). *Flow Patterns Around Heart Valves: A Digital Computer Method for Solving the Equations of Motion.* PhD thesis, Albert Einstein College of Medicine of Yeshiva University.

[Peskin, 1977] Peskin, C. S. (1977). Numerical analysis of blood flow in the heart. *Journal of Computational Physics*, 25(3):220 – 252.

[Peskin, 2002] Peskin, C. S. (2002). The immersed boundary method. *Acta Numerica*, 11:479–517.

[Pinelli et al., 2010] Pinelli, A., Naqavi, I., Piomelli, U., and Favier, J. (2010). Immersed-boundary methods for general finite-difference and finite-volume navier–stokes solvers. *Journal of Computational Physics*, 229:9073–9091.

[Roma et al., 1999] Roma, A. M., Peskin, C. S., and Berger, M. J. (1999). An adaptive version of the immersed boundary method. *Journal of Computational Physics*, 153(2):509 – 534.

[Runborg, 2012] Runborg, O. (2012). Lecture notes on verifying numerical convergence rates.

[Schlichting et al., 2017] Schlichting, H., Gersten, K., Krause, E., and Oertel, H. (2017). *Boundary-layer theory.* Springer.

[Schwertfirm et al., 2008] Schwertfirm, F., Mathew, J., and Manhart, M. (2008). Improving spatial resolution characteristics of finite difference and finite volume schemes by approximate deconvolution pre-processing. *Computers and Fluids*, 37(9):1092 – 1102.

[Stolz and Adams, 1999] Stolz, S. and Adams, N. A. (1999). An approximate deconvolution procedure for large-eddy simulation. *Physics of Fluids*, 11(7):1699–1701.

[Tornberg and Engquist, 2004] Tornberg, A.-K. and Engquist, B. (2004). Numerical approximations of singular source terms in differential equations. *J. Comput. Phys.*, 200(2):462–488.

[Uhlmann, 2004] Uhlmann, M. (2004). New results on the simulation of particulate flows. Technical Report 1038, CIEMAT, Madrid, Spain.

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

[Verstappen and Veldman, 2003] Verstappen, R. and Veldman, A. (2003). Symmetry-preserving discretization of turbulent flow. *Journal of Computational Physics*, 187(1):343 – 368.

[Williamson, 1980] Williamson, J. (1980). Low-storage runge-kutta schemes. *Journal of Computational Physics*, 35(1):48 – 56.

A. Appendix

A.1. Cell-centered deconvolution of velocity on non-uniform grids

The Cell-centered deconvolution of a u-momentum cell approximating the egde-averaged velocity at the center of the u-momentum cell from the cell-averaged velocities reads:

$$u^{y}_{is,j} = \beta_1 u^{xy}_{is-1,j} + \beta_2 u^{xy}_{is,j} + \beta_3 u^{xy}_{is+1,j}. \tag{A.1}$$

To find the coefficients, we take the Taylor expansion of each term on the right hand side around x_{is}:

$$\begin{aligned}
u^{xy}_{is-1,j} &= \frac{1}{\Delta x_{is-1}} \int_{x_{is}-\frac{\Delta x_{is}}{2}-\Delta x_{is-1}}^{x_{is}-\frac{\Delta x_{is}}{2}} u^{y}\ dx \\
&\approx \frac{1}{\Delta x_{is-1}} \int_{x_{is}-\frac{\Delta x_{is}}{2}-\Delta x_{is-1}}^{x_{is}-\frac{\Delta x_{is}}{2}} u^{y}_{is,j} + (x - x_{is}) \frac{du}{dx}\bigg|^{y}_{is,j} + \frac{(x - x_{is})^2}{2!} \frac{d^2u}{dx^2}\bigg|^{y}_{is,j}\ dx \\
&\approx \frac{1}{\Delta x_{is-1}} \left[u^{y}_{is,j} x + \frac{(x - x_{is})^2}{2} \frac{du}{dx}\bigg|^{y}_{is,j} + \frac{(x - x_{is})^3}{6} \frac{d^2u}{dx^2}\bigg|^{y}_{is,j} \right]\Bigg|^{x_{is}-\frac{\Delta x_{is}}{2}}_{x_{is}-\frac{\Delta x_{is}}{2}-\Delta x_{is-1}}
\end{aligned}$$

Similarly, for $u_{is,j}$ and $u_{is+1,j}$ momentum cells we write:

$$\begin{aligned}
u^{xy}_{is,j} &\approx \frac{1}{\Delta x_{is}} \left[u^{y}_{is,j} x + \frac{(x - x_{is})^2}{2} \frac{du}{dx}\bigg|^{y}_{is,j} + \frac{(x - x_{is})^3}{6} \frac{d^2u}{dx^2}\bigg|^{y}_{is,j} \right]\Bigg|^{x_{is}+\frac{\Delta x_{is}}{2}}_{x_{is}-\frac{\Delta x_{is}}{2}}, \\
u^{xy}_{is+1,j} &\approx \frac{1}{\Delta x_{is+1}} \left[u^{y}_{is,j} x + \frac{(x - x_{is})^2}{2} \frac{du}{dx}\bigg|^{y}_{is,j} + \frac{(x - x_{is})^3}{6} \frac{d^2u}{dx^2}\bigg|^{y}_{is,j} \right]\Bigg|^{x_{is}+\frac{\Delta x_{is}}{2}+\Delta x_{is+1}}_{x_{is}+\frac{\Delta x_{is}}{2}}.
\end{aligned}$$

Upon substituting the limits of the integral, we write:

$$\begin{aligned}
u^{xy}_{is-1,j} &= u^{y}_{is,j} - \left(\frac{\Delta x_{is} + \Delta x_{is-1}}{2} \right) \frac{du}{dx}\bigg|^{y}_{is,j} + \left(\frac{\Delta x^2_{is}}{8} + \frac{\Delta x_{is} \Delta_{is-1}}{4} + \frac{\Delta x^2_{is-1}}{6} \right) \frac{d^2u}{dx^2}\bigg|^{y}_{is,j}, \\
u^{y}_{is,j} &= u^{y}_{is,j} + \frac{\Delta x_{is,j}}{24}\ \frac{d^2u}{dx^2}\bigg|_{is,j}, \\
u^{xy}_{is+1,j} &= u^{y}_{is,j} + \left(\frac{\Delta x_{is+1} + \Delta x_{is}}{2} \right) \frac{du}{dx}\bigg|^{y}_{is,j} + \left(\frac{\Delta x^2_{is}}{8} + \frac{\Delta x_{is} \Delta_{is+1}}{4} + \frac{\Delta x^2_{is+1}}{6} \right) \frac{d^2u}{dx^2}\bigg|^{y}_{is,j}.
\end{aligned}$$

Now, we can construct the linear system of equations to solve for the coefficients:

$$\begin{bmatrix} 1 & 1 & 1 \\ -\Delta x_{is} - \Delta x_{is-1} & 0 & \Delta x_{is+1} + \Delta x_{is} \\ 3\Delta x_{is}^2 + 6\Delta x_{is}\Delta_{is-1} + 4\Delta x_{is-1}^2 & \Delta x_{is,j} & 3\Delta x_{is}^2 + 6\Delta x_{is}\Delta_{is+1} + 4\Delta x_{is+1}^2 \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

A.2. Cell-centered deconvolution of pressure on non-uniform grids

The cell-centered deconvolution for the pressure approximates the edge-averaged pressure at the pressure grid points position and it is given by the following formula:

$$p^y_{i,j} = \beta_4 p^{xy}_{i-1,j} + \beta_5 p^{xy}_{i,j} + \beta_6 p^{xy}_{i+1,j}. \tag{A.2}$$

The coefficients are found by expanding the right hand side around x_i :

$$p^{xy}_{i-1,j} \approx \frac{1}{\Delta x_{i-1}} \left[p^y_{i,j} x + \frac{(x-x_i)^2}{2} \frac{dp}{dx}\bigg|^y_{i,j} + \frac{(x-x_i)^3}{6} \frac{d^2p}{dx^2}\bigg|^y_{i,j} \right] \Bigg|^{x_i - \frac{\Delta x_{is-1}}{2}}_{x_i - \Delta x_{is-1} - \frac{\Delta x_{is-2}}{2}},$$

$$p^{xy}_{i,j} \approx \frac{1}{\Delta x_{i}} \left[p^y_{i,j} x + \frac{(x-x_i)^2}{2} \frac{dp}{dx}\bigg|^y_{i,j} + \frac{(x-x_i)^3}{6} \frac{d^2p}{dx^2}\bigg|^y_{i,j} \right] \Bigg|^{x_i + \frac{\Delta x_{is}}{2}}_{x_i - \frac{\Delta x_{is-1}}{2}},$$

$$p^{xy}_{i+1,j} \approx \frac{1}{\Delta x_{i+1}} \left[p^y_{i,j} x + \frac{(x-x_i)^2}{2} \frac{dp}{dx}\bigg|^y_{i,j} + \frac{(x-x_i)^3}{6} \frac{d^2p}{dx^2}\bigg|^y_{i,j} \right] \Bigg|^{x_i + \Delta x_{is} + \frac{\Delta x_{is+1}}{2}}_{x_i + \frac{\Delta x_{is}}{2}}.$$

Substituting the limits of the integrals and the definition of pressure cell spacing introduced in equation (2.5) yields:

$$p^{xy}_{i-1,j} = p^y_{i,j} - \left(\frac{3\Delta x_{is-1} + \Delta x_{is-2}}{4} \right) \frac{dp}{dx}\bigg|^y_{i,j} + \left(\frac{7\Delta x_{is-1}^2 + 5\Delta x_{is-1}\Delta x_{is-2} + \Delta x_{is-2}^2}{24} \right) \frac{d^2p}{dx^2}\bigg|^y_{i,j},$$

$$p^{xy}_{i,j} = p^y_{i,j} + \left(\frac{\Delta x_{is} - \Delta x_{is-1}}{4} \right) \frac{dp}{dx}\bigg|^y_{i,j} + \left(\frac{\Delta x_{is}^2 - \Delta x_{is}\Delta x_{is-1} + \Delta x_{is-1}^2}{24} \right) \frac{d^2p}{dx^2}\bigg|^y_{i,j},$$

$$p^{xy}_{i+1,j} = p^y_{i,j} + \left(\frac{3\Delta x_{is} + \Delta x_{is+1}}{4} \right) \frac{dp}{dx}\bigg|^y_{i,j} + \left(\frac{7\Delta x_{is}^2 + 5\Delta x_{is}\Delta x_{is+1} + \Delta x_{is+1}^2}{24} \right) \frac{d^2p}{dx^2}\bigg|^y_{i,j}.$$

The corresponding linear system of equations solving for the coefficients is then:

$$\begin{bmatrix} 1 & 1 & 1 \\ -3\Delta x_{is-1} - \Delta x_{is-2} & \Delta x_{is} - \Delta x_{is-1} & 3\Delta x_{is} + \Delta x_{is+1} \\ 7\Delta x_{is-1}^2 + 5\Delta x_{is-1}\Delta x_{is-2} + \Delta x_{is-2}^2 & \Delta x_{is}^2 - \Delta x_{is}\Delta x_{is-1} + \Delta x_{is-1}^2 & 7\Delta x_{is}^2 + 5\Delta x_{is}\Delta x_{is+1} + \Delta x_{is+1}^2 \end{bmatrix} \begin{bmatrix} \beta_4 \\ \beta_5 \\ \beta_6 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

A.3. Fourth-order explicit scheme for convected velocity

The explicit fourth-order scheme evaluating the transported velocity at the left edge of a u-momentum cell reads:

$$u^y_{i,j} = \phi_1 u^{xy}_{is-2,j} + \phi_2 u^{xy}_{is-1,j} + \phi_3 u^{xy}_{is,j} + \phi_4 u^{xy}_{is+1,j}. \tag{A.3}$$

Expanding each term on the right hand side around x_i with $k = -2, -1, 0, 1$ we obtain:

$$\begin{aligned} u^{xy}_{is+k,j} &= \frac{1}{\Delta x}\int_{x_{i+k\Delta x}}^{x_i+(k+1)\Delta x} u^y dx \\ &\approx \frac{1}{\Delta x}\int_{x_{i+k\Delta x}}^{x_i+(k+1)\Delta x} u^y_{i,j} + (x-x_i)\frac{du}{dx}\bigg|^y_{i,j} + \frac{(x-x_i)^2}{2!}\frac{d^2u}{dx^2}\bigg|^y_{i,j} + \frac{(x-x_i)^3}{3!}\frac{d^3u}{dx^3}\bigg|^y_{i,j} + \\ &\qquad O(\Delta x^4)\ dx \\ &\approx \frac{1}{\Delta x}\left[u^y_{i,j}x + \frac{(x-x_i)^2}{2}\frac{du}{dx}\bigg|^y_{i,j} + \frac{(x-x_i)^3}{6}\frac{d^2u}{dx^2}\bigg|^y_{i,j} + \frac{(x-x_i)^4}{24}\frac{d^3u}{dx^3}\bigg|^y_{i,j}\right]\Bigg|^{x_i+(k+1)\Delta x}_{x_{i+k\Delta x}}. \end{aligned}$$

Substituting the limits of the integral for each cell we get:

$$\begin{aligned} u^{xy}_{is-2} &= u^y_{i,j} - \frac{3\Delta x}{2}\frac{du}{dx}\bigg|^y_{i,j} + \frac{7\Delta x^2}{6}\frac{d^2u}{dx^2}\bigg|^y_{i,j} - \frac{15\Delta x^3}{24}\frac{d^3u}{dx^3}\bigg|^y_{i,j}, \\ u^{xy}_{is-1} &= u^y_{i,j} - \frac{\Delta x}{2}\frac{du}{dx}\bigg|^y_{i,j} + \frac{\Delta x^2}{6}\frac{d^2u}{dx^2}\bigg|^y_{i,j} - \frac{\Delta x^3}{24}\frac{d^3u}{dx^3}\bigg|^y_{i,j}, \\ u^{xy}_{is} &= u^y_{i,j} + \frac{\Delta x}{2}\frac{du}{dx}\bigg|^y_{i,j} + \frac{\Delta x^2}{6}\frac{d^2u}{dx^2}\bigg|^y_{i,j} + \frac{\Delta x^3}{24}\frac{d^3u}{dx^3}\bigg|^y_{i,j}, \\ u^{xy}_{is+1} &= u^y_{i,j} + \frac{3\Delta x}{2}\frac{du}{dx}\bigg|^y_{i,j} + \frac{7\Delta x^2}{6}\frac{d^2u}{dx^2}\bigg|^y_{i,j} + \frac{15\Delta x^3}{24}\frac{d^3u}{dx^3}\bigg|^y_{i,j}. \end{aligned}$$

Solving the following linear system of equations yields the coefficients of equation

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ -3\Delta x & -\Delta x & \Delta x & 3\Delta x \\ 7\Delta x^2 & \Delta x^2 & \Delta x^2 & 7\Delta x^2 \\ -15\Delta x^3 & -\Delta x^3 & \Delta x^3 & 15\Delta x^3 \end{bmatrix}\begin{bmatrix}\phi_1\\ \phi_2\\ \phi_3\\ \phi_4\end{bmatrix} = \begin{bmatrix}1\\0\\0\\0\end{bmatrix} \rightarrow \begin{bmatrix}\phi_1\\ \phi_2\\ \phi_3\\ \phi_4\end{bmatrix} = \begin{bmatrix}-1/12\\ 7/12\\ 7/12\\ -1/12\end{bmatrix}$$

The same procedure is followed to derive the coefficients on a non-unifrom grid configuration. We avoid writing these coefficients as they are lengthy and their derivation is straightforward.

A.4. Fourth-order explicit scheme for the first derivative

The explicit fourth-order scheme evaluating the first derivative at a u-momentum cell left edge reads:

$$\left.\frac{\partial u}{\partial x}\right|^{y}_{i,j} = \phi_5 u^{xy}_{is-2} + \phi_6 u^{xy}_{is-1} + \phi_7 u^{xy}_{is} + \phi_8 u^{xy}_{is+1}. \tag{A.4}$$

This stencil is the same as that used in (A.3). Hence we only have to modify the linear system of equations such that the coefficient of the first derivative is unity and all other terms vanish.

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ -\frac{3\Delta x}{2} & -\frac{\Delta x}{2} & \frac{\Delta x}{2} & \frac{3\Delta x}{2} \\ 7\Delta x^2 & \Delta x^2 & \Delta x^2 & 7\Delta x^2 \\ -15\Delta x^3 & -\Delta x^3 & \Delta x^3 & 15\Delta x^3 \end{bmatrix} \begin{bmatrix} \phi_5 \\ \phi_6 \\ \phi_7 \\ \phi_8 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \rightarrow \begin{bmatrix} \phi_5 \\ \phi_6 \\ \phi_7 \\ \phi_8 \end{bmatrix} = \begin{bmatrix} 1/12 \\ -5/4 \\ 5/4 \\ -1/12 \end{bmatrix}$$

A.5. Fourth-order interpolation of convective velocities aligned with the momentum cell on non-uniform grids

The fourth-order interpolation of the divergence-free convective velocity aligned with the momentum cell reads:

$$\mathsf{u}^{y}_{i,j} = \lambda_1 u^{y}_{is-2,j} + \lambda_2 u^{y}_{is-1,j} + \lambda_3 u^{y}_{is,j} + \lambda_4 u^{y}_{is+1,j}. \tag{A.5}$$

Expanding each term on the right hand side around $x_{i,j}$:

$$\begin{aligned}
\mathsf{u}^{y}_{is-2,j} &= u^{y}_{i,j} - \left(\frac{\Delta x_{is-2,j}}{2} + \Delta x_{is-1,j}\right)\left.\frac{du}{dx}\right|^{y}_{i,j} + \frac{\left(\frac{\Delta x_{is-2,j}}{2} + \Delta x_{is-1}\right)^2}{2}\left.\frac{d^2u}{dx^2}\right|^{y}_{i,j} \\
&\qquad - \frac{\left(\frac{\Delta x_{is-2,j}}{2} + \Delta x_{is-1}\right)^3}{6}\left.\frac{d^3u}{dx^3}\right|^{y}_{i,j}, \\
\mathsf{u}^{y}_{is-1,j} &= u^{y}_{i,j} - \left(\frac{\Delta x_{is-1,j}}{2}\right)\left.\frac{du}{dx}\right|^{y}_{i,j} + \frac{\left(\frac{\Delta x_{is-1,j}}{2}\right)^2}{2}\left.\frac{d^2u}{dx^2}\right|^{y}_{i,j} - \frac{\left(\frac{\Delta x_{is-1,j}}{2}\right)^3}{6}\left.\frac{d^3u}{dx^3}\right|^{y}_{i,j}, \\
\mathsf{u}^{y}_{is,j} &= u^{y}_{i,j} + \left(\frac{\Delta x_{is,j}}{2}\right)\left.\frac{du}{dx}\right|^{y}_{i,j} + \frac{\left(\frac{\Delta x_{is,j}}{2}\right)^2}{2}\left.\frac{d^2u}{dx^2}\right|^{y}_{i,j} + \frac{\left(\frac{\Delta x_{is,j}}{2}\right)^3}{6}\left.\frac{d^3u}{dx^3}\right|^{y}_{i,j}, \\
\mathsf{u}^{y}_{is+1,j} &= u^{y}_{i,j} + \left(\Delta x_{is,j} + \frac{\Delta x_{is+1,j}}{2}\right)\left.\frac{du}{dx}\right|^{y}_{i,j} + \frac{\left(\Delta x_{is,j} + \frac{\Delta x_{is+1,j}}{2}\right)^2}{2}\left.\frac{d^2u}{dx^2}\right|^{y}_{i,j} \\
&\qquad + \frac{\left(\Delta x_{is,j} + \frac{\Delta x_{is+1,j}}{2}\right)^3}{6}\left.\frac{d^3u}{dx^3}\right|^{y}_{i,j}.
\end{aligned}$$

The linear system of equations solving for the coefficients becomes:

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ -\tfrac{\Delta x_{is-2,j}}{2} - \Delta x_{is-1,j} & -\tfrac{\Delta x_{is-1,j}}{2} & \tfrac{\Delta x_{is,j}}{2} & \Delta x_{is,j} + \tfrac{\Delta x_{is+1,j}}{2} \\ \left(\tfrac{\Delta x_{is-2,j}}{2} + \Delta x_{is-1}\right)^2 & \left(\tfrac{\Delta x_{is-1,j}}{2}\right)^2 & \left(\tfrac{\Delta x_{is,j}}{2}\right)^2 & \left(\Delta x_{is,j} + \tfrac{\Delta x_{is+1,j}}{2}\right)^2 \\ \left(\tfrac{\Delta x_{is-2,j}}{2} + \Delta x_{is-1}\right)^3 & -\left(\tfrac{\Delta x_{is-1,j}}{2}\right)^3 & \left(\tfrac{\Delta x_{is,j}}{2}\right)^3 & \left(\Delta x_{is,j} + \tfrac{\Delta x_{is+1,j}}{2}\right)^3 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

A.6. Fourth-order Laplacian operator on uniform grids

The fourth-order Laplacian operator is found by applying the fourth-order divergence operator over pressure cells on the fourth-order gradient operator over the momentum cells. Employing equation (3.6) we can write:

$$\begin{aligned} (D_{4x}G_{4x}p)^{xy}_{i,j} &= \frac{(G_{4x}p)^{y}_{is,j} - (G_{4x}p)^{y}_{is-1,j}}{\Delta x} \\ &= \frac{-\beta_1 (G_{4x}p)^{xy}_{is-2,j} + (\beta_1 - \beta_2)(G_{4x}p)^{xy}_{is-1,j} + (\beta_2 - \beta_3)(G_{4x}p)^{xy}_{is,j} + \beta_3 (G_{4x}p)^{xy}_{is+1,j}}{\Delta x} \end{aligned}$$

Using equation (3.7) we write the fouth-order approximation of the gradient operator over momentum cells:

$$\begin{aligned} (G_{4x}p)^{xy}_{is-2,j} &= \frac{p^{y}_{i-1,j} - p^{y}_{i-2,j}}{\Delta x} = \frac{-\beta_4 p^{xy}_{i-3,j} + (\beta_4 - \beta_5) p^{xy}_{i-2,j} + (\beta_5 - \beta_6) p^{xy}_{i-1,j} + \beta_6 p^{xy}_{i,j}}{\Delta x} \\ (G_{4x}p)^{xy}_{is-1,j} &= \frac{p^{y}_{i,j} - p^{y}_{i-1,j}}{\Delta x} = \frac{-\beta_4 p^{xy}_{i-2,j} + (\beta_4 - \beta_5) p^{xy}_{i-1,j} + (\beta_5 - \beta_6) p^{xy}_{i,j} + \beta_6 p^{xy}_{i+1,j}}{\Delta x} \\ (G_{4x}p)^{xy}_{is,j} &= \frac{p^{y}_{i+1,j} - p^{y}_{i,j}}{\Delta x} = \frac{-\beta_4 p^{xy}_{i-1,j} + (\beta_4 - \beta_5) p^{xy}_{i,j} + (\beta_5 - \beta_6) p^{xy}_{i+1,j} + \beta_6 p^{xy}_{i+2,j}}{\Delta x} \\ (G_{4x}p)^{xy}_{is+1,j} &= \frac{p^{y}_{i+2,j} - p^{y}_{i+1,j}}{\Delta x} = \frac{-\beta_4 p^{xy}_{i,j} + (\beta_4 - \beta_5) p^{xy}_{i+1,j} + (\beta_5 - \beta_6) p^{xy}_{i+2,j} + \beta_6 p^{xy}_{i+3,j}}{\Delta x} \end{aligned}$$

Upon substituting the gradients, we find:

$$\begin{aligned} (D_{4x}G_{4x}p)^{xy}_{i,j} =& \frac{1}{\Delta x^2} \{ \beta_1 \beta_4 \; p^{xy}_{i-3} + [-\beta_1(\beta_4 - \beta_5) - (\beta_1 - \beta_2)\beta_4] \; p^{xy}_{i-2} + \\ & [-\beta_1(\beta_5 - \beta_6) + (\beta_1 - \beta_2)(\beta_4 - \beta_5) - (\beta_2 - \beta_3)\beta_4] \; p^{xy}_{i-1,j} + \\ & [-\beta_1 \beta_6 + (\beta_1 - \beta_2)(\beta_5 - \beta_6) + (\beta_2 - \beta_3)(\beta_4 - \beta_5) - \beta_3 \beta_4] \; p^{xy}_{i,j} + \\ & [(\beta_1 - \beta_2)\beta_6 + (\beta_2 - \beta_3)(\beta_5 - \beta_6) + \beta_3(\beta_4 - \beta_5)] \; p^{xy}_{i+1,j} + \\ & [(\beta_2 - \beta_3)\beta_6 + \beta_3(\beta_5 - \beta_6)] \; p^{xy}_{i+2,j} \; + \beta_3 \beta_6 \; p^{xy}_{i+3,j} \}. \end{aligned}$$

In two-dimensional context, one has to solve a 13-point stencil. This wide stencil spans over three pressure cells in each spatial dimension which rises the requirement of three pressure ghost cells.

Declaration

I hereby declare that the thesis submitted is my own unaided work. All direct or indirect sources used are acknowledged as references.

München, den January 10, 2020

Khaled Boulbrachene