UNIVERSITY OF CATANIA

DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE

Level-Set Ghost Fluid Methods for Free Boundary Problems in Incompressible Euler and Navier-Stokes Equations

Valeria Artale

Ph.D in Mathematics for Technology, XXIV cycle

Advisor:

Prof. G. RUSSO

Submitted:

December 2011

Contents

List of Figures

List of Tables

Abstract

The present work is devoted to the study of free boundary problems for Euler and Navier-Stokes equations in primitive variables. The goal of the present work is to elaborate a methodology for numerical modeling of all kinds of incompressible viscous fluids, having in mind possible application to deep water, lava flow simulation and crust formation.

Our approach could be essentially divided in three fundamental components: finite difference for spatial approximation, second order accurate method for temporal discretization and level set methods for boundary representation.

The domain is discretized by a regular Cartesian grid. The boundary is described by level set methods. In this context the boundary is seen as a zero level set of a specific function. Navier-Stokes equations is solved starting from Semi-Lagrangian methods, achieving second order accuracy in time and space. Resolution of Navier-Stokes equations allows a Poisson problem for pressure as an intermediate step. This is solved by multigrid methods. The velocity and the pressure are computed by solving a single implicit system solved iteratively.

Keywords: Navier-Stokes equation, Chorin's projection method, free moving boundary, level set methods, Semi-Lagrangian methods for NS equations.

Introduction

Central task in the natural sciences lies on describing reality as accurately as possible in order to better understand natural phenomena and gain insight into the behavior of objects under given conditions. An important application area for numerical simulation is the study of the behavior of fluid flows. Flows are everywhere and as often we don't realize it. The behavior of liquids and gases, both of which are considered fluids, can be observed in many areas of everyday life, such as waterfall and transformation of clouds. Without an exhaustive knowledge of the fluid flow, it would be impossible to construct aircraft and other vehicles, indeed it is fundamental to know which is the behavior of the flow around the wings or the coachwork. Other applications could be the simulation of waves from a breaking dam, the simulation of blood flow in human circulatory system, the simulation of weather and climate modeling and so on. For modeling flows on the geophysical scale, such as whole rivers or lakes, or for evaluating the results of human activities in nature, the hydrodynamic-numerical models are nowadays indispensable.

The primary aim of this thesis is to explore numerical methods for the governing equation of fluid dynamic, the Navier-Stokes and Euler equations, and to elaborate a strategy for numerical modeling of different kind of incompressible fluid flows with free moving interface (water and lava flow, in particular), having in mind possible interface topology changes, like merging or break-up.

Different points of view are possible in the description of fluid flow: Eulerian, Lagrangian and mixed Euler-Lagrangian approaches. Eulerian methods are characterized by a coordinate system that is fixed and stationary in the frame of reference. The fluid moves in the different computational cells. Lagrangian methods are characterized by a coordinate system that moves with the fluid. The fluid is seen as a collection of particles and the motion of every particles is studied. The mixed Euler-Lagrangian methods, such as Semi-Lagrangian or Arbitrary Lagrangian-Eulerian (ALE) methods, are related to both Eulerian and Lagrangian concepts.
In general, it seems that any computational method for free-surface flow consists of flow modeling and interface modeling, but there are two important components of an algorithm to be included: spatial discretization and flow solver. Collecting the main parts of a numerical modeling procedure for fluid flows with free interface, we summarize the following step:

- *mathematical model*: primitive variables (velocity and pressure), vorticity-stream function formulation;
- *flow modeling*: Eulerian, Lagrangian, mixed Euler-Lagrangian;
- *interface modeling*: level set methods, particle level set, marker level set, Volume-Of-Fluid (VOF), hybrid level set and VOF, front capturing methods (ENO, WENO);

- *spatial discretization*: Finite Element Methods (FEM), finite difference, finite volume, meshless methods;
- *flow solver*

Multiphase flows and free surface problems are difficult to simulate, because the interface separating two fluids could develop complex pattern. There are two basic approaches that can be used to describe the flows with a material interface: “capturing” and “tracking”. In the interface capturing methods, the interface is treated as a region of steep gradient in some quantity, i.e. density. In the interface tracking method the interface is treated explicitly as a sharp discontinuity moving through the grid. We will focus our attention to capturing methods. In general, a possible classification of interface capturing methods is:

- a piecewise polynomial function (*front tracking*);
- level set of some function (*level set methods*)
- a collection of volume fractions (*volum of fluid methods*)

Piecewise polynomial methods allow high order-accurate approximations to a smoothly varying front but it is very difficult to extend it in three spatial dimension. Example of piecewise polynomial methods include boundary integral methods [2, 28, 35] and front tracking method [16, 17, 40].

Level set methods, introduce by Osher and Sethian [30], has applied to a wide variety of problems such as bubble and drops [38]. In the level set methods the interface is seen as the zero level set of some function, named *level set function*. They are very easy to implement and no specific procedure are required in order to describe topological change of the boundary. Level set methods have gained widespread popularity in the computational physics community. At the beginning, the Ghost Fluid Method (GFM) for two phase compressible flows, developed in [14], captures and preserves discontinuities across an interface represented by a suitable level set function. Using GFM the level set methods have been extended to more general cases, such as detonation [15], incompressible flow [3], two-phase incompressible flow [23, 18], Stefan problems for crystal growth [13]. In [6] a level set formulation is derived for the incompressible Navier-Stokes equations with free surface.

Finally the Volume-Of-Fluid method consists on tracking the volume of each cells that contain a portion of the interface. At each time these volumes are used to reconstruct an approximation to the interface, which allows to update the values of the fluid volume in each cell at the next time. Sussman et al. [37] proposed a Coupled Level Set/Volume-Of-Fluid (CLSVOF) method combining the advantages of both methods.

The development of numerical methods for incompressible viscous flow was strongly influenced by the Marker-And-Cell (MAC) method of Harlow et al. [19]. The method consists of a finite difference scheme with an explicit first-order time discretization. Primitive variables are located on a staggered grid, that is velocity and pressure are stored on different grid nodes. One of the main difficulties in the design of numerical methods for the Navier-Stokes equation is the development of an appropriate discrete formulation of the incompressibility condition $\nabla \cdot \mathbf{u} = 0$. The MAC method [19] enforces the incompressibility constraint by deriving a Poisson problem for pressure.

Chorin, [7, 10], developed a numerical method based on a discrete form of the Hodge decomposition, which is very close to the MAC method and showed that his method is first order in time and second order in space. Since the method is only first order accurate in time, it requires a restrictive time step to achieve a reasonable accuracy. Kim and Moin [26] started from Chorin's projection method, replacing the treatment of the non linear terms with a second order explicit Adams-Bashforth scheme and using the staggered grid of the MAC method. Kim and Moin achieved mass conservation by an implicit coupling between the continuity equation and the pressure in the momentum equations. Their method is based on fractional-step method and is second order accurate. Van Kan [22] developed a second order scheme, based on discretizing the spatial terms using the staggered grid. This reduces the PDEs to a system of differential algebraic equations and finally he used an integration technique of projection type for this system, leading to a second order scheme. Bell et al. [4] described a second order projection method for the incompressible Navier-Stokes equations: starting from Chorin's projection method, they solve diffusion-convection equations (by a second order Godunov method) to predict intermediate velocities which are projected onto the space of divergence-free vector fields. Gibou et al [25] presented an unconditionally stable second order accurate projection method. The momentum equation is updated by a second order accurate Semi-Lagrangian method and a backward difference scheme to treat the diffusion term.

The study of all these different numerical methods for fluid flows led us to a specific choice of diverse components of the numerical strategy embraced in this thesis. First of all, in order to describe the interface, we use Eulerian approach since it enables us to use a fixed structured grid. In particular, level set approach is taken. Finally we choose Eulerian and Semi-Lagrangian methods to describe fluid motion. Starting from this methods we develop an original solver in which the pressure and velocity are updated simultaneously, having in mind possible application to deep water, lava flow simulation and crust formation.

The thesis is organized as follows: first of all a synthetic description of the incompressible fluid motion equations and a description of projection method are given, followed by an overview on level-set methods; then a description of numerical methods for the Euler and Navier-Stokes equations is treated. A brief description of pressure Poisson solver by multigrid methods follows. Finally a variety of numerical examples are presented to validate the proposed computational method.

Chapter 1

The mathematical model

In this section we derive the mathematical model for unsteady viscous fluid flow, [32].

1.1 The Navier-Stokes Equations

The motion of a continuous medium is governed by the principles of classical mechanics and thermodynamics for the conservation of mass, momentum and energy. Application of these principles in an absolute frame of reference leads to the following conservation equations in integral form for mass, momentum and energy respectively

$$\frac{d}{dt}\int_{\delta} \rho \, d\delta + \int_{\Sigma} \rho \, \mathbf{U} \cdot \mathbf{n} \, d\Sigma = 0 \tag{1.1}$$

$$\frac{d}{dt}\int_{\delta} \rho \; \mathbf{U} \, d\delta + \int_{\Sigma} (\mathbf{n} \cdot \mathbf{U}) \, \rho \, \mathbf{U} - \mathbf{n}\sigma \, d\Sigma = \int_{\delta} \mathbf{F} \, d\delta \tag{1.2}$$

$$\frac{d}{dt}\int_{\delta} \rho \; E \, d\delta + \int_{\Sigma} \mathbf{n} \cdot (\rho E \mathbf{U} - \sigma \mathbf{U} + \mathbf{q}) \; d\Sigma = \int_{\delta} \mathbf{F} \cdot \mathbf{U} \, d\delta \tag{1.3}$$

where t is the time, ρ is the density, $\mathbf{U}$ is the velocity, $E = e + \mathbf{U}^2/2$ is the total specific energy per unit mass, σ is the stress tensor, $\mathbf{q}$ is the heat-flux vector, $\mathbf{F}$ is the external force, $\mathbf{n}$ is the unit outward normal to the boundary Σ of the fixed control volume δ. The energy equation is valid under the assumption that there is no source or sink of energy in δ. If the property of the medium are continuous functions of space and time and sufficiently differentiable, the conservation equations in *integral form* (1.1)-(1.3) can be transformed into an equivalent set of partial differential equations through the divergence theorem:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho\, \mathbf{U}) = 0 \tag{1.4}$$

$$\frac{\partial}{\partial t}(\rho\, \mathbf{U}) + \nabla \cdot (\rho\, \mathbf{U}\, \mathbf{U} - \sigma) = \mathbf{F} \tag{1.5}$$

$$\frac{\partial}{\partial t}(\rho\, E) + \nabla \cdot (\rho\, E\, \mathbf{U} - \sigma \mathbf{U} + \mathbf{q}) = \mathbf{F} \cdot \mathbf{U} \tag{1.6}$$

One thus obtains the equations in *divergence* or *conservative* form. Equivalent *non conservative* forms are:

$$\frac{D\rho}{Dt} + \rho \nabla \cdot \mathbf{U} = 0 \tag{1.7}$$

$$\rho \frac{D\mathbf{U}}{Dt} - \nabla \cdot \sigma = \mathbf{F} \tag{1.8}$$

$$\rho \frac{De}{Dt} - \sigma \cdot \nabla \mathbf{U} + \nabla \cdot \mathbf{q} = 0 \tag{1.9}$$

where $D/Dt = \partial/\partial t + \mathbf{U} \cdot \nabla$ is the material derivative.

The above equations are based on the Eulerian approach for the description

of the continuum motion. An alternative description is provided by the Lagrangian formulation in which the dependent variables are the characteristic properties of the material particles that are followed in their motion.
The basic dependent variables in Eqs (1.1-1.3) or (1.4-1.6) are ρ, $\mathbf{U}$ and E. Constitutive relationship for the stress tensor σ and for the heat-flux vector $\mathbf{q}$ must be added to these equations in order to obtain a closed system. For Newtonian fluids, the stress tensor is a linear function of the velocity gradient. From this definition results Newtonian's laws, also called the Navier-Stokes law for σ:

$$\sigma = -p\mathbf{I} + \tau \qquad \text{and} \qquad \tau = \lambda(\nabla \cdot \mathbf{U})\,\mathbf{I} + 2\mu\,\mathrm{Def}\,\mathbf{U} \tag{1.10}$$

where $\mathrm{Def}\,\mathbf{U} = (\nabla\mathbf{U} + \nabla\mathbf{U}^t)/2$ is the rate of deformation tensor . In this relation p is the pressure, τ is the viscous stress tensor, λ and μ are two coefficients of viscosity. Furthermore, the fluid is assumed to obey Fourier's law of heat conduction for $\mathbf{q}$:

$$\mathbf{q} = -k\nabla T \tag{1.11}$$

where T is the absolute temperature, and k is the thermal conductivity coefficient. Many fluids, in particular air and water, follow Newton's law and Fourier's law.

1.2 The Navier-Stokes Equation for the Incompressible Flow

An incompressible flow in characterized by the condition that

$$\nabla \cdot \mathbf{U} = 0 \tag{1.12}$$

This condition when introduced into the continuity equation (1.4) implies that

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho\, \mathbf{U}) = 0 \qquad \text{or} \qquad \frac{D\rho}{Dt} = 0 \tag{1.13}$$

This condition states that the density is constant along the fluid particle trajectory. In most cases one usually assumes ρ to be uniform so that this condition is satisfied identically everywhere. In the event that μ is constant, the momentum equation (1.5) reduces to the form:

$$\rho \left[\frac{\partial \mathbf{U}}{\partial t} + \nabla \cdot (\mathbf{U}\, \mathbf{U}) \right] + \nabla p - \mu \Delta \mathbf{U} = \mathbf{F} \tag{1.14}$$

This form is called the *conservative* form of the Navier-Stokes equations. In this case the unknowns are the velocity field and pressure. Applying the incompressibility condition to the (1.8), one can obtain the nonconservative form of the Navier-Stokes equations for incompressible flow equation (1.5):

$$\rho \left[\frac{\partial \mathbf{U}}{\partial t} + (\mathbf{U} \cdot \nabla)\mathbf{U} \right] + \nabla p - \mu \Delta \mathbf{U} = \mathbf{F} \tag{1.15}$$

1.3 Dimensionless form

To define dimensionless variables, characteristic values of all the variables entering the Navier-Stokes equations can be constructed from some reference quantities: a reference length L, a reference velocity U^*, a reference density ρ^* and a reference values μ^* and k^* of the coefficients of viscosity and thermal conductivity.

All other characteristic quantities can be derived from these basic ones; we choose L/U^* as characteristic time t, $\rho^* U^{*2}$ as characteristic σ, $\rho^* U^{*2}/L$ as characteristic F, U^{*2} as characteristic e and E, and $\rho^* U^{*3}$ as characteristic q, so the previous equations (nonconservative and conservative form) remain unchanged in dimensionless form. Furthermore, we use as reference value $\rho^* U^{*2}$ for p and $\mu^* U^*/L$ for τ, so that the constitutive relationship (1.10), becomes

$$\sigma = -p\mathbf{I} + \frac{1}{Re}\tau \qquad \text{and} \qquad \tau = \lambda(\nabla \cdot \mathbf{U})\,\mathbf{I} + 2\mu\,\mathrm{Def}\,\mathbf{U} \tag{1.16}$$

in which new variables are to be interpreted as nondimensional variables. The quantity $Re = V^* L \rho^*/\mu^*$is the Reynolds number, which represents the relative magnitude of inertial and viscous forces: for $Re \simeq 0$ the inertial forces are negligible against the viscous forces (highly viscous fluid), whereas for Re very large the viscous forces can be neglected.

In the following chapter, dimensionless form for Navier-Stokes equation is used.

1.4 Free boundary problem for incompressible fluid

The Navier-Stokes equation for incompressible flow are:

$$\begin{cases} \dfrac{\partial \mathbf{U}}{\partial t} + (\mathbf{U} \cdot \nabla) + \nabla p - \dfrac{1}{Re}\Delta \mathbf{U} = \mathbf{F} \\ \nabla \cdot \mathbf{U} = 0 \end{cases} \tag{1.17}$$

while the Euler equations for incompressible fluid are:

$$\begin{cases} \dfrac{\partial \mathbf{U}}{\partial t} + (\mathbf{U} \cdot \nabla) + \nabla p = \mathbf{F} \\ \nabla \cdot \mathbf{U} = 0 \end{cases} \tag{1.18}$$

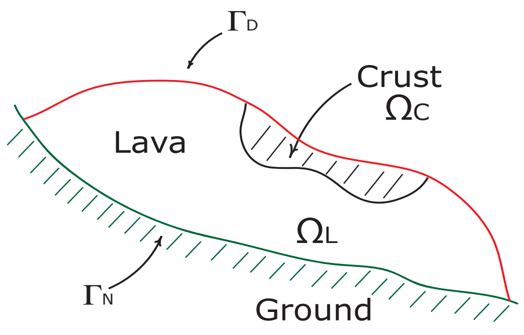


Figure 1.1: Lava flow domain example

These equation are solved in a domain Ω, in which a liquid could fill up the entire physical domain or a portion of it. For example, in Figure 1.1 a very schematic view of lava flow is represented. There is fluid in Ω_L and we solve (1.17) or (1.18) only in this region. Proper boundary condition are imposed on free surface (see chapter 4 for more details).

Chapter 2

Projection methods

Projection methods are a popular class of methods for solving the incompressible Navier-Stokes equations

$$\mathbf{U}_t + (\mathbf{U} \cdot \nabla)\mathbf{U} + \nabla p - \frac{1}{Re}\Delta \mathbf{U} = \mathbf{F} \tag{2.1}$$

$$\nabla \cdot \mathbf{U} = 0 \tag{2.2}$$

The first projection method was developed by Chorin [9, 10]. The basic idea behind this method is to use the momentum equation to solve for an intermediate velocity that is not required to be divergence-free. Then the intermediate velocity is projected to yield a discretely divergence-free velocity and a gradient field. The latter can then be used to update the pressure. Projection methods involve solving for intermediate quantities which are then used to compute the velocity field and the pressure. Because the intermediate quantities are not physical quantities, there has been much confusion related to the boundary conditions. The proper update for the pressure is another source of ambiguity in projection methods. A recent paper by Brown et al. [5] explores and clarifies the role of boundary conditions and different pressure updates in projection methods.

The following, well-known decomposition theorem helps to clarify the role of the pressure in incompressible flow and motivates projection methods [7].

Theorem 1. ***(Hodge deomposition)*** *Let Ω be a smooth, bounded domain, and $\boldsymbol{U}^*$ be a smooth vector field on Ω. The vector field $\boldsymbol{U}^*$ can be decomposed in the form*

$$\boldsymbol{U}^* = \boldsymbol{U} + \nabla\phi \tag{2.3}$$

where

$$\nabla \cdot \boldsymbol{U} = 0 \quad in \quad \Omega, \qquad \boldsymbol{U} \cdot n = 0 \quad on \quad \partial\Omega \tag{2.4}$$

Proof. Taking the divergence of the (2.3) gives the Poisson equation

$$\Delta\phi = \nabla \cdot \mathbf{U}^* \tag{2.5}$$

with the boundary condition given by the normal component of the (2.3)

$$\frac{\partial\phi}{\partial n} = \mathbf{U}^* \cdot n \quad \text{on} \quad \partial\Omega \tag{2.6}$$

The (2.5) and (2.6) define a Neumann problem for ϕ, which has a unique solution, up to an additive constant, provided the solvability condition

$$\int_\Omega \nabla \cdot \mathbf{U}^* \, d\delta = \int_{\partial\Omega} \mathbf{U}^* \cdot n \, dS \tag{2.7}$$

is satisfied. The previous equation is always satisfied, because this is simply the divergence theorem. This defines the gradient part of the decomposition, and the divergence-free field is then defined by

$$\mathbf{U} = \mathbf{U}^* - \nabla\phi \tag{2.8}$$

$\square$

The previous proof describes the general procedure for decomposing an arbitrary smooth vector field into a gradient field and a divergence-free field. Let $\mathcal{P}$ denote the operator which takes vector fields to divergence-free vector fields, as in the decomposition theorem. Then $\mathcal{P}$ is a projection operator

$$\mathcal{P}(\mathbf{U}^*) = \mathbf{U} \tag{2.9}$$

and

$$(I - \mathcal{P})(\mathbf{U}^*) = \nabla\phi \tag{2.10}$$

Applying the projection operator to (2.1) gives the equation

$$\mathbf{U}_t = \mathcal{P}\left(-\mathbf{U}\cdot\nabla\mathbf{U} + \frac{1}{Re}\Delta\mathbf{U} + F\right) \tag{2.11}$$

in which the pressure has been eliminated. This form is equivalent to the original equations (2.1) - (2.2).

The proof of the Hodge decomposition suggests of a scheme for evolving velocity and pressure in time: advance the momentum equation (2.1) to determine an intermediate velocity which is not required to be divergence-free; find the pressure by solving a Poisson problem for pressure and then update the velocity filed by (2.8)

Chapter 3

Level set method

The level set method was created by Osher and Sethian in 1988. The strength of this method lies in implicitly representing dynamic surfaces. This greatly simplifies many of the problems one faces with explicit representations, such as merging of different surfaces.
Implicit functions are widely used in the mathematics and graphics communities for modeling complex dynamic surface such as the surface of water. The strength of the level set method is in modeling and animating implicit functions that dynamically change over time.

3.1 Implicit function

The distinguishing property of an implicit function is that the interface exists where the implicit equation evaluates to zero. For example, in one spatial dimension, suppose we dived the real line in three different region by the points $x_1 = -1$ and $x_2 = 1$. We refer to $\Omega^- = (-1, 1)$ as the *inside* region, $\Omega^+ = (-\infty, -1) \cup (1, \infty)$ as the *outside* region and the set of two points $\partial\Omega = -1, 1$ as the *interface*. This is an *explicit* representation of the interface, because we explicitly write down the points that belong to it. As we said before, an *implicit* interface representation defines the interface as a zero isocontour of some function, called *level set* function. In this case, the level

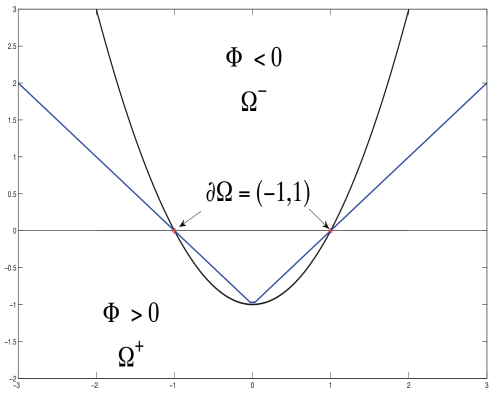


Figure 3.1: 1d Implicit representation of $\partial\Omega = \{-1, 1\}$ and level set function $\phi(x) = x^2 - 1$

set function is $\phi(x) = x^2 - 1$. One can merely extend previous definitions in two dimensions. For example, consider $\phi(\mathbf{x}) = x^2 + y^2 - 1$ where the interface defined by the zero isocontour $\phi(\mathbf{x}) = 0$ is the unit circle defined by $\partial\Omega = \{\mathbf{x} : |\mathbf{x}| = 1\}$. An explicit representation of an interface only contains information about the interface itself (only a finite set of points). That is, given an explicit function, we can find the position of the interface, but we don't have any information about the rest of space in which the interface exists. Instead, an implicit representation contains information about the entire space of the function, including the position of the interface. As it is shown in Figure 3.1 and Figure 3.2, the implicit level set function ϕ represents

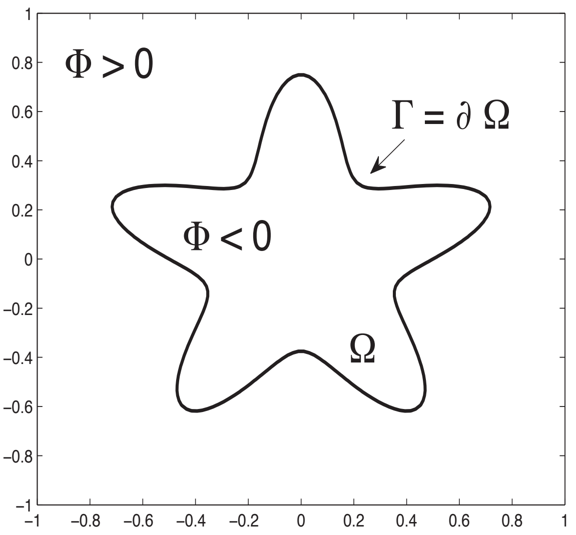


Figure 3.2: Implicit representation of flower shaped domain

the region with the following properties:

$$\begin{cases} \phi\,(x,y) > 0 & (x,y) \notin \Omega \\ \phi\,(x,y) = 0 & (x,y) \in \partial\Omega \\ \phi\,(x,y) < 0 & (x,y) \in \Omega \end{cases} \tag{3.1}$$

In Figure 3.2 is represented a flower-shaped domain: its the level set function has the following expression, [31]

$$\phi = r - 0.5 - (y^5 + 5\,x^4\,y - 10\,x^2\,y^3)/(3\,r^4)$$

where $r = \sqrt{x^2 + y^2}$. The other major difference between explicit-implicit representation is how we represent the function in discrete space. When we are dealing with an explicit object, we make use of splines or triangles

for representing its interface, or surface. For a rigid body, or nearly rigid body object, this explicit framework works fine. However, working with this methodology becomes non-trivial when we want to represent a surface that dynamically changes topology, such as water or fire.There is no easy way of dynamically merging and splitting the surface as it changes form.
Discrete representations of implicit functions are handled using a Cartesian grid that bounds the space that the function exists in. Then we simply represent the space by storing the value of as each grid node. To find the interface itself, we just need to interpolate between nodes that have negative values for ϕ and nodes that have positive values for ϕ.

3.2 Geometric properties

By using level set methods, it is very easy define domain with complex shape. Indeed another advantage of implicit level set representation is the simplicity of performing boolean operations. If ϕ_1 and ϕ_2 are two different implicit functions, then $\phi = min(\phi_1, \phi_2)$ is the level set function representing the *union* of the interior regions of ϕ_1 and ϕ_2. At the same way, $\phi = max(\phi_1, \phi_2)$ is the level set function representing the *intersection* of the interior regions of ϕ_1 and ϕ_2. In Figure 3.4 is shown the zero isocontour of the minimum between two level set functions.
The gradient $\nabla\ \phi$ is perpendicular to the isocontours of ϕ and points in the direction of increasing ϕ. It is useful to compute the unit *normal* $\mathbf{n}$ in

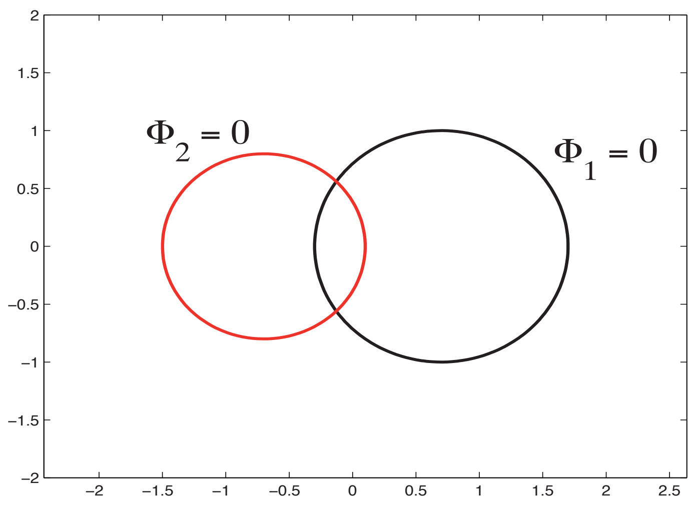


Figure 3.3: Zero isocontour of two level set functions ϕ_1 end ϕ_2

outward direction to the interface:

$$\mathbf{n} = \frac{\nabla\,\phi}{|\nabla\,\phi|}$$

The mean *curvature* of the interface is defined as the divergence of the normal:

$$\kappa = \nabla \cdot \mathbf{n}$$

so that $\kappa > 0$ for convex regions, $\kappa < 0$ for concave regions and $\kappa = 0$ for a plane. As we said, the level set approach is very useful for computing interface evolution. The interface Γ is the zero level set of a level set ϕ:

$$\Gamma = \{\mathbf{x} \in \mathbb{R}^n \; : \; \phi(\mathbf{x}) = 0\}$$

By adding a temporal variable, it easy to capture the evolution of the interface. If $\phi(\mathbf{x})$ is the level set function and Γ is the zero isocontour, one can

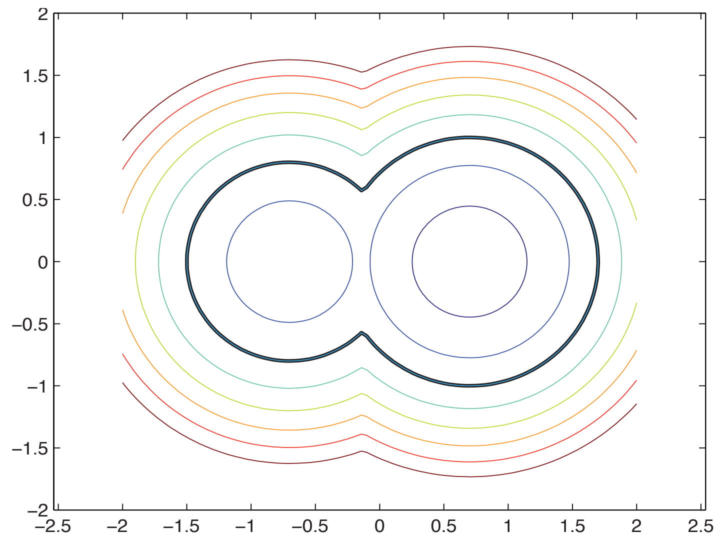


Figure 3.4: Isocontour of $\phi = \min(\phi_1, \phi_2)$; in black the zero level set

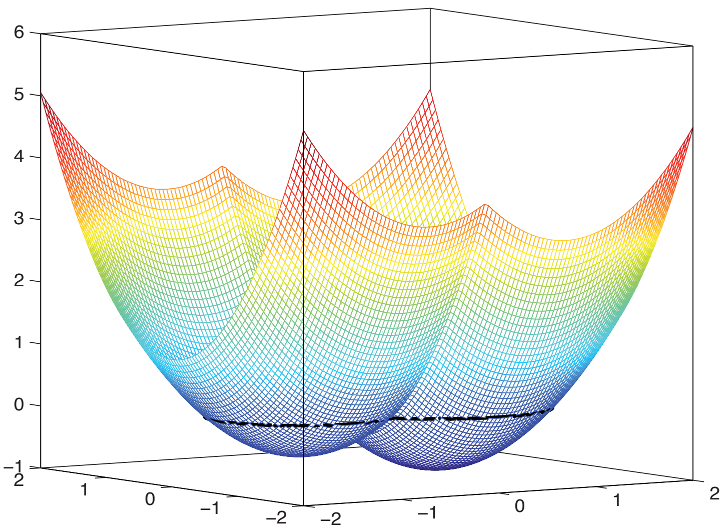


Figure 3.5: Level set function $\phi = \min(\phi_1, \phi_2)$; in black the zero level set

write $\phi(\mathbf{x}, t)$ such that the interface is a function of the time:

$$\Gamma(t) = \{\mathbf{x} \in \mathbb{R}^n \; : \; \phi(\mathbf{x}, t) = 0\}$$

The interface $\Gamma(t)$ is updated by solving a transport equation for ϕ:

$$\frac{\partial \phi}{\partial t} + \mathbf{v} \cdot \nabla \phi = 0 \tag{3.2}$$

where $\mathbf{v}$ is a velocity field . This equation follows from the fact that if a point $\mathbf{x}(t)$ lies on the interface then:

$$\frac{d\phi}{dt}(\mathbf{x}(t), t) = 0 \tag{3.3}$$

If the normal component of the velocity is considered:

$$v_n = \mathbf{v} \cdot \frac{\nabla \phi}{|\nabla \phi|} \tag{3.4}$$

the (3.2) becomes:

$$\frac{\partial \phi}{\partial t} + v_n \, |\nabla \phi| = 0 \tag{3.5}$$

3.3 Signed Distance functions and Reinitialization

Exact solution of (3.2) reflects the position of an interface moving with speed $\mathbf{v}$. The level set function can develop large or small gradients around the zero level set, so that the numerical solution of the level set equation will

become progressively less accurate. In many numerical implementations of the level set method, the level set function is replaced as a distance function by reinitializing the level set function [34]. The signed distance function (shortly sdf) is a particular implicit functions, such that at each point in space, the value of ϕ is the signed distance to the closest point on the interface Γ (zero level set of sdf). The level set reinitialization step replaces the original level set function with a new level set function that shares the same zero isocontour, but is a distance function.
Numerically one can reinitialize the level set function by solving the following equation to the steady state:

$$\phi_t + \mathcal{S}(\phi^0)\,(|\nabla\phi| - 1) = 0 \tag{3.6}$$

with $\phi(\mathbf{x}, 0) = \phi^0(\mathbf{x})$ initial condition and $\mathcal{S}(\phi^0)$ sign function. The zero level set $\phi^0(\mathbf{x})$ represents the location of the interface. Steady state solution of the previous equation gives a distance function with $|\nabla\phi| = 1$ and solving up to time τ, $\phi(\mathbf{x}, \tau)$ is the signed distance function for all points within distance τ from the interface.

3.3.1 First order approximation

The (3.6) equation can be discretized by using upwind methods. The first order 1D version used in [38] is given by

$$\phi_i^{n+1} = \phi_i^n - \Delta t\, S(\phi_i^0) G(\phi_i^n) \tag{3.7}$$

where:

$$G(\phi_i) = \begin{cases} \max(|a^+|^2, |b^-|) - 1 & \phi_i^0 > 0 \\ \max(|a^-|^2, |b^+|) - 1 & \phi_i^0 < 0 \end{cases} \tag{3.8}$$

with

$$a = D_x^- \phi_i = (\phi_i - \phi_{i-1})/\Delta x$$

$$b = D_x^+ \phi_i = (\phi_{i+1} - \phi_i)/\Delta x$$

and for any real number k, it is $k^- = \min(k, 0)$ and $k^+ = \max(k, 0)$. The method used to solve the equation (3.6) is usually upwind methods, where the discrete derivatives are computed by upwind differencing according to the direction of the characteristics. In particular, this means that when differencing across the interface, this property will be violated. So the discretization of the derivatives near the interface is not truly upwind, in the sense that part of the information is coming from the wrong side of the level set.
So the method needs to be modified. Russo and Smereka in [38] proposed a new upwind scheme obteined by a simple correction of the previous schemes. The new method is:

$$\phi_i^{n+1} = \begin{cases} \phi_i^n \ \frac{\Delta t}{h}(S(\phi_i^0)\ |\phi_i^n| \ - D_i) & \text{se } \phi_i^0\, \phi_{i-1}^0 < 0 \text{ oppure } \phi_i^0\, \phi_{i+1}^0 < 0 \\ \\ \phi_i^n - \Delta t\, S(\phi_i^0) G(\phi_i) & \text{otherwise} \end{cases}$$

The scheme is applicable in a dimension by dimension framework. Since the location of the interface is preserved, the loss of area/volume becomes independent of the number of iteration of the algorithm.

3.3.2 High order approximation

The method can be extended to general ENO reconstruction in order to develop higher order schemes. In this section a second order scheme is presented, [34].
Given five points of the stencil around the node x_i, $(X(k), H(k))\,, k \in [-2, 2]$ such that $X(k) = x_{i+k}$, $H(k) = \phi_{i+k}$. The left and right derivatives a and b are given as follows. First of all, divided differences are computed:

$$\begin{aligned} D^1_{k+\frac{1}{2}} &= \frac{H(k+1) - H(k)}{X(k+1) - X(k)}, \qquad k \in [-2, 1] \\ D^2_k &= \frac{D^1_{k+\frac{1}{2}} - D^1_{k-\frac{1}{2}}}{X(k+1) - X(k-1)}, \quad k \in [-1, 0] \end{aligned}$$

Define the minmod function

$$MM(\alpha, \beta) = \begin{cases} \alpha \ \text{ se } \ |\alpha| \leq |\beta| & \text{e} \quad \alpha\beta > 0 \\ \beta \ \text{ se } \ |\alpha| > |\beta| & \text{e} \quad \alpha\beta > 0 \\ 0 \ \text{ se } \ \alpha\beta \leq 0. \end{cases} \tag{3.9}$$

Then:

$$a = D^1_{-\frac{1}{2}} + MM(D^2_1, D^2_2)\,\left(X(0) - X(-1)\right)$$

$$b = D^1_{\frac{1}{2}} + MM(D^2_2, D^2_3)\,\left(X(0) - X(1)\right)$$

If the point x_i is within one grid cell from the interface, then the stencil will include the intersection of the function with the axis, for example by fitting a third-order polynomial through the grid points near the zero. The extension of the scheme to two dimension is straightforward.

3.4 Reinitialization Results

Consider the square domain $[-2, 2]^2$ and the level set function:

$$\phi_0 = ((x-1)^2 + (y-1)^2 + 0.1)\sqrt{x^2 + y^2 - 1};$$

The zero level set is the circle centered in the origin and radius 1. The function ϕ_0 isn't a signed distance function.
In Figure 3.6 we can see the result of applying second order scheme. The grid is 100x100 , the time step is $\Delta t = 0.5\Delta x$ and the number of iteration is $k = 0, 10, 20, 40$, from the top left to the lower right.
A characteristic property of the signed distance function is that $|\phi| = 1$. In Figure 3.7 the gradient of ϕ is plotted as a function of the number of iterations.

Observing Figure 3.6 and Figure 3.7, one could see that after 50 iteration the level set function is a signed distance function in the all domain. If the method is used as an intermediate step of a more general algorithm, it should be useful compute the signed distance function only in a narrow band, near the zero level set.

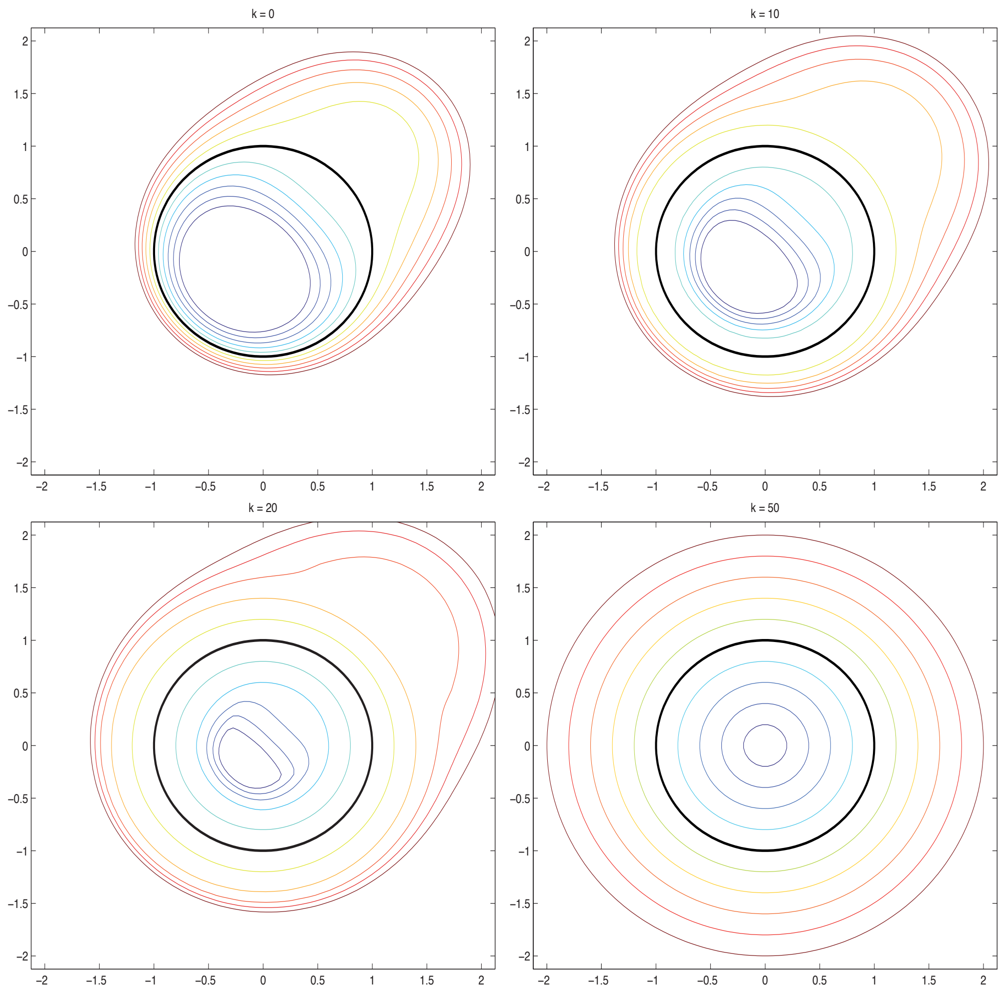


Figure 3.6: Level-set reinitialization; in black the zero level set of ϕ; The contours run from -1 to 1 and are spaced by 0.2

3.5 Final remarks: Discretization of the Domain

The whole computational domain Ω is a rectangular box discretized by a regular square grid. In Ω a level-set function ϕ, is defined such that its zero isocontour Γ is the boundary of the physic domain. Sometimes it should be

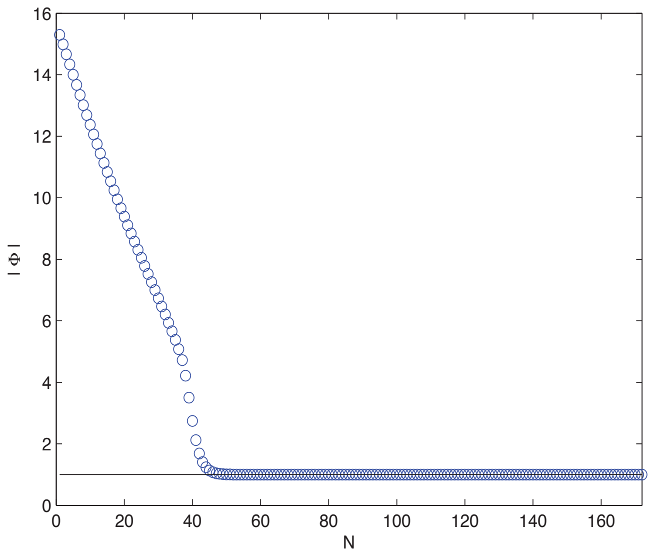


Figure 3.7: $|\phi|$ is plotted as a function of iterations number

useful to define more than one level set function in order to describe domain with particular shape, i.e. domain depicted in Figure 3.8, in which we would like to describe the motion of a fluid. In this context, we define a level set function $\phi_f(x,t)$ with zero level set $\Gamma_f(x,t)$ to define fluid surface and a level set function $\phi_g(x)$ with zero level set $\Gamma_g(x)$ to define ground surface. The

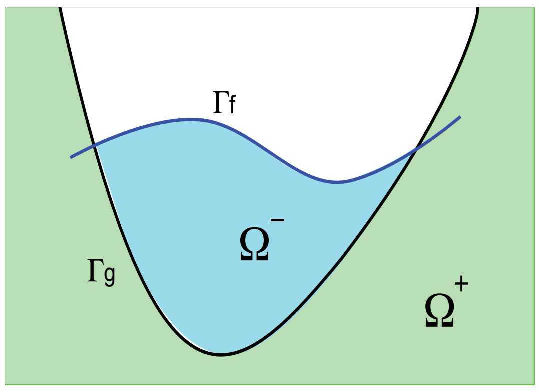


Figure 3.8: General Domain

region in which we want to solve the problem is Ω^- that is equivalent to the region in which the level set function $\phi(x,t) = \min(\phi_f, -\phi_g)$ is negative. The domain Ω is discretized by a regular square grid. Scanning grid nodes

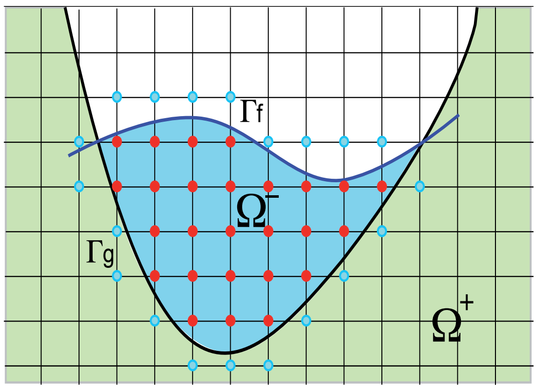


Figure 3.9: Discretization of General Domain

$x_{j,k}$, it is possible to identify which points are inside Ω^- and which points are outside Ω^- but close to an inside points, called ghost points. So active grid nodes are composed by two sets of points, defined as:

- inside, $\mathcal{P}_i = \{x_{j,k} \in \Omega : \phi(x_{j,k}) < 0\}$, marked with a red dots in Figure 3.9
- ghost, $\mathcal{P}_g = \{x_{j,k} \in \Omega : \phi(x_{j,k})\phi(x_{j+1,k}) < 0$ or $\phi(x_{j,k})\phi(x_{j-1,k}) < 0$ or $\phi(x_{j,k})\,\phi(x_{j,k-1}) < 0$ or $\phi(x_{j,k})\,\phi(x_{j,k+1}) < 0\}$, marked with a blu dots in Figure 3.9

For simplicity we omitted the t term in function ϕ.

Chapter 4

Numerical methods for the Navier-Stokes equation

In this chapter I'm going to describe the main methods studied and developed in my research. We focused our attention on MAC method, semi-lagrangian method and a new iterative technique to solve Navier-Stokes equations.

4.1 Mac method

The Mac method was probably the first primitive variable method for incompressible flow. The first paper on this approach was written by Harlow and Welch ([19]). They called this method marker-and-cell (MAC) method and can be applied to time dependent motion of viscous, incompressible fluid in two-dimensional Cartesian grid. The solution technique makes use of finite-difference approximations applied to the Navier-Stokes equations. The dependent variable are velocity and pressure and neither vorticity nor stream function is considered. It is assumed that velocity field is conservative.

The nondimensional form of the Navier-Stokes equations is

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \tag{4.1}$$

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} + \frac{\partial uv}{\partial y} + \frac{\partial p}{\partial x} = \frac{1}{Re}(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + gx) \tag{4.2}$$

$$\frac{\partial v}{\partial t} + \frac{\partial uv}{\partial x} + \frac{\partial v^2}{\partial y} + \frac{\partial p}{\partial y} = \frac{1}{Re}(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + gy) \tag{4.3}$$

where Re is the Reynold's number and $g = (gx, gy)$ is external forces (i.e. gravity). Initial conditions are required and need to satisfy the (4.1). The MAC method is based on finite difference discretisations and on the solution of a Poisson equation to determine the pressure behaviour. The domain is discretized using a *staggered grid*, in which the different unknown variables are not located at the same grid points, but are evaluated at different nodes: the pressure is located in the cell centers, the horizontal velocity u in the midpoints of the vertical edges, the vertical velocity v in the midpoints of the horizontal edges (Fig. 4.1). Discretization of (4.1) gives

$$\frac{u_{j+1/2,k} - u_{j-1/2,k}}{\Delta x} + \frac{v_{j,k+1/2} - u_{j,k-1/2}}{\Delta y} = 0 \tag{4.4}$$

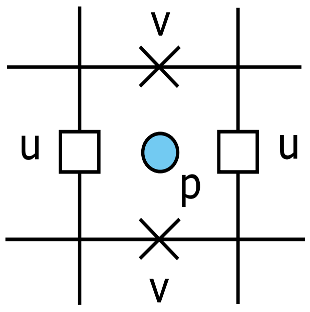


Figure 4.1: Staggered grid

While the (4.2) becames

$$u^{n+1}_{j+1/2,k} = F^n_{j+1/2,k} - \frac{\Delta t}{\Delta x}\left(p^{n+1}_{j+1,k} - p^{n+1}_{j,k}\right) \tag{4.5}$$

where

$$F^n_{j+1/2,k} = u^n_{j+1/2,k} - \Delta t\left(\frac{u^2_{j+1,k} - u^2_{j,k}}{\delta x} + \frac{(u\,v)_{j+1/2,k+1/2} - (u\,v)_{j+1/2,k-1/2}}{\Delta y}\right)^n +$$
$$\frac{\Delta t}{Re}\left(\frac{u_{j-1/2,k} - 2u_{j+1/2,k} + u_{j+3/2,k}}{\Delta x^2} + \frac{u_{j+1/2,k-1} - 2u_{j+1/2,k} + u_{j+1/2,k+1}}{\Delta y^2}\right)^n + \Delta t\, gx$$

The momentum equation for v becomes:

$$v^{n+1}_{j,k+1/2} = G^n_{j,k+1/2} - \frac{\Delta t}{\Delta y}\left(p^{n+1}_{j,k+1} - p^{n+1}_{j,k}\right) \tag{4.6}$$

where

$$G^n_{j,k+1/2} = v^n_{j,k+1/2} - \Delta t \left(\frac{v^2_{j,k+1} - v^2_{j,k}}{\Delta y} + \frac{(u\,v)_{j+1/2,k+1/2} - (u\,v)_{j-1/2,k+1/2}}{\Delta x} \right)^n +$$

$$\frac{\Delta t}{Re} \left(\frac{v_{j,k-1/2} - 2v_{j,k+1/2} + v_{j,k+3/2}}{\Delta x^2} + \frac{v_{j-1,k+1/2} - 2v_{jk+1/2,} + v_{j+1,k+1/2}}{\Delta y^2} \right)^n + \Delta t\, gy$$

The values $u_{j+1,k}$ and $(uv)_{j+1/2,k+1/2}$ are simple averages computed as follow:

$$u_{j+1,k} = \frac{u_{j+1/2,k} + v_{j+3/2,k}}{2}$$

and

$$(u\,v)_{j+1/2,k+1/2} = \frac{(u_{j+1/2,k} + u_{j+1/2,k+1})\,(v_{j+1,k+1/2} + v_{j,k+1/2})}{4}$$

To compute the pressure is sufficient substitute the (4.5) e (4.6) nella (4.4)

$$\left(\frac{p^{n+1}_{j-1,k} - 2\,p^{n+1}_{j,k} + p^{n+1}_{j+1,k}}{\Delta x^2} + \frac{p^{n+1}_{j,k-1} - 2\,p^{n+1}_{j,k} + p^{n+1}_{j,k+1}}{\Delta y^2} \right) =$$

$$\frac{1}{\Delta t} \left(\frac{F^n_{j+1/2,k} - F^n_{j-1/2,k}}{\Delta x} + \frac{G^n_{j,k+1/2} - G_{j,k-1/2}{}^n}{\Delta y} \right) \tag{4.7}$$

This equation is can be solved each time step using iterative techniques or direct Poisson solver. In this contest we use multigrid method.

The discrete equation to advance the velocity field can be written in vector form as:

$$\mathbf{u}^{n+1} = \mathbf{F}^n - \Delta t \nabla P^{n+1} \qquad \mathbf{F} = (F, G) \tag{4.8}$$

The Poisson equation for the pressure can be written as

$$\Delta P^{n+1} = \frac{1}{\Delta t} \nabla \cdot \mathbf{F}^n \tag{4.9}$$

The MAC method is very close to the projection method proposed by Chorin and seen in the previous section.

Since discretized momentum equation are treated in an explicit manner, a restriction on the maximum time step for a stable solution is required:

$$0.25(\|u\| + \|v\|)^2 \Delta t Re \leq 1$$

$$\frac{\Delta t}{Re \Delta x^2} \leq 0.25 \qquad \text{assuming that} \qquad \Delta x = \Delta y$$

The method seen before has the advantage of being very simple to implement. One weakness is that it is an explicit method, and not suitable for simulations of low Reynolds number flows or Stokes flows. For an explicit method, the time step should be less than the viscous time scale and it should be possible that the time step becomes very small and computational time huge. Moreover the method is second order in space but only first order in time. We overcome all these difficulties using implicit semi-lagrangian methods.

4.1.1 Boundary condition

The discretization of the momentum equation involves values of the velocity and pressure that lie out the computational domain.

Velocity Boundary Condition

In general, on rigid wall one could choose between *free-slip* (velocity component tangent to the boundary vanish along the normal derivative of the velocity component tangent to the boundary) and *no-slip* boundary condition (the fluid has the same velocity of the boundary). Other boundary condition are *outflow* (the total velocity does not change in normal direction), *inflow* (the velocity are assigned) and *periodic* boundary conditions. The choice of which type of previous boundary condition depends on the particular problem describing the fluid flow.

On free surface the situation is quite different. Let's consider the situation depicted in Figure 4.2, in which only horizontal component of the velocity is shown. There is liquid in Ω^- and for example gas in Ω^+. The interface between fluid and gas Γ is the zero level set of a suitable function $\phi(x,t)$. In discretizing the momentum equation for the horizontal component u of the velocity in the grid node $(i+1/2,j)$, the value $u(i+1/2,j+1)$, where $(i+1/2,j+1)$ is a ghost point, is required. Since the velocity field is defined only in Ω^-, we use a strategy to extrapolate the velocity from inside region Ω^- to outside region Ω^+. The simplest extrapolation that can be performed is the constant extrapolation.
Given a function u define d only in a portion of a space and a level set function ϕ such that $\phi <= 0$ defines the region where u is known and $\phi > 0$ define the region where u is unknown. Function u is extrapolated as a constant along

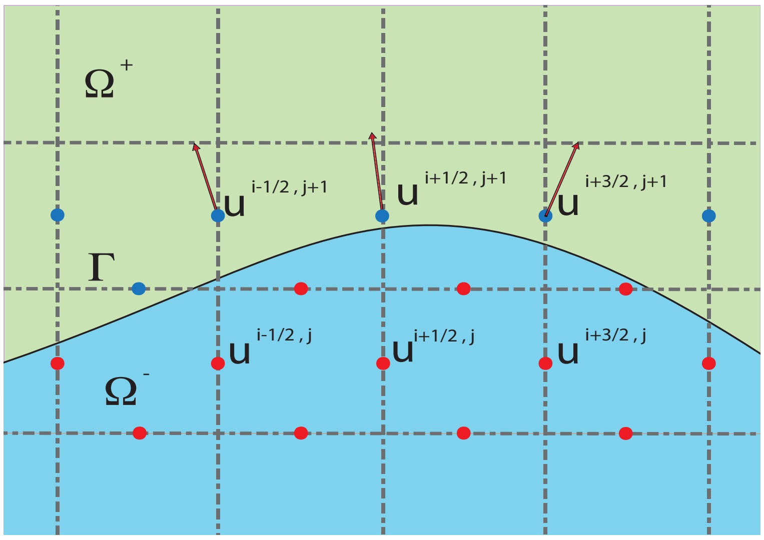


Figure 4.2: Velocity boundary condition on free surface

the normal $\mathbf{n}$ defined as:

$$\mathbf{n} = \frac{\nabla\,\phi}{|\nabla\,\phi|}$$

To extrapolate in normal direction, the following PDE is used

$$\frac{\partial u}{\partial t} + \mathbf{n}\cdot\nabla u = 0 \tag{4.10}$$

This equation is solved until the steady state is reached, since at this state $\mathbf{n}\cdot\nabla u = 0$, that means that u will be constant along the characteristic direction $\mathbf{n}$. Sometimes the u value are required only in a narrow band of Γ, so the (4.10) is solved only for a few time step [1, 14].

Pressure Boundary Condition

Solving the pressure Poisson equation (4.9) requires boundary value for the pressure. These result from multiplying the discrete momentum equation

(4.8) with $\mathbf{n} = (n_x, n_y)$

$$\begin{aligned} \nabla P^{n+1} \cdot \mathbf{n} &= \frac{\partial p^{n+1}}{\partial x} n_x + \frac{\partial p^{n+1}}{\partial y} n_y = \\ &= -\frac{1}{\Delta t}((u^{n+1} - F^n) n_x + (v^{n+1} - G^n) n_y) \end{aligned} \tag{4.11}$$

On a left vertical rigid wall the normal is $\mathbf{n} = (-1, 0)$ and the discrete (4.11)

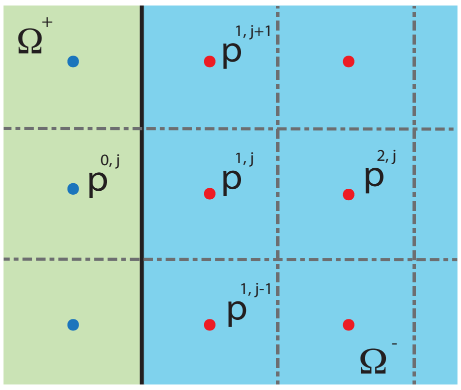


Figure 4.3: Pressure boundary condition

becomes:

$$\frac{p^{n+1}_{0,j} - p^{n+1}_{1,j}}{\Delta x} = \frac{1}{\Delta t}(u^{n+1}_{1/2,j} - F^n_{1/2,j}) \tag{4.12}$$

If we now instert this in (4.13), we obtain

$$\begin{aligned} &\frac{p^{n+1}_{2,j} - p^{n+1}_{1,j}}{\Delta x^2} + \frac{p^{n+1}_{1,j+1} - 2\, p^{n+1}_{1,j} + p^{n+1}_{1,j-1}}{\Delta y^2} = \\ &\frac{1}{\Delta t}\left(\frac{F^n_{3/2,j} - u^{n+1}_{1/2,j}}{\Delta x} + \frac{G^n_{1,j+1/2} - G^n_{1,j-1/2}}{\Delta y}\right) \end{aligned} \tag{4.13}$$

This equation does not depend on the value $F^n_{1/2,j}$, that can be selected arbitrarily. The simplest choice is $F^n_{1/2,j} = u^{n+1}_{1/2,j}$, which leads to a zero Neumann boundary condition for pressure. Analogous results can be obtained on other rigid walls.

Because of Neumann boundary condition, the system $\mathcal{A}x = b$ associated to pressure Poisson problem is singular. The common strategy used in this case consists in considering an augmented matrix, [20, 25]

$$\begin{pmatrix} \mathcal{A} & s \\ s & 0 \end{pmatrix} \begin{pmatrix} x \\ \alpha \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix}$$

where s denotes the null eigenvector of $\mathcal{A}$. In this way, though the original matrix is singular, the augmented matrix is not.

Finally, on free surface, we impose $p = 0$.

4.2 High order projection method: second order semi-lagrangian

We first consider the advection-diuffsion equation

$$\frac{\partial \phi}{\partial t} + \mathbf{u} \cdot \nabla \phi = \mu \, \Delta \phi \tag{4.14}$$

and write it in lagrangian form

$$\frac{d\,\phi}{d\,t} = \mu\,\Delta\phi \tag{4.15}$$

$$\frac{d\,\mathbf{x}}{d\,t} = \mathbf{u}(\mathbf{x}, t) \tag{4.16}$$

Semi-lagrangian methods are a generalization of the classical method of characteristics proposed by Courant - Isaacson - Rees for hyperbolic equation. From a lagrangian point of view, solving previous equations means solve the (4.15) along the characteristic lines (4.16). This leads to a decoupling of the advection and diffusion terms and a scheme that is unconditionally stable. But, as the fluid particles move, they may reach points that are not grid nodes, so the new mesh could result irregular and distorted. Hence, each time step a remeshing is required and this operation may be very expensive from a computational point of view.

In the semi-lagrangian approach, the computational mesh is fixed, no remeshing is requires. At each time step, a discrete set of particle is tracked backward over a single time step along its characteristic line up to the departure points. In other word, the solution is reconstructed by integrating the equation along characteristic curves, starting from any grid point x_i and tracking back the departure point x_d in the upwind direction. The (4.15) becomes:

$$\frac{\phi^{n+1} - \phi^n_d}{\Delta t} \;=\; \mu\,\Delta\left(\frac{\phi^{n+1} + \phi^n_d}{2}\right) \tag{4.17}$$

$$\frac{d\,\mathbf{x}}{d\,t} = \mathbf{u}(\mathbf{x}, t), \qquad \mathbf{x}^{n+1} = \mathbf{x}(t^{n+1}) = x_i \tag{4.18}$$

where ϕ_d^n is the value of ϕ at the departure point x_d from which the characteristic curve originates from at time level n and x_i is any grid point that is an arrival point. Using the explicit second order midpoint rule for locating the departure point, we have:

$$\hat{\mathbf{x}} = \mathbf{x}^{n+1} - \frac{\Delta t}{2}\,\mathbf{u}^n(\mathbf{x}^{n+1}) \tag{4.19}$$

$$\mathbf{x}_d^n = \mathbf{x}^{n+1} - \Delta t\,\mathbf{u}^{n+1/2}(\hat{\mathbf{x}}) \tag{4.20}$$

where $\mathbf{u}^{n+1/2}$ is a linear combination of the velocities at the two previous time steps t^n and t^{n-1}, that is $\mathbf{u}^{n+1/2} = \frac{3}{2}\mathbf{u}^n - \frac{1}{2}\mathbf{u}^{n-1}$. The point $\hat{\mathbf{x}}$ is not necessary a grid point, so an interpolation procedure is required to compute the value $\mathbf{u}^{n+1/2}(\hat{\mathbf{x}})$. At the same way the value ϕ_d^n need to be interpolated from ϕ^n in $\mathbf{x}_d^n$.

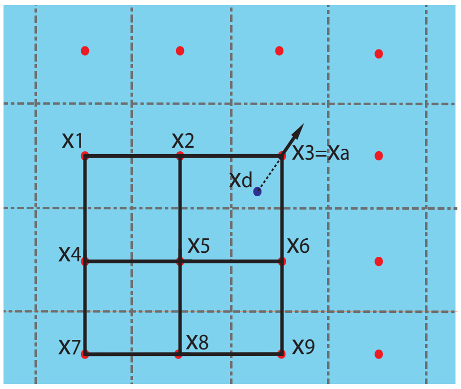


Figure 4.4: Arrival and departure point of a characteristic curve and nine point stencil in upwind direction

In Figure 4.4 there is a sketch of Semi-Lagrangian transport: x_d is the departure point of characteristic curve arriving to x_a, grid node. In order to compute ϕ_d^n, biquadratic interpolation is used, which is constructed from the value of solution at nine different points. The stencil is in upwind direction with respect to the velocity component. In particular, once that the cell in which the point x_d is determined and the point $x_{i,j}$ is the grid node close to x_d in the upwind direction, the stencil has the following structure

$$S_x = [i, i+a, i+2\,a] \qquad \text{and} \qquad S_y = [j, j+b, j+2\,b] \tag{4.21}$$

where $a = sign(u_{i,j})$ and $b = sign(v_{i,j})$. In Figure 4.4 the $x_{i,j}$ is x_3.

Let's write now the Navier-Stokes equations in non dimensional and compact form:

$$\mathbf{U}_t + (\mathbf{U}\cdot\nabla)\mathbf{U} + \nabla p - \frac{1}{Re}\Delta\mathbf{U} = \mathbf{F} \tag{4.22}$$

$$\nabla\cdot\mathbf{U} = 0 \tag{4.23}$$

and rewrite them in semi-lagrangian form

$$\frac{d\,\mathbf{U}}{d\,t} = -\nabla p + \frac{1}{Re}\Delta\mathbf{U} + \mathbf{F} \tag{4.24}$$

$$\nabla\cdot\mathbf{U} = 0 \tag{4.25}$$

The Crank-Nicolson scheme is often used for implicit treatment of viscous term. However , Xiu et al. [41] showed that the method seems to be second-order in time, but it is fill first-order in time. The backward differentiation formula (BDF) offers a more convenient choice for its unconditional stability and simplicity, [25].

Using semi-lagrangian and BDF, the momentum equation becomes:

$$\frac{1}{\Delta t}\left(\frac{3}{2}\mathbf{U}^{n+1} - 2\mathbf{U}_d^n + \frac{1}{2}\mathbf{U}_d^{n-1}\right) = -\nabla p^{n+1} + \frac{1}{Re}\Delta\mathbf{U}^{n+1} + F^{n+1} \tag{4.26}$$

where $\mathbf{U}_d^n$ is the velocity at the departure point $\mathbf{x}_d^n$ at time level n and $\mathbf{U}_d^{n-1}$ is the velocity at the departure point $\mathbf{x}_d^{n-1}$ at time level $n-1$. The departure point $\mathbf{x}_d^n$ is computed by solving (4.27)-(4.28) over one single time step Δt, while the point $\mathbf{x}_d^{n-1}$ is given by:

$$\hat{\mathbf{x}} = \mathbf{x}^{n+1} - \mathbf{u}^n(\mathbf{x}^{n+1}) \tag{4.27}$$

$$\mathbf{x}_d^n = \mathbf{x}^{n+1} - 2\Delta t\, \mathbf{u}^n(\hat{\mathbf{x}}) \tag{4.28}$$

The (4.26) is solved by projection method. First, given the velocity $\mathbf{U}^n$, an intermediate velocity field is computed by ignoring the pressure:

$$\frac{1}{\Delta t}\left(\frac{3}{2}\mathbf{U}^{*} - 2\mathbf{U}_d^n + \frac{1}{2}\mathbf{U}_d^{n-1}\right) = \frac{1}{Re}\Delta\mathbf{U}^{*} + F^{n+1} \tag{4.29}$$

that could be written as

$$\left(\frac{3}{2}Id - \frac{1}{Re}\Delta\right)\mathbf{U}^{*} = 2\mathbf{U}_d^n - \frac{1}{2}\mathbf{U}_d^{n-1} + \Delta t\, F^{n+1} \tag{4.30}$$

To satisfy the incompressibility condition, a potential function $\tilde{P}$ is now defined by solving the following Poisson problem:

$$\Delta \tilde{P} = \nabla \cdot \mathbf{U}^* \tag{4.31}$$

Finally the fluid velocity $\mathbf{U}^{n+1}$ is projected to the divergence free field:

$$\mathbf{U}^{n+1} = \mathbf{U}^* - \nabla \tilde{P} \tag{4.32}$$

Describing previous method, the assumption of constant time step is done. When the time step is not constant we propose a simple alternative structure. Let's define $h_{n+1} = t^{n+1} - t^n$ and $h_n = t^n - t^{n-1}$. By performing a Taylor expansion:

$$\mathbf{U}_n = \mathbf{U}_{n+1} - h_{n+1}\mathbf{U}'_{n+1} + \frac{1}{2}h_{n+1}^2\mathbf{U}''_{n+1} + O(h^3) \tag{4.33}$$

$$\mathbf{U}_{n-1} = \mathbf{U}_{n+1} - (h_n + h_{n+1})\mathbf{U}'_{n+1} + \frac{1}{2}(h_n + h_{n+1})^2\mathbf{U}''_{n+1} + O(h^3) \tag{4.34}$$

Multiplying the (4.33) by $(h_n + h_{n+1})^2$ and the (4.34) by $-h_{n+1}^2$, let's sum new expression each other:

$$\begin{gathered}(h_n + h_{n+1})^2 (\mathbf{U}_n - \mathbf{U}_{n-1}) - h_{n+1}^2 (\mathbf{U}_{n-1} - \mathbf{U}_{n+1}) = \\ -(h_n + h_{n+1})^2 h_{n+1}\mathbf{U}'_{n+1} + (h_n + h_{n+1})h_{n+1}^2\mathbf{U}'_{n+1}\end{gathered}$$

Then, the derivative $\mathbf{U}'_{n+1}$ is given by:

$$\mathbf{U}'_{n+1} = \frac{(h_n^2 + 2\,h_n h_{n+1})\mathbf{U}_{n+1} - (h_n + h_{n+1})^2\mathbf{U}_n + h_{n+1}^2\mathbf{U}_{n-1}}{h_n\,h_{n+1}\,(h_n + h_{n+1})}$$

Therefore the left hand side of (4.26) becomes:

$$\frac{(h_n^2 + 2\,h_n h_{n+1})\mathbf{U}^{n+1} - (h_n + h_{n+1})^2 \mathbf{U}_d^n + h_{n+1}^2 \mathbf{U}_d^{n-1}}{h_n\, h_{n+1}\,(h_n + h_{n+1})}$$

Boundary condition for velocity and pressure

Following the approach of [5, 25, 26], proper boundary condition on $\partial\Omega$ for the intermediate velocity and pressure on rigid walls are:

$$\begin{aligned} \mathbf{n}\cdot\mathbf{U}^* &= \mathbf{n}\cdot\mathbf{U}^{n+1} \\ \mathbf{n}\cdot\nabla P &= 0 \\ \tau\cdot\mathbf{U}^* &= \tau\cdot\mathbf{U}^{n+1} + \tau\cdot\nabla P \end{aligned}$$

On free surface, we always extrapolate the velocity from inside to outside region and impose $p = 0$.

4.3 Iterative method

In our computation, we have seen that previous method is second order in time and space, but it doesn't preserve stationary solution. We have introduce an original technique based on an iterative solution of the Navier-Stokes equations that preserves stationary solution at discrete level. Starting from the Semi-Lagrangian form of the Navier-Stokes equation, taking the divergence of both side of

$$\frac{1}{\Delta t}\left(\frac{3}{2}\mathbf{U}^{n+1} - 2\mathbf{U}_d^n + \frac{1}{2}\mathbf{U}_d^{n-1}\right) = -\nabla p^{n+1} + \frac{1}{Re}\Delta\mathbf{U}^{n+1} + F^{n+1} \qquad (4.35)$$

and imposing the discrete incompressibility condition $\nabla \cdot \mathbf{U}^{n+1} = 0$, we obtain:

$$-\Delta p = F_p, \text{ where } F_p = \nabla \cdot \left(-\frac{2}{\Delta t}\mathbf{U}^n + \frac{1}{2\Delta t}\mathbf{U}^{n-1} - F^{n+1} \right)$$

Boundary condition for velocity and pressure

On rigid walls, we choose free-slip boundary conditions. To compute the right boundary condition for pressure, multiply by n the equation:

$$\frac{d\,\mathbf{U}}{d\,t} = -\nabla p + \frac{1}{Re}\Delta \mathbf{U} + \mathbf{F} \tag{4.36}$$

and obtain:

$$\frac{\partial p^{(n+1)}}{\partial n} = \mu\,\Delta \mathbf{U}^{(n+1)} \cdot n + \mathbf{F} \cdot n \tag{4.37}$$

in which $\frac{d\,\mathbf{U}}{d\,t} \cdot n = 0$ because of free-slip boundary condition.

On free surface, extrapolated value of velocity are used, while the boundary conditions for pressure are:

$$\begin{aligned} \nabla \cdot \mathbf{U}^{n+1} &= 0 \\ \frac{\partial \mathbf{U}^{n+1} \cdot \tau}{\partial n} &= 0 \\ p &= 0 \end{aligned}$$

Chapter 5

Poisson Solver

Solution of Navier-Stokes equations allows a Poisson problem for pressure as an intermediate step. In this section we briefly describe the technique used to solve this kind of problem (for more details see [11, 12]).

5.1 Second order discretization

Let $d \geq 1$ an integer, $D = [-1,1]^d$ the computational domain, $\Omega \subset D$ a domain such that $\partial\Omega \cap \partial D = \emptyset$. We assume Ω is a smooth domain, i.e. the boundary $\partial\Omega \in C^1$. Suppose that the boundary can be partitioned in Γ_D, Γ_N such that, for example, $\Gamma_D \bigcup \Gamma_N = \partial\Omega$, and $\overset{\circ}{\Gamma}_D \cap \overset{\circ}{\Gamma}_N = \emptyset$.

The model problem to be solved is:

$$\begin{aligned} -\Delta u &= f \quad \text{in } \Omega && (5.1)\\ u &= g_D \text{ on } \Gamma_D && (5.2)\\ \frac{\partial u}{\partial n} &= g_N \text{ on } \Gamma_N && (5.3) \end{aligned}$$

where n is the outward unit normal, $f : \Omega \to \mathbb{R}$, $g_D : \Gamma_D \to \mathbb{R}$, $g_N : \Gamma_N \to \mathbb{R}$ are assigned functions. The domain Ω is represented by a level set function ϕ.

5.2 Discretization of the problem

Once the computational domain is discretized in regular cells, denote with $N_i = |\Omega_h|$ the number of grid nodes inside Ω and with $N_g = |\Gamma_h|$ the number of ghost nodes, where inside and ghost nodes are defined as in Section 3.5.

The goal is to write a $(N_i + N_g) \times (N_i + N_g)$ linear system. Discretizing the Laplace operator by the standard five points stencil, the N_i equations for the inside points can easily write as, (Fig. 5.1):

$$\frac{4u_{i,j} - (u_{i,j-1} + u_{i,j+1} + u_{i-1,j} + u_{i+1,j})}{h^2} = f_{i,j}. \tag{5.4}$$

To close the linear system, we must write an equation for each ghost point G. In order to close the linear system of equations (5.4) for inside grid points, an equation for each ghost point has to be written. Let G be a ghost point. The outward unit normal in G, using the level set function ϕ, $\hat{\mathbf{n}}_G = (n_G^x, n_G^y) = \nabla\phi / |\nabla\phi|$, using a second order accurate discretization for $\nabla\phi$, such as central difference in G. By the signed distance function, it is possible to compute the closest boundary point to G, called B (see Fig. 5.2):

$$B = G - \hat{\mathbf{n}}_G \cdot \phi(G). \tag{5.5}$$

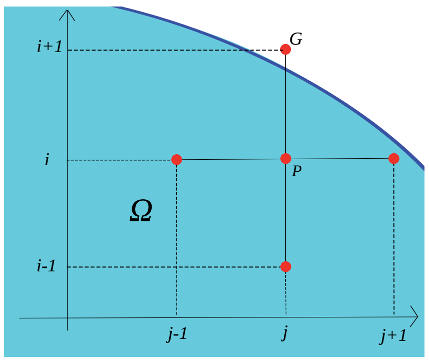


Figure 5.1: Five-point stencil centered at P. The grid point G is outside the domain and it is named *ghost point*.

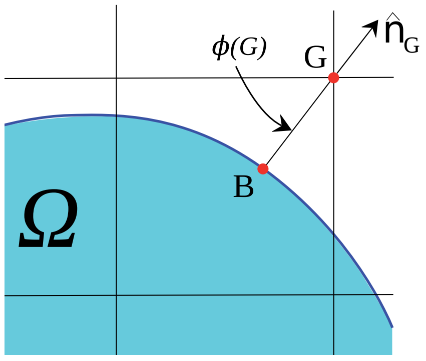


Figure 5.2: Computation of the boundary closest point B to the ghost point G from the normal unit vector, obtained from the signed distance function.

The point B could lie on Γ_D or Γ_N. If $B \in \Gamma_D$, the boundary conditions are:

$$u_h(B) = g_D(B) \tag{5.6}$$

while, if $B \in \Gamma_N$, we have:

$$\frac{\partial u_h}{\partial n}(B) = g_N(B)$$

where $u_h(B)$ and $\dfrac{\partial u_h}{\partial n}(B)$ are suitable reconstructions of the exact solution and its normal derivative by the numerical solution u_h. We choose

$$u_h(B) = \tilde{u}(B), \quad \frac{\partial u_h}{\partial n}(B) = (\nabla \tilde{u} \cdot \hat{\mathbf{n}})|_B = \left(\nabla \tilde{u} \cdot \nabla \tilde{\phi}/|\nabla \tilde{\phi}|\right)\Big|_B \tag{5.7}$$

where $\tilde{u}$ and $\tilde{\phi}$ are biquadratic interpolant respectively of u and ϕ on the upwind (with respect to the normal) nine-point stencil defined in 4.2

5.3 Relaxation scheme: fictitious time

A simple Gauss-Seidel scheme applied to the linear system fails to converge. The main idea to use Gauss-Sidel-like smoother consists in keeping a Gauss-Seidel-type iteration for inner equations and *relaxing* the boundary conditions. Therefore, the model problem 5.1 (5.1)-(5.3) is transformed into the following *associate time-dependent problem*:

$$\begin{aligned}
\frac{\partial \tilde{u}}{\partial t} &= \Delta \tilde{u} + f && \text{in } \Omega && (5.8)\\
\frac{\partial \tilde{u}}{\partial t} &= \mu_D (g_D - \tilde{u}) && \text{on } \Gamma_D && (5.9)\\
\frac{\partial \tilde{u}}{\partial t} &= \mu_N \left(g_N - \frac{\partial \tilde{u}}{\partial n} \right) && \text{on } \Gamma_N && (5.10)\\
\tilde{u} &= \tilde{u}_0 && \text{in } \Omega, \text{ when } t = 0 && (5.11)
\end{aligned}$$

where μ_D and μ_N are two positive constants. If the solution of the problem (5.8)-(5.11) is stationary (i.e. $\partial \tilde{u}/\partial t \to 0$, as $t \to +\infty$), then $u(x) = \lim_{t\to+\infty} \tilde{u}(t,x)$ is a solution of the model problem (5.1)-(5.3). A relaxation scheme can be therefore obtained discretizing the problem (5.8)-(5.11), where the time t is a fictitious time.

5.3.1 Discretization of relaxation scheme

For any grid point of Ω_h, set of grid nodes inside Ω, we have an equation deriving form discretization of (5.8) in such point. Using forward Euler in time and central difference in space and taking the maximum time step consented by the CFL condition (i.e. $\Delta t = h^2/4$):

$$u_{i,j}^{(n+1)} = 1/4\left(h^2 f_{i,j} + u_{i-1,j}^{(n)} + u_{i+1,j}^{(n)} + u_{i,j-1}^{(n)} + u_{i,j+1}^{(n)}\right). \tag{5.12}$$

Eq. (5.12) is equivalent to (5.1) using central difference in space and Jacobi iteration.

For any ghost point G we compute the projection point B on the boundary by previous formula (5.5) and discretize (5.9) or (5.10) if respectively $G \in \Gamma_D$ or $G \in \Gamma_N$. We use forward Euler in time and the discretizations (5.7) in space. We obtain the iterative scheme:

$$u_G^{(n+1)} = u_G^{(n)} + \mu_D \Delta t\left(g_D(B) - u_h^{(n)}(B)\right) \tag{5.13}$$

if $G \in \Gamma_D$, or

$$u_G^{(n+1)} = u_G^{(n)} + \mu_N \Delta t\left(g_D(B) - \frac{\partial u_h}{\partial n}^{(n)}(B)\right) \tag{5.14}$$

if $G \in \Gamma_N$.

The reconstructions $u_h^{(n)}(B)$ and $\dfrac{\partial u_h}{\partial n}^{(n)}(B)$ are the ones described in Sec. 5.1, more precisely they are (5.7) for **second order** accuracy (of the solution and in the gradient).

Constants μ_D and μ_N of (5.13) and (5.14) are obtained by CFL condition, i.e. in such a way that the coefficient of $u_P^{(n)}$ in the right-hand side of (5.13) and (5.14) is positive. Such conditions read:

$$\mu_D \Delta t < 1, \qquad \frac{\mu_N \Delta t}{h} < \frac{1}{\sqrt{2}} \text{ for first order accuracy;}$$

$$\mu_D \Delta t < 1, \qquad \frac{\mu_N \Delta t}{h} < \frac{2}{3\sqrt{2}} \text{ for second order accuracy.} \tag{5.15}$$

5.4 Multigrid components

In this section we describe a multigrid approach to speed up the convergence of the relaxation scheme (5.8)-(5.11). Let I_h be a general subset of D_h, set of grid nodes. Let introduce the linear space of grid functions over I_h and define $S(I_h) = \{\mathbf{w}_h \colon I_h \to \mathbb{R}\}$. From now on, we shall consider the two space dimension, i.e. $d = 2$, but the results are valid also for $d > 2$.

Using the simplified notation, the iterative scheme (5.12)-(5.14) converges to the solution of the problem:

$$\begin{cases} -\Delta_h \mathbf{u}_h = \mathbf{f}_h \\ L_h \mathbf{u}_h = \mathbf{g}_h \end{cases} \tag{5.16}$$

where:

- $\mathbf{u}_h \in S(\Omega_h \cup \Gamma_h)$ is the unknown;
- $\Delta_h \colon S(\Omega_h \cup \Gamma_h) \to S(\Omega_h)$ is the standard discrete version of the Laplacian operator:

  $$\Delta_h \mathbf{w}_h(x,y) = \frac{1}{h^2}\left(\mathbf{w}_h(x+h,y) + \mathbf{w}_h(x-h,y) - 4\mathbf{w}_h(x,y) \right. \\ \left. + \mathbf{w}_h(x,y+h) + \mathbf{w}_h(x,y-h)\right)$$

  for any $\mathbf{w}_h \in S(\Omega_h \cup \Gamma_h)$ and $(x,y) \in \Omega_h$;

- $\mathbf{f}_h \in S(\Omega_h)$ is defined by $f_h(P) = f(P)$ for any grid point $P \in \Omega_h$;

- $L_h \colon S(\Omega_h \cup \Gamma h) \to S(\Gamma_h)$ is the discrete version of boundary conditions:

$$L_h \mathbf{w}_h(G) = \begin{cases} \mathcal{L}_{St_G}[u](B) & \text{if} \quad B \in \Gamma_D \\ \left. \left(\nabla \mathcal{L}_{St_G}[u] \cdot \dfrac{\nabla \mathcal{L}_{St_G}[\phi]}{|\nabla \mathcal{L}_{St_G}[\phi]|} \right) \right|_B & \text{if} \quad B \in \Gamma_N \end{cases} \tag{5.17}$$

  for any $\mathbf{w}_h \in S(\Omega_h \cup \Gamma_h)$ and $G \in \Gamma_h$;

- $\mathbf{g}_h \in S(\Gamma_h)$ is defined by:

$$\mathbf{g}_h(G) = \begin{cases} g_D(B) & \text{if} \quad B \in \Gamma_D \\ g_N(B) & \text{if} \quad B \in \Gamma_N \end{cases}$$

  for any ghost point $G \in \Gamma_h$.

For any spatial step h, let introduce an exact solver:

$$\mathbf{u}_h = \mathcal{S}_h \left(\mathbf{f}_h, \mathbf{g}_h \right)$$

of the system (5.16), and denote by

$$\Re_h \colon S(\Omega_h \cup \Gamma_h) \times S(\Omega_h) \times S(\Gamma_h) \longrightarrow S(\Omega_h \cup \Gamma_h) \tag{5.18}$$

the relaxation operator. The iterative scheme

$$\mathbf{u}_h^{(m+1)} = \Re_h \left(\mathbf{u}_h^{(m)}, \mathbf{f}_h, \mathbf{g}_h \right) \tag{5.19}$$

converges to the solution of (5.16) as $n \to +\infty$. In details, the iteration

(5.19) summarize the iterative scheme (5.12)-(5.14).

In order to explain the multigrid approach, we just describe the two-grid correction scheme (TGCS), because all the others schemes, such as V-cycle, W-cycle or Full multigrid, can be easily derived from it (see [39, Sections 2.4, 2.6] for more details). The TGCS consists into the following algorithm:

1. Set initial guess $\mathbf{u}_h = 0$

2. Relax ν_1 times on the finest grid: for k from 1 to ν_1 do

$$\mathbf{u}_h := \Re_h \left(\mathbf{u}_h, \mathbf{f}_h, \mathbf{g}_h\right)$$

3. Compute the following defects:

$$\mathbf{r}_h^{\Omega} = f_h + \Delta_h \, \mathbf{u}_h$$

$$\mathbf{r}_h^{\Gamma} = g_h - L_h \mathbf{u}_h$$

4. Transfer the defects to a coarser grid with spatial step $2h$ by a suitable *restriction operator*

$$\mathbf{r}_{2h}^{\Omega} = I_{2h}^{h} \left(\mathbf{r}_h^{\Omega}\right)$$

$$\mathbf{r}_{2h}^{\Gamma} = I_{2h}^{h} \left(\mathbf{r}_h^{\Gamma}\right)$$

5. Solve exactly the residual problem in the coarser grid

$$\mathbf{e}_{2h} = \mathcal{S}_{2h} \left(\mathbf{r}_{2h}^{\Omega}, \mathbf{r}_{2h}^{\Gamma}\right)$$

6. Transfer the error to the finest grid by a suitable *interpolation operator*

$$\mathbf{e}_h = I_h^{2h}\left(\mathbf{e}_{2h}\right)$$

7. Correct the fine-grid approximation

$$\mathbf{u}_h : = \mathbf{u}_h + \mathbf{e}_h$$

8. Relax ν_2 times on the finest grid: for k from 1 to ν_2 do

$$\mathbf{u}_h : = \Re_h\left(\mathbf{u}_h, \mathbf{f}_h, \mathbf{g}_h\right)$$

In the next section, a description of step (4) an (6) is given.

5.4.1 Transfer grid operators

Restriction operator

In general, by the stencil notation

$$I_{2h}^{h} = \begin{bmatrix} & \vdots & \vdots & \vdots & \\ \cdots & t_{-1,-1} & t_{-1,0} & t_{-1,1} & \cdots \\ \cdots & t_{0,-1} & t_{0,0} & t_{0,1} & \cdots \\ \cdots & t_{1,-1} & t_{1,0} & t_{1,1} & \cdots \\ & \vdots & \vdots & \vdots & \end{bmatrix}_{2h}^{h}$$

we will intend the restriction operator I^h_{2h} defined by:

$$I^h_{2h}\mathbf{w}_h(x,y) = \sum_{(i,j)\in R_k} t_{i,j}\mathbf{w}_h(x+jh, y+ih),$$

where only a finite number of coefficients $t_{i,j}$ is different from zero, and $R_k \equiv \{-k,\ldots,k\}^2$ for some positive integer k. In practice $k=1$ allows second order restriction operator.

Let us suppose we have extended the defect to the whole computational domain D_h (as it is carefully described in Sec. 5.4.1). Since we have different operators for inner equations and for boundary conditions, the defect is smooth separately inside Ω_h and along the ghost point Γ_h (or $D_h - \Omega_h$, because of the extension), but it is not smooth in all $\Omega_h \cup \Gamma_h$. For this reason, it is convenient to transfer separately to the coarse grid the defects $\mathbf{r}^\Omega_h$ and $\mathbf{r}^\Gamma_h$. Let us define the new restriction operator:

$$\mathcal{I}^h_{2h}\colon S(Z_h) \longrightarrow S(Z_{2h}), \tag{5.20}$$

where Z_h is an arbitrary subset of D_h, and where we intend $Z_{2h} = Z_h \cap \Omega_{2h}$. Let $(x,y) \in Z_{2h}$. Let $\mathcal{N}(x,y) = \{(x+jh, y+ih)\colon j,i = -1,0,1\}$ a neighborhood of (x,y).

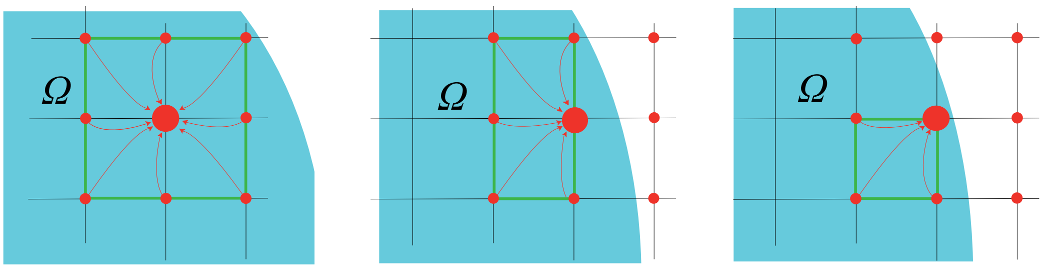
Now consider the maximum full rectangle $\mathcal{T}$ with vertices belonging to $\mathcal{N}(x,y)$ and such that $\mathcal{T}\cap D_h \subseteq Z_h$ (see Fig. 5.3, where $Z_h = \Omega_h$). Therefore, the stencil we use in (x,y) to transfer $\mathbf{w}_h$ to a coarse grid depends on the size of $\mathcal{T}$. Now let $\mathcal{T}\cap D_h$ be a 3×2 points. We can suppose the vertices of $\mathcal{T}$ are $(x+jh, y+ih)$, with $j\in\{-1,0\}$, $i\in\{-1,1\}$. In this case, the stencil we will use is:

$$\left(I_{2h}^{h}\mathbf{w}_h\right)(x,y) = \frac{1}{16}\begin{bmatrix} 2 & 2 & 0 \\ 4 & 4 & 0 \\ 2 & 2 & 0 \end{bmatrix}_{2h}^{h}(x,y), \tag{5.21}$$

while, if $\mathcal{T}$ is a 2×2 points, with vertex $(x+jh, y+ih)$, $j,i\in\{-1,0\}$, the stencil will be:

$$\left(I_{2h}^{h}\mathbf{w}_h\right)(x,y) = \frac{1}{16}\begin{bmatrix} 0 & 0 & 0 \\ 4 & 4 & 0 \\ 4 & 4 & 0 \end{bmatrix}_{2h}^{h}(x,y), \tag{5.22}$$

This three case are summarized in Fig. 5.3 (where $Z_h = \Omega_h$).



$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}_{2h}^{h} \qquad \frac{1}{16}\begin{bmatrix} 2 & 2 & 0 \\ 4 & 4 & 0 \\ 2 & 2 & 0 \end{bmatrix}_{2h}^{h} \qquad \frac{1}{16}\begin{bmatrix} 0 & 0 & 0 \\ 4 & 4 & 0 \\ 4 & 4 & 0 \end{bmatrix}_{2h}^{h}$$

Figure 5.3: Upper, the nine points of $\mathcal{N}(x,y)$ and the green boundary of the rectangle $\mathcal{T}$. The bold red point is on the coarser and finer grids, while the little red points are on the finer grid. The arrows represent the action of the restriction operators. Below, the respective stencil to be used.

Extension of the defect

Finally we talk about the extension of the defect $\mathbf{r}_h^\Gamma$ from $S(\Gamma_h)$ to $S(D_h - \Omega_h)$. In every ghost point we store the defect of the boundary condition concerning that ghost point but it is geometrically referred to a boundary point B placed along the normal direction. Switching to a coarse grid, some ghost point G_1 may not be ghost point in the fine grid (see Fig. 5.4).

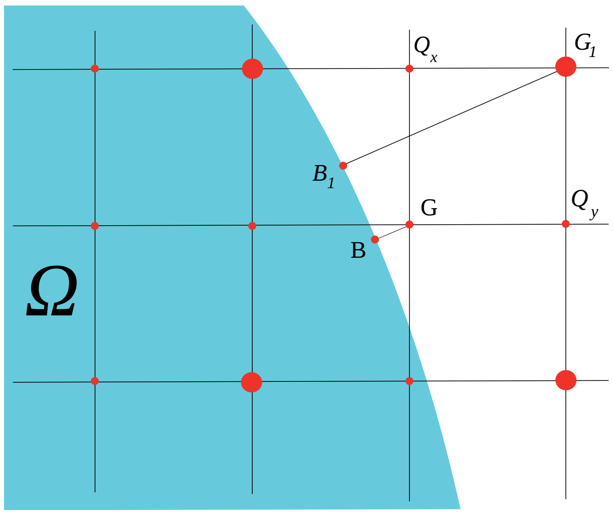


Figure 5.4: Red bold and small points are grid points of Ω_h, while red bold points are grid points of Ω_{2h}. G_1 is a ghost point on the coarser grid, but not on the finer grid, then no value of the defect is stored in it. Q_x and Q_y are the two upwind near points to G_1.

Then, no acceptable value of the defect is stored in G_1. Indeed, we expect that $\mathbf{r}_{2h}$ has in the ghost point G_1 the defect of the boundary conditions referred to B_1. Hence, if we extend the defect $\mathbf{r}_h$ outside Ω_h constant along the normal lines to the boundary, we will find $\mathbf{r}_h^\Gamma(G_1)$ as an approximation of the defect of the boundary conditions in B_1. After coarsening (performed using only points outside Ω_h, as described before), the ghost points of the coarser grid will contain the expected values of the defect.

The extension of the defect $\mathbf{r}_h^\Gamma$ is performed by solving the same transport equation solved for computing velocity boundary condition on free surface

$$\frac{\partial r^\Gamma}{\partial \tau} + \nabla r^\Gamma \cdot \mathbf{n} = 0$$

in a few steps of a fictitious time τ. The extension process can be resumed introducing an extension operator, which in practice depends only on the set of ghost point Γ_h and on the discretized signed distance function ϕ_h. Therefore:

$$\mathcal{E}[\Gamma_h;\phi_h]\colon S(\Gamma_h) \longrightarrow S(D_h - \Omega_h), \tag{5.23}$$

and then the extension procedure and the restriction of the defect $\mathbf{r}_h^\Gamma$ can be resumed as follows:

$$I_{2h}^h(\mathcal{E}[\Gamma_h;\phi_h](\mathbf{r}_h^\Gamma)).$$

Interpolation

Since the interpolation operator will act on the error, which is continuous across the boundary, we can use the standard stencil for linear interpolation operator:

$$I_h^{2h} = \frac{1}{4}\left]\begin{matrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{matrix}\right[_h^{2h}. \tag{5.24}$$

Chapter 6

Numerical Test

6.1 Workflow of the general method

Let Ω a free surface domain in which we would describe the motion of a certain fluid. Let's assign:

- a level set function ϕ_0 such that its zero level set Γ is equivalent to $\partial\Omega$
- an initial condition for velocity field U_0 such that $\nabla \cdot U_0 = 0$

Until the final time or steady state is reached:

- reinitialize the level set function to the signed distance function ϕ;
- scan grid nodes to identify inside, ghost and outside points;
- extrapolate the velocity out of the interface in normal direction, to "cover" ghost points;
- solve the Euler or Navier-Stokes equation in Ω^-

- extrapolate the new velocity out of the interface in normal direction, to "cover" ghost points;
- update the interface by solving the level set equation for ϕ

6.2 Euler equation - Mac method

This is a very simple test, in which we would like to solve Euler equation on a parabolic-shaped basin. In Figure 6.1 snapshot are taken at time t=0, 0.15, 0.60, 0.83 from the top left to right.

6.3 Comparison with SPH method

Smoothed Particle Hydrodynamics (SPH) is a Langrangian meshless method. The fluid is divided into a set of particles, each carrying information about its position, velocity, pressure and any other quantity needed by the model.

Since the method is Lagrangian, the particles move according to the governing equation of the fluid, without being constrained to a fixed mesh. This allows them to move freely with respect to one another, without the issues related to mesh deformations which are often found in methods such as finite elements. As in fixed-mesh method, it necessary to compute derivatives using information from a finite number of points: in the SPH method, the interpolation points are particles which move with the flow, and the interpolation of any quantity depends on kernel estimation (kernels are functions which tend to the delta function as the length scale h tend to zero), [21, 27].

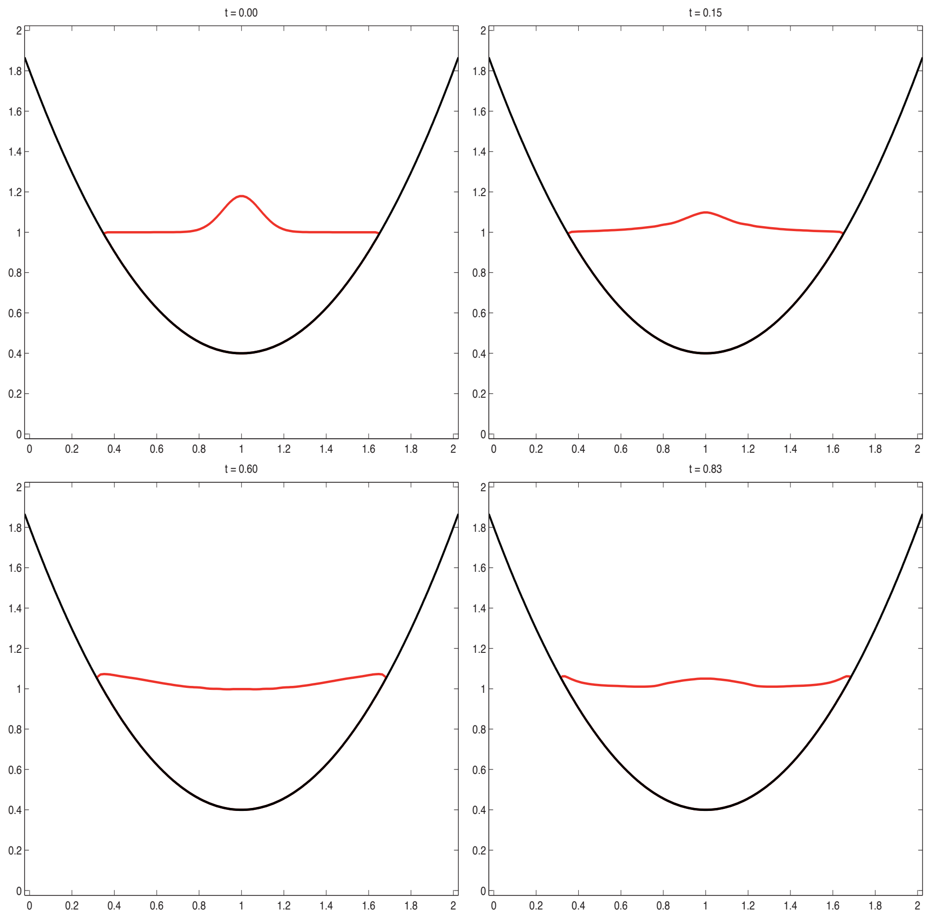


Figure 6.1: Fluid flow on parabolic ground, t = 0, 0.15, 0.60, 0.83, test (6.2)

We compare first-order in time Sem-Lagrangian method with SPH method implemented by Dr. G.Bilotta. Initial level set function is a gaussian defined in $[0, 2] \times [0, 0.5]$

$$\phi_0 = y - (0.1\ \exp(-(x-1)^2/(0.1)^2) + 0.1);$$

Comparison are based on two different grid size. In Figure 6.2, magenta clored lines correspond to 512×64 grid size, black colored lines correspond to 256×64 grid size and blu dots are fluid particles.

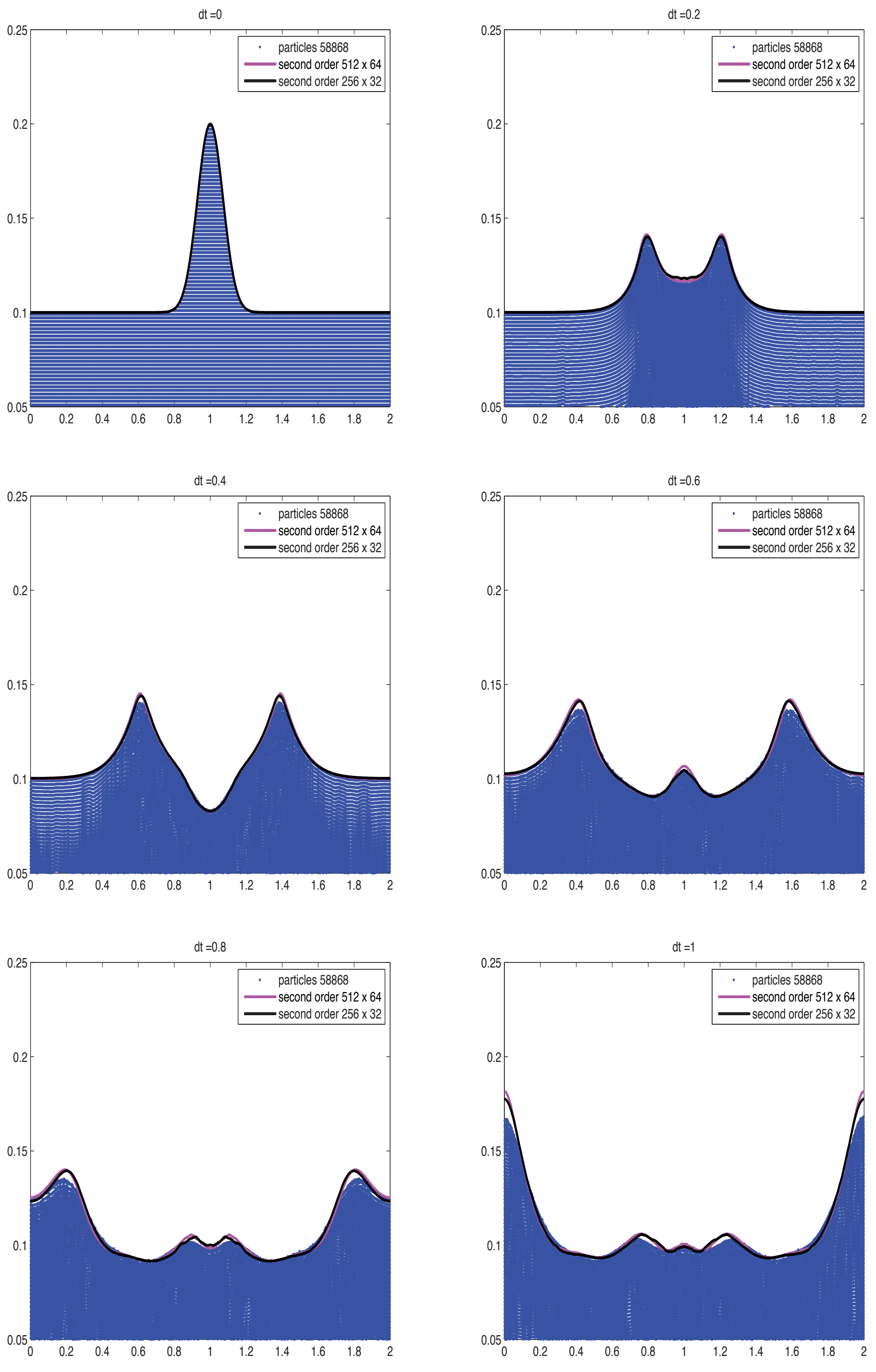


Figure 6.2: SPH vs 1st order in time semi-lagrangian

The results are very encouraging: using two different method, with two different approaches, we obtain same results except in the convex areas near the free surface of the fluid. In the concave areas there is a good agreement between two methods. This may results from the fact that SPH is a first order approximation and accuracy also depends on near particle, in fact it increases in high density regions and decreases in low density regions, Figure 6.3.

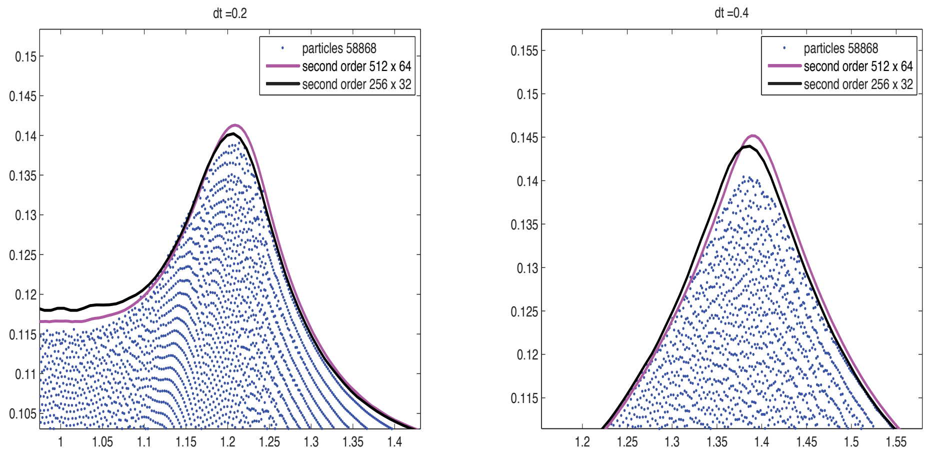


Figure 6.3: Zoom on convex areas of the free surface

6.4 Another comparison

In Figure 6.4 and Figure 6.5 flow evolution is plotted. Computational domain is $[-1, 1] \times [0, 0.25]$. The gaussian $\phi_0 = y - (a \exp(-(x-b)^2/c^2) + 0.1)$ with $a = 0.1$, $b = 0.$ and $c = .1$ is the starting level set. First order in time Semi-Lagrangian methods for Euler equations are used. The blu, red and black lines correspond respectively to 128×32, 256×64 and 512×128 grid size.

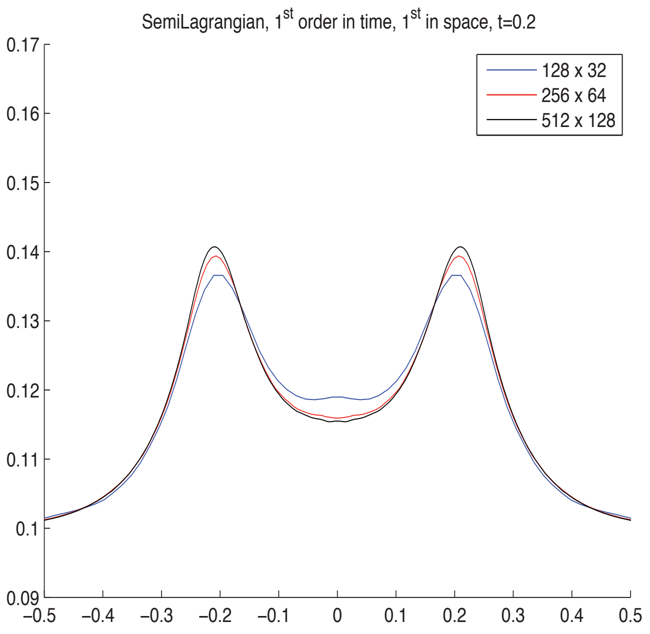


Figure 6.4: Fluid flow evolution using Semi-Lagrangian $O(\Delta t, \Delta x)$, test (6.4)

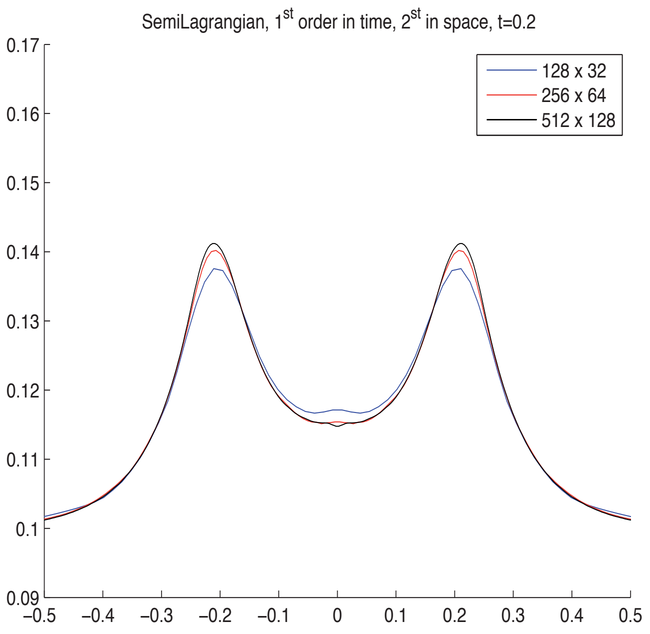


Figure 6.5: Fluid flow evolution using Semi-Lagrangian $O(\Delta t, \Delta x^2)$, test (6.4)

6.5 Verify accuracy of new iterative method

In order to test the accuracy of the method, consider a single vortex flow in $[-0.5, 0.5]^2$. The exact solution is

$$
\begin{aligned}
u(x,y,t) &= -\cos(\pi\, x)\sin(\pi\, y)\cos(\pi\, t)\\
v(x,y,t) &= \sin(\pi\, x)\cos(\pi\, y)\cos(\pi\, t)\\
p(x,y,t) &= -1/4\, \cos(\pi\, t)(\cos(2\pi\, x)+\cos(2\pi\, y))
\end{aligned}
$$

In Figure 6.6 are plotted error computed comparing the exact solution with numerical solution. As we can see, a CFL=1 is a good choice in order to obtain good results.

CFL	Bestfitting Slope
0.125	1.68
0.5	1.88
1	1.92

Table 6.1: Bestfitting of error on exact solution at different CFL, test (6.5)

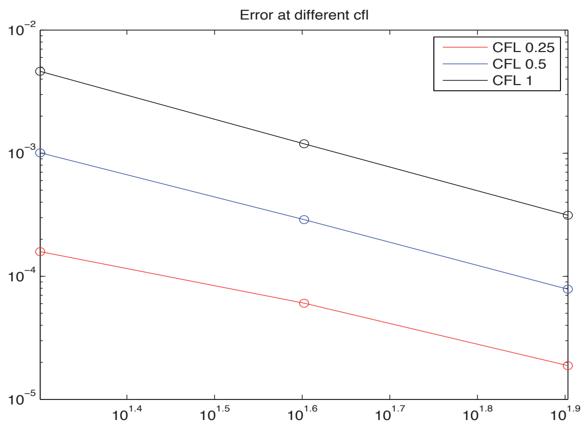


Figure 6.6: Error at different CFL, , test (6.5)

6.6 Stationarity of new iterative method

The iterative method seen for the Navier-Stokes equation, preserves the stationary solution. Starting from a solution u_e for the stationary Euler equations, we have seen that the same function satisfy the Navier-Stokes equation, in which the external forces are set equal to $-\Delta u_e$.
The function $U_e = (u_e, v_e)$

$$\begin{aligned} u_e &= -\cos(\pi\, x)\sin(\pi\, y) \\ v_e &= \cos(\pi\, y)\sin(\pi\, x) \end{aligned}$$

is a stationary solution for the Euler equation:

$$\frac{d\,U}{d\,t} + \nabla\, p = 0$$

If we consider the Navier-Stokes equation:

$$\frac{d\,U}{d\,t} + \nabla\, p = \Delta U + F$$

we can see that U_e satisfy the left hand side. Then, U_e is a stationary solution for the Navier-Stokes equation if $F = -\Delta U_e$. In Figure 6.7 is plotted error estimate. The stationary solution is reached at time t=0.6

	CFL	Bestfitting Slope
u	0.125	1.81
v	0.125	1.91

Table 6.2: Bestfitting slop of error computed on velocity component with respect to exact solution at CFL=0.125, test (6.6)

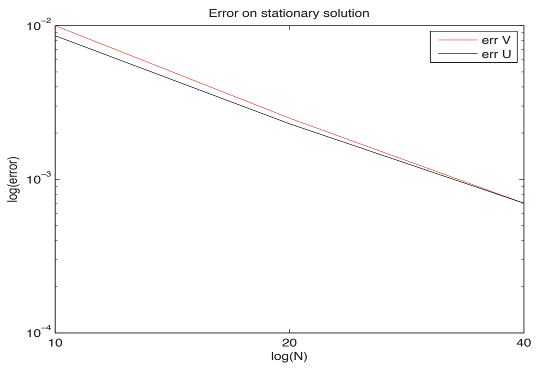


Figure 6.7: Error on stationary solution, test (6.6)

6.7 Comparison between new iterative method and shallow water solver

In this section we compare the results obtained by new iterative scheme with a central scheme for conservations laws, [33]. As we ca see in Figure 6.8,

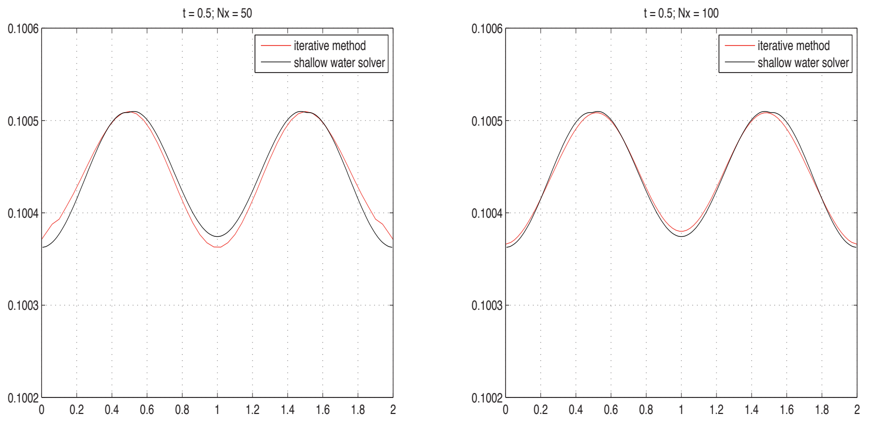


Figure 6.8: Compare Shallow water solver with Semi-Lagrangian scheme, test(6.7)

there is a good agreement in results. The difference between two interfaces, (in red interface from the iteraritve scheme and in black the water depth from shallow water scheme) is very small, respect to the spatial step.

6.8 Sitting-Bubble

Figure 6.9 and Figure 6.10 show the results of evolution of a bubble of fluid "sits" on a same properties fluid film in $[0,2]^2$, . Tests are run at different Reynold's number and snapshot are taken at time t=0, 0.18, 0.27, 0.342 from the top left to the bottom right. For $Re = 10$ deposition regime

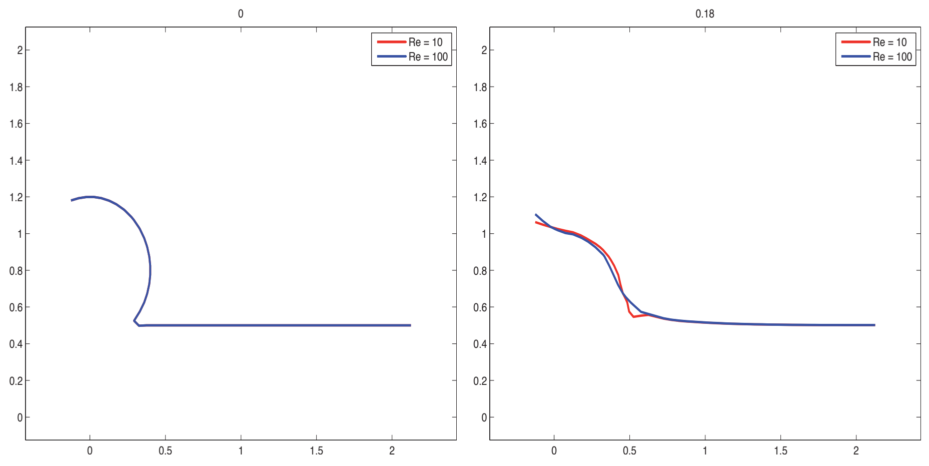


Figure 6.9: Sitting Bubble at different Reynold's number, Re=10 in red, Re=100 in blue, t=0, 0.18, test (6.8)

is observed (red line): the bubble sits on the layer without waves formation. For $Re = 100$ spreading regime is observed (blue line): while sitting, the bubble causes a waves that propagates on layer surface towards right side of the domain.

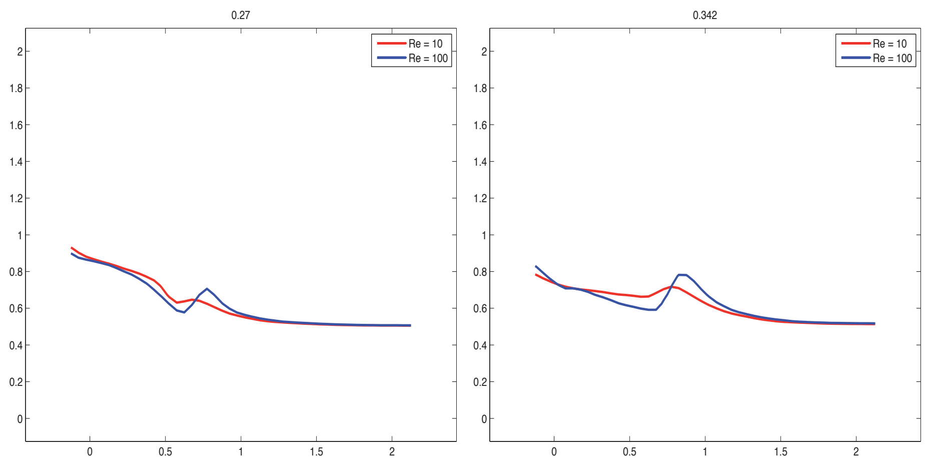


Figure 6.10: Sitting Bubble at different Reynold's number, Re=10 in red, Re=100 in blue, t=0.27, 0.342, test (6.8)

6.9 Rising Bubble

In this benchmark problem proposed for example in [36], a rising bubble is simulated in $[0,1] \times [0,1]$ from t=0 to t=0.25. We solve the Navier Stokes equation with no-slip boundary condition on the wall and Reynolds number 100. The initial configuration is a circle with center in $x_c = 0.5, y_c = 0.5$ and radius $R_c = 0.25$. The initial level set is $\phi = -\left(\sqrt{(x-x_c)^2 + (y-y_c)^2} - R_c\right)$. We suppose that the fluid is only in the region where $\phi < 0$

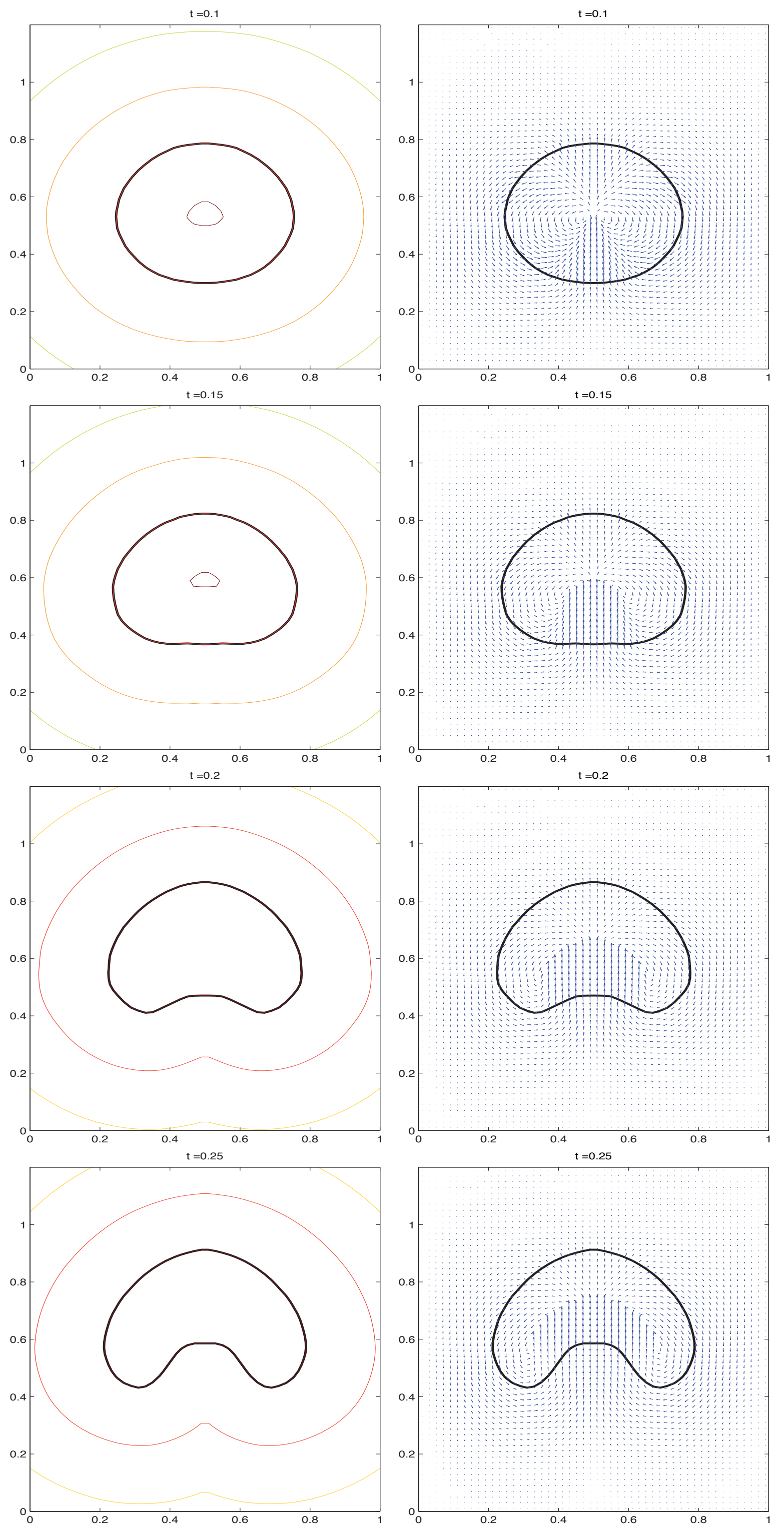


Figure 6.11: Evolution of Rising-Bubble (left column) and plot of the velocity filed (right column), t=0.1, 0.15, 0.2, 0.25

Chapter 7

Conclusion

We have presented the general computational approach that is well suited for numerical simulation of all kinds of unsteady fluid flows. The approach essentially relies on three basic components: level set methods for boundary representation finite difference scheme for spatial discretization and Eulerian or Semi-Lagrangian for temporal discretization. Two numerical methods have been applied to fluid flow simulation: 1) Chorin's projection or MAC method and 2) Semi-lagrangian method.

The first two methods are incredibly flexible and relatively efficient and, above all, easily understood. They may be applied to the computation of flows in fixed domains and to the simulation of free boundary value problems. Unfortunally they are explicit and this may cause a very small step size in computation. Moreover they are first order accurate in time and second order accurate in space.

Semi-lagrangian methods are second order accurate in time, therefore more accurate then the previous one. Starting from semi-lagrangian backward differentiation formula, we have developed an original iterative method, which is second order accurate in time and space and which preserves stationary solution at discrete level. The general system of Navier-Stokes equations with proper boundary condition, is split in two subproblems (one for the velocity and one for pressure) which are solved iteratively.

Future work

Next step is to include a solid body in the fluid and simulate the flow past this object. Our idea is to treat solid body as a fluid with a huge viscosity and so treat the problem as usual. The different things consist in boundary condition to be applied on the interface Γ between the "two fluids": the quantity as viscosity, density, pressure and velocity may have a jump on Γ. All these ingredients lay the foundations for lava flow simulation and crust formation. First of all is necessary to couple Navier-Stokes equations with heat equation because of viscosity dependence on temperature. Different rheologies may be considered: Newton, Herschel-Bulkley and Bingham. Actually, real lava rheologies is not known but most common rheology assumed for it is Bingham rheology. Finally we would like to implement our code in three-spatial dimension and validate it using multiphysics modeling and simulation software, as Comsol.

Acknowledments

First of all I would like to express my gratitude to my advisor Professor Giovanni Russo for the continuous support of my Ph.D study and research, for its motivation and knowledge. I would like to thank him for allowing me to come in contact with such interesting person met and topic treated during my experience abroad. I would like to thank Professor Giuseppe Mulone, who, although he was not my professor anymore, he always found the time to listen and to give support. I would like to express my sincere gratitude to Professor Mark Sussman for hosting me. I am very thankful to Dr. Giuseppe Bilotta always kind and always disposed to help me. I am grateful to every Professor met during my PhD (it's impossible to list everyone), because I have learnt something from each of them.

Ed ora, dopo i ringraziamenti "formali", passiamo ai ringraziamenti "informali". Inutile dire che ringrazio immensamente i miei genitori, per aver sempre sostenuto le mie scelta, per avermi spinto e invogliata quando era il momento giusto. Ringrazio mia sorella, per avermi fatto sentire sempre la sua presenza. Ringrazio Enzo, per essere così comprensivo, presente e il mio costante punto di riferimento.

Bibliography

[1] **T. D. Aslam**, *A partial differential equation approach to multidimensional extrapolation*, J Comp. Phys 193 (2003) 349Ð355.

[2] **G. R. Baker, D. I. Meiron, S.A.Orszag**, *Boundary Integral Methods for Axisymmetric and Three-Dimensional Rayleigh-Tayrlo Instability*, Physica D 12:19, 31, 1984.

[3] **R. Caiden, R. Fedkiw, C. Anderson**, *A Numerical Methods for Two Phase Flow. Consisting of Separate Compressible and. Incompressible Regions*, J. Comp. Phys. 166:1-27, 2001

[4] **J. B. Bell, P. Colella, H. M. Glaz**, *A second order Projection method for the incompressible Navier-Stokes equations*, J. Comp. Phys. 85, 257-283, 1989

[5] **D. L. Brown, R. Cortez, M. L. Minion**, *Accurate projection methods for the incompressible Navier-Stokes equations*, J. Comp. Phys. 168 (2001) 464-499.

[6] **Y. C. Chang, T. Y. Hou, B. Merriman, S. Osher***A Level Set Formulation of Eulerian Interface Capturing Methods for Incompressible Fluid Flows* J. Comput. Phys. 124, 449Ð464, 1996.

[7] **A. J. Chorin, J. E. Marsden**, *A Mathematical Introduction to Fluid Mechanics*, 3rd Edition, Springer, New York, 1998.

[8] **A. J. Chorin, J. E. Marsden**, *A Numerical Method for Solving Incompressible Viscous Flow Problems*, J. Comput. Phys. 135, 118-195.

[9] **A. J. Chorin**, *Numerical solutions of the Navier-Stokes equations*, Math. Comput. 22 (1968) 745-762.

[10] **A. J. Chorin**, *On the convergence of discrete approximations to the Navier-Stokes equations*, Math. Comput. 23 (1969) 341-353.

[11] **A. Coco, G. Russo**, *A fictitious time method for the solution of Poisson's equation in an arbitrary domain embedded in a square grid*, Journal of Computation Physics. Under revision.

[12] **A. Coco, G. Russo**, *Multigrid approach for Poisson' s equation with mixed boundary condition in an arbitrary domain*, Submitted. Preprint available in http://arxiv.org/pdf/1111.0983.

[13] **D. Enright, D. Nguyen, F. Gibou, R. Fedkiw**, *Using the Particle Level Set Method and a Second Order Accurate Pressure Boundary Condition for Free Surface Flows* Accepted to 4th ASME-JSME Joint Fluids Conference paper number: FEDSM2003-45144

[14] **R.P. Fedkiw, T. Aslam, B. Merriman, S. Osher**, *A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method)*, J. Comput. Phys. 152: 457Ð492, 1999.

[15] **R.P. Fedkiw, T. Aslam, T. Xu**, *The ghost fluid method for deflagration and detonation discontinuities*, J. Comput. Phys. 154:457Ð492, 1999

[16] **J. Glimm, O.McBryan**, *A Computational Model fo Interface* Adv. Appl. Math., 6:422 435, 1985

[17] **J. W. Grove**, *The interaction of Shocks with Fluid Interface* Adv. Appl. Math., 10:201 227, 1989

[18] **X-D Liu, R. Fedkiw, M. Kang**, *A boundary condition capturing method for Poisson's equation on irregular domain*, J. Comput. Phys, 154:151, 2000

[19] **F. H. Harlow, J. E. Welch**, *Numerical Calculation of Time Dependent Viscous Incompressible Flow of Fluid with Free Surface* Phys. Fluids 8, 2182 (1965)

[20] **H. Henshaw**, *A fourth-order accurate method for the incompressible Navier-Stokes equations on overlapping grids*, J. Comput. Phys, 113: 13-25, 1994

[21] **A. Herault, G. Bilotta, R.A. Dalrymple**, *SPH on GPU with CUDA* Journal of Hydraulic Research Vol. 48 Extra Issue (2010), pp. 74Ð79

[22] **J. van Kan***A Second-Order Accurate Pressure-Correction Scheme for Viscous Incompressible Flow*, J. Sci. and Stat. Comput. 7, pp. 870-891

[23] **M. Kang, R. Fedkiw, X-D Liu** *A boundary condition capturing method for multiphase incompressible flow*, J. Sci. Comput. 15: 323-60, 2000

[24] **F. Losasso, R. Fedkiw, S. Osher**, *Spatially adaptive techniques for level set methods and incompressible flow*, Computers and Fluids, 35:995Ð1010, 2006.

[25] **C. Min, F. Gibou**, *A Second Order Accurate Projection Method for the Incompressible Navier-Stokes equations on Non-Graded Adaptive Grids*, J. Comput. Phys. 219 912-929, 2006

[26] **P. Moin, J. Kim,**, *Application of a fractional-Step Method to Incompressible Navier-Stokes equations*, J. Comput. Phys. 59 308-323, 1985.

[27] **J.J. Monaghan***Smoothed particle hydrodynamics*, Rep. Prog. Phys. 68 (2005) 1703Ð1759.

[28] **H. N. Oguz, A. Prospetti** *Dynamic of Bubble Growth and Detachment from a Needle* J. Fluid. Mech. 257:111 145, 1993.

[29] **S. Osher, R. Fedkiw**, *Level Set Methods and Dynamic Implicit Surface*, Springer.

[30] **S. Osher, J. A. Sethian**, *Fronts Propagating with Curvature-Dependent Speed: Algorithms based on Hamilton-Jacobi Formulation*, J. Comput. Phys. 79:12, 49, 1988

[31] **J. Papac, F. Gibou, C. Ratsch**, *Efficient symmetric discretization for the Poisson, heat and Stefan-type problems with Robin boundary conditions* J. Computl Phys 229: 875Ð889, 2010.

[32] **R. Peyret, T.D. Taylor**, *Computational Methods for Fluid Flow*, Edition, Springer, New York.

[33] **G. Russo**, *Central Scheme for conservation laws with application to shallow water eqaution*

[34] **G. Russo, P. Smereka**, *A remark on Computing Distance Functions*, J. Comput. Phys. 163 no. 1, 51-67, 2000.

[35] **J.A. Sethian**, *Turbulent Combustion in Open and Closed Vessel* J. Comput. Phys. 54:425, 456, 1984

[36] **J.A. Sethian**, *An Improved Sharp Interface Method for Viscoelastic and Viscous Two-Phase Flows* J. Sci. Comput DOI 10.1007/s10915-007-9173-5

[37] **M. Sussman, E.G. Puckett**, *A coupled level set and volume of fluid method for computing 3d and axisymmetric incompressible twophase flow*, J. Comput. Phys., 162:301Ð337, 2000.

[38] **M. Sussman, P. Smereka, S. Osher**, *A level set method for computing solutions to incompressible two phase flow*, J. Comput. Phys. 119 146, 1994.

[39] **U.Trottemberg, C. Oosterlee, and A. Schuller**, *Multigrid. Academic* Press, 2000.

[40] **S.O.Unverdi, G. Tryggvason**, *A front-tracking method for viscous, incompressible, multifluid flows* J. Comput. Phys. 100: 25:37, 1992

[41] **D. Xiu, G. Karniadakis**, *A Semi-Lagrangian High-Order Method for Navier-Stokes Equations*, J. Comput. Phys. 172 658-684, 2001.