Fixed mesh methods in computational mechanics

Ramon Codina and Joan Baiges

May 12, 2010

Abstract

In many coupled problems of practical interest the domain of at least one of the problems evolves in time. The Arbitrary Eulerian Lagrangian (ALE) approach is a tool very often employed to cope with this domain motion. However, in this work we aim at describing numerical techniques that allow us to use a fixed mesh for the approximation of moving boundary problems, particularly using the finite element approach. This type of formulations is often termed embedded or immersed boundary methods. Emphasis will be put in describing a particular version of the ALE formulation using fixed meshes that we have developed, and that we call fixed-mesh ALE method (FM-ALE). Methods able to deal with fixed meshes are closely linked to the approximate imposition of boundary conditions. Some possibilities to do this approximation are also described.

Keywords: moving domains, fixed mesh methods, ALE, approximate boundary conditions

1 Introduction

In the classical ALE approach to solve problems in computational fluid dynamics, the mesh in which the computational domain is discretized is deformed (see for example [16, 33, 35]). This is done according to a prescribed motion of part of its boundary, which is transmitted to the interior nodes in a way as smooth as possible so as to avoid mesh distortion. The FM-ALE formulation has a different motivation. Instead of assuming that the computational domain is defined by the mesh boundary, we assume that there is a function that defines the boundary of the domain where the flow takes place. We will refer to it as the boundary function. It may be given, for example, by the shape of a body that moves within the fluid, or it may need to be computed, as in the case of level set functions. It may be also defined discretely, by a set of points. When this boundary function moves, the flow domain changes, and that must be taken into account at the moment of writing the conservation equations that govern the flow,

which need to be cast in the ALE format. However, our purpose here is to explain how to use always a background fixed mesh. A review of the method will be presented.

Other possibilities to use a single grid in the whole simulation can be found in the literature, each one having advantages and drawbacks. They were designed as an alternative to body fitted meshes and can be divided into two main groups, corresponding in fact to two ways of prescribing the boundary conditions on the moving boundary [13]:

- Force term. The interaction of the fluid and the solid is taken into account through a force term, which appears either in the strong or in the weak form of the flow equations. Among this type of methods, let us cite for example the Immersed Boundary method as a variant of the Penalty method, where punctual forces are added to the momentum equation, and the Fictitious Domain method, where the solid boundary conditions are imposed through a Lagrange multiplier.

- Approximate boundary conditions. Instead of adding a force term, these methods impose the boundary conditions in an approximate way once the discretization has been carried out, either by modifying the differential operators near the interface (in finite differences) or by modifying the unknowns near the interface.

The Immersed Boundary Method in its original form [48] consists in adding punctual penalty forces in the domain boundary so that the boundary conditions are fulfilled. The forces are computed from a fluid-structure (elastic) interaction problem at the interface. The method is first order accurate even if second order approximation schemes are used, although formal second order accuracy has been reported in [40]. The more recent Immersed Interface Method achieves higher order accuracy by avoiding the use of the Dirac delta distribution to define the forcing terms (see [41, 42, 56]).

The Penalty method is similar to the previous one in the sense that a force term is added to the momentum equations. The difference is due to the fact that the penalty parameter is not computed from a fluid-structure interaction as in the original immersed boundary method, but it is simply required to be large enough to enforce the boundary conditions approximately [53].

Another approach is the use of Lagrange multipliers to enforce the boundary conditions. However, the finite element subspaces for the bulk and Lagrange multiplier fields must satisfy the classical inf-sup condition, which usually leads to the need for stabilization. Moreover, additional degrees of freedom must be added to the problem. The use of Lagrange multipliers is the basis of the Fictitious Domain Method [23, 26].

Another possibility for imposing boundary conditions is the use of preexisting grid nodes to impose boundary conditions. This is the case of the FM-ALE method, which applies a variant of the ALE method to a fixed mesh strategy [2, 14, 15, 32] and the hybrid Cartesian/immersed boundary methods for Cartesian grids [19, 45, 57].

Most of these methods have been well tested in the literature for both steady and moving interfaces. Generally, the last case is treated by applying directly the former at each time step. In this work we review all these formulations from a unified point of view.

The chapter is organized as follows. In Section 2 we present a general overview of the physical problem and equations with which fixed mesh methods have to deal, which we particularize for the finite element method. In Section 3 we describe several existing possibilities for imposing Dirichlet boundary conditions in fixed mesh methods. Imposing Dirichlet boundary conditions is an essential ingredient of any fixed grid strategy, and it is very closely related to treatment of the ALE equations in moving domains. In Section 4 we present different possibilities to treat moving subdomains with fixed-mesh strategies. Finally, a summary of the methods described closes the chapter in Section 5.

2 Problem statement

In this section we state the continuous problem to be solved. Fixed mesh methods applied to problems in moving domains appear in different applications. However, to fix ideas we will concentrate our developments to fluid-structure interaction problems in which the solid (structure) either deforms or has rigid body motions. The fluid domain will therefore change in time. Our main interest is to describe strategies able to deal with this situation. A more detailed description of the problem can be found in [2, 14].

2.1 Flow equations

Let us consider a region $\Omega^0 \subset \mathbb{R}^d$ ($d = 2, 3$) where a flow will take place during a time interval $[0, T]$. However, we consider the case in which the fluid at time t occupies only a subdomain $\Omega(t) \subset \Omega^0$ (note in particular that $\Omega(0) \subset \Omega^0$). Suppose also that the boundary of $\Omega(t)$ is defined by part of $\partial\Omega^0$ and a moving boundary that we call $\Gamma_{\text{free}}(t) = \partial\Omega(t) \setminus \partial\Omega^0 \cap \partial\Omega(t)$. This moving part of $\partial\Omega(t)$ may correspond to the boundary of a moving solid immersed in the fluid.

In order to cope with the time-dependency of $\Omega(t)$, we may use the ALE approach, with the particular feature of considering a variable definition of the domain velocity. Let $\boldsymbol{\chi}_t$ be a family of invertible mappings, which for all $t \in [0, T]$ map a point $\boldsymbol{X} \in \Omega(0)$ to a point $\boldsymbol{x} = \boldsymbol{\chi}_t(\boldsymbol{X}) \in \Omega(t)$, with $\boldsymbol{\chi}_0 = \boldsymbol{I}$, the identity. If $\boldsymbol{\chi}_t$ is given by the motion of the particles, the resulting formulation would be Lagrangian, whereas if $\boldsymbol{\chi}_t = \boldsymbol{I}$ for all t, $\Omega(t) = \Omega(0)$ and the formulation would be Eulerian.

Let now $t' \in [0, T]$, with $t' \leq t$, and consider the mapping

$$\begin{aligned} \boldsymbol{\chi}_{t,t'} : \Omega(t') &\longrightarrow \Omega(t) \\ \boldsymbol{x}' &\mapsto \boldsymbol{x} = \boldsymbol{\chi}_t \circ \boldsymbol{\chi}_{t'}^{-1}(\boldsymbol{x}'). \end{aligned}$$

Given a function $f : \Omega(t) \times (0, T) \longrightarrow \mathbb{R}$ we define

$$\left.\frac{\partial f}{\partial t}\right|_{\boldsymbol{x}'} (\boldsymbol{x}, t) := \frac{\partial (f \circ \boldsymbol{\chi}_{t,t'})}{\partial t}(\boldsymbol{x}', t), \quad \boldsymbol{x} \in \Omega(t),\ \boldsymbol{x}' \in \Omega(t').$$

In particular, the domain velocity taking as a reference the coordinates of $\Omega(t')$ is given by

$$\boldsymbol{u}_{\rm dom} := \left.\frac{\partial \boldsymbol{x}}{\partial t}\right|_{\boldsymbol{x}'} (\boldsymbol{x}, t). \tag{1}$$

The incompressible Navier-Stokes formulated in $\Omega(t)$, accounting also for the motion of this domain, can be written as follows: find a velocity $\boldsymbol{u} : \Omega(t) \times (0, T) \longrightarrow \mathbb{R}^d$ and a pressure $p : \Omega(t) \times (0, T) \longrightarrow \mathbb{R}$ such that

$$\rho \left[\left.\frac{\partial \boldsymbol{u}}{\partial t}\right|_{\boldsymbol{x}'} (\boldsymbol{x}, t) + (\boldsymbol{u} - \boldsymbol{u}_{\rm dom}) \cdot \nabla \boldsymbol{u} \right] - \nabla \cdot (2\mu \nabla^S \boldsymbol{u}) + \nabla p = \rho \boldsymbol{f}, \tag{2}$$

$$\nabla \cdot \boldsymbol{u} = 0, \tag{3}$$

where $\nabla^S \boldsymbol{u}$ is the symmetrical part of the velocity gradient, ρ is the fluid density, μ is the viscosity and $\boldsymbol{f}$ is the vector of body forces.

Initial and boundary conditions have to be appended to problem (2)-(3). The boundary conditions on $\Gamma_{\rm free}(t)$ can be of two different types: a) p (or the normal stress) given, $\boldsymbol{u}$ unknown on $\Gamma_{\rm free}$; b) $\boldsymbol{u}$ given, p (or the normal stress) unknown on $\Gamma_{\rm free}$. On the rest of the boundary of $\Omega(t)$ the usual boundary conditions can be considered. In general, we consider these boundary conditions of the form

$$\begin{aligned} \boldsymbol{u} &= \bar{\boldsymbol{u}} \quad \text{on } \Gamma_D, \\ \boldsymbol{n} \cdot \boldsymbol{\sigma} &= \bar{\boldsymbol{t}} \quad \text{on } \Gamma_N, \end{aligned}$$

where $\boldsymbol{n}$ is the external normal to the boundary, $\boldsymbol{\sigma} = -p\boldsymbol{I} + 2\mu\nabla^S \boldsymbol{u}$ is the Cauchy stress tensor and $\bar{\boldsymbol{u}}$ and $\bar{\boldsymbol{t}}$ are the given boundary data. The components of the boundary Γ_D and Γ_N are obviously disjoint and such that $\Gamma_D \cup \Gamma_N = \partial\Omega$, and therefore time-dependent.

We will also introduce the time and spatial discretization of the problem since some of the methods we will describe are directly applied to the discrete problem. We will focus in the finite element method, although other possibilities are obviously possible.

2.1.1 The time-discrete problem

Let us start introducing some notation. Consider a uniform partition of $[0, T]$ into N time intervals of length δt. Let us denote by f^n the approximation of a time dependent function f at time level $t^n = n\delta t$. We will also denote

$$\begin{aligned} \delta f^{n+1} &= f^{n+1} - f^n, \\ \delta_t f^{n+1} &= \frac{f^{n+1} - f^n}{\delta t}, \\ f^{n+\theta} &= \theta f^{n+1} + (1-\theta) f^n, \quad \theta \in [1/2, 1]. \end{aligned}$$

Suppose we are given a computational domain at time t^n, with spatial coordinates labeled $\boldsymbol{x}^n$, and $\boldsymbol{u}^n$ and p^n are known in this domain. The velocity $\boldsymbol{u}^{n+1}$ and the pressure p^{n+1} can then be found as the solution to the problem

$$\rho \left[\delta_t \boldsymbol{u}^{n+1}\big|_{\boldsymbol{x}^n} + (\boldsymbol{u}^{n+\theta} - \boldsymbol{u}^{n+\theta}_{\text{dom}}) \cdot \nabla \boldsymbol{u}^{n+\theta} \right] - \nabla \cdot (2\mu \nabla^S \boldsymbol{u}^{n+\theta}) + \nabla p^{n+1} = \rho \boldsymbol{f}^{n+1}, \tag{4}$$

$$\nabla \cdot \boldsymbol{u}^{n+\theta} = 0, \tag{5}$$

where now $\delta_t \boldsymbol{u}^{n+1}|_{\boldsymbol{x}^n} = (\boldsymbol{u}^{n+1}(\boldsymbol{x}) - \boldsymbol{u}^n(\boldsymbol{x}^n))/\delta t$, being $\boldsymbol{x} = \boldsymbol{\chi}_{t^{n+\theta},t^n}(\boldsymbol{x}^n)$ the spatial coordinates in $\Omega(t^{n+\theta})$. The domain velocity given by (1), with $\boldsymbol{x}' = \boldsymbol{x}^n$, is approximated as

$$\boldsymbol{u}^{n+\theta}_{\text{dom}} = \frac{1}{\theta \delta t} \left(\boldsymbol{\chi}_{t^{n+\theta},t^n}(\boldsymbol{x}^n) - \boldsymbol{x}^n \right). \tag{6}$$

2.1.2 The fully discrete problem

The next step is to consider the spatial discretization of problem (4)-(5). As for the time discretization, different options are possible. For the sake of conciseness, here we simply describe the straight-forward Galerkin finite element formulation, even though there are instabilities that might appear due to dominant convective term or incompatible velocity-pressure interpolations. In order to overcome these numerical problems of the standard Galerkin method, a stabilized finite element formulation can be applied. The formulation we use is presented in [11]. It is based on the subgrid scale concept introduced in [34], although when linear elements are used it reduces to the Galerkin/least-squares method described for example in [18].

Let $\{\Omega^e\}^{n+1}$ be a finite element partition of the domain $\Omega(t^{n+1})$, with index e ranging from 1 to the number of elements n_{el} (which may be different at different time steps). We denote with a subscript h the finite element approximation to the unknown functions, and by $\boldsymbol{v}_h$ and q_h the velocity and pressure test functions associated to $\{\Omega^e\}^{n+1}$, respectively.

The finite element discretized method consists of finding $\boldsymbol{u}^{n+1}_h$ and p^{n+1}_h such that

$$\begin{aligned} &\int_\Omega \boldsymbol{v}_h \cdot \rho \ \delta_t \boldsymbol{u}^{n+1}\big|_{\boldsymbol{x}^n} + \int_\Omega 2\nabla^S \boldsymbol{v}_h : \mu \nabla^S \boldsymbol{u}^{n+\theta} \\ &\quad + \int_\Omega \boldsymbol{v}_h \cdot (\rho \, (\boldsymbol{u}^{n+\theta} - \boldsymbol{u}^{n+\theta}_{\text{dom}}) \cdot \nabla \boldsymbol{u}^{n+\theta}) - \int_\Omega p^{n+1}_h \nabla \cdot \boldsymbol{v}_h \\ &\quad = \int_\Omega \boldsymbol{v}_h \cdot \boldsymbol{f}^{n+\theta}, \end{aligned} \tag{7}$$

$$\int_\Omega q_h \nabla \cdot \boldsymbol{u}^{n+\theta}_h = 0. \tag{8}$$

2.2 Solid body equations

Let us now consider the solid body domain $\Omega_s(t) \subset \Omega^0$, which also evolves in time. The solid mechanics problem formulated in a purely Lagrangian approach in $\Omega_s(t)$

can be written as follows:

$$\rho_s \frac{d^2 \boldsymbol{d}}{dt^2} = \nabla \cdot \boldsymbol{\sigma}_s + \rho_s \boldsymbol{b}, \tag{9}$$

$$\rho_s J = \rho_{s0}, \tag{10}$$

where ρ_s is the solid density, $\boldsymbol{d}$ is the displacement vector, $\boldsymbol{\sigma}_s$ is the Cauchy stress tensor, $\boldsymbol{b}$ is the vector of body forces and

$$\boldsymbol{F} = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{X}}, \quad J = \det(\boldsymbol{F}).$$

For linear elastic isotropic materials we have $\sigma_s = \lambda_s (\nabla \cdot \boldsymbol{d}) \boldsymbol{I} + 2\mu_s \nabla^s \boldsymbol{d}$. Rigid bodies could be treated as well in the methods to be described in the following.

The time discretization and finite element approximation of the above equations is standard. We omit the details, which can be found for example in [2].

3 Approximate imposition of boundary conditions

3.1 Motivation

As indicated previously, moving domain methods on fixed meshes are closely linked to methods to approximate boundary conditions on non-matching meshes, since the latter are a crucial ingredient of the former. For example, let us consider a fluid-structure interaction problem as described above. A classical way to proceed is to solve equations (7)-(8) coupled with the discretized version of equations (9)-(10) in an iterative manner, using the continuity of velocities and stresses on the interface boundary as transmission conditions between the fluid and the solid. In order to have a stable scheme, stress conditions need to be applied to the solid (transmitting the stresses exerted by the fluid) and velocity (Dirichlet) conditions to the fluid (fixing the motion of the boundary of the fluid domain by the motion obtained in the solid). The key point is precisely the prescription of these conditions on meshes that, as time evolves, will not match the flow domain. We concentrate now on the description of methods to treat this problem, which can be stated for stationary problems as well.

3.2 Problem setting

Let us now focus in the solution of the flow problem, for which we will use a non-matching finite element discretization. The ideas to be presented are extendable to other numerical formulations and other physical problems. However, some of the difficulties we shall mention are characteristic of flow problems. Likewise, we will consider general non-structured meshes, the application to Cartesian meshes being obvious.

An issue of special relevance when using non-matching grids is the imposition of Dirichlet boundary conditions. Let us describe the problem to be solved. Consider the

situation depicted in Fig. 1. A domain $\Omega \subset \mathbb{R}^d$, $d = 2, 3$, with boundary $\Gamma = \partial\Omega$ (red curve in Fig. 1), is covered by a mesh that occupies a domain $\Omega_h = \Omega_{\rm in} \cup \Omega_\Gamma$, where $\Omega_{\rm in} \subset \Omega$ is formed by the elements interior to Ω and Ω_Γ is formed by a set of elements cut by Γ. In turn, let us split $\Omega_\Gamma = \Omega_{\Gamma,{\rm in}} \cup \Omega_{\Gamma,{\rm out}}$, where $\Omega_{\Gamma,{\rm in}} = \Omega \cap \Omega_\Gamma$ and $\Omega_{\Gamma,{\rm out}}$ is the interior of $\Omega_\Gamma \setminus \Omega_{\Gamma,{\rm in}}$. Note that $\Omega = \Omega_{\rm in} \cup \Omega_{\Gamma,{\rm in}}$. For simplicity, we will assume that the intersection of Γ with the element domains is a piecewise polynomial curve (in 2D) or surface (in 3D) of the same order as the finite element interpolation.

Suppose we want to solve a boundary value problem for the unknown u in Ω with the mesh of Ω_h already created and boundary conditions $u = \bar{u}$ on Γ. The obvious choice would be:

- Obtain the nodes of Γ (circles in Fig. 1) from the intersection with the element edges.
- Split the elements of $\Omega_{\Gamma,{\rm in}}$ so as to obtain a grid matching the boundary Γ.
- Prescribe the boundary condition $u_h = \bar{u}$ in the classical way, where u_h denotes the approximate solution.

This strategy leads to a local remeshing close to Γ that is involved from the computational point of view. Obviously, the implementation of the strategy described is very simple for unstructured simplicial meshes, but it is not so easy if one wants to use other element shapes and, definitely, prevents from using Cartesian meshes. Moreover, if the boundary Γ evolves in time (a situation considered later) the number of degrees of freedom changes at each time instant, thus modifying the structure and sparsivity of the matrix of the final algebraic system. This is clearly an inconvenience even when using unstructured simplicial meshes. There are several methods for imposing boundary conditions which avoid the need for locally remeshing. Some of these methods are described next.

3.3 Immersed boundary method

The immersed boundary method was introduced in 1972 by C. Peskin (see [39, 48] and [49] for an overview), and since then it has been widely used to simulate fluid-structure interaction problems in moving domains. The key point of the immersed boundary method is how Dirichlet boundary conditions are imposed: in the immersed boundary method boundary conditions are imposed through the introduction of a force in the momentum equation. In principle this force is introduced only in the fluid-solid interface by means of a Dirac-delta function, but in practice this function has to be extended to the nodes surrounding the interface due to the discrete nature of finite element meshes, and the Dirac-delta function is smoothed.

Suppose that the interface is represented by a set of Lagrangian points of coordinates $\boldsymbol{x}_{\Gamma,i}$, $i = 1, 2, ..., P$ (points on the red curve in Fig. 1), which typically correspond to nodes in the Lagrangian solid body mesh, and that the force to be added is

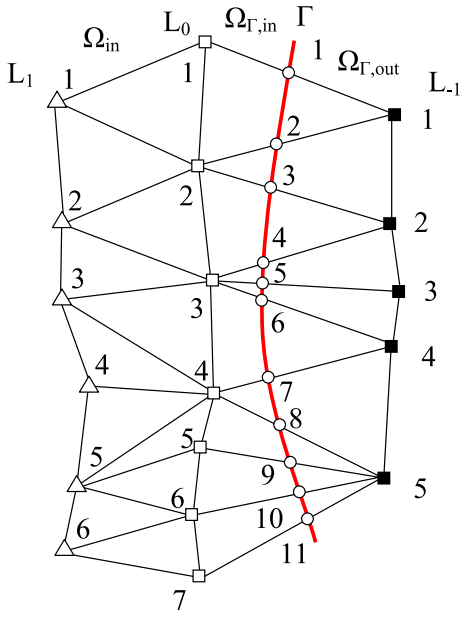

Figure 1: Setting

elastic. If the interface is considered a membrane, it should be computed from the coupling of the membrane with the fluid, and this is what is described for example in [49]. However, a simpler option, often used in the applications, is to consider this force as proportional to the deviation of the boundary value of the velocity to the boundary condition, in a spring-like model. If k is the constant of proportionality, the punctual force introduced by the immersed boundary method applied to the Navier-Stokes equations would be of the form

$$\boldsymbol{f} = \sum_{i=1}^{P} k(\boldsymbol{u} - \bar{\boldsymbol{u}})\delta(|\boldsymbol{x} - \boldsymbol{x}_{\Gamma,i}|),$$

where summation extends to all nodes representing the boundary. This force, when smoothed and introduced in the weak form of the problem yields the right-hand-side term

$$\int_{\Omega} \boldsymbol{v}_h \cdot \boldsymbol{f} = \int_{\Omega} \boldsymbol{v}_h \cdot \sum_{i=1}^{P} k(\boldsymbol{u} - \bar{\boldsymbol{u}})\bar{\delta}(|\boldsymbol{x} - \boldsymbol{x}_{\Gamma,i}|).$$

The *elastic/penalty* constant k has to be large enough to guarantee that boundary conditions are strongly enough imposed, and $\bar{\delta}$ is the smoothed Dirac-delta function which controls the extent of the applied force.

In the continuous problem, k would be unbounded, and the Dirac-delta function would not be smoothed at all. However, using very large values for k and very sharp $\bar{\delta}$ functions highly ill-conditions the discrete system of equations to be solved, which is the reason for the smoothing. Note that this force extends not only over the fluid-solid interface, but over the whole computational domain and needs to be integrated.

However, the $\bar{\delta}$ function limits its extent to one or two layers of nodes at each side of the interface. The immersed boundary method is qualified to be a diffusive method in [44]. The reason stems from the discretization of the Dirac delta function which has a finite support. Based on the same coupling principle, Wang and Liu [54] devised the Extended Immersed Boundary method to allow volumetric deformation of the elastic solid.

3.4 Penalty methods and Nitsche's method

Penalty methods are similar to Peskin's methods in that they introduce a force to impose boundary conditions. The main difference is the definition of this force. While in the immersed boundary method this force is many times understood as an elastic spring, in penalty methods the nature of the force is purely numerical, and it even depends on the discretization through a parameter which represents the element size. Moreover, the extent of the penalty force in penalty methods is only over the interface surface and not over the whole computational domain. The expression for the penalty force is:

$$\boldsymbol{f} = \frac{\alpha}{h}(\boldsymbol{u} - \bar{\boldsymbol{u}}),$$

which introduced in the discrete weak form of the problem yields

$$\int_\Gamma \boldsymbol{v}_h \cdot \boldsymbol{f} = \int_\Gamma \boldsymbol{v}_h \cdot \frac{\alpha}{h}(\boldsymbol{u}_h - \bar{\boldsymbol{u}}), \tag{11}$$

where α is the penalty parameter and h is the element size. The larger α is the stronger the boundary conditions are imposed, but also the more ill-conditioned the resulting system of equations becomes. Note that, unlike in the immersed boundary method, the integral extends only over the contour of the domain.

Nitsche's method can be understood as an improvement of the original penalty method. If the penalty forces (11) are added, there is no reason why the test function associated to the boundary nodes must vanish, and therefore this contribution must be accounted for. If this is not done, a poor approximation of the unknowns close to the interface is obtained, even if the boundary condition is well approximated.

If we define the operator

$$\boldsymbol{\sigma}(\boldsymbol{v}, q) := -2\boldsymbol{n} \cdot \mu \nabla^S \boldsymbol{v} + q\boldsymbol{n},$$

the term to be added to the right-hand-side of (7) is

$$-\int_\Gamma \boldsymbol{v}_h \cdot \boldsymbol{\sigma}(\boldsymbol{u}_h, p_h) + \int_\Gamma \boldsymbol{v}_h \cdot \frac{\alpha}{h}(\boldsymbol{u}_h - \bar{\boldsymbol{u}}). \tag{12}$$

Of course, terms involving $\boldsymbol{u}_h$ and p_h should be moved to the left-hand-side (LHS) of (7), since they contribute to the system matrix. The problem with (12) is that it yields a non-symmetric problem, even in the Stokes case. In order to avoid this, the Dirichlet

condition $\boldsymbol{u} = \bar{\boldsymbol{u}}$ is weighted by $\boldsymbol{\sigma}(\boldsymbol{v}_h, q_h)$. Thus, instead of adding only (11) to the LHS of (7), the method consists in adding

$$-\int_\Gamma \boldsymbol{v}_h \cdot \boldsymbol{\sigma}(\boldsymbol{u}_h, p_h) - \int_\Gamma \boldsymbol{u}_h \cdot \boldsymbol{\sigma}(\boldsymbol{v}_h, q_h) + \int_\Gamma \bar{\boldsymbol{u}} \cdot \boldsymbol{\sigma}(\boldsymbol{v}_h, q_h) + \int_\Gamma \boldsymbol{v}_h \cdot \frac{\alpha}{h}(\boldsymbol{u}_h - \bar{\boldsymbol{u}}). \tag{13}$$

Again, terms involving $\boldsymbol{u}_h$ and p_h should be moved to the LHS of (7). The resulting problem is symmetric if the Stokes problem is considered. Let us remark that both in (11) and (13) it is important to evaluate the unknowns at the time step of consideration, since explicit treatments are usually highly unstable. When added to (7), the natural choice is to evaluate (11) and (13) at $n + \theta$.

Nitsche's method was proposed for the Poisson problem and analyzed in this case. It can be shown that the higher the value of α, the better the approximation to the boundary condition at the expense of a poorer approximation to the differential equation. However, for any value of α it is possible to show that the method is stable and optimally convergent. See [37] for a proof, including more general boundary conditions than used here (although for Poisson's problem). The good performance of Nitsche's method has been exploited also in other contexts, such as the imposition of boundary conditions for discontinuous finite element approximations (see the original work in [1] and the extension in [29], for example), the imposition of transmission conditions in domain decomposition with non-matching grids (as in [4, 28], among many others) or also in some stabilized finite element methods for which this method fits nicely [8].

3.5 Lagrange multiplier techniques

Another possibility to enforce boundary conditions, which does not involve a large penalty or elastic terms, is the use of Lagrange multipliers. Lagrange multipliers consist of adding new equations to the global system of equations that enforce the boundary conditions. New unknowns (the Lagrange multipliers) need also to be added to the problem. The main advantage of this procedure is that the system does not become ill-conditioned. On the other hand, the system of equations becomes larger, and the space for the Lagrange multipliers has to be carefully chosen so that the final formulation is stable (or stabilization techniques should be used, see for example [3, 17]).

Using Lagrange multipliers to impose boundary conditions to the Navier-Stokes equations consists of adding the term

$$\int_\Gamma \boldsymbol{v}_h \cdot \boldsymbol{\lambda}_h$$

to the LHS of the momentum equation (7) and to add the new equations for the Lagrange multipliers

$$\int_\Gamma \boldsymbol{\gamma}_h \cdot (\boldsymbol{u}_h - \bar{\boldsymbol{u}}) = \boldsymbol{0},$$

where $\boldsymbol{\lambda}_h$ are the Lagrange multipliers of the finite element problem and $\boldsymbol{\gamma}_h$ their associated test functions. Note that the Lagrange multipliers contribution to the momentum equation corresponds exactly to that of the stresses through the boundary, that is to say, $\boldsymbol{\lambda} = \boldsymbol{\sigma}(\boldsymbol{u}, p)$ for the continuous problem. In the discrete case, their introduction avoids the need for post-processing the stresses in order to compute the forces exerted by the fluid on the solid body.

The Lagrange multiplier technique for imposing boundary conditions is associated to the *fictitious domain* method, in which the flow problem is solved over all the computational domain, including the region occupied by the solid body [22, 24]. We describe it later as a method to treat problems in moving domains.

3.6 Minimization of boundary errors with external degrees of freedom

Another possibility in order to avoid the ill-conditioning due to penalty terms (which appears in the immersed boundary and penalty methods), and the need of adding new degrees of freedom to the system of equations (which appears if Lagrange multipliers are used) is to use currently existing degrees of freedom in order to enforce boundary conditions. This involves replacing momentum equations in nodes adjacent to the fluid-solid interface with the equations that enforce boundary conditions.

There are several methods which use pre-existing degrees of freedom in order to enforce boundary conditions but we will focus here in our own approach to the problem. We summarize next the strategy proposed in [12] to prescribe Dirichlet boundary conditions on a generic immersed boundary, that we denote as before by Γ.

Let u_h be the unknown solution of a problem posed in $\Omega \subset \Omega^0$ for which we want to prescribe a condition on Γ. Let Ω_Γ be the set of elements cut by Γ, which is split as $\Omega_\Gamma = \Omega_{\Gamma,\mathrm{in}} \cup \Omega_{\Gamma,\mathrm{out}}$, where $\Omega_{\Gamma,\mathrm{in}} = \Omega \cap \Omega_\Gamma$ and $\Omega_{\Gamma,\mathrm{out}}$ is the interior of $\Omega_\Gamma \setminus \Omega_{\Gamma,\mathrm{in}}$. Let also Ω_{in} be such that $\Omega = \Omega_{\mathrm{in}} \cup \Omega_{\Gamma,\mathrm{in}}$. For simplicity, we will assume that the intersection of Γ with the element domains can be exactly represented by the classical isoparametric mapping. For the notation to be used, see again Fig. 1.

Suppose that the unknown u_h is interpolated as

$$\begin{aligned} u_h(\boldsymbol{x}) &= \sum_{a=1}^{n_{\mathrm{in}}} I^a_{\mathrm{in}}(\boldsymbol{x}) U^a_{\mathrm{in}} + \sum_{b=1}^{n_{\mathrm{out}}} I^b_{\mathrm{out}}(\boldsymbol{x}) U^b_{\mathrm{out}} \\ &= \boldsymbol{I}_{\mathrm{in}}(\boldsymbol{x}) \boldsymbol{U}_{\mathrm{in}} + \boldsymbol{I}_{\mathrm{out}}(\boldsymbol{x}) \boldsymbol{U}_{\mathrm{out}}, \end{aligned}$$

where $I^a_{\mathrm{in}}(\boldsymbol{x})$ and $I^b_{\mathrm{out}}(\boldsymbol{x})$ are the standard interpolation functions, n_{in} is the number of nodes in Ω_{in}, the domain where the problem needs to be solved (including layer L_0) and n_{out} the number of nodes in layer L_{-1} (see Fig. 1).

The objective is to compute $\boldsymbol{U}_{\mathrm{out}}$. Suppose that u_h needs to be prescribed to a given

function $\bar{u}$ on Γ. The main idea is to compute $\boldsymbol{U}_{\text{out}}$ by minimizing the functional

$$J_2(\boldsymbol{U}_{\text{in}}, \boldsymbol{U}_{\text{out}}) = \int_\Gamma (u_h(\boldsymbol{x}) - \bar{u}(\boldsymbol{x}))^2 = \int_\Gamma (\boldsymbol{I}_{\text{in}}(\boldsymbol{x})\boldsymbol{U}_{\text{in}} + \boldsymbol{I}_{\text{out}}(\boldsymbol{x})\boldsymbol{U}_{\text{out}} - \bar{u}(\boldsymbol{x}))^2 . \tag{14}$$

Suppose now that the problem for u_h in Ω_{in} leads to an algebraic equation of the form

$$\boldsymbol{K}_{\text{in,in}}\boldsymbol{U}_{\text{in}} + \boldsymbol{K}_{\text{in,out}}\boldsymbol{U}_{\text{out}} = \boldsymbol{F}_{\text{in}}. \tag{15}$$

The domain integrals in matrices $\boldsymbol{K}_{\text{in,in}}$ and $\boldsymbol{K}_{\text{in,out}}$ extend only over Ω. The nodal values $\boldsymbol{U}_{\text{out}}$ are merely used as degrees of freedom to interpolate u_h in the domain Ω. If (15) is supplemented with the equation resulting from the minimization of functional (14), the system to be solved is finally

$$\begin{bmatrix} \boldsymbol{K}_{\text{in,in}} & \boldsymbol{K}_{\text{in,out}} \\ \boldsymbol{N}_\Gamma & \boldsymbol{M}_\Gamma \end{bmatrix} \begin{bmatrix} \boldsymbol{U}_{\text{in}} \\ \boldsymbol{U}_{\text{out}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{F}_{\text{in}} \\ \boldsymbol{f}_\Gamma \end{bmatrix}, \tag{16}$$

where

$$\boldsymbol{M}_\Gamma = \int_\Gamma \boldsymbol{I}_{\text{out}}^{\text{t}}(\boldsymbol{x})\boldsymbol{I}_{\text{out}}(\boldsymbol{x}), \quad \boldsymbol{f}_\Gamma = \int_\Gamma \boldsymbol{I}_{\text{out}}^{\text{t}}(\boldsymbol{x})\bar{u}(\boldsymbol{x}), \quad \boldsymbol{N}_\Gamma = \int_\Gamma \boldsymbol{I}_{\text{out}}^{\text{t}}(\boldsymbol{x})\boldsymbol{I}_{\text{in}}(\boldsymbol{x}).$$

It is important to note that this implementation maintains the connectivity of the background mesh.

4 Dealing with moving subdomains

In the previous section we have seen several possibilities to apply boundary conditions when non-matching grids are used. However, we have not discussed yet how to deal with moving subdomains. The most important issue in this type of problems is how to compute temporal derivatives. This is clearly defined for ALE strategies, but in fixed mesh methods it is not that straight-forward, and there are several possibilities for computing temporal derivatives in regions close to the fluid-solid interface. Each of these possibilities defines a family of methods.

Before starting to describe the various methods, let us state which is the key issue when dealing with moving subdomains in fixed mesh strategies. Let us consider the situation depicted in Fig. 2 at t^n, where we have a circular solid body immersed in a fluid which we want to simulate using a fixed mesh strategy. We have depicted in green nodes inside the fluid computational domain. Let us now consider the situation at time step t^{n+1}, where the circular body has been horizontally displaced. There are nodes which were outside the flow computational domain at t^n, which are inside the computational domain at t^{n+1} (green nodes inside the red circle in Fig. 2). These nodes are called *newly created nodes*. Since the unknown values at these nodes at t^n is not known, computing temporal derivatives is not immediate. The following sections describe some of the proposed strategies to do so.

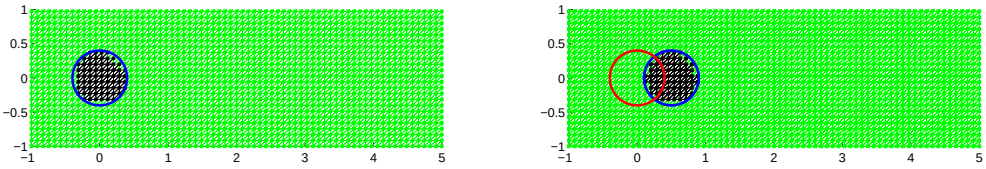

Figure 2: Green: nodes and elements inside the computational domain. Left: domain configuration at t^n. Right: domain configuration at t^{n+1}. The red circle corresponds to the domain configuration at t^n.

4.1 The fictitious domain method

In the fictitious domain method [22, 24] we work with a computational domain which does not coincide with the physical flow domain. Instead, we use as computational domain for the flow problem the whole region covered by the finite element mesh, that is, we solve in both the green and black regions in Fig. 2. Obviously, we are not interested in the solution inside the solid body region, which is physically meaningless.

The key point of the fictitious domain method is that, since we solve for the whole computational domain, we can use the results at t^n for computing temporal derivatives at t^{n+1} *even for the newly created nodes*. Let us say, however, that results at t^n for newly created nodes at t^{n+1} do not come from a physical problem but from a *fictitious* one, and that it is not clear how results in both the physical and the fictitious problem are related.

The fictitious domain method has been extensively used to simulate two and three-dimensional flow problems with moving boundaries having a known trajectory, and in particular to the solution of a Couette problem and a helical ribbon mixer [7]. As another example, it has been applied to the solution of the flow around a moving disk [25]. In fictitious domain methods, the motion of the object needs not necessarily to be known a-priori, and aerodynamic forces can be taken into account to couple the fluid dynamics and the kinematics of the rigid body. Pan [46] predicts the path of a ball falling in a viscous fluid (at low Reynolds numbers); in [20], the authors solve the two-dimensional flow around an airfoil that is free to rotate around its center of mass, the sedimentation of particles in a box, and a three-dimensional case involving two spherical particles. Using the same method, Juárez [36] simulates the sedimentation of an elliptic body in a two-dimensional viscous fluid. This fictitious domain method is also well-suited for shape optimization problems [21].

4.2 The fixed-mesh ALE method

The key point of the fixed-mesh ALE method is that, even if we are using a fixed-mesh method, domain movement cannot be obviated, and it is necessary to take it into account with an ALE method. However, since we are interested in using a fixed-mesh, we develop a strategy which allows us to always work with the background fixed mesh and, at the same time, includes the ALE terms which appear due to the domain movement. We have developed this strategy in [2, 14, 15, 32].

Suppose Ω^0 is meshed with a finite element mesh M^0 and that at time level t^n the domain $\Omega(t^n)$ is meshed with a finite element mesh M^n (as we will see, close to M^0). Let $\boldsymbol{u}^n$ be the velocity already computed on $\Omega(t^n)$. The purpose is to obtain the fluid region $\Omega(t^{n+1})$ and the velocity field $\boldsymbol{u}^{n+1}$. The former may move according to a prescribed kinematics, for example due to the motion of a solid, or can be an unknown of the problem. If the classical ALE method is used, M^n would deform to another mesh defined at t^{n+1}. The key idea is not to use this mesh to compute $\boldsymbol{u}^{n+1}$ and p^{n+1}, but to re-mesh in such a way that the new mesh is, essentially, M^0 once again.

The steps of the algorithm to achieve the goal described are the following:

1. Define $\Gamma_{\text{free}}^{n+1}$ by updating the function that defines it.
2. Deform *virtually* the mesh M^n to M_{virt}^{n+1} using the classical ALE concepts and compute the mesh velocity $\boldsymbol{u}_m^{n+1}$.
3. Write down the ALE Navier-Stokes equations on M_{virt}^{n+1}.
4. *Split the elements* of M^0 cut by $\Gamma_{\text{free}}^{n+1}$ to define a mesh on $\Omega(t^{n+1})$, M^{n+1}.
5. *Project* the ALE Navier-Stokes equations from M_{virt}^{n+1} to M^{n+1}.
6. Solve the equations on M^{n+1} to compute $\boldsymbol{u}^{n+1}$ and p^{n+1}.

A global idea of the meshes involved in the process is represented in Fig. 3. Note in particular that at each time steps two sets of nodes have to be appropriately dealt with, namely, the so called newly created nodes and the boundary nodes. Contrary to other fixed grid methods, newly created nodes are treated in a completely natural way using the FM-ALE approach: the value of the velocity there is directly given by the projection step from M_{virt}^{n+1} to M^{n+1}. Boundary nodes require either additional unknowns with respect to those of mesh M^0 or an appropriate imposition of boundary conditions, as explained in the previous Section.

4.3 Chimera strategies

The Chimera method was first envisaged as a tool for simplifying the mesh generation [5, 51, 52]. Independent meshes are generated for each component (object) of the

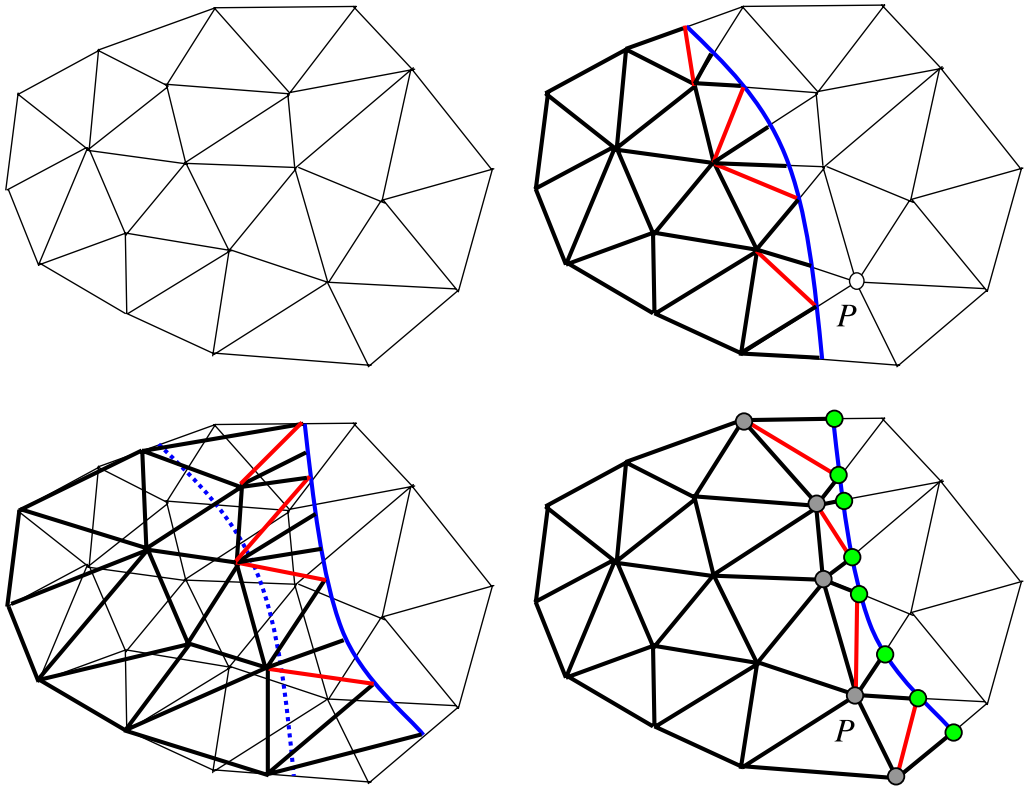

Figure 3: Two dimensional FM-ALE schematic. Top-left: original finite element mesh M^0 of Ω^0. Top-right: finite element mesh M^n of $\Omega(t^n)$, with the elements represented by a thick line and the elements of M^0 represented by thin line. The blue line represents Γ^n_{free} and the red edges indicate the splitting of M^0 to obtain M^n. Bottom-left: updating of M^n to M^{n+1}_{virt} using the classical ALE strategy. The position of $\Gamma^{n+1}_{\text{free}}$ is again shown using a solid blue line and the previous position Γ^n_{free} using a dotted blue line. Bottom-right: Mesh M^{n+1} of $\Omega(t^{n+1})$, represented by a thick line. The edges that split elements of M^0 are again indicated in red. Boundary nodes, where approximate boundary conditions need to be imposed, are drawn in green, whereas newly created nodes are drawn in gray.

computational domain, enabling a flexibility on the choice of the type of element as well as on their orientation that could not be possible when meshing complex three dimensional geometries [6,27]. Then, as a direct application, the Chimera method has also been used as a mesh refinement technique [47]. In addition, if it is implemented efficiently, it is a very efficient tool to treat flows with moving components [9, 50, 55].

Through the simple example sketched in Figure 4, let us briefly explain the Chimera method applied to the motion of moving objects inside a fluid domain:

- Independent meshes are generated for the so called background mesh and the mesh around the solid, which is called the *patch mesh*.

- The mesh around the solid is placed on the background mesh. The set of the two overset grids is the so called *composite grid*. Elements of the background located inside the patch are removed, with the possibility of having a non-empty intersection between the resulting mesh and the patch mesh. This process, known as *hole cutting*, defines two interfaces, one of the domain attached to the solid and the other, known as *apparent interface*, of the hole created in the background mesh. Nodes on this last interface are called *fringe nodes*. Nodes between the apparent interface and the outer boundary of the patch mesh are the *overlapping nodes*.

- The problem defined on the patch mesh is again a fluid-structure interaction problem. The main idea is to write the conservation equations on this domain in a frame of reference that moves with the rigid body motion of the solid. Then, the mesh deformation in the fluid region will be small, even if the solid deforms. Of course, the patch mesh will remain constant if the solid is a rigid body. In order to write the momentum conservation equation in the frame of reference following the rigid body modes of the body, Coriolis, centrifugal forces and acceleration forces have to be added to the Navier-Stokes equations. These are of the form

$$2\rho\,\boldsymbol{\omega}\times\boldsymbol{u},\quad \rho\,\boldsymbol{\omega}\times(\boldsymbol{\omega}\times\boldsymbol{x})\quad\text{and}\quad -\rho\,\boldsymbol{a}_s,$$

  respectively, where $\boldsymbol{\omega}$ is the angular velocity of the frame reference, $\boldsymbol{x}$ the position vector and $\boldsymbol{a}_s$ the acceleration.

- The FSI problem on the patch mesh is then coupled to the flow problem on the background mesh using a *domain decomposition strategy*. Typically, a Dirichlet-Dirichlet coupling is used (this is the so called Schwartz's method). This however requires a wide enough overlap to allow the associated iteration-by-subdomain strategy converge. Nevertheless, it is shown in [30, 31] that it is also possible to apply a Dirichlet condition on the apparent (non-smooth) interface and a Neumann condition on the outer boundary of the patch domain. Note that variables on the patch mesh and the background mesh will be referred to different reference systems, and thus the former need to be properly transformed to the fixed reference where they are usually required.

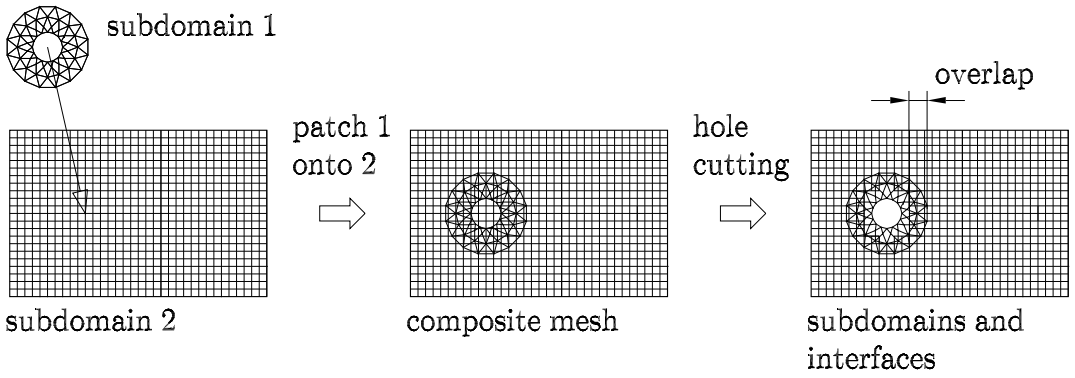

Figure 4: Chimera method: principles.

The reader can see the details of this approach in [30], where several examples solved using this method can be found.

4.4 Other possibilities

As we have already seen, a special treatment is required for newly created nodes. In many publications, the previous time step values are computed using ad hoc arguments, that sometimes lead to good approximations from the practical point of view when small time steps are used. As an example, in [43] the authors extrapolate the velocity and pressure from the nearest fluid nodes at the previous time step. In [10], the Navier-Stokes equations are correctly expressed in an ALE framework, but the velocity is taken as the solid velocity. It is worth to note that if the solid is deformable and has been solved together with the fluid in a coupled way (as in the original immersed boundary method [48] or in the fluid-solid approach in [58]), this velocity is physically meaningful. This is not the case, however, in the case of rigid bodies or bodies with rigid boundaries. A possibility to deal with this situation is to write the Navier-Stokes equations in a non-inertial frame of reference attached to the body, as in [38], where an immersed boundary method is used.

5 Summary

The purpose of this chapter has been to review some methods that allow to use fixed meshes in problems with time dependent domains, with particular emphasis on those in which the authors have been involved. The connection with the approximate imposition of boundary conditions has been highlighted.

Summarizing, the methods described herein are:

- Approximate imposition of Dirichlet boundary conditions:
  - Introduction of forces on the boundaries.

  - Penalization of the boundary conditions, including Nitsche's method.
  - Use of Lagrange multipliers to enforce boundary conditions.
  - Use of inactive degrees of freedom to optimize the imposition of boundary conditions.

- Treatment of time dependent domains:
  - Solving in the whole physical domain, both the solid and the fluid, with an appropriate modeling of the forces at the interface. This includes the original version of the immersed boundary method.
  - Fictitious domain method, in which the solid is considered to be filled with a fictitious fluid and boundary conditions are imposed through Lagrange multipliers.
  - Fixed-mesh ALE method, based on the classical ALE approach but projecting the equations always to a fixed mesh.
  - Chimera strategies based on domain decomposition method, which at least allow to remove the mesh deformation due to rigid body motions of the solids inside the fluid.

Many variants of all these methods exist. Our purpose here has been to describe the main ideas behind them, without entering the details that can be found in the references.

References

[1] D.N. Arnold. An interior penalty finite element method with discontinuous elements. *SIAM Journal on Numerical Analysis*, 19:742–760, 1982.

[2] J. Baiges and R. Codina. The fixed-mesh ale approach applied to solid mechanics and fluid-structure interaction problems. *International Journal for Numerical Methods in Engineering*, 81:1529–1557, 2010.

[3] H.J.C Barbosa and T.J.R Hughes. The finite element method with Lagrangian multipliers on the boundary: circumventing the Babuška-Brezzi condition. *Computer Methods in Applied Mechanics and Engineering*, 85:109–128, 1991.

[4] R. Becker, P. Hansbo, and R. Stenberg. A finite element method for domain decomposition with non-matching grids. *Mathematical Modelling and Numerical Analysis*, 37:209–225, 2003.

[5] J.A. Benek, P.G. Buning, and J.L. Steger. A 3-D Chimera grid embedding technique. In *7th AIAA Computational Fluid Dynamics Conference*, pages 322–331, Cincinnati (USA), July 15-17 1985. AIAA-1985-1523.

[6] J.A. Benek, T.L. Tonegan, and N.E. Suhs. Extended Chimera grid embedding system with application to viscous flow. In *8th AIAA Computational Fluid Dynamics Conference*, pages 283–291, June 9-11 1987. AIAA-87-1126.

[7] F. Bertrand, P.A. Tanguy, and F. Thibault. A three-dimensional fictitious domain method for incompressible fluid flow problems. *Int. J. Num. Meth. Fluids*, 25:719–736, 1997.

[8] E. Burman, M.A. Fernández, and P. Hansbo. Continuous interior penalty finite element method for Oseen's equations. *SIAM Journal on Numerical Analysis*, 44:1248–1274, 2006.

[9] J.-J. Chattot and Y. Wang. Improved treatment of interstecting bodies with the Chimera method and validation with a simple and fast flow solver. *Computers & Fluids*, 27(5-6):721–740, 1998.

[10] Y. Cheny and O. Botella. The LS-STAG method: A new immersed boundary / level-set method for the computation of incompressible viscous flows in complex moving geometries with good conservation properties. *Journal of Computational Physics*, 2008. Submitted.

[11] R. Codina. A stabilized finite element method for generalized stationary incompressible flows. *Computer Methods in Applied Mechanics and Engineering*, 190:2681–2706, 2001.

[12] R. Codina and J. Baiges. Approximate imposition of boundary conditions in immersed boundary methods. *Ijnme*, 80:1379–1405, 2009.

[13] R. Codina and G. Houzeaux. Implementation aspects of coupled problems in CFD involving time dependent domains, in *Verification and Validation Methods for Challenging Multiphysics Problems*, G. Bugeda, J.C Courty, A. Guilliot, R. Höld, M. Marini, T. Nguyen, K. Papailiou, J. Périaux and D. Schwamborn (Eds.), pages 99–123. CIMNE, Barcelona, 2006.

[14] R. Codina, J. Houzeaux, H. Coppola-Owen, and J. Baiges. The fixed-mesh ALE approach for the numerical approximation of flows in moving domains. *Journal of Computational Physics*, 228:1591–1611, 2009.

[15] H. Coppola-Owen and R. Codina. A finite element model for free surface flows on fixed meshes. *International Journal for Numerical Methods in Fluids*, 54:1151–1171, 2007.

[16] J. Donea, P. Fasoli-Stella, and S. Giuliani. Lagrangian and eulerian finite element techniques for transient fluid structure interaction problems. In *Transactions Fourth SMIRT*, page B1/2, 1977.

[17] A. Ern and J.-L. Guermond. *Theory and Practice of Finite Elements*. Springer-Verlag, 2004.

[18] L.P. Franca and S.L. Frey. Stabilized finite element methods: II. The incompressible Navier-Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 99:209–233, 1992.

[19] A. Gilmanov and F. Sotiropoulos. A hybrid Cartesian/immersed boundary method for simulating flows with 3D, geometrically complex, moving bodies. *Journal of Computational Physics*, 207:457–492, 2005.

[20] R. Glowinski, T.-W. Pan, T.I. Hesla, D.D. Joseph, and J. Périaux. A distributed Lagrange multiplier/fictitious domain method for flows around moving rigid bodies: application to particulate flow. *Int. J. Num. Meth. Fluids*, 30:1043–1066, 1999.

[21] R. Glowinski, T.-W. Pan, J. Kearsley, and J. Périaux. Numerical simulation and optimal shape for viscous flow by a fictitious domain method. *Int. J. Num. Meth. Fluids*, 20:685–711, 1995.

[22] R. Glowinski, T.-W. Pan, and J. Périaux. Fictitious domain methods for the Dirichlet problem and its generlization to some flow problems. In K. Morgan, E. Oñate, J. Périaux, J. Péraire, and O.C. Zienkiewicz, editors, *Finite Element in Fluids, New Trends and Applications*, pages 347–368. Pineridge Press, Barcelona (Spain), 1993.

[23] R. Glowinski, T.-W. Pan, and J. Périaux. A fictitious domain method for Dirichlet problems and applications. *Computer Methods in Applied Mechanics and Engineering*, 111:203–303, 1994.

[24] R. Glowinski, T.-W. Pan, and J. Périaux. A fictitious domain method for Dirichlet problems and applications. *Computer Methods in Applied Mechanics and Engineering*, 111:203–303, 1994.

[25] R. Glowinski, T.-W. Pan, and J. Périaux. On a domain embedding method for flow around moving rigid bodies. In Petter E. Bjørstad, Magne Espedal, and David Keyes, editors, *Ninth international Conference of Domain Decomposition Methods*. ddm.org, 1998. Proceedings from the Ninth International Conference, June 1996, Bergen, Norway.

[26] R. Glowinski, T.W. Pan, T.I. Hesla, D.D. Joseph, and J. Périaux. A distributed Lagrange multiplier/fictitious domain method for flows around moving rigid bodies: application to particulate flow. *International Journal for Numerical Methods in Fluids*, 30:1043–1066, 1999.

[27] E. Guilmineau, J. Piquet, and P. Queutey. Two-dimensional turbulent viscous flow simulation past airfoils at fixed incidence. *Computers & Fluids*, 26(2):135–162, 1997.

[28] A. Hansbo and P. Hansbo. An unfitted finite element method, based on Nitsche's method, for elliptic interface problems. *Computer Methods in Applied Mechanics and Engineering*, 191:5537–5552, 2002.

[29] P. Hansbo and M.G. Larson. Discontinuous Galerkin methods for incompressible and nearly incompressible elasticity by Nitsche's method. *Computer Methods in Applied Mechanics and Engineering*, 191:1895–1908, 2002.

[30] G. Houzeaux and R. Codina. A Chimera method based on a Dirichlet/Neumann (Robin) coupling for the Navier-Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 192:3343–3377, 2003.

[31] G. Houzeaux and R. Codina. An overlapping iteration-by-subdomain Dirichlet/Robin domain decomposition method for advection–diffusion problems. *Journal of Computational and Applied Mathematics*, 158:243–276, 2003.

[32] G. Houzeaux and R. Codina. A finite element model for the simulation of lost foam casting. *International Journal for Numerical Methods in Fluids*, 46:203–226, 2004.

[33] A. Huerta and W.K. Liu. Viscous flow with large free surface motion. *Computer Methods in Applied Mechanics and Engineering*, 69:277–324, 1988.

[34] T.J.R. Hughes. Multiscale phenomena: Green's function, the Dirichlet-to-Neumann formulation, subgrid scale models, bubbles and the origins of stabilized formulations. *Computer Methods in Applied Mechanics and Engineering*, 127:387–401, 1995.

[35] T.J.R. Hughes, W.K. Liu, and T.K. Zimmerman. Lagrangian-eulerian finite element formulation for incompressible viscous flows. *Computer Methods in Applied Mechanics and Engineering*, 29:329–349, 1981.

[36] L.H. Juárez. Numerical simulation of the sedimentation of an elliptic body in an incompressible viscous fluid. *C. R. Acad. Sci. Paris*, 329(Série IIb):221–224, 2001.

[37] M. Juntunen and R. Stenberg. Nitsche's method for general boundary conditions. Helsinki University of Technology, Institute of Mathematics, Research Reports A530, 2007.

[38] D. Kima and H. Choi. Immersed boundary method for flow around an arbitrarily moving body. *Journal of Computational Physics*, 212:662–680, 2006.

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

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

[41] R.J. LeVeque and Z. Li. The immersed interface method for elliptic equations with discontinuous coefficients and singular sources. *SIAM Journal on Numerical Analysis*, 31 (4):1019–1044, 1994.

[42] R.J. LeVeque and Z. Li. Immersed interface method for incompressible Navier-Stokes equations. *SIAM Journal on Scientific and Statistical Computing*, 18 (3):709–735, 1997.

[43] R. Löhner, J.R. Cebral, F.F. Camelli, J.D. Baum, and E.L. Mestreau. Adaptive embedded/immersed unstructured grid techniques. *Archives of Computational Methods in Engineering*, 14:279–301, 2007.

[44] S. Marella, S. Krishnan, H. Liu, and H.S. Udaykumar. Sharp interface cartesian grid method i: An easily implemented technique for 3d moving boundary computations. *Journal of Computational Physics*, 210:1–31, 2005.

[45] J. Mohd-Yusof. Combined immersed boundaries/B-splines methods for simulations of flows in complex geometries. *CTR annual research briefs, Stanford University, NASA Ames*, 1997.

[46] T.-W. Pan. Numerical simulation of the motion of a ball falling in an incompressible viscous fluid. *C. R. Acad. Sci. Paris*, 327(Série IIb):1035–1038, 1999.

[47] E. Pärt-Enander. *Overlapping Grids and Applications in Gas Dynamics*. PhD thesis, Uppsala Universitet, Sweden, 1995.

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

[49] C.S Peskin. The immersed boundary method. *Acta Numerica*, pages 479–517, 2002.

[50] N.C. Prewitt, D.M. Belk, and W. Shyy. Parallel computing of overset grids for aerodynamic problems with moving objects. *J. Progr. Aero. Sci.*, 36:117–172, 2000.

[51] J.L. Steger and J.A. Benek. On the use of composite grid schemes in computational aerodynamics. *Computer Methods in Applied Mechanics and Engineering*, 64:301–320, 1987.

[52] J.L. Steger, F.C. Dougherty, and J.A. Benek. A Chimera grid scheme. In K. N. Ghia and U. Ghia, editors, *Advances in Grid Generation*, volume ASME FED-5, pages 59–69, 1983.

[53] J.V. Voorde, J. Vierendeels, and E. Dick. Flow simulations in rotary volumetric pumps and compressors with the fictitious domain method. *Journal of Computational and Applied Mathematics*, 168:491–499, 2004.

[54] X. Wang and WKLiu. Extended immersed boundary method using fem and rkpm. *Computer Methods in Applied Mechanics and Engineering*, 193:1305–1321, 2004.

[55] Z.J. Wang and V. Parthasarathy. A fully automated Chimera methodology for multiple moving body problems. *Int. J. Num. Meth. Fluids*, 33:919–938, 2000.

[56] S. Xu and Z.J. Wang. An immersed interface method for simulating the interaction of a fluid with moving boundaries. *Journal of Computational Physics*, 216:454–493, 2006.

[57] J.H. Ferziger Y.H. Tseng. A ghost-cell immersed boundary method for flow in complex geometry. *Journal of Computational Physics*, 192:593–623, 2003.

[58] H. Zhao, J.B. Freund, and R.D. Moser. A fixed-mesh method for incompressible flow-structure systems with finite solid deformations. *Journal of Computational Physics*, 227:3114–3140, 2008.