A NURBS-based immersed methodology for fluid-structure interaction

Hugo Casquero[a,*], Carles Bona-Casas[a], Hector Gomez[a]

[a] *Departamento de Métodos Matemáticos, Universidade da Coruña, Campus de A Coruña, 15071, A Coruña, Spain.*

Abstract

We introduce an isogeometric, immersed, and fully-implicit formulation for fluid-structure interaction (FSI). The method focuses on viscous incompressible flows and nonlinear hyperelastic incompressible solids, which are a common case in various fields, such as, for example, biomechanics. In our FSI method, we utilize an Eulerian mesh on the whole domain and a Lagrangian mesh on the solid domain. The Lagrangian mesh is arbitrarily located on top of the Eulerian mesh in a non-conforming fashion. Due to the formulation of our problem, based on the Immersed Finite Element Method (IFEM), we do not need mesh update or remeshing algorithms. The fluid-structure interface is the boundary of the Lagrangian mesh, but cuts arbitrarily the Eulerian mesh. The generalized-α method is used for time discretization and NURBS-based isogeometric analysis is employed for the spatial discretization on both fluid and solid domains. The information transfer between the two meshes is carried out using the NURBS functions, which avoids the use of the so-called discretized delta functions. The higher order and especially the higher continuity of NURBS functions allow us to deal with severe mesh distortion in the Lagrangian mesh in comparison with classical $\mathcal{C}^0$ linear piecewise functions as we prove numerically. Our numerical solutions attain good agreement with theoretical data for free-falling objects in two and three dimensions, which confirms the feasibility of our methodology.

Keywords:
Fluid-structure interaction, Particulate flow, Immersed methods, Isogeometric analysis, NURBS, Viscous incompressible flows, Variational multiscale, Nonlinear elasticity

1. Introduction

In the past few decades, numerous research efforts have been directed to the development of modeling and simulation techniques for fluid-structure interaction (FSI) problems. See, for example [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]. An efficient, accurate, and robust computational technique is essential in studying complicated FSI problems, which are ubiquitous in all

[*] Coresponding author.
Email address: hugo.casquero@udc.es (Hugo Casquero)

fields of engineering. Body-fitted approaches [1] have been widely utilized in FSI problems and are typically based upon the arbitrary Lagrangian Eulerian (ALE) description [11, 12, 13, 14]. The ALE description is used to track the fluid-solid interface properly, which is one of the main challenges in a FSI technique. These approaches are usually very accurate, but mesh update or remeshing algorithms are needed, which are time consuming and can lead to inaccuracies in the variables that are projected onto the mesh in some cases.

A different type of FSI methodology was introduced by Peskin in the 1970's [15]. Peskin developed the immersed boundary (IB) method, which may be applied to solids that take the form of a fiber network and are completely submerged into the fluid. In Peskin's method, a uniform, fixed Eulerian mesh spans the whole domain and the submerged solid is represented by a fiber network which is arbitrarily located on top of the Eulerian mesh. A discretized Dirac delta function with $\mathcal{C}^1$ continuity is employed to transfer information between the solid and the Eulerian mesh. This information transfer represents the interaction between the fluid and the solid, which is accounted for by distributing interaction forces in the fluid and interpolating the solid velocities from the fluid velocities. In this way, the action of the solid on the fluid is represented by a forcing function. Thus, on the Eulerian mesh only the forced Navier-Stokes equations need to be solved and no mesh-update algorithms are necessary. The IB method has been applied to a large variety of problems such as the design of prosthetic cardiac valves [16], wave propagation in cochlea [17], biofilm processes [18], and swimming motions of marine worms [19].

In the 2000's, the IB method was enhanced and extended by Liu and his coworkers giving rise to a new algorithm called immersed finite element method (IFEM) [20]. One of the major limitations of the IB method is that the solid carries mass, but does not occupy volume. This limitation is circumvented in the IFEM. The IFEM uses finite elements for both the fluid and the solid. In origin, the IFEM also employed a discretized delta function for the information transfer between meshes, but it was removed later in [21] and the finite element shape functions were used instead. The IFEM has been used to solve several problems of biomechanics such as stent deployment [22], cell migration, and cell-substrate adhesion [6] and hemodynamics [23, 24]. We note, however, that as far as we are aware, the IFEM has not been applied to flows with very large Reynolds number and these problems could be difficult to tackle by the IFEM because the boundary layer at the fluid-structure interface must be resolved by the Eulerian mesh.

We believe that the IFEM was a very significant development which opened the door to the solution of FSI problems that could not have been computed otherwise. Nevertheless, the generality of the IFEM provides opportunities for different interpretations, and opens the way to potential improvements. In particular, some of the limitations of the IFEM seem to emanate from its underlying technology, namely the finite element method. For instance, higher-order robustness and the possibility of handling severe mesh distortions in the solid domain seem to be areas in which there is opportunity for improvement. Our motivation here is to address theses points by extending the IFEM to the isogeometric analysis (IGA) realm. Isogeometric analysis was created to fill the gap between computer aided design (CAD) and simulation [25]. The main idea of IGA is to utilize the functions that are commonly used in CAD as shape functions in analysis. Up to now, non uniform rational B-splines (NURBS) have been the functions most widely employed in IGA. If we compare NURBS with Lagrange functions, which are typically used in the finite element method, NURBS have several advantages due to their higher continuity between elements and some of them are the following: First, spatial derivatives of second or higher order may be computed easily, which permits straightforward solution to higher-order partial differential equations [26, 27, 28, 29, 30, 31, 32, 33, 34]. Second, NURBS can handle severe mesh distortion in discretizations of solid mechanics as shown in [35]. Also, NURBS improve the accuracy on a per-degree-of-freedom basis as shown in several applications [36, 37]. IGA has been successfully applied to a large number of fields including body-fitted FSI problems [14, 38], fluid mechanics [39, 40], phase-filed models [26, 41, 42, 43, 44, 45], structural vibrations [46, 47], quasi-incompressible elasticity [48], shell modeling [49, 50], contact problems [51, 52, 53], shape optimization [54], and electromagnetics [55]. A book about IGA has been written, detailing the methodology and its applications [56].

In this paper, we propose a new method to solve immersed FSI using isogeometric analysis both for the fluid and the solid. Therefore, we take advantage of all the aforementioned benefits of NURBS. The NURBS functions are also used to transfer the information between meshes. It is important to notice that this information transfer always reduces the efficiency and the accuracy of the overall algorithm, so we have designed our algorithms to minimize the amount of information transferred between meshes.

Our method only uses the *ab initio* laws of fluid and solid mechanics. Due to the applications we are interested in, the solids are treated as incompressible, nonlinear and hyperelastic, but other material models could be considered (for example, in [21] a similar algorithm is presented where compressible solids are considered, and in [57] the authors propose an immersed method capable of dealing with rigid and deformable solids). The fluid is governed by the Navier-Stokes equations of viscous incompressible flows. For the space discretization, we use the variational multiscale (VMS) method developed by Hughes et al. [58, 59, 60]. Previous studies on the IFEM employed explicit or semi-implicit first-order accurate time integration schemes. Here, we propose a fully-implicit second-order accurate time integration algorithm based on the generalized-α method [61, 62].

The outline of this paper is as follows. In Section 2 we present the mathematical formulation of the immersed FSI problem at the continuous level. We start describing the kinematics of the fluid-solid system. Then, we focus on the strong form and the weak form of the problem. Section 3 describes the discretization process. We employ a semi-discrete formulation and make use of the VMS method which includes additional terms in comparison with classical stabilized methods [63]. The interaction between the fluid and the solid is described in detail in this section. We also provide the main ideas to implement the method. Section 4 displays several numerical examples in 2D and 3D. The first example is a free-falling cylinder in 2D and our numerical results are checked against an exact solution when it is available. The second example is a free falling sphere in 3D and the numerical results are again compared with a theoretical solution. Our results show good agreement with the theoretical data in both cases, which confirms the viability of the approach. In a third example, we consider a parallelogram submerged in a flow as a test problem to show that higher-order and higher-continuity NURBS functions allow to deal with larger deformations in the Lagrangian mesh in comparison with classical $\mathcal{C}^0$ linear piecewise functions. Section 4 finishes with a simulation of six hollow cylinders submerged in a flow in order to exhibit the capability of the method to handle problems with several immersed solids. Finally, in Section 5, some concluding remarks will be drawn.

2. Formulation of the immersed problem at the continuous level

Let $\Omega \subset \mathbb{R}^d$ be an open set, where d is the number of spatial dimensions. The spatial domain Ω encloses the entirety of our fluid-solid system. Although using standard techniques for problems on moving domains (e.g., ALE or space-time methods; see [1]), our methodology can be applied to the case in which Ω depends upon time, all the numerical examples presented in this paper are posed on a fixed domain Ω, so we will restrict our presentation to this case from the outset. In what follows, $\Omega_t^f \subset \Omega$ denotes the open subset of Ω occupied by the fluid at time t. For simplicity, we will call Ω_t^f "fluid domain". The solid domain, which is also an open subset of Ω, will be denoted Ω_t^s . The solid and fluid domains define a partition of Ω, in such a way that $\overline{\Omega} = \overline{\Omega_t^f \cup \Omega_t^s}$. Note that although both the fluid and solid domain depend on time, Ω has been assumed to be time independent, as this is the relevant case for all the numerical examples in this paper which involve free-falling objects and particulate flows. The fluid and solid domains are not allowed to overlap ($\Omega_t^f \cap \Omega_t^s = \emptyset$, where $\emptyset$ is the empty set), but they meet at the solid-fluid interface that we call Γ_t^I ($\overline{\Omega_t^f} \cap \overline{\Omega_t^s} = \Gamma_t^I$). The boundaries of Ω, Ω_t^f and Ω_t^s are denoted by Γ, Γ_t^f and Γ_t^s , respectively, and are assumed to have a well-defined unit outward normal. For future reference, we denote these normals by $\boldsymbol{n}$, $\boldsymbol{n}^f$ and $\boldsymbol{n}^s$, where the subscript t is omitted for notational simplicity. Fig. 1 shows three possible configurations. Note that the set Ω_t^s is allowed to be non-connected [see Fig. 1(c)].

2.1. Kinematics

To introduce the basic notation about motions and configurations, let us define a reference configuration Ω_0^s for our solid body. In principle, a reference configuration for the fluid domain could also be defined, but we will not make use of it. We consider the mapping $\boldsymbol{\varphi} : \Omega_0^s \times (0, T) \mapsto \mathbb{R}^d$, where $(0, T)$ is the time interval of interest[*]. We assume that $\boldsymbol{\varphi}$ is sufficiently smooth, orientation preserving and invertible [64]. Points $\boldsymbol{X}$ in Ω_0^s are called material points or particles, while points in $\mathbb{R}^d$ are denoted $\boldsymbol{x}$ and are called spatial points. To define the mapping $\boldsymbol{\varphi}$ we make use of the displacement field $\overline{\boldsymbol{u}}^s : \Omega_0^s \times (0, T) \mapsto \mathbb{R}^d$. In particular,

$$\boldsymbol{\varphi}(\boldsymbol{X}, t) = \boldsymbol{X} + \overline{\boldsymbol{u}}^s(\boldsymbol{X}, t) \tag{1}$$

[*] In a situation like that represented in Fig. 1(c), in which we have several solids immersed into the fluid, we would have to define one mapping $\boldsymbol{\varphi}$ associated to each of the solids.

For convenience, we assume that $\Omega_t^s = \boldsymbol{\varphi}(\Omega_0^s, t)$ and, thus we associate Ω_0^s with the undeformed configuration of the solid. The deformation gradient $\boldsymbol{F} : \Omega_0^s \times (0,T) \mapsto \mathbb{R}^{d\times d}$ is defined as

$$\boldsymbol{F} = \nabla_{\boldsymbol{X}} \boldsymbol{\varphi} = \boldsymbol{I} + \nabla_{\boldsymbol{X}} \overline{\boldsymbol{u}}^s \tag{2}$$

where the subscript $\boldsymbol{X}$ in $\nabla_{\boldsymbol{X}}$ emphasizes that the gradient applies to the variables $\boldsymbol{X}$. In (2), $\boldsymbol{I}$ denotes the identity tensor in $\mathbb{R}^{d\times d}$. Let us also define the Cauchy-Green deformation tensor $\boldsymbol{C} : \Omega_0^s \times (0,T) \mapsto \mathbb{R}^{d\times d}_{\text{sym}}$ as

$$\boldsymbol{C} = \boldsymbol{F}\boldsymbol{F}^T,$$

where $\mathbb{R}^{d\times d}_{\text{sym}}$ denotes the space of symmetric tensors in $\mathbb{R}^{d\times d}$, and $\boldsymbol{F}^T$ represents the transpose of $\boldsymbol{F}$. We will also make use of the Green-Lagrange strain tensor $\boldsymbol{E} : \Omega_0^s \times (0,T) \mapsto \mathbb{R}^{d\times d}_{\text{sym}}$ defined as

$$\boldsymbol{E} = \frac{1}{2}(\boldsymbol{C} - \boldsymbol{I}) \tag{3}$$

Finally, we define the velocity $\overline{\boldsymbol{v}}^s : \Omega_0^s \times (0,T) \mapsto \mathbb{R}^d$, and the acceleration $\overline{\boldsymbol{a}}^s : \Omega_0^s \times (0,T) \mapsto \mathbb{R}^d$ of a material point as

$$\begin{aligned} \overline{\boldsymbol{v}}^s(\boldsymbol{X},t) &= \frac{\partial \overline{\boldsymbol{u}}^s(\boldsymbol{X},t)}{\partial t} && (4) \\ \overline{\boldsymbol{a}}^s(\boldsymbol{X},t) &= \frac{\partial \overline{\boldsymbol{v}}^s(\boldsymbol{X},t)}{\partial t} = \frac{\partial^2 \overline{\boldsymbol{u}}^s(\boldsymbol{X},t)}{\partial t^2} && (5) \end{aligned}$$

Associated to the function $\overline{\boldsymbol{v}}^s$, we define $\boldsymbol{v}^s : \Omega_t^s \times (0,T) \mapsto \mathbb{R}^d$ as

$$\overline{\boldsymbol{v}}^s(\boldsymbol{X},t) = \boldsymbol{v}^s(\boldsymbol{\varphi}(\boldsymbol{X},t),t) \quad \text{for all} \quad \boldsymbol{X} \in \Omega_0^s,\ t \in [0,T] \tag{6}$$

Note that although $\overline{\boldsymbol{v}}^s$ and $\boldsymbol{v}^s$ represent the same physical quantity, namely the solid velocity, they are different functions, and thus, we use different notation for them. In what follows, we will call $\overline{\boldsymbol{v}}^s$ Lagrangian or material velocity and $\boldsymbol{v}^s$ Eulerian or spatial velocity. In general, we will use a bar for Lagrangian functions whenever there is possibility of confusion. Applying the chain rule to (6), we obtain

$$\overline{\boldsymbol{a}}^s(\boldsymbol{X},t) = \frac{\partial \overline{\boldsymbol{v}}^s}{\partial t}(\boldsymbol{X},t) = \frac{\partial \boldsymbol{v}^s}{\partial t}(\boldsymbol{x},t) + \boldsymbol{v}^s(\boldsymbol{x},t)\cdot\nabla_{\boldsymbol{x}}\boldsymbol{v}^s(\boldsymbol{x},t) \quad \text{if} \quad \boldsymbol{x} = \boldsymbol{\varphi}(\boldsymbol{X},t) \tag{7}$$

In what follows, we will also make use of the standard notation

$$\dot{\boldsymbol{\alpha}} = \frac{\mathrm{d}\boldsymbol{\alpha}}{\mathrm{d}t} = \frac{\partial \boldsymbol{\alpha}}{\partial t} + \boldsymbol{v}\cdot\nabla_{\boldsymbol{x}}\boldsymbol{\alpha} \tag{8}$$

for a generic Eulerian function $\boldsymbol{\alpha}$. For future reference, we also state the relation between the gradient of $\boldsymbol{\alpha}$ and the gradient of its Lagrangian counterpart $\overline{\boldsymbol{\alpha}}$, which is given by

$$\nabla_{\boldsymbol{x}}\boldsymbol{\alpha}(\boldsymbol{x},t) = \nabla_{\boldsymbol{X}}\overline{\boldsymbol{\alpha}}(\boldsymbol{X},t)\boldsymbol{F}^{-1}(\boldsymbol{X},t) \quad \text{if} \quad \boldsymbol{x} = \boldsymbol{\varphi}(\boldsymbol{X},t) \tag{9}$$

2.2. Strong form of the problem

To derive the strong form of the governing equations, we assume that both fluid and solid densities are constant, which implies incompressibility of both fluid and solid phases. As a consequence, the entire fluid-solid system is also incompressible, even if the density may be non-homogeneous in space when fluid and solid densities are different (for a thorough study of incompressible media with variable density, see [65]). Under these considerations, and assuming that gravity is the only external force acting on the system, the governing equations may be written in Eulerian coordinates as

$$\rho^f \frac{\mathrm{d}\boldsymbol{v}^f}{\mathrm{d}t} = \nabla_{\boldsymbol{x}} \cdot \boldsymbol{\sigma}^f + \rho^f \boldsymbol{g} \qquad \text{in} \quad \Omega_t^f \times (0,T) \tag{10.1}$$

$$\nabla_{\boldsymbol{x}} \cdot \boldsymbol{v}^f = 0 \qquad \text{in} \quad \Omega_t^f \times (0,T) \tag{10.2}$$

$$\rho^s \frac{\mathrm{d}\boldsymbol{v}^s}{\mathrm{d}t} = \nabla_{\boldsymbol{x}} \cdot \boldsymbol{\sigma}^s + \rho^s \boldsymbol{g} \qquad \text{in} \quad \Omega_t^s \times (0,T) \tag{10.3}$$

$$\nabla_{\boldsymbol{x}} \cdot \boldsymbol{v}^s = 0 \qquad \text{in} \quad \Omega_t^s \times (0,T) \tag{10.4}$$

$$\boldsymbol{v}^f = \boldsymbol{v}^s \qquad \text{on} \quad \Gamma_t^I \times (0,T) \tag{10.5}$$

$$\boldsymbol{\sigma}^f \boldsymbol{n}^f = -\boldsymbol{\sigma}^s \boldsymbol{n}^s \qquad \text{on} \quad \Gamma_t^I \times (0,T) \tag{10.6}$$

where ρ^f, $\boldsymbol{v}^f$ and $\boldsymbol{\sigma}^f$ denote the fluid density, velocity and Cauchy stress tensor, respectively, while ρ^s, $\boldsymbol{v}^s$ and $\boldsymbol{\sigma}^s$ denote the same quantities for the solid. Eq. (10.5) represents a kinematic constraint that equates the fluid velocity with that of the solid at the interface, while (10.6) ensures correct transmission of forces at the interface. Eqs. (10.1) and (10.3) are associated to the unknowns $\boldsymbol{v}^f$ and $\boldsymbol{v}^s$, while constraints (10.2) and (10.4) are associated to Lagrange multipliers represented by pressure fields p^f and p^s in the stress tensors $\boldsymbol{\sigma}^f$ and $\boldsymbol{\sigma}^s$. We also note that in order to get a well-posed initial/boundary-value problem, we will need to impose initial conditions on Ω_t^f and boundary conditions on $\Gamma \cap \Gamma_t^f$ for a fluid mechanics problem along with initial conditions on Ω_t^s and boundary conditions on $\Gamma \cap \Gamma_t^s$ for a solid mechanics problem.

2.2.1. Global velocity and pressure fields

The key idea of the method is to define global functions $\boldsymbol{v} : \Omega \times (0,T) \mapsto \mathbb{R}^d$, $p : \Omega \times (0,T) \mapsto \mathbb{R}$, such that

$$\boldsymbol{v} = \begin{cases} \boldsymbol{v}^f & \text{on} \quad \Omega_t^f \times (0,T) \\ \boldsymbol{v}^s & \text{on} \quad \Omega_t^s \times (0,T) \end{cases} \quad ; \quad p = \begin{cases} p^f & \text{on} \quad \Omega_t^f \times (0,T) \\ p^s & \text{on} \quad \Omega_t^s \times (0,T) \end{cases} \tag{11}$$

The function $\boldsymbol{v}$ is globally continuous due to the kinematic constraint (10.5), but the pressure field could be discontinuous in the limit case of a solid body with zero measure (see [66, 67]). Using the function $\boldsymbol{v}$, the strong form of the governing equations in Eulerian coordinates (10) may be rewritten as

$$\begin{aligned}
&\rho^f \frac{\mathrm{d}\boldsymbol{v}}{\mathrm{d}t} = \nabla_{\boldsymbol{x}} \cdot \boldsymbol{\sigma}^f + \rho^f \boldsymbol{g} + \boldsymbol{\mathcal{F}} && \text{in} \quad \Omega \times (0,T) && (12.1)\\
&\nabla_{\boldsymbol{x}} \cdot \boldsymbol{v} = 0 && \text{in} \quad \Omega \times (0,T) && (12.2)\\
&\boldsymbol{\sigma}^f \boldsymbol{n}^f = -\boldsymbol{\sigma}^s \boldsymbol{n}^s && \text{on} \quad \Gamma_t^I \times (0,T) && (12.3)
\end{aligned}$$

where $\boldsymbol{\mathcal{F}}$ will be, in principle, a discontinuous function and it is defined in such a way that Eq. (12.1) is equivalent to Eqs. (10.1) and (10.3)

$$\boldsymbol{\mathcal{F}} = \begin{cases} 0, & \boldsymbol{x} \in \Omega_t^f \\ \left(\rho^f - \rho^s\right)\left(\dot{\boldsymbol{v}} - \boldsymbol{g}\right) + \nabla_{\boldsymbol{x}} \cdot \left(\boldsymbol{\sigma}^s - \boldsymbol{\sigma}^f\right), & \boldsymbol{x} \in \Omega_t^s \end{cases} \tag{13}$$

Note that Eq. (10.5) is automatically satisfied due to the continuity of $\boldsymbol{v}$ across the fluid-solid interface, and thus, it is no longer necessary. In principle, Eq. (12.3) requires tracking the fluid-solid interface, but as we will show later, our variational formulation imposes this constraint naturally, and no algorithmical treatment will be needed.

2.2.2. Constitutive equations

To completely define Eqs. (12) and (13) we need to determine the stress tensors $\boldsymbol{\sigma}^f$ ans $\boldsymbol{\sigma}^s$. We will assume the fluid to be Newtonian and incompressible. Therefore, $\boldsymbol{\sigma}^f : \Omega \times (0,T) \mapsto \mathbb{R}^{d\times d}$

$$\boldsymbol{\sigma}^f = -p\boldsymbol{I} + 2\mu \nabla_{\boldsymbol{x}}^{\mathrm{sym}} \boldsymbol{v} \tag{14}$$

where $\mu > 0$ is the dynamic viscosity and $\nabla_{\boldsymbol{x}}^{\mathrm{sym}} \boldsymbol{v} = (\nabla_{\boldsymbol{x}} \boldsymbol{v} + \nabla_{\boldsymbol{x}} \boldsymbol{v}^T)/2$. Note that we have extended the definition of $\boldsymbol{\sigma}^f$ to the entire fluid-solid system, even if it does not have a clear physical meaning on Ω_t^s.

The Cauchy stress tensor of the solid is given by the expression

$$\boldsymbol{\sigma}^s = -p\boldsymbol{I} + \boldsymbol{F}\boldsymbol{S}\boldsymbol{F}^T/J \tag{15}$$

where $J = \det(\boldsymbol{F})$ and $\boldsymbol{S}$ is the second Piola-Kirchhoff stress tensor. We will consider hyperelastic materials, assuming the existence of a stored elastic-energy function per unit volume of the undeformed configuration ψ. Following the standard additive decomposition of ψ (see, e.g., [68]) into a volumetric part depending only on J and an isochoric part, we obtain

$$\psi(J, \boldsymbol{C}) = \psi^{\mathrm{dil}}(J) + \psi^{\mathrm{iso}}(\boldsymbol{C}) \tag{16}$$

Here, we will use the Neo-Hookean materials with dilatation penalty proposed in [68], which are defined as,

$$\psi^{\mathrm{dil}}(J) = \frac{1}{2}\kappa^s \left(\frac{1}{2}(J^2 - 1) - \ln J\right) \tag{17}$$

$$\psi^{\mathrm{iso}}(J^{-2/d}\boldsymbol{C}) = \frac{1}{2}\mu^s \left(J^{-2/d}\mathrm{tr}(\boldsymbol{C}) - d\right) \tag{18}$$

where $\mathrm{tr}(\cdot)$ denotes the trace operator, κ^s is the bulk modulus and μ^s is the shear modulus. For an incompressible material $\psi^{\mathrm{dil}}(J) = 0$, and the second stress tensor is computed as

$$\boldsymbol{S} = \frac{\partial \psi^{\mathrm{iso}}}{\partial \boldsymbol{E}} \tag{19}$$

To compute the stress tensor $\boldsymbol{S}$ we need the Lagrangian displacement of the solid $\overline{\boldsymbol{u}}^s(\boldsymbol{X}, t)$, which will be obtained from Eq. (4), repeated here for completeness

$$\frac{\partial \overline{\boldsymbol{u}}^s}{\partial t}(\boldsymbol{X}, t) = \boldsymbol{v}(\boldsymbol{x}, t) \quad \text{if} \quad \boldsymbol{x} = \boldsymbol{\varphi}(\boldsymbol{X}, t), \tag{20}$$

Eq. (20) holds for all $\boldsymbol{X} \in \Omega_0^s$ and for all $t \in [0, T]$.

2.2.3. Initial and boundary conditions

As initial conditions, we set an initial velocity for the entire fluid-solid system. Since the solid displacements are also necessary to evaluate the solid stress tensor, we need to specify also initial displacements on Ω_0^s . Thus, our initial conditions are given by

$$\begin{array}{rcll} \boldsymbol{v}(\boldsymbol{x}, 0) & = & \boldsymbol{v}_I(\boldsymbol{x}), & \boldsymbol{x} \in \Omega \qquad (21) \\ \overline{\boldsymbol{u}}^s(\boldsymbol{X}, 0) & = & 0, & \boldsymbol{X} \in \Omega_0^s \qquad (22) \end{array}$$

Note that Eq. (22) associates the reference configuration Ω_0^s with the undeformed configuration.

To define suitable boundary conditions we note that the boundary Γ can be subdivided into non-overlapping sets $\mathcal{B}^s = \Gamma \cap \Gamma_t^s$ and $\mathcal{B}^f = \Gamma \cap \Gamma_t^f$ such that $\Gamma = \overline{\mathcal{B}^s \cup \mathcal{B}^f}$ and $\emptyset = \mathcal{B}^s \cap \mathcal{B}^f$. Note, that in general $\mathcal{B}^s \neq \Gamma_t^s$, because part of Γ_t^s may not belong to Γ [see an example in Fig. 1(b)]. To define a well-posed problem we need to specify standard boundary conditions for a fluid- and solid-mechanics problem on $\mathcal{B}^s$ and $\mathcal{B}^f$, respectively. Since we have limited ourselves to the case in which Ω is fixed in time, we only consider homogeneous displacement boundary conditions on $\mathcal{B}^s$. On $\mathcal{B}^f$, however, we allow for more general boundary conditions and we actually split $\mathcal{B}^f$ as $\mathcal{B}^f = \overline{\mathcal{B}_D^f \cup \mathcal{B}_N^f}$, with $\mathcal{B}_D^f \cap \mathcal{B}_N^f = \emptyset$. On $\mathcal{B}_D^f$ we impose a given velocity $\boldsymbol{v}_B$ whereas on $\mathcal{B}_N^f$ we prescribe a traction vector $\boldsymbol{h}$. Thus, our boundary conditions are defined as

$$\boldsymbol{v} = \boldsymbol{v}_B \qquad \text{on} \quad \mathcal{B}_D^f \tag{23.1}$$

$$\boldsymbol{\sigma}^f \boldsymbol{n}^f = \boldsymbol{h} \qquad \text{on} \quad \mathcal{B}_N^f \tag{23.2}$$

$$\overline{\boldsymbol{u}}^s = 0 \qquad \text{on} \quad \mathcal{B}^s \tag{23.3}$$

2.3. Weak form

Let $\mathcal{S}_v$ and $\mathcal{S}_p$ denote trial solution spaces for velocity and pressure, defined as follows

$$\mathcal{S}_v = \left\{ \boldsymbol{v} \ \middle|\ \boldsymbol{v}(\cdot, t) \in \left(\mathcal{H}^1(\Omega)\right)^d, \ \boldsymbol{v} = \boldsymbol{v}_B \quad \text{on} \quad \mathcal{B}_D^f, \ \boldsymbol{v} = 0 \quad \text{on} \quad \mathcal{B}^s \right\} \tag{24.1}$$

$$\mathcal{S}_p = \left\{ p \ \middle|\ p(\cdot, t) \in \mathcal{L}^2(\Omega), \quad \int_\Omega p \, \mathrm{d}\Omega = 0 \quad \text{if} \quad \mathcal{B}_N^f = \emptyset \right\} \tag{24.2}$$

Here, $\mathcal{L}^2(\Omega)$ denotes the space of scalar-valued functions that are square-integrable on Ω, while $\left(\mathcal{H}^1(\Omega)\right)^d$ denotes the space of square-integrable $\mathbb{R}^d$-valued functions with square-integrable derivatives on Ω. Analogously, we define weighting function spaces $\mathcal{V}_v$, $\mathcal{V}_p$ for the momentum and continuity equations as follows

$$\mathcal{V}_v = \left\{ \boldsymbol{w} \mid \boldsymbol{w}(\cdot) \in \left(\mathcal{H}^1(\Omega)\right)^d, \ \boldsymbol{w} = 0 \quad \text{on} \quad \mathcal{B}_D^f \cup \mathcal{B}^s \right\} \tag{25.1}$$

$$\mathcal{V}_p = \left\{ q \mid q(\cdot) \in \mathcal{L}^2(\Omega) \right\} \tag{25.2}$$

To derive a weak form of our FSI problem, we begin by multiplying Eq. (12.1) with a weight function $\boldsymbol{w} \in \mathcal{V}_v$. Then, we integrate over the domain Ω, and use integration by parts to obtain

$$\left(\boldsymbol{w}, \rho^f(\dot{\boldsymbol{v}} - \boldsymbol{g})\right)_\Omega + \left(\nabla_{\boldsymbol{x}}\boldsymbol{w}, \boldsymbol{\sigma}^f\right)_\Omega - (\boldsymbol{w}, \boldsymbol{h})_{\mathcal{B}_N^f} - (\boldsymbol{w}, \boldsymbol{\mathcal{F}})_{\Omega_t^s} = 0 \tag{26}$$

where $(\cdot,\cdot)_\Omega$ denotes the $\mathcal{L}^2$ inner product over the domain Ω. To derive (26) we have used the fact that $\boldsymbol{w} = 0$ on $\mathcal{B}_D^f \cup \mathcal{B}^s$ and we have replaced $\boldsymbol{\sigma}^f \boldsymbol{n}^f$ with $\boldsymbol{h}$ on the third term of the left hand side. Let us focus now on the last term of the left hand side of (26)

$$\begin{aligned}
(\boldsymbol{w}, \boldsymbol{\mathcal{F}})_{\Omega_t^s} &= \left(\boldsymbol{w}, (\rho^f - \rho^s)(\dot{\boldsymbol{v}} - \boldsymbol{g})\right)_{\Omega_t^s} + \left(\boldsymbol{w}, \nabla_{\boldsymbol{x}} \cdot (\boldsymbol{\sigma}^s - \boldsymbol{\sigma}^f)\right)_{\Omega_t^s} \\
&= \left(\boldsymbol{w}, (\rho^f - \rho^s)(\dot{\boldsymbol{v}} - \boldsymbol{g})\right)_{\Omega_t^s} + \left(\nabla_{\boldsymbol{x}}\boldsymbol{w}, \boldsymbol{\sigma}^f - \boldsymbol{\sigma}^s\right)_{\Omega_t^s} \\
&\quad + \left(\boldsymbol{w}, (\boldsymbol{\sigma}^s - \boldsymbol{\sigma}^f)\boldsymbol{n}^s\right)_{\Gamma_t^I} + \left(\boldsymbol{w}, (\boldsymbol{\sigma}^s - \boldsymbol{\sigma}^f)\boldsymbol{n}^s\right)_{\mathcal{B}^s} \\
&= \left(\boldsymbol{w}, (\rho^f - \rho^s)(\dot{\boldsymbol{v}} - \boldsymbol{g})\right)_{\Omega_t^s} + \left(\nabla_{\boldsymbol{x}}\boldsymbol{w}, \boldsymbol{\sigma}^f - \boldsymbol{\sigma}^s\right)_{\Omega_t^s}
\end{aligned} \tag{27}$$

where we have used the fact that $\Gamma_t^s = \overline{\Gamma_t^I \cup \mathcal{B}^s}$. We have also used Eqs. (10.6), $\boldsymbol{n}^s = -\boldsymbol{n}^f$, and (25.1) to annihilate the boundary integrals. If we now pull back the integrals in Eq. (27) to the reference configuration Ω_0^s making use of the mapping $\boldsymbol{\varphi}$, and we use expression (7), we obtain

$$\begin{aligned}
(\boldsymbol{w}(\boldsymbol{x}), \boldsymbol{\mathcal{F}}(\boldsymbol{x},t))_{\Omega_t^s} &= \left(\boldsymbol{w}(\boldsymbol{\varphi}(\boldsymbol{X},t)), (\rho^f - \rho^s)\left(\frac{\partial \boldsymbol{v}}{\partial t}(\boldsymbol{\varphi}(\boldsymbol{X},t),t) - \boldsymbol{g}\right) J(\boldsymbol{X},t)\right)_{\Omega_0^s} \\
&\quad + \left(\boldsymbol{w}(\boldsymbol{\varphi}(\boldsymbol{X},t)), (\rho^f - \rho^s)\boldsymbol{v}(\boldsymbol{\varphi}(\boldsymbol{X},t),t) \cdot \nabla_{\boldsymbol{x}}\boldsymbol{v}(\boldsymbol{\varphi}(\boldsymbol{X},t),t) J(\boldsymbol{X},t)\right)_{\Omega_0^s} \\
&\quad + \left(\nabla_{\boldsymbol{x}}\boldsymbol{w}(\boldsymbol{\varphi}(\boldsymbol{X},t), \left[\boldsymbol{\sigma}^f(\boldsymbol{\varphi}(\boldsymbol{X},t),t) - \boldsymbol{\sigma}^s(\boldsymbol{\varphi}(\boldsymbol{X},t),t))\right] J(\boldsymbol{X},t)\right)_{\Omega_0^s}
\end{aligned} \tag{28}$$

Note that we may call the right hand side of (28) a mixed Lagrangian-Eulerian form because we are integrating over the reference configuration, but all derivatives are taken with respect to physical space variables. As we will see next, this is an important aspect of our algorithm and one that will permit us significant computational savings, and perhaps also increased accuracy (see Remark 2 in Section 3.1 and Section 4.1). Eq. (28) completes the derivation of the weak form of (12.1). The weak form of Eq. (12.2) can be derived by multiplying the equation with a function $q \in \mathcal{V}_p$, and integrating over Ω to obtain

$$(q, \nabla_{\boldsymbol{x}} \cdot \boldsymbol{v})_\Omega = 0 \tag{29}$$

Now, we combine Eqs. (26), (28) and (29), dividing everywhere through the fluid density and renaming the p as the pressure divided by ρ^f. We also use the classical notation $\nu = \mu/\rho^f$. Under these circumstances, our weak form may be written as

$$B^{ED}\left(\{\boldsymbol{w}, q\}, \{\boldsymbol{v}, p\}\right) - L^{ED}\left(\boldsymbol{w}\right) + B^{LD}\left(\boldsymbol{w}, \boldsymbol{v}\right) - L^{LD}\left(\boldsymbol{w}\right) = 0 \tag{30}$$

where

$$\begin{aligned} B^{ED}\left(\{\boldsymbol{w}, q\}, \{\boldsymbol{v}, p\}\right) = & \left(\boldsymbol{w}(\boldsymbol{x}), \frac{\partial \boldsymbol{v}}{\partial t}(\boldsymbol{x}, t) + \boldsymbol{v}(\boldsymbol{x}, t) \cdot \nabla_{\boldsymbol{x}} \boldsymbol{v}(\boldsymbol{x}, t)\right)_{\Omega} - \left(\nabla_{\boldsymbol{x}} \cdot \boldsymbol{w}(\boldsymbol{x}), p(\boldsymbol{x})\right)_{\Omega} \\ & + \left(\nabla_{\boldsymbol{x}}^{\text{sym}} \boldsymbol{w}(\boldsymbol{x}), 2\nu \nabla_{\boldsymbol{x}}^{\text{sym}} \boldsymbol{v}(\boldsymbol{x}, t)\right)_{\Omega} + \left(q(\boldsymbol{x}), \nabla_{\boldsymbol{x}} \cdot \boldsymbol{v}(\boldsymbol{x}, t)\right)_{\Omega} \end{aligned} \tag{31}$$

$$L^{ED}\left(\boldsymbol{w}\right) = \left(\boldsymbol{w}(\boldsymbol{x}), \boldsymbol{g}\right)_{\Omega} + \left(\boldsymbol{w}(\boldsymbol{x}), \boldsymbol{h}(\boldsymbol{x})/\rho^f\right)_{\mathcal{B}_N^f} \tag{32}$$

$$\begin{aligned} B^{LD}\left(\boldsymbol{w}, \boldsymbol{v}\right) = & -\left(\boldsymbol{w}(\boldsymbol{\varphi}(\boldsymbol{X}, t)), \left(1 - \frac{\rho^s}{\rho^f}\right) \frac{\partial \boldsymbol{v}(\boldsymbol{\varphi}(\boldsymbol{X}, t), t)}{\partial t} J(\boldsymbol{X}, t)\right)_{\Omega_0^s} \\ & - \left(\boldsymbol{w}(\boldsymbol{\varphi}(\boldsymbol{X}, t)), \left(1 - \frac{\rho^s}{\rho^f}\right) \boldsymbol{v}(\boldsymbol{\varphi}(\boldsymbol{X}, t), t) \cdot \nabla_{\boldsymbol{x}} \boldsymbol{v}(\boldsymbol{\varphi}(\boldsymbol{X}, t), t) J(\boldsymbol{X}, t)\right)_{\Omega_0^s} \\ & - \left(\nabla_{\boldsymbol{x}}^{\text{sym}} \boldsymbol{w}(\boldsymbol{\varphi}(\boldsymbol{X}, t)), 2\nu \nabla_{\boldsymbol{x}}^{\text{sym}} \boldsymbol{v}(\boldsymbol{\varphi}(\boldsymbol{X}, t), t) J(\boldsymbol{X}, t)\right)_{\Omega_0^s} \\ & + \left(\nabla_{\boldsymbol{x}}^{\text{sym}} \boldsymbol{w}(\boldsymbol{\varphi}(\boldsymbol{X}, t)), \frac{1}{\rho^f} \boldsymbol{F}(\boldsymbol{X}, t) \boldsymbol{S}(\boldsymbol{X}, t) \boldsymbol{F}^T(\boldsymbol{X}, t)\right)_{\Omega_0^s} \end{aligned} \tag{33}$$

$$L^{LD}\left(\boldsymbol{w}\right) = -\left(\boldsymbol{w}(\boldsymbol{\varphi}(\boldsymbol{X}, t)), \left(1 - \frac{\rho^s}{\rho^f}\right) \boldsymbol{g} J(\boldsymbol{X}, t)\right)_{\Omega_0^s} \tag{34}$$

Note that $J(\boldsymbol{X}, t) = 1$ due to the incompressibility constraint, but we have carried it in our derivations for generality reasons. Clearly, to evaluate the operators B^{LD} and L^{LD} we need the displacement field on the solid $\overline{\boldsymbol{u}}^s$, which will be computed using (20). In an IGA or finite-element context, perhaps the most natural approach would be to derive a weak form of (20) to eventually approximate $\overline{\boldsymbol{u}}^s$ by a member of a finite element space. This is of course a viable approach, but we find more appealing (and, very likely, more efficient) the possibility of approximating directly the strong form of (20) using an isogeometric collocation-like approach [69, 70, 71, 72]. Thus, we do not proceed further with Eq. (20) at this point.

2.4. Scale separation and variational multiscale method

We wish to propose an algorithm that is stable and accurate for arbitrary combinations of the velocity and pressure spaces, as well as for problems with low and high Reynolds numbers. Thus, we derive a variational multiscale formulation of our weak form (30), following [39]. To do so, a direct sum decomposition of the solution spaces $\mathcal{S}_v$ and $\mathcal{S}_p$ is established into coarse-scale ($\widetilde{\mathcal{S}}_v$ and $\widetilde{\mathcal{S}}_p$) and fine-scale subspaces ($\mathcal{S}'_v$ and $\mathcal{S}'_p$). Thus,

$$\mathcal{S}_v = \widetilde{\mathcal{S}}_v \oplus \mathcal{S}'_v \tag{35}$$

$$\mathcal{S}_p = \widetilde{\mathcal{S}}_p \oplus \mathcal{S}'_p \tag{36}$$

Therefore, for all $\boldsymbol{v} \in \mathcal{S}_v$ and for all $p \in \mathcal{S}_p$ there exists a unique decomposition

$$\boldsymbol{v} = \widetilde{\boldsymbol{v}} + \boldsymbol{v}', \ \widetilde{\boldsymbol{v}} \in \widetilde{\mathcal{S}}_v \ \text{ and } \ \boldsymbol{v}' \in \mathcal{S}'_v \tag{37}$$

$$p = \widetilde{p} + p', \ \widetilde{p} \in \widetilde{\mathcal{S}}_p \ \text{ and } \ p' \in \mathcal{S}'_p \tag{38}$$

This decomposition is also applied to the weighting functions $\boldsymbol{w}$ and q. If we substitute the splitting (37)–(38) into our weak form (30), and neglect the effect of the velocity fine scales on the operator B^{LD}, that is, $B^{LD}(\widetilde{\boldsymbol{w}}, \widetilde{\boldsymbol{v}} + \boldsymbol{v}') = B^{LD}(\widetilde{\boldsymbol{w}}, \widetilde{\boldsymbol{v}})$, we obtain

$$B^{ED}\left(\{\widetilde{\boldsymbol{w}}, \widetilde{q}\}, \{\widetilde{\boldsymbol{v}} + \boldsymbol{v}', \widetilde{p} + p'\}\right) - L^{ED}\left(\widetilde{\boldsymbol{w}}\right) + B^{LD}\left(\widetilde{\boldsymbol{w}}, \widetilde{\boldsymbol{v}}\right) - L^{LD}\left(\widetilde{\boldsymbol{w}}\right) = 0 \tag{39}$$

Following [39], we make the following assumptions

1. The velocity fine scales vanish on the boundary, that is, $\boldsymbol{v}' = 0$ on Γ.
2. $(\widetilde{\boldsymbol{w}}, \frac{\partial \boldsymbol{v}'}{\partial t}) = 0$. More sophisticated approaches using dynamic velocity fine scales have been devised [73], leading to enhanced accuracy. However, we decided to favor simplicity and neglected the dynamics of the velocity fine scales.
3. $(\nabla^{\text{sym}}_{\boldsymbol{x}} \widetilde{\boldsymbol{w}}, 2\nu \nabla^{\text{sym}}_{\boldsymbol{x}} \boldsymbol{v}') = 0$. Note that this term can be omitted selecting a projector that enforces the orthogonality of the coarse and fine scales in the semi-norm induced by this term as demonstrated in [60].
4. The flow is incompressible, that is, $\nabla_{\boldsymbol{x}} \cdot (\widetilde{\boldsymbol{v}} + \boldsymbol{v}') = 0$.

Under the above hypotheses, the operator B^{ED} in Eq. (39) can be expressed as

$$\begin{aligned} B^{ED}\left(\{\widetilde{\boldsymbol{w}}, \widetilde{q}\}, \{\widetilde{\boldsymbol{v}} + \boldsymbol{v}', \widetilde{p} + p'\}\right) = & \left(\widetilde{\boldsymbol{w}}, \frac{\partial \widetilde{\boldsymbol{v}}}{\partial t} + \widetilde{\boldsymbol{v}} \cdot \nabla_{\boldsymbol{x}} \widetilde{\boldsymbol{v}}\right)_\Omega \\ & - (\nabla_{\boldsymbol{x}} \cdot \widetilde{\boldsymbol{w}}, \widetilde{p})_\Omega + (\nabla^{\text{sym}}_{\boldsymbol{x}} \widetilde{\boldsymbol{w}}, 2\nu \nabla^{\text{sym}}_{\boldsymbol{x}} \widetilde{\boldsymbol{v}})_\Omega \\ & + (\widetilde{q}, \nabla_{\boldsymbol{x}} \cdot \widetilde{\boldsymbol{v}})_\Omega + (\widetilde{\boldsymbol{w}}, \boldsymbol{v}' \cdot \nabla \widetilde{\boldsymbol{v}})_\Omega \\ & - (\nabla_{\boldsymbol{x}} \widetilde{\boldsymbol{w}}, (\widetilde{\boldsymbol{v}} + \boldsymbol{v}') \otimes \boldsymbol{v}')_\Omega - (\nabla_{\boldsymbol{x}} \cdot \widetilde{\boldsymbol{w}}, p')_\Omega \\ & - (\nabla_{\boldsymbol{x}} \widetilde{q}, \boldsymbol{v}')_\Omega \end{aligned} \tag{40}$$

3. Formulation of the immersed problem at the discrete level

Here, we apply a semi-discrete formulation to the problem defined in Eqs. (39)–(40). Our space discretization is based on IGA. We also make use of an isogeometric collocation-like process to transmit information from the fluid to the solid. We discretize in time using a second-order accurate, and fully-implicit method based on generalized-α.

3.1. Space discretization

To discretize the weak form (39) we need to define conforming finite-dimensional trial solution spaces $\mathcal{S}_{\boldsymbol{v}}^h \subset \widetilde{\mathcal{S}}_{\boldsymbol{v}}$ and $\mathcal{S}_p^h \subset \widetilde{\mathcal{S}}_p$, where h is a mesh parameter. We also define finite-dimensional weighting function spaces $\mathcal{V}_{\boldsymbol{v}}^h \subset \widetilde{\mathcal{V}}_{\boldsymbol{v}}$ and $\mathcal{V}_p^h \subset \widetilde{\mathcal{V}}_p$. Following standard arguments, we will replace $\widetilde{\boldsymbol{v}}$ and $\widetilde{p}$ in Eq. (39) with $\boldsymbol{v}^h \in \mathcal{S}_{\boldsymbol{v}}^h$ and $p^h \in \mathcal{S}_p^h$. The weighting functions $\widetilde{\boldsymbol{w}}$ and $\widetilde{q}$ will also be replaced with $\boldsymbol{w}^h \in \mathcal{V}_{\boldsymbol{v}}^h$ and $q^h \in \mathcal{V}_p^h$. If we ignore boundary conditions at this stage, our discrete spaces verify $\mathcal{S}_v^h = \mathcal{V}_v^h$ and $\mathcal{S}_p^h = \mathcal{V}_p^h$. In our algorithm, all the discrete spaces are spanned by NURBS, using the concept of IGA (for an introduction to IGA, see [56] or [25]). Let us denote by $\{N_A(\boldsymbol{x})\}_{A=1}^{n_{ED}}$ a set of NURBS basis functions defined in physical space. These functions define the discrete spaces we introduced above, which have dimension n_{ED}, where the subscript ED emphasizes that these spaces are defined in Eulerian coordinates. In what follows, we will suppose that p denotes the degree of the NURBS basis functions. Assuming that Dirichlet boundary conditions will be strongly enforced on the discrete space at a later stage, we can write

$$\boldsymbol{v}^h(\boldsymbol{x},t) = \sum_{A=1}^{n_{ED}} \boldsymbol{v}_A(t) N_A(\boldsymbol{x}); \quad p^h(\boldsymbol{x},t) = \sum_{A=1}^{n_{ED}} p_A(t) N_A(\boldsymbol{x}) \tag{41}$$

$$\boldsymbol{w}^h(\boldsymbol{x}) = \sum_{A=1}^{n_{ED}} \boldsymbol{w}_A N_A(\boldsymbol{x}); \quad q^h(\boldsymbol{x}) = \sum_{A=1}^{n_{ED}} q_A N_A(\boldsymbol{x}) \tag{42}$$

Invoking the isoparametric concept, the basis functions $\{N_A(\boldsymbol{x})\}_{A=1}^{n_{ED}}$ are also used to define a computational mesh on Ω. This computational mesh is used to compute the integrals on Ω that appear on the weak form (39) and will be called Eulerian mesh. Under these considerations, the semi-discrete form of our problem may be obtained by replacing $\widetilde{\boldsymbol{w}} \leftarrow \boldsymbol{w}^h$, $\widetilde{\boldsymbol{v}} \leftarrow \boldsymbol{v}^h$, $\widetilde{p} \leftarrow p^h$, $\widetilde{q} \leftarrow q^h$, and computing $\boldsymbol{v}'$ and p' as

$$\boldsymbol{v}' = -\,\tau_M \boldsymbol{r}_M\left(\boldsymbol{v}^h, p^h\right) \tag{43.1}$$

$$p' = -\,\tau_C r_C\left(\boldsymbol{v}^h\right) \tag{43.2}$$

where τ_M and τ_C are defined as in [39], and

$$\boldsymbol{r}_M\left(\boldsymbol{v}^h, p^h\right) = \frac{\partial \boldsymbol{v}^h}{\partial t} + \boldsymbol{v}^h \cdot \nabla_{\boldsymbol{x}} \boldsymbol{v}^h + \nabla_{\boldsymbol{x}} p^h - \nu \Delta_{\boldsymbol{x}} \boldsymbol{v}^h - \boldsymbol{g} \tag{44.1}$$

$$r_C\left(\boldsymbol{v}^h\right) = \nabla_{\boldsymbol{x}} \cdot \boldsymbol{v}^h \tag{44.2}$$

Proceeding this way, we obtain

$$B_{MS}^{ED}\left(\{\boldsymbol{w}^h, q^h\}, \{\boldsymbol{v}^h, p^h\}\right) - L^{ED}\left(\boldsymbol{w}^h\right) + B^{LD}\left(\boldsymbol{w}^h, \boldsymbol{v}^h\right) - L^{LD}\left(\boldsymbol{w}^h\right) = 0 \tag{45}$$

where

$$\begin{aligned}B^{ED}_{MS}\left(\{\boldsymbol{w}^h, q^h\}, \{\boldsymbol{v}^h, p^h\}\right) = &\left(\boldsymbol{w}^h(\boldsymbol{x}), \frac{\partial \boldsymbol{v}^h(\boldsymbol{x},t)}{\partial t} + \boldsymbol{v}^h(\boldsymbol{x},t)\cdot\nabla_{\boldsymbol{x}}\boldsymbol{v}^h(\boldsymbol{x},t)\right)_\Omega \\ &- \left(\nabla_{\boldsymbol{x}}\cdot\boldsymbol{w}^h(\boldsymbol{x}), p^h(\boldsymbol{x})\right)_\Omega + \left(\nabla^{\text{sym}}_{\boldsymbol{x}}\boldsymbol{w}^h(\boldsymbol{x}), 2\nu\nabla^{\text{sym}}_{\boldsymbol{x}}\boldsymbol{v}^h(\boldsymbol{x},t)\right)_\Omega \\ &+ \left(q^h(\boldsymbol{x}), \nabla_{\boldsymbol{x}}\cdot\boldsymbol{v}^h(\boldsymbol{x},t)\right)_\Omega - \left(\boldsymbol{w}^h(\boldsymbol{x}), \tau_M\boldsymbol{r}_M(\boldsymbol{u}^h,p^h)\cdot\nabla\boldsymbol{v}^h\right)_\Omega \\ &+ \left(\nabla_{\boldsymbol{x}}\boldsymbol{w}^h(\boldsymbol{x}), (\boldsymbol{v}^h - \tau_M\boldsymbol{r}_M(\boldsymbol{v}^h,p^h))\otimes\tau_M\boldsymbol{r}_M(\boldsymbol{v}^h,p^h)\right)_\Omega \\ &+ \left(\nabla_{\boldsymbol{x}}\cdot\boldsymbol{w}^h(\boldsymbol{x}), \tau_C\boldsymbol{r}_C(\boldsymbol{u}^h)\right)_\Omega + \left(\nabla_{\boldsymbol{x}}q^h, \tau_M\boldsymbol{r}_M(\boldsymbol{u}^h,p^h)\right)_\Omega \qquad (46)\end{aligned}$$

To compute the operator B^{LD} in Eq. (45) we need a discrete Lagrangian displacement that we define next. The discrete Lagrangian displacement lives in another discrete space $\mathcal{S}^h_{\overline{\boldsymbol{u}}^s} = \text{span}\{\overline{N}_B(\boldsymbol{X})\}^{n_{LD}}_{B=1}$, which is defined on the material description. Therefore, the discrete displacement can be written as

$$\overline{\boldsymbol{u}}^h(\boldsymbol{X},t) = \sum_{B=1}^{n_{LD}} \overline{\boldsymbol{u}}_B(t)\overline{N}_B(\boldsymbol{X}) \qquad (47)$$

Note that we have dropped the superscript s on the displacement, as all variables defined in material description refer to the solid, and thus, there is no possibility of confusion. Using Eq. (47) we can define a discrete mapping $\boldsymbol{\varphi}^h(\boldsymbol{X},t)$ that will be used to push forward the solid mesh to physical space. Note that when pushed forward, the solid mesh will be located on top of the Eulerian mesh defined on Ω in an arbitrary, non-conforming fashion. All integrals on Ω^s_0 [see the weak form (45)] will be computed on the mesh defined by the $\overline{N}_B$'s. Henceforth, this mesh will be called Lagrangian mesh because it is defined in the material description[†].

Let us assume that the $\overline{N}_B$'s are defined from the tensor product of open knot vectors. Associated to those knot vectors, we define the set $\widehat{\mathcal{M}}^h$, which contains the Greville points in parametric coordinates. Greville points can be obtained by averaging knots, and a precise definition may be found in [74]. We will use the notation $\widehat{\mathcal{M}}^h = \{\widehat{\boldsymbol{\tau}}_i\}^{n_{LD}}_{i=1}$, where the $\widehat{\boldsymbol{\tau}}_i$'s are the Greville points in parametric space. For open knot vectors, $\#\widehat{\mathcal{M}}^h = \dim(\mathcal{S}^h_{\overline{\boldsymbol{u}}^s}) = n_{LD}$, where $\#\widehat{\mathcal{M}}^h$ denotes the cardinal of $\widehat{\mathcal{M}}^h$. Assuming that the initial solid geometry is mapped from the parametric space using a NURBS geometrical mapping $\boldsymbol{\phi}$, the Greville points in parametric space are mapped to the solid initial geometry as $\widetilde{\boldsymbol{\tau}}_i = \boldsymbol{\phi}(\widehat{\boldsymbol{\tau}}_i)$, defining the set $\widetilde{\mathcal{M}}^h = \{\widetilde{\boldsymbol{\tau}}_i\}^{n_{LD}}_{i=1}$. Then, at each time t, they are pushed forward to physical space using the mapping $\boldsymbol{\varphi}^h(\cdot,t)$. Let us call $\mathcal{M}^h = \{\boldsymbol{\tau}_i\}^{n_{LD}}_{i=1}$ the set of Greville points in physical space at time t, where $\boldsymbol{\tau}_i = \boldsymbol{\varphi}^h(\widetilde{\boldsymbol{\tau}}_i,t)$. Using these ideas, we compute the Lagrangian displacement from the Eulerian velocity by collocating Eq. (20) at the Greville points as,

$$\frac{\partial\overline{\boldsymbol{u}}^h}{\partial t}(\widetilde{\boldsymbol{\tau}}_i,t) = \boldsymbol{v}^h(\boldsymbol{\varphi}^h(\widetilde{\boldsymbol{\tau}}_i,t),t) \quad \text{for all} \quad \widetilde{\boldsymbol{\tau}}_i \in \widetilde{\mathcal{M}}^h \qquad (48)$$

[†] Note that we always use NURBS elements and the nomenclature Lagrangian mesh should not induce the reader to think that we are using classical Lagrangian elements.

Isogeometric collocation approaches have been recently introduced as a highly efficient alternative to Galerkin methods with higher-order convergence rates [69, 70]. Within the realm of isogeometric collocation, there are several possibilities to select the collocation points [69]. Here, we use Greville points which have shown good accuracy and stability for all cases of practical importance. Note that expression (48) represents a system of dn_{LD} equations that we use to determine the dn_{LD} control variables of the discrete displacement $\overline{\boldsymbol{u}}^h$.

Remarks:

1. Eq. (44.1) contains second derivatives of $\boldsymbol{v}^h$. If the Eulerian mesh is $\mathcal{C}^0$-continuous across element interfaces, a special procedure is needed for reconstructing second derivatives in order to avoid Dirac layers between elements [75]. However, we are able to evaluate second derivatives of $\boldsymbol{v}^h$ directly since all our Eulerian meshes are $\mathcal{C}^1$-continuous, that is, we do not need any special procedure for reconstructing second derivatives.
2. Note that our space discretization is substantially different from those employed in [21] and [76]. In our formulation, the only variable that is defined on the Lagrangian space $\mathcal{S}^h_{\overline{\boldsymbol{u}}^s}$ is the displacement field $\overline{\boldsymbol{u}}^h$. For example, the term $\boldsymbol{v}^h(\boldsymbol{\varphi}^h(\boldsymbol{X},t))$ of the operator B^{LD} is *not* projected onto $\mathcal{S}^h_{\overline{\boldsymbol{u}}^s}$, but it is directly evaluated as a function of the space $\mathcal{S}^h_v$. A similar argument applies, for example, to the term $\boldsymbol{w}^h(\boldsymbol{\varphi}^h(\boldsymbol{X},t))$. In contrast, in the above-mentioned works [21, 76], all the shape functions of the fluid Eulerian mesh need to be projected onto the solid mesh, and expressed as a linear combination of the $\overline{N}_B$'s. The computational cost of this projection is probably negligible using classical Lagrange finite elements, but represents a significant computational overhead for NURBS functions with degree $\mathsf{p} \geq 2$. In this case, the basis functions are not interpolatory and we need to solve a linear system of size n_{LD} to project each of the Eulerian basis functions onto the Lagrangian mesh. Our algorithm avoids this computational overhead altogether. For comparison purposes, we also implemented the isogeometric version of the approach proposed in [21] and [76], where all the basis functions of $\mathcal{S}^h_v$ are projected onto $\mathcal{S}^h_{\overline{\boldsymbol{u}}^s}$. For basis functions of degree $\mathsf{p} \geq 2$, our algorithm was significantly faster and, at the very least, as accurate as that based on the projection of Eulerian basis functions. Earlier works also report on the fact that, according to numerical evidence, the Lagrangian mesh needs to be at least twice as fine as the Eulerian mesh. As suggested in [76], this could be related to the projection of the fluid shape functions onto the space $\mathcal{S}^h_{\overline{\boldsymbol{u}}^s}$. In some situations, the projection on $\mathcal{S}^h_{\overline{\boldsymbol{u}}^s}$ of Eulerian shape functions with significant support on the solid is zero if the Lagrangian mesh is not fine enough. As we will show in our numerical examples, our method also circumvents this problem, and we have been able to perform very accurate computations on Lagrangian meshes which have the same element size as the Eulerian mesh. This is also related to the fact that the order of the Eulerian meshes that we utilize in our simulations is $\mathsf{p} \geq 2$ and, as a result, the support of the basis functions is $\mathsf{p}+1 \geq 3$ elements in each direction. Since the support of the basis functions is larger, the above-described situation will be probably avoided.

3. From an implementation point of view, the semi-discrete weak form (45) is computed as follows: We first loop over the elements of the Eulerian mesh defined on Ω to compute the terms B_{MS}^{ED} and L^{ED}. Then, we loop over the elements of the Lagrangian mesh defined on Ω_0^s to subtract the fluid contributions and add the terms of the solid mechanics equations. One of the crucial advantages of this implementation is that when we loop over the elements on Ω we do not need to know where the solid is located with respect to the Eulerian mesh to compute the integrals.
4. Following [76], when we loop over the Lagrangian mesh to subtract the contributions of the fluid mechanics equations computing B^{LD} the stabilization terms are not subtracted. Note also that the residuals from which we define the velocity and pressure fine scales (44) are those of the fluid mechanics equations, which correspond to the righ-hand sides of Eq. (44). The residual r_C will approach zero on the whole computational domain, including the solid domain as the whole system is incompressible. However, the residual $\boldsymbol{r}_M$ will approach zero within the fluid domain, but not within the solid domain. Therefore, we do not have a residual-based stabilization in the classical sense of this term. We do note, nevertheless, that our stabilization is consistent in the entire fluid-solid system because τ_M tends to zero as the mesh is refined. This solution is less than satisfactory for us, but computing the residuals in a fully consistent way would produce a significant computational overhead, and would require an algorithmical tracking on the solid position when we loop over the Eulerian mesh. This can be incorporated into our algorithm, but one of its most appealing features would be lost. Another way to make the stabilization completely residual based would be not to construct the stabilization terms of the fluid mechanics equations in the elements covered by the structure. This would introduce inefficiencies in the algorithm and would require to construct suitable stabilization terms in the solid to deal with the incompressibility constraint. For these reasons, we did not consider this option, either. In practical computations, the behavior of our algorithm has been very satisfactory, but we acknowledge that with this implementation it may not be possible to achieve higher-order convergence rates within the solid domain. We believe this point deserves more investigation, and we hope to address it in the future.

3.2. Time discretization

At this stage our formulation remains continuous in time. Here, we propose a monolithic and fully-implicit algorithm based on the generalized-α method. The generalized-α algorithm was first proposed by Chung and Hulbert [61] for the structural mechanics equations, and later extended by Jansen et al. [62] to first-order systems. To illustrate our method, let us divide the time interval of interest $[0, T]$ into a sequence of subintervals (t_n, t_{n+1}) with fixed time-step size $\Delta t = t_{n+1} - t_n$. To describe our algorithm, we define the following residual vectors

$$\boldsymbol{R}^M = \left\{R_{A,i}^M\right\}; \quad \boldsymbol{R}^C = \left\{R_A^C\right\}; \quad \boldsymbol{R}^P = \left\{R_{B,i}^P\right\} \tag{49}$$

Here, $A \in \{1, \ldots, n_{ED}\}$ is a control-variable index, B is an index that takes values from 1 to n_{LD}, and i is a dimension index which runs from 1 to d. The components of the residual vectors are given by

$$R^M_{A,i} = B^{ED}_{MS}\left(\{N_A \boldsymbol{e}_i, 0\}, \{\boldsymbol{v}^h, p^h\}\right) - L^{ED}\left(N_A \boldsymbol{e}_i\right) + B^{LD}\left(N_A \boldsymbol{e}_i, \boldsymbol{v}^h\right) - L^{LD}\left(N_A \boldsymbol{e}_i\right) \quad (50)$$

$$R^C_A = B^{ED}_{MS}\left(\{0, N_A\}, \{\boldsymbol{v}^h, p^h\}\right) \quad (51)$$

$$R^P_{B,i} = \boldsymbol{e}_i \cdot \left[\frac{\partial \overline{\boldsymbol{u}}^h}{\partial t}(\widetilde{\boldsymbol{\tau}}_B, t) - \boldsymbol{v}^h\left(\boldsymbol{\varphi}^h(\widetilde{\boldsymbol{\tau}}_B, t), t\right)\right] \quad (52)$$

Let us call $\boldsymbol{V}_n$, $\boldsymbol{P}_n$ and $\boldsymbol{A}_n$ the time-discrete approximation of the global vector of control variables of $\boldsymbol{v}^h(\cdot, t_n)$, $p^h(\cdot, t_n)$, and $\frac{\partial \boldsymbol{v}^h}{\partial t}(\cdot, t_n)$, respectively. Similarly, $\overline{\boldsymbol{U}}_n$ and $\overline{\boldsymbol{V}}_n$ are the time-discrete approximation of the global vector of control variables of $\overline{\boldsymbol{u}}^h(\cdot, t_n)$ and $\frac{\partial \overline{\boldsymbol{u}}^h}{\partial t}(\cdot, t_n)$. Let us also define $\overline{\boldsymbol{U}}^G_n$ as the vector that collects the time-discrete approximation to the solid displacements at the Greville points, that is, $\overline{\boldsymbol{U}}^G_n \approx \{\overline{u}^h_i(\widetilde{\boldsymbol{\tau}}_A, t_n)\}^{n_{LD}}_{A=1}$ for all $i = 1, \ldots, d$. Analogously, we define $\overline{\boldsymbol{V}}^G_n \approx \{\frac{\partial \overline{u}^h_i}{\partial t}(\widetilde{\boldsymbol{\tau}}_A, t_n)\}^{n_{LD}}_{A=1}$ for all $i = 1, \ldots, d$. Using this notation, our time-integration algorithm may be defined as follows: given $\boldsymbol{V}_n$, $\boldsymbol{A}_n$, $\overline{\boldsymbol{U}}_n$, and $\overline{\boldsymbol{V}}_n$, find $\boldsymbol{V}_{n+1}$, $\boldsymbol{A}_{n+1}$, $\boldsymbol{V}_{n+\alpha_f}$, $\boldsymbol{A}_{n+\alpha_m}$, $\boldsymbol{P}_{n+1}$, $\overline{\boldsymbol{U}}_{n+1}$, $\overline{\boldsymbol{V}}_{n+1}$, $\overline{\boldsymbol{U}}_{n+\alpha_f}$, and $\overline{\boldsymbol{V}}_{n+\alpha_m}$ such that

$$\boldsymbol{R}^M(\boldsymbol{V}_{n+\alpha_f}, \boldsymbol{A}_{n+\alpha_m}, \boldsymbol{P}_{n+1}, \overline{\boldsymbol{U}}^G_{n+\alpha_f}) = 0 \quad (53)$$

$$\boldsymbol{R}^C(\boldsymbol{V}_{n+\alpha_f}, \boldsymbol{A}_{n+\alpha_m}, \boldsymbol{P}_{n+1}) = 0 \quad (54)$$

$$\boldsymbol{R}^P(\overline{\boldsymbol{U}}^G_{n+\alpha_f}, \overline{\boldsymbol{V}}^G_{n+\alpha_m}, \boldsymbol{V}_{n+\alpha_f}) = 0 \quad (55)$$

$$\boldsymbol{V}_{n+\alpha_f} = \boldsymbol{V}_n + \alpha_f(\boldsymbol{V}_{n+1} - \boldsymbol{V}_n) \quad (56)$$

$$\boldsymbol{A}_{n+\alpha_m} = \boldsymbol{A}_n + \alpha_m(\boldsymbol{A}_{n+1} - \boldsymbol{A}_n) \quad (57)$$

$$\overline{\boldsymbol{U}}^G_{n+\alpha_f} = \overline{\boldsymbol{U}}^G_n + \alpha_f(\overline{\boldsymbol{U}}^G_{n+1} - \overline{\boldsymbol{U}}^G_n) \quad (58)$$

$$\overline{\boldsymbol{V}}^G_{n+\alpha_m} = \overline{\boldsymbol{V}}^G_n + \alpha_m(\overline{\boldsymbol{V}}^G_{n+1} - \overline{\boldsymbol{V}}^G_n) \quad (59)$$

$$\boldsymbol{V}_{n+1} = \boldsymbol{V}_n + \Delta t((1-\gamma)\boldsymbol{A}_n + \gamma \boldsymbol{A}_{n+1}) \quad (60)$$

$$\overline{\boldsymbol{U}}^G_{n+1} = \overline{\boldsymbol{U}}^G_n + \Delta t((1-\gamma)\overline{\boldsymbol{V}}^G_n + \gamma \overline{\boldsymbol{V}}^G_{n+1}) \quad (61)$$

Note that although $\boldsymbol{V}_{n+1}$ and $\boldsymbol{A}_{n+1}$ are treated separately, they are not independent as Eq. (60) must hold true. The same argument applies to $\overline{\boldsymbol{U}}^G_{n+1}$ and $\overline{\boldsymbol{V}}^G_{n+1}$, which are related through Eq. (61). In Eqs. (53)–(61), α_m, α_f and γ are real-valued parameters that control the accuracy and stability of the algorithm. Jansen et al. [62] showed that second-order accuracy can be attained by taking

$$\gamma = \frac{1}{2} + \alpha_m - \alpha_f, \quad (62)$$

while unconditional stability (for a linear problem) requires

$$\alpha_m \geq \alpha_f \geq \frac{1}{2} \quad (63)$$

A particular feature of generalized-α is that the method parameters can be expressed in terms of ϱ_∞ (the spectral radius of the amplification matrix as $\Delta t \to \infty$) in such a way that conditions (62)–(63) are automatically satisfied. This can be accomplished by taking

$$\alpha_m = \frac{1}{2}\left(\frac{3-\varrho_\infty}{1+\varrho_\infty}\right) \tag{64}$$

$$\alpha_f = \gamma = \frac{1}{1+\varrho_\infty} \tag{65}$$

with $\varrho_\infty \in [0,1]$. By using Eqs. (64)–(65), all the eigenvalues of the amplification matrix take on the value $-\varrho_\infty$ when $\Delta t \to \infty$, which suggests that ϱ_∞ can be used to control high-frequency dissipation with the guarantee that second-order accuracy and unconditional stability for a linear problem are granted. In our computations, we have taken $\varrho_\infty = 1/2$, which represents an adequate balance between accuracy and robustness in our simulations.

3.3. Implementation

Eqs. (53)–(55) constitute a nonlinear system. To solve this nonlinear system we use Newton-Raphson's algorithm which results in a two-stage predictor multicorrector method defined as follows:

Predictor stage: Set

$$\boldsymbol{V}_{n+1,(0)} = \boldsymbol{V}_n \tag{66}$$

$$\boldsymbol{A}_{n+1,(0)} = \frac{(\gamma-1)}{\gamma}\boldsymbol{A}_n \tag{67}$$

$$\boldsymbol{P}_{n+1,(0)} = \boldsymbol{P}_n \tag{68}$$

$$\overline{\boldsymbol{U}}_{n+1,(0)} = \overline{\boldsymbol{U}}_n \tag{69}$$

$$\overline{\boldsymbol{V}}_{n+1,(0)} = \frac{(\gamma-1)}{\gamma}\overline{\boldsymbol{V}}_n \tag{70}$$

where the subscript 0 on the left-hand side quantities is the Newton-Raphson iteration index. Note that the predictions are based on Eqs. (66), (68) and (69), while (67) and (70) are only a consequence of Eqs. (60) and (61).

Multicorrector stage: Repeat the following steps for $i = 0, 1, 2, ..., i_{max}$, or until convergence is achieved:

1. Evaluate the global unknowns at intermediate time levels

$$\boldsymbol{V}_{n+\alpha_f,(i)} = \boldsymbol{V}_n + \alpha_f(\boldsymbol{V}_{n+1,(i)} - \boldsymbol{V}_n) \tag{71}$$

$$\boldsymbol{A}_{n+\alpha_m,(i)} = \boldsymbol{A}_n + \alpha_m(\boldsymbol{A}_{n+1,(i)} - \boldsymbol{A}_n) \tag{72}$$

$$\boldsymbol{P}_{n+1,(i)} = \boldsymbol{P}_{n+1,(i)} \tag{73}$$

$$\overline{\boldsymbol{U}}_{n+\alpha_f,(i)} = \overline{\boldsymbol{U}}_n + \alpha_f(\overline{\boldsymbol{U}}_{n+1,(i)} - \overline{\boldsymbol{U}}_n) \tag{74}$$

$$\overline{\boldsymbol{V}}_{n+\alpha_m,(i)} = \overline{\boldsymbol{V}}_n + \alpha_m(\overline{\boldsymbol{V}}_{n+1,(i)} - \overline{\boldsymbol{V}}_n) \tag{75}$$

2. Use the intermediate time levels of the i-th Newton iteration to compute the tangent matrix $\boldsymbol{K}_{,(i)}$ and the residual vector $\boldsymbol{R}_{,(i)}$, where

$$\boldsymbol{K}_{,(i)} = \begin{pmatrix} \boldsymbol{K}_{11,(i)} & \boldsymbol{K}_{12,(i)} & \boldsymbol{K}_{13,(i)} \\ \boldsymbol{K}_{21,(i)} & \boldsymbol{K}_{22,(i)} & \boldsymbol{K}_{23,(i)} \\ \boldsymbol{K}_{31,(i)} & \boldsymbol{K}_{32,(i)} & \boldsymbol{K}_{33,(i)} \end{pmatrix}; \quad \boldsymbol{R}_{,(i)} = \left\{ \begin{array}{c} \boldsymbol{R}^{M}_{,(i)} \\ \boldsymbol{R}^{C}_{,(i)} \\ \boldsymbol{R}^{P}_{,(i)} \end{array} \right\} \tag{76}$$

Note that the residual $\boldsymbol{R}_{,(i)}$ may be linearized with respect to different variables, leading to distinct tangent matrices. Here, we linearize with respect to $\boldsymbol{A}_{n+1}$, $\boldsymbol{P}_{n+1}$ and $\overline{\boldsymbol{V}}^{G}_{n+1}$. Once we have decided to linearize with respect to $\boldsymbol{A}_{n+1}$ and $\overline{\boldsymbol{V}}^{G}_{n+1}$, the variables $\boldsymbol{V}_{n+1}$ and $\overline{\boldsymbol{U}}^{G}_{n+1}$ as well as all the α-levels become linearly dependent upon $\boldsymbol{A}_{n+1}$ and $\overline{\boldsymbol{V}}^{G}_{n+1}$ through Eqs. (56)–(61). Using this choice, the tangent matrix is given by

$$\boldsymbol{K}_{11,(i)} = \frac{\partial \boldsymbol{R}^{M}_{,(i)}}{\partial \boldsymbol{A}_{n+1,(i)}}; \quad \boldsymbol{K}_{12,(i)} = \frac{\partial \boldsymbol{R}^{M}_{,(i)}}{\partial \boldsymbol{P}_{n+1,(i)}}; \quad \boldsymbol{K}_{13,(i)} = \frac{\partial \boldsymbol{R}^{M}_{,(i)}}{\partial \overline{\boldsymbol{V}}^{G}_{n+1,(i)}}; \tag{77}$$

$$\boldsymbol{K}_{21,(i)} = \frac{\partial \boldsymbol{R}^{C}_{,(i)}}{\partial \boldsymbol{A}_{n+1,(i)}}; \quad \boldsymbol{K}_{22,(i)} = \frac{\partial \boldsymbol{R}^{C}_{,(i)}}{\partial \boldsymbol{P}_{n+1,(i)}}; \quad \boldsymbol{K}_{23,(i)} = \frac{\partial \boldsymbol{R}^{C}_{,(i)}}{\partial \overline{\boldsymbol{V}}^{G}_{n+1,(i)}}; \tag{78}$$

$$\boldsymbol{K}_{31,(i)} = \frac{\partial \boldsymbol{R}^{P}_{,(i)}}{\partial \boldsymbol{A}_{n+1,(i)}}; \quad \boldsymbol{K}_{32,(i)} = \frac{\partial \boldsymbol{R}^{P}_{,(i)}}{\partial \boldsymbol{P}_{n+1,(i)}}; \quad \boldsymbol{K}_{33,(i)} = \frac{\partial \boldsymbol{R}^{P}_{,(i)}}{\partial \overline{\boldsymbol{V}}^{G}_{n+1,(i)}}, \tag{79}$$

We note that $\boldsymbol{K}_{23}$ and $\boldsymbol{K}_{32}$ vanish. In our implementation, we neglect the contributions $\boldsymbol{K}_{13}$ and $\boldsymbol{K}_{31}$ to the tangent matrix. Under these assumptions, and considering that $\boldsymbol{K}_{23}$ and $\boldsymbol{K}_{32}$ are zero, the global linear system may be decoupled as

$$\begin{pmatrix} \boldsymbol{K}_{11,(i)} & \boldsymbol{K}_{12,(i)} \\ \boldsymbol{K}_{21,(i)} & \boldsymbol{K}_{22,(i)} \end{pmatrix} \left\{ \begin{array}{c} \Delta\boldsymbol{A}_{n+1,(i)} \\ \Delta\boldsymbol{P}_{n+1,(i)} \end{array} \right\} = - \left\{ \begin{array}{c} \boldsymbol{R}^{M}_{,(i)} \\ \boldsymbol{R}^{C}_{,(i)} \end{array} \right\} \tag{80}$$

$$\boldsymbol{K}_{33,(i)} \Delta\overline{\boldsymbol{V}}^{G}_{n+1,(i)} = -\boldsymbol{R}^{P}_{,(i)} \tag{81}$$

with the additional advantage that the unknowns of each greville point in the global vector $\Delta\overline{\boldsymbol{V}}^{G}_{n+1,(i)}$ are decoupled in such a way that (81) is solved as n_{LD} independent systems of equations of size $d \times d$. We remark that the matrix $\boldsymbol{K}_{33,(i)}$ has this special shape because we are linearizing with respect to the displacements at Greville points $\overline{\boldsymbol{U}}^{G}_{n+1}$. If we were to linearize with respect to their associated control variables $\overline{\boldsymbol{U}}_{n+1}$, the dn_{LD} equations would be coupled, leading to a more intensive computation.
The submatrices $\boldsymbol{K}_{lm}$ for $l, m \in \{1, 2, 3\}$ may be computed using the chain rule. For

example,

$$\begin{aligned}\boldsymbol{K}_{11,(i)} &= \frac{\partial \boldsymbol{R}^M_{,(i)}}{\partial \boldsymbol{A}_{n+1,(i)}} \\ &= \frac{\partial \boldsymbol{R}^M_{,(i)}}{\partial \boldsymbol{V}_{n+\alpha_f,(i)}} \frac{\partial \boldsymbol{V}_{n+\alpha_f,(i)}}{\partial \boldsymbol{V}_{n+1,(i)}} \frac{\partial \boldsymbol{V}_{n+1,(i)}}{\partial \boldsymbol{A}_{n+1,(i)}} + \frac{\partial \boldsymbol{R}^M_{,(i)}}{\partial \boldsymbol{A}_{n+\alpha_m,(i)}} \frac{\partial \boldsymbol{A}_{n+\alpha_m,(i)}}{\partial \boldsymbol{A}_{n+1,(i)}} \\ &= \alpha_f \gamma \Delta t \frac{\partial \boldsymbol{R}^M_{,(i)}}{\partial \boldsymbol{V}_{n+\alpha_f,(i)}} + \alpha_m \frac{\partial \boldsymbol{R}^M_{,(i)}}{\partial \boldsymbol{A}_{n+\alpha_m,(i)}}. \end{aligned} \tag{82}$$

Finally, we note that the global linear system (80) has the same size as that of a fully-implicit incompressible Navier-Stokes formulation and is solved up to a given tolerance using the GMRES method [77] with incomplete *LU* factorization [78, 79, 80].

3. Update the Newton-Rapshon iterates as follows

$$\boldsymbol{V}_{n+1,(i+1)} = \boldsymbol{V}_{n+1,(i)} + \gamma \Delta t \Delta \boldsymbol{A}_{n+1,(i)} \tag{83}$$

$$\boldsymbol{A}_{n+1,(i+1)} = \boldsymbol{A}_{n+1,(i)} + \Delta \boldsymbol{A}_{n+1,(i)} \tag{84}$$

$$\boldsymbol{P}_{n+1,(i+1)} = \boldsymbol{P}_{n+1,(i)} + \Delta \boldsymbol{P}_{n+1,(i)} \tag{85}$$

$$\overline{\boldsymbol{U}}^G_{n+1,(i+1)} = \overline{\boldsymbol{U}}^G_{n+1,(i)} + \gamma \Delta t \Delta \overline{\boldsymbol{V}}^G_{n+1,(i)} \tag{86}$$

$$\overline{\boldsymbol{V}}^G_{n+1,(i+1)} = \overline{\boldsymbol{V}}^G_{n+1,(i)} + \Delta \overline{\boldsymbol{V}}^G_{n+1,(i)} \tag{87}$$

Note that the updates of $\boldsymbol{A}_{n+1}$, $\boldsymbol{P}_{n+1}$ and $\overline{\boldsymbol{V}}^G_{n+1}$ are straightforward while $\boldsymbol{V}_{n+1}$ and $\overline{\boldsymbol{U}}^G_{n+1}$ are updated consistently with Eqs. (60)–(61) that define the generalized-α method.

4. From the i-th Newton iterate of the time-discrete Greville values of the solid displacement $\overline{\boldsymbol{U}}^G_{n+1,(i)}$, compute their corresponding control variables $\overline{\boldsymbol{U}}_{n+1,(i)}$. This may be done independently for each of the spatial directions. Therefore, let us define $\overline{\boldsymbol{\mathcal{U}}}^G_{n+1,(i)}$ as the entries of $\overline{\boldsymbol{U}}^G_{n+1,(i)}$ that correspond to the first spatial direction. Similarly, we define $\overline{\boldsymbol{\mathcal{U}}}_{n+1,(i)}$ as the restriction of $\overline{\boldsymbol{U}}_{n+1,(i)}$ to the first spatial dimension. Then, we can compute $\overline{\boldsymbol{\mathcal{U}}}_{n+1,(i)}$ by solving the linear system

$$\boldsymbol{M} \overline{\boldsymbol{\mathcal{U}}}_{n+1,(i)} = \overline{\boldsymbol{\mathcal{U}}}^G_{n+1,(i)} \tag{88}$$

where the entries of the matrix $\boldsymbol{M}$ are given by

$$\boldsymbol{M} = \{M_{AB}\};\ M_{AB} = \overline{N}_B(\widetilde{\boldsymbol{\tau}}_A);\ A, B \in \{1, \ldots, n_{LD}\} \tag{89}$$

Note that the matrix $\boldsymbol{M}$ depends neither on time nor on the spatial direction. Thus, $\boldsymbol{M}$ can be pre-computed and pre-factorized at the beginning of the simulation. Whenever we need to solve system (88), we only perform the corresponding backward and forward substitutions. The process defined by Eqs. (88)–(89) is repeated for the remaining spatial directions. This completes one nonlinear iteration.

Remark:

> We acknowledge that our strategy of using a non-consistent tangent matrix (see Step 2 of the Multicorrector stage) may limit the size of our time steps. However, computing the consistent tangent matrix produces a significant computational overhead, so we chose not to do so. Other authors have reported on the use of matrix-free Newton-Krylov methods [21]. In our implementation, utilizing the tangent matrix (80)–(81) vastly outperformed the use of matrix-free techniques. Using matrix-free methods, the computational cost of a time step was higher and we could not take time steps as large as those employed with the tangent matrix (80)–(81). In the future, we will investigate in more detail the possibility of using the consistent tangent matrix.

4. Numerical Examples

In this section we present four numerical examples computed using the discretization introduced in Sections 3.1 and 3.2. The code used to perform these simulations has been developed on top of the PetIGA framework [81, 82, 83], which adds NURBS discretization capabilities and integration of forms to the scientific library PETSc [84, 85].

4.1. Free falling cylinder

An object falling in a fluid will try to adjust its velocity to a terminal or settling velocity that we call v_T. The terminal velocity of a falling object is reached when the sum of the drag force (F_D), which increases with velocity, and buoyancy (F_b) equals the downward force of gravity (F_g) acting on the object. Since the net force on the object is then zero, the object moves with constant velocity. That is, to obtain the exact solution for the settling velocity of a rigid cylinder with infinite length and radius a which is released to fall in a channel of width $2L$, we impose

$$F_D + F_b = F_g \tag{90}$$

where

$$F_b = g\rho^f \pi a^2 \tag{91}$$

$$F_g = g\rho^s \pi a^2 \tag{92}$$

Here, g is the Euclidean norm of $\boldsymbol{g}$. We assume that for this problem the relevant Reynolds number is based on the terminal velocity and the cylinder diameter, that is,

$$R_e = \frac{2\rho^f v_T a}{\mu} \tag{93}$$

According to [86], under the assumption of creeping flow ($R_e << 1$), and undeformable solid, F_D can be expressed as

$$F_D = \frac{4\pi\mu v_T}{\ln\left(\frac{L}{a}\right) - 0.9157 + 1.7244\left(\frac{a}{L}\right)^2 - 1.7302\left(\frac{a}{L}\right)^4}, \tag{94}$$

which using Eqs. (90)–(92) leads to the terminal velocity

$$v_T = \frac{\left(\rho^s - \rho^f\right) g a^2}{4\mu} \left[\ln\left(\frac{L}{a}\right) - 0.9157 + 1.7244 \left(\frac{a}{L}\right)^2 - 1.7302 \left(\frac{a}{L}\right)^4 \right] \tag{95}$$

In this numerical example, we will validate our numerical results against the exact solution (95). The infinite length of the cylinder allows us to neglect the dynamics in the direction of the axis of the cylinder and therefore we set up a two-dimensional domain, which has to be long enough in the direction of gravity such that the cylinder can reach its terminal velocity. The solid has to be sufficiently stiff in order to complete the whole simulation without experiencing perceptible deformations. We have used a Neo-Hookean model with dilatational penalty in 2D, whose stored energy function can be obtained by setting $d = 2$ in Eqs. (16)–(18). Taking the derivative of the stored energy function we get the second Piola-Kirchhoff stress tensor

$$\boldsymbol{S} = \mu^s J^{-2/d} \left(\boldsymbol{I} - \frac{1}{d} \mathrm{tr}(\boldsymbol{C}) \boldsymbol{C}^{-1} \right) + \frac{1}{2} \kappa^s \left(J^2 - 1\right) \boldsymbol{C}^{-1} \tag{96}$$

where $d = 2$. Note that following [20], for this example, and also for the remaining examples, we have included also ψ^{dil} into the stored energy functional to derive $\boldsymbol{S}$. Consistently with what was reported in [20], we found that disregarding the dilatational part of the stored energy function produced non-negligible errors in the incompressibility constraint. Thus, following [20] we used relatively small values of κ^s for our computations (on the order of magnitude of $\mu^s/10$). Within this range, we did not find the numerical solution to be sensitive to the value of κ^s. The remaining physical parameters are taken as follows: gravity acceleration $g = 981$ cm/s^2, dynamic viscosity $\mu = 5$ dyn/cm^2, fluid density $\rho^f = 1$ g/cm^3, solid density $\rho^s = 1.25$ g/cm^3 and shear modulus $\mu^s = 33550$ g/(cm s^2). The dimensions of the computational domain are $2L \times 3L$ with $L = 2$ cm and the radius of the cylinder is $a = 0.25$ cm. Introducing these parameters in Eq. (95), we obtain $v_{\mathrm{T}} = 0.912$ cm/s and the Reynolds number is $R_e = 0.0912 \ll 1$.

The geometry of the problem, boundary conditions and Lagrangian mesh are represented in Fig. 2. NURBS shape functions, unlike Lagrange shape functions, are able to represent the axial section of a cylinder exactly when NURBS of second order or higher are used. Note that one face of each of the elements adjacent to the cylinder axis has been degenerated by placing multiple control points at the same location, and thus many parameter values map to the same point in physical space. Such a mapping is not invertible, but it is still analysis-suitable since the Gauss quadrature points never fall on the singularity itself.

Both the Lagrangian mesh and the Eulerian mesh are comprised of quadratic NURBS elements. The shape functions of the Eulerian mesh are at least $\mathcal{C}^1$-continuous everywhere. The shape functions of the Lagrangian mesh are $\mathcal{C}^0$-continuous (in the circumferential direction) along two diametral lines, and at least $\mathcal{C}^1$-continuous everywhere else. All the simulations are performed with the same Lagrangian mesh which has 11 elements in the radial direction and 48 elements in the circumferential direction. The time step used in the simulations is $\Delta t = 10^{-3}$ s. We have computed the average solid velocity in the direction of gravity for simulations with different Eulerian meshes. The points chosen to calculate that average are the images of the Greville points. As shown in Fig. 3, the more we refine the Eulerian mesh, the closer we get to the analytical solution. The relative difference between the analytical and numerical terminal velocity is 5.7% with 100×150 elements, 2.1% with 150×225 elements, and 0.5% with 200×300 elements.

Fig. 4 shows the vortices and the recirculation that appear close to the free falling cylinder. In Fig. 5, we plot the pressure subtracting its hydrostatic component along with the Lagrangian mesh and the finest Eulerian mesh used in our computations. It is noticeable that the elements of the Lagrangian mesh are even bigger than the elements of the Eulerian mesh. Despite this fact, the simulation yields very accurate results as it can be seen in Fig 3. Based on previously reported numerical results, the possibility of using a Lagrangian mesh coarser than the Eulerian mesh seems to be a unique feature of of NURBS functions with $\mathsf{p} \geq 2$ that is not attained with classical Lagrange elements.

In Fig. 6, we consider several fluid viscosities while the rest of parameters remain fixed. We then compare the numerical result with the analytical solution for each case. In Fig. 7, we repeat the same procedure, but this time we vary the solid density while keeping the other parameters fixed. In both cases, the difference between the numerical result and the analytical solution increases as the Reynolds number increases. This comes as no surprise, since the analytical solution is deduced under the assumption of creeping flow and as a result it looses its accuracy for predicting the correct terminal velocity when the Reynolds number gets close to one or is even higher. This is precisely what Figs. 6 and 7 show.

Finally, for comparison purposes, we also computed this example by using an alternative algorithm that projects the shape functions of the Eulerian mesh onto the Lagrangian mesh using a collocation-like process for each shape function (see Remark 2 in Section 3.1). The idea is to project the Eulerian shape functions using a procedure similar to that used for the displacement field. Note that the projection of a shape function, though, is a linear process, unlike the projection of the displacement, which is nonlinear. We have also tried projecting the Eulerian velocity and its time derivative and using these projected quantities when looping over the elements of the Lagrangian mesh. For that, we have made use of Eq. (9). The variation of the results when using these projected quantities is very subtle (data not shown) and the algorithm becomes slower.

4.2. Free falling sphere

The second example is a 3D computation of a sphere of radius a falling in a fluid. The terminal velocity of the simulation is going to be compared with an analytical solution in order to validate the method in a three-dimensional setup. As it has been mentioned in the previous section, we impose Eq. (90). Although this time the drag force, buoyancy and the downward force of gravity acting on the object are given by

$$F_D = \frac{1}{2}\pi a^2 C_D \rho^f v_T^2 \tag{97}$$

$$F_b = \frac{4}{3}\pi a^3 g \rho^f \tag{98}$$

$$F_g = \frac{4}{3}\pi a^3 g \rho^s \tag{99}$$

If we knew the value of the drag coefficient C_D, we could calculate the terminal velocity of the free falling sphere. Theoretical solutions under the assumption of creeping flow ($R_e \ll 1$) show that the drag coefficient of a sphere falling in an unbounded domain takes the value

$$C_D = \frac{24}{R_e} \tag{100}$$

where R_e is the same as in Eq. (93), but this time a refers to the radius of the sphere. This expression for the drag coefficient yields the terminal velocity

$$v_{Stokes} = \frac{2g}{9\mu} a^2 (\rho^s - \rho^f) \tag{101}$$

We have renamed v_T to v_{Stokes} not only because this solution was derived by Stokes [87] but to emphasize that it is valid only under the limit of $R_e \ll 1$ and in an unbounded medium. Since we cannot perform a computation on an infinite domain, we need to correct Eq. (101) to account for the presence of rigid walls. The effects caused by the rigid walls involve an increase in the viscous dissipation which decreases the speed of the sphere. In particular, the correction factor due to the wall-attachment effects K for creeping flow of a Newtonian fluid is defined by

$$K = \frac{v_{Stokes}}{v_T} \tag{102}$$

It describes the decrease in the speed resulting from the presence of the walls as a function of the ratio of the radius of the falling sphere and a cylinder which contains the surrounding fluid. Being a the radius of the sphere and A the radius of the cylinder, Bohlin [88] obtains for $R_e \ll 1$ and $a/A < 0.6$,

$$K = \left[1 - 2.10443\left(\frac{a}{A}\right) + 2.08877\left(\frac{a}{A}\right)^3 + \dots\right]^{-1} \tag{103}$$

Experimental works [89] also obtain expressions which yield very similar results under the assumptions stated above. We can now compare the theoretical result with our numerical result. As in the previous section, the solid has to be sufficiently stiff in order to complete the whole simulation without experiencing perceptible deformations and the length of the computational domain in the gravity direction needs to be long enough such that the sphere can reach its terminal velocity. The material used for the sphere is the same as that used for the cylinder in the previous section, but $d = 3$ in this case. The physical parameters of this example are the following: gravity acceleration $g = 981$ cm/s^2, dynamic viscosity $\mu = 10$ dyn/cm^2, fluid density $\rho^f = 1$ g/cm^3, solid density $\rho^s = 1.5$ g/cm^3 and shear modulus $\mu^s = 33550$ g/(cm s^2). The computational domain is a cylinder of 4 cm of height and $A = 2$ cm of radius. The radius of the sphere is $a = 0.25$ cm. The geometry of the problem, boundary conditions and Lagrangian mesh are represented in Fig. 8.

The Eulerian mesh has $60 \times 60 \times 67$ quadratic elements and it is nonuniform, as the mesh is finer through the path of the sphere. The geometry was generated by extruding a circle in the vertical (z) direction. To define the circle, we used a map that transforms each edge of the parameter space into a quarter of the circumference (details on the parameterization of the circle can be found, for example, in [90]). This makes the mapping singular at four points on the boundary, which produces four singular lines when the circle is extruded in vertical direction. Everywhere else, the basis functions are at least $\mathcal{C}^1$ continuous. The Lagrangian mesh has $9 \times 20 \times 40$ quadratic elements, the continuity between elements is $\mathcal{C}^1$ with the exception of three circles in which the continuity is $\mathcal{C}^0$ in an angular direction (this choice for the representation of a sphere has been used previously in FSI simulations [1]). NURBS shape functions, unlike Lagrange shape functions, are able to represent a sphere and a cylinder exactly when NURBS of second order or higher are used. The time step used in the simulation is $\Delta t = 10^{-3}$ s.

We have computed the average velocity in the gravity direction of the solid during the simulation. Again, the points chosen to calculate that average have been the images of the Greville points. As it can be seen in Fig. 9, the terminal average velocity of the simulation is very similar to the analytical result. In particular, the relative difference between the numerical and analytical terminal velocity is 3.2% . In Fig. 10(a), we plot the magnitude of the velocity, the streamlines and the Lagrangian mesh at time $t = 0.1$ s and in Fig. 10(b), we plot the velocity in z direction in two perpendicular slices at time $t = 0.1$ s together with the Eulerian mesh.

4.3. Parallelogram submerged in a flow

In this case we consider a parallelogram embedded in a tube with Poiseuille-like flow. The same material model as in Section 4.1 is used, although a very low shear modulus is established so that the solid is very soft. All simulations are performed using the following physical parameters: gravity acceleration $g = 981$ cm/s^2, dynamic viscosity $\mu = 0.01$ dyn/ cm^2, fluid density $\rho^f = 1$ g/cm^3 and solid density $\rho^s = 1.5$ g/cm^3.

The problem setup is shown in Fig. 11. We assume that for this problem the relevant Reynolds number is based on the inflow velocity (78 cm/s) and the side of the tube perpendicular to the flow (25 cm); its value is $R_e = 195000$. The initial position of the solid is such that its long side is perpendicular to the flow. The velocity gradient along the long side of the parallelogram will cause a rotation of the solid. During that process, it will experiment large deformations due to its low stiffness. As it will be shown, these large deformations can distort the Lagrangian mesh up to a point where the simulation fails. In order to establish some criterion, we will consider the simulations to be successful if the solid is able to rotate 90 degrees.

The time step used in the simulations is $\Delta t = 10^{-3}$ s. We will work with a fixed number of control variables in all the simulations of this section. In particular, 234 control variables. First, we take $\mu^s = 400$g/(cm s^2). If we use a Lagrangian mesh with order $\mathsf{p} = 1$ and $\mathcal{C}^0$ continuity between elements, it collapses when the parallelogram has rotated approximately 45 degrees. Nevertheless, a Lagrangian mesh with order $\mathsf{p} = 2$ and $\mathcal{C}^1$ continuity between elements is able to rotate 90 degrees as shown in Fig. 12. We then decrease the shear modulus to $\mu^s = 300\text{g}/(\text{cm}\,\text{s}^2)$. The simulation fails for the two previous meshes with this new value of the shear modulus, but a Lagrangian mesh with order $\mathsf{p} = 3$ and $\mathcal{C}^2$ continuity between elements is able to complete the simulation without any problems as shown in Fig. 13. We are therefore able to deal with larger deformations as we increase the order of the mesh and the continuity between elements for the same number of degrees of freedom. The same fact was shown for isogeometric structural analysis in [35]. These results suggest that NURBS with higher order and higher continuity may increase the robustness of immersed FSI methods.

4.4. Hollow cylinders immersed in a flow

The fourth and final example consists of six hollow cylinders immersed in a water flow. The cylinders are filled with water and the problem setup is shown in Fig. 14. The inner radius of the cylinders is $r_{in} = 1.35$ cm, while the exterior radius is $r_{ex} = 1.5$ cm. The dimensions of the channel are 45×25 cm. The flow is driven by a parabolic inflow velocity with a maximum of 50 cm/s, the lateral boundaries are assigned zero velocity, and a zero-traction boundary condition is applied at the outflow. The initial condition for the velocity is a paraboloid with a maximum of 50 cm/s. We use the same solid material model as in Section 4.1. The shear modulus is $\mu^s = 335$ g/(cm s^2), which produces highly deformable solids. We take $\rho^s = 1.5$ g/cm^3. The fluid is water with dynamic viscosity $\mu = 0.01$ dyn/cm^2 and density $\rho^f = 1.0$ g/cm^3. Defining the Reynolds number as in Section 4.3, we have $R_e = 125000$.

In this case, we use relatively thin structures for which we use solid elements with two $\mathcal{C}^1$ quadratic elements, that is, four $\mathcal{C}^1$ quadratic basis functions in the through-thickness direction. This way of modeling shell-like structures with solid NURBS elements was introduced in [25]. The Eulerian mesh has 210×175 quadratic elements and each Lagrangian mesh has 2×140 quadratic elements. The time step used in the simulation is $\Delta t = 10^{-3}$ s.

In Fig. 5 we plot the velocity magnitude and the pressure at different time steps where can see how the presence of the solids modifies the flow conditions. This example shows how the presence of thin solids can produce large pressure gradients that need to be resolved by the Eulerian mesh. Again, the ability of higher-order NURBS to approximate rough functions may increase the overall robustness of the algorithm.

5. Conclusions

We propose an immersed fluid-structure interaction method which uses NURBS-based isogeometric analysis for both the viscous incompressible fluid and the nonlinear hyperelastic solid. The high inter-element continuity of NURBS allows us to deal with larger deformations in the Lagrangian mesh, which increases the robustness of the overall algorithm. We are able to run very accurate simulations with the same element size in both meshes or even with bigger element size in the Lagrangian mesh, while works using standard $\mathcal{C}^0$ finite elements consistently reported the need for a Lagrangian mesh at least twice as fine as the Eulerian mesh. The information transfer between the Eulerian mesh and the Lagrangian mesh is kept to a minimum, and it is carried out by a collocation-like process that also exploits the high continuity of NURBS. We propose a fully-implicit, second-order accurate time integration algorithm based on the generalized-α method, which leads to a monolithic formulation. We validated our computational framework using theoretical data in two and three dimensions.

Acknowledgements

HG, HC and CB-C were partially supported by the European Research Council through the FP7 Ideas Starting Grant (project # 307201). HG was also partially supported by Xunta de Galicia, and by Ministerio de Economía y Competitividad (contract # DPI2013-44406-R). The last two grants are cofinanced with FEDER funds. This support is gratefully acknowledged. We acknowledge the open source scientific libraries PETSc, PetIGA, and their developers.

References

[1] Y. Bazilevs, K. Takizawa, T. E. Tezduyar, Computational fluid-structure interaction: methods and applications, John Wiley & Sons, 2012.

[2] Y. Zhang, Y. Bazilevs, S. Goswami, C. Bajaj, T. J. R. Hughes, Patient-specific vascular NURBS modeling for isogeometric analysis of blood flow, Computer Methods in Applied Mechanics and Engineering 196 (2007) 2943–2959.

[3] Y. Bazilevs, M.-C. Hsu, Y. Zhang, W. Wang, T. Kvamsdal, S. Hentschel, J. Isaksen, Computational vascular fluid-structure interaction: Methodology and application to cerebral aneurysms, Biomechanics and Modeling in Mechanobiology 9 (2010) 481–498.

[4] K. Takizawa, T. E. Tezduyar, Computational methods for parachute fluid-structure interactions, Archives of Computational Methods in Engineering 19 (2012) 125–169.

[5] K. Takizawa, B. Henicke, T. E. Tezduyar, M.-C. Hsu, Y. Bazilevs, Stabilized space-time computation of wind-turbine rotor aerodynamics, Computational Mechanics 48 (2011) 333–344.

[6] W. K. Liu, Y. Liu, D. Farrell, L. Zhang, X. Wang, Y. Fukui, N. Patankar, Y. Zhang, C. Bajaj, J. Lee, J. Hong, X. Chen, H. Hsu, Immersed finite element method and its applications to biological systems, Computer Methods in Applied Mechanics and Engineering 195 (2006) 1722–1749.

[7] C. A. Felippa, K. Park, C. Farhat, Partitioned analysis of coupled mechanical systems, Computer methods in applied mechanics and engineering 190 (2001) 3247–3270.

[8] U. Küttler, W. A. Wall, Fixed-point fluid–structure interaction solvers with dynamic relaxation, Computational Mechanics 43 (2008) 61–72.

[9] E. H. Brummelen, P. Geuzaine, Fundamentals of fluid–structure interaction, Wiley Online Library, 2010.

[10] S. Badia, A. Quaini, A. Quarteroni, Splitting methods based on algebraic factorization for fluid-structure interaction, SIAM Journal on Scientific Computing 30 (2008) 1778–1805.

[11] J. Donéa, P. Fasoli-Stella, S. Giuliani, Lagrangian and eulerian finite element techniques for transient fluid-structure interaction problems, Transactions of the 4th International Conference on Structural Mechanics in Reactor Technology Volume B: Thermal and Fluid/Structure Dynamics Analysis B(1/2) (1977) 1–12.

[12] T. Belytschko, J. M. Kennedy, Computer models for subassembly simulation, Nuclear Engineering and Design 49 (1978) 17–38.

[13] T. J. R. Hughes, W. K. Liu, T. K. Zimmermann, Lagrangian-eulerian finite element formulation for incompressible viscous flows, Computer methods in applied mechanics and engineering 29 (1981) 329–349.

[14] Y. Bazilevs, V. M. Calo, T. J. R. Hughes, Y. Zhang, Isogeometric fluid-structure interaction: Theory, algorithms, and computations, Computational Mechanics 43 (2008) 3–37.

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

[16] D. McQueen, C. Peskin, Computer-assisted design of butterfly bileaflet valves for the mitral position, Scandinavian Journal of Thoracic and Cardiovascular Surgery 19 (1985) 139–148.

[17] R. Beyer Jr., A computational model of the cochlea using the immersed boundary method, Journal of Computational Physics 98 (1992) 145–162.

[18] R. Dillon, L. Fauci, A. Fogelson, D. Gaver III, Modeling biofilm processes using the immersed boundary method, Journal of Computational Physics 129 (1996) 57–73.

[19] L. Fauci, C. Peskin, A computational model of aquatic animal locomotion, Journal of Computational Physics 77 (1988) 85–108.

[20] L. Zhang, A. Gerstenberger, X. Wang, W. K. Liu, Immersed finite element method, Computer Methods in Applied Mechanics and Engineering 193 (2004) 2051–2067.

[21] W. K. Liu, D. Kim, S. Tang, Mathematical foundations of the immersed finite element method, Computational Mechanics 39 (2007) 211–222.

[22] M. Gay, L. Zhang, W. K. Liu, Stent modeling using immersed finite element method, Computer Methods in Applied Mechanics and Engineering 195 (2006) 4358–4370.

[23] Y. Liu, L. Zhang, X. Wang, W. K. Liu, Coupling of navier-stokes equations with protein molecular dynamics and its application to hemodynamics, International Journal for Numerical Methods in Fluids 46 (2004) 1237–1252.

[24] Y. Liu, W. K. Liu, Rheology of red blood cell aggregation by computer simulation, Journal of Computational Physics 220 (2006) 139–154.

[25] T. J. R. Hughes, J. A. Cottrell, Y. Bazilevs, Isogeometric analysis CAD, finite elements, NURBS, exact geometry and mesh refinement, Computacional Methods in Applied Mechanics and Engineering 194 (2005) 4135–4195.

[26] H. Gomez, V. M. Calo, Y. Bazilevs, T. J. R. Hughes, Isogeometric analysis of the Cahn-Hilliard phase-field model, Computacional Methods in Applied Mechanics and Engineering 197 (2008) 4333–4352.

[27] H. Gomez, T. J. Hughes, X. Nogueira, V. M. Calo, Isogeometric analysis of the isothermal navier–stokes–korteweg equations, Computer Methods in Applied Mechanics and Engineering 199 (2010) 1828–1840.

[28] H. Gomez, J. París, Numerical simulation of asymptotic states of the damped kuramoto-sivashinsky equation, Physical Review E 83 (2011) 046702.

[29] H. Gomez, X. Nogueira, A new space–time discretization for the swift–hohenberg equation that strictly respects the lyapunov functional, Communications in Nonlinear Science and Numerical Simulation 17 (2012) 4930–4946.

[30] H. Gomez, X. Nogueira, An unconditionally energy-stable method for the phase field crystal equation, Computer Methods in Applied Mechanics and Engineering 249 (2012) 52–61.

[31] U. Thiele, A. J. Archer, M. J. Robbins, H. Gomez, E. Knobloch, Localized states in the conserved swift-hohenberg equation with cubic nonlinearity, Physical Review E 87 (2013) 042915.

[32] H. Gomez, L. Cueto-Felgueroso, R. Juanes, Three-dimensional simulation of unstable gravity-driven infiltration of water into a porous medium, Journal of Computational Physics 238 (2013) 217–239.

[33] L. Dedè, M. J. Borden, T. J. Hughes, Isogeometric analysis for topology optimization with a phase field model, Archives of Computational Methods in Engineering 19 (2012) 427–465.

[34] J. Kiendl, K.-U. Bletzinger, J. Linhard, R. Wüchner, Isogeometric shell analysis with kirchhoff–love elements, Computer Methods in Applied Mechanics and Engineering 198 (2009) 3902–3914.

[35] S. Lipton, J. Evans, Y. Bazilevs, T. Elguedj, T. J. R. Hughes, Robustness of isogeometric structural discretizations under severe mesh distortion, Computer Methods in Applied Mechanics and Engineering 199 (2010) 357–373.

[36] I. Akkerman, Y. Bazilevs, V. M. Calo, T. J. R. Hughes, S. Hulshoff, The role of continuity in residual-based variational multiscale modeling of turbulence, Computational Mechanics 41 (2008) 371–378.

[37] J. A. Evans, Y. Bazilevs, I. Babuška, T. J. Hughes, n-widths, sup-infs, and optimality ratios for the k-version of the isogeometric finite element method, Computer Methods in Applied Mechanics and Engineering 198 (2009) 1726–1741.

[38] Y. Bazilevs, V. M. Calo, Y. Zhang, T. J. R. Hughes, Isogeometric fluid-structure interaction analysis with applications to arterial blood flow, Computational Mechanics 38 (2006) 310–322.

[39] Y. Bazilevs, V. M. Calo, J. A. Cottrell., T. J. R. Hughes, A. Reali, G. Scovazzi, Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows, Computer Methods in Applied Mechanics and Engineering 197 (2007) 173–201.

[40] J. A. Evans, T. J. Hughes, Isogeometric divergence-conforming b-splines for the unsteady navier–stokes equations, Journal of Computational Physics 241 (2013) 141–167.

[41] H. Gomez, T. J. R. Hughes, Provably unconditionally stable, second-order time-accurate, mixed variational methods for phase-field models, Journal of Computational Physics 230 (2011) 5310–5327.

[42] R. Dhote, H. Gomez, R. Melnik, J. Zu, Isogeometric analysis of a dynamic thermo-mechanical phase-field model applied to shape memory alloys, Computational Mechanics (2013) 1–16.

[43] G. Vilanova, I. Colominas, H. Gomez, Capillary networks in tumor angiogenesis: From discrete endothelial cells to phase-field averaged descriptions via isogeometric analysis, International Journal for Numerical Methods in Biomedical Engineering 29 (2013) 1015–1037.

[44] G. Vilanova, I. Colominas, H. Gomez, Coupling of discrete random walks and continuous modeling for three-dimensional tumor-induced angiogenesis, Computational Mechanics (2013) 1–16.

[45] J. Liu, H. Gomez, J. A. Evans, T. J. Hughes, C. M. Landis, Functional entropy variables: A new methodology for deriving thermodynamically consistent algorithms for complex fluids, with particular reference to the isothermal navier–stokes–korteweg equations, Journal of Computational Physics 248 (2013) 47–86.

[46] J. A. Cottrell, A. Reali, Y. Bazilevs, T. J. R. Hughes, Isogeometric analysis of structural vibrations, Computer Methods in Applied Mechanics and Engineering 195 (2006) 5257–5296.

[47] T. J. Hughes, A. Reali, G. Sangalli, Duality and unified analysis of discrete approximations in structural dynamics and wave propagation: Comparison of p-method finite elements with k-method nurbs, Computer methods in applied mechanics and engineering 197 (49) (2008) 4104–4124.

[48] T. Elguedj, Y. Bazilevs, V. M. Calo, T. J. Hughes, $\overline{B}$ and $\overline{F}$ projection methods for nearly incompressible linear and non-linear elasticity and plasticity using higher-order nurbs elements, Computer methods in applied mechanics and engineering 197 (2008) 2732–2762.

[49] D. Benson, Y. Bazilevs, M. Hsu, T. J. R. Hughes, Isogeometric shell analysis: The reissner-mindlin shell, Computer Methods in Applied Mechanics and Engineering 199 (2010) 276–289.

[50] J. Kiendl, Y. Bazilevs, M.-C. Hsu, R. Wüchner, K.-U. Bletzinger, The bending strip method for isogeometric analysis of kirchhoff–love shell structures comprised of multiple patches, Computer Methods in Applied Mechanics and Engineering 199 (2010) 2403–2416.

[51] L. De Lorenzis, I. Temizer, P. Wriggers, G. Zavarise, A large deformation frictional contact formulation using NURBS-based isogeometric analysis, International Journal for Numerical Methods in Engineering 87 (2011) 1278–1300.

[52] R. Dimitri, L. De Lorenzis, M. Scott, P. Wriggers, R. Taylor, G. Zavarise, Isogeometric large deformation frictionless contact using t-splines, Computer Methods in Applied Mechanics and Engineering 269 (2014) 394–414.

[53] R. Dimitri, L. De Lorenzis, P. Wriggers, G. Zavarise, Nurbs-and t-spline-based isogeometric cohesive zone modeling of interface debonding, Computational Mechanics (2014) 1–20.

[54] W. Wall, M. Frenzel, C. Cyron, Isogeometric structural shape optimization, Computer Methods in Applied Mechanics and Engineering 197 (2008) 2976–2988.

[55] A. Buffa, G. Sangalli, R. Vázquez, Isogeometric analysis in electromagnetics: B-splines approximation, Computer Methods in Applied Mechanics and Engineering 199 (2010) 1143–1152.

[56] J. A. Cottrell, T. J. R. Hughes, Y. Bazilevs, Isogeometric Analysis Toward Integration of CAD and FEA, Wiley, 2009.

[57] C. Hesch, A. Gil, A. Arranz Carreño, J. Bonet, P. Betsch, A mortar approach for fluid–structure interaction problems: Immersed strategies for deformable and rigid bodies, Computer Methods in Applied Mechanics and Engineering.

[58] T. J. Hughes, Multiscale phenomena: Green's functions, the dirichlet-to-neumann formulation, subgrid scale models, bubbles and the origins of stabilized methods, Computer methods in applied mechanics and engineering 127 (1995) 387–401.

[59] T. J. R. Hughes, G. Feijóo, L. Mazzei, J.-B. Quincy, The variational multiscale method - a paradigm for computational mechanics, Computer Methods in Applied Mechanics and Engineering 166 (1998) 3–24.

[60] T. J. R. Hughes, G. Sangalli, Variational multiscale analysis: The fine-scale green's function, projection, optimization, localization, and stabilized methods, SIAM Journal on Numerical Analysis 45 (2007) 539–557.

[61] J. Chung, G. Hulbert, A time integration algorithm for structural dynamics with improved numerical dissipation: The generalized- method., Journal of Applied Mechanics 60 (1993) 371–375.

[62] K. Jansen, C. Whiting, G. Hulbert, Generalized-α method for integrating the filtered navier-stokes equations with a stabilized finite element method, Computer Methods in Applied Mechanics and Engineering 190 (2000) 305–319.

[63] A. N. Brooks, T. J. Hughes, Streamline upwind/petrov-galerkin formulations for convection dominated flows with particular emphasis on the incompressible navier-stokes equations, Computer methods in applied mechanics and engineering 32 (1982) 199–259.

[64] J. E. Marsden, T. J. Hughes, Mathematical foundations of elasticity, Courier Dover Publications, 1994.

[65] P.-L. Lions, Mathematical topics in fluid mechanics. Vol. 1. Incompressible models, Oxford Science, 1996.

[66] M.-C. Hsu, D. Kamensky, Y. Bazilevs, M. Sacks, T. Hughes, Fluid-structure interaction analysis of bioprosthetic heart valves: significance of arterial wall deformation, Computational Mechanics 54 (2014) 1055–1071.

[67] D. Kamensky, M.-C. Hsu, D. Schillinger, J. A. Evans, A. Aggarwal, Y. Bazilevs, M. S. Sacks, T. J. Hughes, A variational immersed boundary framework for fluid–structure interaction: Isogeometric implementation and application to bioprosthetic heart valves, Computer Methods in Applied Mechanics and Engineering, (2014) in review. Also available as ICES Report 14-12.

[68] J. C. Simo, T. J. R. Hughes, Computational inelasticity, Springer, 2008.

[69] F. Auricchio, L. Da Veiga, T. J. R. Hughes., A. Reali, G. Sangalli, Isogeometric collocation methods, Mathematical Models and Methods in Applied Sciences 20 (2010) 2075–2107.

[70] D. Schillinger, J. Evans, A. Reali, M. Scott, T. J. R. Hughes, Isogeometric collocation: Cost comparison with galerkin methods and extension to adaptive hierarchical nurbs discretizations, Computer Methods in Applied Mechanics and Engineering 267 (2013) 170–232.

[71] H. Gomez, A. Reali, G. Sangalli, Accurate, efficient, and (iso) geometrically flexible collocation methods for phase-field models, Journal of Computational Physics.

[72] H. Gomez, A. Reali, An isogeometric collocation approach for bernoulli-euler beams and krichhoff plates, To appear in Computer Methods in Applied Mechanics and Engineering.

[73] R. Codina, J. Principe, O. Guasch, S. Badia, Time dependent subscales in the stabilized finite element approximation of incompressible flow problems, Computer Methods in Applied Mechanics and Engineering 196 (2007) 2413–2430.

[74] C. De Boor, A practical guide to splines, Vol. 27, Springer-Verlag New York, 1978.

[75] K. E. Jansen, S. S. Collis, C. Whiting, F. Shaki, A better consistency for low-order stabilized finite element methods, Computer Methods in Applied Mechanics and Engineering 174 (1999) 153–170.

[76] C. Hesch, A. Gil, A. Arranz Carreo, J. Bonet, On continuum immersed strategies for fluid-structure interaction, Computer Methods in Applied Mechanics and Engineering 247-248 (2012) 51–64.

[77] Y. Saad, M. H. Schultz, Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM Journal on scientific and statistical computing 7 (1986) 856–869.

[78] T. Dupont, R. P. Kendall, H. Rachford, Jr, An approximate factorization procedure for solving self-adjoint elliptic difference equations, SIAM Journal on Numerical Analysis 5 (1968) 559–573.

[79] T. A. Oliphant, An implicit, numerical method for solving two-dimensional time-dependent diffusion problems, Quarterly of Applied Mathematics 19 (1961) 221–229.

[80] T. F. Chan, H. A. Van der Vorst, Approximate and incomplete factorizations, Springer, 1997.

[81] N. Collier, L. Dalcin, V. M. Calo, Petiga: High-performance isogeometric analysis, arXiv preprint:1305.4452.

[82] M. Bernal, V. M. Calo, N. Collier, G. Espinosa, F. Fuentes, J. Mahecha, Isogeometric analysis of hyperelastic materials using petiga, Procedia Computer Science 18 (2013) 1604–1613.

[83] P. A. Vignal, N. Collier, V. M. Calo, Phase field modeling using petiga, Procedia Computer Science 18 (2013) 1614–1623.

[84] S. Balay, M. F. Adams, J. Brown, P. Brune, K. Buschelman, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, K. Rupp, B. F. Smith, H. Zhang, PETSc Web page, `http://www.mcs.anl.gov/petsc` (2014).

[85] S. Balay, M. F. Adams, J. Brown, P. Brune, K. Buschelman, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, K. Rupp, B. F. Smith, H. Zhang, PETSc users manual, Tech. Rep. ANL-95/11 - Revision 3.4, Argonne National Laboratory (2013).

[86] J. Happel, H. Brenner, Low Reynolds number hydrodynamics - with special applications to particulate media., Springer, 1983.

[87] G. Stokes, On the effect of internal friction of fluids on the motion of pendulums, Trans. Cambridge Phil Soc. 9 (1851) 6106.

[88] X. Bohlin, On the drag on a rigid sphere moving in a viscous fluid inside a cylindrical tube, Trans Roy. Inst. Teck. 155 (1960) 1–63.

[89] A. Francis, Wall effect in falling ball method for viscosity, Physics 4 (1933) 403–406.

[90] A. Vuong, C. Heinrich, B. Simeon, Isogat: A 2d tutorial matlab code for isogeometric analysis, Computer Aided Geometric Design 27 (2010) 644–655.

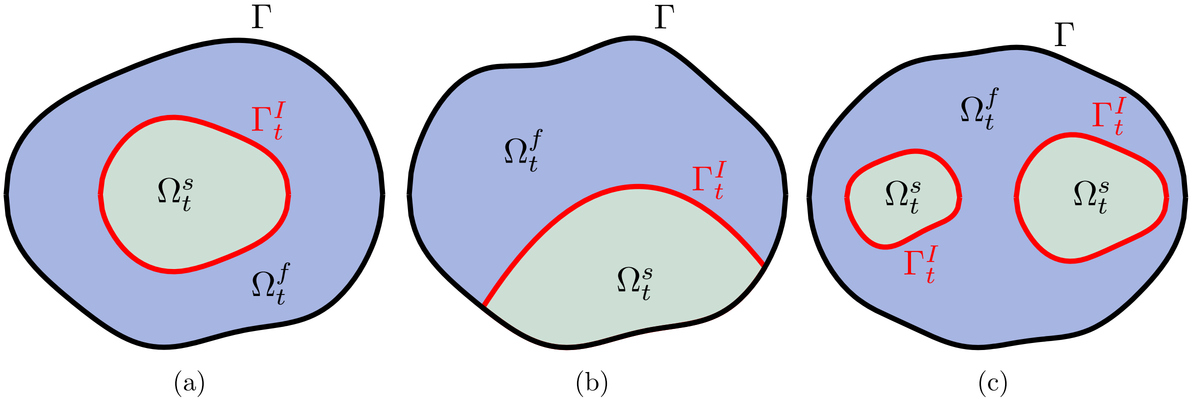


Figure 1: (Color online) Three possible configurations of the solid domain Ω_t^s (green), the fluid domain Ω_t^f (blue), and the fluid-solid interface Γ_t^I (red). Note that the solid domain is allowed to be non-connected [see snapshot (c)]. The domain Ω is implicitly defined as the set enclosed by Γ.

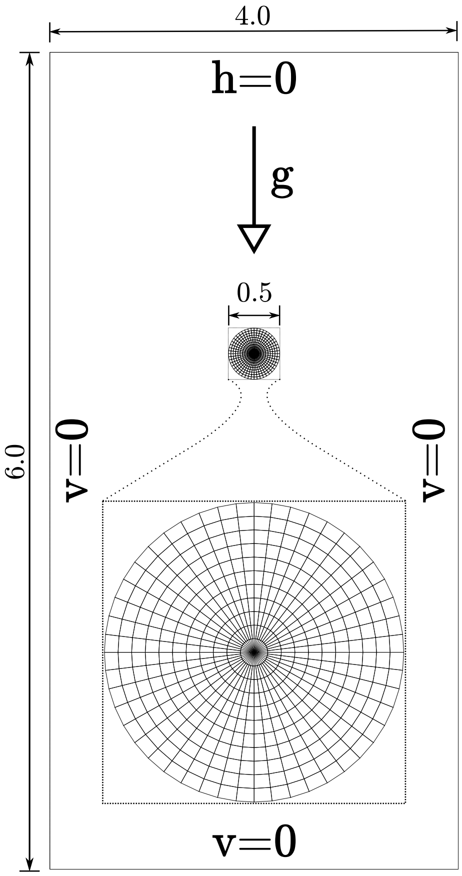


Figure 2: Geometry, boundary conditions, and Lagrangian mesh for the free falling cylinder example. Note that we impose non-slip boundary conditions on the lateral boundaries as this is the case for which the exact solution applies. The inset shows a zoom in of the Lagrangian mesh. The lengths are in centimeters.

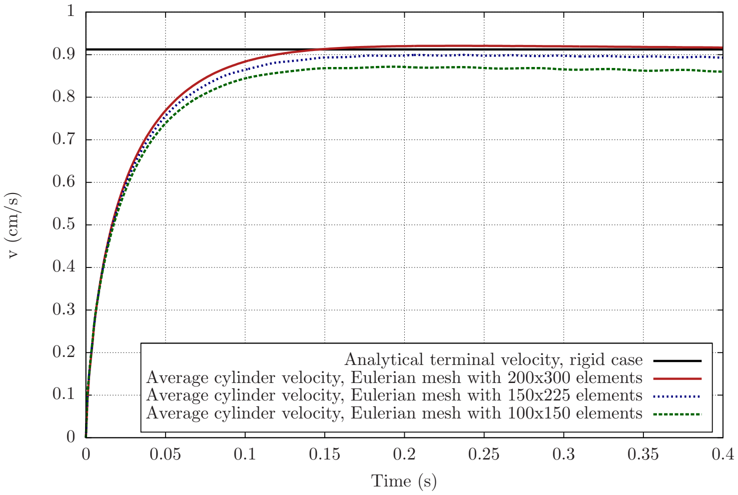


Figure 3: (Color online) Exact terminal velocity (black line) and time evolution of the average cylinder velocity obtained from the computations (red, blue and green lines). All computations were performed using a Lagrangian mesh composed of 11 × 48 elements (knot spans). The Eulerian mesh was refined to show convergence of the numerical solution to the exact solution. In all cases, we used quadratic basis functions.

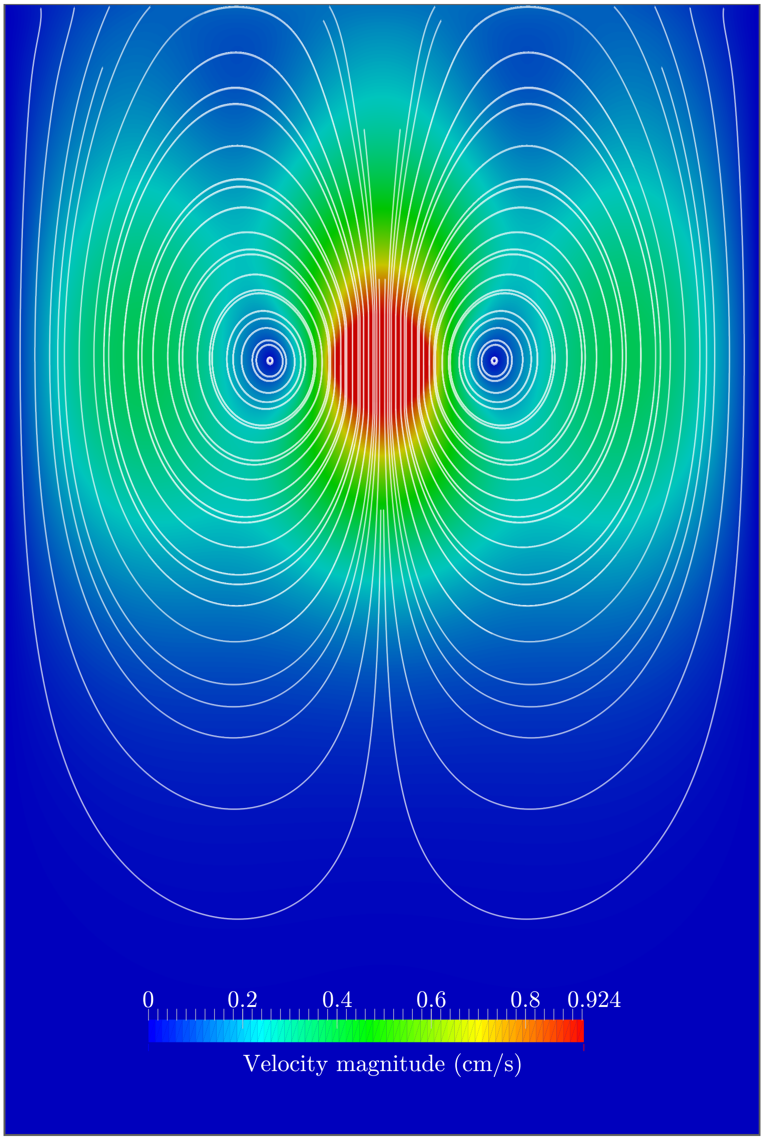


Figure 4: (Color online) Velocity magnitude along with the streamlines at time $t = 0.35$ s for the free-falling cylinder example. The plot clearly shows two large vortices close to the cylinder, which is the expected laminar-flow solution. The Eulerian mesh is composed of 200×300 elements and the Lagrangian mesh has 11×48 elements. In all cases, we used quadratic basis functions. At the scale of the plot, the solution looks symmetric as expected.

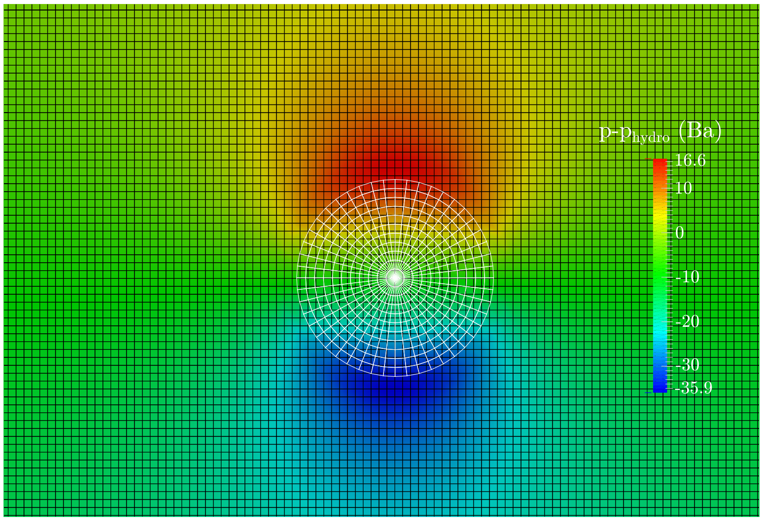


Figure 5: (Color online) Contour plot of the pressure (subtracting the hydrostatic component) along with the spatial discretizations. The Eulerian mesh is composed of 200 × 300 elements and the Lagrangian mesh has 11 × 48 elements. In all cases, we used quadratic basis functions. Note that the elements of the Lagrangian mesh (white lines) are even bigger than the elements of the Eulerian mesh (black lines), and, yet, the solution is very accurate as shown in Fig. 3.

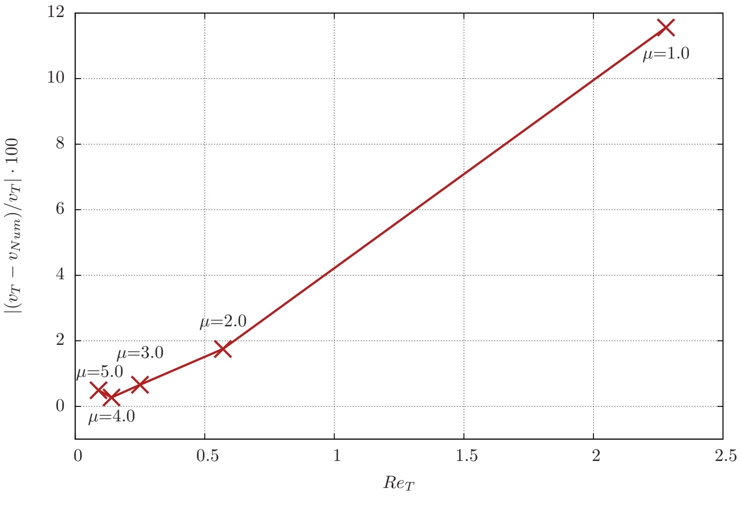


Figure 6: Relative difference between the analytical and numerical terminal velocities for several values of the fluid viscosity. The analytical solution is only acceptable when $R_e \ll 1$ since it is a solution for creeping flow. When $R_e \ll 1$, the analytical and numerical solutions show very good agreement. As the Reynolds number gets close to one or higher, the difference between the two solutions increases as it must happen. The Eulerian mesh has 200 × 300 elements and the Lagrangian mesh has 11 × 48 elements. We used quadratic basis functions.

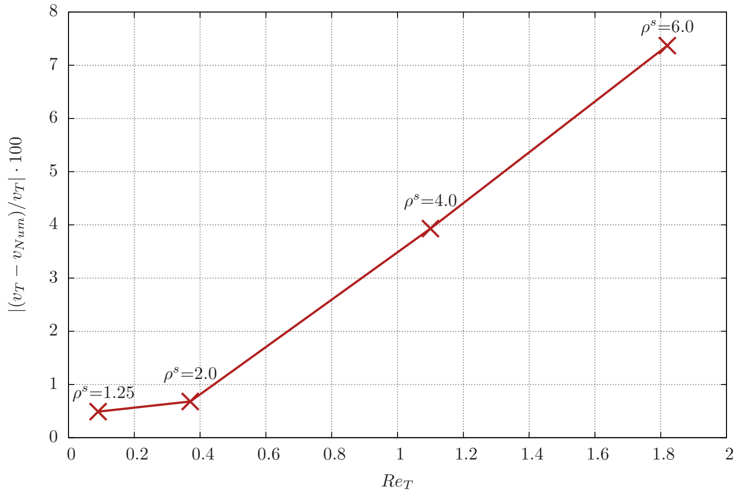


Figure 7: Relative error of the numerical terminal velocity for several values of the solid density. The exact solution is only acceptable when $R_e \ll 1$ since it is a solution for creeping flow. When $R_e \ll 1$, the exact and numerical solutions show good agreement. As the Reynolds number gets close to one or higher, the difference between the two solutions increases because the theoretical solution becomes inaccurate. The Eulerian mesh is composed of 200×300 elements and the Lagrangian mesh has 11×48 elements. We used quadratic basis functions.

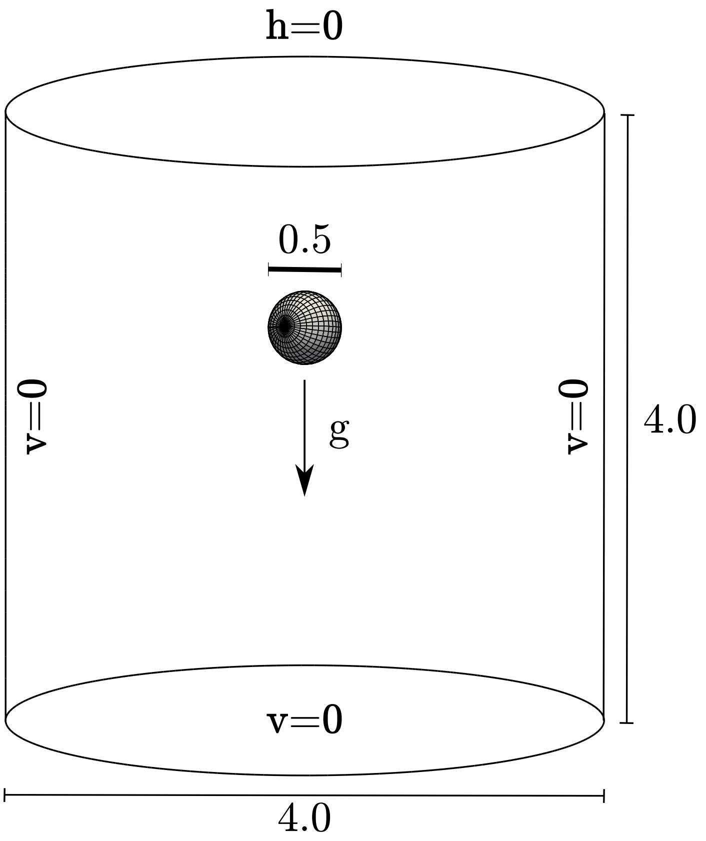


Figure 8: Geometry, boundary conditions, and Lagrangian mesh for the free falling sphere. The lengths are in centimeters.

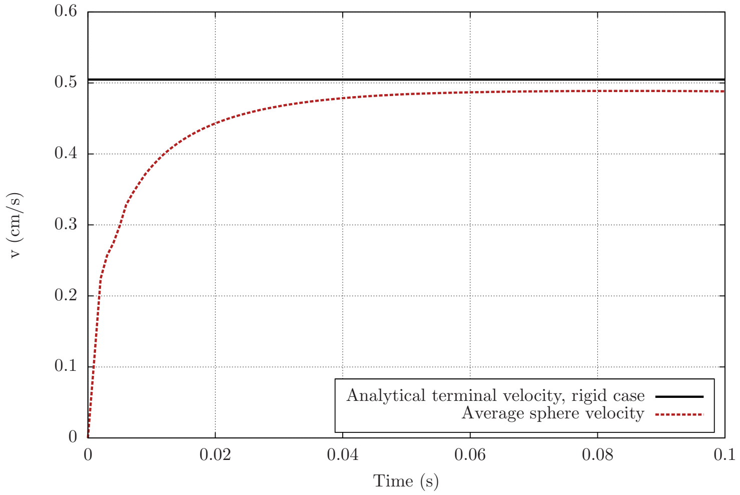
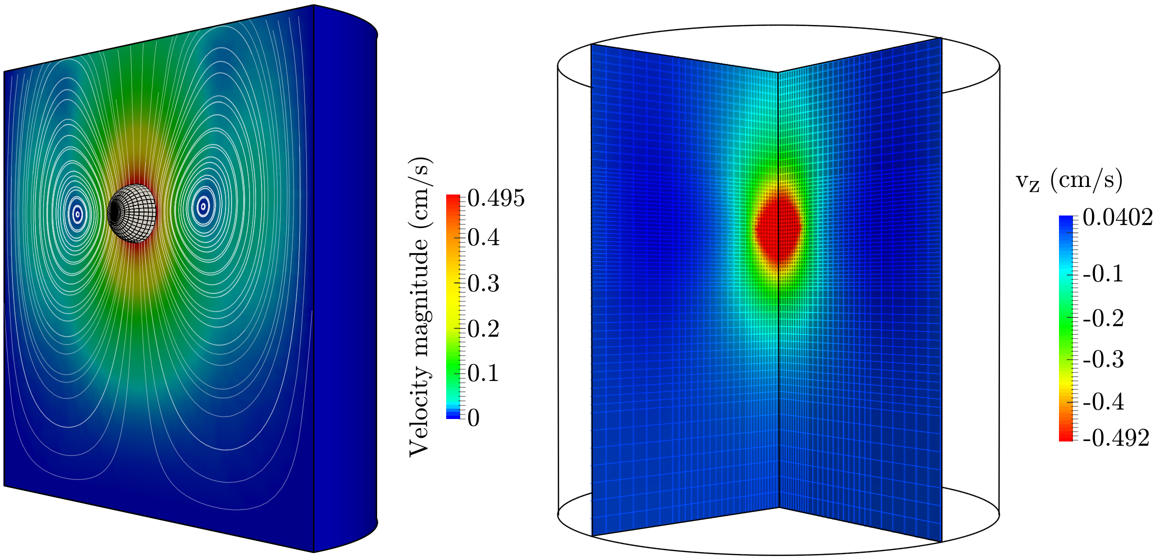


Figure 9: (Color online) Analytical terminal velocity (black line) and time evolution of the average sphere velocity obtained from the computations. The Eulerian mesh is composed of 60 × 60 × 67 elements and it is nonuniform, having higher spatial resolution around the sphere's trajectory [see Fig. 10(b)]. The Lagrangian mesh has 9 × 20 × 40 elements. We used quadratic basis functions.



(a) Velocity magnitude, $t = 0.1$ s.

(b) Vertical velocity, $t = 0.1$ s.

Figure 10: (Color online) Contour plot of the velocity magnitude, streamlines, and Lagrangian mesh at time $t = 0.1$ s (a). Contour plot of the velocity in z (vertical) direction in two cut planes, along with the Eulerian mesh. On the Eulerian mesh, the size of the elements is smaller close to the sphere's trajectory (b).

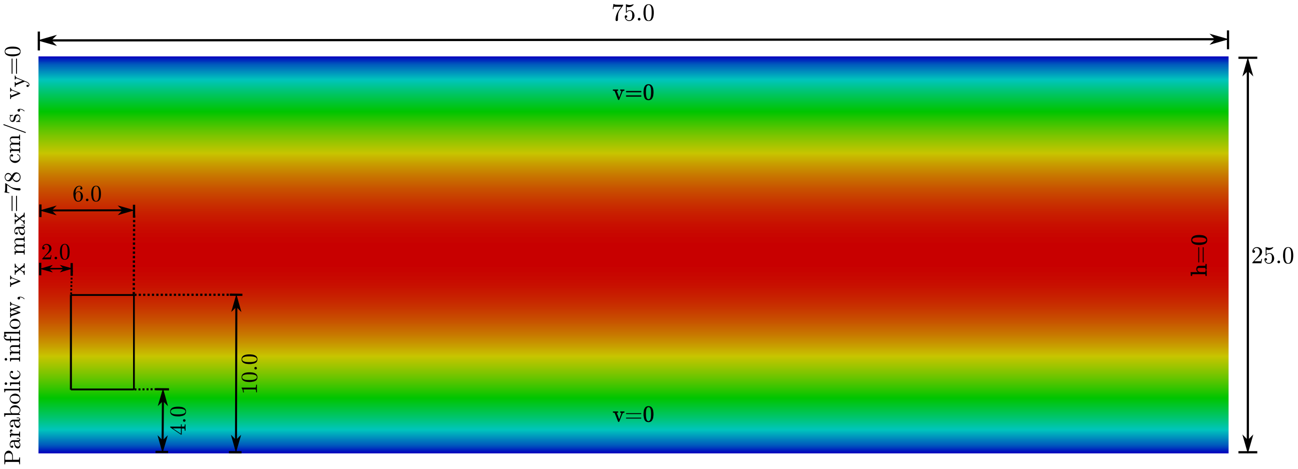
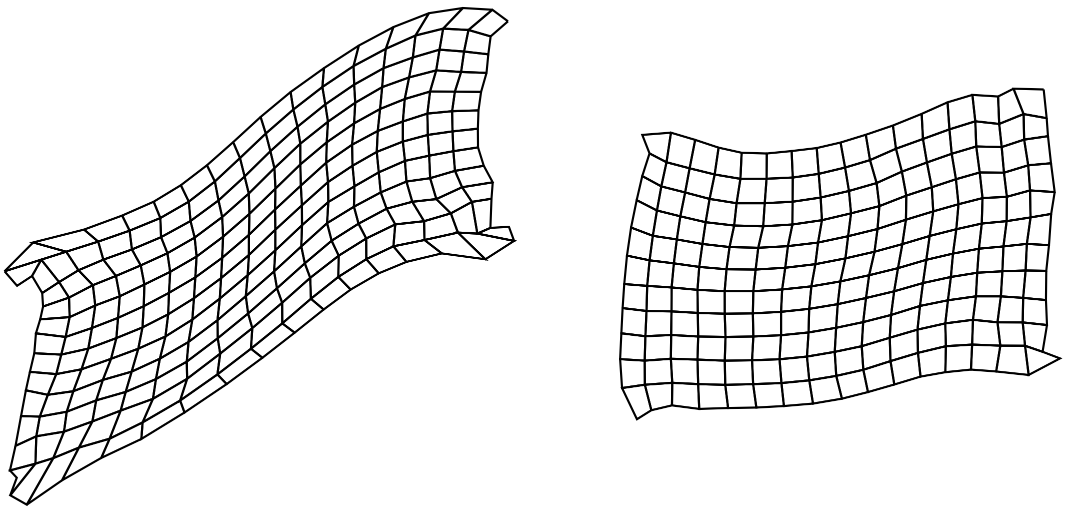


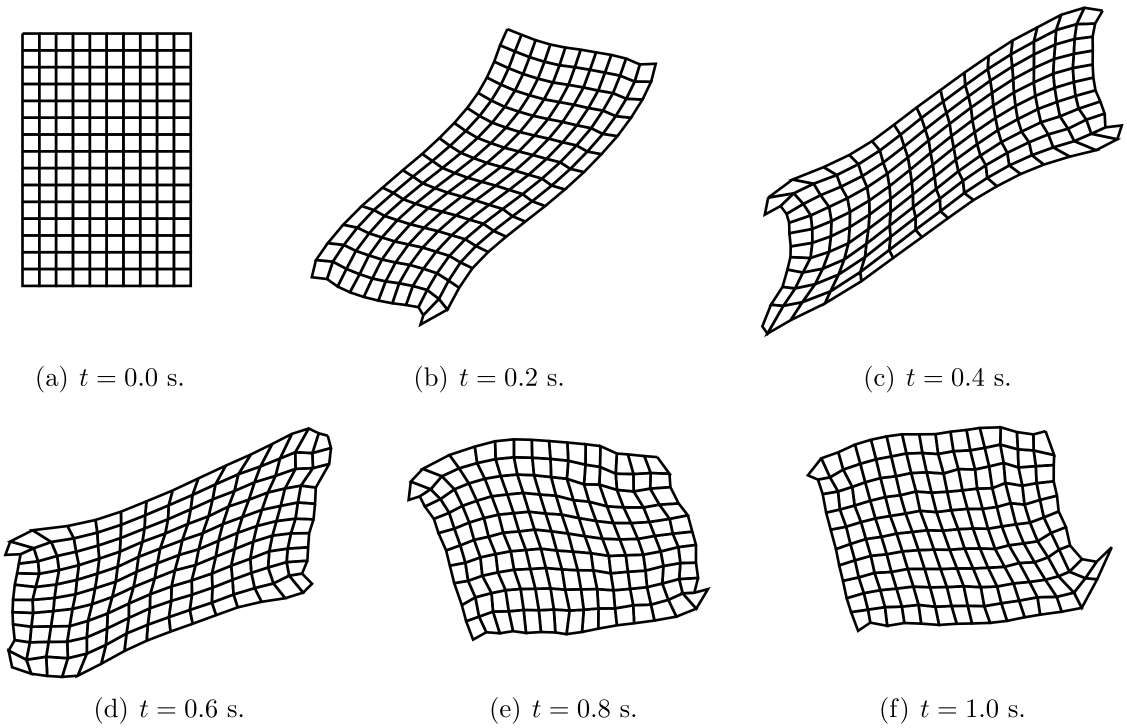
Figure 11: (Color online) Geometry, boundary conditions, initial position of the solid, and initial velocity for the parallelogram submerged in a flow. The color scale represents horizontal velocity with blue color corresponding to zero and red corresponding to 78 cm/s. The lengths are in centimeters.



(a) $\mathsf{p} = 1$, $t = 0.412$ s.

(b) $\mathsf{p} = 2$, $t = 1.000$ s.

Figure 12: Physical net of the Lagrangian mesh with 12×17 $\mathcal{C}^0$ linear elements a time step before the failure of the simulation (a), and physical net of the Lagrangian mesh with 11×16 C^1 quadratic elements when it has already completed the simulation (b). The shear modulus is $\mu^s = 400$ g/(cms^2) and the number of control variables is 13×18 in both meshes.



(a) $t = 0.0$ s.

(b) $t = 0.2$ s.

(c) $t = 0.4$ s.

(d) $t = 0.6$ s.

(e) $t = 0.8$ s.

(f) $t = 1.0$ s.

Figure 13: Physical net of the Lagrangian mesh with 10×15 $\mathcal{C}^2$ cubic elements at different time steps along the simulation. The shear modulus is $\mu^s = 300$ g/(cms^2) and the number of control variables is 13×18.

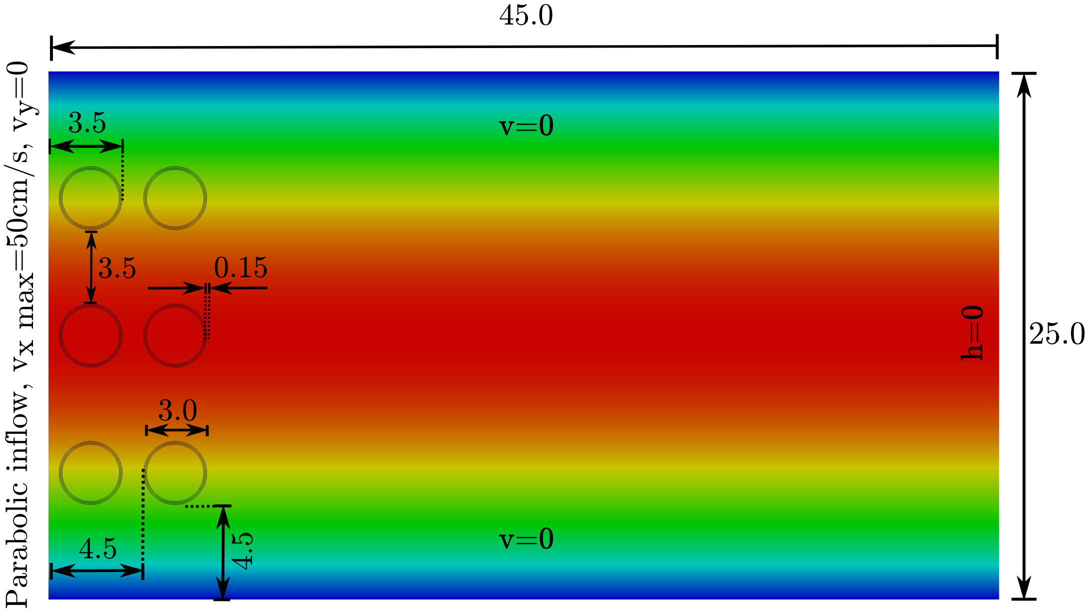
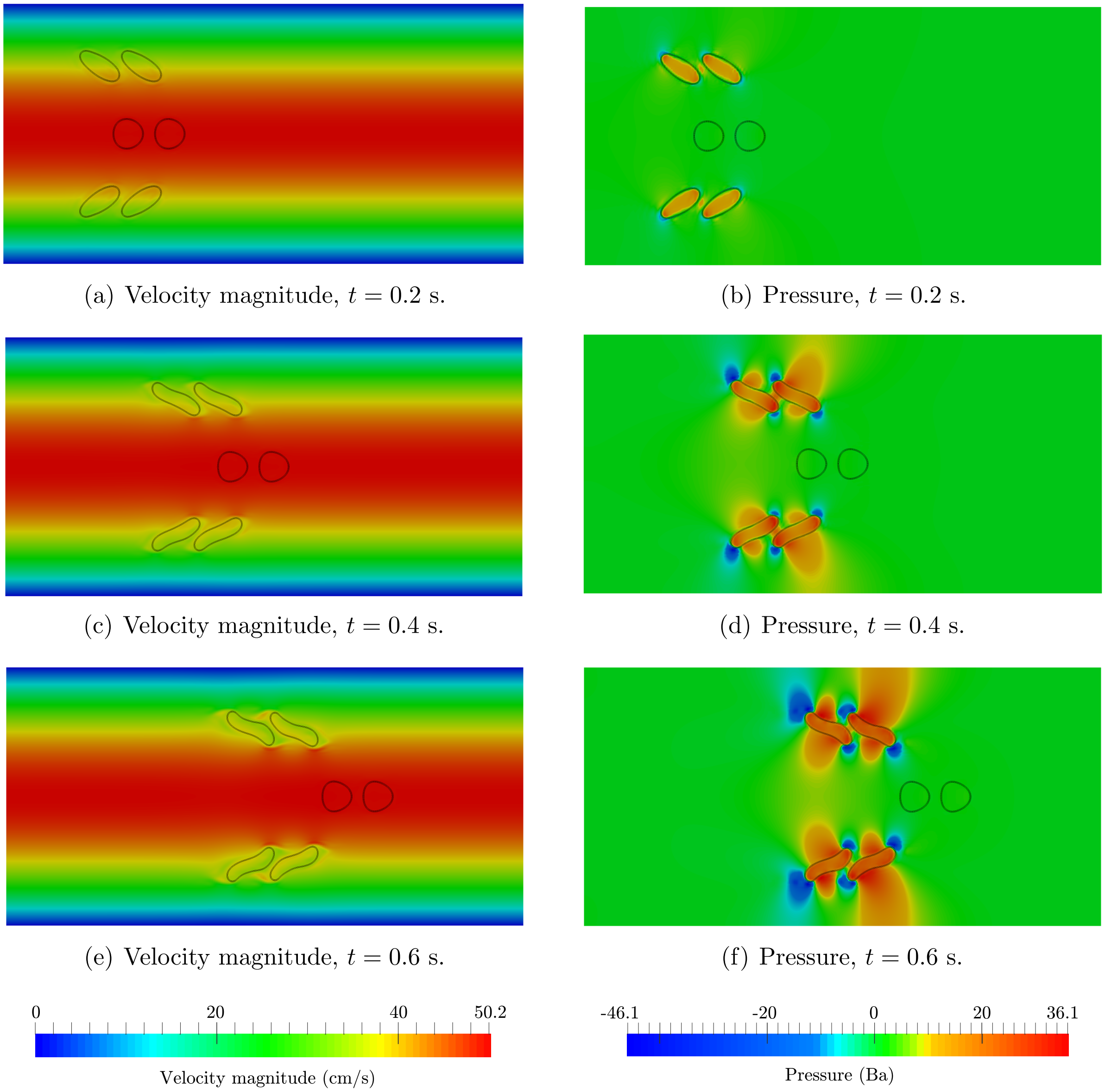


Figure 14: (Color online) Geometry, boundary conditions, initial position of the solids, and initial velocity for the 6 hollow cylinders immersed in a flow. The color scale represents horizontal velocity with blue color corresponding to zero and red corresponding to 50 cm/s. The lengths are in centimeters.



(a) Velocity magnitude, $t = 0.2$ s.

(b) Pressure, $t = 0.2$ s.

(c) Velocity magnitude, $t = 0.4$ s.

(d) Pressure, $t = 0.4$ s.

(e) Velocity magnitude, $t = 0.6$ s.

(f) Pressure, $t = 0.6$ s.

Figure 15: (Color online) Contour plot of the velocity magnitude (left) and pressure (right) at different time steps. The simulation shows how the solid objects subjected to a velocity gradient experience large deformations, while those in the middle of the channel travel practically undeformed.