Comput. Methods Appl. Mech. Engrg. 209–212 (2012) 197–211

Contents lists available at SciVerse ScienceDirect

Comput. Methods Appl. Mech. Engrg.

journal homepage: www.elsevier.com/locate/cma

High order methods for the approximation of the incompressible Navier–Stokes equations in a moving domain

G. Pena [a,*], C. Prud'homme [b], A. Quarteroni [c,d]

[a] CMUC, University of Coimbra, Largo D. Dinis, Apartado 3008, 3001-454 Coimbra, Portugal
[b] Laboratoire Jean Kuntzmann, Université Joseph Fourier Grenoble 1, BP 53 38041 Grenoble Cedex 9, France
[c] SB-MATHICSE-CMCS, École Polytechnique Fédérale de Lausanne, MA C2 573 (Bâtiment MA), Station 8, CH-1015 Lausanne, Switzerland
[d] MOX, Politechico di Milano, Piazza Leonardo da Vinci, 32-20133 Milan, Italy

ARTICLE INFO

Article history:
Received 6 July 2010
Received in revised form 20 July 2011
Accepted 30 September 2011
Available online 8 October 2011

Keywords:
Spectral element method
Incompressible Navier–Stokes equations
Arbitrary Lagrangian–Eulerian framework
Algebraic factorization methods

ABSTRACT

In this paper we address the numerical approximation of the incompressible Navier–Stokes equations in a moving domain by the spectral element method and high order time integrators. We present the Arbitrary Lagrangian Eulerian (ALE) formulation of the incompressible Navier–Stokes equations and propose a numerical method based on the following kernels: a Lagrange basis associated with Fekete points in the spectral element method context, BDF time integrators, an ALE map of high degree, and an algebraic linear solver. In particular, the high degree ALE map is appropriate to deal with a computational domain whose boundary is described with curved elements. Finally, we apply the proposed strategy to a test case.

1. Introduction

The accurate approximation of the incompressible Navier–Stokes equations for flows in moving domains is an important subject of research in applied mathematics. This type of problem appears in many important fluid dynamics applications, including fluid–structure interaction problems [8,38,10,33] or free surface flows [21,3]. The main difficulties of simulating this problem are:

(i) how to discretize the system of equations in a domain that evolves in time, see [22,9];
(ii) the techniques to solve the associated algebraic system.

The first item is related with the problem formulation and its space/time discretization. Since the domain changes its shape in time, a common technique to keep track of its evolution is the Arbitrary Lagrangian Eulerian (ALE) frame [22,9]. The ALE frame introduces a vector function that represents the domain velocity of deformation. Its numerical approximation has been discussed in the context of the spectral element method, Ho and Rønquist [21] and Bouffanais [3], or the finite element method, see Nobile [30]. Another option is the map sketched in Pena and Prud'homme in [33] that we present in this paper in full detail. A relevant aspect in devising numerical schemes in the ALE framework is the so called *Geometric Conservation Law* (GCL). A numerical scheme satisfies the GCL if it can represent a constant solution through time. Although it is neither a necessary nor a sufficient condition for convergence/stability of the schemes, in some cases, the fulfillment of the GCL implies stability independently of the domain's rate of deformation, see [30,59,60].

Regarding the space discretization, the starting point in discretizing in space the Navier–Stokes equations (in a fixed domain) in the primitive variable formulation is the choice of discrete spaces for velocity and pressure. It is a well known fact that the discrete velocity and pressure spaces cannot be chosen independently. Indeed, a discrete compatibility condition enforces that a certain gap must exist between these spaces. If such a condition is violated, then the linear system associated with the discretization fails to have a unique solution. This is the so called Brezzi–Babuska–Ladizenskaya *inf–sup condition*, see [42]. In the literature one can find a few possible choices of spaces that fulfill such condition. For some examples, see [2,46,1,50]. For an extensive analysis, see [4]. In the context of the spectral element method, it is known that, for instance, choosing velocities as continuous polynomials of degree N and pressures as piecewise discontinuous polynomials of degree N or $N-1$ violates the inf–sup condition, see [2]. At an algebraic level, this violation is reflected by the existence of non-constant pressures (defined all over the domain) whose discrete gradient is zero, leading to the non uniqueness of the solution of the Stokes/Navier–Stokes equations. One of the most popular and widely used discretizations that is free of spurious pressure modes

[*] Corresponding author. Tel.: +351 239791164; fax: +351 239793069.
E-mail address: gpena@mat.uc.pt (G. Pena).

0045-7825/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.cma.2011.09.016

was studied by Bernadi and Maday [2] and Rønquist [43]. It consists of approximating the velocities with polynomials of degree N and pressures with piecewise discontinuous polynomials of degree $N-2$. However, the corresponding error estimates are not optimal regarding the polynomial order of the approximation spaces. This is due to the fact that the inf–sup constant decreases as the polynomial order increases. A comparison of the approximation properties of these spaces can be found in Pena and Prud'homme [33].

In the context of the spectral element method, the work by Patera in [31] provided the bases for the modern *multidomain spectral method*. This version of the SEM pushes the method to deal with arbitrary geometries, combining its spectral properties with the flexibility of the finite element method. Although in the beginning it was only applied to geometries that were partitioned into quadrangular sub-domains, the monography by Sherwin and Karniadakis [24] provided a further extension of this method to geometries that could be partitioned into simplices, thus giving even more flexibility to the SEM. The definition of global basis functions can be done using Lagrange polynomials associated with suitable point sets. In the literature, several point sets have been proposed, for tensorized and simplicial domains. The Gaussian points (Gauss, Gauss–Radau and Gauss–Lobatto) are usually employed to construct Lagrange bases in tensorized geometries due to their well behaved Lebesgue constants. For simplicial domains, there is no equivalent of the Gaussian points. The equidistributed points, see [5], are a first alternative, but these do not have low Lebesgue constants and, in the context of the Galerkin method, they lead to very ill conditioned linear systems, see [32]. Other choices in the triangular case, more robust with respect to interpolation, are the Electrostatic [20], Fekete [51], and Heinrichs [19] and more recently, Warpblend [52] points. A very interesting property shared by the Electrostatic, Fekete and Warpblend points is that, on the edges of the triangle where they are defined, they coincide with the Gauss–Lobatto points. This feature allows the use of hybrid meshes (composed of quadrangles and triangles) in a continuous Galerkin setting. For the triangular spectral element method, the Fekete points are usually a good choice since they provide good numerical stability properties to the linear systems involved, see [32].

Apart from the space discretization problem, several solution strategies have been proposed to solve the unsteady Navier–Stokes equations. We highlight two of them: (i) *fully coupled methods* and (ii) *splitting methods*. Splitting methods decouple the calculation of the velocity and pressure field, by performing a splitting, either in the differential equations, see for instance [16–18], or at the algebraic level, see [40,41,45,12,11]. Such decoupling of the variables makes the calculation of the solution faster, however at the cost of introducing some error in the approximation, called *splitting error*. The differential type of splitting also introduces an artificial boundary condition (that needs to be derived) for the pressure operator. On the other hand, algebraic splitting methods do not have this requirement. Fully coupled algorithms do not introduce splitting error. Instead, they try to solve the fully coupled velocity–pressure system of equations, e.g. by the Uzawa approach or with a suitable preconditioner for the whole linear system, see [28,29,47,32]. See [5,6] for an extensive discussion. The solver proposed in this paper account for a first contribution to a complete high order ALE framework.

In the following sections, we propose a numerical strategy to solve the incompressible unsteady Navier–Stokes equations set in a moving domain. In Section 2, we present the equations written in the ALE frame of reference. Regarding the space discretization, the spectral element method is briefly introduced in Section 3.1. In our framework, we use the Lagrange basis associated with Fekete points in triangular elements, which, at the best of our knowledge, has never been used in this type of moving domain application. The description of a high order ALE map, responsible, at each time step, for describing the computational domain where the Navier–Stokes equations are to be solved, is detailed in Section 3.2. This map combines a piecewise linear harmonic extension of the boundary data, see [30,21,61,62], and a correction procedure on the boundary faces to ensure a high order description of the moving domain. In the following Sections 3.3 and 3.4, the fully discrete numerical method is presented. For the stabilization of the velocity field (in the presence of convection dominated flows), we adopt the *interior penalty* (IP) stabilization. Thus far, only linear diffusion–reaction–convection equations have been IP-stabilized while using spectral elements as the space discretization, see [55,56], or the Oseen equation, see [58], hence the use of this stabilization is new in this context. This stabilization technique has also been employed for the discretization of the Navier–Stokes equations with finite elements, see [57]. A combination of Backward Differentiation Formulas (BDF) and an extrapolation formula of the same order is used to fully discretize in time the system of equations (the same BDFq formula is used to approximate the mesh velocity associated with the ALE map). Once the differential system is fully discretized and a linear system is obtained (see Section 3.5), we consider two approaches to solve it: a LU factorization and the Yosida-q schemes. The latter have been extensively used for Navier–Stokes equations in rigid domains, see [11,12], but their adaptation to domains with moving boundaries is new to our knowledge.

Section 4 provides insight on the accuracy of the methodology proposed. First, in Section 4.1, we verify that we obtain indeed spectral convergence with the approximation spaces detailed in Section 3.3 (in the case of a domain with rigid boundaries). In the same subsection, we also show that if the boundary is curved (not affine), using curved geometrical elements is crucial in order to achieve spectral convergence. This motivates us, in the context of the ALE framework, to define, at each timestep, computational meshes that have curved edges in the boundary of the domain (and straight in the interior), which means defining ALE maps that induce this type of mesh in the computational domain, see Section 3.2. The (optimal) approximation property of our ALE map construction is verified in Section 4.2. Finally, in Section 4.3, we compare two temporal discretization strategies — BDFq schemes coupled with LU factorization and the extension of the Yosida-q schemes — in the ALE context and display the (expected) order of convergence in time.

All the computations in this paper were done with the Feel++ (Finite Element Embedded Library in c++), formerly known as the Life library, see [35,34,36,33,37].

2. Differential problem

Let us denote by Ω_{t_0} a reference configuration, for instance, the domain filled by the fluid at time $t = t_0$ in which we want to solve the Navier–Stokes equations. The position of a point in the current domain Ω_t, $t > t_0$, is denoted by $\mathbf{x}$ (in the Eulerian coordinate system) and by $\mathbf{Y}$ in the reference domain Ω_{t_0}. The system's evolution is studied in the interval $I = [t_0, T]$. In the following, quantities in bold like $\mathbf{x}$ or $\mathbf{Y}$ represent vectorial variables. The same is valid for vectorial and scalar fields as everywhere else in the paper.

We introduce a family of mappings $\mathcal{A}_t$ that for each t, associates to a point $\mathbf{Y} \in \Omega_{t_0}$ a point $\mathbf{x} \in \Omega_t$:

$$\mathcal{A}_t : \Omega_{t_0} \to \Omega_t, \quad \mathbf{x}(\mathbf{Y}, t) = \mathcal{A}_t(\mathbf{Y}), \quad t \in I. \tag{1}$$

For every $t, \mathcal{A}_t$ is assumed to be an homeomorphism in $\overline{\Omega}_{t_0}$, i.e., $\mathcal{A}_t$ is a continuous bijection from the closure $\overline{\Omega}_{t_0}$ onto $\overline{\Omega}_t$, as well as its inverse, from $\overline{\Omega}_t$ onto $\overline{\Omega}_{t_0}$. We also assume that the application

$$t \mapsto \mathbf{x}(\mathbf{Y}, t), \quad \mathbf{Y} \in \Omega_{t_0}$$

is differentiable almost everywhere in I. The application $\mathcal{A}_t$ is called the *ALE map*.

Let $f : \Omega_t \times I \rightarrow \mathbb{R}$ be a function defined in the Eulerian frame, and $\hat{f} := f \circ \mathcal{A}_t$ the corresponding function defined in the ALE framework, defined as

$$\hat{f} : \Omega_{t_0} \times I \rightarrow \mathbb{R}, \quad \hat{f}(\mathbf{Y}, t) = f(\mathcal{A}_t(\mathbf{Y}), t) \tag{2}$$

and conversely,

$$f(\mathbf{x}, t) = \hat{f}\left(\mathcal{A}_t^{-1}(\mathbf{x}), t\right).$$

Another ingredient is the ALE time derivative of f, defined as

$$\left.\frac{\partial f}{\partial t}\right|_{\mathbf{Y}} : \Omega_t \times I \rightarrow \mathbb{R}, \quad \left.\frac{\partial f}{\partial t}\right|_{\mathbf{Y}}(\mathbf{x}, t) = \frac{\partial \hat{f}}{\partial t}\left(\mathcal{A}_t^{-1}(\mathbf{x}), t\right).$$

We then define the *domain velocity of deformation* as

$$\mathbf{w}(\mathbf{x}, t) = \left.\frac{\partial \mathbf{x}}{\partial t}\right|_{\mathbf{Y}}. \tag{3}$$

In the ALE framework, the unsteady incompressible Navier–Stokes equations read as

$$\rho\left.\frac{\partial \mathbf{u}}{\partial t}\right|_{\mathbf{Y}} - \mathbf{div}_{\mathbf{x}}(2\nu\mathbf{D}_{\mathbf{x}}(\mathbf{u})) + \rho((\mathbf{u} - \mathbf{w}) \cdot \nabla_{\mathbf{x}})\mathbf{u} + \nabla_{\mathbf{x}} p = \mathbf{f}, \ \text{ in } \Omega_t \times I, \tag{4}$$

$$\mathrm{div}_{\mathbf{x}}(\mathbf{u}) = 0, \ \text{ in } \Omega_t \times I, \tag{5}$$

where all differential operators are defined w.r.t. the Eulerian coordinate system, except the ALE time derivative. The symmetric part of the *rate of deformation tensor* $\mathbf{D}_{\mathbf{x}}$ in (4) is defined as

$$\mathbf{D}_{\mathbf{x}}(\mathbf{u}) = \frac{1}{2}(\nabla_{\mathbf{x}}\mathbf{u} + (\nabla_{\mathbf{x}}\mathbf{u})^T).$$

The constant ρ is the *density* of the fluid. For simplicity of the exposition, we will consider homogeneous Dirichlet on Γ_t^D and Neumann boundary conditions on Γ_t^N. These subsets of the boundary satisfy $\partial\Omega_t = \Gamma_t^D \cup \Gamma_t^N, \Gamma_t^D \cap \Gamma_t^N = \emptyset$.

In order to derive the weak formulation for problem (4) and (5), we introduce function spaces for trial and test functions built with the ALE map $\mathcal{A}_t$ and spaces defined in the reference domain. Let $\mathbf{V}(\Omega_t)$ and $Q(\Omega_t)$ be defined as

$$\mathbf{V}(\Omega_t) = \left\{\mathbf{v} : \Omega_t \times I \rightarrow \mathbb{R}^d, \ \mathbf{v} = \hat{\mathbf{v}} \circ \mathcal{A}_t^{-1}, \ \hat{\mathbf{v}} \in \mathbf{H}^1_{\Gamma^D}(\Omega_{t_0})\right\} \tag{6}$$

and

$$Q(\Omega_t) = \left\{q : \Omega_t \times I \rightarrow \mathbb{R}, \ q = \hat{q} \circ \mathcal{A}_t^{-1}, \ \hat{q} \in L^2(\Omega_{t_0})\right\}, \tag{7}$$

where $\mathbf{H}^1_{\Gamma^D}(\Omega_{t_0})$ is the subset of $\mathbf{H}^1(\Omega_{t_0})$ whose functions are vector valued and have zero trace on $\Gamma^D = \mathcal{A}_t^{-1}\left(\Gamma_t^D\right)$.

For $\mathbf{u}$, $\mathbf{v}$, $\boldsymbol{\beta} \in \mathbf{V}(\Omega_t)$ and p, $q \in Q(\Omega_t)$, we introduce the following notations

$$(\mathbf{u}, \mathbf{v})_{\Omega_t} = \int_{\Omega_t} \mathbf{u} \cdot \mathbf{v} \; dx,$$

$$a(\mathbf{u}, \mathbf{v})_{\Omega_t} = 2\nu \int_{\Omega_t} \mathbf{D}_{\mathbf{x}}(\mathbf{u}) : \nabla_{\mathbf{x}}\mathbf{v} \; dx,$$

$$b(\mathbf{v}, p)_{\Omega_t} = \int_{\Omega_t} \mathrm{div}_{\mathbf{x}}(\mathbf{u}) p \; dx,$$

$$c(\mathbf{u}, \mathbf{v}; \boldsymbol{\beta})_{\Omega_t} = \rho \int_{\Omega_t} [\boldsymbol{\beta} \cdot \nabla_{\mathbf{x}}]\mathbf{u} \cdot \mathbf{v} \; dx.$$

With these notations, the weak formulation of the Navier–Stokes equations in the ALE framework reads as follows.

Problem 2.1. For almost every $t \in I$, find $\mathbf{u}(t) \in \mathbf{V}(\Omega_t)$, with $\mathbf{u}(t_0) = \mathbf{u}_0$ in Ω_{t_0} and $p(t) \in Q(\Omega_t)$, such that

$$\begin{aligned} &\rho\left(\left.\frac{\partial \mathbf{u}}{\partial t}\right|_{\mathbf{Y}}, \mathbf{v}\right)_{\Omega_t} + c(\mathbf{u}, \mathbf{v}; \mathbf{u} - \mathbf{w})_{\Omega_t} + a(\mathbf{u}, \mathbf{v})_{\Omega_t} - b(\mathbf{v}, p)_{\Omega_t} \\ &\quad = (\mathbf{f}, \mathbf{v})_{\Omega_t}, \quad \forall \mathbf{v} \in \mathbf{V}(\Omega_t), b(\mathbf{u}, q)_{\Omega_t} = 0, \quad \forall q \in Q(\Omega_t) \end{aligned} \tag{8}$$

3. Numerical approximation

3.1. Construction of the spectral element space

We address now the discretization of the system of equations (8) and start by introducing some concepts and notations.

3.1.1. Notations and preliminaries

We denote by $\widehat{K}$ a reference element, which is either a *d-simplex* (interval, triangle, tetrahedron) or a *d-hypercube* (interval, quadrangle, hexahedron) where d is the topological dimension of Ω_{t_0}. In $\widehat{K}$ we build the polynomial space $\mathbb{P}_N(\mathcal{T}^d)(\mathbb{Q}_N(\mathcal{Q}^d))$, corresponding to the space of polynomials of total degree less or equal to N (degree smaller or equal than N), for d = 1, 2, 3. In the following, we shall denote this space by $\mathbb{P}_N(\widehat{K})$. Given an arbitrary element, say K, in a triangulation of Ω_{t_0}, we define the geometrical mapping $\boldsymbol{\varphi}_K : \widehat{K} \rightarrow K$, that is a polynomial in $\mathbb{P}_N(\widehat{K})$.

We are now ready to define the polynomial spaces necessary for the spectral element method.

3.1.2. The spectral element space

Let $\mathcal{T}_{t_0,\delta}$ be a triangulation of the reference domain Ω_{t_0} into N_{el} elements that we denote by K_e, where $\delta = (h, N_{\mathrm{geo}})$. Let $\Omega_{t_0,\delta}$ be a domain, obtained by the union of all elements in the triangulation $\mathcal{T}_{t_0,\delta}$, that approximates Ω_{t_0}. The domain $\Omega_{t_0,\delta}$ and the elements of the triangulation $\mathcal{T}_{t_0,\delta}$ satisfy the assumptions detailed in [32].

We define the *spectral element space* as

$$\mathcal{F}_N(\mathcal{T}_{t_0,\delta}) = \left\{v \in C^0(\overline{\Omega_{t_0,\delta}}) : v_{|K_e} \in \mathbb{P}_N(K_e), \quad \forall K_e \in \mathcal{T}_{t_0,\delta}\right\}, \tag{9}$$

where

$$\mathbb{P}_N(K_e) = \left\{p : \ p = \hat{p} \circ \boldsymbol{\varphi}_{K_e}^{-1}, \quad \hat{p} \in \mathbb{P}_N(\widehat{K})\right\}. \tag{10}$$

In the following, for this space we build the Lagrange nodal basis associated with Fekete points, see [51]. This set of points, also called *high order nodes*, is associated with the space $\mathcal{F}_N(\mathcal{T}_{t_0,\delta})$. They are obtained by collecting, for each K_e, the image of the Fekete points in the reference element through the geometrical mapping. Details on the construction of function bases for this space can be found in [32].

3.2. Construction of the discrete ALE map

We denote by $\Omega_{t,\delta}$ the discrete computational domain where the Navier–Stokes equations are to be solved, at time t and

$$\mathbf{g}_{t,\delta} : \partial\Omega_{t_0,\delta} \rightarrow \partial\Omega_{t,\delta}$$

the map that transforms the boundary of $\Omega_{t_0,\delta}$ onto the boundary of $\Omega_{t,\delta}$ (which we assume known *a priori*). The discrete ALE map $\mathcal{A}_{t,\delta}$ then satisfies

$$\mathcal{A}_{t,\delta_{|\Omega_{t_0,\delta}}} = \mathbf{g}_{t,\delta}, \quad \mathcal{A}_{t,\delta}(\Omega_{t_0,\delta}) = \Omega_{t,\delta}. \tag{11}$$

To build $\mathcal{A}_{t,\delta}$, and given the fact that in practice we only have a description of the boundary of $\Omega_{t,\delta}$, we start by dividing the boundary $\partial\Omega_{t,\delta}$ into two parts: $\partial\Omega^{\mathcal{D}}_{t,\delta}$ a part of the boundary that remains fixed in time and $\partial\Omega^{\sigma}_{t,\delta}$ the part the moves with t. Clearly, $\Omega_{t,\delta} = \partial\Omega^{\mathcal{D}}_{t,\delta} \cup \partial\Omega^{\sigma}_{t,\delta}$. Let us assume that we have a description of $\partial\Omega^{\sigma}_{t,\delta}$ in terms of polynomials of degree N.

We then make the following assumptions: (i) the upper and lower parts of the boundary $\partial\Omega^{\sigma}_{t,\delta}$ are described by polynomials p of degree N defined in $[x_0, x_L]$; (ii) $\Omega_{t_0,\delta}$ can be covered exactly by a triangulation composed of elements with straight edges. This assumption is not as restrictive as it seems because mesh generators can typically create triangulations for N_{geo} = 1, 2.

Remark 3.1. Some popular open source mesh generators, such as GMSH, provide high order mesh generation, see [13]. This means that the mesh generator can produce triangulations such that $N_{\text{geo}} > 2$.

3.2.1. Standard harmonic extension

Then, given the description of the boundary in terms of polynomials, we perform a standard harmonic extension of the function $\mathbf{g}_{t,\delta}$ by solving.

Problem 3.1. Find $\mathcal{A}_{t,\delta_1} \in (\mathcal{F}_1(\mathcal{T}_{t_0,\delta}))^d$ such that

$$\begin{cases} \int_{\Omega_{t_0,\delta}} \nabla \mathcal{A}_{t,\delta_1} : \nabla \mathbf{z}\, dx = 0, & \forall \mathbf{z} \in (\mathcal{F}_1(\mathcal{T}_{t_0,\delta}))^d, \\ \mathcal{A}_{t,\delta_1} = \mathbf{g}_{t,\delta}, & \text{on } \partial\Omega_{t_0,\delta}. \end{cases} \tag{12}$$

Remark 3.2. The integrals in the previous problem and the following ones are calculated with the use of numerical quadrature formulas. In the case the integrator function is polynomial, we use enough points for each integral so that it is calculated exactly. If the function is not polynomial, then we use a large number of points so that the quadrature error is negligible. We stress that in both cases, the function being integrated and the geometrical mapping are taken into account.

We obtain an ALE map $\mathcal{A}_{t,\delta_1}$, where $\delta_1 = (h, 1)$, that once applied to the triangulation $\mathcal{T}_{t_0,\delta}$ generates a mesh $\mathcal{T}_{t,\delta_1}$ for $\Omega_{t,\delta}$, i.e., $\mathcal{T}_{t,\delta_1} = \mathcal{A}_{t,\delta_1}(\mathcal{T}_{t_0,\delta})$. This process is depicted in Fig. 2. To make the exposition of the following steps easier, we fix an element in $\mathcal{T}_{t_0,\delta}$ and its image through the transformation $\mathcal{A}_{t,\delta_1}$, say K_{t_0} and K_t, respectively. In Fig. 2 we can identify these elements with the shaded elements on the left and right columns, respectively. In the same figure, we illustrate two different possible configurations using this technique, which we shall use throughout the presentation of this method.

3.2.2. Introducing high order nodes

The next step is to project $\mathcal{A}_{t,\delta_1}$ onto the space $(\mathcal{F}_{N_{\text{geo}}}(\mathcal{T}_{t_0,\delta}))^d$. Let $\mathcal{B} = \{\boldsymbol{\phi}_i\}_i$ be a nodal basis for this space (in our simulations, $\mathcal{B}$ is the Lagrange basis associated with Fekete points). With these notations, the projection of $\mathcal{A}_{t,\delta_1}$ onto $(\mathcal{F}_{N_{\text{geo}}}(\mathcal{T}_{t_0,\delta}))^d$, denoted $\mathcal{A}^*_{t,\delta}$, is

$$\mathcal{A}^*_{t,\delta} = \sum_i \alpha_i \boldsymbol{\phi}_i,$$

where the coefficients α_i are determined by interpolating $\mathcal{A}_{t,\delta_1}$ over the nodal points associated with $(\mathcal{F}_{N_{\text{geo}}}(\mathcal{T}_{t_0,\delta}))^d$. Since the basis is nodal, these coefficients represent the evaluation of $\mathcal{A}_{t,\delta_1}$ at these nodes.

We now change the value of the degrees of freedom, associated with edges that are in contact with the curved wall. If $\mathbf{x}_0$ is a point belonging to the mentioned edge that corresponds to a degree of freedom and (x_t, y_t), see Fig. 1, are the coordinates of its image through $\mathcal{A}^*_{t,\delta}$, then we take (see Fig. 3).

$$\mathcal{A}^*_{t,\delta}(\mathbf{x}_0) = (x_t, p(x_t)).$$

This shifting in the coordinates solves the problem of making the edges of the elements conform with the curved boundary. However, this might create a map that has a singular Jacobian since part of the interior of the element in the reference domain is mapped outside the corresponding element in the computational mesh, see Fig. 4(a). It may be also that there are no points mapped outside the element and the transformation is valid, see Fig. 4(b), however the approximation may be poor.

3.2.3. Shifting the nodes on edges and faces

The final step to obtain a valid ALE map is to shift also the nodes on the faces of the elements of the reference domain and obtain a situation as in Fig. 5. The coordinates of the new nodes are obtained using a transformation of Gordon–Hall type, see [14,15,32,5].

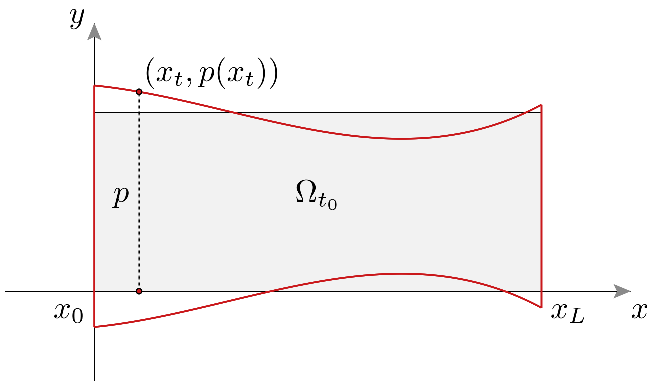


Fig. 1. Description of the reference (shaded and rectangular region) domain $\Omega_{t_0,\delta}$ and computational domain $\Omega_{t,\delta}$ (curved region). The top and bottom boundaries of the computational domain are described in terms of polynomials.

Let us assume that the edges, denoted Γ_i, of the curved elements in $\Omega_{t,\delta}$, denoted K_t^{curved}, are parameterized by functions $\pi_i : [-1, 1] \to \Gamma_i$. We assume that the parameterizations verify $\pi_0(-1) = \pi_2(1)$, $\pi_1(-1) = \pi_0(1)$ and $\pi_2(-1) = \pi_1(1)$, see [32] for more details.

A transformation that extends smoothly the boundary mappings to the interior of K_t^{curved} can be found, for instance, in [5,32]. Here, we use the one from the latter manuscript. The idea is to define a transformation from the reference element, $\widehat{\Omega}$, onto K_t^{curved}. The transformation has the form

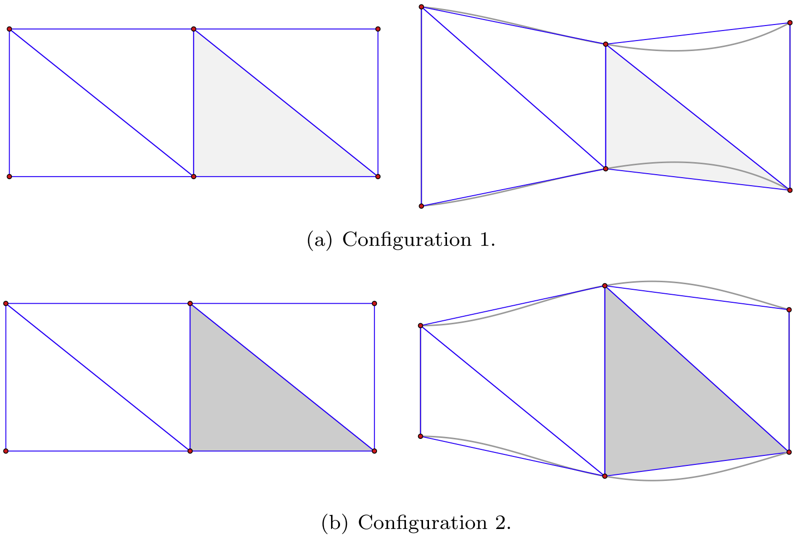
$$\begin{aligned} \boldsymbol{\varphi}_{K_t^{curved}}(\xi, \eta) = {} & \frac{1-\eta}{2}\pi_2(\xi) - \frac{1+\xi}{2}\pi_2(-\eta) + \frac{1-\xi}{2}\pi_1(-\eta) \\ & - \frac{1+\eta}{2}\pi_1(\xi) + \left(1 + \frac{\xi+\eta}{2}\right)\pi_0(-\xi) \\ & - \frac{1+\xi}{2}\pi_0(-1-\xi-\eta) + \frac{1+\xi}{2}\pi_2(1) + \frac{\xi+\eta}{2}\pi_1(1) \end{aligned}$$

for all $(\xi, \eta) \in \widehat{\Omega}$.

Let us denote the geometrical transformation from the element onto K_t by $\boldsymbol{\varphi}_{K_t}$ and M_i the set of high order nodes belonging to the topological subentity of dimension i. This means that M_0 are the vertices of K_t, M_1 are the nodes on the edges of K_t and M_2 are the high order nodes that need to be shifted. First, we apply $\boldsymbol{\varphi}_{K_t}^{-1}$ to $M = M_0 \cup M_1 \cup M_2$. We obtain a set of points that lie exactly in $\widehat{\Omega}$. Moreover, $\boldsymbol{\varphi}_{K_t}^{-1}(M_0)$ are the vertices of $\widehat{\Omega}$, $\boldsymbol{\varphi}_{K_t}^{-1}(M_1)$ lie on the edges of $\widehat{\Omega}$ and $\boldsymbol{\varphi}_{K_t}^{-1}(M_2)$ lie on the face of $\widehat{\Omega}$. We know *a priori*, which edges from K_t are curved and build parametrizations of them. Therefore, we can construct $\boldsymbol{\varphi}_{K_t^{curved}}$ and apply it to $\boldsymbol{\varphi}_{K_t}^{-1}(M_2)$. Like this, we obtain a new set of points, which are the shifted high order nodes in the element K_t^{curved}. Let us denote this new set of points by M_2^{curved}. The final stage is to replace the value of the degrees of freedom lying in the face of K_t in the map $\mathcal{A}^*_{t,\delta}$ with the values of M_2^{curved}. Let $\mathcal{A}_{t,\delta}$ denote the updated map.

Remark 3.3. Although we did not make any considerations about the orientation of the vertices, edges or faces of the elements, all the previous transformations respect that orientation and therefore replacing the values in M_2 by the ones in M_2^{curved} is sufficient to build the correct ALE map. See [32] for more details regarding the orientation of the elements of a triangulation.

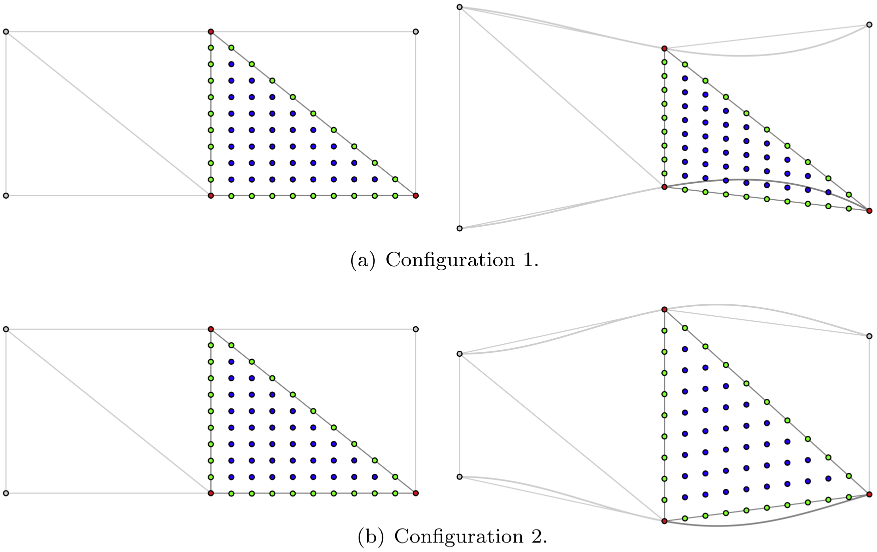
Remark 3.4. In [32] the deduction of a similar transformation for triangulations composed with quadrangles is also done. For the three dimensional case, we refer the reader to [25] or [48].



(a) Configuration 1.

(b) Configuration 2.

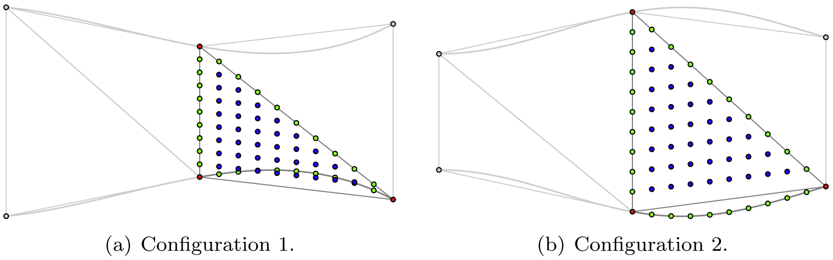
Fig. 2. Transformation from the triangulation in the reference domain $\Omega_{t_0,\delta}$ (left) to the triangulation in the computational domain, $\mathcal{T}_{t,\delta_1}$ through the map $\mathcal{A}_{t,\delta_1}$. On the right, the straight edges correspond to the triangulation $\mathcal{T}_{t,\delta_1}$ and the curved boundaries correspond to $\Omega_{t,\delta}$.



(a) Configuration 1.

(b) Configuration 2.

Fig. 3. Effect of $\mathcal{A}^*_{t,\delta}$ on a equidistributed point set defined in an element of the reference mesh.



(a) Configuration 1.

(b) Configuration 2.

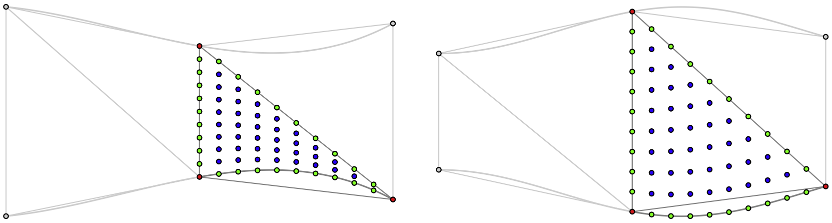
Fig. 4. Effect of $\mathcal{A}^*_{t,\delta}$ if only the degrees of freedom in the edges are shifted.

Remark 3.5. We highlight that the construction just presented does not depend on the extension operator that was used to generate the first mesh in the computational domain. Other procedures can be applied, see [3].

A numerical study of the approximation properties of the ALE map just presented is done in Section 4.2.

Advantages and disadvantages. A consequence of the definition of the map just presented is that it is affine for elements that do not share an edge with the curved boundary. For these elements T,

$$\mathcal{A}_{t,\delta_{|T}} = \mathcal{A}_{t,\delta_1}{}_{|T}.$$



(a) Configuration 1. (b) Configuration 2.

Fig. 5. Transformation from the reference mesh to the computational one (update on faces degrees of freedom).

This also means that the geometric mapping associated with these elements is affine. This situation is illustrated in Fig. 6. The advantage of this property is that when integrating linear/bilinear forms in these elements, a constant Jacobian is associated with the geometrical transformation and therefore a minimal order quadrature can be used (in the sense that the geometrical transformation does not need to be taken into account). Only the elements that intersect the curved boundary, the quadrature order has to account for the non constant Jacobian.

Remark 3.6. We stress that — thanks to the correction of the map in the boundaries of the domain that moves while maintaining affine elements in the interior of the triangulation — the map (i) describes the curved boundary with the expected accuracy (compared with affine geometrical elements) and (ii) can be calculated very cheaply. This an improvement with respect to the maps proposed by solving the harmonic extension using either first order or *N*th order elements. Indeed, the former is cheaply computed but does not describe the geometry of the domain with enough accuracy to ensure spectral convergence (see Section 4.1) whereas the latter is very expensive from a computational point of view although it generates a map that describes the boundary that ensures spectral accuracy. The map we propose takes advantages of both options: retain accuracy at a low cost.

A final remark concerns possible strategies in the case the boundary's deformation is "large", as it can happen that an interior straight edge of the mesh intersects the curved boundary. This originates an invalid element and several techniques are proposed in literature to deal with this issue, see [26,27]. However, since we might now want to change the structure of our reference mesh, other possibilities consist of using a control function in the Laplace operator of the harmonic extension to follow the boundary movement, see for instance [23] or employing a reference mesh that is refined near the curved boundary. A final, more costly possibility, is to re-mesh the whole domain. In our approach, since the displacements we consider are small, re-meshing is not necessary.

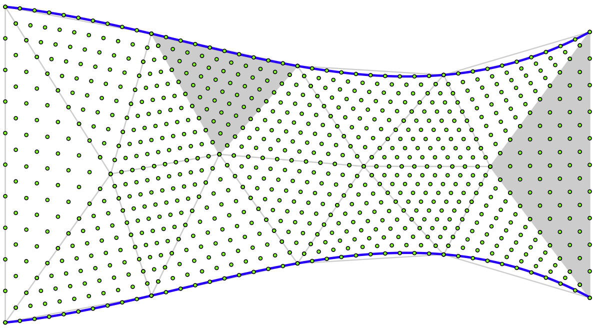


Fig. 6. The effect of the mapping $\mathcal{A}_{t,\delta}$ to an equidistributed point set in the reference domain (Configuration 1). The shadowed elements are, from left to right, triangles where the geometrical transformation is of high degree or linear, respectively.

Remark 3.7. We have introduced an ALE map based on a harmonic extension operator and a displacement correction procedure in the degrees of freedom (of the ALE map) that lie on the curved boundary. We point out that the overall construction of the ALE map proposed is independent of the operator used to generate the first order ALE map. Other strategies, such as the Winslow smoother, see [63], could be used too.

3.3. Variational formulation of the semi-discrete problem

We address now the discretization in space of the system of equations (8). Let (see Fig. 7)

$$\mathbf{V}_\delta(\Omega_{t,\delta}) = \Big\{ \mathbf{v} : \Omega_{t,\delta} \times I \to \mathbb{R}^d,\ \mathbf{v} = \hat{\mathbf{v}} \circ \mathcal{A}_{t,\delta}^{-1},\\ \hat{\mathbf{v}} \in \mathbf{H}^1_{\Gamma^D}(\Omega_{t_0}) \cap (\mathcal{F}_N(\mathcal{T}_{t_0,\delta}))^d \Big\} \tag{13}$$

and

$$Q_\delta(\Omega_{t,\delta}) = \Big\{ q : \Omega_{t,\delta} \times I \to \mathbb{R},\ q = \hat{q} \circ \mathcal{A}_{t,\delta}^{-1},\ \hat{q} \in \mathcal{F}_M(\mathcal{T}_{t_0,\delta}) \Big\} \tag{14}$$

where $N \geqslant 2$ and $M = N - 1$ or $M = N - 2$. $\mathbf{V}_\delta(\Omega_{t,\delta})$ and $Q_\delta(\Omega_{t,\delta})$ are the finite dimensional function spaces in which velocity and pressure will be discretized for any time $t > 0$, respectively.

We introduce the *semi-discrete domain velocity*, $\mathbf{w}_\delta$, defined as

$$\mathbf{w}_\delta(\mathbf{x}, t) = \frac{\partial \mathcal{A}_{t,\delta}}{\partial t} \circ \mathcal{A}_{t,\delta}^{-1}, \quad \forall \mathbf{x} \in \Omega_{t,\delta},\ t > 0. \tag{15}$$

Since the construction of the discrete ALE map lies upon the discretization of a differential problem for a given mesh, we also call this quantity *mesh velocity*.

The semi-discrete variational problem reads as follows.

Problem 3.2. For almost every $t \in I$, find $\mathbf{u}_\delta(t) \in (\mathcal{F}_N(\mathcal{T}_{t,\delta}))^d$, with $\mathbf{u}_\delta(t_0) = \mathbf{u}_{0,\delta}$ in $\Omega_{t_0,\delta}$ and $p_\delta(t) \in Q(\Omega_{t,\delta})$, such that

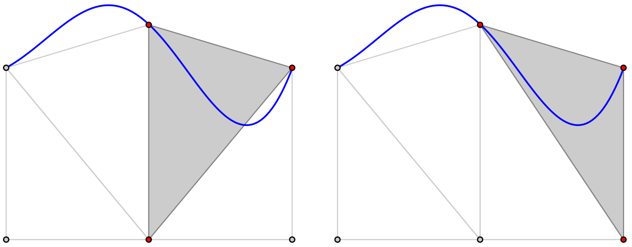


Fig. 7. Invalid element created due to the high distortion in the boundary (left). Edge swap technique to correct possible invalid elements (right).

$$\rho\left(\frac{\partial \mathbf{u}_\delta}{\partial t}\Big|_{\mathbf{Y}}, \mathbf{v}\right)_{\Omega_{t,\delta}} + c(\mathbf{u}_\delta, \mathbf{v}; \mathbf{u}_\delta - \mathbf{w}_\delta)_{\Omega_{t,\delta}} + a(\mathbf{u}_\delta, \mathbf{v})_{\Omega_{t,\delta}} - b(\mathbf{v}, p)_{\Omega_{t,\delta}}$$
$$= (\mathbf{f}, \mathbf{v})_{\Omega_{t,\delta}}, \forall \mathbf{v} \in \mathbf{V}_\delta(\Omega_{t,\delta}), b(\mathbf{u}, q)_{\Omega_{t,\delta}} = 0, \quad \forall q \in Q_\delta(\Omega_{t,\delta}). \tag{16}$$

Remark 3.8. To enhance the stability of the spatial discretization, we add to the first equation of system (16) the quantity $s(\mathbf{u}_\delta, \mathbf{v}; \mathbf{u}_\delta)_{\Omega_{t,\delta}}$ defined by

$$s(\mathbf{u}, \mathbf{v}; \boldsymbol{\beta})_{\Omega_t} = \frac{\rho}{2}\int_{\Omega_t} \mathrm{div}_{\mathbf{x}}(\boldsymbol{\beta})\mathbf{u} \cdot \mathbf{v} dx.$$

This term is consistent with the Navier–Stokes equations, since at the fully continuous level, $\mathrm{div}_{\mathbf{x}}(\mathbf{u}) = 0$. For a more detailed explanation of the importance of adding such term to the formulation in the context of stability, see [30].

Interior penalty stabilization. Often fluid flows are dominated by the convection hence a suitable stabilization has to be operated on the variational formulation. In our formulation, this might happen because of the velocity of the fluid or, in the ALE framework, because of the contribution of mesh velocity. In the problems addressed in this paper, the latter is the main reason to include such a term in the variational formulation since it is the mesh velocity, and not the fluid velocity, that makes the flow dominated by the convection.

In our approach, we consider the *interior penalty* (IP) stabilization technique. Let us first introduce some notations. Let $\mathcal{F}_I$ be the set of internal faces of a triangulation $\mathcal{T}_\delta$, where $\delta = (h, N_{\mathrm{geo}})$. Given a face $F \in \mathcal{F}_I$, let T_1 and T_2 be the elements of $\mathcal{T}_\delta$ that share F, that is, $F = T_1 \cap T_2$. Let $v \in H^1(\Omega_\delta)$ and $\mathbf{v} \in \mathbf{H}^1(\Omega_\delta)$. We denote by v_1, v_2, respectively, $\mathbf{v}_1$, $\mathbf{v}_2$ the restrictions of v and $\mathbf{v}$ to the elements T_1 and T_2. Let $\mathbf{n}_1$ and $\mathbf{n}_2$ be the exterior normals of T_1 and T_2. Then, the *jumps* of v and $\mathbf{v}$ across F are defined as

$$[\![v]\!]_F = v_1\mathbf{n}_1 + v_2\mathbf{n}_2, \tag{17}$$
$$[\![\mathbf{v}]\!]_F = \mathbf{v}_1 \cdot \mathbf{n}_1 + \mathbf{v}_2 \cdot \mathbf{n}_2. \tag{18}$$

In the case of the tensor function $\nabla\mathbf{v}$, we define the jump as

$$[\![\nabla\mathbf{v}]\!]_F = \nabla\mathbf{v}_1\mathbf{n}_1 + \nabla\mathbf{v}_2\mathbf{n}_2.$$

The stabilization term to be added to the variational formulation reads

$$j(\mathbf{u}, \mathbf{v}; \boldsymbol{\beta})_{\Omega_\delta} = \gamma \sum_{F \in \mathcal{F}_I} \int_F |\boldsymbol{\beta} \cdot \mathbf{n}| \frac{h_F^2}{N^{3.5}} [\![\nabla\mathbf{u}]\!]_F \cdot [\![\nabla\mathbf{v}]\!]_F \, ds, \tag{19}$$

where h_F denotes the length of the face F and N the degree of the velocity approximation and γ is the *stabilization parameter*. In the system of equations (16), the term to add (to the first equation) to account for the IP stabilization is $j(\mathbf{u}_\delta, \mathbf{v}; \mathbf{u}_\delta - \mathbf{w}_\delta)_{\Omega_{t,\delta}}$.

3.4. Time integration

We start by approximating the time derivative by a *backward differentiation formula* of order q (BDFq) and linearize the nonlinear convective term by an extrapolation formula of order q. Given $\Delta t \in (0, T)$, we set $t_0 = 0$, $t_n = t_0 + n\Delta t$ (for any $n \geqslant 1$) and $N_T = [\frac{T}{\Delta t}]$ (i.e., the integer part of $\frac{T}{\Delta t}$); then as follows.

Problem 3.3. For each $n \geqslant q - 1$, we look for the solution $(\mathbf{u}_\delta^{n+1}, p_\delta^{n+1}) \in (\mathcal{F}_N(\mathcal{T}_{t_{n+1},\delta}))^d \times Q_\delta(\Omega_{t_{n+1},\delta})$, with $\mathbf{u}_\delta^0 = \mathbf{u}_{0,\delta}$ in $\Omega_{t_0,\delta}$, such that

$$\rho\frac{\beta_{-1}}{\Delta t}(\mathbf{u}_\delta^{n+1}, \mathbf{v})_{\Omega_{t_{n+1},\delta}} + a(\mathbf{u}_\delta^{n+1}, \mathbf{v})_{\Omega_{t_{n+1},\delta}} + s(\mathbf{u}_\delta^{n+1}, \mathbf{v}; \mathbf{u}_\delta^*)_{\Omega_{t_{n+1},\delta}}$$
$$- b(\mathbf{v}, p_\delta^{n+1})_{\Omega_{t_{n+1},\delta}} + c(\mathbf{u}_\delta^{n+1}, \mathbf{v}; \mathbf{u}_\delta^* - \mathbf{w}_\delta^{n+1})_{\Omega_{t_{n+1},\delta}}$$
$$= \left(\tilde{\mathbf{f}}_\delta^{n+1}, \mathbf{v}\right)_{\Omega_{t_{n+1},\delta}}, \quad \forall \mathbf{v} \in \mathbf{V}_\delta(\Omega_{t_{n+1},\delta}) \tag{20}$$
$$b(\mathbf{u}_\delta^{n+1}, q)_{\Omega_{t_{n+1},\delta}} = 0, \quad \forall q \in Q_\delta(\Omega_{t_{n+1},\delta}),$$

where

$$\tilde{\mathbf{f}}_\delta^{n+1} = \mathbf{f}^{n+1} + \rho\sum_{j=0}^{q-1}\frac{\beta_j}{\Delta t}\mathbf{u}_\delta^{n-j}$$

Notice that the functions $\mathbf{u}_\delta^{n-j}$ should be defined in $\Omega_{t_{n-j},\delta}$, which might not coincide with the integration domain $\Omega_{t_{n+1},\delta}$. However, these quantities can be ported from their domain of definition to the current one by applying ALE maps. More precisely, if we denote by $\mathbf{u}_\delta^{n-j,*}$ the approximation of $\mathbf{u}(t_{n-j})$ defined in $\Omega_{t_{n-j},\delta}$, then

$$\mathbf{u}_\delta^{n-j} = \mathbf{u}_\delta^{n-j,*} \circ \mathcal{A}_{t_{n+1},\delta} \circ \mathcal{A}_{t_{n-j},\delta}^{-1}.$$

Similar considerations are valid every time a quantity defined in a domain of the type $\Omega_{t_k,\delta}$ needs to be ported to the current computational domain $\Omega_{t_{n+1},\delta}$.

In Eq. (20), there are two quantities that we have not yet defined, or at least said how to calculate: $\mathbf{u}_\delta^*$ and $\mathbf{w}_\delta^{n+1}$. Regarding the former, this is a linearization of the convective term of the Navier–Stokes equations. We define $\mathbf{u}_\delta^*$ as (see [39])

$$\mathbf{u}_\delta^* = \begin{cases} \mathbf{u}_\delta^n, & q = 1, \\ 2\mathbf{u}_\delta^n - \mathbf{u}_\delta^{n-1}, & q = 2, \\ 3\mathbf{u}_\delta^n - 3\mathbf{u}_\delta^{n-1} + \mathbf{u}_\delta^{n-2}, & q = 3, \\ 4\mathbf{u}_\delta^n - 6\mathbf{u}_\delta^{n-1} + 4\mathbf{u}_\delta^{n-2} - \mathbf{u}_\delta^{n-3}, & q = 4. \end{cases} \tag{21}$$

Regarding $\mathbf{w}_\delta^{n+1}$, the discrete time derivative of the discrete ALE map, we also adopt the BDFq schemes to approximate it. For instance, for $q = 2$, we have

$$\mathbf{w}_\delta^{n+1} = \frac{1}{\Delta t}\left(\frac{3}{2}\mathcal{A}_{t_{n+1},\delta} - 2\mathcal{A}_{t_n,\delta} + \frac{1}{2}\mathcal{A}_{t_{n-1},\delta}\right) \circ \mathcal{A}_{t_{n+1},\delta}^{-1}. \tag{22}$$

Numerical schemes of the type (20) have been analyzed in literature in the context of a linear advection diffusion problem. It has been shown in Nobile [30] that when applying the Backward Euler time integration method (equivalent to our method with $q = 1$) to the advection diffusion problem in the non-conservative form, the scheme is only conditionally stable. The stability condition (derived in [30]) is

$$\Delta t < \left(\|\mathrm{div}(\mathbf{w}_\delta^n)\|_{L^\infty(\Omega_{t_n,\delta})} + \sup_{t \in (t_n, t_{n+1})} \|J_{\mathcal{A}_{t_n,t_{n+1}}} \mathrm{div}(\mathbf{w}_\delta)\|_{L^\infty(\Omega_{t,\delta})}\right)^{-1} \tag{23}$$

for all $n = 1, \ldots, N_T$. We remark that only geometrical quantities are involved in (23). If the mesh velocity is calculated such that it is divergence free, then the scheme is unconditionally stable. This is a sufficient condition to satisfy the Geometric Conservation Law (GCL), see Remark 3.9.

Also in [30], for the case $q = 2$, again in the context of a linear advection diffusion equation, it is shown that the method is conditionally stable and the time step restriction depends only on geometrical quantities, just like (23).

Remark 3.9 (*Geometric Conservation Law*). We say that an equation/numerical scheme satisfies the *Geometric Conservation Law* (GCL) if it is able to reproduce a constant solution (in the absence of source terms and proper boundary conditions).

Let us suppose that $\mathbf{u}_\delta^i \equiv \tilde{\mathbf{u}}$ and $p_\delta^i \equiv 0$ are constant, for all $i = 0, \ldots, n$. Notice that if a constant velocity is solution of the Navier–Stokes system, then the pressure is zero all over the domain, in the presence of homogeneous Neumann boundary conditions.

Then, from the system of equations (20), in order that $(\tilde{\mathbf{u}}, 0)$ be a solution of (20), we need that

$$\int_{\Omega_{t_{n+1}}} \frac{\beta_{-1}}{\Delta t}\mathbf{u}_\delta^{n+1}\cdot\mathbf{v}dx = \int_{\Omega_{t_{n+1}}}\sum_{j=0}^{q-1}\frac{\beta_j}{\Delta t}\mathbf{u}_\delta^{n-j}\cdot\mathbf{v}dx, \quad \forall \mathbf{v}\in\mathbf{V}_\delta(\Omega_{t_{n+1},\delta}),$$

which is true if

$$\beta_{-1} = \sum_{j=0}^{q-1}\beta_j. \tag{24}$$

The previous condition is necessary and sufficient if $\tilde{\mathbf{u}} \neq \mathbf{0}$. On the other hand, equality (24) is a consequence of the consistency of the BDFq schemes. Therefore, our formulation of the Navier–Stokes equations in the ALE frame satisfies the GCL, for all BDFq schemes considered.

3.5. Fully discrete system

Let us consider basis functions for the spaces $\mathbf{V}_\delta(\Omega_{t_{n+1},\delta})$ and $Q_\delta(\Omega_{t_{n+1},\delta})$, say

$$\mathbf{V}_\delta(\Omega_{t_{n+1},\delta}) = \text{span}\{\boldsymbol{\phi}_i\}_{i=1}^{N_u}, Q_\delta(\Omega_{t_{n+1},\delta}) = \text{span}\{\psi_i\}_{i=1}^{N_p}.$$

In practice, the construction of these spaces is done by considering their reference counterparts, $\mathbf{V}_\delta(\Omega_{t_0,\delta})$ and $Q_\delta(\Omega_{t_0,\delta})$, and applying the ALE map to the reference triangulation, $\mathcal{T}_{t_0,\delta}$.

We introduce the following matrices and vectors (we omit the superscript $n+1$ to indicate the dependence of the basis functions and the matrices on n to simplify the notation):

$$\begin{array}{ll}
G_\delta(i,j) = -b(\boldsymbol{\phi}_i,\psi_j)_{\Omega_{t_{n+1},\delta}}, & 1\leqslant i\leqslant N_u, 1\leqslant j\leqslant N_p,\\
D_\delta(i,j) = b(\boldsymbol{\phi}_j,\psi_i)_{\Omega_{t_{n+1},\delta}}, & 1\leqslant j\leqslant N_u, 1\leqslant i\leqslant N_p,\\
H_\delta(i,j) = a(\boldsymbol{\phi}_i,\boldsymbol{\phi}_j)_{\Omega_{t_{n+1},\delta}}, & 1\leqslant i,j\leqslant N_u,\\
C_\delta(i,j) = c\left(\boldsymbol{\phi}_i,\boldsymbol{\phi}_j;\mathbf{u}_\delta^*-\mathbf{w}_\delta^{n+1}\right)_{\Omega_{t_{n+1},\delta}} + s\left(\boldsymbol{\phi}_i,\boldsymbol{\phi}_j;\mathbf{u}_\delta^*\right)_{\Omega_{t_{n+1},\delta}}, & 1\leqslant i,j\leqslant N_u,\\
M_\delta(i,j) = (\boldsymbol{\phi}_i,\boldsymbol{\phi}_j)_{\Omega_{t_{n+1},\delta}} & 1\leqslant i,j\leqslant N_u,\\
\mathbf{F}_\delta(j) = \left(\tilde{\mathbf{f}}_\delta^{n+1},\boldsymbol{\phi}_j\right)_{\Omega_{t_{n+1},\delta}}, & 1\leqslant j\leqslant N_u
\end{array}$$

and

$$F_\delta = \rho\frac{\beta_{-1}}{\Delta t}M_\delta + \nu H_\delta + C_\delta.$$

Then Problem 3.3 is equivalent to solve, for each $n \geqslant 1$ a system of the form

$$\underbrace{\begin{bmatrix} F_\delta & G_\delta \\ D_\delta & 0 \end{bmatrix}}_{A_N}\begin{bmatrix}\mathbf{U}_\delta^{n+1} \\ \mathbf{P}_\delta^{n+1}\end{bmatrix} = \begin{bmatrix}\mathbf{F}_\delta \\ 0\end{bmatrix} \tag{25}$$

where $\mathbf{U}_\delta^{n+1}$ and $\mathbf{P}_\delta^{n+1}$ denote the vector representations of $\mathbf{u}_\delta^{n+1}$ and p_δ^{n+1} in the bases of $\mathbf{V}_\delta(\Omega_{t_{n+1},\delta})$ and $Q_\delta(\Omega_{t_{n+1},\delta})$, respectively and $N = N_u + N_p$.

Remark 3.10. When the IP stabilization term is considered in the variational formulation, its contribution is added to matrix C_δ. In this case, the components of C_δ are defined as

$$\begin{aligned} C_\delta(i,j) = {} & c\left(\boldsymbol{\phi}_i,\boldsymbol{\phi}_j;\mathbf{u}_\delta^*-\mathbf{w}_\delta^{n+1}\right)_{\Omega_{t_{n+1},\delta}} + s\left(\boldsymbol{\phi}_i,\boldsymbol{\phi}_j;\mathbf{u}_\delta^*\right)_{\Omega_{t_{n+1},\delta}} \\ & + j\left(\boldsymbol{\phi}_i,\boldsymbol{\phi}_j;\mathbf{u}_\delta^*-\mathbf{w}_\delta^{n+1}\right)_{\Omega_{t_{n+1},\delta}}, \end{aligned}$$

for $1 \leqslant i, j \leqslant N_u$.

3.6. Linear algebra solution strategy

In the following, we analyse the convergence properties of two strategies: a direct solver using a LU factorization and the Yosida-q schemes proposed in [40,45,12,11]. Numerical results using these schemes are presented in Section 4.3.2.

From now on, we subscript the matricies by N and not by δ.

3.6.1. LU preconditioner

The LU factorization of the matrix A_N are calculated with the help of the Ifpack library provided by Trilinos, see [44]. This factorization is calculated using the KLU algorithm, see [7,49]. When solving similar systems, as is the case in Section 4.3, we reuse the LU factorization as preconditioner until the number of iterations needed to solve the linear system is equal to 10. Once this value is attained, the factorization is recalculated. A better strategy to determine when to recalculate the preconditioner is described in [53].

3.6.2. The Yosida-q schemes

An efficient solution technique of the Navier–Stokes equations are splitting methods, of either differential or algebraic type, see [17,18,16,41,40,45,5] for a few references. The former methods split the differential operators of the equations, while the latter splits the linear system like (25) using an inexact block LU factorization. In this section, we review the algebraic factorization methods known as *Yosida-q schemes.*

We start by noticing that A_N can be factorized as follows

$$A_N = \begin{bmatrix} I_N & 0 \\ D_N F_N^{-1} & I_N \end{bmatrix}\begin{bmatrix} F_N & 0 \\ 0 & S_N \end{bmatrix}\begin{bmatrix} I_N & F_N^{-1}G_N \\ 0 & I_N \end{bmatrix}. \tag{26}$$

In the first version of the Yosida scheme, introduced by Quarteroni et al. [41], the central ideas is to approximate the matrix F_N^{-1}, in the pressure Schur complement $S_N = -\frac{\Delta t}{\beta_{-1}}D_N F_N^{-1}G_N$, by a second order in time approximation

$$F_N^{-1} \approx \frac{\Delta t}{\beta_{-1}}M_N^{-1}.$$

This leads to the approximate matrix $\widetilde{A}_N$ given by

$$\widetilde{A}_N = \begin{bmatrix} F_N & 0 \\ D_N & -\frac{\Delta t}{\beta_{-1}}D_N M_N^{-1}G_N \end{bmatrix}\begin{bmatrix} I_N & F_N^{-1}G_N \\ 0 & I_N \end{bmatrix}.$$

It was shown by Quarteroni et al. [40] that this scheme applied to the unsteady Stokes equations, together with a BDF2 time discretization leads to second order in time convergence for the velocity, order 3/2 for the pressure and unconditional stability.

Remark 3.11. Replacing the pressure Schur complement by $S_N^{app} = -\frac{\Delta t}{\beta_{-1}}D_N M_N^{-1}G_N$, called *approximate pressure Schur complement*, allows one to reduce the computational cost to solve system (25), while introducing a splitting error of the same order as the time discretization used for the Navier–Stokes equations. Since the S_N^{app} is s.p.d. we can use the preconditioned conjugate gradient method to invert it. A preconditioner can be built using a preconditioner for M_N, which can be, for instance, an algebraic multigrid preconditioner (like the one provided by Trilinos). Moreover, if M_N is lumped, we can build the approximate Schur matrix and factorize it once. We can then reuse this factorization to precondition subsequent solves of the Yosida method.

Later versions of this first Yosida scheme, now called *Yosida*-2 scheme, have been proposed and improved the order of convergence in time for velocity and pressure. By introducing a matrix J_N in the inexact block LU factorization

$$\widetilde{A}_N = \begin{bmatrix} F_N & 0 \\ D_N & -\frac{\Delta t}{\beta_{-1}}D_N M_N^{-1}G_N \end{bmatrix}\begin{bmatrix} I_N & F_N^{-1}G_N \\ 0 & J_N \end{bmatrix}$$

and choosing it carefully, one can obtain schemes of order q for the velocity and $q-1/2$ for the pressure, named *Yosida*-q. This choice is based on the minimization of the splitting error originated from approximating A_N with $\widetilde{A}_N$, see [45,12,11].

All three Yosida schemes differ only in the expression of matrix J_N. While for Yosida-2 it is equal to the identity matrix, the higher order versions take more involved expressions. If we define

$$B_N = -D_N \frac{\Delta t}{\beta_{-1}} M_N^{-1} F_N \frac{\Delta t}{\beta_{-1}} M_N^{-1} G_N$$

then for Yosida-3

$$J_N = B_N^{-1} S_N^{app}. \tag{27}$$

The fourth order version of the Yosida schemes, Yosida-4, is obtained by replacing B_N in (27) by

$$\widehat{B}_N = B_N \left(S_N^{app}\right)^{-1} B_N + B_N + D_N \left(\frac{\Delta t}{\beta_{-1}} M_N^{-1} F_N\right)^2 \frac{\Delta t}{\beta_{-1}} M_N^{-1} G_N.$$

Though appearing complex to calculate, the three Yosida schemes can be summarized in Algorithm 1.

Algorithm: A step of the Yosida algorithm.

```
given $\mathbf{f}$ and $\mathbf{g}$,
solve $F_N \tilde{\mathbf{u}} = \mathbf{f}$
solve $S_N^{app} \tilde{\mathbf{p}} = \mathbf{g} + D_N \tilde{\mathbf{u}}$
if $q > 2$ then
  solve $\mathbf{z} = B_N \tilde{\mathbf{p}}$
  solve $S_N^{app} \mathbf{p} = \mathbf{z}$
  if $q = 4$ then
    compute $\mathbf{p}_B = B_N \mathbf{p} + \mathbf{z} + D_N \left(\frac{\Delta t}{\beta_{-1}} M_N^{-1} F_N\right)^2 \frac{\Delta t}{\beta_{-1}} M_N^{-1} G_N \tilde{\mathbf{p}}$
    solve $S_N^{app} \mathbf{p} = \mathbf{p}_B$
  end if
else
  $\mathbf{p} = \tilde{\mathbf{p}}$
end if
$F_N(\mathbf{u} - \tilde{\mathbf{u}}) = -G_N \mathbf{p}$
return $(\mathbf{u}, \mathbf{p})$
```

A complete analysis on the convergence properties of all Yosida schemes, for a time-dependent Stokes problem, is provided in [11].

4. Numerical experiments

In this section we present some numerical tests to the spectral elements, the ALE map, the preconditioning strategy and the whole solver for the Navier–Stokes equations.

4.1. Using high order geometrical elements

We start by checking the convergence of the spectral element method for the case of a fixed domain and to provide an argument for using curved geometrical elements. We consider a modified Kovasznay solution of the steady Stokes equations, see [54]. The exact solution is

$$\begin{aligned} \mathbf{u}(x,y) &= \left[1 - e^{\lambda x}\cos(2\pi y), \frac{\lambda}{2\pi} e^{\lambda x}\sin(2\pi y)\right]^T, \\ p(x,y) &= -\frac{e^{2\lambda x}}{2}, \lambda = \frac{1}{2\nu} - \sqrt{\frac{1}{4\nu^2} + 4\pi^2}. \end{aligned} \tag{28}$$

The viscosity is $\nu = 0.035$ and the forcing term for the momentum equation reads

$$\begin{aligned} \mathbf{f}(x,y) = {} & (e^{\lambda x}((\lambda^2 - 4\pi^2)\nu\cos(2\pi y) - \lambda e^{\lambda x}), \quad e^{\lambda x}\nu\sin(2\pi y) \\ & \times (-\lambda^2 + 4\pi^2))^T. \end{aligned} \tag{29}$$

The domain Ω where the equations are defined is obtained as follows: we take $\Omega^* = (-0.5, 1) \times (-0.5, 1.5)$ and deform the edge $\partial\Omega^*_{bottom} = (-0.5, 1) \times \{-0.5\}$ according to the law

$$\boldsymbol{\varphi}(\mathbf{Y}) = [\mathbf{Y}, 0.08(\mathbf{Y} + 0.5)(\mathbf{Y} - 1)(\mathbf{Y}^2 - 1)]^T, \quad \mathbf{Y} \in (-0.5, 1).$$

The rest of the boundary of Ω^* remains the same. We denote by $\partial\Omega_{bottom} = \boldsymbol{\varphi}(\partial\Omega^*_{bottom})$ the curved edge. Dirichlet boundary conditions are then derived from the exact solution. A plot of the domain's shape, velocity and pressure profile are depicted in Fig. 8.

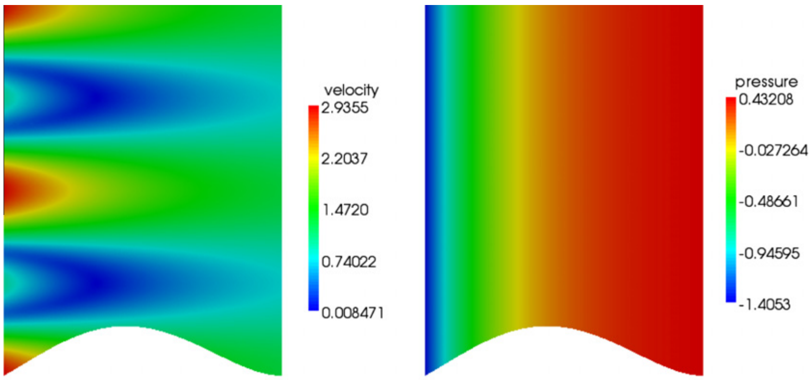
Let us fix $h > 0$ and $N_{\text{geo}} = 1, 2, 3, 4$. We consider a triangulation $\mathcal{T}_\delta$ of Ω, where $\delta = (h, N_{\text{geo}})$ and the respective domain Ω_δ it induces, that is,

$$\overline{\Omega_\delta} = \bigcup_{T \in \mathcal{T}_\delta} T.$$

Remark 4.1. Notice that if geometrical elements of order four are used, the boundary of Ω is described exactly for all h, yielding no error in approximating the geometry of the domain.

In Fig. 9 we plot the error in $\mathbf{H}^1(\Omega_\delta)$-norm for the velocity and $L^2(\Omega_\delta)$-norm for the pressure. We observe that spectral convergence is achieved using all four types of geometrical elements. This is however misleading since the error in the geometry approximation is not taken into account in the error quantities we calculated. We merely showed that spectral convergence is achieved without taking into account the error in the geometry approximation.

In order to take into account for the latter and motivate the use of high order geometrical elements, we propose two alternatives.



(a) Magnitude of velocity (b) Pressure

Fig. 8. Plot of the solution of the Kovasznay problem.

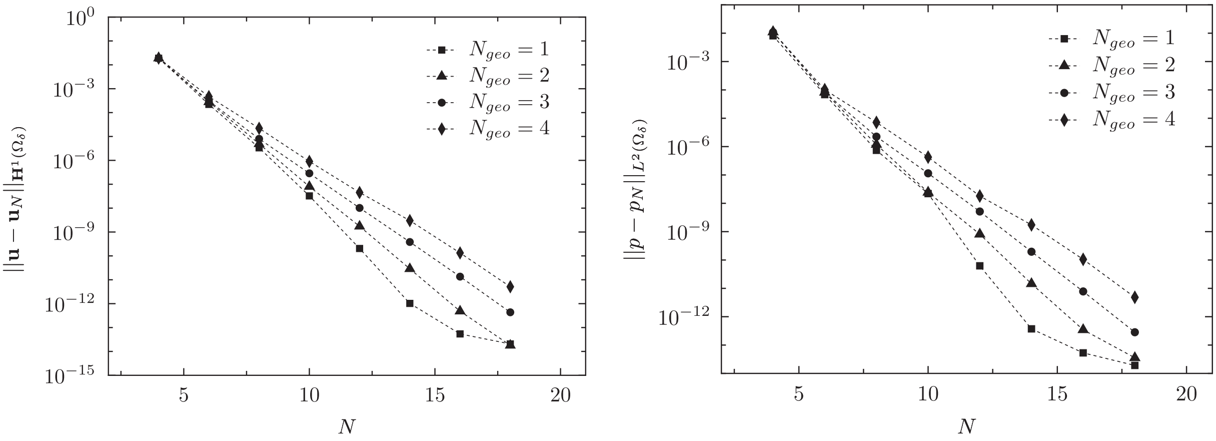


Fig. 9. Convergence plots for the modified Kovasznay example using several high order geometrical elements and the $\mathbb{P}_N - \mathbb{P}_{N-2}$ method.

The first relies on computing the error of the approximation in the $\mathbf{H}^1(\Omega)$-norm then we obtain the plots shown in Fig. 10. The solution $\mathbf{u}_N$ of the problem is nodally projected onto the velocity space of the same polynomial degree associated with the fourth order mesh (that describes the domain exactly). By taking into account the approximation of the domain in the error, we observe that for $N_{\text{geo}} = 1, 2, 3$, the error stagnates. There is convergence only when the geometry is exactly described. In general this kind of testcase is not really satisfactory because there would be extrapolations involved while interpolating the results for $N_{\text{geo}} = 1, 2, 3$ on the order 4 mesh. Nevertheless we checked that there is no extrapolation in this testcase and chose to display this convergence plot which we found informative.

The second (proper) way is to calculate the *drag* and *lift* forces on the curved boundary. These coefficients are respectively the first and second component of the vector force

$$\int_{\partial\Omega_{bottom}} (-p\mathbf{I} + \nu\nabla\mathbf{u})\mathbf{n}ds.$$

We use as reference drag and lift the values obtained in the fourth order mesh (the one that describes the domain exactly) and with an approximation space of high degree for velocity and pressure. In Fig. 11 we plot the error between several approximations for the domain/fluid variables and exact drag/lift values obtained using the fourth order mesh and the $\mathbb{P}_8 - \mathbb{P}_6$ method.

We obtain similar results as in Fig. 10: the error stagnates, unless we approximate the geometry of the domain exactly. We also observe the expected convergence in h for the drag and lift coefficients.

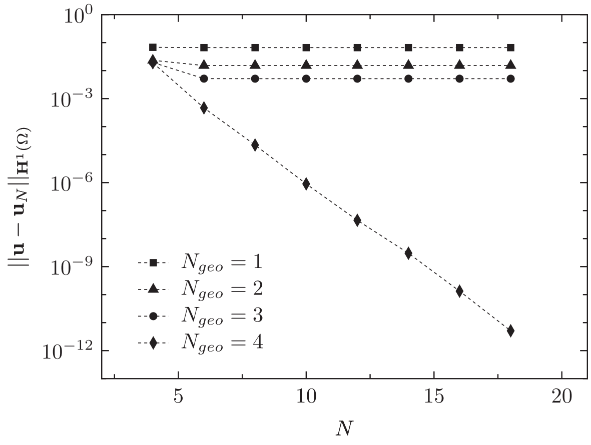


Fig. 10. Convergence plot in the $\mathbf{H}^1(\Omega)$-norm for the modified Kovasznay example using high order geometrical elements and the $\mathbb{P}_N - \mathbb{P}_{N-2}$ method.

Remark 4.2. We highlight that measuring the error for the lift/drag does not involve any interpolation procedure to the high order mesh and should be preferred to measuring the error in the velocity and pressure fields.

4.2. ALE map approximation properties

We present now some numerical results to assess the accuracy of the ALE map describing the boundary of a 2D domain. Let us consider the reference domain $\Omega_{t_0} = (0,5) \times (-1,1)$.

We define Ω_{t_1} as the domain we obtain by moving the upper and lower edges of the rectangle Ω_{t_0} using the following displacement functions:

- upper boundary: $\boldsymbol{\eta}(x,1) = [x, 1 + 0.3\cos(x)]^T$
- lower boundary: $\boldsymbol{\eta}(x,-1) = [x, -1.1 - 0.3\cos(x)]^T$.

In Fig. 12(a)–(c) we show the application of the ALE maps $\mathcal{A}_{h,N_{\text{geo}}} : \Omega_{t_0} \to \Omega_{t_1,h,N_{\text{geo}}}$ constructed using polynomials of degree two to five to a mesh of the reference domain.

We also wanted to determine the accuracy at which the ALE maps describe the boundary of the domain Ω_{t_1}. For this, we measured the error

$$\|(\mathcal{A}_{h,N_{\text{geo}}}(\cdot,1) - \boldsymbol{\eta}(\cdot,1)) \cdot \mathbf{e}_2\|_{L^2(0,5)}$$

in the upper boundary of the reference domain and

$$\|(\mathcal{A}_{h,N_{\text{geo}}}(\cdot,-1) - \boldsymbol{\eta}(\cdot,-1)) \cdot \mathbf{e}_2\|_{L^2(0,5)}$$

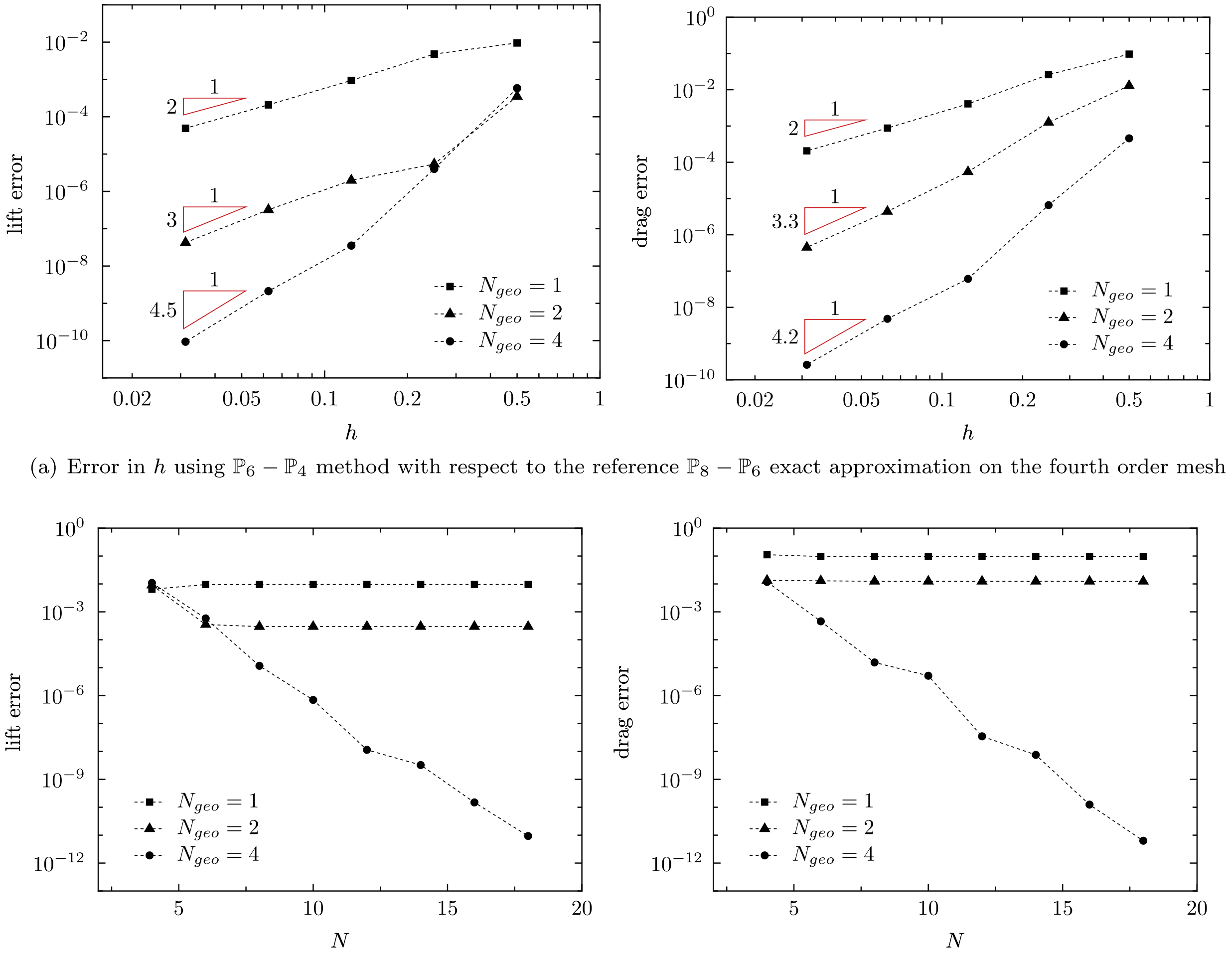
in the lower part of Ω_{t_0}. We plot the sum of both quantities in Fig. 13. The error decreases with the expected rates, i.e., $\mathcal{O}(h^{N_{\text{geo}}+1})$.

4.3. Incompressible Navier–Stokes equations in a moving domain

Consider $\Omega_{t_0} = (0,5) \times (-1,1)$ and Ω_t obtained from the reference domain by applying a vectorial field $\mathbf{d}$ to $\Gamma_t^L = I \times \{-1\}$, where $I = [0,5]$, while not moving the rest of the boundary of Ω_{t_0}. The displacement law is defined by

$$\mathbf{d}(\mathbf{Y},t) = 0.02((\mathbf{Y}-2.5)^2 + 5)\mathbf{x}(5-\mathbf{Y})$$
$$(f(t)\chi(t \in [1,3]) + \chi(t > 3)), \tag{30}$$

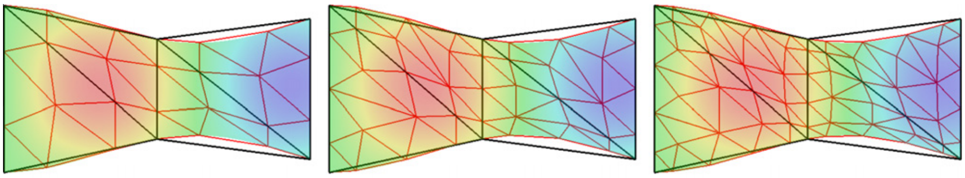
with $t \in I$, $\chi(\cdot)$ is the characteristic function and



(a) Error in h using $\mathbb{P}_6 - \mathbb{P}_4$ method with respect to the reference $\mathbb{P}_8 - \mathbb{P}_6$ exact approximation on the fourth order mesh

(b) Error using high order geometrical elements

Fig. 11. Convergence plot in the $\mathbf{H}^1(\Omega)$-norm for the modified Kovasznay example using high order geometrical elements and the $\mathbb{P}_N - \mathbb{P}_{N-2}$ method.



(a) $\mathcal{A}_{h,3}$ map (b) $\mathcal{A}_{h,4}$ map (c) $\mathcal{A}_{h,5}$ map

Fig. 12. The thick lines define the $\mathbb{P}_1$ coarse mesh used in the construction of the ALE maps. Inside each of the elements of this mesh, we observe the $\mathbb{P}_1$ triangulation constructed on top of the high order nodes. In this figure, $h = 2$.

$$f(t) = -0.15625t^7 + 2.1875t^6 - 12.46875t^5 + 37.1875t^4 - 62.34375t^3 + 59.0625t^2 - 29.53125t + 6.0625.$$

In Fig. 14 we see the effect of the displacement law in Ω_{t_0} and depict the new shape of Ω_t.

Remark 4.3. The function f satisfies $f(1) = 0$, $f(3) = 1$ and $f^{(k)}(1) = f^{(k)}(3) = 0$, $k = 1, 2, 3$. It is the only polynomial of degree 7 that satisfies these conditions. It was constructed such that the variation of the mesh velocity in time is smooth enough.

We now build the solution that we use to benchmark our ALE Navier–Stokes solver. Let $\hat{\mathbf{u}} = (1 - y^2, 0)^T$ and $\hat{p} = -2\nu(x - 5)$ be the solution of the steady Navier–Stokes equations in the reference domain Ω_{t_0}.

We consider Eqs. (4) and (5) defined in Ω_t with $\mathbf{f} \equiv \mathbf{0}$. Regarding boundary conditions, we define

$$\Gamma_t^N = \{5\} \times (-1, 1) \quad \text{and} \quad \Gamma_t^D = \partial\Omega_t \setminus \Gamma_t^N.$$

We set as boundary conditions

$$\mathbf{u} = \hat{\mathbf{u}}, \quad \text{on} \quad \Gamma_t^D \tag{31}$$

and

$$(-p\mathbf{I} + \mathbf{D}_{\mathbf{x}}(\mathbf{u}))\mathbf{n} = (-\hat{p}\mathbf{I} + \mathbf{D}_{\mathbf{x}}(\hat{\mathbf{u}}))\mathbf{n}, \quad \text{on} \quad \Gamma_t^N,$$

where $\mathbf{I}$ is the $d \times d$ identity matrix.

Remark 4.4. We remark that since the boundary of Ω_t deforms inside Ω_{t_0}, Eq. (31) makes sense in the part of the domain that changes in time. Also, the pair $(\hat{\mathbf{u}}, \hat{p})$ is the solution of (4) and (5)

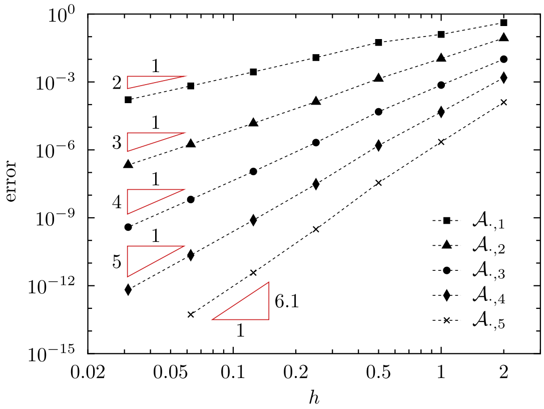


Fig. 13. Convergence plot for the high order ALE maps.

with the boundary conditions that we presented. In Fig. 14 we show the evolution of the domain Ω_t for three time steps. We also show in the same figure, the macro structure of a fourth order mesh, that is, a triangulation such that $N_{\text{geo}} = 4$, that describes the geometry of the domain exactly, see [33] for more details on this tool.

This benchmark test allows us to test the Navier–Stokes ALE framework with spectral elements in space, high order time integration, high order geometrical elements and also the IP stabilization.

The discretization of this problem is done with the scheme proposed in Problem 3.3. We try two strategies to solve the linear system (25): the first, using a direct method and the second, using algebraic splitting, more precisely, the Yosida-q schemes.

Let us first define the following error quantities that we are going to measure in order to assess the accuracy of the solver. We denote E_u the error in the velocity and E_p the error in the pressure and we define them as

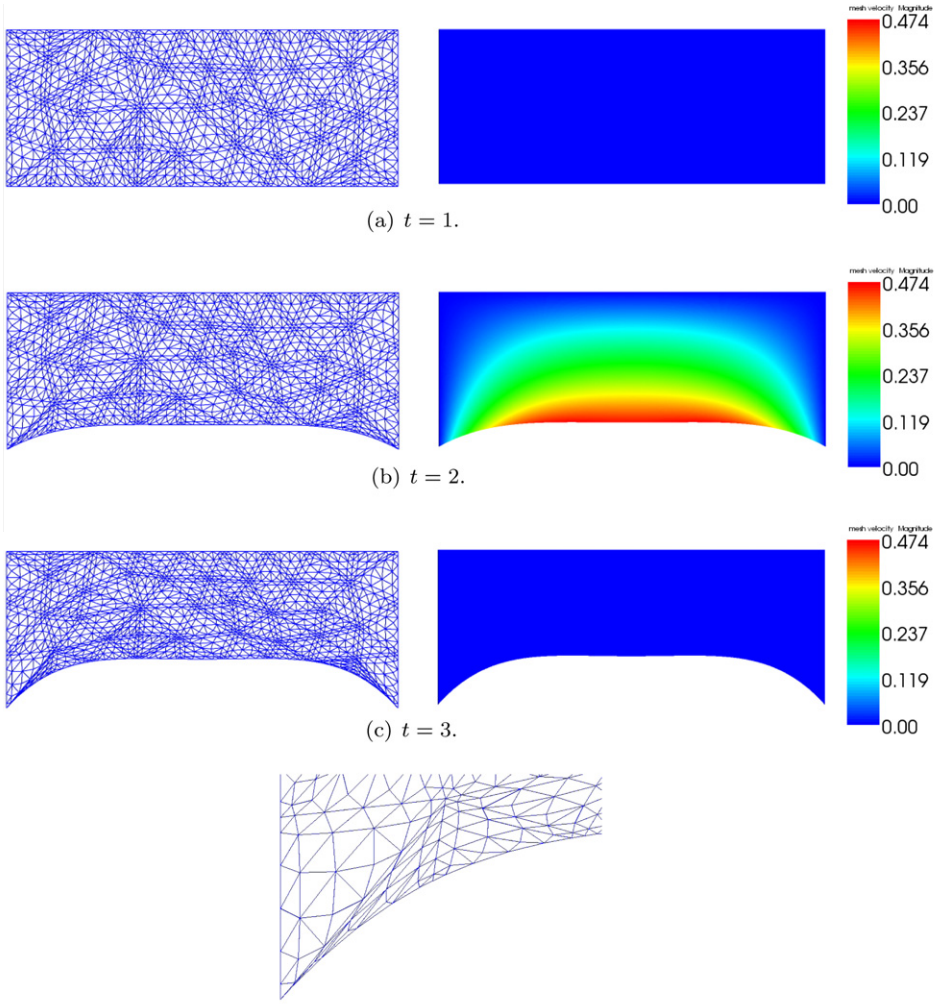
$$E_u = \left(\Delta t \sum_{n=0}^{N_T} \|\mathbf{u}(t_n) - \mathbf{u}_\delta^n\|^2_{\mathbf{H}^1(\Omega_{t_n,\delta})} \right)^{1/2}$$

and

$$E_p = \left(\Delta t \sum_{n=0}^{N_T} \|p(t_n) - p_\delta^n\|^2_{L^2(\Omega_{t_n,\delta})} \right)^{1/2}.$$

4.3.1. Using a direct method

In Fig. 15 we plot the error quantities E_u and E_p for two choices of approximation spaces for velocity and pressure and N_{geo}, $N = 2$,



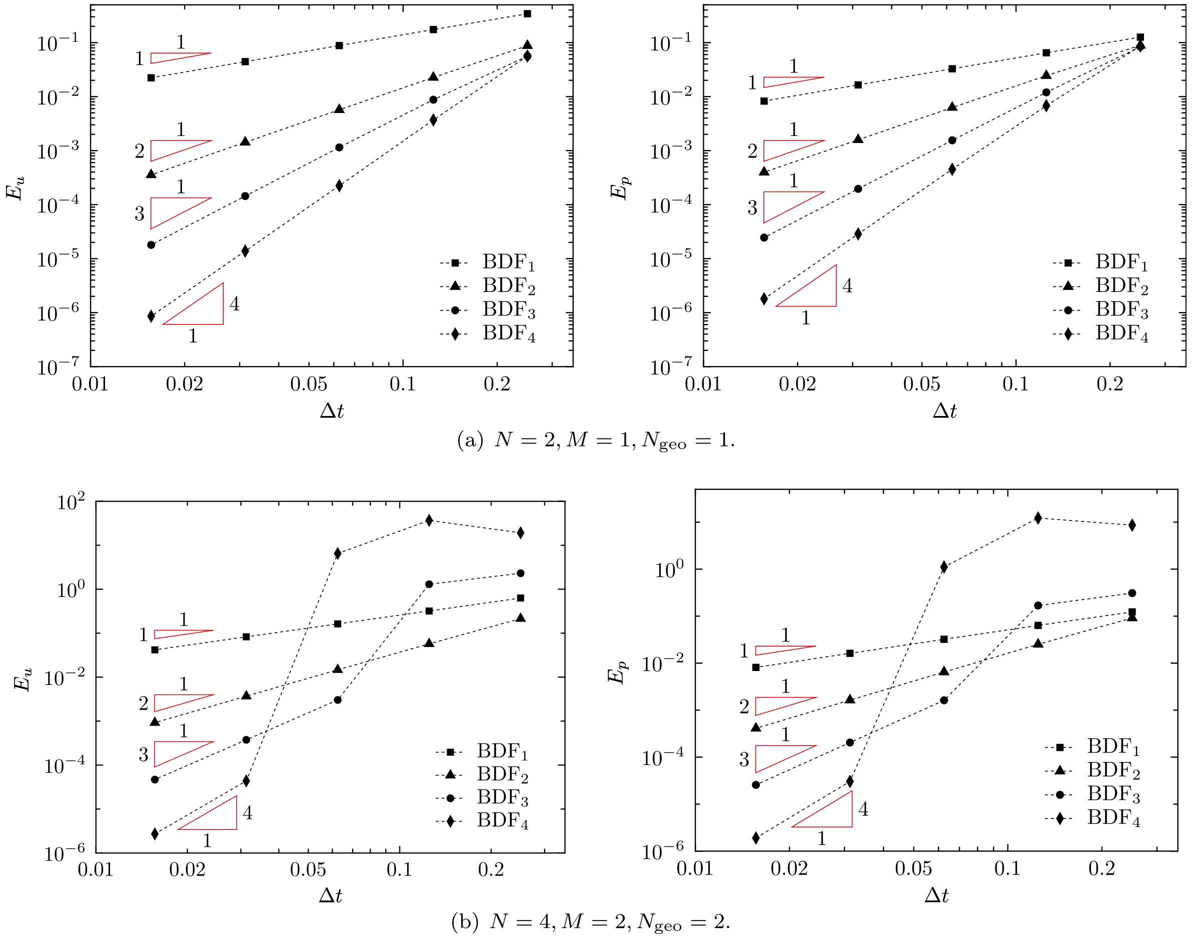
(a) $t = 1$.

(b) $t = 2$.

(c) $t = 3$.

(d) Zoom of the left lower corner of the mesh.

Fig. 14. Plot of the high order mesh used in visualizing the simulations (left) and the magnitude of the mesh velocity (right) at different time steps ($\nu = 10^{-3}$).



(a) $N = 2, M = 1, N_{\text{geo}} = 1$.

(b) $N = 4, M = 2, N_{\text{geo}} = 2$.

Fig. 15. Plot of the errors E_u and E_p for different choices of velocity–pressure spaces, geometrical elements and BDFq schemes.

$M = 1$ and $N_{\text{geo}} = 1$, see Fig. 15(a), and $N = 4$, $M = 2$ and $N_{\text{geo}} = 2$, see Fig. 15(b). We also consider different integration time strategies. We considered for this test $h = 0.5$, $\nu = 10^{-3}$, $\rho = 1$. We highlight that the flow is convection dominated (without the stabilization term, the method would not be stable) and we have stabilized the equations by the interior penalty term. We took $\gamma = 0.1$ in (19). These results were obtained by solving directly the linear system (25) with a LU factorization.

From Fig. 15 we confirm the expected convergence order for the proposed methods in time. Using a BDFq time integrator, a linear extrapolation of the convective term of the same order and an approximation of the mesh velocity also with a BDFq formula, the time discretization error is of the order of Δt^q, $q = 1, 2, 3, 4$. In Fig. 15, the convergence order of each scheme can be grasped from the slope of each curve. From the same figures, we observe that when the polynomial degree for the space approximation is increased, the stability constraint on the time-step for $q = 2, 3$ becomes more restrictive.

We remark that in Fig. 15(a) and (b), the numerical schemes used describe the solution of the problem exactly in space, though not the geometry. We also tested the above numerical schemes using a fourth order geometry. In this case, the orders of

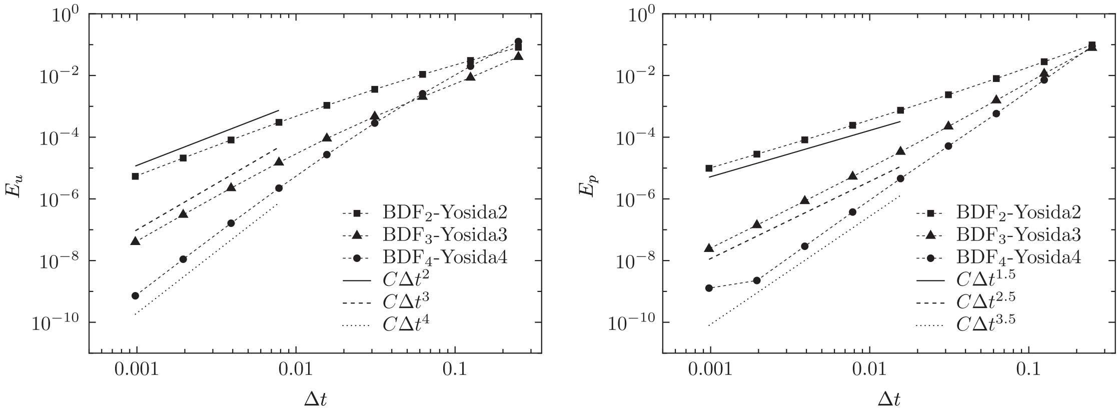


Fig. 16. Plot of the errors E_u and E_p for $N = 2$, $M = 1$, $N_{\text{geo}} = 1$ and different BDFq and Yosida-q schemes.

convergence in Δt are the same as the ones reported for the cases in Fig. 15(a) and (b), although the stability limitations on Δt are more severe for BDF3 and BDF4. Again, in this case, the BDF1 and BDF2 schemes remain stable.

4.3.2. Using the Yosida-q schemes

In the benchmark using Yosida-q schemes, we considered $h = 0.5$, $\nu = 0.05$, $\rho = 1$ and the stabilization parameter $\gamma = 0$. The error quantities E_u and E_p are plotted in Fig. 16. We notice that the convergence orders for the BDFq-Yosida-q agree with the rates predicted for the time-dependent Stokes equations in Gervasio [11].

Similar tests were conducted using $\nu = 10^{-3}$. The convergence orders for the pressure were slightly better in this case, for the range of Δt considered. This was due to the fact that the splitting error introduced by the Yosida-q schemes was much smaller than the error introduced by the corresponding BDF method. Regarding the stability of the methods, we did not observe considerable differences between using BDFq and Yosida-q schemes, for $q = 2, 3, 4$.

5. Conclusions

In this paper, we propose a numerical strategy to solve the unsteady incompressible Navier–Stokes equations defined in a domain that changes in time.

A full discretization scheme is presented, using the triangular spectral element method combined with Lagrange basis functions constructed on Fekete points and BDFq schemes to discretize the time derivative and the ALE mesh velocity. The non-linear convective term of the Navier–Stokes equations is linearized with a formula of the same order as the BDFq scheme. We propose a discrete ALE map that is able to describe curved boundaries, as long as the domain deformation is small. Its approximation properties in the moving boundary of the domain are of order $\mathcal{O}(h^{N_{\rm geo}+1})$ in the $L^2(\Omega)$-norm.

Regarding the full method, we concluded that if the mesh velocity is approximated with the same BDF scheme as the velocity and the extrapolation formula, we showed that the method converges with order $\mathcal{O}(\Delta t^q)$ when using a direct solver for the linear system (that appears at each time step) and it converges with order $\mathcal{O}(\Delta t^q)$ and $\mathcal{O}\left(\Delta t^{q-\frac{1}{2}}\right)$ in the velocity and pressure, respectively, when using the Yosida-q schemes.

Regarding our current work, we are (i) developing scalable solutions strategies for solving the Navier–Stokes equations in 2D/3D in moving domains, within the high order framework we presented in this paper and (ii) extending the high order ALE map construction to more general domains and deformations. In particular, the crucial step which corrects the high order geometrical nodes needs to be improved and generalized as well as the extension operator used to generate the first order ALE map. With respect to the latter, we are looking to other operators that overcome the limitations of the standard harmonic extension in the case for large deformations of the moving domain.

Acknowledgements

The first author was partially supported by Fundação para a Ciência e Tecnologia through grant SFRH/BD/22243/2005 of POCI2010/FEDER. The second author was supported by the ISLE/CHPID grant from the region Rhônes-Alpes and the PEPS/META-LIFE grant from the CNRS. We acknowledge the support provided by the European Research Council Advanced Grant ‘‘Mathcard, Mathematical Modelling and Simulation of the Cardiovascular System’’ Project ERC-2008-AdG 227058.

References

[1] M. Ainsworth, P. Coggins, A uniformly stable family of mixed hp-finite elements with continuous pressures for incompressible flow, IMA J. Numer. Anal. 22 (2) (2002) 307–327.

[2] C. Bernardi, Y. Maday, Uniform inf–sup conditions for the spectral discretization of the Stokes problem, Math. Models Methods Appl. Sci. 9 (3) (1999) 395–414.

[3] R. Bouffanais, Simulation of shear-driven flows: transition with a free surface and confined turbulence, Ph.D. thesis, no. 3837, EPF Lausanne, 2007.

[4] F. Brezzi, M. Fortin, Mixed and Hybrid Finite Element Methods, Springer-Verlag New York, Inc., New York, NY, USA, 1991.

[5] C. Canuto, M.Y. Hussaini, A. Quarteroni, T.A. Zang, Spectral Methods: Fundamentals in Single Domains, Springer-Verlag, New York, Berlin, 2006.

[6] C. Canuto, M.Y. Hussaini, A. Quarteroni, T.A. Zang, Spectral Methods: Evolution to Complex Geometries and Applications to Fluid Dynamics, Springer-Verlag, New York, Berlin, 2007.

[7] T.A. Davis, Direct Methods for Sparse Linear Systems (Fundamentals of Algorithms 2), Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2006.

[8] S. Deparis, M. Fernández, L. Formaggia, Acceleration of a fixed point algorithm for fluid-structure interaction using transpiration conditions, Math. Model. Numer. Anal. 37 (4) (2003) 601–616.

[9] J. Donea, S. Giuliani, J.P. Halleux, An arbitrary Lagrangian–Eulerian finite element method for transient dynamic fluid-structure interactions, Comput. Methods Appl. Mech. Engrg. 33 (1–3) (1982) 689–723.

[10] L. Formaggia, A. Quarteroni, A. Veneziani (Eds.), Cardiovascular Mathematics: Modeling and Simulation of the Circulatory System, MS & A, vol. 1, Springer, 2009.

[11] P. Gervasio, Convergence analysis of high order algebraic fractional step schemes for time-dependent Stokes equations, SIAM J. Numer. Anal. 46 (4) (2008) 1682–1703.

[12] P. Gervasio, F. Saleri, A. Veneziani, Algebraic fractional-step schemes with spectral methods for the incompressible Navier–Stokes equations, J. Comput. Phys. 214 (1) (2006) 347–365.

[13] C. Geuzaine, J.-F. Remacle, Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities, Int. J. Numer. Methods Engrg. 79 (11) (2009) 1309–1331.

[14] W.J. Gordon, C.A. Hall, Construction of curvilinear coordinate systems and their applications to mesh generation, Int. J. Numer. Methods Engrg. 7 (1973) 461–477.

[15] W.J. Gordon, C.A. Hall, Transfinite element methods: blending-function interpolation over arbitrary curved element domains, Numer. Math. 21 (1973) 109–129.

[16] J.L. Guermond, P. Minev, J. Shen, Error analysis of pressure-correction schemes for the time-dependent Stokes equations with open boundary conditions, SIAM J. Numer. Anal. 43 (1) (2005) 239–258.

[17] J.L. Guermond, J. Shen, A new class of truly consistent splitting schemes for incompressible flows, J. Comput. Phys. 192 (1) (2003) 262–276.

[18] J.L. Guermond, J. Shen, Velocity-correction projection methods for incompressible flows, SIAM J. Numer. Anal. 41 (1) (2003) 112–134.

[19] W. Heinrichs, Improved Lebesgue constants on the triangle, J. Comput. Phys. 207 (2) (2005) 625–638.

[20] J.S. Hesthaven, C.H. Teng, Stable spectral methods on tetrahedral elements, SIAM J. Sci. Comput. 21 (6) (2000) 2352–2380.

[21] L.W. Ho, E.M. Rønquist, Spectral element solution of steady incompressible viscous free-surface flows, Finite Elem. Anal. Des. 16 (3-4) (1994) 207–227.

[22] T.J.R. Hughes, W.K. Liu, T.K. Zimmermann, Lagrangian–Eulerian finite element formulation for incompressible viscous flows, Comput. Methods Appl. Mech. Engrg. 29 (3) (1981) 329–349.

[23] H. Kanchi, A. Masud, A 3D adaptive mesh moving scheme, Int. J. Numer. Methods Fluids 54 (6-8) (2007) 923–944.

[24] G.E. Karniadakis, S.J. Sherwin, Spectral/hp element methods for computational fluid dynamics, second ed., Oxford University Press, Oxford, 2004.

[25] G.E. Karniadakis, S.J. Sherwin, A new triangular and tetrahedral basis for high-order (hp) finite element methods, Int. J. Numer. Methods Engrg. 38 (22) (1995) 3775–3802.

[26] J. Peiró, S.J. Sherwin, Mesh generation in curvilinear domains using high-order elements, Int. J. Numer. Methods Engrg. 53 (2002) 107–223.

[27] S. Dey, R.M. O'Bara, M.S. Shephard, Curvilinear mesh generation in 3D, Comput. Aided Geom. Des. 33 (2001) 199–209.

[28] D. Kay, D. Loghin, A. Wathen, A preconditioner for the steady-state Navier–Stokes equations, SIAM J. Sci. Comput. 24 (1) (2002) 237–256.

[29] D. Kay, E. Lungu, A block preconditioner for high–order mixed finite element approximations to the Navier–Stokes equations, SIAM J. Sci. Comput. 27 (6) (2006) 1867–1880.

[30] F. Nobile. Numerical approximation of fluid-structure interaction problems with application to haemodynamics, Ph.D. thesis, no. 2458, EPF Lausanne, 2001.

[31] A.T. Patera, A spectral element methods for fluid dynamics: laminar flow in a channel expansion, J. Comput. Phys. 54 (1984) 468–488.

[32] G. Pena, Spectral element approximation of the incompressible Navier–Stokes equations evolving in a moving domain and applications, Ph.D. thesis,, no. 4529, École Polytechnique Fédérale de Lausanne, November 2009.

[33] G. Pena, C. Prud'homme, Construction of a high order fluid-structure interaction solver, JCAM 234 (7) (2010). <http://dx.doi.org/10.1016/j.cam.2009.08.093>.

[34] C. Prud'homme, A domain specific embedded language in C++for automatic differentiation, projection, integration and variational formulations, Sci. Program. 14 (2) (2006) 81–110. <http://iospress.metapress.com/link.asp?id=8xwd8r59hg1hmlcl>.

[35] C. Prud'homme. Life: overview of a unified C++ implementation of the finite and spectral element methods in 1D, 2D and 3D, in: Workshop On State-Of-The-Art In Scientific And Parallel Computing, Lecture Notes in Computer Science, p. 10. Springer-Verlag.

[36] C. Prud'homme. Life: a modern and unified c++implementation of finite-element and spectral-elements methods in 1d, 2d and 3d: overview and applications, in: ICIAM.

[37] C. Prud'homme, V. Chabannes, G. Pena, S. Veys, Feel++: A Computational Framework for Galerkin Methods, in preparation.

[38] A. Quarteroni, L. Formaggia, Mathematical modelling and numerical simulation of the cardiovascular system, in: Modelling of Living Systems, Handbook of Numerical Analysis Series (2003).

[39] A. Quarteroni, R. Sacco, F. Saleri, Numerical Mathematics (Texts in Applied Mathematics), Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[40] A. Quarteroni, F. Saleri, A. Veneziani, Analysis of the Yosida method for the Incompressible Navier–Stokes equations, J. Math. Pures Appl. 78 (5) (1999) 473–503.

[41] A. Quarteroni, F. Saleri, A. Veneziani, Factorization methods for the numerical approximation of Navier–Stokes equations, Comput. Methods Appl. Mech. Engrg. 188 (1–3) (2000) 505–526.

[42] A. Quarteroni, A. Valli, Numerical Approximation of Partial Differential Equations, Springer-Verlag, Berlin, Heidelberg, New York,, 1994.

[43] E.M. Rønquist. Optimal spectral element methods for the unsteady three-dimensional incompressible Navier–Stokes equations, Ph,D, thesis, Massachusetts Institute of Technology, 1988.

[44] M. Sala and M. Heroux. Robust algebraic preconditioners with IFPACK 3.0. Technical Report SAND-0662, Sandia National Laboratories, 2005.

[45] F. Saleri, A. Veneziani, Pressure correction algebraic splitting methods for the incompressible Navier–Stokes equations, SIAM J. Numer. Anal. 43 (1) (2005) 174–194.

[46] C. Schwab, M. Suri, Mixed hp finite element methods for Stokes and non-Newtonian flow, Comput. Methods Appl. Mech. Engrg. 175 (1999) 217–241.

[47] D. Silvester, H. Elman, D. Kay, A. Wathen, Efficient preconditioning of the linearized Navier–Stokes equations for incompressible flow, J. Comput. Appl. Math. 128 (1–2) (2001) 261–279.

[48] P. Solin, K. Segeth, I. Dolezel, Higher-Order Finite Element Methods, Chapman and Hall/CRC, Boca Raton, London, New York, 2004.

[49] K. Stanley. Klu: a "clark kent" sparse lu factorization algorithm for circuit matrices, in: 2004 SIAM Conference on Parallel Processing for Scientific Computing (PP04), 2004.

[50] R. Stenberg, M. Suri, Mixed hp finite element methods for problems in elasticity and Stokes flow, Numer. Math. 72 (3) (1996) 367–389.

[51] M.A. Taylor, B.A. Wingate, R.E. Vincent, An algorithm for computing Fekete points in the triangle, SIAM J. Numer. Anal. 38 (5) (2000) 1707–1720.

[52] T. Warburton, An explicit construction for interpolation nodes on the simplex, J. Engrg. Math. 56 (3) (2006) 247–262.

[53] C. Winkelmann, Interior penalty finite element approximation of Navier–Stokes equations and application to free surface flows, Ph.D. thesis, Lausanne, 2007.

[54] L.I.G. Kovasznay, Laminar flow behind a two-dimensional grid, Proc. Camb. Philos. Soc. 44 (1948) 58–62.

[55] E. Burman, A. Quarteroni, B. Stamm, Stabilization strategies for high order methods for transport dominated problems, Bollettino U.M.I. Series IX 1 (1) (2008) 57–77.

[56] E. Burman, A. Ern, Continuous interior penalty hp-finite element methods for advection and advection-diffusion equations, Math. Comp 76 (259) (2007) 119–1140 (EPFL-IACS report 02.2005).

[57] E. Burman, M.A. Fernandez, Continuous interior penalty finite element method for the time-dependent Navier–Stokes equations: space discretization and convergence, Numer. Math. 107 (1) (2007) 39–77.

[58] M. Braack, E. Burman, V. John, G. Lube, Stabilized finite element methods for the generalized Oseen problem, Comput. Methods Appl. Mech. Engrg. 196 (4–6) (2007) 853–866.

[59] L. Formaggia, F. Nobile, A stability analysis for the arbitrary Lagrangian Eulerian formulation with finite elements, East–West J. Numer. Math. 7 (2) (1999) 105–131.

[60] L. Formaggia, F. Nobile, Stability analysis of second-order time accurate schemes for ALE-FEM, Comput. Methods Appl. Mech. Engrg. 193 (39-41) (2004) 4097–4116.

[61] Th. Richter, Th. Wick, Finite elements for fluid-structure interaction in ALE and fully Eulerian coordinates, Comput. Methods Appl. Mech. Engrg. 199 (41–44) (2010) 2633–2642.

[62] C.M. Murea, S. Sy, A fast method for solving fluid-structure interaction problems numerically, Int. J. Numer. Methods Fluids 60 (2009) 1149–1172.

[63] A. Winslow, Numerical solution of the quasilinear Poisson equations in a nonuniform triangle mesh, J. Comput. Phys. 2 (1967) 149–172.