Comput. Methods Appl. Mech. Engrg. 197 (2008) 4216–4232

Contents lists available at ScienceDirect

Comput. Methods Appl. Mech. Engrg.

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

# Modular vs. non-modular preconditioners for fluid–structure systems with large added-mass effect

Santiago Badia [a,b,*], Annalisa Quaini [c], Alfio Quarteroni [c,d]

[a] CIMNE, Universitat Politècnica de Catalunya, Jordi Girona 1-3, Edifici C1, 08034 Barcelona, Spain
[b] Computational Mathematics and Algorithms, MS-1320, Sandia National Laboratories, Albuquerque, NM 87185-1320, United States
[c] IACS – Chair of Modeling and Scientific Computing, EPFL, CH-1015 Lausanne, Switzerland
[d] MOX – Dipartimento di Matematica "F. Brioschi" – Politecnico di Milano, Piazza Leonardo da Vinci 32, 20133 Milano, Italy

A R T I C L E  I N F O

*Article history:*
Received 4 February 2008
Received in revised form 12 April 2008
Accepted 21 April 2008
Available online 6 May 2008

*Keywords:*
Fluid–structure interaction
Partitioned procedures
Preconditioners
Semi-implicit coupling
Added-mass effect
Hemodynamics

A B S T R A C T

In this article we address the numerical simulation of fluid–structure interaction (FSI) problems featuring large added-mass effect. We analyze different preconditioners for the coupled system matrix obtained after space–time discretization and linearization of the FSI problem. The classical Dirichlet–Neumann preconditioner has the advantage of "modularity" because it allows to reuse existing fluid and structure codes with minimum effort (simple interface communication). Unfortunately, its performance is very poor in case of large added-mass effects. Alternatively, we consider two non-modular approaches. The first one consists in preconditioning the coupled system with a suitable diagonal scaling combined with an ILUT preconditioner. The system is then solved by a Krylov method. The drawback of this procedure is that the combination of fluid and structure codes to solve the coupled system is not straightforward. The second non-modular approach we consider is a splitting technique based on an inexact block-LU factorization of the linear FSI system. The resulting algorithm computes the fluid velocity separately from the coupled pressure–structure system at each iteration, reducing the computational cost. Independently of the preconditioner, the efficiency of semi-implicit algorithms (i.e., those that treat geometric and fluid nonlinearities in an explicit way) is highlighted and their performance compared to the one of implicit algorithms. All the methods are tested on three-dimensional blood-vessel systems. The algorithm combining the non-modular ILUT preconditioner with Krylov methods proved to be the fastest.

## 1. Introduction

In coupled fluid–structure systems, the fluid acts over the structure as an extra mass (usually called added-mass [27]) at the interface. The importance of the extra inertia term appearing in the structure equation increases with the quotient $\rho_f/\rho_s$, where $\rho_f$ and $\rho_s$ are the fluid and the structure density, respectively. Therefore, when the structure density is much bigger than the fluid one, the added-mass effect is almost negligible. However, some problems involve a fluid and a structure whose densities are of the same order of magnitude. We focus on those cases, in which the added-mass effect becomes important.

Fluid–structure interaction problems are usually solved via partitioned procedures, stemming from a domain decomposition viewpoint. These algorithms consist in the evaluation of independent fluid and structure problems, coupled via transmission conditions in an iterative fashion. The Dirichlet–Neumann (DN) algorithm is one of the most popular partitioned procedures in FSI. A Dirichlet boundary condition (continuity of velocities) is imposed at the interface for the fluid sub-problem, whereas the structure sub-problem is supplemented with Neumann boundary conditions (continuity of stresses). The DN-algorithm iterates over these two problems until convergence. These are Richardson (also called fixed point) iterations on the interface displacement and they are denoted as coupling iterations.

Fluid–structure algorithms were initially developed for aeroelastic applications, where typically $\rho_s \gg \rho_f$. In this case, the classical DN-algorithm (that we will denote by DN–Richardson) converges in a few iterations. Thus, it is common practice in computational aeroelasticity to use an explicit treatment of the coupling, that is only one coupling iteration is performed per time step (see, e.g., [30,15]). Unfortunately, the convergence properties of the DN–Richardson algorithm depend heavily on the added-mass effect. In fact, when the density of the structure is comparable to the fluid one, the method fails to converge (see, e.g., [28,35]). In order to enforce convergence, relaxation is needed [26]. The relaxation parameter diminishes as the added-mass effect

* Corresponding author. Address: CIMNE, Universitat Politècnica de Catalunya, Jordi Girona 1-3, Edifici C1, 08034 Barcelona, Spain.
*E-mail addresses:* sbadia@cimne.upc.edu (S. Badia), annalisa.quaini@epfl.ch (A. Quaini), alfio.quarteroni@epfl.ch (A. Quarteroni).

0045-7825/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.cma.2008.04.018

increases and it might become so small that convergence is reached extremely slowly [11].

Many interesting applications are located in the large added-mass effect range, like most of FSI problems involving light and thin-walled structures (e.g., sail–wind systems or airbags). In particular, we are interested in the simulation of the deformation of the arterial walls, whose density is almost identical to the blood one, in the circulatory system.

Despite its inefficiency in case of a large added-mass effect, the DN–Richardson algorithm has still been used. The reason relies on its *modularity*. A FSI algorithm that only requires interface data transfer between the two codes, without any modification of the sources, is called modular. A modular FSI algorithm allows to reuse existing (and already optimized) fluid and structure codes. For the DN–Richardson method, the interface communication consists in the transfer of the proper boundary conditions on the interface. A simple master code can perform the communication between codes only by handling interface information. Furthermore, Neumann and Dirichlet boundary conditions are input data for any fluid and structure solver and no change needs to be made on these two solvers.

Since the nineties, many works have been focused on the development of FSI algorithms capable of improving the convergence velocity of modular algorithms. Some of them suggested the use of dynamic evaluations of the relaxation parameters based on line-search techniques, like steepest descent or Aitken acceleration (see e.g. [26]). In this minimization approach, robust Krylov methods have replaced Richardson iterations in [21,17,25]. Other works proposed to diminish the computational cost by reducing the coupled fluid–structure problem to a pressure–structure problem, using the continuous projection method [16] or algebraic block-LU factorizations [5]. A third approach consists in modifying the boundary conditions at the interface. The Neumann–Dirichlet method has even worse convergence properties than the DN one. The Neumann–Neumann algorithm slightly reduces the number of iterations, but every iteration is more expensive, making its efficiency similar to the one of the DN (see [13]). Recently, two improved partitioned procedures have been designed: one sets Robin boundary conditions on the interface [4], while the other enforces the continuity of velocities in a weak way by applying Nitsche's method and introducing a penalization term that acts as a pseudo-compressibility [10].

In [18] a simplified monolithic FSI algorithm embedding the structure into the fluid problem has been proposed. There, the $(d-1)$-dimensional ($d$ being the space dimension) structure is modeled as a membrane. The same idea of writing the FSI problem only in terms of fluid unknowns is presented in [29], where an algebraic law for approximating the structure problem is employed. In this paper, we consider the elasticity equation for a $d$-dimensional structure. In any case, the use of non-modular preconditioners for the FSI system has received much less attention. The first reason is the fact that they are not needed in applications with a negligible added-mass effect because partitioned procedures are very efficient. The second reason is the loss of modularity. Existing fluid and structure codes can still be reused, but the coupling of the codes is more involved than bare interface communication. In fact, fluid and structure matrices must be stored in a unique FSI matrix, which has to be accessed to compute the preconditioner. However, as we show in the present work, non-modular algorithms should not be dismissed. We claim the efficiency of non-modular preconditioners for problems affected by a large added-mass effect.

The basic aspects of our non-modular approach are the use of fluid and structure problems in terms of velocities, the use of a single finite element partition for the whole domain and the use of the same velocity finite element space for fluid and structure problems (that can easily be attained by using stabilization techniques). In this frame, the continuity of velocities is straightforward and the continuity of stresses is imposed weakly.

The solution of the monolithic system is preconditioned in two steps. Since fluid and structure entries are not of the same order, we first apply a suitable scaling of the FSI system. In a second step, the scaled system is preconditioned by an incomplete LU factorization (the ILUT preconditioner). The preconditioned FSI system is solved using a Krylov method, e.g. GMRES or BiCGStab. We denote this combination by ILUT-GMRES and ILUT-BiCGStab, respectively. Every iteration of the Krylov method requires to solve a linear system with the preconditioner as system matrix. The solution of this systems is very simple and cheap thanks to the ILU structure of the preconditioner. This is a main difference with respect to the DN preconditioner, where the solution of the system with the preconditioner as system matrix involves expensive fluid and structure evaluations. Thus, for an equal number of outer Krylov method iterations, the non-modular approach is much faster.

The third method we consider is a non-modular FSI algorithm based on the reduction of the FSI system to a coupled pressure–structure system. This method, denoted by the name PIC (pressure-interface correction), originates from an inexact block-LU factorization of the FSI system (see [5]). The approximation introduced by the inexact factors perturbs the system but does not spoil the accuracy when using first order algorithms.

All the approaches listed above aim at reducing the computational cost by abating coupling iterations. Another way to serve the same purpose is to reduce the nonlinear iterations by adopting more efficient linearization techniques. The use of a full Newton algorithm is suggested in [17]. Even though the Newton method reduces the number of nonlinear iterations, every iteration is more expensive because shape derivative evaluations are needed, making the implementation complicated. Quasi-Newton algorithms have been suggested in [20,24,21]. In this paper, we insist on the idea of considering the nonlinearity in an explicit way, leading to semi-implicit algorithms. Semi-implicit procedures do not endanger the stability whereas the computational cost is drastically reduced (no nonlinear iterations are performed). The accuracy of semi-implicit algorithms has already been studied in [5].

Let us give the outline of the article. In Section 2, we state the mathematical formulation of the FSI problem and detail the choices of our monolithic approach. In Section 3, we recall the DN–Richardson and DN-GMRES algorithms. In Section 4, we introduce the ILUT preconditioner for the monolithic system. Section 5 is devoted to an inexact block-LU factorization of the FSI system. In Sections 6 and 7, we carry out a set of numerical experiments. Then, in Section 8, we draw some important conclusions on the optimal range of applicability of the methods here proposed.

## 2. Problem setting

Consider an heterogeneous mechanical system which covers a bounded, polyhedral and moving domain $\Omega_t \subset \mathbb{R}^d$ ($d = 2, 3$, being the space dimension), where time $t$ spans the interval of analysis $[0, T]$. This domain is divided into a domain $\Omega_t^s$ occupied by a solid structure and its complement $\Omega_t^f$ occupied by the fluid. The fluid–structure interface $\Sigma_t$ is the common boundary between $\Omega_t^f$ and $\Omega_t^s$, i.e. $\Sigma_t = \partial\Omega_t^f \cap \partial\Omega_t^s$. Furthermore, $\boldsymbol{n}_\mathrm{f}$ is the outward normal of $\Omega_t^f$ on $\Sigma_t$ and $\boldsymbol{n}_\mathrm{s}$ is its counterpart for the structure domain. The initial configuration $\Omega_0$ at $t = 0$ is considered as the reference one. In particular, we assume an incompressible and Newtonian fluid and an elastic structure.

The fluid problem is governed by the incompressible Navier–Stokes equations:

$$\delta_t \boldsymbol{u} + \boldsymbol{u}\cdot\nabla\boldsymbol{u} - \frac{1}{\rho_f}\nabla\cdot\boldsymbol{\sigma}^f = \boldsymbol{f}_f \quad \text{in } \Omega_t^f \times (0,T),$$
$$\nabla\cdot\boldsymbol{u} = 0 \quad \text{in } \Omega_t^f \times (0,T),$$

where $\boldsymbol{u}$ is the fluid velocity, $\boldsymbol{\sigma}^f$ the Cauchy stress tensor and $\boldsymbol{f}_f$ the body force. For Newtonian fluids, $\boldsymbol{\sigma}^f$ has the following expression:

$$\boldsymbol{\sigma}^f(\boldsymbol{u},p) = -p\mathbf{I} + 2\mu\epsilon(\boldsymbol{u}),$$

where $p$ is the pressure, $\mu$ is the fluid viscosity, and

$$\epsilon(\boldsymbol{u}) = \frac{1}{2}(\nabla\boldsymbol{u} + (\nabla\boldsymbol{u})^{\mathrm{T}})$$

is the strain rate tensor, with $\nabla$ denoting the spatial gradient operator.

The structure is governed by the elastodynamics equation

$$\partial_t^2\boldsymbol{\eta} - \frac{1}{\rho_s}\nabla\cdot\boldsymbol{\sigma}^s = \boldsymbol{f}_s \quad \text{in } \Omega_t^s \times (0,T),$$

where $\boldsymbol{\eta}$ is the structure displacement and $\boldsymbol{f}_s$ the body force. This equation must be supplemented with a constitutive law that relates the structural displacement $\boldsymbol{\eta}$ and the Cauchy stress tensor $\boldsymbol{\sigma}^s$. As a simple example, in our numerical simulations we used the linear Saint-Venant Kirchhoff three-dimensional elastic model, where the solid stress is defined as:

$$\boldsymbol{\sigma}^s(\boldsymbol{\eta}) = 2\mu_\ell\epsilon(\boldsymbol{\eta}) + \lambda_\ell(\nabla\cdot\boldsymbol{\eta})\mathbf{I},$$

Here, $\epsilon(\boldsymbol{\eta}) = (\nabla\boldsymbol{\eta} + (\nabla\boldsymbol{\eta})^{\mathrm{T}})/2, \mu_\ell$ and $\lambda_\ell$ are the Lamé constants. Of course, other structure models can be chosen according to the specific problem under consideration.

These two problems are coupled on the interface by two transmission conditions. Due to the fact that we are dealing with viscous fluids, the continuity of velocities (normal and tangential)

$$\boldsymbol{u} = \partial_t\boldsymbol{\eta} \quad \text{on } \Sigma_t \times (0,T)$$

must be satisfied. On the other hand, the continuity of stresses

$$\boldsymbol{\sigma}^s\cdot\boldsymbol{n}_s + \boldsymbol{\sigma}^f\cdot\boldsymbol{n}_f = 0 \quad \text{on } \Sigma_t \times (0,T)$$

must hold, due to the action–reaction principle.

In order to describe the evolution of the whole domain $\Omega_t$, we define two families of mappings:

$$\mathscr{L} : \Omega_0^s \times [0,T] \to \Omega_t^s, (\boldsymbol{x}_0, t) \to \boldsymbol{x} = \mathscr{L}(\boldsymbol{x}_0, t) \tag{1}$$

and

$$\mathscr{A} : \Omega_0^f \times [0,T] \to \Omega_t^f, (\boldsymbol{x}_0, t) \to \boldsymbol{x} = \mathscr{A}(\boldsymbol{x}_0, t). \tag{2}$$

The map $\mathscr{L}_t = \mathscr{L}(\cdot,t)$ tracks the solid domain in time, $\mathscr{A}_t = \mathscr{A}(\cdot,t)$ the fluid domain and they must agree on $\Sigma_t$:

$$\mathscr{L}_t = \mathscr{A}_t \quad \text{on } \Sigma_t, \tag{3}$$

in order to define an homeomorphism over $\Omega_t$.

We adopt a purely Lagrangian approach for the structure. Thus, if $\hat{\boldsymbol{\eta}}$ denotes the displacement of the solid medium evaluated at the reference configuration, then:

$$\mathscr{L}_t(\boldsymbol{x}_0) = \boldsymbol{x}_0 + \hat{\boldsymbol{\eta}}(\boldsymbol{x}_0, t).$$

Apart from (3), the fluid domain mapping $\mathscr{A}_t$ is arbitrary. This mapping can be defined as an appropriate extension operator of its value on the interface:

$$\mathscr{A}_t(\boldsymbol{x}_0) = \boldsymbol{x}_0 + \mathrm{Ext}(\hat{\boldsymbol{\eta}}(\boldsymbol{x}_0, t)|_{\Sigma_0}).$$

A classical choice is to consider a harmonic extension in the reference domain. $\mathscr{A}_t$ is called the *Arbitrary Lagrangian–Eulerian* (ALE) mapping, since in general it does not track the fluid particles (in that case the formulation would be purely Lagrangian).

We can now write the velocity ALE time derivative (for the fluid):

$$\delta_t\boldsymbol{u}|_{\boldsymbol{x}_0} = \partial_t\boldsymbol{u} + \boldsymbol{w}\cdot\nabla\boldsymbol{u},$$

which is the variation of the velocity for a particle that moves with the fluid mapping $\mathscr{A}_t$. The domain velocity $\boldsymbol{w}$ is calculated using the following expression:

$$\boldsymbol{w}(\boldsymbol{x}, t) = \delta_t\boldsymbol{x}|_{\boldsymbol{x}_0} = \delta_t\mathscr{A}_t \circ \mathscr{A}_t^{-1}(\boldsymbol{x}).$$

The fluid–structure problem (where the fluid problem is stated in the ALE formulation) in its strong form reads as follows:

(i) *Geometry problem*: Find the fluid domain displacement:

$$\mathscr{A}_t(\boldsymbol{x}_0) = \boldsymbol{x}_0 + \mathrm{Ext}(\hat{\boldsymbol{\eta}}|_{\Sigma_0}),$$
$$\boldsymbol{w} = \partial_t\mathscr{A}_t \circ \mathscr{A}_t^{-1}, \quad \Omega_t^f = \mathscr{A}_t(\Omega_0^f). \tag{4}$$

(ii) *Fluid–structure problem*: Find velocity $\boldsymbol{u}$, pressure $p$ and displacement $\boldsymbol{\eta}$ such that

$$\begin{aligned}
&\delta_t\boldsymbol{u}|_{\boldsymbol{x}_0} + (\boldsymbol{u}-\boldsymbol{w})\cdot\nabla\boldsymbol{u} - \frac{1}{\rho_f}\nabla\cdot\boldsymbol{\sigma}^f = \boldsymbol{f}_f \quad \text{in } \Omega_t^f \times (0,T),\\
&\nabla\cdot\boldsymbol{u} = 0 \quad \text{in } \Omega_t^f \times (0,T),\\
&\partial_t^2\boldsymbol{\eta} - \frac{1}{\rho_s}\nabla\cdot\boldsymbol{\sigma}^s = \boldsymbol{f}_s \quad \text{in } \Omega_t^s \times (0,T),\\
&\boldsymbol{u} = \partial_t\boldsymbol{\eta} \quad \text{on } \Sigma_t \times (0,T),\\
&\boldsymbol{\sigma}^s\cdot\boldsymbol{n}_s + \boldsymbol{\sigma}^f\cdot\boldsymbol{n}_f = 0 \quad \text{on } \Sigma_t \times (0,T).
\end{aligned} \tag{5}$$

## 2.1. Weak formulation

For the variational formulation of the fluid–structure problem (4) and (5), we indicate with $L^2(\Omega)$ the space of square integrable functions in a spatial domain $\Omega$ and with $H^1(\Omega)$ the space of functions in $L^2(\Omega)$ with first derivatives in $L^2(\Omega)$. We use $(\cdot,\cdot)_\Omega$ and $\langle\cdot,\cdot\rangle_\Omega$ to denote the $L^2$ product and a duality pair in $\Omega$, respectively.

Let us define the following spaces, for any given $t \in [0,T)$:

$$\begin{aligned}
&V^f(t) := \{\boldsymbol{v} : \Omega_t^f \to \mathbb{R}^d, \boldsymbol{v} = \hat{\boldsymbol{v}} \circ (\mathscr{A}_t)^{-1}, \hat{\boldsymbol{v}} \in (H^1(\Omega_0^f))^d\},\\
&V_0^f(t) := \{\boldsymbol{v} \in V^f(t), \boldsymbol{v}|_{\Sigma_t} = \boldsymbol{0}\},\\
&Q(t) := \{q : \Omega_t^f \to \mathbb{R}, q = \hat{q} \circ (\mathscr{A}_t)^{-1}, \hat{q} \in L^2(\Omega_0^f)\},\\
&\widehat{V}^s := \{\hat{\boldsymbol{v}} : \Omega_0^s \to \mathbb{R}^d, \hat{\boldsymbol{v}} \in (H^1(\Omega_0^s))^d\}.
\end{aligned}$$

$\mathscr{A}_t^{-1}$ is assumed Lipschitz continuous in order for $V^f(t) \subset (H^1(\Omega_t^f))^d$ and $Q(t) \subset L^2(\Omega_t^f)$. The variational formulation of the fluid–structure problem is: given $t \in (0,T)$, find $(\boldsymbol{u}, p, \hat{\boldsymbol{\eta}}) \in V^f(t) \times Q(t) \times \widehat{V}^s$ such that

$$\rho_f(\delta_t\boldsymbol{u}|_{\boldsymbol{x}_0}, \boldsymbol{v}_0^f)_{\Omega_t^f} + \mathscr{N}(\boldsymbol{u}-\boldsymbol{w}; \boldsymbol{u}, p, \boldsymbol{v}_0^f, q)_{\Omega_t^f} = \langle\boldsymbol{f}_f, \boldsymbol{v}_0^f\rangle_{\Omega_t^f}, \tag{6a}$$

$$\rho_s(\partial_{tt}\hat{\boldsymbol{\eta}}, \hat{\boldsymbol{v}}^s)_{\Omega_0^s} + \langle\hat{\boldsymbol{\sigma}}^s, \nabla\hat{\boldsymbol{v}}^s\rangle_{\Omega_0^s} = \langle\hat{\boldsymbol{f}}_s, \hat{\boldsymbol{v}}^s\rangle_{\Omega_0^s} - \langle\boldsymbol{\sigma}^f\cdot\boldsymbol{n}_f, \boldsymbol{v}_s\rangle_{\Sigma_t}, \tag{6b}$$

$$\boldsymbol{u} = \partial_t\hat{\boldsymbol{\eta}} \circ (\mathscr{A}_t)^{-1} \quad \text{on } \Sigma_t, \tag{6c}$$

for all $(\boldsymbol{v}_0^f, q, \hat{\boldsymbol{v}}^s) \in V_0^f(t) \times Q(t) \times \widehat{V}^s$, where

$$\mathscr{N}(\boldsymbol{a}; \boldsymbol{u}, p, \boldsymbol{v}, q)_\Omega = 2\mu(\epsilon(\boldsymbol{u}), \epsilon(\boldsymbol{v}))_\Omega + \rho_f\int_\Omega(\boldsymbol{a}\cdot\nabla\boldsymbol{u})\cdot\boldsymbol{v}\,\mathrm{d}\Omega - (p, \nabla\cdot\boldsymbol{v})_\Omega + (\nabla\cdot\boldsymbol{u}, q)_\Omega.$$

The continuity of velocities has been enforced in a strong way by (6c). On the contrary, the continuity of stresses on the interface is satisfied in a weak way by choosing test functions $\boldsymbol{v}^f \in V^f(t)$ for the momentum conservation equation of the fluid problem. In fact, the fluid interface load can be seen as the variational residual of the weak form of the momentum conservation equation for test functions that do not vanish on $\Sigma_t$:

$$\langle \boldsymbol{\sigma}^f \cdot \boldsymbol{n}_{\mathrm{f}}, \boldsymbol{v}^f \rangle_{\Sigma_t} = \rho_{\mathrm{f}}(\delta_t \boldsymbol{u}|_{\boldsymbol{x}_0}, \boldsymbol{v}^f)_{\Omega_t^f} + \mathscr{N}(\boldsymbol{u}-\boldsymbol{w}; \boldsymbol{u}, p, \boldsymbol{v}^f, q)_{\Omega_t^f} - \langle \boldsymbol{f}_{\mathrm{f}}, \boldsymbol{v}^f \rangle_{\Omega_t^f} = -\langle \mathscr{R}(\boldsymbol{u}, p), \boldsymbol{v}^f \rangle_{\Omega_t^f}.$$

Therefore, for the last term in Eq. (6b) we have the following equality:

$$\langle \boldsymbol{\sigma}^f \cdot \boldsymbol{n}_{\mathrm{f}}, \boldsymbol{v}^s \rangle_{\Sigma_t} = -\langle \mathscr{R}(\boldsymbol{u}, p), \mathscr{E}_t(\boldsymbol{v}^s|_{\Sigma_t}) \rangle_{\Omega_t^f}$$

for all $\boldsymbol{v}^s \in V^s(t)$ (where, abusing of notation, $V^s(t) = \mathscr{L}_t(\widehat{V}^s)$), $\mathscr{E}_t$ being an arbitrary extension operator from the trace space associated to $V^s(t)$ to $V^f(t)$.

The weak transmission of the fluid loads at the interface is crucial when carrying out stability and convergence analysis. We refer to [9] for an insight on the improved rates of convergence obtained by using weak definition of fluxes. Stability results for the FSI problem relying on this weak transmission can be found in [35,28].

### 2.2. The fully discrete problem: space and time discretization

Let $\widehat{V}_h^f \subset [H^1(\Omega_0^f)]^d, \widehat{V}_{0,h}^f \subset [H_0^1(\Omega_0^f)]^d, \hat{Q}_h^f \subset L^2(\Omega_0^f)$ and $\widehat{V}_h^s \subset [H^1(\Omega_0^s)]^{d-1}$ be the finite element spaces approximating $V^f, V_0^f, Q$ and $\widehat{V}^s$ at the reference configuration, respectively. With an abuse of notation, we can define the finite element spaces for a given time step $t^n$ using the domain maps (1) and (2), e.g. $V_h^f(t^n) = \mathscr{A}_{t^n}(\widehat{V}_h^f)$.

The standard Galerkin approximation of the incompressible Navier–Stokes equations may fail for two different reasons. First, the method exhibits instabilities when the convective term is dominant. On the other hand, pressure stability can only be obtained for velocity–pressure finite element spaces $(Q_h^f, V_h^f)$ that satisfy a discrete *inf–sup* condition (see [7]). The simplest combinations of velocity–pressure pairs (e.g. equal order nodal interpolation) do not satisfy this condition and are unstable.

Both pitfalls can be overcome by resorting to a *stabilized* formulation. In this work, we consider the *orthogonal subgrid scales* (OSS) technique proposed by Codina in [12]. It allows to use equal order velocity–pressure pairs (like the $\mathbb{P}_1 - \mathbb{P}_1$ pair adopted in this work) and stabilizes the convective term for high Reynolds numbers. We refer to [2] for the numerical analysis of the OSS technique in the ALE framework. The stabilized version of the fluid problem is obtained by using the form

$$\mathscr{N}_{\mathrm{s}}(\boldsymbol{a}_h; \boldsymbol{u}_h, p_h, \boldsymbol{v}_h, q_h)_\Omega := \mathscr{N}(\boldsymbol{a}_h; \boldsymbol{u}_h, p_h, \boldsymbol{v}_h, q_h)_\Omega + \mathscr{S}(\boldsymbol{a}_h; \boldsymbol{u}_h, p_h, \boldsymbol{v}_h, q_h)_\Omega,$$

where the perturbation term introduced by OSS (in its quasi-static form) reads

$$\mathscr{S}(\boldsymbol{a}_h; \boldsymbol{u}_h, p_h, \boldsymbol{v}_h, q_h)_\Omega = (\tau_1 \Pi^\perp(\boldsymbol{a}_h \cdot \nabla \boldsymbol{u}_h + \nabla p_h), \boldsymbol{a}_h \cdot \nabla \boldsymbol{v}_h + \nabla q_h)_\Omega + (\tau_2 \Pi^\perp(\nabla \cdot \boldsymbol{u}_h), \nabla \cdot \boldsymbol{v}_h)_\Omega. \quad (7)$$

Here, $\Pi^\perp(\cdot)$ is the $L^2$ orthogonal projection onto the finite element space, i. e.:

$$\Pi^\perp(\cdot) = \mathscr{I}(\cdot) - \Pi(\cdot)$$

where $\Pi(\cdot)$ is the $L^2$ projection onto the finite element space and $\mathscr{I}(\cdot)$ the identity operator. We use the following expressions for the stabilization parameters

$$\tau_1 = \left[ c_1 \frac{\mu}{\rho h^2} + c_2 \frac{|\boldsymbol{a}_h|}{h} \right]^{-1}, \quad \tau_2 = \frac{h^2}{c_1 \tau_1},$$

where $c_1$ and $c_2$ are appropriate constants, justified in [12] through a Fourier analysis. We refer to [12] for a thorough description of this stabilization technique.

With regard to time discretization, we have considered the backward Euler scheme for the fluid equations and the mid-point rule for the structure [35] for simplicity. In any case, the splitting methods suggested below can be easily extended to other time integration schemes. By defining the backward Euler operator $\delta_t$ as $\delta_t f^{n+1} = (f^{n+1} - f^n)/\delta t$ and denoting by $\mathrm{Ext}_h(\cdot)$ a discretized version of the extension operator $\mathrm{Ext}(\cdot)$, at each time level $t^{n+1}$, the fully discretized fluid–structure problem reads:

(i) *Geometry problem*: Find the fluid domain displacement

$$\mathscr{A}_{t^{n+1}}(\boldsymbol{x}_0) = \boldsymbol{x}_0 + \mathrm{Ext}_h(\hat{\boldsymbol{\eta}}_h^{n+1}|_{\Sigma_0}), \quad \boldsymbol{w}_h^{n+1} = \delta_t \mathscr{A}_{t^{n+1}} \circ \mathscr{A}_{t^{n+1}}^{-1}, \Omega_{t^{n+1}}^f = \mathscr{A}_{t^{n+1}}(\Omega_0^f). \quad (8)$$

(ii) *Fluid–structure problem*: Find $(\boldsymbol{u}_h^{n+1}, p_h^{n+1}, \hat{\boldsymbol{\eta}}_h^{n+1}) \in V_h^f \times Q_h \times \hat{V}_h^s$ such that

$$\rho_{\mathrm{f}}(\delta_t \boldsymbol{u}_h^{n+1}|_{\boldsymbol{x}_0}, \boldsymbol{v}_h^f)_{\Omega_{t^{n+1}}^f} + \mathscr{N}_{\mathrm{s}}(\boldsymbol{u}_h^{n+1} - \boldsymbol{w}_h^{n+1}; \boldsymbol{u}_h^{n+1}, p_h^{n+1}, \boldsymbol{v}_h^f, q_h)_{\Omega_{t^{n+1}}^f} = \langle \boldsymbol{f}_f^{n+1}, \boldsymbol{v}_h^f \rangle_{\Omega_{t^{n+1}}^f} \quad (9a)$$

$$\rho_{\mathrm{s}}\left(\frac{\dot{\hat{\boldsymbol{\eta}}}_h^{n+1} - \dot{\hat{\boldsymbol{\eta}}}_h^n}{\delta t}, \hat{\boldsymbol{v}}_h^s\right)_{\Omega_0^s} + \left\langle \boldsymbol{\sigma}^s\left(\frac{\hat{\boldsymbol{\eta}}_h^{n+1} + \hat{\boldsymbol{\eta}}_h^n}{2}\right), \nabla \cdot \hat{\boldsymbol{v}}_h^s \right\rangle_{\Omega_0^s} = \langle \boldsymbol{f}_{\mathrm{s}}^{n+1}, \hat{\boldsymbol{v}}_h^s \rangle_{\Omega_0^s} - \langle \mathscr{R}(\boldsymbol{u}_h^{n+1}, p_h^{n+1}), \mathscr{E}_h(\boldsymbol{v}_h^s|_{\Sigma_t}) \rangle_{\Omega_{t^{n+1}}^f}, \quad (9b)$$

$$\left(\frac{\dot{\hat{\boldsymbol{\eta}}}_h^{n+1} + \dot{\hat{\boldsymbol{\eta}}}_h^n}{2}, \hat{\boldsymbol{v}}_h^s\right)_{\Omega_0^s} = \left(\frac{\hat{\boldsymbol{\eta}}_h^{n+1} - \hat{\boldsymbol{\eta}}_h^n}{\delta t}, \hat{\boldsymbol{v}}_h^s\right)_{\Omega_0^s}, \quad (9c)$$

$$\boldsymbol{u}_h^{n+1} = \delta_t \dot{\hat{\boldsymbol{\eta}}}_h^{n+1} \circ \mathscr{A}_{t^{n+1}}^{-1} \quad \text{on } \Sigma_t \quad (9d)$$

for all $(\boldsymbol{v}_h^f, q_h, \hat{\boldsymbol{v}}_h^s) \in V_{0,h}^f \times Q_h \times \widehat{V}_h^s$.

The fluid domain $\Omega_{t^{n+1}}^f$ defined by $\mathscr{A}_{t^{n+1}}$ does depend on $\hat{\boldsymbol{\eta}}_h^{n+1}$ and the fluid problem depends on $\Omega_{t^{n+1}}^f$ in a nonlinear way. We consider a *fixed point algorithm* to linearize both this dependence and the convective term in (9a). The linearization of the fluid–structure problem (8) and (9) by the fixed point algorithm consists of: given the predictions $\tilde{\boldsymbol{\eta}}_h^{n+1}$ and $\tilde{\boldsymbol{u}}_h^{n+1}$

Step 1: Calculate the fluid domain displacement as in (8) but replacing the first equation with

$$\mathscr{A}_{t^{n+1}}(\boldsymbol{x}_0) = \boldsymbol{x}_0 + \mathrm{Ext}_h(\tilde{\boldsymbol{\eta}}_h^{n+1}|_{\Sigma_0}).$$

Step 2: Solve the fluid–structure problem as in (9) but replacing the momentum Eq. (9a) by the linearized version:

$$\rho_{\mathrm{f}}(\delta_t \boldsymbol{u}_h^{n+1}|_{\boldsymbol{x}_0}, \boldsymbol{v}_h^f)_{\Omega_{t^{n+1}}^f} + \mathscr{N}_{\mathrm{s}}(\tilde{\boldsymbol{u}}_h^{n+1} - \boldsymbol{w}_h^{n+1}; \boldsymbol{u}_h^{n+1}, p_h^{n+1}, \boldsymbol{v}_h^f, q_h)_{\Omega_{t^{n+1}}^f} = \langle \boldsymbol{f}_{\mathrm{f}}^{n+1}, \boldsymbol{v}_h^f \rangle_{\Omega_{t^{n+1}}^f}. \quad (10)$$

Step 3: Check the stopping criterion. If it is not satisfied, update $\tilde{\boldsymbol{\eta}}_h^{n+1} = \hat{\boldsymbol{\eta}}_h^{n+1}, \tilde{\boldsymbol{u}}_h^{n+1} = \boldsymbol{u}_h^{n+1}$ and go to Step 1.

We have ended up with a *fully discretized and linearized fluid–structure problem* that can be solved by a linear solver. Notice that the fluid and structure problems are strongly coupled: the fluid solution depends on $\hat{\boldsymbol{\eta}}_h^{n+1}$ through (9d), whereas to solve the structure problem in (9b) $\boldsymbol{u}_h^{n+1}$ and $p_h^{n+1}$ are needed.

A method that deals with the fluid–structure coupling in an explicit way replaces (9d) by the condition $\boldsymbol{u}_h^{n+1} = \delta_t \tilde{\boldsymbol{\eta}}_h^{n+1} \circ \mathscr{A}_{t^{n+1}}^{-1}$. Otherwise, the coupling is implicit (also called strong coupling).

There exist two ways for an algorithm to treat the nonlinearities given by the convective term and by the fluid domain: explicitly and implicitly. In the first case, only one fixed point iteration is performed per time step. In the other case, nonlinear iterations are performed till convergence of the fixed point, Newton or quasi-Newton algorithm.

The FSI algorithms treating nonlinearity explicitly are called semi-implicit. In general, the treatment of the fluid domain in an explicit way does not affect the unconditional stability of the coupled FSI problem, even when the added-mass effect is critical. Let us remark that this is not the case for the fluid–structure coupling. Explicit or weak coupling is unstable when the added-mass effect is important.

In particular, if the problem is discretized with a first order method (in time) and the condition

$$\tilde{\boldsymbol{u}}_h^{n+1} = \boldsymbol{w}_h^{n+1}, \quad \text{on } \Sigma_t,$$

is satisfied (e.g. by taking $\tilde{\boldsymbol{u}}_h^{n+1} = \boldsymbol{u}_h^n$ and $\tilde{\boldsymbol{\eta}}_h^{n+1} = \hat{\boldsymbol{\eta}}_h^n$), the semi-implicit method keeps the stability properties of the implicit procedure (see [28]). Examples of semi-implicit algorithms can be found in [16,5]. Semi-implicit methods treat explicitly nonlinearity (reducing CPU cost) and implicitly the fluid–structure coupling (keeping stability). The optimal convergence of the monolithic semi-implicit method has been checked in [5].

## 2.3. The linear fluid–structure system

We aim at writing the fluid–structure system yielded by the linearized and fully discretized FSI problem. We start by introducing the unknowns for the fluid problem: $\mathbf{U}_{\mathrm{f}}^{n+1}, \mathbf{U}_{\sigma}^{n+1}$ and $\mathbf{P}^{n+1}$ are the arrays of nodal values for the velocity of the inner nodes, the velocity of the interface nodes, and the pressure. The structural unknowns are $\mathbf{D}_s^{n+1}$ and $\dot{\mathbf{D}}_s^{n+1}$, the arrays of nodal values for $\hat{\boldsymbol{\eta}}_h^{n+1}$ and $\dot{\boldsymbol{\eta}}_h^{n+1}$. We also consider the structure velocity $\mathbf{U}_s^{n+1} = \delta_t \mathbf{D}_s^{n+1}$. Assuming matching grids and equal interpolation spaces for the fluid velocity and structure displacement, we can state the discrete continuity of velocities as follows:

$$\mathbf{U}_{\sigma}^{n+1} = \delta_t \mathbf{D}_{\sigma}^{n+1}.$$

More involved situations would require the use of mortar methods (see, e.g., [6]), for example.

In order to write the fully discretized coupled problem for a given time value $t^{n+1}$, we need to define a set of matrices. We denote by $K_{\alpha\beta}$ the matrix that includes viscous and convective terms, where the subindexes $\alpha$ and $\beta$ indicate the position of fluid nodes: the *value* $\sigma$ is used for nodes on $\Sigma_t$, $f$ otherwise. Using the same notation, we also define the mass matrices $M_{\alpha\beta}$, the fluid matrix $C_{\alpha\beta} = \frac{1}{\delta t} M_{\alpha\beta} + K_{\alpha\beta}$, the gradient matrix $G_\alpha$ and the divergence matrix $D_\alpha$. In these matrices we already include the corresponding stabilization terms. We also indicate with $L_p^\tau$ the matrix associated to the pressure stabilization. Finally, let us denote with $N_{\alpha\beta}$ the matrix associated to the structure written in terms of structure velocity, where the subindexes $\alpha$ and $\beta$ take the values $\sigma$ for interface nodes and $s$ for inner structure nodes.

At a given time value $t^{n+1}$, Eqs. (10), (9b), (9c) and (9d) can be written in matrix form as:

$$A\mathbf{X}^{n+1} = \mathbf{b}^{n+1}, \tag{11}$$

where

$$A = \begin{bmatrix} C_{ff} & G_{\mathrm{f}} & C_{f\sigma} & 0 \\ D_{\mathrm{f}} & L_p^\tau & D_\sigma & 0 \\ C_{\sigma f} & G_\sigma & C_{\sigma\sigma} + N_{\sigma\sigma} & N_{\sigma s} \\ 0 & 0 & N_{s\sigma} & N_{ss} \end{bmatrix},$$

$$\mathbf{X}^{n+1} = \begin{bmatrix} \mathbf{U}_{\mathrm{f}}^{n+1} \\ \mathbf{P}^{n+1} \\ \mathbf{U}_{\sigma}^{n+1} \\ \mathbf{U}_{s}^{n+1} \end{bmatrix}, \quad \mathbf{b}^{n+1} = \begin{bmatrix} \mathbf{b}_{\mathrm{f}}^{n+1} \\ \mathbf{b}_{p}^{n+1} \\ \mathbf{b}_{\sigma}^{n+1} \\ \mathbf{b}_{s}^{n+1} \end{bmatrix}, \tag{12}$$

The right-hand side terms $\mathbf{b}_{\mathrm{f}}^{n+1}, \mathbf{b}_p^{n+1}, \mathbf{b}_\sigma^{n+1}$ and $\mathbf{b}_s^{n+1}$ account for body forces, time integration and stabilization terms, and the structure terms related to the fact that the structure equation is stated in terms of velocities.

We refer to [5] for a more detailed exposition of the discrete FSI system.

**Remark 1.** It is also possible to linearize the fluid and structure problems through Newton methods. Again, the block structure of matrix $A$ is left unchanged and our procedures can be applied.

**Remark 2.** The orthogonal projection in the stabilization term (7) complicates the assembling of the fluid block. Therefore, for practical purposes, only the term

$$\begin{aligned}&(\tau_1(\tilde{\boldsymbol{u}}_h^{n+1} \cdot \nabla \boldsymbol{u}_h^{n+1} + \nabla p_h^{n+1}), \tilde{\boldsymbol{u}}_h^{n+1} \cdot \nabla \boldsymbol{v}_h^f + \nabla q_h)_{\Omega^f_{t^{n+1}}} \\ &\quad + (\tau_2(\nabla \cdot \boldsymbol{u}_h^{n+1}), \nabla \cdot \boldsymbol{v}_h^f)_{\Omega^f_{t^{n+1}}}\end{aligned} \tag{13}$$

is assembled in the matrix, whereas the missing term is treated explicitly and sent to the right-hand side

$$\begin{aligned}&(\tau_1 \Pi(\tilde{\boldsymbol{u}}_h^n \cdot \nabla \boldsymbol{u}_h^n + \nabla p_h^n), \tilde{\boldsymbol{u}}_h^{n+1} \cdot \nabla \boldsymbol{v}_h^f + \nabla q_h)_{\Omega^f_{t^{n+1}}} \\ &\quad + (\tau_2 \Pi(\nabla \cdot \boldsymbol{u}_h^n), \nabla \cdot \boldsymbol{v}_h^f)_{\Omega^f_{t^{n+1}}}.\end{aligned} \tag{14}$$

Alternatively, we could use the algebraic subgrid scales (ASGS) technique (see [22]), which introduces the stabilization term

$$\begin{aligned}&(\tau_1(\rho_{\mathrm{f}} \delta_t \boldsymbol{u}_h^{n+1}|_{\boldsymbol{x}_0} + \tilde{\boldsymbol{u}}_h^{n+1} \cdot \nabla \boldsymbol{u}_h^{n+1} + \nabla p_h^{n+1}), \tilde{\boldsymbol{u}}_h^{n+1} \cdot \nabla \boldsymbol{v}_h^f + \nabla q_h)_{\Omega^f_{t^{n+1}}} \\ &\quad + (\tau_2(\nabla \cdot \boldsymbol{u}_h^{n+1}), \nabla \cdot \boldsymbol{v}_h^f)_{\Omega^f_{t^{n+1}}}.\end{aligned}$$

**Remark 3.** $L_p^\tau$ is a weighted Laplacian matrix that comes from the term $(\tau_1 \nabla p_h^{n+1}, \nabla q_h)_{\Omega^f_{t^{n+1}}}$.

**Remark 4.** In case of considering non-matching grids and a mortar method on the interface, the monolithic system has to be modified. Two different interface arrays must be considered: the interface fluid velocity $\mathbf{U}_{\sigma,f}^{n+1}$ and the interface structure velocity $\mathbf{U}_{\sigma,s}^{n+1}$. For instance, considering the structure interface as the master, and the fluid interface as the slave, we can easily define the rectangular matrix $Y$ that projects the structure interface velocity into the fluid interface space. The continuity of velocities is imposed as

$$\mathbf{U}_{\sigma,f}^{n+1} = Y\mathbf{U}_{\sigma,s}^{n+1}.$$

Matrix $Y$ involves an inverse mass matrix (better if lumped) on the fluid interface. Then, we must multiply the blocks $C_{\sigma f}$, $G_\sigma$ and $C_{\sigma\sigma}$ by $Y$ on the right and solve the problem with $\mathbf{U}_{\sigma,s}^{n+1}$ as interface unknown. The assembling of these matrices is advised for the preconditioning of the system matrix by ILU-type preconditioners.

## 2.4. Features of the monolithic system

In this section, we describe our monolithic formulation. Our goal is to end up with a simple FSI system suitable for iterative solvers and ILU-type preconditioners.

Firstly, we rely on a finite element partition of the overall domain. It implies matching grids on the fluid–structure interface. This approach is reasonable when there is an interest in solving the problem with non-modular preconditioners (one whole system).

On the other hand, we make use of the same finite element spaces for fluid velocity and structure displacement (or velocity). This is extremely simple when using stabilization techniques because the velocity–pressure pair can circumvent the discrete inf–sup condition. In that case, the same finite element interpolation spaces can be used for fluid velocity, pressure and structure unknowns. For example, for the numerical experiments in Sections 6 and 7, we use $\mathbb{P}_1/\mathbb{P}_1$ finite elements for the fluid and $\mathbb{P}_1$ finite elements for the structure.

Moreover, we reformulate the structure equations in terms of velocities. This is attained by a simple modification of the right-hand side and does not affect at all the generality of the formulation.

By virtue of the previous conditions, the velocity unknowns are defined over the whole domain (fluid and structure), the problem is discretized using one finite element partition and all the unknowns are interpolated with the same finite element space.

In this frame, the transmission conditions are easily imposed. The continuity of velocities on the interface is implicitly enforced by the finite element space interpolation used over the whole domain. The continuity of stresses is imposed weakly. The weak transmission of stresses simply arises from the fact that shape functions on the interface nodes have support on fluid and structure sub-domains (see [5]). In this way, the final system has the clear form reported in (12).

Another option would be to impose the transmission of stresses in a strong form. Once the fluid problem is computed, the stresses are integrated on the boundary elements by evaluating the fluid stress on the Gauss points, and passed to the structure solver. The monolithic matrix in this case reads as

$$A = \begin{bmatrix} C_{ff} & G_f & C_{f\sigma} & 0 \\ D_f & L_p^\tau & D_\sigma & 0 \\ J_{\sigma f} & J_p & N_{\sigma\sigma} & N_{\sigma s} \\ 0 & 0 & N_{s\sigma} & N_{ss} \end{bmatrix},$$

where $J_{\sigma f}$ comes from the term

$$\langle \nu \boldsymbol{n}_f \cdot \nabla \boldsymbol{u}_h^{n+1} \boldsymbol{v}_h^s \rangle_{\Sigma_t}$$

and $J_p$ from

$$\langle -p_h^{n+1} \mathbf{I} \cdot \boldsymbol{n}_f, \boldsymbol{v}_h^s \rangle_{\Sigma_t}.$$

This approach destroys the symmetry of the system (in case of using the Stokes problem), affects the unconditional stability of (12) and spoils the order of accuracy of the method (see [9]). For these reasons, we consider the weak transmission of stresses.

Last but not least, an appropriate fluid formulation is important for the efficiency of ILU-type preconditioners applied to the FSI system. Inf–sup stable elements yield linear systems that are indefinite since they represent saddle-point problems. By using stabilized formulations the zero pressure block of these systems is replaced by a semi-positive definite matrix. This improves remarkably the efficiency of iterative solvers preconditioned with ILU-type preconditioners (see e.g. [34,1,19,8]).

In the next sections, we consider different preconditioners for the monolithic FSI system.

## 3. The Dirichlet–Neumann preconditioner

The FSI system can be reformulated as an interface problem. This is achieved by writing system (12) only in terms of $\mathbf{U}_\sigma$ thanks to the Schur complements of fluid and structure sub-problems. Omitting the time step superscript for simplicity, the interface problem is:

$$(\tilde{C}_\sigma + \tilde{N}_\sigma)\mathbf{U}_\sigma = \tilde{\mathbf{b}}_\sigma,$$

with

$$\tilde{C}_\sigma = C_{\sigma\sigma} - [C_{\sigma f} G_\sigma] \begin{bmatrix} C_{ff} & G_f \\ D_f & L_p^\tau \end{bmatrix}^{-1} \begin{bmatrix} C_{f\sigma} \\ D_\sigma \end{bmatrix},$$

$$\tilde{N}_\sigma = N_{\sigma\sigma} - N_{\sigma s} N_{ss}^{-1} N_{s\sigma},$$

$$\tilde{\mathbf{b}}_\sigma = \mathbf{b}_\sigma - [C_{\sigma f} G_\sigma] \begin{bmatrix} C_{ff} & G_f \\ D_f & L_p^\tau \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{b}_f \\ \mathbf{b}_p \end{bmatrix} - N_{\sigma s} N_{ss}^{-1} \mathbf{b}_s.$$

The interface system preconditioned with the Dirichlet–Neumann preconditioner $\tilde{N}_\sigma$ reads as follows:

$$\tilde{N}_\sigma^{-1}(\tilde{C}_\sigma + \tilde{N}_\sigma)\mathbf{U}_\sigma = \tilde{N}_\sigma^{-1}\tilde{\mathbf{b}}_\sigma. \tag{15}$$

This Schur complement preconditioner can also be understood as an incomplete block-LU factorization of the FSI system matrix $A$ (see [31]). The preconditioned system must be solved with a matrix-free iterative solver. In the next two sections, we introduce two different choices.

### 3.1. Richardson algorithm for the preconditioned interface system

The classical Dirichlet–Neumann algorithm can be understood as Richardson iterations over system (15):

$$\mathbf{U}_\sigma^{k+1} = \mathbf{U}_\sigma^k + \tilde{N}_\sigma^{-1}(\tilde{\mathbf{b}}_\sigma - (\tilde{N}_\sigma + \tilde{C}_\sigma)\mathbf{U}_\sigma^k).$$

We can easily infer that it is equivalent to the following iterative procedure:

(i) *Fluid problem* (Dirichlet boundary condition)

$$\begin{bmatrix} C_{ff} & G_f \\ D_f & L_p^\tau \end{bmatrix} \begin{bmatrix} \mathbf{U}_f^{k+1} \\ \mathbf{P}^{k+1} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_f - C_{f\sigma}\mathbf{U}_\sigma^k \\ \mathbf{b}_p - D_\sigma \mathbf{U}_\sigma^k \end{bmatrix} \tag{16a}$$

(ii) Structure problem (Neumann boundary condition)

$$\begin{bmatrix} N_{\sigma\sigma} & N_{\sigma s} \\ N_{s\sigma} & N_{ss} \end{bmatrix} \left[ \mathbf{U}_\sigma^{k+1} \mathbf{U}_s^{k+1} \right] = \begin{bmatrix} \mathbf{b}_\sigma^{k+1} - C_{\sigma\sigma}\mathbf{U}_\sigma^{k+1} - C_{\sigma f}\mathbf{U}_f^{k+1} - G_\sigma \mathbf{P}^{k+1} \\ \mathbf{b}_s^{k+1} \end{bmatrix}. \tag{16b}$$

This is the most appealing feature of the DN–Richardson method: every iteration of the algorithm can be performed by separate fluid and structure solvers. We only need to modify the boundary conditions.

The iterative process must be supplemented with an appropriate stopping criterion. For instance, for the numerical experiments in Sections 6 and 7 we use:

$$\frac{\|\mathbf{U}_\sigma^{k+1} - \mathbf{U}_\sigma^k\|}{\|\mathbf{U}_\sigma^0\|} \leqslant \epsilon. \tag{17}$$

Every iteration of the DN–Richardson algorithm is expensive, because it involves to solve one fluid and one structure problem. A cheaper preconditioner has been suggested in [36]. The fluid and structure problems are replaced by ILU-type preconditioners of the respective system matrices. This preconditioner is not modular and less effective than the original one, but the computational cost of every iteration is reduced.

### 3.2. GMRES algorithm for the preconditioned interface system

Instead of using Richardson iterations, we can apply the GMRES algorithm to the preconditioned interface problem (15). The resulting method is denoted by DN-GMRES. It is much faster and robust than DN–Richardson, because it involves orthonormal iterations. Moreover, convergence is always assured, at worst after as many iterations as degrees of freedom at the interface (not practical for real applications). The GMRES methods requires to compute and store the Krylov base associated to $Q = \tilde{N}_\sigma^{-1}(\tilde{C}_\sigma + \tilde{N}_\sigma)$, starting from the preconditioned residual $\mathbf{r}^0 = \tilde{N}_\sigma^{-1}[\tilde{\mathbf{b}}_\sigma - (\tilde{C}_\sigma + \tilde{N}_\sigma)\mathbf{U}_\sigma^0]$, where $\mathbf{U}_\sigma^0$ is the initial guess. The Krylov space generated for the $m$-th iteration of the GMRES method is

$$\mathscr{K}_m := \text{span}\{\mathbf{r}^0, Q\mathbf{r}^0, Q^2\mathbf{r}^0, \ldots, Q^m\mathbf{r}^0\} = \text{span}\{\mathbf{z}^0, \mathbf{z}^1, \ldots, \mathbf{z}^m\}. \tag{18}$$

Given $\mathbf{z}^k$, in order to get $\mathbf{z}^{k+1}$ we must evaluate a matrix-vector product

$$\tilde{N}_\sigma^{-1}(\tilde{N}_\sigma + \tilde{C}_\sigma)\mathbf{z}^k = \mathbf{z}^k + \tilde{N}_\sigma^{-1}\tilde{C}_\sigma\mathbf{z}^k$$

This algorithm can be rearranged in such a way that every matrix-vector product is evaluated by the DN–Richardson code, simply setting to zero the body force:

(i) Given $\mathbf{U}_\sigma^0$, solve one Richardson iteration of (16a) to get $\mathbf{U}_\sigma^1$ and compute the initial residual as:

$$\mathbf{r}^0 = \mathbf{U}_\sigma^1 - \mathbf{U}_\sigma^0.$$

(ii) Initialize the Krylov base with $\mathbf{z}^0 = \mathbf{r}^0/\|\mathbf{r}^0\|$ and at every GMRES iteration (see [33], Section 6.5]) obtain the matrix vector product $\mathbf{w} = Q\mathbf{z}^k$ as follows:

(a) Fluid problem (Dirichlet boundary conditions and zero forcing term)

$$\begin{bmatrix} C_{ff} & G_f \\ D_f & L_p^\tau \end{bmatrix}\begin{bmatrix} \mathbf{v}_f \\ \mathbf{q} \end{bmatrix} = -\begin{bmatrix} C_{f\sigma}\mathbf{z}^k \\ D_\sigma\mathbf{z}^k \end{bmatrix}; \tag{19a}$$

(b) *Structure problem* (Neumann boundary conditions and zero forcing term)

$$\begin{bmatrix} N_{\sigma\sigma} & N_{\sigma s} \\ N_{s\sigma} & N_{ss} \end{bmatrix}\begin{bmatrix} \mathbf{v}_\sigma \\ \mathbf{v}_s \end{bmatrix} = -\begin{bmatrix} C_{\sigma\sigma}\mathbf{z}^k + C_{\sigma f}\mathbf{v}_f + G_\sigma\mathbf{q} \\ \mathbf{0} \end{bmatrix}. \tag{19b}$$

(c) Evaluate $\mathbf{w} = \mathbf{z}^k - \mathbf{v}_\sigma$.

Implementing the DN-GMRES method by reusing the DN–Richardson master allows to use separate fluid and structure solvers. Unluckily, the performance of the DN-GMRES algorithm is still negatively affected by the added-mass effect.

**Remark 5.** At every GMRES iteration we get

$$\mathbf{U}_\sigma^k = \underset{\mathbf{y}\in\mathscr{K}_k}{\operatorname{argmin}}\|\tilde{N}_\sigma^{-1}[\tilde{\mathbf{b}}_\sigma - (\tilde{C}_\sigma + \tilde{N}_\sigma)\mathbf{y}]\|,$$

which can also be written as

$$\|\tilde{\mathbf{U}}_\sigma^{k+1} - \mathbf{U}_\sigma^k\|,$$

where $\tilde{\mathbf{U}}_\sigma^{k+1}$ is obtained from $\mathbf{U}_\sigma^k$ by solving one iteration of the Richardson algorithm (16a). By taking $\epsilon\|\mathbf{U}_\sigma^0\|$ as tolerance, we impose the same stopping criterion used for the DN–Richardson method. This is the choice adopted in the numerical experiments.

**Remark 6.** The GMRES algorithm is performed over the interface unknowns. Therefore, the Krylov base elements only have the dimensions of $\mathbf{U}_\sigma$. The memory requirements are clearly reduced.

**Remark 7.** The DN-GMRES algorithm could be implemented in a modular way. The computation of the initial residual is nothing else but one iteration of the DN–Richardson algorithm and the rest of the matrix-vector products can be computed using (19a), with separate fluid and structure evaluations. However, we must set to zero the right-hand side term in both sub-problems. Assuming that this can be done without modifying the source codes, the DN-GMRES would keep modularity. In any case, a modular DN-GMRES algorithm is extremely inefficient; fluid and structure matrices do not change in the iterative process and could be assembled only once. An efficient implementation of the DN-GMRES algorithm requires a master with access to fluid and structure blocks to perform the iterative process without reassembling matrices.

### *3.3. The reduction factor for the residual norm of the DN-GMRES method for a model problem*

The purpose of this subsection is to understand how the added-mass effect affects the convergence of the DN-GMRES algorithm. To fulfill it, we consider the simplified fluid–structure model proposed in [11].

We take a rectangular fluid domain $\Omega^f \in \mathbb{R}^2$ of height $R$ and length $L$ (see Fig. 2 in [11]). The structure domain $\Omega^s$ coincides with the interface, that is $\overline{\Omega^s} = \Sigma$. Under the hypothesis of dealing with a thin structure, having a membrane behaviour and neglecting all the displacements but the normal one, we derive the structure model:

$$\rho_s h\partial_{tt}\eta + a\eta - b\partial_{xx}\eta = f_\Sigma(x,t) \quad \text{in } \Omega^s \times (0,T).$$

Here, $\eta = \eta(x,t)$ is the displacement in the direction of $\boldsymbol{n}^f$, $h$ is the thickness of the structure, $a = Eh/R^2(1-\nu^2)$, $E$ being the Young modulus and $\nu$ the Poisson coefficient, $b = kGh$, $G$ being the shear stress modulus and $k$ the Timoshenko shear correction factor, and $f_\Sigma(x,t)$ the forcing term coming from the fluid.

The model for the fluid is linear, incompressible, and inviscid. The deformation of the structure is assumed to be so small that the fluid domain $\Omega^f$ can be considered fixed. Hence, the fluid model is the following:

$$\begin{aligned} &\rho_f\delta_t\boldsymbol{u} + \nabla p = \mathbf{0} \quad \text{in } \Omega^f \times (0,T), \\ &\nabla\cdot\boldsymbol{u} = 0 \quad \text{in } \Omega_t^f \times (0,T), \\ &u = \delta_t\eta \quad \text{on } \Sigma \times (0,T), \end{aligned}$$

with suitable boundary conditions on $\partial\Omega^f \setminus \Sigma$ and initial conditions; $u$ denotes $\boldsymbol{u}\cdot\boldsymbol{n}_f$ on $\Sigma$.

For the time discretization of the FSI system we choose the implicit Euler scheme for the fluid problem and first order backward difference scheme for the structure one. The time-discrete problem reads:

$$\begin{aligned} &\rho_f\delta_t\boldsymbol{u}^{n+1} + \nabla p^{n+1} = \mathbf{0} \quad \text{in } \Omega^f \times (0,T), \\ &\nabla\cdot\boldsymbol{u}^{n+1} = 0 \quad \text{in } \Omega_t^f \times (0,T), \\ &u = \delta_t\eta^{n+1} \quad \text{on } \Sigma \times (0,T), \end{aligned} \tag{20}$$

and

$$\begin{aligned} &\rho_s h\frac{\eta^{n+1} - 2\eta^n + \eta^{n-1}}{\delta t^2} + a\eta^{n+1} - b\partial_{xx}\eta^{n+1} \\ &\quad = p^{n+1} \quad \text{in } \Omega^s \times (0,T). \end{aligned} \tag{21}$$

It can be shown [11,4] that problem (20) and (21) corresponds to the following discrete added-mass problem for the structure:

$$\begin{aligned} &(\rho_s h\mathscr{I} + \rho_f\mathscr{M})\frac{\eta^{n+1} - 2\eta^n + \eta^{n-1}}{\delta t^2} + a\eta^{n+1} + b\mathscr{L}\eta^{n+1} \\ &\quad = \hat{p}^{n+1} \quad \text{on } \Omega^s \times (0,T), \end{aligned} \tag{22}$$

where $\mathscr{I}$ denotes the identity operator, $\mathscr{M} : H^{-1/2}(\Sigma) \to H^{1/2}(\Sigma)$ stands for the added-mass operator and $\mathscr{L} = -\partial_{xx}$ is the Laplace operator. $\hat{p}^{n+1}$ takes into account non-homogeneous boundary conditions on $\partial\Omega^f \setminus \Sigma$.

Let us indicate with $\mathscr{Q}$ the linear, invertible, and continuous operator

$$\mathscr{Q} = \left(\frac{\rho_s h}{\delta t^2} + a\right)\mathscr{I} + b\mathscr{L} + \frac{\rho_f}{\delta t^2}\mathscr{M},$$

which can be split as $\mathscr{Q} = \mathscr{Q}_f + \mathscr{Q}_s$, where $\mathscr{Q}_f$ and $\mathscr{Q}_s$ are the linear operators associated to the fluid and structure sub-domains:

$$\mathscr{Q}_f = \frac{\rho_f}{\delta t^2}\mathscr{M}, \quad \mathscr{Q}_s = \left(\frac{\rho_s h}{\delta t^2} + a\right)\mathscr{I} + b\mathscr{L}.$$

Solving (22) with the DN-GMRES algorithm means to solve the problem $\mathscr{Q}\eta^{n+1} = G$ ($G$ accounting for $\eta^n, \eta^{n-1}$ and $\hat{p}^{n+1}$) with the GMRES method based on $\mathscr{Q}_s$ as preconditioner. To analyze the DN-GMRES algorithm we express $\eta$ as

$$\eta = \sum_{i=1}^{\infty} \eta_i g_i, \quad \text{with } g_i = \sqrt{\frac{2}{L}} \sin\left(\mathrm{i}\pi \frac{x}{L}\right).$$

The functions $g_i$ are eigenfunctions of both the added-mass and the Laplace operators. Let $\mu_i$ (see [11]) and $\lambda_i$ (see [4]) be the respective eigenvalues:

$$\mu_i = \frac{L}{\mathrm{i}\pi \tanh\left(\mathrm{i}\pi \frac{R}{L}\right)}, \quad \text{and} \quad \lambda_i = \left(\frac{\mathrm{i}\pi}{L}\right)^2,$$

for $i = 1, \ldots, \infty$. The operator $\mathcal{Q}_s$ is continuous and coercive. Also $\mathcal{Q}_f$ is continuous [11].

The reduction factor $\rho$ for the residual norm of the DN-GMRES method is given by:

$$\rho = \sqrt{1 - \frac{\sigma_{\min}}{\sigma_{\max}}}, \tag{23}$$

(see, e.g., [14]), where

$$\sigma_{\min} = \inf_{\eta \neq 0} \frac{(\mathcal{Q}_s^{-1}\mathcal{Q}\eta, \eta)}{(\eta, \eta)}, \quad \sigma_{\max} = \sup_{\eta \neq 0} \frac{(\mathcal{Q}_s^{-1}\mathcal{Q}\eta, \eta)}{(\eta, \eta)}.$$

We have:

$$\sigma_{\min} = \inf_{\eta \neq 0} \frac{(\mathcal{Q}_s^{-1}(\mathcal{Q}_f + \mathcal{Q}_s)\eta, \eta)}{(\eta, \eta)} = 1 + \inf_{\eta \neq 0} \frac{(\mathcal{Q}_s^{-1}\mathcal{Q}_f\eta, \eta)}{(\eta, \eta)} = 1, \tag{24}$$

since the operator $\mathcal{Q}_f$ is positive on $L^2(\Sigma)$ and $\mu_i \to 0$ and $\lambda_i \to \infty$ as $i \to \infty$. For the supremum we get:

$$\sigma_{\max} = 1 + \sup_{\eta \neq 0} \frac{(\mathcal{Q}_s^{-1}\mathcal{Q}_f\eta, \eta)}{(\eta, \eta)} = 1 + \frac{\rho_f \mu_{\max}}{\rho_s h + a\delta t^2 + \delta t^2 b \lambda_{\min}}.$$

In [4], it is proved that the DN–Richardson algorithm applied to the simplified problem (22) converges to the monolithic solution only if the relaxation parameter $\omega \in (0, \omega_{\max}]$, with

$$\omega_{\max} = \frac{2}{1 + \frac{\rho_f \mu_1}{\rho_s h + a\delta t^2 + \delta t^2 b \lambda_1}}.$$

Thus, $\sigma_{\max} = \frac{2}{\omega_{\max}}$. Plugging this result and (24) into (23), we obtain

$$\rho = \sqrt{1 - \frac{\omega_{\max}}{2}} = \sqrt{\frac{\rho_f \mu_1}{\rho_f \mu_1 + \rho_s h + a\delta t^2 + \delta t^2 b \lambda_1}}.$$

Since $0 < \rho < 1$, the advantage of the DN-GMRES algorithm is that convergence is always reached, whereas the DN–Richardson method has a constraint on the relaxation parameter. However, as the added-mass effect gets critical, $\omega_{\max} \to 0$; so the reduction factor $\rho \to 1$ and convergence slows down.

## 4. ILU preconditioners

The first problem related to the monolithic FSI matrix is the discrepancy between the entries in the different blocks. In order to solve this issue, we consider a diagonal scaling of the matrix (applied on the left). The diagonal scaling we performed for the numerical simulations in Sections 6 and 7 is the following. Let $D$ be the diagonal matrix whose element are the diagonal coefficients of $A$ (12). Instead of solving system (11), we solve:

$$\widehat{A}\mathbf{X}^{n+1} = \hat{\mathbf{b}}^{n+1},$$

where $\widehat{A} := D^{-1}A$ and $\hat{\mathbf{b}}^{n+1} := D^{-1}\mathbf{b}^{n+1}$.

The system matrix $\widehat{A}$ is preconditioned by an incomplete LU factorization $P$, the so-called ILUT preconditioner (see [33]). The ILUT preconditioner allows to fix a threshold (entries smaller than the threshold are discarded) and the level of fill-in (that defines the maximum number of non-zero entries per row). Again, we make use of left-preconditioning:

$$P^{-1}\widehat{A}\mathbf{X}^{n+1} = P^{-1}\hat{\mathbf{b}}^{n+1} \tag{25}$$

This method is non-modular, in the sense that the whole monolithic matrix is needed to compute the preconditioner.

In the non-modular approach, we aim at solving the FSI linear system through standard iterative methods. The preconditioned system is solved by a matrix-free Krylov method. Because of the non-symmetric nature of the system, we consider the GMRES and BiCGStab algorithms. The GMRES method is based on the minimization of the residual of the preconditioned system (25). This algorithm requires to store the Krylov base, where every element of the base is an array of size the number of unknowns. Due to memory constraints, the maximum number of Krylov elements that can be stored must be limited. When this limit is reached, the GMRES method must be re-started. The BiCGStab algorithm is based on a quasi-minimization of the residual that does not require to store the Krylov base, drastically reducing the memory usage. The GMRES algorithm (without re-starting) requires a lower number of iterations than BiCGStab; however, the latter performs better than the re-started GMRES.

## 5. An inexact block-LU factorization

In [5] two semi-implicit algorithms have been derived from splitting techniques designed for the FSI problem at the fully discrete level. In that work, inf–sup stable finite element pairs were used for velocity and pressure, and the methods (called PIC and FSY) were tested on a 2D benchmark involving a one-dimensional structure. Now, the goal is to solve realistic 3D applications with the PIC and FSY schemes in order to understand their efficiency and compare it to that of the methods presented in Sections 3 and 4.

In this section, we briefly extend the semi-implicit algorithms based on inexact factorizations to stabilized finite element methods with equal velocity–pressure interpolation and generalize them to the case of a $d$-dimensional structure. Letting the subscript $S$ indicate both the inner structure and interface nodes, the matrix and the vectors in (12) can be rewritten as

$$A = \begin{bmatrix} C_{ff} & G_f & C_{fS} \\ D_f & L_p^\tau & D_S \\ C_{Sf} & G_S & N_S \end{bmatrix},$$

$$\mathbf{X}^{n+1} = \begin{bmatrix} \mathbf{U}_f^{n+1} \\ \mathbf{P}^{n+1} \\ \mathbf{U}_S^{n+1} \end{bmatrix}, \quad \mathbf{b}^{n+1} = \begin{bmatrix} \mathbf{b}_f^{n+1} \\ \mathbf{b}_p^{n+1} \\ \mathbf{b}_S^{n+1} \end{bmatrix}. \tag{26}$$

The PIC and FSY schemes derive from an inexact block-LU factorization, carried out over the FSI system matrix in (26). The exact $L$ and $U$ factors read:

$$A = \begin{bmatrix} C_{ff} & 0 & 0 \\ D_f & S_{pp} & S_{pS} \\ C_{Sf} & S_{Sp} & S_{SS} \end{bmatrix} \begin{bmatrix} I & C_{ff}^{-1}G_f & C_{ff}^{-1}C_{fS} \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} = LU, \tag{27}$$

where the $S$-matrices are Schur complements. Their formal definition is:

$$S_{pp} = L_p^\tau - D_f C_{ff}^{-1} G_f,$$
$$S_{pS} = D_S - D_f C_{ff}^{-1} C_{fS},$$
$$S_{Sp} = G_S - C_{Sf} C_{ff}^{-1} G_f,$$
$$S_{SS} = N_{SS} - C_{Sf} C_{ff}^{-1} C_{fS}.$$

The presence of the inverse fluid matrix $C_{ff}^{-1}$ makes the exact LU factorization unaffordable. Therefore, we resort to inexact factorizations in order to reduce the computational complexity. The exact

$L$ and $U$ factors in (27) are replaced by inexact ones in which $C_{ff}^{-1}$ is substituted by the zeroth order term of its Neumann expansion:

$$C_{ff}^{-1} = \left(\frac{1}{\delta t}M_{ff} + K_{ff}\right)^{-1} = \delta t M_{ff}^{-1} + \mathcal{O}(\delta t^2) \simeq \delta t M_{ff}^{-1}. \tag{28}$$

**Remark 8.** In order to reduce the computational cost, a lumped mass matrix $M_{ff}$ is used.

**Remark 9.** When using the OSS technique, none of the stabilization terms is multiplied by $\delta t^{-1}$; the time derivative terms in the residual disappear with the orthogonal projection. We can include all the stabilization terms in $K_{ff}$ and use the previous expansion with a lumped mass matrix. However, for some other techniques, like *algebraic subgrid scales* or Galerkin/least-squares, there are stabilization terms that are multiplied by $\delta t^{-1}$. Matrix $M_{ff}$ is not a standard mass matrix anymore and cannot be lumped, making its inversion more involved.

We denote by $T_{\alpha\beta}$ the approximated Schur complements, in which $C_{ff}^{-1}$ is replaced by $\delta t M_{ff}^{-1}$. The subindexes $\alpha$ and $\beta$ can take the values $p$ for pressure and $S$ for interface and inner structure nodes. Consequently, the lower block-triangular matrix $L$ is approximated by:

$$L_{\mathrm{PIC}} := \begin{bmatrix} C_{ff} & 0 & 0 \\ D_{\mathrm{f}} & T_{pp} & T_{pS} \\ C_{Sf} & T_{Sp} & T_{SS} \end{bmatrix}.$$

Using the same approximation (28) for the upper block-triangular matrix $U$, the following inexact $U$ factor is obtained:

$$U_{\mathrm{PIC}} := \begin{bmatrix} I & \delta t M_{ff}^{-1} G_{\mathrm{f}} & \delta t M_{ff}^{-1} C_{fS} \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix}.$$

The system matrix for the PIC scheme is obtained by replacing matrices $L$ and $U$ with $L_{\mathrm{PIC}}$ and $U_{\mathrm{PIC}}$ ($A_{\mathrm{PIC}} = L_{\mathrm{PIC}}U_{\mathrm{PIC}}$), while the FSY algorithm substitutes only the $L$ factor ($A_{\mathrm{FSY}} = L_{\mathrm{PIC}}U$).

An inexact factorization involves a perturbation error that can be reduced if it is applied to the incremental system (instead of the non-incremental (11)):

$$A(\mathbf{X}^{n+1} - \mathbf{X}^*) = \mathbf{b}^{n+1} - A\mathbf{X}^*, \tag{29}$$

where $\mathbf{X}^*$ is the vector made of $\mathbf{U}_{\mathrm{f}}^*, \mathbf{P}^*$ and $\mathbf{U}_S^*$ which are predictions of $\mathbf{U}_{\mathrm{f}}^{n+1}, \mathbf{P}^{n+1}$ and $\mathbf{U}_S^{n+1}$. For instance, a first order prediction would be $\mathbf{X}^* = \mathbf{X}^n$.

The algorithms based on the inexact factorizations applied to the incremental FSI system can be rearranged into three-steps procedures. For the PIC scheme that procedure is the following:

(i) *Computation of the intermediate velocity*:

$$C_{ff}\widetilde{\mathbf{U}}_{\mathrm{f}}^{n+1} = \mathbf{b}_{\mathrm{f}}^{n+1} - G_{\mathrm{f}}\mathbf{P}^* - C_{fS}\mathbf{U}_S^*; \tag{30a}$$

(ii) *Solution of the coupled pressure-interface system*:

$$\begin{bmatrix} T_{pp} & T_{pS} \\ T_{Sp} & T_{SS} \end{bmatrix} \begin{bmatrix} \mathbf{P}^{n+1} - \mathbf{P}^* \\ \mathbf{U}_S^{n+1} - \mathbf{U}_S^* \end{bmatrix} = \begin{bmatrix} -D_{\mathrm{f}}\widetilde{\mathbf{U}}_{\mathrm{f}}^{n+1} \\ \mathbf{b}_S^{n+1} - C_{Sf}\widetilde{\mathbf{U}}_{\mathrm{f}}^{n+1} \end{bmatrix} - \begin{bmatrix} 0 & D_S \\ G_S & N_{SS} \end{bmatrix} \begin{bmatrix} \mathbf{P}^* \\ \mathbf{U}_S^* \end{bmatrix}; \tag{30b}$$

(iii) *Computation of the end-of-step velocity*:

$$\frac{1}{\delta t}M_{ff}\mathbf{U}_{\mathrm{f}}^{n+1} = \frac{1}{\delta t}M_{ff}\widetilde{\mathbf{U}}_{\mathrm{f}}^{n+1} - G_{\mathrm{f}}(\mathbf{P}^{n+1} - \mathbf{P}^*) - C_{fS}(\mathbf{U}_S^{n+1} - \mathbf{U}_S^*). \tag{30c}$$

The incremental version of the FSY scheme shares the first two steps (30a) and (30b) with the PIC method, whereas the third one becomes

(iv) *Computation of the end-of-step velocity*:

$$C_{ff}\mathbf{U}_{\mathrm{f}}^{n+1} = C_{ff}\widetilde{\mathbf{U}}_{\mathrm{f}}^{n+1} - G_{\mathrm{f}}(\mathbf{P}^{n+1} - \mathbf{P}^*) - C_{f\sigma}(\mathbf{U}_\sigma^{n+1} - \mathbf{U}_\sigma^*).$$

The latter step differs from (30c) and is actually more expensive due to the presence of the stiffness matrix $C_{ff}$. Since we are interested in comparing the efficiency of different methods, in the numerical simulations we will only consider the PIC algorithm.

Clearly, the numerical complexity of the PIC scheme lies in Step 2, where the pressure is coupled to the structure velocity. In the following, we will denote by $T$ the system matrix of the pressure–structure problem (30b). The added-mass can only have an effect on matrix $T$, whose size is much smaller than that of the original FSI system matrix (26). For the solution of system (30b) we adopt a matrix-free method, which prevents us from assembling the matrix. In particular, in Section 7 we consider the GMRES and the BiCGStab algorithms and the corresponding PIC schemes are called PIC-GMRES and PIC-BiCGStab.

A key point in the solution of system (30b) is the choice of a good preconditioner for $T$. The computation of the ILU preconditioner would require the evaluation of the elements of $T$. Hence, it is too expensive and does not make much sense, since we want to avoid the cost of assembling $T$ by adopting a matrix-free method. In the simulation of the carotid bifurcation (Section 7), we employed two preconditioners: the point-diagonal and the block-diagonal one. The former proves to be cheaper in terms of CPU time (see Fig. 10a).

**Remark 10.** Herein, the PIC algorithm has been considered as a solver, with the corresponding perturbation. However, this inexact block-LU factorization can also be used as preconditioner (see [5]).

## 6. Numerical results for the straight cylindrical pipe

Through our numerical experimentation we aim at analyzing how the added-mass effect affects the performance of the different FSI algorithms considered above. Our goal is to simulate the propagation of a pressure pulse in a straight pipe with deformable boundaries as the structure density varies. We consider both the fully 3D problem, whose fluid domain is a cylinder of radius $R_0$ 0.5 cm and length $L = 6$ cm, and its 2D approximation, obtained by intersecting the pipe with a plane. The fluid and structure physical parameters used in the simulations are listed in Table 1: a double line separates the common ones from the ones of the 2D problem only (see [28]), which are separated also from the parameters of the 3D problem (see [13]).

On the inflow section we impose the following Neumann boundary condition:

$$\boldsymbol{\sigma}_{\mathrm{in}}^f = -\frac{P_{\mathrm{in}}}{2}\left[1 - \cos\left(\frac{\pi t}{2.5 \times 10^{-3}}\right)\right]\mathbf{n}_{\mathrm{f}},$$

while on the outflow section an homogeneous Neumann condition has been imposed. The amplitude $P_{\mathrm{in}}$ of the pressure pulse has been taken equal to $2 \times 10^4$ dyne/cm$^2$ and the time duration of the pulse is 5 ms. We solve the problem over the time interval [0, 0.012] s.

**Table 1**
Fluid and structure physical properties for the numerical tests

| | |
|---|---|
| Fluid density: $\rho_{\mathrm{f}} = 1.0$ g/cm$^3$ | Fluid viscosity: $\mu = 0.035$ poise |
| Structure density: $\rho_{\mathrm{s}} = 1.1$ g/cm$^3$ | Wall thickness: $h = 0.1$ cm |
| Young modulus: $E = 7 \times 10^5$ dyne/cm$^2$ | Viscoelastic parameter: $\gamma = 10^{-1}$ dyne s |
| Shear modulus: $G = 2.5 \times 10^5$ dyne/cm$^2$ | Poisson coefficient: $\nu = 0.4$ |
| Lamé constants: $\mu_\ell = 10^6$ dyne/ cm$^2$, $\lambda_\ell = 1.73 \times 10^6$ dyne/cm$^2$ | |

For both problems we choose a conforming space discretization between fluid and structure: stabilized $\mathbb{P}_1 - \mathbb{P}_1$ finite elements for the fluid and $\mathbb{P}_1$ finite elements for the structure.

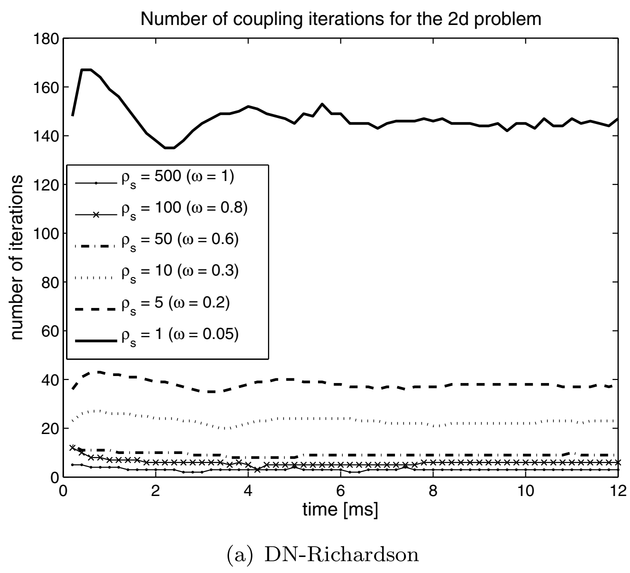
The software that has been used is ZEPHYR, a multi-physics finite element code written in Fortran and developed at CIMNE-UPC (Barcelona). For the ILUT preconditioner and iterative solvers, we have used SPARSKIT, developed by Saad (see [32]). All the simulations were performed on a 3.2 GHz Pentium 4 with 2 GB of RAM.

### *6.1. Comparison between the DN–Richardson and DN-GMRES methods*

We solve the 2D problem with the DN–Richardson and DN-GMRES algorithms (semi-implicit version) on a structured mesh of $61 \times 21$ fluid nodes and $61 \times 4$ structure nodes, with time step $\delta t = 2 \times 10^{-4}$ s. We consider different values of the structure density $\rho_s = 500, 100, 50, 10, 5, 1$ g/cm$^3$. Similar results have been reported in [11,28] for inf–sup stable finite elements for the fluid and simplified structural models under the hypotheses of plane stress and membrane deformations.

We choose to adopt the explicit treatment of the nonlinearities in order to focus on the fluid–structure coupling iterations.

Fig. 1 shows the number of coupling iterations needed by the two algorithms to satisfy the stopping criterion ((17) with tolerance $10^{-4}$) at each time step, for the different densities. The number of subiterations for the DN–Richardson algorithm increases dramatically as the structure density approaches the fluid one. Notice in the legend the relaxation parameter $\omega$ taken in each case: it corresponds to the highest value under which we have convergence of the coupling iterations. The relaxation parameter can be interpreted as an index of "stiffness" of the fluid–structure coupling. When using the DN-GMRES algorithm the number of subiterations increases only slightly as the structure density decreases. In fact, the two methods are almost equivalent in the case of high structure densities, but the advantage of employing GMRES instead



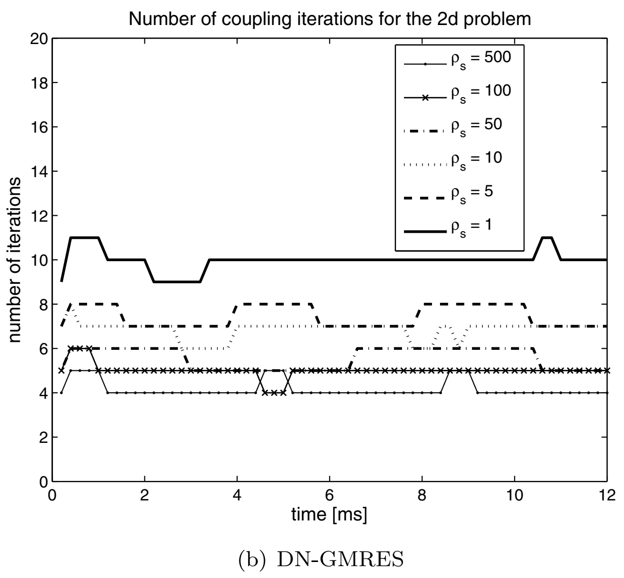
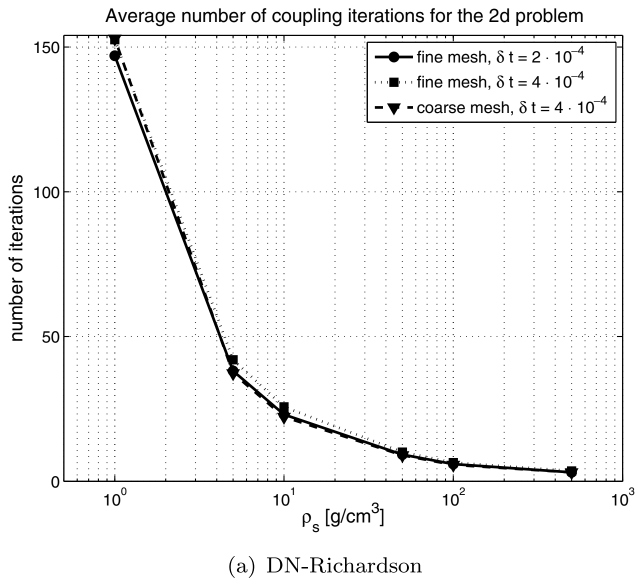


**Fig. 1.** Number of coupling iterations needed to satisfy the convergence criterion at each time step for (a) the DN–Richardson and (b) DN-GMRES methods.



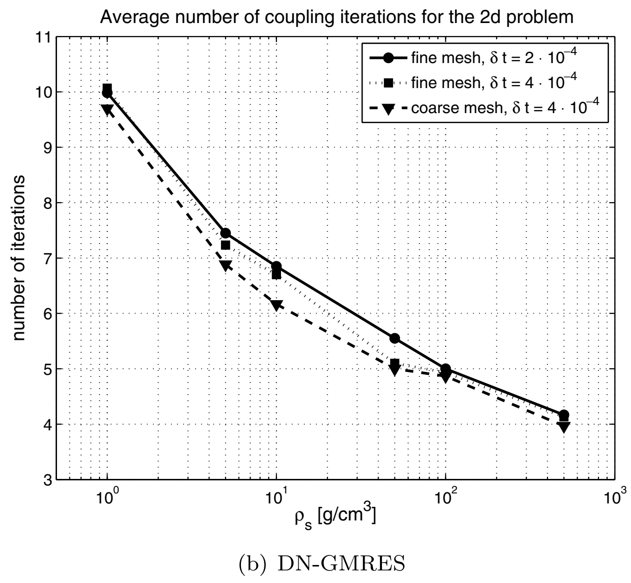


**Fig. 2.** Average number of coupling iterations for (a) the DN–Richardson and (b) DN-GMRES methods as the structure density varies. Comparison for different meshes and different time step sizes.

of Richardson iterations becomes clear in presence of a strong added-mass effect. Moreover, no relaxation is needed for the convergence of the DN-GMRES algorithm.

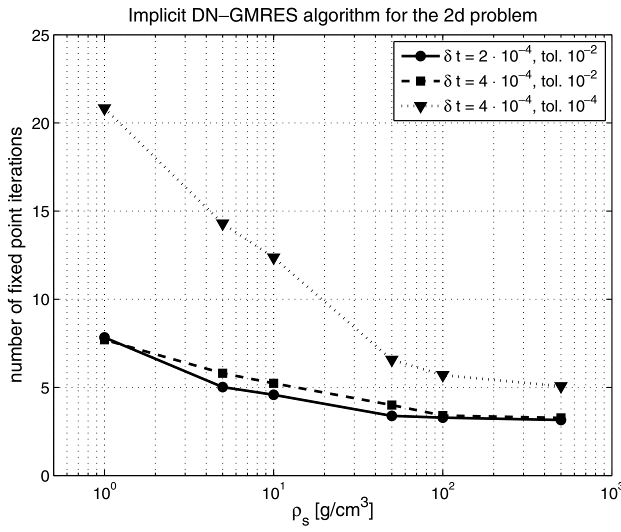
To better show the improvement of the DN-GMRES algorithm we report in Fig. 2 the average number of coupling iterations over the time interval for the two methods as the structure density varies. Both methods are fairly insensitive to mesh size variations. The coarser structured mesh used for the comparison has $41 \times 16$ fluid nodes and $41 \times 3$ structure nodes.

## 6.2. The DN-GMRES method: implicit and semi-implicit versions

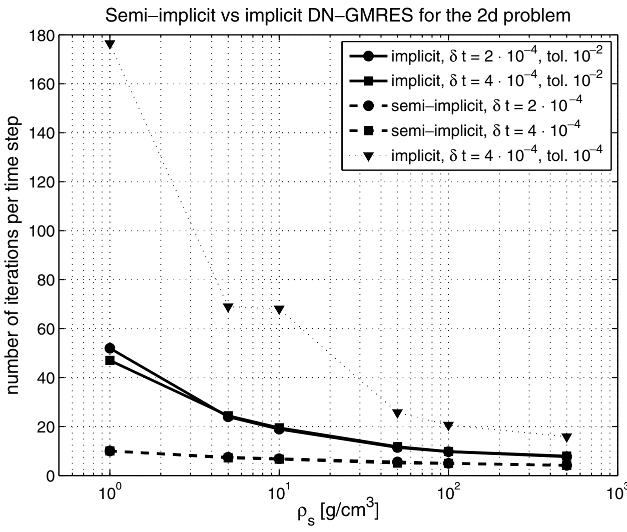
In order to check the computational savings allowed by the explicit treatment of the nonlinearities, we compare the implicit and semi-implicit versions of the DN-GMRES algorithm for the 2D problem.

Fig. 3a shows the average number of nonlinear iterations of a fixed point algorithm for two different time step sizes, two different tolerances of the nonlinear loop, and for all the structure densities specified in Section 6.1. For high $\rho_s$ the nonlinearity is mainly due to the convective term in the fluid equations, while as $\rho_s$ decreases the domain nonlinearities become more important.

The implicit DN-GMRES method uses two nested loops: an external one dealing with the nonlinearity and an internal one solving every linearized system. Thus, the implicit method is computationally intensive, with a high number of fluid structure evaluations (loosely speaking, number of nonlinear iterations times number of average coupling iterations). We plot the accumulative number of iterations, i.e. the sum of the number of GMRES iterations required by every fixed point iteration, of the implicit DN-GMRES for the 2D test problem in Fig. 3b. On the contrary, the DN–Richardson method allows to use only one loop that deals with

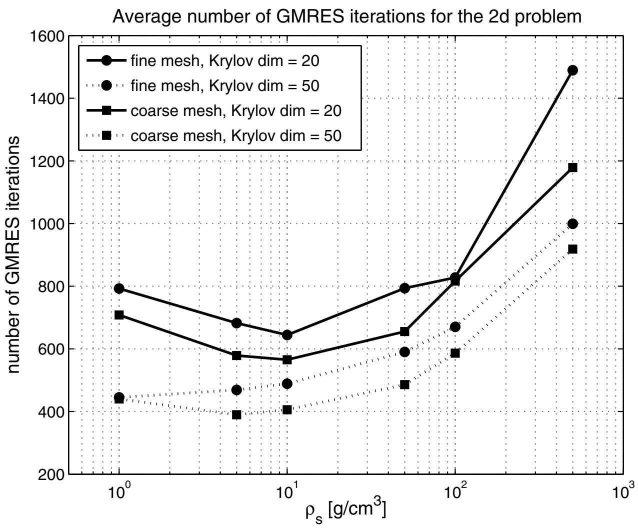


(a) Nonlinear iterations

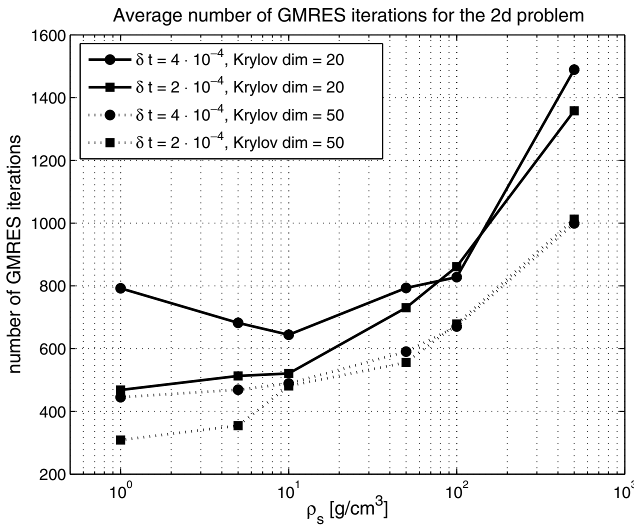


(b) Total coupling iterations

**Fig. 3.** (a) Average number of fixed point iterations for the implicit version and (b) average number of total GMRES iterations per time step for the implicit and semi-implicit versions, for different structure densities and time step sizes.



(a) Dependence with $h$



(b) Dependence with $\delta t$

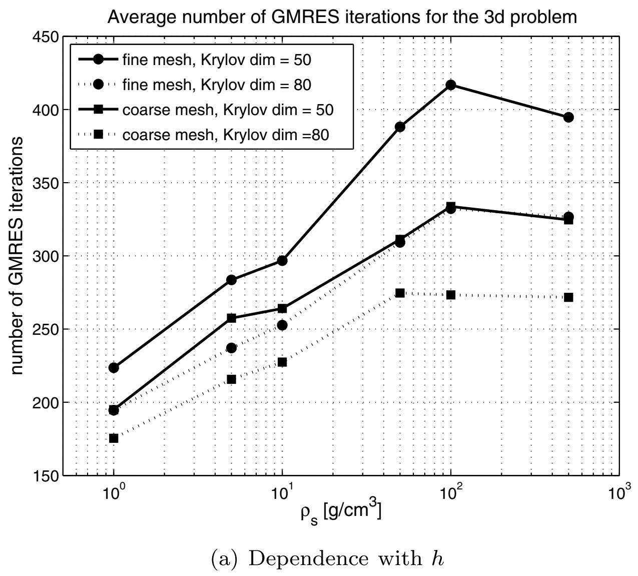
**Fig. 4.** Average number of GMRES iterations to solve the monolithic system for the 2D problem, for different $\rho_s$. Comparison for (a) different meshes and (b) different time step sizes. (a) Dependence with $h$ and (b) Dependence with $\delta t$.

both nonlinear and coupling iterations (see [3]). Even though the nonlinear iterations are not so ill-posed as coupling iterations, the number of accumulative iterations increases a lot. In Fig. 3b, we compare the average number of GMRES iterations per time step for the implicit and semi-implicit versions of DN-GMRES, as $\rho_s$ varies. In the case of a low density structure, an explicit treatment of the nonlinearity reduces drastically the CPU cost because no nonlinear iterations must be performed; when using the ALE formulation, every nonlinear iteration of the shape domain involves to compute a Laplacian problem. The difference between the CPU cost of semi-implicit and implicit schemes gets even bigger with a tighter tolerance, as expected. Therefore, in hemodynamics applications it is very appealing to deal explicitly with the geometrical and fluid nonlinearities, while keeping the fluid–structure system coupled.

The computational savings obtained by a semi-implicit treatment of the nonlinearity are also reported in Section 7.3 for a realistic 3D problem.

### 6.3. The ILUT-GMRES and ILUT-BiCGStab methods

We apply our non-modular approach to the 2D and 3D problems for the same values of $\rho_s$ reported in Section 6.1. The preconditioners adopted are the incomplete LU factors of the scaled monolithic system with 20 non-zero entries per row and threshold 0.1. For the GMRES method, two different values for the maximum dimension of the Krylov space (20 and 50 for the 2D problem, 50 and 80 for the 3D one) are taken into account. Again, we consider the semi-implicit versions. The main goal of this section is not to compare re-started GMRES and BiCGStab iterative solvers; as



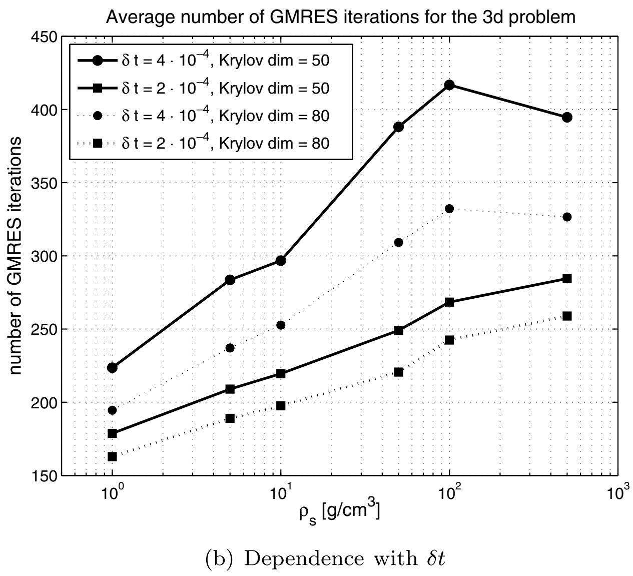
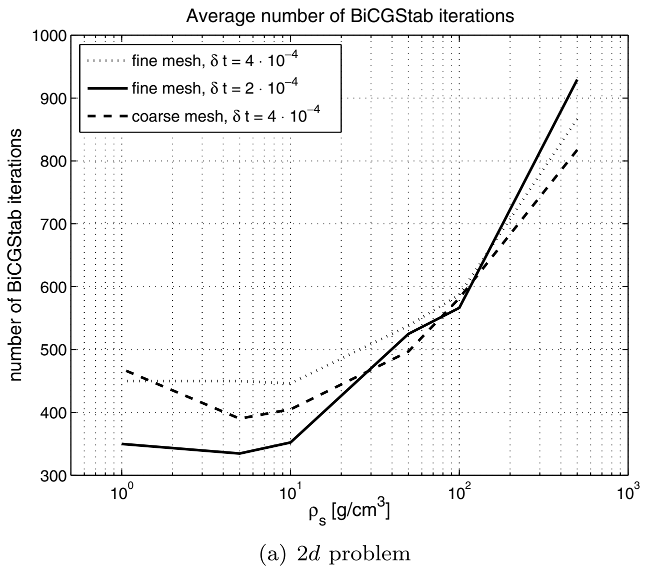


**Fig. 5.** Average number of GMRES iterations to solve the monolithic system for the 3D problem, for different $\rho_s$. Comparison for (a) different meshes and (b) different time step sizes.



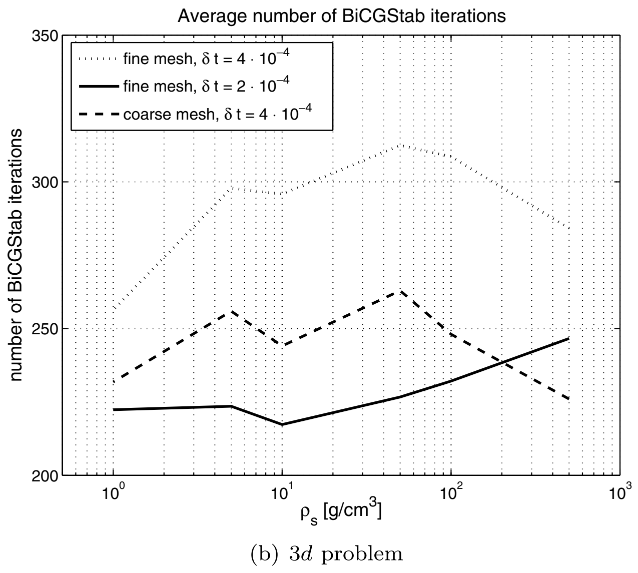
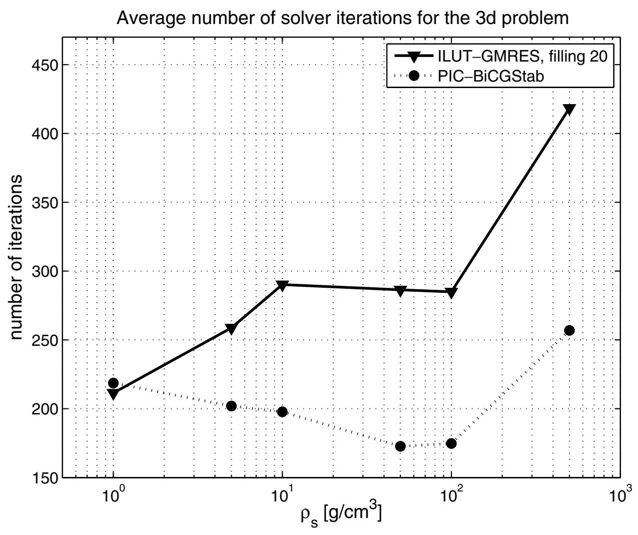


**Fig. 6.** Average number of BiCGStab iterations to solve the monolithic system for the (a) 2D and (b) 3D problem, for different $\rho_s$, meshes and time step sizes.

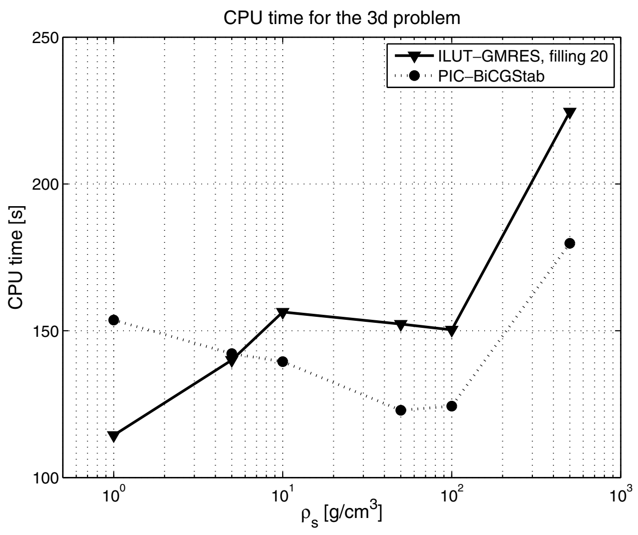
commented in Section 4, the best choice will strongly depend on the available computer memory. Our purpose is rather to show how ILUT preconditioners behave as the structure density approaches the fluid one.

For the 2D problem, the meshes are the same used for the tests in Section 6.1. For the 3D case we considered two unstructured meshes: the coarse one with average element size $h = 0.14$ (4347 nodes and 21163 tetrahedra) and the fine one with average element size $h = 0.12$ (6452 nodes and 32190 tetrahedra). The tolerance used for the iterative solvers is $10^{-4}$.

In Figs. 4 and 5, we observe the number of GMRES iterations for the bi-dimensional and three-dimensional problems, respectively,

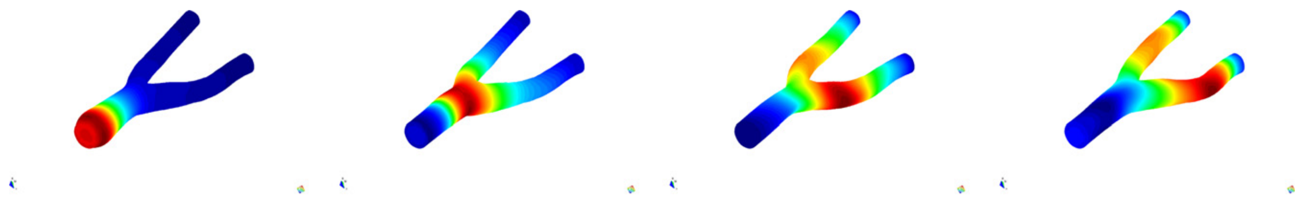


(a) Solver iterations



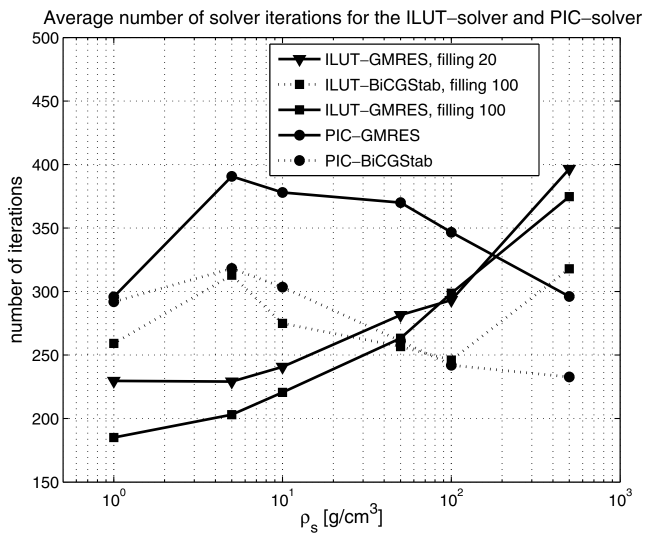
(b) CPU time

**Fig. 7.** (a) Average number of solver iterations and (b) CPU time for the ILUT-solver and the PIC methods as the structure density varies. (a) Solver iterations and (b) CPU time.

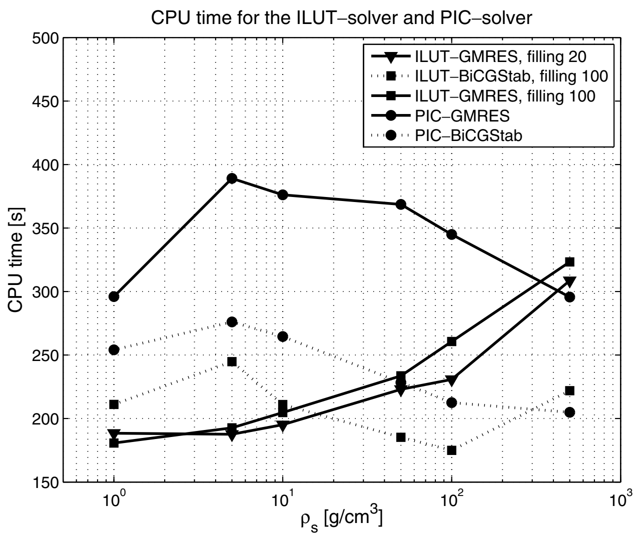


(a) $t = 3$ ms (b) $t = 6$ ms (c) $t = 9$ ms (d) $t = 12$ ms

**Fig. 8.** Propagation of the initial pressure pulse, moving from the inflow to the outflow section. Solution at every 3 ms.



(a) Solver iterations



(b) CPU time

**Fig. 9.** (a) Average number of solver iterations and (b) CPU time for the ILUT-solver and the PIC methods as the structure density varies.

on two different meshes and with two different time step sizes ($\delta t = 2 \times 10^{-4}$ s and $\delta t = 4 \times 10^{-4}$ s). Refining the mesh causes an increase in the iterations number, while the number of iterations decreases with the time step. This can be explained by the fact that the starting point for the GMRES method is the solution at the previous time step. However, the difference in the number of iterations with respect to the mesh size and the time step reduces as the Krylov space dimension gets bigger and as the added-mass effect becomes important. For both problems increasing the maximum dimension of the Krylov space ensures faster convergence of the GMRES method, because it reduces the re-starting of the method. Furthermore, the algorithm shows better convergence properties for problems with large added-mass effect.

The convergence of the ILUT-BiCGStab algorithm for the 2D and 3D case is shown in Fig. 6a and b, respectively. ILUT-BiCGStab shows the same behavior than ILUT-GMRES in the 2D problem (Fig. 6a), while the trend is more irregular for the 3D test (Fig. 6b).

As a conclusion, non-modular ILUT preconditioners are suitable for large added-mass problems, because they do not exhibit the ill behavior of the DN preconditioner as $\rho_s/\rho_f$ decreases (reported in Fig. 2).

## 6.4. Comparison between the ILUT-solver and PIC-solver

We compare now CPU cost and number of iterations of the two non-modular preconditioners with respect to the structure density for the 3D straight artery. The solver iterations and CPU cost for the ILUT-GMRES and PIC-BiCGStab methods are reported in Fig. 7. For large added-mass effect, ILUT-GMRES requires less CPU cost whereas PIC-BiCGStab is cheaper for larger values of $\rho_s$. The CPU cost of ILUT-GMRES decreases as the added-mass effect becomes more important while the BiCGStab method exhibits a slight increase of CPU cost.

## 7. Numerical results for the carotid bifurcation

Our goal is now to simulate a pressure wave in the carotid bifurcation using the same fluid and solid properties as in the straight pipe case. The geometry is a realistic one first used in [23]. The fluid and the structure are initially at rest and the same Neumann boundary conditions of the straight pipe are imposed at both the inlet and the outlet. The average inflow diameter is 0.67 cm, the time step used is $\delta t = 2 \times 10^{-4}$ s and the time interval is $[0, 0.012]$ s. Fig. 8 shows the fluid pressure together with the structural deformation amplified by a factor 10 at time $t = 3, 6, 9, 12$ ms.

Again we choose a conforming space discretization between fluid and structure: stabilized $\mathbb{P}_1 - \mathbb{P}_1$ finite elements for the fluid and $\mathbb{P}_1$ finite elements for the structure.

## 7.1. Comparison between the ILUT-solver, PIC-solver and DN-GMRES methods

We first compare the ILUT-solver and the PIC methods. Numerical tests for the PIC method similar to those for the ILUT-GMRES in the previous section can be found in [5] for inf–sup stable elements. In particular, we consider the ILUT-BiCGStab method, the ILUT-GMRES one with different fill-ins for the preconditioners,

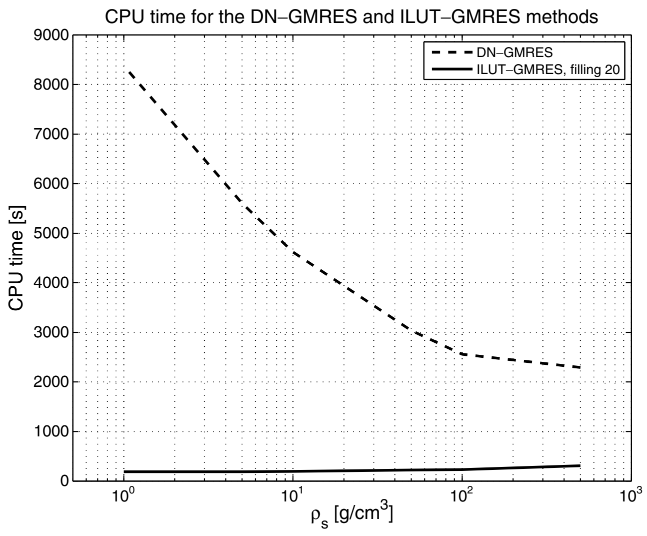
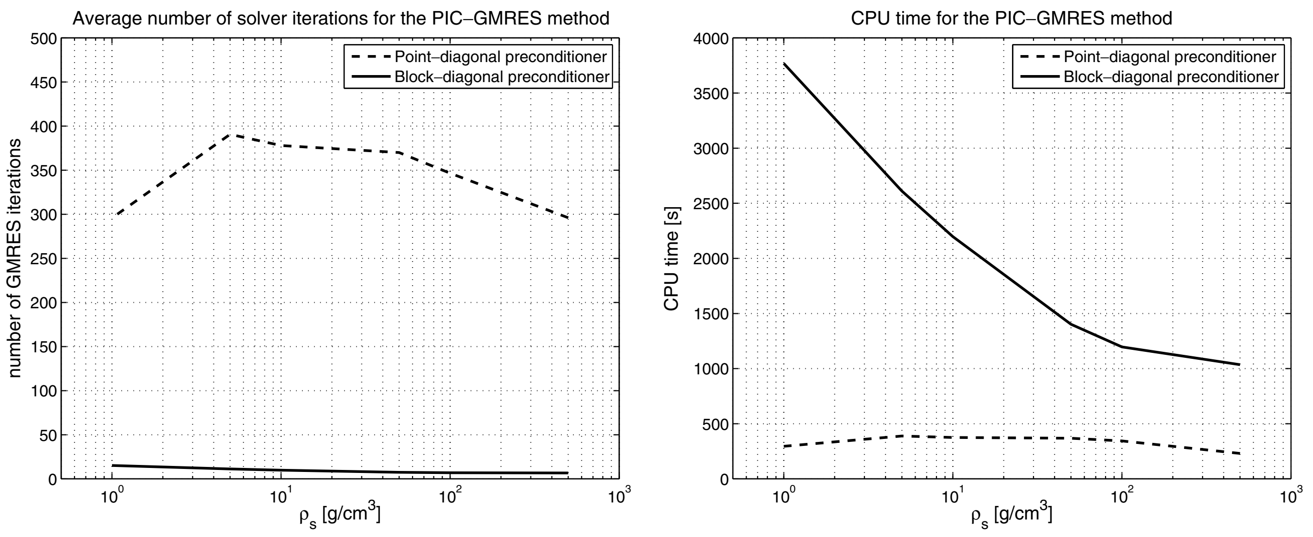


**Fig. 11.** CPU time for the ILUT-GMRES and the DN-GMRES methods as the structure density varies.



(a) Solver iterations (b) CPU time

**Fig. 10.** (a) Average number of GMRES iterations and (b) CPU time for the PIC-GMRES method with different preconditioners as the structure density varies.

the PIC-GMRES and PIC-BiCGStab algorithms. The tolerance for the iterative method is $10^{-4}$ for all the schemes. When the GMRES method is adopted the maximum dimension of the Krylov base is set to 40. The unstructured mesh we used has diameter $h = 0.11$ (8737 nodes and 40814 tetrahedra).

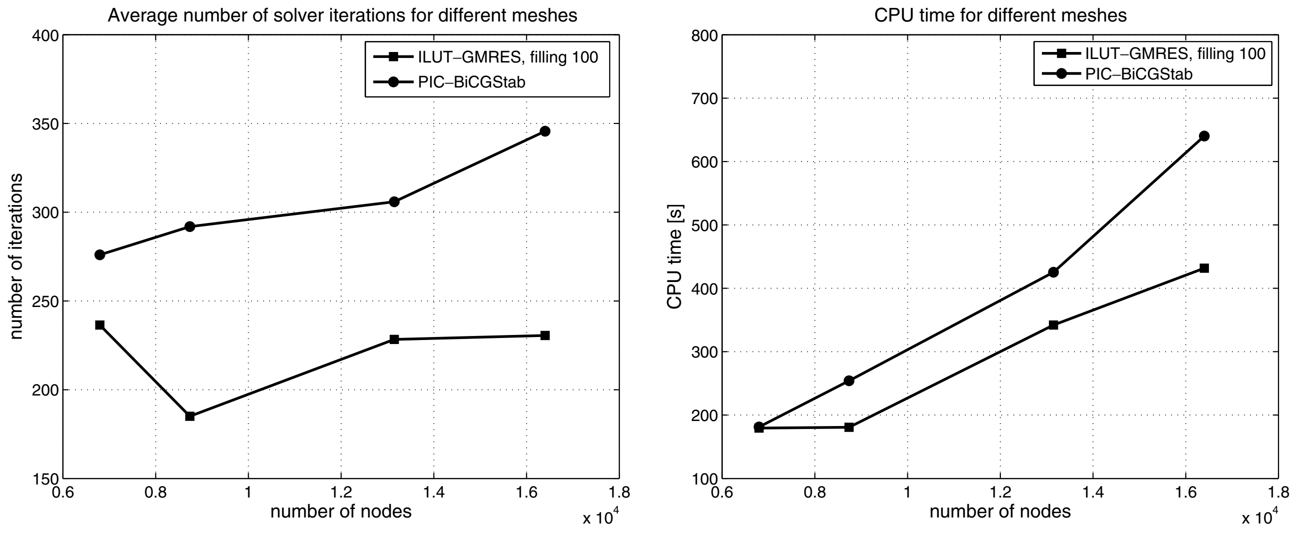
Fig. 9a shows the average number of solver iterations for the structure densities

$$\rho_s = 500, 100, 50, 10, 5, 1 \text{ g/cm}^3.$$

As already noticed in Section 6.3, the decreasing of the structure density improves the performances of the ILUT-GMRES method. Moreover, increasing the fill-in of the preconditioners reduces the number of GMRES iterations up to $\rho_s = 100$. This reduction in the number of iterations does not correspond to a decrease in the CPU time for $\rho_s > 1$, as it can be seen in Fig. 9b. In fact, the more accurate ILU factorizations require fewer iterations to converge but the cost to compute the incomplete factors (and sometimes the overall CPU cost) increases. For low structure densities the ILUT-BiCGStab behaves worse than the ILUT-GMRES. In any case, both methods have very similar CPU cost for large added-mass effect problems. The choice of the iterative solver (GMRES vs. BiCGStab) will depend on the size of the problem and computer memory (see Section 4). While the PIC-BiCGStab method always converges in less iterations and faster than the PIC-GMRES.

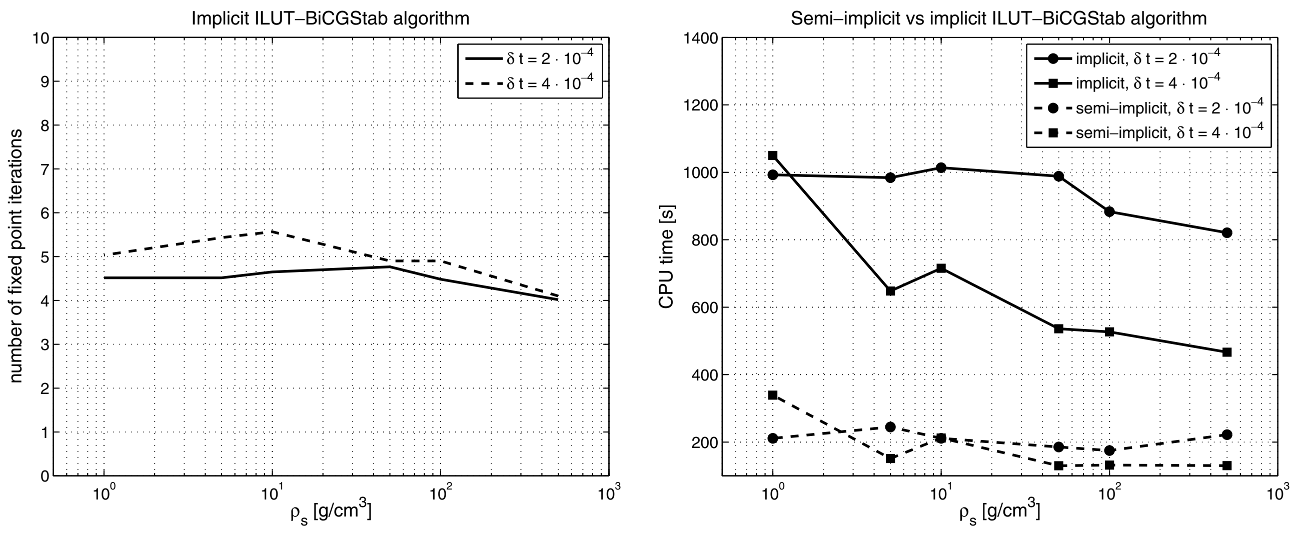
The PIC-solver methods whose results are reported in Fig. 9 employ the point-diagonal preconditioner to solve system (30b). We also considered the block-diagonal one. Obviously, this latter drastically reduces the number of solver iterations (Fig. 10a) but it is much more time consuming than the point-diagonal preconditioner (Fig. 10b).

The DN-GMRES algorithm is much more expensive in terms of CPU time than the other two methods. That is the reason why the results are not reported in the same graph but in a separated one (Fig. 11). Even though it represents an improvement with



(a) Solver iterations (b) CPU time

**Fig. 12.** (a) Average number of solver iterations and (b) CPU time for the ILUT-GMRES and the PIC-BiCGStab methods for different meshes.



(a) Nonlinear iterations (b) Total coupling iterations

**Fig. 13.** (a) Average number of fixed point iterations for the implicit version and (b) average CPU time per time step for the implicit and semi-implicit versions of the ILUT-BiCGStab method; values for different structure densities and time step sizes.

respect to the DN–Richardson algorithm, the DN-GMRES one is not competitive for realistic hemodynamics problem (see Fig. 12).

### *7.2. The ILUT-GMRES and the PIC-BiCGStab methods for hemodynamics problems*

Now we restrict our attention to the problem with the largest added-mass effect, i.e. we set $\rho_s = 1$. Fig. 3a reports the average number of solver iterations and Fig. 3b the CPU time required by the ILUT-GMRES and PIC-BiCGStab methods to solve the bifurcation problem on four different meshes. From the coarsest to the finest, the meshes we used have 6796, 8737, 13,148 and 16,402 nodes (31,138, 40,418, 62,879 and 79,528 tetrahedra, respectively). The PIC-BiCGStab method takes always more iterations to converge than the ILUT-GMRES one. The gap between the iterations number seems to increase with the refinement of the mesh. The CPU times needed by the two methods to complete the simulation show the same tendency. Thus, the ILUT-GMRES algorithm remains the less time-consuming also when the size of the problem increases.

### *7.3. The ILUT-solver: implicit and semi-implicit versions*

As done in Section 6.2 for the DN-GMRES method and the 2D straight artery, we show the efficiency of a semi-implicit treatment of the nonlinearity for ILUT preconditioners. We solved the carotid bifurcation problem with ILUT-BiCGStab. We considered two different time step values and all the structure densities specified in Section 7.1. Fig. 13a shows the average number of fixed point iterations (with tolerance $10^{-2}$) for the implicit treatment of the nonlinearity. The CPU cost is reported in Fig. 13b. For the implicit algorithm, the number of nonlinear iterations is fairly insensitive to structure density variations whereas the CPU cost reduces when $\rho_s$ increases. In any case, the computational savings associated to a semi-implicit treatment of the nonlinearity are clear in all situations.

## 8. Conclusions

In this work, we focused on the numerical simulation of FSI problems characterized by a strong added-mass effect. We took into account two different preconditioners for the coupled system obtained after linearization and full discretization of the FSI problem.

The first one is the classical Dirichlet–Neumann preconditioner. Two modular algorithms based on that preconditioner (the DN–Richardson and the DN-GMRES ones) have been considered. The reduction factor for the DN-GMRES method has been obtained for a model problem.

The second preconditioner is a non-modular ILUT preconditioner for the whole FSI system. We have introduced an appropriate monolithic formulation to be used with this preconditioner. Several aspects of this formulation have been also discussed in [18,35].

The theoretical negative impact of the added-mass effect on the reduction factor agrees with the numerical experiments. The performances of DN–Richardson and DN-GMRES have been compared to those of two methods (ILUT-GMRES and ILUT-BiCGStab) yielded by the non-modular ILUT preconditioner for the whole FSI system.

Another non-modular approach has been considered: the PIC scheme presented in [5], here extended to the case of *d*-dimensional structure and the use of stabilized finite elements methods.

The advantages of the explicit treatment for the nonlinearities of the FSI problem have been underlined. Thus, we dealt with the semi-implicit versions of all the methods mentioned above. This allowed us to focus on the fluid–structure coupling and on the effects of the added-mass.

We have carried out a broad set of numerical experiments. For problems with large added-mass effect we can draw the following conclusions:

- The DN-GMRES algorithm represents an improvement of the DN–Richardson one. However, they both perform well in case of high structure densities but suffer in case of critical added-mass effects.
- Unlike the DN-algorithms, the performance of the ILUT-solver methods is not deteriorated when the structure density approaches the fluid one. This good behavior in the large added-mass effect range pays off for the loss of modularity, also in the case of the PIC methods.
- The ILUT-solver method proved to be the least expensive in terms of CPU time for large problems. The PIC scheme is very competitive for smaller problems. Anyways, both non-modular preconditioners prove to be much more efficient than the modular DN-algorithm approach for the applications under consideration.
- A clear reduction of the CPU cost can be attained by considering a semi-implicit treatment of the nonlinearities.

**Acknowledgements**

Santiago Badia acknowledges the support of the European Community through the Marie Curie contract NanoSim (MOIF-CT-2006-039522). Annalisa Quaini acknowledges the support of the Swiss National Science Foundation under Grant 200021-109378/1.

## References

[1] J. Atanga, D. Silvester, Iterative methods for stabilized mixed velocity–pressure finite elements, Int. J. Numer. Methods Fluids 14 (1992) 71–81.
[2] S. Badia, R. Codina, Analysis of a stabilized finite element approximation of the transient convection–diffusion equation using an ALE framework, SIAM J. Numer. Anal. 44 (5) (2006) 2159–2197.
[3] S. Badia, R. Codina, On some fluid–structure iterative algorithms using pressure segregation methods. Application to aeroelasticity, Int. J. Numer. Methods Engrg. 72 (2007) 46–71.
[4] S. Badia, F. Nobile, C. Vergara, Fluid–structure partitioned procedures based on Robin transmission conditions, J. Comput. Phys. (2008), doi:10.1016/j.jcp.2008.04.006.
[5] S. Badia, A. Quaini, A. Quarteroni, Splitting methods based on algebraic factorization for fluid–structure interaction, SIAM J. Sci. Comput. 30 (4) (2008) 1778–1805.
[6] F. Ben Belgacem, The mortar finite element method with lagrange multipliers, Numer. Math. 84 (1999) 173–197.
[7] F. Brezzi, M. Fortin, Mixed and Hybrid Finite Element Methods, Springer-Veralg, 1991.
[8] F. Brezzi, M. Fortin, L.D. Marini, Error analysis of constant pressure aprroximations of Darcy's law, Comput. Methods Appl. Mech. Engrg. 195 (2006) 1547–1559.
[9] F. Brezzi, T.J.R. Hughes, E. Süli, Variational approximation of flux in conforming finite element methods for elliptic partial differential equations: a model problem, Rend. Mat. Acc. Lincei s 9 (12) (2001) 167–183.
[10] E. Burman, M.A. Fernández, Stabilized explicit coupling for fluid–structure interaction using Nitsche's method, C.R. Acad. Sci. Paris Sér. I Math. 345 (2007) 467–472.
[11] P. Causin, J.F. Gerbeau, F. Nobile, Added-mass effect in the design of partitioned algorithms for fluid–structure problems, Comput. Methods Appl. Mech. Engrg. 194 (42–44) (2005) 4506–4527.
[12] R. Codina, Stabilized finite element approximation of transient incompressible flows using orthogonal subscales, Comput. Methods Appl. Mech. Engrg. 191 (2002) 4295–4321.
[13] S. Deparis, M. Discacciati, G. Fourestey, A. Quarteroni, Fluid–structure algorithms based on Steklov–Poincaré operators, Comput. Methods Appl. Mech. Engrg. 195 (41–43) (2006) 5797–5812.
[14] H. Elman, D. Silvester, A. Wathen, Finite Elements and Fast Iterative Solvers, Oxford Science Publications, 2005.
[15] C. Farhat, P. Geuzaine, G. Brown, Application of a three-field nonlinear fluid–structure formulation to the prediction of the aeroelastic parameters of an F-16 fighter, Comput. Fluids 32 (2003) 3–29.
[16] M.A. Fernández, J.F. Gerbeau, C. Grandmont, A projection semi-implicit scheme for the coupling of an elastic structure with an incompressible fluid, Int. J. Numer. Methods Engrg. 69 (4) (2007) 794–821.

[17] M.A. Fernández, M. Moubachir, A Newton method using exact Jacobians for solving fluid–structure coupling, Comput. Struct. 83 (2–3) (2005) 127–142.

[18] C. Figueroa, I. Vignon-Clementel, K. Jansen, T. Hughes, C. Taylor, A coupled momentum method for modeling blood flow in three-dimensional deformable arteries, Comput. Methods Appl. Mech. Engrg. 195 (2006) 5685–5706.

[19] B. Fisher, A. Ramage, D.J. Silvester, A.J. Wathen, On parameter choice and iterative convergence for stabilised discretisations of advection–diffusion problems, Comput. Methods Appl. Mech. Engrg. 179 (1999) 179–195.

[20] J.F. Gerbeau, M. Vidrascu, A quasi-Newton algorithm based on a reduced model for fluid–structure interaction problems in blood flows, Math. Modell. Numer. Anal. 37 (4) (2003) 631–648.

[21] M. Heil, An efficient solver for the fully coupled solution of large-displacement fluid–structure interaction problems, Comput. Methods Appl. Mech. Engrg. 193 (2004) 1–23.

[22] T.J.R. Hughes, Multiscale phenomena: Green's function, the Dirichlet-to-Neumann formulation, subgrid scale models, bubbles and the origins of stabilized formulations, Comput. Methods Appl. Mech. Engrg. 127 (1995) 387–401.

[23] G. Karner, K. Perktold, M. Hofer, D. Liepsch, Flow characteristics in an anatomically realistic compliant carotid artery bifurcation model, Methods Biomech. Biomed. Engrg. 2 (39–41) (1999) 171–185.

[24] H.G. Matthies, J. Steindorf, Partitioned strong coupled algorithms for fluid–structure interaction, Comput. Struct. 81 (2003) 805–812.

[25] C. Michler, E.H. van Brummelen, R. de Borst, An interface Newton–Krylov solver for fluid–structure interaction, Int. J. Numer. Methods Fluids 47 (10–11) (2005) 1189–1195.

[26] D.P. Mok, W.A. Wall, Partitioned analysis schemes for transient interaction of incompressible flows and nonlinear flexible structures, in: W.A. Wall, K.U. Bletzinger, K. Schweizerhof (Eds.), Trends in Computational Structural Mechanics, CIMNE, Barcelona, Spain, 2001.

[27] H.J.-P. Moran, R. Ohayon, Fluid–Structure Interaction: Applied Numerical Methods, John Wiley & Sons, 1995.

[28] F. Nobile. Numerical Approximation of Fluid–Structure Interaction problems with application to Haemodynamics. Ph.D. Thesis, École Polytechnique Fédérale de Lausanne, 2001.

[29] F. Nobile, C. Vergara, An effective fluid–structure interaction formulation for vascular dynamics by generalized Robin conditions, SIAM J. Sci. Comput. 30 (2) (2008) 731–763.

[30] S. Piperno, C. Farhat, Partitioned procedures for the transient solution of coupled aeroelastic problems – Part II: energy transfer analysis and three-dimensional applications, Comput. Methods Appl. Mech. Engrg. 190 (2001) 3147–3170.

[31] A. Quarteroni, A. Valli, Domain Decomposition Methods for Partial Differential Equations, Oxford Science Publications, 1999.

[32] Y. Saad. SPARSKIT: A basic tool for sparse matrix computation. Technical Report CSRD TR 1029, CSRD, University of Illinois, 1990.

[33] Y. Saad, Iterative Methods for Sparse Linear Systems, PWS Publishing, Boston, MA, 1996.

[34] P.J. Silvester, N. Kechkar, Stabilised bilinear-constant velocity–pressure finite elements for the conjugate gradient solution of the Stokes problem, Comput. Methods Appl. Mech. Engrg. 79 (1990) 71–86.

[35] P. Le Tallec, J. Mouro, Fluid structure interaction with large structural displacements, Comput. Methods Appl. Mech. Engrg. 190 (2001) 3039–3067.

[36] T. Washio, T. Hisada, H. Watanabe, T.E. Tezduyar, A robust preconditioner for fluid–structure interaction problems, Comput. Methods Appl. Mech. Engrg. 194 (39–41) (2005) 4027–4047.