A Parallel Solver for FSI Problems with Fictitious Domain Approach

Daniele Boffi 1,2, Fabio Credali 1,2,*, Lucia Gastaldi 3 and Simone Scacchi 4

1 Computer, Electrical and Mathematical Sciences and Engineering Division, King Abdullah University of Science and Technology, Thuwal 23955, Saudi Arabia
2 Dipartimento di Matematica “F. Casorati”, Università degli Studi di Pavia, Via Ferrata 5, 27100 Pavia, Italy
3 Dipartimento di Ingegneria Civile, Architettura, Territorio, Ambiente e di Matematica, Università degli Studi di Brescia, Via Branze 43, 25123 Brescia, Italy
4 Dipartimento di Matematica “F. Enriques”, Università degli Studi di Milano, Via Saldini 50, 20133 Milano, Italy
* Correspondence: fabio.credali@kaust.edu.sa

Abstract: We present and analyze a parallel solver for the solution of fluid structure interaction problems described by a fictitious domain approach. In particular, the fluid is modeled by the non-stationary incompressible Navier–Stokes equations, while the solid evolution is represented by the elasticity equations. The parallel implementation is based on the PETSc library and the solver has been tested in terms of robustness with respect to mesh refinement and weak scalability by running simulations on a Linux cluster.

Keywords: fluid–structure interactions; fictitious domain; preconditioners; parallel solver

MSC: 65N30; 65N12; 74F10; 65F08

1. Introduction

The analysis of fluid–structure interaction (FSI) problems is important for several applications in science and engineering; typical examples that we have in mind are, for instance, the study of fluid-dynamics of heart valves, particulate flows, and propagation of free surfaces. FSI problems are challenging both from the mathematical and computational point of view: the difficulties originate from the necessity of handling interactions between several objects and the presence of nonlinear terms in the governing equations. As a consequence, in past years, several approaches have been presented for the numerical modeling of FSI problems: among those, we mention the Arbitrary Lagrangian Eulerian formulation (ALE) [1–4], the unfitted Nitsche method [5], the level set formulation [6], and the fictitious domain approach [7,8]. It is important to notice that each method is effective for a selected class of problems, since there is no method that can be applied to all possible situations.

Our fictitious domain approach with distributed Lagrange multiplier was considered first in [9] as evolution of the immersed boundary method [10,11], originally introduced by C. Peskin during the Seventies for cardiac simulations of heart valves. This approach is based on the idea that the fluid domain is fictitiously extended also to the region occupied by the immersed body. In particular, the fluid is governed by the incompressible time dependent Navier–Stokes equations, while the structure can be characterized by either linear or nonlinear constitutive laws for viscous elastic materials. The fluid dynamics is studied on a fixed Eulerian mesh, while we resort to a Lagrangian description to represent the motion and the deformation of the immersed structure; the Lagrangian frame is built by introducing a reference domain which is mapped, at each time step, into the actual position of the body.

In order to get accurate results, simulations of FSI problems require, in general, huge computational resources, both in terms of time and memory. In this framework, the design
of robust parallel solvers is an important tool to perform, in a reasonable amount of time, computations involving a large number of time steps and fine space discretizations. Several works are focused on this task, mainly in the setting of ALE formulations \cite{12–18} or by applying a variational transfer in order to couple fluid and solids \cite{19,20}. To the best of our knowledge, preconditioners for the fictitious domain formulation with distributed Lagrange multiplier have been introduced only recently in \cite{21}.

Here, we continue the analysis of the parallel solver introduced in \cite{21}: we focus our attention on the robustness with respect to mesh refinement, with different choices of time step, and to the weak scalability. We also describe some implementation issues. The finite element discretization is performed by choosing the \((Q_2,P_1)\) element for velocities and pressures of the fluid and the \(Q_1\) element for the structure variables; the time marching scheme is a first order semi-implicit finite difference algorithm. Moreover, the fluid–structure coupling matrix is assembled by exact computations over non-matching meshes as described in \cite{22}; the computational costs of this procedure will be described. At each time step, the linear system arising from the discretization is solved by the GMRES method, combined with either a block-diagonal or a block-triangular preconditioner. Our parallel implementation is based on the PETSc library from Argonne National Laboratory \cite{23,24} and on the parallel direct solver Mumps \cite{25,26}, which is used to invert the diagonal blocks in the block preconditioners.

After recalling some functional analysis notation, in Section 3, we present the mathematical model describing fluid structure interaction problems in the spirit of the fictitious domain approach. In Section 4, we describe the numerical method we implemented for our simulations and in Section 5 we introduce two possible choices of preconditioner for our parallel solver. Finally, in Section 6 we present some numerical tests aiming at assessing the robustness with respect to mesh refinement and the weak scalability.

2. Notation

We recall some useful functional analysis notation \cite{27}. Let us consider an open and bounded domain \(D\). The space of square integrable functions is denoted by \(L^2(D)\), with scalar product \((\cdot,\cdot)_D\). In particular, \(L^2_0(D)\) is the subspace of functions with null mean over \(D\). We denote Sobolev spaces by \(W^{s,p}(D)\); with \(s \in \mathbb{R}\) referring to the differentiability and \(p \in [1,\infty]\) to the summability exponent. When \(p = 2\), we adopt the classical notation \(H^s(D) = W^{s,2}(D)\). In addition, \(H^1_D(D) \subset H^1(D)\) is the space of functions with zero trace on the boundary \(\partial D\). For vector valued spaces the dimension is explicitly indicated.

3. Continuous Formulation

We simulate fluid–structure interaction problems characterized by a visco-elastic incompressible solid body immersed in a viscous incompressible fluid. We denote by \(\Omega^f_t\) and \(\Omega^s_t\) the two regions in \(\mathbb{R}^d\) (with \(d = 2,3\)) occupied by the fluid and the structure, respectively, at the time instant \(t\); the interface between these two regions is denoted by \(\Gamma_t\). The evolution of such a system takes place inside \(\Omega\), that is the union of \(\Omega^f_t\) and \(\Omega^s_t\): this new domain is independent of time and we assume that it is connected and bounded with Lipschitz continuous boundary \(\partial \Omega\). It is worth mentioning that, even if we are going to consider only thick solids, also the evolution of thin structures can be treated by our mathematical model.

The dynamics of the fluid is studied by considering an Eulerian description, associated with the variable \(x\). On the other hand, the evolution of the immersed body is modeled by a Lagrangian description: we introduce the solid reference domain \(B\), associated with the variable \(s\), so that the deformation can be represented by the map \(X : B \rightarrow \Omega^s_t\). This means that \(x \in \Omega^s_t\) is the image at the time \(t\) of a certain point \(s \in B\) and the motion of the structure is represented by the kinematic equation

\[
\mathbf{u}_s(x,t) = \frac{\partial X}{\partial t}(s,t) \quad \text{for} \quad x = X(s,t),
\]
where the material velocity is denoted by $u_s$. The deformation gradient $\nabla_s X$ is denoted by $F$ and $J(s, t) = \det F(s, t)$. Since we are assuming that the solid body is incompressible, the determinant $J$ is constant in time.

In our model, we consider a Newtonian fluid with density $\rho_f$ and viscosity $\nu_f > 0$, so that the Cauchy stress tensor can be written as

$$\sigma_f = -p_f I + \nu_f \varepsilon(u_f),$$

where $u_f$ denotes the velocity of the fluid and $p_f$ its pressure, and where $I$ denotes the identity tensor. Therefore, the dynamics in $\Omega_f^t$ is governed by the incompressible Navier–Stokes equations

$$\rho_f \left( \frac{\partial u_f}{\partial t} + u_f \cdot \nabla u_f \right) = \operatorname{div} \sigma_f$$

$$\operatorname{div} u_f = 0.$$

For the solid, we consider a viscous-hyperelastic material with density $\rho_s$ and viscosity $\nu_s > 0$; for this type of materials, the Cauchy stress tensor $\sigma_s$ can be seen as the sum of two contributions: a viscous part, similar to the one of the fluid

$$\sigma_s^v = -p_s I + \nu_s \varepsilon(u_s),$$

and an elastic part, which can be written, moving from Eulerian to Lagrangian setting, in terms of the Piola–Kirchhoff stress tensor $P$

$$P(F(s, t)) = J(s, t) \sigma_e(s, t) F(s, t)^{-\top}$$

for $x = X(s, t)$.

In particular, hyperelastic materials are characterized by a positive energy density $W(F)$, which is related with $P$ since $P(F) = \partial W / \partial F$. Consequently, the elastic potential energy of the solid body can be expressed as

$$E(X(t)) = \int_B W(F(s, t)) \, ds.$$

Finally, the system is described by the following equations in strong form

$$\rho_f \left( \frac{\partial u_f}{\partial t} + u_f \cdot \nabla u_f \right) = \operatorname{div} \sigma_f \quad \text{in } \Omega_f^t$$

$$\operatorname{div} u_f = 0 \quad \text{in } \Omega_f^t$$

$$\rho_s \frac{\partial^2 X}{\partial t^2} = \operatorname{div}_s \left( J \sigma_s^v F^{-\top} + P(F) \right) \quad \text{in } B$$

$$\operatorname{div} u_s = 0 \quad \text{in } \Omega_s^t$$

and completed by two transmission conditions to enforce continuity of velocity and stress along the interface $\Gamma_t$

$$u_f = u_s \quad \text{on } \Gamma_t$$

$$\sigma_f n_f = -(\sigma_s^v + J^{-1} P F^\top) n_s \quad \text{on } \Gamma_t,$$

where $n_f$ and $n_s$ denote the outer normals to $\Omega_f^t$ and $\Omega_s^t$, respectively. Moreover, we consider the following initial and boundary conditions
\[
\begin{align*}
\mathbf{u}_f(0) &= \mathbf{u}_{f,0} \quad \text{in } \Omega_0^f \\
\mathbf{u}_s(0) &= \mathbf{u}_{s,0} \quad \text{in } \Omega_0^s \\
X(0) &= X_0 \quad \text{in } B \\
\mathbf{u}_f &= 0 \quad \text{on } \partial \Omega.
\end{align*}
\tag{9}
\]

The idea of the fictitious domain approach is to extend the first two equations in (7) to the whole domain $\Omega$ so that all the involved variables are defined on a domain which is independent of time. Consequently, following [9], we introduce two new unknowns

\[
\mathbf{u} = \begin{cases}
\mathbf{u}_f & \text{in } \Omega_f^f \\
\mathbf{u}_s & \text{in } \Omega_f^s
\end{cases}, \quad
p = \begin{cases}
p_f & \text{in } \Omega_f^f \\
p_s & \text{in } \Omega_f^s.
\end{cases}
\tag{10}
\]

In this new setting, (1) becomes a constraint on $\mathbf{u}$, since we have to impose that

\[
\mathbf{u}(X(s,t),t) = \frac{\partial X}{\partial t}(s,t) \quad \text{for } s \in B.
\tag{11}
\]

This condition can be weakly enforced by employing a distributed Lagrange multiplier. To this end, we set $\Lambda = (H^1(B))^d$, the dual space of $H^1(B)^d$, and denote by $\langle \cdot, \cdot \rangle$ the duality pairing between $\Lambda$ and $H^1(B)^d$. Notice that, for $Y \in H^1(B)^d$, we have the following property

\[
\langle \mu, Y \rangle = 0 \quad \forall \mu \in \Lambda \implies Y = 0.
\tag{12}
\]

At this point, following [9,28], the equations in (7), endowed with conditions (8) and (9), can be written in variational form.

**Problem 1.** For given $\mathbf{u}_0 \in H^1_0(\Omega)^d$ and $X_0 \in W^{1,\infty}(B)$, find $\mathbf{u}(t) \in H^1_0(\Omega)^d$, $p(t) \in L^2_0(\Omega)$, $X(t) \in H^1(B)^d$, and $\lambda(t) \in \Lambda$ such that for almost all $t \in (0,T)$:

\[
\begin{align*}
\rho_f \left( \frac{\partial}{\partial t} \mathbf{u}(t), \mathbf{v} \right)_{\Omega} + b(\mathbf{u}(t), \mathbf{u}(t), \mathbf{v}) + a(\mathbf{u}(t), \mathbf{v}) - (\text{div } \mathbf{v}, p(t))_{\Omega} + (\lambda(t), \mathbf{v}(X(\cdot,t))) &= 0 \quad \forall \mathbf{v} \in H^1_0(\Omega)^d \\
(\text{div } \mathbf{u}(t), q)_{\Omega} &= 0 \quad \forall q \in L^2_0(\Omega) \\
(p_s - \rho_f) \frac{\partial^2 X}{\partial t^2}(t), Y \rangle_B + (\mathbb{P}(\mathbf{F}(t)), \nabla_s Y)_B - \langle \lambda(t), Y \rangle &= 0 \quad \forall Y \in H^1(B)^d \\
\langle \mu, \mathbf{u}(X(\cdot,t),t) - \frac{\partial X}{\partial t}(t) \rangle_B &= 0 \quad \forall \mu \in \Lambda \\
\mathbf{u}(x,0) &= \mathbf{u}_0(x) \quad \text{in } \Omega \\
X(s,0) &= X_0(s) \quad \text{in } B.
\end{align*}
\]

In particular,

\[
a(\mathbf{u}, \mathbf{v}) = (\mathbf{v}, \mathbf{E}(\mathbf{u}), \mathbf{E}(\mathbf{v}))_{\Omega} \\
b(\mathbf{u}, \mathbf{v}, \mathbf{w}) = \frac{\rho_f}{2} (\mathbf{u} \cdot \nabla \mathbf{v}, \mathbf{w})_{\Omega} - (\mathbf{u} \cdot \nabla \mathbf{v}, \mathbf{w})_{\Omega}.
\tag{13}
\]

Moreover, $\nu$ is the extended viscosity with value $\nu_f$ in $\Omega_f^f$ and $\nu_s$ in $\Omega_f^s$. For our numerical tests, we are going to consider $\nu_f = \nu_s$ since it is a reasonable assumption for biological models [29].

For our simulations, we consider a simplified version of the problem: we drop the convective term of the Navier–Stokes equations and we assume that fluid and solid materials have the same density, i.e., $\rho_s = \rho_f$. We focus on this case since it is interesting to see how the solver behaves when this assumption is combined with a semi-implicit time advancing scheme in the setting of the fictitious domain approach. This is actually the
The time interval piecewise bilinear elements on quadrilaterals. Other stable combinations of finite element piecewise quadratic velocities and discontinuous piecewise linear pressures. 

\[ X \] can be replaced by the scalar product in \( L^2 \) spaces for our class of problems have been studied in [33], both from the theoretical and satisfying the inf-sup condition for the Stokes problem. In particular, we work with the \( V \) element spaces solid. For the fluid, we consider a partition \( T \) implicitly treated by considering the position of the structure at the previous time step as setting \( g \) denote the subdivision nodes by \( n \) depending on time, setting \( g^n = g(t_n) \), the time derivative is approximated as

\[
\frac{\partial g}{\partial t}(t_{n+1}) \approx \frac{g^{n+1} - g^n}{\Delta t}.
\] (14)

Moreover, the nonlinear coupling terms \( \langle \lambda(t), v(X(\cdot,t)) \rangle \) and \( \langle \mu, u(X(\cdot,t), t) \rangle \) are semi-implicitly treated by considering the position of the structure at the previous time step as \( \langle \lambda^{n+1}, v(X^n) \rangle \) and \( \langle \mu, u^{n+1}(X^n) \rangle \).

For the discretization in space, we work with quadrilateral meshes for both fluid and solid. For the fluid, we consider a partition \( T_h^\Omega \) of \( \Omega \) with meshsize \( h_\Omega \) and two finite element spaces \( V_h \subset H^1(\Omega)^d \) and \( Q_h \subset L^2(\Omega) \) for velocity and pressure, respectively, satisfying the inf-sup condition for the Stokes problem. In particular, we work with the \( (Q_2, P_1) \) pair, which is one of the most popular Stokes elements, making use of continuous piecewise quadratic velocities and discontinuous piecewise linear pressures.

For the solid domain, we choose a partition \( T_h^B \) of \( B \) with meshsize \( h_B \), independent of \( T_h^\Omega \). We then consider two finite dimensional spaces \( S_h \subset H^1(B)^d \) and \( \Lambda_h \subset \Lambda \). We assume that \( S_h = \Lambda_h \) and we approximate both the mapping \( X \) and the Lagrange multiplier \( \lambda \) with piecewise bilinear elements on quadrilaterals. Other stable combinations of finite element spaces for our class of problems have been studied in [33], both from the theoretical and the numerical point of view.

We notice that, since \( \Lambda_h \) is included in \( L^2(B)^d \), at a discrete level, the duality pairing can be replaced by the scalar product in \( L^2(B)^d \)

\[
\langle \mu_h, Y_h \rangle = (\mu_h, Y_h)_B \quad \forall \mu_h \in \Lambda_h, \forall Y_h \in S_h.
\] (15)

Therefore, we get the following fully discrete problem.
Problem 3. Given $u_{0,h} \in V_h$ and $X_{0,h} \in S_h$, for all $n = 1, \ldots, N$ find $u^n_h \in V_h$, $p^n_h \in Q_h$, $X^n_h \in S_h$, and $\lambda^n_h \in \Lambda_h$ fulfilling:

$$
\rho \frac{(u^{n+1}_h - u^n_h)}{\Delta t}, v_h \biggm|_{\Omega} + a(u^{n+1}_h, v_h) - (\text{div} v_h, p^{n+1}_h)_{\Omega} + \left( \lambda^{n+1}_h, v_h(X^n_h) \right)_B = 0 \quad \forall v_h \in V_h
$$

$$(\text{div} u^{n+1}_h, q_h)_{\Omega} = 0 \quad \forall q_h \in Q_h$$

$$(\mathbb{P}(\mathbb{P}^{n+1}_h), \nabla_s Y_h)_B - (\lambda^{n+1}_h, Y_h)_B = 0 \quad \forall Y_h \in S_h$$

$$(\mu_h, u^n_h(X^n_h) - \frac{X^{n+1}_h - X^n_h}{\Delta t})_B = 0 \quad \forall \mu_h \in \Lambda_h$$

$$u^0_h = u_{0,h} \quad X^0_h = X_{0,h}.$$  

Assuming for simplicity $\mathbb{P}(\mathbb{P}) = \mathbb{P}$, Problem 3 can be represented in matrix form as

$$
\begin{bmatrix}
A_f & -B^T & 0 & C_f(X^n_h)^T \\
-B & 0 & 0 & 0 \\
0 & 0 & A_s & -C_s \\
C_f(X^n_h)^T & 0 & -\frac{1}{\Delta t} C_s & 0
\end{bmatrix}
\begin{bmatrix}
u^{n+1}_h \\
p^{n+1}_h \\
X^{n+1}_h \\
\lambda^{n+1}_h
\end{bmatrix} =
\begin{bmatrix} g_1 \\
g_2 \\
g_2 \\
g_2 \end{bmatrix}
$$

with

$$A_f = \frac{\rho_f}{\Delta t} M_f + K_f$$

$$(M_f)_{ij} = \left( \phi_j, \phi_i \right)_{\Omega}, \quad (K_f)_{ij} = a(\phi_j, \phi_i)$$

$$B_{ki} = (\text{div} \phi_i, \psi_k)_{\Omega}$$

$$(A_s)_{ij} = \kappa \left( \nabla_s \chi_j, \nabla_s \chi_i \right)_B$$

$$(C_f(X^n_h))_{ij} = \left( \chi_{ij}, \phi_j(X^n_h) \right)_B, \quad (C_s)_{ij} = \left( \chi_{ij}, \chi_j \right)_B$$

$$g_1 = \frac{\rho_f}{\Delta t} M_f u^0_h, \quad g_2 = -\frac{1}{\Delta t} C_s X^0_h.$$

Here, $\phi_i$ and $\psi_k$ denote the basis functions of $V_h$ and $Q_h$ respectively, while $\chi_j$ are the basis functions of the space defined on $B$. We observe that, since $S_h = \Lambda_h$, $C_s$ is the mass matrix in $S_h$ or $\Lambda_h$.

We can see that the matrix in (16) splits into four blocks, defined as follows:

$$
A_{11} = \begin{bmatrix} A_f & -B^T \\
-B & 0 \\
0 & 0
\end{bmatrix}, \quad A_{12} = \begin{bmatrix} 0 & C_f(X^n_h)^T \\
0 & 0
\end{bmatrix}
$$

$$A_{21} = \begin{bmatrix} 0 & 0 \\
C_f(X^n_h)^T & 0
\end{bmatrix}, \quad A_{22} = \begin{bmatrix} A_s & -C_s \end{bmatrix}$$

where $A_{11}$ is related to the fluid dynamic, $A_{12}$ to the solid evolution, while $A_{12}$ and $A_{21}$ contain the coupling term.

Particular attention has to be paid to the assembly of the coupling matrix $C_f(X^n_h)$, since it involves the integration over $\mathcal{B}$ of solid and mapped fluid basis functions. In order to compute these integrals, we need to know how each element $E$ of $\mathcal{T}_h^S$ is mapped into the fluid domain. We implement an exact quadrature rule by computing, at each time step, the intersection between the fluid mesh $\mathcal{T}_h^F$ and a mapped solid element $X^n_h(E)$. In order to detect all the intersections, each solid element is tested against all the fluid elements making use of a bounding box technique, which, in this particular case, is trivial since the
fluid is discretized with a Cartesian mesh of squares; then the intersections are explicitly computed by means of the Sutherland–Hodgman algorithm. For more details about the procedure in a similar situation, we refer to [22]. The implementation of this composite rule is quite involved and it is not straightforwardly parallelizable.

In general, when $P(F)$ is nonlinear, we use a solver for nonlinear systems of equations such as the Newton iterator method.

5. Parallel Preconditioners

The design of an efficient parallel solver influences two aspects of the numerical method: first, the finite element matrices need to be assembled in parallel on each processor; second, the solution of the saddle point system arising from the discretization has to be solved saving computational resources, in terms of both memory and execution time. For this purpose, we implemented a Fortran90 code based on the library PETSc from Argonne National Laboratory [23,24]. Such library is built on the MPI standard and it offers advanced data structures and routines for the parallel solution of partial differential equations, from basic vector and matrix operations to more complex linear and nonlinear equation solvers. In our code, vectors and matrices are built and subassembled in parallel on each processor.

Our parallel solver adopts two possible choices of preconditioner:

- block-diagonal preconditioner
  \[
  \begin{bmatrix}
  A_{11} & 0 \\
  0 & A_{22}
  \end{bmatrix}
  \]

- block-triangular preconditioner
  \[
  \begin{bmatrix}
  A_{11} & 0 \\
  A_{21} & A_{22}
  \end{bmatrix}
  \]

We solve the linear system making use of the parallel GMRES method combined with the action of our preconditioners, which consists of the exact inversion of the diagonal blocks performed by the parallel direct solver Mumps [25,26].

6. Numerical Tests

The proposed preconditioners have been widely studied in [21] in terms of robustness with respect to mesh refinement, strong scalability and refinement of the time step. In this work, after reporting new results in terms of optimality, we analyze the weak scalability of our solver. We focus on both linear and nonlinear models describing the solid material.

In the GMRES solver, we adopt as stopping criterion a $10^{-8}$ reduction of the Euclidean norm of the relative residual and a restart parameter of 200. In the case of the nonlinear model, the stopping criterion adopted for the Newton method is a $10^{-6}$ reduction of the Euclidean norm of the relative residual.

Our simulations were run on the Shaheen cluster at King Abdullah University of Science and Technology (KAUST, Saudi Arabia). It is a Cray XC40 cluster constituted by 6174 dual sockets computing nodes, based on 16 core Intel Haswell processors running at 2.3 GHz. Each node has 128 GB of DDR4 memory running at 2300 MHz.

6.1. Linear Solid Model

We consider a quarter of the elastic annulus \( \{ x \in \mathbb{R}^2 : 0.3 \leq |x| \leq 0.5 \} \) included in \( \Omega = [0, 1]^2 \); in particular, the solid reference domain corresponds to the resting configuration of the body; that is

\[
B = \{ s = (s_1, s_2) \in \mathbb{R}^2 : s_1, s_2 \geq 0, 0.3 \leq |s| \leq 0.5 \}.
\]

The dynamics of the system is generated by stretching the annulus and observing how internal forces bring it back to the resting condition. In this case, \( \Omega_0 \) coincides with the stretched annulus. Four snapshots of the evolution are shown in Figure 1.
The solid behavior is governed by a linear model; therefore, \( P(F) = \kappa F \), with \( \kappa = 10 \). We choose fluid and solid materials with the same density \( \rho_f = \rho_s = 1 \) and the same viscosity \( \nu_f = \nu_s = 0.1 \). We impose no slip conditions for the velocity on the upper and right edge of \( \Omega \), while on the other two edges, we allow the motion of both fluid and structure along the tangential direction. Finally, the following initial conditions are considered

\[
\mathbf{u}(\mathbf{x}, 0) = 0, \quad \mathbf{X}(s, 0) = \left( \frac{s_1}{1.4}, 1.4 s_2 \right).
\]

Figure 1. Four snapshots of the evolution of the structure with linear constitutive law.

In Table 1, we report the results for the optimality test, where the robustness of the solver is studied by refining the mesh and keeping fixed the number of processors. In particular, we set the time step to \( \Delta t = 0.01 \) and the final time of our simulation to \( T = 2 \). The number of processors used for the simulation is 32. The time \( T_{ass} \) needed to assemble the matrix of the problem increases moderately, while the time \( T_{coup} \), needed for the assembly of the coupling matrix by computing the intersection between the involved meshes, exhibits a superlinear growth. In terms of preconditioners, we can see that block-diag is not robust with respect to mesh refinement since the number of GMRES iterations grows from 13 to 430; clearly, this phenomenon affects also the time \( T_{sol} \) we need to solve the system. On the other hand, block-tri is robust since the number of GMRES iterations remains bounded by 14 when the mesh is refined. Therefore, \( T_{sol} \) presents only a moderate growth and, for 1,074,054 dofs, it is 30 times smaller than the value we obtain for block-diag preconditioner.

Table 1. Refinements of the mesh in the linear solid model. The simulations are run on the Shaheen cluster. procs = number of processors; dofs = degrees of freedom; \( T_{ass} \) = CPU time to assemble the stiffness and mass matrices; \( T_{coup} \) = CPU time to assemble the coupling term; its = GMRES iterations; \( T_{sol} \) = CPU time to solve the linear system; \( T_{tot} \) = total simulation CPU time. The quantities \( T_{coup} \), its and \( T_{sol} \) are averaged over the time steps. All CPU times are reported in seconds.

<table>
<thead>
<tr>
<th>dofs</th>
<th>( T_{ass} ) (s)</th>
<th>( T_{coup} ) (s)</th>
<th>its</th>
<th>Block-diag ( T_{sol} ) (s)</th>
<th>7</th>
<th>( T_{tot} ) (s)</th>
<th>Block-tri ( T_{sol} ) (s)</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td>30,534</td>
<td>( 1.02 \times 10^{-2} )</td>
<td>( 9.98 \times 10^{-2} )</td>
<td>13</td>
<td>( 1.14 \times 10^{-1} )</td>
<td>42.01</td>
<td>( 6.93 \times 10^{-2} )</td>
<td>( 33.83 )</td>
<td></td>
</tr>
<tr>
<td>120,454</td>
<td>( 2.12 \times 10^{-2} )</td>
<td>( 1.09 \times 10^{-2} )</td>
<td>31</td>
<td>( 8.30 \times 10^{-1} )</td>
<td>390.90</td>
<td>( 2.40 \times 10^{-1} )</td>
<td>( 266.17 )</td>
<td></td>
</tr>
<tr>
<td>269,766</td>
<td>( 9.20 \times 10^{-2} )</td>
<td>( 7.60 \times 10^{-2} )</td>
<td>97</td>
<td>( 5.41 \times 10^{-1} )</td>
<td>2.55 \times 10^{3}</td>
<td>( 6.47 \times 10^{-1} )</td>
<td>( 1.65 \times 10^{3} )</td>
<td></td>
</tr>
<tr>
<td>478,470</td>
<td>( 1.31 \times 10^{-1} )</td>
<td>( 25.04 \times 10^{-1} )</td>
<td>192</td>
<td>( 18.75 \times 10^{-1} )</td>
<td>( 9.07 \times 10^{3} )</td>
<td>12</td>
<td>( 1.14 \times 10^{-1} )</td>
<td>( 5.24 \times 10^{3} )</td>
</tr>
<tr>
<td>746,566</td>
<td>( 1.23 \times 10^{-1} )</td>
<td>( 85.32 \times 10^{-1} )</td>
<td>422</td>
<td>( 67.92 \times 10^{-1} )</td>
<td>( 3.07 \times 10^{4} )</td>
<td>13</td>
<td>( 2.17 \times 10^{-1} )</td>
<td>( 1.75 \times 10^{4} )</td>
</tr>
<tr>
<td>1,074,054</td>
<td>( 1.81 \times 10^{-1} )</td>
<td>( 196.88 \times 10^{-1} )</td>
<td>430</td>
<td>( 97.19 \times 10^{-1} )</td>
<td>( 5.90 \times 10^{4} )</td>
<td>14</td>
<td>( 3.21 \times 10^{-1} )</td>
<td>( 4.00 \times 10^{4} )</td>
</tr>
</tbody>
</table>

The weak scalability of the proposed parallel solver is analyzed in Table 2. Again, we choose \( T = 2 \) and \( \Delta t = 0.01 \). We perform six tests by doubling both the global number of dofs and the number of processors. Thanks to the resources provided by PETSc, the time
The assembly of stiffness and mass matrices is perfectly scalable. On the other hand, the assembly procedure for the coupling matrix is much more complicated: in order to detect all the intersections between solid and fluid elements, the algorithm consists of two nested loops. For each solid element (outer loop), we check its position with respect to all the fluid elements (inner loop). In particular, only the outer loop is distributed over all the processors. Consequently, $T_{\text{coup}}$ is not scalable since the number of fluid dofs, analyzed in serial, increases at each test. We now discuss the behavior of the two proposed preconditioners. It is evident that block-diag is not scalable since the number of GMRES iteration drastically increases as we increase dofs and procs, clearly affecting $T_{\text{sol}}$ and $T_{\text{tot}}$. On the other hand, block-tri behaves well: even if it is not perfectly scalable, the number of iterations slightly increases from 8 to 18 and $T_{\text{sol}}$ ranges from $2.24 \times 10^{-1}$ s to 11.43 s.

**Table 2.** Weak scalability for the linear solid model. The simulations are run on the Shaheen cluster. Same format as Table 1.

<table>
<thead>
<tr>
<th>_procs</th>
<th>dofs</th>
<th>$T_{\text{ass}}$ (s)</th>
<th>$T_{\text{coup}}$ (s)</th>
<th>$T_{\text{sol}}$ (s)</th>
<th>$T_{\text{tot}}$ (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>68,070</td>
<td>$8.55 \times 10^{-2}$</td>
<td>3.95</td>
<td>6.25 $\times 10^{-1}$</td>
<td>933.43</td>
</tr>
<tr>
<td>8</td>
<td>135,870</td>
<td>1.00 $\times 10^{-1}$</td>
<td>5.23</td>
<td>2.16 $\times 10^{-3}$</td>
<td>9</td>
</tr>
<tr>
<td>16</td>
<td>269,766</td>
<td>1.01 $\times 10^{-1}$</td>
<td>8.77</td>
<td>10.23</td>
<td>11</td>
</tr>
<tr>
<td>32</td>
<td>539,926</td>
<td>9.24 $\times 10^{-2}$</td>
<td>59.27</td>
<td>108.05</td>
<td>14</td>
</tr>
<tr>
<td>64</td>
<td>1,074,054</td>
<td>1.90 $\times 10^{-1}$</td>
<td>48.00</td>
<td>113.59</td>
<td>18</td>
</tr>
<tr>
<td>128</td>
<td>2,152,614</td>
<td>1.90 $\times 10^{-1}$</td>
<td>98.63</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

**6.2. Nonlinear Solid Model**

For this test, we set again the fluid domain $\Omega$ to be the unit square; the immersed solid body is a bar represented, at resting configuration, by the rectangle $B = \Omega_0^s = [0, 0.4] \times [0.45, 0.55]$. During the time interval $[0, 1]$, the structure is pulled down by a force applied at the middle point of the right edge. Therefore, when released, the solid body returns to its resting configuration by the action of internal forces. Four snapshots of the evolution are shown in Figure 2.

The energy density of the solid material is given by the potential strain energy function of an isotropic hyperelastic material; in particular, we have

$$W(F) = (\gamma / 2\eta) \exp (\eta [\text{tr}(F^\top F) - 2]),$$

where $\text{tr}(F^\top F)$ denotes the trace of $F^\top F$, while $\gamma = 1.333$ and $\eta = 9.242$. It can be proved that $W$ is a strictly locally convex strain energy function, as discussed in [34–36].

Also for this test we assume that fluid and solid materials share the same density, equal to 1, and the same viscosity, equal to 0.2. The velocity is imposed to be zero at the boundary of $\Omega$, while the following initial conditions are imposed

$$u(x, 0) = 0, \quad X(s, 0) = s.$$ 

Results for the mesh refinement test are reported in Table 3: we consider the evolution of the system during the time interval $[0, 2]$, with time step $\Delta t = 0.002$. The number of processors for the simulations is set to 64, while the number of dofs increases from 21,222 to 741,702. As for the linear case, $T_{\text{ass}}$ increases moderately and $T_{\text{coup}}$ follows a superlinear growth. Both preconditioners are robust with respect to mesh refinement: the number of Newton iterations is 2 for each test and the average number of GMRES iterations per nonlinear iteration is bounded by 15 for block-diag and by 10 for block-tri. This behavior of block-diag is in contrast with the results we obtained for the linear solid model: this is due to the finer time step chosen for this simulation.
In order to study the weak scalability, we choose $T = 0.1$ and $\Delta t = 0.002$. The results, reported in Table 4, are similar to the results obtained for the linear case. As before, $T_{coup}$ is not scalable due to the algorithm we implemented for the assembling of the coupling term. Even if it is not perfectly scalable, block-tri performs pretty well since the average number of linear iterations per nonlinear iteration increases only from 15 to 19. On the other hand, the good behavior of block-diag registered in Table 3 is not confirmed: the average number of linear iterations reaches 101, showing a lack of weak scalability, as already seen in Table 2.

Table 4. Weak scalability for the nonlinear solid model. The simulations are run on the Shaheen cluster. Same format as Table 3.

<table>
<thead>
<tr>
<th>Procs</th>
<th>DoFs</th>
<th>$T_{ass}$ (s)</th>
<th>$T_{coup}$ (s)</th>
<th>nit</th>
<th>its</th>
<th>$T_{sol}$ (s)</th>
<th>$T_{tot}$ (s)</th>
<th>nit</th>
<th>its</th>
<th>$T_{sol}$ (s)</th>
<th>$T_{tot}$ (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>83,398</td>
<td>$1.01 \times 10^{-1}$</td>
<td>2.21</td>
<td>3</td>
<td>23</td>
<td>6.76</td>
<td>448.65</td>
<td>3</td>
<td>15</td>
<td>5.50</td>
<td>386.13</td>
</tr>
<tr>
<td>8</td>
<td>156,910</td>
<td>$1.59 \times 10^{-1}$</td>
<td>3.77</td>
<td>3</td>
<td>38</td>
<td>15.49</td>
<td>963.03</td>
<td>3</td>
<td>16</td>
<td>8.87</td>
<td>627.84</td>
</tr>
<tr>
<td>16</td>
<td>330,630</td>
<td>$1.62 \times 10^{-1}$</td>
<td>8.92</td>
<td>3</td>
<td>49</td>
<td>36.58</td>
<td>2.28 $\times 10^{3}$</td>
<td>3</td>
<td>17</td>
<td>17.84</td>
<td>1.34 $\times 10^{3}$</td>
</tr>
<tr>
<td>32</td>
<td>741,702</td>
<td>$2.60 \times 10^{-1}$</td>
<td>25.18</td>
<td>3</td>
<td>67</td>
<td>123.99</td>
<td>7.46 $\times 10^{3}$</td>
<td>3</td>
<td>18</td>
<td>48.85</td>
<td>3.70 $\times 10^{3}$</td>
</tr>
<tr>
<td>64</td>
<td>1,316,614</td>
<td>$2.61 \times 10^{-1}$</td>
<td>69.12</td>
<td>3</td>
<td>101</td>
<td>328.18</td>
<td>1.99 $\times 10^{4}$</td>
<td>3</td>
<td>19</td>
<td>97.28</td>
<td>8.38 $\times 10^{3}$</td>
</tr>
</tbody>
</table>

Figure 2. Four snapshots of the evolution of the structure with nonlinear constitutive law.
7. Conclusions

We analyzed two preconditioners, block-diagonal and block-triangular, for saddle point systems originating from the finite element discretization of fluid–structure interaction problems with fictitious domain approach. We have focused only on the case where the fluid and solid domains have the same densities, which, based on previous studies, should be the most challenging to solve. In particular, the analysis has been conducted by studying the robustness with respect to mesh refinement and weak scalability, applying the parallel solver to both linear and nonlinear problems.

Only block-triangular appears to be robust in terms of mesh refinement for linear and nonlinear problems; on the other hand, block-diagonal works well when the time step is very small.

Moreover, by studying the weak scalability, we can notice two further limitations of the proposed method, which will be the subject of future studies. First, the time to assemble the coupling matrix is not scalable: it is based on two nested loops, related to solid and fluid elements, respectively; but only the external one is conducted in parallel over the processors. In order to improve this procedure, one may subdivide the involved meshes into clusters or, alternatively, one may keep track of the position of the solid body at the previous time instant. Second, since the action of the preconditioners consists of the exact inversion of two matrices, the time for solving the linear system slightly increases when the mesh is refined. Some preliminary results have shown that, in contrast with the fluid block $A_{11}$, the solid block $A_{22}$ does not behave well when an inexact inversion is applied: our future studies will be primarily focused on overcoming this problem.

As concerns the modeling approach, several extensions are possible. First of all, we have neglected the convective term in the Navier–Stokes equation; second, we have considered an isotropic nonlinear constitutive law for the structure instead of anisotropic variants and, finally, we have focused only on two dimensional problems.

Author Contributions: Conceptualization, D.B. and L.G.; Methodology, S.S.; Software, F.C. and S.S.; Validation, F.C. and S.S.; Resources, D.B.; Writing—Original draft, F.C. and S.S.; Writing—Review & editing, F.C., D.B., L.G. and S.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: The authors are member of GNCS/INdAM research group. D.B., F.C. and L.G. are partially supported by IMATI/CNR. Moreover, D.B. and L.G. are partially supported by PRIN/MIUR.

Conflicts of Interest: The authors declare no conflict of interest.

References


**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.