Journal of Scientific Computing (2020) 82:28
https://doi.org/10.1007/s10915-019-01113-y

A Parallel Newton Multigrid Framework for Monolithic Fluid-Structure Interactions

L. Failer[3] · T. Richter[1,2]

Received: 28 April 2019 / Revised: 27 September 2019 / Accepted: 7 November 2019 /
Published online: 21 January 2020

Abstract
We present a monolithic parallel Newton-multigrid solver for nonlinear nonstationary three dimensional fluid-structure interactions in arbitrary Lagrangian Eulerian (ALE) formulation. We start with a finite element discretization of the coupled problem, based on a remapping of the Navier–Stokes equations onto a fixed reference framework. The strongly coupled fluid-structure interaction problem is discretized with finite elements in space and finite differences in time. The resulting nonlinear and linear systems of equations are large and show a very high condition number. We present a novel Newton approach that is based on two essential ideas: First, a condensation of the solid deformation by exploiting the discretized velocity-deformation relation $d_t\mathbf{u} = \mathbf{v}$, second, the Jacobian of the fluid-structure interaction system is simplified by neglecting all derivatives with respect to the ALE deformation, an approximation that has shown to have little impact. The resulting system of equations decouples into a joint momentum equation and into two separate equations for the deformation fields in solid and fluid. Besides a reduction of the problem sizes, the approximation has a positive effect on the conditioning of the systems such that multigrid solvers with simple smoothers like a parallel Vanka-iteration can be applied. We demonstrate the efficiency of the resulting solver infrastructure on a well-studied 2d test-case and we also introduce a challenging 3d problem.

Keywords Fluid-structure interactions · Finite elements · Multigrid · Parallel computing

✉ T. Richter
thomas.richter@ovgu.de

L. Failer
lukas.failer@ma.tum.de

1 Otto-von-Guericke Universität Magdeburg, 39106 Magdeburg, Germany

2 Interdisciplinary Center for Scientific Computing, Heidelberg University, 69120 Heidelberg, Germany

3 Technische Universität München, 85748 Garching bei München, Germany

1 Introduction

Fluid-structure interactions appear in various problems ranging from classical applications in engineering like the design of ships or aircrafts, to the design of wind turbines. But they are also present in bio/medical systems describing the blood flow in the heart or in general problems involving the cardiovascular system. The typical challenge of fluid-structure interactions is two-fold. First, the special coupling character that stems from the coupling of a hyperbolic-type equation—the solid problem—with a parabolic-type equation—the Navier–Stokes equations. Second, the moving domain character brings along severe nonlinearities that have a non-local character, as geometrical changes close to the moving fluid-solid interface might have big impact on the overall solution.

Numerical approaches can usually be classified into *monolithic approaches*, where the coupled fluid-structure interaction system is taken as one entity and into *partitioned approaches*, where two separate problems—for fluid and solid—are formulated and where the coupling between them is incorporated in terms of an outer (iterative) algorithm. This second approach has the advantage that difficulties are isolated and that perfectly suited numerical schemes can be used for each of the subproblems. There are however application classes where partitioned approaches either fail or lack efficiency. The *added mass effect* [10] exactly describes this special stiffness connected to fluid-structure interactions. It is typical for problems with similar densities in the fluid and the solid—as it happens in the interaction of blood and tissue or in the interaction of water and the solid structure of a vessel. Here, monolithic approaches are considered to be favorable.

Monolithic approaches all give rise to strongly coupled, usually very large and nonlinear algebraic systems of equations. Although there has been substantial progress in designing efficient numerical schemes for tackling the nonlinear problems [16,21,23] (usually by Newton's method) and the resulting linear systems [2,11,13,19,28,32,36], the computational effort is still immense and numerically accurate results for 3d problems are still rare.

In this contribution we present an approximated Newton scheme for solving nonstationary fluids structure interactions in a strictly monolithic formulation. The idea is based on the observation that the Newton convergence rate does not significantly worsen, if we neglect the derivatives with respect to the ALE deformation, see [33, Section 5.2.3]. Although convergence rates slightly suffer, overall computational times can be reduced due to lesser effort for assembling the matrix. Here, we exploit this structure of the reduced Jacobian to achieve an exact splitting of the monolithic Jacobian into a coupled problem for the velocities of fluid and solid and into a second step, where separate update problems are solved for solid and fluid deformation. Apart from the approximation of the Jacobian, no further splitting error is introduced. The benefit of this approach is twofold: instead of one large system with 7 coupled unknowns (pressure, velocity field and deformation field in 3d) we solve one coupled system of four unknowns (pressure and velocities) and two separate problems involving the deformations of each domain. Second, separating a reduced velocity problem has a positive effect on the system matrices such that efficient preconditioners and smoothers can be applied that are suitable for easy parallelization. Finally, we use the newly developed solver to introduce and test a new three dimensional benchmark configuration that is based on the configurations described by Hron and Turek [23].

In the following section we give a brief presentation of the fluid-structure interaction problem in a variational Arbitrary Lagrangian Eulerian formulation. Section 3 shortly presents the discretization of the equations in space and time. As formulation and discretization are based on established techniques, these two sections are rather concise. The nonlinear and linear solution framework is described in Sect. 4, where we start by an approximation of the Jacobian that results in a natural partitioning of the linear systems, which in turn are approximated by parallel multigrid methods. Numerical test-cases demonstrate the efficiency and scalability in Sect. 5. Here, we also present a new and challenging 3d configuration for benchmarking fluid-structure interactions. We conclude in Sect. 6.

2 Governing Equations

Here, we present the monolithic formulation for fluid-structure interactions, coupling the incompressible Navier–Stokes equations and an hyperelastic solid, based on the St. Venant Kirchhoff material. For details we refer to [33].

On the d-dimensional domain, partitioned in reference configuration $\Omega = \mathcal{F} \cup \mathcal{I} \cup \mathcal{S}$, where $\mathcal{F}$ is the fluid domain, $\mathcal{S}$ the solid domain and $\mathcal{I}$ the fluid-structure interface, we denote by $\mathbf{v}$ the velocity field, split into fluid velocity $\mathbf{v}_f := \mathbf{v}|_{\mathcal{F}}$ and solid velocity $\mathbf{v}_s := \mathbf{v}|_{\mathcal{S}}$, and by $\mathbf{u}$ the deformation field, again with $\mathbf{u}_s := \mathbf{u}|_{\mathcal{S}}$ and $\mathbf{u}_f := \mathbf{u}|_{\mathcal{F}}$. The boundary of the fluid domain $\Gamma_f := \partial\mathcal{F} \setminus \mathcal{I}$ is split into inflow boundary Γ_f^{in} and wall boundary Γ_f^{wall}, where we usually assume Dirichlet conditions, $\Gamma_f^D := \Gamma_f^{in} \cup \Gamma_f^{wall}$, and a possible outflow boundary Γ_f^{out}, where we enforce the do-nothing outflow condition [22]. The solid boundary $\Gamma_s = \partial\mathcal{S} \setminus \mathcal{I}$ is split into Dirichlet part Γ_s^D and a Neumann part Γ_s^N.

We formulate the coupled fluid-structure interaction problem in a strictly monolithic scheme by mapping the moving fluid domain onto the reference state via the ALE map $T_f(t) : \mathcal{F} \to \mathcal{F}(t)$, constructed by a fluid domain deformation $T_f(t) = \text{id} + \mathbf{u}_f(t)$. In the solid domain, this map $T_s(t) = \text{id} + \mathbf{u}_s(t)$ denotes the Lagrange-Euler mapping and as the deformation field $\mathbf{u}$ will be defined globally on Ω we simply use the notation $T(t) = \text{id} + \mathbf{u}(t)$ with the deformation gradient $\mathbf{F} := \nabla T$ and its determinant $J := \det(\mathbf{F})$. We find the global (in fluid and solid domain) velocity and deformation fields $\mathbf{v}$ and $\mathbf{u}$ and the pressure p in the function spaces

$$\begin{aligned}\mathbf{v}(t) \in \mathbf{v}^D(t) + H_0^1(\Omega; \Gamma_f^D \cup \Gamma_s^D)^d, \quad \mathbf{u}(t) \in \mathbf{u}^D(t)\\ + H_0^1(\Omega; (\partial\mathcal{F} \setminus \mathcal{I}) \cup \Gamma_s^D)^d, \quad p \in L^2(\mathcal{F})\end{aligned}$$

as solution to

$$\begin{aligned}&\big(J(\partial_t \mathbf{v} + (\mathbf{F}^{-1}(\mathbf{v} - \partial_t \mathbf{u}) \cdot \nabla)\mathbf{v}), \phi\big)_{\mathcal{F}} + \big(J\boldsymbol{\sigma}_f \mathbf{F}^{-T}, \nabla\phi\big)_{\mathcal{F}}\\ &\quad + (\rho_s^0 \partial_t \mathbf{v}, \phi)_{\mathcal{S}} + (\mathbf{F}\boldsymbol{\Sigma}_s, \nabla\phi)_{\mathcal{S}} = (J\rho_f \mathbf{f}, \phi)_{\mathcal{F}} + (\rho_s^0 \mathbf{f}, \phi)_{\mathcal{S}}\\ &\big(J\mathbf{F}^{-1} : \nabla\mathbf{v}^T, \xi\big)_{\mathcal{F}} = 0\\ &(\partial_t \mathbf{u} - \mathbf{v}, \psi_s)_{\mathcal{S}} = 0\\ &(\nabla\mathbf{u}, \nabla\psi_f)_{\mathcal{F}} = 0,\end{aligned} \tag{1}$$

where the test functions are given in

$$\phi \in H_0^1(\Omega; \Gamma_f^D \cup \Gamma_s^D)^d, \quad \xi \in L^2(\mathcal{F}), \quad \psi_f \in H_0^1(\mathcal{F})^d, \quad \psi_s \in L^2(\mathcal{S})^d.$$

By ρ_s^0 we denote the solid's density, by $\mathbf{u}^D(t) \in H^1(\Omega)^d$ and $\mathbf{v}^D(t) \in H^1(\Omega)^d$ extensions of the Dirichlet data into the domain. The Cauchy stress tensor of the Navier–Stokes equations in ALE coordinates is given by

$$\boldsymbol{\sigma}_f(\mathbf{v}, p) = -p_f I + \rho_f \nu_f (\nabla\mathbf{v}\mathbf{F}^{-1} + \mathbf{F}^{-T}\nabla\mathbf{v}^T)$$

with the kinematic viscosity ν_f and the density ρ_f. In the solid we consider the St. Venant Kirchhoff material with the Piola Kirchhoff tensor

$$\boldsymbol{\Sigma}_s(\mathbf{u}) = 2\mu_s \mathbf{E}_s + \lambda_s \operatorname{tr}(\mathbf{E}_s) I, \quad \mathbf{E}_s := \frac{1}{2}(\mathbf{F}^T\mathbf{F} - I)$$

and with the shear modulus μ_s and the Lamé coefficient λ_s. In (1) we construct the ALE extension $\mathbf{u}_f = \mathbf{u}|_{\mathcal{F}}$ by a simple harmonic extension. A detailed discussion and further literature on the construction of this extension is found in [33,40].

For shorter notation, we denote by $U := (\mathbf{v}, \mathbf{u}, p_f)$ the solution and by $\Phi := (\xi, \phi, \psi_f, \psi_s)$ the test functions.

3 Discretization

We give a very brief presentation on the numerical approximation of System (1). In time, we use the theta time-stepping scheme, which includes the backward Euler method, the Crank-Nicolson scheme and variants like the fractional step theta method, see [37]. In space we use conforming finite elements.

3.1 Temporal Discretization

For discretization in time we split the temporal interval $I = [0, T]$ into discrete time steps $0 = t_1 < t_2 < \cdots < t_N = T$ with the step size $k := t_n - t_{n-1}$. For simplicity we assume that the subdivision is uniform. By $U_n \approx U(t_n)$ we denote the approximation at time t_n. We choose the theta time-stepping method for temporal discretization with $\theta \in [0, 1]$. To simplify the presentation we introduce

$$\begin{aligned}
A_F(U, \phi) &:= \big(J(\mathbf{F}^{-1}\mathbf{v} \cdot \nabla)\mathbf{v}, \phi\big)_{\mathcal{F}} \\
&\qquad + \big(\rho_f \nu_f J(\nabla\mathbf{v}\mathbf{F}^{-1} + \mathbf{F}^{-T}\nabla\mathbf{v}^T)\mathbf{F}^{-T}, \nabla\phi\big)_{\mathcal{F}} - \big(J\rho_f \mathbf{f}, \phi\big)_{\mathcal{F}} \\
A_S(U, \phi) &:= \big(\mathbf{F}\boldsymbol{\Sigma}_s, \nabla\phi\big)_{\mathcal{S}} - \big(\rho_s^0 \mathbf{f}, \phi\big)_{\mathcal{S}}, \quad A_{ALE}(U, \psi_f) := \big(\nabla\mathbf{u}, \nabla\psi_f\big)_{\mathcal{F}} \\
A_p(U, \phi) &:= \big(Jp\mathbf{F}^{-1}, \nabla\phi\big)_{\mathcal{F}}, \quad A_{div}(U, \xi) := \big(J\mathbf{F}^{-1} : \nabla\mathbf{v}^T, \xi\big)_{\mathcal{F}}.
\end{aligned} \tag{2}$$

Then, one time step $t_{n-1} \mapsto t_n$ of the theta scheme is given as

$$\begin{aligned}
&\underbrace{\big(\bar{J}_n(\mathbf{v}_n - \mathbf{v}_{n-1}), \phi\big)_{\mathcal{F}} - \big((\bar{J}_n\bar{\mathbf{F}}^{-1}(\mathbf{u}_n - \mathbf{u}_{n-1}) \cdot \nabla)\bar{\mathbf{v}}_n, \phi\big)_{\mathcal{F}}}_{F_{NS}(U_n,\phi)} \\
&\quad + \underbrace{kA_p(U_n, \phi) + k\theta A_F(U_n, \phi)}_{F_{NS}(U_n,\phi)} \\
&\quad + \big(\rho_s^0(\mathbf{v}_n - \mathbf{v}_{n-1}), \phi\big)_{\mathcal{S}} + k\theta A_S(U_n, \phi) = -k(1-\theta)A_F(U_{n-1}, \phi) \\
&\quad - k(1-\theta)A_S(U_{n-1}, \phi) \\
&kA_{div}(U_n, \xi) = 0 \\
&kA_{ALE}(U_n, \psi_f) = 0 \\
&\big(\mathbf{u}_n, \psi_s\big)_{\mathcal{S}} - k\theta\big(\mathbf{v}_n, \psi_s\big)_{\mathcal{S}} = \big(\mathbf{u}_{n-1}, \psi_s\big) + k(1-\theta)\big(\mathbf{v}_{n-1}, \psi_s\big)_{\mathcal{S}},
\end{aligned} \tag{3}$$

with $\bar{J}_n = 1/2(J_{n-1} + J_n)$ and $\bar{\mathbf{F}}_n = 1/2(\mathbf{F}_{n-1} + \mathbf{F}_n)$. Note that the ALE extension equation A_{ALE}, the divergence equation A_{div} and the pressure coupling A_p are completely implicit. A discussion of this scheme and results on its stability for fluid-structure interactions are found in [33,35]. We consider $\theta = 1/2 + \mathcal{O}(k)$ to get second order convergence and good stability properties.

The last equation in (3) gives a relation for the new deformation at time t_n

$$\mathbf{u}_n = \mathbf{u}_{n-1} + k\theta\mathbf{v}_n + k(1-\theta)\mathbf{v}_{n-1} \text{ in } \mathcal{S} \tag{4}$$

and we will use this representation to eliminate the unknown deformation $\mathbf{u}_n$ and base the solid stresses purely on the last time step and the unknown velocity $\mathbf{v}_n$, i.e. by expressing the deformation gradient as

$$\mathbf{F}_n = \mathbf{F}(\mathbf{u}_n) \mathrel{\widehat{=}} \mathbf{F}(\mathbf{u}_{n-1}, \mathbf{v}_{n-1}; \mathbf{v}_n) = I + \nabla\big(\mathbf{u}_{n-1} + k\theta\mathbf{v}_n + k(1-\theta)\mathbf{v}_{n-1}\big) \text{ in } \mathcal{S}. \tag{5}$$

Removing the solid deformation from the momentum equation will help to reduce the algebraic systems in Sect. 4. A similar technique within a Eulerian formulation and using a characteristics method is presented in [30,31].

3.2 Finite Elements

In space, we discretize with conforming finite elements by choosing discrete function spaces $U_h \in X_h$ and $\Phi_h \in Y_h$. We only consider finite element meshes that resolve the interface $\mathcal{I}$ in the reference configuration, such that the ALE formulation will always exactly track the moving interface. In our setting, implemented in the finite element library Gascoigne 3D [5] we use quadratic finite elements for all unknowns and add stabilization terms based on local projections [4,18,29,33] to satisfy the inf-sup condition. Where transport is dominant, additional stabilization terms of streamline upwind type [23,34,38] or of local projection type [14,33] are added. As the remainder of this manuscript only considers the fully discrete setting, we refrain from indicating spatial or temporal discrete variables with the usual subscripts.

For each time step $t_{n-1} \mapsto t_n$ we introduce the following short notation for the system of algebraic equations that is based on the splitting of the solution into unknowns acting in the fluid domain $(\mathbf{v}_f, \mathbf{u}_f)$, on the interface $(\mathbf{v}_i, \mathbf{u}_i)$ and those on the solid $(\mathbf{v}_s, \mathbf{u}_s)$. The pressure variable p acts in the fluid and on the interface.

$$\mathcal{A}(U) := \begin{pmatrix} \mathcal{D}(p, \mathbf{v}_f, \mathbf{u}_f, \mathbf{v}_i, \mathbf{u}_i, \mathbf{v}_s, \mathbf{u}_s) \\ \mathcal{M}^f(p, \mathbf{v}_f, \mathbf{u}_f, \mathbf{v}_i, \mathbf{u}_i) \\ \mathcal{M}^i(p, \mathbf{v}_f, \mathbf{u}_f, \mathbf{v}_i, \mathbf{u}_i, \mathbf{v}_s) \\ \mathcal{M}^s(p, \mathbf{v}_i, \mathbf{u}_i, \mathbf{v}_s) \\ \mathcal{E}(\mathbf{u}_f, \mathbf{u}_i) \\ \mathcal{U}^i(\mathbf{v}_i, \mathbf{u}_i, \mathbf{v}_s, \mathbf{u}_s) \\ \mathcal{U}^s(\mathbf{v}_i, \mathbf{u}_i, \mathbf{v}_s, \mathbf{u}_s) \end{pmatrix} = \begin{pmatrix} \mathcal{B}_1 \\ \mathcal{B}_2 \\ \mathcal{B}_3 \\ \mathcal{B}_4 \\ \mathcal{B}_5 \\ \mathcal{B}_6 \\ \mathcal{B}_7 \end{pmatrix} =: \mathcal{B} \tag{6}$$

$\mathcal{D}$ describes the divergence equation which acts in the fluid domain and on the interface, $\mathcal{M}$ the two momentum equations, acting in the fluid domain, on the interface and in the solid domain (which is indicated by a corresponding index), $\mathcal{E}$ describes the ALE extension in the fluid domain and $\mathcal{U}$ is the relation between solid velocity and solid deformation, acting on the interface degrees of freedom and in the solid. Note that $\mathcal{M}^i$ and $\mathcal{M}^s$, the term describing the momentum equations, do not directly depend on the solid deformation $\mathbf{u}_s$ as we express the deformation gradient by the velocity, see (5).

4 Solution of the Algebraic Systems

In fluid-structure interactions the solid and fluid problem are coupled via interface conditions. Forces in normal direction along the interface have to be equal (dynamic coupling condition) and the fluid domain has to follow the solid motion (kinematic and geometric coupling condition). If the solid motion is rather small and slow the energy exchange happens mainly via the dynamic coupling conditions. This allows the use of explicit time-stepping schemes for the mesh motion and ALE transformation for these examples. We want to follow a different approach and use a fully implicit time-stepping with an inexact Jacobian in the Newton algorithm. We neglect the derivatives with respect to the ALE deformation. Thereby, we have to solve in every Newton step a linear system of the same complexity as in the case of a partitioned time-stepping scheme.

In [33, chapter 5] we give a numerical study on different linearization techniques. It is found that the overall computational time can be reduced by neglecting the ALE derivatives in the Jacobian. Even for the fsi-3 benchmark problem of Hron and Turek [24] it is more efficient (in terms of overall computational time) to omit these derivatives at the cost of some additional Newton steps. Neglecting the ALE derivatives will be crucial for the reduction step described in the following section.

As we only change the Jacobian, we still apply a fully implicit time-stepping scheme and take advantage of its stability properties. Furthermore the transport due to the mesh motion is well approximated. For small time step sizes we will still observe super-linear convergence as with an exact Newton algorithm. In addition, the simplified structure of the matrix simplifies the development of preconditioners significantly as we will see later.

4.1 Relation to Approaches in Literature

Many (perhaps most) works on solvers for fluid-structure interactions are based on partitioned schemes, where highly tuned schemes can be applied to the two subproblems and acceleration methods are developed for the coupling. For an overview on some methods we refer to contributions in [8,9] and the literature cited therein. We focus on problems with a dominant added mass effect, where monolithic approaches are believed to be more efficient [21].

In the following we assume that the monolithic problem is approximated with a Newton scheme. It has been documented [2,32] that the Jacobian is very ill-conditioned with condition numbers exceeding those in fluid or solid mechanics by far. Furthermore, the systems are (in particular in 3d) so large that direct solvers are not applicable. In addition we found [32] that the condition numbers may be so large that direct solvers do not even converge well.[1] All successful solution strategies will therefore feature some kind of partitioning, usually be means of a decoupled preconditioner within a GMRES iteration. In [28] an overview on state of the art precondition techniques for iterative fluid-structure interaction solvers is given.

Multigrid solvers have first been used to accelerate the solution of the subproblems within an iterative scheme. A fully monolithic geometric multigrid approach was presented in [23] for 2d fsi problems. Here, the multigrid smoother was based on a Vanka iteration. In [7] the authors analyzed a highly simplified model problem and showed that a partitioned iteration as smoother should result in ideal multigrid performance with improved convergence rates on deeper mesh hierarchies. An algebraic multigrid method with applications in 2d and 3d

[1] These results where found in [32] for the direct solver UMFPACK [12]. As similar study in [2] could validate our estimates for the condition numbers but did not experience a deterioration of convergence rates in the solver MUMPS [1].

[19] was based on a Gauss-Seidel splitting in the smoother. In [32] we presented a fully geometric monolithic multigrid method with a smoother that is based on a partitioning into fluid and solid problem and a block decomposition of each equation. This approach has been extended to incompressible materials and also to direct-to-steady-state solutions [2].

Some of these contributions employ parallelism. Recently, a block-preconditioned parallel GMRES iteration was presented [25] and showed good performance on various 2d and 3d test cases. A Gauss-Seidel decoupling with highly efficient and massively parallel preconditioners based on the SIMPLE scheme for the fluid and multigrid for a linear elasticity problem is presented in [13].

4.2 Linearization and Splitting

Each time step of the fully discrete problem is solved by Newton's method. Evaluating the Jacobian is cumbersome due to the moving domain character of the fluid problem. First presentations of the derivatives of the fsi problem with respect to the mesh motion based on the concept of shape derivatives have been given by Fernandez and Moubachir [17]. Details in the spirit of our formulation in ALE coordinates are given in [33, Section 5.2.2]. Based on the notation (6) let $U^{(0)}$ be an initial guess (usually taken from the last time step) we iterate for $l = 0, 1, 2, \ldots$

$$\mathcal{A}'(U^{(l)})W^{(l)} = \mathcal{B} - \mathcal{A}(U^{(l)}), \quad U^{(l+1)} := U^{(l)} + \omega^{(l)} W^{(l)}, \tag{7}$$

with a line search parameter $\omega^{(l)} > 0$ and the Jacobian $\mathcal{A}'(U)$ evaluated at U. Each linear problem can be written as

$$\left(\begin{array}{ccc|cc|cc}
0 & \mathcal{D}_{\mathbf{v}_f} & \mathcal{D}_{\mathbf{u}_f} & \mathcal{D}_{\mathbf{v}_i} & \mathcal{D}_{\mathbf{u}_i} & 0 & 0 \\
\mathcal{M}^f_p & \mathcal{M}^f_{\mathbf{v}_f} & \mathcal{M}^f_{\mathbf{u}_f} & \mathcal{M}^f_{\mathbf{v}_i} v\mathcal{M}^f_{\mathbf{u}_i} & 0 & 0 & \\
\mathcal{M}^i_p & \mathcal{M}^i_{\mathbf{v}_f} & \mathcal{M}^i_{\mathbf{u}_f} & \mathcal{M}^i_{\mathbf{v}_i} & \boldsymbol{\mathcal{M}}^{\boldsymbol{i}}_{\mathbf{u}_i} & \mathcal{M}^i_{\mathbf{v}_s} & \boldsymbol{\mathcal{M}}^{\boldsymbol{i}}_{\mathbf{u}_s} \\
\mathcal{M}^s_p & 0 & 0 & \mathcal{M}^s_{\mathbf{v}_i} & \boldsymbol{\mathcal{M}}^{\boldsymbol{s}}_{\mathbf{u}_i} & \mathcal{M}^s_{\mathbf{v}_s} & \boldsymbol{\mathcal{M}}^{\boldsymbol{s}}_{\mathbf{u}_s} \\
\hline
0 & 0 & \mathcal{E}^f_{\mathbf{u}_f} & 0 & \mathcal{E}^f_{\mathbf{u}_i} & 0 & 0 \\
\hline
0 & 0 & 0 & \mathcal{U}^i_{\mathbf{v}_i} & \mathcal{U}^i_{\mathbf{u}_i} & \mathcal{U}^i_{\mathbf{v}_s} & \mathcal{U}^i_{\mathbf{u}_s} \\
0 & 0 & 0 & \mathcal{U}^s_{\mathbf{v}_i} & \mathcal{U}^s_{\mathbf{u}_i} & \mathcal{U}^s_{\mathbf{v}_s} & \mathcal{U}^s_{\mathbf{u}_s}
\end{array}\right)
\begin{pmatrix} \delta\mathbf{p} \\ \mathbf{v}_f \\ \mathbf{u}_f \\ \mathbf{v}_i \\ \mathbf{u}_i \\ \mathbf{v}_s \\ \mathbf{u}_s \end{pmatrix}
=
\begin{pmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \\ \mathbf{b}_4 \\ \mathbf{b}_5 \\ \mathbf{b}_6 \\ \mathbf{b}_7 \end{pmatrix}, \tag{8}$$

where the right hand side vector $\mathbf{B} = \mathcal{B} - \mathcal{A}(U^{(l)})$ is the Newton residual. The Jacobian shows the coupling structure of the nonlinear problem (6). The indices $\mathcal{M}^f$, $\mathcal{M}^i$, $\mathcal{M}^s$ correspond to the degrees of freedom, whether it belongs to a Lagrange node in the fluid, on the interface or in the solid. The subnodes correspond to the dependency on the unknown solution component, pressure, velocity and deformation, each in the different domains.

Three of the entries in bold letters, $\boldsymbol{\mathcal{M}}^{\boldsymbol{s}}_{\mathbf{u}_i}$, $\boldsymbol{\mathcal{M}}^{\boldsymbol{s}}_{\mathbf{u}_s}$ and $\boldsymbol{\mathcal{M}}^{\boldsymbol{i}}_{\mathbf{u}_s}$ are zero. As the deformation gradient is expressed in terms of the velocity, see (5), the dependency of the solid equation on the solid's deformation does not appear. The entry $\boldsymbol{\mathcal{M}}^{\boldsymbol{i}}_{\mathbf{u}_i}$ belongs to test functions ϕ that live on the interface. Thus, it contributes to both the solid equation and the fluid equation, e.g.

$$\langle \boldsymbol{\mathcal{M}}^{\boldsymbol{i}}_{\mathbf{u}_i}(\psi), \phi \rangle = \left(\frac{d}{d\mathbf{u}_i} (J\boldsymbol{\sigma}_f \mathbf{F}^{-T})(\psi), \nabla\phi \right)_{\mathcal{F}} + \left(\underbrace{\frac{d}{d\mathbf{u}_i} (\mathbf{F}\boldsymbol{\Sigma}_s)(\psi),}_{=0} \nabla\phi \right)_{\mathcal{S}},$$

where only the solid part will vanish, compare (2). The remaining part belongs to the ALE map and these terms require the highest computational effort.

Corresponding terms are found in $\mathcal{M}^f_{\mathbf{u}_f}$, $\mathcal{M}^i_{\mathbf{u}_f}$, $\mathcal{M}^f_{\mathbf{u}_i}$ and also in $\mathcal{D}_{\mathbf{u}_f}$ and $\mathcal{D}_{\mathbf{u}_i}$, which are all highlighted and marked in gray. We will set these matrix entries to zero and note once more that this is the only approximation within our Newton-multigrid scheme. Sorting the unknowns as $(p, \mathbf{v}_f, \mathbf{v}_i, \mathbf{v}_s, \mathbf{u}_f, \mathbf{u}_i, \mathbf{u}_s)$, the reduced system takes the following form and reveals a block structure

$$\left(\begin{array}{cccc|c|cc}
0 & \mathcal{D}_{\mathbf{v}_f} & \mathcal{D}_{\mathbf{v}_i} & 0 & \mathbf{0} & \mathbf{0} & 0 \\
\mathcal{M}^f_p & \mathcal{M}^f_{\mathbf{v}_f} & \mathcal{M}^f_{\mathbf{v}_i} & 0 & \mathbf{0} & \mathbf{0} & 0 \\
\mathcal{M}^i_p & \mathcal{M}^i_{\mathbf{v}_f} & \mathcal{M}^i_{\mathbf{v}_i} & \mathcal{M}^i_{\mathbf{v}_s} & \mathbf{0} & \mathbf{0} & 0 \\
\mathcal{M}^s_p & 0 & \mathcal{M}^s_{\mathbf{v}_i} & \mathcal{M}^s_{\mathbf{v}_s} & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & \mathcal{E}^f_{\mathbf{u}_f} & \mathcal{E}^f_{\mathbf{u}_i} & 0 \\
\hline
0 & 0 & \mathcal{U}^i_{\mathbf{v}_i} & \mathcal{U}^i_{\mathbf{v}_s} & 0 & \mathcal{U}^i_{\mathbf{u}_i} & \mathcal{U}^i_{\mathbf{u}_i} \\
0 & 0 & \mathcal{U}^s_{\mathbf{v}_i} & \mathcal{U}^s_{\mathbf{v}_s} & 0 & \mathcal{U}^s_{\mathbf{u}_i} & \mathcal{U}^s_{\mathbf{u}_i}
\end{array}\right)
\begin{pmatrix} \delta\mathbf{p} \\ \mathbf{v}_f \\ \mathbf{v}_i \\ \mathbf{v}_s \\ \mathbf{u}_f \\ \mathbf{u}_i \\ \mathbf{u}_s \end{pmatrix}
=
\begin{pmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \\ \mathbf{b}_4 \\ \mathbf{b}_5 \\ \mathbf{b}_6 \\ \mathbf{b}_7 \end{pmatrix}. \tag{9}$$

The dropped ALE derivatives (bold face zeros) are the most costly parts in matrix assembly. While skipping these terms does worsen Newton convergence rates, the overall computational time can still benefit. This has been shown in [33, Section 5.2.3] considering a benchmark problem with large deformation. This reduced linear system decomposes into three sub-steps. First, the coupled momentum equation, living in fluid and solid domain and acting on pressure and velocity

$$\begin{pmatrix}
0 & \mathcal{D}_{\mathbf{v}_f} & \mathcal{D}_{\mathbf{v}_i} & 0 \\
\mathcal{M}^f_p & \mathcal{M}^f_{\mathbf{v}_f} & \mathcal{M}^f_{\mathbf{v}_i} & 0 \\
\mathcal{M}^i_p & \mathcal{M}^i_{\mathbf{v}_f} & \mathcal{M}^i_{\mathbf{v}_i} & \mathcal{M}^i_{\mathbf{v}_s} \\
\mathcal{M}^s_p & 0 & \mathcal{M}^s_{\mathbf{v}_i} & \mathcal{M}^s_{\mathbf{v}_s}
\end{pmatrix}
\begin{pmatrix} \delta\mathbf{p} \\ \mathbf{v}_f \\ \mathbf{v}_i \\ \mathbf{v}_s \end{pmatrix}
=
\begin{pmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \\ \mathbf{b}_4 \end{pmatrix}. \tag{10}$$

Second, the update equation for the deformation on the interface and within the solid domain

$$\begin{pmatrix} \mathcal{U}^i_{\mathbf{u}_i} & \mathcal{U}^i_{\mathbf{u}_i} \\ \mathcal{U}^s_{\mathbf{u}_i} & \mathcal{U}^s_{\mathbf{u}_i} \end{pmatrix}
\begin{pmatrix} \mathbf{u}_i \\ \mathbf{u}_s \end{pmatrix}
=
\begin{pmatrix} \mathbf{b}_6 \\ \mathbf{b}_7 \end{pmatrix}
-
\begin{pmatrix} \mathcal{U}^i_{\mathbf{v}_i} & \mathcal{U}^i_{\mathbf{v}_s} \\ \mathcal{U}^s_{\mathbf{v}_i} & \mathcal{U}^s_{\mathbf{v}_s} \end{pmatrix}
\begin{pmatrix} \delta\mathbf{v}_i \\ \delta\mathbf{v}_s \end{pmatrix}, \tag{11}$$

which, as a finite element discretization of the zero-order equation $\mathbf{u}_n = \mathbf{u}_{n+1} + k(1-\theta)\mathbf{v}_{n-1} + k\theta\mathbf{v}_n$, only involves the mass matrix on both sides, such that this update can be performed by one vector-addition. Finally it remains to solve for the ALE extension equation

$$\mathcal{E}^f_{\mathbf{u}_f}\delta\mathbf{u}_f = \mathbf{b}_5 - \mathcal{E}^f_{\mathbf{u}_i}\delta\mathbf{u}_i \tag{12}$$

a simple elliptic equation, usually either the vector Laplacian or a linear elasticity problem, see [33, section 5.2.5]. The main effort lies in the momentum equations (10), which is still a coupled fluid-solid problem with saddle-point character due to the incompressibility.

Details on the derivatives appearing in (10) are given in [17,41,42] and in [33, Section 5.2.2] in the framework of this work. Note however that most of these terms, including all derivatives of the Navier–Stokes equations in direction of the fluid domain deformation $\mathbf{u}_f$ are dropped.

4.3 Solution of the Linear Problems

The efficient solution of the linear systems arising in Newton approximations to nonlinear fluid-structure interaction problems is still an open problem. Lately some progress has been done in the direction of multigrid preconditioners for the monolithic problem [2,19,32,33]. In all these contributions it has proven to be essential to apply a partitioning into fluid-problem and solid-problem within the smoother.

We shortly present the linear algebra framework used in the software library *Gascoigne 3D* [5]. We are using equal-order finite element for all unknowns, namely pressure, velocity and deformation such that we can locally block all degrees of freedom in each Lagrange point. The solution U_h is written as

$$U_h(x) = \sum_{i=1}^{N_h} \mathbf{U}_i \phi_h^{(i)}(x), \quad \mathbf{U}_i = \begin{pmatrix} \mathbf{p}_i \\ \mathbf{v}_i \\ \mathbf{u}_i \end{pmatrix} \in \mathbb{R}^{2d+1}.$$

By N_h we denote the number of degrees of freedom (for every unknown), by d the dimension. Likewise, the system matrix $\mathbf{A}$ is a matrix with block structure, i.e. $\mathbf{A} \in \mathbb{R}^{N_h(2d+1)\times N_h(2d+1)}$ with $\mathbf{A}_{ij} \in \mathbb{R}^{(2d+1)\times(2d+1)}$. Considering the approximation scheme described in (10), (11) and (12), the coupled momentum equation has $n_c^{\mathcal{M}} = d+1$ components and the extension problem consists of $n_c^{\mathcal{E}} = d$ components. In general, the complete linear algebra module is acting on generic matrices and vectors with a block structure and local blocks of size $n_c \times n_c$ and n_c, respectively. The linear solver is designed by the following approach:

(I) As outer iteration we employ a GMRES method. Usually very few (< 10) iterations are required such that restarting strategies are not used.
(II) The GMRES solver is preconditioned by a geometric multigrid method in V-cycle [3,26]. The finite element mesh on each multigrid level resolves the fluid-solid interface.
(III) As smoother in the multigrid solver we use a Vanka type iteration which we will outline in some detail.

The smoother for the velocity problem and the smoother for the ALE extension problem is of Vanka type. Let $\mathcal{N}_h$ be the set of degrees of freedom of the discretization on mesh level Ω_h. By $\mathcal{P} = \{P_1, \dots, P_{n_\mathcal{P}}\}$ with $P_i \subset \mathcal{N}_h$ we denote a partitioning of unknowns into local patches. In the most simple case, P_i includes all degrees of freedom in one element of the mesh. Larger patches, e.g. by combining 4 adjacent elements in 2d or 8 elements in 3d are possible. By $n_\mathcal{P}$ we denote the number of patches and by n_p the size of each patch, which is the number of degrees of freedom in the patch. For simplicity, we assume that all patches in $\mathcal{P}$ have the same size. By $\mathcal{R}_i : \mathbb{R}^N \to \mathbb{R}^{n_p}$ we denote the restriction of a global vector to the degrees of freedom in one patch, by $\mathcal{R}_i^T$ the prolongation. Given a block vector $\mathbf{x} \in \mathbb{R}^{N_h n_c}$ and a block matrix $\mathbf{A} \in \mathbb{R}^{N_h n_c \times N_h n_h}$ we denote by

$$\mathbf{x}_i := \mathcal{R}_i \mathbf{x}, \quad \mathbf{A}_i := \mathcal{R}_i \mathbf{A} \mathcal{R}_i^T$$

the restrictions to the degrees of freedom of one patch P_i. We iterate

$$\begin{aligned} d_h^{(l)} &= b_h - A_h x_h^{(l)}, \\ x_h^{(l+1)} &= x_h^{(l)} + \omega_V \sum_{P \subset \Omega_h} \mathcal{R}_i^T \mathbf{A}_i^{-1} \mathcal{R}_i d_h^{(l)}, \end{aligned} \tag{13}$$

with a damping parameter $\omega_V \approx 0.8$. This smoother can also be considered as a domain decomposition iteration with minimal overlap. Numerical tests have shown that this simple Jacobi coupling is more efficient than a corresponding Gauss-Seidel iteration.

The local matrices $\mathbf{A}_i$ are inverted exactly using the library *Eigen* [20]. They are of substantial size, for $d = 3$, the local matrices corresponding to the momentum equations (10) have dimension 108×108, if small patches are used, and 500×500 if the smoother is based on the larger patches.

Table 1 Parameters of the benchmark problems in 2d (left) and 3d (right): average inflow velocity, fluid- and solid-parameters, reference length L and resulting Reynolds number

	2d configuration	3d configuration
$\bar{\mathbf{v}}$	$2\,\mathrm{m\ s^{-1}}$	$1.75\,\mathrm{m\ s^{-1}}$
ρ_s, ρ_f	$1000\,\mathrm{kg\ m^{-3}}$	$1000\,\mathrm{kg\ m^{-3}}$
μ_s	$2\times 10^6\,\mathrm{kg\ m^{-1}\ s^{-2}}$	$2\times 10^6\,\mathrm{kg\ m^{-1}\ s^{-2}}$
λ_s	$8\times 10^6\,\mathrm{kg\ m^{-1}\ s^{-2}}$	$8\times 10^6\,\mathrm{kg\,m^{-1}\,s^{-2}}$
ν_f	$0.001\,\mathrm{m^2\,s^{-1}}$	$0.001\,\mathrm{m^2\ s^{-1}}$
L	$0.1\,\mathrm{m}$	$0.1\,\mathrm{m}$
Re	200	175

4.4 Parallelization

Basic features of *Gascoigne 3D* [5] are parallelized based on *OpenMP* [27]. For parallelization of the assembly of residuals and the matrix as well as application of the Vanka smoother (13) we use a coloring of the patches $\mathcal{P}$ such that no collisions appear. The usual memory bottleneck of finite element simulations will limit the parallel efficiency of matrix vector product and Vanka smoother. We will present some data on the parallel performance in Sect. 5.5.4.

5 Numerical Results

5.1 Problem Configuration

Two different test-cases are considered to study the performance of the discretization and the solvers that have been presented in Sects. 3 and 4 . First, we perform a numerical study based on the 2d fsi-3 benchmark problem that has been defined by Hron and Turek [24]. Second, we present a new 3d benchmark configuration that is based on the Hron & Turek problem.

5.1.1 2d Configuration

As two dimensional configuration we solve the nonstationary 2d fsi-3 benchmark problem that has been introduced by Hron and Turek [24] and since then has been revisited in many contributions [21,34] or [33, chapter 7]. We present results for this well established benchmark problem in order to validate the discretization and to compare the performance of the solver with results published in literature. The material parameters are given in Table 1 and the parameters yield a Reynolds number Re = 200 showing a periodic flow pattern.

5.1.2 3d Configuration

Figure 1 shows the geometric configuration of the 3d benchmark problem. The computational domain with dimension 2.8 m × 0.41 m × 0.41 m is hexahedral with a cylinder cut out of it

$$\Omega = \{(x,y,z)\in\mathbb{R}^3 \,|\, 0 < x < 2.8,\ 0 < y < 0.41,\ 0 < z < 0.41\} \setminus \bar{\Omega}_{\text{cyl}},$$
$$\Omega_{\text{cyl}} = \{(x,y,z)\in\mathbb{R}^3 \,|\, (x-0.5)^2 + (y-0.2)^2 < 0.05^2,\ 0 < z < 0.41\}.$$

The midpoint of the cylinder is slightly non-symmetric to allow for a stable oscillatory flow at low Reynolds numbers. Attached to the cylinder is an elastic beam with approximate dimension 0.35 × 0.02 × 0.2 given in initial state at time $t = 0$ as

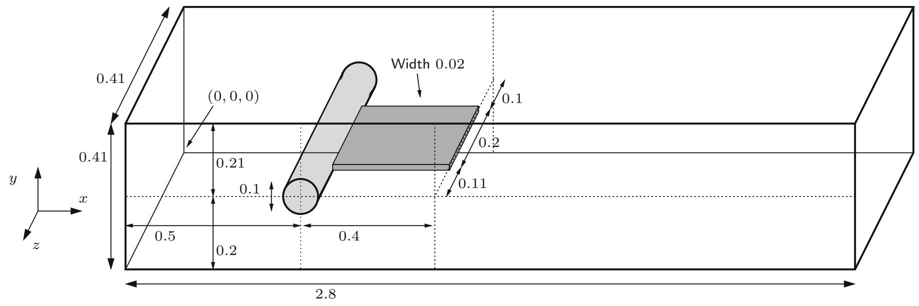


Fig. 1 Configuration of the 3d benchmark problem

$$\mathcal{S} = \{(x, y, z) \in \mathbb{R}^3 \,|\, 0.5 < x < 0.9,\ 0.19 < y < 0.21,\ 0.1 < z < 0.3\} \setminus \bar{\Omega}_{\text{cyl}}$$

The reference fluid domain at time $t = 0$ is given by

$$\mathcal{F} = \Omega \setminus \bar{\mathcal{S}}.$$

Boundary conditions The boundary of the domain is split into the *inflow boundary* Γ_f^{in} at $x = 0$, the *outflow boundary* Γ_f^{out} at $x = 2.8$, the *wall boundaries* at $z = 0$ and $z = 0.41$ as well as $y = 0$ and $y = 0.41$ as well as the *cylinder boundary* Γ_f^{cyl} at $(x-0.5)^2+(y-0.2)^2 = 0.05^2$. On the inflow boundary Γ_f^{in} we prescribe a bi-parabolic profile

$$\mathbf{v}^{in} = \bar{\mathbf{v}}\frac{36y(0.41-y)z(0.41-z)}{0.41^4},$$

that satisfies $|\Gamma_f^{in}|^{-1}\int_{\Gamma_f^{in}} \mathbf{v}^{in}\,\mathrm{d}s = \bar{\mathbf{v}}$, where $\bar{\mathbf{v}}$ is the average velocity. For regularization we suggest to introduce a transient start-up of the inflow

$$\mathbf{v}^{in}(t) = \mathbf{v}^{in}\begin{cases}\left(\frac{1}{2} - \frac{1}{2}\cos(\pi t)\right) & 0 \le t < 1\\ 1 & t \ge 1.\end{cases}$$

On the remaining boundaries $\Gamma_f^{wall} \cup \Gamma_f^{cyl}$ the no-slip condition $\mathbf{v} = 0$ is prescribed. For the deformation $\mathbf{u}$ (both the solid deformation and the ALE extension), a no-slip condition $\mathbf{u} = 0$ is prescribed on all boundaries. On the outer boundaries Γ_{wall}, Γ_{in} and Γ_{out} this condition can be relaxed to allow for larger mesh deformations, see [33, Section 5.3.5].

Material Parameters Similar material parameters as for the 2d set are taken and the values are given in Table 1. These parameters give a Reynolds number of $Re = 175$ and a periodic flow pattern arises.

5.2 Quantities of Interest

For the 2d configuration, we present the displacement at the tip of the flag at the point $A = (0.6, 0.2)$ in x- and y-direction. In the case of the 3d configuration we take the point $B = (0.9, 0.2, 0.3)$ on the back face of the beam and present the displacement in x-, y- and z-direction. These values are evaluated at every time-point. In addition we compute the drag and lift values around the beam and cylinder. To compute the lift $\mathbf{f} \cdot \mathbf{e}_1$ and drag forces $\mathbf{f} \cdot \mathbf{e}_2$ with $\mathbf{e}_i = (\delta_{ij})_{j=1}^3 \in \mathbb{R}^3$ and

$$\mathbf{f} = \int_{\Gamma_f^{cyl} \cup \mathcal{I}} J \boldsymbol{\sigma}_f \mathbf{F}^{-T} n \, \mathrm{d}\Gamma, \tag{14}$$

we evaluate the residual representation

$$\begin{aligned} \mathbf{f}_n = & \big(\bar{J}_n(\mathbf{v}_n - \mathbf{v}_{n-1}), \mathbf{1}_{\Gamma_{cyl}}\big)_{\mathcal{F}} - \big(\bar{J}_n \bar{\mathbf{F}}^{-1}(\mathbf{u}_n - \mathbf{u}_{n-1}) \cdot \nabla \bar{\mathbf{v}}_n, \mathbf{1}_{\Gamma_{cyl}}\big)_{\mathcal{F}} \\ & + A_p(U_n, \mathbf{1}_{\Gamma_{cyl}}) + k\theta A_F(U_n, \mathbf{1}_{\Gamma_{cyl}}) + k(1-\theta) A_F(U_{n-1}, \mathbf{1}_{\Gamma_{cyl}}) \\ & + k(1-\theta) A_S(U_{n-1}, \mathbf{1}_{\Gamma_s})) + k\theta A_S(U_n, \mathbf{1}_{\Gamma_s}) \end{aligned}$$

where $\mathbf{1}_{\Gamma_{cyl}}$ is a finite element testfunction which is one along the cylinder Γ_{cyl} and zero elsewhere. Thereby we can compute the mean drag and lift value on every time interval $I_n = [t_n, t_{n+1}]$ with very high precision. Details on the evaluation of such surface integrals for flow problems are given in [6] and in [33, Section 6.6.2] in the case of fluid-structure interactions.

5.3 Approximative Newton Scheme (2d benchmark)

We start by investigating the effect of the approximation of the Jacobian in our reduced Newton scheme. The 2d fsi-3 benchmark problem by Hron and Turek is evaluated on the time interval $I = [5, 5.5]$, where the dynamic is fully evolved and large deformations appear. A similar study with the same parameters and discretization has been performed in [33, chapter 5.2.3], however, based on the full monolithic Jacobian and using a direct solver for the linear problems. The comparison with the results in [33] enables to evaluate the effects of the presented inexact Jacobian on the Newton scheme. On the time interval $I = [5, 5.5]$ the oscillations are fully developed such that significant oscillations appear and the geometric nonlinearities, that come from the ALE mapping, have to be taken into account.

We only update the Jacobian of (10), the momentum equation, if the nonlinear convergence rate, that is measured as

$$\rho_l = \frac{\|\mathcal{B} - \mathcal{A}(U^{(l)})\|_\infty}{\|\mathcal{B} - \mathcal{A}(U^{(l-1)})\|_\infty}, \tag{15}$$

is above a given threshold γ_{nt}. The Jacobian of (12), the mesh motion problem, is only assembled once in the first time step, as we use a linear elasticity law. Like in [33], we investigate the behavior for the parameters

$$\gamma_{nt} \in \{0, 0.2, 0.5\},$$

where $\gamma_{nt} = 0$ corresponds to the assembly of the approximated Jacobian in every Newton step. We solve the linear systems in every Newton step using a direct solver without any parallelization. The computations are performed on an Intel(R) Core(TM) i7-7700 CPU @ 3.60GHz. For the time-stepping we use the suggested implicitly shifted Crank-Nicolson scheme with $\theta = 0.5 + 2k$ and the time step size $k = 0.005$s. For spatial discretization we choose equal-order biquadratic elements on a mesh with 80 960 dofs (mesh level 4). The Newton algorithm is stopped if the relative error reduces by eight orders of magnitude (relative tol $= 10^{-8}$).

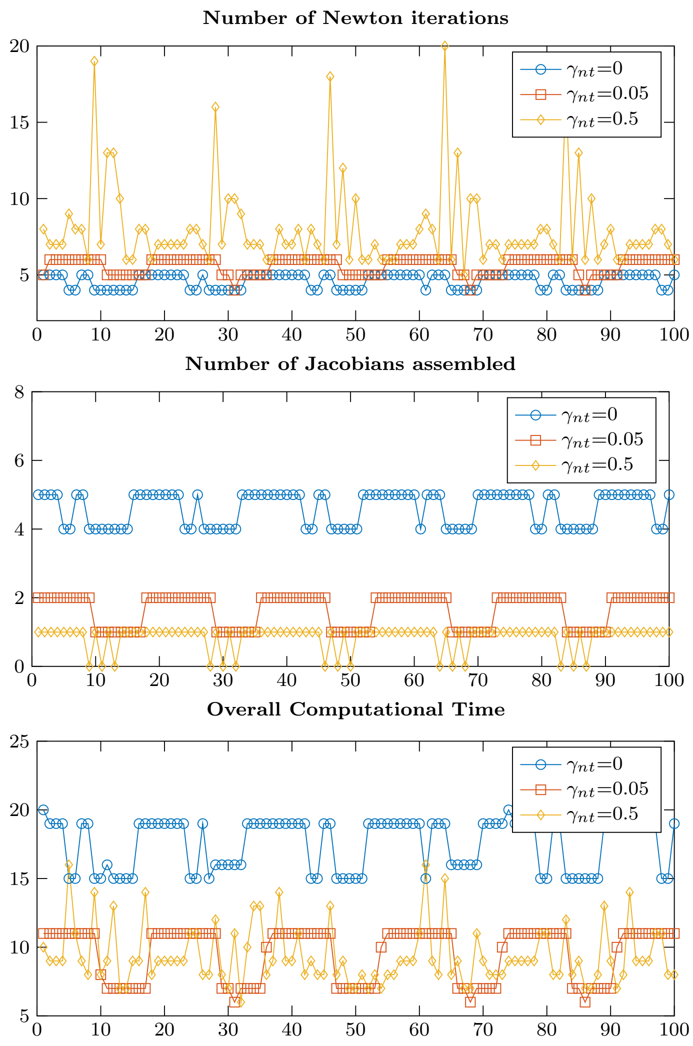


Fig. 2 Study on the effect of the non-exact Newton scheme for the 2d benchmark problem. The Jacobian is only reassembled, if the Newton rate is above γ_{nt}. Top: number of Newton iterations per time step. Middle: Number of Jacobians assembled in each time step. Bottom: overall computational time in each time step

In Fig. 2 we show the results for each time step in the interval $I = [5, 5.5]$. The top row shows that the least number of Newton steps are required, if $\gamma_{nt} = 0$ is used. This is expected as $\gamma_{nt} = 0$ corresponds to the full Newton scheme that allows for quadratic convergence. While the effect is small for $\gamma_{nt} = 0.05$, the resulting Newton iteration count strongly increases for $\gamma_{nt} = 0.5$, where up to 20 steps are required, compared to a limit of 5 steps for $\gamma_{nt} = 0$ and 6 steps for $\gamma_{nt} = 0.05$. In the middle plot of Fig. 2 we give the number of Jacobians that have to be assembled. For $\gamma_{nt} = 0$ these numbers obviously correspond to the number of Newton steps, as the Jacobian is newly assembled in each step. For $\gamma_{nt} = 0.05$ and $\gamma_{nt} = 0.5$ the required number of assemblies is strongly limited. Finally, the lower plot shows the resulting computational time. Although $\gamma_{nt} = 0$ yields the best convergence rates,

Table 2 Accumulated number of Newton steps, assemblies of the Jacobian in Equation (10) and the total time (in seconds) for all 100 time steps for different values of γ_{nt}

Matrix ass. tolerance	$\gamma_{nt} = 0.0$	$\gamma_{nt} = 0.05$	$\gamma_{nt} = 0.2$	$\gamma_{nt} = 0.5$
Total Newton steps	460	559	741	800
Jacobians assembled	460	164	110	85
Total Time (seconds)	1753	950	899	936

Table 3 Degrees of freedom for 2d and 3d configuration on every refinement level

Mesh level	1	2	3	4	5	6
dofs 2d	1440	5360	20,640	80,960	320,640	1,276,155
dofs 3d	63,826	463,988	3,531,304	–	–	–

it requires the highest computational time. The choice $\gamma_{nt} = 0.05$ reduces the computational time by a factor of 2 while still giving very robust convergence. These results are in agreement with the study in [33]. With respect to computational time, Table 2 shows that $\gamma_{nt} = 0.2$ is most efficient, as the reduced time to assemble the Jacobian and the increased time, due to more Newton steps balances best. These results also reveal the large computational time that is required for assembling the Jacobian and preparing the multigrid smoother.

We can see in Table 2, where we collect the accumulated numbers for the complete interval $I = [5, 5.5]$ that we need 460 Newton steps, if we assemble the Jacobian in (10) in every Newton step. As we neglect the sensitivity information with respect to the mesh motion, we still have an inexact Newton scheme. Nevertheless, we need less Newton steps compared to the use of an exact Jacobian as in [33], where 532 Newton steps were required for the same setting. This is in line with the numerical tests on the inexact Jacobian for the 2d fsi-3 benchmark results in [33], where in first numerical studies no disadvantages due to the inexact Jacobian could be observed. Nevertheless, the better convergence rate is surprising. The direct solver UMFPACK [12] has difficulties to solve the exact Jacobian accurately enough as reported in [32,33], which could be the reason for the higher number of Newton steps. The condition numbers for the matrices of the subproblems (10), (11) and (12) are much better then for the exact Jacobian as already analyzed in [33]. To conclude the modification of the Newton scheme only has minor influences on the Newton convergence rate, at least in the here presented numerical study.

5.4 Reference Values

All presented results in the following sections are initiated by using a time-stepping scheme with $k = 0.004$s to compute a solution on the time interval $I = [0, 8]$ on all mesh levels indicated in Table 3. The corresponding solutions at time $t = 8$s act as initial values for further computations on the interval $I = [8, 10]$ based on the time step sizes $k = 0.004$s, $k = 0.002$s and $k = 0.001$s. To avoid inaccuracies in the reference values due a rapid change of the numerical discretization parameters, we only present results on the interval $I = [9, 10]$. A similar approach on adaptive time-stepping schemes is demonstrated in [15] and shows accurate results.

Table 4 Results of the 2d fsi-3 Benchmark with time step size $k = 0.004$s, $k = 0.002s$ and $k = 0.001$s. We indicate the average between maximum and minimum value within the interval $I = [9, 10]$ as well as the deviation from the average to the maximum and the minimum, see (16)

Level	$u_x \times 10^{-3}$	$u_y \times 10^{-3}$	drag $\times 10^2$	lift$\times 10^2$
2	-2.5207 ± 2.4006	1.2285 ± 32.6701	4.4132 ± 0.2599	0.0921 ± 1.6816
3	-3.3174 ± 3.1032	1.2753 ± 36.8303	4.5564 ± 0.2941	0.0998 ± 1.4003
4	-2.8430 ± 2.6869	1.4665 ± 34.6516	4.5892 ± 0.2703	0.0363 ± 1.5581
5	-2.8716 ± 2.7174	1.4960 ± 34.8656	4.6031 ± 0.2778	0.0248 ± 1.5730
6	-2.8644 ± 2.7111	1.4995 ± 34.8329	4.6043 ± 0.2787	0.0237 ± 1.5737
Lev	$u_x \times 10^{-3}$	$u_y \times 10^{-3}$	drag $\times 10^2$	lift$\times 10^2$
2	-2.6363 ± 2.5088	1.1688 ± 33.2886	4.4445 ± 0.2741	0.0667 ± 1.5742
3	-3.2725 ± 3.0748	1.2874 ± 36.7999	4.5753 ± 0.2964	0.0683 ± 1.3963
4	-2.8466 ± 2.6874	1.4604 ± 34.6813	4.5915 ± 0.2702	0.0319 ± 1.5509
5	-2.8850 ± 2.7255	1.4774 ± 34.9795	4.6037 ± 0.2786	0.0252 ± 1.5675
6	-2.8841 ± 2.7250	1.4785 ± 34.9845	4.6050 ± 0.2798	0.0242 ± 1.5699
Lev	$u_x \times 10^{-3}$	$u_y \times 10^{-3}$	drag $\times 10^2$	lift$\times 10^2$
2	-2.7866 ± 2.6462	1.1851 ± 33.9983	4.4712 ± 0.2887	0.0439 ± 1.4837
3	-3.2432 ± 3.0478	1.2869 ± 36.7179	4.5884 ± 0.2979	0.0531 ± 1.4114
4	-2.8317 ± 2.6716	1.4550 ± 34.6089	4.5925 ± 0.2686	0.0297 ± 1.5425
5	-2.8844 ± 2.7234	1.4674 ± 34.9896	4.6034 ± 0.2775	0.0250 ± 1.5610
6	-2.8900 ± 2.7290	1.4690 ± 35.0322	4.6049 ± 0.2791	0.0245 ± 1.5659

5.4.1 Reference Values for the 2d Configuration

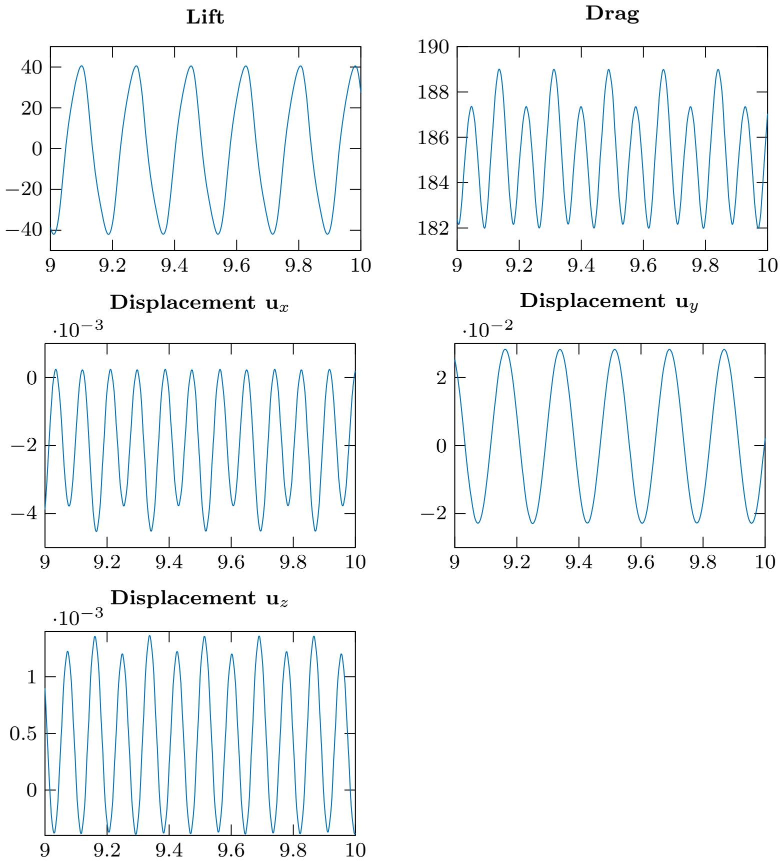
We summarize the results in Table 4. Within the interval $I = [9, 10]$ we indicate for each functional $j(\cdot)$ the average of minimum and maximum as well as the deviation, i.e.

$$\frac{\min_{t\in I} j(t) + \max_{t\in I} j(t)}{2} \pm \frac{\max_{t\in I} j(t) - \min_{t\in I} j(t)}{2}. \tag{16}$$

The presented values in the table for different time step sizes and spatial mesh sizes indicate convergence of the algorithm in space and a dominance of the spatial discretization error on the coarse grids in comparison to the temporal discretization error. These results are in very good agreement to the values found in literature [36].

5.4.2 Reference Values for the 3d Configuration

In 3d, we evaluate the displacement of the elastic beam in the point B and also compute the drag and lift coefficients around the whole cylinder and the flag. In Fig. 3 we show the different functionals as function over the time interval $I = [9, 10]$. In addition, we summarized the average of maximal and minimal value as well as the deviation (see 16) for different meshes and for different time step sizes k. To draw a conclusion on the convergence or to present reference values, the computation has to be repeated on even finer meshes in the future.



lev	$u_x \cdot 10^{-3}$	$u_y \cdot 10^{-3}$	$u_z \cdot 10^{-3}$	drag $\cdot 10^2$	lift
1	-5.131 $\pm$ 5.501	1.784 $\pm$ 36.391	-0.772 $\pm$ 0.772	1.863 $\pm$ 0.099	3.752 $\pm$ 70.452
2	-2.943 $\pm$ 3.157	1.503 $\pm$ 30.098	-0.315 $\pm$ 0.315	1.863 $\pm$ 0.027	-1.491 $\pm$ 49.471
3	-2.176 $\pm$ 2.419	2.766 $\pm$ 25.687	-0.196 $\pm$ 0.196	1.857 $\pm$ 0.036	-0.704 $\pm$ 41.347

lev	$u_x \cdot 10^{-3}$	$u_y \cdot 10^{-3}$	$u_z \cdot 10^{-3}$	drag $\cdot 10^2$	lift
1	-4.330 $\pm$ 4.581	2.941 $\pm$ 35.043	0.438 $\pm$ 2.423	1.841 $\pm$ 0.098	3.063 $\pm$ 66.003
2	-2.788 $\pm$ 3.011	1.647 $\pm$ 29.590	0.484 $\pm$ 1.117	1.863 $\pm$ 0.025	-1.272 $\pm$ 49.646
3	-2.161 $\pm$ 2.401	2.750 $\pm$ 25.643	0.490 $\pm$ 0.881	1.857 $\pm$ 0.035	-0.728 $\pm$ 41.407

lev	$u_x \cdot 10^{-3}$	$u_y \cdot 10^{-3}$	$u_z \cdot 10^{-3}$	drag $\cdot 10^2$	lift
1	-3.875 $\pm$ 4.114	0.659 $\pm$ 35.614	0.135 $\pm$ 2.234	1.824 $\pm$ 0.091	1.910 $\pm$ 64.380
2	-2.650 $\pm$ 2.881	1.566 $\pm$ 29.224	0.435 $\pm$ 1.091	1.861 $\pm$ 0.025	-1.119 $\pm$ 48.186
3	-2.143 $\pm$ 2.383	2.699 $\pm$ 25.594	0.486 $\pm$ 0.877	1.855 $\pm$ 0.035	-0.717 $\pm$ 41.299

Fig. 3 3d fsi-3 configuration. Top: functional values as function over the time interval $I = [9, 10]$ for $k = 0.001$s and mesh level 3. Bottom: results for time step sizes $k = 0.004$s, $k = 0.002$s, $k = 0.001$s and all three mesh levels. We indicate the average between maximum and minimum value within the interval $I = [9, 10]$ as well as the deviation from the average to the maximum and the minimum, see (16)

5.5 Performance of the Linear Solver

To test the linear iterative solver presented in Sect. 4, we recomputed the solution on different mesh levels for the 2d and 3d benchmark configuration on the time interval $I = [9, 9.5]$ with time step size $k = 0.002$s (250 steps). The beam oscillates in this time interval. Hence, due to the strong coupling, the solution of the Newton system is very challenging and the fluid as well as the solid elasticity problem have both to be solved very accurately.

The Newton algorithm in every time step terminates, if the residual is reduced by eight orders of magnitude (relative tol $= 10^{-8}$) or if the absolute value, so the residual, falls below 10^{-8}. In every Newton step, the iterative solver for the linear problem (10) reduces the error by a factor of 10^{-4}. The parameter $\gamma_{nt} = 0.05$ is chosen as in Sect. 5.3 to decide, if the Jacobian of the momentum equation (10) is reassembled in the next Newton step. The mesh motion subproblem (12) is a linear elasticity problem and hence can be solved very efficiently with the geometric multigrid solver. Nevertheless, as we have to solve it after every Newton step, the solution of the linear system has still a high contribution to the computational time. The matrix for the linear meshmotion problem (12) only has to be assembled once in the first step.

In the following, we will only present averaged values. By "mean time per Newton step" we denote the average time of each step, measured over all 250 time steps. Hence, this average value also includes the time to reassemble the Jacobian, whose assembly incidence depends on the Newton rate, see Sect. 5.3. To make the values comparable with other solution approaches, we additionally present the mean time to assemble one Jacobian of the momentum equation (10). In the case of the direct solver, this includes the times for preparation and computation of the LU decomposition. In the case of the ILU and Vanka smoother the assemble times include the time to compute the ILU or the LU of the block matrices $\mathbf{A}_i$.

5.5.1 Dependency on the Vanka Patch Size (2d fsi-3)

Concerning the Vanka smoother, the question arises, how large we should choose the patches P_i to solve the linear system coming from the momentum equation (10) most efficiently. The simple structure of the Vanka solver enables to use different patch sizes in the fluid and solid domain. To test different blocking strategies we recorded the computational time for the 2d fsi-3 benchmark on the finest mesh level 6 and present the mean number of Newton steps and matrix assemblies per time step in Table 5. We either choose patches consisting of one element ($n_p = 3^2 \cdot 3 = 27$) or patches stretching over four adjacent elements ($n_p = 5^2 \cdot 3 = 75$). This yields local matrices of size $\mathbf{A}_i \in \mathbb{R}^{27\times 27}$ or $\mathbf{A}_i \in \mathbb{R}^{75\times 75}$ if larger patches are used.

We can observe that the minimal number of GMRES steps to solve (10) in every Newton step can be obtained, by using $n_p = 75$. If we only use the degrees of freedom of one element ($n_p = 27$) as block on the solid domain, the number of GMRES steps increases and the Newton convergence suffers. This effect cannot be observed, if we only use smaller patches within the fluid domain, but large patches in the solid. As computational times are reasonable small for 2d computations, we will always use larger patches of size $n_p = 75$ in the Vanka smoother for all 2d computations to follow.

In 3d the same blocking strategy would correspond to combining 8 elements to one block, resulting in $n_p = 5^3 \cdot 4 = 500$ and matrices of size $\mathbf{A}_i \in \mathbb{R}^{500\times 500}$. This strategy is forbiddingly expensive with increasing memory and time consumption for each block-LU. As the results in Table 5 show that it is sufficient to use small patches in the fluid domain, we will combine large patches with $n_p = 500$ in the solid with smaller patches of size $n_p = 3^3 \cdot 4 = 108$ in the fluid domain for all 3d computations to follow.

Table 5 Vanka Blocking strategy for the 2d test case. We either choose every element as one block or combine 4 elements to one block on the fluid ($\mathcal{F}$) and solid ($\mathcal{S}$) domain. We present the number of Newton steps, matrix assembles and number of GMRES steps per linear solve of (10) on mesh level 6 and the computational time relative to the $\mathcal{F}$: 27/$\mathcal{S}$: 27 case

n_P	$\mathcal{F}$: 27/$\mathcal{S}$: 27	$\mathcal{F}$: 75/$\mathcal{S}$: 75	$\mathcal{F}$: 75/$\mathcal{S}$: 27	$\mathcal{F}$: 27/$\mathcal{S}$: 75
Newton steps	6.87	5.10	6.83	5.10
Matrix assemblies	2.87	1.23	2.86	1.23
GMRES per Newton	20.58	12.60	17.16	15.26
Relative comp. time	100%	55%	95%	61%

5.5.2 Geometric Multigrid Performance in 2d and 3d (Sequential Computations)

All computations have been carried out on an Intel(R) Core(TM) i7-7700 CPU @ 3.60GHz. Single Core performance only is used in this section. In Fig. 4 we show the results for both 2d and 3d benchmark problems on sequences of meshes.

In the top row we present the memory consumption (in 3d, the finest mesh level exceeded the available memory). In particular the 3d results show the expected superiority of iterative solvers as compared to the direct linear Solver UMFPACK [12] with a non-optimal scaling. The Vanka smoother requires slightly more memory which comes from the overlap of degrees of freedom between the different blocks. The middle plot of Fig. 4 shows the resulting computational time. According to our previous study [32], the multigrid method is not able to beat the direct solver in 2d. The situation dramatically changes in 3d, where the direct solver shows a strongly non-optimal scaling. The multigrid solvers shows nearly linear scaling for both ILU and Vanka smoothing. Concerning the ILU smoother, this is an improvement to our previous study presented in [32], where the multigrid solver was performed in a purely monolithic setting and an ILU that consists of local blocks coupling pressure, velocity and deformation. Here no convergence could be achieved on fine meshes. We note that the 3d benchmark problem considered in this paper is by far more challenging than the problem investigated in [32,33] as it comprises very large deformation and hence strong nonlinearities in the solid and also in the ALE map. The lowest row shows the time for one assembly of the Jacobian, including the computational times for preparing the direct solver, the ILU smoother and the Vanka smoother. Here, the main discrepancies between the direct solver and the multigrid methods arise. Since we do not recompute the Jacobian in every Newton step (not even in every time step), it is no inconsistency that the assembly time is larger than the complete time per Newton step. In 2d the results appear slightly sub-optimal. This is due to the necessity to assemble the matrices along the complete multigrid hierarchy yielding a scaling of order $\mathcal{O}(n \log n)$.

The average number of Newton steps and the average number of Jacobians assembled for all time steps within the intervals are gathered in Table 6. In addition, we present the mean number of GMRES steps to solve the linearized momentum equation, problem (10), once. The values show that the average number of matrix assemblies in each time step can be below 1. This is due to the approximation of the Jacobian by reassembling it, only if the convergence rates deteriorate. Both multigrid approaches, Vanka and ILU are very robust with regard to mesh refinement. The linear iteration counts rise only slightly.

Figure 5 shows the average number of GMRES steps required for both Vanka and for ILU smoothing in every time step. The values fluctuate due to the oscillatory motion of the beam.

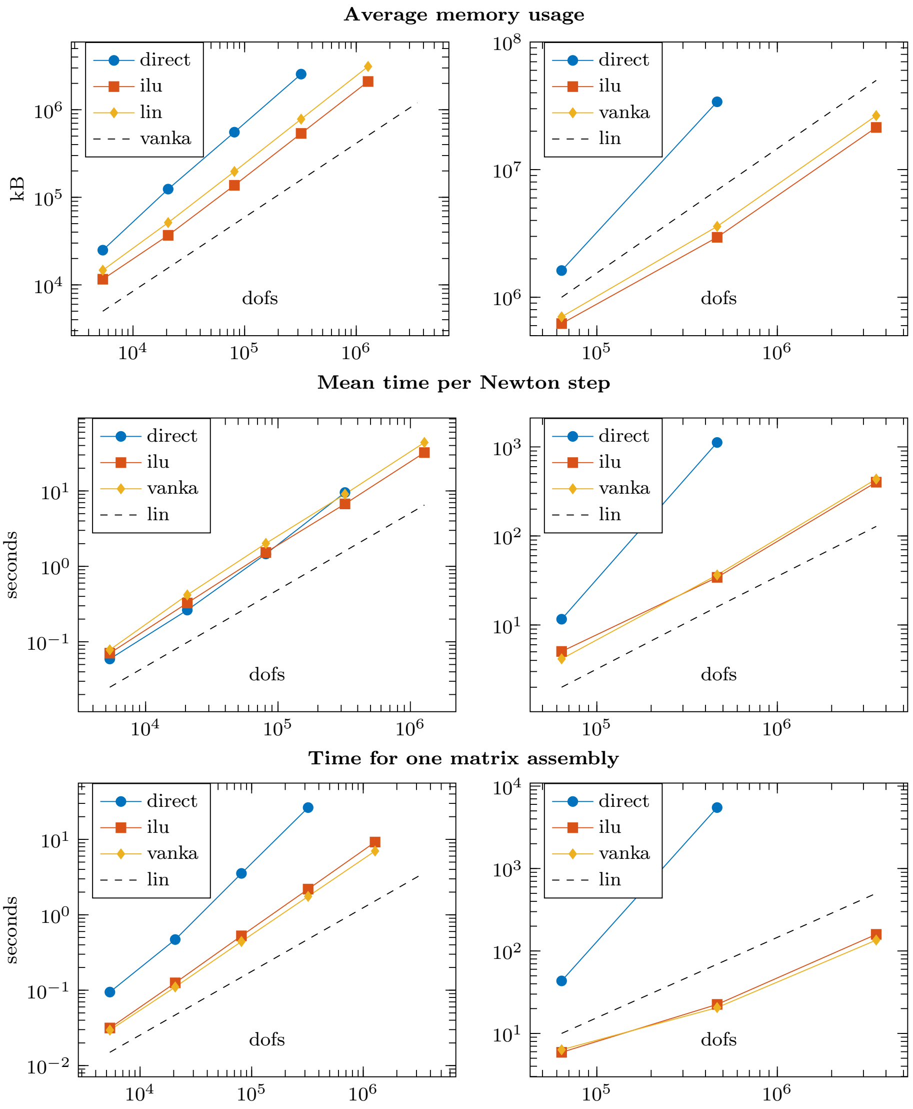


Fig. 4 Performance of the multigrid solver in 2d (left) and 3d (right) in comparison to a direct solver. On different meshes with increasing numbers of degrees of freedom, we compare the performance of the direct solver UMFPACK [12] with the multigrid solver based on ILU smoothing and based on Vanka smoothing. No parallelization is employed. From top to bottom: average memory usage, average time per Newton step and time to assemble one system matrix including preparation of the direct solver and the smoothers. Note that we do not reassemble the Jacobian in every Newton step, and therefore, assembly times can be higher than mean Newton times (which include the assembly)

According to Fig. 4 we need 43.88s for each Newton step on mesh level 6 in the 2d configuration. And according to Table 6 an average of 5.1 Newton steps. The mean computational time per time step is $43.88\text{s} \cdot 5.1 = 223.49$ s, whereby an average of $7.01\text{s} \cdot 1.23 = 8.66\text{s}$ are used to construct the Jacobian. Most of the computational time is spent by the linear solver. In every Newton step the linear solver needs about $(223.9\text{s} - 8.66\text{s})/5.1 = 42\text{s}$.

Table 6 Average number of Newton steps, matrix assemblies per time step and average number of GMRES steps within each Newton step. Top: 2d benchmark. Bottom: 3d benchmark

	Mesh level 4			Mesh level 5			Mesh level 6		
	Direct	ILU	Vanka	Direct	ILU	Vanka	Direct	ILU	Vanka
Newton steps	5.15	5.04	5.14	5.21	5.04	5.16	–	5.17	5.10
Matrix assemblies	1.14	0.90	0.90	1.14	0.97	0.96	–	0.94	1.23
GMRES per Newton	–	11.07	9.53	–	11.07	10.65	–	13.08	12.60
	Mesh level 1			Mesh level 2			Mesh level 3		
	Direct	ILU	Vanka	Direct	ILU	Vanka	Direct	ILU	Vanka
Newton steps	5.27	5.40	5.22	5.27	5.58	5.23	–	5.24	5.15
Matrix assemble	1.00	1.23	0.99	0.95	1.22	0.95	–	1.22	1.00
GMRES per Newton	–	13.20	4.81	–	14.52	9.35	–	15.33	10.59

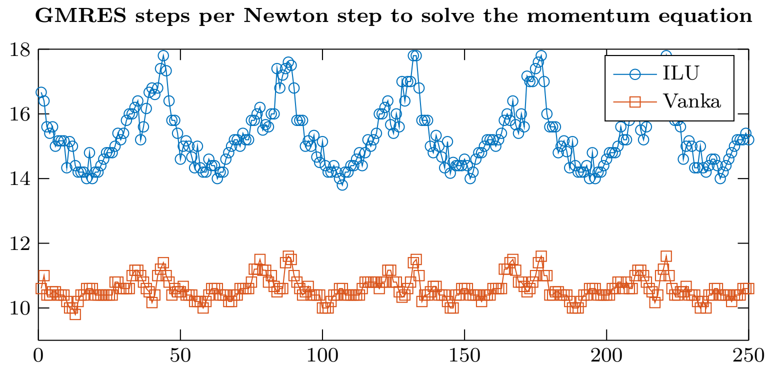


Fig. 5 3d fsi-3 configuration: Mean number of GMRES steps to solve the momentum equation (10) within every Newton step plotted over time steps

5.5.3 Dependency on the Time Step Size

The approach for splitting and reducing the Jacobian is based on two steps, skipping the ALE derivatives in the fluid problem and inserting the discretized velocity-deformation relation (4) into the solid's stress tensor, see (5). Both steps rely on time-stepping. Small time steps will limit the domain motion from step to step, such that the influence of the ALE derivative and also the update in (4) is small. In Table 7 we investigate the robustness of the solution approach versus the time step size. Besides average iteration counts we also indicate the total number of GMRES steps that would be required to advance one time unit (1s), i.e. the number of steps necessary for $1/k$ time steps. This value stands for the overall effort of the method. However, we do not relate this effort to the resulting error. Further, this value does not include times for assembling the Jacobian that can be essential. The values in Table 7 show an increase of Newton steps with larger time step sizes, in particular for $k = 0.01$s. This is expected due to the decreased impact of the mass matrix and since we drop the ALE derivatives. The number of GMRES iterations per time Newton step is very stable up to $k = 0.004$s and starts to increase at $k = 0.01$s. Despite this, the overall number of GMRES steps per time unit still shows increased efficiency for larger time step sizes.

Table 7 Average number of Newton steps, matrix assemblies per time step and average number of GMRES steps within each Newton step using the Vanka smoother. Further we indicate the average number of GMRES required to advance one time unit, e.g. for $1/k$ time steps. Top: 2d benchmark at meshlevel 6. Bottom: 3d benchmark at meshlevel 3

	$k = 0.01$s	$k = 0.004$s	$k = 0.002$s	$k = 0.001$s
Newton steps	8.16	5.92	5.10	4.82
Matrix assemblies	4.5	1.72	1.23	0.86
GMRES per Newton	21.63	15.34	12.61	10.71
GMRES per time unit	17 650	22 703	32 155	51 622
	$k = 0.01$s	$k = 0.004$s	$k = 0.002$s	$k = 0.001$s
Newton steps	8.24	6.09	5.15	4.94
Matrix assemblies	4.32	1.50	1.00	0.80
GMRES per Newton	13.58	10.86	10.59	10.08
GMRES per time unit	11 190	16 534	27 269	49 795

To classify these results in the context of the 2d benchmark problem we refer to [35], where a meta-analysis on the temporal discretization of various solution approaches to this benchmark problem is given. It is found that time steps $k \ll 0.01$s are used by nearly all contributors to this benchmark and that $k \approx 0.01$s must be considered as too large in terms of the approximation error. A possible explanation for the need of such small time steps is found in high frequent oscillations of the solid that have to be resolved. With this in mind, the solution approach presented here can be regarded as highly efficient and appropriate for nonstationary fluid-structure interactions.

Considering stationary problems the coupling character of fluid-structure interactions strongly changes. Instead of a parabolic/hyperbolic-type setup, we then deal with a coupling of elliptic/elliptic character. The velocity-deformation equation $\mathbf{v}_s = \partial_t \mathbf{u}_s$ is not present as the solid's velocity, being zero, is not even part of the system. For a direct-to-steady-state solution approaches with stronger coupling must be considered. We refer to [32] and in particular to [2], where a variant of the former approach is realized for stationary problems.

5.5.4 Parallelization

The Vanka smoother (based on a Jacobi iteration) has the advantage that it can be easily parallelized. We introduce a cell wise coloring of the Vanka patches. Colors are attributed by a simple ad hoc algorithm. We run over all patches; if a patch is not already labeled with a color, we will label it and block all its neighbours that share a common degree of freedom for this color. Then, we continue with the next color. This algorithm is not optimal in terms of "numbers of colors" and also not optimal in terms of "balanced number of elements per color" but adequate for our purpose with a moderate number of cores. As different patch sizes for fluid and solid domain are used in 3d, a different color is always allocated to fluid and solid patches, such that a good load balancing is possible. The finest mesh level in 3d is partitioned into 22 colors (13 within the fluid, 9 in the solid domain), see Fig. 6, whereby the number of patches in each color ranges between 6 716 and 2 within the fluid domain, and is constant with 80 patches per color within the solid. About 99% of the fluid patches belong to colors containing at least 500 patches, such that very little overhead must be expected due to suboptimality of partitioning the remaining colors (as long as a moderate number of threads

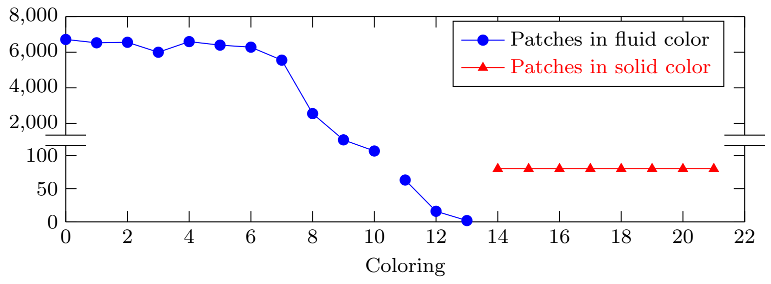


Fig. 6 Coloring for avoiding memory collisions in the parallel Vanka smoother for the finest 3d mesh with about $3.5 \cdot 10^6$ dofs. The fluid patches consist of 108 dofs each, while the solid patches couple 500 dofs. The smallest fluid color has only 2 patches

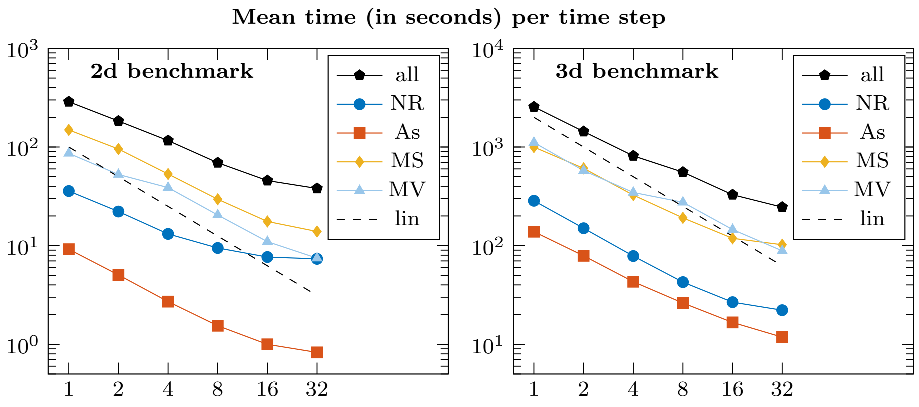


Fig. 7 Strong scalability test. Mean time per time step (all) to compute the Newton residual (NR), assemble the Jacobian in (10) (As), multilevel solver (MS), matrix-vector multiplication (MV) in 2d (left) and 3d (right) using 1–32 threads on mesh level 6 and level 3

is considered). Our algorithms yields solid colors with 80 patches each. While 80 is dividable by 16, it is not dividable by 32. Hence, the potential efficiency of functions depending on this coloring is reduced to about 0.8 (for 32 threads).

Furthermore, we parallelized the matrix vector product. Although in principle trivial to parallelize, we suffer from the usually memory bandwidth restrictions that will limit possible speedups for matrix vector products. All parallelization is done in *OpenMP* [27]. We note that the parallelization is not the focus of this work. Only first steps have been undertaken and the implementation allows for further optimization.

Similar to Sect. 5.5.2 we recompute the 2d and 3d problem on the time-interval $I = [9, 9.5]$ with the step size $k = 0.002$s using the finest refinement levels 6 (in 2d) and 3 (in 3d). The mean computational time per time step on an Intel(R) Xeon(R) Gold 6150 CPU @ 2.70GHz is given in Fig. 7 in a strong scalability test. In 3d we can observe that the parallelization of all ingredients scales rather well. If we double the number of cores the computational time reduces by a factor of 0.57. With 32 threads we achieve a speed up of about 10 in comparison to single core performance. The drop in efficiency from 16 to 32 threads (in 3d) is clearly visible in the assembly of the residual and the application of the Vanka smoother, two functions that strongly depend on the coloring of the patches.

Table 8 Distribution of the computational time to the main ingredients of the Newton-multigrid solver: integration of the nonlinear residual, assembly of the Jacobian, matrix-vector products and application of the Vanka smoother. The numbers do not add up to 100% as some parts, like the memory management, are not included in the measurement

# Threads	Total (%)	Residual (%)	Matrix (%)	MV product (%)	Vanka (%)
1	100	11	5	44	39
4	100	10	5	43	40
16	100	8	5	45	36

In Table 8 we show, how the distribution of the computational time to the different ingredients develops for an increasing numbers of threads. These results belong to the 3d benchmark problem on the finest mesh level 3. The numbers show that more than 80% of the time is spend in linear algebra routines like sparse matrix-vector products and the application of the Vanka smoother. These operations are mainly limited by the memory bandwidth. The very low contribution of only 5% for the matrix assembly could lead to the conclusion that a matrix free implementation might be the proper choice. However, our implementation requires less than one matrix assembly per time step in average. The multigrid smoother however is applied many hundred times (about 5 Newton steps, 10 GMRES steps each, several Vanka steps). A matrix free implementation on such a low number of threads would hence strongly increase the overall time.

5.5.5 Comparison to Results in the Literature

A direct comparison of the computational times to results published in literature is difficult, since different numerical test-cases are considered. The complexity depends on the problem size (number of degrees of freedom) but also on the density of the system matrices (depending on the finite element space). Another important factor is the impact of the nonlinearity which is influenced by the deformation of the solid and also the time step size. The 3d test case that we discussed in [32] features less deformation and is simpler as compared to the 3d benchmark problem introduced here. Further, computational times are often not presented in full detail. Besides the algebraic parts of the solver, assembly times have an important impact. Thereby the influence due to the strategy for smart Jacobian re-assemblies is significant. Finally, different hardware is used throughout all contributions.

To give a comparison of the various approaches we compute the ratio of “degrees of freedom per second for each time step” and indicate an accurate description of the setup used. Table 9 lists the results for 2d benchmark problems (mostly the fsi-3 problem) and different 3d problems. We only list contributions that can be regarded as fully monolithic and that handle nonlinear fluid-structure interactions including geometric nonlinearities by domain motion.

Due to the reasons mentioned above, these results have to be interpreted with care. They show however that the 2d benchmark problem is handled well by all approaches with little room for improvement. In 3d the direct solver clearly fails. Furthermore, multigrid approaches are superior on fine meshes with a little advantage for the reduced Newton-multigrid approach presented here.

Table 9 Comparison of different strategies found in literature. We indicate degrees of freedom, time step size, number of cores used and the resulting ratio of "degrees of freedom per second computed for each time step". Higher values indicate better performance. Finally we give the source for the values

Approach (2D)	Dofs	k	Cores	Ratio	Source
Direct (UMFPACK)	320,640	0.002 s	1	6463 s^{-1}	Fig. 4, Tab. 6
Direct (MUMPS)	1,800,000	0.01 s	64	4147 s^{-1}	[25, Fig. 8]
Geometric MG	1,202,816	0.01 s	1	2440 s^{-1}	[2, Tab. 1, Tab. 12]^1
GMRES	16,000,000	0.01 s	64	65,040 s^{-1}	[25, Fig. 9]^2
AMG (BGS)	79,869	0.01 s	4	4991 s^{-1}	[19, Sec. 8.3, Fig. 9]^3
Red. Newton-MG	1,276,155	0.002 s	16	27,943 s^{-1}	Fig. 4, Tab. 6
Approach (3D)	**Dofs**	**k**	**Cores**	**Ratio**	**Source**
Direct (UMFPACK)	463,988	0.002 s	1	78.5 s^{-1}	Fig. 4, Tab. 6
Geometric MG	7,600,776	0.01 s	1	3781 s^{-1}	[32, Tab. II, VI, Fig. 4]^5
Geometric MG	120,902	0.01 s	1	245 s^{-1}	[2, Tab. 1, Tab. 12]^{4,5}
GMRES	14,000,000	0.005 s	256	1223 s^{-1}	[25, Fig. 14, 15]^5
AMG (BGS)	3,063,312	0.0001 s	16	9881 s^{-1}	[19, Tab III, V]^5
Red. Newton-MG	3,531,304	0.002 s	32	14,413 s^{-1}	Fig. 4, Tab. 6

^{1} Matrix assembly times in this approach are very high, the performance of the linear solver alone is very good and comparable to the other (iterative) approaches, see [2]

^{2} considers a different 2d test case. [25] only gives computational time per linear iteration with an average of 6 Newton steps per time step [39]

^{3} different 2d test case. A variant, BGS(AMG) with 6 150 s^{-1} is presented, see [19, Fig. 9]

^{4} See remark 1 in the 2d table

^{5} These contributions consider different test-cases

6 Summary

We have introduced a Newton multigrid framework for monolithic fluid-structure interactions in ALE coordinates. The solver is based on two reduction techniques in the Jacobian: first, a condensation of the solid deformation by representing the deformation gradient on the velocity only and second, by skipping the ALE derivatives within the Navier–Stokes equations. This second steps leads to an approximated Newton method but we could show (also in preliminary works) that the time-to-solution even benefits from this approximation, as the computational time for assembling the ALE derivatives is very high. The reduction has two positive effect: the large system of 7 unknowns (in 3d) decomposes into one fluid-solid problem in pressure and velocity with 4 unknowns and two partitioned systems with 3 unknowns each for solving solid and fluid deformation. The second effect is the better conditioning of the coupled system that allows for the use of very simple multigrid smoothers that are easy to parallelize. Also, while ILU smoothing applied to the monolithic system was not convergent in our previous contribution [32], performed well for smoothing the global momentum equations. Combined with first steps of parallelization and in comparison to our past approaches based on a monolithic solution of the complete pressure-velocity-deformation system and partitioned smoothers we could significantly reduce the computational time.

As basis for future benchmarking of 3d fluid-structure interactions we presented an extension of the 2d benchmark problems by Hron and Turek [24] that is by far more challenging (due to larger deformations and a strong dynamic behavior) as compared to a first test case introduced in our past work [32] which has also been considered in [2,25] in very similar studies. It will still require further effort to establish reference values for this new 3d benchmark case.

Our work includes some first simple steps of parallelization which have to be extended in future work. In particular, in order to overcome the memory bandwidth limitations which are common in such memory extensive computations, distributed memory paradigms have to be incorporated [26]. Further, some benefit can be expected by using GPU acceleration for matrix vector product and Vanka smoother.

Acknowledgements Open Access funding provided by Projekt DEAL. Both authors acknowledge the financial support by the Federal Ministry of Education and Research of Germany, Grant Number 05M16NMA (TR) and 05M16WOA (LF). TR further acknowledges the support of the GRK 2297 MathCoRe, funded by the Deutsche Forschungsgemeinschaft, Grant Number 314838170 and LF acknowledges the support of the GRK 1754, funded by the Deutsche Forschungsgemeinschaft, Grant Number 188264188.

References

1. Amestoy, P.R., Guermouche, A., L'Excellent, J.Y., Pralet, S.: Hybrid scheduling for the parallel solution of linear systems. Parallel Comput. **32**(2), 136–156 (2006)
2. Aulisa, E., Bna, S., Bornia, G.: A monolithic Ale Newton-Krylov solver with multigrid-Richardson–Schwarz preconditioning for incompressible fluid-structure interaction. Comput. Fluids **174**, 213–228 (2018)
3. Becker, R., Braack, M.: Multigrid techniques for finite elements on locally refined meshes. Numer. Linear Algebra Appl. **7**, 363–379 (2000). Special Issue
4. Becker, R., Braack, M.: A finite element pressure gradient stabilization for the Stokes equations based on local projections. Calcolo **38**(4), 173–199 (2001)
5. Becker, R., Braack, M., Meidner, D., Richter, T., Vexler, B.: The finite element toolkit GASCOIGNE. http://www.gascoigne.uni-hd.de
6. Braack, M., Richter, T.: Solutions of 3D Navier–Stokes benchmark problems with adaptive finite elements. Comput. Fluids **35**(4), 372–392 (2006)
7. Brummelen, E., Zee, K., Borst, R.: Space/time multigrid for a fluid-structure-interaction problem. Appl. Numer. Math. **58**(12), 1951–1971 (2008)
8. Bungartz, H.J., Schäfer, M. (eds.): Fluid-Structure Interaction. Modelling, Simulation, Optimisation. Lecture Notes in Computational Science and Engineering, vol. 53. Springer (2006). ISBN-10: 3-540-34595-7
9. Bungartz, H.J., Schäfer, M. (eds.): Fluid-Structure Interaction II. Modelling, Simulation, Optimisation. Lecture Notes in Computational Science and Engineering. Springer (2010)
10. Causin, P., Gereau, J., Nobile, F.: Added-mass effect in the design of partitioned algorithms for fluid-structure problems. Comput. Methods Appl. Mech. Eng. **194**, 4506–4527 (2005)
11. Crosetto, P., Deparis, S., Fourestey, G., Quarteroni, A.: Parallel algorithms for fluid-structure interaction problems in haemodynamics. SIAM J. Sci. Comput. **33**(4), 1598–1622 (2011). https://doi.org/10.1137/090772836
12. Davis, T.: Umfpack, an unsymmetric-pattern multifrontal method. ACM Trans. Math. Soft. **30**(2), 196–199 (2014)
13. Deparis, S., Forti, D., Grandperrin, G., Quarteroni, A.: Facsi: a block parallel preconditioner for fluid-structure interaction in hemodynamics. J. Comput. Phys. **327**, 700–718 (2016). https://doi.org/10.1016/j.jcp.2016.10.005

14. Failer, L.: Optimal control for time dependent nonlinear fluid-structure interaction. Ph.D. thesis, Technische Universität München (2017)
15. Failer, L., Wick, T.: Adaptive time-step control for nonlinear fluid-structure interaction. J. Comput. Phys. **366**, 448–477 (2018)
16. Fernández, M., Gerbeau, J.F.: Algorithms for fluid-structure interaction problems. In: Formaggia, L., Quarteroni, A., Veneziani, A. (eds.) Cardiovascular Mathematics: Modeling and Simulation of the Circulatory System, MS & A, vol. 1, pp. 307–346. Springer, Berlin (2009)
17. Fernández, M., Moubachir, M.: A newton method using exact Jacobians for solving fluid-structure coupling. Comput. Struct> **83**, 127–142 (2005)
18. Frei, S.: Eulerian finite element methods for interface problems and fluid-structure interactions. Ph.D. thesis, Universität Heidelberg (2016). https://doi.org/10.11588/heidok.00021590
19. Gee, M., Küttler, U., Wall, W.: Truly monolithic algebraic multigrid for fluid-structure interaction. Int. J. Numer. Method Eng. **85**, 987–1016 (2010)
20. Guennebaud, G., Jacob, B., et al.: Eigen v3. http://eigen.tuxfamily.org (2010)
21. Heil, M., Hazel, A., Boyle, J.: Solvers for large-displacement fluid-structure interaction problems: segregated vs monolithic approaches. Comput. Mech. **43**, 91–101 (2008)
22. Heywood, J., Rannacher, R., Turek, S.: Artificial boundaries and flux and pressure conditions for the incompressible Navier–Stokes equations. Int. J. Numer. Math. Fluids. **22**, 325–352 (1992)
23. Hron, J., Turek, S.: A monolithic FEM/Multigrid solver for an ALE formulation of fluid-structure interaction with applications in biomechanics. In: Bungartz, H.J., Schäfer, M. (eds.) Fluid-Structure Interaction: Modeling, Simulation, Optimization. Lecture Notes in Computational Science and Engineering, pp. 146–170. Springer, Berlin (2006)
24. Hron, J., Turek, S.: Proposal for numerical benchmarking of fluid-structure interaction between an elastic object and laminar incompressible flow. In: Bungartz, H.J., Schäfer, M. (eds.) Fluid-Structure Interaction: Modeling, Simulation, Optimization. Lecture Notes in Computational Science and Engineering, pp. 371–385. Springer, Berlin (2006)
25. Jodlbauer, D., Langer, U., Wick, T.: Parallel block-preconditioned monolithic solvers for fluid-structure interaction problems. Int. J. Numer. Methods Eng. **117**(6), 623–643 (2019)
26. Kimmritz, M., Richter, T.: Parallel multigrid method for finite element simulations of complex flow problems on locally refined meshes. Numer. Linear Algebra Appl. **18**(4), 615–636 (2010)
27. Klemm, M., Supinski, B., (eds.): OpenMP Application Programming Interface Specification Version 5.0. Independently published (2019)
28. Langer, U., Yang, H.: Recent development of robust monolithic fluid-structure interaction solvers. In: Fluid-Structure Interaction. Modeling, Adaptive Discretization and Solvers. Radon Series on Computational and Applied Mathematics, vol. 20, pp. 169–192. de Gruyter (2017)
29. Molnar, M.: Stabilisierte Finite Elemente für Strömungsprobleme auf bewegten Gebieten. Master's thesis, Universität Heidelberg (2015)
30. Pironneau, O.: An energy preserving monolithic eulerian fluid-structure numerical scheme. Chinese Annals of Mathematics **39**, (2016). https://doi.org/10.1007/s11401-018-1061-9
31. Pironneau, O.: An Energy stable Monolithic Eulerian Fluid-Structure Numerical Scheme with compressible materials (2019). arXiv:1607.08083
32. Richter, T.: A monolithic geometric multigrid solver for fluid-structure interactions in ALE formulation. Int. J. Numer. Meth. Eng. **104**(5), 372–390 (2015)
33. Richter, T.: Fluid-structure Interactions. Models, Analysis and Finite Elements. Lecture Notes in Computational Science and Engineering, vol. 118. Springer, Berlin (2017)
34. Richter, T., Wick, T.: Finite elements for fluid-structure interaction in ALE and Fully Eulerian coordinates. Comput. Methods Appl. Mech. Eng. **199**(41–44), 2633–2642 (2010)
35. Richter, T., Wick, T.: On time discretizations of fluid-structure interactions. In: Carraro, T., Geiger, M., Körkel, S., Rannacher, R. (eds.) Multiple Shooting and Time Domain Decomposition Methods. Contributions in Mathematical and Computational Science, vol. 9, pp. 377–400. Springer, Berlin (2015)
36. Turek, S., Hron, J., Madlik, M., Razzaq, M., Wobker, H., Acker, J.: Numerical simulation and benchmarking of a monolithic multigrid solver for fluid–structure interaction problems with application to hemodynamics. Technical report, Fakultät für Mathematik, TU Dortmund (2010). Ergebnisberichte des Instituts für Angewandte Mathematik, Nummer 403
37. Turek, S., Rivkind, L., Hron, J., Glowinski, R.: Numerical study of a modified time-stepping theta-scheme for incompressible flow simulations. J. Sci. Comput. **28**(2–3), 533–547 (2006)
38. Wall, W.: Fluid-structure interaction with stabilized finite elements. Ph.D. thesis, University of Stuttgart (1999). Urn:nbn:de:bsz:93-opus-6234
39. Wick, T.: Personal communication. University of Hannover (September 2019)

40. Yirgit, S., Schäfer, M., Heck, M.: Grid movement techniques and their influence on laminar fluid-structure interaction rpoblems. J. Fluids Struct. **24**(6), 819–832 (2008)
41. Zee, K., Brummelen, E., Borst, R.: Goal-oriented error estimation and adaptivity for free-boundary problems: the domain-map linearization approach. SIAM J. Sci. Comput. **32**(2), 1074–1092 (2010)
42. Zee, K., Brummelen, E., Borst, R.: Goal-oriented error estimation and adaptivity for free-boundary problems: the shape-linearization approach. SIAM J. Sci. Comput. **32**(2), 1093–1118 (2010)