Computational Mechanics
https://doi.org/10.1007/s00466-025-02689-z

ORIGINAL PAPER

Adaptive tolerances for staggered solution algorithms in fluid-structure interaction

Christian Kühne[1] · Georgios Bletsos[2] · Sven Lutz Möller[1] · Alexander Düster[1] · Lars Radtke[1,3]

Received: 20 February 2025 / Accepted: 21 August 2025

Abstract
Staggered solution algorithms are a well-known alternative to monolithic approaches for solving strongly coupled multi-field problems. A coupling between the subproblems is achieved by exchanging coupling quantities between the solvers. Implicit coupling schemes are realized by letting the solvers solve each load or time step repeatedly until convergence up to a given coupling tolerance is achieved. In fluid-structure interaction as well as a variety of other problem classes, the equations governing the individual fields are nonlinear. Accordingly, each solver performs an inner iterative solution procedure that terminates once an inner tolerance is reached. The basic idea of this work builds on the possibility to adaptively adjust these inner tolerances based on carefully designed rules while preserving the black-box nature of the solvers. The resulting coupling scheme yields significant improvements in computational efficiency compared to classical schemes where the inner solver tolerances are held fixed. This is demonstrated in several numerical examples. The idea is tested in combination with state-of-the-art convergence acceleration schemes and can be realized within any staggered solution approach by only minor modifications to the participating solvers.

1 Introduction

Multi-field problems are an important topic for engineering applications. These problems deal with the interaction between two or more subproblems, such as those describing a fluid and those describing an elastic structural subproblem. Such multi-field problems are called fluid-structure interaction (FSI) problems and are a widespread area of research. FSI plays an important role in, for example, biomedical applications such as cardiovascular blood flow in the arteries [4, 42], maritime applications such as flexible propellers [28, 41], or floating wind turbines [50].

Various solution techniques have been developed to address the complex coupling between the subproblems in FSI. These techniques can be classified into partitioned and monolithic coupling approaches [3, 8]. In monolithic approaches, the coupled problem is represented by a single system of equations solved by a single solver. This approach yields a stable coupling but requires the implementation of specially designed software. Partitioned approaches solve the two subproblems independently, realizing the coupling by iteratively exchanging coupling quantities at the coupling interface until convergence is reached. Specifically, we apply a staggered scheme in which the subproblems are solved in sequence. This allows for the reuse of existing specialized field solvers, which can be treated as black boxes. However, these methods tend to be less stable than monolithic approaches. To enhance the stability and convergence rate of staggered algorithms, acceleration methods such as Aitken relaxation [1, 23, 31] or interface quasi-Newton least-squares [10, 12, 20, 42] are used.

In recent decades, several advanced staggered coupling algorithms have been developed to solve multi-field problems faster or with greater precision. For instance, the multi-level quasi-Newton coupling algorithms from [13] achieve faster simulations by solving the problems on multiple grid levels and computing the quasi-Newton terms on

✉ Christian Kühne
christian.kuehne@tuhh.de

[1] Hamburg University of Technology, Numerical Structural Analysis with Application in Ship Technology, Institute for Ship Structural Design and Analysis (M-10), Am Schwarzenberg Campus 4(C), 21073 Hamburg, Germany

[2] Hamburg University of Technology, Institute for Fluid Dynamics and Ship Theory (M-8), Am Schwarzenberg Campus 4(C), 21073 Hamburg, Germany

[3] Chair of Structural Mechanics, University of Rostock, Albert-Einstein-Str. 2, 18057 Rostock, Germany

Published online: 12 December 2025

coarser grids. Other advanced coupling methods presented in [7, 33] involve parallel simulations of the subproblems to reduce computational time.

Due to the nature of most FSI applications, the underlying equations of the subproblems are nonlinear and require an inner iterative solution procedure. Nonlinear solvers for the different fields can vary, for example with regard to the initial conditions used for the solution process. In [18], the impact of resetting the solver's information on the initial conditions between successive coupling iterations within a time step was investigated. We refer to these as solvers with and without resetting properties. A solver with resetting properties uses the same initial values in each coupling iteration within a time step (typically, the converged result from the previous time step). A solver without resetting uses the last solution from the previous coupling iteration as the initial condition for the next one. The choice for one of these solver variants has a large impact on the efficiency of the method in terms of the number of coupling iterations and the inner solver iterations. However, this aspect has not yet been studied in detail in the literature.

Generally, adaptive discretization parameters are a promising way to reduce the computational time of FSI and other simulations. Various adaptive strategies have been developed to reduce computation time or enhance accuracy. One possible approach involves adaptive time-stepping schemes as described in [32], while [44] investigated adaptive remeshing strategies for FSI simulations. In [21, 45], adaptively adjusted tolerances were used for a conjugate gradient (CG) solver [22] applied within nonlinear problems solved with the Newton-Raphson method. Iterative solver schemes, like the CG-solver, continue until the desired quantity reaches a specified solver tolerance. Typically, these tolerances are set at the start of the simulation and remain fixed. However, they may also be adaptively adjusted within each nonlinear iteration (Newton-Raphson iteration) while solving the linear system. Generally, the idea of adaptive tolerances is to change the tolerances of an inner iterative scheme according to certain rules based on an outer solution procedure in order to speed up the computation while preserving the accuracy [21, 45].

In this paper, we apply the concept of adaptive tolerances in the FSI context, where each subfield solver iterates multiple times per time step until the coupling iterations are considered converged. Furthermore, these solvers undergo an inner nonlinear iterative solution process until an inner tolerance is satisfied. The idea of setting these inner solver tolerances in an adaptive way was investigated in the master thesis of one of the authors[1]. Unlike [21], where the linear system is solved with a CG method using adaptive tolerances, we use fixed tolerances to solve the linear system. The adaptively adjusted tolerances are applied one level higher, i.e., on the level of the solver's inner iterations – for example based on the Newton-Raphson algorithm in structural mechanics or a pressure-corrector scheme in fluid mechanics.

In [46] the maximum number of iterations for each subfield solver is limited to a fixed value during the coupling iterations. This can have an effect on the computational cost similar to that expected from adjusting the tolerances of the solver, which is the main focus of this work. Further, we combine this with an investigation of the effects of resetting a solver's inner state in each coupling iteration.

The remainder of this paper is structured as follows. Section 2 introduces the solution strategies for coupled problems and the implementation aspects for designing a coupled solver including the option to reset or not reset the inner state of the solvers. After that, in Sec. 3, the idea of adaptive tolerances is described for a simple problem, and more advanced rules for selecting the tolerance are introduced. Section 4 introduces the fluid and structural subproblem, and Sec. 5 focuses on different numerical investigations demonstrating the potential of the idea. Section 6 gives a summary of the findings.

2 Solution algorithms for coupled problems

In order to establish a common notation, we formulate the solution algorithms for dynamic problems that are solved using a time-marching scheme. To this end, the problem is solved at the discrete time instances t_k, and the solution at the initial time instance $t_0 = 0$ is known. In a general sense, one may formulate a coupled problem involving two subproblems $\mathbf{r}_k^a = \mathbf{0}$ and $\mathbf{r}_k^b = \mathbf{0}$ as

$$\begin{cases} \mathbf{r}_k^a(\mathbf{y}_k^a, \mathbf{c}^a(\mathbf{y}_k^b)) = \mathbf{0}, \\ \mathbf{r}_k^b(\mathbf{y}_k^b, \mathbf{c}^b(\mathbf{y}_k^a)) = \mathbf{0}. \end{cases} \tag{1}$$

Therein, $\mathbf{y}_k^a$ denotes the state of subproblem a, and $\mathbf{c}_k^a(\mathbf{y}_k^b)$ denotes the coupling quantities that appear in subproblem a and depend on the state of the other subproblem b. For $\mathbf{y}_k^b$ and $\mathbf{c}_k^b(\mathbf{y}_k^a)$, the situation is reversed accordingly.

A monolithic solution strategy based on the Newton-Raphson algorithm can then be formulated as follows:

1. Choose initial guess:

$$\begin{bmatrix} \mathbf{y}_{k+1,0}^a \\ \mathbf{y}_{k+1,0}^b \end{bmatrix} = \begin{bmatrix} \mathbf{y}_k^a \\ \mathbf{y}_k^b \end{bmatrix} \tag{2}$$

[1] Unpublished but can be provided upon request: S. L. Möller. "Beschleunigung von gekoppelten Simulationen durch adaptive Toleranzanpassung am Beispiel der Fluid-Struktur-Interaktion". Master thesis. Hamburg University of Technology, 2021

2. Solve linearized system:

$$\begin{bmatrix} \mathbf{D}^{a,a}_{k+1,i} & \mathbf{D}^{a,b}_{k+1,i} \\ \mathbf{D}^{b,a}_{k+1,i} & \mathbf{D}^{b,b}_{k+1,i} \end{bmatrix} \begin{bmatrix} \Delta\mathbf{y}^{a} \\ \Delta\mathbf{y}^{b} \end{bmatrix} = \begin{bmatrix} -\mathbf{r}^{a}_{k+1,i} \\ -\mathbf{r}^{b}_{k+1,i} \end{bmatrix} \tag{3}$$

3. Update solution:

$$\begin{bmatrix} \mathbf{y}^{a}_{k+1,i+1} \\ \mathbf{y}^{b}_{k+1,i+1} \end{bmatrix} = \begin{bmatrix} \mathbf{y}^{a}_{k+1,i} \\ \mathbf{y}^{b}_{k+1,i} \end{bmatrix} + \begin{bmatrix} \Delta\mathbf{y}^{a} \\ \Delta\mathbf{y}^{b} \end{bmatrix}. \tag{4}$$

4. Check convergence:

$$\text{If } \left\| \begin{bmatrix} \mathbf{r}^{a}_{k+1,i} \\ \mathbf{r}^{b}_{k+1,i} \end{bmatrix} \right\| > tol, \text{ increase i and go to Step 2 .} \tag{5}$$

5. Finalize time step:

$$\text{Set } \mathbf{y}^{a}_{k+1} = \mathbf{y}^{a}_{k+1,i+1}, \quad \mathbf{y}^{b}_{k+1} = \mathbf{y}^{b}_{k+1,i+1}. \tag{6}$$

$$\text{If } k < N^{\text{ts}} \text{ increase } k \text{ and go to Step 1 .} \tag{7}$$

In the above steps,

$$\mathbf{c}^{a}_{k,i} = \mathbf{c}^{a}(\mathbf{y}^{b}_{k,i}), \quad \mathbf{r}^{a}_{k,i} = \mathbf{r}^{a}_{k}(\mathbf{y}^{a}_{k,i}, \mathbf{c}^{a}_{k,i}), \quad \mathbf{D}^{a,b}_{k,i} = \left.\frac{\partial \mathbf{r}^{a}_{k}}{\partial \mathbf{y}^{b}_{k}}\right|_{\mathbf{y}^{a}_{k,i}, \mathbf{c}^{a}_{k,i}}, \tag{8}$$

and $\mathbf{c}^{b}_{k,i}$, $\mathbf{r}^{b}_{k,i}$, $\mathbf{D}^{b,a}_{k,i}$, $\mathbf{D}^{a,a}_{k,i}$, and $\mathbf{D}^{b,b}_{k,i}$ are defined accordingly. In particular, the cross derivatives $\mathbf{D}^{a,b}_{k,i}$ and $\mathbf{D}^{b,a}_{k,i}$ need to be determined anew for different coupling scenarios, making the implementation of monolithic solvers challenging as well as problem-dependent. The number of time steps is denoted by N^{ts}.

A staggered solution strategy avoids computations of cross derivatives and can be formulated as a nested iteration. To this end, it is common practice to introduce an operator formulation, where the solution of each subproblem is abbreviated as $\mathbf{c}^{b}_{k+1} = \mathcal{S}^{a}_{k}(\mathbf{c}^{a}_{k+1})$. A classical staggered solution algorithm will then evaluate the operators $\mathcal{S}^{a}_{k}$ and $\mathcal{S}^{b}_{k}$ sequentially to compute better approximations of $\mathbf{c}^{a}_{k+1}$, denoted as $\mathbf{c}^{a}_{k+1,j}$. We refer to such iterations that are controlled by a coupling software as *coupling iterations* with an index *j*. The algorithm can be described by the following steps.

1. Provide a suitable initial guess using a predictor $\mathcal{P}_k$:

$$\mathbf{c}^{a}_{k+1,j=0} = \mathcal{P}_{k}(\mathbf{c}^{a}_{k})$$

2. Call solver *a*:

$$\mathbf{c}^{b}_{k+1,j+1} = \mathcal{S}^{a}_{k}(\mathbf{c}^{a}_{k+1,j})$$

3. Call solver *b*:

$$\tilde{\mathbf{c}}^{a}_{k+1,j+1} = \mathcal{S}^{b}_{k}(\mathbf{c}^{b}_{k+1,j+1})$$

4. Compute residuals:

$$\mathbf{r}^{a}_{k+1,j} = \tilde{\mathbf{c}}^{a}_{k+1,j+1} - \mathbf{c}^{a}_{k+1,j} \tag{9}$$

$$\mathbf{r}^{b}_{k+1,j} = \mathbf{c}^{b}_{k+1,j+1} - \mathbf{c}^{b}_{k+1,j} \tag{10}$$

5. Check convergence:

$$\text{If } \frac{\|\mathbf{r}^{a}_{k+1,j}\|}{\sqrt{m^{a}}} > tol^{\mathbf{c}^{a}} \quad \text{or} \quad \frac{\|\mathbf{r}^{b}_{k+1,j}\|}{\sqrt{m^{b}}} > tol^{\mathbf{c}^{b}} : \text{ go to Step 6.} \tag{11}$$

$$\text{Else: go to step 7.} \tag{12}$$

6. Relax solution / accelerate convergence

$$\mathbf{c}^{a}_{k+1,j+1} = \mathcal{A}_{k,j}(\tilde{\mathbf{c}}^{a}_{k+1,j+1}) \text{ increase j and go to Step 2 .} \tag{13}$$

7. Finalize time step:

$$\text{Set } \mathbf{c}^{a}_{k+1} = \mathbf{c}^{a}_{k+1,j+1}, \quad \mathbf{c}^{b}_{k+1} = \mathbf{c}^{b}_{k+1,j+1}. \tag{14}$$

$$\text{If } k < N^{\text{ts}} \text{ increase } k \text{ and go to Step 1 .} \tag{15}$$

In Eq. (11), m^a and m^b denote the number of components in $\mathbf{r}^{a}_{k+1,j}$ and $\mathbf{r}^{b}_{k+1,j}$, respectively. Of course, the order in which the two subproblems are solved within each coupling iteration can be swapped. Concrete examples for possible prediction operators $\mathcal{P}_k$ and acceleration operators $\mathcal{A}_{k,j}$ are given in the next subsections. At this point, it should be noted that the subscripts of all operators are chosen carefully. While $\mathcal{P}_k$ yields the same results for the same input regardless of the coupling iteration *j* (but depends on the time step *k*), the results of $\mathcal{A}_{k,j}$ depend on both *k* and *j*. Above, we assumed that also the solver operators $\mathcal{S}^{a}_{k}$ ignore the coupling iteration *j* and yield the same results for the same input within a given time step *k*. However, looking into the inner structure of the solvers, it becomes clear that solver operators may be equipped with a coupling iteration index as well, i.e., $\mathcal{S}^{a}_{k,j}$.

The inner structure of a solver typically involves an iterative solution process itself due to the nonlinearity of the subproblems. We denote such iterations as *inner iterations* with an index *i*. To this end, approximations of the state $\mathbf{y}^{a}_{k,j}$ are denoted as $\mathbf{y}^{a,i}_{k,j}$. As mentioned in the introduction, different choices for the initial guess $\mathbf{y}^{a,i=0}_{k,j}$ become available in the scope of a staggered solution algorithm where each time step is solved multiple times. If the initial guess is always taken as the solution of the previous time step, we

write $\mathcal{S}_k^a$ (the operator depends only on the time step) – but if it is taken as the last computed result (possibly from the previous coupling iteration of the same time step), we write $\mathcal{S}_{k,j}^a$. Intuitively, we call $\mathcal{S}_k^a$ a *resetting solver* and $\mathcal{S}_{k,j}^a$ a *non-resetting solver*. From an implementation point of view, the latter does not need to reset its current state, provided that all intermediate results $\boldsymbol{y}_{k,j}^{a,i}$ are stored using the same block of memory. Nevertheless, resetting solvers are much more common, see, e.g., [27, 28, 42 and [16] – and many authors do not even mention which initial guess is used for the solver's inner iterations.

Without loss of generality, we assume here that the Newton-Raphson method is used within the solvers. In this situation, $\mathbf{c}_k^b = \mathcal{S}_k^a(\mathbf{c}_k^a)$ (or $\mathbf{c}_k^b = \mathcal{S}_{k,j}^a(\mathbf{c}_k^a)$) abbreviates the same steps as performed in the monolithic strategy:

1. Choose initial guess:

$$\mathbf{y}_{k+1,j+1}^{a,i=0} = \mathbf{y}_k^a \text{ (for } \mathcal{S}_k^a \text{)} \quad \text{or} \quad \mathbf{y}_{k+1,j+1}^{a,i=0} = \mathbf{y}_{k+1,j}^a \text{ (for } \mathcal{S}_{k,j}^a \text{)} \tag{16}$$

2. Solve linearized system:

$$\left.\frac{\partial \mathbf{r}_{k+1}^a}{\partial \mathbf{y}_{k+1}^a}\right|_{\mathbf{y}_{k+1,j+1}^{a,i},\mathbf{c}_{k+1,\tilde{j}}^a} \Delta\mathbf{y}^{a,i} = -\mathbf{r}^a(\mathbf{y}_{k+1,j+1}^{a,i}, \mathbf{c}_{k+1,\tilde{j}}^a) \tag{17}$$

3. Update solution:

$$\mathbf{y}_{k+1,j+1}^{a,i+1} = \mathbf{y}_{k+1,j+1}^{a,i} + \Delta\mathbf{y}^{a,i} \tag{18}$$

4. Check convergence:

$$\text{If } \|\mathbf{r}^a(\mathbf{y}_{k+1,j+1}^{a,i+1}, \mathbf{c}_{k+1,\tilde{j}}^a)\| > tol^a \text{ increase } i \text{ and go to Step 2.} \tag{19}$$

5. Finalize time step:

Set $\mathbf{y}_{k+1,j+1}^a = \mathbf{y}_{k+1,j+1}^{a,i+1}$,
evaluate $\mathbf{c}_{k+1,j+1}^b(\mathbf{y}_{k+1,j+1}^a)$ and set $N_{k,j}^a = i$.
If global convergence is achieved, (20)
set $\mathbf{y}_{k+1}^a = \mathbf{y}_{k+1,j+1}^a$ and increase k.
If $k < N^{\text{ts}}$ go to Step 1 .

In Step 2, $\tilde{j} = j$ for solver a and $\tilde{j} = j+1$ for solver b. It is noted that throughout this process, the coupling quantities $\mathbf{c}_{k+1}^a$ that depend on the solution of subproblem b remain fixed. The interplay of two resetting or non-resetting solvers inside the staggered coupling algorithm is illustrated in Fig. 1. At each time step k, one or more coupling iterations j take place. Depending on whether we use a resetting or non-resetting solver, the initial value for the iterations within a solver changes. This applies to both solvers (a and b), but in Fig. 1 we focus on the details of solver b. At the beginning of each time step, we use a predictor $\mathcal{P}_k$ to calculate an initial guess of the coupling quantity $\mathbf{c}_{k+1,j=0}^a$. During each coupling iteration, solver a and b receive their respective coupling quantities as inputs, and the subproblem solvers start the solution process. In the first coupling iteration, the initial guess for the subproblem solvers is the solution from the previous time step. If the coupling iteration converges, we proceed to the next time step. Otherwise, we use an accelerator $\mathcal{A}_{k,j}$ and proceed with the next coupling iteration $j+1$. For a resetting solver, the initial guess in the next coupling iteration is again the solution from the previous time step. For a non-resetting solver, the initial guess for the subproblem solver is the last computed result from the previous coupling iteration within the same time step.

2.1 Prediction operator

The predictor $\mathcal{P}_k$ computes an initial guess for the time step k. There exist different types of predictors, and among the

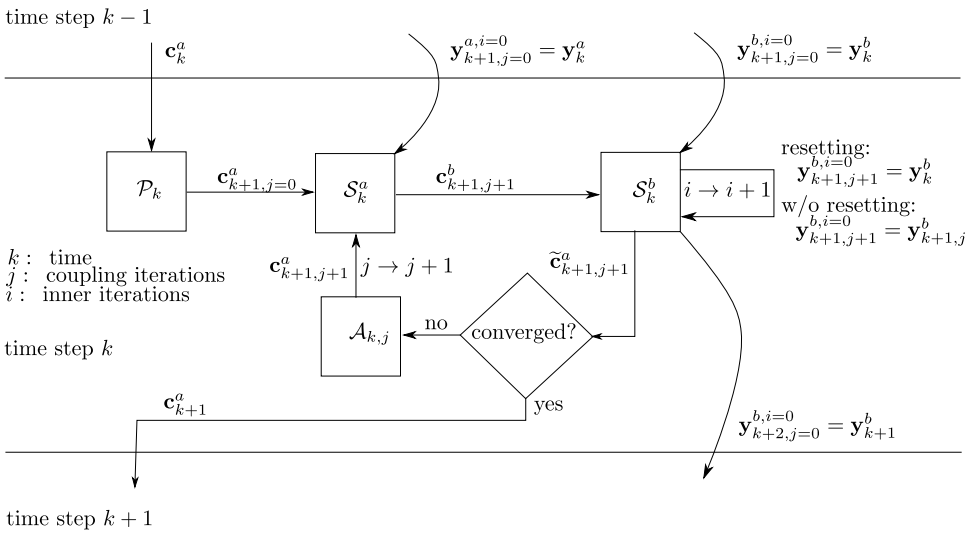


Fig. 1 Staggered coupling algorithm with illustration of resetting details for solver $\mathcal{S}^b$

most popular ones are those based on a polynomial extrapolation. Here, we fit a parabola through the solution of the last three time steps. Then, the tangent to that parabola is computed at t_{k-1} and evaluated at t_{k+1}. Finally, this yields the following formula for the initial guess, which is valid for a constant time step size Δt only:

$$\mathbf{c}^a_{k+1,0} = \frac{1}{2}\mathbf{c}^a_{k-2} - 2\mathbf{c}^a_{k-1} + \frac{5}{2}\mathbf{c}^a_k \tag{21}$$

This type of predictor was also successfully used in [10] to improve the efficiency of partitioned FSI simulations.

2.2 Acceleration operator

The accelerator $\mathcal{A}_{k,j}$ provides an improved coupling quantity for the next coupling iteration. Commonly used methods include variants of the Aitken relaxation [1, 23, 31], Broyden's method [5], and other quasi-Newton methods. The quasi-Newton least squares (QNLS) method introduced in [49] and further developed in the scope of FSI simulation in [10, 11, 20, 40] is used in this work. It is based on an approximation of the classical Newton method by a minimization of the residual in a least-squares sense. Unlike the mentioned relaxation method, this yields an update of the coupling quantity $\mathbf{c}^a$ in a direction $\Delta\mathbf{c}^a_{k+1,j}$ that is not parallel to the residual, but to an approximation of the Newton-Raphson increment, i.e., the product of the inverse of the tangent matrix $\mathbf{D}^{b,a}_{k+1,i}$ and the residual $\mathbf{r}^a_{k+1,j}$:

$$\mathbf{c}^a_{k+1,j+1} = \mathbf{c}^a_{k+1,j} + \Delta\mathbf{c}^a_{k+1,j} \approx \mathbf{c}^a_{k+1,j} - \left(\mathbf{D}^{b,a}_{k+1,i}\right)^{-1}\mathbf{r}^a_{k+1,j}. \tag{22}$$

2.3 Implementation aspects

Generally, a solver for a subproblem needs to be manipulated in order to participate in a coupled simulation. In our coupling manager *comana*, see [27], we try to do this in a minimally invasive way, as illustrated in Algorithms 1 and 2. Basically, only an implicit coupling loop has to be introduced once the time loop has been identified. At the beginning and end, a general communication routine is called in which data may be sent to and/or received from the coupling manager. While only one call to the communication routine is enough to realize the above coupling strategy, we found it convenient to use two calls. Considering solver *a*, in the first call of the `communicate()` routine, we typically

- Set the coupling quantities $\mathbf{c}^a$ as boundary conditions,
- Set other parameters, e.g., the time step size.

In the second call, we typically

- Request (part of) the solution as coupling quantities $\mathbf{c}^b$,
- Set other parameters, e.g., the time step size for the next time step
- Provide the solver with information on the global convergence state.

Of course, the communication routine needs access to the data structures within the solver to achieve this. To this end, in the initialization routine, an interface is established that expects a reference to the solver data denoted as `problem`.

Due to the fact that interfaces designed in such a way only allow input and output exchange – without involving quantities connected to the inner quantities of a solver (e.g., the tangent stiffness matrix) – they are said to be of a *black-box* nature. We would like to emphasize that enabling adaptive tolerances within a solver, as introduced in the next section, requires only little implementation effort and preserves the black-box nature. Basically, only one more scalar value must be included as a parameter, similar to the time step size, which may be sent and received when working with adaptive time-marching schemes. Our experience

```
1:
2:  for k = 1, ..., N^ts do                    ▷ time loop
3:
4:
5:      solveProblem()
6:
7:
8:
9:
10:
11:     updateState()
12: end for
```

Algorithm 1 Classical solver.

```
1:  initialize(problem)
2:  for k = 1, ..., N^ts do                    ▷ time loop
3:      while true do                          ▷ coupling loop
4:          communicate()
5:          solveProblem()
6:          communicate()
7:          if isGloballyConverged() then
8:              break
9:          end if
10:     end while
11:     updateState()
12: end for
```

Algorithm 2 Coupled solver.

when connecting various solvers to our coupling manager *comana* is that it is generally much harder to achieve a separation of the solution step and the state update step (e.g., into routines `solveProblem()` and `updateState()`) than adding the possibility to send and receive additional parameters. Accordingly, an implementation of the adaptive tolerance proposed here within another coupling software, e.g., *preCICE* [6], should not pose any problems. Further, a combination of the strategy with algorithms beyond the classical staggered one, e.g., the parallel coupling algorithms investigated in [34], is directly possible.

3 Adaptive tolerances

Looking at the inner iterations in Eqs. (16) – (20), it is clear that not only the initial guess but also a convergence tolerance is needed in order to fully specify the solver-internal solution algorithm. The central idea of this work builds on the possibility to change this tolerance from one coupling iteration to the next. We therefore introduce a more explicit notation for the solution operators

$$\begin{aligned} \mathbf{c}^b_{k+1} &= \mathcal{S}^a_k(\mathbf{c}^a_{k+1}, tol^a) \quad \text{and} \\ \mathbf{c}^b_{k+1} &= \mathcal{S}^a_{k,j}(\mathbf{c}^a_{k+1}, tol^a) \end{aligned} \tag{23}$$

to distinguish solvers with and without resetting, respectively. The idea to set the tolerances in an adaptive way is inspired by the observation that only the results of the last coupling iteration are actually kept, while all results obtained in previous coupling iterations of that time step are dumped. To this end, it is promising to compute such preliminary results with a lower accuracy than desired for the final solution.

3.1 A simple rule and a simple problem

A simple rule following the above idea is to set the tolerance to a large value $tol_{\max}$ at the beginning of each time step and set it to the actually desired (small) tolerance $tol_{\min}$ after a certain number of coupling iterations N^{fine}. To make sure that the coupling iteration converges to the correct solution, the small tolerance is enforced directly if the coupling iterations converge before N^{fine} is reached and at least one more iteration is performed based on $tol_{\min}$. A simplified staggered solution procedure with fixed tolerances and resetting solvers is given in Algorithm 3. For comparison, a variant of the same algorithm with adaptive tolerances and non-resetting solvers is given in Algorithm 4. Since the tolerance is switched only once in this simple problem, we call it here *switched tolerance*. The algorithms include only the solution of a single time step and do not make use of convergence acceleration methods, i.e. $\mathbf{c}^a_{k+1,j+1} = \tilde{\mathbf{c}}^a_{k+1,j+1}$ here. Therefore, we do not consider the interaction between the solver tolerances and the coupling accelerator for this case. It should be noted that, as shown in [46], an inaccurate solution to the subproblems could negatively impact the QNLS accelerator, given that the input data is less precise. We

1: $\mathbf{c}^a_{k+1,0} = \mathbf{0}$
2: $tol^a = tol^a_{\min}$
3: $tol^b = tol^b_{\min}$
4: $j = 0$
5: **while** `true` **do**
6: $\quad \mathbf{c}^b_{k+1,j+1} = \mathcal{S}^a_k(\mathbf{c}^a_{k+1,j})$
7: $\quad \mathbf{c}^a_{k+1,j+1} = \mathcal{S}^b_k(\mathbf{c}^b_{k+1,j+1})$
8: $\quad \mathbf{r}^a = \mathbf{c}^a_{k+1,j+1} - \mathbf{c}^a_{k+1,j}$
9: $\quad \mathbf{r}^b = \mathbf{c}^b_{k+1,j+1} - \mathbf{c}^b_{k+1,j}$
10: $\quad$ **if** `isConverged(`$\mathbf{r}^a$`,` $\mathbf{r}^b$`)` **then**
11:
12: $\quad\quad$ **break**
13:
14: $\quad$ **end if**
15:
16:
17:
18:
19: $\quad j = j + 1$
20: **end while**

Algorithm 3 Fixed tolerances.

1: $\mathbf{c}^a_{k+1,0} = \mathbf{0}$
2: $tol^a = tol^a_{\max}$
3: $tol^b = tol^b_{\max}$
4: $j = 0$
5: **while** `true` **do**
6: $\quad \mathbf{c}^b_{k+1,j+1} = \mathcal{S}^a_{k,j}(\mathbf{c}^a_{k+1,j})$
7: $\quad \mathbf{c}^a_{k+1,j+1} = \mathcal{S}^b_{k,j}(\mathbf{c}^b_{k+1,j+1})$
8: $\quad \mathbf{r}^a = \mathbf{c}^a_{k+1,j+1} - \mathbf{c}^a_{k+1,j}$
9: $\quad \mathbf{r}^b = \mathbf{c}^b_{k+1,j+1} - \mathbf{c}^b_{k+1,j}$
10: $\quad$ **if** `isConverged(`$\mathbf{r}^a$`,` $\mathbf{r}^b$`)` **then**
11: $\quad\quad$ **if** $tol^a = tol^a_{\min}$ **and** $tol^b = tol^b_{\min}$ **then**
12: $\quad\quad\quad$ **break**
13: $\quad\quad$ **end if**
14: $\quad$ **end if**
15: $\quad$ **if** $j > N^{\text{fine}}$ **or** `isConverged(`$\mathbf{r}^a$`,` $\mathbf{r}^b$`)` **then**
16: $\quad\quad tol^a = tol^a_{\min}$
17: $\quad\quad tol^b = tol^b_{\min}$
18: $\quad$ **end if**
19: $\quad j = j + 1$
20: **end while**

Algorithm 4 Switched tolerances.

investigated the effects of a simple finite-difference-based quasi-Newton accelerator, however, this did not yield new insights. The QNLS accelerator is used in the subsequent FSI simulations.

In order to illustrate the potential of adaptive tolerances, we consider a simple test problem

$$\begin{cases} r^a = (y^a)^3 + y^a - 2\,(c^a)^2 + 3\,y^a\,c^a - 10 & = 0, \\ r^b = (y^b)^4 + (y^b)^2 - 2\,(c^b)^2 + 3\,y^b\,c^b + y^b - 10 & = 0. \end{cases} \tag{24}$$

The coupling quantities are simply set to

$$c^a = y^b \quad \text{and} \quad c^b = y^a. \tag{25}$$

Due to the time-independent nature of the subproblems, the coupling loop is executed only once to solve the coupled problem. Figure 2a and b illustrate the solution as the intersection of the zero level-sets (red and blue line) of each of the two residuals.

In the following, we investigate the effect of resetting as well as that of adaptive tolerances according to the explained rule. For all computations, we set $tol^a_{\min} = tol^b_{\min} = 10^{-10}$ and the coupling tolerance $tol^a_{\text{cpl}} = tol^b_{\text{cpl}} = 10^{-10}$. The values of $tol^a_{\max} = tol^b_{\max} = tol_{\max}$ and N^{fine} are varied. As a measure for the computation time, the total number of inner solver iterations

$$N^{\text{total}} = \sum_{j=1}^{N_k} N^a_{0,j} + N^b_{0,j}. \tag{26}$$

is determined. In Appendix B, we show that this is a reasonable measure for the overall computational time. The speed-up of computational time

$$\eta = \left(1 - \frac{N^{\text{total}}}{N^{\text{total}}_{\text{ref}}}\right) \cdot 100\% \tag{27}$$

is defined via the total number of inner iterations determined for the (classical) solver with resetting and fixed tolerance $N^{\text{total}}_{\text{ref}}$ and that corresponding to the investigated solution variant N^{total}.

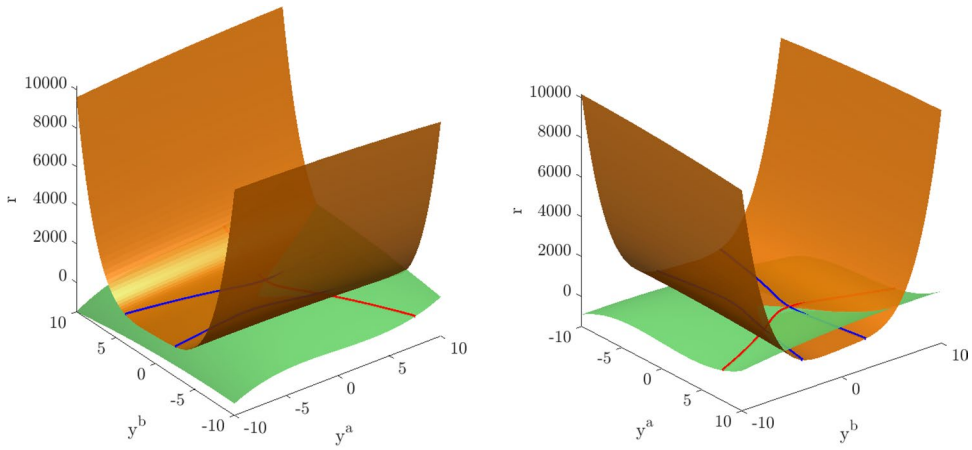
3.1.1 Results for $\mathcal{S}^a_k$ / $\mathcal{S}^b_k$ (solvers with resetting)

To begin with, we consider the results obtained for resetting solvers. The parameters were set to $tol_{\max} = 10^{-3}$ and $N^{\text{fine}} = 1$. In Fig. 3a and b, the inner solver residuals and the coupling residual are plotted against the total number of inner solver iterations. The vertical lines indicate the start of a coupling iteration. During the staggered solution process, one solver waits until the other has converged before starting its own solution process. During the waiting period, the residual remains unchanged. Comparing the result for fixed tolerances in Fig. 3a with the one for adaptive tolerances in Fig. 3b, one can see that the adaptive tolerance lead to a lower number of inner solver iterations in total. While 94 inner solver iterations are performed to reach the final solution with fixed tolerances, only 86 are needed when using adaptive tolerances. For this problem, the number of coupling iterations remains 7, regardless of the resetting.

In this example, the savings from switching to an adaptive tolerance scheme are not that high in terms of the total number of inner solver iterations. Therefore, in the next step, the solvers without resetting are investigated.

3.1.2 Results for $\mathcal{S}^a_{k,j}$ / $\mathcal{S}^b_{k,j}$ (solvers without resetting)

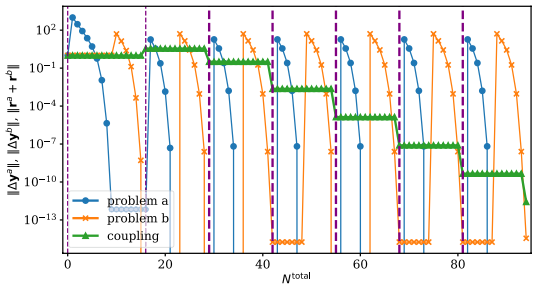
Solving the test problem using solvers without resetting results in less than half of the total number of inner solver iterations, compared to the previous results, as depicted in Fig. 4a. The problem is solved using only 36 inner solver iterations in total. Again, the number of coupling iterations is 7.



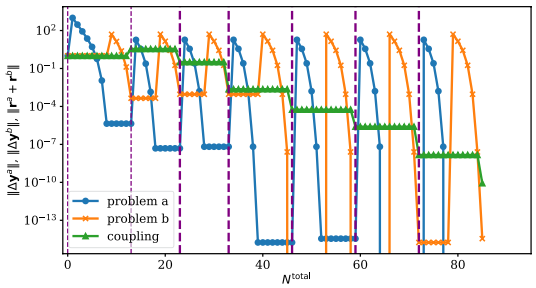
(a) First view of the surface plots.

(b) Second view of the surface plots.

Fig. 2 Illustration of the test problem from Eq. (24)

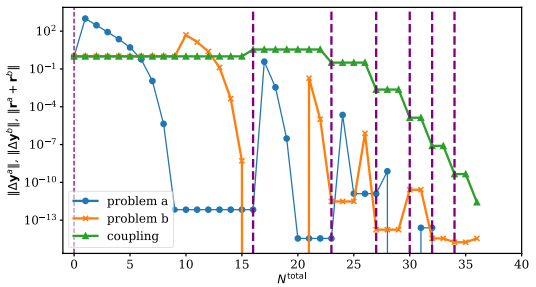


(a) Solver *with resetting* and fixed tolerances.

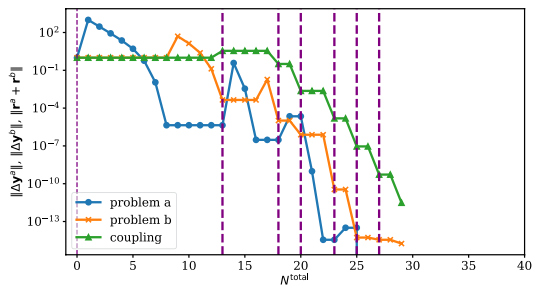


(b) Solver *with resetting* and *adaptive tolerances*

Fig. 3 Convergence of the residuals when solving the problem in Eq. (24) using different solver setups according to Algorithm 4



(a) Solver *without resetting* and *fixed tolerances*.



(b) Solver *without resetting* and *adaptive tolerances*.

Fig. 4 Convergence of the residuals when solving the problem in Eq. (24) using different solver setups according to Algorithm 4

Adding to the solver without resetting the adaptive tolerance rule, even less total inner iterations are required to solve the test problem. With the chosen set of parameters, the total number of inner iterations reduces to 29, as can be seen in Fig. 4b.

As described in Algorithm 4, the small tolerance $tol_{\min}$ is used whenever the coupling iterations convergence before N^{fine} is reached to ensure an accurate solution. For all four solver setups, the number of coupling iterations remains the same. The required computational time varies proportionally to the number of inner solver iterations. Comparing the standard case (fixed tolerances and solvers with resetting) with the advanced case (adaptive tolerances and solvers without resetting), a speed-up of nearly 70% can be observed.

3.1.3 Parameter study

Up to now, the simple test problem was solved with one set of parameters for $tol_{\max}$ and N^{fine}. In the next step, a parameter study is carried out to investigate the effects of different settings. The parameter $tol_{\max}$ is varied from 10^{-10} to 10^{8} and N^{fine} from 1 to 10. This variation of parameters is applied to both solver variants, i.e., with and without resetting.

We begin with the results for the solver with resetting. In Fig. 5, the total number of inner iterations is plotted against $tol_{\max}$ for different values of N^{fine}. Depending on the chosen set of parameters, the solver with adaptive tolerances requires more or less inner solver iterations.

In the extreme case with $tol_{\max} = 10^{-2}$ and $N^{\text{fine}} = 10$, the solver needs 40% more iterations compared to the case with a fixed tolerance set to $tol_{\min}$. In the best case, the solver needs approximately 11% less total inner solver iterations. One can also observe that the total number of inner solver iterations becomes independent of N^{fine} for $tol_{\max} \geq 10^{4}$.

Investigating the results for the solver without resetting in Fig. 6, it is seen that a larger set of parameters leads to a reduction in the total number of inner solver iterations. In the best case, a speed-up of approximately 75% is achieved compared to the solver with resetting and fixed tolerances. This is a speed-up of 36% compared to the solver without resetting and fixed tolerances. In the worst case, the solver without resetting still results in a speed-up compared to the best result of the solver with resetting.

The results of both parameter studies are summarized in Table 1.

The results shown for the parameter study of the simple test problem lead to the conclusion that using a solver without resetting in combination with adaptive tolerances yields the best performance gain from among the investigated setups. Both can lead to a speed-up in terms of the total number of inner solver iterations, depending on the chosen rule.

3.2 Advanced rules

In the above example, the tolerances were adjusted based on the coupling iteration index j. Since the typical number of coupling iterations required is problem-dependent, more general rules are formulated.

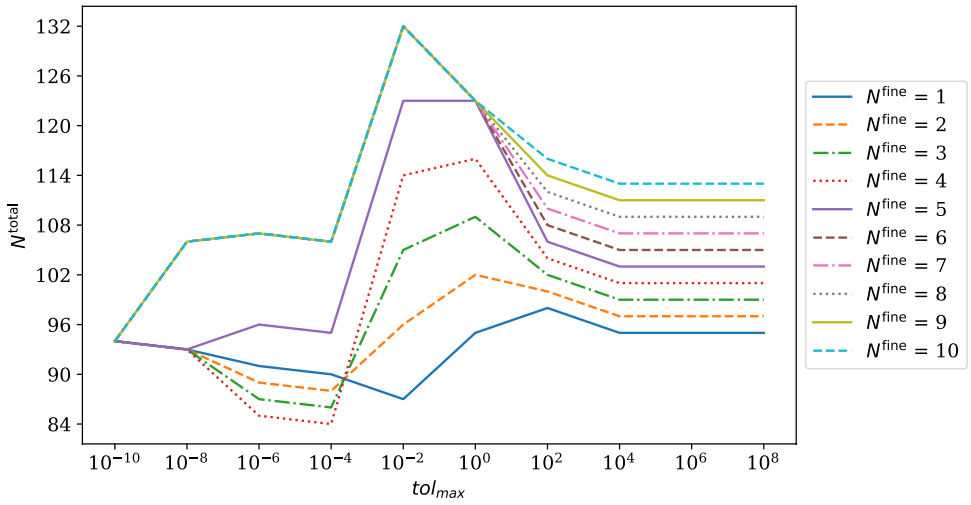


Fig. 5 Total number of inner solver iterations for different $tol_{\max}$ and N^{fine} and solver with resetting

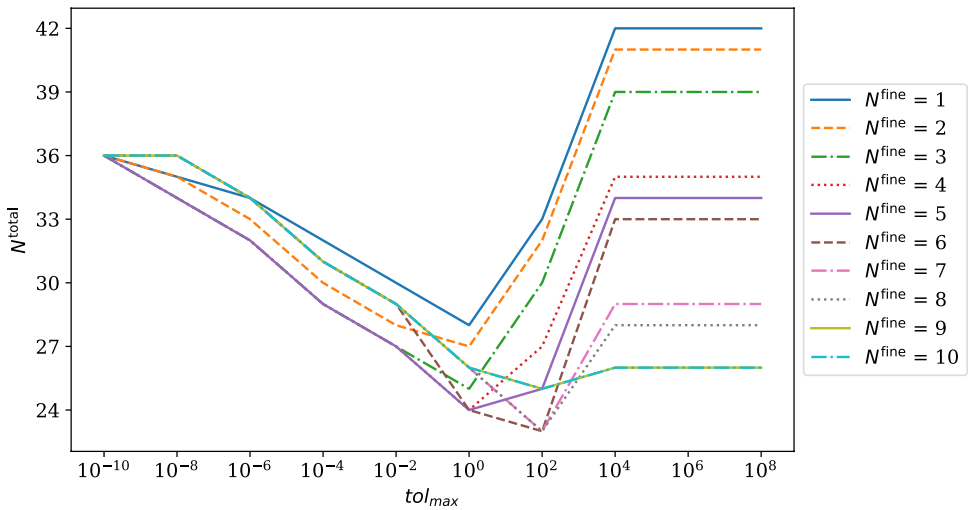


Fig. 6 Total number of inner solver iterations for different $tol_{\max}$ and N^{fine} and solver without resetting

Table 1 Results for best parameter settings

simulation parameters rule	results inner iter.	
	η in %	N^{total}
Resetting fixed	0	94
Resetting adaptive	10.6	84
w/o resetting fixed	61.7	36
w/o resetting adaptive	75.5	23

Like the simple rule introduced before, the advanced rules are based on a lower and an upper limit for the tolerances $tol^a_{\min}$ and $tol^a_{\max}$, respectively. It is emphasized that, for any rule, the final coupling iteration is always performed using the low tolerance $tol^a_{\min}$ for both solvers, as in Lines 6 – 8 in Algorithm 4.

3.2.1 Rule A

For this rule, we set the inner solver tolerance according to Eq. (28), where α is a parameter that must be provided by the user. This means that in every coupling iteration j, the tolerance decreases. The rule reads

$$tol^{a,\mathrm{A}} = \max(tol^a_{\max} \cdot \alpha^{-j}, tol^a_{\min}). \tag{28}$$

Using a basis $\alpha > 1$, it always holds that $\alpha^{-j} < 1$ and, thus, $tol^{a,\mathrm{A}} < tol^a_{\max}$.

If an estimation of the number of coupling iterations per time step N is available – e.g., from previous experience – the basis α can be computed with

$$\alpha = \left(\frac{tol_{\max}}{tol_{\min}}\right)^{\frac{1}{N}}. \tag{29}$$

3.2.2 Rule B

The next rule uses the coupling residual $\mathbf{r}^a_{k+1,j+1}$ for the computation of the inner solver tolerance multiplied with a factor F with

$$tol^{a,\mathrm{B}} = \max(\min(F\,\|\mathbf{r}^a_{k+1,j+1}\|, tol^a_{\max}), tol^a_{\min}). \tag{30}$$

Of course, only a factor $F > 0$ makes sense in this rule. Furthermore, $F < 1$ is chosen here, as it is questionable to demand a lower accuracy within the solvers than within the coupling iterations. However, this depends on the actual residual that is compared to the tolerance, as elaborated in Sec. 4.2 for the individual solvers for the considered FSI problem.

3.2.3 Rule C

The last rule has the same structure as Rule B, but uses the low tolerance in the first coupling iterations. This results in

$$tol^{a,\mathrm{C}} = \begin{cases} tol^a_{\min} & \text{if } j = 1, \\ tol^{a,\mathrm{B}} & \text{else}\ . \end{cases} \tag{31}$$

Instabilities are often observed within the first few coupling iterations. The idea behind this rule is to prevent an enlargement of these instabilities which may occur due to less accurate subproblem solutions.

4 Fluid-structure interaction

In this section, we briefly recap the formulation of fluid-structure interaction problems. For a detailed introduction into the topic, we refer to standard textbooks [3, 8, 9] and articles with extensive theory, e.g., [2, 27, 42].

4.1 Modeling

We consider the interaction between a hyperelastic solid in the domain Ω^{s}_t and a viscous fluid in the domain Ω^{f}_t.

The solid is described by the balance of linear momentum in a total Lagrangian formulation

$$\rho^{\mathrm{s}}_0\,\ddot{\boldsymbol{d}} - \mathrm{Div}(\boldsymbol{F}\,\boldsymbol{S}) = \rho^{\mathrm{s}}_0\,\boldsymbol{b}^{\mathrm{s}} \quad \text{in } \Omega^{\mathrm{s}}_0, \tag{32}$$

where ρ^{s}_0 is the initial density, $\ddot{\boldsymbol{d}}$ is the acceleration, i.e., the second temporal derivative of the displacement $\boldsymbol{d}$. The deformation gradient, the second Piola-Kirchhoff stress, and the body load are denoted as $\boldsymbol{F}$, $\boldsymbol{S}$, and $\boldsymbol{b}^{\mathrm{S}}$, respectively. With $\mathrm{Div}(\cdot)$, we refer to the divergence with respect to the reference configuration Ω_0. Equation (32) is equipped with boundary conditions of Dirichlet type

$$\boldsymbol{d} = \bar{\boldsymbol{d}} \quad \text{on } \Gamma^{\mathrm{s},\,\mathrm{D}}_0 \tag{33}$$

and of Neumann type

$$\boldsymbol{F}\,\boldsymbol{S}\,\boldsymbol{N}^{\mathrm{s}} = \bar{\boldsymbol{t}} \quad \text{on } \Gamma^{\mathrm{s},\,\mathrm{N}}_0, \tag{34}$$

where $\bar{\boldsymbol{d}}$ and $\bar{\boldsymbol{t}}$ denote a given displacement and a given traction, respectively, and $\boldsymbol{N}^{\mathrm{s}}$ is the unit outward normal of $\Gamma^{\mathrm{s},\,\mathrm{N}}_0$. Generally, we assume that $\partial\Omega^{\mathrm{s}} = \Gamma^{\mathrm{s,N}} \cup \Gamma^{\mathrm{s,D}}$ and $\Gamma^{\mathrm{s,N}} \cap \Gamma^{\mathrm{s,D}} = \emptyset$. In the scope of FSI simulations, the FSI interface $\Gamma^{\mathrm{FSI}} \subset \Gamma^{s,N}$. Throughout this paper, homogeneous initial conditions are assumed.

The fluid is described by the incompressible Navier-Stokes equations in an arbitrary Lagrangian-Eulerian formulation

$$\dot{\boldsymbol{v}} + \mathrm{grad}(\boldsymbol{v})\,(\boldsymbol{v} - \hat{\boldsymbol{v}}) - \frac{1}{\rho^{\mathrm{f}}}\,\mathrm{div}(\boldsymbol{\sigma}) = \boldsymbol{0}, \tag{35}$$

$$\mathrm{div}(\boldsymbol{v}) = 0, \tag{36}$$

where $\boldsymbol{v}$ denotes the fluid velocity, $\hat{\boldsymbol{v}} = \dot{\hat{\boldsymbol{d}}}$ denotes the domain velocity, i.e., the temporal derivative of the domain displacement $\hat{\boldsymbol{d}}$. The fluid density is denoted as ρ^{f}, and $\mathrm{grad}(\cdot)$ and $\mathrm{div}(\cdot)$ refer to the gradient and divergence with respect to the coordinates of the current configuration Ω^{f}_t. The Cauchy stress is computed as

$$\boldsymbol{\sigma} = \rho^{\mathrm{f}}\,p\,\boldsymbol{1} + \eta\,\dot{\boldsymbol{\varepsilon}}, \tag{37}$$

where $\dot{\boldsymbol{\varepsilon}} = \mathrm{symgrad}(\boldsymbol{v})$ denotes the strain-rate tensor, i.e., the symmetric gradient of the velocity, and η denotes the dynamic viscosity of the fluid. Eqs. (35) and (36) are equipped with velocity boundary conditions

$$\boldsymbol{v} = \bar{\boldsymbol{v}} \quad \text{and} \quad \mathrm{grad}(p)\cdot\boldsymbol{n}^{\mathrm{f}} = 0 \quad \text{and} \quad \hat{\boldsymbol{d}} = \bar{\hat{\boldsymbol{d}}} \quad \text{on } \Gamma^{\mathrm{f},\boldsymbol{v}}_t \tag{38}$$

and pressure boundary conditions

$$p = \bar{p} \quad \text{and} \quad \mathrm{grad}(\boldsymbol{v})\,\boldsymbol{n}^{\mathrm{f}} = \boldsymbol{0} \quad \text{and} \quad \hat{\boldsymbol{d}} = \bar{\hat{\boldsymbol{d}}} \quad \text{on } \Gamma^{\mathrm{f},p}_t, \tag{39}$$

where $\boldsymbol{n}^{\mathrm{f}}$ is the unit outward normal of $\Gamma^{\mathrm{f},\boldsymbol{v}}_t$ and $\Gamma^{\mathrm{f},p}_t$. Generally, we assume that $\partial\Omega^{\mathrm{f}} = \Gamma^{\mathrm{f},\boldsymbol{v}} \cup \Gamma^{\mathrm{f},p}$ and $\Gamma^{\mathrm{f},\boldsymbol{v}} \cap \Gamma^{\mathrm{f},p} = \emptyset$. Additionally, the domain displacement $\hat{\boldsymbol{d}}$ is prescribed everywhere on the boundary of Ω^{f}_0. It is extended into the domain by solving the Laplace equation. Alternative methods to achieve this extension are given in [39]. In the scope of FSI simulations, the FSI interface $\Gamma^{\mathrm{FSI}} \subset \Gamma^{\mathrm{f},\boldsymbol{v}}$. The coupling conditions on the interface can be stated as

$$\bar{\hat{\boldsymbol{d}}} = \boldsymbol{d} \quad \text{on } \Gamma^{\mathrm{FSI}}_0, \tag{40}$$

$$\bar{\boldsymbol{v}} = \hat{\boldsymbol{v}} = \dot{\boldsymbol{d}} \quad \text{on } \Gamma^{\mathrm{FSI}}_t, \tag{41}$$

$$\bar{\boldsymbol{t}} = -\boldsymbol{\sigma}\boldsymbol{n}^{\mathrm{f}} \quad \text{on } \Gamma^{\mathrm{FSI}}_0. \tag{42}$$

4.2 Discretization

We discretize the structure subproblem using the high-order finite elements as implemented in the in-house software *AdhoC++* [51]. To this end, we make use of hierarchic shape functions based on integrated Legendre polynomials, as explained in [47]. A weak form is constructed from (32) and then discretized by hexahedral elements. The element geometry is described using the quasi-regional mapping approach [15, 25]. The discretization finally leads to the nonlinear semi-discrete equation of motion

$$\mathbf{M}\,\ddot{\mathbf{d}} + \mathbf{C}\,\dot{\mathbf{d}} + \mathbf{i}(\mathbf{d}) = \mathbf{e}, \tag{43}$$

where $\mathbf{M}$ is the mass matrix, $\mathbf{C} = a\,\mathbf{M}$ is the mass proportional damping matrix, $\mathbf{i}$ is the internal load vector, $\mathbf{e}$ is the external load vector, and $\mathbf{d}$ is the degree of freedom vector. Equation (43) is discretized in time using the Newmark method [36]. This yields a nonlinear system of equations that is solved in every time step k using the Newton-Raphson method. Based on the effective stiffness matrix $\mathbf{K}_k^{\mathrm{eff},i}$ and the effective residual vector $\mathbf{r}_k^{\mathrm{eff,i}}$, the linear system to be solved in each Newton-Raphson iteration i reads

$$\mathbf{K}_k^{\mathrm{eff},i}\Delta\mathbf{d}_k^i = -\mathbf{r}_k^{\mathrm{eff,i}}. \tag{44}$$

Afterwards, the solution is updated by $\mathbf{d}_{k+1}^{i+1} = \mathbf{d}_{k+1}^i + \Delta\mathbf{d}_k^i$. The iteration is terminated once the convergence criterion

$$\|\Delta\mathbf{d}_k^i\| < tol^{\mathcal{S}} \tag{45}$$

is fulfilled. We note that this is not the only possible convergence criterion and may not be the first choice for a general application. Here, it is chosen in order to allow for a more direct relation of the prescribed tolerance and the resulting accuracy of the computed displacement. Further, for the less involved benchmark problem, we investigate the effect of the prescribed tolerance on the accuracy of the displacement through a comparison with a reference solution. Following [24], the increment might be a reasonable convergence criterion of the Newton-Raphson solver, as long as the tangent stiffness matrix is not ill-conditioned and the initial iterate is not far from the solution.

The fluid subproblem is discretized using the finite volume method employed by *FreSCo+* [43]. The solution process follows a pressure correction scheme based on the SIMPLE algorithm [37]. The segregated algorithm uses a cell-centered, collocated storage arrangement for all transport quantities. The temporal term is discretized by a first-order accurate in time backward Euler method. The spatial discretization is second-order accurate and employs the QUICK scheme [29] for the discretization of the convective term. The domain velocity $\hat{v}$ is computed based on the swept-volume approach for the treatment of the space conservation law [14], in accordance with the temporal discretization scheme. The employed algorithm is highly parallelizable using a domain decomposition approach based on a Single Program Multiple Data (SPMD) message-passing model, i.e., each process runs the same program on its own subset of data. The MPI protocol is employed for data communications within the fluid subproblem.

Upon discretization of the momentum and pressure-correction equations, a linearized algebraic equation system

$$\mathbf{A}_k^{\phi}\boldsymbol{\phi}_k = \mathbf{B}_k^{\phi} \tag{46}$$

arises and is solved by an iterative process for each time step. The iterations of the algebraic equation systems are of less importance in the context of this work, and focus is solely placed on the iterations of the pressure-velocity coupling, which herein refer to the inner iterations of the fluid solver. Consider i to be the index denoting the inner iterations of the fluid. Then, the residual of each inner iteration can be computed as

$$\mathbf{r}_k^{\phi,i} = \mathbf{A}_k^{\phi,i}\boldsymbol{\phi}_k^{i-1} - \mathbf{B}_k^{\phi,i}, \tag{47}$$

where $\boldsymbol{\phi}_k^{i-1}$ denotes a fluid velocity component or pressure correction and $\mathbf{r}_k^{\phi,i} \in \mathbb{R}^{N_{\mathrm{CV}}}$, with N_{CV} denoting the number of control volumes used to discretize the fluid domain. To evaluate the convergence of the pressure-velocity iterative coupling process, the L^1-norm of (47) is computed and normalized for each discretized equation. The time step is considered converged when

$$\max\left(\frac{\left\|\mathbf{r}_k^{v_x,i}\right\|_1}{\left\|\mathbf{A}_k^{v_x,i}\mathbf{v}_{\mathrm{x}_k}^{i-1}\right\|_1}, \frac{\left\|\mathbf{r}_k^{v_y,i}\right\|_1}{\left\|\mathbf{A}_k^{v_y,i}\mathbf{v}_{\mathrm{y}_k}^{i-1}\right\|_1}, \frac{\left\|\mathbf{r}_k^{v_z,i}\right\|_1}{\left\|\mathbf{A}_k^{v_z,i}\mathbf{v}_{\mathrm{z}_k}^{i-1}\right\|_1}, \frac{1}{N_{\mathrm{CV}}}\left\|\mathbf{r}_k^{p,i}\right\|_1\right) \leq tol^{\mathcal{F}} \tag{48}$$

where the first three components correspond to the normalized velocity residuals (for a 3D simulation) and the last one to the normalized pressure correction. Note that the normalization followed for the velocity components results in a dimensionless metric, while the pressure correction is only normalized by the number of cells due to the lack of appropriate initialization schemes explicitly for pressure correction.

5 Numerical investigation

In this section, we apply the rules introduced above in the scope of FSI simulations. First, we consider a well-known two-dimensional benchmark problem. Afterwards, a three-dimensional FSI problem is solved to show the applicability of the proposed ideas in the scope of more complex applications.

5.1 Lid-driven cavity flow

The lid-driven cavity flow is a well-known benchmark problem from fluid dynamics, see, e.g., [17, 19]. In [35], it was augmented by a flexible bottom and has served as a benchmark problem, e.g., in [48]. We also used it previously to test several algorithms within our coupling software *comana*, see, e.g., [27, 38]. This problem was also investigated in [46], in combination with limiting the number of the inner solver iterations.

Figure 7 illustrates the underlying boundary value problem of the lid-driven cavity flow, with a flexible bottom indicated by the dotted line. At the inlet (upper part of the left boundary) and at the top boundary, a time-dependent velocity boundary condition is prescribed. The maximum value in x-direction is

$$\bar{v} = \left(1 - \cos\left(\frac{2\,\pi\,t}{5\,\mathrm{s}}\right)\right)\ \mathrm{m/s}. \tag{49}$$

The simulation time is $30.0\,\mathrm{s}$. At the outlet (upper part of the right boundary), the pressure is set to $\bar{p} = 0\,\mathrm{Pa}$. The fluid has a density of $1\ \frac{\mathrm{kg}}{\mathrm{m}^3}$ and a dynamic viscosity of $3 \cdot 10^{-2}\,\mathrm{Pa\,s}$.

The structure is clamped on the left and right end and has a thickness in y-direction of $2 \cdot 10^{-3}\,\mathrm{m}$. A Saint Venant-Kirchhoff material model is chosen with density $500\ \frac{\mathrm{kg}}{\mathrm{m}^3}$, Young's modulus $250\frac{\mathrm{N}}{\mathrm{m}^2}$, and Poisson's ratio of 0. For the following coupled problem and throughout the rest of this work, the QNLS method uses only data from the current time step and applies filtering to manage linear dependencies between the data [20]. We use a relative filtering tolerance of $10e^{-9}$ and an absolute filtering tolerance of $10e^{2}$.

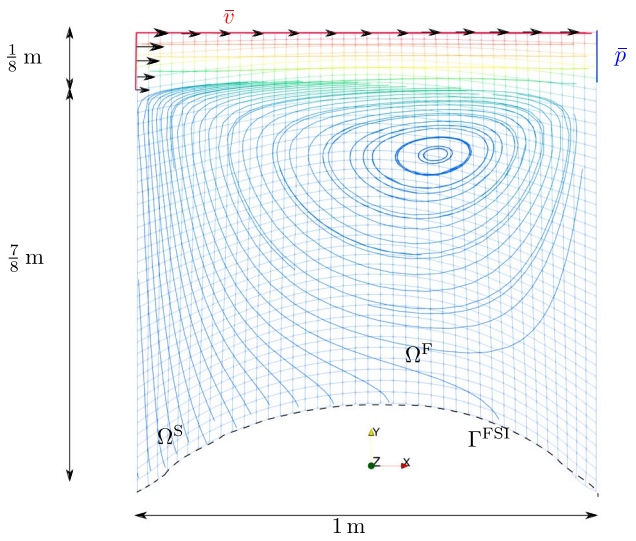


Fig. 7 Illustration of the lid-driven cavity flow

5.1.1 Discretization

We would like to note that the published reference solutions are discretization dependent and do not correspond to the converged solution of the underlying problem, see [35, 48]. Accordingly, deviations are expected when employing different discretization schemes in space and/or time. For our purposes in this work, it is nevertheless well suited. In order to test the efficiency of the rules to set tolerances adaptively, we simply require that the results obtained with a given discretization remain the same (up to the prescribed tolerances). Whether or not the discretized system actually describes the underlying physics reasonably well is only of secondary interest. To this end, we employ the rather coarse spatial discretization from [48] combined with a larger time step size of $\Delta t = 2 \cdot 10^{-2}\,\mathrm{s}$.

The fluid mesh consists of 40 by 40 finite volumes in $x-$ and $y-$ direction, respectively. The structural mesh consists of four by two elements in the $x-$ and $y-$ direction, with an ansatz order of $p = 6$ in the longitudinal direction and $p = 4$ in thickness direction. The Newmark method is used for time integration with $\beta = 0.25$ and $\gamma = 0.5$.

In both solvers, three-dimensional meshes are used with a single finite element / finite volume in the z-direction. Accordingly, both meshes contain hexahedral elements only. For a minimal computational overhead, we chose an order of $p = 1$ for the finite elements in this direction.

5.1.2 Results

To begin with, we compute the problem using very low tolerances (see Tab. 2, first line) to determine a reference solution. The displacement of the center point at the bottom is shown in Fig. 8. An error measure is introduced by considering the displacement in y-direction over all time steps:

$$e^{\mathrm{P}} = \frac{\sqrt{\sum_{k=1}^{N^{\mathrm{ts}}} (d_y^{\mathrm{ref}}(\boldsymbol{x}^{\mathrm{A}}, t_k) - d_y(\boldsymbol{x}^{\mathrm{A}}, t_k))^2}}{\sqrt{\sum_{k=1}^{N^{\mathrm{ts}}} d_y^{\mathrm{ref}}(\boldsymbol{x}^{\mathrm{A}}, t_k)^2}} \tag{50}$$

For the fluid, we measure the error of the traction in y-direction over all time steps:

Table 2 Tolerances for the simulations of the lid-driven cavity flow

	Coupling tolerances		Inner tolerances	
	Displacement	Traction	Structure	Fluid
	$tol^{\boldsymbol{d}}$	$tol^{\boldsymbol{t}}$	$tol^{\mathcal{S}}$	$tol^{\mathcal{F}}$
Fine reference	10^{-8}	10^{-5}	10^{-12}	10^{-10}
Previous studies	10^{-6}	10^{-3}	10^{-10}	10^{-8}
After efficiency study	10^{-6}	10^{-3}	10^{-6}	10^{-6}
Rule A, B, and C (max)	10^{-6}	10^{-3}	10^{-3}	10^{-3}

$$e^{\mathrm{T}} = \frac{\sqrt{\sum_{k=1}^{N^{\mathrm{ts}}} (t_y^{\mathrm{ref}}(\boldsymbol{x}^{\mathrm{A}}, t_k) - t_y(\boldsymbol{x}^{\mathrm{A}}, t_k))^2}}{\sqrt{\sum_{k=1}^{N^{\mathrm{ts}}} t_y^{\mathrm{ref}}(\boldsymbol{x}^{\mathrm{A}}, t_k)^2}} \tag{51}$$

In previous studies, we used larger tolerances (see Tab. 2, second line) – and based on the reference solution, we determined an error of $e^{\mathrm{P}} = 6.34 \cdot 10^{-4}$ and $e^{\mathrm{T}} = 6.47 \cdot 10^{-1}$. Comparing the efficiency of the classical approach with that of the novel ones using adaptive tolerances, we would like to achieve the same accuracy. At the same time, we have to make sure that none of the tolerances is actually too low, which could lead to an overestimation of the efficiency gain. This is because, without such a check, a solver may perform many unnecessary inner iterations, as the accuracy is limited by the other solver and/or the coupling tolerance. In this situation, adaptive tolerances will yield a gain in efficiency by reducing the number of inner iterations without affecting the accuracy, however, the same could be achieved simply by selecting a larger tolerance in the first place.

To this end, we start with the tolerance values from previous studies and increase each tolerance individually while keeping the others low. The displacement $\mathbf{d}(\boldsymbol{x}^{\mathrm{A}})$ of the structure at the point $\boldsymbol{x}^{\mathrm{A}} = (0.5 \;\; 0.0 \;\; 0.0)^T$ from the previous studies is depicted in Fig. 8. The corresponding errors are shown in Fig. 9. The same has been done for the tractions in y-direction at point $\boldsymbol{x}^{\mathrm{A}}$, as depicted in Fig. 10. Comparing Figs. 9 and 10, it is seen that the displacement calls for lower tolerances.

For the efficiency studies, all tolerance values are selected as the next lower power of 10 that yields the desired accuracy. The tolerances are listed in Table 2. Compared to the tolerances from the previous study, only the solver tolerances are found to be unnecessarily small and are therefore increased, while the coupling tolerances remain unchanged. The last line of Table 2 shows the maximum values of the inner tolerance, which will be used within the different adaptive tolerance rules for the later simulations. For all rules, a solver without resetting will be used.

Regarding the inner tolerance in $\mathcal{S}$, we would like to mention that a slight increase in the desired tolerance, e.g., to 10^{-4}, would make it possible to increase $tol^{\mathcal{S}}$ by many orders of magnitude. This is because in most time steps, only a single inner solver (Newton-Raphson) iteration is performed. To this end, it is expected that only little can be gained within the structure solver for this particular example and that the adaptive tolerances may have no effect at all on the inner iteration in $\mathcal{S}$ if the desired accuracy is increased.

5.1.3 Adaptive tolerances

Next, we compare the different solver rules to investigate their influence on the computational time. As a reference for calculating a speed-up, the solver with resetting and fixed tolerance is used. For Rule A, we chose $\alpha = 2.0$, and for Rules B and C, we set $F = 0.1$. Figure 11 shows the accumulated number of inner iterations in $\mathcal{S}$ and Fig. 12 for $\mathcal{F}$. Starting with the structural solver $\mathcal{S}$, using a solver without resetting leads to a speed-up of 23.0 %. Rules B and C cannot improve this speed-up and lead to a reduction while still performing better in terms of computational time than the reference case. Rule A leads to an improvement and a

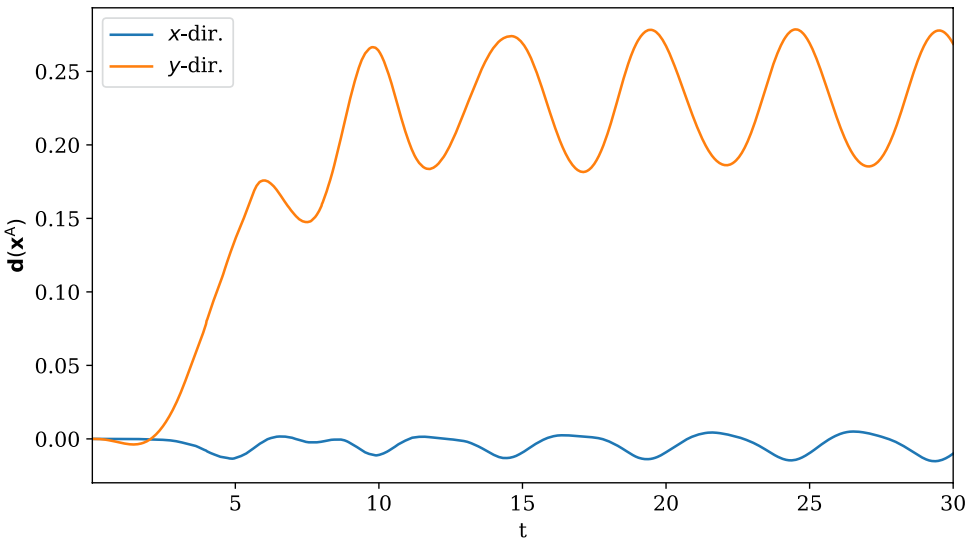


Fig. 8 Displacement $\mathbf{d}(\boldsymbol{x}^{\mathrm{A}})$ over time

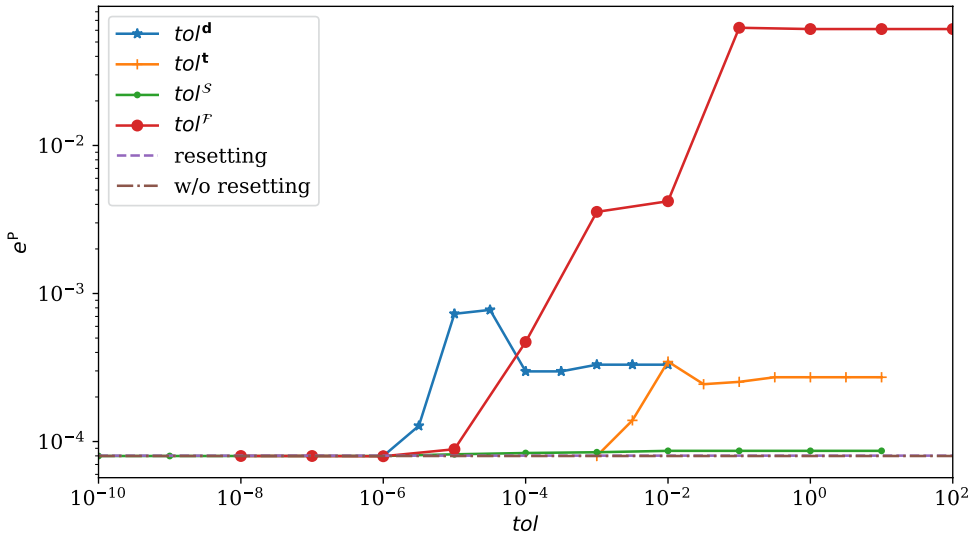


Fig. 9 Displacement error e^{P} for varying tolerances

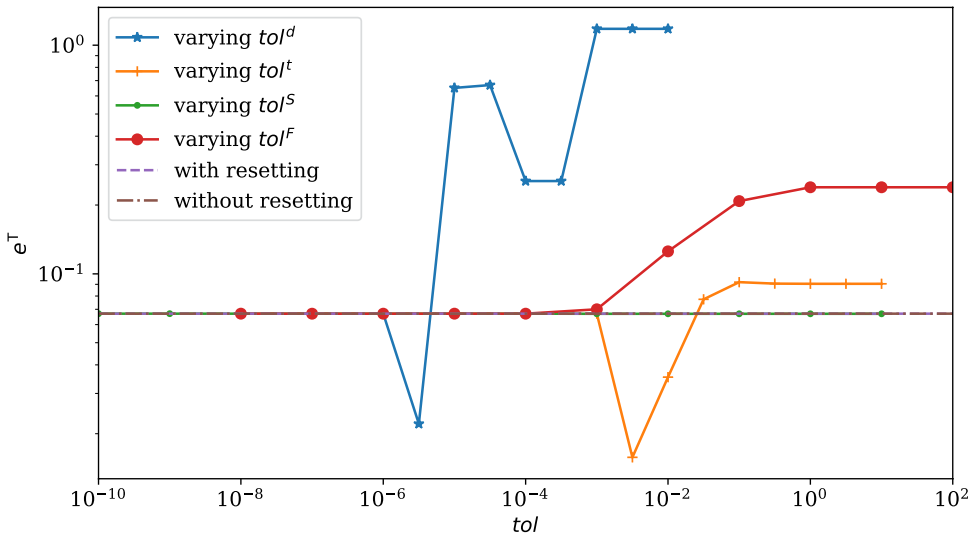


Fig. 10 Traction error e^{T} for varying tolerances

speed-up of $39.5\,\%$. The effects on the fluid solver $\mathcal{F}$ are depicted in Fig. 12.

The solver without resetting and fixed tolerance achieves a speed-up of $34.2\,\%$. In this case, using an adaptive tolerance increases the speed-up for all investigated rules. The best results could be simulated with Rules A and B, with a speed-up of approximately $58\,\%$.

Figure 13 shows the accumulated number of coupling iterations over the simulation time for the different rules. Using a solver with or without resetting does not lead to noticeable differences. Comparing that with the number of coupling iterations for the simulations using adaptive tolerances, one can see that they need approximately $25\,\%$ more coupling iterations. In [46], the trend of increased coupling iterations due to less accurate subproblem solutions was also observed.

Nevertheless, within an FSI simulation, the computational overhead of a coupling iteration is only marginal compared to the effort of an inner iteration.

The results are summarized in Tab. 3, where the total values of the different iteration numbers are also listed. As in the simple problem, we were able to confirm that it is beneficial to use a solver without resetting properties and to employ adaptive tolerance adjustment. In most FSI simulations, the fluid solver is the most time-demanding part of the simulation, which underlines the advantage of using a solver without resetting and employ adaptive tolerances, as the speed-up is not negligibly. For this example, the best choice was using a solver without resetting and Rule A for adjusting the tolerance. One has to keep in mind that these results have the desired accuracy because all rules compute the final coupling iteration using the low tolerance.

It is important to mention that the rules employed are not claimed to be the best possible choice for adaptive

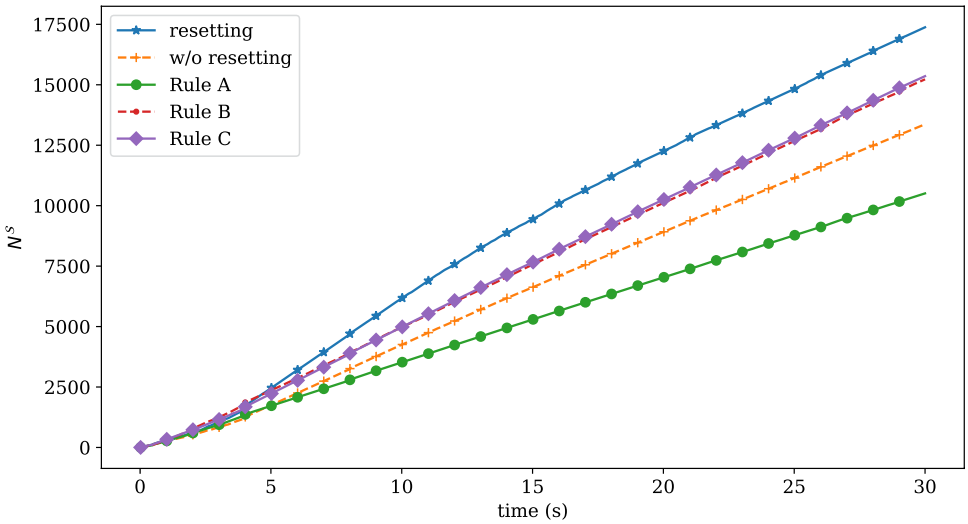


Fig. 11 Accumulated number of inner iterations in $\mathcal{S}$

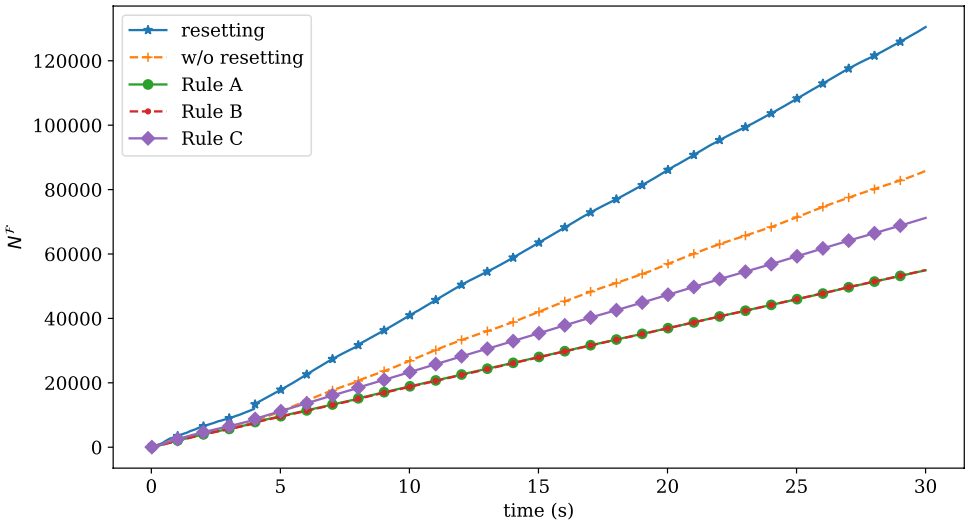


Fig. 12 Accumulated number of inner iterations in $\mathcal{F}$

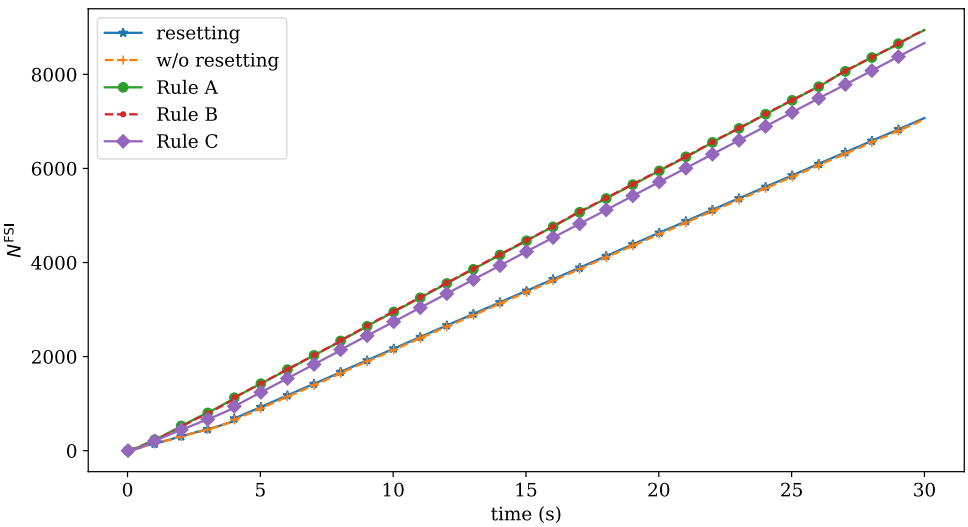


Fig. 13 Accumulated number of coupling iterations

Table 3 Effect of resetting and adaptive tolerance on inner iterations for the lid-driven cavity flow

Simulation parameters rule	Results inner iter.					
	η in %			Total no. iter.		
	Fluid	Structure	Coupl.	Fluid	Structure	Coupl.
Resetting	0	0	0	130474	17378	7072
w/o resetting	34.2	23.0	0.4	85809	13367	7043
Rule A	57.9	39.5	− 26.4	54946	10508	8942
Rule B	57.8	12.4	− 26.5	55011	15227	8948
Rule C	45.4	11.6	− 22.6	71201	15357	8667

tolerances, but they give good results in terms of the speed-up of computational time. Other approaches such as using Rule A with the exponent $-j^2$ have been investigated, but they did not achieve the desired speed-up.

5.2 Pressure pulse in an elastic tube

In this section, we investigate the performance of the adaptive tolerances for a well-known FSI benchmark problem.

5.2.1 Simulation setup

The phenomenon under investigation is a pulse wave propagating in a three-dimensional elastic tube. A similar variant of this problem was investigated previously in [10, 26, 38] and the references therein. In [46], this problem was investigated based on the strategy of limiting the number of inner solver iterations. In addition to the standard problem, we investigate different structural configurations leading to increased difficulty in coupling. The tube has a length of $L = 0.05\,\mathrm{m}$ and a diameter of $D = 0.01\,\mathrm{m}$. The fluid has a density of $1000\,\frac{\mathrm{kg}}{\mathrm{m}^3}$ and a dynamic viscosity of $3 \cdot 10^{-3}\,\mathrm{Pa\,s}$. The simulation time is $0.018\,\mathrm{s}$, and a time step size of $\Delta t = 10^{-4}\,\mathrm{s}$ is chosen. At the inlet, a pressure pulse boundary condition

$$p(t) = \begin{cases} 1333.2\,\mathrm{Pa} & \text{if } t \leq 0.003\,\mathrm{s}, \\ 0 & \text{else} \end{cases} \tag{52}$$

is prescribed. At the outlet, the pressure is set to $0\,\mathrm{Pa}$.

The fluid mesh consists of 132 000 cells, as shown in Fig. 14.

The structure domain has the same length and inner diameter as the fluid domain. The mesh consists of one element in the thickness direction, six elements in the circumferential direction, and nine elements in the axial direction. The elements have a high-order geometry description, which is based on the quasi-regional mapping approach (see [15, 25]). The structure mesh is shown in Fig. 15, where the black lines indicate the elements. Note that the uncolored edges only appear due to the low post-processing resolution

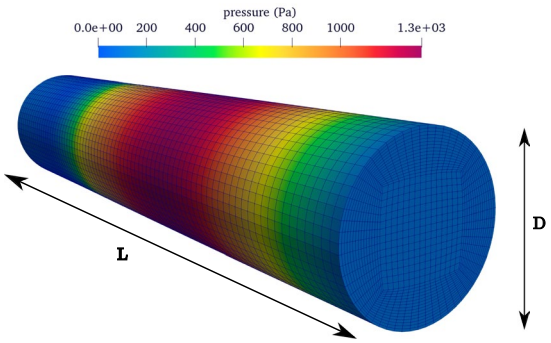


Fig. 14 Finite-volume mesh for the fluid domain with pressure distribution at time $t = 0.0025\,\mathrm{s}$

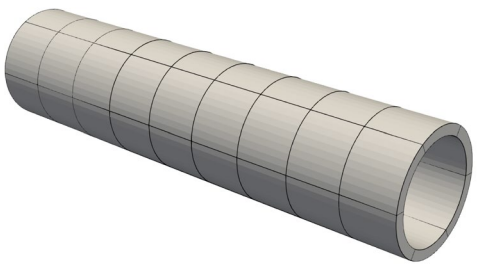


Fig. 15 High-order finite element mesh for the structure domain of the tube (black lines indicate element boundaries)

– actually, the element geometries are described with polynomials of third order.

A Saint Venant-Kirchhoff material model is chosen with a Young's modulus $E = 10^6\,\frac{\mathrm{N}}{\mathrm{m}^2}$ and a Poisson's ratio of $\nu = 0.3$. The density is set to $\rho = 1000\,\frac{\mathrm{kg}}{\mathrm{m}^3}$. As a time-integration scheme, a Newmark method is used with $\beta = 0.49$ and $\gamma = 0.9$. The structure is clamped on both ends (at the inlet and the outlet). Different values are investigated regarding the wall thickness d. Four setups are compared, namely $d = 0.001\,\mathrm{m}$, $d = 0.0005\,\mathrm{m}$, $d = 0.00025\,\mathrm{m}$, and $d = 0.000167\,\mathrm{m}$. This results in an increasing added-mass effect with decreasing thickness. The different setups, including the displacement field at $t = 0.005\,\mathrm{s}$, are shown in Fig. 16. Similar to a directly increased added-mass effect due to an increased fluid density, we expect this change to make the problem more prone to instabilities. As expected, also the wave velocity, i.e., the velocity of the pulse, as it

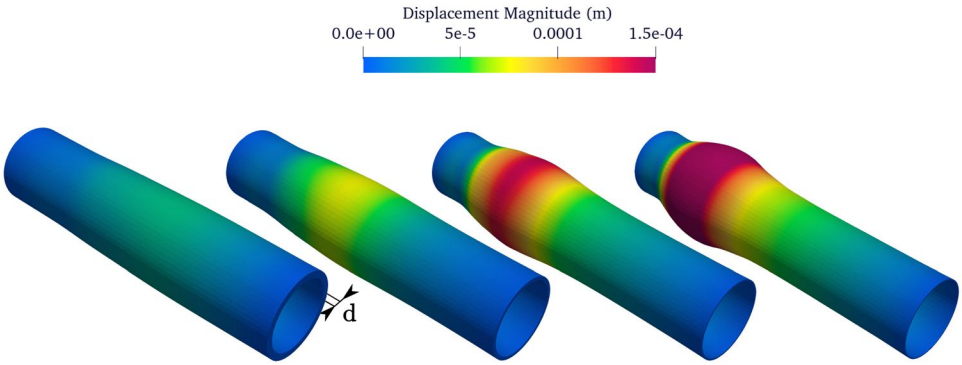


Fig. 16 Illustration of the displacement magnitude field at $t = 0.005\,\mathrm{s}$ for different wall thicknesses (a scaling factor of 10 was used)

Table 4 Tolerances for the simulations of the pressure pulse in an elastic tube

	Coupling tolerances		Inner tolerances	
	Absolute tolerance	Relative tolerance	Structure	Fluid
	tol_{abs}	tol_{rel}	$tol^{\mathcal{S}}$	$tol^{\mathcal{F}}$
Resetting	10^{-9}	10^{-4}	10^{-10}	10^{-6}
Rule A (max)	10^{-9}	10^{-4}	10^{-5}	10^{-3}

travels through the tube, decreases with decreasing wall thickness.

5.2.2 Adaptive tolerances

Since it constitutes the most promising rule based on the previous investigations, we employ only Rule A here. The convergence criteria for this example are more involved than the simple criteria considered so far. As in [27], an absolute criterion $\|\mathbf{r}_{k+1,j}\| < tol_{\mathrm{abs}}$ and a relative criterion $\frac{\|\mathbf{r}_{k+1,j}\|}{\|\mathbf{r}_{k=0,j}\|} < tol_{\mathrm{rel}}$ are used. Like in several other investigations [26, 30, 38], this yields more a reliable coupling and shall be tested in combination with the adaptive tolerances here in order to demonstrate the general applicability of this concept. The setting for the coupling tolerances as well as the minimum values of the inner solver tolerances $tol^{\mathcal{S}}$ and $tol^{\mathcal{F}}$ are chosen as in Tab. 4. The maximum value for the inner solver tolerance in Rule A has the same difference in the exponent as in the lid-driven cavity flow problem from the previous chapter.

5.2.3 Results

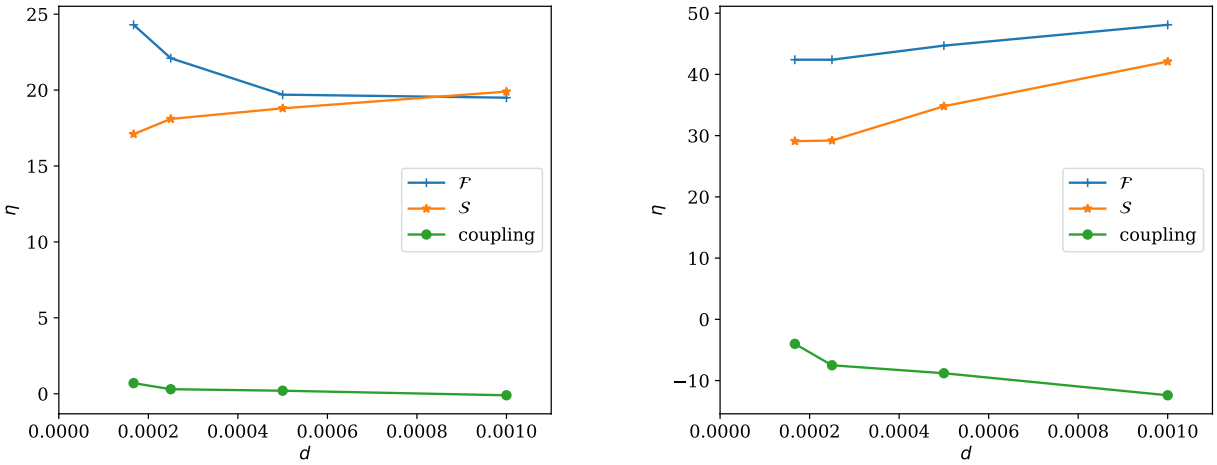
The results in terms of speed-up are illustrated in Fig. 17 for simulations using solvers without resetting and simulations using solvers that additionally make use of adaptive tolerances.

Starting with the results for the solvers without resetting and fixed tolerances in Fig. 17a , one can see that, independently of the thickness *d*, a speed-up in the range of $19 - 24\,\%$ can be achieved for the inner iterations of the fluid solver and $17\,\%$ to $22\,\%$ for the inner iterations of the structural solver. The overall number of coupling iterations is nearly the same as for the solver with resetting.

Looking at the results for the simulations based on solvers without resetting and adaptive tolerances in Fig. 17b , we observe an even higher speed-up on the structural side with $29\,\%$ to $42\,\%$ and on the fluid side with $42\,\%$ to $48\,\%$ throughout the simulation. The speed-up with regard to the number of coupling iterations for the different structure setups varies in the range of $-12\,\%$ to $-4\,\%$.However, as already mentioned, the computational effort for the coupling is negligible compared to the inner solver iterations. Of course, the total number of coupling iterations increases with increasing added-mass effect for all the solver variants considered. These additional results are summarized in the appendix A in Tab. 5 – 7 for each structural setup, where the total values of the different iteration numbers are listed.

6 Conclusion

In this paper, we presented an adaptive tolerance approach for the simulation of partitioned FSI problems. We showed that the idea of adaptive tolerances can be easily realized within the framework of solving multi-field problems.



(a) Solver without resetting and fixed tolerances. (b) Solver without resetting and adaptive tolerances.

Fig. 17 Speed-up for different wall thicknesses

Simulations were performed for a simple test problem to describe the main idea of adaptive tolerances combined with solvers with and without resetting. In addition, partitioned FSI simulations were performed to show the potential of adaptive tolerances according to different rules. We were able to show that using a solver without resetting properties combined with an adaptive tolerance rule serves to significantly reduce the computational effort in terms of the number of inner solver iterations.

Investigating the idea within a simple test problem, we observed that inner solvers without resetting properties lead to a computational speed-up. A speed-up of $61.7\,\%$ was achieved compared to the reference case using a solver with resetting. This speed-up could be further improved by using adaptively adjusted tolerances for the inner solver iterations. This led to the conclusion that a combination of a solver without resetting and adaptive tolerances yields the most efficient method.

The simple rule for choosing the inner tolerance for the simple test problem combined with the solver without resetting resulted in the best case in a speed-up of $75.5\,\%$.

These results were tested on the lid-driven cavity flow, a numerical FSI benchmark problem, using more advanced rules to adaptively set the inner solver tolerance. In our work, these rules depend on the number of coupling iterations or the coupling residual. Both sets of rules lead to a speed-up of the overall FSI simulation. The best rule combined with a solver without resetting achieved a speed-up of $57.7\,\%$ for the fluid solver and $26.3\,\%$ for the structural solver. The number of total coupling iterations increased, but in terms of computational time, the resulting overhead is negligible compared to the inner solver iterations.

As a more advanced application, the simulation of a pressure pulse through an elastic three-dimensional tube confirmed the previous investigations. Different structural configurations were considered using four different wall thickness values. These different configurations lead to increased difficulty in coupling. A speed-up was achieved for all considered wall thicknesses. The computational time of the structural solvers decreased in the range of $29\,\%$ to $42\,\%$. On the fluid side, a speed-up in the range of $42\,\%$ to $48\,\%$ was achieved. Of course, these results are problem-dependent and may look different for other problems.

Further investigations could be performed using the methods introduced for other application simulations, such as a cardiovascular anastomosis or maritime applications. Other more advanced rules could also be tested. It is also necessary to carry out further investigations regarding the interaction between the adaptive tolerances and the accelerator.

A Results for the pressure pulse in an elastic tube problem

Table 5 Effects of resetting and adaptive tolerance on inner iterations for the pressure pulse in an elastic tube problem. Thickness 1 mm

Simulation parameters rule	Results inner iter.					
	η in %			Total no. iter.		
	Fluid	Structure	Coupl.	Fluid	Structure	Coupl.
Resetting	0	0	0	60633	5657	1906
w/o resetting	19.5	19.9	− 0.1	48819	4534	1907
Rule A	48.1	42.1	− 12.4	31460	3274	2143

Table 6 Effects of resetting and adaptive tolerance on inner iterations for the pressure pulse in an elastic tube problem. Thickness 0.5 mm

Simulation parameters rule	Results inner iter.					
	η in %			Total no. iter.		
	Fluid	Structure	Coupl.	Fluid	Structure	Coupl.
Resetting	0	0	0	70645	6713	2234
w/o resetting	19.7	18.8	0.2	56752	5449	2229
Rule A	44.7	34.8	− 8.8	39042	4405	2431

Table 7 Effects of resetting and adaptive tolerance on inner iterations for the pressure pulse in an elastic tube problem. Thickness 0.25 mm

Simulation parameters rule	Results inner iter.					
	η in %			Total no. iter.		
	Fluid	Structure	Coupl.	Fluid	Structure	Coupl.
Resetting	0	0	0	87124	8377	2778
w/o resetting	22.1	18.1	0.3	67878	6862	2771
Rule A	42.4	29.2	− 7.5	50207	5927	2985

Table 8 Effects of resetting and adaptive tolerance on inner iterations for the pressure pulse in an elastic tube problem. Thickness 0.167 mm

Simulation parameters rule	Results inner iter.					
	η in %			Total no. iter.		
	Fluid	Structure	Coupl.	Fluid	Structure	Coupl.
Resetting	0	0	0	100792	9844	3224
w/o resetting	24.3	17.1	0.7	76259	8158	3203
Rule A	42.4	29.1	− 4.0	58034	6980	3352

B Time measure for the pressure pulse in an elastic tube problem with wall thickness 1 mm

To confirm the assumption that the number of inner solver iterations serves as an effective performance metric, we simulate the pressure pulse in an elastic tube with wall thickness 1mm (see Sec. 5.2) and measure the speed-up based on the total simulation runtime. All simulations are performed using a single kernel. The results, summarized in Tab. 9, demonstrate that the speed-up of the total simulation runtime closely matches the speed-up achieved in the given number of inner solver iterations.

Table 9 Effects of resetting and adaptive tolerance on inner iterations and total simulation runtime for the pressure pulse in an elastic tube problem. Thickness 1 mm

Simulation parameters rule	Results inner iter.						
	η in %			Total no. iter.			Speed-up
	Fluid	Structure	Coupl.	Fluid	Structure	Coupl.	Total runtime
Resetting	0	0	0	60633	5657	1906	0%
w/o resetting	19.5	19.9	− 0.1	48819	4534	1907	19.9%
Rule A w/o resetting	48.1	42.1	− 12.4	31460	3274	2143	45.0%

Acknowledgements The authors gratefully acknowledge the support of the DFG (Deutsche Forschungsgemeinschaft) under DU 405/20-1 (grant number 503865803).

Funding Open Access funding enabled and organized by Projekt DEAL.

References

1. Aitken AC (1950) Studies in practical mathematics, on the iterative solution of a system of linear equations. Proc of the Royal Soc of Edinburgh Sect A Math and Phys Sci 63(1):52–60. https://doi.org/10.1017/S008045410000697X
2. Bazilevs Y, Calo VM, Hughes TJR, Zhang Y (2008) Isogeometric fluid-structure interaction: theory, algorithms, and computations. Comput Mech 43(1):3–37. https://doi.org/10.1007/s00466-008-0315-x
3. Bazilevs Y, Takizawa K, Tezduyar TE (2013) Computational fluid-structure interaction: methods and applications. Wiley Series in Computational Mechanics. John Wiley & Sons, Hoboken
4. Bletsos G, Rung T, Radtke L (2024) Hemodynamics in arterial by pass graft anastomoses with varying cuff sizes and proximal flow paths: a fluid-structure interaction study. Comput Methods Biomech Biomed Engin 28(7):1066–1085. https://doi.org/10.1080/10255842.2024.2310747
5. Broyden CG (1965) A class of methods for solving nonlinear simultaneous equations. Math of Comput 19:577–593. https://doi.org/10.1090/S0025-5718-1965-0198670-6
6. Bungartz HJ, Lindner F, Gatzhammer B, Mehl M, Scheufele K, Shukaev A, Uekermann B (2016) preCICE - A fully parallel library for multi-physics surface coupling. Comput Fluids 141:250–258. https://doi.org/10.1016/j.compfluid.2016.04.003. (Advances in Fluid-Structure Interaction)
7. Bungartz HJ, Lindner F, Mehl M, Uekermann B (2015) A plug-and-play coupling approach for parallel multi-field simulations. Comput Mech 55(6):1119–1129. https://doi.org/10.1007/s00466-014-1113-2
8. Bungartz HJ, Mehl M, Schäfer M (eds) (2010) Fluid-structure interaction II, modelling, simulation, optimisation. Vol. 73. Lecture Notes in Computational Science and Engineering. Springer. isbn: 978-3-642-14205-5
9. Bungartz HJ, Schäfer M (eds) (2006) Fluid-structure interaction, modelling, simulation and optimisation. Vol. 53. Lecture Notes in Computational Science and Engineering. Springer. isbn: 978-3-540-34595-4
10. Degroote J, Bathe KJ, Vierendeels J (2009) Performance of a new partitioned procedure versus a monolithic procedure in fluid-structure interaction. Comput Struct 87(11):793–801. https://doi.org/10.1016/j.compstruc.2008.11.013
11. Degroote J, Bruggeman R, Haelterman R, Vierendeels J (2008) Stability of a coupling technique for partitioned solvers in FSI applications. Comput Struct 86(23–24):2224–2234. https://doi.org/10.1016/j.compstruc.2008.05.005
12. Degroote J, Haelterman R, Annerel S, Bruggeman P, Vierendeels J (2010) Performance of partitioned procedures in fluid-structure interaction. Comput Struct 88(7–8):446–457
13. Degroote J, Vierendeels J (2012) Multi-level quasi-Newton coupling algorithms for the partitioned simulation of fluid-structure interaction. Comput Methods Appl Mech Eng 225:14–27. https://doi.org/10.1016/j.cma.2012.03.010
14. Demirdžić I, Perić M (1988) Space conservation law in finite volume calculations of fluid flow. Int J Numer Meth Fluids 8:1037–1050
15. Düster A, Bröker H, Rank E (2001) The p-version of the finite element method for threedimensional curved thin walled structures. Int J Numer Meth Eng 52(7):673–703. https://doi.org/10.1002/nme.222
16. Erbts P, Hartmann S, Düster A (2015) A partitioned solution approach for electro-thermomechanical problems. Arch Appl Mech 85:1075–1101. https://doi.org/10.1007/s00419-014-0941-z
17. Erturk E, Corke TC, Gökçöl C (2005) Numerical solutions of 2-D steady incompressible driven cavity flow at high Reynolds numbers. Int J Numer Meth Fluids 48(7):747–774
18. Gallinger TG (2010) Effiziente Algorithmen zur partitionierten Lösung stark gekoppelter Probleme der Fluid-Struktur-Wechselwirkung. Doctoral thesis. Technische Universität München
19. Ghia U, Ghia KN, Shin CT (1982) High-Re solutions for incompressible flow using the Navier–Stokes equations and a multigrid method. J Comput Phys 48(3):387–411
20. Haelterman R, Bogaers AEJ, Scheufele K, Uekermann B, Mehl M (2016) Improving the performance of the partitioned QN-ILS procedure for fluid-structure interaction problems: filtering. Comput Struct 171(Supplement C):9–17. https://doi.org/10.1016/j.compstruc.2016.04.001
21. Hartmann S, Tebbens JD, Quint KJ, Meister A (2009) Iterative solvers within sequences of large linear systems in nonlinear structural mechanics. ZAMM - J Appl Math and Mech / Zeitschrift für Angewandte Math und Mechanik 89(9):711–728. https://doi.org/10.1002/zamm.200800211
22. Hestenes MR, Stiefel E (1952) Methods of conjugate gradients for solving linear systems. J Res Nat Bur Standards 49:409–436
23. Irons B, Tuck RC (1969) A version of the Aitken Accelerator for Computer implementation. Int J Numer Meth Eng 1:275–277
24. Kelley CT (2003) Society for industrial and applied mathematics
25. Királyfalvi G, Szabó BA (1997) Quasi-regional mapping for the p-version of the finite element method. Finite Elem Anal Des 27(1):85–97. https://doi.org/10.1016/S0168-874X(97)00006-1
26. König M (2018) Partitioned solution strategies for strongly-coupled fluid-structure interaction problems in maritime applications. Doctoral thesis. https://doi.org/10.15480/882.1736
27. König M, Radtke L, Düster A (2016) A flexible C++ framework for the partitioned solution of strongly coupled multifield problems. Comput Math Appl 72(7):1764–1789. https://doi.org/10.1016/j.camwa.2016.07.031
28. Lampe T, Radtke L, Abdel-Maksoud M, Düster A (2020) A partitioned solution approach for the simulation of dynamic behaviour and acoustic signature of flexible cavitating marine propellers. Ocean Eng 197:106854. https://doi.org/10.1016/j.oceaneng.2019.106854
29. Leonard BP (1979) A stable and accurate convective modelling procedure based on quadratic upstream interpolation. Comput Methods Appl Mech Eng 19(1):59–98. https://doi.org/10.1016/0045-7825(79)90034-3
30. Lund J, Ferreira González D, Neitzel-Petersen JC, Radtke M, Abdel-Maksoud L, Düster A (2023) Validation of a partitioned

fluid-structure interaction simulation for turbo machine rotors. Ships Offshore Struct 18(6):775–786. https://doi.org/10.1080/17445302.2022.2069389
31. Macleod AJ (1986) Acceleration of vector sequences by multi-dimensional Δ^2 methods. Commun Appl Numer Methods 2(4):385–392. https://doi.org/10.1002/cnm.1630020409
32. Mayr M, Wall WA, Gee MW (2018) Adaptive time stepping for fluid-structure interaction solvers. Finite Elem Anal Des 141:55–69. https://doi.org/10.1016/j.finel.2017.12.002
33. Mehl M, Uekermann BB, Bijl H, Blom D, Gatzhammer B, van Zuijlen A (2016) Parallel coupling numerics for partitioned fluid-structure interaction simulations. Comput Math Appl 71(4):869–891. https://doi.org/10.1016/j.camwa.2015.12.025
34. Mehl M, Uekermann B, Bijl H, Blom D, Gatzhammer B, Zuijlen A (2016) Parallel coupling numerics for partitioned fluid–structure interaction simulations. Comput Math Appl 71(4):869–891. https://doi.org/10.1016/j.camwa.2015.12.025
35. Mock DP (2001) Partitionierte Lösungsansätze in der Strukturdynamik und der Fluid-Struktur- Interaktion. Doctoral thesis
36. Newmark NM (1959) A method of computation for structural dynamics. J Eng Mech Division ASCE 85:67-EM3-94
37. Patankar SV, Spalding DB (1972) A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows. Int J Heat Mass Transf 15(10):1787–1806. https://doi.org/10.1016/0017-9310(72)90054-3
38. Radtke L (2020) A partitioned solution approach for fluid-structure interaction problems in the arterial system. Doctoral thesis. https://doi.org/10.15480/882.3638
39. Radtke L, Bletsos G, Kühl N, Suchan T, Rung T, Düster A, Welker K (2023) Parameter- free shape optimization: various shape updates for engineering applications. Aerospace 10(9):751. https://doi.org/10.3390/aerospace10090751
40. Radtke L, König M, Düster A (2017) The influence of geometric imperfections in cardiovascular FSI simulations. Comput Math Appl 74(7):1675–1689. https://doi.org/10.1016/j.camwa.2017.04.012
41. Radtke L, Lampe T, Abdel-Maksoud M, Düster A (2018) A partitioned solution approach for the simulation of the dynamic behaviour of flexible marine propellers. Ship Technol Res 0(0):1–14. https://doi.org/10.1080/09377255.2018.1542782
42. Radtke L, Larena-Avellaneda A, Debus ES, Düster A (2016) Convergence acceleration for partitioned simulations of the fluid-structure interaction in arteries. Comput Mech 57(6):901–920. https://doi.org/10.1007/s00466-016-1268-0
43. Rung T, Wöckner K, Manzke M, Brunswig J, Ulrich C, Stück A (2009) Challenges and perspectives for maritime CFD applications. Jahrbuch der Schiffbautechnischen Gesellschaft 103:127–139
44. Saksono PH, Dettmer WG, Perić D (2007) An adaptive remeshing strategy for flows with moving boundaries and fluid-structure interaction. Int J Numer Meth Eng 71(9):1009–1050
45. Sartorti R, Kühne C, Radtke L, Düster A (2025) Stabilization techniques and adaptive conjugate gradient solver tolerances for the finite cell method. Adv Comput Sci Eng 3:24–45. https://doi.org/10.3934/acse.2025003
46. Spenke T, Delaissé N, Degroote J, Hosters N (2024) On the number of subproblem iterations per coupling step in partitioned fluid-structure interaction simulations. Int J Numer Meth Eng 125(7):e7420. https://doi.org/10.1002/nme.7420
47. Szabó BA, Babuška I (2011) Introduction to finite element analysis: formulation, verification, and validation. Wiley-Blackwell, Chichester
48. Vázquez JGG (2007) Nonlinear analysis of orthotropic membrane and shell structures including fluid-structure interaction. Doctoral thesis. Universitat Politècnica de Catalunya
49. Vierendeels J, Lanoye L, Degroote J, Verdonck P (2007) Implicit coupling of partitioned fluid-structure interaction problems with reduced order models. Comput Struct 85(11):970–976. https://doi.org/10.1016/j.compstruc.2006.11.006
50. Wiegard B, König M, Radtke L, Lund J, Netzbandt S, Abdel-Maksoud M, Düster A (2021) Fluid-structure interaction and stress analysis of a floating wind turbine. Mar Struct 78:102970. https://doi.org/10.1016/j.marstruc.2021.102970
51. Zander N, Bog T, Elhaddad M, Espinoza R, Hu H, Joly A, Wu C, Zerbe P, Düster A, Kollmannsberger S, Parvizian J, Ruess M, Schillinger D, Rank E (2014) FCMLab: a finite cell research toolbox for MATLAB. Adv Eng Softw 74:49–63. https://doi.org/10.1016/j.advengsoft.2014.04.004