Technische Universität München

TUM School of Engineering and Design

Enhanced Algorithms for Fluid-Structure Interaction Simulations: Accurate Temporal Discretization and Robust Convergence Acceleration

Shahrokh Shayegan

Vollständiger Abdruck der von der TUM School of Engineering and Design der Technischen Universität München zur Erlangung eines

Doktors der Ingenieurwissenschaften (Dr.-Ing.)

genehmigten Dissertation.

Vorsitz:

Prof. Dr.-Ing. habil. Fabian Duddeck

Prüfer*innen der Dissertation:

1. Prof. Dr.-Ing. Kai-Uwe Bletzinger
2. Prof. Dr.-Ing. habil. Roland Wüchner
3. Prof. Riccardo Rossi, Ph.D.

Die Dissertation wurde am 08.11.2022 bei der Technischen Universität München eingereicht und durch die TUM School of Engineering and Design am 24.04.2023 angenommen.

Schriftenreihe des Lehrstuhls für Statik
TU München

Band 58

Shahrokh Shayegan

Enhanced Algorithms for Fluid-Structure Interaction Simulations: Accurate Temporal Discretization and Robust Convergence Acceleration

München 2023

Veröffentlicht durch

Kai-Uwe Bletzinger
Lehrstuhl für Statik
Technische Universität München
Arcisstr. 21
80333 München

Telefon: +49(0)89 289 22422
Telefax: +49(0)89 289 22421
E-Mail: kub@tum.de
Internet: www.bgu.tum.de/st/startseite/

ISBN: 978-3-943683-70-7

To my lovely parents, Ehteram and Hossein

Abstract

The present work proposes novel methods and strategies for enhancing the robustness, accuracy, and efficiency of multiphysics simulations with a particular focus on fluid-structure interaction (FSI) problems. At its core, the present work deals with two main aspects of the FSI problems: stable and accurate temporal discretization of FSI problems, including non-matching time steps, and enhanced robustness and efficiency of interface quasi-Newton convergence acceleration methods for the FSI simulations.

An end-of-step-equilibrium form of the generalized-α scheme, abbreviated as EG-α scheme, is presented for the temporal discretization of FSI problems. The equilibrium equation of the problem is satisfied exactly at discrete time points instead of the classical averaged form. The EG-α scheme particularly benefits the temporal discretization of stabilized formulations for incompressible Navier-Stokes equations on fixed and moving domains. For the structural problems, a variant of the EG-α scheme for first-order systems is proposed that avoids doubling the number of unknowns in the problem's linear system of equations. The consistent temporal discretization of the FSI problem using the EG-α scheme alleviates the need for temporal interpolation of interface quantities between the fluid and solid domains required by the classical form of the G-α scheme.

The present contribution also proposes multi-time-step coupling algorithms for partitioned, strongly coupled FSI problems. Both fluid and structure subcycling schemes are proposed and analyzed with a particular focus on the role of discrete coupling conditions in multi-time-step FSI problems, interpolation of the kinematic quantities from coarse to fine time scales, and transferring of interface forces between different time levels. The proposed schemes allow accurate and stable multi-time-step FSI simulations with high time step ratios.

In this work, some of the issues surrounding the robustness and efficiency of the interface quasi-Newton methods are addressed. The efficient and robust solution of the least squares problem, arising from the interface quasi-Newton methods, via QR factorization is addressed. The use of column scaling technique in filtering based on QR factorization is proposed, and the positive effect of column scaling on the robustness and efficiency of the interface quasi-Newton methods is demonstrated. Furthermore, this work proposes a strategy for combining filtering techniques and automatic determination of the time step history in the IQN-ILS method. A major question concerning the reuse of information from the previous time steps in the IQN-ILS method is the optimal combination of values for the maximum time step history and the filtering tolerance. The proposed strategy reduces the need for trial and error to determine the optimal number of previous time steps in the IQN-ILS method and achieves a good performance for various FSI problems with a set of default parameters.

Kurzfassung

In der vorliegenden Arbeit werden neue Methoden und Strategien zur Erhöhung der Robustheit, Genauigkeit und Effizienz von Multiphysik-Simulationen mit besonderem Schwerpunkt auf Fluid-Struktur-Interaktionsproblemen (FSI) vorgeschlagen. Im Kern befasst sich diese Arbeit mit zwei Aspekten von FSI-Problemen: stabile und genaue zeitliche Diskretisierung von FSI-Problemen, einschließlich nicht übereinstimmender Zeitschritte, und Robustheit und Effizienz von Quasi-Newton-Methoden für FSI-Simulationen.

Für die zeitliche Diskretisierung der FSI-Problemen wird eine End-of-step-equilibrium-Form des generalized-α Schemas, abgekürzt EG-α Schema, präsentiert. Die Gleichgewichtsgleichung des Problems wird anstelle der klassischen Durchschnittsform genau zu diskreten Zeitpunkten erfüllt. Das EG-α Schema ist besonders vorteilhaft für die zeitliche Diskretisierung von stabilisierten Formulierungen für inkompressible Navier-Stokes-Gleichungen. Für Strukturprobleme wird eine Variante des EG-α Schemas vorgeschlagen, die eine Verdoppelung der Anzahl der Unbekannten in den linearen Gleichungssystemen des Problems vermeidet. Durch die konsistente zeitliche Diskretisierung des FSI-Problems mit dem EG-α Schema entfällt die Notwendigkeit der zeitlichen Interpolation von Größen zwischen Fluid und Struktur, die bei der klassischen Form des G-α Schemas erforderlich ist.

In diesem Beitrag werden auch Multi-time-step-Kopplungsalgorithmen für partitionierte, stark gekoppelte FSI-Probleme vorgeschlagen. Es werden sowohl Fluid- als auch Struktur-Subcycling-Verfahren vorgeschlagen, wobei ein besonderer Schwerpunkt auf der Rolle diskreter Kopplungsbedingungen in Multi-time-step-FSI-Problemen, der Interpolation der kinematischen Größen von groben zu feinen Zeitskalen und der Übertragung von Schnittstellenkräften zwischen verschiedenen Zeitskalen liegt. Die vorgeschlagenen Schemata ermöglichen genaue und stabile Multi-time-step-FSI-Simulationen mit hohen Zeitschrittverhältnissen.

In dieser Arbeit werden einige der Fragen bezüglich der Robustheit und Effizienz der Quasi-Newton-Methoden für FSI-Probleme behandelt. Die Verwendung der Spaltenskalierung bei der Filterung auf Basis der QR-Faktorisierung wird vorgeschlagen, und der positive Effekt der Spaltenskalierung auf die Robustheit und Effizienz der Quasi-Newton-Methoden wird demonstriert. Darüber hinaus wird in dieser Arbeit eine Strategie zur Kombination von Filterverfahren und automatischer Bestimmung des Zeitschrittverlaufs in der IQN-ILS-Methode vorgeschlagen. Eine wichtige Frage bezüglich der Wiederverwendung von Informationen aus den vorherigen Zeitschritten in der IQN-ILS-Methode ist die optimale Kombination von Werten für die maximale Zeitschritt-Historie und die Filtertoleranz. Die vorgeschlagene Strategie reduziert die Notwendigkeit von Versuch und Irrtum, um die optimale Anzahl von vorherigen Zeitschritten in der IQN-ILS-Methode zu bestimmen und erreicht eine gute Leistung für verschiedene FSI-Probleme mit Default-Parametern.

Acknowledgements

My Dissertation Journey has been truly transformative, both personally and professionally. I would like to take a moment and express my deepest gratitude to those who have contributed to the successful completion of my dissertation.

First and foremost, I would like to express my sincere appreciation to Prof. Kai-Uwe Bletzinger, my supervisor, for granting me the opportunity to conduct my research at the chair of Structural Analysis. His strong support, academic freedom, and guidance have been instrumental in shaping my research endeavors. I am grateful to him for fostering an exceptional working environment that allowed me to explore and thrive in my research interests.

I would like to extend my thanks to Prof. Roland Wüchner for the initial introduction to the topic and for his invaluable guidance throughout my journey. His assistance and thought-provoking discussions have significantly contributed to my technical and personal growth. Additionally, the collaborations he facilitated played a crucial role in the success of this work, and I am truly grateful for his mentorship.

I would like to thank Prof. Riccardo Rossi for reviewing my thesis and his invaluable scientific inputs throughout the years. Our numerous technical discussions have greatly elevated the scientific caliber of my research. I am also thankful to him for providing me with the opportunity of a research stay at CIMNE, Barcelona, enriching my academic experience.

My sincere appreciation goes out to all my colleagues at the chair of Structural Analysis for their exceptional cooperation and support throughout the years. The vibrant working atmosphere and camaraderie have motivated me daily. I would like to express my gratitude to my office mates, Reza, Aditya, Daniel, and Manuel, for creating a fun and enjoyable working environment throughout the years. Special thanks are due to Reza, Andreas, Altug, and Maziar, with whom I collaborated on various topics and from whom I have learned invaluable insights. I would also like to extend my thanks to all other colleagues and friends at the chair, particularly Inigo, Klaus, Tobi, Ihar, Armin, Philipp, Ann-Kathrin, Mate, and Martin, for the wonderful time we shared together.

I am deeply grateful to my friends and family, whose support and belief in me have been the pillars of my success. I owe a profound debt of gratitude to my parents for their constant care, support, and sacrifices throughout my journey. Their unwavering love and encouragement have propelled me forward and made this achievement possible.

The support by TUM International Graduate School of Science and Engineering (IGSSE) is gratefully acknowledged.

Shahrokh Shayegan
Technische Universität München
May, 2023

Contents

Chapter 1

Introduction

1.1 Motivation

Fluid-structure interaction (FSI) is of importance in many scientific and engineering applications. Some Examples of FSI are aerodynamics and flutter of airplane wings, wind turbines, vortex-induced vibrations, blood flow in arteries and circulatory system, Coriolis mass flow meters, lightweight and membrane structures in flows, and turbomachinery. The simulation of FSI problems has presented many formidable numerical challenges and has been one of the leading areas of research in the field of multiphysics and coupled simulations. The inherently dynamic nature of many FSI problems demands special attention to the temporal discretization and raises the issue of maintaining temporal stability and accuracy in coupled problems. Another important issue in the FSI problems is the computational cost per the desired level of accuracy. In this work, we deal with a class of partitioned iterative coupling strategies that allow the coupling of established and customized solvers for fluid and structural problems with minimal intrusion. The solvers exchange only limited data (related to the boundary conditions) restricted to the coupling interface. In such an approach, the computational cost is mostly dominated by the cost of the fluid and structural solvers in each coupling iteration. Therefore, accelerating the convergence of the coupling iterations is crucial for achieving a reasonable computational cost for a desired level of accuracy. Furthermore, the robustness of the convergence acceleration technique is a prerequisite for any potential gain through reducing the number of coupling iterations. The temporal accuracy and stability of FSI problems, including non-matching time steps, and enhancing the robustness and efficiency

of convergence acceleration techniques for the FSI simulations are the main themes of the present work.

The FSI problems pose a unique challenge of handling deforming domains due to moving boundaries. Many interface-tracking and interface-capturing techniques have been developed and matured over the years. Arguably, enhancing the robustness and accuracy of the spatial discretization in the interface-capturing methods has been the main focus of researchers in recent years. In comparison, the temporal discretization aspects of the problems with moving interfaces is less explored. The generalized-α family of time integration schemes is one of the established and proven time integration schemes for the FSI problems. Such schemes satisfy the equilibrium equation in an average sense in every time step and require the evaluation of terms at intermediate time instants. Those intermediate evaluations lead to higher computational costs (especially in interface-capturing methods) and often a tedious and more complicated implementation. Those issues are more pronounced when using (residual-based) stabilization techniques in the discretization of flow problems due to the added complexity of discretizing the stabilization terms. Therefore, it is highly desirable to avoid evaluating the terms at intermediate time instants while still taking advantage of the temporal accuracy, robustness, and user-controlled high-frequency damping offered by the generalized-α family of methods.

In an FSI simulation, one may wish to use a combination of different (and possibly specialized) time integration schemes in the solid and fluid domains in order to meet the stability and accuracy requirements of integrating initial value problems of different characteristics. Depending on the nature of the problem, one may use different combinations of implicit and/or explicit time integration schemes for different subdomains. The widely varying physical properties of the interacting fluids and structures and the different accuracy and stability requirements of the numerical schemes utilized for solving the fluid and structure problem often pose different requirements on the temporal discretizations in the structural and fluid domains. It is highly desirable to choose the temporal discretizations in the fluid and solid domains based on their respective physical and numerical requirements. Another aspect of the temporal discretization of the problem is the choice of time step size in the structure and fluid domains. The required time step size in the two subdomains could be vastly different due to different physical parameters and the type of governing equations in the two subproblems leading to different physical time scales in the structure and fluid, stability requirements due to the type of the utilized time integration scheme (e.g., explicit schemes in each of the subdomains) , convergence requirements of the nonlinear solvers in the presence of highly nonlinear phenomena, e.g., high-Reynolds turbulent flows. In the literature, schemes with different time step sizes for different subdomains are sometimes referred to as asynchronous, multi-time-step, multi-rate, or subcycling schemes. Under the premise that such methods do not impair the stability and accuracy of the simulations, substantial gains in computational

efficiency (through reductions in computational time and transferring of data) can be achieved.

One of the earliest convergence acceleration techniques used in strongly coupled partitioned FSI simulations is the Gauss-Seidel method with or without relaxation. The stability issues of partitioned methods based on the Gauss-Seidel scheme, due to the artificial added-mass effect, have been extensively studied in the literature. The convergence of Gauss-Seidel iterations is usually slow but can be accelerated by using dynamic relaxation, e.g., Aitken relaxation. More stable and efficient convergence acceleration techniques have been since introduced and successfully used for partitioned strongly coupled problems in the literature. Interface quasi-Newton methods have been since developed based on the idea of approximating Jacobians using least squares models. The IQN-ILS method, which is based on approximating the inverse of the Jacobian of the interface problem, has been one of the most successful convergence acceleration techniques for the partitioned FSI problems in recent years. It has been shown that by reusing data from the previous time steps, the IQN-ILS method can exhibit excellent efficiency in time-dependent partitioned coupled simulations. However, The performance of this technique depends on user-defined parameters that might have to be tuned for different problems. Another technique that has been proven effective (and sometimes essential) in enhancing the robustness and efficiency of interface quasi-Newton methods is the filtering of columns in the matrix used for the arising least squares problem. Various filtering techniques with different levels of robustness and efficiency have been used in the literature. However, there are still some unaddressed issues surrounding the robustness and efficiency of those methods in practice.

1.2 Objectives

One of the goals of the present work is to alleviate the need to evaluate the terms of equilibrium equations in the FSI problems at intermediate time instants within a time step while retaining the temporal accuracy, robustness, and user-controlled high-frequency damping of the generalized-α time integration scheme. Satisfying the equilibrium equations exactly at the discrete time points is particularly advantageous for solving fluid problems (specially stabilized formulations for incompressible Navier-Stokes equations) on moving domains in terms of numerical accuracy and complexity of implementation. Those advantages transfer naturally to FSI simulations.

Another goal of this work is to enhance the robustness and efficiency of interface quasi-Newton methods for the convergence acceleration of FSI problems. Despite a plethora of research on quasi-Newton methods for partitioned FSI problems and tremendous advancements in this field, there are still some unaddressed issues surrounding the robustness and efficiency of those methods in practice. The present contribution intends to address a few of those issues. Some of the fine details regarding the algorithms involved

in the interface quasi-Newton methods are revisited, and several algebraic techniques are proposed to be used in order to enhance the robustness and the efficiency of the interface quasi-Newton methods at the implementation level as well as at the algorithmic level. The proposed solutions are relevant for many variants of interface quasi-Newton methods in the literature, especially for the IQN-ILS method.

The final goal of this work is to propose and analyze multi-time-step algorithms for partitioned strongly coupled FSI problems. The proposed schemes allow accurate and stable multi-time-step FSI simulations with high time step ratios. An important factor in the design of the multi-time-step schemes considered here is the applicability in a wide range of FSI applications. The iterative strongly coupled method allows the simulation of FSI problems with strong added mass effect. The previous contributions regarding the time discretization and convergence acceleration of FSI problems are incorporated into the proposed multi-time-step algorithms. The attributes of the presented schemes allow a straightforward integration into already existing software environments capable of performing FSI simulations.

1.3 Outline of the contributions

In the following, a short description of the content of each chapter is given, and a summary of the main contributions of this work is presented.

Chapter 2 deals with the time discretization of FSI problems using an end-of-step-equilibrium form of the generalized-α time integration scheme.

- An end-of-step-equilibrium form of the generalized-α time integration scheme for incompressible Navier-Stokes equations is proposed. Using this scheme, the equilibrium equation of the problem is satisfied exactly at discrete time points instead of the classical averaged form. The benefits of the end-of-step-equilibrium form of the scheme are demonstrated for the stabilized finite element formulations for incompressible Navier-Stokes equation on fixed and moving domains.
- A comprehensive analysis and comparison of different variants of the generalized-α scheme for incompressible Navier-Stokes equations on fixed and moving domains is presented.
- The end-of-step-equilibrium form of the generalized-α scheme for first-order systems is presented for the time discretization of the structure in FSI problems. The proposed form of the scheme avoids doubling the number of unknowns in the problem's linear system of equations, which is common in the state-space (displacement-velocity) formulations. The presented scheme is consistent with the time integration scheme used in the fluid problem.
- The consistent temporal discretization of the FSI problem using the end-of-step-equilibrium generalized-α scheme is presented, which avoids

the need for temporal interpolation of interface values between the fluid and solid domains in order to maintain the second-order accuracy for the whole FSI problem.

Chapter 3 presents the techniques for enhancing the robustness and efficiency of interface quasi-Newton methods for convergence acceleration of FSI problems.

- Algebraic techniques for the efficient solution of the least squares problem using QR factorization via Householder and modified Gram-Schmidt orthogonalization are presented.
- Using the column scaling technique is proposed for the interface quasi-Newton methods, and the benefits of column scaling for the conditioning of the least squares problem and the filtering of columns in the interface quasi-Newton methods are demonstrated.
- Different Choices for filtering criterion based on the diagonal entries of the R-factor in the QR factorization are analyzed and compared.

Chapter 4 presents a strategy for the automatic determination of time step history in interface quasi-Newton methods.

- A strategy for combining filtering techniques and automatic determination of the number of previous time steps in IQN-ILS method based on modifying the standard Householder QR factorization with column pivoting is proposed.
- Using an incremental condition estimator for the accurate estimation of minimum and maximum singular values in the least squares problem of interface quasi-Newton methods is proposed. This technique is beneficial for filtering techniques based on the QR factorization with or without column pivoting.

Chapter 5 presents multi-time-step algorithms for partitioned strongly coupled FSI problems.

- Fluid and structure subcycling schemes for partitioned strongly coupled FSI problems based on Dirichlet-Neumann decomposition are proposed and analyzed. The proposed schemes allow accurate and stable multi-time-step FSI simulations with high time step ratios.
- The proposed fluid subcycling scheme is based on an almost linear fluid acceleration profile in each coarse time step and a cubic Hermite interpolation for the fluid velocity from coarse to fine time steps.
- The fluid subcycling scheme based on the discrete velocity continuity condition is shown to yield considerably more accurate results compared to the traditional approach based on the discrete displacement continuity condition.

- The proposed structure subcycling scheme based on the discrete velocity continuity condition is shown to be stable and accurate for very high time step ratios, while the scheme based on the discrete displacement continuity condition is unstable even for very small time step ratios.

Chapter 6 demonstrates the application of the contributions in chapters 2-4 to an industrial FSI example of an omega-shaped Coriolis mass flow meter.

Chapter 2

Time discretization of fluid-structure interaction problems

2.1 Time discretization of the fluid problem

2.1.1 Introduction

This section is concerned with the time discretization of stabilized finite element formulations for unsteady incompressible Navier-Stokes (INS) equations on fixed and moving domains. Analyzing and enhancing the temporal accuracy and stability of time integration schemes have been an important topic of research in the field of unsteady flow simulations. The implicit time integration schemes have been popular for discretizing the INS equations due to the presence of the incompressibility constraint and to avoid restrictions on the time step size due to the CFL condition as in explicit schemes [1]. A variety of time integration schemes have been used for discretizing unsteady INS equations: the well-known backward Euler, Crank-Nicolson and second-order backward differentiation (BDF2) schemes [2, 3, 4], implicit one-step-θ schemes [4, 3, 1, 5, 6], fractional-step-θ scheme [7, 8], linear continuous and discontinuous finite element in time [5], different variants of Runge-Kutta methods [1, 9, 10, 11], and generalized-α scheme [12, 5, 13, 14, 15, 16], among others. The generalized-α scheme (hereafter referred to as G-α) for the first-order systems was proposed in [12] for the time discretization of compressible Navier-Stokes equations and was later applied to INS equations. This scheme has since

become a popular choice in fluid dynamics and FSI problems, [72, 206, 61, 210, 79, 209, 65, 211], due to its second-order accuracy, unconditional stability for linear systems and flexibility in choosing the desired high-frequency numerical dissipation. The equilibrium equation of the problem is satisfied in an average sense in every time step using the G-α scheme. Different terms in the equilibrium equation are evaluated using generalized averaged values at time instants within each time step. The differential-algebraic nature of the INS equations poses a dilemma with regard to the temporal evaluation of the pressure term and the incompressibility constraint. The aforementioned terms can be evaluated using the generalized average values, similar to the rest of the terms, or evaluated fully implicitly leading to different variants of the G-α scheme for INS equations. In [16], a review and comparison of two G-α variants for INS equations are presented.

Another level of difficulty is added to the time discretization of INS equations by the introduction of stabilized formulations. It is well known that the application of standard finite element method to the INS equations might lead to numerical instabilities. The reason for such instabilities is twofold. First, the system resulting from the INS equations has a saddle-point structure. The combination of finite element spaces for velocity and pressure can not be chosen arbitrarily in such problems and should satisfy the so-called inf-sup condition (also called the LBB condition) [17, 18, 19, 20]. The standard finite elements with equal-order interpolation for the velocity and pressure do not satisfy the inf-sup condition and lead to wiggles and instabilities in the solution. For a more detailed discussion about the stable mixed finite elements and the inf-sup condition, see, e.g., [22, 23] and the references therein. The second type of numerical instabilities in the Galerkin discretization of the INS equations occurs in the convection-dominated flows. Both of the aforementioned numerical instabilities can be remedied using stabilization techniques. Stabilized finite element methods have been successfully applied to many different numerical problems. Some of the prominent and influential stabilization techniques targeting the instabilities mentioned above and the improvement of the conservation properties of the numerical schemes are SUPG [24, 25], PSPG [26], grad-div stabilization (also known as bulk viscosity or LSIC) [27], the combination of those methods, e.g., SUPG/PSPG [28, 29], SUPG/PSPG/grad-div [30, 31], GLS [32], and VMS [33, 34]. The extra terms introduced by the mentioned residual-based stabilization techniques complicates the time discretization of the INS equations. In particular, the previously mentioned dilemma regarding the evaluation of specific terms either fully implicitly or using generalized averaged values also enters the stabilization terms through the residuals.

The consistent discretization of the INS equations on moving domains poses yet another challenge for the time integration schemes with weighted average evaluation of the equilibrium equation. While the previously mentioned challenges of the temporal evaluation of different terms remain, an important new question arises regarding the instance of the moving domain, on which

different terms of the equilibrium equation are evaluated. The most common choice is to solve the INS on the most recent instance of the moving domain at the end of each time step. However, this choice of the integration domain is not necessarily consistent with all the time integration schemes, e.g., the G-α scheme. Furthermore, different variants of the G-α scheme may require evaluations of different terms on multiple instants of the moving domain in order to preserve consistency of the results. The evaluation of the equations on the intermediate instances of the moving domain complicates the solution procedure and might even increase computational cost, specially in unfitted approaches for solving problems with moving domains. Therefore, a time integration scheme that consistently evaluates the terms of the equilibrium equation on the instance of the moving domain at exactly the end of each time step is preferred in such situations.

In this section, a variant of the G-α family of time integration schemes for INS equations is presented, which satisfies the equilibrium at the end of each time step instead of the classical weighted average form. Furthermore, different variants of the G-α scheme for INS equations are systematically analyzed and compared. The consistent temporal evaluation of the different terms in the equilibrium equation is studied for stabilized formulations of the INS equations on fixed and moving domains. The new variant is beneficial for accurately satisfying the incompressibility constraint in INS equations, reducing complexity in temporal discretization of residual-based stabilized formulations, and consistent evaluation of the equilibrium equations of the fluid problems on the instance of the moving domain at the end of each time step.

2.1.2 Incompressible Navier-Stokes equations

Let $\Omega \subset \mathbb{R}^d$, $d = \{2,3\}$, be the domain where the INS equations are solved, and Γ be the boundary of Ω, i.e., $\Gamma = \partial\Omega$. The INS equations over the time interval $[0,T]$ read:

$$\begin{aligned} \rho\partial_t \boldsymbol{v} + \rho\left(\boldsymbol{v}\cdot\nabla\right)\boldsymbol{v} - 2\mu\nabla\cdot\left(\nabla^s\boldsymbol{v}\right) + \nabla p &= \boldsymbol{f} \quad \text{in } \Omega,\ t\in(0,T], \\ \nabla\cdot\boldsymbol{v} &= 0 \quad \text{in } \Omega,\ t\in(0,T], \end{aligned} \tag{2.1}$$

where $\boldsymbol{v}$ and p denote the fluid velocity and the pressure, ρ and μ stand for the constant density and dynamic viscosity, respectively, $\boldsymbol{f}$ represents the body forces, and $\nabla^s\boldsymbol{v} = \frac{1}{2}\left(\nabla\boldsymbol{v} + \left(\nabla\boldsymbol{v}\right)^T\right)$ is the symmetric strain-rate tensor. Eq. 2.1 is subjected to the following initial and boundary conditions:

$$\begin{aligned} \boldsymbol{v} &= \boldsymbol{v}_0 && \text{in } \Omega,\ t=0, \\ \boldsymbol{v} &= \boldsymbol{v}^D && \text{on } \Gamma^D,\ t\in(0,T], \\ \left(2\mu\left(\nabla^s\boldsymbol{v}\right) - p\boldsymbol{I}\right)\cdot\boldsymbol{n} &= \boldsymbol{t}^N && \text{on } \Gamma^N,\ t\in(0,T], \end{aligned} \tag{2.2}$$

where Γ^D and Γ^N are, respectively, the Dirichlet and Neumann portions of the boundary, such that $\Gamma^D \cup \Gamma^N = \Gamma$ and $\Gamma^D \cap \Gamma^N = \varnothing$, $\boldsymbol{I}$ is the identity matrix, $\boldsymbol{n}$ is the unit normal vector, and $\boldsymbol{t}^N$ is the normal traction prescribed on the Neumann boundaries.

A few concise definitions are shortly introduced, which will facilitate the presenting the weak form of INS equations. We denote by $L^2(\Omega)$ the space of the square-integrable functions in Ω. The L^2 inner product of two scalar-valued functions in Ω is denoted by

$$(w, v)_\Omega \coloneqq \int_\Omega wv \, \mathrm{d}\Omega. \tag{2.3}$$

With a slight abuse of notation, the product of any two functions in domain Ω is denoted by the same notation as in 2.3, even if the functions are not defined in $L^2(\Omega)$ (assuming that the integral is well defined). The L^2 inner product of two functions on $\Gamma = \partial\Omega$ is denoted by

$$(w, v)_\Gamma \coloneqq \int_\Gamma wv \, \mathrm{d}\Gamma. \tag{2.4}$$

The space of functions with the distributional derivatives up to the first order belonging to $L^2(\Omega)$ is denoted by $H^1(\Omega)$. The space of functions in $H^1(\Omega)$ with zero traces on $\Gamma = \partial\Omega$ is denoted by $H_0^1(\Omega)$, and the space of functions in $H^1(\Omega)$ that satisfy the Dirichlet boundary conditions on Γ^D is denoted by $H_D^1(\Omega)$. The dual space of $H^1(\Omega)$ is denoted by $H^{-1}(\Omega)$. Similar notations hold for vector-valued functions, albeit using bold symbols. At every $t \in (0, T]$, the velocity and pressure in the INS equations belong, respectively, to the spaces $\boldsymbol{\mathcal{V}} \coloneqq \boldsymbol{H}_D^1(\Omega)$ and $\mathcal{P} \coloneqq L^2(\Omega)/\mathbb{R}$, and the corresponding test functions used in the weak form belong to $\boldsymbol{\mathcal{W}} \coloneqq \boldsymbol{H}_0^1(\Omega)$ and $\mathcal{Q} \coloneqq L^2(\Omega)/\mathbb{R}$, respectively. With these definitions at hand, the weak form of the problem reads:

At every $t \in (0, T]$, find $[\boldsymbol{v}, p] \in \boldsymbol{\mathcal{V}} \times \mathcal{P}$ such that

$$\begin{aligned} (\boldsymbol{w}, \rho\partial_t \boldsymbol{v})_\Omega + (\boldsymbol{w}, \rho(\boldsymbol{v}\cdot\nabla)\boldsymbol{v})_\Omega + (\nabla^s \boldsymbol{w}, 2\mu\nabla^s \boldsymbol{v})_\Omega - (\nabla\cdot\boldsymbol{w}, p)_\Omega &= (\boldsymbol{w}, \boldsymbol{f})_\Omega, \\ (q, \nabla\cdot\boldsymbol{v})_\Omega &= 0, \end{aligned} \tag{2.5}$$

for every $[\boldsymbol{w}, q] \in \boldsymbol{\mathcal{W}} \times \mathcal{Q}$, while satisfying the initial and boundary conditions. If Eq. 2.5 is discretized using the standard (Bubnov-) Galerkin Finite element method, the discrete solutions $\boldsymbol{v}^h$ and p^h belong to the finite-dimensional spaces $\boldsymbol{\mathcal{V}}^h \subset \boldsymbol{\mathcal{V}}$ and $\mathcal{P}^h \subset \mathcal{P}$, respectively. The semi-discrete problem reads:

At every $t \in (0, T]$, find $[\boldsymbol{v}^h, p^h] \in \boldsymbol{\mathcal{V}}^h \times \mathcal{P}^h$ such that

$$\begin{aligned}
&\left(\boldsymbol{w}^h, \rho \partial_t \boldsymbol{v}^h\right)_\Omega + \left(\boldsymbol{w}^h, \rho\left(\boldsymbol{v}^h \cdot \nabla\right)\boldsymbol{v}^h\right)_\Omega + \left(\nabla^s \boldsymbol{w}^h, 2\mu \nabla^s \boldsymbol{v}^h\right)_\Omega \\
&- \left(\nabla \cdot \boldsymbol{w}^h, p^h\right)_\Omega = \left(\boldsymbol{w}^h, \boldsymbol{f}\right)_\Omega, \\
&\left(q^h, \nabla \cdot \boldsymbol{v}^h\right)_\Omega = 0,
\end{aligned} \tag{2.6}$$

for every $[\boldsymbol{w}^h, q^h] \in \boldsymbol{\mathcal{W}}^h \times \mathcal{Q}^h$, while satisfying the initial and boundary conditions.

However, the standard Galerkin Finite element formulation is generally not stable for INS equations and requires stabilization. The stabilized formulation in the present work is based on the VMS framework and follows closely the formulation presented in [35] and originally introduced in [36, 37, 6]. For more references on the origins and derivation of different VMS formulations, the interested readers are referred to [38, 39] and the references therein. In the following, the VMS formulation used in this work is concisely introduced.

2.1.2.1 Variational multiscale method

In the VMS method with a two-scale decomposition, the velocity and pressure are decomposed into *large-scale* (also called *coarse-scale*) and *subgrid-scale* (also called *fine-scale*, *subscale*, *small-scale*) components

$$\begin{aligned}
\boldsymbol{v} &= \boldsymbol{v}^h + \boldsymbol{v}', \\
p &= p^h + p',
\end{aligned} \tag{2.7}$$

where $\boldsymbol{v}^h$ and p^h represent the large-scale component of the velocity and pressure resolved by the finite elements, while $\boldsymbol{v}'$ and p' represent the subgrid-scale components of the velocity and pressure that are unresolved by the finite elements. Accordingly, the functional spaces of the velocity and pressure are also decomposed into large and fine-scale components

$$\begin{aligned}
\boldsymbol{\mathcal{V}} &= \boldsymbol{\mathcal{V}}^h \oplus \boldsymbol{\mathcal{V}}', \\
\mathcal{P} &= \mathcal{P}^h \oplus \mathcal{P}',
\end{aligned} \tag{2.8}$$

where $\boldsymbol{\mathcal{V}}^h$ and $\mathcal{P}^h$ are the finite-dimensional space of the function used for the discretization of the problem, whereas $\boldsymbol{\mathcal{V}}'$ and $\mathcal{P}'$ are the infinite-dimensional spaces of the velocity and pressure fine scales. A similar decomposition is done for the spaces of the test functions. Using the introduced scale decomposition, the continuous problem, Eq. 2.5, can be split into the large-scale and fine-scale problems as follows:

At every $t \in (0,T]$, find $[\boldsymbol{v}^h, p^h] \in \boldsymbol{\mathcal{V}}^h \times \mathcal{P}^h$ and $[\boldsymbol{v}', p'] \in \boldsymbol{\mathcal{V}}' \times \mathcal{P}'$ such that

$$\begin{aligned}
&\left(\boldsymbol{w}^h, \rho\partial_t(\boldsymbol{v}^h + \boldsymbol{v}')\right)_\Omega + \left(\boldsymbol{w}^h, \rho\left(\boldsymbol{v}^a \cdot \nabla\right)(\boldsymbol{v}^h + \boldsymbol{v}')\right)_\Omega \\
&+ \left(\nabla^s \boldsymbol{w}^h, 2\mu\nabla^s(\boldsymbol{v}^h + \boldsymbol{v}')\right)_\Omega - \left(\nabla \cdot \boldsymbol{w}^h, p^h + p'\right)_\Omega = \left(\boldsymbol{w}^h, \boldsymbol{f}\right)_\Omega, \\
&\left(q^h, \nabla \cdot (\boldsymbol{v}^h + \boldsymbol{v}')\right)_\Omega = 0,
\end{aligned} \tag{2.9}$$

and

$$\begin{aligned}
&\left(\boldsymbol{w}', \rho\partial_t(\boldsymbol{v}^h + \boldsymbol{v}')\right)_\Omega + \left(\boldsymbol{w}', \rho\left(\boldsymbol{v}^a \cdot \nabla\right)(\boldsymbol{v}^h + \boldsymbol{v}')\right)_\Omega \\
&- \left(\boldsymbol{w}', 2\mu\nabla \cdot \left(\nabla^s(\boldsymbol{v}^h + \boldsymbol{v}')\right)\right)_\Omega + \left(\boldsymbol{w}', \nabla(p^h + p')\right)_\Omega = \left(\boldsymbol{w}', \boldsymbol{f}\right)_\Omega, \\
&\left(q', \nabla \cdot (\boldsymbol{v}^h + \boldsymbol{v}')\right)_\Omega = 0,
\end{aligned} \tag{2.10}$$

for every $[\boldsymbol{w}^h, q^h] \in \boldsymbol{\mathcal{W}}^h \times \mathcal{Q}^h$ and $[\boldsymbol{w}', q'] \in \boldsymbol{\mathcal{W}}' \times \mathcal{Q}'$. The convective velocity in the second term of Eqs. 2.9 and 2.10 is shown by $\boldsymbol{v}^a$, which may either take the form $\boldsymbol{v}^a = \boldsymbol{v}^h + \boldsymbol{v}'$ or $\boldsymbol{v}^a = \boldsymbol{v}^h$. Each choice for $\boldsymbol{v}^a$ leads to a different VMS formulation. The former choice of $\boldsymbol{v}^a$ leads to the so-called nonlinear scale splitting (or nonlinear subgrid scales) [39], while the latter leads to linear scale splitting (or linear subgrid scales). The nonlinear scale splitting is used in [40, 41, 37, 6, 13], among others, which opens the doors to modeling turbulence using the VMS framework.

After Integrating by parts the terms in Eq. 2.9 that contain derivatives of the subgrid scales, Eq. 2.9 can be reformulated as

$$\begin{aligned}
&\left(\boldsymbol{w}^h, \rho\partial_t \boldsymbol{v}^h\right)_\Omega + \left(\boldsymbol{w}^h, \rho\left(\boldsymbol{v}^a \cdot \nabla\right)\boldsymbol{v}^h\right)_\Omega + \left(\nabla^s \boldsymbol{w}^h, 2\mu\nabla^s \boldsymbol{v}^h\right)_\Omega - \left(\nabla \cdot \boldsymbol{w}^h, p^h\right)_\Omega \\
&+ \left(q^h, \nabla \cdot \boldsymbol{v}^h\right)_\Omega + \left(\boldsymbol{w}^h, \rho\partial_t \boldsymbol{v}'\right)_\Omega - \left(\rho\left(\boldsymbol{v}^a \cdot \nabla\right)\boldsymbol{w}^h + 2\mu\nabla \cdot \left(\nabla^s \boldsymbol{w}^h\right) + \nabla q^h, \boldsymbol{v}'\right)_{\Omega^K} \\
&- \left(\nabla \cdot \boldsymbol{w}^h, p'\right)_{\Omega^K} = \left(\boldsymbol{w}^h, \boldsymbol{f}\right)_\Omega.
\end{aligned} \tag{2.11}$$

In deriving Eq. 2.11, it is assumed that the subgrid scales vanish on the boundaries of the elements and the boundary terms arising from the integration by parts are equal to zero. Furthermore, The following short notation is used: $(\cdot,\cdot)_{\Omega^K} = \sum_{K=1}^{ne} (\cdot,\cdot)_{\Omega^K}$, where Ω^K denotes the interior of the element K and ne is the total number of the elements in the mesh.

Eq. 2.11 governs the evolution of large-scale quantities. The left-hand-side terms on the second line of equation Eq. 2.11 describe the action of the subgrid scales on the large-scale quantities. A concrete definition of the fine-scale velocity and pressure is needed in order to close the large-scale problem. Those definitions are sought in the solution of the fine-scale problem, Eq. 2.10. However, Eq. 2.10 is defined in the infinite-dimensional space of the fine scales and, therefore, can not be solved directly. Instead, the fine-scale

velocity and pressure in each element are modeled as

$$\begin{aligned} \partial_t \boldsymbol{v}' + (\boldsymbol{\tau}^v)^{-1} \boldsymbol{v}' &= \Pi(\boldsymbol{r}^m), \\ (\tau^p)^{-1} p' &= \Pi(r^c), \end{aligned} \tag{2.12}$$

where $\boldsymbol{\tau}^v$ and τ^p are the so-called stabilization parameters, Π denotes a projection onto the space of the subgrid scales, which will be discussed shortly (the same symbol is used here for the projection of the scalar-valued and vector-valued functions), $\boldsymbol{r}^m$ and r^c are the strong-form residuals of the coarse scale momentum and the continuity equations, defined as

$$\begin{aligned} \boldsymbol{r}^m &= \boldsymbol{f} - \rho \partial_t \boldsymbol{v}^h - \rho (\boldsymbol{v}^a \cdot \nabla) \boldsymbol{v}^h + 2\mu \nabla \cdot \left(\nabla^s \boldsymbol{v}^h \right) - \nabla p^h, \\ r^c &= -\nabla \cdot \boldsymbol{v}^h. \end{aligned} \tag{2.13}$$

The stabilization parameters $\boldsymbol{\tau}^v$ and τ^p play a central role in the stabilized formulations, and their inverses approximate the action of the spatial operators acting on the fine scales in Eq. 2.10. In [33, 34, 42], it is shown that the stabilization parameters represent local approximations to the fine-scale Green's operator. In practice, in particular, for the INS equations, the definitions of the stabilization parameters are derived using asymptotic scaling arguments [13]. In [37, 39], an approximate Fourier analysis is used to motivate the design of the stabilization parameters. Following [43, 37], $\boldsymbol{\tau}^v$ is defined in each element as a diagonal matrix: $\boldsymbol{\tau}^v = \mathrm{diag}(\tau^v)$. Furthermore, τ^v and τ^p are defined as

$$\tau^v = \left(\frac{c_1 \mu}{h^2} + \frac{c_2 \rho \| \boldsymbol{v}^a \|}{h} \right)^{-1}, \tag{2.14}$$

$$\tau^p = \frac{h^2}{c_1 \tau^v}, \tag{2.15}$$

where c_1 and c_2 are algorithmic constants usually defined as $c_1 = 4$ and $c_2 = 2$ for elements with linear shape functions, h is the characteristic length of each element, and $\| \boldsymbol{v}^a \|$ is the norm of the advective velocity. There are other definitions for the stabilization parameters available in the literature, e.g., in [13], which typically have similar asymptotic behavior with respect to h, ρ, μ, and $\| \boldsymbol{v}^a \|$. Following [44], the values of c_1 and c_2 for higher-order elements are replaced by $c_1 r^2$ and $c_2 r$, respectively, where r is the interpolation order of the elements. In the present work, linear and quadratic elements are used. It is worth noting that there are other definitions for c_1 and c_2 for higher-order elements, e.g., in [45], where c_1 is taken to be proportional to r^4 (See also [44] for a discussion in this regard).

The inclusion of the $\partial_t \boldsymbol{v}'$ term in Eq. 2.12 is introduced in [37, 6] and leads to the so-called dynamic subscales. In a dynamic subscale formulation, the velocity subscales are tracked in time, and the effect of the time dependency

of the subscales on the large-scale problem is also taken into account through the $\rho\left(\boldsymbol{w}^h, \partial_t \boldsymbol{v}'\right)_\Omega$ term in Eq. 2.11. In [6], it is shown that dynamic subscales lead to the correct behavior of the time integration scheme, and allow to use arbitrary small time steps regardless of the size of the spatial discretization. Dropping the $\partial_t \boldsymbol{v}'$ term in Eq. 2.12, and subsequently, the $\rho\left(\boldsymbol{w}^h, \partial_t \boldsymbol{v}'\right)_\Omega$ term in Eq. 2.11, leads to quasi-static subscales [37, 6], which is the formulation used classically in the stabilized methods.

If the projection Π in Eq. 2.12 is taken to be $\boldsymbol{I}$ (unity), the so-called algebraic subgrid-scale (ASGS) formulation arises [43]. This choice implies that the subgrid scales belong to the space of the large-scale residuals. ASGS is the *classical* choice in the stabilized formulations. Another option for the projection Π is presented in [36, 37], where the space of the subgrid scales is chosen to be orthogonal to the finite element space (where the large-scale quantities are defined). Therefore Π is taken as $\Pi_h^\perp = \boldsymbol{I} - \Pi_h$, where Π_h represents the L^2 projection onto the proper finite element space and $\Pi_h^\perp$ is its orthogonal projection. This formulation is called the orthogonal subgrid-scale (OSS) method. Considering the above-mentioned choices of Π, Eq. 2.12 can be rewritten as

$$\partial_t \boldsymbol{v}' + \left(\boldsymbol{\tau}^v\right)^{-1} \boldsymbol{v}' = \boldsymbol{r}^m - \boldsymbol{\xi}^h, \tag{2.16}$$

$$\left(\tau^p\right)^{-1} p' = r^c - \delta^h, \tag{2.17}$$

where $\boldsymbol{\xi}^h$ and δ^h are equal to zero in the ASGS formulation, and in the case of OSS formulation, $\boldsymbol{\xi}^h$ and δ^h represent, respectively, the projection of the residuals $\boldsymbol{r}^m$ and r^c onto the space of finite elements:

$$\begin{aligned} \left(\boldsymbol{w}^h, \boldsymbol{\xi}^h\right)_{\Omega^K} &= \left(\boldsymbol{w}^h, \boldsymbol{r}^m\right)_{\Omega^K} \qquad && \forall \boldsymbol{w}^h \in \boldsymbol{\mathcal{W}}^h, \\ \left(q^h, \delta^h\right)_{\Omega^K} &= \left(q^h, r^c\right)_{\Omega^K} && \forall q^h \in \mathcal{Q}^h. \end{aligned} \tag{2.18}$$

If the space of the subgrid scales is orthogonal to the finite element space, simplifications can be made in $\boldsymbol{r}^m$ and r^c compared to the definitions provided in Eq. 2.13. It is immediately apparent that the projection of the term $\rho \partial_t \boldsymbol{v}^h$ onto the space of the subgrid scales is zero due to the choice of orthogonal subgrid scales. Furthermore, in the case of minimal regularity, the body force term $\boldsymbol{f}$ in Eq. 2.13 can be neglected without upsetting the order of accuracy of the method [37]. In [37, 46], it is suggested that dropping the term $2\mu \nabla \cdot \left(\nabla^s \boldsymbol{v}^h\right)$ in Eq. 2.13 leads to a formulation that is still consistent when using orthogonal subgrid scales (see remark 4 in [37]). Note that using linear elements, the terms with second spatial derivatives are exactly zero in any case. As a side note, it should be remarked that by virtue of the incompressibility constraint $\nabla \cdot \boldsymbol{v}^h = 0$, the term $2\mu \nabla \cdot \left(\nabla^s \boldsymbol{v}^h\right)$ in Eq. 2.13 is usually replaced by $\Delta \boldsymbol{v}^h$ in practice, where Δ is the Laplacian operator (see [13, 37]). Another direct consequence of the choice of orthogonal subgrid scales is that the term $\rho\left(\boldsymbol{w}^h, \partial_t \boldsymbol{v}'\right)_\Omega$ in Eq. 2.11 is equal to zero. It should be remarked that in

order to get truly orthogonal subgrid scales, the projections in Eq. 2.18 must be weighted by the local stabilization parameters [37]. Therefore, the L^2 projections used in Eq. 2.18 are generally an approximation.

2.1.3 G-α scheme for the first-order systems

The G-α scheme, hereafter refereed to as the G-α scheme, was originally proposed in [47] for the time integration of the second-order ordinary differential equations (ODE) in structural mechanics. The extension of the G-α scheme to the first-order ODEs, in particular to the equations arising from the spatial discretization of the compressible Navier-Stokes equations, was later proposed in [12]. It has since become a popular choice in fluid dynamics and fluid-structure interaction (FSI) simulations, due to its second-order accuracy, unconditional stability for linear systems and flexibility in choosing the amount of the desired high-frequency dissipation introduced by the time integration scheme.

Let the time interval $[0,T]$ be divided into N_t subintervals, with the discrete points in time $\{t_n\}_{n=0}^{N_t}$, and let $\Delta t_n = t_{n+1} - t_n$ be the time step size in the nth interval. Let us introduce a model first-order ODE which facilitates discussing some of the concepts discussed in the following. The model problem reads:

$$\begin{aligned} \partial_t v - \lambda v = 0, \quad & t \in (0,T], \\ v = v_0, \quad & t = 0, \end{aligned} \tag{2.19}$$

where v is a scalar variable and λ is a constant scalar. The G-α scheme satisfies the equilibrium equation of the problem in an average sense in every time step. Integrating the model problem, Eq. 2.19, using the G-α scheme leads to the following average equilibrium equation in the nth time step:

$$\tilde{\dot{v}}_{n+\alpha_m} - \lambda v_{n+\alpha_f} = 0, \tag{2.20}$$

with α_f and α_m being algorithmic parameters of the G-α scheme, and $v_{n+\alpha_f}$ and $\tilde{\dot{v}}_{n+\alpha_m}$ (the reason for using $\tilde{}$ will be made clear later) representing the generalized midpoint values of the variable v and its time derivative, respectively:

$$\begin{aligned} v_{n+\alpha_f} &= (1-\alpha_f)\, v_n + \alpha_f v_{n+1}, \\ \tilde{\dot{v}}_{n+\alpha_m} &= (1-\alpha_m)\, \tilde{\dot{v}}_n + \alpha_m \tilde{\dot{v}}_{n+1}. \end{aligned} \tag{2.21}$$

The variable v and its time derivative $\tilde{\dot{v}}$ are related to each other using the following (Newmark) update formula:

$$v_{n+1} = v_n + \Delta t \left((1-\gamma)\, \tilde{\dot{v}}_n + \gamma \tilde{\dot{v}}_{n+1} \right), \tag{2.22}$$

where γ is another algorithmic parameter of the G-α scheme.

The generalized midpoint values $v_{n+\alpha_f}$ and $\tilde{\dot{v}}_{n+\alpha_m}$ correspond, respectively,

to the intermediate time instants $t_{n+\alpha_f}$ and $t_{n+\alpha_m}$, within the nth time step:

$$\begin{aligned} t_{n+\alpha_f} &= (1-\alpha_f)\, t_n + \alpha_f t_{n+1}, \\ t_{n+\alpha_m} &= (1-\alpha_m)\, t_n + \alpha_m t_{n+1}. \end{aligned} \tag{2.23}$$

Note in Eq. 2.20 that the term containing the discrete time derivative of v is evaluated at $t_{n+\alpha_m}$ and the term containing the variable v (and, in general, any other possible term except those containing the time derivative) is evaluated at $t_{n+\alpha_f}$.

In order to achieve second-order accuracy in time, unconditional stability and optimal amount of numerical dissipation, the parameters α_f, α_m and γ can not be chosen arbitrarily and are dependent on each other. It is shown in [12] that second-order accuracy in time is for the variable v is guaranteed if

$$\gamma = \frac{1}{2} + \alpha_m - \alpha_f, \tag{2.24}$$

and unconditional stability is achieved if

$$\alpha_m \geqslant \alpha_f \geqslant \frac{1}{2}. \tag{2.25}$$

The amount of the high-frequency numerical dissipation is conveniently controlled by the user-controlled parameter $\rho_\infty \in [0,1]$ which is the spectral radius of the amplification matrix of the scheme as $\Delta t \to \infty$. Having decided about the desired amount of the high-frequency dissipation through ρ_∞, the parameters α_f, α_m and γ are determined as [12]

$$\alpha_f = \frac{1}{1+\rho_\infty}, \quad \alpha_m = \frac{1}{2}\left(\frac{3-\rho_\infty}{1+\rho_\infty}\right), \quad \gamma = \frac{1}{2} + \alpha_m - \alpha_f. \tag{2.26}$$

Instead of the linear initial value problem (IVP) represented by Eq. 2.19, many problems of interest are governed by a nonlinear IVP

$$\begin{aligned} \partial_t v - \mathcal{N}(v) = 0, \quad t \in (0, T], \\ v = v_0, \quad t = 0, \end{aligned} \tag{2.27}$$

where $\mathcal{N}(v)$ is any nonlinear operator depending on v (which itself is time dependent). Considering the integration of the nonlinear term using the G-α scheme, there are two apparent choices for the quadrature rule conceivable: the generalized midpoint quadrature rule (GMQ) and the generalized trapezoidal quadrature rule (GTQ) leading, respectively, to

$$\mathcal{N}^{GMQ}_{n+\alpha_f}(\alpha_f, v_n, v_{n+1}) = \mathcal{N}(v_{n+\alpha_f}), \tag{2.28}$$

$$\mathcal{N}^{GTQ}_{n+\alpha_f}(\alpha_f, v_n, v_{n+1}) = (1-\alpha_f)\,\mathcal{N}(v_n) + \alpha_f \mathcal{N}(v_{n+1}). \tag{2.29}$$

For linear systems, the generalized midpoint and generalized trapezoidal quadrature rules lead to the same results. However, the two lead to (usually slightly) different results when treating nonlinear problems. The role and effect of the aforementioned quadrature rules in the time integration of the Navier-Stokes equations will be discussed in the following. The naming convention used in this work for the quadrature rules in Eqs. 2.28–2.29 is adopted from [48], with an additional emphasis on the *quadrature* (in time) for the nonlinear terms. We do so, in order to remove the ambiguity between the quadrature rules for nonlinear terms in the context of the G-α family of time integration schemes, as used in the present work, with the use of "generalized midpoint rule" and "generalized trapezoidal rule" in, e.g., [5, 37], which are other names used for the one-step-θ scheme. It is worth mentioning that in [49, 50], quadrature rules similar to those presented in Eqs. 2.28–2.29 are refereed to as the *tangent* and *secant* versions, when discretizing nonlinear terms in the Navier-Stokes equations using the one-step-θ family of time integration schemes.

2.1.4 G-α scheme for the incompressible Navier-Stokes equations

The application of the G-α scheme to the INS equations requires the evaluation of intermediate values for the velocity, pressure, and acceleration variables

$$\boldsymbol{v}^h_{n+\alpha_f} = (1-\alpha_f)\,\boldsymbol{v}^h_n + \alpha_f \boldsymbol{v}^h_{n+1}, \tag{2.30}$$

$$p^h_{n+\alpha_f} = (1-\alpha_f)\,p^h_n + \alpha_f p^h_{n+1}, \tag{2.31}$$

$$\tilde{\boldsymbol{v}}^h_{n+\alpha_m} = (1-\alpha_m)\,\tilde{\boldsymbol{v}}^h_n + \alpha_m \tilde{\boldsymbol{v}}^h_{n+1}. \tag{2.32}$$

Additionally, the velocity and the acceleration variables are related through the following formula:

$$\boldsymbol{v}^h_{n+1} = \boldsymbol{v}^h_n + \Delta t\left((1-\gamma)\,\tilde{\boldsymbol{v}}^h_n + \gamma \tilde{\boldsymbol{v}}^h_{n+1}\right). \tag{2.33}$$

Following the G-α rules presented in section 2.1.3, the INS equations are satisfied in an average sense in the nth time step and read in the time-discrete form as

$$\begin{aligned}
&\left(\boldsymbol{w}^h, \rho\tilde{\boldsymbol{v}}^h_{n+\alpha_m}\right)_\Omega + \left(\boldsymbol{w}^h, \rho(\boldsymbol{v}^a_{n+\alpha_f}\cdot\nabla)\boldsymbol{v}^h_{n+\alpha_f}\right)_\Omega + \left(\nabla^s \boldsymbol{w}^h, 2\mu\nabla^s \boldsymbol{v}^h_{n+\alpha_f}\right)_\Omega \\
&- \left(\nabla\cdot\boldsymbol{w}^h, p^h_{n+\alpha_1}\right)_\Omega + \left(q^h, \nabla\cdot\boldsymbol{v}^h_{n+\alpha_2}\right)_\Omega + \left(\mathcal{L}^*_{n+\alpha_f}\left(\boldsymbol{w}^h, q^h\right), \boldsymbol{v}'_{n+\alpha_f}\right)_{\Omega^K} \\
&- \left(\nabla\cdot\boldsymbol{w}^h, p'_{n+\alpha_f}\right)_{\Omega^K} = \left(\boldsymbol{w}^h, \boldsymbol{f}_{n+\alpha_f}\right)_\Omega,
\end{aligned} \tag{2.34}$$

where

$$\mathcal{L}^*_{n+\alpha_f}\left(\boldsymbol{w}^h, q^h\right) = -\left(\rho(\boldsymbol{v}^a_{n+\alpha_f}\cdot\nabla)\boldsymbol{w}^h + 2\mu\nabla\cdot\left(\nabla^s\boldsymbol{w}^h\right) + \nabla q^h\right), \tag{2.35}$$

$$\boldsymbol{v}'_{n+\alpha_f} = \tau^v_{n+\alpha_f} \boldsymbol{r}^m_{n+\alpha_{f,m}}, \tag{2.36}$$

$$p'_{n+\alpha_f} = \tau^p_{n+\alpha_f} r^c_{n+\alpha_2}, \tag{2.37}$$

$$\tau^v_{n+\alpha_f} = \left(\frac{c_1 \mu}{h^2} + \frac{c_2 \rho \| \boldsymbol{v}^a_{n+\alpha_f} \|}{h} \right)^{-1}, \tag{2.38}$$

$$\tau^p_{n+\alpha_f} = \frac{h^2}{c_1 \tau^v_{n+\alpha_f}}, \tag{2.39}$$

$$\boldsymbol{r}^m_{n+\alpha_{f,m}} = \boldsymbol{f}_{n+\alpha_f} - \rho \dot{\tilde{\boldsymbol{v}}}^h_{n+\alpha_m} - \rho \left(\boldsymbol{v}^a_{n+\alpha_f} \cdot \nabla \right) \boldsymbol{v}^h_{n+\alpha_f} + 2\mu \nabla \cdot \left(\nabla^s \boldsymbol{v}^h_{n+\alpha_f} \right) - \nabla p^h_{n+\alpha_1}, \tag{2.40}$$

$$r^c_{n+\alpha_2} = -\nabla \cdot \boldsymbol{v}^h_{n+\alpha_2}. \tag{2.41}$$

Note the use of α_1 and α_2 in the subscripts of the terms $p^h_{n+\alpha_1}$ and $\nabla \cdot \boldsymbol{v}^h_{n+\alpha_2}$ in Eqs. 2.34, 2.37, 2.40, and 2.41, which indicate the evaluation of the terms $p^h_{n+\alpha_1}$ and $\nabla \cdot \boldsymbol{v}^h_{n+\alpha_2}$ at time instants yet to be defined. Different choices for α_1 and α_2 lead to different variants of the formulation, which differ only in the temporal evaluations of at least one of the terms included in the pair $\left\{ p^h_{n+\alpha_1}, \nabla \cdot \boldsymbol{v}^h_{n+\alpha_2} \right\}$. Note also the special case in the subscript of the term $\boldsymbol{r}^m_{n+\alpha_{f,m}}$ in Eqs. 2.38 and 2.40, where it indicates that $\boldsymbol{r}^m_{n+\alpha_{f,m}}$ contains a term which is evaluated at the intermediate time instant $t_{n+\alpha_m}$ and other terms which are evaluated at the intermediate time instant $t_{n+\alpha_f}$. At first sight, the simultaneous presence of the terms evaluated at $t_{n+\alpha_m}$ and $t_{n+\alpha_f}$ renders using a generalized trapezoidal quadrature rule impossible, a topic which will be discussed later.

The issue surrounding the temporal evaluation of the terms $p^h_{n+\alpha_1}$ and $\nabla \cdot \boldsymbol{v}^h_{n+\alpha_2}$, when using time integration schemes that require evaluation at intermediate time instants, arises from the differential-algebraic nature of the INS equations due to the presence of the incompressibility constraint and the role of pressure field as the Lagrange multiplier in imposing this condition. It is often argued that the pressure should not be subjected to time integration due to its role as Lagrange multiplier. Furthermore, It has been suggested that including contributions from the velocity at previous time steps in the incompressibility constraint should be avoided. That is to avoid the propagation of the errors in the incompressibility constraint due to the nonsolenoidal initial velocity field or not fully satisfying the incompressibility constraint in the previous time steps. These considerations prompt the evaluation of pressure term and incompressibility constraint at the end of the time step, where unknowns are sought, i.e., choosing $\alpha_1 = \alpha_2 = 1$ and replacing the pair $\left\{ p^h_{n+\alpha_1}, \nabla \cdot \boldsymbol{v}^h_{n+\alpha_2} \right\}$ in Eq. 2.34– 2.41 with $\left\{ p^h_{n+1}, \nabla \cdot \boldsymbol{v}^h_{n+1} \right\}$. This approach can be interpreted as applying the G-α scheme only to the terms of the momentum equation containing velocities, and treating the DAE aspects of the INS equations fully implicitly as done, e.g., in the backward

Euler time integration scheme.

On the other hand, it is possible to apply the G-α scheme, initially conceived for ODEs, without distinction to all terms of the INS equations that constitute the DAE system. Following the approach presented in [2] for the application of the Trapezoidal rule to a model DAE system, the discrete INS equations (without stabilization, for simplicity) are written in the so-called time-singular form

$$\begin{bmatrix} \boldsymbol{M} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \partial_t \boldsymbol{V} \\ \partial_t \boldsymbol{P} \end{bmatrix} + \begin{bmatrix} \boldsymbol{N} + \boldsymbol{A} & -\boldsymbol{B} \\ \boldsymbol{B}^T & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{V} \\ \boldsymbol{P} \end{bmatrix} = \begin{bmatrix} \boldsymbol{F} \\ 0 \end{bmatrix}, \tag{2.42}$$

where $\boldsymbol{M}$, $\boldsymbol{N}$, $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{F}$ represent the discrete counterparts of the inertial, (nonlinear) convection, diffusion, pressure, and right-hand side terms, respectively. Eq. 2.42 can be rewritten as

$$\boldsymbol{\mathcal{M}} \partial_t \boldsymbol{Y} + \boldsymbol{\mathcal{C}} \boldsymbol{Y} = \boldsymbol{\mathcal{F}}, \tag{2.43}$$

where

$$\boldsymbol{Y} = \begin{bmatrix} \boldsymbol{V} \\ \boldsymbol{P} \end{bmatrix}, \quad \boldsymbol{\mathcal{M}} = \begin{bmatrix} \boldsymbol{M} & 0 \\ 0 & 0 \end{bmatrix}, \quad \boldsymbol{\mathcal{C}} = \begin{bmatrix} \boldsymbol{N} + \boldsymbol{A} & -\boldsymbol{B} \\ \boldsymbol{B}^T & 0 \end{bmatrix}, \quad \boldsymbol{\mathcal{F}} = \begin{bmatrix} \boldsymbol{F} \\ 0 \end{bmatrix}. \tag{2.44}$$

The application of the G-α scheme to Eq. 2.43 is straightforward and leads to the time-discrete equilibrium equation

$$\boldsymbol{\mathcal{M}} \tilde{\dot{\boldsymbol{Y}}}_{n+\alpha_m} + \boldsymbol{\mathcal{C}} \boldsymbol{Y}_{n+\alpha_f} = \boldsymbol{\mathcal{F}}_{n+\alpha_f}. \tag{2.45}$$

Considering the relation of Eq. 2.45 to Eq. 2.42 and Recalling that the term $\boldsymbol{\mathcal{C}} \boldsymbol{Y}_{n+\alpha_f}$ contains the pressure term and the incompressibility constraint from the INS equations, it is evident that the mentioned terms should be evaluated at the intermediate time instant $t_{n+\alpha_f}$, which leads to a natural extension of the G-α scheme to the DAE system of the INS equations. Using this point of view, the application of the G-α scheme to the VMS formulation presented in Eq. 2.34– 2.41 prompts choosing $\alpha_1 = \alpha_2 = \alpha_f$ and replacing the terms in the pair $\left\{p^h_{n+\alpha_1}, \nabla \cdot \boldsymbol{v}^h_{n+\alpha_2}\right\}$ with $\{p^h_{n+\alpha_f}, \nabla \cdot \boldsymbol{v}^h_{n+\alpha_f}\}$, where $p^h_{n+\alpha_f}$ and $\boldsymbol{v}^h_{n+\alpha_f}$ are defined by Eqs. 2.31 and 2.30, respectively. Note that using $p^h_{n+\alpha_f}$ raises the need for proper initial value for pressure and storage of the old pressure values in each time step.

The two versions of the G-α scheme for INS equations with $\{p^h_{n+1}, \nabla \cdot \boldsymbol{v}^h_{n+1}\}$ and $\{p^h_{n+\alpha_f}, \nabla \cdot \boldsymbol{v}^h_{n+\alpha_f}\}$ are compared in [16] using inf-sup stable Taylor-Hood elements. It is shown in [16] that the version with the pair $\{p^h_{n+1}, \nabla \cdot \boldsymbol{v}^h_{n+1}\}$ leads to a first-order temporal approximation for pressure, which is a reduction from the theoretical second-order accuracy, while using the pair $\{p^h_{n+\alpha_f}, \nabla \cdot \boldsymbol{v}^h_{n+\alpha_f}\}$ retains the expected second-order accuracy for both velocity and pressure, as underlined in [14] as well.

Another consequence of the choice between p^h_{n+1} and $p^h_{n+\alpha_f}$ in the time

discretization of the INS equations is related to the evaluation of the integrated reaction force and the corresponding normal traction (flux) on certain parts of the boundary with Dirichlet boundary conditions. The integrated reaction force on the boundary is used, e.g., for calculating the drag and lift forces on the objects in a flow, and normal traction is required in, e.g., fluid-structure interaction simulations. We take a short detour to explain how the integrated boundary tractions are calculated and will then make the connection to the time integration of the INS equations using the G-α scheme and the choice between p^h_{n+1} and $p^h_{n+\alpha_f}$.

After solving the INS equations for $\boldsymbol{v}^h$ and p^h, there are two possible approaches for calculating the total reaction force on a boundary Γ^s with a strong homogeneous Dirichlet boundary condition. The first possibility is to use the solution to $\boldsymbol{v}^h$ and p^h to calculate the total reaction force acting on the subset of the boundary denoted by Γ^s using *direct evaluation* of the integral

$$F = \int_{\Gamma^s} \left(2\mu\left(\nabla^s \boldsymbol{v}^h\right) - p^h \boldsymbol{I}\right) \cdot \boldsymbol{n} \, \mathrm{d}\Gamma^s. \tag{2.46}$$

Let $\boldsymbol{n}^D$ and $\boldsymbol{n}^L$ be unit vectors denoting the direction of the incoming flow and its normal, respectively (for simplicity assume $\boldsymbol{n}^D = [1,0,0]^T$ and $\boldsymbol{n}^L = [0,1,0]^T$ in 3D). The drag and lift forces F^D and F^L are calculated using the direct evaluation approach as

$$F^D = \int_{\Gamma^s} \left(\left(2\mu\left(\nabla^s \boldsymbol{v}^h\right) - p^h \boldsymbol{I}\right) \cdot \boldsymbol{n}\right) \cdot \boldsymbol{n}^D \, \mathrm{d}\Gamma^s, \tag{2.47}$$

$$F^L = \int_{\Gamma^s} \left(\left(2\mu\left(\nabla^s \boldsymbol{v}^h\right) - p^h \boldsymbol{I}\right) \cdot \boldsymbol{n}\right) \cdot \boldsymbol{n}^L \, \mathrm{d}\Gamma^s. \tag{2.48}$$

Instead of the direct evaluation of the total reaction force as in Eq. 2.46, one can calculate the so-called *variationally consistent* reaction force using the information already contained in the variational form of the solved problem [51, 52, 53, 54, 55, 56, 57, 58, 59]. The variationally consistent total reaction force on the boundary Γ^s with strong homogeneous Dirichlet boundary condition is calculated as

$$\begin{aligned} F = \left(\boldsymbol{w}^h, \rho\partial_t \boldsymbol{v}^h\right)_\Omega + n\left(\boldsymbol{w}^h, \boldsymbol{v}^h, \boldsymbol{v}^a\right) + a\left(\boldsymbol{w}^h, \boldsymbol{v}^h\right) - \left(\nabla \cdot \boldsymbol{w}^h, p^h\right)_\Omega + \left(q^h, \nabla \cdot \boldsymbol{v}^h\right)_\Omega \\ + \left(\mathcal{L}^*\left(\boldsymbol{w}^h, q^h\right), \boldsymbol{v}'\right)_{\Omega^K} - \left(\nabla \cdot \boldsymbol{w}^h, p'\right)_{\Omega^K} - \left(\boldsymbol{w}^h, \boldsymbol{f}\right)_\Omega, \end{aligned} \tag{2.49}$$

for arbitrary q^h and $\boldsymbol{w}^h \in \boldsymbol{H}^1(\Omega)$ such that $\boldsymbol{w}^h\big|_{\Gamma^s} = \boldsymbol{w}^s$, where $\boldsymbol{w}^s$ is a known value and $\boldsymbol{w}^h = \boldsymbol{0}$ on all other boundaries except Γ^s. The drag and lift forces F^D and F^L ensue from Eq. 2.49 by setting $\boldsymbol{w}^s = \boldsymbol{n}^D$ and $\boldsymbol{w}^s = \boldsymbol{n}^L$, respectively. The calculation of the boundary reaction forces (total fluxes in general) through a variationally consistent approach using Eq. 2.49 is shown to possess superior accuracy and robustness compared to the calculation using direct evaluation of integrals as in Eq. 2.46 [51, 52]. The traction on the

boundary Γ^s, denoted by $\boldsymbol{t}^s$, can be derived using the same variationally consistent approach:

$$\left(\boldsymbol{w}^h, \boldsymbol{t}^s\right)_{\Gamma^s} = \left(\boldsymbol{w}^h, \rho\partial_t \boldsymbol{v}^h\right)_\Omega + n\left(\boldsymbol{w}^h, \boldsymbol{v}^h, \boldsymbol{v}^a\right) + a\left(\boldsymbol{w}^h, \boldsymbol{v}^h\right) - \left(\nabla\cdot\boldsymbol{w}^h, p^h\right)_\Omega + \left(q^h, \nabla\cdot\boldsymbol{v}^h\right)_\Omega + \left(\mathcal{L}^*\left(\boldsymbol{w}^h, q^h\right), \boldsymbol{v}'\right)_{\Omega^K} - \left(\nabla\cdot\boldsymbol{w}^h, p'\right)_{\Omega^K} - \left(\boldsymbol{w}^h, \boldsymbol{f}\right)_\Omega, \quad (2.50)$$

for arbitrary q^h and for $\boldsymbol{w}^h \in \boldsymbol{\mathcal{W}}^s(\Omega)$, with $\boldsymbol{\mathcal{W}}^s(\Omega)$ being the complement of $\boldsymbol{\mathcal{W}}(\Omega)$ in $\boldsymbol{H}^1(\Omega)$, that is, $\boldsymbol{\mathcal{W}} \oplus \boldsymbol{\mathcal{W}}^s = \boldsymbol{H}^1$ and $\boldsymbol{\mathcal{W}}(\Omega)$ being the usual space of the test functions with zero traces on the boundary Γ^s for the INS momentum equation. The variationally consistent tractions are conservative quantities if the discrete form of the variational formulation is conservative [52]. However, there is a subtlety when dealing with the INS equations in the advective form. As shown in [60] and later extended in [6] to the case of dynamic subgrid scales in VMS formulations, an inf-sup stable Galerkin formulation for the advective form of the INS equations does not generally conserve the linear momentum. However, using stabilized formulations allows us to retain the momentum-conserving property in the discrete sense if the contribution of the velocity subgrid scale is included in the advective velocity, that is, $\boldsymbol{v}^a = \boldsymbol{v}^h + \boldsymbol{v}'$. The presence of the term $\left(\boldsymbol{w}^h, \rho\left(\boldsymbol{v}'\cdot\nabla\right)\boldsymbol{v}^h\right)_\Omega$ in the weak form is a crucial component in preserving the conservation of momentum [60, 6]. Therefore, using VMS methods with nonlinear scale splitting, i.e., including the contribution of the velocity subgrid scale in the advective velocity, guarantees the variationally consistent tractions in Eq. 2.50 to be truly conserved quantities.

Going back to the topic of time integration of the INS equations using the G-α scheme, choosing to evaluate the pressure terms in Eq. 2.34 at the intermediate time instant $t_{n+\alpha_f}$ leads to a clear interpretation of the time-discretized variationally consistent tractions at the intermediate time instant $t_{n+\alpha_f}$:

$$\left(\boldsymbol{w}^h, \boldsymbol{t}^s_{n+\alpha_f}\right)_{\Gamma^s} = \left(\boldsymbol{w}^h, \rho\tilde{\dot{\boldsymbol{v}}}^h_{n+\alpha_m}\right)_\Omega + n\left(\boldsymbol{w}^h, \boldsymbol{v}^h_{n+\alpha_f}, \boldsymbol{v}^a_{n+\alpha_f}\right) + a\left(\boldsymbol{w}^h, \boldsymbol{v}^h_{n+\alpha_f}\right) - \left(\nabla\cdot\boldsymbol{w}^h, p^h_{n+\alpha_f}\right)_\Omega + \left(q^h, \nabla\cdot\boldsymbol{v}^h_{n+\alpha_2}\right)_\Omega + \left(\mathcal{L}^*_{n+\alpha_f}\left(\boldsymbol{w}^h, q^h\right), \boldsymbol{v}'_{n+\alpha_f}\right)_{\Omega^K} - \left(\nabla\cdot\boldsymbol{w}^h, p'_{n+\alpha_f}\right)_{\Omega^K} - \left(\boldsymbol{w}^h, \boldsymbol{f}_{n+\alpha_f}\right)_\Omega, \quad (2.51)$$

for arbitrary q^h and $\boldsymbol{w}^h$ as defined for Eq. 2.50. The tractions at time instant t_{n+1} are then calculated using the G-α update formula

$$\boldsymbol{t}^s_{n+1} = \frac{1}{\alpha_f}\boldsymbol{t}^s_{n+\alpha_f} - \frac{(1-\alpha_f)}{\alpha_f}\boldsymbol{t}^s_n. \quad (2.52)$$

Note that the calculation of $\boldsymbol{t}^s_{n+1}$ requires the storage of $\boldsymbol{t}^s_n$. Calculating the time-dependent drag and lift forces follows the same consideration as for

the time-discretized tractions. Interpreting the tractions on the left-hand side of Eq. 2.51 to be at the intermediate time instant $t_{n+\alpha_f}$ is also crucial in retaining the second-order accuracy of the G-α scheme in the coupled simulations, e.g., FSI, as shown in [61].

On the other hand, if the pressure in the fourth term on the right-hand side of Eq. 2.51 is taken to be at t_{n+1} instead of $t_{n+\alpha_f}$, there appears a dilemma in interpreting to which time instant the time-discretized tractions on the left-hand side of Eq. 2.51 belong. The reason is that the calculated tractions would seemingly correspond to $\left(2\mu\left(\nabla^s \boldsymbol{v}^h_{n+\alpha_f}\right) - p^h_{n+1}\boldsymbol{I}\right)\cdot\boldsymbol{n}$, which contains the evaluation of terms at two different time instants.

Two variants of the G-α scheme for the INS equations have been analyzed so far that incorporate the pairs $\{p^h_{n+1}, \nabla\cdot\boldsymbol{v}^h_{n+1}\}$ and $\{p^h_{n+\alpha_f}, \nabla\cdot\boldsymbol{v}^h_{n+\alpha_f}\}$ in the time discretization of Eq. 2.34– 2.41, respectively, and are otherwise identical in the discretization of the other terms. In the context of the time integration of INS equations, we refer to the variant of the G-α scheme with the pair $\{p^h_{n+1}, \nabla\cdot\boldsymbol{v}^h_{n+1}\}$ as the G-α scheme with *first-order accurate pressure*, or G-α with p–$O(\Delta t)$ for short, and we call the variant with the pair $\{p^h_{n+\alpha_f}, \nabla\cdot\boldsymbol{v}^h_{n+\alpha_f}\}$ the *direct* form of the G-α scheme since this variant is the direct application of the G-α scheme to the INS equations as a system of DAEs. As recently reviewed and compared in [16] as well, these two variants make up the bulk of the literature on the application of the G-α scheme to the INS equations, with the former variant being the more common choice despite possessing deficiencies underlined previously and also shown in [16]. The common denominator of the two approaches is the evaluation and interpretation of the pressure term and the incompressibility constraint at the same time instant.

Another seemingly less well-known variant of the G-α scheme for the INS equations is pointed out in the following. Starting from the direct form of the G-α scheme with the pair $\{p^h_{n+\alpha_f}, \nabla\cdot\boldsymbol{v}^h_{n+\alpha_f}\}$, we assume that the old velocity field $\boldsymbol{v}^h_n$ is solenoidal in every time step, i.e., $\nabla\cdot\boldsymbol{v}^h_n = 0$, and therefore, the incompressibility constraint in the strong form reduces to $\nabla\cdot\boldsymbol{v}^h_{n+1} = 0$. This assumption prompts using the pair $\{p^h_{n+\alpha_f}, \nabla\cdot\boldsymbol{v}^h_{n+1}\}$ in Eq. 2.34– 2.41. This approach follows the discussion presented in [2] regarding the application of the trapezoidal rule to the INS equations. As mentioned in [2], by assuming $\boldsymbol{v}^h_n$ to be solenoidal, one solves a (slightly) different, but still consistent, problem than the original one. This variant of the scheme prevents the errors in the incompressibility constraint from propagating in time while retaining the second-order temporal accuracy for pressure. As in the direct form of the G-α, proper initial values for pressure and storage of the old pressure values in each time step are required. We call this variant of the G-α scheme the *shortened* form of the G-α scheme, following a similar naming convention used in [2] for the application of the trapezoidal rule to DAE systems. A similar approach is presented in [10] for the θ schemes, and the resulting schemes are termed *pressure corrected θ-schemes*. In addition to replacing $\boldsymbol{v}^h_{n+\alpha_f}$ with $\boldsymbol{v}^h_{n+1}$ in the incompressibility constraint, $p^h_{n+\alpha_f}$ can be also replaced by

$\bar{p}^h_{n+\alpha_f}$, where $\bar{p}^h_{n+\alpha_f}$ denotes the pressure degree of freedom at the intermediate time instant $t_{n+\alpha_f}$ while $p^h_{n+\alpha_f}$ denotes, as usual, the generalized average pressure at $t_{n+\alpha_f}$ as given by Eq. 2.31. This change helps with preserving the (skew-)symmetry of the off-diagonal blocks in the global system of linear equations. In order to further clarify this point, consider the direct form of the G-α scheme applied to the INS equations without the stabilization terms for simplicity. Taking into account the G-α rules Eqs. 2.30–2.33, the updates for the unknown velocity and pressure DoFs in the nth time step, $\boldsymbol{U}_{n+1}$ and P_{n+1}, are calculated through solving the following algebraic system in every iteration of the nonlinear solution strategy:

$$\begin{bmatrix} \frac{\alpha_m}{\gamma\Delta t}\boldsymbol{M} + \alpha_f \boldsymbol{N}^{lin} + \alpha_f \boldsymbol{A} & -\alpha_f \boldsymbol{B} \\ \alpha_f \boldsymbol{B}^T & 0 \end{bmatrix} \begin{bmatrix} \Delta \boldsymbol{V}_{\boldsymbol{n+1}} \\ \Delta \boldsymbol{P}_{n+1} \end{bmatrix} = - \begin{bmatrix} \boldsymbol{R}^m \\ \boldsymbol{R}^c \end{bmatrix}, \tag{2.53}$$

where $\boldsymbol{M}$, $\boldsymbol{N}^{lin}$, $\boldsymbol{A}$, and $\boldsymbol{B}$ represent the discrete counterparts of the inertial, (linearized) convection, diffusion, and pressure terms, respectively. $\boldsymbol{R}^m$ and $\boldsymbol{R}^c$ denote the residuals of the weak forms of the momentum and continuity equations, respectively. Note that the (skew-)symmetry of the off-diagonal blocks, arising from the pressure and the incompressibility constraint, is preserved after time integration. If it is desired to remove the coefficient in front of the $\boldsymbol{M}$, all of the terms in Eq. 2.53 are divided by $\frac{\alpha_m}{\gamma\Delta t}$.

In the case of applying the shortened G-α scheme to the INS equations, the updates for the unknown velocity and pressure vectors of DoFs in the nth time step, $\boldsymbol{U}_{n+1}$ and $\bar{\boldsymbol{P}}_{n+\alpha_f}$, are calculated through solving the following algebraic system in every iteration of the nonlinear solution strategy:

$$\begin{bmatrix} \frac{\alpha_m}{\gamma\Delta t}\boldsymbol{M} + \alpha_f \boldsymbol{N}^{lin} + \alpha_f \boldsymbol{A} & -\boldsymbol{B} \\ \boldsymbol{B}^T & 0 \end{bmatrix} \begin{bmatrix} \Delta \boldsymbol{V}_{\boldsymbol{n+1}} \\ \Delta \bar{\boldsymbol{P}}_{n+\alpha_f} \end{bmatrix} = - \begin{bmatrix} \boldsymbol{R}^m \\ \boldsymbol{R}^c \end{bmatrix}, \tag{2.54}$$

and the pressure values at t_{n+1} are subsequently updated through

$$\boldsymbol{P}_{n+1} = \frac{1}{\alpha_f}\bar{\boldsymbol{P}}_{n+\alpha_f} - \frac{(1-\alpha_f)}{\alpha_f}\boldsymbol{P}_n. \tag{2.55}$$

Note that the (skew-)symmetry of the off-diagonal blocks is preserved in Eq. 2.54. That is the main reason for using the pair $\{\bar{p}^h_{n+\alpha_f}, \nabla\cdot\boldsymbol{v}^h_{n+1}\}$ instead of $\{p^h_{n+\alpha_f}, \nabla\cdot\boldsymbol{v}^h_{n+1}\}$ in Eqs. 2.34–2.41 when using the shortened form of G-α scheme. Another possibility for keeping the symmetry properties of the off-diagonal blocks of the algebraic system when using the shortened form of the G-α scheme is to use the pair $\{p^h_{n+\alpha_f}, \nabla\cdot\boldsymbol{v}^h_{n+1}\}$ and scale the incompressibility constraint as $\alpha_f \nabla\cdot\boldsymbol{v}^h_{n+1} = 0$. This approach is most useful when using inf-sup stable elements since other terms are added to the off-diagonal and diagonal blocks of the system using the stabilized methods. Another advantage of using $\bar{p}^h_{n+\alpha_f}$ as pressure unknown is the straightforward interpretation of the

pressure $\bar{p}^h_{n+\alpha_f}$ as the Lagrange multiplier at the intermediate time instant, which satisfies the (shortened) incompressibility constraint.

2.1.5 An end-of-step-equilibrium form of G-α scheme for INS equations

2.1.5.1 Second-order accurate time derivatives using the G-α scheme

It should be pointed out that the time derivative of velocity, although part of the INS equations, may not be of much interest in many applications. Furthermore, the time derivative of pressure is absent in the INS equations. However, as pointed out in [16], the time derivatives of velocity and pressure are needed in Hemodynamic simulations as "inputs to reduced-order models of the upstream and downstream vasculature, which are commonly used as boundary conditions coupled to the three-dimensional domain"[16]. Therefore, it is worth examining the order of accuracy of velocity and pressure time derivatives in the INS equations using the G-α time integration scheme. Furthermore, the second-order accurate acceleration variable plays a crucial role in the alternative form of the G-α scheme, as will be presented shortly.

The G-α scheme for the INS equations possesses second-order (temporal) accuracy for the velocity variable and can retain the second-order accuracy for the pressure variable, as discussed previously. On the contrary, it is commonly believed that the G-α scheme demonstrates only first-order accuracy for the acceleration variable and the time derivative of the pressure variable if needed [16]. However, it is shown in [48] how the second-order accuracy of acceleration can be restored using the G-α scheme for second-order systems (possessing displacement, velocity, and acceleration variables) arising from structural mechanics. The present section discusses how the second-order accuracy for the time derivatives of velocity and pressure can be retained using the G-α scheme. The following discussion is an adaptation of the results first presented in [48] to the present discussion of the time integration of INS equations.

Recall the first-order model problem introduced in Eq. 2.27. Denoting by $v^{ex}(t)$ the exact solution of v at time t, the equilibrium equation of the model problem using the exact solution at time t reads

$$\partial_t v^{ex}(t) - \mathcal{N}(v^{ex}(t)) = \partial_t v^{ex}(t) - \mathcal{N}^{ex}(t) = 0. \tag{2.56}$$

On the other hand, the time-discretized equilibrium equation using the G-α scheme at the nth time step reads

$$\tilde{\dot{v}}_{n+\alpha_m} - \mathcal{N}_{n+\alpha_f}(\alpha_f, v_n, v_{n+1}) = 0, \tag{2.57}$$

where either of the definitions in Eq. 2.28 or 2.29 could be used for $\mathcal{N}_{n+\alpha_f}$. Recall that by satisfying the condition in Eq. 2.24 for the G-α scheme, the second-order temporal accuracy is obtained for v at all discrete time instants,

e.g., for v_{n+1}, we have

$$v_{n+1} = v^{ex}(t_{n+1}) + \mathcal{O}(\Delta t^2), \tag{2.58}$$

which if inserted in Eq. 2.57 leads to

$$\tilde{\dot{v}}_{n+\alpha_m} - \mathcal{N}^{ex}_{n+\alpha_f}(\alpha_f, v_n, v_{n+1}) = \mathcal{O}(\Delta t^2). \tag{2.59}$$

Using

$$\mathcal{N}^{ex}_{n+\alpha_f}(\alpha_f, v_n, v_{n+1}) = (1-\alpha_f)\mathcal{N}^{ex}(v_n) + (\alpha_f)\mathcal{N}^{ex}(v_{n+1}) + \mathcal{O}(\Delta t^2) \tag{2.60}$$

in Eq. 2.59 in conjunction with the definition of the exact equilibrium equation in Eq. 2.56, one finally arrives at the following relation between $\tilde{\dot{v}}$, the acceleration variable of the G-α scheme, and the exact value of acceleration $\partial_t v^{ex}$:

$$(1-\alpha_m)\,\tilde{\dot{v}}_n + \alpha_m \tilde{\dot{v}}_{n+1} = (1-\alpha_f)\,\partial_t v^{ex}(t_n) + \alpha_f \partial_t v^{ex}(t_{n+1}) + \mathcal{O}(\Delta t^2). \tag{2.61}$$

Motivated by Eq. 2.61, one can introduce a new variable $\dot{v}$ for acceleration with the following relation to $\tilde{\dot{v}}$:

$$(1-\alpha_m)\,\tilde{\dot{v}}_n + \alpha_m \tilde{\dot{v}}_{n+1} = (1-\alpha_f)\,\dot{v}_n + \alpha_f \dot{v}_{n+1}. \tag{2.62}$$

In contrast to $\tilde{\dot{v}}$, which is generally only a first-order accurate approximation to acceleration (hence distinguished by $\tilde{}$), $\dot{v}$ enjoys second-order temporal accuracy. $\tilde{\dot{v}}$ is referred to as the *auxiliary* acceleration or an *acceleration-like* variable, while $\dot{v}$ is called the *true* acceleration variable. The true acceleration can be calculated in a post-processing step using Eq. 2.61 and, the its price is the storage of an extra variable.

With regard to the INS equations, it is immediately apparent that the second-order accurate true acceleration vector $\dot{\boldsymbol{v}}^h_{n+1}$ can be calculated using

$$(1-\alpha_m)\,\tilde{\dot{\boldsymbol{v}}}^h_n + \alpha_m \tilde{\dot{\boldsymbol{v}}}^h_{n+1} = (1-\alpha_f)\,\dot{\boldsymbol{v}}^h_n + \alpha_f \dot{\boldsymbol{v}}^h_{n+1}, \tag{2.63}$$

where $\tilde{\dot{\boldsymbol{v}}}^h_{n+1}$ itself is updated using Eq. 2.33.
In [16], the pressure time derivative $\dot{p}^h_{n+1}$ is calculated using the same G-α update rule as in Eq. 2.33:

$$p^h_{n+1} = p^h_n + \Delta t\left((1-\gamma)\,\tilde{\dot{p}}^h_n + \gamma \tilde{\dot{p}}^h_{n+1}\right), \tag{2.64}$$

which implicitly implies that the pressure and its time derivative are related through a first-order ODE. It is demonstrated in [16] that the pressure derivative calculated using Eq. 2.64 is first-order accurate in time, an observation also expected from the discussion in the present section. In analogy to the velocity time derivative, and given that the pressure variable retains second-order accuracy, a true pressure time derivative $\dot{p}^h_{n+1}$ can be calculated using

$$(1-\alpha_m)\,\tilde{\dot{p}}^h_n + \alpha_m \tilde{\dot{p}}^h_{n+1} = (1-\alpha_f)\,\dot{p}^h_n + \alpha_f \dot{p}^h_{n+1}. \tag{2.65}$$

2.1.5.2 Deriving the end-of-step-equilibrium form of G-α

Recall the time-discretized INS equations given in Eqs. 2.34–2.41 and the earlier comment about the presence of terms evaluated at different time instants $t+\alpha_m$ and $t+\alpha_f$ preventing us from restating the entirety of the INS equations using a generalized trapezoidal quadrature rule. In light of the possibility for calculating the true acceleration using Eq. 2.63, the idea arises to replace $\tilde{\dot{\boldsymbol{v}}}^h_{n+\alpha_m}$ with $\dot{\boldsymbol{v}}^h_{n+\alpha_f}$ in Eqs. 2.34–2.41. Doing so allows us to use the generalized trapezoidal quadrature rule for all terms in the time-discretized INS equations. The terms at time instant t_{n+1} and with coefficient α_f can be collected together and separated from those at time instant t_n and with the coefficient $(1-\alpha_f)$. Requiring the equilibrium to hold for each collected group of terms at t_{n+1} and t_n leads to the following equilibrium equation in the nth time step:

$$\begin{aligned}
&\left(\boldsymbol{w}^h, \rho\dot{\boldsymbol{v}}^h_{n+1}\right)_\Omega + \left(\boldsymbol{w}^h, \rho(\boldsymbol{v}^a_{n+1}\cdot\nabla)\boldsymbol{v}^h_{n+1}\right)_\Omega + \left(\nabla^s\boldsymbol{w}^h, 2\mu\nabla^s\boldsymbol{v}^h_{n+1}\right)_\Omega \\
&- \left(\nabla\cdot\boldsymbol{w}^h, p^h_{n+1}\right)_\Omega + \left(q^h, \nabla\cdot\boldsymbol{v}^h_{n+1}\right)_\Omega + \left(\mathcal{L}^*_{n+1}\left(\boldsymbol{w}^h, q^h\right), \boldsymbol{v}'_{n+1}\right)_{\Omega^K} \\
&- \left(\nabla\cdot\boldsymbol{w}^h, p'_{n+1}\right)_{\Omega^K} = \left(\boldsymbol{w}^h, \boldsymbol{f}_{n+1}\right)_\Omega,
\end{aligned} \tag{2.66}$$

where

$$\mathcal{L}^*_{n+1}\left(\boldsymbol{w}^h, q^h\right) = -\left(\rho(\boldsymbol{v}^a_{n+1}\cdot\nabla)\boldsymbol{w}^h + 2\mu\nabla\cdot\left(\nabla^s\boldsymbol{w}^h\right) + \nabla q^h\right), \tag{2.67}$$

$$\boldsymbol{v}'_{n+1} = \tau^v_{n+1}\boldsymbol{r}^m_{n+1}, \tag{2.68}$$

$$p'_{n+1} = \tau^p_{n+1} r^c_{n+1}, \tag{2.69}$$

$$\tau^v_{n+1} = \left(\frac{c_1\mu}{h^2} + \frac{c_2\rho\|\boldsymbol{v}^a_{n+1}\|}{h}\right)^{-1}, \tag{2.70}$$

$$\tau^p_{n+1} = \frac{h^2}{c_1\tau^v_{n+1}}, \tag{2.71}$$

$$\boldsymbol{r}^m_{n+1} = \boldsymbol{f}_{n+1} - \rho\dot{\boldsymbol{v}}^h_{n+1} - \rho\left(\boldsymbol{v}^a_{n+1}\cdot\nabla\right)\boldsymbol{v}^h_{n+1} + 2\mu\nabla\cdot\left(\nabla^s\boldsymbol{v}^h_{n+1}\right) - \nabla p^h_{n+1}, \tag{2.72}$$

$$r^c_{n+1} = -\nabla\cdot\boldsymbol{v}^h_{n+1}. \tag{2.73}$$

Eqs. 2.66–2.73 are accompanied by the following update formula for the true acceleration $\dot{\boldsymbol{v}}^h_{n+1}$:

$$\dot{\boldsymbol{v}}^h_{n+1} = \frac{\alpha_m}{\alpha_f \gamma \Delta t}\left(\boldsymbol{v}^h_{n+1} - \boldsymbol{v}^h_n\right) + \frac{(\alpha_m - \gamma)}{\alpha_f \gamma}\tilde{\dot{\boldsymbol{v}}}^h_n - \frac{(1-\alpha_f)}{\alpha_f}\dot{\boldsymbol{v}}^h_n. \tag{2.74}$$

The update formula in Eq. 2.74 is derived by combining Eqs. 2.33 and 2.63 and rearranging terms. Note in Eq. 2.74 that the storage of only $\tilde{\dot{\boldsymbol{v}}}^h_n$ is necessary and not that of $\tilde{\dot{\boldsymbol{v}}}^h_{n+1}$. The algorithmic parameters α_f, α_m and γ are calculated using the formulae given in Eq. 2.26. Note that the Eqs. 2.66–2.73 are written in terms of the end-of-step velocity, pressure, and (true) acceleration and that the equilibrium equation is satisfied also at the end of the time step, as opposed to the standard form of G-α, where the equilibrium is satisfied in an average sense in each time step. Therefore, we call the new form of the G-α scheme for the INS equations the *end-of-step-equilibrium* G-α scheme, or EG-α for short. This idea was also used in [62] to derive a similar form of the G-α scheme for the second-order systems, arising from the simulation of the constrained multibody mechanical systems represented by index-3 DAEs. It is also worth highlighting the work in [63], where (in our terminology) an end-of-step-equilibrium form of the one-step-θ scheme for the INS equations is derived. The advantage of the EG-α scheme in the present work over the one-step-θ scheme utilized in [63] is that the EG-α scheme is second-order accurate for all admissible values of ρ_∞, whereas the one-step-θ scheme is second-order accurate only for $\theta = 1/2$, which is equivalent to the Crank-Nicolson scheme that provides no high-frequency dissipation and is in practice more prone to numerical oscillations. Furthermore, the G-α family of schemes introduces less numerical dissipation in the low and mid-frequency range than the one-step-θ family for the same value of ρ_∞ [5].

The variationally consistent traction on the boundary Γ^s using the EG-α scheme is calculated as

$$\begin{aligned}\left(\boldsymbol{w}^h, \boldsymbol{t}^s_{n+1}\right)_{\Gamma^s} &= \left(\boldsymbol{w}^h, \rho\dot{\boldsymbol{v}}^h_{n+1}\right)_\Omega + n\left(\boldsymbol{w}^h, \boldsymbol{v}^h_{n+1}, \boldsymbol{v}^a_{n+1}\right) + a\left(\boldsymbol{w}^h, \boldsymbol{v}^h_{n+1}\right) \\ &- \left(\nabla\cdot\boldsymbol{w}^h, p^h_{n+1}\right)_\Omega + \left(q^h, \nabla\cdot\boldsymbol{v}^h_{n+1}\right)_\Omega + \left(\mathcal{L}^\star_{n+1}\left(\boldsymbol{w}^h, q^h\right), \boldsymbol{v}'_{n+1}\right)_{\Omega^K} \\ &- \left(\nabla\cdot\boldsymbol{w}^h, p'_{n+1}\right)_{\Omega^K} - \left(\boldsymbol{w}^h, \boldsymbol{f}_{n+1}\right)_\Omega.\end{aligned} \tag{2.75}$$

Note that $\boldsymbol{t}^s_{n+1}$ is directly calculated from Eq. 2.75, and no storage and updating of the old traction values $\boldsymbol{t}^s_n$ is required. The same holds for calculating the drag and lift forces on Γ^s.

Since the INS equations are nonlinear, a predictor-multicorrector solution procedure consistent with the time integration scheme is required. The predictor assumes that the values of velocity and pressure remain the same when moving from one time step to the next. Using the EG-α scheme, the values of the variables at the predictor stage (the zeroth nonlinear iteration)

are:

$$\boldsymbol{V}_{n+1}^{h,(0)} = \boldsymbol{V}_n^h \tag{2.76}$$

$$\boldsymbol{P}_{n+1}^{h,(0)} = \boldsymbol{P}_n^h \tag{2.77}$$

$$\dot{\boldsymbol{V}}_{n+1}^{h,(0)} = \frac{(\alpha_m - \gamma)}{\alpha_f \gamma} \tilde{\dot{\boldsymbol{V}}}_n^h - \frac{(1-\alpha_f)}{\alpha_f} \dot{\boldsymbol{V}}_n^h. \tag{2.78}$$

Different linearization approaches are conceivable for INS equations. In the present work, a Picard linearization of the nonlinear convective term is carried out and the derivatives of the stabilization terms with respect to the velocity and pressure unknowns are excluded from the left-hand side matrix. For a detailed discussion on different linearization techniques for INS equations, see [35, 1, 13, 206]. After spatial discretization, the semi-discrete problem in matrix form can be generally expressed as

$$\boldsymbol{\mathcal{M}} \begin{bmatrix} \partial_t \boldsymbol{V} \\ 0 \end{bmatrix} + \boldsymbol{\mathcal{C}} \begin{bmatrix} \boldsymbol{V} \\ \boldsymbol{P} \end{bmatrix} = \begin{bmatrix} \boldsymbol{F} \\ 0 \end{bmatrix}. \tag{2.79}$$

From Eq. 2.74, the derivative of the true acceleration variable in the EG-α scheme with respect to the velocity variable reads

$$\frac{\partial \dot{\boldsymbol{V}}_{n+1}^h}{\boldsymbol{V}_{n+1}^h} = \frac{\alpha_m}{\alpha_f \gamma \Delta t}. \tag{2.80}$$

Therefore, the linear system of equations to be solved in the kth nonlinear iteration is

$$\left(\frac{\alpha_m}{\alpha_f \gamma \Delta t} \boldsymbol{\mathcal{M}} + \boldsymbol{\mathcal{C}} \right) \begin{bmatrix} \Delta \boldsymbol{V}^{(k+1)} \\ \Delta \boldsymbol{P}^{(k+1)} \end{bmatrix} = -\boldsymbol{R}_{n+1}^{(k)}, \tag{2.81}$$

where $\Delta \boldsymbol{V}_{n+1}^{(k+1)} = \boldsymbol{V}_{n+1}^{(k+1)} - \boldsymbol{V}_{n+1}^{(k)}$, $\Delta \boldsymbol{P}_{n+1}^{(k+1)} = \boldsymbol{P}_{n+1}^{(k+1)} - \boldsymbol{P}_{n+1}^{(k)}$, and $\boldsymbol{R}_{n+1}^{(k)}$ is the residual of INS equations (LHS minus RHS of Eq. 2.66) at the kth iteration.

After solving the linear system of equations in the kth nonlinear iteration, the vectors of variables are updated in the corrector stage as follows:

$$\boldsymbol{V}_{n+1}^{(k+1)} = \boldsymbol{V}_{n+1}^{(k)} + \Delta \boldsymbol{V}_{n+1}^{(k+1)}, \tag{2.82}$$

$$\boldsymbol{P}_{n+1}^{(k+1)} = \boldsymbol{P}_{n+1}^{(k)} + \Delta \boldsymbol{P}_{n+1}^{(k+1)}, \tag{2.83}$$

$$\dot{\boldsymbol{V}}_{n+1}^{(k+1)} = \dot{\boldsymbol{V}}_{n+1}^{(k)} + \frac{\alpha_m}{\alpha_f \gamma \Delta t} \Delta \boldsymbol{V}_{n+1}^{(k+1)}. \tag{2.84}$$

The vector of auxiliary variable $\tilde{\dot{\boldsymbol{V}}}_{n+1}$ is updated at the end of time step using Eq. 2.62.

The satisfaction of the equilibrium equation at the discrete time points has advantages in specific applications and situations, as will be shown in

section 2.1.6. Furthermore, by satisfying the incompressibility constraint exactly at the discrete time points, the errors in the continuity equation are not propagated in time, a feature that is achieved in the G-α scheme only by solving a (slightly) modified problem through the shortened form of the G-α scheme. Moreover, no initial value is required for pressure when using the EG-α scheme, as opposed to the direct and shortened forms of the G-α scheme. An initial value is required for the acceleration in the EG-α and all other previously mentioned variants of the G-α scheme. When using EG-α, if no restart information from previous simulations is available, the initial value for $\dot{\boldsymbol{v}}^h$ is also set to $\tilde{\dot{\boldsymbol{v}}}^h$. The EG-$\alpha$ scheme retains the second-order accuracy for the velocity, pressure, and acceleration variables. Lastly, implementing the EG-α scheme is relatively easy compared to the G-α scheme, particularly in existing frameworks geared towards working with the end-of-step values, e.g., through using the BDF2 time integration scheme, as opposed to the generalized averaged values used in the G-α scheme.

It is also worth mentioning that, as pointed out in [64, 14, 15, 65], the G-α scheme is equivalent to the midpoint rule (equivalent to Crank-Nicolson for linear systems) for $\rho_\infty = 1$ and is equivalent to the BDF2 scheme for $\rho_\infty = 0$. Therefore, one may think of the G-α scheme as the interposition between the midpoint and the BDF2 schemes. On the other hand, the EG-α scheme is equivalent to an end-of-step-equilibrium form of the Crank-Nicolson scheme (trapezoidal rule) for $\rho_\infty = 1$ and equivalent to the BDF2 scheme for $\rho_\infty = 0$ and can be thought of as the interpolation between those two schemes for the rest of the admissible values of ρ_∞.

2.1.5.3 Time integration of the dynamic algebraic and orthogonal subgrid scales

So far, we have dealt with the quasi-static subgrid scales in our VMS formulation. In this section, the dynamic subgrid scales, first in combination with the ASGS formulation, are discussed. The evolution of the dynamic velocity subscales is governed by Eq. 2.16, with $\boldsymbol{\xi}^h$ set to zero. The first "natural" choice for discretizing Eq. 2.16 in time is to use the same time integration scheme for the large-scale problem [6]. In the case of the G-α scheme, the evolution of the dynamic subgrid scales on each integration point of each element follows

$$\rho\tilde{\dot{\boldsymbol{v}}}'_{n+\alpha_m} + \left(\tau^v_{n+\alpha_f}\right)^{-1}\boldsymbol{v}'_{n+\alpha_f} = \boldsymbol{r}^m_{n+\alpha_{f,m}}, \tag{2.85}$$

where $\tau^v_{n+\alpha_f}$ and $\boldsymbol{r}^m_{n+\alpha_{f,m}}$ are defined according to Eq. 2.36 and Eq. 2.40, respectively. As before, the subscript $n+\alpha_{f,m}$ means that $\boldsymbol{r}^m_{n+\alpha_{f,m}}$ contains terms that are evaluated at $t_{n+\alpha_f}$ and $t_{n+\alpha_m}$. Closed-form formulae for $\boldsymbol{v}'_{n+1}$ and $\tilde{\dot{\boldsymbol{v}}}'_{n+1}$ are given in [66] and are restated here for comparison (attention:ρ

is missing):

$$\boldsymbol{v}'_{n+1} = \frac{\alpha_m \tau^v_{n+\alpha_f} + (\alpha_f - 1)\gamma\Delta t}{\alpha_m \tau^v_{n+\alpha_f} + \alpha_f \gamma \Delta t}\, \boldsymbol{v}'_n + \frac{(\alpha_m - \gamma)\, \tau^v_{n+\alpha_f} \Delta t}{\alpha_m \tau^v_{n+\alpha_f} + \alpha_f \gamma \Delta t}\, \tilde{\boldsymbol{v}}'_n + \left(\frac{\alpha_m \gamma}{\Delta t} + \frac{\alpha_f}{\tau^v_{n+\alpha_f}} \right)^{-1} \boldsymbol{r}^m_{n+\alpha_{f,m}}, \tag{2.86}$$

$$\tilde{\boldsymbol{v}}'_{n+1} = \frac{(\alpha_m - 1)\, \tau^v_{n+\alpha_f} + \alpha_f\, (\gamma - 1)\, \Delta t}{\alpha_m \tau^v_{n+\alpha_f} + \alpha_f \gamma \Delta t}\, \tilde{\boldsymbol{v}}'_n + \frac{\tau^v_{n+\alpha_f}}{\alpha_m \tau^v_{n+\alpha_f} + \alpha_f \gamma \Delta t} \left(\boldsymbol{r}^m_{n+\alpha_{f,m}} - \frac{1}{\tau^v_{n+\alpha_f}} \boldsymbol{v}'_n \right). \tag{2.87}$$

Eqs. 2.86 and 2.87 are used to update the fine-scale velocity and acceleration variables at the end of the time step. The intermediate quantities $\tilde{\boldsymbol{v}}'_{n+\alpha_m}$ and $\boldsymbol{v}'_{n+\alpha_f}$ for use in Eq. 2.34 are calculated with the help of Eqs. 2.86 and 2.87. Note that the term $\left(\boldsymbol{w}^h, \rho \tilde{\boldsymbol{v}}'_{n+\alpha_m}\right)_\Omega$ is added to Eq. 2.34 when using dynamic ASGS formulation. The variables $\tilde{\boldsymbol{v}}'_n$ and $\boldsymbol{v}'_n$ on the right-hand side of Eqs. 2.86 and 2.87 are stored at the integration points of each element. It is also worth noting that the scaling factor in front of $\boldsymbol{r}^m_{n+\alpha_{f,m}}$ in Eq. 2.86 works as the effective stabilization parameter when using the G-α_f scheme. This interpretation follows [6], where a similar interpretation of the stabilization parameter is given when using the one-step-θ scheme.

Another possibility for the time discretization of the subgrid scales is introduced in [67], where the first-order backward Euler scheme is used for the time discretization of the subgrid scales instead of using the same time integration scheme used for the large-scale problem. It is shown in [67] that the error introduced by evaluating the subgrid scales at t_{n+1}, instead of the intermediate time instant, introduces an error of order $\mathcal{O}(\Delta t^2)$, which is the same order as the error in the time integration scheme of the large-scale problem. Therefore, a lower-order time integration scheme can be used for the fine-scale problem without upsetting the second-order temporal accuracy of the large-scale quantities. Using the backward Euler scheme for the subgrid scales helps with the stability of the problem in time as well. The strong-form residuals of the INS equations are not smooth and are discontinuous across inter-element boundaries. Consequently, the possibility of developing high-frequency oscillations might increase. Therefore, it is reasonable to use a dissipative scheme for the time discretization of the subgrid scales [67]. Adding numerical dissipation to the integration of the subgrid scale is also possible using Eqs. 2.85–2.87. However, it simultaneously dictates adding more numerical dissipation to the large-scale problem since the same time integration scheme is used for the large and fine-scale problems. Another advantage of using the backward Euler scheme for the subgrid scales is the

reduced storage requirements only for storing $\boldsymbol{v}'_n$, instead of $\boldsymbol{v}'_n$ and $\tilde{\boldsymbol{v}}'_n$ in Eqs. 2.86 and 2.87. This storage saving could be of importance, in particular, when using higher-order elements since $\boldsymbol{v}'_n$ and $\tilde{\boldsymbol{v}}'_n$ should be stored on all integration points.

Discretizing Eq. 2.16 using backward Euler scheme leads to

$$\rho \frac{\boldsymbol{v}'_{n+1} - \boldsymbol{v}'_n}{\Delta t} + \left(\tau^v_{n+1}\right)^{-1} \boldsymbol{v}'_{n+1} = \boldsymbol{r}^m_{n+1}. \tag{2.88}$$

Closed-form formulae for the velocity subscale and its derivative are straightforward to derive and read

$$\boldsymbol{v}'_{n+1} = \left(\frac{\rho}{\Delta t} + \frac{1}{\tau^v_{n+1}}\right)^{-1} \left(\boldsymbol{r}^m_{n+1} + \frac{\rho \boldsymbol{v}'_n}{\Delta t}\right), \tag{2.89}$$

$$\dot{\boldsymbol{v}}'_{n+1} = \frac{\boldsymbol{v}'_{n+1} - \boldsymbol{v}'_n}{\Delta t} = \left(\frac{\rho}{\Delta t} + \frac{1}{\tau^v_{n+1}}\right)^{-1} \left(\frac{\boldsymbol{r}^m_{n+1}}{\Delta t} + \frac{\rho \boldsymbol{v}'_n}{\Delta t^2}\right) - \frac{\boldsymbol{v}'_n}{\Delta t}, \tag{2.90}$$

where the scaling factor of the term $\boldsymbol{r}^m_{n+1}$ is the effective stabilization parameter. Note that it is similar to the relation used in the step-size-dependent stabilization parameter. However, as remarked in [67], the difference is that the subgrid scales are properly integrated and tracked in time when using Eq. 2.89, which allows using arbitrarily time step sizes and preventing instabilities due to small time step sizes.

Eqs. 2.88–2.90 fit very well with the large-scale INS equations integrated by the EG-α scheme, Eqs. 2.66–2.73, since all the terms are evaluated at the end of the time step t_{n+1}. Therefore, when using a dynamic ASGS formulation integrated by the EG-α scheme, $\boldsymbol{v}'_{n+1}$ in Eq. 2.66 is replaced by the definition given in Eq. 2.89, and the term $\left(\boldsymbol{w}^h, \rho \dot{\boldsymbol{v}}'_{n+1}\right)_\Omega$ is added to the left-hand side of Eq. 2.66, where $\dot{\boldsymbol{v}}'_{n+1}$ is calculated according to Eq. 2.90. One could also integrate the subgrid scales using the EG-α. However, we avoid doing so for the same reasons stated for the case of G-α scheme (The EG-α scheme would pose even higher storage requirements due to the inclusion of true and auxiliary acceleration variables for the subgrid scales).

If an OSS formulation is used instead of ASGS, the quantity $\boldsymbol{\xi}^h$ in Eq. 2.16 is not zero. Therefore, $\boldsymbol{\xi}^h_{n+1}$, the projection of $\boldsymbol{r}^h_{n+1}$ onto the space of finite elements, is calculated using

$$\left(\boldsymbol{w}^h, \boldsymbol{\xi}^h_{n+1}\right)_{\Omega K} = \left(\boldsymbol{w}^h, \boldsymbol{r}^m_{n+1}\right)_{\Omega K} \qquad \forall \boldsymbol{w}^h \in \boldsymbol{\mathcal{W}}^h, \tag{2.91}$$

where $\boldsymbol{r}^m_{n+1}$ is defined in Eq. 2.72. Furthermore, $\boldsymbol{r}^m_{n+1}$ is replaced by $\boldsymbol{r}^m_{n+1} - \boldsymbol{\xi}^h_{n+1}$ on the right-hand sides of Eqs. 2.88–2.90, which then leads to the time discretization of Eq. 2.16 using backward Euler scheme. Since the information about the finite element residuals is required to carry out the projection in Eq. 2.91, a closed-form formulation of the large-scale problem is no longer possible,

when using the OSS variant. Instead, the fine and large-scale problems are usually solved iteratively until convergence, see, e.g, –. Theoretically, if the calculated subgrid scales are truly orthogonal to the space of the coarse scales as done in –, the term $\left(\boldsymbol{w}^h, \rho\dot{\boldsymbol{v}}'_{n+1}\right)_\Omega$ does not need to be added to Eq. 2.66, and the contribution of – is removed from Eq. 2.72 for $\boldsymbol{r}^m_{n+1}$, and subsequently from $\boldsymbol{\xi}^h_{n+1}$, due to the orthogonality assumption between the fine and large-scale spaces. As suggested in [37], the terms $\boldsymbol{f}_{n+1}$ and $2\mu\nabla\cdot\left(\nabla^s\boldsymbol{v}^h_{n+1}\right)$ could be also dropped from $\boldsymbol{r}^m_{n+1}$ and the calculation of $\boldsymbol{\xi}^m_{n+1}$, when using orthogonal subgrid scales.

2.1.6 Incompressible Navier-Stokes equations on moving domains

This section focuses on the time integration of the INS equations on moving domains. The EG-α and different variants of the G-α are derived consistently and analyzed for simulations on moving domains, and the advantage of the EG-α scheme for such simulations is highlighted. The moving meshes are handled using the Arbitrary Lagrangian-Eulerian (ALE) method [68, 69, 70] in this work. However, the EG-α scheme benefits also other approaches for simulations on moving domains.

In an ALE formulation, the deforming domain at the current time t $\Omega_t = \Omega(t)$ is referred to as the *current configuration*. On the other hand, the ALE or *reference configuration*, typically chosen to be equal to the initial configuration, is denoted by Ω_0. We distinguish between the spatial (Eulerian) coordinates $\boldsymbol{x} \in \Omega_t$ and the reference (ALE) coordinates $\mathbf{x} \in \Omega_0$ (there is also a third coordinate expressing the material (Lagrangian) coordinate, which we do not directly deal with in the present section). The current and the reference configurations are related through the mapping $\mathcal{A}_t$, which is assumed to be sufficiently regular and invertible (see [71] for more complete definitions and derivation):

$$\mathcal{A}_t : \Omega_0 \to \Omega_t, \qquad \boldsymbol{x}(\mathbf{x}, t) = \mathcal{A}_t(\mathbf{x}) \quad \forall \mathbf{x} \in \Omega_0. \tag{2.92}$$

The time derivative of the mapping $\mathcal{A}_t$ is denoted by $\boldsymbol{v}^m$, i.e., $\boldsymbol{v}^m = \partial_t\mathcal{A}_t(\mathbf{x}) = \partial_t\boldsymbol{x}(\mathbf{x}, t)$, and is referred to as the *domain velocity* (or *mesh velocity* in a discrete setup). The ALE time derivative of a vector-valued function $\boldsymbol{v}$ defined in the current configuration $\left.\frac{\partial\boldsymbol{v}}{\partial t}\right|_{\mathbf{x}}$ is known to be related to its local (partial) time derivative in the spatial frame $\left.\frac{\partial\boldsymbol{v}}{\partial t}\right|_{x}$ through

$$\left.\frac{\partial\boldsymbol{v}}{\partial t}\right|_{\mathbf{x}} = \left.\frac{\partial\boldsymbol{v}}{\partial t}\right|_{x} + \left(\nabla_x\boldsymbol{v}\right)\boldsymbol{v}^m, \tag{2.93}$$

or equivalently,

$$\left.\frac{\partial\boldsymbol{v}}{\partial t}\right|_{x} = \left.\frac{\partial\boldsymbol{v}}{\partial t}\right|_{\mathbf{x}} - \left(\nabla_x\boldsymbol{v}\right)\boldsymbol{v}^m, \tag{2.94}$$

where ∇_x denotes the gradient with respect to the spatial coordinates $\boldsymbol{x}$. From Eq. 2.94, it is clear that the difference between the two types of time derivatives is the appearance of the advective-like term $(\nabla_x \boldsymbol{v})\,\boldsymbol{v}^m = (\boldsymbol{v}^m \cdot \nabla_x)\,\boldsymbol{v}$. In the following, the subscript from the partial time derivative. Similarly, the subscripts from the spatial operators with respect to the spatial coordinates are dropped, e.g., the gradient in the spatial coordinates is denoted by ∇. With these definitions at hand, the ALE formulation of the INS equations on a moving domain reads

$$\begin{aligned}
&\left(\boldsymbol{w}^h, \rho\partial_t \boldsymbol{v}^h\right)_{\Omega_t} + \left(\boldsymbol{w}^h, \rho\left((\boldsymbol{v}^a - \boldsymbol{v}^m)\cdot\nabla\right)\boldsymbol{v}^h\right)_{\Omega_t} + \left(\nabla\boldsymbol{w}^h, \boldsymbol{\sigma}^{v,h}\right)_{\Omega_t} \\
&+ \left(\nabla\boldsymbol{w}^h, \boldsymbol{\sigma}^{p,h}\right)_{\Omega_t} + \left(q^h, \nabla\cdot\boldsymbol{v}^h\right)_{\Omega_t} - \left(\rho\left((\boldsymbol{v}^a - \boldsymbol{v}^m)\cdot\nabla\right)\boldsymbol{w}^h + \nabla q^h, \boldsymbol{v}'\right)_{\Omega_t^K} \\
&- \left(\nabla\cdot\boldsymbol{w}^h, p'\right)_{\Omega_t^K} = \left(\boldsymbol{w}^h, \boldsymbol{f}\right)_{\Omega_t},
\end{aligned} \tag{2.95}$$

with

$$\boldsymbol{\sigma}^{v,h} = 2\mu\nabla^s\boldsymbol{v}^h, \quad \boldsymbol{\sigma}^{p,h} = -p^h\boldsymbol{I}, \tag{2.96}$$

$$\boldsymbol{v}' = \tau^v\boldsymbol{r}^m, \tag{2.97}$$

$$p' = \tau^p r^c, \tag{2.98}$$

$$\tau^v = \left(\frac{c_1\mu}{h^2} + \frac{c_2\rho\|\boldsymbol{v}^a - \boldsymbol{v}^m\|}{h}\right)^{-1}, \tag{2.99}$$

$$\tau^p = \frac{h^2}{c_1\tau^v}, \tag{2.100}$$

$$\boldsymbol{r}^m = \boldsymbol{f} - \rho\partial_t\boldsymbol{v}^h - \rho\left((\boldsymbol{v}^a - \boldsymbol{v}^m)\cdot\nabla\right)\boldsymbol{v}^h + \nabla\cdot\left(\boldsymbol{\sigma}^{v,h}\right) + \nabla\cdot\left(\boldsymbol{\sigma}^{p,h}\right), \tag{2.101}$$

$$r^c = -\nabla\cdot\boldsymbol{v}^h. \tag{2.102}$$

The main difference between Eqs. 2.95–2.102 and the INS equations on fixed domains, Eqs. 2.11–2.15, is the appearance of the mesh velocity $\boldsymbol{v}^m$ in the convective terms in Eqs. 2.95, 2.99, and 2.101. Furthermore, the integrals are calculated on the current domain Ω_t. Also, note that the pressure-dependent component of the stress term, $\boldsymbol{\sigma}^{p,h}$ in Eq. 2.95, is kept intact for clearer presentation and comparison with the formulations presented hereafter.

The application of the G-α family of time integration schemes to Eqs. 2.95–2.102 follows mostly the same procedure as for the INS equations on a fixed domain. However, an important question arises regarding the instance of Ω_t, on which the terms of Eq. 2.95 are evaluated. The most common choice in the literature is to solve Eq. 2.95 on the most recent instance of Ω_t, i.e., $\Omega_{n+1} = \Omega\,(t_{n+1})$ in the time interval $[t_n, t_{n+1}]$. However, in [72], the equations are evaluated on $\Omega_{n+\alpha_f}$, the domain at the intermediate time instant $t_{n+\alpha_f}$, when using the G-α scheme. Similarly, in [73], the integrals are evaluated on

the domain at an intermediate time instant when using the one-step-θ scheme. In [63] it is argued that the equations should be solved on the instance of the moving domain corresponding to the unknowns, i.e., Ω_{n+1} in the case of the (end-of-step-equilibrium form of) one-step-θ scheme. A test case of INS equations on moving domains with constant-in-time analytical solution for pressure is studied in [63], where it is shown that numerical results using integration on the domain at an intermediate time instant, within $[t_n, t_{n+1}]$, does not preserve the expected constant-in-time value for pressure. A similar numerical test case will be later studied in the present work. Apart from the test case studied in [63], not many comparisons are available in the literature regarding the choice of the integration domain's instance when using the G-α or related time integration schemes. In the following, we will study, in more detail, the question of which instance of the moving domain Ω_t should be used when using the G-α family of time integration schemes and will later present numerical evidence supporting our choice.

In deriving Eqs. 2.95–2.102, only the inertial terms are written in the ALE reference frame, while all the other terms are stated with respect to the spatial frame, and the computations are carried out on the current domain Ω_t, which leads to a similar form as the INS equations in the Eulerian coordinates. The geometrical nonlinearity related to the moving domain is dealt with by explicitly moving and updating the computational domain in every time step. Another possibility is to transform all the terms in Eq. 2.95 to a fixed reference domain Ω_0 and, by doing so, avoid the need for moving the computational domain. In this case, the geometrical nonlinearity explicitly enters the INS equations. This approach is followed in [74, 50, 75, 49], among others. We will use this approach to pursue the "correct" choice of the integration domain when using the G-α family of time integration schemes.

A key component in the transformation between the spatial and the reference configurations is the deformation gradient (Jacobian) of the domain mapping

$$\mathbf{F} = \frac{\partial \mathcal{A}_t(\mathbf{x})}{\partial \mathbf{x}} = \frac{\partial \boldsymbol{x}(\mathbf{x}, t)}{\partial \mathbf{x}} = \nabla_{\mathrm{x}} \boldsymbol{x}, \tag{2.103}$$

and its determinant $\mathrm{J} = \det(\mathbf{F})$. We will also require the inverse and the transpose of the deformation gradient, $\mathbf{F}^{-1}$ and $\mathbf{F}^{T}$, respectively. For each function $\boldsymbol{v}$ in the spatial coordinates, the corresponding representation in the reference coordinates is denoted by upright type letter $\mathbf{v} = \boldsymbol{v} \circ \mathcal{A}_t$, i.e.,

$$\mathbf{v}(\mathbf{x}, t) = \boldsymbol{v}(\mathcal{A}_t(\mathbf{x}), t). \tag{2.104}$$

Given the function $\mathbf{v}$ stated in the reference configuration, its representation in the current configuration is denoted by $\boldsymbol{v} = \mathbf{v} \circ \mathcal{A}_t^{-1}$, i.e.,

$$\boldsymbol{v}(\boldsymbol{x}, t) = \mathbf{v}\left(\mathcal{A}_t^{-1}(\boldsymbol{x}), t\right). \tag{2.105}$$

We will later deal with the transformation of variables between domains at

two different time instants as well. For instance, when dealing with the time interval $[t_n, t_{n+1}]$, the variable $\boldsymbol{v}_n$ stated on domain Ω_{n+1} refers formally to $\boldsymbol{v}_n \circ \mathcal{A}_n \circ \mathcal{A}_{n+1}^{-1}$. However, such details are dropped in the following for the sake of brevity.

The transformation of the INS equations from the current configuration to the reference configuration can be split into two steps. First, all the integrals should be evaluated on the reference domain instead of the current domain, which is achieved with the help of the determinant of the deformation gradient

$$\int_{\Omega_t} 1 \, \mathrm{d}\Omega_t = \int_{\Omega_0} \mathrm{J} \, \mathrm{d}\Omega_0. \tag{2.106}$$

Secondly, all the spatial operators should be taken with respect to the reference coordinates instead of the spatial coordinates. The gradients of the scalar p, vector $\boldsymbol{v}$, and tensor $\boldsymbol{\sigma}$ in the spatial coordinates are respectively transformed to the reference coordinates by

$$\nabla p = \mathbf{F}^{-T} \nabla_{\mathrm{x}} \mathrm{p}, \tag{2.107}$$

$$\nabla \boldsymbol{v} = \nabla_{\mathrm{x}} \mathbf{v} \; \mathbf{F}^{-1}, \tag{2.108}$$

$$\nabla \boldsymbol{\sigma} = \nabla_{\mathrm{x}} \boldsymbol{\sigma} \; \mathbf{F}^{-1}. \tag{2.109}$$

The transformation of the divergence of vector $\boldsymbol{v}$, i.e., $\nabla \cdot \boldsymbol{v}$ to the reference configuration leads to

$$\nabla_{\mathrm{x}} \cdot \left(\mathrm{J} \mathbf{F}^{-1} \mathbf{v} \right). \tag{2.110}$$

The direct evaluation of Eq. 2.110 would seemingly require the second order derivatives of the domain's deformation with respect to $\mathbf{x}$. However, those second-order derivatives cancel out and the following relation ensues (see [76] for the proof):

$$\nabla_{\mathrm{x}} \cdot \left(\mathrm{J} \mathbf{F}^{-1} \mathbf{v} \right) = \mathrm{J} \, \mathrm{tr} \left(\nabla_{\mathrm{x}} \mathbf{v} \; \mathbf{F}^{-1} \right), \tag{2.111}$$

where $\mathrm{tr}\,()$ denotes the trace operator.

With these definitions at hand, the INS equations transformed to the reference domain read

$$\begin{aligned} &\left(\mathbf{w}^h, \rho \mathrm{J} \partial_t \mathbf{v}^h \right)_{\Omega_0} + \left(\mathbf{w}^h, \rho \mathrm{J} \left(\mathbf{F}^{-1} (\mathbf{v}^a - \mathbf{v}^m) \cdot \nabla_{\mathrm{x}} \right) \mathbf{v}^h \right)_{\Omega_0} + \left(\nabla_{\mathrm{x}} \mathbf{w}^h, \mathrm{J} \boldsymbol{\sigma}^{v,h} \mathbf{F}^{-T} \right)_{\Omega_0} \\ &\quad + \left(\nabla_{\mathrm{x}} \mathbf{w}^h, \mathrm{J} \boldsymbol{\sigma}^{p,h} \mathbf{F}^{-T} \right)_{\Omega_0} + \left(\mathrm{q}^h, \mathrm{J} \, \mathrm{tr} \left(\nabla_{\mathrm{x}} \mathbf{v}^h \; \mathbf{F}^{-1} \right) \right)_{\Omega_0} \\ &\quad - \left(\rho \left(\mathbf{F}^{-1} (\mathbf{v}^a - \mathbf{v}^m) \cdot \nabla_{\mathrm{x}} \right) \mathbf{w}^h + \mathbf{F}^{-T} \nabla_{\mathrm{x}} \mathrm{q}^h, \mathrm{J} \mathbf{v}' \right)_{\Omega_0^K} \\ &\quad - \left(\mathrm{tr} \left(\nabla_{\mathrm{x}} \mathbf{w}^h \; \mathbf{F}^{-1} \right), \mathrm{J} \mathrm{p}' \right)_{\Omega_0^K} = \left(\mathbf{w}^h, \mathrm{J} \mathbf{f} \right)_{\Omega_0}, \end{aligned} \tag{2.112}$$

with

$$\boldsymbol{\sigma}^{v,h} = \mu\left(\nabla_{\mathrm{x}}\mathbf{v}^h\ \mathbf{F}^{-1} + \mathbf{F}^{-T}\nabla_{\mathrm{x}}^{-T}\mathbf{v}^h\right) - \mathrm{p}^h\boldsymbol{I}, \quad \boldsymbol{\sigma}^{p,h} = -\mathrm{p}^h\boldsymbol{I}, \tag{2.113}$$

$$\mathbf{v}' = \tau^v\mathbf{r}^m, \tag{2.114}$$

$$\mathrm{p}' = \tau^p\mathrm{r}^c, \tag{2.115}$$

$$\tau^v = \left(\frac{c_1\mu}{\mathrm{h}^2} + \frac{c_2\rho\|\mathbf{v}^a - \mathbf{v}^m\|}{\mathrm{h}}\right)^{-1}, \tag{2.116}$$

$$\tau^p = \frac{\mathrm{h}^2}{c_1\tau^v}, \tag{2.117}$$

$$\mathbf{r}^m = \mathbf{f} - \rho\partial_t\mathbf{v}^h - \rho\left(\mathbf{F}^{-1}(\mathbf{v}^a - \mathbf{v}^m)\cdot\nabla_{\mathrm{x}}\right)\mathbf{v}^h + \nabla_{\mathrm{x}}\cdot\left(\boldsymbol{\sigma}^{v,h}\mathbf{F}^{-1}\right) + \nabla_{\mathrm{x}}\cdot\left(\boldsymbol{\sigma}^{p,h}\mathbf{F}^{-1}\right), \tag{2.118}$$

$$\mathrm{r}^c = -\operatorname{tr}\left(\nabla_{\mathrm{x}}\mathbf{v}^h\ \mathbf{F}^{-1}\right). \tag{2.119}$$

2.1.6.1 G-α time integration scheme for INS equations on moving domains

We now focus on the application of the G-α time integration scheme to Eqs. 2.112–2.119. The approach followed here is to apply the G-α scheme consistently to all terms, including the J and **F** terms that represent the effect of the geometrical nonlinearity of the domain's movement. Subsequently, the time discrete equations can be transformed back to the current configuration, which then determines the instance of the integration domain. According to the G-α rules the inertial term should be evaluated at $t_{n+\alpha_m}$, leading to

$$\left(\mathbf{w}^h, \rho\mathrm{J}_{n+\alpha_m}\tilde{\dot{\mathbf{v}}}^h_{n+\alpha_m}\right)_{\Omega_0}. \tag{2.120}$$

If transformed back to the the spatial configuration, Eq. 2.120 leads to

$$\left(\boldsymbol{w}^h, \rho\tilde{\dot{\boldsymbol{v}}}^h_{n+\alpha_m}\right)_{\Omega_{n+\alpha_m}}. \tag{2.121}$$

Similarly, the second term in Eq. 2.112 is integrated using the G-α scheme as

$$\left(\mathbf{w}^h, \rho\mathrm{J}_{n+\alpha_f}\left(\mathbf{F}^{-1}_{n+\alpha_f}(\mathbf{v}^a_{n+\alpha_f} - \mathbf{v}^m_{n+\alpha_f})\cdot\nabla_{\mathrm{x}}\right)\mathbf{v}^h_{n+\alpha_f}\right)_{\Omega_0}, \tag{2.122}$$

which, if transformed back to the spatial coordinates leads to

$$\left(\boldsymbol{w}^h, \rho\left((\boldsymbol{v}^a_{n+\alpha_f} - \boldsymbol{v}^m_{n+\alpha_f})\cdot\nabla\right)\boldsymbol{v}^h_{n+\alpha_f}\right)_{\Omega_{n+\alpha_f}}. \tag{2.123}$$

There are two issues with Eq. 2.120. The first issue is that, per definition, α_m is not bound to be in $[0,1]$. Therefore, for certain choices of the parameter

α_m, $\mathrm{J}_{n+\alpha_m}$ results from an extrapolation (as opposed to interpolation) of the mesh deformations at t_n and t_{n+1}. The extrapolated mesh deformation at $t_{n+\alpha_m}$ is not guaranteed to lead to a valid regular mapping between the reference and the current domains (Equivalently, one can say that sufficient mesh quality on $\Omega_{n+\alpha_m}$, used in Eq. 2.121, is not guaranteed). In contrast, α_f is bound to be in the interval $[0,1]$ and, therefore, $\mathrm{J}_{n+\alpha_f}$ is always an interpolation between J_n and J_{n+1}, which are expected to have reasonable values by solving an auxiliary mesh deformation (mesh moving) problem at t_n and t_{n+1}. The second issue with Eq. 2.120 is that it corresponds to integration on the domain $\Omega_{n+\alpha_m}$ if transformed back to the spatial coordinates, while all the other terms would be integrated on $\Omega_{n+\alpha_f}$. Evaluation of terms on two instances of the domain leads to an extra complexity and cost of the solution procedure when solving the equations on moving domains (obviously, this disadvantage does not exist when solving the equations on the reference domain). In order to alleviate this issue, we make use of the true accelerations defined in section 2.1.5. Recall that the true acceleration $\dot{\boldsymbol{v}}^h$ is evaluated at the intermediate time instant $t_{n+\alpha_f}$, in contrast to the auxiliary acceleration variable $\tilde{\dot{\boldsymbol{v}}}^h$. Therefore, the time-discrete form of the inertial term in Eq. 2.112 can be restated as

$$\left(\mathbf{w}^h, \rho \mathrm{J}_{n+\alpha_f} \dot{\mathbf{v}}^h_{n+\alpha_f}\right)_{\Omega_0}. \tag{2.124}$$

One can then use the relation $\dot{\mathbf{v}}^h_{n+\alpha_f} = \tilde{\dot{\mathbf{v}}}^h_{n+\alpha_m}$ in order to rewrite Eq. 2.124 as

$$\left(\mathbf{w}^h, \rho \mathrm{J}_{n+\alpha_f} \tilde{\dot{\mathbf{v}}}^h_{n+\alpha_m}\right)_{\Omega_0}, \tag{2.125}$$

which, if transformed back to the spatial coordinates, leads to

$$\left(\boldsymbol{w}^h, \rho \tilde{\dot{\boldsymbol{v}}}^h_{n+\alpha_m}\right)_{\Omega_{n+\alpha_f}}. \tag{2.126}$$

Similar to the case of applying the G-α scheme to INS equations on the fixed domains, one can think of three variants of the G-α scheme for INS on moving domains: the direct, shortened, and p–$O(\Delta t)$ variants. As before, the difference between the three variants is in the time discretization of the pressure term and the continuity equation. The direct G-α scheme is derived by direct application of the G-α scheme to all terms of the equations in a straightforward manner. Therefore, the deformation gradient and its determinant, $\mathbf{F}$ and J, in all of the terms are evaluated at $t_{n+\alpha_f}$ when solving the system in the fixed reference configuration. Equivalently, all the terms are evaluated on the domain $\Omega_{n+\alpha_f}$ when solving the system in the current configuration. The disadvantage of evaluating the integrals on a domain at the intermediate time instant is the need to form the additional domain $\Omega_{n+\alpha_f}$ as the computational mesh in a discrete problem. In addition to the extra complexity and computational cost of handling the intermediate computational mesh, it may sometimes be impossible (or prohibitively expensive) to form

$\Omega_{n+\alpha_f}$, e.g., when the spatial discretization changes between two time steps or when using embedded or immersed techniques. The most common alternative is evaluating the gradients and integrals on the domain Ω_{n+1}, the most recent instance of the moving domain, and apply the G-α rules only to the velocity and pressure terms. This approach translates to evaluating the deformation gradient and its determinant, $\mathbf{F}$ and J, in all terms, at t_{n+1} when solving the system in the fixed reference configuration. However, using F_{n+1} and J_{n+1} in the equations is inconsistent with the G-α rules. We will compare the validity and accuracy of the two choices for the integration domain in the numerical studies presented in section 2.1.7. By consistent application of the direct G-α scheme, the time discretization of the pressure term and the continuity equation, formulated in the reference configuration read, respectively

$$\left(\nabla_{\mathrm{x}}\mathbf{w}^h, \mathrm{J}_{n+\alpha_f}\boldsymbol{\sigma}^{p,h}_{n+\alpha_f}\mathbf{F}^{-T}_{n+\alpha_f}\right)_{\Omega_0}, \tag{2.127}$$

$$\left(\mathrm{q}^h, \mathrm{J}_{n+\alpha_f}\,\mathrm{tr}\left(\nabla_{\mathrm{x}}\mathbf{v}^h_{n+\alpha_f}\,\mathbf{F}^{-1}_{n+\alpha_f}\right)\right)_{\Omega_0} = 0. \tag{2.128}$$

The derivation of the shortened and p–$O(\Delta t)$ forms of the G-α scheme is not as straightforward as the direct form when dealing with INS equations on moving domains. Recall that on fixed domains, the shortened G-α scheme is derived by reducing the continuity equation from $\nabla\cdot\boldsymbol{v}^h_{n+\alpha_f} = 0$ to $\nabla\cdot\boldsymbol{v}^h_{n+1} = 0$. Extending this idea to the case of moving domains, one may try to replace Eq. 2.128 with

$$\left(\mathrm{q}^h, \mathrm{J}_{n+\alpha_f}\,\mathrm{tr}\left(\nabla_{\mathrm{x}}\mathbf{v}^h_{n+1}\,\mathbf{F}^{-1}_{n+\alpha_f}\right)\right)_{\Omega_0} = 0. \tag{2.129}$$

The advantage of Eq. 2.129 is that $\mathbf{F}$ and J in the continuity equation are evaluated at $t_{n+\alpha_f}$, which is the same in all the other terms. This choice amounts to evaluating all terms on the same instance of the domain $\Omega_{n+\alpha_f}$ when solving the system in the current configuration. However, the calculated velocity $\boldsymbol{v}^h_{n+1}$ would be divergence-free on the domain $\Omega_{n+\alpha_f}$ and not on Ω_{n+1}. Therefore, one can instead restate the shortened form of the continuity equation in the reference configuration as

$$\left(\mathrm{q}^h, \mathrm{J}_{n+1}\,\mathrm{tr}\left(\nabla_{\mathrm{x}}\mathbf{v}^h_{n+1}\,\mathbf{F}^{-1}_{n+1}\right)\right)_{\Omega_0} = 0. \tag{2.130}$$

Eq. 2.130 can be justified by evaluating the continuity equation using a GTQ rule instead of the GMQ rule and assuming that $\left(\mathrm{q}^h, \mathrm{J}_n\,\mathrm{tr}\left(\nabla_{\mathrm{x}}\mathbf{v}^h_n\,\mathbf{F}^{-1}_n\right)\right)_{\Omega_0} = 0$ holds from the previous time step. When transforming the equations to the current configuration, the disadvantage of Eq. 2.130 is that it equates to evaluating the continuity equation on the domain Ω_{n+1}, while all the other terms are evaluated on $\Omega_{n+\alpha_f}$. Therefore, we will need to deal with multiple instances of the domain. Regarding the pressure term using the shortened form of G-α, if the continuity equation is discretized according to Eq. 2.129,

then the pressure term is unchanged compared to the direct form of the G-α scheme and follows Eq. 2.127. However, if the continuity equation is discretized using Eq. 2.130, it seems logical to discretize the pressure term in time also using the GTQ rule leading to

$$(1-\alpha_f)\left(\nabla_{\mathrm{x}}\mathbf{w}^h, \mathrm{J}_n\boldsymbol{\sigma}_n^{p,h}\mathbf{F}_n^{-T}\right)_{\Omega_0} + \alpha_f\left(\nabla_{\mathrm{x}}\mathbf{w}^h, \mathrm{J}_{n+1}\boldsymbol{\sigma}_{n+1}^{p,h}\mathbf{F}_{n+1}^{-T}\right)_{\Omega_0}, \tag{2.131}$$

which is equal to the pressure term in Eq. 2.127 if the domain's deformation is linear in time. After transforming the equations back to the current configuration, the time discretization of the pressure term corresponding to Eq. 2.131 prompts evaluations on yet another instance of the domain Ω_n, necessary for the first part of Eq. 2.131, in addition to already evaluating some terms on $\Omega_{n+\alpha_f}$ and Ω_{n+1}. Furthermore, more ambiguities arise in the time discretization of the stabilization terms due to the residual-based stabilization technique used in the present work. In the weak form in Eq. 2.112, the term containing the subgrid-scale velocity $\mathbf{v}'$ comprises many terms, including contribution from the pressure gradient in the strong form of the large-scale residuals (see Eq. 2.118). Discretizing the pressure in the stabilization terms using a GTQ rule in time, similar to Eq. 2.131, leads to terms in which $\mathbf{F}$ and J are evaluated at different time instants. Such terms would not be straightforwardly transformed to the current configuration. For the reasons just mentioned and considering that in many cases, the domain's deformation behaves almost linearly in time in each time step, we choose to discretize the pressure term in time using Eq. 2.127 when using the shortened form of the G-α scheme.

The p–$O(\Delta t)$ form of the G-α scheme is derived similarly to the shortened form. The continuity equation is discretized using Eq. 2.130. Additionally, the pressure term is also evaluated at t_{n+1} as

$$\left(\nabla_{\mathrm{x}}\mathbf{w}^h, \mathrm{J}_{n+1}\boldsymbol{\sigma}_{n+1}^{p,h}\mathbf{F}_{n+1}^{-T}\right)_{\Omega_0}. \tag{2.132}$$

Using the p–$O(\Delta t)$ G-α, difficulties surround the time discretization of the terms containing the fine-scale velocities, similar to the shortened form of the G-α scheme discussed previously. For that reason, we will also analyze and compare the behavior of the p–$O(\Delta t)$ G-α variant when using inf-sup stable P2P1 (Taylor-Hood) elements without stabilization terms.

In order to concisely present and later compare different variants of the G-α scheme using different choices for the integration domain, we define and make use of the variables α_1, α_2, α_3, α_4, and α_5, which are used to identify time instants. The time evaluation of many terms in the INS equations is shared among all three variants of the G-α scheme. The terms making a difference among the variants are identified using the newly defined variables. α_1 is used for the pressure term, α_2 for the divergence of the velocity, α_3 for the integration domains (or for deformation gradient and its determinant), α_4 for the integration domain of the pressure term specifically, and α_5 for

the integration domain of the continuity equation. Each of the variables α_1, α_2, α_3, α_4, and α_5 can take a value from the set $\{\alpha_f, 1\}$. For example, for $\alpha_3 = \alpha_f$, we have $\Omega_{n+\alpha_3} = \Omega_{n+\alpha_f}$.

The final form of the time-discretized INS equations on the reference domain using the three variants of the G-α scheme can by concisely stated using

$$
\begin{aligned}
R(\alpha_1,\alpha_2,\alpha_3,\alpha_4,\alpha_5) &= \left(\mathbf{w}^h, \rho \mathrm{J}_{n+\alpha_3} \tilde{\dot{\mathbf{v}}}^h_{n+\alpha_m}\right)_{\Omega_0} \\
&+ \left(\mathbf{w}^h, \rho \mathrm{J}_{n+\alpha_3} \left(\mathbf{F}^{-1}_{n+\alpha_3}(\mathbf{v}^a_{n+\alpha_f} - \mathbf{v}^m_{n+\alpha_f}) \cdot \nabla_{\mathrm{x}}\right) \mathbf{v}^h_{n+\alpha_f}\right)_{\Omega_0} \\
&+ \left(\nabla_{\mathrm{x}} \mathbf{w}^h, \mathrm{J}_{n+\alpha_3} \boldsymbol{\sigma}^{v,h}_{n+\alpha_f} \mathbf{F}^{-T}_{n+\alpha_3}\right)_{\Omega_0} + \left(\nabla_{\mathrm{x}} \mathbf{w}^h, \mathrm{J}_{n+\alpha_4} \boldsymbol{\sigma}^{p,h}_{n+\alpha_1} \mathbf{F}^{-T}_{n+\alpha_4}\right)_{\Omega_0} \\
&+ \left(\mathrm{q}^h, \mathrm{J}_{n+\alpha_5} \operatorname{tr}\left(\nabla_{\mathrm{x}} \mathbf{v}^h_{n+\alpha_2} \ \mathbf{F}^{-1}_{n+\alpha_5}\right)\right)_{\Omega_0} \\
&- \left(\rho \left(\mathbf{F}^{-1}_{n+\alpha_3}(\mathbf{v}^a_{n+\alpha_f} - \mathbf{v}^m_{n+\alpha_f}) \cdot \nabla_{\mathrm{x}}\right) \mathbf{w}^h + \mathbf{F}^{-T}_{n+\alpha_3} \nabla_{\mathrm{x}} \mathrm{q}^h, \mathrm{J}_{n+\alpha_3} \mathbf{v}'_{n+\alpha_f}\right)_{\Omega^K_0} \\
&- \left(\operatorname{tr}\left(\nabla_{\mathrm{x}} \mathbf{w}^h \ \mathbf{F}^{-1}_{n+\alpha_5}\right), \mathrm{J}_{n+\alpha_5} \mathrm{p}'_{n+\alpha_2}\right)_{\Omega^K_0} = \left(\mathbf{w}^h, \mathrm{J}_{n+\alpha_3} \mathbf{f}_{n+\alpha_f}\right)_{\Omega_0}, \qquad (2.133)
\end{aligned}
$$

with

$$
\boldsymbol{\sigma}^{v,h}_{n+\alpha_f} = \mu \left(\nabla_{\mathrm{x}} \mathbf{v}^h_{n+\alpha_f} \ \mathbf{F}^{-1}_{n+\alpha_3} + \mathbf{F}^{-T}_{n+\alpha_3} \nabla^{-T}_{\mathrm{x}} \mathbf{v}^h_{n+\alpha_f}\right), \quad \boldsymbol{\sigma}^{p,h}_{n+\alpha_1} = -\mathrm{p}^h_{n+\alpha_1} \boldsymbol{I}, \qquad (2.134)
$$

$$
\mathbf{v}'_{n+\alpha_f} = \tau^v_{n+\alpha_f} \mathbf{r}^m_{n+\alpha_f}, \qquad (2.135)
$$

$$
\mathrm{p}'_{n+\alpha_2} = \tau^p_{n+\alpha_f} \mathrm{r}^c_{n+\alpha_2}, \qquad (2.136)
$$

$$
\tau^v_{n+\alpha_f} = \left(\frac{c_1 \mu}{\mathrm{h}^2} + \frac{c_2 \rho \| \mathbf{v}^a_{n+\alpha_f} - \mathbf{v}^m_{n+\alpha_f} \|}{\mathrm{h}}\right)^{-1}, \qquad (2.137)
$$

$$
\tau^p_{n+\alpha_f} = \frac{\mathrm{h}^2}{c_1 \tau^v_{n+\alpha_f}}, \qquad (2.138)
$$

$$
\begin{aligned}
\mathbf{r}^m_{n+\alpha_f} =& \mathbf{f}_{n+\alpha_f} - \rho \tilde{\dot{\mathbf{v}}}^h_{n+\alpha_m} - \rho \left(\mathbf{F}^{-1}_{n+\alpha_3}(\mathbf{v}^a_{n+\alpha_f} - \mathbf{v}^m_{n+\alpha_f}) \cdot \nabla_{\mathrm{x}}\right) \mathbf{v}^h_{n+\alpha_f} \\
&+ \nabla_{\mathrm{x}} \cdot \left(\boldsymbol{\sigma}^{v,h}_{n+\alpha_f} \mathbf{F}^{-1}_{n+\alpha_3}\right) + \nabla_{\mathrm{x}} \cdot \left(\boldsymbol{\sigma}^{p,h}_{n+\alpha_1} \mathbf{F}^{-1}_{n+\alpha_4}\right), \qquad (2.139)
\end{aligned}
$$

$$
\mathrm{r}^c_{n+\alpha_2} = -\operatorname{tr}\left(\nabla_{\mathrm{x}} \mathbf{v}^h_{n+\alpha_2} \ \mathbf{F}^{-1}_{n+\alpha_5}\right). \qquad (2.140)
$$

Transforming Eqs. 2.133–2.140 to the current domain leads to the final form of the INS equations on the current moving domain and discretized in time

using the three variants of the G-α scheme

$$
\begin{aligned}
R(\alpha_1,\alpha_2,\alpha_3,\alpha_4,\alpha_5) &= \left(\boldsymbol{w}^h, \rho\tilde{\dot{\boldsymbol{v}}}^h_{n+\alpha_m}\right)_{\Omega_{n+\alpha_3}} \\
&+ \left(\boldsymbol{w}^h, \rho\left((\boldsymbol{v}^a_{n+\alpha_f} - \boldsymbol{v}^m_{n+\alpha_f})\cdot\nabla\right)\boldsymbol{v}^h_{n+\alpha_f}\right)_{\Omega_{n+\alpha_3}} \\
&+ \left(\nabla\boldsymbol{w}^h, \boldsymbol{\sigma}^{v,h}_{n+\alpha_f}\right)_{\Omega_{n+\alpha_3}} + \left(\nabla\boldsymbol{w}^h, \boldsymbol{\sigma}^{p,h}_{n+\alpha_1}\right)_{\Omega_{n+\alpha_4}} \\
&+ \left(q^h, \nabla\cdot\boldsymbol{v}^h_{n+\alpha_2}\right)_{\Omega_{n+\alpha_5}} - \left(\rho\left((\boldsymbol{v}^a_{n+\alpha_f} - \boldsymbol{v}^m_{n+\alpha_f})\cdot\nabla\right)\boldsymbol{w}^h + \nabla q^h, \boldsymbol{v}'_{n+\alpha_f}\right)_{\Omega^K_{n+\alpha_3}} \\
&- \left(\nabla\cdot\boldsymbol{w}^h, p'_{n+\alpha_2}\right)_{\Omega^K_{n+\alpha_5}} = \left(\boldsymbol{w}^h, \boldsymbol{f}_{n+\alpha_f}\right)_{\Omega_{n+\alpha_3}},
\end{aligned}
\tag{2.141}
$$

with

$$
\boldsymbol{\sigma}^{v,h}_{n+\alpha_f} = 2\mu\nabla^s\boldsymbol{v}^h_{n+\alpha_f}, \quad \boldsymbol{\sigma}^{p,h}_{n+\alpha_1} = -p^h_{n+\alpha_1}\boldsymbol{I}, \tag{2.142}
$$

$$
\boldsymbol{v}'_{n+\alpha_f} = \tau^v_{n+\alpha_f}\boldsymbol{r}^m_{n+\alpha_f}, \tag{2.143}
$$

$$
p'_{n+\alpha_2} = \tau^p_{n+\alpha_f} r^c_{n+\alpha_2}, \tag{2.144}
$$

$$
\tau^v_{n+\alpha_f} = \left(\frac{c_1\mu}{h^2} + \frac{c_2\rho\|\boldsymbol{v}^a_{n+\alpha_f} - \boldsymbol{v}^m_{n+\alpha_f}\|}{h}\right)^{-1}, \tag{2.145}
$$

$$
\tau^p_{n+\alpha_f} = \frac{h^2}{c_1\tau^v_{n+\alpha_f}}, \tag{2.146}
$$

$$
\begin{aligned}
\boldsymbol{r}^m_{n+\alpha_f} &= \boldsymbol{f}_{n+\alpha_f} - \rho\tilde{\dot{\boldsymbol{v}}}^h_{n+\alpha_m} - \rho\left((\boldsymbol{v}^a_{n+\alpha_f} - \boldsymbol{v}^m_{n+\alpha_f})\cdot\nabla\right)\boldsymbol{v}^h_{n+\alpha_f} \\
&\quad + \nabla\cdot\left(\boldsymbol{\sigma}^{v,h}_{n+\alpha_f}\right) + \nabla\cdot\left(\boldsymbol{\sigma}^{p,h}_{n+\alpha_1}\right),
\end{aligned}
\tag{2.147}
$$

$$
r^c_{n+\alpha_2} = -\nabla\cdot\boldsymbol{v}^h_{n+\alpha_2}. \tag{2.148}
$$

Different variants of the G-α scheme for INS equations on moving domains are gathered in Table 2.1, together with their respective choices for the parameters α_1, α_2, α_3, α_4, and α_5. The naming of the G-α variants in Table 2.1 conforms best to the case of solving the equations in the current configuration. Despite that, we also use the same naming convention for the corresponding G-α variants when solving the equations in the reference configuration. A comparison of the accuracy of the G-α variants in Table 2.1 will be carried out in section 2.1.7.

The mesh velocity $\mathbf{v}^m_{n+\alpha_f}$ in Eq. 2.133, equivalently $\boldsymbol{v}^m_{n+\alpha_f}$ in Eq. 2.141, is evaluated at the intermediate time instant $t_{n+\alpha_f}$ and is so far assumed to be exact in time through an analytical solution. In practice, the discrete mesh velocity is calculated from the mesh deformation. If the domain's boundaries deform, the internal mesh should be adapted to preserve a reasonably good mesh quality (or equivalently, to keep a reasonably smooth ALE mapping if the

Table 2.1: Different variants of the G-α scheme for INS equations on moving domains.

G-α variant's identifier	**Parameter values**
direct G-α on $\Omega_{n+\alpha_f}$	$\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4 = \alpha_5 = \alpha_f$
direct G-α on Ω_{n+1}	$\alpha_1 = \alpha_2 = \alpha_f, \alpha_3 = \alpha_4 = \alpha_5 = 1$
shortened G-α, multi-domain	$\alpha_1 = \alpha_3 = \alpha_4 = \alpha_f, \alpha_2 = \alpha_5 = 1$
shortened G-α on $\Omega_{n+\alpha_f}$	$\alpha_1 = \alpha_3 = \alpha_4 = \alpha_5 = \alpha_f, \alpha_2 = 1$
shortened G-α on Ω_{n+1}	$\alpha_1 = \alpha_f, \alpha_2 = \alpha_3 = \alpha_4 = \alpha_5 = 1$
p–$O(\Delta t)$ G-α, multi-domain	$\alpha_3 = \alpha_f, \alpha_1 = \alpha_2 = \alpha_4 = \alpha_5 = 1$
p–$O(\Delta t)$ G-α, multi-domain, P2P1	$\alpha_3 = \alpha_f, \alpha_1 = \alpha_2 = \alpha_4 = \alpha_5 = 1$
p–$O(\Delta t)$ G-α on Ω_{n+1}	$\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4 = \alpha_5 = 1$

equations are solved in the reference domain). The mesh adaptation is usually carried out by solving an auxiliary problem called the *mesh moving* or *mesh motion* problem at the discrete time points $t_i, i = 1, \ldots, n+1$. The outcome of the mesh moving problem is the mesh displacement $\mathbf{d}_i^m$, $i = 1, \ldots, n+1$, at the discrete time points. See, e.g., [77, 78] for details on different mesh moving strategies. In [72, 79], the mesh velocity $\mathbf{v}_{n+1}^m$ is taken to be equal to $\tilde{\dot{\mathbf{d}}}_{n+1}^m$, the time derivative of the mesh displacement according to the G-α scheme for the first-order systems

$$\tilde{\dot{\mathbf{d}}}_{n+1}^m = \frac{1}{\gamma^m \Delta t}\left(\mathbf{d}_{n+1}^m - \mathbf{d}_n^m\right) - \frac{1-\gamma^m}{\gamma^m}\left(\tilde{\dot{\mathbf{d}}}_n^m\right), \tag{2.149}$$

where γ^m is chosen to be equal to the γ parameter used for the time integration of the fluid problem, which is in turn dependent on the α_f and α_m parameters of the G-α scheme for the INS equations. Subsequently, $\tilde{\dot{\mathbf{d}}}_{n+\alpha_f}^m$ is calculated to be used as the weighted average mesh velocity in Eq. 2.133 or 2.141. However, as explained in section 2.1.5.1, the time derivative calculated using Eq. 2.149, in conjunction with γ, α_f and α_m of the G-α scheme, is in general only first-order accurate and is not as accurate as the true time derivative of the mesh displacement $\dot{\mathbf{d}}_{n+1}^m$, calculated as

$$\dot{\mathbf{d}}_{n+1}^m = \frac{\alpha_m}{\alpha_f \gamma \Delta t}\left(\mathbf{d}_{n+1}^m - \mathbf{d}_n^m\right) - \frac{(1-\alpha_f)}{\alpha_f}\dot{\mathbf{d}}_n^m - \frac{(\alpha_m - \gamma)}{\alpha_f \gamma}\tilde{\dot{\mathbf{d}}}_n^m. \tag{2.150}$$

Therefore, one approach for accurately calculating the mesh velocity is to calculate the true time derivative of the mesh displacement at t_{n+1}, according to Eq. 2.150, and later use $\mathbf{v}_{n+\alpha_f}^m = \dot{\mathbf{d}}_{n+\alpha_f}^m$ as the weighted average of the mesh velocity in Eq. 2.133 or 2.141. Another approach is to calculate $\tilde{\dot{\mathbf{d}}}_{n+1}^m$ according to Eq. 2.149 and later use $\tilde{\dot{\mathbf{d}}}_{n+\alpha_m}^m$ (and not $\tilde{\dot{\mathbf{d}}}_{n+\alpha_f}^m$) as the weighted

average of the mesh velocity in Eq. 2.133 or 2.141. The latter approach is valid based on the relation $\tilde{\mathbf{d}}^m_{n+\alpha_m} = \dot{\mathbf{d}}^m_{n+\alpha_f} = \mathbf{v}^m_{n+\alpha_f}$, shown previously in section 2.1.5.1. Furthermore, it is also more consistent with the original rules of the G-α scheme to evaluate the time derivative terms at the intermediate time instant $t_{n+\alpha_m}$.

The conclusion we would like to draw from the present investigation is that the consistent application of the G-α scheme to the INS equations on moving domains requires the evaluation of the integrals on the domain $\Omega_{n+\alpha_f}$, and that the integration on domain Ω_{n+1} is not consistent with the time integration scheme. Numerical studies will be carried out in section 2.1.7 in order to verify this point.

2.1.6.2 EG-α time integration scheme for INS equations on moving domains

Recall from section 2.1.5 that in the EG-α scheme, the equilibrium equation in the interval $[t_n, t_{n+1}]$ is satisfied at t_{n+1} in contrast to the weighted equilibrium equation in the G-α scheme. This advantage of the EG-α scheme is of even greater importance in the context of INS equations on moving domains. The INS equations transformed to the reference domain and discretized in time using the EG-α reads

$$\begin{aligned}
&\left(\mathbf{w}^h, \rho \mathrm{J}_{n+1}\dot{\mathbf{v}}^h_{n+1}\right)_{\Omega_0} + \left(\mathbf{w}^h, \rho \mathrm{J}_{n+}\left(\mathbf{F}^{-1}_{n+1}(\mathbf{v}^a_{n+1} - \mathbf{v}^m_{n+1})\cdot\nabla_{\mathrm{x}}\right)\mathbf{v}^h_{n+1}\right)_{\Omega_0}\\
&+ \left(\nabla_{\mathrm{x}}\mathbf{w}^h, \mathrm{J}_{n+1}\boldsymbol{\sigma}^h_{n+1}\mathbf{F}^{-T}_{n+1}\right)_{\Omega_0} + \left(\mathrm{q}^h, \mathrm{J}_{n+1}\,\mathrm{tr}\left(\nabla_{\mathrm{x}}\mathbf{v}^h_{n+1}\ \mathbf{F}^{-1}_{n+1}\right)\right)_{\Omega_0}\\
&- \left(\rho\left(\mathbf{F}^{-1}_{n+1}(\mathbf{v}^a_{n+1} - \mathbf{v}^m_{n+1})\cdot\nabla_{\mathrm{x}}\right)\mathbf{w}^h + \mathbf{F}^{-T}_{n+1}\nabla_{\mathrm{x}}\mathrm{q}^h, \mathrm{J}_{n+1}\mathbf{v}'_{n+1}\right)_{\Omega^K_0}\\
&- \left(\mathrm{tr}\left(\nabla_{\mathrm{x}}\mathbf{w}^h\ \mathbf{F}^{-1}_{n+1}\right), \mathrm{J}_{n+1}\mathrm{p}'_{n+1}\right)_{\Omega^K_0} = \left(\mathbf{w}^h, \mathrm{J}_{n+1}\mathbf{f}_{n+1}\right)_{\Omega_0},
\end{aligned} \tag{2.151}$$

with

$$\boldsymbol{\sigma}^h_{n+1} = \mu\left(\nabla_{\mathrm{x}}\mathbf{v}^h_{n+1}\ \mathbf{F}^{-1}_{n+1} + \mathbf{F}^{-T}_{n+1}\nabla^{-T}_{\mathrm{x}}\mathbf{v}^h_{n+1}\right) - \mathrm{p}^h_{n+1}\boldsymbol{I}, \tag{2.152}$$

$$\mathbf{v}'_{n+1} = \tau^v_{n+1}\mathbf{r}^m_{n+1}, \tag{2.153}$$

$$\mathrm{p}'_{n+1} = \tau^p_{n+1}\mathrm{r}^c_{n+1}, \tag{2.154}$$

$$\tau^v_{n+1} = \left(\frac{c_1\mu}{\mathrm{h}^2} + \frac{c_2\rho\|\mathbf{v}^a_{n+1} - \mathbf{v}^m_{n+1}\|}{\mathrm{h}}\right)^{-1}, \tag{2.155}$$

$$\tau^p_{n+1} = \frac{\mathrm{h}^2}{c_1\tau^v_{n+1}}, \tag{2.156}$$

$$\mathbf{r}_{n+1}^{m} = \mathbf{f}_{n+1} - \rho\dot{\mathbf{v}}_{n+1}^{h} - \rho\left(\mathbf{F}_{n+1}^{-1}(\mathbf{v}_{n+1}^{a} - \mathbf{v}_{n+1}^{m})\cdot\nabla_{\mathrm{x}}\right)\mathbf{v}_{n+1}^{h} + \nabla_{\mathrm{x}}\cdot\left(\boldsymbol{\sigma}_{n+1}^{h}\mathbf{F}_{n+1}^{-1}\right), \tag{2.157}$$

$$\mathrm{r}_{n+1}^{c} = -\,\mathrm{tr}\left(\nabla_{\mathrm{x}}\mathbf{v}_{n+1}^{h}\ \mathbf{F}_{n+1}^{-1}\right). \tag{2.158}$$

The mesh velocity $\mathbf{v}_{n+1}^{m}$ is either known analytically or is taken to be equal to the true time derivative of the mesh displacement, i.e., $\mathbf{v}_{n+1}^{m} = \dot{\mathbf{d}}_{n+1}^{m}$, where $\dot{\mathbf{d}}_{n+1}^{m}$ is calculated according to Eq. 2.150.

Transforming Eqs. 2.151–2.158 back to the spatial configuration leads to the final form of the INS equations on moving domains discretized in time by the EG-α scheme

$$\begin{aligned}
&\left(\boldsymbol{w}^{h}, \rho\dot{\boldsymbol{v}}_{n+1}^{h}\right)_{\Omega_{n+1}} + \left(\boldsymbol{w}^{h}, \rho\left((\boldsymbol{v}_{n+1}^{a} - \boldsymbol{v}_{n+1}^{m})\cdot\nabla\right)\boldsymbol{v}_{n+1}^{h}\right)_{\Omega_{n+1}} + \left(\nabla\boldsymbol{w}^{h}, \boldsymbol{\sigma}_{n+1}^{h}\right)_{\Omega_{n+1}} \\
&+ \left(q^{h}, \nabla\cdot\boldsymbol{v}_{n+1}^{h}\right)_{\Omega_{n+1}} - \left(\rho\left((\boldsymbol{v}_{n+1}^{a} - \boldsymbol{v}_{n+1}^{m})\cdot\nabla\right)\boldsymbol{w}^{h} + \nabla q^{h}, \boldsymbol{v}_{n+1}^{\prime}\right)_{\Omega_{n+1}^{K}} \\
&- \left(\nabla\cdot\boldsymbol{w}^{h}, p_{n+1}^{\prime}\right)_{\Omega_{n+1}^{K}} = \left(\boldsymbol{w}^{h}, \boldsymbol{f}_{n+1}\right)_{\Omega_{n+1}},
\end{aligned} \tag{2.159}$$

with

$$\boldsymbol{\sigma}_{n+1}^{h} = 2\mu\nabla^{s}\boldsymbol{v}_{n+1}^{h} - p_{n+1}^{h}\boldsymbol{I}, \tag{2.160}$$

$$\boldsymbol{v}_{n+1}^{\prime} = \tau_{n+1}^{v}\boldsymbol{r}_{n+1}^{m}, \tag{2.161}$$

$$p_{n+1}^{\prime} = \tau_{n+1}^{p} r_{n+1}^{c}, \tag{2.162}$$

$$\tau_{n+1}^{v} = \left(\frac{c_1\mu}{h^2} + \frac{c_2\rho\|\boldsymbol{v}_{n+1}^{a} - \boldsymbol{v}_{n+1}^{m}\|}{h}\right)^{-1}, \tag{2.163}$$

$$\tau_{n+1}^{p} = \frac{h^2}{c_1\tau_{n+1}^{v}}, \tag{2.164}$$

$$\boldsymbol{r}_{n+1}^{m} = \boldsymbol{f}_{n+1} - \rho\dot{\boldsymbol{v}}_{n+1}^{h} - \rho\left((\boldsymbol{v}_{n+1}^{a} - \boldsymbol{v}_{n+1}^{m})\cdot\nabla\right)\boldsymbol{v}_{n+1}^{h} + \nabla\cdot\left(\boldsymbol{\sigma}_{n+1}^{h}\right), \tag{2.165}$$

$$r_{n+1}^{c} = -\nabla\cdot\boldsymbol{v}_{n+1}^{h}. \tag{2.166}$$

The main advantage of the EG-α scheme over the G-α scheme for the INS equations on moving domains is that the integrals are directly evaluated on Ω_{n+1}, and there is no need for evaluations on $\Omega_{n+\alpha_f}$.

2.1.7 Results and discussion

In this section, two examples of flows with analytical solutions on fixed domains are used to study the temporal accuracy of the velocity, pressure and their time derivatives using different variants of the G-α and EG-α schemes. Afterwards, two numerical benchmarks with more complex flow structures are presented and the stability and accuracy of the schemes, using different

algorithmic parameters, are compared. Finally, two numerical benchmarks of flows on moving domains are presented. In addition to studying similar quantities of interest as in flows on fixed grids, the effect of the choice of the integration domain on the accuracy of the results will be studied.

2.1.7.1 Pulsatile flow in a channel

In this example, a fully developed flow in a channel subjected to an oscillatory streamwise pressure gradient is studied. The channel is assumed to be oriented along the x-axis. The pressure gradient comprises constant and oscillatory components and is defined by the Fourier series

$$-\frac{\partial p}{\partial x} = g_0 + \sum_{n=1}^{N} g_n e^{in\omega t}, \tag{2.167}$$

where i is the imaginary unit, g_0 is the absolute value of the constant part of the pressure gradient, g_n denotes the amplitude of the oscillatory part of the pressure gradient at the n-th mode, and ω is the fundamental angular frequency of the oscillations defined by $\omega = 2\pi/T$, with T being the period of oscillation.

The velocity in the streamwise direction is defined by [80]

$$\boldsymbol{v}_x = \frac{g_0}{2\mu}\left(r^2 - y^2\right) + i\sum_{n=1}^{N}\frac{g_n}{\rho n\omega}\left(\frac{\cosh\left(y\sqrt{in\omega/\nu}\right)}{\cosh\left(r\sqrt{in\omega/\nu}\right)} - 1\right)e^{in\omega t}, \tag{2.168}$$

where the first and the second terms on the right-hand side of Eq. 2.168 correspond, respectively, to the constant and oscillatory components of the velocity. Furthermore, y denotes the y-coordinate measured from the middle of the channel (see Fig. 2.1), and r is equal to $h/2$, where h is the width of the channel. ρ, μ, and ν are the constant density, dynamic viscosity and kinematic viscosity values, respectively. Eq. 2.168 leads to a complex value, and its real component is used as the reference value for the velocity.

The analytical solution of the pressure takes the form

$$p = -\left(g_0 + \sum_{n=1}^{N} g_n e^{in\omega t}\right)(x - l) + p_0, \tag{2.169}$$

where p_0 is the reference pressure value at the outlet of the channel (see Fig 2.1), x is the distance from the inlet along the x-axis, and l is the length of the channel.

For the simulations in this section, the Fourier series is truncated at $n = 1$, and the following parameter values are used: $g_0 = 17$, $g_1 = 35 - 45i$, $p_0 = 0$, $\omega = 2\pi$, $h = 0.5$, $l = 1$, $\rho = 1$, $\mu = 0.05$. The velocity profile in the channel, normalized by the maximum velocity, is depicted in Fig. 2.2 at different

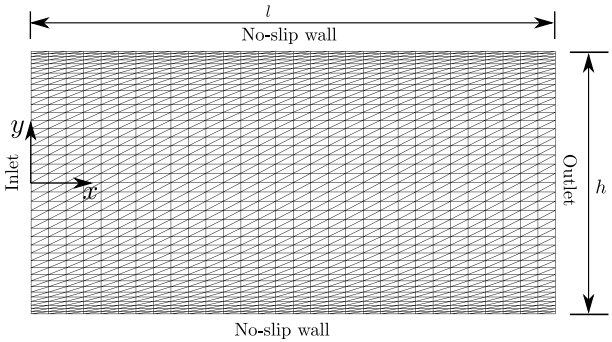


Figure 2.1: Pulsatile flow in channel: the computational domain and an example 30×25 mesh.

fractions of the period of oscillation. A similar pulsatile flow problem, but in a rigid pipe instead of a channel, is used in [16] using the Womersley solution [81].

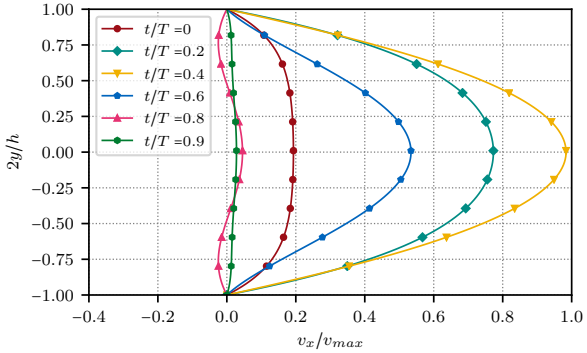


Figure 2.2: Pulsatile flow in channel: velocity profile along the width of the channel at different fractions of the period of oscillation.

In the numerical examples, no-slip boundary conditions are applied to the top and bottom walls of the channel. The velocity profile is applied at the inlet, and the *do-nothing* boundary condition is applied at the outlet, which sets the mean pressure across the outlet boundary to zero [82, 83]. For a modification of the do-nothing condition that allows the flow to leave the channel without disturbance when using the so-called *strain tensor formulation* of the viscous term used in this work, see, e.g., [76, p. 54]. The initial values are calculated using the exact values of velocity and pressure, according to Eqs. 2.168 and 2.169, respectively. The domain is discretized using triangular elements with

quadratic shape functions in order to avoid the lack of consistency in the numerical solution (due to the spatial discretization) commonly occurring when using stabilized formulations with linear shape functions. An example computational mesh with 30 elements along the length and 25 elements along the height of the channel is shown in Fig. 2.1 for better visibility. In the actual computations, a similar mesh with respectively 60 and 50 elements along the length and width of the channel is used.

We are interested in verifying the temporal order of accuracy of the EG-α scheme and comparing it with the three variants of the G-α scheme: the direct, shortened and p–$O(\Delta t)$ variants for the velocity, pressure, and their time derivatives. For problems with smooth enough solutions and by fully satisfying the continuity equation, the direct and the shortened forms of the G-α scheme are equivalent and are simply referred to as the G-α scheme. The p–$O(\Delta t)$ form of the G-α scheme differs from the direct and shortened forms in approximating the pressure (and its time derivative if required). In the following, the numerical rate of convergence of the relative error of velocity measured in H^1-norm and the pressure in L^2-norm, together with the relative errors of their time derivatives measured in L^2-norm are studied as the time step size is uniformly refined. For instance, the relative error of velocity is calculated as $\|\boldsymbol{v}^h - \boldsymbol{v}^e\|_{H^1} / \|\boldsymbol{v}^e\|_{H^1}$, where $\boldsymbol{v}^h$ and $\boldsymbol{v}^e$ are the numerical and analytical values of velocity, respectively. The errors are calculated at the end of one period of oscillation ($T = 1$). The absolute and relative tolerances of the nonlinear solver in each time step are set to 10^{-13} and 10^{-11}, respectively. For all the time integration schemes in this section, $\rho_\infty = 0.5$ is used.

The relative error of the velocity measured in H^1-norm is plotted in Fig. 2.3a. As expected, the calculated velocities using the EG-α scheme and all three variants of the G-α scheme are second-order accurate in time, and the convergence lines are overlapping.

The relative error of the pressure plotted in Fig. 2.3b shows some differences among the schemes. While the EG-α and the direct and shortened G-α schemes result in second-order accurate approximations for pressure, the p–$O(\Delta t)$ form of the G-α provides only first-order temporal accuracy for pressure. On the other hand, the direct and shortened G-α scheme require a consistent initial value for the pressure, while the p–$O(\Delta t)$ G-α and EG-α schemes do not require any initial value for the pressure.

The errors in the time derivatives of velocity and pressure are plotted in Fig. 2.4 for the EG-α scheme and the direct and shortened forms of the G-α scheme. The p–$O(\Delta t)$ G-α scheme is skipped in this comparison since it provides only first-order accurate pressure approximation and would naturally lead to a first-order accurate pressure time derivative. The plots reveal that while the calculated time derivatives of the velocity and pressure, $\dot{\boldsymbol{v}}^h$ and $\dot{p}^h$, using the EG-α are second-order accurate in time, the G-α scheme provides by default only first order accurate approximations for the time derivatives of the velocity and pressure, $\tilde{\boldsymbol{v}}^h$ and $\tilde{p}^h$. As verified by Fig. 2.4, second-order accurate time derivatives of the velocity and pressure can also be obtained

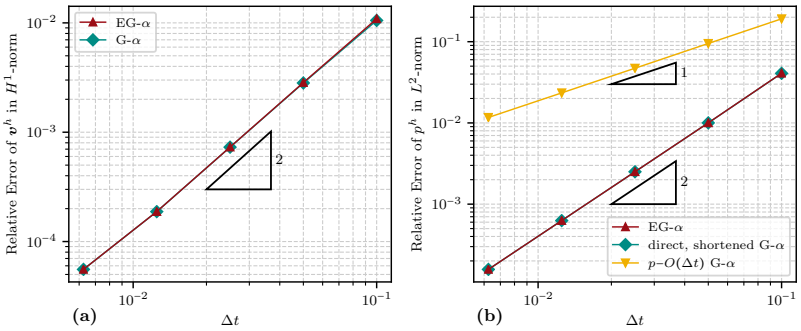


Figure 2.3: Pulsatile flow in channel: (a) Relative error of velocity $\boldsymbol{v}^h$; (b) Relative error of pressure p^h.

from the G-α scheme as well by performing a post-processing step at every time step, as shown in section 2.1.5.1.

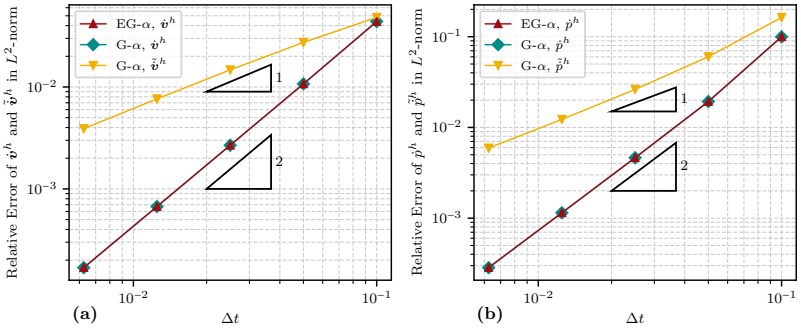


Figure 2.4: Pulsatile flow in channel: (a) Relative error of velocity time derivatives $\dot{\boldsymbol{v}}^h$ and $\tilde{\dot{\boldsymbol{v}}}^h$; (b) Relative error of pressure time derivatives $\dot{p}^h$ and $\tilde{\dot{p}}^h$.

2.1.7.2 Taylor-Green vortex problem in 2D

The example presented in section 2.1.7.1 was a uni-directional flow. In this section, the convergence of the time integration schemes in a more complex flow pattern is studied. The Taylor-Green solution, originally derived in [84], is used as a reference solution for benchmarking in CFD [85, 86, 87]. The Taylor-Green vortex problem admits a spatially periodic and time-dependent

solution for the velocity and pressure

$$\boldsymbol{v}_x = -\cos(\pi x)\sin(\pi y)\, e^{-2\pi^2 \nu t}, \tag{2.170}$$

$$\boldsymbol{v}_y = \sin(\pi x)\cos(\pi y)\, e^{-2\pi^2 \nu t}, \tag{2.171}$$

$$p = -\frac{\cos(2\pi x)\cos(2\pi y)}{4} e^{-4\pi^2 \nu t}, \tag{2.172}$$

where ν denotes the kinematic viscosity.

We consider the solution on a square domain with $0 \leqslant x, y \leqslant 1$. The computational mesh is a 150 by 150 structured grid in which each quadrilateral cell is divided into four triangular elements with quadratic shape functions. In the simulations, traction boundary conditions are applied to all boundaries. The tractions and initial conditions are calculated using the exact values for the velocity and pressure given in Eqs. 2.170–2.172. The initial pressure and velocity fields, together with the velocity streamlines, are depicted in Fig. 2.5.

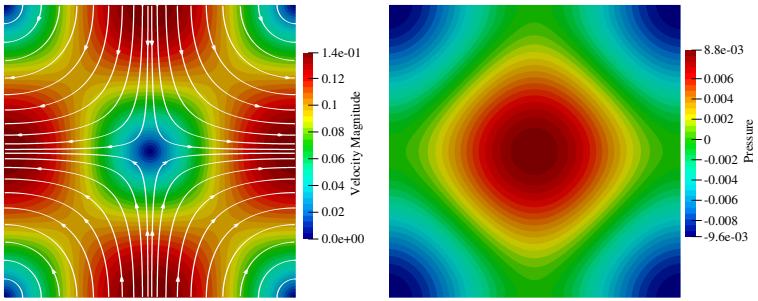


Figure 2.5: Taylor-Green vortex problem: (a) the velocity field together with the velocity streamlines; (b) the pressure field.

In the numerical examples in this section, The errors are calculated at $t = 1.0$. For all the time integration schemes in this section, ρ_∞ is set to 0.5. The absolute and relative tolerances of the nonlinear solver are set to 10^{-13} and 10^{-11}, respectively, and a direct solver is used to solve the resulting system of equations.

The relative errors of the velocity and pressure are plotted in Fig. 2.6, which shows that the velocities calculated using the EG-α scheme and all three variants of the G-α scheme are second-order accurate in time. In this example, the velocity using the G-α scheme has a slightly smaller error than the EG-α scheme, which points to a slightly smaller coefficient in the leading error term of velocity. On the contrary, the pressure calculated using the EG-α scheme shows a slightly smaller error than the pressure calculated using the direct

and shortened forms of the G-α scheme. All three of those schemes possess second-order temporal accuracy for pressure. Finally, the pressure calculated using the p–$O(\Delta t)$ form of the G-α scheme is only first-order accurate in time, similar to the pulsatile flow example.

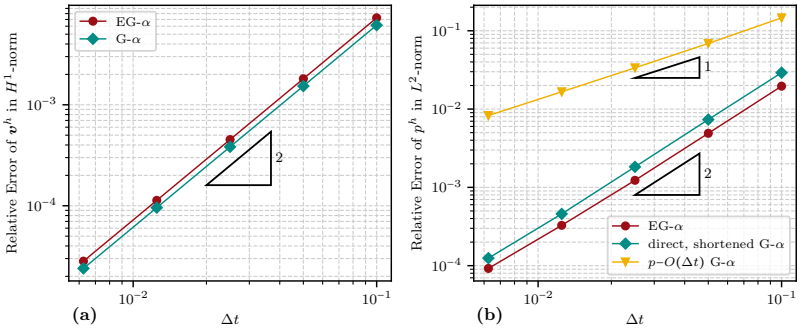


Figure 2.6: Taylor-Green vortex problem: (a) Relative error of velocity $\boldsymbol{v}^h$; (b) Relative error of pressure p^h.

The errors in the time derivatives of velocity and pressure are plotted in Fig. 2.7 for the EG-α scheme and the direct and shortened forms of the G-α scheme. The time derivatives of the velocity $\dot{\boldsymbol{v}}^h$ and pressure $\dot{p}^h$ calculated using the EG-α are second-order accurate in time. Without doing post-processing on the time derivatives, the G-α scheme provides only first-order accurate approximations for the time derivatives of the velocity and pressure, $\tilde{\dot{\boldsymbol{v}}}^h$ and $\tilde{\dot{p}}^h$. As in the previous example, second-order accurate time derivatives of the velocity and pressure are recovered for the G-α scheme by performing the post-processing step at every time step.

The results of this section are consistent with those of section 2.1.7.1 and confirm the second order temporal accuracy of the EG-α scheme for the velocity, pressure, and their time derivatives. Furthermore, the results suggest that the direct and shortened forms of the G-α scheme should be preferred over its p–$O(\Delta t)$ form due to possessing higher temporal accuracy for the pressure.

2.1.7.3 Flow around cylinder

The numerical examples in sections 2.1.7.1 and 2.1.7.2 focused on verifying the temporal order of accuracy of the schemes by comparison to analytical solutions. In this section, in addition to the general accuracy of the schemes, the stability of the time integration schemes is studied using more complex flow problems.

The well-known example of flow around a cylinder in a channel is studied in the present section. The flow develops into periodic vortex shedding

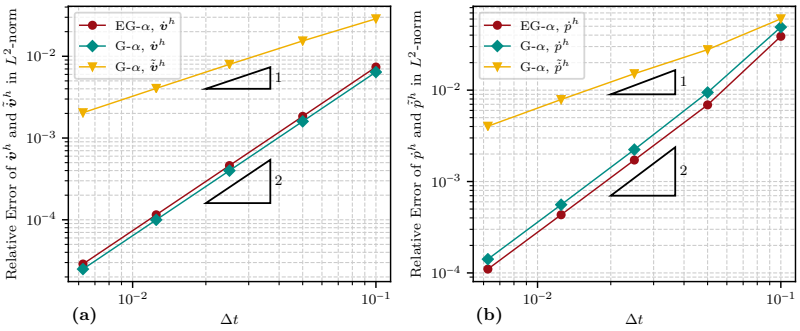


Figure 2.7: Taylor-Green vortex problem: (a) Relative error of velocity time derivatives $\dot{\boldsymbol{v}}^h$ and $\tilde{\dot{\boldsymbol{v}}}^h$; (b) Relative error of pressure time derivatives $\dot{p}^h$ and $\tilde{\dot{p}}^h$.

downstream of the cylinder, known as the von Kármán vortex street. Due to the vortex shedding, oscillating drag and lift forces are exerted on the cylinder. The specific benchmark problem solved in this section is taken from [88], where it is denoted by "test case 2D-2". The geometry of the problem solved here follows the definitions provided in detail in [88]. The inflow velocity is prescribed by

$$\boldsymbol{v}_x(y) = 4U_m \frac{y\,(h-y)}{h^2}, \quad \boldsymbol{v}_y = 0, \tag{2.173}$$

with $U_m = 1.5$ m/s. The fluid has a density of 1.0 kg/m^3 and kinematic viscosity of 10^{-3} m^2/s, leading to a Reynolds number $Re = \bar{U}D/\nu = 100$, where $\bar{U} = 2U_m/3$ is the mean velocity across the inlet. It has been suggested to run the simulation with a coarse time step for about 3.5 seconds in order to obtain the initial conditions for the actual simulation with a fine time step. Here, a different initialization procedure similar to the approach in [89] is used. The simulation is started with zero initial conditions and the inflow velocity is ramped up to the nominal value in the first 3 seconds. Therefore, the time-dependent inlet velocity takes the form

$$\boldsymbol{v}_x\,(y,t) = \begin{cases} \boldsymbol{v}_x\,(y)\,\dfrac{1-\cos\left(\frac{\pi t}{3}\right)}{2} & \text{if } t < 3 \\ \boldsymbol{v}_x\,(y) & \text{otherwise} \end{cases}, \tag{2.174}$$

with $\boldsymbol{v}_x\,(y)$ defined by Eq. 2.173.

In order to compare the stability of different variants of the schemes, we deliberately do not use a very fine mesh to more closely replicate the challenging situation in which high-frequency oscillations occur while still simulating the correct behavior of the flow. Two different meshes with medium and coarse resolutions are used here. In addition, elements with linear shape functions are used in all the simulations in this section. The coarse mesh,

depicted in Fig. 2.8, has 8632 triangular elements and 4327 nodes. The medium mesh follows a similar distribution of the nodes as in the coarse mesh but is finer and has 33162 elements and 16874 nodes.

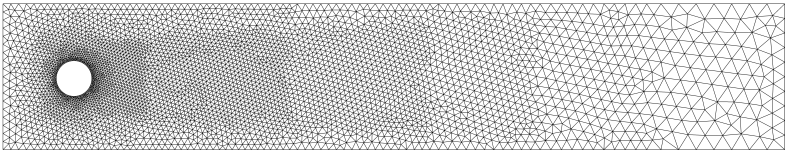


Figure 2.8: Flow around cylinder: The coarse mesh.

The simulations are carried out for 8 seconds with a time step size of 0.1 seconds and a relative tolerance of the nonlinear solver in every time step equal to 10^{-8}. Fig. 2.9 shows the velocity field and streamlines, together with the pressure contours at the end of a simulation on the coarse mesh and using the EG-α scheme.

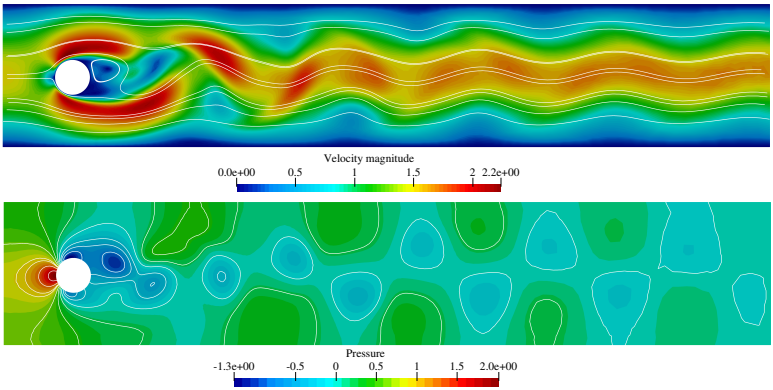


Figure 2.9: Flow around cylinder: (a) the velocity field and the streamlines; (b) the pressure contours.

Two values of interest in the flow around the cylinder example are the drag and lift coefficients, C_D and C_L, which are defined respectively as

$$C_D = \frac{2F_D}{\rho \bar{U}^2 D}, \quad C_L = \frac{2F_L}{\rho \bar{U}^2 D}, \tag{2.175}$$

where F_D and F_l are the drag and lift forces on the cylinder, calculated in the present work using the variationally consistent approach (see section 2.1.4). The evolution of the drag coefficient on the cylinder over time is plotted in Fig. 2.10 for the coarse and the medium meshes. The drag coefficient

undergoes harmonic oscillations as expected with the values of C_D lying in the ranges [3.18, 3.124] and [3.17, 3.123] using the coarse and medium meshes, respectively, which are satisfactory when compared to the results presented in [88] where the lower and upper bounds of the maximum value of C_D are 3.22 and 3.4 respectively.

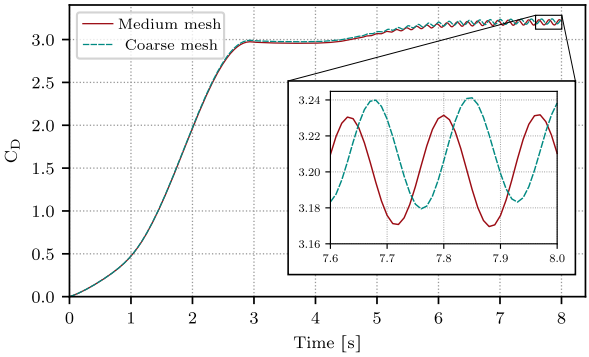


Figure 2.10: Flow around cylinder: Evolution of the drag coefficient C_D.

The evolution of the lift coefficient on the cylinder is plotted in Fig. 2.11 for the coarse and the medium meshes. The magnitude of the lift coefficient oscillations has almost converged to 1.0 at $t = 8$, and a longer simulation is generally required to reach a fully developed results. However, the results after 8 seconds are enough for the comparisons carried out in this section. For comparison, the lower and upper bounds of the results reported in [88] for the maximum value of C_L are 0.99 and 1.01 respectively.

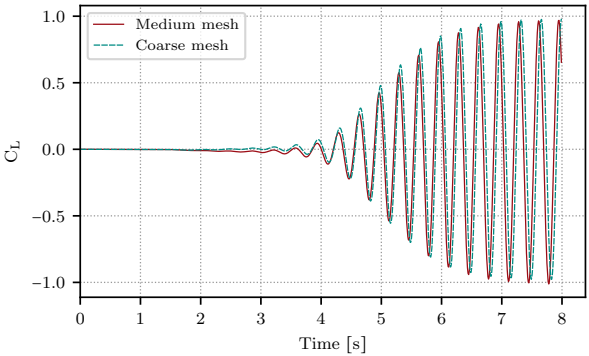


Figure 2.11: Flow around cylinder: Evolution of the lift coefficient C_L.

We are interested in comparing the stability and accuracy of the EG-α scheme to the direct and shortened variants of the G-α scheme. In order to do so, we compare the values of the drag and lift coefficients as well as the pressure value in front of the cylinder and the flow velocity in the y-direction at a point located 0.15 m downstream of the cylinder for different schemes and different values of ρ_∞. The results for $\rho_\infty = 0.5$ are plotted in Fig. 2.12 in a time interval right at the end of the simulation. The plots reveal that the results using the EG-α, direct G-α and shortened G-α schemes perform very similarly. Any possible high-frequency oscillation is damped out because of the introduced high-frequency dissipation by setting $\rho_\infty = 0.5$. There is a slight difference in the pressure values obtained by the shortened G-α scheme compared to the other two schemes. This difference is attributed to the fact that the shortened G-α scheme actually solves an ever slightly different problem (due to the shortening of the time-discrete continuity equation).

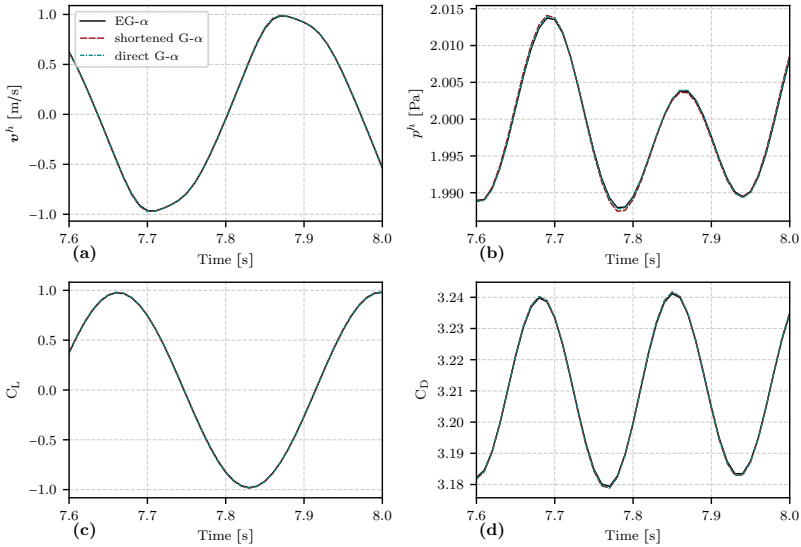


Figure 2.12: Flow around cylinder: comparison of the EG-α, direct G-α and shortened G-α schemes with $\rho_\infty = 0.5$. (a) velocity at 0.15 m downstream of cylinder; (b) pressure in front of cylinder; (c) lift coefficient; (d) drag coefficient.

While the choice of $\rho_\infty = 0.5$ is commonly recommended for the G-α scheme, a higher value of ρ_∞, i.e., less numerical dissipation, is enough for many problems and leads to more accurate results. The schemes are also compared for $\rho_\infty = 0.95$, leading to a minimal amount of high-frequency dissipation. The results, plotted in Fig. 2.12, reveal that while the EG-α and

the shortened G-α schemes lead to stable results for all the plotted values, the pressure and drag coefficient calculated using the direct G-α scheme develop high-frequency oscillations which grow in time. This is due to the fact that very small errors in the continuity equation are propagated and accumulate over time when using the direct form of the G-α scheme. The shortened G-α scheme avoids the propagation of such errors by shortening the continuity equation and solving the modified problem. The EG-α solves the equilibrium equation, including the continuity equation, at every discrete time point separately and, therefore, does not suffer from the propagation of the errors in the continuity equation.

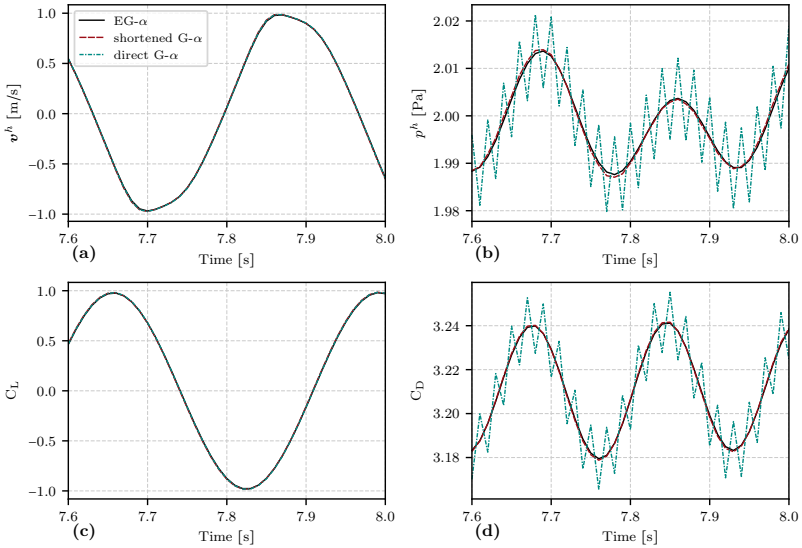


Figure 2.13: Flow around cylinder: comparison of the EG-α, direct G-α and shortened G-α schemes with $\rho_\infty = 0.95$. (a) velocity at 0.15 m downstream of cylinder; (b) pressure in front of cylinder; (c) lift coefficient; (d) drag coefficient.

2.1.7.4 Oscillating driven cavity flow

In this section, another example of unsteady flows is studied in order to verify the findings in section 2.1.7.3. The driven cavity is a well-known benchmark in the CFD community. However, it is usually used for verifying steady-state solutions. An unsteady oscillating version of the benchmark is proposed in [90]. The computational domain is a unit square. No-slip boundary conditions are applied on the left, bottom, and right sides of the square. On the top

boundary, an oscillating velocity along the x-axis is applied:

$$\boldsymbol{v}_x = U \sin(\omega t), \tag{2.176}$$

with $U = 1$ m/s and $\omega = \pi$. The fluid has a density of 1 kg/m^3 and kinematic viscosity of 0.002 m^2/s, leading to a Reynolds number $Re = 500$. Since the pressure is defined up to a constant in this enclosed domain, the pressure value at a point located in the middle of the bottom boundary is set to zero. Two triangular meshes are considered for the simulations, a coarse and a fine mesh with 60 and 140 divisions along each side, respectively. The meshes are refined towards the boundaries. For the sake of clarity, only the bottom left quarter of the coarse mesh is shown in Fig. 2.14.

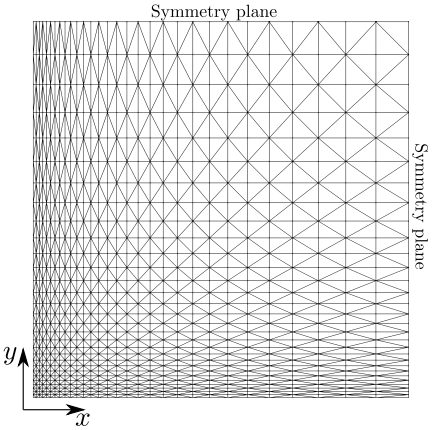


Figure 2.14: Oscillating driven cavity flow: The bottom left quarter of the coarse mesh with 60 divisions along each side.

The simulations in this section are carried out for 10 seconds with a time step size of 0.02 seconds, triangular elements with linear shape functions, and a relative tolerance of the nonlinear solver in every time step equal to 10^{-9}. Fig. 2.15 shows the velocity field and streamlines, together with the pressure contours at the end of a simulation on the coarse mesh and using the EG-α scheme.

A value of interest in the current example is the integrated skin friction coefficient (or simply friction coefficient) on the top boundary, defined as

$$C_F = \frac{2F_F}{\rho U^2}, \tag{2.177}$$

where F_F is the integrated shear force on the top boundary and U is the characteristic velocity. In [90], a slightly different definition of the friction

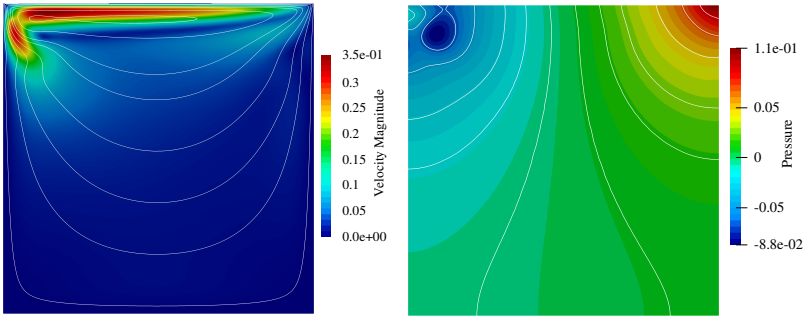


Figure 2.15: Oscillating driven cavity flow: (a) the velocity field and the streamlines; (b) the pressure contours at $t = 10\ s$.

coefficient is used, which would be equal to $Re \cdot C_F/2$, with Re being the Reynolds number and C_F defined by Eq. 2.177. The friction coefficients calculated using the EG-α scheme on the fine and coarse meshes are plotted in Fig. 2.16. For comparison, Fig. 2.16 plots also the reference values from [90] obtained using a structured 121×121 grid and with a time step size of 2×10^{-4}, and rescaled to follow the friction coefficient definition in Eq. 2.177. For the numerical stability studies in this section, the coarse mesh is deliberately used.

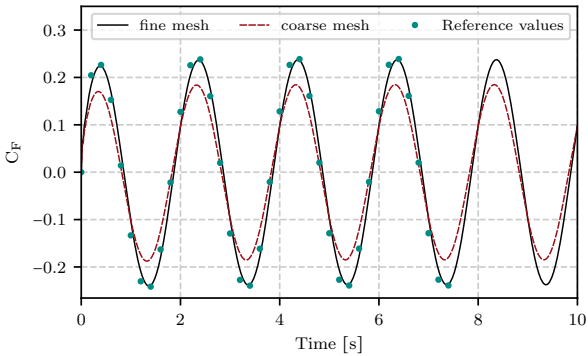


Figure 2.16: Oscillating driven cavity flow: The evolution of the friction coefficient using the fine and coarse meshes and comparison with the reference values from [90]. The reference values are resealed here to match the present definition of the friction coefficient.

Similarly to the friction coefficient, Eq. 2.177, we define a force coefficient

for the normal force exerted on the top boundary by the flow:

$$C_N = \frac{2F_N}{\rho U^2}, \tag{2.178}$$

where F_N is the integrated normal force applied on the top boundary by the flow. Note that in the current example, C_F is independent of the pressure value, but C_N depends on the pressure. In addition to C_F and C_N, we monitor the pressure value at a point located in the middle of the top boundary, with the coordinates [0.5, 1], and the velocity value at a point with the coordinates [0.5, 0.75] (the left bottom corner of the domain has the coordinates [0,0]).

The results from the last two seconds of the simulations using the EG-α, direct G-α, and shortened G-α schemes with $\rho_\infty = 0.5$ are shown in Fig. 2.17. The Results in Fig. 2.17 indicate that the three schemes perform very similarly in this example, and are practically indistinguishable when enough numerical high-frequency damping is used.

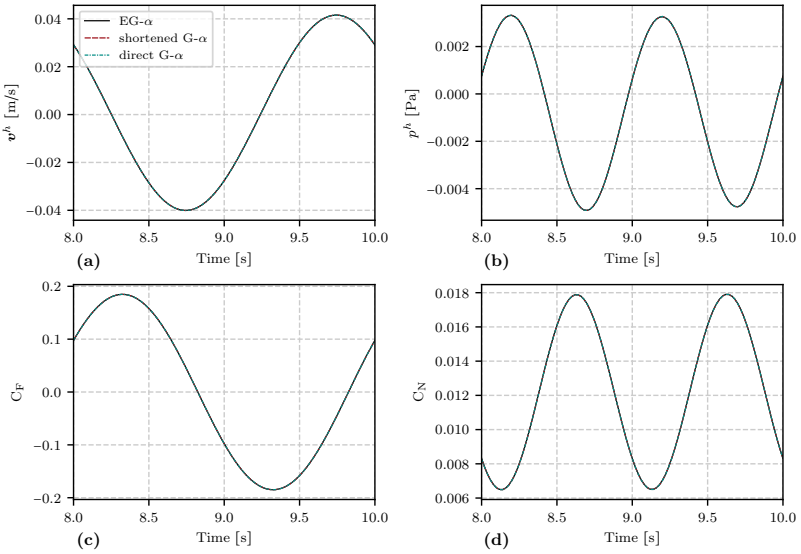


Figure 2.17: Oscillating driven cavity flow: comparison of the EG-α, direct G-α and shortened G-α schemes with $\rho_\infty = 0.5$. (a) velocity along the x-axis at the point [0.5, 0,75]; (b) pressure at the point [0.5, 0,75]; (c) friction coefficient; (d) normal force coefficient.

In order to compare the three schemes with lower amount of algorithmic high-frequency dissipation, the simulations are repeated with $\rho_\infty = 0.95$. The results from the last two seconds of the simulations are shown in Fig. 2.18. The

pressure value calculated using the direct G-α scheme develops high-frequency oscillations which grow in time. Consequently, the normal force coefficient C_N starts to oscillate using the direct G-α scheme. The velocity and the friction coefficient C_F calculated using the direct G-α scheme are oscillation free due to the independence of C_F from the pressure values. In contrast to the direct G-α scheme, the results using the EG-α and the shortened G-α are stable in the current example, even when using very small amounts of high-frequency dissipation.

The results of this section are in agreement with the observations made in section 2.1.7.3 and indicate that the EG-α and the shortened G-α schemes are generally more stable than the direct G-α scheme. While the direct G-α scheme performs as well as the other two schemes for higher amounts of high-frequency dissipation, it is prone to developing pressure oscillations when using lower amounts of high-frequency dissipation. The situation may get worse for the direct G-α scheme when solving more complex flow problems with rough initial and boundary conditions.

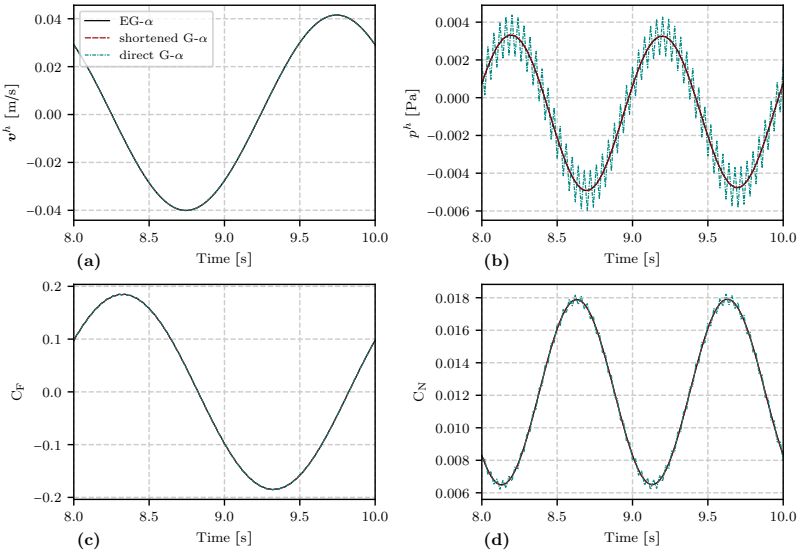


Figure 2.18: Oscillating driven cavity flow: comparison of the EG-α, direct G-α and shortened G-α schemes with $\rho_\infty = 0.95$. (a) velocity along the x-axis at the point [0.5, 0,75]; (b) pressure at the point [0.5, 0,75]; (c) friction coefficient; (d) normal force coefficient.

2.1.7.5 Poiseuille flow on moving domain

After studying the accuracy and stability of the EG-α and variants of G-α schemes on fixed grids, the accuracy of different schemes is compared on moving grids in the current and the following section. Particular focus is placed on the correct instance of the integration domain when solving the equations in the current configuration (equivalently, the correct time instant at which the domain's deformation gradient $\mathbf{F}$ and its determinant J are evaluated when solving the equations in the reference configuration).

The current example is taken from [63] and consists of the Poiseuille flow in a channel with a moving grid. The velocity field in the Poiseuille flow is steady-state and has a quadratic profile along the width of the channel. The steady-state pressure field changes linearly in space along the length of the channel. The domain is discretized using a structured quadrilateral 10×6 mesh with quadratic shape functions in each element. The computational domain, together with the initial mesh, is depicted in Fig. 2.19. The physical

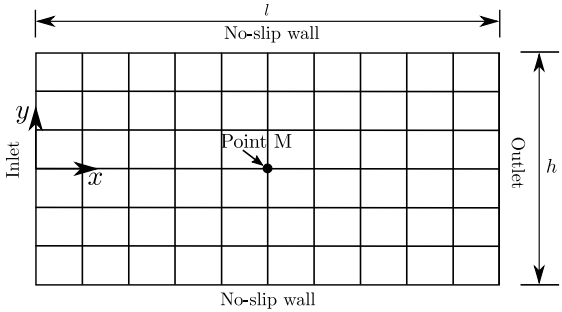


Figure 2.19: Poiseuille flow on moving domain: The computational domain and an the 10×6 initial mesh.

parameters of the problem are $h = 1$, $l = 2$, $\rho = 1$, and $\nu = 0.2$. A parabolic velocity profile with the maximum value of 0.5 is applied at the Inlet, and do-nothing boundary condition is applied at the outlet. The computational grid undergoes a deformation such that the vertical middle line of the mesh is moved 0.8 units to the right at one extreme and 0.8 units to the left at the other one, according to the horizontal elongation function $x = 0.8\sin(\pi t/2)$. The deformed mesh at the two extremes of the deformation is depicted in Fig. 2.20. The simulations are carried out using a time step size of 0.05.

We intend to compare the accuracy of the EG-α scheme and the variants of the G-α scheme listed in Table 2.1. The time evolution of the pressure value at point M in the middle of the domain (in the current configuration) is monitored using different schemes. The schemes are expected to preserve the constant value for pressure at point M. Note that since the velocity field is steady-state and the same everywhere along the x-axis, no difference in

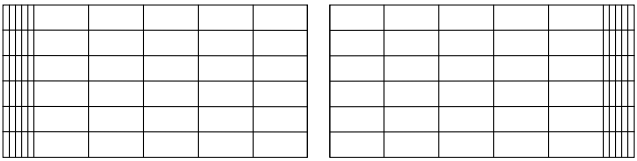


Figure 2.20: Poiseuille flow on moving domain: The computational domain at the (a) left extreme of the deformation; (b) right extreme of the deformation.

the velocity values among the different variants of the schemes is revealed in this example. The evolution of the pressure value at point M using different schemes is plotted in Fig. 2.21.

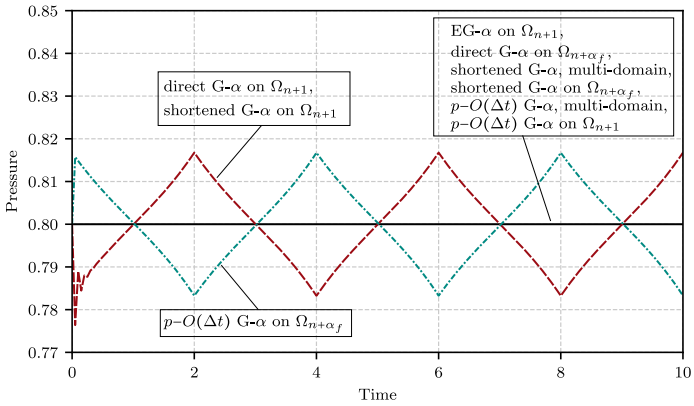


Figure 2.21: Poiseuille flow on moving domain: The evolution of the pressure value in the middle of the domain using different schemes. Refer to Table 2.1 for the details of the schemes.

The EG-α scheme used on domain Ω_{n+1} yields correct results for the pressure. This is the advantage of the EG-α scheme for problems on moving domains and is the result of satisfying the equilibrium equation at t_{n+1}. Among the variants of the G-α scheme, two groups of variants lead to the correct results for the pressure. The first group consists of the direct G-α evaluated on $\Omega_{n+\alpha_f}$, shortened G-α evaluated on multiple instances of Ω, and shortened G-α evaluated on $\Omega_{n+\alpha_f}$ (see Table 2.1 for the naming convention). The first group of G-α variants is characterized by the evaluation of the pressure term at $t_{n+\alpha_f}$ and on the intermediate domain $\Omega_{n+\alpha_f}$. The second group of the G-α variants, leading to the correct results for pressure, comprises the p–$O(\Delta t)$ G-α evaluated on Ω_{n+1} and p–$O(\Delta t)$ G-α evaluated on multiple instances of Ω. This second group of the schemes is characterized by the evaluation of the

pressure term at t_{n+1} and on the domain Ω_{n+1}. There are also variants of the G-α that do not yield correct results for the pressure, namely: the direct and shortened G-α evaluated on Ω_{n+1} and the p–$O(\Delta t)$ G-α evaluated on $\Omega_{n+\alpha_f}$. The last group of variants has a mismatch between the time instant at which the pressure term and its integral are evaluated.

The conclusion drawn from the present example, which is strictly valid only for the pressure approximation, is that the EG-α scheme yields the expected results for the pressure. For the G-α family of schemes, the integrals should be evaluated on a domain consistent with the time instant at which the pressure term is evaluated. Of course, from an accuracy point of view, using a variant of the G-α with a second-order approximation for pressure is more desirable. As mentioned previously, Due to a combination of the constant velocity field along the x-axis and the grid's motion being also uni-directional in the x-direction, the current example is incapable of revealing any difference among different variants of the G-α scheme in the approximation of velocity. Therefore, a more complex flow on a moving domain will be studied in the following section in order to gain a full understating and comparison of the different schemes.

2.1.7.6 Taylor-Green vortex problem in ALE coordinates

In this section, the Taylor-Green flow example, presented in section 2.1.7.2, is revisited by solving the problem on a moving domain. The velocity and the pressure fields are not constant in time and space and are, therefore, suitable for comparing different time integration schemes on moving domains. The same physical parameters as in section 2.1.7.2 are used here. However, the domain moves along the x and y axes according to the following formulae:

$$x = (1 - \cos(\omega t))\, xy\, (x - h)\, (y - h)\,, \tag{2.179}$$

$$y = (1 - \cos(\omega t))\, xy\, (x - h)\, (y - h)\,, \tag{2.180}$$

where $\omega = 2\pi$ is the frequency of the oscillatory motion and $h = 1$ is the length of each side of the domain. The nodes on the boundary do not move, and the grid's internal nodes move along one of the diagonals of the square domain. The deformed mesh at the extreme of oscillatory motion is depicted in Fig. 2.22 for a mesh with 25 divisions along each edge of the domain (for better clarity). In the actual simulations, a mesh with 50 divisions along each domain edge is used. The simulations are carried out for 2 seconds with a time step size of 0.025.

First, the temporal order of accuracy of the calculated mesh velocity using the G-α family of schemes is compared. It was pointed out in section 2.1.6.1 that the mesh velocity calculated using Eq. 2.149 is generally first-order accurate. In contrast, the true mesh velocity calculated using Eq. 2.150 is second-order accurate in time. The errors in the calculated mesh velocities $\tilde{\dot{\mathrm{d}}}^m$ and $\dot{\mathrm{d}}^m$ using Eqs. 2.149 and 2.150 are measured at $t = 0.9$ and are plotted

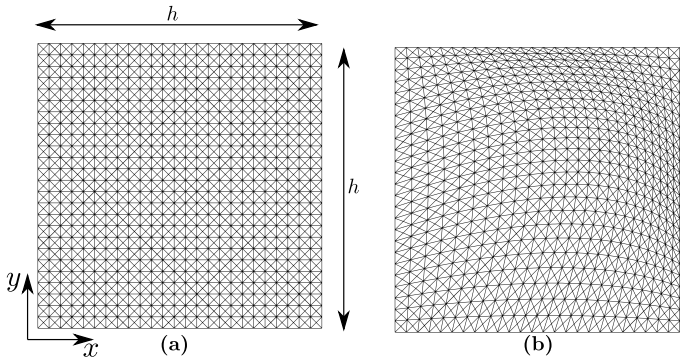


Figure 2.22: Taylor-Green vortex problem on a moving domain: (a) the initial mesh and the computational domain at one extreme of the oscillatory motion; (b) the computational domain at the other extreme of the oscillatory motion for a mesh with 25 elements along each edge.

against a uniform refinement of the time step size in Fig. 2.23. It is verified that the calculated mesh velocity using Eqs. 2.149 and 2.150 are indeed first and second-order accurate in time, respectively.

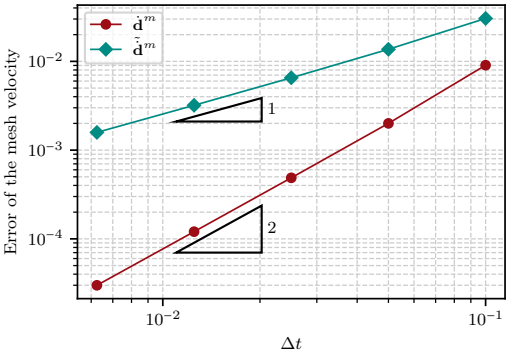


Figure 2.23: Taylor-Green vortex problem on a moving domain: The global error of the calculated mesh velocities $\tilde{\dot{\mathrm{d}}}^m$ and $\dot{\mathrm{d}}^m$ using, respectively, Eqs. 2.149 and 2.150.

Next, the accuracy of the calculated pressure and velocity along the x-axis at point P, located in the middle of the top boundary, using different variants of the G-α scheme listed in Table 2.1 are compared. The analytical solution of both the pressure and the velocity in x-direction at point P is zero throughout time (see Fig. 2.5).

The evolution of the pressure and velocity at point P approximated using

the direct G-α variants are plotted in Fig.2.24. The results reveal that using the direct G-α and evaluating the terms on $\Omega_{n+\alpha_f}$ yields values very close to the analytical solution. Those values would converge to the analytical solution using finer time steps and spatial discretizations. In contrast, using the direct G-α and conveniently evaluating the terms on Ω_{n+1} yields results which deviate from the expected values.

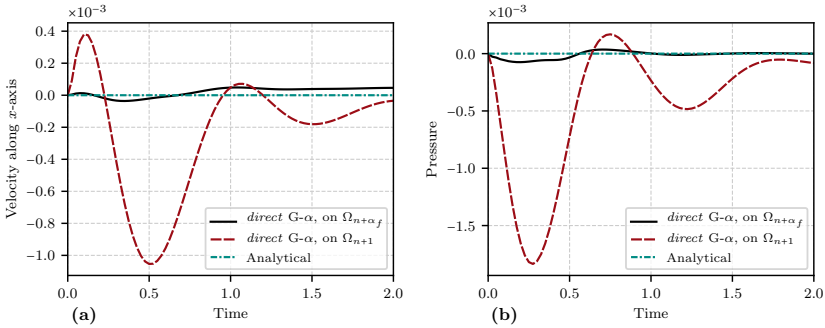


Figure 2.24: Taylor-Green vortex problem on a moving domain: comparison of the direct G-α variants in the approximation of (a) the velocity along the x-axis; (b) the pressure at point P.

The results using the shortened G-α variants are presented in Fig. 2.25 and indicate that the only variant of the shortened G-α scheme capable of approximating the expected analytical solutions is the one with evaluations of the integrals on multiple instances of the domain. This is a drawback of the shortened G-α scheme for flows on moving domains since the terms containing the continuity equation must be evaluated on Ω_{n+1}, while all the other terms are evaluated on $\Omega_{n+\alpha_f}$. This drawback is alleviated when solving the equations in the reference configuration. Note in Fig. 2.25 that using the shortened G-α and evaluating all the integrals on $\Omega_{n+\alpha_f}$ does not yield the correct results, neither for the pressure nor for the velocity.

The results using the p–$O(\Delta t)$ G-α variants are presented in Fig. 2.26. Using the p–$O(\Delta t)$ G-α scheme and evaluating the integrals on multiple instances of the domain, according to Table 2.1, generally leads to accepted results. However, the pressure and velocity approximations are not as accurate as the direct and shortened variants with the correct setups. The combination of the p–$O(\Delta t)$ G-α scheme and evaluation of integrals on multiple instances is also tried together with the inf-sup stable P2P1 elements, which lead to similar results.

The EG-α scheme, used on the domain Ω_{n+1}, is compared in Fig. 2.27 to the most accurate variants of the G-α family of schemes, that is, the direct G-α used on the domain $\Omega_{n+\alpha_f}$ and shortened G-α with the evaluation of integrals on multiple instances of the domain. The EG-α alpha and direct

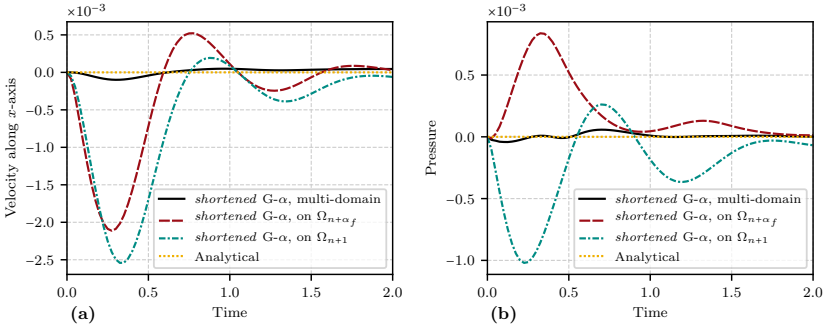


Figure 2.25: Taylor-Green vortex problem on a moving domain: comparison of the shortened G-α variants in the approximation of (a) the velocity along the x-axis; (b) the pressure at point P.

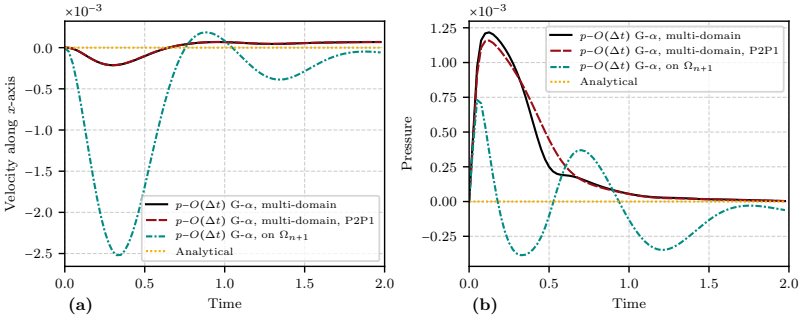


Figure 2.26: Taylor-Green vortex problem on a moving domain: comparison of the p–$O(\Delta t)$ G-α variants in the approximation of (a) the velocity along the x-axis; (b) the pressure at point P.

G-α schemes behave very similarly, with the EG-α scheme being slightly more accurate in approximating the pressure. The shortened G-α scheme performs slightly worse than the two other schemes in the approximation of velocity.

After the quantitative comparison of the schemes using the results at point P, the global solution of the pressure field using different variants of the schemes is plotted and compared in Fig. 2.28. The sub-figures on the top row of Fig. 2.28 show results using the EG-α, direct and shortened G-α schemes using their corresponding correct integration domains. The sub-figures on the bottom row of the figure show examples of distorted pressure fields approximated using the wrong integration domains. A similar comparison for the velocity field is made using Fig. 2.29, showing examples of the distorted velocity fields in the sub-figures in the bottom row.

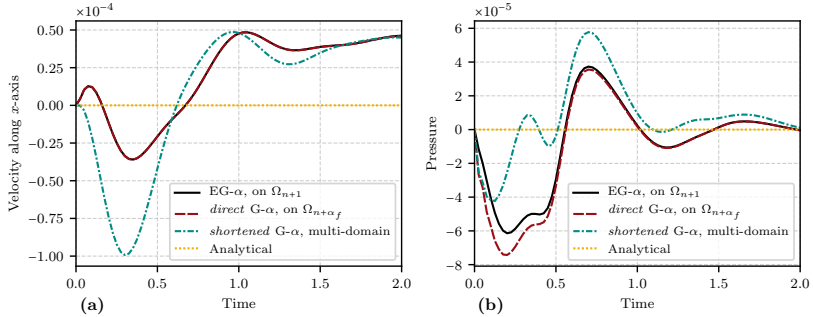


Figure 2.27: Taylor-Green vortex problem on a moving domain: comparison of the EG-α scheme with the best performing variants of G-α in the approximation of (a) the velocity along the x-axis; (b) the pressure at point P.

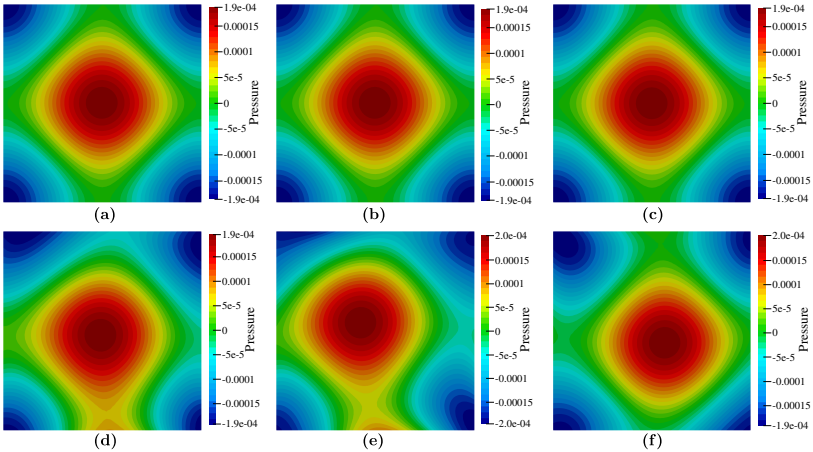


Figure 2.28: Taylor-Green vortex problem in ALE framework: pressure field at time $t = 2.0$ using (a) EG-α on Ω_{n+1}; (b) direct G-α on $\Omega_{n+\alpha_f}$; (c) shortened G-α, multi-domain; (d) p–$O(\Delta t)$ G-α on Ω_{n+1}; (e) direct G-α on Ω_{n+1}; (f) shortened G-α on $\Omega_{n+\alpha_f}$

2.2 Time discretization of the solid problem

2.2.1 The solid problem

The solid problem is posed in the Lagrangian reference frame. The governing equation of the solid is determined by the momentum equation

$$\rho \partial_{tt} \boldsymbol{d} - \nabla_X \cdot (\mathbf{F}\boldsymbol{\Sigma}) = \rho \boldsymbol{f} \quad \text{in } \Omega_0, \ t \in (0, T], \tag{2.181}$$

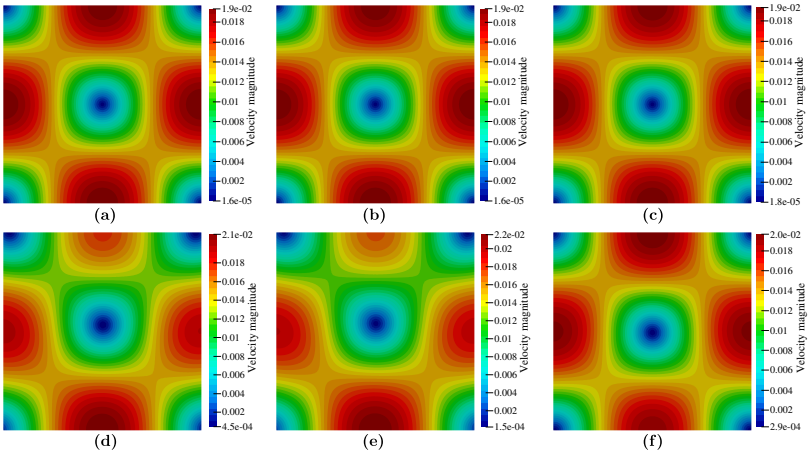


Figure 2.29: Taylor-Green vortex problem in ALE framework: velocity field at time $t = 2.0$ using (a) EG-α on Ω_{n+1}; (b) direct G-α on $\Omega_{n+\alpha_f}$; (c) shortened G-α, multi-domain; (d) p–$O(\Delta t)$ G-α on Ω_{n+1}; (e) direct G-α on Ω_{n+1}; (f) shortened G-α on $\Omega_{n+\alpha_f}$

where ρ is the solid density in the reference frame, ∂_{tt} stands for the second derivative with respect to time, $\mathbf{F}$ is the deformation gradient of the solid deformation mapping, Σ is the second Piola–Kirchhoff stress, and $\boldsymbol{f}$ is the body force per unit mass. The deformation gradient $\mathbf{F}$ is defined as

$$\mathbf{F} = \frac{\partial \mathcal{A}_t(\boldsymbol{X})}{\partial \boldsymbol{X}} = \frac{\partial \boldsymbol{x}(\boldsymbol{X}, t)}{\partial \boldsymbol{X}} = \nabla_X \boldsymbol{x}, \tag{2.182}$$

where $\boldsymbol{X}$ and $\boldsymbol{x}$ stand for the material and spatial coordinates, respectively. For St. Venant Kirchhoff material, the second Piola–Kirchhoff stress is defined as

$$\boldsymbol{\Sigma} = 2\mu \boldsymbol{E} + \lambda \operatorname{tr}(\boldsymbol{E})\, \boldsymbol{I}, \tag{2.183}$$

where μ and λ are the Lamé parameters, and $\boldsymbol{E}$ is the Green–Lagrange strain tensor defied as

$$\boldsymbol{E} = \frac{1}{2}\left((\mathbf{F})^T \mathbf{F} - \boldsymbol{I}\right). \tag{2.184}$$

The solid problem in Eq. 2.181 is subjected to the following initial and boundary conditions

$$\begin{aligned} \boldsymbol{d} &= \boldsymbol{d}_0 && \text{in } \Omega_0,\ t = 0, \\ \partial_t \boldsymbol{d} &= \boldsymbol{v}_0 && \text{in } \Omega_0,\ t = 0, \\ \boldsymbol{d} &= \boldsymbol{d}^{s,D} && \text{on } \Gamma_0^{s,D},\ t \in (0, T], \\ \mathbf{F}\boldsymbol{\Sigma} \cdot \boldsymbol{n}_0 &= \boldsymbol{t}^{s,N} && \text{on } \Gamma_0^{s,N},\ t \in (0, T], \end{aligned} \tag{2.185}$$

where $\boldsymbol{d}_0$ and $\boldsymbol{v}_0$ are the initial values for displacement and velocity respectively, $\Gamma_0^{s,D}$ and $\Gamma_0^{f,N}$ represent the Dirichlet and Neumann portions of the solid boundary Γ, $\boldsymbol{n}_0$ is the unit normal vector of the solid domain's boundary in the reference configuration, and $\boldsymbol{t}^{s,N}$ is the normal traction prescribed on the Neumann section of the solid domain's boundary. Let $\boldsymbol{\mathcal{V}} \coloneqq \boldsymbol{H}_D^1(\Omega_0)$ and $\boldsymbol{\mathcal{W}} \coloneqq \boldsymbol{H}_0^1(\Omega_0)$ be the spaces of trial solution and test functions, respectively, and $\boldsymbol{\mathcal{V}}_h$ and $\boldsymbol{\mathcal{W}}_h$ be their corresponding finite-dimensional spaces. After spatial discretization using Galerkin method, the semi-discrete form of the solid problem reads: Find $\boldsymbol{d} \in \boldsymbol{\mathcal{V}}_h$ such that $\forall \boldsymbol{w} \in \boldsymbol{\mathcal{W}}_h$

$$B(\boldsymbol{w}, \boldsymbol{d}) = F(\boldsymbol{w}), \tag{2.186}$$

where

$$B(\boldsymbol{w}, \boldsymbol{d}) = (\boldsymbol{w}, \rho \partial_{tt} \boldsymbol{d})_{\Omega_0} + (\nabla_X \boldsymbol{w}, \mathbf{F}\boldsymbol{\Sigma})_{\Omega_0}, \tag{2.187}$$

$$F(\boldsymbol{w}) = (\boldsymbol{w}, \rho \boldsymbol{f})_{\Omega_0} + \left(\boldsymbol{w}, \boldsymbol{t}^{s,N}\right)_{\Gamma_0^{f,N}}. \tag{2.188}$$

The solid problem in Eq. 2.181 is a second-order system in time. It is possible to write the system in Eq. 2.181 as a system of two coupled first-order equations in time. By introducing $\boldsymbol{v}$ as an independent variable, the solid problem can be restated as

$$\rho \partial_t \boldsymbol{v} - \nabla_X \cdot (\mathbf{F}\boldsymbol{\Sigma}) = \rho \boldsymbol{f} \quad \text{in } \Omega_0,\ t \in (0, T], \tag{2.189}$$

$$\partial_t \boldsymbol{d} = \boldsymbol{v} \quad \text{in } \Omega_0,\ t \in (0, T], \tag{2.190}$$

where the second equation is a kinematic constraint between the displacement and velocity variables. Recasting the solid problem, as in Eqs. 2.189-2.190, has been attractive in the literature in the context of FSI simulations [50, 203, 204, 79]. This approach has the advantage that one can treat the solid and fluid problems uniformly. On the other hand, the number of degrees of freedoms (DoFs) in the system represented by Eqs. 2.189-2.190, and subsequently the size of the system of linear equations arising from the spatial discretization, is doubled compared to the system in Eq. 2.181 since there are now two unknown variables instead of one. However, as was shown in [205], it is possible to reduce the unknown variables to only $\boldsymbol{d}$ (or equally $\boldsymbol{v}$) by taking advantage of Eq. 2.190 and algebraic manipulations through the update formulas of the time integration scheme.

2.2.2 G-α time integration scheme for second-order systems

The G-α scheme for second-order was introduced in [47] and includes as subsets many of popular time integration schemes in structural mechanics, e.g., Newmark [212], HHT-α [213], WBZ-α [214]. For a unified presentation of those family of schemes, see [187, 189]. Let the time interval $[0, T]$ be divided into N_t subintervals, with the discrete points in time $\{t_n\}_{n=0}^{N_t}$, and

let $\Delta t_n = t_{n+1} - t_n$ be the time step size in the nth interval. Using the G-α scheme, the Following kinematic variables, evaluated at different instants within the time step, are required

$$\boldsymbol{d}_{n+\alpha_f} = (1-\alpha_f)\boldsymbol{d}_n + \alpha_f \boldsymbol{d}_{n+1}, \tag{2.191}$$

$$\dot{\boldsymbol{d}}_{n+\alpha_f} = (1-\alpha_f)\dot{\boldsymbol{d}}_n + \alpha_f \dot{\boldsymbol{d}}_{n+1}, \tag{2.192}$$

$$\tilde{\ddot{\boldsymbol{d}}}_{n+\alpha_m} = (1-\alpha_m)\tilde{\ddot{\boldsymbol{d}}}_n + \alpha_m \tilde{\ddot{\boldsymbol{d}}}_{n+1}, \tag{2.193}$$

where $\boldsymbol{d}_{n+1}$, $\dot{\boldsymbol{d}}_{n+1}$ and $\tilde{\ddot{\boldsymbol{d}}}_{n+1}$ are the displacement, velocity and *auxiliary* acceleration variables at t_{n+1}, and $\boldsymbol{d}_n$, $\dot{\boldsymbol{d}}_n$ and $\tilde{\ddot{\boldsymbol{d}}}_n$ represent the same variables at time t_n, respectively. The values α_f and α_f^m are algorithmic parameters of the scheme. Evaluating the auxiliary acceleration variable $\tilde{\ddot{\boldsymbol{d}}}$ at a different time instant than the rest of the terms is one of the mechanisms used in the G-α scheme for introducing an optimal amount of numerical high-frequency damping. However, the auxiliary acceleration variable $\tilde{\ddot{\boldsymbol{d}}}$(also called the acceleration-like variable) is shown in [48] to possess only first-order accuracy in time (in contrast to the second-order accuracy for displacement and velocity variables). The second-order accuracy of the acceleration can be restored by introducing the *true* acceleration variable $\ddot{\boldsymbol{d}}$, which is related to the auxiliary acceleration variable $\tilde{\ddot{\boldsymbol{d}}}$ through the following relation [48]:

$$(1-\alpha_m)\,\tilde{\ddot{\boldsymbol{d}}}_n + \alpha_m \tilde{\ddot{\boldsymbol{d}}}_{n+1} = (1-\alpha_f)\,\ddot{\boldsymbol{d}}_n + \alpha_f \ddot{\boldsymbol{d}}_{n+1}. \tag{2.194}$$

Using the second-order accurate acceleration values is crucial in the fluid-subcycling method presented in section 5.3.1. Therefore, the true acceleration $\ddot{\boldsymbol{d}}$ is calculated and tracked in time. The kinematic variables are related through the following two update formulae

$$\boldsymbol{d}_{n+1} = \boldsymbol{d}_n + \Delta t \dot{\boldsymbol{d}}_n + \Delta t^2 \left((\frac{1}{2} - \beta)\tilde{\ddot{\boldsymbol{d}}}_n + \beta \tilde{\ddot{\boldsymbol{d}}}_{n+1} \right), \tag{2.195}$$

$$\dot{\boldsymbol{d}}_{n+1} = \dot{\boldsymbol{d}}_n + \Delta t \left((1-\gamma)\,\tilde{\ddot{\boldsymbol{d}}}_n + \gamma \tilde{\ddot{\boldsymbol{d}}}_{n+1} \right), \tag{2.196}$$

where β and γ are algorithmic parameters of the scheme. The following relations for the algorithmic parameters ensure the second-order accuracy and unconditional stability of the scheme

$$\alpha_f = \frac{1}{1+\rho_\infty}, \quad \alpha_m = \frac{2-\rho_\infty}{1+\rho_\infty}, \quad \beta = \frac{(1+\alpha_m-\alpha_f)^2}{4} \quad \gamma = \frac{1}{2} + \alpha_m - \alpha_f, \tag{2.197}$$

where $0 \leqslant \rho_\infty \leqslant 1$ is the spectral radius of the amplification matrix of the scheme as $\Delta t \to \infty$ and is used to control the amount of numerical high-frequency damping of the scheme.

The equilibrium equation of the problem is satisfied in a weighted average sense in each time step. For the solid model problem, the time-discretized equilibrium equation reads

$$\boldsymbol{M}\tilde{\ddot{\boldsymbol{d}}}_{n+\alpha_m} + \boldsymbol{K}\boldsymbol{d}_{n+\alpha_f} = \boldsymbol{\lambda}_{n+\alpha_f}. \tag{2.198}$$

Note that the solid interface force $\boldsymbol{\lambda}$ is also evaluated at $t_{n+\alpha_f}$.

The G-α scheme for first-order systems can also be used for the time integration of the solid problems by rewriting the structural equations as a system of two first-order equations and without increasing the number of unknowns, as shown in [205]. In this work, the structural problem is discretized using the EG-α scheme in order to have a consistent and uniform time discretization of the FSI problem.

2.2.3 EG-α time integration scheme for structural problems

In this section the EG-α scheme for the time discretization of structural mechanics problems is derived. The main advantage of the EG-α for structural mechanics problems is that it allows a uniform and consistent temporal discretization of the FSI problems. It is possible to directly apply the EG-α scheme derived in section 2.1.5 to Eqs. 2.189-2.190. However, the resulting linear system of equations would have double the number of unknowns corresponding to the displacement and velocity variables, $\boldsymbol{d}$ and $\boldsymbol{v}$, respectively. Inspired by the approach in [205] for the classical averaged form of the G-α scheme for first-order systems, a variant of the EG-α for structural problems without increasing the number of unknowns in the system of linear equations is derived in the following.

The variables involved in the temporal discretization of Eqs. 2.189-2.190 using the EG-α scheme are: $\boldsymbol{d}$, $\tilde{\dot{\boldsymbol{d}}}$, $\dot{\boldsymbol{d}}$, $\boldsymbol{v}$, $\tilde{\boldsymbol{v}}$, $\dot{\boldsymbol{v}}$, which are related to each other using the following G-α formulae:

$$\boldsymbol{d}_{n+1} = \boldsymbol{d}_n + \Delta t((1-\gamma)\tilde{\dot{\boldsymbol{d}}}_n + \gamma\tilde{\dot{\boldsymbol{d}}}_{n+1}), \tag{2.199}$$

$$\boldsymbol{v}_{n+1} = \boldsymbol{v}_n + \Delta t\left((1-\gamma)\tilde{\boldsymbol{v}}_n + \gamma\tilde{\boldsymbol{v}}_{n+1}\right), \tag{2.200}$$

$$(1-\alpha_m)\,\tilde{\dot{\boldsymbol{d}}}_n + \alpha_m\tilde{\dot{\boldsymbol{d}}}_{n+1} = (1-\alpha_f)\,\dot{\boldsymbol{d}}_n + \alpha_f\dot{\boldsymbol{d}}_{n+1}, \tag{2.201}$$

$$(1-\alpha_m)\,\tilde{\boldsymbol{v}}_n + \alpha_m\tilde{\boldsymbol{v}}_{n+1} = (1-\alpha_f)\,\dot{\boldsymbol{v}}_n + \alpha_f\dot{\boldsymbol{v}}_{n+1}. \tag{2.202}$$

Discretizing the kinematic constraint in Eq. 2.190 using the EG-α scheme leads to

$$\dot{\boldsymbol{d}}_{n+1} = \boldsymbol{v}_{n+1}, \quad \dot{\boldsymbol{d}}_n = \boldsymbol{v}_n. \tag{2.203}$$

Therefore, only one of the variables $\dot{\boldsymbol{d}}$ and $\boldsymbol{v}$ is required to be stored. Replacing $\dot{\boldsymbol{d}}_{n+1}$ and $\dot{\boldsymbol{d}}_n$ with $\boldsymbol{v}_{n+1}$ and $\boldsymbol{v}_n$ in Eq. 2.201 and inserting the resulting

expression for $\boldsymbol{v}_{n+1}$ in Eq. 2.199 leads to

$$\boldsymbol{v}_{n+1} = \frac{\alpha_m}{\alpha_f \gamma \Delta t}(\boldsymbol{d}_{n+1} - \boldsymbol{d}_n) + \frac{\gamma - \alpha_m}{\alpha_f \gamma}\tilde{\dot{\boldsymbol{d}}}_n - \frac{1-\alpha_f}{\alpha_f}\boldsymbol{v}_n. \tag{2.204}$$

Inserting Eq. 2.204 and the expression for $\dot{\boldsymbol{v}}_{n+1}$ from Eq. 2.202 into Eq. 2.200 leads to

$$\dot{\boldsymbol{v}}_{n+1} = \frac{\alpha_m^2}{\alpha_f^2 \gamma^2 \Delta t^2}(\boldsymbol{d}_{n+1} - \boldsymbol{d}_n) + \frac{\alpha_m(\gamma - \alpha_m)}{\alpha_f^2 \gamma^2 \Delta t}\tilde{\dot{\boldsymbol{d}}}_n - \frac{\alpha_m}{\alpha_f^2 \gamma \Delta t}\boldsymbol{v}_n + \frac{\gamma - \alpha_m}{\alpha_f \gamma}\tilde{\dot{\boldsymbol{v}}}_n - \frac{1-\alpha_f}{\alpha_f}\dot{\boldsymbol{v}}_n. \tag{2.205}$$

From Eqs. 2.204 and 2.205, we have

$$\frac{\partial \boldsymbol{v}_{n+1}}{\partial \boldsymbol{d}_{n+1}} = \frac{\alpha_m}{\alpha_f \gamma \Delta t}, \quad \frac{\partial \dot{\boldsymbol{v}}_{n+1}}{\partial \boldsymbol{d}_{n+1}} = \frac{\alpha_m^2}{\alpha_f^2 \gamma^2 \Delta t^2}. \tag{2.206}$$

For a semi-discretized linear structural problem with the mass matrix $\boldsymbol{M}$, stiffness matrix $\boldsymbol{K}$ and damping matrix $\boldsymbol{C}$, and external force vector $\boldsymbol{F}$, the solution to $\boldsymbol{d}_{n+1}$ can be found through solving the following system:

$$\begin{aligned}
&\left(\frac{\alpha_m^2}{\alpha_f^2 \gamma^2 \Delta t^2}\boldsymbol{M} + \frac{\alpha_m}{\alpha_f \gamma \Delta t}\boldsymbol{C} + \boldsymbol{K}\right)\boldsymbol{d}_{n+1} = \\
&\boldsymbol{M}\left(\frac{\alpha_m^2}{\alpha_f^2 \gamma^2 \Delta t^2}\boldsymbol{d}_n - \frac{\alpha_m(\gamma - \alpha_m)}{\alpha_f^2 \gamma^2 \Delta t}\tilde{\dot{\boldsymbol{d}}}_n + \frac{\alpha_m}{\alpha_f^2 \gamma \Delta t}\boldsymbol{v}_n - \frac{\gamma - \alpha_m}{\alpha_f \gamma}\tilde{\dot{\boldsymbol{v}}}_n + \frac{1-\alpha_f}{\alpha_f}\dot{\boldsymbol{v}}_n\right) \\
&+\boldsymbol{C}\left(\frac{\alpha_m}{\alpha_f \gamma \Delta t}\boldsymbol{d}_n - \frac{\gamma - \alpha_m}{\alpha_f \gamma}\tilde{\dot{\boldsymbol{d}}}_n + \frac{1-\alpha_f}{\alpha_f}\boldsymbol{v}_n\right) \\
&+\boldsymbol{F}_{n+1}.
\end{aligned} \tag{2.207}$$

After solving for $\boldsymbol{d}_{n+1}$, the values of $\boldsymbol{v}_{n+1}$, $\dot{\boldsymbol{v}}_{n+1}$, $\tilde{\dot{\boldsymbol{d}}}_{n+1}$, and $\tilde{\dot{\boldsymbol{v}}}_{n+1}$ are updated using Eqs. 2.204, 2.205, 2.201, and 2.202, respectively. Note that the values of $\tilde{\dot{\boldsymbol{d}}}_{n+1}$, and $\tilde{\dot{\boldsymbol{v}}}_{n+1}$ do not necessarily need to be stored separately and can overwrite the values of $\tilde{\dot{\boldsymbol{d}}}_n$, and $\tilde{\dot{\boldsymbol{v}}}_n$ before proceeding to the next time step.

For nonlinear problems, a predictor-multicorrector approach is suitable. The predictor used here is based on the assumption that the value of velocity remains the same when moving from one time step to the next. The values of the kinematic variables at the predictor stage (the zeroth nonlinear iteration)

read:

$$\boldsymbol{v}_{n+1}^{(0)} = \boldsymbol{v}_n \tag{2.208}$$

$$\boldsymbol{d}_{n+1}^{(0)} = \boldsymbol{d}_n - \frac{(\gamma - \alpha_m)\Delta t}{\alpha_m}\tilde{\boldsymbol{d}}_n + \frac{\gamma \Delta t}{\alpha_m}\boldsymbol{v}_n \tag{2.209}$$

$$\dot{\boldsymbol{v}}_{n+1}^{(0)} = \frac{\gamma - \alpha_m}{\alpha_f \gamma}\tilde{\dot{\boldsymbol{v}}}_n - \frac{1 - \alpha_f}{\alpha_f}\dot{\boldsymbol{v}}_n. \tag{2.210}$$

The nonlinear problem in the kth iteration is solved using

$$\left(\frac{\alpha_m^2}{\alpha_f^2 \gamma^2 \Delta t^2}\boldsymbol{M} + \frac{\alpha_m}{\alpha_f \gamma \Delta t}\boldsymbol{C} + \boldsymbol{K}^t\right)\Delta \boldsymbol{d}_{n+1}^{(k+1)} = -\boldsymbol{R}_{n+1}^{(k)}, \tag{2.211}$$

where $\Delta \boldsymbol{d}_{n+1}^{(k+1)} = \boldsymbol{d}_{n+1}^{(k+1)} - \boldsymbol{d}_{n+1}^{(k)}$, $\boldsymbol{K}^t$ is the tangent stiffness matrix, and $\boldsymbol{R}_{n+1}^{(k)}$ is the residual of the momentum equation at the kth iteration, defined as

$$\boldsymbol{R}_{n+1}^{(k)} = \boldsymbol{M}\dot{\boldsymbol{v}}_{n+1}^{(k)} + \boldsymbol{C}\boldsymbol{v}_{n+1}^{(k)} + \boldsymbol{G}_{n+1}^{(k)}(\boldsymbol{d}_{n+1}^{(k)}) - \boldsymbol{F}_{n+1}, \tag{2.212}$$

where $\boldsymbol{G}_{n+1}^{(k)}$ is the vector of internal forces at the kth iteration.

After solving the linear system of equations in the kth nonlinear iteration, the kinematic variables are updated in the corrector stage as follows:

$$\boldsymbol{d}_{n+1}^{(k+1)} = \boldsymbol{d}_{n+1}^{(k)} + \Delta \boldsymbol{d}_{n+1}^{(k+1)}, \tag{2.213}$$

$$\boldsymbol{v}_{n+1}^{(k+1)} = \boldsymbol{v}_{n+1}^{(k)} + \frac{\alpha_m}{\alpha_f \gamma \Delta t}\Delta \boldsymbol{d}_{n+1}^{(k+1)}, \tag{2.214}$$

$$\dot{\boldsymbol{v}}_{n+1}^{(k+1)} = \dot{\boldsymbol{v}}_{n+1}^{(k)} + \frac{\alpha_m^2}{\alpha_f^2 \gamma^2 \Delta t^2}\Delta \boldsymbol{d}_{n+1}^{(k+1)}. \tag{2.215}$$

The values of the auxiliary variables $\tilde{\boldsymbol{d}}_{n+1}$, and $\tilde{\dot{\boldsymbol{v}}}_{n+1}$ are updated at the end of the time step using Eqs. 2.201, and 2.202, respectively.

2.3 Fluid-structure interaction problem

In an FSI problem, the initial configuration of the problem's domain $\Omega_0 \subset \mathbb{R}^d$, $d = \{2, 3\}$, is partitioned into the solid and fluid subdomains such that there are no overlapping or gaps between the two domains, i.e., $\Omega_0 = \Omega_0^s \cup \Omega_0^f$. The shared interface of the fluid and structural domains is denoted by Γ_0^i, i.e., $\Gamma_0^i = \partial\Omega_0^s \cap \partial\Omega_0^f$. The problem's domain undergoes deformations during an FSI simulation. The initial configuration often serves as the reference configuration. Let $\mathcal{A}_t^s$ and $\mathcal{A}_t^f$ be sufficiently regular and invertible mappings that relate the reference and the current configurations of the solid and fluid

subdomains, i.e.,

$$\mathcal{A}_t^s : \Omega_0^s \to \Omega_t^s, \qquad \boldsymbol{x}(\mathbf{x},t) = \mathcal{A}_t^s(\mathbf{x}) \quad \forall \mathbf{x} \in \Omega_0^s, \tag{2.216}$$

$$\mathcal{A}_t^f : \Omega_0^f \to \Omega_t^f, \qquad \boldsymbol{x}(\mathbf{x},t) = \mathcal{A}_t^f(\mathbf{x}) \quad \forall \mathbf{x} \in \Omega_0^f. \tag{2.217}$$

The mappings $\mathcal{A}_t^s$ and $\mathcal{A}_t^f$ define the motion of the solid and fluid subdomains respectively and coincide on the FSI interface Γ^i. The partitioning of the problem's domain holds as well for any time $0 < t$ such that $\Omega(t) \equiv \Omega_t = \Omega_t^s \cup \Omega_t^f$ and $\Gamma_t^i = \partial\Omega_t^s \cap \partial\Omega_t^f$. In the present work, the structural problem is solved in Lagrangian reference frame and the fluid problem is solved in the ALE reference frame [68, 69, 70]. Therefore, the motion of the solid domain coincides with the motion of the material particles. However, the motion of the fluid domain does not define the motion of the material particles.

2.3.1 Motion of the fluid problem

The ALE mapping $\mathcal{A}_t^f$ defines the motion of the fluid domain and is updated in accordance with the motion of the FSI interface. The mapping $\mathcal{A}_t^f$ can be characterized using a partial differential equation subjected to Dirichlet boundary conditions on the FSI interface due to the motion of the structure. Let $\boldsymbol{x}$ and $\mathbf{x}$ be the coordinates of the current and reference configuration of the fluid domain, respectively, and $\boldsymbol{d}^m$ be the displacement of the domain with respect to the reference configuration. It holds that

$$\boldsymbol{x}(\mathbf{x},t) = \mathbf{x} + \boldsymbol{d}^m(\mathbf{x},t) = \mathcal{A}_t^f(\mathbf{x}). \tag{2.218}$$

The displacement of the reference domain is governed by a PDE representing a fictitious elastic structure

$$-\nabla_{\mathrm{x}} \cdot (2\mu^m \nabla_{\mathrm{x}}^s \boldsymbol{d}^m + \lambda^m (\nabla_{\mathrm{x}} \cdot \boldsymbol{d}^m) \boldsymbol{I}) = \mathbf{0} \quad \text{in } \Omega_0^f, \tag{2.219}$$

where μ^m and λ^m are the Lamé parameters of the fictitious elastic structure which are selected with the aim of preserving the mesh quality of the deformed fluid domain throughout the simulation. An effective approach presented in [78, 202] for preserving the mesh quality is to make the smaller elements, which are usually placed next to the FSI interface, stiffer by dividing the parameters of the fictitious structure by the Jacobian determinant of the mapping from the reference cell to each element in the mesh. Eq. 2.219 is subjected to the following Dirichlet boundary condition

$$\boldsymbol{d}^m = \boldsymbol{d}^i \quad \text{on } \Gamma_0^i, \tag{2.220}$$

where $\boldsymbol{d}^i$ is the displacement of the FSI interface dictated by the structure's motion.

Let $\boldsymbol{\mathcal{V}}^m \coloneqq \boldsymbol{H}_D^1(\Omega_0^f)$ and $\boldsymbol{\mathcal{W}}^m \coloneqq \boldsymbol{H}_0^1(\Omega_0^f)$ be the spaces of trial and test functions, respectively, and $\boldsymbol{\mathcal{V}}_h^m$ and $\boldsymbol{\mathcal{W}}_h^m$ be their corresponding finite-

dimensional spaces. After spatial discretization using the Galerkin method, the semi-discrete form of the mesh moving problem reads: Find $\boldsymbol{d}^m \in \boldsymbol{\mathcal{V}}_h^m$ such that $\forall \boldsymbol{w}^m \in \boldsymbol{\mathcal{W}}_h^m$

$$B^m(\boldsymbol{w}^m, \boldsymbol{d}^m) = (\nabla_{\mathrm{x}}^s \boldsymbol{w}^m, 2\mu^m \nabla_{\mathrm{x}}^s \boldsymbol{d}^m)_{\Omega_0^f} + (\nabla_{\mathrm{x}} \cdot \boldsymbol{w}^m, \lambda^m \nabla_{\mathrm{x}} \cdot \boldsymbol{d}^m)_{\Omega_0^f} = 0. \tag{2.221}$$

The displacement in the fluid domain is then related to the mesh velocity through the time integration rules of the fluid solver.

2.3.2 Coupling of the fluid and solid problems

The numerical FSI simulation is carried out by solving the fluid, fluid mesh deformation, and solid problems together with appropriate interface coupling conditions. The first interface condition is the so-called *geometric coupling condition* which specifies that the solid and the fluid subdomains match at the FSI interface throughout the time without any gaps or overlapping between them. In other words, the deformation mappings of the solid and the fluid domains, $\mathcal{A}_t^s$ and $\mathcal{A}_t^f$, coincide at the FSI interface at all times. Assuming that the initial configurations of the solid and fluid domains match exactly at the FSI interface, the condition on the solid and fluid deformation mappings leads to

$$\mathcal{A}_t^f(\mathbf{x}, t) = \mathcal{A}_t^s(\mathbf{x}, t) \Rightarrow \mathbf{x} + \boldsymbol{d}^m = \mathbf{x} + \boldsymbol{d}^s \Rightarrow \boldsymbol{d}^m = \boldsymbol{d}^s \quad \forall \mathbf{x} \in \Gamma_0^i. \tag{2.222}$$

In other words, The displacement of the solid matches the mesh displacement of the fluid domain on the FSI interface. This coupling condition has a geometrical nature and plays an essential role in guaranteeing the consistency of coupling the two problems solved in different reference frames [206]. The geometric coupling condition can be also stated in the spatial configuration as

$$\boldsymbol{d}^m \circ (\mathcal{A}_t^f)^{-1} = \boldsymbol{d}^s \circ (\mathcal{A}_t^s)^{-1} \quad \text{on } \Gamma_t^i. \tag{2.223}$$

For the sake of simplicity, we will hereafter write the condition as $\boldsymbol{d}^m = \boldsymbol{d}^s$ even on Γ_t^i, which belongs to the current configuration.

Another interface coupling condition between the solid and the fluid problems is the *kinematic coupling condition* which states that the solid particle velocity matches the fluid velocity on the FSI interface. In certain situations, the kinematic condition can be relaxed to include only the equality of the normal velocities of the solid and fluid on the FSI interface. Note that the particle velocity in the solid $\boldsymbol{v}^s$ is defined in the Lagrangian reference frame, but the fluid velocity $\boldsymbol{v}^f$ is defined in the spatial configuration. Therefore, the kinematic condition on the FSI interface is more correctly written as $\boldsymbol{v}^f = \boldsymbol{v}^s \circ (\mathcal{A}_t^s)^{-1}$. However, for the sake of brevity, the kinematic condition is hereafter written as

$$\boldsymbol{v}^f = \boldsymbol{v}^s \text{ on } \Gamma_t^i. \tag{2.224}$$

The final coupling condition on the FSI interface is the *dynamic coupling condition*, stating that the fluid and solid normal stresses on the FSI interface are in equilibrium, i.e.,

$$\mathbf{F}^s \boldsymbol{\Sigma}^s \cdot \boldsymbol{n}_0^s + \boldsymbol{\sigma}^f \cdot \boldsymbol{n}_t^f = 0. \tag{2.225}$$

Note again that the solid problem is stated in the initial configuration, while the fluid problem is stated in the deformed configuration. In order to simplify the notation, the solid stresses are hereafter stated in terms of the transformed Cauchy stresses $\boldsymbol{\sigma}^s$ consistent with the second Piola–Kirchhoff stresses, leading to the following simple presentation of the dynamic coupling condition

$$\boldsymbol{\sigma}^s \cdot \boldsymbol{n}_t^s + \boldsymbol{\sigma}^f \cdot \boldsymbol{n}_t^f = 0 \text{ on } \Gamma_t^i. \tag{2.226}$$

Furthermore, the normal vector on the FSI boundary is unified by setting $\boldsymbol{n} = \boldsymbol{n}_t^s = -\boldsymbol{n}_t^f$, leading to the following simple presentation of the dynamic coupling condition

$$\boldsymbol{\sigma}^s \boldsymbol{n} = \boldsymbol{\sigma}^f \boldsymbol{n} \;\Rightarrow \boldsymbol{\lambda}^s = \boldsymbol{\lambda}^f \text{on } \Gamma_t^i, \tag{2.227}$$

where $\boldsymbol{\lambda}^s$ and $\boldsymbol{\lambda}^f$ are the traction vectors on the FSI boundary.

In the present work, the overall FSI problem is decomposed into the fluid and the solid problems using a Dirichlet-Neumann decomposition approach. The geometric and the kinematic coupling conditions are included in the mesh-moving and the fluid problems as Dirichlet boundary conditions. On the other hand, the dynamic coupling condition is contained in the solid problem as a Neumann boundary condition on the FSI interface. The normal stresses of the fluid problem on the FSI interface are calculated using the so-called variationally consistent approach through the information already contained in the variational form of the discrete fluid problem [51, 52, 53, 54, 56, 58, 59] (see Eq. 2.49).

2.3.3 Partitioned strongly-coupled FSI problem

The FSI problem is solved here using a partitioned strongly coupled approach based on the Dirichlet-Neumann decomposition of the overall problem. The solid, mesh-moving and fluid problems are solved iteratively until the coupling conditions are fully satisfied, and convergence is achieved (up to a numerical tolerance). In addition to being straightforward to implement, this approach fits naturally to (most) FSI problems, as the motion of the fluid domain is determined by the motion of the structure and, in return, the normal stresses of the fluid are applied on the structure which subsequently prompts the deformation of the structure. This coupling approach can also be very efficient when combined with advanced convergence acceleration techniques. The partitioned strongly coupled technique considered in the present study requires only the iterative exchange of interface data between the subdomains

without the need to solve additional systems on the FSI interface, which can be costly for large FSI problems, in particular in the context of coupling domains with non-matching time step sizes.

The partitioned FSI problem can be concisely written in terms of operators acting on the interface data. These nonlinear interface operators represent the fluid and the structural solvers with the input and output data restricted to the FSI interface. The Dirichlet-to-Neumann map of the fluid solver is represented by the operator

$$\boldsymbol{\lambda} = \boldsymbol{\mathcal{F}}(\boldsymbol{d}), \tag{2.228}$$

where $\boldsymbol{d}$ and $\boldsymbol{\lambda} = -\boldsymbol{\sigma} \cdot \boldsymbol{n}$ are the fluid displacement and reaction force on the FSI interface, respectively (Note that instead of the displacement, the solver operators can be formulated in terms of velocity as well). The mesh-moving problem is included in the fluid interface operator, i.e., the mesh-moving problem is solved using the interface displacement as input, the fluid domain is updated, the mesh velocity is calculated, and finally, the fluid problem is solved using the interface velocity as input. The Neumann-to-Dirichlet map of the structural solver is represented by

$$\boldsymbol{d} = \boldsymbol{\mathcal{S}}(\boldsymbol{\lambda}). \tag{2.229}$$

Using a Gauss-Seidel type scheme for solving the partitioned FSI problem, the fluid and the structure are solved sequentially in an iterative manner. One iteration of solving the fluid and the structure sequentially can be represented by the combined operator $\boldsymbol{\mathcal{H}}$

$$\boldsymbol{\mathcal{H}}(\boldsymbol{d}) \coloneqq \boldsymbol{\mathcal{S}} \circ \boldsymbol{\mathcal{F}}(\boldsymbol{d}). \tag{2.230}$$

The coupled problem can be solved using fixed point iterations $\boldsymbol{d}^{(l+1)} = \boldsymbol{\mathcal{H}}\left(\boldsymbol{d}^{(l)}\right)$, possibility with under-relaxation, where $\boldsymbol{d}^{(l)}$ denotes the displacement iterate at the lth iteration. Alternatively, the interface problem can be stated as a root-finding problem $\boldsymbol{\mathcal{R}}(\boldsymbol{d}) \coloneqq \boldsymbol{\mathcal{H}}(\boldsymbol{d}) - \boldsymbol{d} = 0$, where $\boldsymbol{\mathcal{R}} \coloneqq \boldsymbol{\mathcal{H}} - \boldsymbol{\mathcal{I}}$ represents the interface residual operator. The root-finding problem is solved in this work using an interface quasi-Newton method. Denoting by $\boldsymbol{r} \coloneqq \boldsymbol{\mathcal{R}}(\boldsymbol{d}) = \boldsymbol{\mathcal{H}}(\boldsymbol{d}) - \boldsymbol{d}$ the interface residual vector, the interface quasi-Newton method is formulated as

$$\delta\boldsymbol{d}^{(l)} = -\boldsymbol{K}^{(l)}\boldsymbol{r}^{(l)}, \tag{2.231}$$

where $\boldsymbol{K}^{(l)}$ is an approximation to the inverse of the interface Jacobian $\boldsymbol{\mathcal{R}}'^{(l)} = \frac{\partial \boldsymbol{\mathcal{R}}}{\partial \boldsymbol{d}}\Big|_{\boldsymbol{d}^{(l)}}$. Various quasi-Newton methods have been proposed in the literature for the convergence acceleration of partitioned strongly coupled problems, which differ mainly in generating the approximation of the interface Jacobian or directly its inverse [111, 120, 121, 110, 119, 122]. Chapters 3 and 4 in this work are concerned with the convergence acceleration of partitioned strongly coupled problems using quasi-Newton methods.

2.3.4 Time discretization of the coupled problem

In this section, the time discretization of the coupling conditions is discussed. The dynamic coupling condition is satisfied in every time step at t_{n+1}

$$\boldsymbol{\lambda}^s_{n+1} = \boldsymbol{\lambda}^f_{n+1}. \tag{2.232}$$

As discussed previously, the fluid interface force $\boldsymbol{\lambda}^f_{n+1}$ is updated from $\boldsymbol{\lambda}^f_{n+\alpha^f_f}$ when using the G-α scheme. Furthermore, the solid interface force is required at the intermediate time instant $t_{n+\alpha^s_f}$, which can be interpolated using the values $\boldsymbol{\lambda}^s_{n+1}$ and $\boldsymbol{\lambda}^s_n$. The effect of this consistent (temporal) interpolation of the interface forces between the fluid and structure on the stability and accuracy of FSI simulations was first studied in [61] using the G-α time integration scheme. However, $\boldsymbol{\lambda}^s_{n+1}$ and $\boldsymbol{\lambda}^f_{n+1}$ are naturally evaluated at t_{n+1} using the EG-α scheme. Therefore, no further interpolation of interface forces is required.

The kinematic coupling conditions 5.3 are simultaneously satisfied at the continuous level. However, at the discrete level, the continuity of displacements, velocities and accelerations at the coupling interface can hold simultaneously only if the same time step size and time integration scheme with the same set of parameters is used in the fluid and solid subdomains. Here, three approaches for handling discrete kinematic coupling constraints in FSI problems are discussed. A common approach in the literature is based on strongly satisfying the discrete displacement continuity condition at t_{n+1}

$$\boldsymbol{d}^m_{n+1} = \boldsymbol{d}^s_{n+1} \quad \text{on } \Gamma^i_t. \tag{2.233}$$

Subsequently, the fluid velocity on the FSI boundary $\boldsymbol{v}^{f,\Gamma^i}_{n+1}$ is calculated such that it is consistent with the time integration scheme of the fluid domain. Note that in this approach, the kinematic coupling condition, Eq. 2.224, is in general not guaranteed to be exactly satisfied at the discrete level. The input to the combined mesh-moving and fluid interface operator, Eq. 2.228, is naturally the interface displacement $\boldsymbol{d}_{n+1} = \boldsymbol{d}^m_{n+1} = \boldsymbol{d}^s_{n+1}$, i.e.,

$$\boldsymbol{\lambda}_{n+1} = \boldsymbol{\mathcal{F}}\left(\boldsymbol{d}_{n+1}\right). \tag{2.234}$$

Consequently, the FSI convergence acceleration technique operates as well on the interface displacement, i.e., it predicts in every coupling iteration a new iterate for $\boldsymbol{d}_{n+1}$ [106, 91, 110, 111, 92, 115, 119]. The same approach is used in [177, 178], where the quasi-Newton convergence acceleration techniques are used in the FSI problems with non-matching time steps.

In the second approach for treating the time-discretized geometric and kinematic coupling conditions, the discrete velocity continuity condition is

strongly satisfied at t_{n+1}, i.e.,

$$\boldsymbol{v}_{n+1}^{f} = \boldsymbol{v}_{n+1}^{s} \quad \text{on } \Gamma_t^i. \tag{2.235}$$

Subsequently, the mesh displacement on the FSI boundary is calculated consistently using the rules of the time integration scheme of the fluid domain. In this approach, the satisfaction of the geometric coupling condition is generally not guaranteed at the discrete level. The input to the combined mesh-moving and fluid interface operator is naturally the interface velocity $\boldsymbol{v}_{n+1} = \boldsymbol{v}_{n+1}^{m} = \boldsymbol{v}_{n+1}^{s}$, i.e.,

$$\boldsymbol{\lambda}_{n+1} = \boldsymbol{\mathcal{F}}\left(\boldsymbol{v}_{n+1}\right), \tag{2.236}$$

and the combined fluid and solid interface operator is

$$\boldsymbol{\mathcal{H}}\left(\boldsymbol{v}_{n+1}\right) \coloneqq \boldsymbol{\mathcal{S}} \circ \boldsymbol{\mathcal{F}}\left(\boldsymbol{v}_{n+1}\right). \tag{2.237}$$

The FSI convergence acceleration technique operates on the interface velocity (instead of displacement), i.e., it predicts in every coupling iteration a new iterate for $\boldsymbol{v}_{n+1}$.

Choosing to fulfill either Eq. 2.233 or 2.235 at the discrete level has its advantage. Satisfying the geometric coupling condition ensures that the solid and fluid domains match at the discrete level at all times, which follows the theoretical assumptions. On the other hand, the kinematic coupling condition is one of the two physically-related coupling conditions which drive the FSI problem. Therefore, satisfying the kinematic coupling condition is expected to have an important role in the physical accuracy of the results. A final possibility is to simultaneously enforce the continuity of discrete displacement, Eq. 2.233, and discrete velocity, Eq. 2.235, across the interface (dictated by the values in the solid domain in a Dirichlet-Neumann decomposition). However, the relation between the fluid velocity and displacement values on the coupling interface will not be anymore consistent with the time integration scheme utilized in the fluid domain. Furthermore, more operations are required for exchanging data between the fluid and structural solvers in a partitioned coupling environment. We will later compare these approaches for satisfying the kinematic coupling condition in the context of coupling domains with non-matching time steps. When coupling domains with matching time steps, choosing to satisfy either the discrete displacement or velocity continuity conditions generally has no significant impact on the simulation results. However, the choice of discrete kinematic continuity condition significantly impacts the FSI problems with non-matching time steps, as shown in section 5.3.

The final remark in this section is on the consistent prediction of the kinematic values at the beginning of each time step. As stated previously, a predictor-multicorrector solution procedure is used in both fluid and solid problems for solving the nonlinear systems (see Eqs. 2.76–2.76, and 2.208–2.210). The predictor stage in both problems is consistent with the EG-α

scheme and follows the same assumption that the velocity does not change when proceeding from one time step to the next. Consistent predictions for the kinematic variables of the moving mesh on the FSI interface at the beginning of the nth time step are given by

$$\boldsymbol{v}_{n+1}^{m} = \boldsymbol{v}_{n}^{m} \tag{2.238}$$

$$\boldsymbol{d}_{n+1}^{m} = \boldsymbol{d}_{n}^{m} - \frac{(\gamma - \alpha_m)\Delta t}{\alpha_m} \tilde{\boldsymbol{d}}_{n}^{m} + \frac{\gamma \Delta t}{\alpha_m} \boldsymbol{v}_{n}^{m} \tag{2.239}$$

$$\dot{\boldsymbol{v}}_{n+1}^{m} = \frac{\gamma - \alpha_m}{\alpha_f \gamma} \tilde{\boldsymbol{v}}_{n}^{m} - \frac{1 - \alpha_f}{\alpha_f} \dot{\boldsymbol{v}}_{n}^{m}. \tag{2.240}$$

The presented EG-α time integration scheme for the mesh motion, fluid, and structural problems is used for the temporal discretization of the FSI problems in chapters 3, 4, 5, and 6.

Chapter 3

Enhancing the robustness and efficiency of interface quasi-Newton methods

3.1 Introduction

In this work, we deal with a class of partitioned coupling strategies that allow the coupling of black-box solvers [91, 92, 93]. In such an approach, the black-box solvers exchange only limited data (related to the boundary conditions) restricted to the coupling interface. The scope of the present work further narrows down to partitioned *strong* (*implicit*) coupling methods in which the coupling conditions are fully satisfied through carrying out coupling iterations. One of the earliest coupling schemes used in strongly coupled partitioned FSI simulations is the (block) Gauss-Seidel method with or without relaxation [94, 95, 96, 97, 98]. The stability issues of partitioned methods based on the Gauss-Seidel scheme, due to the artificial added-mass effect, has been extensively studied in the literature [99, 100, 101, 102, 103]. The convergence of Gauss-Seidel (fixed-point) iterations is usually slow but can be accelerated by using dynamic relaxation, e.g., Aitken relaxation [91, 103, 104]. The fixed-point iterations with dynamic relaxation technique is simple to implement, and it can be effective in accelerating the convergence of the coupling iterations in certain cases. However, it still suffers from numerical instabilities due to strong added-mass effect. More stable and efficient convergence acceleration

techniques have been since introduced and successfully used for partitioned strongly coupled problems in the literature. The Interface-GMRES(R) method is introduced and analyzed in [105, 106, 107, 108], where the convergence of the fixed-point iterations is accelerated by Krylov-subspace methods. The application of vector extrapolation methods for the convergence acceleration of strongly coupled FSI simulations is studied in [109]. In [110], a method is introduced for solving the partitioned FSI problem using a block quasi-Newton method. In this method, later known as IBQN-LS [111, 112, 113, 92], the Jacobians of the fluid and the structural solvers are approximated using least squares models. A few other quasi-Newton methods have been since developed based on the idea of approximating Jacobians using least squares models. In [111], the IQN-ILS method is introduced for approximating the inverse of the Jacobian of the interface problem in the partitioned FSI simulations. The IQN-ILS method, further analyzed in [113, 114, 92, 112, 115], has been one of the most successful convergence acceleration techniques for the partitioned FSI problems in recent years. It has been shown that by reusing data from the previous time steps (a concept introduced first for the Interface-GMRES(R)), the IQN-ILS method (also referred to as QN-ILS [115, 112]) can exhibit excellent efficiency in time-dependent partitioned coupled simulations [113, 114, 116, 115, 117, 118, 119]. However, The performance of this technique depends on user-defined parameters that might have to be tuned for different problems. Another quasi-Newton method that performs well in partitioned coupled problems is the MVQN method [120, 121] (also referred to as IQN-MVJ or MVJ [116, 118, 119, 122]). This method has the advantage that it does not depend on a user-defined parameter (in contrast to IQN-ILS with reuse of information from the past time steps). However, the MVQN method requires the explicit storage of a dense square matrix with a size equal to the number of degrees of freedom (DOF) on the FSI interface. Therefore, the MVQN could be prohibitively costly in large problems. Different solutions have been suggested in the literature to address the issue of high costs associated with the explicit storage of the inverse Jacobian in the MVQN method [119, 122], albeit at the expense of reintroducing some user-defined parameters. In [123, 124, 116, 117, 119, 125], different aspects of parallel and/or efficient implementation of quasi-Newton methods are discussed. For some comparisons among different quasi-Newton methods for the convergence acceleration of partitioned coupled problems, see [92, 113, 116, 118, 121, 126, 122].

The remainder of this chapter is organized as follows. In section 3.2, the general concept of the interface quasi-Newton methods and, in particular, the IQN-ILS method are briefly introduced. In section 3.3, the issues surrounding the efficiency and robustness of some of the algorithms used in solving the least squares problem arising from the interface quasi-Newton methods are addressed. In sections 3.4–3.6, the techniques for enhancing the robustness and efficiency of the interface quasi-Newton methods, particularly the IQN-ILS method, in terms of convergence acceleration of the partitioned coupling iterations are introduced. The numerical results are presented and discussed

in section 3.7.

3.2 Quasi-Newton methods for solving partitioned FSI problems

By treating the solvers as black-box, the fluid and the structural solvers can be concisely represented as nonlinear operators with input and the output data restricted to the FSI interface. The fluid solver together with the mesh motion solver is represented by $\boldsymbol{y} = \boldsymbol{\mathcal{F}}(\boldsymbol{x})$ and the structural solver is represented by $\boldsymbol{x} = \boldsymbol{\mathcal{S}}(\boldsymbol{y})$, where $\boldsymbol{x}$ and $\boldsymbol{y}$ denote the position of the nodes and the traction on the FSI interface, respectively. If a Gauss-Seidel type scheme is used for solving the partitioned FSI problem, the fluid and the structure are solved sequentially in an iterative manner. We define a new operator $\boldsymbol{\mathcal{H}}$, which represents one iteration of solving the fluid and the structure sequentially: $\boldsymbol{\mathcal{H}}(\boldsymbol{x}) \coloneqq \boldsymbol{\mathcal{S}} \circ \boldsymbol{\mathcal{F}}(\boldsymbol{x})$. The coupled problem can be written as a fixed point problem $\boldsymbol{x} = \boldsymbol{\mathcal{H}}(\boldsymbol{x})$ or equivalently as a root-finding problem $\boldsymbol{\mathcal{R}}(\boldsymbol{x}) \coloneqq \boldsymbol{\mathcal{H}}(\boldsymbol{x}) - \boldsymbol{x} = 0$, where $\boldsymbol{\mathcal{R}} \coloneqq \boldsymbol{\mathcal{H}} - \boldsymbol{\mathcal{I}}$ is called the interface residual operator. The fixed point problem can be solved using fixed point iterations $\boldsymbol{x}^{(l+1)} = \boldsymbol{\mathcal{H}}\left(\boldsymbol{x}^{(l)}\right)$, where $\boldsymbol{x}^{(l)}$ denotes the iterate at the lth iteration. Alternatively, one can solve the root-finding problem using Newton-Raphson technique. If we denote the output of the $\boldsymbol{\mathcal{H}}$ operator in every iteration by $\tilde{\boldsymbol{x}}$, i.e., $\tilde{\boldsymbol{x}} \coloneqq \boldsymbol{\mathcal{H}}(\boldsymbol{x})$, the interface residual vector can be defined as $\boldsymbol{r} \coloneqq \boldsymbol{\mathcal{R}}(\boldsymbol{x}) = \boldsymbol{\mathcal{H}}(\boldsymbol{x}) - \boldsymbol{x}$. The Newton-Raphson method is formulated as

$$\boldsymbol{\mathcal{R}}'^{(l)} \delta \boldsymbol{x}^{(l)} = -\boldsymbol{r}^{(l)}, \tag{3.1}$$

where $\boldsymbol{\mathcal{R}}'^{(l)} = \frac{\partial \boldsymbol{\mathcal{R}}}{\partial \boldsymbol{x}}\Big|_{\boldsymbol{x}^{(l)}}$ is the Jacobian of $\boldsymbol{\mathcal{R}}$ with respect to $\boldsymbol{x}$ at the lth iteration, and $\delta \boldsymbol{x}^{(l)} = \boldsymbol{x}^{(l+1)} - \boldsymbol{x}^{(l)}$. Deriving and assembling the Jacobian could be a tedious or impossible task, if the black-box solvers do not provide the information needed for calculating the Jacobian. A solution to this problem is to approximate the Jacobian

$$\boldsymbol{J}^{(l)} \approx \boldsymbol{\mathcal{R}}'^{(l)} \tag{3.2}$$

or its inverse

$$\boldsymbol{K}^{(l)} \coloneqq \left(\boldsymbol{J}^{(l)}\right)^{-1} \approx \left(\boldsymbol{\mathcal{R}}'^{(l)}\right)^{-1} \tag{3.3}$$

using only the input and output information of the solvers in every iteration. This is the basis for the quasi-Newton methods.

3.2.1 IQN-ILS method

Assuming that l coupling iterations have already been carried out, the set of vectors $\tilde{\boldsymbol{x}}^{(l)}, \tilde{\boldsymbol{x}}^{(l-1)}, \ldots, \tilde{\boldsymbol{x}}^{(1)}, \tilde{\boldsymbol{x}}^{(0)}$ and $\boldsymbol{r}^{(l)}, \boldsymbol{r}^{(l-1)}, \ldots, \boldsymbol{r}^{(1)}, \boldsymbol{r}^{(0)}$ are available. After each coupling iteration, the difference between each vector of that iteration and the corresponding vector from the previous iteration is calculated:

$$\Delta\tilde{\boldsymbol{x}}^{(l)} = \tilde{\boldsymbol{x}}^{(l)} - \tilde{\boldsymbol{x}}^{(l-1)}, \quad \Delta\boldsymbol{r}^{(l)} = \boldsymbol{r}^{(l)} - \boldsymbol{r}^{(l-1)}. \tag{3.4}$$

The available difference vectors at the lth iteration are appended to the matrices $\boldsymbol{W}^{(l)}$ and $\boldsymbol{V}^{(l)}$ from the left

$$\boldsymbol{W}^{(l)} = \left[\Delta\tilde{\boldsymbol{x}}^{(l)} \mid \Delta\tilde{\boldsymbol{x}}^{(l-1)} \mid \ldots \mid \Delta\tilde{\boldsymbol{x}}^{(1)}\right], \quad \boldsymbol{V}^{(l)} = \left[\Delta\boldsymbol{r}^{(l)} \mid \Delta\boldsymbol{r}^{(l-1)} \mid \ldots \mid \Delta\boldsymbol{r}^{(1)}\right]. \tag{3.5}$$

When dealing with time-dependent problems, the information from the previous time steps might be relevant for the current time step due to the similar characteristics of the interface fixed point problem in the consecutive time steps. Reusing information from the previous time steps for accelerating the convergence of the coupling iterations was first introduced for the Interface-GMRES(R) method in [105, 106] and was subsequently used for the IQN-ILS method in [111, 113]. It has been demonstrated, e.g., in [111, 113, 115, 118, 116], that reusing the information from the previous time steps can substantially accelerate the convergence of the quasi-Newton iterations. In this technique, the difference vectors from s previous time steps are reused to enrich the $\boldsymbol{V}$ and $\boldsymbol{W}$ matrices, where s is a user-defined parameter.

The vector $\Delta\boldsymbol{r} = 0 - \boldsymbol{r}^{(l)}$ is approximated as a linear combination of the previous $\Delta\boldsymbol{r}^{(i)}$, i.e.,

$$\Delta\boldsymbol{r} \approx \boldsymbol{V}^{(l)}\boldsymbol{\alpha}, \tag{3.6}$$

where the vector $\boldsymbol{\alpha}$ contains the unknown coefficients of the linear combination. Eq. 3.6 is an over-determined system of linear equations, whose solution is equal to the solution of the following linear least squares problem

$$\min_{\boldsymbol{\alpha}} \left\| \Delta\boldsymbol{r} - \boldsymbol{V}^{(l)}\boldsymbol{\alpha} \right\|_2. \tag{3.7}$$

In the present work, Eq. 3.7 is solved using the QR factorization of $\boldsymbol{V}^{(l)}$. The details of the factorization process will be discussed in section 3.3. Let $\boldsymbol{V}^{(l)} = \boldsymbol{Q}_1\boldsymbol{R}$ be the *thin* (*economy-size*) QR factorization of $\boldsymbol{V}^{(l)}$. Then, the solution of $\boldsymbol{\alpha}$ in Eq. 3.7 can be found by solving the triangular system

$$\boldsymbol{R}\boldsymbol{\alpha} = \boldsymbol{Q}_1^T\Delta\boldsymbol{r}. \tag{3.8}$$

After solving for $\boldsymbol{\alpha}$, $\Delta\tilde{\boldsymbol{x}}$ is written as a linear combination of the previous $\Delta\tilde{\boldsymbol{x}}^{(i)}$ using the same $\boldsymbol{\alpha}$ coefficients, which are now known: $\Delta\tilde{\boldsymbol{x}} \approx \boldsymbol{W}^{(l)}\boldsymbol{\alpha}$. By noting that $\Delta\boldsymbol{r} = \Delta\tilde{\boldsymbol{x}} - \Delta\boldsymbol{x}$, one finally arrives at a relation for the prediction of the next iterate $\boldsymbol{x}^{(l+1)}$:

$$\Delta\boldsymbol{x} = \boldsymbol{x}^{(l+1)} - \boldsymbol{x}^{(l)} = \boldsymbol{W}^{(l)}\boldsymbol{\alpha} - \Delta\boldsymbol{r}. \tag{3.9}$$

By inserting the solution of $\boldsymbol{\alpha}$ from Eq. 3.8 in Eq. 3.9 and comparing the

results to Eq. 3.1, it becomes clear that one can regard

$$\boldsymbol{K}^{(l)} = \boldsymbol{W}^{(l)} \boldsymbol{R}^{-1} \boldsymbol{Q}_1^T - \boldsymbol{I} \tag{3.10}$$

as an approximation for the inverse of the Jacobian in the quasi-Newton iterations.

3.3 Solving the least squares problem

In this section, a brief description of the methods for the calculation of the QR factorization and solving the least squares problem is given in order to facilitate the presentation of the techniques used for enhancing the efficiency and robustness of the algorithms used in the interface quasi-Newton methods. In the following, the matrices are denoted by bold capital letters, e.g., $\boldsymbol{V}$. An element of matrix $\boldsymbol{V}$ is denoted by $V_{i,j}$, i.e., $V_{i,j} = \boldsymbol{V}(i,j)$. A column of the matrix, e.g., the jth column, is denoted by $\boldsymbol{V}_{:,j}$, i.e., $\boldsymbol{V}_{:,j} = \boldsymbol{V}(:,j)$. A block of matrix $\boldsymbol{V}$ is denoted by a subscript, e.g., $\boldsymbol{V}_1$ is the first column block of $\boldsymbol{V}$, and $\boldsymbol{V}_{11}$ denotes the first diagonal sub-block of $\boldsymbol{V}$. Hereafter, the superscript l representing the coupling iterations, e.g., in $\boldsymbol{V}^{(l)}$, is dropped as it is clear that the problem is solved in the current iteration. A new superscript k will be used for denoting the iterations in the QR factorization process. We are interested in solving the least squares problem

$$\min_{\boldsymbol{\alpha}=[\alpha_1,\ldots,\alpha_m]^T} \|\Delta \boldsymbol{r} - \boldsymbol{V}\boldsymbol{\alpha}\|_2 , \tag{3.11}$$

where $\boldsymbol{V} \in \mathbb{R}^{n\times m}$, $\Delta \boldsymbol{r} \in \mathbb{R}^n$ and $\boldsymbol{\alpha} \in \mathbb{R}^m$. The QR factorization of $\boldsymbol{V}$ leads to

$$\boldsymbol{V} = \boldsymbol{Q}\begin{bmatrix}\boldsymbol{R}\\ 0\end{bmatrix} = [\boldsymbol{Q}_1 \mid \boldsymbol{Q}_2]\begin{bmatrix}\boldsymbol{R}\\ 0\end{bmatrix} = \boldsymbol{Q}_1\boldsymbol{R}, \tag{3.12}$$

where $\boldsymbol{Q} \in \mathbb{R}^{n\times n}$ is an orthogonal matrix, and $\boldsymbol{R} \in \mathbb{R}^{m\times m}$ is upper triangular. If matrix $\boldsymbol{Q}$ is partitioned into two blocks $\boldsymbol{Q}_1$ and $\boldsymbol{Q}_2$ such that $\boldsymbol{Q}_1$ holds the first m columns, matrices $\boldsymbol{Q}_1$ and $\boldsymbol{Q}_2$ form orthogonal bases for the *range* of $\boldsymbol{A}$ and its complement, respectively [127, 128]. $\boldsymbol{V} = \boldsymbol{Q}_1\boldsymbol{R}$ is called the *thin* (or *economy-size*) QR factorization of $\boldsymbol{V}$. Owing to the fact that the vector 2-norm is invariant with respect to the orthogonal transformations, one can write

$$\|\Delta \boldsymbol{r} - \boldsymbol{V}\boldsymbol{\alpha}\|_2^2 = \left\|\boldsymbol{Q}^T\Delta \boldsymbol{r} - \boldsymbol{Q}^T\boldsymbol{V}\boldsymbol{\alpha}\right\|_2^2 = \left\|\boldsymbol{Q}_1^T\Delta \boldsymbol{r} - \boldsymbol{R}\boldsymbol{\alpha}\right\|_2^2 + \left\|\boldsymbol{Q}_2^T\Delta \boldsymbol{r}\right\|_2^2 . \tag{3.13}$$

Let

$$\boldsymbol{b} \coloneqq \boldsymbol{Q}^T\Delta \boldsymbol{r} = \begin{bmatrix}\boldsymbol{Q}_1^T\Delta \boldsymbol{r}\\ \boldsymbol{Q}_2^T\Delta \boldsymbol{r}\end{bmatrix} = \begin{bmatrix}\boldsymbol{b}_1\\ \boldsymbol{b}_2\end{bmatrix}. \tag{3.14}$$

The solution $\boldsymbol{\alpha}_{ls}$ to the least squares problem is found by solving the upper triangular system

$$\boldsymbol{R}\boldsymbol{\alpha}_{ls} = \boldsymbol{b}_1 = \boldsymbol{Q}_1^T \Delta \boldsymbol{r}, \tag{3.15}$$

and $\|\boldsymbol{b}_2\|_2$ is equal to the norm of the minimum residual:

$$\|\boldsymbol{b}_2\|_2 = \|\Delta \boldsymbol{r} - \boldsymbol{V}\boldsymbol{\alpha}_{ls}\|_2 . \tag{3.16}$$

3.3.1 Solving the least squares problem using Householder QR factorization

In the earlier works on the interface quasi-Newton methods using Householder QR factorization, e.g., in [111, 113, 92], *economy-size* QR factorization is used for solving the least squares problem. This requires the explicit calculation and storage of the $\boldsymbol{Q}_1$ matrix. In the current section, it is shown that the explicit representation of $\boldsymbol{Q}_1$ is not required, and a more efficient algorithm can be used instead.

Householder transformation (or reflection) $\boldsymbol{P}$ is a symmetric and orthogonal matrix of the form

$$\boldsymbol{P} = \boldsymbol{I} - \gamma \boldsymbol{v}\boldsymbol{v}^T, \quad \gamma = \frac{2}{\boldsymbol{v}^T\boldsymbol{v}}. \tag{3.17}$$

For any vector $\boldsymbol{a}$, the Householder vector $\boldsymbol{v}$ can be defined such that $\boldsymbol{P}\boldsymbol{a} = \|\boldsymbol{a}\|_2 \boldsymbol{e}_1$, where $\boldsymbol{e}_1$ is the first unit vector. There are some practical considerations regarding the computation of the Householder vector $\boldsymbol{v}$ (see, e.g., [128, p. 235]). For the sake of brevity, it is sufficient here to assume that there is a function available that if given a vector $\boldsymbol{a}$, generates the Householder vector: $[\gamma, \boldsymbol{v}] = \text{house}(\boldsymbol{a})$. Note that when applying a Householder transformation to a vector, matrix $\boldsymbol{P}$ is never explicitly created, i.e.,

$$\boldsymbol{P}\boldsymbol{a} = \left(\boldsymbol{I} - \gamma \boldsymbol{v}\boldsymbol{v}^T\right)\boldsymbol{a} = \boldsymbol{a} - \gamma\left(\boldsymbol{v}^T\boldsymbol{a}\right)\boldsymbol{v}. \tag{3.18}$$

Likewise, the premultiplication of a matrix $\boldsymbol{A}$ by $\boldsymbol{P}$ entails a matrix-vector multiplication and a rank one update to matrix $\boldsymbol{A}$:

$$\boldsymbol{P}\boldsymbol{A} = \left(\boldsymbol{I} - \gamma \boldsymbol{v}\boldsymbol{v}^T\right)\boldsymbol{A} = \boldsymbol{A} - \gamma \boldsymbol{v}\left(\boldsymbol{v}^T\boldsymbol{A}\right). \tag{3.19}$$

The Householder QR factorization is calculated by transforming matrix $\boldsymbol{V}$ in iterations. In every iteration, the elements of a column under the main diagonal of $\boldsymbol{V}$ are eliminated. Starting from $\boldsymbol{V}^{(1)} = \boldsymbol{V}$, one Householder transformation is applied at every iteration, i.e., $\boldsymbol{V}^{(k+1)} = \boldsymbol{P}^{(k)}\boldsymbol{V}^{(k)}$, for $1 \le k \le m$. Therefore, after the kth iteration, we have

$$\boldsymbol{V}^{(k+1)} = \boldsymbol{P}^{(k)} \dots \boldsymbol{P}^{(1)}\boldsymbol{V} = \begin{bmatrix} \boldsymbol{R}_{11} & \boldsymbol{R}_{12} \\ 0 & \tilde{\boldsymbol{V}}^{(k+1)} \end{bmatrix}, \tag{3.20}$$

where the first k rows and columns of matrix $\boldsymbol{V}$ have been already transformed.

The lower right block of matrix $\boldsymbol{V}$ which is not transformed yet after the kth iteration is denoted by $\tilde{\boldsymbol{V}}^{(k+1)}$. In Eq. 3.20, $\boldsymbol{P}^{(k)}$ has the form

$$\boldsymbol{P}^{(k)} = \begin{bmatrix} \boldsymbol{I}_{k-1} & 0 \\ 0 & \tilde{\boldsymbol{P}}^{(k)} \end{bmatrix}, \tag{3.21}$$

where $\tilde{\boldsymbol{P}}^{(k)}$ is a Householder transformation. After the mth iteration, the QR factorization is complete, and we have

$$\boldsymbol{V}^{(m+1)} = \boldsymbol{P}^{(m)} \dots \boldsymbol{P}^{(1)} \boldsymbol{V} = \begin{bmatrix} \boldsymbol{R} \\ 0 \end{bmatrix}. \tag{3.22}$$

By comparing Eq. 3.22 with Eq. 3.12, it becomes apparent that

$$\boldsymbol{P}^{(m)} \dots \boldsymbol{P}^{(1)} = \boldsymbol{Q}^T \quad \text{and} \quad \boldsymbol{P}^{(1)} \dots \boldsymbol{P}^{(m)} = \boldsymbol{Q}. \tag{3.23}$$

This is the basis for the so-called *factored-form* representation of $\boldsymbol{Q}$ [127, 128]. Instead of explicitly forming and storing $\boldsymbol{Q}_1$ for the calculation of $\boldsymbol{Q}_1^T \Delta \boldsymbol{r}$, as done in the *economy-size* QR factorization, one can apply $\boldsymbol{Q}^T$ to $\Delta \boldsymbol{r}$ in the *factored-form* after the factorization is completed, i.e.,

$$\boldsymbol{Q}^T \Delta \boldsymbol{r} = \boldsymbol{P}^{(m)} \dots \boldsymbol{P}^{(1)} \Delta \boldsymbol{r}. \tag{3.24}$$

In order to do so, the Householder vector $\boldsymbol{v}^{(k)}$ and the scalar $\gamma^{(k)}$ from all iterations of the factorization must be stored in order to recursively calculate $\boldsymbol{Q}^T \Delta \boldsymbol{r}$ in a post-factorization loop. Recall that in the kth factorization step, the elements below the main diagonal in the kth column of $\boldsymbol{V}$ are annihilated. If each Householder vector $\boldsymbol{v}^{(k)}$ is constructed such that $\boldsymbol{v}^{(k)}(1) = 1.0$, then $\boldsymbol{v}^{(k)}(1)$ does not need to be stored and the remaining elements of $\boldsymbol{v}^{(k)}$ can be stored in the empty space below the main diagonal in the corresponding column of $\boldsymbol{V}$. This algorithm is usually available in the state-of-the-art general-purpose linear algebra libraries. In the context of interface quasi-Newton methods for the coupled interface problems, the fact that matrix $\boldsymbol{Q}$ (or $\boldsymbol{Q}_1$) does not need to be explicitly created and stored is appreciated in [125] as well, where a parallel implementation of Householder QR factorization is used for solving the least squares problem. In [125], the Householder vectors $\boldsymbol{v}^{(k)}$ are stored but not constructed to have $\boldsymbol{v}^{(k)}(1) = 1.0$. Therefore, the Householder vectors can't be stored in the space below the main diagonal of $\boldsymbol{V}$. If care is not taken, the Householder vectors must be stored separately, which for matrices with large number of rows requires almost as much memory as matrix $\boldsymbol{V}$ itself. However, we underline that even storing all the Householder vectors is not strictly necessary in the application of interface quasi-Newton methods. The only use of $\boldsymbol{Q}^T$, in our application, is to calculate the product $\boldsymbol{Q}^T \Delta \boldsymbol{r}$. Furthermore, $\boldsymbol{Q}^T$ is the product of successive Householder matrices $\boldsymbol{P}^{(k)}$, which are available through the Householder vectors $\boldsymbol{v}^{(k)}$ in the factorization process. Therefore, it is possible to apply the Householder transformation in

every iteration of the factorization to the corresponding block of the vector $\Delta \boldsymbol{r}$ and by doing so, gradually calculate the product $\boldsymbol{Q}^T \Delta \boldsymbol{r}$ *on-the-fly* (in the factorization loop) [129]. Using this technique, not only the Householder matrix $\boldsymbol{P}$ and matrix $\boldsymbol{Q}$ (or $\boldsymbol{Q}_1$), but also the Householder vectors $\boldsymbol{v}^{(k)}$ and scalars $\gamma^{(k)}$ from all factorization iterations do not strictly need to be stored, and the post-factorization loop for recursively calculating $\boldsymbol{Q}^T \Delta \boldsymbol{r}$ in the *factored-form* is spared. The necessary steps for the least squares solution via Householder QR factorization and employing the *on-the-fly* calculation of $\boldsymbol{Q}^T \Delta \boldsymbol{r}$ are presented in Algorithm 1.

Algorithm 1 Solving the linear least squares problem $\min_{\boldsymbol{\alpha}} \|\Delta \boldsymbol{r} - \boldsymbol{V}\boldsymbol{\alpha}\|_2$ by Householder QR factorization, employing the *on-the-fly* calculation of $\boldsymbol{Q}^T \Delta \boldsymbol{r}$ in the factorization loop.

Input: $\boldsymbol{V} \in \mathbb{R}^{n \times m}, \Delta \boldsymbol{r} \in \mathbb{R}^n$
Output: $\boldsymbol{\alpha} \in \mathbb{R}^m$

for $k = 1, \ldots, m$ **do** ▷ QR factorization loop
  $\tilde{\boldsymbol{V}}^{(k)} = \boldsymbol{V}(k:n, k:m)$ ▷ The block of $\boldsymbol{V}$ that is not transformed yet
  $\left[\gamma^{(k)}, \boldsymbol{v}^{(k)}\right] = \text{house}\left(\tilde{\boldsymbol{V}}^{(k)}_{:,k}\right)$
  $\tilde{\boldsymbol{P}}^{(k)} \tilde{\boldsymbol{V}}^{(k)} = \tilde{\boldsymbol{V}}^{(k)} - \gamma^{(k)} \boldsymbol{v}^{(k)} \left(\boldsymbol{v}^{(k)^T} \tilde{\boldsymbol{V}}^{(k)}\right)$
  $\widetilde{\Delta \boldsymbol{r}} = \Delta \boldsymbol{r}(k:n)$ ▷ Trailing part of $\Delta \boldsymbol{r}$ that is not transformed yet
  $\tilde{\boldsymbol{P}}^{(k)} \widetilde{\Delta \boldsymbol{r}} = \widetilde{\Delta \boldsymbol{r}} - \gamma^{(k)} \left(\boldsymbol{v}^{(k)^T} \widetilde{\Delta \boldsymbol{r}}\right) \boldsymbol{v}^{(k)}$ ▷ Calculating $\boldsymbol{Q}^T \Delta \boldsymbol{r}$ *on-the-fly*
  if $k = m$ **then**
    $\boldsymbol{b} = \Delta \boldsymbol{r}$ ▷ Auxiliary variable $\boldsymbol{b}$ is used here only for clarity
  end if
end for
$\boldsymbol{b} = [\boldsymbol{b}_1, \boldsymbol{b}_2]^T$ ▷ Partition $\boldsymbol{b}$ after the mth row. note that $\boldsymbol{b}_1 = \boldsymbol{Q}_1^T \Delta \boldsymbol{r}$
$\boldsymbol{R} = \text{triu}\left(\boldsymbol{V}^{(m+1)}(:m, :m)\right)$ ▷ Upper triangular part of $\boldsymbol{V}^{(m+1)}$
$\boldsymbol{R}\boldsymbol{\alpha} = \boldsymbol{b}_1$ ▷ Solve the triangular system using back substitution

3.3.2 Solving the least squares problem via modified Gram-Schmidt orthogonalization

The Gram-Schmidt process produces the *thin* QR factorization $\boldsymbol{V} = \boldsymbol{Q}_1 \boldsymbol{R}$ by successively orthogonalizing the columns of $\boldsymbol{V}$. Given $\boldsymbol{V} \in \mathbb{R}^{n \times m}$, $\boldsymbol{Q}_1 \in \mathbb{R}^{n \times m}$ is orthogonal and $\boldsymbol{R} \in \mathbb{R}^{m \times m}$ is upper triangular. There are different versions of the Gram-Schmidt process. For a survey on different variants of the Gram-Schmidt process, see [130]. The column-oriented version of the *modified* Gram-Schmidt, presented in Algorithm 2, is studied here. The modified Gram-Schmidt algorithm is numerically more stable than the *classical* Gram-Schmidt. However the loss of orthogonality occurs in the modified Gram-Schmidt algorithm as well, albeit more gradually. It is shown that the loss of

orthogonality in the modified Gram-Schmidt algorithm is proportional to the condition number of matrix $\boldsymbol{V}$ [130][127, p. 61].

Algorithm 2 Modified Gram-Schmidt algorithm

Input: $\boldsymbol{V} \in \mathbb{R}^{n\times m}$
Output: $\boldsymbol{Q}_1 \in \mathbb{R}^{n\times m}$, $\boldsymbol{R} \in \mathbb{R}^{m\times m}$

$R_{1,1} = \|\boldsymbol{V}_{:,1}\|_2$, $\boldsymbol{Q}_1 = \dfrac{\boldsymbol{V}_{:,1}}{R_{1,1}}$
for $j = 2, \ldots, m$ **do**
  $\boldsymbol{v}^{(j)} = \boldsymbol{V}_{:,j}$
  for $i = 1, \ldots, j-1$ **do**
    $R_{i,j} = \boldsymbol{Q}_{:,i}^T \boldsymbol{v}^{(j)}$
    $\boldsymbol{v}^{(j)} = \boldsymbol{v}^{(j)} - R_{i,j}\boldsymbol{Q}_{:,i}$
  end for
  $R_{j,j} = \|\boldsymbol{v}^{(j)}\|_2$ $\boldsymbol{Q}_{:,j} = \dfrac{\boldsymbol{v}^{(j)}}{R_{j,j}}$
end for

In the previous works on the use of modified Gram-Schmidt orthogonalization in the context of interface quasi-Newton methods, e.g., [123, 117, 119], the product $\boldsymbol{Q}_1^T\Delta\boldsymbol{r}$ is calculated explicitly after the QR factorization $\boldsymbol{V} = \boldsymbol{Q}_1\boldsymbol{R}$ is carried out. This product is subsequently used as the right hand side of the triangular system $\boldsymbol{R}\boldsymbol{\alpha} = \boldsymbol{Q}_1^T\Delta\boldsymbol{r}$. This procedure does not always give an accurate solution. Here we point out to a more stable procedure. As noted, e.g., in [131, pp. 272–273] or [127, pp. 64–65], due to the loss of orthogonality which happens in modified Gram-Schmidt algorithm as well, the explicit calculation of $\boldsymbol{Q}_1^T\Delta\boldsymbol{r}$ followed by $\boldsymbol{R}\boldsymbol{\alpha} = \boldsymbol{Q}_1^T\Delta\boldsymbol{r}$ does not guarantee an accurate solution and should not be used. Instead, one sweep of the orthogonalization algorithm is carried out in order to make $\Delta\boldsymbol{r}$ orthogonal to the columns of the $\boldsymbol{Q}_1$ matrix. This can be symbolically written as

$$[\boldsymbol{V}, \Delta\boldsymbol{r}] = \left[\boldsymbol{Q}_1, \boldsymbol{q}'\right]\begin{bmatrix}\boldsymbol{R} & \boldsymbol{b}\\ 0 & 1\end{bmatrix}. \tag{3.25}$$

Then, $\boldsymbol{b} \in \mathbb{R}^m$ is used to solve the system $\boldsymbol{R}\boldsymbol{\alpha} = \boldsymbol{b}$, and $\boldsymbol{q}'$ is the residual of the least squares problem. The steps for solving the least square problem $\min_{\boldsymbol{\alpha}} \|\Delta\boldsymbol{r} - \boldsymbol{V}\boldsymbol{\alpha}\|_2$ via the modified Gram-Schmidt algorithm are presented in algorithm 3.

Based on the discussion in section 3.3.1 and the current section, it is concluded that the explicit calculation of $\boldsymbol{Q}^T\Delta\boldsymbol{r}$ or $\boldsymbol{Q}_1^T\Delta\boldsymbol{r}$ for solving the least squares solution via either Householder or (modified) Gram-Schmidt QR factorization should be avoided, by reason of efficiency and stability of the algorithms.

Algorithm 3 Solving the least squares problem $\min_{\boldsymbol{\alpha}} \|\Delta \boldsymbol{r} - \boldsymbol{V}\boldsymbol{\alpha}\|_2$ via the modified Gram-Schmidt algorithm

Input: $\boldsymbol{V} \in \mathbb{R}^{n \times m}, \Delta \boldsymbol{r} \in \mathbb{R}^n$
Output: $\boldsymbol{\alpha} \in \mathbb{R}^m$

1: $[\boldsymbol{Q}_1, \boldsymbol{R}]$ = modified Gram-Schmidt$(\boldsymbol{V})$ ▷ Calculate the QR factorization of $\boldsymbol{V}$ using algorithm 2
2: $\Delta \boldsymbol{r}^{(1)} = \Delta \boldsymbol{r}$ ▷ The intermediate variables $\Delta \boldsymbol{r}^{(j)}$ are used for clarity
3: allocate $\boldsymbol{b} = [b_1, \ldots, b_m]^T$
4: **for** $j = 1, \ldots, m$ **do**
5: $\quad b_j = \boldsymbol{Q}_{:,j}^T \Delta \boldsymbol{r}^{(j)}$
6: $\quad \Delta \boldsymbol{r}^{(j+1)} = \Delta \boldsymbol{r}^{(j)} - b_j \boldsymbol{Q}_{:,j}$
7: **end for**
8: $\boldsymbol{q}' = \Delta \boldsymbol{r}^{(m+1)}$ ▷ $\boldsymbol{q}'$ is the minimum norm residual vector
9: $\boldsymbol{R}\boldsymbol{\alpha} = \boldsymbol{b}$ ▷ Solve the triangular system using back substitution

3.3.3 Updating Householder QR factorization after removing a column

Matrices $\boldsymbol{V}$ and $\boldsymbol{W}$ in the interface quasi-Newton methods change between two consecutive coupling iterations by inserting a new column (from left) and possibly by removing a column (from right) if a sliding window scheme is used. In addition, a column might be removed from anywhere in the matrix due to filtering (see Section 3.4). For the sake of efficiency, it is desirable to update the existing QR factorization when a column is inserted or removed, rather than computing a new factorization from scratch.

The task of updating Gram-Schmidt QR factorization of a general matrix is first addressed in [132]. In the context of interface quasi-Newton methods, updating the (modified) Gram-Schmidt QR factorization by inserting or removing columns is dealt with in [123, 117, 119] . However, in the previous studies on quasi-Newton methods, where householder QR factorization in conjunction with filtering techniques is used, e.g., in [114, 92, 125], the QR factorization is discarded and subsequently recalculated every time a column is removed from matrix $\boldsymbol{V}$. Consequently, the computational cost is substantially increased, considering that removing columns might happen frequently in the filtering process. Householder QR factorization can be updated as well after a column is removed. However, a simple extension of the updating algorithms for the Gram-Schmidt QR factorization to the case of Householder QR factorization requires the explicit storage of the *full* $\boldsymbol{Q}$ matrix, which is highly undesirable for big matrices. Hereafter, it is shown how the Householder QR factorization can be updated after a column is removed[1] from matrix $\boldsymbol{V}$ by filtering, without explicitly creating and storing matrix $\boldsymbol{Q}$ or $\boldsymbol{Q}_1$ (see also [133]). The updating process is carried out through applying a series of

[1] Updating Householder QR factorization by inserting a column always requires the explicit creation of the $\boldsymbol{Q}$ matrix. Therefore, it is not suitable for our application.

Givens transformations. A Givens rotation in $\mathbb{R}^2$ is presented by the matrix

$$\boldsymbol{G} = \begin{bmatrix} c & s \\ -s & c \end{bmatrix}, \tag{3.26}$$

where c and s stand for $cos(\theta)$ and $sin(\theta)$, respectively. If a vector $\boldsymbol{w} = [w_1, w_2]^T$ is premultiplied by the transpose of $\boldsymbol{G}$, i.e., $\boldsymbol{G}^T\boldsymbol{w}$, it is rotated counterclockwise by θ. One could define θ such that the premultiplication by $\boldsymbol{G}^T$ would zero out the second element in $\boldsymbol{w}$, i.e., $\boldsymbol{G}^T\boldsymbol{w} = [w'_1, 0]^T$. In practice, explicit calculation and storage of θ or $\boldsymbol{G}$ is not needed and only c and s are calculated. See [128, p. 240] for a stable algorithm for the calculating c and s. Note that the same task can be equally well carried out using clockwise rotations through $\boldsymbol{Gw}$, as done in [131], or using plane reflectors, as done in [132].

In $\mathbb{R}^n$, the Givens rotation $\boldsymbol{G}_{(i,j)} \in \mathbb{R}^{n\times n}$ takes the form of a rank two modification of the identity matrix. $\boldsymbol{G}^T_{(i,j)}$ is determined such that when the vector $\boldsymbol{w} \in \mathbb{R}^n$ is premultiplied by $\boldsymbol{G}^T_{(i,j)}$, the ith element of $\boldsymbol{w}$ is changed and the jth element of $\boldsymbol{w}$ is set to zero. See, e.g., [128, p. 241] for details on how to apply Givens transformations to vectors and matrices without explicitly forming $\boldsymbol{G}^T_{(i,j)}$. Recall that the goal is to solve the least squares problem 3.11 by updating the QR factorization, after a column is removed. Given a Householder QR factorization for $\boldsymbol{V} \in \mathbb{R}^{n\times m}$

$$\boldsymbol{V} = \boldsymbol{Q}\begin{bmatrix} \boldsymbol{R} \\ 0 \end{bmatrix}, \tag{3.27}$$

one can update the factorization after removing a column from matrix $\boldsymbol{V}$ to arrive at the updated factorization

$$\tilde{\boldsymbol{V}} = \tilde{\boldsymbol{Q}}\begin{bmatrix} \tilde{\boldsymbol{R}} \\ 0 \end{bmatrix}, \tag{3.28}$$

where $\tilde{\boldsymbol{V}} \in \mathbb{R}^{n\times(m-1)}$, $\tilde{\boldsymbol{Q}} \in \mathbb{R}^{n\times n}$ and $\tilde{\boldsymbol{R}} \in \mathbb{R}^{(m-1)\times(m-1)}$. Assume that the kth column of $\boldsymbol{V}$, i.e., $\boldsymbol{V}_{:,k}$, is to be removed ($1 < k < m$). Denote by $\boldsymbol{V}_1$ and $\boldsymbol{V}_2$ the column blocks before and after $\boldsymbol{V}_{:,k}$, respectively. Then, one can write

$$[\boldsymbol{V}_1 \mid \boldsymbol{V}_{:,k} \mid \boldsymbol{V}_2] = \boldsymbol{Q}\begin{bmatrix} \boldsymbol{R}_1 & \boldsymbol{R}_{:,k} & \boldsymbol{R}_2 \\ 0 & 0 & 0 \end{bmatrix}. \tag{3.29}$$

By removing the kth column from matrices $\boldsymbol{V}$ and $\boldsymbol{R}$, we have

$$[\boldsymbol{V}_1 \mid \boldsymbol{V}_2] = \boldsymbol{Q}\begin{bmatrix} \boldsymbol{R}_1 & \boldsymbol{R}_2 \\ 0 & 0 \end{bmatrix} = \boldsymbol{Q}\begin{bmatrix} \check{\boldsymbol{R}} \\ 0 \end{bmatrix}. \tag{3.30}$$

Matrix $\check{\boldsymbol{R}} \in \mathbb{R}^{m\times(m-1)}$ is upper Hessenberg. In order to restore the triangular

form of the R-factor, a series of Givens transformations are applied in order to zero out the elements under the main diagonal in the columns $k, \dots, m-1$ of matrix $\check{\boldsymbol{R}}$:

$$\boldsymbol{G}_{(m-1,m)}^T \dots \boldsymbol{G}_{(k,k+1)}^T \begin{bmatrix} \check{\boldsymbol{R}} \\ 0 \end{bmatrix} = \boldsymbol{G}^T \begin{bmatrix} \check{\boldsymbol{R}} \\ 0 \end{bmatrix} = \begin{bmatrix} \tilde{\boldsymbol{R}} \\ 0 \end{bmatrix}. \tag{3.31}$$

Note that the mth row of $\check{\boldsymbol{R}}$ is zeroed out. The updated R-factor is $\tilde{\boldsymbol{R}} \in \mathbb{R}^{(m-1)\times(m-1)}$. The updated $\tilde{\boldsymbol{Q}}$ can be also calculated as

$$\tilde{\boldsymbol{Q}} = \boldsymbol{Q}\boldsymbol{G} = \boldsymbol{Q}\boldsymbol{G}_{(k,k+1)} \dots \boldsymbol{G}_{(m-1,m)}. \tag{3.32}$$

However, as stated previously, storing $\tilde{\boldsymbol{Q}}$ explicitly is undesirable. For the updated least squares problem, one needs to calculate

$$\tilde{\boldsymbol{b}} \coloneqq \tilde{\boldsymbol{Q}}^T \Delta \boldsymbol{r} = (\boldsymbol{Q}\boldsymbol{G})^T \Delta \boldsymbol{r} = \boldsymbol{G}^T \boldsymbol{Q}^T \Delta \boldsymbol{r} \tag{3.33}$$

By recalling that $\boldsymbol{b} = \boldsymbol{Q}^T \Delta \boldsymbol{r}$ has been already calculated, without the explicit generation of $\boldsymbol{Q}$ (see section 3.3.1), one can write

$$\tilde{\boldsymbol{b}} = \boldsymbol{G}^T \boldsymbol{b} = \boldsymbol{G}_{(m-1,m)}^T \dots \boldsymbol{G}_{(k,k+1)}^T \boldsymbol{b}. \tag{3.34}$$

Therefore $\tilde{\boldsymbol{b}}$ is calculated by applying the series of Givens rotations to $\boldsymbol{b}$ and neither $\boldsymbol{Q}$, nor $\tilde{\boldsymbol{Q}}$ are created explicitly. Finally, $\tilde{\boldsymbol{b}}_1 = \tilde{\boldsymbol{b}}(1 : m-1)$ is used to solve the triangular system

$$\tilde{\boldsymbol{R}}\boldsymbol{\alpha} = \tilde{\boldsymbol{b}}_1 \tag{3.35}$$

3.4 Filtering

It is not uncommon that some of the columns in matrix $\boldsymbol{V}$ become almost linearly dependent on the other columns. This could happen due to stagnation of the iterations or when the information from many previous time steps is reused, leading to matrix $\boldsymbol{V}$ having many columns, which increases the probability of occurrence of (almost) linearly dependent columns. The practice of removing some of the columns of matrix $\boldsymbol{V}$ for avoiding ill-conditioned systems or (in some cases) for improving the convergence rate in quasi-Newton algorithms has been referred to as filtering [115, 117, 119, 121, 125]. When using a QR factorization of $\boldsymbol{V}$, linearly dependent columns may lead to the occurrence of small diagonal elements in the R-factor. Therefore, one can try to detect small diagonal elements in the R-factor as a measure for linearly dependent columns, and subsequently remove the corresponding columns from matrix $\boldsymbol{V}$ (and $\boldsymbol{W}$). In this work, we focus on the filtering techniques based on detecting small diagonal entries of the R-factor in QR factorization. The first use of such a technique in the context of interface quasi-Newton methods was in [111], where an absolute criterion is used for detecting small

diagonal elements. This absolute criterion is problem dependent and does not take into account the scale of matrix $\boldsymbol{V}$. In [115], the relative cut-off criterion $|R_{i,i}| < \epsilon \cdot \|\boldsymbol{R}\|_F$ is used instead, where $\|\boldsymbol{R}\|_F$[2] is the Frobenius norm of matrix $\boldsymbol{R}$. Not much has been discussed in [115] or later publications regarding the choice of $\|\boldsymbol{R}\|_F$ in the relative criterion. We will elaborate more on this topic in section 3.6.

Note that filtering based on the diagonal entries of $\boldsymbol{R}$ might filter out some columns merely because they have smaller norms compared to the other columns (see the results in section 3.7 and also [115]). Therefore, another method for filtering is proposed in [115]. This method, later referred to as $QR2$ filtering in [117, 119], filters out the columns of $\boldsymbol{V}$ which lead to a small diagonal elements in the R-factor during the construction of the QR factorization via the modified Gram-Schmidt algorithm. In this filtering technique, the ratio of the norm of each orthogonalized vector to the norm of the corresponding column in $\boldsymbol{V}$ is monitored during the factorization process, and the column is removed from $\boldsymbol{V}$ if this ratio falls below a certain threshold. The filtering criterion in this technique can be also rewritten as [115]

$$|R_{i,i}| < \epsilon \cdot \|\boldsymbol{R}_{:,i}\|_2, \tag{3.36}$$

where $\boldsymbol{R}_{:,i}$ is the ith column of the R-factor. In the $QR2$ filtering technique, the QR factorization must be recalculated in every coupling iteration after a new column is appended to the left of matrix $\boldsymbol{V}$ [119, 117]. This can be considered as a drawback of this filtering technique since otherwise, the QR factorization via modified Gram-Schmidt algorithm could be updated rather than recalculated after a column is inserted. Moreover, the $QR2$ filtering technique, as presented in [115], is specific to modified Gram-Schmidt orthogonalization, and therefore, can not be used in conjunction with Householder QR factorization.

3.5 Column scaling

We propose to use the *column scaling* technique (sometimes called column weighting [128]), when solving the least squares problem in interface quasi-Newton methods for partitioned coupled simulations. For any positive diagonal matrix $\boldsymbol{D} \in \mathbb{R}^{m \times m}$, one can rewrite the least squares system 3.11 as ([128, p. 306][134, p. 185])

$$\min_{\boldsymbol{\alpha}} \|\Delta\boldsymbol{r} - \boldsymbol{V}\boldsymbol{\alpha}\|_2 = \min_{\boldsymbol{\alpha}} \left\|\Delta\boldsymbol{r} - \boldsymbol{V}\boldsymbol{D}^{-1}\boldsymbol{D}\boldsymbol{\alpha}\right\|_2. \tag{3.37}$$

Through a change of variables, we arrive at a new least squares problem

$$\min_{\bar{\boldsymbol{\alpha}}} \left\|\Delta\boldsymbol{r} - \bar{\boldsymbol{V}}\bar{\boldsymbol{\alpha}}\right\|_2, \tag{3.38}$$

[2] In [115], the Frobenius norm of $\boldsymbol{R}$ is denoted by $\|\boldsymbol{R}\|_2$. In this work, $\|\boldsymbol{R}\|_2$ refers to the spectral norm (matrix 2-norm).

where $\bar{\boldsymbol{V}} = \boldsymbol{V}\boldsymbol{D}^{-1}$ is said to be *scaled*. Note that since $\boldsymbol{D}$ is diagonal, no matrix-matrix or matrix-vector multiplication is carried out. Postmultiplying by $\boldsymbol{D}^{-1}$ scales each column of $\bar{\boldsymbol{V}}$ by the corresponding diagonal entry of $\boldsymbol{D}^{-1}$. After solving 3.38 for $\bar{\boldsymbol{\alpha}}$ and recalling that from 3.37 we have $\bar{\boldsymbol{\alpha}} = \boldsymbol{D}\boldsymbol{\alpha}$, the solution of the original least squares problem is determined by

$$\boldsymbol{\alpha} = \boldsymbol{D}^{-1}\bar{\boldsymbol{\alpha}}. \tag{3.39}$$

Note again that no explicit matrix-vector multiplication is carried out. One can choose matrix $\boldsymbol{D}$ according to the requirements of the application field. However, one particular choice of scaling has desirable properties for the present application. Let $\boldsymbol{D} = \operatorname{diag}(d_1, \ldots, d_m)$ be the scaling matrix. If the diagonal elements of $\boldsymbol{D}$ are chosen to be the 2-norms of the columns of $\boldsymbol{V}$, i.e., $d_j = \|\boldsymbol{V}_{:,j}\|_2$ for $1 \leq j \leq m$, then the columns of matrix $\bar{\boldsymbol{V}}$ have the same (unit) 2-norm $\left\|\bar{\boldsymbol{V}}_{:,j}\right\|_2 = 1$, for $1 \leq j \leq m$. Consequently, when filtering techniques are used, no column is removed from matrix $\bar{\boldsymbol{V}}$ merely because it has a small size compared to the other columns. In other words, the judgment about the linear dependence among the columns of $\boldsymbol{V}$ is not polluted by the disperse column norms of $\boldsymbol{V}$. This is important since usually the norms of the columns decrease as the convergence of the coupling iterations is approached. Those columns represent the state of the system more accurately. However they might get filtered out because of their small norm if no scaling is used. This is a major reason for the (sometimes prohibitively) slow convergence of the coupling iterations when a higher filtering tolerance is used and matrix $\boldsymbol{V}$ is not scaled (see section 3.7).

Another advantage of scaling the columns of $\boldsymbol{V}$ by their norms is that, matrix $\bar{\boldsymbol{V}}$ has a better conditioning than matrix $\boldsymbol{V}$. In [135], it is shown that choosing the scaling such that the columns of $\bar{\boldsymbol{V}}$ have equal 2-norm approximately minimizes the condition number of $\bar{\boldsymbol{V}}$. To be more precise, the condition number of $\bar{\boldsymbol{V}}$ will be no more than a factor of $\sqrt{m}$ away from the minimum obtainable value (see [135] and also [134, p. 187]). The steps for solving the least square problem 3.11 using column scaling are presented in algorithm 4.

Algorithm 4 Solving the linear least squares problem $\min_{\boldsymbol{\alpha}} \|\Delta\boldsymbol{r} - \boldsymbol{V}\boldsymbol{\alpha}\|_2$, incorporating column scaling

Input: $\boldsymbol{V} \in \mathbb{R}^{n \times m}, \Delta\boldsymbol{r} \in \mathbb{R}^n$
Output: $\boldsymbol{\alpha} \in \mathbb{R}^m$

1: $\boldsymbol{d} = \left[\|\boldsymbol{V}_{:,1}\|_2 \mid \ldots \mid \|\boldsymbol{V}_{:,m}\|_2\right]$ ▷ $\boldsymbol{D} = \operatorname{diag}(d_1, \ldots, d_m)$
2: $\bar{\boldsymbol{V}} = \left[\dfrac{\boldsymbol{V}_{:,1}}{d_1} \mid \ldots \mid \dfrac{\boldsymbol{V}_{:,m}}{d_m}\right]$ ▷ $\bar{\boldsymbol{V}} = \boldsymbol{V}\boldsymbol{D}^{-1}$
3: solve $\min_{\bar{\boldsymbol{\alpha}}} = \left\|\Delta\boldsymbol{r} - \bar{\boldsymbol{V}}\bar{\boldsymbol{\alpha}}\right\|_2$ using either Algorithm 1 or 3
4: $\boldsymbol{\alpha} = \left[\dfrac{\bar{\alpha}_1}{d_1} \mid \ldots \mid \dfrac{\bar{\alpha}_m}{d_m}\right]$ ▷ $\boldsymbol{\alpha} = \boldsymbol{D}^{-1}\bar{\boldsymbol{\alpha}}$

Column scaling can be used, equally well, in conjunction with Householder and (modified) Gram-Schmidt QR factorizations. Furthermore, when using column scaling, the QR factorization via (modified) Gram-Schmidt can still be updated as before by either inserting or removing columns. In an implementation of column scaling for quasi-Newton methods, both matrices $\boldsymbol{V}$ and $\bar{\boldsymbol{V}}$ do not need to be stored. One possibility is to store only $\bar{\boldsymbol{V}}$. Before a new column is appended to $\bar{\boldsymbol{V}}$, it is scaled by its 2-norm. The norms of the columns are also stored in a separate vector. There is another potentially more efficient way for incorporating column scaling in an implementation of interface quasi-Newton methods. The key is the invariance of QR factorization under column scaling [136][131, p. 272]. It means that if we have $\boldsymbol{V} = \boldsymbol{QR}$, the QR factorization of $\bar{\boldsymbol{V}} = \boldsymbol{V}\boldsymbol{D}^{-1}$ is $\boldsymbol{Q}(\boldsymbol{R}\boldsymbol{D}^{-1})$, for any positive diagonal matrix $\boldsymbol{D}$. Therefore, instead of scaling the columns of $\boldsymbol{V}$ and then carrying out the QR factorization of $\bar{\boldsymbol{V}}$, one can carry out the QR factorization of $\boldsymbol{V}$ and subsequently scale the columns of the R-factor to arrive at $\bar{\boldsymbol{R}} = \boldsymbol{R}\boldsymbol{D}^{-1}$. Doing so can save the computational cost associated with scaling the columns of $\boldsymbol{V}$ if the size of the interface coupled problem (therefore the number of rows in $\boldsymbol{V}$) is very big.

With the columns of matrices $\bar{\boldsymbol{V}}$ and $\bar{\boldsymbol{R}}$ having equal and unit norms, the diagonal entries of the (scaled) R-factor, $\bar{R}_{i,i}$ for $1 \leq i \leq m$, are better indicative of the possible presence of linear dependence among the columns of $\bar{\boldsymbol{V}}$. Therefore, one possible filtering criterion is

$$\left|\bar{R}_{i,i}\right| < \epsilon. \tag{3.40}$$

Considering that the column norms of a matrix are invariant under orthogonal transformations, assuming that no considerable loss of orthogonality occurs in the factorization process, Eq. 3.40 can be rewritten as

$$\left|\bar{R}_{i,i}\right| = \frac{|R_{i,i}|}{\|\boldsymbol{V}_{:,i}\|_2} = \frac{|R_{i,i}|}{\|\boldsymbol{R}_{:,i}\|_2} < \epsilon, \tag{3.41}$$

where $\boldsymbol{V}_{:,i}$ and $\boldsymbol{R}_{:,i}$ stand for the ith column of the (unscaled) matrices $\boldsymbol{V}$ and $\boldsymbol{R}$, respectively. This means that for the sake of filtering, the scaling can possibly be taken into account implicitly in the filtering criterion as opposed to explicitly carrying out column scaling on the matrices. On the other hand, there are a few drawbacks associated with such an approach. As discussed previously, matrix $\boldsymbol{R}$ (without scaling) has usually worse conditioning than matrix $\bar{\boldsymbol{R}}$ (with column scaling such that the columns have equal 2-norms). Furthermore, since scaling is not carried out explicitly on the columns of $\boldsymbol{R}$, the effect of scaling must be taken into account implicitly on all further possible operations on the $\boldsymbol{R}$ matrix. Such operations include filtering with possibly more elaborate filtering criteria and estimating the condition number (see section 3.6). Note that the filtering criterion 3.41 is the same criterion 3.36 used in the $QR2$ filtered modified Gram-Schmidt method in [115]. We can now interpret the $QR2$ filtering method as a technique that also takes

into account the column scaling implicitly in the filtering criterion. However, the QR factorization can not be updated and must be recalculated after a new column is appended to the left of matrix $\boldsymbol{V}$ using the $QR2$ filtering technique [119, 117].

3.6 Choice of filtering criterion

In this section we elaborate on the choices of filtering criteria (which directly affects the choice of the filtering tolerance) and show their relation with each other and to the singular values and condition number of matrix $\boldsymbol{V} \in \mathbb{R}^{n\times m}$ (or $\tilde{\boldsymbol{V}}$). Let

$$\boldsymbol{V} = \check{\boldsymbol{U}}\begin{bmatrix}\boldsymbol{\Sigma}\\ 0\end{bmatrix}\check{\boldsymbol{V}}^T \tag{3.42}$$

be the singular value decomposition (SVD) of matrix $\boldsymbol{V}$. $\check{\boldsymbol{U}} \in \mathbb{R}^{n\times n}$ and $\check{\boldsymbol{V}} \in \mathbb{R}^{m\times m}$ hold the left and right singular vectors, respectively. The diagonal matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{m\times m}$ holds the singular values, σ_i, of $\boldsymbol{V}$, which are sorted from biggest to smallest, i.e.,

$$\boldsymbol{\Sigma} = \operatorname{diag}(\sigma_1, \sigma_1, \ldots, \sigma_m), \quad \sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_m \geq 0. \tag{3.43}$$

The minimum and maximum singular values provide useful information regarding the conditioning of a matrix. The maximum and minimum singular values of $\boldsymbol{V}$ are designated by $\sigma_{\max}(\boldsymbol{V})$ and $\sigma_{\min}(\boldsymbol{V})$, respectively, i.e., $\sigma_{\max}(\boldsymbol{V}) \coloneqq \sigma_1$ and $\sigma_{\min}(\boldsymbol{V}) \coloneqq \sigma_m$.

We need a few matrix norm definitions (see, e.g., [127, 128] for details) in further discussions. The matrix 2-norm (also known as the spectral norm), subordinate to the vector 2-norm, is defined for matrix $\boldsymbol{V}$ as

$$\|\boldsymbol{V}\|_2 = \max_{\boldsymbol{y}\neq 0} \frac{\|\boldsymbol{V}\boldsymbol{y}\|_2}{\|\boldsymbol{y}\|_2} = \max_{\|\boldsymbol{y}\|_2=1} \|\boldsymbol{V}\boldsymbol{y}\|_2. \tag{3.44}$$

This matrix norm is not straightforward to calculate. The following two relations relate the 2-norm of a matrix and its pseudo inverse to the singular values of $\boldsymbol{V}$:

$$\|\boldsymbol{V}\|_2 = \sigma_{\max}(\boldsymbol{V}), \quad \left\|\boldsymbol{V}^\dagger\right\|_2 = \frac{1}{\sigma_{\min}(\boldsymbol{V})}. \tag{3.45}$$

A commonly used matrix norm is the Frobenius norm

$$\|\boldsymbol{V}\|_F = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{m} |V_{i,j}|^2}. \tag{3.46}$$

The 1-norm and ∞-norm of matrix $\boldsymbol{V}$ are defined as the maximum column

sum and row sum of $\boldsymbol{V}$ respectively, i.e.,

$$\|\boldsymbol{V}\|_1 = \max_{1\le j\le m}\sum_{i=1}^{n}|V_{i,j}|\,, \quad \|\boldsymbol{V}\|_\infty = \max_{1\le i\le n}\sum_{j=1}^{m}|V_{i,j}|\,. \tag{3.47}$$

The (2-norm) condition number of matrix $\boldsymbol{V}\in\mathbb{R}^{n\times m}$ is defined as

$$\kappa(\boldsymbol{V}) = \|\boldsymbol{V}\|_2\left\|\boldsymbol{V}^\dagger\right\|_2 = \frac{\sigma_{\max}(\boldsymbol{V})}{\sigma_{\min}(\boldsymbol{V})}. \tag{3.48}$$

It is convenient to work with the inverse of the condition number, called the reciprocal condition number, defined as

$$\kappa^{-1}(\boldsymbol{V}) = \frac{1}{\kappa(\boldsymbol{V})} = \frac{\sigma_{\min}(\boldsymbol{V})}{\sigma_{\max}(\boldsymbol{V})}. \tag{3.49}$$

Let $\boldsymbol{R}$ be the R-factor from the QR factorization of matrix $\boldsymbol{V}$ and assume that no substantial loss of orthogonality has happened during the factorization. It is known that the singular values and condition numbers of matrices $\boldsymbol{V}$ and $\boldsymbol{R}$ are equal (see, e.g., [137]).

$$\sigma_{\max}(\boldsymbol{V}) = \sigma_{\max}(\boldsymbol{R}), \quad \sigma_{\min}(\boldsymbol{V}) = \sigma_{\min}(\boldsymbol{R}), \quad \kappa(\boldsymbol{V}) = \kappa(\boldsymbol{R}) \tag{3.50}$$

We now focus on estimating the maximum and minimum singular values of matrix $\boldsymbol{R}$ based solely on the information available from the entries of $\boldsymbol{R}$. Those estimations will be subsequently used for estimating $\kappa^{-1}(\boldsymbol{R})$. The estimated values will be designated by an overhead hat, i.e., $\hat{\sigma}_{\max}(\boldsymbol{R}) \approx \sigma_{\max}(\boldsymbol{R})$ and $\hat{\sigma}_{\min}(\boldsymbol{R}) \approx \sigma_{\min}(\boldsymbol{R})$. We first deal with estimating the minimum singular value of $\boldsymbol{R}$. Let $d_{\min}(\boldsymbol{R})$ be the minimum diagonal element in $\boldsymbol{R}$, i.e.,

$$d_{\min}(\boldsymbol{R}) = \min_{i=1,\dots,m}|R_{i,i}|\,. \tag{3.51}$$

It can be shown that $d_{\min}(\boldsymbol{R})$ is bounded by the singular values of $\boldsymbol{R}$ (see [138]):

$$\sigma_{\min}(\boldsymbol{R}) \le d_{\min}(\boldsymbol{R}) \le \sigma_{\min}(\boldsymbol{R})\left(\frac{\sigma_{\max}(\boldsymbol{R})}{\sigma_{\min}(\boldsymbol{R})}\right)^{1-\frac{1}{m}}. \tag{3.52}$$

$d_{\min}(\boldsymbol{R})$ is an upper bound for $\sigma_{\min}(\boldsymbol{R})$. The bounds in 3.52 are not tight, and as discussed in [138], the probability of $d_{\min}(\boldsymbol{R})$ being much larger than $\sigma_{\min}(\boldsymbol{R})$ is not insignificant. However, solely based on the elements of $\boldsymbol{R}$ and without any further operations (e.g., column pivoting) there is no better approximation for $\sigma_{\min}(\boldsymbol{R})$ at hand. Therefore, as an estimation for the minimum singular value of $\boldsymbol{R}$, we have

$$\hat{\sigma}_{\min}(\boldsymbol{R}) = d_{\min}(\boldsymbol{R}). \tag{3.53}$$

There are more possibilities for estimating the maximum singular value of $\boldsymbol{R}$. Recall that $\sigma_{\max}(\boldsymbol{R})$ is equal to $\|\boldsymbol{R}\|_2$. We will look at matrix norm equivalences in order to find approximations for $\|\boldsymbol{R}\|_2$ (the relations among the 2-norm, 1-norm, ∞-norm and the Frobenius norm can be found, e.g., in [128]). An upper bound for the 2-norm of $\boldsymbol{R}$ is available through either of the two following relations: For any matrix $\boldsymbol{A} \in \mathbb{R}^{n\times m}$, it holds that

$$\|\boldsymbol{A}\|_2 \le \sqrt{\|\boldsymbol{A}\|_1 \|\boldsymbol{A}\|_\infty}, \quad \|\boldsymbol{A}\|_2 \le \|\boldsymbol{A}\|_F \le \sqrt{m}\|\boldsymbol{A}\|_2. \tag{3.54}$$

Therefore, two estimates (both are upper bounds) for the maximum singular value of $\boldsymbol{R}$ are

$$\hat{\sigma}_{\max}(\boldsymbol{R}) = \sqrt{\|\boldsymbol{R}\|_1 \|\boldsymbol{R}\|_\infty} \quad \text{and} \quad \hat{\sigma}_{\max}(\boldsymbol{R}) = \|\boldsymbol{R}\|_F. \tag{3.55}$$

For any matrix $\boldsymbol{A} \in \mathbb{R}^{n\times m}$, the following relation between the 2-norm and 1-norm of $\boldsymbol{A}$ holds:

$$\frac{1}{\sqrt{n}}\|\boldsymbol{A}\|_1 \le \|\boldsymbol{A}\|_2 \le \sqrt{m}\|\boldsymbol{A}\|_1. \tag{3.56}$$

Therefore, another upper bound estimate for the maximum singular value of $\boldsymbol{R}$ is

$$\hat{\sigma}_{\max}(\boldsymbol{R}) = \sqrt{m}\|\boldsymbol{R}\|_1, \tag{3.57}$$

which is expected to be a not very tight estimation. For comparison, $\|\boldsymbol{R}\|_1$ will be also used in the numerical examples as an estimation for the maximum singular value of $\boldsymbol{R}$. Another possibility for estimating the maximum singular value of $\boldsymbol{R}$ is based on the maximum column norm of $\boldsymbol{R}$, i.e., $\max \|\boldsymbol{R}_{:,j}\|_2$, for $1 \le j \le m$. The following relation holds between the maximum singular value and the maximum column norm of $\boldsymbol{R}$ (see [139]):

$$\max \|\boldsymbol{R}_{:,j}\|_2 \le \sigma_{\max}(\boldsymbol{R}) \le \sqrt{m} \cdot (\max \|\boldsymbol{R}_{:,j}\|_2), \quad 1 \le j \le m. \tag{3.58}$$

Therefore, we have the following two estimations for the maximum singular value of $\boldsymbol{R}$

$$\hat{\sigma}_{\max}(\boldsymbol{R}) = \max \|\boldsymbol{R}_{:,j}\|_2 \quad \text{and} \quad \hat{\sigma}_{\max}(\boldsymbol{R}) = \sqrt{m} \cdot (\max \|\boldsymbol{R}_{:,j}\|_2), \quad 1 \le j \le m. \tag{3.59}$$

Based on the fact that $\max \|\boldsymbol{R}_{:,j}\|_2$ and $\sqrt{m} \cdot (\max \|\boldsymbol{R}_{:,j}\|_2)$ are respectively lower and upper bounds for the maximum singular value of $\boldsymbol{R}$, a heuristic value between them might be a better estimation. The following estimation for the maximum singular value of $\boldsymbol{R}$ is suggested in [140]:

$$\hat{\sigma}_{\max}(\boldsymbol{R}) = \sqrt[3]{m} \cdot (\max \|\boldsymbol{R}_{:,j}\|_2), \quad 1 \le j \le m. \tag{3.60}$$

Recalling that $\hat{\kappa}^{-1}(\boldsymbol{R}) = \dfrac{\hat{\sigma}_{\min}(\boldsymbol{R})}{\hat{\sigma}_{\max}(\boldsymbol{R})}$, it is now possible to determine estimations

for the reciprocal condition number of $\boldsymbol{R}$ based on $\hat{\sigma}_{\min}(\boldsymbol{R}) = d_{\min}(\boldsymbol{R})$ and any $\hat{\sigma}_{\max}$ derived previously in this section. For example, if the Frobenius norm is used to estimate the maximum singular value of $\boldsymbol{R}$, we have

$$\hat{\kappa}^{-1}(\boldsymbol{R}) = \frac{d_{\min}(\boldsymbol{R})}{\|\boldsymbol{R}\|_F}. \tag{3.61}$$

Recall the filtering criterion $\frac{|R_{i,i}|}{\|\boldsymbol{R}\|_F} < \epsilon$ from section 3.4. Using this filtering criterion, the minimum diagonal element of $\boldsymbol{R}$ (i.e., $d_{\min}(\boldsymbol{R})$) is bigger than $\epsilon \cdot \|\boldsymbol{R}\|_F$ after the filtering is carried out. By comparing this filtering criterion to Eq. 3.61, it becomes apparent that the filtering tolerance, ϵ, is a threshold for the (estimated) minimum reciprocal condition number of $\boldsymbol{R}$ after the filtering is carried out. This relation is of importance since one gets an estimation of the reciprocal condition number of $\boldsymbol{R}$ without doing any further operations. Furthermore, by comparing how accurately any particular filtering criterion allows to estimate the reciprocal condition number, the confidence in a chosen filtering tolerance can be assessed. The correlation between the estimated and the actual reciprocal condition numbers differs from one coupled problem to another. Therefore, when studying the effect of filtering tolerances for a variety of numerical problems, a filtering criterion that allow a more accurate estimation of the reciprocal condition number is more desirable.

We have now a basis for comparing and choosing filtering criterion to be used in interface quasi-Newton methods, using either unscaled matrix $\boldsymbol{R}$ or scaled matrix $\bar{\boldsymbol{R}}$ (see section 3.5). Based on the correlation with estimation for the reciprocal condition number, different filtering criteria are defined and listed in Table 3.1. These filtering criteria will be compared in numerical test cases in section 3.7.

Table 3.1: Different filtering criteria compared in the numerical examples. $R_{i,i}$, $1 \le i \le m$, are the diagonal entries of $\boldsymbol{R} \in \mathbb{R}^{m \times m}$. $\max\|\boldsymbol{R}_{:,j}\|_2$ for, $1 \le i \le m$, represents the maximum column norm of $\boldsymbol{R} \in \mathbb{R}^{m \times m}$. If column scaling is carried out, $\boldsymbol{R}$ is replaced by $\bar{\boldsymbol{R}}$, and some of the relations are simplified.

Criterion 1	Criterion 2	Criterion 3	Criterion 4
$\frac{\lvert R_{i,i}\rvert}{\sqrt{m}\lVert\boldsymbol{R}\rVert_1} < \epsilon$	$\frac{\lvert R_{i,i}\rvert}{\sqrt{\lVert\boldsymbol{R}\rVert_1\lVert\boldsymbol{R}\rVert_\infty}} < \epsilon$	$\frac{\lvert R_{i,i}\rvert}{\sqrt{m}\cdot(\max\lVert\boldsymbol{R}_{:,j}\rVert_2)} < \epsilon$	$\frac{\lvert R_{i,i}\rvert}{\lVert\boldsymbol{R}\rVert_F} < \epsilon$
Criterion 5	**Criterion 6**	**Criterion 7**	
$\frac{\lvert R_{i,i}\rvert}{\sqrt[3]{m}\cdot(\max\lVert\boldsymbol{R}_{:,j}\rVert_2)} < \epsilon$	$\frac{\lvert R_{i,i}\rvert}{\lVert\boldsymbol{R}\rVert_1} < \epsilon$	$\frac{\lvert R_{i,i}\rvert}{\max\lVert\boldsymbol{R}_{:,j}\rVert_2} < \epsilon$	

If column scaling is used such that the columns of matrix $\bar{\boldsymbol{R}}$ have equal and unit norms, $\max\left\|\bar{\boldsymbol{R}}_{:,j}\right\|_2$ is equal to unity. Therefore, some of the relations in Table 3.1 are simplified. Filtering criterion 7 simplifies to $\left|\bar{R}_{i,i}\right| < \epsilon$, and

implicitly comprises an approximation of the maximum singular value of $\bar{\boldsymbol{R}}$ equal to unity. This is true when column scaling is carried out explicitly, as well as when column scaling is taken into account implicitly in the filtering criterion, as done in $QR2$ filtering method introduced in [115]. Note that the $QR2$ filtering method was interpreted in section 3.5 as a method that also implicitly takes into account column scaling in the filtering criterion $\left|\bar{R}_{i,i}\right| < \epsilon$. With the columns of matrix $\bar{\boldsymbol{R}}$ having equal and unit norms, it is straightforward to show that $\|\bar{\boldsymbol{R}}\|_F = \sqrt{m}$, where m is the number of columns in $\bar{\boldsymbol{R}}$. On the other hand, since $\max \left\|\bar{\boldsymbol{R}}_{:,j}\right\|_2 = 1$, it is also true that $\sqrt{m} \cdot (\max \left\|\bar{\boldsymbol{R}}_{:,j}\right\|_2) = \sqrt{m}$. Therefore, filtering criteria 3 and 4, in Table 3.1, are equal when column scaling is used. Furthermore, when using those filtering criteria, no further operations need to be carried out to calculate the (Frobenius or maximum column) norms. This is another advantage of applying column scaling.

3.7 Results and discussion

All the simulation in this section are carried out in the Kratos Multiphysics[3] open-source framework [141, 142]. The least squares problem entailed in the IQN-ILS method is solved using Householder QR factorization. A linear predictor is used in all simulations for predicting the displacements of the nodes on the FSI interface in all simulations. A few different convergence criteria are used for evaluating the convergence status of coupling iterations at each time step. The first criterion is an absolute one, based on the magnitude of the interface residual:

$$\frac{\|\boldsymbol{r}^{(l)}\|_2}{\sqrt{n}} < \varepsilon_{abs}^{d}, \tag{3.62}$$

where $\boldsymbol{r}^{(l)} = \boldsymbol{\mathcal{R}}\left(\boldsymbol{x}^{(l)}\right)$ is the interface (displacement) residual at the lth coupling iteration and n is the number of DOFs on the FSI interface. Instead of an absolute criterion, one can use a relative one

$$\frac{\|\boldsymbol{r}^{(l)}\|_2}{\|\tilde{\boldsymbol{x}}^{(l)}\|_2} < \varepsilon_{rel}^{d}. \tag{3.63}$$

In certain applications it might be necessary to make sure that the equilibrium of interface tractions is fully satisfied. For further discussions on this topic, see [117]. Therefore, in addition to the criterion 3.63, one can simultaneously use a relative convergence criterion based on the interface tractions:

$$\frac{\|\boldsymbol{f}^{(l)} - \boldsymbol{f}^{(l-1)}\|_2}{\|\boldsymbol{f}^{(l)}\|_2} < \varepsilon_{rel}^{f}, \tag{3.64}$$

[3] `https://github.com/KratosMultiphysics/Kratos`

where $\boldsymbol{f}^{(l)}$ and $\boldsymbol{f}^{(l-1)}$ are the interface traction vectors at the current and previous coupling iteration, respectively. In the numerical examples presented hereafter, the choice of convergence criteria among 3.62, 3.63 and 3.64 will be made such that a better comparison can be done with the results available in the literature.

3.7.1 Wave propagation in a 3D elastic tube

This test case simulates the pressure wave propagation through a flexible elastic tube. The benchmark problem appeared first in [143, 144] and is also described in [145, 146]. This benchmark problem has been widely used for the verification and comparison of quasi-Newton methods for the convergence acceleration of partitioned FSI simulations (see, e.g., [113, 120, 116, 115, 118, 119]). The problem consists of a straight elastic tube containing a fluid, with a length of 0.05 m, an inner diameter of 0.01 m, and a wall thickness of 0.001 m. Initially, the fluid and the structure (i.e., the tube's wall) are at rest. The two ends of the tube are fixed. The pressure at the outlet boundary is set to zero for the duration of the simulation. During the first 3×10^{-3} seconds of the simulation, an overpressure of 1333.2 Pa is applied at the inlet boundary and set to zero afterward. Consequently, a pressure wave propagates through the tube. The simulation is carried out for 10^{-2} s. The tube's wall is described by a linear elastic (Saint Venant–Kirchhof) material with a density of 1200 kg/m^3, Young's modulus of 3×10^5 N/m^2 and Poisson's ratio of 0.3. The fluid is assumed to be incompressible and has a density of 1000 kg/m^3 and a dynamic viscosity of 3×10^{-3} Pa s.

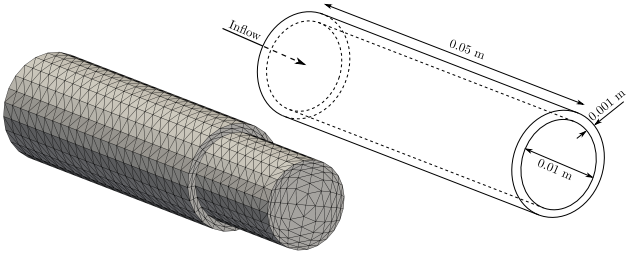


Figure 3.1: Wave propagation in a 3D elastic tube: computational domain's dimensions and the spatial discretization.

The fluid domain is discretized using 13824 tetrahedral finite elements, utilizing linear, equal-order interpolation for velocity and pressure (referred to as P1P1[4] element hereafter) and stabilized through the Variational Multiscales Method (VMS) [147, 148]. The structural domain is discretized using 9216

[4] To be specific, by P1P1 we designate the tetrahedral (in 3D) or triangular (in 2D) finite elements with linear interpolation for velocity and pressure.

linear tetrahedral elements based on a Total Lagrangian formulation. The number of degrees of freedom (DOF) in the fluid and structural domains are 11220 and 7128, respectively. The fluid and the structural domains have matching spatial discretizations on the FSI interface. The number of (displacement) DOFs on the FSI interface is 2376. The time integration schemes used is the 2nd order backward differentiation (BDF2) scheme for the fluid domain and Bossak (WBZ – α) scheme [149] for the structural domain. The time step size for both domains is 10^{-4} s. For assessing the convergence of the coupling iterations, relative convergence criteria for interface displacement residuals 3.63 and tractions 3.64 are used simultaneously with the convergence tolerances $\varepsilon^{d}_{rel} = 10^{-5}$ and $\varepsilon^{f}_{rel} = 10^{-4}$, respectively. Fig. 3.2 illustrates the pressure wave propagation in the elastic tube.

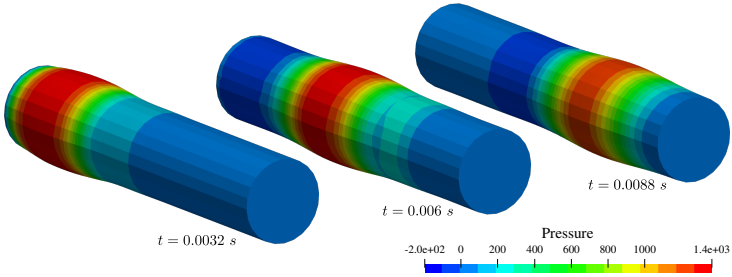


Figure 3.2: Wave propagation in a 3D elastic tube: pressure contours on the deformed fluid domain. The deformations are scaled by a factor of 5.

We first compare the different convergence criteria, presented in Table 3.1, with respect to how accurately they allow to estimate the minimum and the maximum singular values, as well as the reciprocal condition number of matrix $\boldsymbol{R}$ (or $\bar{\boldsymbol{R}}$). In order to have a fair comparison, a set of unique matrices needs to be used for all the filtering criteria. Therefore, a simulation is carried out using IQN-ILS with information from 30 previous time steps and with no filtering applied. A unique set of 406 $\boldsymbol{V}$ matrices used in the quasi-Newton method, from all the coupling iterations between the time steps 25 and 100, is generated and stored. For each of the filtering criteria, QR decomposition of all the matrices in the unique set is calculated, while column filtering is carried out using the same filtering tolerance $\epsilon = 10^{-5}$. The analysis is carried out on the matrices with and without applying column scaling. We are interested in knowing how the estimated values (denoted by the overhead hat symbol) associated with the filtering criteria, compares to the exact values. We monitor the ratio $\frac{\hat{\sigma}_{\max}(\boldsymbol{R})}{\sigma_{\max}(\boldsymbol{R})}$ for which a value of one is ideal. The ratio $\frac{\sigma_{\min}(\boldsymbol{R})}{d_{\min}(\boldsymbol{R})}$ quantifies how well $d_{\min}(\boldsymbol{R})$, the smallest diagonal entry of $\boldsymbol{R}$ (or $\bar{\boldsymbol{R}}$), estimates the exact minimum singular value of $\boldsymbol{R}$ (or $\bar{\boldsymbol{R}}$). This ratio admits a value between zero and one. A higher value for this ratio indicates a better

estimation. Finally, the ratio $\frac{\kappa^{-1}(\boldsymbol{R})}{\hat{\kappa}^{-1}(\boldsymbol{R})}$ indicates how the estimated reciprocal condition number compares to the exact one. A value of one for this ratio is ideal.

The box (candle) plots of the ratios $\frac{\hat{\sigma}_{\max}(\bar{\boldsymbol{R}})}{\sigma_{\max}(\bar{\boldsymbol{R}})}$ and $\frac{\sigma_{\min}(\bar{\boldsymbol{R}})}{d_{\min}(\bar{\boldsymbol{R}})}$, using column scaling, are depicted for different filtering criteria (see Table 3.1) in Figs. 3.3a and 3.3b, respectively.

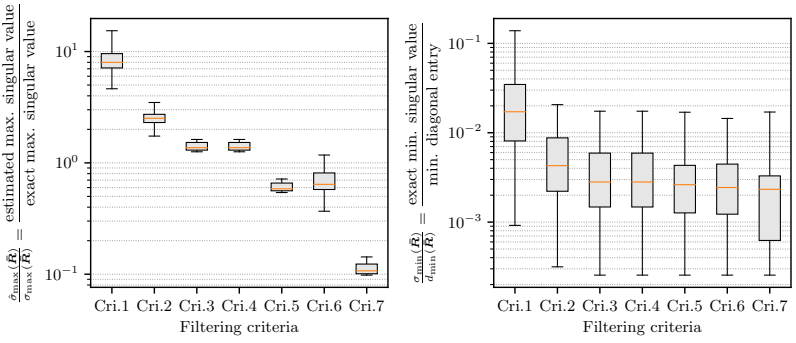
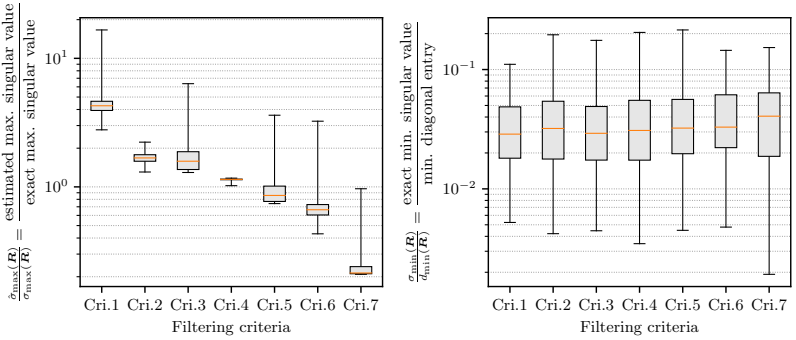


Figure 3.3: Wave propagation in a 3D elastic tube: comparison of different filtering criteria (listed in Table 3.1) based on the accuracy of the estimation for (a) the maximum singular value; (b) the minimum singular value. Column scaling is applied.

Fig. 3.3a reveals that filtering criteria 1–4 lead to overestimating, and filtering criteria 5–7 lead to mostly underestimating the maximum singular value of $\bar{\boldsymbol{R}}$. Filtering criteria 3 and 4 generate equal results, which also follows the expectation from the theoretical analysis in section 3.6. Concerning the accuracy of the estimation for the maximum singular value, filtering criteria 3 and 4 provide the best upper bound estimation. Filtering criterion 5 also provides a good estimation, albeit it underestimates the exact value. Filtering criteria 1 and 7 lead to the highest amount of overestimation and underestimation, respectively. The plot in Fig. 3.3b shows that the minimum diagonal element of $\bar{\boldsymbol{R}}$ overestimates the minimum singular value of $\bar{\boldsymbol{R}}$, sometimes by a factor of more than 10^3.

The above analysis can be carried out for the matrices without applying column scaling as well. The results are depicted in Fig. 3.4. The results from the estimation for the maximum singular value of $\boldsymbol{R}$, depicted in Fig. 3.4a, reveal that the filtering criteria 1, 3, 5, 6 and 7 do not provide very tight estimations. The reason is that these estimations rely on the size of the biggest column (in the sense of either $\|\boldsymbol{R}\|_1$ or $\max\|\boldsymbol{R}_{:,j}\|_2$) of the matrix. Without column scaling, an estimation of the maximum singular value that relies on the column with the biggest size may lead to less tight bounds.



(a) (b)

Figure 3.4: Wave propagation in a 3D elastic tube: comparison of different filtering criteria (listed in Table 3.1) based on the accuracy of the estimation for (a) the maximum singular value; (b) the minimum singular value. Column scaling is not applied.

Our primary interest in comparing different filtering criteria is to quantify how well a particular criterion allows to estimate the reciprocal condition number of the $\boldsymbol{R}$ (or $\bar{\boldsymbol{R}}$) matrix. This comparison is presented in Fig. 3.5. The first point to note is that all of the filtering criteria lead to overestimating the reciprocal condition number. However, the amount by which they overestimate the exact value differs from one another. Filtering criterion 1 allows a better estimation of the reciprocal condition number. Recall from Figs. 3.3 and 3.4 that filtering criterion 1 leads to the highest amount of overestimation of the maximum singular value. On the other hand, by using merely the minimum diagonal entry of the R-factor as an indication, the minimum singular value is overestimated (by a few orders of magnitude in some cases). Consequently, the overestimation of the maximum singular value by criterion 1, while not desirable as such, allows to compensate the overestimation of the minimum singular value to some degree and, therefore, leads to a better estimation of the reciprocal condition number. Filtering criterion 7 leads to the worst estimation of the reciprocal condition number.

We now focus on the effects of column scaling on the performance of the quasi-Newton method in the present example. It is worth comparing the number of columns removed by filtering with and without column scaling. In Fig. 3.6, the number of matrix columns before filtering and after filtering with and without column scaling are plotted. Filtering criterion 4 (based on the Frobenius norm) and filtering tolerance $\epsilon = 10^{-5}$ are used for both cases. From Fig. 3.6, it is evident that considerably more columns are removed from the matrix when no column scaling is applied. This behavior is observed across many numerical examples. In order to better study the origin of this behavior, it is constructive to look more carefully at what happens during

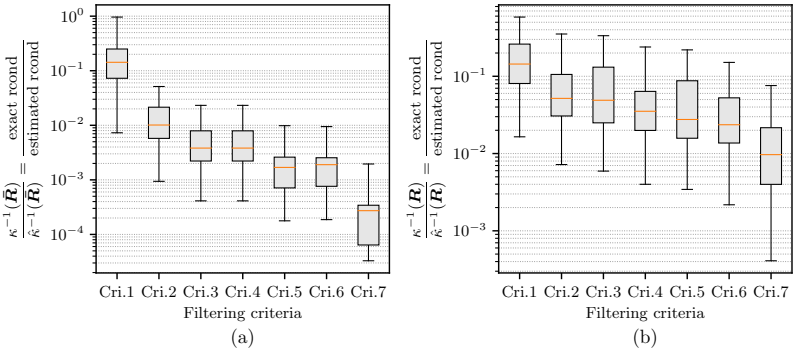


Figure 3.5: Wave propagation in a 3D elastic tube: comparison of different filtering criteria (listed in Table 3.1) based on the accuracy of the estimation for the reciprocal condition number of the matrices (a) together with column scaling; (b) without column scaling.

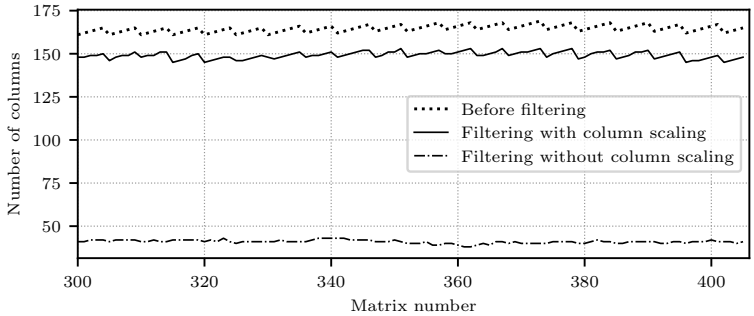


Figure 3.6: Wave propagation in a 3D elastic tube: the number of columns before and after filtering with and without applying column scaling. The last 106 matrices (out of 406 generated matrices) are used here.

the filtering process when column scaling is carried out compared to when it is not. Fig. 3.7 plots the norms of the columns in matrix $\boldsymbol{V}$, together with the diagonal entries of $\boldsymbol{R}$ and $\bar{\boldsymbol{R}}$ resulting from QR factorization with and without column scaling at a certain coupling iteration. In matrix $\boldsymbol{V}$, The columns corresponding to the initial coupling iterations in each time step have frequently bigger norms than the other columns. As the coupled problem approaches convergence in each time step, the columns' norms are reduced. Since information from previous time steps is reused in IQN-ILS, the pattern of mostly decreasing column norms among the columns added in one time step, followed by an increase in the column norm when moving to the next time step is repeated. This pattern is observed in the column norms of matrix

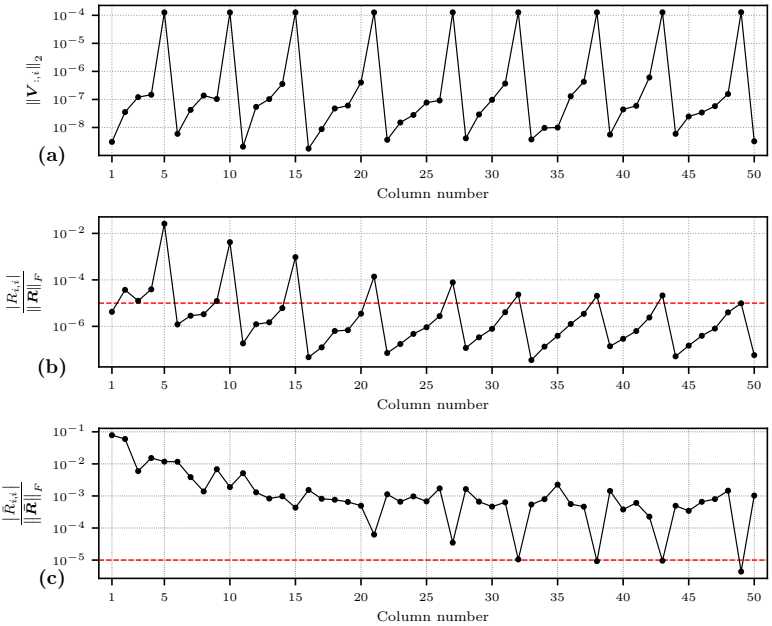


Figure 3.7: Wave propagation in a 3D elastic tube: (a) Column norms of matrix $\boldsymbol{V}$; (b) the absolute values of the diagonal entries of matrix $\boldsymbol{R}$, divided by the Frobenius norm of $\boldsymbol{R}$; (c) the absolute values of the diagonal entries of matrix $\bar{\boldsymbol{R}}$, divided by the Frobenius norm of $\bar{\boldsymbol{R}}$. The dashed horizontal lines in (b) and (c) represent the filtering tolerance $\epsilon = 10^{-5}$. For clarity, only the first 50 columns of the matrices are shown here.

$\boldsymbol{V}$ plotted in Fig. 3.7a. The absolute values of the diagonal entries in $\boldsymbol{R}$, presented in 3.7b, follow a similar pattern as the column norms of matrix $\boldsymbol{V}$ (and $\boldsymbol{R}$). The dependence of the diagonal entries of $\boldsymbol{R}$ on the disperse column norms adversely affects the decision about the presence of linear dependence among the columns of $\boldsymbol{R}$. In Fig. 3.7b, the diagonal entries are scaled by the Frobenius norm of $\boldsymbol{R}$, and a horizontal dashed line, representing the filtering tolerance $\epsilon = 10^{-5}$, is depicted as well. It is evident that, due to the small norm of their corresponding columns, the majority of data points in 3.7b (including the one from the most recent iteration) fall below the filtering tolerance and, therefore, are filtered out. Furthermore, the columns from the final iterations of each time step frequently produce smaller diagonal entries and end up being filtered out first. However, these columns may better represent the system at each time step since they are from iterations closer to the convergence. Filtering out those columns, particularly in the current time step, might be detrimental to the convergence rate of the quasi-Newton method. When column scaling is applied such that the columns have equal

norms, the diagonal entries of the $\bar{\boldsymbol{R}}$ matrix are not affected by the disperse norms of the columns. Therefore, the decision about the linear dependency among the columns of $\bar{\boldsymbol{V}}$ based on the diagonal entries of $\bar{\boldsymbol{R}}$ is more reliable. As shown in Fig. 3.7c, the diagonal entries of $\bar{\boldsymbol{R}}$ do not follow the pattern of the column norms depicted in 3.7a.

As pointed out in section 3.5, column scaling often leads to better-conditioned matrices. The amount of improvement in the conditioning of the matrix varies from case to case and is usually dependent on the dispersion of the column norms in matrix $\boldsymbol{V}$. In Fig. 3.8, the reciprocal condition numbers of the matrices with and without column scaling are plotted. In the

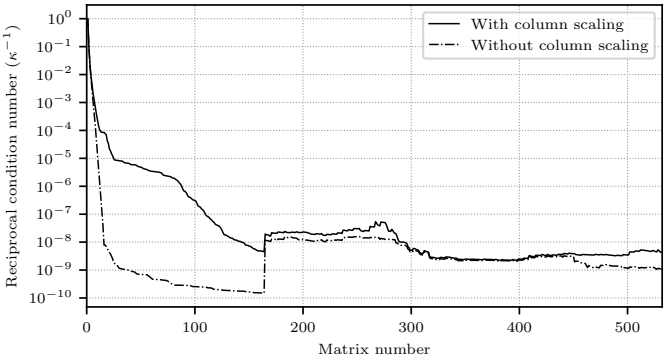


Figure 3.8: Wave propagation in a 3D elastic tube: reciprocal condition number of $\boldsymbol{V}$ matrices (without applying column scaling) and $\bar{\boldsymbol{V}}$ matrices (with applying column scaling). The matrices from all coupling iterations of all time steps are used for the comparison. No filtering is applied, and information from 30 previous time steps is reused.

present example, the improvement in the conditioning is more pronounced in the matrices from the early phase of the simulation, where more coupling iterations are performed per times step (because the information from many prior time steps is not fully available yet).

The average numbers of iterations per time step are presented in Table 3.2 for simulations with different combinations of input parameters. In order to allow comparison with the results from the literature (e.g., [115]), filtering criterion 4, based on the Frobenius norm, is used for all the simulations. The first point to notice in Table 3.2 is that for filtering tolerance of 10^{-4} and higher, the simulations in which column scaling is not applied do not converge before the maximum number of coupling iterations of 40 is reached (the corresponding entries in Table 3.2 are designated by "div."). The reason for that is excessive removal of the wrong columns when column scaling is not applied as already discussed (see Fig. 3.7 and the related discussion about it). The convergence is achieved in all the simulations in which column scaling is

Table 3.2: Wave propagation in a 3D elastic tube: average number of coupling iterations per time step.

time step history	column scaling	filtering tolerance (ϵ)							
		10^{-14}	10^{-8}	10^{-7}	10^{-6}	10^{-5}	10^{-4}	10^{-3}	10^{-2}
0	true	13.12	13.12	13.12	13.12	13.12	13.12	13.12	13.13
	false	13.12	13.12	13.14	13.17	13.31	div.	div.	div.
10	true	6.74	6.73	6.77	6.70	6.77	6.75	6.75	7.25
	false	6.73	6.73	6.67	7.01	8.98	div.	div.	div.
20	true	6.24	6.33	6.32	6.24	6.41	6.43	6.32	6.43
	false	6.31	6.29	6.31	6.79	9.14	div.	div.	div.
30	true	6.32	6.33	6.32	6.35	6.32	6.32	6.12	6.06
	false	6.37	6.30	6.27	6.72	9.19	div.	div.	div.
40	true	6.21	6.21	6.22	6.33	6.33	6.35	6.02	5.92
	false	6.21	6.23	6.34	6.75	8.93	div.	div.	div.

applied. For the filtering tolerance of 10^{-5}, the simulations without column scaling do converge, albeit considerably slower than the simulations with lower filtering tolerances. From the results in Table 3.2 and the discussion presented earlier about the plots in Fig. 3.7, it can be inferred that without column scaling, the efficiency and robustness of IQN-ILS in conjunction with filtering is too sensitive to the choice of filtering tolerance in the present example. Moreover, without column scaling, it is not at all possible to explore higher values for filtering tolerance. On the other hand, when column scaling is applied, all simulations converge. Moreover, for any particular value of the time step history, the average number of iterations per time step does not change drastically with different filtering tolerances.

It is worth emphasizing that, while the primary purpose of filtering is to prevent ill-conditioning of the matrix in the least squares problem at hand, filtering sometimes leads to enhanced efficiency, i.e., a lower average number of iterations per time step. The reason being that sometimes by removing the columns that are not necessarily (almost) linearly dependent on the other columns but do not represent the interface coupled problem accurately (e.g., because they are from iterations far from the convergence), the remaining columns of $\tilde{\boldsymbol{V}}$ describe the status of the interface fixed point problem more accurately. In the current example, a higher number of previous time steps (e.g., 40) and filtering with a higher tolerance (e.g., 10^{-3} or 10^{-2}) together with column scaling leads to the lowest number of coupling iterations. A more detailed discussion on the optimal combination of time step history and filtering tolerance for IQN-ILS is beyond the scope of the present work. The emphasis in the present study lies on the importance of column scaling in enhancing the robustness and efficiency of quasi-Newton methods.

3.7.2 Flow-induced oscillation of a flexible beam in the wake of a square bluff body

This benchmark problem was introduced by Wall and Ramm [150, 151] (hereafter, we refer to this benchmark problem as Wall benchmark for the sake

of brevity). A flexible beam attached to a square bluff body is submerged in incompressible fluid flow. Due to the interaction with the wake of the square body, the flexible beam starts to oscillate. The rationale for choosing this example is to analyze the not fully harmonic motion of the flexible beam in the initial phase of the simulation. The geometry and the boundary conditions of the problem are illustrated in Fig. 3.9.

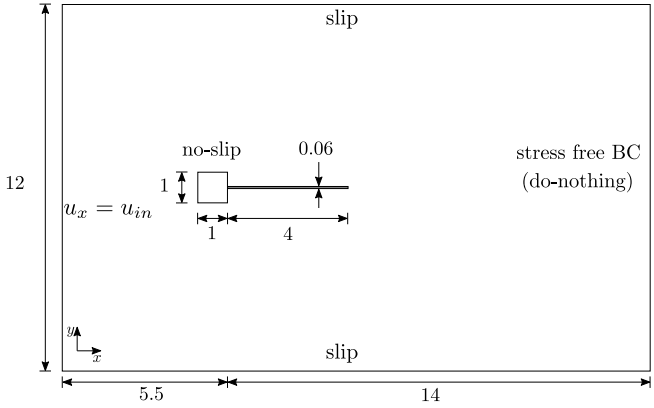


Figure 3.9: Wall benchmark: fluid domain's spatial discretization

The fluid domain is discretized using 5008 P1P1 finite elements, stabilized by VMS. The Fluid mesh is illustrated in Fig. 3.10. The structure is discretized using 20 nine-nodded quadratic Total Lagrangian elements. The number of DOFs in the fluid and structural domains are 10444 and 369, respectively. The fluid and the structural domains have non-matching spatial discretizations on the FSI interface. The nearest element interpolation method [152] is used for transferring the data between the two domains. The number of fluid domain's mesh displacement DOFs on the FSI interface (used in the quasi-Newton method) is 306.

The flow initialization follows the procedure in [72]. The fluid and the structure are initially at rest. At $t = 0$, the inflow velocity is applied instantaneously. The inflow velocity is $u_{in} = 51.3$. Zero-traction (do-nothing) boundary condition is applied at the outlet, and slip boundary conditions are applied at the top and bottom domain boundaries. The fluid is considered to be incompressible, with a density of 1.18×10^{-3} and a dynamic viscosity of 1.82×10^{-4}. For the structure, we assume plane stress and small strain elastic behavior with Young's modulus of 2.5×10^{6} and Poisson's ratio of 0.35. Bossak time integration scheme is used for both the fluid and the structural domains. The coupled simulation is carried out for 3 s with a time step size of 5×10^{-3} s for both domains. To assess the convergence of the coupling iterations, the absolute convergence criteria 3.62 with the absolute tolerances

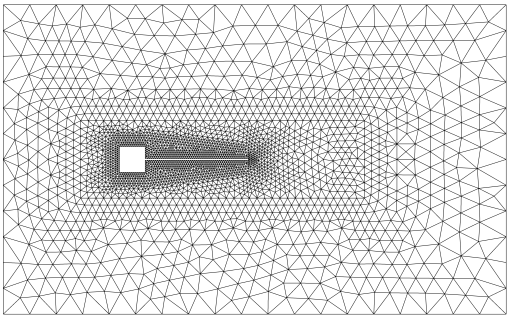


Figure 3.10: Wall benchmark: fluid domain's spatial discretization.

of $\varepsilon_{abs}^{d} = 10^{-7}$ is used. Fig. 3.11 illustrates the pressure contours around the flexible beam at time instance $t = 2.55$ s. The plot of the vertical displacement at the tip of the beam against the time is depicted in Fig. 3.12.

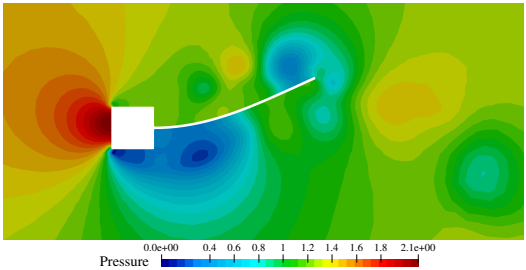


Figure 3.11: Wall benchmark: pressure contours around the flexible beam at $t = 2.55$ s.

The set of matrices used for the comparison of the filtering criteria is generated by carrying out a simulation using IQN-ILS with information from 10 previous time steps and without filtering. A unique set of 1950 matrices used in the quasi-Newton method, from coupling iterations between the time steps 20 and 600, is generated. For each filtering criterion, QR decomposition of all the $\boldsymbol{V}$ (or $\tilde{\boldsymbol{V}}$) matrices in the unique set is calculated, and column filtering is carried out using the same filtering tolerance $\epsilon = 10^{-5}$. The comparison of the filtering criteria with regard to the estimation for the extremum singular values is presented in Fig. 3.13 for the cases where column scaling is applied. The results generally follow the same trend as in the example in section 3.7.1. As indicated by the box plots in Fig. 3.13a, filtering criterion 1 gives rise to the highest amount of overestimation for the maximum singular value. Filtering criteria 3 and 4 (which are equal if column scaling is used) provide tight upper bounds for the maximum singular value of $\tilde{\boldsymbol{R}}$ in the present example. Criterion 5 leads, as well, to a close estimation of the maximum

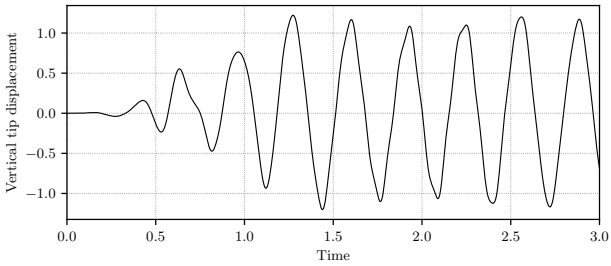


Figure 3.12: Wall benchmark: time history of the vertical displacement at the tip of the beam.

singular value, although it mostly underestimates the exact value. For the minimum singular value of $\bar{\boldsymbol{R}}$, filtering based on the minimum diagonal entry of $\bar{\boldsymbol{R}}$ results in overestimating the exact value. The box plots in Fig. 3.13b confirm this anticipated overestimation.

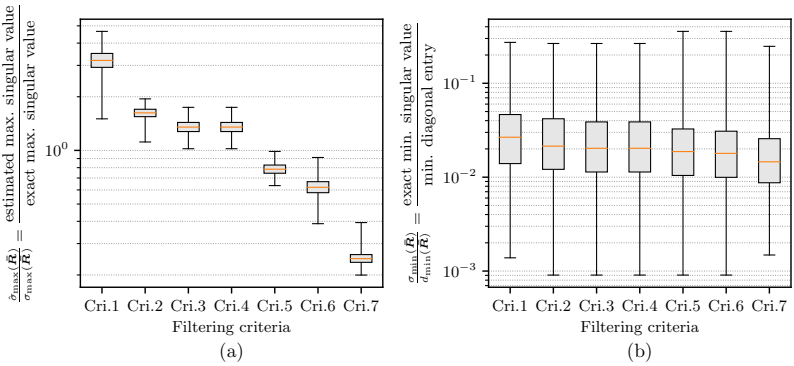


Figure 3.13: Wall benchmark: comparison of different filtering criteria (listed in Table 3.1) based on the accuracy of the estimation for (a) the maximum singular value; (b) the minimum singular value. Column scaling is applied.

For the matrices without column scaling, the estimations for the maximum and minimum singular values of $\boldsymbol{R}$ are presented in Figs. 3.14a and 3.14b, respectively. The plots in 3.14a suggest that without column scaling, the estimations for the maximum singular value of $\boldsymbol{R}$ based on the size of the biggest column in $\boldsymbol{R}$ result in less tight bounds. The same observation was made for the wave propagation in elastic tube example, presented in section 3.7.1, as well. The comparison of the filtering criteria with regard to the accuracy of the estimated reciprocal condition number of the R-factor is presented in Fig. 3.15. Compared to the other filtering criteria, filtering

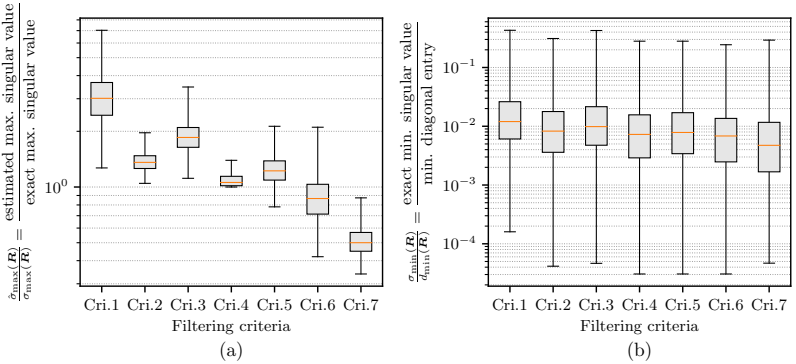


Figure 3.14: Wall benchmark: comparison of different filtering criteria (listed in Table 3.1) based on the accuracy of the estimation for (a) the maximum singular value; (b) the minimum singular value. Column scaling is not applied.

criterion 1 and 7 provide the best and worst estimations for the reciprocal condition number, respectively.

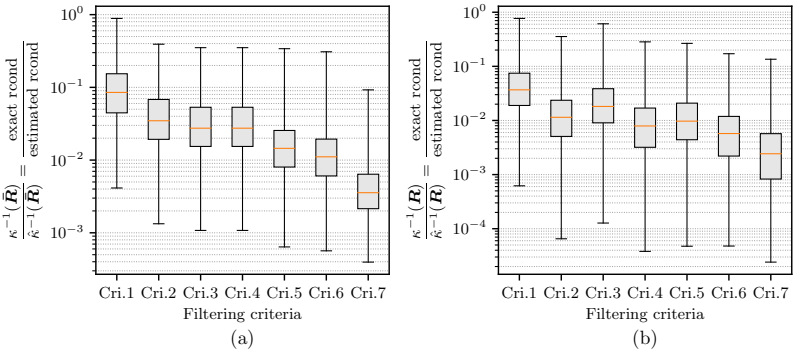


Figure 3.15: Wall benchmark: comparison of different filtering criteria (listed in Table 3.1) based on the accuracy of the estimation for the reciprocal condition number of the matrices (a) with column scaling; (b) without column scaling.

In Fig. 3.16, the reciprocal condition numbers of the matrices $\boldsymbol{V}$ and $\bar{\boldsymbol{V}}$ are plotted. By comparing the two graphs in Fig. 3.16, it is apparent that when column scaling is applied, the reciprocal condition number of the matrices are very often a few orders of magnitude higher than in the matrices without column scaling.

The average numbers of iterations per time step are presented in Table 3.3

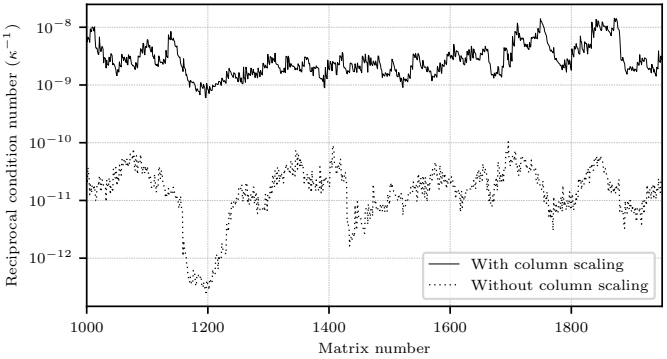


Figure 3.16: Wall benchmark: reciprocal condition number of $\boldsymbol{V}$ matrices (without applying column scaling) and $\bar{\boldsymbol{V}}$ matrices (with applying column scaling). For the sake of clarity, the last 950 available matrices (out of a total number of 1950 matrices) are used for the comparison. No filtering is applied.

for different combinations of input parameters. Filtering criterion 4, based on the Frobenius norm, is used for all the simulations. In contrast to the

Table 3.3: Wall benchmark: average number of coupling iterations per time step.

time step history	column scaling	filtering tolerance (ϵ)							
		10^{-14}	10^{-8}	10^{-7}	10^{-6}	10^{-5}	10^{-4}	10^{-3}	10^{-2}
0	true	4.74	4.74	4.74	4.74	4.74	4.74	4.74	4.74
	false	4.74	4.74	4.74	4.74	4.74	4.77	4.88	6.07
2	true	3.73	3.75	3.73	3.76	3.74	3.76	3.73	3.67
	false	3.73	3.73	3.72	3.74	3.74	3.83	4.03	4.50
5	true	4.02	4.01	4.02	4.02	3.89	3.82	3.78	3.70
	false	4.03	4.0	3.98	3.99	4.04	4.09	4.21	4.42
10	true	4.33	4.37	4.33	4.17	4.07	3.87	3.79	3.72
	false	4.35	4.35	4.33	4.31	4.26	4.30	4.49	4.46
15	true	4.58	4.57	4.58	4.43	4.13	3.87	3.79	3.72
	false	4.63	4.60	4.61	4.58	4.42	4.62	5.02	4.60

previous example in section 3.7.1, reusing information from a relatively low number of previous time steps leads to the lowest number of iterations per time step. The best results are achieved using 2 previous time steps. This observation may be explained by the fact that the behavior of the coupled problem in the current example is less predictable and more instantaneous in the initial phase of the simulation. Therefore, information from many previous time steps may not contribute to an accurate prediction at the current time step. By analyzing the results in Table 3.3 corresponding to different time step histories and lowest filtering tolerance (10^{-14}), it is confirmed that by

increasing the time step history in IQN-ILS the number of coupling iterations per time step increases. When column scaling is carried out, for any value (bigger than zero) of time step history presented in Table 3.3, the average number of coupling iterations per time step almost always decreases as the filtering tolerance increases. This trend is more pronounced for higher values of time step history. The same is not true when column scaling is not applied.

3.7.3 Flow-induced oscillation of a flexible beam in the wake of a cylinder

This benchmark problem was proposed by Turek and Hron in [89], and has been since used frequently in the literature. See, e.g., [113, 124, 119, 118, 153] for the application of quasi-Newton methods on this benchmark problem. There are three different configurations of the benchmark available. We choose the FSI3 configuration, which entails the strongest added-mass effect and the highest Reynolds number. This benchmark problem is conceptually similar to the example presented in section 3.7.2. Here, the fully harmonic motion of the flexible beam in the later stages of the simulation is analyzed, as opposed to the initial and not fully harmonic behavior studied in section 3.7.2. For the exact dimensions and geometry of the benchmark, see [89]. The fluid is considered to be incompressible, with density of 10^3 kg/m^3 and kinematic viscosity of 10^{-3} m^2/s. For the structure we assume plane stress and small strain elastic behavior, with a density of 10^3 kg/m^3, Young's modulus of 5.6×10^6 N/m^2 and Poisson's ratio of 0.4. A parabolic inflow profile with the mean flow value of 2 m/s is applied at the left boundary. In the first 2 seconds of the simulations, the inflow velocity is smoothly increased from zero to the nominal value. Do-nothing (zero traction) boundary condition is applied at the outlet. No-slip condition is applied at the top and bottom domain boundaries, as well as at the cylinder. The fluid domain is discretized using 30530 P1P1 finite elements stabilized by VMS. The Fluid mesh is illustrated in Fig. 3.17. The structure (the flexible beam) is discretized using 3290 linear quadrilateral Total Lagrangian elements. The number of DOFs in the fluid and structural domains are 63588 and 10620, respectively. The fluid and the structural domains have matching spatial discretizations on the FSI interface. The number of DOFs on the FSI interface (used in the quasi-Newton method) is 1455.

For the sake of the comparisons to follow in the remainder of the current section, a portion of the simulation between the times 19.6 s and 20 s is used. At t = 19.6 s, when the comparison starts, all the information from the previous time steps, needed for the IQN-ILS are available in each simulation. Bossak time integration scheme is used for both the fluid and the structural domains. The coupled simulation is carried out with the time step size of 10^{-3} s for both domains. For assessing the convergence of the coupling iterations, the absolute convergence criteria 3.63, with a tolerances of $\varepsilon^d_{rel} = 10^{-6}$ is used.

Fig. 3.18 illustrates the pressure contours in the whole fluid domain at

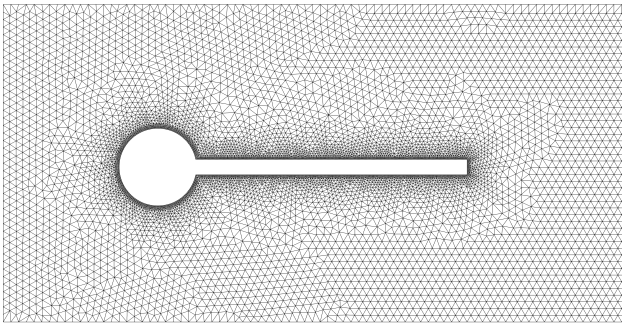


Figure 3.17: FSI3 benchmark: fluid domain's spatial discretization around the flexible beam. The grid extends uniformly downstream of the illustrated section.

the time instance of t = 19.743 s. The plots of the horizontal and vertical displacement at the tip of the beam against the time are depicted in Fig. 3.19.

Figure 3.18: FSI3 benchmark: pressure contours at t = 19.743 s.

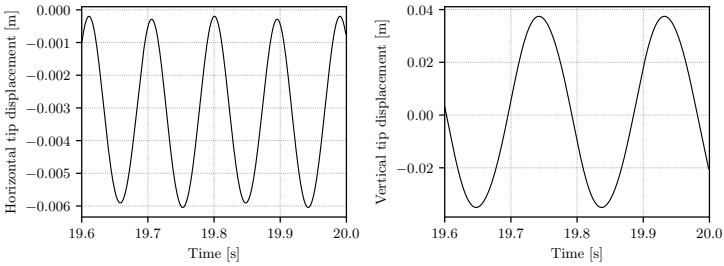


Figure 3.19: FSI3 benchmark: time history of the horizontal and vertical displacement at the tip of the beam.

The set of matrices used for comparing the different filtering criteria is generated by carrying out a simulation using IQN-ILS with information from 8 previous time steps and without filtering. A unique set of 1089 matrices, from coupling iterations of all 400 time steps, is generated. Sebsequently, filtering is carried out using the filtering tolerance $\epsilon = 10^{-5}$. The comparison of

the filtering criteria with regard to the estimation for the extremum singular values is presented in Figs. 3.20 and 3.21 for the matrices with and without columns scaling, respectively. The results generally follow the same trend as in the example in sections 3.7.1 and 3.7.2. Filtering criterion 4 provides a tight upper bound for the maximum singular value of $\bar{\boldsymbol{R}}$ and $\boldsymbol{R}$. Filtering criterion 5 leads, as well, to a good estimation of the maximum singular value, although it mostly underestimates the exact value. Filtering based on the minimum diagonal entry of the R-factor results in overestimating the exact minimum singular value of $\bar{\boldsymbol{R}}$ and $\boldsymbol{R}$. The comparison of the filtering criteria

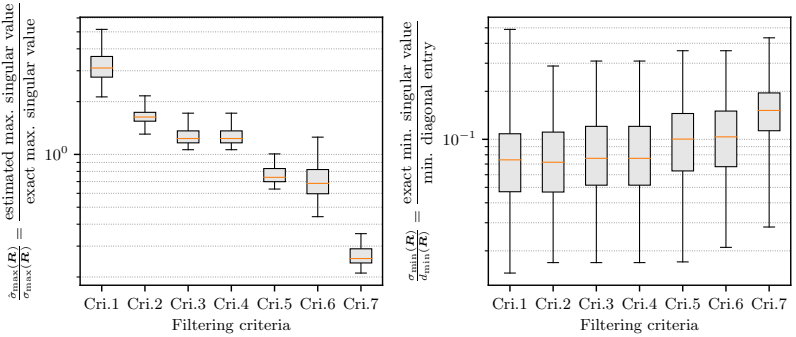


Figure 3.20: FSI3 benchmark: comparison of different filtering criteria (listed in Table 3.1) based on the accuracy of the estimation for (a) the maximum singular value; (b) the minimum singular value. Column scaling is applied.

with regard to the accuracy of the estimated reciprocal condition number of the R-factor is presented in Fig. 3.22. Filtering criteria 1 and 7 lead to the best and worst estimation for the reciprocal condition number, respectively. Nonetheless, criterion 1 provides an estimation for the reciprocal condition number that is slightly less than the exact value in some cases.

The influence of column scaling on the conditioning of the matrix used in the least squares solution is shown in Fig. 3.23. As in the previous two examples, the reciprocal condition numbers of the matrices with column scaling are higher than those of the matrices without column scaling.

The average numbers of iterations per time step are presented in Table 3.4 for different combinations of input parameters. Filtering criterion 4 (listed in Table 3.1), based on the Frobenius norm, is used for all the simulations. In contrast to the previous example in section 3.7.2, reusing information from a moderate number of previous time steps helps with decreasing the average number of coupling iterations per time step. The best results are achieved using information from 8 previous time steps and with little to no filtering. By analyzing the results in Table 3.4 corresponding to each value of time

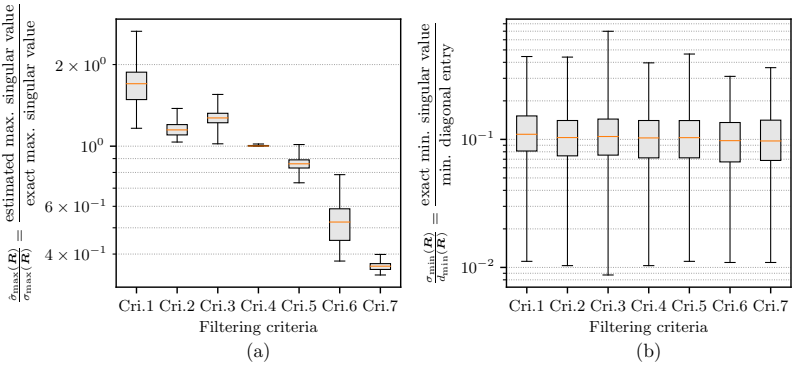


Figure 3.21: FSI3 benchmark: comparison of different filtering criteria (listed in Table 3.1) based on the accuracy of the estimation for (a) the maximum singular value; (b) the minimum singular value. Column scaling is not applied.

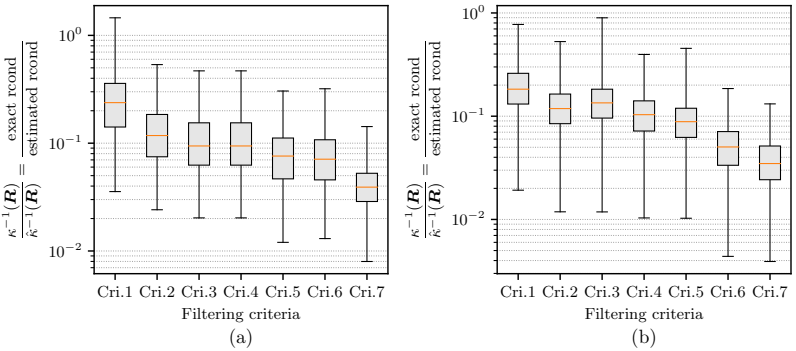


Figure 3.22: FSI3 benchmark: comparison of different filtering criteria (listed in Table 3.1) based on the accuracy of the estimation for the reciprocal condition number of the matrices (a) with column scaling; (b) without column scaling.

step history, it is revealed that the average number of iterations per time step mostly increases as higher values of filtering tolerance are used. However, the difference in performance using different filtering tolerances is not substantial when column scaling is used, except for the filtering tolerance of 10^{-2}. On the other hand, when column scaling is not applied, filtering with a tolerance higher than 10^{-5} is detrimental to the convergence of the coupling iterations. It can be deduced that column scaling enhances the robustness of filtering in IQN-ILS by reducing the sensitivity of the number of coupling iteration to the filtering tolerance.

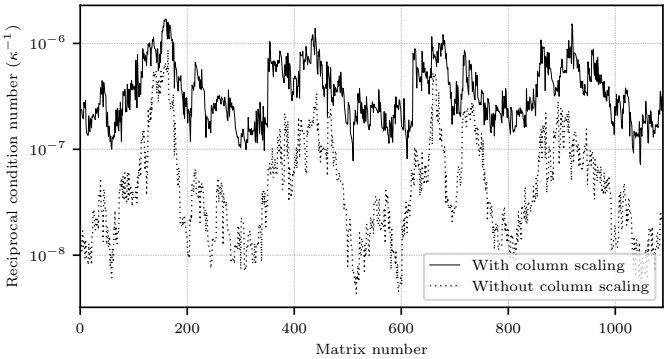


Figure 3.23: FSI3 benchmark: reciprocal condition number of $\boldsymbol{V}$ matrices (without applying column scaling) and $\tilde{\boldsymbol{V}}$ matrices (with applying column scaling). No filtering is applied.

Table 3.4: FSI3 benchmark: average number of coupling iterations per time step.

time step history	column scaling	filtering tolerance (ϵ)							
		10^{-14}	10^{-8}	10^{-7}	10^{-6}	10^{-5}	10^{-4}	10^{-3}	10^{-2}
0	true	10.81	10.81	10.81	10.81	10.81	10.81	10.82	10.82
	false	10.82	10.81	10.81	10.82	10.89	div.	div.	div.
5	true	3.82	3.82	3.81	3.79	3.82	4.16	4.42	5.05
	false	3.82	3.84	3.84	3.88	4.46	div.	div.	div.
8	true	3.72	3.71	3.73	3.71	3.74	3.96	4.07	4.90
	false	3.72	3.72	3.72	3.73	4.41	div.	div.	div.
10	true	3.81	3.81	3.82	3.81	3.86	4.06	4.06	4.83
	false	3.81	3.82	3.82	3.90	4.44	div.	div.	div.
15	true	4.54	4.52	4.62	4.61	4.71	4.52	4.08	4.93
	false	4.54	4.20	4.12	4.04	4.55	div.	div.	div.

Chapter 4

A strategy for automatic determination of time step history in interface quasi-Newton methods

4.1 Introduction

The convergence acceleration of partitioned FSI problems using interface quasi-Newton methods was reviewed in chapter 3, and techniques for enhancing the robustness and efficiency of those methods were proposed. The IQN-ILS method with information from previous time steps, together with filtering techniques and column scaling, prove to be very effective in accelerating the convergence of partitioned FSI problems. However, the determination of optimal parameters controlling the filtering and the number of previous time steps in IQN-ILS method requires a lot of trial and error for every new problem, as shown in section 3.7. An approach for alleviating the trial and error for the IQN-ILS method with filtering is proposed in [119], where a practically infinite number of previous time steps are reused together with filtering in order to remove the outdated or contradicting columns from the least square problem. In [119], the so-called $QR1$ and $QR2$ filtering techniques are used (see [115, 117, 119] or section 3.4), and it is concluded that the $QR1$ filtering is not robust enough for this purpose, and the $QR2$ filtering technique shows "fairly good" results but is relatively costly due to need for frequently recalculating the QR factorization. In the present work, a strategy for combining filtering

techniques and automatic determination of the number of previous time steps (time step history) in IQN-ILS method based on modifying the standard Householder QR factorization with column pivoting is proposed. The *basic solution* to the least squares problem via Householder QR factorization with column pivoting is used as the base algorithm for filtering the columns of the $\boldsymbol{V}$ matrix involved in the least squares solution. The base algorithm is modified using several techniques in order to find a good compromise between filtering outdated and contradicting columns in matrix $\boldsymbol{V}$. The proposed strategy is not parameter-free. However, it allows for combining the determination of parameters such that only one user-defined parameter is left to be chosen. Furthermore, the performance of the strategy is not very sensitive to the user-defined parameter, and a default value leads to quite good results for many scenarios, as will be shown using several numerical examples. The present chapter builds upon chapter 3. See section 3.2 for a concise presentation of the IQN-ILS method, section 3.3 for the introduction to the least squares problem in interface quasi-Newton methods, section 3.4 for filtering techniques, and section 3.5 for column scaling.

4.2 Solving the least squares problem using Householder QR factorization with column pivoting

In the following, the matrices are denoted by bold capital letters, e.g., $\boldsymbol{V}$. An element of matrix $\boldsymbol{V}$ is denoted by $V_{i,j}$, i.e., $V_{i,j} = \boldsymbol{V}(i,j)$. A column of the matrix, e.g., the jth column, is denoted by $\boldsymbol{V}_{:,j}$, i.e., $\boldsymbol{V}_{:,j} = \boldsymbol{V}(:,j)$.

Householder transformation (or reflection) $\boldsymbol{P}$ is a symmetric and orthogonal matrix of the form

$$\boldsymbol{P} = \boldsymbol{I} - \frac{1}{\gamma}\boldsymbol{v}\boldsymbol{v}^T, \quad \gamma = \frac{1}{2}\boldsymbol{v}^T\boldsymbol{v}. \tag{4.1}$$

The vector $\boldsymbol{v}$, called the Householder vector, can be defined such that when the Householder transformation $\boldsymbol{P}$ is applied to a vector $\boldsymbol{a}$, all elements of $\boldsymbol{a}$ after its first element are set to zero, i.e., $\boldsymbol{P}\boldsymbol{a} = \|\boldsymbol{a}\|_2 \boldsymbol{e}_1$, where $\boldsymbol{e}_1$ is the first unit vector. There are some practical considerations regarding the computation of the Householder vector $\boldsymbol{v}$ (see [127, sec. 2.3.1] or [128, sec. 5.1.3]). For the sake of brevity, it is sufficient here to assume that the function

$$[\gamma, \boldsymbol{v}] = \text{genHouseholder}(\boldsymbol{a}) \tag{4.2}$$

is available that, given a vector $\boldsymbol{a}$, generates the Householder vector.

Note that when applying a Householder transformation to a vector, matrix $\boldsymbol{P}$ is never explicitly created, i.e.,

$$\boldsymbol{P}\boldsymbol{a} = \left(\boldsymbol{I} - \frac{1}{\gamma}\boldsymbol{v}\boldsymbol{v}^T\right)\boldsymbol{a} = \boldsymbol{a} - \frac{1}{\gamma}\left(\boldsymbol{v}^T\boldsymbol{a}\right)\boldsymbol{v}. \tag{4.3}$$

Likewise, the premultiplication of a matrix $\boldsymbol{A}$ by $\boldsymbol{P}$ entails a matrix-vector multiplication and a rank one update to matrix $\boldsymbol{A}$:

$$\boldsymbol{P}\boldsymbol{A} = \left(\boldsymbol{I} - \frac{1}{\gamma}\boldsymbol{v}\boldsymbol{v}^T\right)\boldsymbol{A} = \boldsymbol{A} - \frac{1}{\gamma}\boldsymbol{v}\left(\boldsymbol{A}^T\boldsymbol{v}\right)^T. \tag{4.4}$$

We define the function

$$\text{applyHouseholder}\left(\gamma, \boldsymbol{v}, \boldsymbol{A}\right), \tag{4.5}$$

which applies the Householder transformation $\boldsymbol{P}$ formed by γ and $\boldsymbol{v}$ to the matrix $\boldsymbol{A}$.

An elementary permutation is mathematically represented by a matrix $\boldsymbol{\Pi}$ and, if multiplied by matrix $\boldsymbol{A}$, interchanges two columns of $\boldsymbol{A}$. When applying a permutation to matrix $\boldsymbol{A}$, the permutation matrix $\boldsymbol{\Pi}$ or the product $\boldsymbol{A}\boldsymbol{\Pi}$ are not explicitly calculated. Instead, two columns of $\boldsymbol{A}$ are interchanged, and the effect of the permutation matrix $\boldsymbol{\Pi}$ is concisely encapsulated in a permutation vector $\boldsymbol{piv}$. Permutation vector $\boldsymbol{piv}$ is an integer vector that holds the relation between the column indices of $\boldsymbol{V}\boldsymbol{\Pi}$ and $\boldsymbol{V}$. For example, $j = \boldsymbol{piv}(i)$ means that the ith column of $\boldsymbol{V}\boldsymbol{\Pi}$ was the jth column of $\boldsymbol{V}$.

The Householder QR factorization is calculated by transforming matrix $\boldsymbol{V}$ in iterations. Starting from $\boldsymbol{V}^{(1)} = \boldsymbol{V}$, the elements of a column under the main diagonal of $\boldsymbol{V}$ are eliminated in every iteration. When column pivoting is carried out, the first step in every iteration of the factorization is to choose the next pivot column among the columns of $\boldsymbol{V}$ that are not transformed yet. We will discuss the criterion for choosing the pivot column later. Subsequently, the chosen pivot column is permuted to the front of the trailing block of $\boldsymbol{V}$ that is not yet transformed. Finally, a Householder transformation is generated and applied. Assume that $k-1$ iterations have already been carried out, and we have

$$\boldsymbol{V}^{(k)} = \boldsymbol{P}^{(k-1)} \dots \boldsymbol{P}^{(1)}\ \boldsymbol{V}\ \boldsymbol{\Pi}^{(1)} \dots \boldsymbol{\Pi}^{(k-1)} = \begin{bmatrix} \boldsymbol{R}_{11} & \boldsymbol{R}_{12} \\ 0 & \tilde{\boldsymbol{V}}^{(k)} \end{bmatrix}, \tag{4.6}$$

where the first k rows and columns of matrix $\boldsymbol{V}$ have already been transformed. The lower right block of matrix $\boldsymbol{V}$ that is not transformed yet after $k-1$ iterations is denoted by $\tilde{\boldsymbol{V}}^{(k)}$. Let $\tilde{\boldsymbol{V}}^{(k)} = \left[\tilde{\boldsymbol{V}}^{(k)}_{:,k} \mid \dots \mid \tilde{\boldsymbol{V}}^{(k)}_{:,m}\right]$ be a column partitioning of $\tilde{\boldsymbol{V}}^{(k)}$, and let $s_j^{(k)}$, $k \le j \le m$, be the norm of the jth column of $\tilde{\boldsymbol{V}}^{(k)}$, i.e, $s_j^{(k)} = \|\tilde{\boldsymbol{V}}^{(k)}_{:,j}\|_2$, $k \le j \le m$. In the standard pivoting strategy, the pivot column is chosen to be the column with the largest norm among the columns of $\tilde{\boldsymbol{V}}^{(k)}$, i.e.,

$$\text{pivot column index} = p \mid s_p^{(k)} = \max\left(s_j^{(k)}\right), \quad k \le j \le m. \tag{4.7}$$

Subsequently, the pth and the kth columns are interchanged. The column norms $s_j^{(k)}$, $k \le j \le m$ do not need to be recalculated in every iteration. In practice, the column norms are updated using the values from the previous factorization iteration. The column norms are calculated once at the beginning of the factorization, i.e.,

$$s_j^{(1)} = \|\boldsymbol{V}_{:,j}\|_2, \quad 1 \le j \le m. \tag{4.8}$$

In the following iterations, the column norms are updated as [127, pp. 103–104][128, p. 277]

$$\left(s_j^{(k+1)}\right)^2 = \left(s_j^{(k)}\right)^2 - \left(R_{k,j}\right)^2, \quad 1 \le k \le m, \quad k+1 \le j \le m. \tag{4.9}$$

Updating column norms instead of recalculating them reduces the computational overhead associated with column pivoting. However, special care must be taken in order to prevent excessive numerical cancellation in the partial column norm updating process[154].

After the mth iteration, the QR factorization is completed:

$$\boldsymbol{V}^{(m+1)} = \boldsymbol{P}^{(m)} \dots \boldsymbol{P}^{(1)} \; \boldsymbol{V} \; \boldsymbol{\Pi}^{(1)} \dots \boldsymbol{\Pi}^{(m)} = \boldsymbol{Q}^T \boldsymbol{V} \boldsymbol{\Pi} = \begin{bmatrix} \boldsymbol{R} \\ 0 \end{bmatrix}. \tag{4.10}$$

Using the standard column pivoting, the diagonal elements of the R-factor form a non-increasing sequence [127, p. 103], i.e.,

$$R_{1,1} \ge R_{2,2} \ge \dots \ge R_{m,m}. \tag{4.11}$$

The QR factorization with standard column pivoting, first introduced in [155], is presented in Algorithm 5.

4.2.1 Basic solution via QR factorization with column pivoting

QR factorization with column pivoting is a useful technique for solving rank-deficient least squares problems. We are interested in solving the least squares problem

$$\min_{\boldsymbol{\alpha}=[\alpha_1,\dots,\alpha_m]^T} \|\Delta \boldsymbol{r} - \boldsymbol{V}\boldsymbol{\alpha}\|_2, \tag{4.12}$$

where $\boldsymbol{V} \in \mathbb{R}^{n \times m}$, $\Delta \boldsymbol{r} \in \mathbb{R}^n$ and $\boldsymbol{\alpha} \in \mathbb{R}^m$. Suppose $\boldsymbol{V}$ is exactly rank-deficient and has rank r ($r < m$).

The QR factorization of $\boldsymbol{V}$ using column pivoting leads to

$$\boldsymbol{Q}^T \boldsymbol{V} \boldsymbol{\Pi} = \begin{bmatrix} \boldsymbol{R}_{11} & \boldsymbol{R}_{12} \\ & 0 \end{bmatrix}, \tag{4.13}$$

Algorithm 5 Householder QR factorization with standard column pivoting

Input: $\boldsymbol{V} \in \mathbb{R}^{n\times m}$
Output: $--$

for $k = 1, \ldots, m$ **do** ▷ The QR factorization loop
  $\tilde{\boldsymbol{V}}^{(k)} = \left[\tilde{\boldsymbol{V}}^{(k)}_{:,k} \mid \ldots \mid \tilde{\boldsymbol{V}}^{(k)}_{:,m}\right] = \boldsymbol{V}(k:n, k:m)$ ▷ $\tilde{\boldsymbol{V}}^{(k)}$ is the block of $\boldsymbol{V}$ that is not transformed yet
  $s^{(k)}_j = \|\tilde{\boldsymbol{V}}^{(k)}_{:,j}\|_2,\ k \le j \le m$
  $p = \operatorname{argmax}\left(s^{(k)}_j\right), \quad k \le j \le m$
  $\tilde{\boldsymbol{V}}^{(k)}_{:,k} \leftrightarrows \tilde{\boldsymbol{V}}^{(k)}_{:,p}$
  $\boldsymbol{piv}(k) \leftrightarrows \boldsymbol{piv}(p)$
  $\left[\gamma^{(k)}, \boldsymbol{v}^{(k)}\right] = \text{genHouseholder}\left(\tilde{\boldsymbol{V}}^{(k)}_{:,k}\right)$
  $\text{applyHouseholder}\left(\gamma, \boldsymbol{v}, \tilde{\boldsymbol{V}}^{(k)}\right)$
  $\widetilde{\Delta \boldsymbol{r}} = \Delta \boldsymbol{r}(k:n)$ ▷ $\widetilde{\Delta \boldsymbol{r}}$ is the trailing part of $\Delta \boldsymbol{r}$ that is not transformed yet
  $\text{applyHouseholder}\left(\gamma, \boldsymbol{v}, \widetilde{\Delta \boldsymbol{r}}\right)$ ▷ Gradually calculating $\boldsymbol{Q}^T \Delta \boldsymbol{r}$ *on-the-fly*
  if $k = m$ **then**
    $\boldsymbol{b} = \Delta \boldsymbol{r}$ ▷ Auxiliary variable $\boldsymbol{b}$ is used here only for clarity
  end if
end for
$\boldsymbol{b} = [\boldsymbol{b}_1, \boldsymbol{b}_2]^T$ ▷ Partition $\boldsymbol{b}$ after the mth row. note that $\boldsymbol{b}_1 = \boldsymbol{Q}_1^T \Delta \boldsymbol{r}$
$\boldsymbol{R} = \text{triu}\left(\boldsymbol{V}^{(m+1)}(:m, :m)\right)$ ▷ $\boldsymbol{R}$ is the upper triangular square part of $\boldsymbol{A}$ after the factorization
$\boldsymbol{R}\boldsymbol{\alpha} = \boldsymbol{b}_1$ ▷ Solve the triangular system using back substitution

where $\boldsymbol{R}_{11} \in \mathbb{R}^{r\times r}$ and $\boldsymbol{R}_{12} \in \mathbb{R}^{r\times(m-r)}$. The elements in the lower right block of the R-factor are zero. Since the 2-norm is invariant under orthogonal transformations, it holds that

$$\|\Delta \boldsymbol{r} - \boldsymbol{V}\boldsymbol{\alpha}\|_2^2 = \left\|\boldsymbol{Q}^T \Delta \boldsymbol{r} - (\boldsymbol{Q}^T \boldsymbol{V}\boldsymbol{\Pi})(\boldsymbol{\Pi}^T \boldsymbol{\alpha})\right\|_2^2. \tag{4.14}$$

We define

$$\boldsymbol{b} \coloneqq \boldsymbol{Q}^T \Delta \boldsymbol{r} = \begin{bmatrix} \boldsymbol{Q}_1^T \Delta \boldsymbol{r} \\ \boldsymbol{Q}_2^T \Delta \boldsymbol{r} \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \end{bmatrix}, \tag{4.15}$$

and

$$\tilde{\boldsymbol{\alpha}} \coloneqq \boldsymbol{\Pi}^T \boldsymbol{\alpha}, \quad \tilde{\boldsymbol{\alpha}} = \begin{bmatrix} \tilde{\boldsymbol{\alpha}}_1 \\ \tilde{\boldsymbol{\alpha}}_2 \end{bmatrix}. \tag{4.16}$$

Using these definitions, Eqs. 4.13 and 4.14 can be rewritten as

$$\|\Delta \boldsymbol{r} - \boldsymbol{V}\boldsymbol{\alpha}\|_2^2 = \|(\boldsymbol{b}_1 - \boldsymbol{R}_{12}\tilde{\boldsymbol{\alpha}}_2) - \boldsymbol{R}_{11}\tilde{\boldsymbol{\alpha}}_1\|_2^2 + \|\boldsymbol{b}_2\|_2^2. \tag{4.17}$$

Using Eq. 4.16, 4.17, the general solution to the least squares problem 4.12 is

$$\boldsymbol{\alpha} = \boldsymbol{\Pi}\tilde{\boldsymbol{\alpha}} = \boldsymbol{\Pi}\begin{bmatrix} \boldsymbol{R}_{11}^{-1}(\boldsymbol{b}_1 - \boldsymbol{R}_{12}\tilde{\boldsymbol{\alpha}}_2) \\ \tilde{\boldsymbol{\alpha}}_2 \end{bmatrix}. \tag{4.18}$$

If $\tilde{\boldsymbol{\alpha}}_2$ is set to zero, we arrive at the *basic solution* [128, p. 292][127, p. 106]

$$\boldsymbol{\alpha}_b = \boldsymbol{\Pi}\begin{bmatrix} \boldsymbol{R}_{11}^{-1}\boldsymbol{b}_1 \\ 0 \end{bmatrix}. \tag{4.19}$$

As discussed previously, permutation matrix $\boldsymbol{\Pi}$ is not created explicitly in Eq. 4.19. Instead, $\tilde{\boldsymbol{\alpha}}_b = \boldsymbol{R}_{11}^{-1}\boldsymbol{b}_1$ is calculated first. Then, the elements of $\boldsymbol{\alpha}_b$ are determined from the elements of $\tilde{\boldsymbol{\alpha}}_b$ using the permutation information encapsulated in vector $\boldsymbol{piv}$:

$$\boldsymbol{\alpha}_b\,(\boldsymbol{piv}\,(i)) = \tilde{\boldsymbol{\alpha}}_b(i), \quad 1 \le i \le r. \tag{4.20}$$

Note that the *basic solution* is generally different from the *minimum norm solution*. The *minimum norm solution* can be calculated by applying a further orthogonal transformation on 4.13 in order to annihilate $\boldsymbol{R}_{12}$ [127, p. 111][128, p. 289].

In a numerical setup, matrix $\boldsymbol{V}$ might become almost (as opposed to exactly) rank-deficient. Therefore, the lower right block in 4.13 will have a small norm instead of being exactly zero. In the *basic solution*, a subset of columns formed by the first r columns of $\boldsymbol{R}$ is used for calculating the solution. The part of the solution corresponding to the trailing $m-r$ columns of $\boldsymbol{R}$ is set to zero. Therefore, the *basic solution* via QR factorization with column pivoting can be used as a *filtering* technique for interface quasi-Newton methods. After QR factorization with column pivoting is carried out, the *basic solution* is calculated and, subsequently, the columns of matrix $\boldsymbol{V}$ (and $\boldsymbol{W}$) corresponding to the trailing $m-r$ columns of $\boldsymbol{V}\boldsymbol{\Pi}$ are removed.

The remaining question is how to determine the numerical rank r of matrix $\boldsymbol{V}$. Recall from Eq. 4.11 that the diagonal elements of the R-factor from a QR factorization with column pivoting form a non-increasing sequence. The diagonal elements of the R-factor approximate the singular values of $\boldsymbol{V}$. Although there are analytically generated matrices for which this relation does not hold, this approximation usually works well in practice. The numerical ϵ-rank r of $\boldsymbol{R}$ is determined by

$$r = \max_{1\le j\le m}(j) \;\mid\; \frac{R_{j,j}}{R_{1,1}} \ge \epsilon, \tag{4.21}$$

where $\frac{R_{j,j}}{R_{1,1}}$ approximates the reciprocal condition number of the leading block of $\boldsymbol{R}$, and ϵ is a user-defined threshold for the minimum allowed reciprocal condition number. Note that we are not necessarily interested in determining

the correct numerical rank of $\boldsymbol{V}$. We use the *basic solution* via QR factorization with column pivoting as a column subset selection or filtering technique. The ϵ-rank determined by Eq. 4.21 is affected by the choice of ϵ and should be interpreted in the context of column subset selection.

In the following, we investigate the application of QR factorization with column pivoting on an example $\boldsymbol{V}$ matrix arising from the least squares problem in the IQN-ILS method. Matrix $\boldsymbol{V}$ has 100 columns and is taken from the numerical example presented in section 4.3 using IQN-ILS with reuse of information from 18 previous time steps. In all the following graphs, the discrete data points are connected via a line to present their sequence better. In Fig. 4.1, a comparison among the singular values and the diagonal elements of the R-factor from a QR factorization with and without column pivoting is presented. The diagonal elements of the R-factor from the QR factorization with column pivoting, presented in Fig. 4.1b, approximate the distribution of the singular values of $\boldsymbol{V}$, presented in Fig. 4.1a, very well. On the other hand, without column pivoting, the diagonal elements of the R-factor, presented in Fig. 4.1c, unsurprisingly do not approximate the distribution of the singular values well. When performing column pivoting, the last columns of the permuted matrix, corresponding to the smallest singular values, can be dropped in the *basic solution* of the least squares problem. It shows the advantage and the potential of performing column pivoting.

Besides the approximation of the singular values, there is another important factor to bear in mind when performing column pivoting, and that is the column ordering in matrix $\boldsymbol{R}$ (and $\boldsymbol{V\Pi}$). Matrix $\boldsymbol{V}$ is formed by appending data columns from consecutive iterations and time steps. Hence, the ordering of the columns is important. In the *basic solution* via QR factorization with column pivoting, the trailing columns of the permuted matrix are dropped if the reciprocal condition number falls below the user-defined threshold. Therefore, it is worth looking at the column ordering of the permuted matrix when column pivoting is carried out. The graph in Fig. 4.2a shows the situation where all the columns have the same column index after column pivoting as they had before (no column permutation has happened). Fig. 4.2b plots the column numbers after column pivoting against the column numbers in the original matrix. It is evident that the columns of matrix $\boldsymbol{V}$ are completely "shuffled" in the column pivoting process. This is a disadvantage in the present application, where we intend to use filtering via QR factorization with column pivoting as a means of automatically limiting the number of previous time steps used in the IQN-ILS method. Note that the excessive column interchanges are not due to the columns of $\boldsymbol{V}$ having disperse norms since the columns of $\boldsymbol{V}$ are scaled to have equal norms. The permutations are required for approximating the sequence of the singular values of $\boldsymbol{V}$.

To conclude, the generally good approximation of the singular values of matrix $\boldsymbol{V}$ via the standard column pivoting strategy comes at the price of completely changing the column ordering in matrix $\boldsymbol{V}$. Recall that our goal is to use column pivoting as a column subset selection or filtering technique.

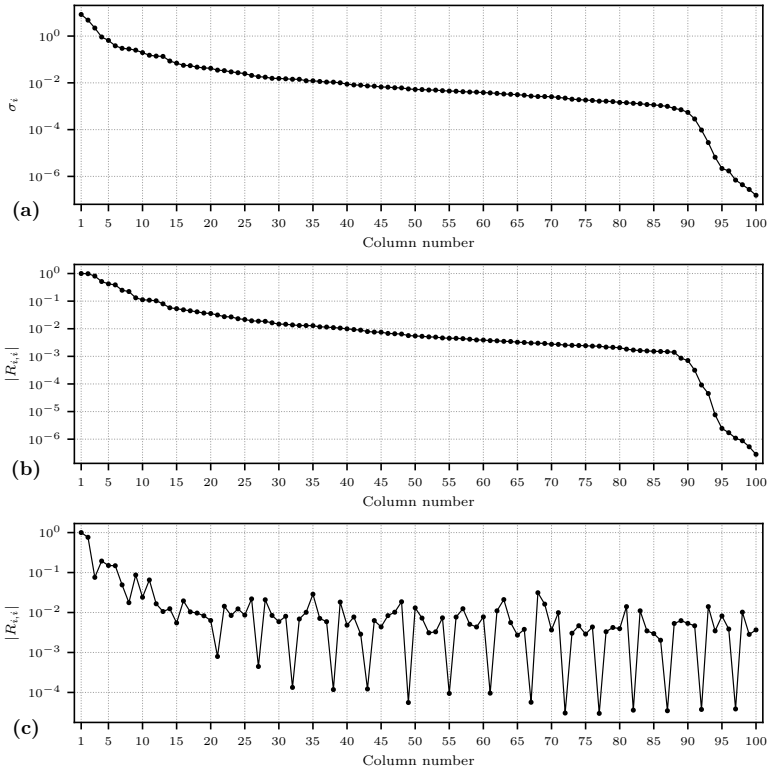


Figure 4.1: Comparison of SVD and QR factorization with and without column pivoting on an example matrix $\boldsymbol{V}$. (a) singular values from SVD; (b) diagonal elements of the R-factor from QR factorization with standard column pivoting; (c) diagonal elements of the R-factor from QR factorization without column pivoting

In that respect, it is enough to permute the much smaller pivot to the back of the matrix and have an approximation of the distribution of the singular values, while better maintaining the ordering of the columns. In other words, a compromise should be made between a very good approximation of the distribution of the singular values and the column ordering. In the following, we will introduce a few techniques for achieving that goal.

4.2.2 Closest column and threshold pivoting

In this section, the column pivoting strategy in Eq. 4.7 and line 4 of Algorithm 5 is modified in order to emphasize the original column ordering. The column pivoting strategy based on Eq. 4.7 strives in every iteration of the factorization

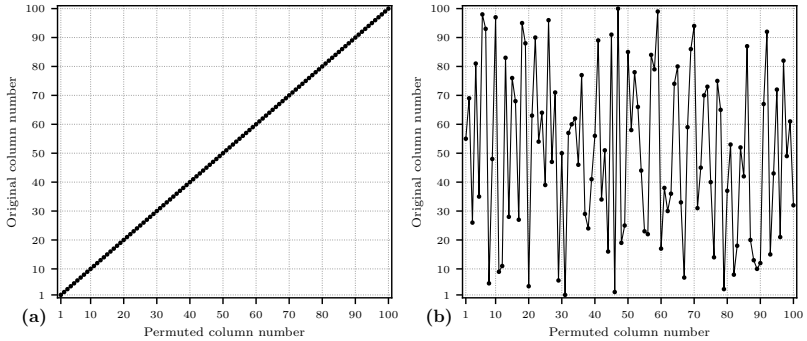


Figure 4.2: Comparison of the original and permuted column numbers in the permutation vector from (a) QR factorization without column pivoting; (b) QR factorization with standard column pivoting

to permute the largest column pivot to the front. For example, if in the kth iteration of the factorization, the mth column has the largest partial norm, then columns m and k are interchanged. A first modification to the pivoting strategy in Eq. 4.7 is to use the *threshold pivoting* strategy

$$\text{pivot column index} = p \mid s_p^{(k)} = \max\left(s_j^{(k)}\right) \text{ and } \frac{s_p^{(k)}}{s_k^{(k)}} > \rho_{piv}, \quad k \le j \le m, \tag{4.22}$$

where $\rho_{piv} > 1.0$ is a threshold for pivoting. In this strategy, the column with the largest partial norm in the kth iteration of the factorization is permuted to the kth column only if its partial column norm is larger than that of the kth column by at least a factor defined by ρ_{piv}. If the conditions in Eq. 4.22 are not met, then no column permutation occurs in the kth iteration of the factorization, and the kth remains in its place. A further modification to the pivoting strategy is what we call the *closest column pivoting* strategy. In this strategy, Instead of the column with the maximum partial norm, the candidate for the pivot column is chosen to be the closet column that satisfies the threshold pivoting condition. The combination of the threshold pivoting and the closest column pivoting strategies leads to the following condition:

$$\text{pivot column index} = \min(j) \mid \frac{s_j^{(k)}}{s_k^{(k)}} > \rho_{piv}, \quad k \le j \le m. \tag{4.23}$$

In this strategy, there is more tendency to preserve the original column ordering of matrix $\boldsymbol{V}$. At the kth iteration of the factorization, a column permutation happens only if a column has a partial norm larger than that of the kth column by a user-defined factor. Otherwise, the kth column remains in its place. Furthermore, if a permutation happens, the kth column is

interchanged with the closest column that meets the pivoting criterion in Eq. 4.23. Based on experience, a value between 5 and 10 for the threshold ρ_{piv} leads to a good compromise between column permutation and preserving the column ordering, with $\rho_{piv} = 10$ being the default value.

An alternative threshold pivoting criterion, discussed, e.g., in [156] for preserving the structure of the QR factorization of sparse matrices is

$$\text{pivot column index} = \min_{k \le p \le m} p \mid s_p^{(k)} \ge \frac{\max\left(s_j^{(k)}\right)}{\rho_{piv}}, \quad k \le j \le m, \tag{4.24}$$

where $\rho_{piv} \ge 1.0$ is a user-defined threshold. In this strategy, the columns with thee larger partial norms are permuted towards the front, as in the standard pivoting strategy. However, instead of permuting the column with the largest partial norm, the closest column with a norm at least as large as a user-defined fraction of the maximum partial norm is permuted to the kth column. Both pivoting criteria 4.24 and 4.23 help preserve the original column ordering compared to the standard pivoting criterion 4.7. In practice, the pivoting criterion in Eq. 4.24 prioritizes moving the columns with larger partial norms to the front more, while the pivoting criterion 4.23 rather prioritizes keeping the original column ordering unless in the kth iteration, a relatively small pivot is placed in the kth column, prompting a permutation.

4.2.3 Cyclic permutation

In the standard column pivoting, a permutation interchanges the position of two columns. In the kth iteration of the factorization, if the pth column is chosen to be the pivot column ($k < p$), then the pth column is moved towards the front by interchanging the kth and the pth columns. By doing so, the sequence of the columns $k, k+1, \ldots, p-1$ is interfered with since the kth column is moved to the back of the sequence. In order to further preserve the ordering of the columns, one can perform a *right cyclic shift* ([140, 157, 156]) of the sequence of columns between k and p instead of column interchanges when a permutation occurs, i.e., for $k < p$,

$$k \longrightarrow k+1, \;\; k+1 \longrightarrow k+2, \;\; \ldots, \;\; p-1 \longrightarrow p, \;\; p \longrightarrow k. \tag{4.25}$$

Note that by *cyclic permutation*, we mean the technique of performing *cyclic shifts* instead of column interchanges. This should not be confused with the *cyclic pivoting* method introduced in [139].

The QR factorization of the example $\boldsymbol{V}$ matrix from section 4.2.1 is recalculated using the threshold and closest column pivoting criteria 4.23 and the cyclic permutation technique. In Fig. 4.3, the diagonal entries of the R-factor after the modifications to the standard pivoting strategy are depicted. As expected, the sequence of the diagonal elements of $\boldsymbol{R}$ does not approximate the distribution of the singular values as well as the standard pivoting strategy would allow (see Fig. 4.1). However, by comparison with

Fig. 4.1a, the diagonal elements of $\boldsymbol{R}$ reflect the general pattern of the singular values of $\boldsymbol{V}$. This is the comprise that we made in order to preserve the column ordering to some degree.

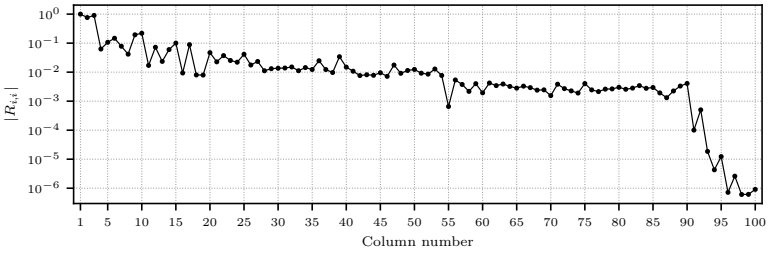


Figure 4.3: Diagonal elements of the R-factor from QR factorization with column pivoting using the threshold and closest column pivoting criteria, and the cyclic permutation technique

In Fig. 4.4, the permutation vector is plotted after the modifications are applied to the standard pivoting strategy. By comparison with Fig. 4.2b, it is evident that the column ordering is quite better preserved compared to the standard column pivoting strategy. At the same time, Fig. 4.4 shows that the columns with significantly smaller partial norms are successfully placed at the trailing part of the matrix and, thus, can be filtered when solving the basic solution via QR factorization with column pivoting.

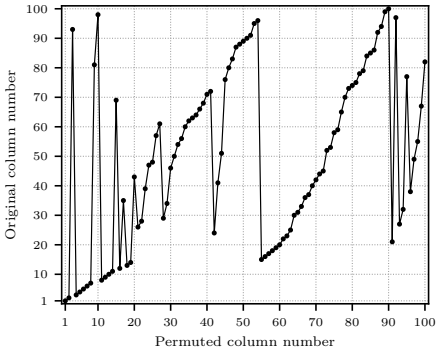


Figure 4.4: The permutation vector from QR factorization with column pivoting using the threshold and closest column pivoting criteria, and the cyclic permutation technique

4.2.4 Age-based column scaling

So far, the columns of matrix $\boldsymbol{V}$ are scaled to have equal and unit norms. This scaling improves the conditioning of the matrix and allows a correct assessment of linear dependency among the columns of $\boldsymbol{V}$. Without column scaling, column pivoting would lead to misleading results as the columns with much bigger norms would always be moved to the front. When the columns of $\boldsymbol{V}$ are scaled to have equal norms, they are in a "neutral" state for column pivoting. No column is prioritized over the other based on the initial column norms.

Recall that we intend to use the filtering (or column subset selection) technique for removing the linearly dependent or less relevant columns as well as for automatically deciding the number of previous time steps used in the IQN-ILS method (i.e., no explicit user-defined parameter for the time step history is required). Through the modifications to the QR factorization with standard column pivoting strategy, a combination of the columns with small pivots and columns from older time steps are placed at the trailing part of matrix $\boldsymbol{R}$. Those columns are excluded from th basic solution via QR factorization with column pivoting (the number of dropped columns depends on a user-defined threshold for deciding the ϵ-rank). Therefore, the combination of threshold pivoting and cyclic permutation techniques, in principle, allows for simultaneously removing the linearly dependent or less relevant columns and controlling the time step history in the IQN-ILS method. However, it is not uncommon in practice that some columns from time steps far back in history remain in matrix $\boldsymbol{V}$, as no explicit parameter for the maximum time step history is used, which can degenerate the convergence rate of the IQN-ILS method. Therefore, a mechanism is required to make sure that the data from very old time steps are consistently removed.

Column scaling does not change the solution of the least square problem. We are free to choose the scaling factor for each column, As long as the diagonal scaling matrix is positive definite. Here, a scaling is chosen that allows prioritizing the columns from newer over the older time steps. In addition to scaling the columns with their corresponding norms, the columns are scaled based on the *age* of their corresponding time step. The age here refers to how far back in history a particular time step is. The age is determined by parameter s. $s = 0$ corresponds to the current time step, $s = 1$ refers to the previous time step, and so on. A function for determining the column scales based on age should return 1 for the columns from the current time step and a progressively smaller value for the columns from the older time steps. The following function is chosen for determining the age-based column scales:

$$\text{scale} = \frac{1}{(\rho_{age})^s}, \tag{4.26}$$

where s is the age of the time step to which each column belongs, and ρ_{age} is a user-defined parameter. Note that the columns from the same time step

have the same age-based scaling. The choice of the scaling function in Eq. 4.26 leads to a straightforward implementation of age-based column scaling. In a new time step, the existing matrix $\boldsymbol{V}$ (from the previous time steps) is divided by the scalar ρ_{age} before the very first column of the new time step is inserted. The age-based scaling affects the singular values of matrix $\boldsymbol{V}$. Therefore, ρ_{age} is recommended to be chosen such that matrix $\boldsymbol{V}$ is not drastically changed since it is known that by having equal column norms, matrix $\boldsymbol{V}$ has close to optimum conditioning. In this work, $\rho_{age} = 1.2$ is used. The age-based scaling factors for columns of the example $\boldsymbol{V}$ matrix from section 4.2.1, with 18 previous time steps, are plotted in Fig. 4.5.

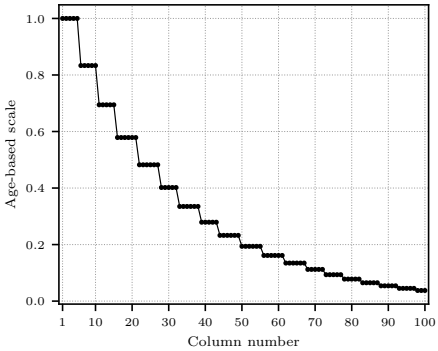


Figure 4.5: Age-based scaling factors for columns of the example matrix $\boldsymbol{V}$ with 18 previous time steps

In Figs. 4.6a and 4.6b, the singular values of the example matrix $\boldsymbol{V}$ without age-based scaling (all columns have equal and unit norm) and with age-based scaling are plotted, respectively. It is shown that the age-based scaling does not change the singular values drastically. The diagonal elements of the R-factor from QR factorization with column pivoting utilizing age-based column scaling are plotted in Fig. 4.6c. By comparing Figs. 4.6c and 4.6b, it can be observed that the diagonal elements of the R-factor approximate the general distribution of the singular values well.

The permutation vector after using age-based column scaling is plotted in Fig. 4.7. By comparing Figs. 4.7 and 4.4, it is clear that the age-based column scaling helps as well with better preserving the column ordering of $\boldsymbol{V}$.

4.2.5 Restricted column pivoting

Restricted column pivoting is a strategy introduced in [158, 159, 156] for restricting the choice of the pivot columns to a *pivot window*, which is a subset of the columns of a matrix. Restricted pivoting was initially used to enhance the efficiency of QR factorization with column pivoting in parallel or block

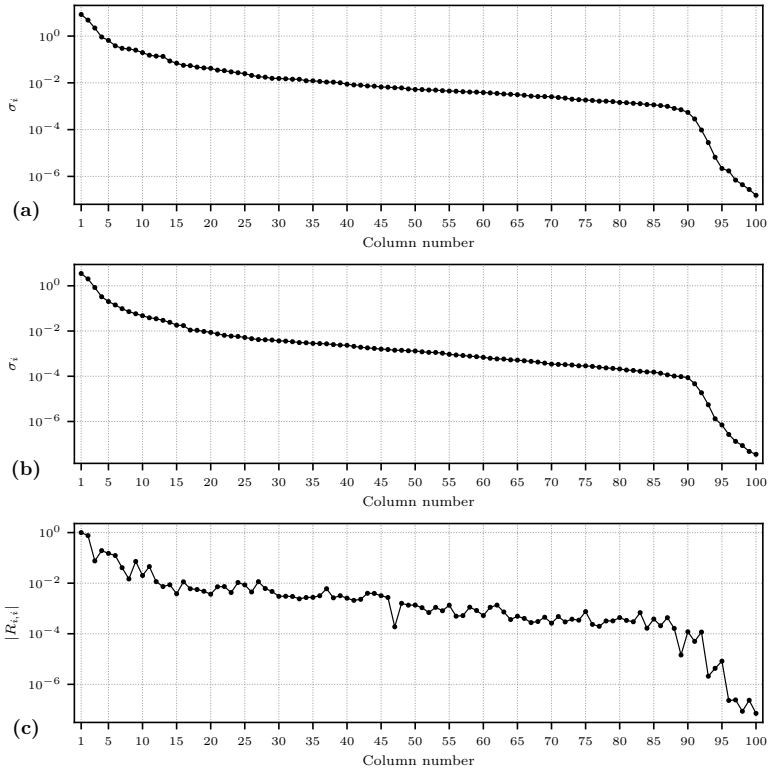


Figure 4.6: The effect of age-based scaling on the example matrix $\boldsymbol{V}$. (a) singular values from SVD without age-based scaling; (b) singular values from SVD with age-based scaling; (c) diagonal elements of the R-factor from QR factorization with column pivoting using the threshold and closest column pivoting criteria, cyclic permutation technique, and age-based scaling

algorithms. In [159], restricted pivoting is used to restrict the choice of pivot columns to the ones in local memory in a distributed memory environment. In [156], the pivot columns are restricted to the ones that preserve the structure and sparsity of a sparse matrix. In the following, the idea of restricted column pivoting is borrowed in order to develop a technique to further control the range of the previous time steps used in the IQN-ILS method.

It occasionally happens that the age-based column scaling introduced in section 4.2.4 does not control the range of the previous time steps to a satisfactory degree unless a sharp scaling function is used. An idea to help control the aforementioned range is to remove columns from very old time steps before removing the almost linearly dependent or less-relevant columns

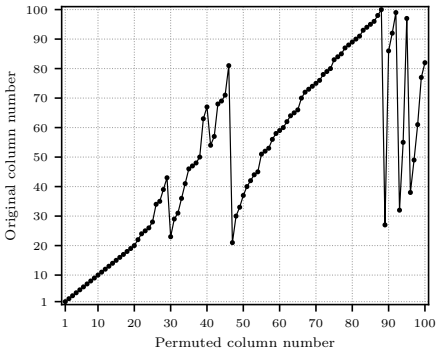


Figure 4.7: The permutation vector from QR factorization with column pivoting using the threshold and closest column pivoting criteria, cyclic permutation technique, and age-based scaling

(depending on the user-defined threshold). In terms of the original column ordering of matrix $\boldsymbol{V}$, removing columns from the middle of the matrix forces removing columns from very old time steps as well. Recall that in the basic solution via QR factorization with column pivoting, the columns at the trailing part of the R-factor are dropped. Therefore, to force the columns from the very old time steps to be removed, we must prevent the pivoting strategy from choosing and moving those old columns to the front of the matrix. Here, the idea of restricted pivoting comes in handy. The pivot window is restricted to a subset of the columns of matrix $\boldsymbol{V}$ excluding the ones from very old time steps. The columns with small pivots are pushed towards the back of the matrix up to the columns from very old time steps (excluded from the pivoting window). In this way, any method that determines the ϵ-rank by processing the columns (or column pivots) of $\boldsymbol{R}$ incrementally would process the columns with small pivots before the columns from the old time steps, and if required (depending on the user-defined threshold), the (almost) linearly dependent columns together with the very old columns are dropped.

The remaining question is: what is the criterion for choosing the size of the pivot window? As previously discussed, restricted pivoting is intended for controlling the range of the previous time steps when the columns from too many time steps are present in matrix $\boldsymbol{V}$. However, restricted column pivoting should not prevent matrix $\boldsymbol{V}$ from growing when columns from fairly recent time steps are used. We choose the number of old time steps whose columns are excluded from the pivot window based on a percentage of the maximum age of columns in matrix $\boldsymbol{V}$. If the oldest time step used in forming matrix $\boldsymbol{V}$ has age s_{max}, the pivot window includes all the columns from s_{win} previous time steps, defined by

$$s_{win} = \text{ceil}\left((1-\xi)\cdot s_{max}\right), \tag{4.27}$$

where ξ, defined as a percentage of s_{max}, controls the range of the old time steps whose columns are excluded from the pivoting window. We use $\xi = 0.1$ in this work. The function ceil(x) in Eq. 4.27 returns the least integer greater than or equal to x. Using Eq. 4.27 together with $\xi = 0.1$ means that if the oldest time step used for forming matrix $\boldsymbol{V}$ has an age less than 10, then $s_{win} = s_{max}$, i.e., the pivot window includes all columns of matrix $\boldsymbol{V}$. If $10 \leq s_{max} < 20$, then the columns from the oldest time step are excluded from the pivot window. If $20 \leq s_{max} < 30$, then the columns from the two oldest time steps with age s_{max} and $s_{max} - 1$ are excluded from the pivot window.

For the example matrix $\boldsymbol{V}$ from section 4.2.1, where the oldest time step used in forming $\boldsymbol{V}$ has an age equal to 18, the presented restricted pivoting strategy excludes the columns from the oldest time step from the pivot window. The diagonal entries of the R-factor using restricted pivoting are plotted in Fig. 4.8. The last three columns (which are from the oldest time step) are kept at the end of the matrix and are the candidates to be dropped.

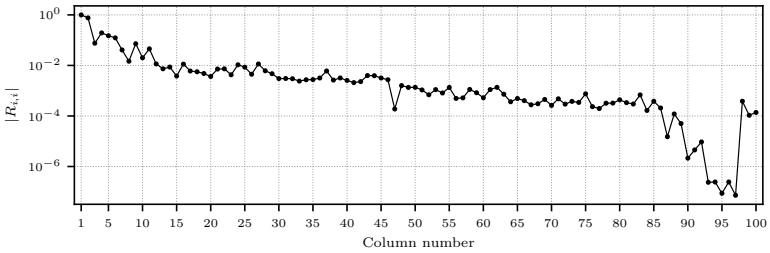


Figure 4.8: Diagonal elements of the R-factor from QR factorization with column pivoting using the threshold and closest column pivoting criteria, cyclic permutation technique, age-based scaling, and restricted column pivoting

The permutation vector plotted in Fig. 4.9 confirms that the last three columns from the oldest time step are excluded from the column window.

4.2.6 Incremental condition estimation

The classical technique for determining the numerical rank of a matrix via QR factorization with column pivoting is based on the diagonal entries of the R-factor as presented in Eq. 4.21, which relies on the fact that the diagonal entries of $\boldsymbol{R}$ via the QR factorization with standard column pivoting form a non-increasing sequence. However, by the modifications made to the standard column pivoting strategy, the diagonal entries of $\boldsymbol{R}$ do not form a strictly non-increasing sequence anymore. Therefore, determining the ϵ-rank using Eq. 4.13 is not reliable anymore. We need a more reliable method for determining the ϵ-rank, which does not rely on the diagonal entries of $\boldsymbol{R}$ forming a non-

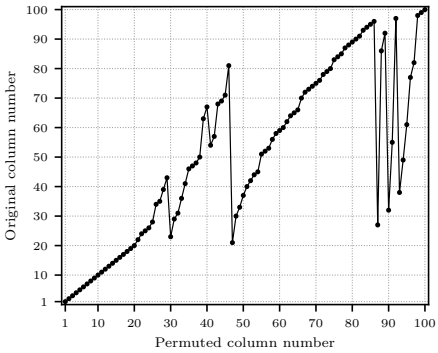


Figure 4.9: The permutation vector from QR factorization with column pivoting using the threshold and closest column pivoting criteria, cyclic permutation technique, age-based scaling, and restricted column pivoting

increasing sequence. We use the *Incremental condition estimator* introduced in [160, 161] for that purpose.

The Incremental condition estimator can incrementally and cheaply estimate the minimum and the maximum singular values (therefore also the condition number) of a matrix generated one column at a time. Those features suit our application very well, where matrix $\boldsymbol{R}$ is generated one column at a time, and we look to determine the ϵ-rank of $\boldsymbol{R}$ incrementally by processing the columns of $\boldsymbol{R}$ starting from the first column and moving towards the back of the matrix. Assume $\boldsymbol{R} \in \mathbb{R}^{m \times m}$ is upper triangular. It is known that

$$\frac{1}{\sigma_{\min}(\boldsymbol{R})} = \left\| \boldsymbol{R}^{-1} \right\|_2, \tag{4.28}$$

where $\sigma_{\min}(\boldsymbol{R})$ is the minimum singular value of $\boldsymbol{R}$. However, computing the inverse of $\boldsymbol{R}$ for calculating $\sigma_{\min}(\boldsymbol{R})$ is undesirable. Very often, an estimation for $\sigma_{\min}(\boldsymbol{R})$ is more desirable. An idea commonly used in estimating $\sigma_{\min}(\boldsymbol{R})$ is to find a large norm solution $\boldsymbol{x}$ to a moderately sized right-hand side $\boldsymbol{b}$ in the following equation ([137, 162, 163, 160]):

$$\boldsymbol{R}^T \boldsymbol{x} = \boldsymbol{b}, \tag{4.29}$$

and, subsequently, calculating

$$\hat{\sigma}_{\min}(\boldsymbol{R}) = \frac{\|\boldsymbol{b}\|_2}{\|\boldsymbol{x}\|_2}, \tag{4.30}$$

where $\hat{\sigma}_{\min}(\boldsymbol{R})$ is an estimation for $\sigma_{\min}(\boldsymbol{R})$ (Note that $\boldsymbol{R}$ and $\boldsymbol{R}^T$ have the same singular values). If $\|\boldsymbol{b}\|_2 = 1$, then $\hat{\sigma}_{\min}(\boldsymbol{R}) = 1/\|\boldsymbol{x}\|_2$, and $\boldsymbol{x}$ is the

approximated singular vector associated with $\hat{\sigma}_{\min}(\boldsymbol{R})$.

Assume that a good estimation for the minimum singular value $\hat{\sigma}_{\min}(\boldsymbol{R}) = 1/\|\boldsymbol{x}\|_2$ is available through a large norm solution to Eq. 4.29 with $\|\boldsymbol{b}\|_2 = 1$. We are interested in finding a large norm solution $\boldsymbol{y}$ to

$$\boldsymbol{R'}^T \boldsymbol{y} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{w} \\ 0 & \gamma \end{bmatrix}^T \boldsymbol{y} = \boldsymbol{b'}, \tag{4.31}$$

where matrix $\boldsymbol{R'}$ is $\boldsymbol{R}$ after a column is added to it, and $\|\boldsymbol{b'}\|_2 = 1$. The incremental condition estimator achieves this goal by calculating $\boldsymbol{y}$ based on $\boldsymbol{x}$ and without accessing $\boldsymbol{R}$ again. The idea of the incremental condition estimator is as follows: set

$$\boldsymbol{b'} = \begin{bmatrix} s\boldsymbol{b} \\ c \end{bmatrix}, \tag{4.32}$$

where $s = \sin\varphi$ and $c = \cos\varphi$. By inserting Eq. 4.32 in 4.31, the solution $\boldsymbol{y}$ is calculated as

$$\boldsymbol{y} = \begin{bmatrix} s\boldsymbol{x} \\ \dfrac{c - s\alpha}{\gamma} \end{bmatrix}, \tag{4.33}$$

where $\alpha = \boldsymbol{w}^T\boldsymbol{x}$. s and c in Eq. 4.33 are found such that $\|\boldsymbol{y}\|_2$ is maximized. The proper values of s and c are calculated analytically in [160, 161]. Having calculated the values of s and c, $\boldsymbol{y}$ is calculated based on $\boldsymbol{x}$ using Eq. 4.33, without accessing $\boldsymbol{R}$. The estimation for the minimum singular value of the updated matrix $\boldsymbol{R'}$ is $\hat{\sigma}_{\min}(\boldsymbol{R'}) = 1/\|\boldsymbol{y}\|_2$, and $\boldsymbol{y}$ is the corresponding estimated singular vector.

The Incremental condition estimator can also be used to calculated an estimated maximum singular value $\hat{\sigma}_{\max}(\boldsymbol{R})$ based on the idea of finding a small norm solution to Eq. 4.29 with $\|\boldsymbol{b}\|_2 = 1$, where $\boldsymbol{b}$ is different than the one used for estimation the minimum singular value. Using this idea, the values of s and c in Eq. 4.33 are determined such that $\|\boldsymbol{y}\|_2$ is minimized [160, 161]. A stable implementation of the incremental condition estimator based on [161] is provided in LAPACK [164]. Note that an estimation for the maximum singular value of $\boldsymbol{R}$ can be also attained using matrix norm inequalities.

Using the incremental condition estimator, $\hat{\sigma}_{\min}(\boldsymbol{R})$ and $\hat{\sigma}_{\max}(\boldsymbol{R})$ are updated as $\boldsymbol{R}$ grows one column at a time, and the ϵ-rank r of matrix $\boldsymbol{R}$ is determined by

$$r = \max_{1 \le j \le m} (j) \;\Big|\; \frac{\hat{\sigma}_{\min}\left(\boldsymbol{R}^{(r)}\right)}{\hat{\sigma}_{\max}\left(\boldsymbol{R}^{(r)}\right)} \ge \epsilon, \tag{4.34}$$

where $\boldsymbol{R}^{(r)} \in \mathbb{R}^{r \times r}$ is a block of $\boldsymbol{R}$ containing the first r columns and rows of $\boldsymbol{R}$. Note that the determination of ϵ-rank can be done incrementally during the factorization process. By doing so, the factorization can be stopped as soon as the ϵ-rank is determined.

4.2.7 The IQN-ILS method with automatic time step history

Recall that the IQN-ILS method with reuse of information from previous time steps requires a problem-dependent user-defined parameter for the maximum reused previous time steps (time step history). In addition, filtering has been shown to enhance the robust and sometimes the efficiency of the IQN-ILS method. Therefore, an additional user-defined parameter is required for the filtering tolerance. Using the developments in the previous sections, we are able to effectively combine the task of determining the time step history with that of column subset selection (filtering). Therefore, no explicit user-defined parameter for the maximum time step history is required. The IQN-ILS method with automatic time step history uses the basic solution via QR factorization with column pivoting as a column subset selection (filtering) technique. The standard pivoting strategy is modified by: threshold pivoting (Eq. 4.24) with $\rho_{piv} = 10$, cyclic permutation, age-based column pivoting (Eq. 4.26) with $\rho_{age} = 1.2$, restricted column pivoting with the pivot window defined by Eq. 4.27 with $\xi = 0.1$, and ϵ-rank determination using incremental condition estimator. Only one parameter remains to be set by the user, and that is, the threshold ϵ for determining the ϵ-rank, which decides the extend to which we keep the columns of $\boldsymbol{R}$ in the column subset selection process. We use $\epsilon = 10^{-4}$ as the default value. The user can change the value of ϵ for decreasing or increasing the number of columns chosen in the column subset selection process in order to increase the convergence rate of the method. That requires prior knowledge about a specific numerical problem or simply trying out the alternative ϵ. However, the presented strategy is not so sensitive to the change of ϵ from the default value.

4.3 Results and discussion

In this section, the proposed strategy for automatic determination of time step history in IQN-ILS method is compared to other well-known convergence accelerator techniques for the simulation of FSI benchmark problems from the literature: Aitken relaxation, IQN-ILS method with different fixed number of previous time steps, and MVQN method. For the Aitken relaxation, an initial relaxation factor of 0.5 is used. For all the quasi-Newton methods, the conditioning of the matrix used for solving the least squares problem is monitored and filtering is applied only for keeping the system well-conditioned. The simulations in this section are carried out in the Kratos Multiphysics[1] open-source framework [141, 142]. The least squares problem entailed in the IQN-ILS method is solved using Householder QR factorization. A linear predictor is used for predicting the displacements of the nodes on the FSI interface in all simulations. A few different convergence criteria are used for evaluating the convergence status of coupling iterations at each time step.

[1] `https://github.com/KratosMultiphysics/Kratos`

The first criterion is an absolute one, based on the magnitude of the interface residual:

$$\frac{\|\boldsymbol{r}^{(l)}\|_2}{\sqrt{n}} < \varepsilon^d_{abs}, \tag{4.35}$$

where $\boldsymbol{r}^{(l)} = \boldsymbol{\mathcal{R}}\left(\boldsymbol{x}^{(l)}\right)$ is the interface (displacement) residual at the lth coupling iteration and n is the number of DOFs on the FSI interface. Instead of an absolute criterion, one can use a relative one

$$\frac{\|\boldsymbol{r}^{(l)}\|_2}{\|\tilde{\boldsymbol{x}}^{(l)}\|_2} < \varepsilon^d_{rel}, \tag{4.36}$$

where $\boldsymbol{r}^{(l)} = \boldsymbol{\mathcal{R}}\left(\boldsymbol{x}^{(l)}\right)$ is the interface residual and $\tilde{\boldsymbol{x}}^{(l)} = \boldsymbol{\mathcal{H}}\left(\boldsymbol{x}^{(l)}\right)$ is the output of the fixed point map at the lth coupling iteration. When used in conjunction with Gauss-Seidel iterations and displacements as interface unknowns, The convergence criteria 4.35 and 4.36 monitor only the displacement residuals and provide no direct information about the convergence of interface traction residuals. In certain applications, it might be necessary to make sure that the equilibrium of interface tractions is fully satisfied. For further discussions on this topic, see [117]. Therefore, in addition to the criterion 4.36, one can simultaneously use a relative convergence criterion based on the interface tractions:

$$\frac{\|\boldsymbol{f}^{(l)} - \boldsymbol{f}^{(l-1)}\|_2}{\|\boldsymbol{f}^{(l)}\|_2} < \varepsilon^f_{rel}, \tag{4.37}$$

where $\boldsymbol{f}^{(l)}$ and $\boldsymbol{f}^{(l-1)}$ are the interface traction vectors at the current and previous coupling iteration, respectively. In the numerical examples presented hereafter, the choice of convergence criteria among Eqs. 4.35, 4.36 and 4.37 will be made such that a better comparison can be done with the results available in the literature.

4.3.1 Driven cavity with flexible bottom

This benchmark problem, proposed in [151, 103], consists of a square cavity with a flexible bottom. The flow inside the cavity is driven by a prescribed periodical velocity at the top boundary. The dimensions of the problem and the boundary conditions are shown in Fig. 4.10a. The definitions of the boundary conditions are taken from [165], which are slightly different from the ones used in [151, 103]. The boundary conditions introduced in [165] lead to a larger deformation of the flexible bottom. The prescribed oscillatory velocity at the top boundary is defined by

$$\bar{u} = 1 - \cos\left(\frac{2\pi t}{5}\right), \tag{4.38}$$

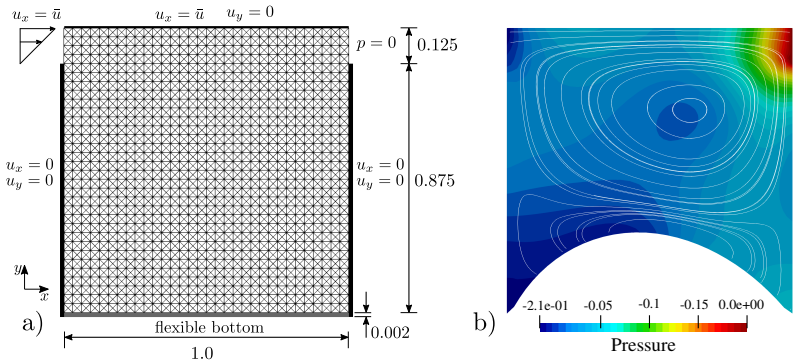


Figure 4.10: Driven cavity: (a) computational domain's dimensions in [m] and the discretization; (b) the pressure contours overlaid by the velocity streamlines at $t = 45.5$ s.

which leads to a velocity range between 0 m/s and 2 m/s. The fluid inflow and outflow defined close to the top of the cavity allow the volume of the cavity to change in time. The fluid domain is discretized using 4096 triangular finite elements, utilizing linear, equal-order interpolation for velocity and pressure (referred to as P1P1[2] element hereafter) and stabilized through the Variational Multiscale Method (VMS) [147, 148]. The Fluid mesh is illustrated in Fig. 4.10. The flexible membrane at the bottom is discretized using 2×32 linear quadrilateral Total Lagrangian elements. The fluid and the structural domains have matching spatial discretizations on the FSI interface. The incompressible fluid has a density of 1.0 kg/m^3 and a kinematic viscosity of 0.01 m^2/s. For the structure, a material with a density of 500 kg/m^3, Young's modulus of 250 N/m^2, and Poisson's ratio of 0 are used. Furthermore, plane stress and small strain elastic behavior is assumed. The coupled simulation is carried out for 70 s with a time step size of 0.1 s for both domains. The absolute convergence criteria 4.35 with the absolute tolerances of $\varepsilon^d_{abs} = 10^{-7}$ is used to check the convergence of the coupling iterations. The pressure contours and the velocity streamlines in the deformed fluid domain at the time instance $t = 45.5$ s are shown in Fig. 4.10b. The vertical displacement of the point in the middle of the flexible membrane is plotted in Fig. 4.11.

The average number of coupling iterations for different convergence acceleration techniques is presented in table 4.1. As expected, the quasi-Newton methods perform better than the Aitken relaxation. Among the quasi-Newton methods, IQN-ILS(3), with information from three previous time steps, performs the best. However, Increasing the time step history to 5 and 10 leads to an increase in the coupling iterations. The IQN-ILS method with automatic time step history (with default parameters) performs almost as good

[2] We designate the triangular (in 2D) or tetrahedral (in 3D) finite elements with linear interpolation for velocity and pressure by P1P1.

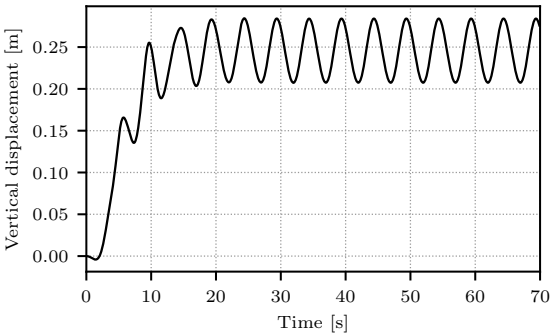


Figure 4.11: Driven cavity: vertical displacement in the middle of the flexible bottom

Table 4.1: Driven cavity: comparison of different convergence acceleration techniques

method	average number of iterations
Aitken Relaxation	9.37
IQN-ILS(0)	5.17
IQN-ILS(3)	3.88
IQN-ILS(5)	4.20
IQN-ILS(10)	4.74
MVQN	4.04
IQN-ILS + automatic time step history, $\epsilon = 10^{-4}$	3.96

as the IQN-ILS(3) and slightly better than MVQN. The main takeaway of the current comparison is that the determination of the time step history in IQN-ILS method can be automatized and combined with the filtering process, without sacrificing too much of performance (in terms of number of coupling iterations).

4.3.2 Flow-induced oscillation of a flexible beam in the wake of a square bluff body

A flexible beam attached to a square bluff body is submerged in incompressible fluid flow. Due to the interaction with the wake of the square body, the flexible beam starts to oscillate. The details of the problem's numerical setup are presented in section 3.7.2.

The average number of coupling iterations using different convergence acceleration techniques is presented in table 4.2 for the Wall benchmark problem. The IQN-ILS method with the automatic determination of time

Table 4.2: Wall benchmark: comparison of different convergence acceleration techniques

method	average number of iterations
Aitken Relaxation	6.6
IQN-ILS(0)	4.74
IQN-ILS(2)	3.73
IQN-ILS(5)	4.02
IQN-ILS(10)	4.33
IQN-ILS(15)	4.58
MVQN	4.36
IQN-ILS + automatic time step history, $\epsilon = 10^{-4}$	3.74

history leads to the least number of coupling iterations. The IQN-ILS(2) method achieves the same performance. The trend of the results for IQN-ILS with fixed number of time step history shows a reduction in efficiency (in terms of number of iterations) as the time step history increases.

4.3.3 Wave propagation in a 3D elastic tube

This benchmark problem consists of a pressure wave propagation through the flexible elastic tube. Initially, the fluid and the structure are at rest. The two ends of the tube are fixed. The pressure at the outlet boundary is set to zero for the duration of the simulation. During the first 3×10^{-3} seconds of the simulation, an overpressure of 1333.2 Pa is applied at the inlet boundary and set to zero afterward. Consequently, a pressure wave propagates through the tube. The problem's detailed description is given in section 3.7.1.

The average number of coupling iterations using different convergence acceleration techniques is presented in table 4.3 for the current problem. The

Table 4.3: Wave propagation in a 3D elastic tube: comparison of different convergence acceleration techniques

method	average number of iterations
Aitken Relaxation	40.2
IQN-ILS(0)	13.12
IQN-ILS(10)	6.74
IQN-ILS(30)	6.32
IQN-ILS(40)	6.21
MVQN	5.34
IQN-ILS + automatic time step history, $\epsilon = 10^{-4}$	6.55
IQN-ILS + automatic time step history, $\epsilon = 10^{-7}$	6.38

first thing to notice in table 4.3 is that the quasi-Newton methods outperform the Aitken relaxation substantially in the current example. The trend of the results for the IQN-ILS method reveals that significantly increasing the time step history leads to better results. However, MVQN is the best performing method in the current example. The performance of MVQN in this example can be attributed to the fact that the approximation for the inverse of interface Jacobian is updated in every time step and then passed to the subsequent time steps. Therefore, information from all previous time steps are implicitly included in the approximation of the Jacobian. The IQN-ILS with automatic determination of time step history has an average performance using the default value of $\epsilon = 10^{-4}$ compared to the other quasi-Newton methods. The proposed column selection strategy with $\epsilon = 10^{-4}$ limits automatically the size of time step history. The user can decrease the ϵ value in order to retain information from older time steps. Using $\epsilon = 10^{-7}$ leads to a slightly better performance, confirming the previous discussion on the effect of time step history size in the current example.

4.3.4 Flexible restrictor flap in converging channel

This example, taken from [166, 167], consists of a flexible restrictor flap in a converging channel. Due to the symmetry, only one half of the channel is simulated. The geometry, the dimensions, and the boundary conditions of the problem are shown in Fig. 4.12. Points A and B, placed on the left side of the flexible flap in Fig. 4.12, are used to sample the pressure and the velocity in the domain. The inflow velocity u_{in} has a parabolic profile

$$u_{in}(y,t) = \frac{y(1-y)}{(0.5)^2} u_{in}(t), \tag{4.39}$$

where the time dependent reference velocity $u_{in}(t)$ increases in time gradually from zero to a nominal value according to

$$u_{in}(t) = \begin{cases} \frac{0.06067}{2}\left(1 - \cos\left(\frac{\pi t}{10}\right)\right) & \text{if } t \leq 10 \\ 0.06067 & \text{otherwise} \end{cases}. \tag{4.40}$$

The fluid domain is discretized using 6922 P1P1 finite elements stabilized by VMS. The Fluid mesh is illustrated in Fig. 4.13. The structure (the flexible flap) is discretized using 4×100 linear quadrilateral Total Lagrangian elements. The number of DOFs in the fluid and structural domains are 14748 and 1515, respectively. The fluid and the structural domains have matching spatial discretizations on the FSI interface. The number of DOFs on the FSI interface is 615. The incompressible fluid has a density of 956 kg/m^3 and dynamic viscosity of 0.145 Pa s. The structural material has a density of 1500 kg/m^3, a Young's modulus of 2.3×10^6 N/m^2 and a Poisson's ratio of 0.45, and plane stress and small strain elastic behavior is assumed. The

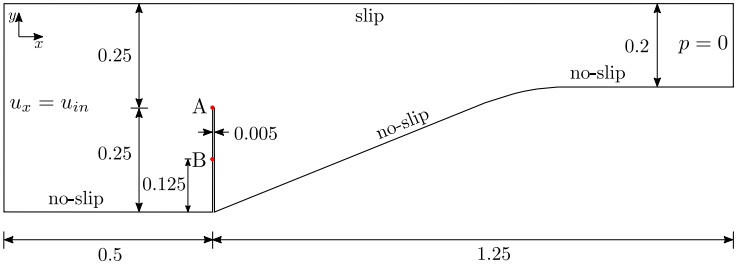


Figure 4.12: Mok benchmark: the domain's dimensions and boundary conditions

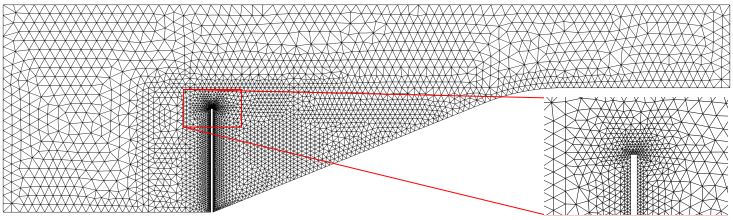


Figure 4.13: Mok benchmark: the fluid's computational grid

coupled simulation is carried out for 25 s with a time step size of 0.1 s for both domains. The absolute convergence criteria 4.35 with the absolute tolerances of $\varepsilon^d_{abs} = 10^{-8}$ is used to check the convergence of the coupling iterations. The horizontal component of the velocity is shown in Fig. 4.14. The pressure and

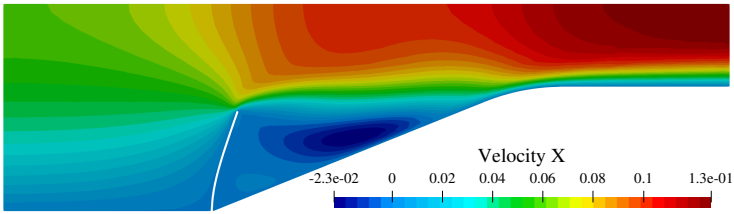


Figure 4.14: Mok benchmark: the contours of fluid velocity in horizontal direction at the end of the simulation

the horizontal component of the velocity at points A and B are plotted in Fig. 4.15.

The average number of coupling iterations using different convergence acceleration techniques is presented in table 4.4 for the Mok benchmark problem. The least number of coupling iterations is achieved using the MVQN method, followed by the IQN-ILS method with automatic determination of time step history. Among the IQN-ILS methods with fixed time step history,

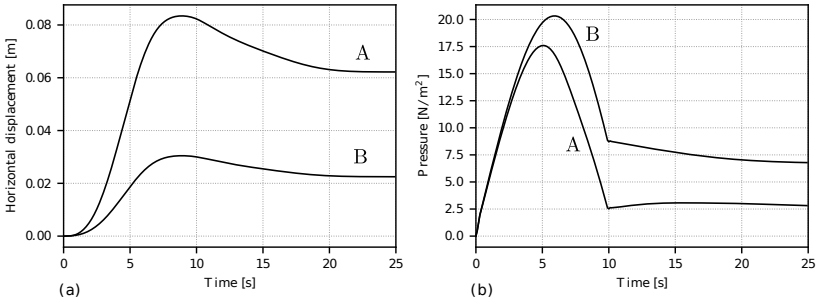


Figure 4.15: Mok benchmark: evolution of (a) horizontal displacements and; (b) pressure values at reference points A and B

Table 4.4: Mok benchmark: comparison of different convergence acceleration techniques

method	average number of iterations
Aitken Relaxation	15.34
IQN-ILS(0)	5.52
IQN-ILS(3)	4.67
IQN-ILS(5)	4.6
IQN-ILS(10)	4.36
IQN-ILS(15)	4.42
MVQN	3.55
IQN-ILS + automatic time step history, $\epsilon = 10^{-4}$	3.72

IQN-ILS(10) performs the best, yet not as good as the proposed column selection strategy.

4.3.5 Flow-induced oscillation of a flexible beam in the wake of a cylinder

This benchmark problem was proposed by Turek and Hron in [89], and has since been used frequently in the literature. There are three different configurations of the benchmark available. We choose the FSI3 configuration, which entails the strongest added-mass effect and the highest Reynolds number. This benchmark problem is conceptually similar to the example presented in section 4.3.2. In this example, we focus on the fully harmonic motion of the flexible beam in the later stages of the simulation. The description and the numerical setup of the benchmark problem are presented in section 3.7.3.

The average number of coupling iterations using different convergence acceleration techniques is presented in table 4.5 for the Turek FSI3 benchmark problem. The IQN-ILS(8) method performs the best in the present example.

Table 4.5: Turek FSI3 benchmark: comparison of different convergence acceleration techniques

method	average number of iterations
Aitken Relaxation	23.7
IQN-ILS(0)	10.81
IQN-ILS(5)	3.82
IQN-ILS(8)	3.72
IQN-ILS(15)	4.54
MVQN	4.81
IQN-ILS + automatic time step history, $\epsilon = 10^{-4}$	4.05

The IQN-ILS method with automatic determination of time step history performs quite good as well. The MVQN method performs worst (among the quasi-Newton methods) in the present example.

Chapter 5

Multi-time-step algorithms for partitioned strongly coupled fluid-structure interaction problems

5.1 Introduction

The present work deals with accurate and efficient multi-time-step coupling algorithms for FSI simulations allowing heterogeneous time integration schemes. The application of subcycling schemes has a long history in certain classes of FSI problems. In particular, there is a rich body of research on fluid subcycling in aeroelastic simulations [170, 171, 172, 173, 174, 175, 176]. The term fluid subcycling means that the time step size of the fluid solver is smaller than that of the structural solver. Hence, for every time step of the structural solver, the fluid solver performs potentially many time steps. Three important choices in the design of the aforementioned fluid subcycling schemes are: transferring the average value of the fluid interface forces over a coarse time step to the structure [174]; handling the interface kinematic coupling conditions on the continuity of discrete interface displacement values; and even distribution of the predicted solid interface displacement to the fluid solver, for which a linear interpolation of interface displacements is advocated [171, 173]. Another dominant theme in [170, 171, 172, 173, 174, 175, 176] is the use of partitioned staggered coupling algorithms for the FSI problem. In the staggered (also called loosely coupled, weakly coupled, explicit, segregated) algorithms, the

structure and fluid problems are solved only once in every time step, i.e., no coupling iterations are performed between the fluid and structure.

While the staggered coupled algorithms are inherently fast and relatively easy to implement, they are known to possess poor stability and accuracy in FSI problems involving high fluid-to-structure density ratios and structures with low stiffness or high slenderness. This class of FSI problems is associated with the strong artificial added mass effect [99, 100, 102]. In such FSI problems, it is often essential to sufficiently satisfy the FSI coupling conditions in each time step in order to reach the desired level of accuracy and stability. The methods that fully satisfy the coupling conditions are referred to as strongly coupled (or implicit) algorithms. Generally, such methods are categorized as either monolithic or partitioned strongly coupled algorithms. The extension of the monolithic approach to FSI problems with non-matching time step sizes for fluid and structure could be computationally very demanding since that entails solving the structure and fluid equations over many time steps simultaneously. On the other hand, the use of some partitioning or domain decomposition methods, in the context of FSI simulations with non-matching time steps, allows the structural and fluid solvers to advance in time as per usual and exchange information at regular intervals in order to solve the interface problem governing the partitioned coupling between the two subdomains. Compared to the staggered algorithms, there are not as many studies available in the FSI literature that investigate, either numerically or analytically, the use of subcycling in the partitioned strongly coupled problems. In [177], the temporal stability of fluid and structure subcycling schemes in partitioned strongly coupled algorithms are studied analytically using a one-dimensional flow in an elastic cylindrical tube and numerically using axisymmetric FSI calculations. For the case of fluid subcycling, a cubic interpolation of interface displacements within a coarse time step (constructed using displacement and acceleration values at the boundaries of the coarse time step) is proposed in [177]. The handling of the interface kinematic conditions is based on the continuity of discrete interface displacement values. In addition, the structure subcycling scheme based on the continuity of the discrete interface displacement is shown to be unstable for bigger fluid-to-structure time step ratios. The convergence of coupling iterations is accelerated using an interface quasi-Newton method [111] formulated using the information at the coarse time steps.

In [178], a multi-rate coupling scheme is proposed for the partitioned strongly coupled problems based on dynamic iterations (waveform relaxation [179]) accelerated by interface quasi-Newton methods [111, 180]. In [178], B-Splines of up to third order are used to interpolate the interface displacements and forces within a time window. In [181], iterative strongly coupled schemes that use different time step sizes for each subdomain are developed for the coupling of heat and wave equations inspired by FSI problems. Furthermore, adaptive time step control for such schemes is proposed. The subdomains are advanced using their own micro time steps within a macro time step used

for coupling. Interpolation of entities between the subdomains is carried out using linear interpolation functions over a macro time step. The convergence of coupling iterations is accelerated using either a relaxation method or a matrix-free Newton-Krylov method.

All the previously cited multi-time-step schemes are based on a Dirichlet-Neumann partitioning of the FSI problem. In [182], a multi-rate extension of the loosely-coupled β-scheme [183, 184] for the case of structure subcycling in FSI problems is proposed. In the algorithm proposed in [182], the structure is advanced in time using a constant Neumann boundary condition predicted from the fluid state at the end of the previous coarse time step. Subsequently, the fluid is advanced one coarse time step using Robin boundary condition set up with the help of the calculated structure's velocity at the last fine time step. In [185], a FETI-based coupling algorithm is used for FSI problems involving fast phenomena requiring explicit time integration schemes in both subdomains but with different time step sizes. The multi-time-step FETI-based coupling schemes were originally developed in the field of structural mechanics, where different methods have been proposed [186, 187, 188, 189, 190, 191, 192, 193]. The common characteristics of the multi-time-step FETI-based coupling methods in the literature are: solving the problem using a dual Schur complement approach, and satisfying the velocity continuity at the common boundaries of the subdomains. The main differences among the FETI-based methods are the satisfaction of kinematic continuity conditions at either the fine or coarse time scales, and interpolation assumptions used for the kinematic entities and Lagrange multipliers. In a different approach to FSI problems with different time scales, a temporal homogenization approach is presented in [194] for long-term simulation of atherosclerotic plaque growth. In this approach, a periodic-in-time fine-scale FSI solution is solved in each long-scale time step and the average of (plaque) growth function is calculated, which is taken into account in the long-scale problem.

Multi-rate coupling schemes have been used as well in other multiphysics problems. In [195], a multi-time-step loosely coupled scheme is presented for coupled free flow and porous media problems. Subcycling in the coupled discrete element method and lattice Boltzmann simulation of particle transport and suspension problems is presented in [196, 197]. Dirichlet-Neumann and Neumann- Neumann waveform relaxation methods are introduced in [198, 199] for parabolic problems and wave equation. Multi-rate Neumann-Neumann waveform relaxation methods are suggested in [200, 201] for heterogeneous coupled heat equations and thermal fluid-structure interaction problems.

In the present work, fluid and structure subcycling schemes are proposed and analyzed for partitioned strongly coupled FSI problems based on Dirichlet-Neumann decomposition. The proposed schemes allow accurate and stable multi-time-step FSI simulations with high time step ratios. An important factor in the design of the multi-time-step schemes considered here is the applicability in a wide range of FSI applications. The iterative partitioned strongly coupled method allows the simulation of FSI problems with strong

added mass effect. Combined with the state-of-the-art interface quasi-Newton methods, the fast convergence of the coupling iterations is achieved. The attributes of the presented schemes allow a straightforward integration into already existing software environments capable of performing FSI simulations. The proposed schemes are applied to some of the challenging FSI benchmark problems, and the effect of subcycling in the fluid and structural solvers are investigated in those problems.

5.2 A model problem

In this section, we first introduce a model problem which facilitates the presentation of the time integration methods as well as the methods related to the coupling of methods with non-matching time step sizes in section 5.3. The model problem is a 1D damped spring-mass system stated by the following ordinary differential equation (ODE)

$$\boldsymbol{M}\partial_{tt}\boldsymbol{d} + \boldsymbol{C}\partial_{t}\boldsymbol{d} + \boldsymbol{K}\boldsymbol{d} = 0, \tag{5.1}$$

where $\boldsymbol{d}$ is the displacement of the mass, $\partial_t \boldsymbol{d}$ and $\partial_{tt}\boldsymbol{d}$ represent the first and second derivatives of $\boldsymbol{d}$ with respect to time, while $\boldsymbol{M}$, $\boldsymbol{C}$ and $\boldsymbol{K}$ are the mass, damping and stiffness of the system, respectively (in order to unify the notation with the general FSI problems, the variables are written using bold symbols). This model problem is used as well in [207, 61, 208] and is shown in [209] to be essentially equivalent in behavior to the 2D linearized problem of a fluid-conveying elastic tube used in [99]. The 1D model problem is handy for studying the influence of time integration schemes in coupled problems, especially concerning the kinematic constraints on the coupling interface.
In order to use the model for studying time discretization in partitioned FSI problems, the model is split into two systems. The splitting is done such that one system models the solid domain in time using a second-order ODE, and the other system models the fluid problem using a first-order ODE. Hereafter, we will refer to the two split systems of the model problem simply as the solid and fluid systems. Let $\boldsymbol{M}^s$ and $\boldsymbol{M}^f$ be the portions of the total mass assigned to the solid and fluid systems, respectively. Furthermore, the stiffness and damping terms are assigned to the solid and fluid systems, respectively, i.e., $\boldsymbol{K}^s = \boldsymbol{K}$ and $\boldsymbol{C}^f = \boldsymbol{C}$. Eq. 5.1 is rearranged as

$$\boldsymbol{M}^s\partial_{tt}\boldsymbol{d}^s + \boldsymbol{K}^s\boldsymbol{d}^s = -\boldsymbol{M}^f\partial_{tt}\boldsymbol{d}^f - \boldsymbol{C}^f\partial_t\boldsymbol{d}^f, \tag{5.2}$$

where the left and right-hand sides of the equation represent the solid and fluid problems respectively. in the present work, the following physical parameters are used for the model problem: $\boldsymbol{M}^s = \boldsymbol{M}^f = 0.5$, $\boldsymbol{K}^s = 1$, $\boldsymbol{C}^f = 0.02$, $\boldsymbol{d}_0^s = \boldsymbol{d}_0^f = 0.1$, $\partial_t\boldsymbol{d}_0^s = \partial_t\boldsymbol{d}_0^f = 0$. Consequently, the undamped natural frequency and the corresponding period of oscillation are $\omega = 1.0$ and $P = 2\pi$, respectively. Eq. 5.2 is subjected to the following continuous-in-time

kinematic constraints:

$$\boldsymbol{d}^f = \boldsymbol{d}^s, \quad \partial_t \boldsymbol{d}^f = \partial_t \boldsymbol{d}^s, \quad \partial_{tt} \boldsymbol{d}^f = \partial_{tt} \boldsymbol{d}^s. \tag{5.3}$$

In the fluid problem, a simple change of variable is carried out as

$$\partial_t \boldsymbol{d}^f = \boldsymbol{v}^f, \tag{5.4}$$

where $\boldsymbol{v}^f$ is the velocity of the fluid. The fluid problem then reads

$$\boldsymbol{M}^f \partial_t \boldsymbol{v}^f + \boldsymbol{C}^f \boldsymbol{v}^f = 0, \tag{5.5}$$

which is a first-order ODE. Eq. 5.5 is subjected to the kinematic coupling conditions. The velocity and displacement of the damper are related through Eq. 5.4 which is similar to the way the fluid velocity and fluid domain's displacement are related on the coupling interface in 2D or 3D FSI problems. A time-discretized form of Eq. 5.4 can be later used to determine the position of the damper through calculating the displacement $\boldsymbol{d}^f$. The equation for the solid problem is stated as

$$\boldsymbol{M}^s \partial_{tt} \boldsymbol{d}^s + \boldsymbol{K}^s \boldsymbol{d}^s = \boldsymbol{\lambda}^s, \tag{5.6}$$

where $\boldsymbol{\lambda}^s$ is the interface reaction force applied to the structure and defined by the equilibrium of forces at the coupling interface

$$\boldsymbol{\lambda}^s = \boldsymbol{\lambda}^f, \quad \text{and } \boldsymbol{\lambda}^f = -\boldsymbol{M}^f \partial_t \boldsymbol{v}^f - \boldsymbol{C}^f \boldsymbol{v}^f. \tag{5.7}$$

5.3 Coupling domains with non-matching time steps

Let us start by motivating the choice of the time grids in the multi-time-step schemes considered here. In the most general setting, the fluid and structural solvers advance in time with their independent and potentially adaptive time step sizes and are required to synchronize their data only at specific time instants. The waveform relaxation methods are iterative coupling methods that are very close to the aforementioned general multi-time-step concept. In waveform relaxation methods, the time interval is usually split into several time windows. In every coupling iteration in each time window, the structure and fluid solvers advance in time with their own time step sizes (in other words, they can both subcycle) till the end of the time window and, subsequently, exchange interface data over the whole time window. This property facilitates adaptive determination of structure and fluid time steps and allows higher interpolation of interface data (if enough data points are available in the time window). On the other hand, the convergence of the dynamic iterations for highly nonlinear problems like FSI is influenced by the length of the time window. Typically, the convergence is faster in shorter time windows. Furthermore, finding a good initial guess for the iterate over the whole time

window is a less trivial task for longer time windows. Finally, the convergence acceleration of waveform iterations is computationally more involved than the classical iterative methods for problems with matching time steps due to the space-time definition of the interface problem in the waveform iteration methods. The more time steps exist in one time window, the more interface data must be stored and operated on for the convergence acceleration, e.g., via quasi-Newton methods. Based on the considerations above, it is decided in this work to choose the coupling windows as short as possible, which is equal to the largest local time step size of the fluid and structural solvers. Designating the larger and smaller time steps of the fluid and structural solvers by the coarse and fine time steps, respectively, the coupling window is chosen to be always equal to the coarse time step. By doing so, the convergence acceleration techniques, e.g., interface quasi-Newton methods, are formulated pointwise in time at the discrete time instants of the coarse time steps without any changes to the formulation required. This approach has the added benefit that the convergence of the coupling iterations is typically achieved faster due to the short coupling window. Furthermore, closed-form definitions of (temporal) interpolation and projection operators between the subdomains are possible since the coupling window is always equal to the coarse time step. On the other hand, the subproblem with the biggest time step size loses the possibility of subcycling in every coupling window in this approach. However, all the other subproblems have the possibility to march with an arbitrary number and size of time steps within a coupling window. The subproblem with the biggest time step size can still adapt its time step when moving from one coupling window to the next, and the coupling window is adapted accordingly.

5.3.1 Fluid subcycling scheme

In this section, we focus on the fluid subcycling case, i.e., the fluid problem is discretized using smaller time steps than the solid problem, either due to the small time scale of the problem or due to the solver requirements. The time grids of the one coupling window are schematically shown in Fig. 5.1. The time indices of the coarse and fine time steps are indicated using n and m, respectively. The coupling between the fluid and solid domains is carried out by iteratively exchanging information through interpolation and projection operators. For the sake of simplicity, it is assumed here that the subproblems have constant time step sizes, and the number of the fine time steps $N^{s,f}$ in each coupling window is an integer number, i.e., $N^{s,f} = \frac{\Delta T^s}{\Delta t^f}$ is an integer.

We now focus on the problem of satisfying the coupling conditions and transferring the data between the solid and fluid subdomains with coarse and fine time steps, respectively. The combination of Dirichlet-Neumann decomposition of the FSI problem and fluid subcycling dictates that the kinematic variable on the FSI interface must be interpolated from the course time step of the solid to the fine time steps in the fluid domain. On the other hand, the fluid tractions on the FSI interface must be transferred from the

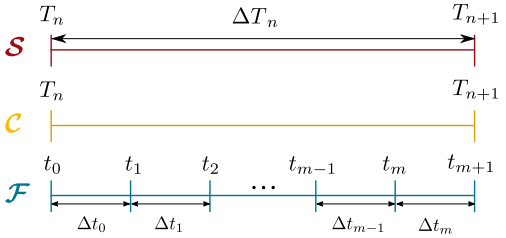


Figure 5.1: From top: the structure, coupling and fluid time grids in the fluid subcycling scheme

fluid domain with fine time step to the solid with the coarse time step. For transferring the interface forces, the most straightforward option is to simply set the values of the fluid and solid tractions at the end of the coarse time step equal, i.e.,

$$\boldsymbol{\lambda}^s_{n+1} = \boldsymbol{\lambda}^f_{n+1}. \tag{5.8}$$

However, using Eq. 5.8 implies a pointwise satisfaction of the dynamic equilibrium condition at the end of the coarse time step. Therefore, any variation in the fluid force within the coarse time step will not be captured in the solid domain, and the interface forces will not be conserved in the coupling window. A better solution for transferring interface forces on the FSI interface arises from the integral form of the dynamic coupling condition over the coarse time step

$$\int_{T_n}^{T_{n+1}} \boldsymbol{\lambda}^s(t)\ \mathrm{d}t = \int_{T_n}^{T_{n+1}} \boldsymbol{\lambda}^f(t)\ \mathrm{d}t. \tag{5.9}$$

In order to evaluate the integrals, we need to know how the forces evolve within the coupling window. We make the assumption that the interface forces vary linearly in time within a fine and coarse time step and are continuous across the time steps. This assumption is reasonable when using the G-α family of time integration schemes since the force term is evaluated as a linear combination of the values at t_n and t_{n+1}. It is then straightforward to formulate the discrete form of Eq. 5.9 over a coupling window and arrive at the following formula for $\boldsymbol{\lambda}^s_{n+1}$:

$$\boldsymbol{\lambda}^s_{n+1} = \frac{2}{\Delta T_n}\Big(\sum_{m=0}^{N^{s,f}-1} \frac{\Delta t_m}{2}(\boldsymbol{\lambda}^f_m + \boldsymbol{\lambda}^f_{m+1})\Big) - \boldsymbol{\lambda}^s_n. \tag{5.10}$$

Using Eq. 5.10, the interface force is preserved in an integral sense over the coupling interface. However, it is not guaranteed that the local maxima and minima in the interface force curve are strictly preserved using Eq. 5.10. It is also worth mentioning that the simple closed-form solution in Eq. 5.10 comes as a result of limiting the coupling window to the coarse time step size of the

solid problem.

Regarding the treatment of the kinematic constraints on the FSI interface, the general approach here is to strongly fulfill the constraints on the discrete interface kinematic variables at the beginning and end of each coarse time step and determine the kinematic variables at the fine time scale through interpolation from the values at the coarse time scale. With the help of the model problem introduced in section 5.2, different approaches for satisfying the FSI kinematic constraints will be presented and compared. The coarse and fine time step sizes are chosen as $\Delta T^s = P/10$ and $\Delta t^f = P/1000$, subsequently, where $P = 2\pi$ is the period of oscillation of the undamped system. Therefore, the structure-to-fluid time step ratio is $N^{s,f} = 100$. The coarse time time size is intentionally chosen to be relatively large in order to test the accuracy and robustness of the presented approaches. For the same reason, the time integration parameters for the solid and fluid problems are set to $\rho_\infty^s = \rho_\infty^f = 0$ unless stated otherwise. As will be discussed later, the chosen time integration parameters are the most challenging for interpolation operations. Typically, larger values for ρ_∞^s and ρ_∞^f are used in practice.

Utilizing the common approach of strongly satisfying the discrete displacement continuity condition, Eq. 2.233, at the end of every coarse time step and assuming a linear variation of displacement within the coarse time step (i.e., the most straightforward option using two data points), the fluid interface displacement is calculated by interpolating between $\boldsymbol{d}_n^s$ and $\boldsymbol{d}_{n+1}^s$ as

$$\boldsymbol{d}_{m+1}^f = (1 - \frac{t_{m+1} - T_n}{\Delta T_n})\boldsymbol{d}_n^s + \frac{t_{m+1} - T_n}{\Delta T_n}\boldsymbol{d}_{n+1}^s, \quad 0 \leqslant m < N^{s,f}. \tag{5.11}$$

The fluid velocity and acceleration will be determined by the time integration scheme in the fluid domain. Therefore, the discrete velocity continuity condition, Eq. 2.235, will not be exactly satisfied at the end of the coarse time step. The simulation results of the model problem using the linear interpolation of the displacement within a coarse time step are plotted in Fig. 5.2. Note that, in all of the following figures, the data are plotted using continuous curves for the sake of a more clear presentation of data. By design, the fluid displacement profile in Fig. 5.2a is linear within every coarse time step. Consequently, it is expected that the fluid velocity is piecewise constant and discontinuous at the boundaries of two consecutive coupling windows, which is observed in Fig. 5.2b. However, there is an overshoot of fluid velocity at the beginning of each coupling window due to the difficulty of precisely capturing discontinuities at the boundaries of the coupling windows by the time integration scheme. Due to the linear evolution of fluid displacements within the coarse time step, the fluid acceleration profile is expected to be constant at zero value. This is generally true in Fig. 5.2c. However, there are significant overshoots at the boundaries of the coupling windows showing as spikes in Fig. 5.2c. These big overshoots are due to the discontinuous fluid velocity profile. Finally, the interface forces in Fig. 5.2d reflect the pattern of the acceleration profile as it is directly related to the acceleration

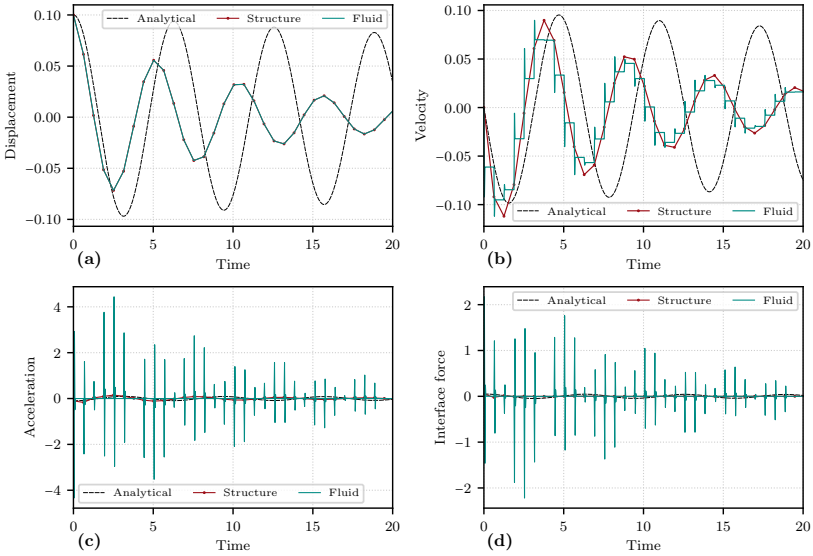


Figure 5.2: Model problem: fluid subcycling results using linear interpolation of displacement values

in our model problem. The numerical solutions in Fig. 5.2 are also generally very diffusive. We can conclude that the assumption of linear variation of the fluid displacement within a coarse time step is not satisfactory, and a better approximation is required. In particular, we will require that the fluid acceleration profile is continuous across the coupling windows, since it will also directly affect the smoothness of the interface forces, as concluded from the results in Fig. 5.2.

An approach based on interpolating the interface displacement while satisfying the continuity of the fluid acceleration across the coupling windows is proposed in [177]. In particular, a cubic interpolation function for the fluid displacement within each coarse time step is constructed using the values of solid displacement and acceleration at the beginning and end of each coupling window, i.e., $\boldsymbol{d}^s_n$, $\boldsymbol{d}^s_{n+1}$, $\dot{\boldsymbol{v}}^s_n$, $\dot{\boldsymbol{v}}^s_{n+1}$. This approach is examined for the model problem, and the results are presented in Fig. 5.3. As expected, the fluid displacement curve in Fig. 5.3a interpolates the solid displacement smoothly. However, the fluid velocity curve in Fig. 5.3b possesses discontinuities across the consecutive coupling windows. Since no constraint on the continuity of the fluid velocity across the coupling windows is imposed, a smooth profile of the fluid velocity is not guaranteed. Even though the solid acceleration data at the beginning and end of the coupling window were utilized in the cubic interpolation of the displacements within a coupling window, the actual evolution of the discrete fluid acceleration obeys the time integration rules

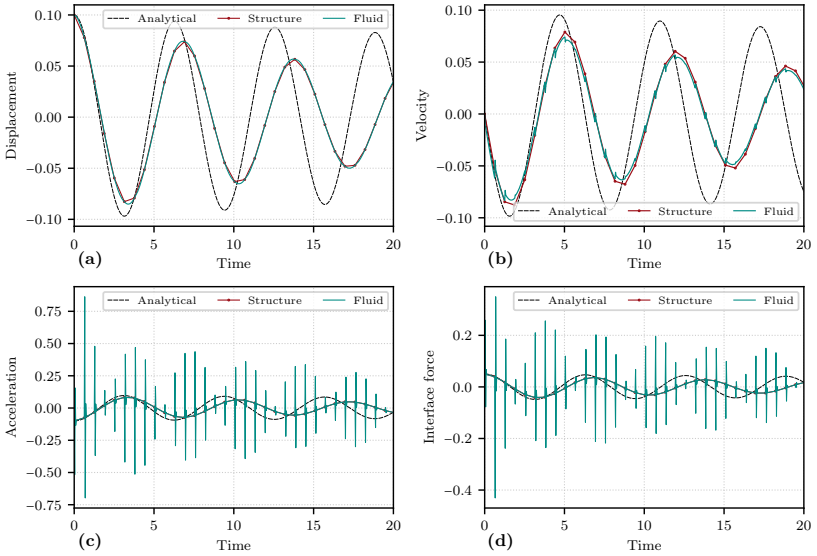


Figure 5.3: Model problem: fluid subcycling results using cubic interpolation of displacement values

and the other kinematic variables in the fluid domain. Hence, the discrete fluid acceleration does not necessarily match the solid acceleration values, specially, at the boundaries of the coupling windows. The fluid acceleration curve in Fig. 5.3c undergoes overshoots across the coupling windows, which is also reflected in the evolution of the interface forces in the fluid domain.

The second approach for tackling the problem of continuity of interface kinematic variables is to strongly fulfill the discrete velocity continuity condition Eq. 2.235 at the beginning and end of each coupling window. The evolution of the displacement and acceleration variables at the interface follows the interface velocity and the time integration scheme used in each subproblem. The simplest method for interpolating the fluid velocity is the linear interpolation between the solid velocities at the beginning and end of each coupling window

$$\boldsymbol{v}^{f}_{m+1} = (1 - \frac{t_{m+1} - T_n}{\Delta T_n})\boldsymbol{v}^{s}_{n} + \frac{t_{m+1} - T_n}{\Delta T_n}\boldsymbol{v}^{s}_{n+1}, \quad 0 \leqslant m < N^{s,f}. \tag{5.12}$$

The simulation results of the model problem using the linear interpolation of the velocity in each coupling window are presented in Fig. 5.4. By construction, the fluid velocity in Fig. 5.4b is linear in time within each coupling window. The fluid displacement curve plotted in Fig. 5.4a is smooth but, as expected, does not interpolate the solid displacement values at the

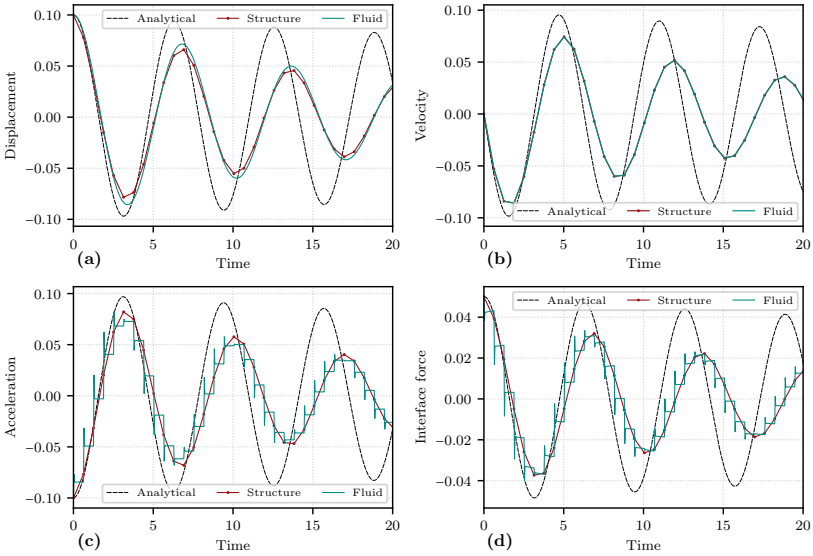


Figure 5.4: Model problem: fluid subcycling results using linear interpolation of velocity values

beginning and end of the coupling windows. The fluid acceleration, plotted in Fig. 5.4c, has a piecewise constant profile and relatively small overshoots at the beginning of each coupling window. The same behavior is also observed in the plot of the fluid interface force in Fig. 5.4d. The (temporal) projection of the fluid forces onto the solid characterized by continuous linear functions in each coupling window can now be better observed. For comparison, the simulation results with the direct (pointwise) transfer of the interface force from fluid to structure at the end of each coarse time step are presented in Fig. 5.5. By comparing the results in Fig. 5.5 to those in 5.4, it is evident that the (temporal) projection of interface forces leads to more accurate results than the direct transfer of forces.

By comparing the results in Fig. 5.4 and Fig. 5.2, it can be concluded that the linear interpolation of the velocities leads to less diffusive results and smaller overshoots in the fluid interface force compared to the case with the linear interpolation of the interface displacements. The plots in Fig. 5.4c and Fig. 5.4d already hint that a smoother interpolation (compared to a linear one) of the interface velocities in the coupling windows might lead to a better representation of the fluid acceleration and interface force.

We can use the solid velocity and acceleration values at the beginning and end of each time window in order to set up a cubic Hermite interpolation for

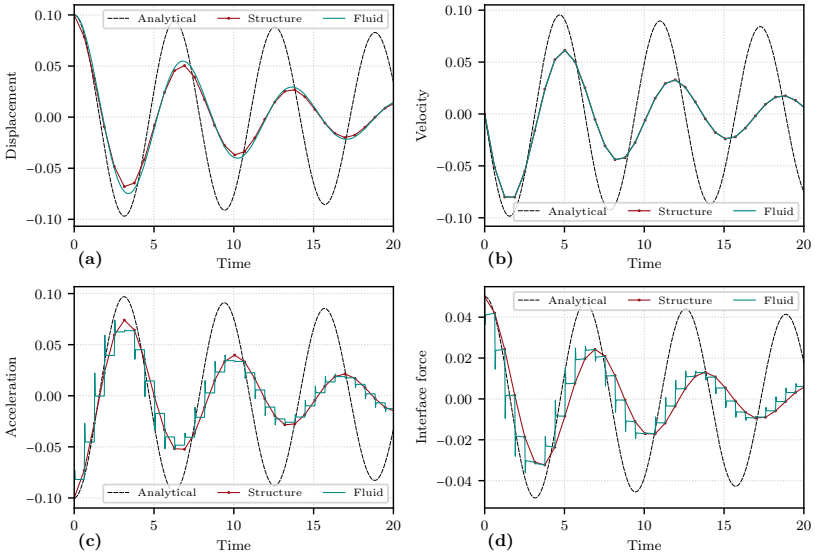


Figure 5.5: Model problem: fluid subcycling results using linear interpolation of velocity values and simple transfer of interface force

the fluid velocity in the same coupling window as

$$\boldsymbol{v}^f_{m+1} = (2\boldsymbol{p}_n - 2\boldsymbol{p}_{n+1} + \dot{\boldsymbol{p}}_n + \dot{\boldsymbol{p}}_{n+1})\xi^3 + (-3\boldsymbol{p}_n + 3\boldsymbol{p}_{n+1} - 2\dot{\boldsymbol{p}}_n - \dot{\boldsymbol{p}}_{n+1})\xi^2 + \dot{\boldsymbol{p}}_n\xi + \boldsymbol{p}_n, \quad 0 \leqslant m < N^{s,f}, \quad \xi = \frac{t_{m+1} - T_n}{\Delta T_n}, \tag{5.13}$$

where the coefficients are chosen as $\boldsymbol{p}_n = \boldsymbol{v}^s_n$, $\boldsymbol{p}_{n+1} = \boldsymbol{v}^s_{n+1}$, $\dot{\boldsymbol{p}}_n = \dot{\boldsymbol{v}}^s_n$, and $\dot{\boldsymbol{p}}_{n+1} = \dot{\boldsymbol{v}}^s_{n+1}$. The simulation results of the model problem utilizing a cubic Hermite interpolation for the velocity in each coupling window are depicted in Fig. 5.6. The fluid velocity curve in Fig. 5.6b is smooth everywhere, owing to the fact that the derivative values (accelerations) are also continuous across consecutive coupling windows. The term "smooth" is used here to imply a C^1 continuous function, i.e., the first derivatives of a function are also continuous. In contrast, the fluid acceleration curve in Fig. 5.6c is only C^0 continuous across the coupling windows. This is due to the fact that the continuity of the second derivative (the temporal derivative of acceleration here) is generally not guaranteed when interpolating values using cubic Hermite interpolation. The kinks in the acceleration curve are also reflected in the fluid interface force depicted in Fig. 5.6d. However, the non-smooth profile of the fluid forces is not transferred to the solid domain due to the (temporal) projection method adopted for transferring the interface forces.

As depicted in Fig. 5.6a, the discrete displacement continuity condition is

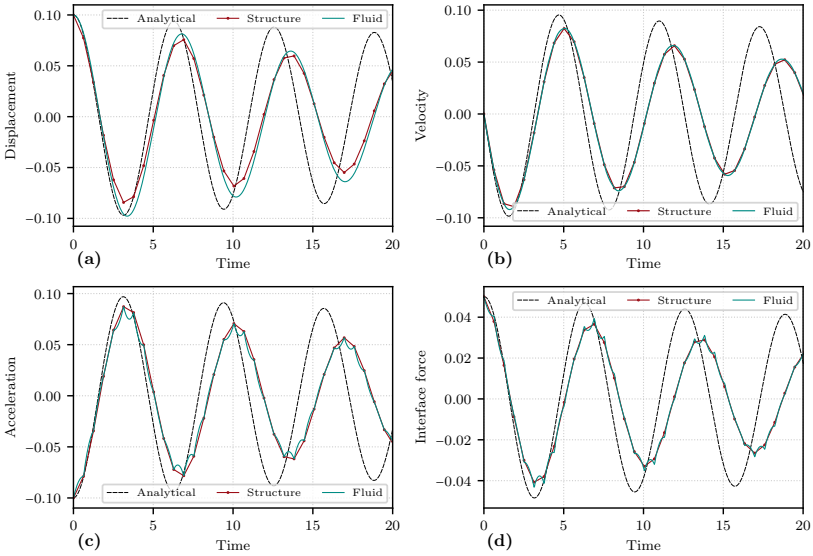


Figure 5.6: Model problem: fluid subcycling results using a cubic Hermite interpolation of velocity values

not exactly satisfied at the boundaries of the coupling windows, i.e., there is a *drift* in the discrete displacement continuity condition Eq. 2.222. A parametric study of the constraint drift will be carried out shortly. The results using the cubic Hermite interpolation of the velocities are generally much better than those using the previously discussed approaches. In particular, much less numerical damping is introduced in the solution due to the coupling of the domains with non-matching time steps. Furthermore, no overshooting occurs in the fluid interface force, although the curve is not smooth due to the fluid acceleration's derivative being discontinuous and possessing varying slopes in the consecutive coupling windows. The smooth and higher order (cubic Hermite) interpolation of the velocities comes at the cost of having no control over the evolution of the accelerations variable within the coupling windows. It is instructive to study the effect of algorithmic parameters on the evolution of the fluid acceleration in each coupling window when using a cubic Hermite interpolation for the fluid velocity. One such parameter is the amount of the high-frequency numerical damping introduced by the G-α time integration scheme in the solid domain and determined by ρ_∞^s. The importance of ρ_∞^s concerning the fluid velocity and acceleration is that the fluid velocities are determined in each coupling window by a cubic Hermite interpolation of the solid velocity and acceleration values whose relation is affected by ρ_∞^s. The evolution of fluid acceleration is plotted in Fig. 5.7 for different values of ρ_∞^s. It is observed that the fluid acceleration profile is

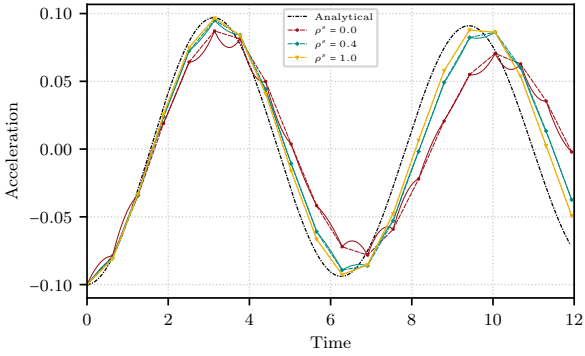


Figure 5.7: Model problem: the effect of ρ^s_∞ on the fluid acceleration profile in fluid subcycling scheme using cubic Hermite interpolation of velocity values. The solid and dashed lines depict the fluid and solid acceleration graphs respectively

linear within each coupling window for $\rho^s_\infty = 1.0$, which implies that the fluid velocity evolves quadratically over each coupling window. As the value of ρ^s_∞ decreases, the fluid acceleration curve in each coupling window deviates from a linear profile. The same behavior is also reflected in the fluid interface force, which is directly related to the fluid acceleration in the model problem.

As a possible solution for alleviating the fluid acceleration's evolution problem, one may think of increasing the interpolation order by adding extra constraints on the acceleration derivatives. However, accurate values for the acceleration derivatives are generally not available. Hence, the higher interpolation functions are prone to generating oscillatory results. Another approach is to reduce the interpolation order to two in order to constrain the evolution of acceleration in each coupling window.

The quadratic interpolation of the velocities requires three data values: the solid velocity values at the beginning and end of the coupling window, $\boldsymbol{v}^s_n$ and $\boldsymbol{v}^s_{n+1}$, and an acceleration value. Considering the requirement that the fluid acceleration must be continuous at the boundaries of the consecutive coupling windows, the fluid acceleration value at the beginning of each coupling window, $\dot{\boldsymbol{v}}^f_n$, is used for constructing the quadratic interpolation. The quadratic fluid velocity profile can be equally achieved using the cubic Hermite interpolation as in Eq. 5.13, for which an acceleration value at the end of the coarse time step is required, which respects the linear acceleration profile. The fluid linear acceleration value $\dot{\boldsymbol{v}}^{f,lin}_{n+1}$ is calculated as

$$\dot{\boldsymbol{v}}^{f,lin}_{n+1} = \frac{2}{\Delta T_n}(\boldsymbol{v}^s_{n+1} - \boldsymbol{v}^s_n) - \dot{\boldsymbol{v}}^f_n. \tag{5.14}$$

Therefore, the quadratic fluid velocity profile can also be achieved by using

the cubic Hermite interpolation, Eq. 5.13, with the coefficient values $\boldsymbol{p}_n = \boldsymbol{v}_n^s$, $\boldsymbol{p}_{n+1} = \boldsymbol{v}_{n+1}^s$, $\dot{\boldsymbol{p}}_n = \dot{\boldsymbol{v}}_n^f$, and $\dot{\boldsymbol{p}}_{n+1} = \dot{\boldsymbol{v}}_{n+1}^{f,lin}$.

The simulation results of the model problem using a quadratic interpolation for the velocity in each coupling window are depicted in Fig. 5.8. The fluid

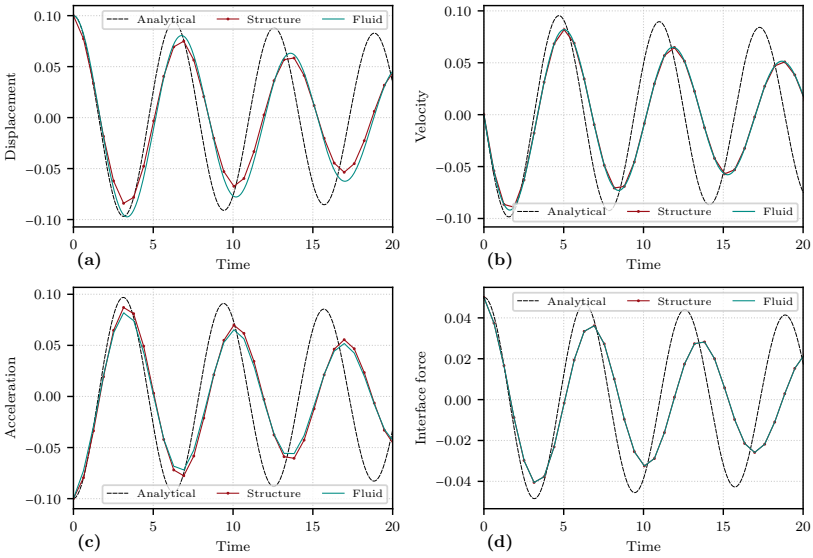


Figure 5.8: Model problem: fluid subcycling results using quadratic Hermite interpolation of velocity values

acceleration in 5.8c evolves linearly over a coupling window. Note that the discrete values of the fluid and solid accelerations do not generally match at the boundaries of the coupling windows when assuming the linear evolution of the fluid acceleration over each coupling window. The quadratic Hermite interpolation of the velocities in a coupling window seems ideal for the present linear model problem. However, the coupled solution using the quadratic Hermite interpolation is, unfortunately, more prone to developing oscillations in nonlinear problems as shown in section 5.4.3. The issue with the quadratic Hermite interpolation is that, while the fluid velocity curve interpolates the solid velocity values $\boldsymbol{v}_n^s$ and $\boldsymbol{v}_{n+1}^s$, the fluid acceleration value at the end of the coarse time step is decoupled from the solid acceleration values. This issue can be best seen in Eq. 5.14, where $\dot{\boldsymbol{v}}_{n+1}^{f,lin}$ is dependent on $\dot{\boldsymbol{v}}_n^f$. Therefore, the accumulation of errors in the fluid acceleration values at the fine time scale may lead to growing oscillations in the interpolated fluid velocity values.

After analyzing the pros and cons of the two approaches using the quadratic or cubic Hermite interpolation of the velocities in coupling windows, we will

try to combine the advantages of the two approaches by requiring the following criteria:

- The fluid velocity curve interpolates the solid velocity values at the beginning and end of the coupling windows.
- Acceleration values (not precisely those of the solid) at the beginning and end of the coupling windows are used as constraints for the interpolation.
- The fluid acceleration curve evolves almost linearly within a coupling window and is continuous (in time) across consecutive coupling windows.
- The acceleration values used in the Hermite interpolation of the velocities should not be totally decoupled from the solid acceleration values at the coarse time scale.

The approach taken here is to construct a cubic Hermite interpolation, Eq. 5.13, using the solid velocities $\boldsymbol{v}_n^s$ and $\boldsymbol{v}_{n+1}^s$. The continuity requirement on the fluid acceleration across the coupling windows leads naturally to using $\dot{\boldsymbol{v}}_n^f$ as the derivative value at the beginning of the coupling window. For the derivative value at the end of the coupling window, a *weighted* acceleration value is used. The weighted acceleration value is a weighted average of the fluid linear acceleration $\dot{\boldsymbol{v}}_{n+1}^{f,lin}$, Eq. 5.14, and the solid acceleration value $\dot{\boldsymbol{v}}_{n+1}^s$ as

$$\dot{\boldsymbol{v}}_{n+1}^{\zeta} = \zeta \dot{\boldsymbol{v}}_{n+1}^{f,lin} + (1-\zeta)\dot{\boldsymbol{v}}_{n+1}^{s}, \quad 0 \leqslant \zeta \leqslant 1. \tag{5.15}$$

The motivation is to constrain the velocity and acceleration values at both ends of the coupling windows while achieving an almost linear profile for the accelerations (and an almost quadratic profile for the interpolated velocities). Based on the aforementioned interpolation criteria, the ζ value should be chosen close to one in order to achieve the almost linear fluid acceleration profile while still including a small contribution from the solid acceleration value $\dot{\boldsymbol{v}}_{n+1}^s$. In this work, $\zeta = 0.9$ is used for all the simulations.

The simulation results of the model problem using a Hermite interpolation for the velocities and employing the almost linear acceleration assumption are plotted in Fig. 5.9. As desired, the plots in Fig. 5.9 are very similar to those using quadratic Hermite interpolation plotted in Fig. 5.8. However, the new approach is more robust than the approach with quadratic Hermite interpolation in the nonlinear FSI simulations as a result of including a contribution of the solid acceleration value in the interpolation constraints (see section 5.4.3). Note that the fluid acceleration curve in Fig. 5.9c is not strictly piecewise linear everywhere, specially close to the peaks. Nonetheless, the fluid acceleration and interface force in Figs. 5.9c and 5.9d are smoother than those in Figs. 5.6c and 5.6d, respectively. The fluid acceleration's evolution in every coupling window gets closer to a linear profile as the value of ρ_∞^s converges to 1.0, as shown in Fig. 5.10.

By choosing to enforce only the discrete velocity continuity condition, Eq. 2.235, at the coarse time level, the evolution of the fluid displacement on

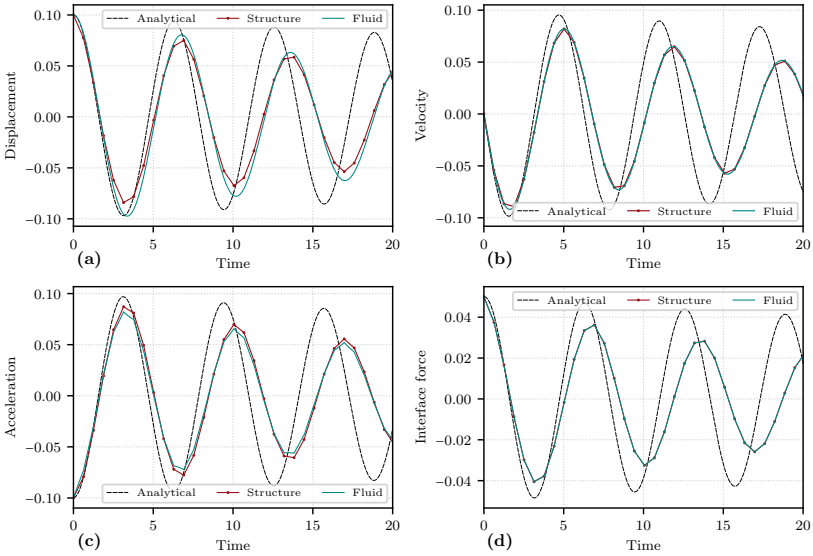


Figure 5.9: Model problem: fluid subcycling results using Hermite interpolation of velocity values and almost linear fluid acceleration assumption

the FSI interface, dictating the fluid domain's deformation in a general FSI problem, is left to be determined by the time integration scheme used in the fluid domain. This approach guarantees a consistent temporal integration of the fluid velocity and (mesh) displacement on the FSI boundary as well as in the interior of the fluid domain. As mentioned previously, the drawback of this approach is that it generally leads to a drift in the discrete displacement continuity condition Eq. 2.233. This drift results partly from using different time integration schemes (or different parameters) in the solid and fluid domains and partly due to the assumptions made in the interpolation of velocities within a coupling window. The drift in the discrete displacement continuity condition is typically more pronounced in the long-term simulations and is affected by the size of the coarse time step. The long-term evolution of the solid and fluid displacements in the model problem is plotted in Fig. 5.11 for the same fine time step size but different values of the coarse time step. It is observed that the drift in the discrete displacement continuity condition is directly related to the coarse time step size. More specifically, the fluid displacement consistent with a cubic (or quadratic) Hermite interpolation of the velocity within a coupling window does not necessarily match the solid displacement values at the coarse time scale. The bigger the coarse time step is, the more assumption is involved in determining the evolution of the interpolated fluid velocity and interface displacement.

Another factor influencing the drift in the discrete displacement continuity

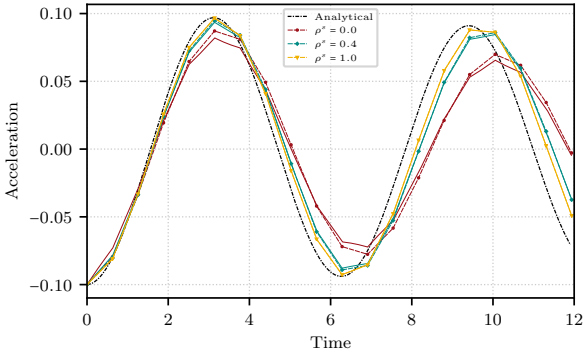


Figure 5.10: Model problem: the effect of ρ^s_∞ on the fluid acceleration profile in fluid subcycling scheme using Hermite interpolation of velocity values and almost linear fluid acceleration assumption. The solid and dashed lines depict the fluid and solid acceleration graphs respectively

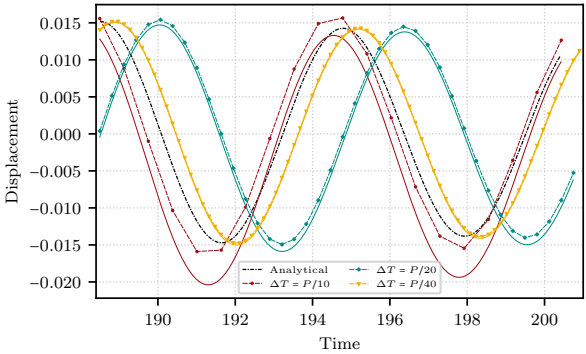


Figure 5.11: Model problem: the effect of the coarse time step size on the long-term drift in the displacement constraint in fluid subcycling scheme using Hermite interpolation of velocity values and almost linear fluid acceleration assumption. The solid and dashed lines depict the fluid and solid displacement graphs respectively

condition is the use of different (also called heterogeneous) time integration schemes in the solid and fluid domains. This work deals with the family of G-α time integration schemes for both fluid and solid problems. The G-α scheme for first-order systems is used in the fluid problem. For the solid problem, there is the possibility of either using the G-α scheme for second-order or first-order systems. Here, the effect of integrating the solid problem using either of the two variants of the G-α scheme on the drift in the displacement constraint is investigated. In Fig. 5.12, the evolution of drift in the displacement constraint,

measured as the difference of the fluid and solid displacements at the coarse time level, is plotted for the two variants of the G-α scheme and different values of ρ_∞^s. For $\rho_\infty^s = 0$, the drift in the displacement constraint is quite higher

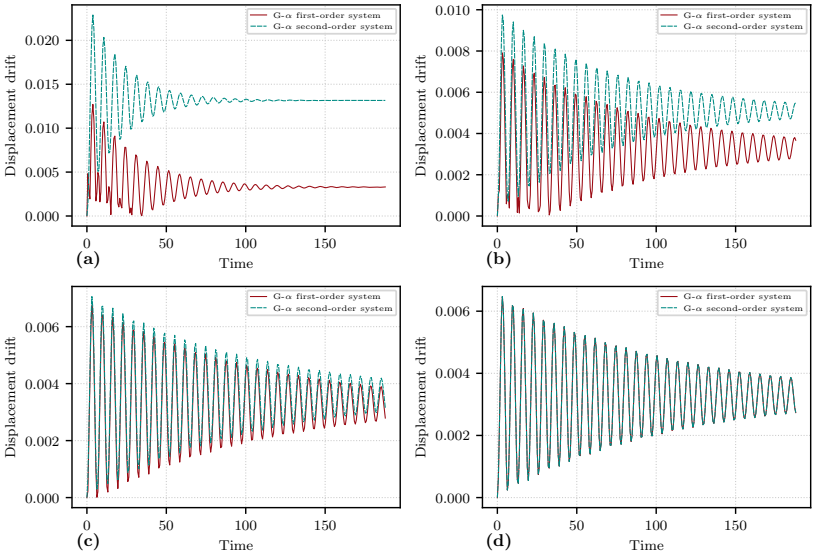


Figure 5.12: Model problem: the effect of ρ_∞^s on the long-term drift in the displacement constraint in fluid subcycling scheme using Hermite interpolation of velocity values and almost linear fluid acceleration assumption. (a) $\rho_\infty^s = 0$; (b) $\rho_\infty^s = 0.4$; (c) $\rho_\infty^s = 0.7$; (c) $\rho_\infty^s = 1.0$

when using the G-α scheme for second-order systems in the solid problem compared to the G-α scheme for first-order systems. The difference between the two variants of the G-α scheme concerning the drift in the displacement constraint decreases as the amount of the high-frequency damping is reduced, and for $\rho_\infty^s = 1$, they produce identical results. The results in Fig. 5.12 suggest that using the G-α scheme for first-order systems in both the fluid and solid problems is favorable for reducing drifts of kinematic constraints in the context of FSI simulations with non-matching time steps.

The present discussion concludes that if the coarse time step of the solid problem is reasonably fine to resolve the evolution of solid velocity adequately, and proper caution is taken in choosing the time integration schemes and their corresponding parameters, then the drift in displacement constraint is kept at a reasonable level by strongly enforcing the discrete velocity continuity condition at the coarse time scale. In practice, it is not so difficult to fulfill the required conditions. Due to the accuracy requirements, the coarse time

steps usually resolve the solid velocity history better than the quite coarse time steps used in the model problem. Moreover, the typical values of ρ_∞^s and ρ_∞^f in G-α methods are 0.5 or higher, which lead to smaller drift in the displacement constraint. Therefore, the preferred approach in this work for dealing with the kinematic coupling conditions is to strongly satisfy the discrete velocity continuity at the coarse time level and choose reasonable time integration parameters (see the results in section 5.4). Nevertheless, a solution based on simultaneously satisfying the continuity of discrete velocities and displacements at the coarse time level will be presented for the cases where the drift in the displacement constraint remains large otherwise.

By strongly satisfying the continuity of displacements at the coarse time level, the fluid interface displacement within a coupling window is determined by interpolation from the solid values at the coarse time level. The kinematics of the FSI interface is then effectively determined by the kinematic values of the solid problem. The interpolation of the displacements must be compatible with the assumptions made on the evolution of the accelerations and velocities. Recall that we assume an almost linear evolution of acceleration and a quadratic evolution of the velocity within a coupling window. Therefore, it is reasonable to assume a cubic evolution of displacement in a coupling window. To that end, the fluid displacement is determined using a cubic Hermite interpolation, Eq. 5.13, with the coefficients $\boldsymbol{p}_n = \boldsymbol{d}_n^s$, $\boldsymbol{p}_{n+1} = \boldsymbol{d}_{n+1}^s$, $\dot{\boldsymbol{p}}_n = \boldsymbol{v}_n^s$, and $\dot{\boldsymbol{p}}_{n+1} = \boldsymbol{v}_{n+1}^s$. The proposed interpolation is also compatible with the fact that the continuity of velocities is strongly enforced at the coarse time scale. Note that two separate functions are set up for interpolating displacements and velocities. Consequently, the evolution of the fluid displacement and velocity within a coarse time step is not necessarily precisely consistent. However, at every time instant of the coarse time level, the fluid interface displacement and velocity become exactly consistent with respect to the time integration scheme of the structural solver. Therefore, any slight inconsistency between the evolution of fluid displacement and velocity within a coupling window does not propagate in time beyond the end of each coupling window. The simulation results of the model problem using separate cubic Hermite interpolations of displacements and velocities (together with almost linear fluid accelerations) in each coupling window are depicted in Fig. 5.13.

The influence of integrating the solid problem using the G-α scheme for first or second-order systems on the drift in the displacement coupling constraint was previously shown in Fig. 5.12. The long-term solution of the model problem using the new approach of interpolating the fluid displacements and velocities is compared in Fig. 5.14 for two variants of the G-α scheme for the solid problem. The overall solution of the model problem is less diffused when integrating the solid problem using the G-α scheme for first-order systems. This observation comes as no surprise since integrating the solid problem using the G-α scheme for firs-order systems is more consistent with the time integration scheme in the fluid domain.

We can now elaborate on the details of the coupling algorithm used for

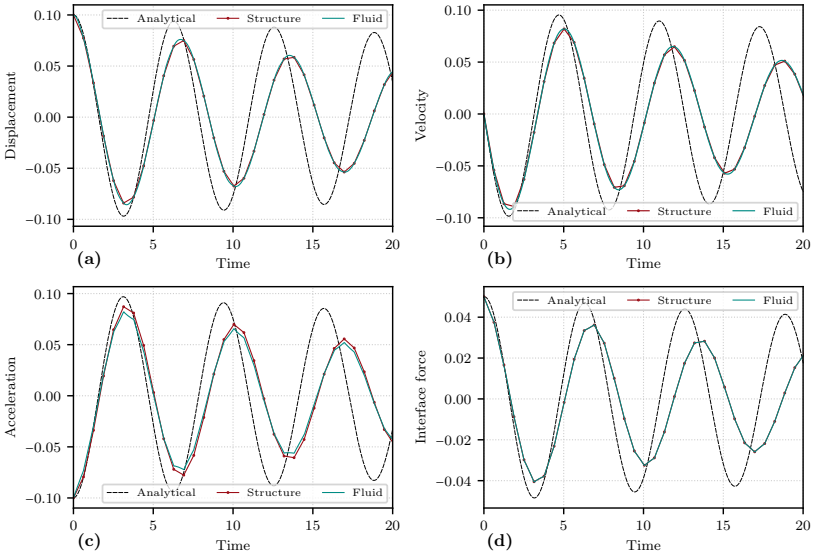


Figure 5.13: Model problem: fluid subcycling results using Hermite interpolation of velocity values and almost linear fluid acceleration assumption, together with a cubic Hermite interpolation of displacement values

FSI simulations with fluid subcycling. The flowchart of the proposed FSI coupling algorithm with fluid subcycling is sketched in Fig. 5.15, where different colors distinguish different groups of coupling actions. There is room for adapting an actual implementation of the algorithm to different code environments based on, e.g., whether the coupling is handled centrally or peer-to-peer or the ability of the solvers to perform all the required actions internally. In every coupling window, the algorithm starts by *checkpointing* the essential data of the fluid solver (with fine time steps) for restarting the fluid solver from the beginning of the coupling window if a new coupling iteration is required. It is assumed that the fluid solver is capable of performing checkpointing/restarting. In the next step, the solid interface velocity at the end of the coupling window $\boldsymbol{v}^{\mathrm{s}}_{n+1}$ is predicted. A linear prediction of $\boldsymbol{v}^{\mathrm{s}}_{n+1}$ is carried out here, requiring $\boldsymbol{v}^{\mathrm{s}}_{n}$ and $\boldsymbol{v}^{\mathrm{s}}_{n-1}$ from the last two coupling windows. It is also possible to use the derivative values in the prediction if one wishes to since they are also available. Limiting the coupling window's size to that of the coarse time step has the advantage that the prediction is probably more accurate due to the coupling window being as short as possible. Next, the interpolation functions for the fluid interface velocity and displacement (if required) within the coupling window are constructed. A more detailed flowchart of the interpolation step is depicted in Fig. 5.16. If a cubic Hermite interpolation is used for interpolating the predicted interface velocity values,

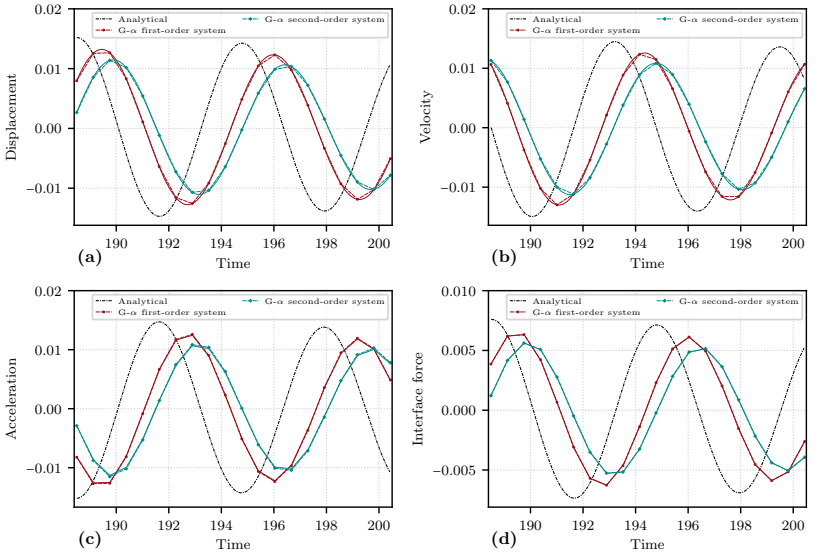


Figure 5.14: Model problem: comparison of integrating the solid problem using the G-α schemes for first or second-order systems in long-term fluid subcycling simulation using Hermite interpolation of velocity values and almost linear fluid acceleration assumption, together with a cubic Hermite interpolation of displacement values

the solid interface acceleration at the end of the coarse time step $\dot{\boldsymbol{v}}^s_{n+1}$ and fluid linear acceleration $\dot{\boldsymbol{v}}^{f,lin}_{n+1}$ are required. However, in every coupling window's very first coupling iteration, the predicted values of $\boldsymbol{v}^s_{n+1}$ and $\dot{\boldsymbol{v}}^s_{n+1}$ might be far from the final solution. Consequently, a cubic Hermite interpolation using those prediction values might lead to a fluid velocity evolution far from the final solution, thus hindering the convergence rate of the iterative coupling. Therefore, a quadratic Hermite interpolation using $\boldsymbol{v}^s_n$, $\boldsymbol{v}^s_{n+1}$, and $\dot{\boldsymbol{v}}^f_n$ is carried out when interpolating the predicted solid interface velocities in the fluid domain. From the second coupling iteration onward, the interpolation function for the velocities is set up as a cubic Hermite interpolation using the values of $\boldsymbol{v}^s_n$, $\boldsymbol{v}^s_{n+1}$, $\dot{\boldsymbol{v}}^f_n$ and $\dot{\boldsymbol{v}}^{f,lin}_{n+1}$. The steps related to the interpolation of displacement can be dropped If one decides to allow the fluid displacement to be determined consistently from the interface velocities with the help of the time integration scheme of the fluid problem. Next, the fluid solver marches through the coupling window using its fine step steps. In every fluid time step m, the fluid interface velocity $\boldsymbol{v}^f_{m+1}$ and displacement $\boldsymbol{d}^f_{m+1}$ (if required) are interpolated using the constructed interpolation functions as the input, and the fluid problem is then solved. At every fluid time step, the local time integral of the interface is calculated using the information available in the

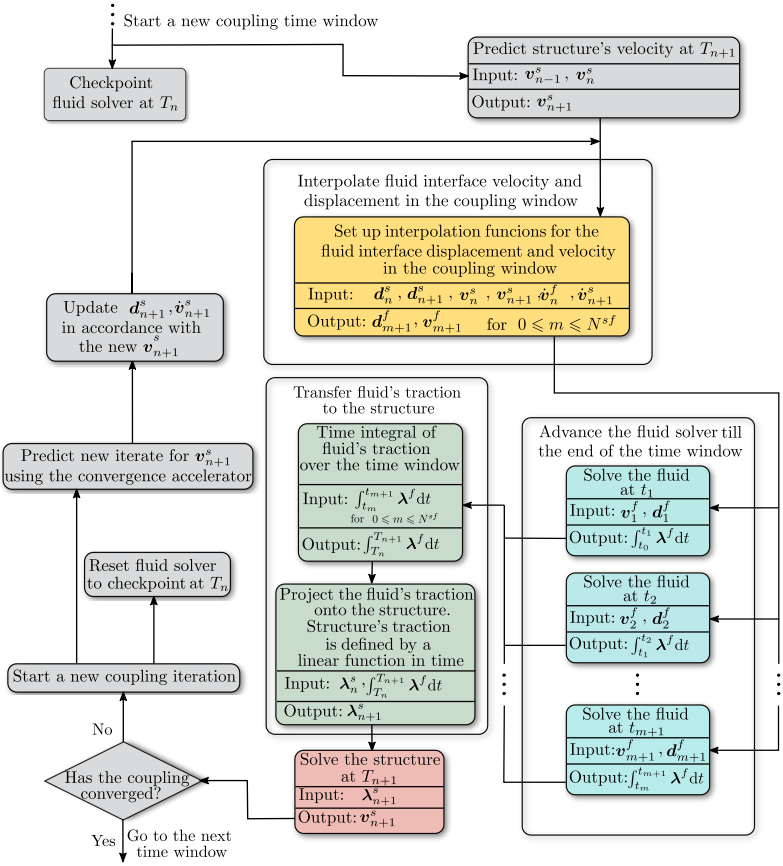


Figure 5.15: The flowchart of the proposed fluid subcycling scheme. The interpolation step is shown in more detail in Fig. 5.16. The steps for interpolating the interface displacement values can be dropped if not required.

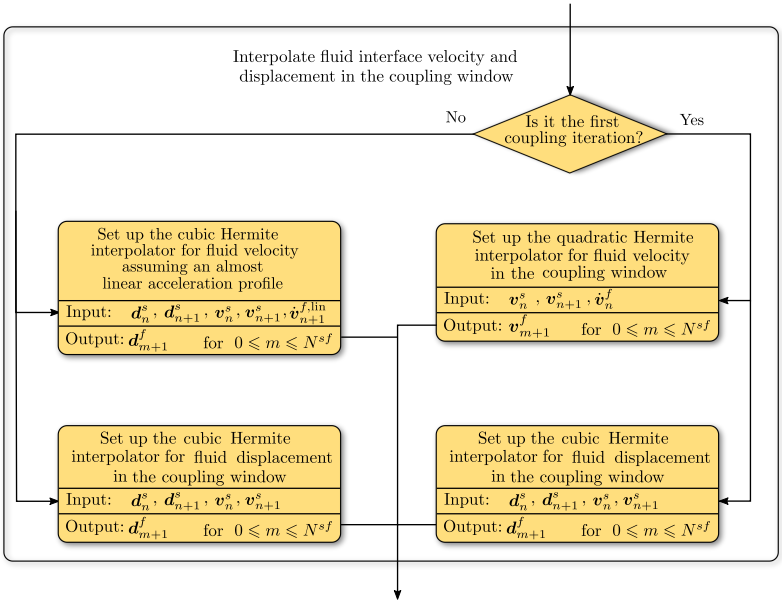


Figure 5.16: The flowchart of the interpolation steps required for the proposed fluid subcycling scheme

time step, i.e., $\boldsymbol{\lambda}^f_m$ and $\boldsymbol{\lambda}^f_{m+1}$. The local integrated interface force is added up on the fly as the fluid solver marches through the coupling window and builds up the total time integral of the fluid interface force required for the projection onto the solid domain. By doing so, we avoid storing the fluid interface values at every fine time step for processing at the end of the coupling window. Next, the fluid interface force is projected onto the solid domain by calculating $\boldsymbol{\lambda}^f_{n+1}$ using Eq. 5.10, and the solid problem is solved in the coarse time step n resulting in $\tilde{\boldsymbol{v}}^s_{n+1}$. Finally, the convergence of the interface velocities is checked by comparing $\tilde{\boldsymbol{v}}^s_{n+1}$ to $\boldsymbol{v}^s_{n+1}$. If the desired convergence is not achieved, a new coupling iteration is started by resetting the fluid solver back to the checkpoint state and predicting a new iterate for $\boldsymbol{v}^s_{n+1}$. Note that the interface quasi-Newton method is set up using the iterate values of velocity at the coarse time scale.

5.3.2 Structure subcycling scheme

This section investigates the structure subcycling scheme for the FSI problems with finer temporal discretization in the solid domain. The difference in the time step sizes in the structure and fluid domains could be due to the different time scales, the specific requirements of certain time integration schemes

(e.g., explicit schemes), or the convergence requirements in the employed nonlinear solvers. Similar to the fluid subcycling schemes in section 5.3.1, the coupling window is equal to the largest time step of the subproblems, i.e., the fluid problem in this case. The time grids of the one coupling window are schematically shown in Fig. 5.17. The time indices of the coarse and fine time steps are indicated using n and m, respectively. The coupling between the fluid

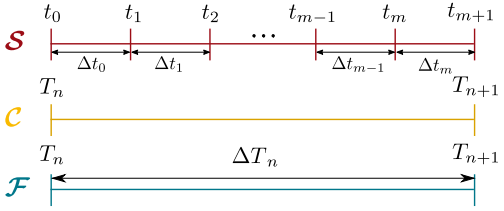


Figure 5.17: From top: the structure, coupling and fluid time grids in the structure subcycling scheme

and solid domains is carried out by iteratively exchanging information within each coupling window. The interface quasi-Newton method is formulated pointwise in time at the discrete time instants of the coupling window. For the sake of simplicity, it is assumed here that the subproblems have constant time step sizes and the number of the fine time steps $N^{f,s}$ in each coupling window is an integer number, i.e., $N^{f,s} = \frac{\Delta T^f}{\Delta t^s}$.

The combination of Dirichlet-Neumann splitting of the FSI problem and structure subcycling indicates that the fluid traction values on the FSI interface can be interpolated within the course time step to derive the solid traction values at the fine time steps. This interpolation might seem similar to the interpolation of velocity values from the coarse to fine time steps in the fluid subcycling scheme. However, accurate information about interface traction's time derivative is unavailable. Therefore, it is not possible to construct a higher order (Hermite) interpolation function for the interface traction using the information available within a coarse time step. We make the assumption that the interface traction varies linearly within a coarse time step. Therefore, the transfer of interface traction from the fluid to the solid domain is carried out using linear interpolation within a coarse time step. The same approach for interpolating interface forces is investigated in [177] for the strongly coupled FSI problems with structure subcycling as well. Furthermore, a similar assumption is made in some FETI-based asynchronous coupling methods in structural mechanics. The implications of the linear interpolation (in time) of the interface traction will be discussed with the help of numerical examples.

As previously discussed, a common approach for dealing with the constraints on the kinematic variables at the FSI interface is to strongly satisfy

the discrete displacement continuity condition, Eq. 2.233. The discrete fluid velocity value at the FSI interface follows the evolution of the interface displacement with the help of the consistent time integration rules employed in the fluid domain. This approach is also used in [177] for handling the interface kinematic constraints in FSI simulations with structure subcycling. Using this approach, the discrete velocity continuity condition, Eq. 2.235, is not guaranteed to be exactly satisfied at the end of the coupling window. In the following, the effects of this structure subcycling scheme on the partitioned simulation of our model problem will be investigated. The same physical and numerical parameters are used for the model problem as in section 5.3.1, except that the fine time steps are used in the solid domain. Each natural period of oscillation of the system is discretized using 10 coarse time steps, and a time step ratio of $N^{f,s} = 100$ is used in the structure subcycling scheme. The simulation results of the model problem using the structure subcycling scheme based on the satisfaction of the discrete displacement continuity condition, Eq. 2.233, at the coarse time level are plotted in Fig. 5.18. The

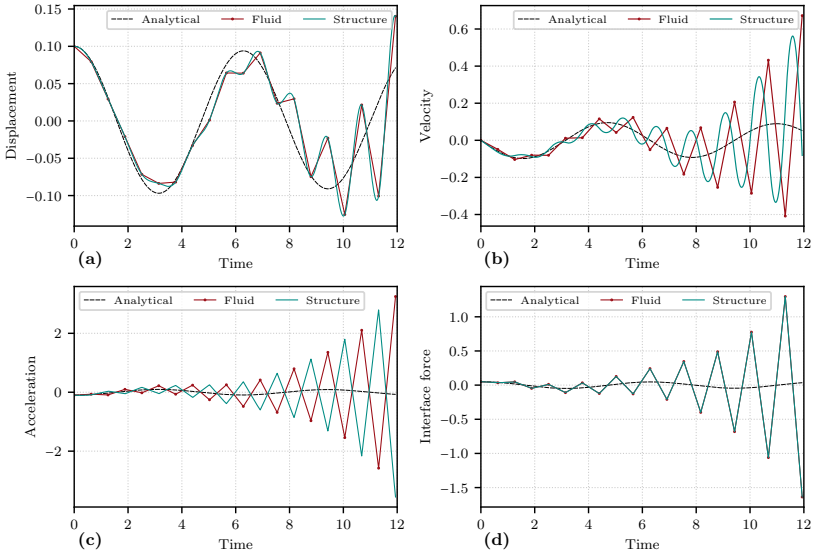


Figure 5.18: Model problem: structure subcycling results based on satisfying the discrete displacement continuity condition

first and most important observation in Fig. 5.18 is that the simulation with the aforementioned structure subcycling scheme is unstable and begins to diverge right from the beginning, even for the present linear model problem. This observation is in line with the analytical and numerical studies of model problems in [177], where it is concluded that the simulations using structure

subcycling and two considered combinations of time integration schemes are unstable for a time step ratio of $N^{f,s} \geqslant 3$.

The critical factor leading to the unstable simulation results in Fig. 5.18 is the choice of strongly satisfying (only) the discrete displacement continuity condition, Eq. 2.233, at the coarse time level. In this approach, the fluid displacement at the coarse time level is determined from the solid displacement value at the end of each coupling window. The displacement graphs in Fig. 5.18a confirm that this goal is achieved in numerical results, despite the growing oscillations in the solution. However, the fluid velocity at the coarse time scale, which is consistently derived from the fluid displacement, does not immediately respond to the change of velocity in the solid domain at the fine time scale and lags behind (see Fig. 5.18b). Subsequently, the fluid and solid acceleration graphs in Fig. 5.18c evolve with (almost) opposite phase. The solid interface forces in Fig. 5.18b are linearly interpolated from the fluid forces, which are directly determined by the fluid acceleration. This faulty cycle is repeated and leads to diverging oscillations in the solution. More numerical insight about the unstable solutions using the structure subcycling scheme is provided in [177], where the instability is attributed to the maximum amplitude of eigenvalues of the amplification matrix being greater than one for two different combinations of time integration schemes in a model problem.

Alternatively, the structure subcycling algorithm in this work is formulated based on satisfying the discrete velocity continuity condition, Eq. 2.235, at the coarse time scale in conjunction with the same (temporal) linear interpolation of the interface forces. The simulation results of the model problem using the proposed structure subcycling scheme are plotted in Fig. 5.19. The results using the subcycling scheme based on the discrete velocity continuity condition are stable. By comparing the results in Fig. 5.19 to those using the fluid subcycling scheme presented in Fig. 5.9, it is evident that the structure subcycling leads to a more accurate solution of the system, which is expected as the model structure contains the stiffness term while the model fluid problem contains the damping term. The solid acceleration curve in Fig. 5.19c is not smooth at the boundaries of the coupling windows due to relatively large coarse time steps combined with the assumed linear evolution of the interface force within a coarse time step. Therefore, a key consideration in using the structure subcycling scheme is whether the assumed linear profile for the interface force is reasonable in combination with the chosen coarse time step size. Another issue to point out is the drift in the displacement constraint observed in Fig. 5.19, which is also directly influenced by the size of the coarse time step and is further exacerbated by choice of $\rho^s_\infty = \rho^f_\infty = 0$.

In order to study the effect of the coarse time step size on the results using the structure subcycling scheme, three simulations of the model problem with the same fine time step size of $\Delta t^s = P/1000$ and different coarse time step sizes of $\Delta T^f = P/10$, $\Delta T^f = P/20$ and $\Delta T^f = P/40$ are carried out. The comparison of the acceleration graphs presented in Fig. 5.20 reveals that the solid acceleration curve becomes more smooth as the coarse time step size

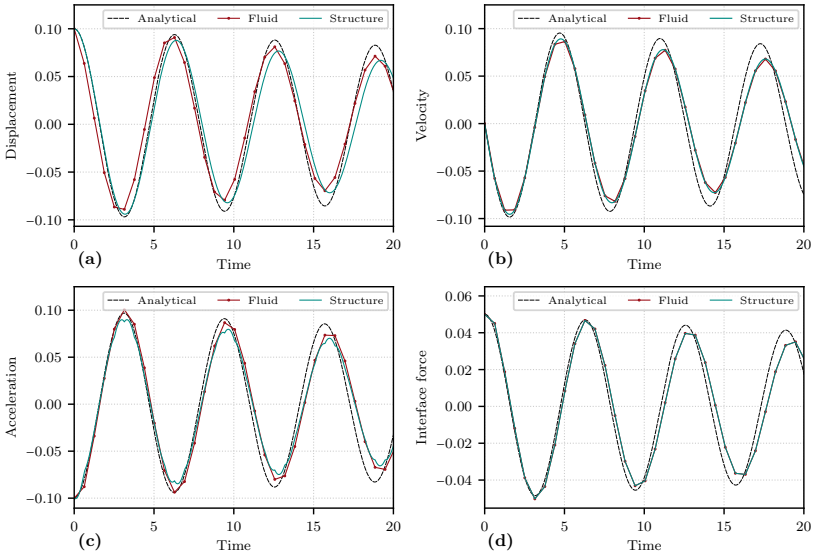


Figure 5.19: Model problem: structure subcycling results based on satisfying the discrete velocity continuity condition

decreases. The displacement values from the three simulations are compared in Fig. 5.21. It is observed that the drift in the displacement constraint is reduced by decreasing the coarse time step size.

Another factor affecting the drift in the displacement constraint is the ρ_∞^s parameter of the G-α time integration scheme in the solid domain. The model problem's solid and fluid displacement graphs are plotted in Fig. 5.22 for three different values of ρ_∞^s. The drift in the displacement constraint increases as the ρ_∞^s parameter is reduced, mainly due to a bigger phase shift between the solid and fluid displacement curves. Recall that a similar trend was also observed for the fluid subcycling scheme. Therefore, it seems logical to increase the ρ_∞^s value in order to control the drift in the displacement constraint, as a result of which the introduced high-frequency damping by the G-α scheme is reduced. However, one should be careful not to remove the high-frequency damping, mainly because of the numerical artifacts introduced in the solid acceleration due to the assumed (temporal) profile of the interface forces in the structure subcycling scheme. Fig. 5.23 shows the long-term response of the model problem using the structure subcycling scheme for the three different values of ρ_∞^s. For $\rho_\infty^s = 1.0$, the solution undergoes nonphysical oscillations, which grow in amplitude as time goes on. This behavior is mainly the accumulation of errors in the solid acceleration and is best observed in the acceleration and interface force graphs in Fig. 5.23c and 5.23d, respectively. The simulation is stable for $\rho_\infty^s = 0.0$, but the results are too diffusive. A good

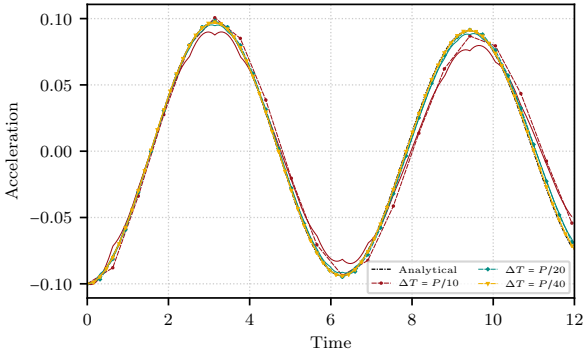


Figure 5.20: Model problem: the effect of the coarse time step size on the solid acceleration profile in the structure subcycling scheme based on satisfying the discrete velocity continuity condition. The solid and dashed lines depict the solid and fluid graphs respectively

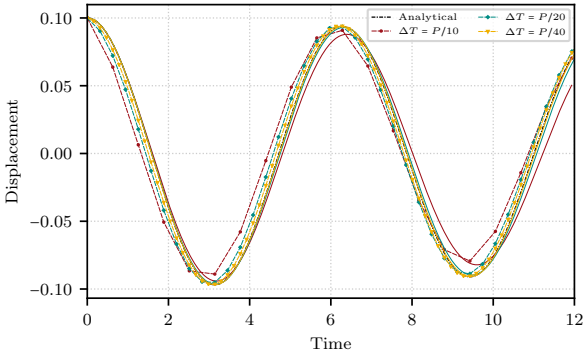


Figure 5.21: Model problem: the effect of the coarse time step size on the drift in the displacement constraint in the structure subcycling scheme based on satisfying the discrete velocity continuity condition. The solid and dashed lines depict the solid and fluid graphs respectively

compromise is achieved by using $\rho_\infty^s = 0.5$.

Similar to the fluid subcycling case, the preferred approach in this work for dealing with the discrete kinematic coupling conditions in FSI simulations is to satisfy the discrete velocity continuity condition, 2.235, and subsequently derive the fluid interface displacement consistently using the time integration rules in the fluid domain. Doing so reduces the amount of transferring and managing data between the structure and fluid solver in a partitioned simulation. By choosing a sensible value for ρ_∞ (e.g., $0.4 \leqslant \rho_\infty \leqslant 0.9$) in the G-$\alpha$ time integration scheme and the coarse time step size such that the temporal

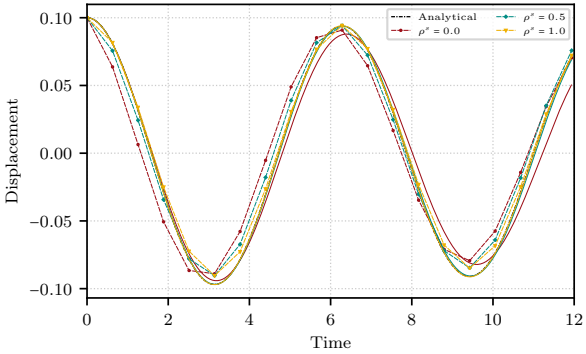


Figure 5.22: Model problem: the effect of ρ^s_∞ on the drift in the displacement constraint in the structure subcycling scheme based on satisfying the discrete velocity continuity condition. The solid and dashed lines depict the solid and fluid graphs respectively

evolution of the solution is reasonably resolved, the drift in the displacement constraint will be minimal in most cases. Consistently deriving the fluid interface displacement using the time integration rules in the fluid domain is particularly advantageous for satisfying the discrete geometric conservation law in the fluid domain. If the drift in the displacement constraint is deemed unacceptably large, one can satisfy both discrete displacement and velocity coupling constraints, Eqs. 2.233 and 2.235, respectively. As discussed in the fluid subcycling case as well, the kinematics of the interface will then be determined by the structure solver, i.e., the interface velocity and displacement at the coarse time scale will be consistent with respect to the time integration scheme of the solid domain but not necessarily with respect to the time integration scheme in the fluid domain.

5.4 FSI Results and discussion

5.4.1 Transverse galloping of a square body

Let us first present the motivation behind choosing the current test case. A large body of research has been devoted to the numerical FSI simulation of vortex-induced oscillation of bodies in flows. The vortex-induced oscillation is characterized by the so-called *lock-in* effect, which is the result of the structure's natural frequency almost coinciding with the flow's vortex shedding frequency, leading to bigger amplitudes of oscillations. With regard to the temporal resolution of the discrete FSI problem, a similar time step size for the fluid and solid domains, which adequately resolves the period of oscillation of the vortex shedding and the structure's oscillation, is sufficient in order to capture the vortex-induced oscillation phenomenon. However, there are other

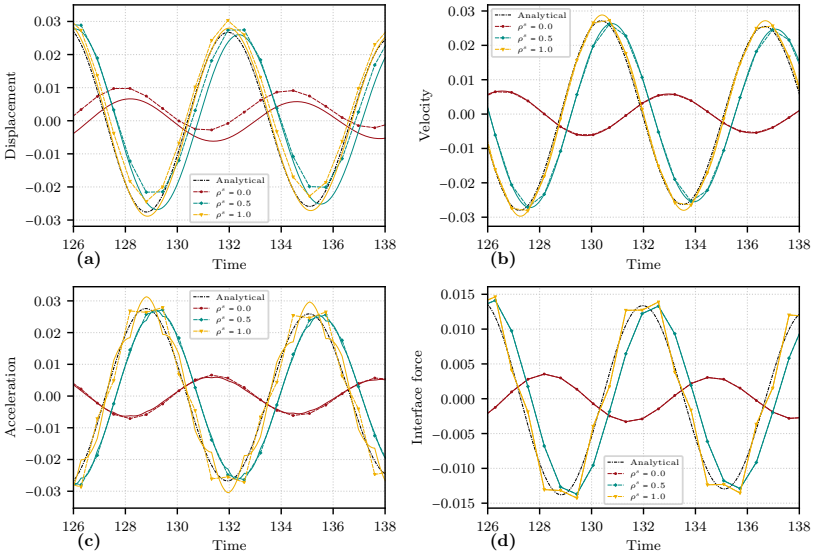


Figure 5.23: Model problem: the effect of ρ^s_∞ on the long-term structure subcycling results based on satisfying the discrete velocity continuity condition. The solid and dashed lines depict the solid and fluid graphs respectively

types of flow-induced vibrations of structures, which occur at frequencies much lower than that of the vortex shedding in the flow [72]. One such phenomenon is the so-called *galloping* of bodies (with a single degree of freedom) immersed in flows. The numerical FSI simulation of traverse galloping of a square body is carried out in, e.g., [215, 72] using matching time step sizes for the fluid and solid domains. This test case is suited for studying the effects of subcycling in the fluid domain on the accuracy of FSI simulations. Since the period of the flow's vortex shedding is much smaller than that of the structural oscillations, a finer time step size is required in the fluid domain in order to resolve the flow's vortex shedding adequately, while the structure's vibrations can be accurately simulated using bigger time steps.

The numerical test case consists of a square body immersed in a 2d cross flow. The body is supported by an elastic spring and a damper with a small damping coefficient such that it oscillates perpendicularly with respect to the flow direction. The geometrical and physical properties of the problem are taken from [72]. The fluid density and dynamic viscosity are $\rho = 1.0$ and $\mu = 0.01$, respectively. The inflow velocity is $v_\infty = 2.5$, which results in a Reynolds number $Re = v_\infty D\rho/\mu = 250$, with $D = 1$ being the side length of the square body. The rigid body has a mass $m = 20$ and is supported by a spring with stiffness $k = 3.08425$ and a damper with the coefficient $c = 0.0581195$. The geometry of the numerical domain is sketched in Fig. 5.24. The fluid domain

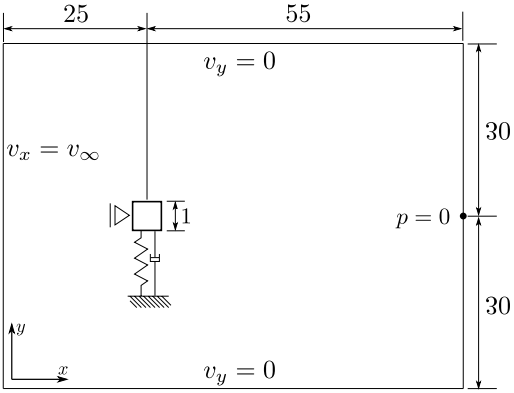


Figure 5.24: Traverse galloping of square body: the computational domain.

is discretized using 11354 triangular stabilized finite elements with linear Lagrange shape functions and based on a variational multiscale formulation. The fluid computational mesh around the square body is depicted in Fig. 5.25.

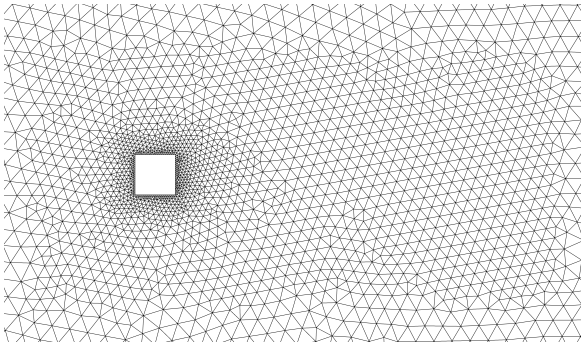


Figure 5.25: Traverse galloping of square body: the fluid's spatial discretization around the square body.

The fluid and solid problems are discretized in time using the G-α scheme for first-order systems with the high-frequency numerical damping determined by $\rho^s_\infty = 0.6$ and $\rho^f_\infty = 0.5$ in the solid and fluid domains, respectively. We are interested in choosing the time step size in the solid and fluid domains based on the local temporal accuracy requirements in each subproblem, determined in the present case by the natural frequency of the structural problem and the flow's vortex shedding frequency. The natural frequency and the corresponding period of oscillation of the structural problem read $f_n = 0.0625$ and $T_n = 16$,

respectively. We aim to discretize each period of structural oscillation using about 20 time steps, leading to a time step size of $\Delta t^s = 0.8$ for the solid problem. On the other hand, the vortex shedding frequency of the flow and its corresponding period of oscillation is approximately $f_v \approx 0.398$ and $T_v \approx 2.512$, respectively. The fluid problem is discretized in time using the time step size $\Delta t^f = 0.04$, leading to almost 63 time steps in each period which suffices for an adequate temporal resolution of each vortex shedding period of the flow. The solid-to-fluid time step ratio is $N^{s,f} = 20$. Note that the time step sizes are chosen solely based on the minimum requirements of capturing the essential physical behaviors in the subproblems.

The results of the FSI simulation with fluid subcycling are presented in Fig. 5.26. Both the fluid and solid quantities (with different time scales) on the FSI interface are plotted in Fig. 5.26. Note that the square rigid body

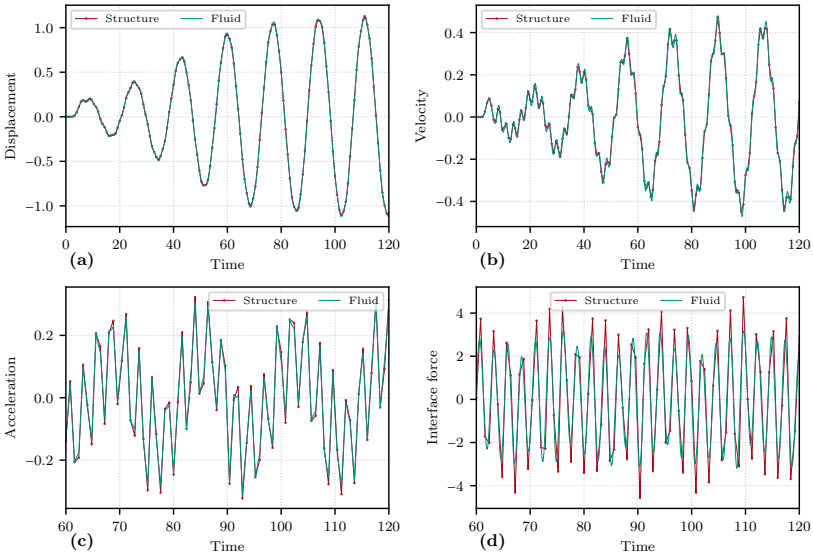


Figure 5.26: Traverse galloping of square body: fluid subcycling results using Hermite interpolation of velocity values and almost linear fluid acceleration assumption

is allowed to move only in the vertical direction. In the present numerical example, the fluid interface displacement is calculated from the fluid interface velocity using formulae consistent with the time integration scheme in the fluid domain. In spite of that, the fluid and solid displacement values on the interface are very close at the coarse time level.The time evolution of the solid and fluid interface velocities plotted in Fig. 5.26b reveals the significance of the Hermite interpolation used for the velocity values at the fine time scale.

The fine details in the fluid interface velocity curve would not have been captured if no derivative information had been leveraged. In fig. 5.26c, the fluid and solid interface acceleration values are plotted over time. For better clarity, the plot of the second half of the time period is presented. The fluid interface acceleration evolves almost linearly within most of the coarse time steps, which is consistent with the assumptions made regarding the temporal profile of the acceleration. The solid and fluid interface forces are compared in Fig. 5.26d. In contrast to the model problem, the fluid interface force does not evolve almost linearly within a coarse time step since the fluid interface reaction also depends on the gradient of velocities and pressure when solving INS equations. The fluid interface forces (at the fine time scale) are projected to the solid interface (at the coarse time level) according to Eq 5.10. Note in Fig. 5.26d that the projection does not preserve the extrema of the fluid interface force curve, which is a characteristic of the projection operator, especially when the time step size in the target domain does not allow to reasonably capture the profile of the fluid interface force. This issue is not very pronounced in the current example, although a slightly smaller time step size in the solid domain would allow a more accurate projection of the fluid interface force.

In order to better analyze the effect of fluid subcycling in the present test case, a comparison is carried out between the results of the simulation with subcycling and those of the simulation with matching time steps for fluid and solid domains. Two simulations with matching time steps are carried out: One with the fine time step size $\Delta t^s = \Delta t^f = 0.04$ and another with the coarse time step size $\Delta t^s = \Delta t^f = 0.8$. The time evolution of the interface displacement from different simulations is plotted in Fig. 5.27. For the simulations with non-matching time steps, both fluid and solid quantities are plotted. The FSI simulation with the matching coarse time step size $\Delta t^s = \Delta t^f = 0.8$ leads to a displacement history that is almost 50 percent different than the results with the fine time step size $\Delta t^s = \Delta t^f = 0.04$ when comparing the extrema values. Additionally, there is a substantial period elongation in the displacement curve when using the coarse time step size for both solid and fluid domains. In contrast, when using subcycling in the flow solver, with $\Delta t^s = 0.8$ and $\Delta t^f = 0.04$, the fluid and solid displacement curves are very similar to the curve from the simulation with matching fine step sizes. There is only a small phase shift due to the effect of high-frequency damping in the solid domain by setting $\rho^s_\infty = 0.6$, which is more pronounced when using $\Delta t^s = 0.8$ compared to $\Delta t^s = 0.04$.

In Fig. 5.28, the velocity values from the three simulations are compared. While the results from the simulation with fluid subcycling, $\Delta t^s = 0.8$ and $\Delta t^f = 0.04$, and the simulation with matching fine time steps, $\Delta t^s = \Delta t^f = 0.04$, capture the small fluctuations in the velocity over time, the velocity values from the simulation with matching coarse time steps, $\Delta t^s = \Delta t^f = 0.8$, are smeared out and are also different in the values of extrema.

The comparison of the acceleration values from the three simulations is

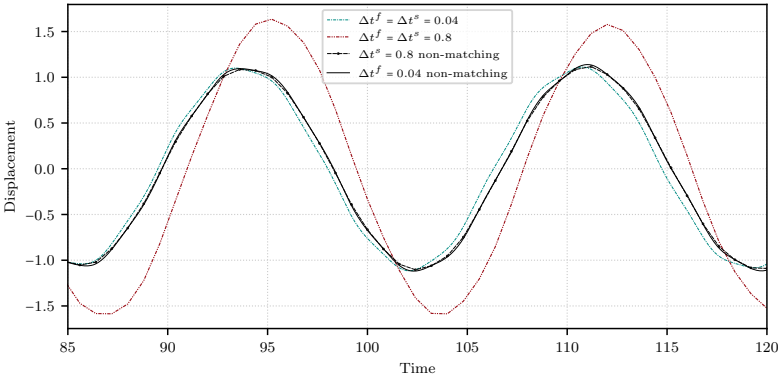


Figure 5.27: Traverse galloping of square body: comparison of displacement from fluid subcycling scheme to those from simulations with fine and coarse matching time steps. For the subcycling scheme, both the fluid and solid displacement graphs are plotted using solid and dashed lines, respectively

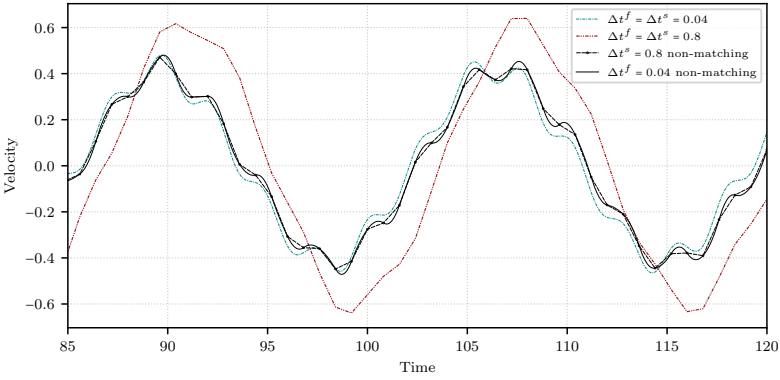


Figure 5.28: Traverse galloping of square body: comparison of velocity from fluid subcycling scheme to those from simulations with fine and coarse matching time steps. For the subcycling scheme, both the fluid and solid velocity graphs are plotted using solid and dashed lines, respectively

carried out in Fig. 5.29. The acceleration curve from the simulation with matching coarse time steps, $\Delta t^s = \Delta t^f = 0.8$, does not possess the (physical) high-frequency fluctuations. In contrast, the high-frequency acceleration fluctuations from the simulation with fluid subcycling, $\Delta t^s = 0.8$ and $\Delta t^f = 0.04$, match very well with the simulation with matching fine time steps. However, the assumption made on the acceleration profile being almost linear within a coarse time step is clearly reflected in the results from the simulation

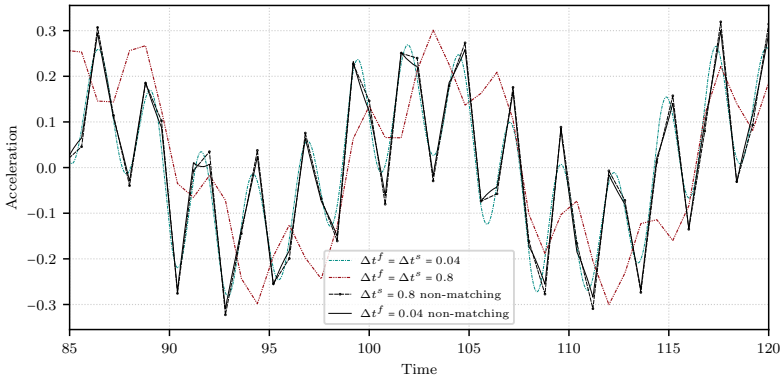


Figure 5.29: Traverse galloping of square body: comparison of acceleration from fluid subcycling scheme to those from simulations with fine and coarse matching time steps. For the subcycling scheme, both the fluid and solid acceleration graphs are plotted using solid and dashed lines, respectively

with non-matching time steps, while the acceleration curve from the simulation with matching fine time steps is smooth.

The interface forces from different simulations are compared in Fig. 5.30. The fluid interface force from the simulation with fluid subcycling evolves very similarly to the interface force from the simulation with matching fine time steps. This observation is consistent with our expectation since the fluid time step size in the problem with not-matching time steps was chosen to resolve the vortex shedding in the flow adequately. The projection of the fluid interface force onto the solid domain (using linear functions in time) can reasonably capture the fluctuations in the interface force, with the caveat that there exist slight overshoots in the extrema values, as previously discussed. In contrast, the interface forces are not captured correctly in the simulation with matching coarse time steps since the flow is not adequately resolved in time. This result was anticipated from analyzing the flow's vortex shedding frequency and natural frequency of the solid problem. The presented coupling methodology for FSI problems with non-matching time steps proves to be accurate and robust enough to allow reproducing the anticipated results in numerical simulations.

For comparison, the results of fluid subcycling using two other fluid subcycling schemes studied in section 5.3.1 will also be presented. The results using a cubic interpolation for displacement (as proposed in [177]) together with the projection of the interface forces are presented in Fig. 5.31. Similar to the model problem, the continuity of the fluid interface velocity across the boundaries of coarse time steps is lost (due to the construction of the interpolation function for displacements) which leads to big spikes in the graph of the fluid interface acceleration and interface force.

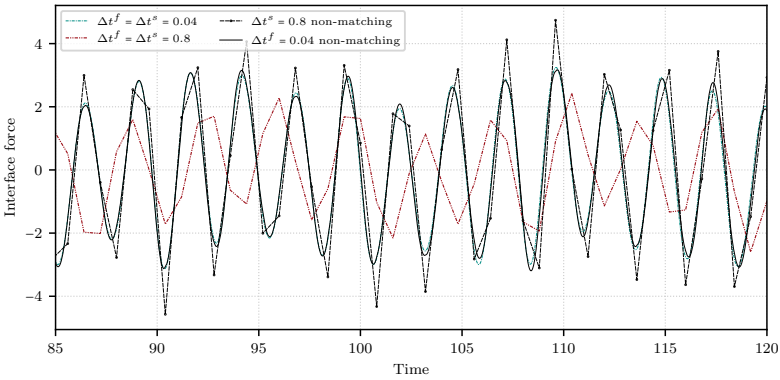


Figure 5.30: Traverse galloping of square body: comparison of interface traction values from fluid subcycling scheme to those from simulations with fine and coarse matching time steps. For the subcycling scheme, both the fluid and solid interface traction graphs are plotted using solid and dashed lines, respectively

The results of fluid subcycling using a linear interpolation of velocities and the projection of the interface forces are presented in Fig. 5.32. The fine details in the fluid interface velocity are not captured due to the linear interpolation. As expected, the fluid acceleration graph overshoots at the beginning of each coupling window. Consequently, the fluid interface force also has discontinuities across the boundary of the coarse time steps, which are more clearly visible in Fig. 5.33.

5.4.2 Flow-induced oscillation of a flexible beam in the wake of a square bluff body

In this benchmark problem, a flexible beam attached to a square bluff body is submerged in incompressible fluid flow. Due to the interaction with the wake of the square body, the flexible beam starts to oscillate. For the detailed description of the problem and setup of the simulation, see section 3.7.2. In this section, an absolute convergence criterion with the absolute tolerances of $\varepsilon_{abs} = 10^{-6}$ is used to assess the convergence of the coupling iterations. Both fluid and solid problems are discretized in time using the G-α scheme for first-order systems with the high-frequency numerical damping determined by $\rho^s_\infty = \rho^f_\infty = 0.5$.

In this example, the effects of both fluid and structure subcycling on the FSI simulation are studied. Furthermore, higher ratios of time step sizes will be used. For the simulation with fluid subcycling, the time step sizes are chosen as $\Delta t^s = 1e^{-2}$ and $\Delta t^f = 5e^{-5}$, leading to a time step ratio of $N^{s,f} = 200$. In the simulation with structure subcycling, the time step sizes are

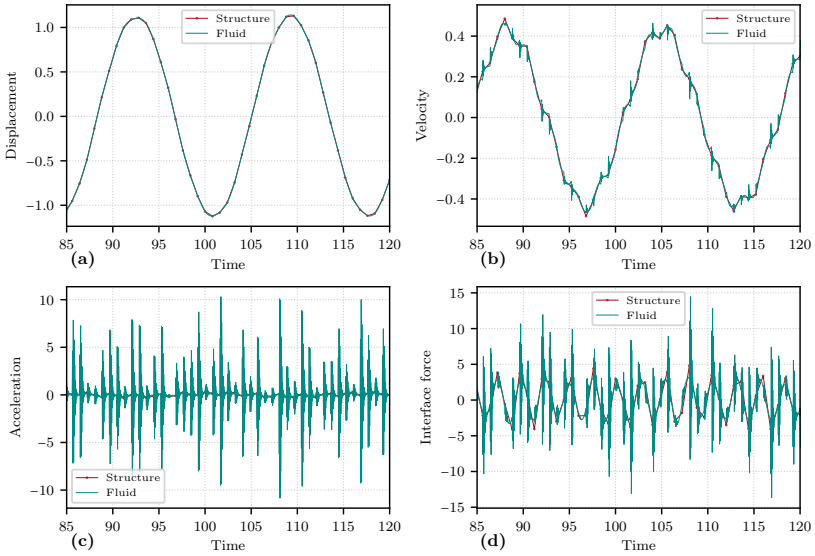


Figure 5.31: Traverse galloping of square body: fluid subcycling results using cubic interpolation of displacement values

chosen as $\Delta t^s = 5e^{-5}$ and $\Delta t^f = 1e^{-2}$ with a time step ratio of $N^{f,s} = 200$. For comparison, two FSI simulations with matching time steps of $\Delta t^s = \Delta t^f = 5e^{-5}$ and $\Delta t^s = \Delta t^f = 1e^{-2}$ are also carried out. The simulation with matching fine time step will be used as the reference for further comparisons. By comparing the results from the fluid and structure subcycling schemes with high time step ratios, we aim to identify the role of accurately resolving the fluid forces and capturing the kinematic behavior of the structure in the accuracy of the FSI simulations. For better clarity, the results from a time interval are presented in which all the simulations are developed. All the results presented in this section are values in the vertical direction of nodal quantities of a point located at the tip of the flexible beam.

The evolution of interface traction from the four simulations is plotted in Fig. 5.34. For the simulations with subcycling, both fluid and solid values are plotted. The quantities at the fine and coarse time scales are plotted as solid and dashed lines, respectively. Barring some fine details, the pattern of the interface traction graph from the simulation with fluid subcycling is similar to that from the simulation with matching fine time steps. There is also a phase shift between the two graphs, partly due to the introduced high-frequency numerical damping in the solid domain combined with a rather large time step size. These observations suggest that, in the current example, the more important factor in accurately capturing the interface traction is the adequate temporal resolution of the flow. However, the fine details in the traction graph

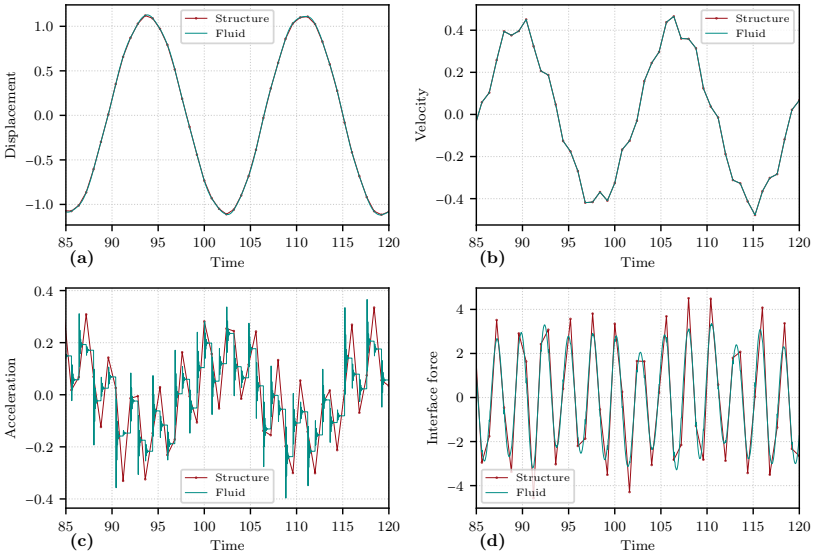


Figure 5.32: Traverse galloping of square body: fluid subcycling results using linear interpolation of velocity values

from the simulation with matching fine time steps can be attributed to the fine temporal resolution of the structure's kinematic behavior, which is not well captured in the simulation with fluid subcycling and large time step ratio. It is also worth noticing that the (temporal) projection of fluid traction onto the structure performs very well in the fluid subcycling scheme. It is observed that the traction graphs from the simulations with structure subcycling and matching coarse time steps are quite similar but different from the reference simulation with matching fine time steps. This observation supports the prior argument that resolving the flow structure properly has the biggest impact on the accuracy of the interface traction in the current example.

The acceleration values from the four simulations are plotted in Fig. 5.35. Here is where the effect of finely resolving the kinematic behavior of the structure is most pronounced. The acceleration graph from the simulation with structure subcycling and that of the simulation with matching coarse time steps have similar phase shifts with respect to the reference results with matching fine time steps. However, the structure subcycling allows capturing the fluctuations in the evolution of the interface acceleration. Recall that the simulations with the coarse time step size in the fluid domain results in similar interface traction. Therefore, capturing the acceleration fluctuations can be attributed to properly resolving higher modes in the oscillation of the structure. The pattern of those acceleration fluctuations in the simulations with structure subcycling is similar to that from the reference simulation with

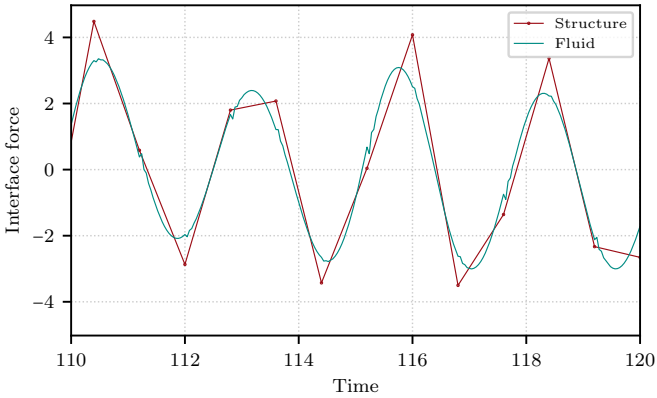


Figure 5.33: Traverse galloping of square body: details of the interface traction from the fluid subcycling scheme using linear interpolation of velocity values

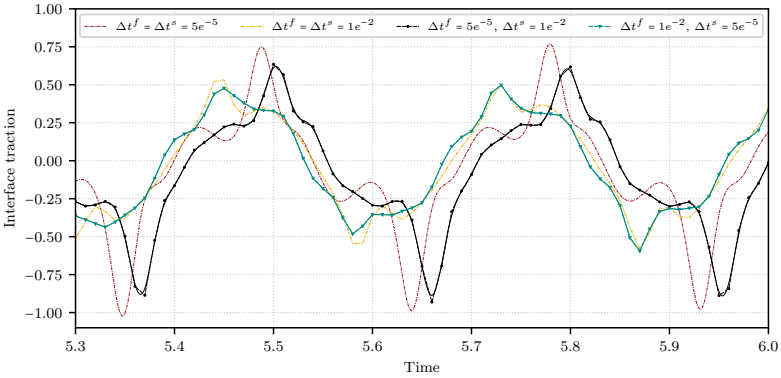


Figure 5.34: Wall Benchmark: comparison of interface traction values from fluid and structure subcycling schemes to those from simulations with fine and coarse matching time steps. For the subcycling schemes, both the fine and coarse time level values are plotted using solid and dashed lines, respectively

matching fine time steps. There are, however, very small (non-physical) high-frequency oscillations in the solid acceleration's graph from the simulation with structure subcycling compared to the reference results. Those high-frequency fluctuations are artifacts due to the assumption in the interpolation of fluid interface traction (at coarse time scale) to the structure (at fine time scale). This issue is magnified by the fact that the structure in the current example

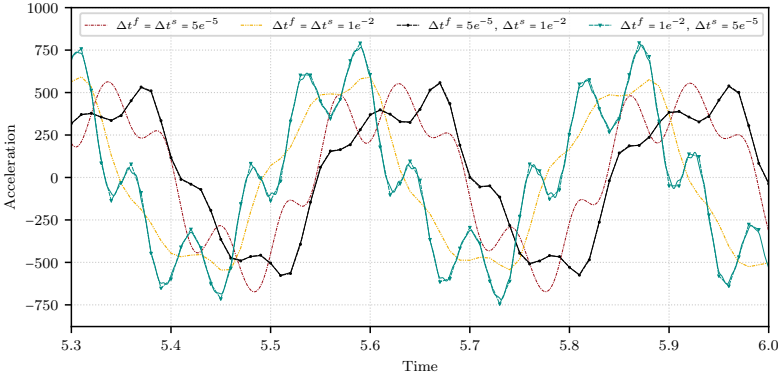


Figure 5.35: Wall Benchmark: comparison of acceleration values from fluid and structure subcycling schemes to those from simulations with fine and coarse matching time steps. For the subcycling schemes, both the fine and coarse time level values are plotted using solid and dashed lines, respectively

possesses no structural damping. In the simulation with matching fine time step, the fluid's viscous forces act as damping on the structure at every fine time step. However, The fluid forces are interpolated within a coarse time step when doing structure subcycling which leads to the fluid damping effects on the structure being affected by the interpolation assumptions. Nevertheless, the small numerical artifacts in the structure acceleration do not significantly impact the system's general response in this example. The acceleration graph from the simulation with fluid subcycling is closer to the reference graph from the simulation with matching fine time step in terms of the phase shift and extrema values but does not quite accurately capture the higher frequency acceleration fluctuations, which were argued to be due to the higher modes of the structure. Note also the almost linear temporal profile of the fluid acceleration within each coarse time step when using fluid subcycling, making it almost indistinguishable from the solid acceleration graph.

The velocity values from the four simulations are compared in Fig. 5.36. The evolution of velocity values from each simulation is consistent with the earlier arguments regarding the role of fluid and structure subcycling in the present example. The graph from the simulation with fluid subcycling is closer to the reference graph with respect to the phase shift and amplitude. On the other hand, the graph from the simulation with structure subcycling better captures the higher modes of velocity oscillation.

The comparison of the displacement histories presented in Fig. 5.37 confirms the observations made earlier that, in the present example, resolving the flow vortices has the most impact on the general accuracy of the FSI simulation. It is also worth mentioning that for both simulations with fluid and structure subcycling, the fluid displacement on the FSI interface is determined

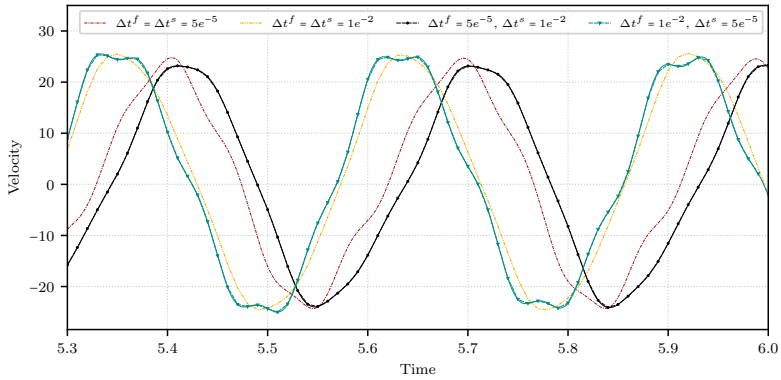


Figure 5.36: Wall Benchmark: comparison of velocity values from fluid and structure subcycling schemes to those from simulations with fine and coarse matching time steps. For the subcycling schemes, both the fine and coarse time level values are plotted using solid and dashed lines, respectively

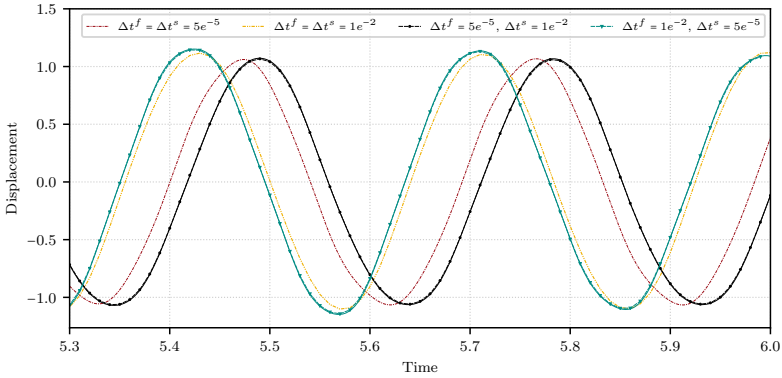


Figure 5.37: Wall Benchmark: comparison of displacement values from fluid and structure subcycling schemes to those from simulations with fine and coarse matching time steps. For the subcycling schemes, both the fine and coarse time level values are plotted using solid and dashed lines, respectively

using the fluid velocity and the time integration scheme in the fluid domain. The drift in displacement constraint is quite small in both cases, such that the fluid and solid displacement graphs are almost indistinguishable in the simulations with subcycling.

In the following, the results from two other fluid subcycling schemes and the structure subcycling scheme based on the discrete displacement continuity condition are presented for comparison. In Fig. 5.38, the results from the fluid

subcycling scheme with cubic interpolation for displacements are presented. The fluid velocity graph possesses discontinuities at the boundaries of the

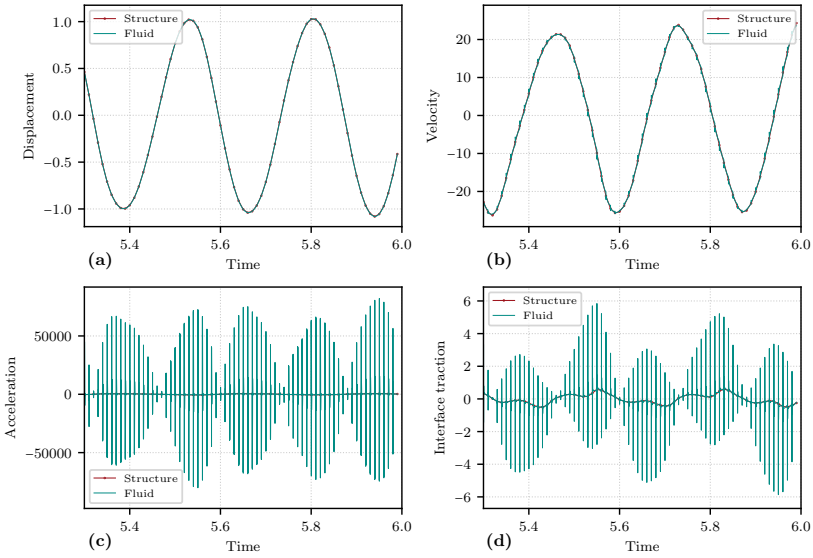


Figure 5.38: Wall benchmark: fluid subcycling results using cubic interpolation of displacement values

coarse time steps, leading to considerable nonphysical oscillations in the fluid acceleration that are also reflected in the interface force graph.

The results using the fluid subcycling scheme with linear interpolation of velocities are illustrated in Fig. 5.39. The general quality of the solution using linear interpolation of velocities seems acceptable in the current example. The chosen coarse time step size is small enough such that a linear profile of velocity within a coarse time step sufficiently resolves the evolution of the velocity in Fig. 5.39b. However, the fluid acceleration curve overshoots at the beginning of each coarse time step and leads to a nonsmooth profile of the fluid interface force in Fig. 5.39d.

The results using the structure subcycling scheme with the discrete displacement continuity condition in conjunction with the linear interpolation of interface forces are presented in Fig. 5.40. As expected, the simulation is unstable and diverges right at the beginning (see section 5.3.2 for an explanation of this behavior), which highlights the importance of satisfying the discrete velocity continuity condition in the proposed structure subcycling scheme based on the Dirichlet-Neumann decomposition.

This section concludes that the presented subcycling schemes can be

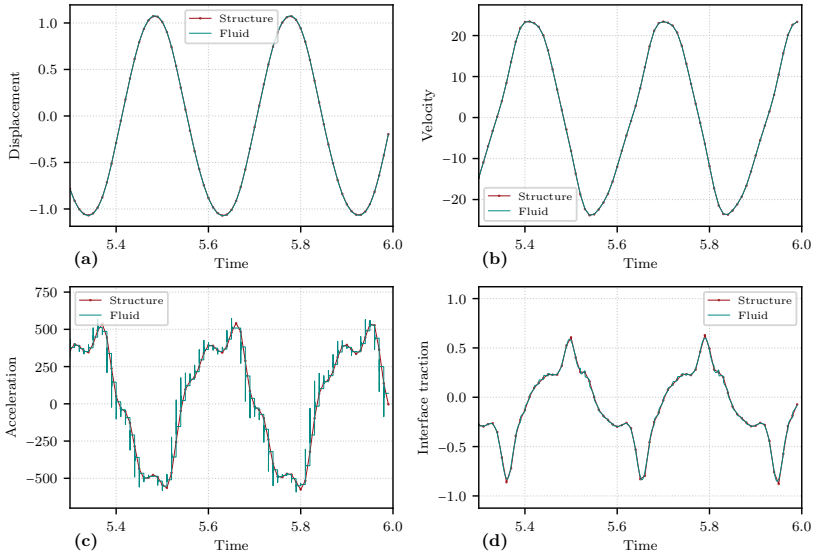


Figure 5.39: Wall benchmark: fluid subcycling results using linear interpolation of velocity values

robustly used with high time step ratios without compromising the accuracy of the numerical results. Furthermore, the precise effect of resolving the flow vortices and the higher structural modes on the accuracy of the FSI simulations are studied using the presented schemes.

5.4.3 Driven cavity with flexible bottom

This benchmark problem consists of a square cavity with a flexible bottom. The flow inside the cavity is driven by a prescribed periodical velocity at the top boundary. The problem definition and the numerical parameters are described in section 4.3.1. Here, a convergence criterion with the absolute tolerances of $\varepsilon_{abs} = 10^{-6}$ is used to check the convergence of the coupling iterations. Both fluid and solid problems are discretized in time using the G-α scheme for first-order systems with the high-frequency numerical damping determined by $\rho^s_\infty = \rho^f_\infty = 0.5$.

Similar to the studies in section 5.4.2, the effects of structure and fluid subcycling on the FSI simulation are investigated. Furthermore, the present example provides a more demanding challenge with regard to satisfying the incompressibility constraint in the INS equations and its effect on the pressure distribution in the domain. For the simulation with structure subcycling, the time step sizes are chosen as $\Delta t^s = 0.005$ and $\Delta t^f = 0.2$ with time step ratio of $N^{f,s} = 40$. In the simulation with fluid subcycling, the time step

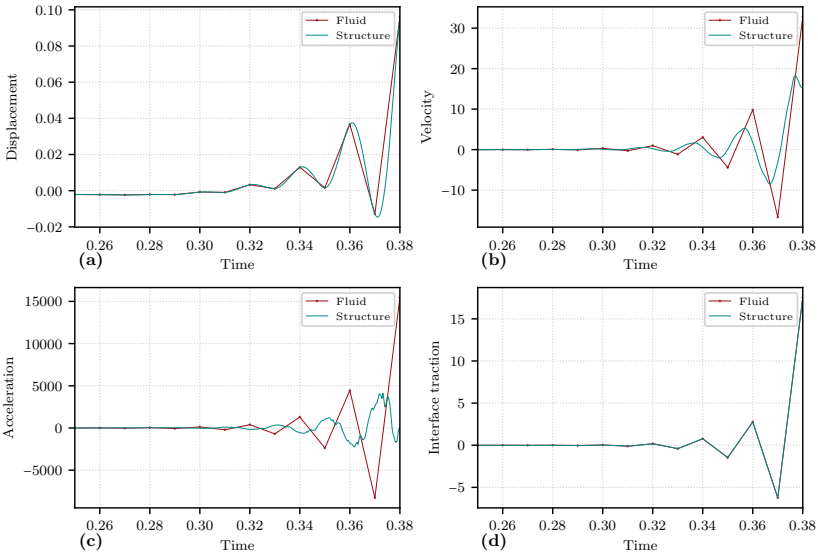


Figure 5.40: Wall benchmark: structure subcycling results based on satisfying the discrete displacement continuity condition

sizes are chosen as $\Delta t^s = 0.2$ and $\Delta t^f = 0.005$ leading to a time step ratio of $N^{s,f} = 40$. For comparison, two FSI simulations with matching time steps of $\Delta t^s = \Delta t^f = 0.005$ and $\Delta t^s = \Delta t^f = 0.2$ are carried out as well. The simulation with matching fine time step will be used as the reference for further comparisons.

The evolution of different quantities at a point located in the middle of the flexible bottom of the cavity is plotted in Fig. 5.41 for the simulation with structure subcycling. The graphs in Fig. 5.41 show that at $t = 30$, the FSI solution is almost fully developed and the flexible bottom of the cavity undergoes a periodic oscillation. The Interface traction graph in Fig. 5.41d reveals that the coarse time step size, $\Delta t^f = 0.2$, used in the fluid domain is fine enough to resolve the general (temporal) profile of the traction's oscillation by the assumption of linear evolution of traction within each coarse time step. In the current example, the fluid's interface displacement is derived consistently from the fluid velocity. The solid and fluid displacement graphs in Fig. 5.41 reveal that the drift in the geometrical constraint is small (and remains small even for longer simulations). In the following, a portion of the time history of the quantities plotted in Fig. 5.41 will be used for a more detailed comparison of the simulations with subcycling and those with matching time steps.

The acceleration values at the reference location from the simulations with structure and fluid subcycling and the two simulations with matching fine and coarse time steps are compared in Fig. 5.42. For the simulations with

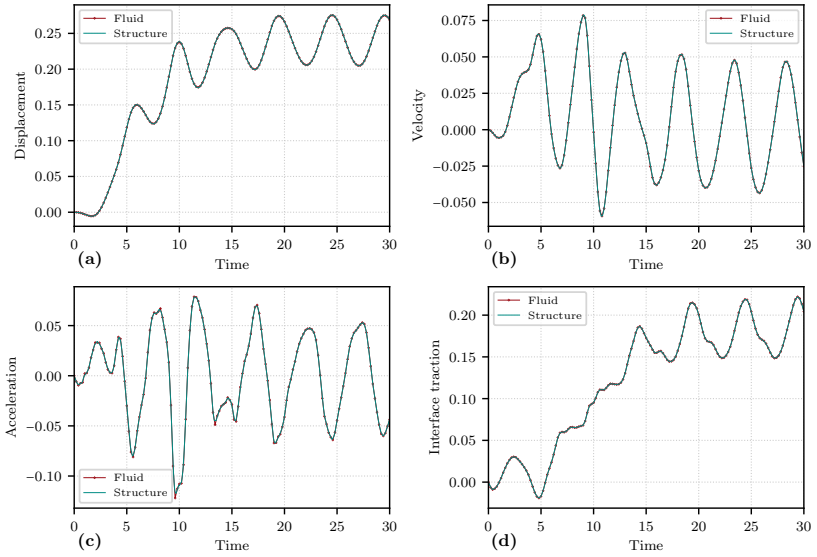


Figure 5.41: Driven cavity with flexible bottom: structure subcycling results based on satisfying the discrete velocity continuity condition

subcycling, the results at the fine and coarse time scales are plotted using solid and dashed lines, respectively. The boundaries of the coarse time steps are also marked using markers. The graphs in Fig. 5.42 are generally not so much different from each other as opposed to the case in section 5.4.2. However, it is still noticeable that the acceleration values from the simulation with structure subcycling is quite close to the reference values from the simulation with matching fine time steps. On the other hand, the results from the simulation with fluid subcycling are quite close to those from the simulation with matching coarse time steps. We can conclude that the adequate temporal resolution of the structure requires finer time steps than the fluid problem, and consequently, the structure subcycling is advantageous in the current example. It is worth noting that the almost linear temporal profile of acceleration within each coarse time step when using fluid subcycling complies well with the theoretical assumptions. The fluid and solid acceleration graphs are also almost indistinguishable in Fig. 5.42 when using either fluid or structure subcycling schemes.

The velocity values at the reference location from the simulations with structure and fluid subcycling and the two simulations with matching fine and coarse time steps are compared in Fig. 5.43. The velocity graphs in Fig. 5.43 confirm that the results from the simulation with structure subcycling are closer to the reference results, indicating that structure subcycling is of advantage in the present case.

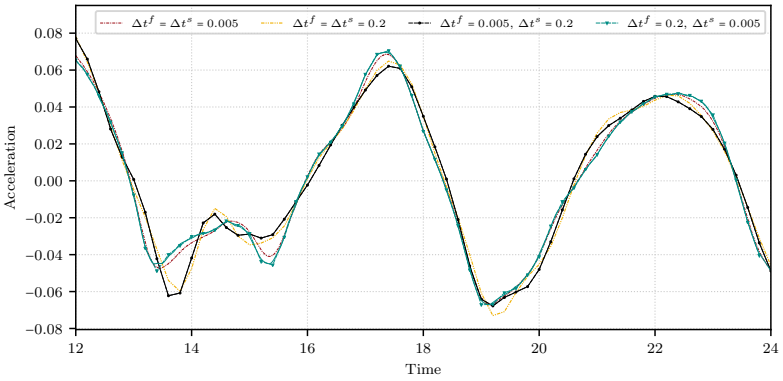


Figure 5.42: Driven cavity with flexible bottom: comparison of acceleration values from fluid and structure subcycling schemes to those from simulations with fine and coarse matching time steps. For the subcycling schemes, both the fine and coarse time level values are plotted using solid and dashed lines, respectively

As shown in Fig. 5.35 and discussed in detail in section 5.4.2, the linear interpolation of fluid traction in the structure subcycling scheme leads to small amplitude (non-physical) fluctuations in the solid acceleration's evolution, especially when no structural damping is present. For the present example, a close-up of the solid and fluid acceleration graphs with structure subcycling is presented in Fig. 5.44 using two different time step ratios of $N^{f,s} = 40$ and $N^{f,s} = 80$. The small amplitude fluctuations in the solid acceleration are observed in the graphs in Fig. 5.44 (for a sense of the scale, compare Fig. 5.44 with Fig. 5.42). The small-amplitude acceleration fluctuations are more pronounced in the simulation with a higher time step ratio of $N^{f,s} = 80$. This observation can be attributed to the fact that, under the same linear temporal profile of traction within a coarse time step as per the assumption, higher modes in the acceleration's evolution are triggered when using a smaller time step size in the solid domain. The fluctuations in the solid acceleration depend on the size of the coarse time step in the fluid domain. If the evolution of the fluid interface traction is adequately resolved by the coarse time step, such that a linear interpolation of traction within a coarse time step is reasonable, the small-amplitude oscillations generally have a minimal effect on the solution of the FSI problem. The latter statement is confirmed by comparing a close-up of the velocity profiles from the two simulations with time step ratios of $N^{f,s} = 40$ and $N^{f,s} = 80$, as shown in Fig. 5.45. The graphs of velocity presented in Fig. 5.45 are very similar, which confirms the previous statement on the effect of the high-frequency small-amplitude acceleration fluctuations.

In the following, a few numerical aspects of the simulations with fluid subcycling will be investigated. In Fig. 5.46, the fluid and solid interface

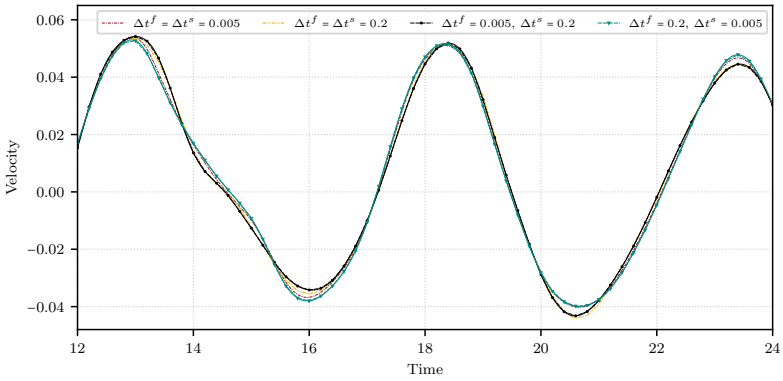


Figure 5.43: Driven cavity with flexible bottom: comparison of velocity values from fluid and structure subcycling schemes to those from simulations with fine and coarse matching time steps. For the subcycling schemes, both the fine and coarse time level values are plotted using solid and dashed lines, respectively

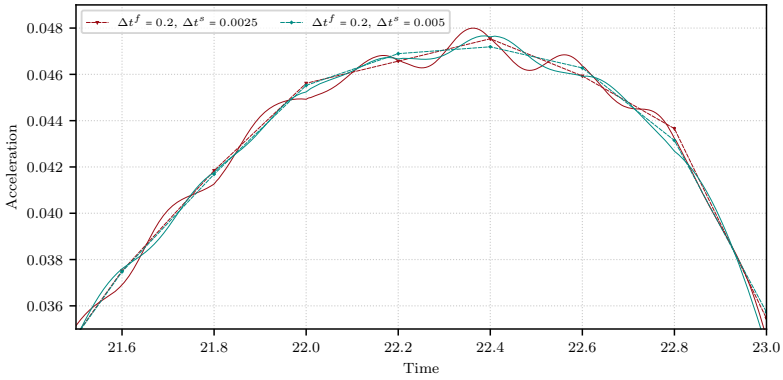


Figure 5.44: Driven cavity with flexible bottom: the effect of fine time step size on the acceleration values from structure subcycling scheme. Both the fine and coarse time level values are plotted using solid and dashed lines, respectively

traction values at the reference point from a portion of the simulation with fluid subcycling are plotted. The fluid traction graph in Fig. 5.46 is not smooth in certain sections. These non-smooth irregularities in the fluid traction are due to the fluid's pressure distribution on the FSI interface, which is affected by the velocity interpolation from the solid domain. Recall that the interpolation is applied to the interface velocities while controlling the evolution of the interface acceleration (i.e., the assumption that fluid

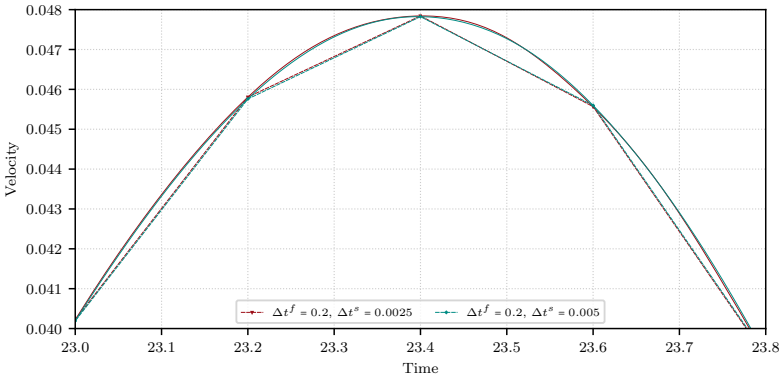


Figure 5.45: Driven cavity with flexible bottom: the effect of fine time step size on the velocity values from structure subcycling scheme. Both the fine and coarse time level values are plotted using solid and dashed lines, respectively

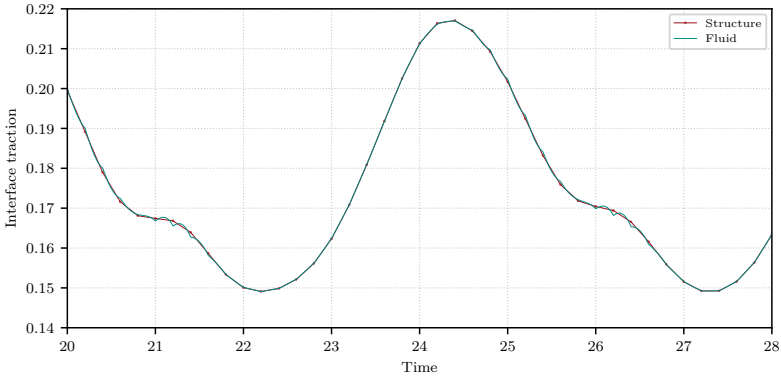


Figure 5.46: Driven cavity with flexible bottom: the interface traction values from the fluid subcycling scheme using Hermite interpolation of velocity values and almost linear fluid acceleration assumption

acceleration evolves linearly within a coarse time step). When solving the INS equations, the pressure in the fluid domain is determined in conjunction with satisfying the incompressibility constraint. However, no measures are taken to control the evolution of fluid pressure's time derivative at the FSI interface within a coarse time step. Therefore, in the problems where the pressure's behavior is sensitive, e.g., solving INS equations in almost closed domains, any slight non-smoothness in the interpolated interface velocity or the corresponding acceleration values might impact the evolution of pressure

in the fluid domain. In the current example, the irregularities in the fluid pressure are relatively small and do not affect the FSI solution due to the projection carried out for determining the solid traction at the FSI interface (See Fig. 5.46). The irregularities in the evolution of fluid pressure increase as the coarse time step size in the solid domain increases in the current example.

It was mentioned in section 5.3 that a quadratic Hermite interpolation of the velocities could also be constructed using the solid velocity values at the two boundaries of the coarse time step and the fluid acceleration value at the beginning of the coarse time step.In the following, the results using a quadratic Hermite interpolation are compared to those using a cubic Hermite interpolation in conjunction with the almost linear acceleration profile. For comparison, simulations with fluid subcycling are carried out using the time step sizes $\Delta t^s = 0.1$ and $\Delta t^f = 0.005$, and the high-frequency numerical damping is determined by $\rho_\infty^s = \rho_\infty^f = 0.5$. The fluid acceleration and velocity values from the two simulations are compared in Figs. 5.47 and 5.48, respectively. Up to $t = 17$ s, the solutions from the two simulations are

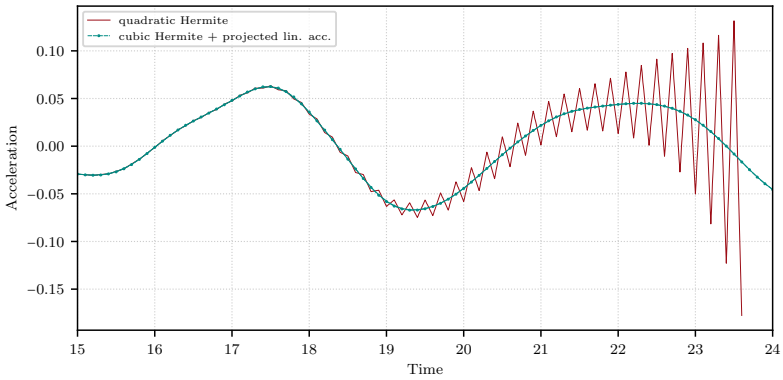


Figure 5.47: Driven cavity with flexible bottom: comparison of the exactly linear and almost linear fluid acceleration assumption in the fluid subcycling scheme using Hermite interpolation

very similar. However, the solution of the simulation with quadratic Hermite interpolation of velocities starts to oscillate and eventually diverges. The reason is the accumulation of numerical errors in the fluid acceleration since only the acceleration value at the beginning of each coarse time step is used for constructing the velocity's interpolation function.

Similar to the previous sections, the results using alternative subcycling schemes are presented hereafter for comparison. In Fig. 5.49, the results using the fluid subcycling scheme with cubic interpolation for displacements are presented. The same numerical behavior, characterized by jumps in the fluid velocity curve at the boundaries of the coarse time step and subsequent spikes

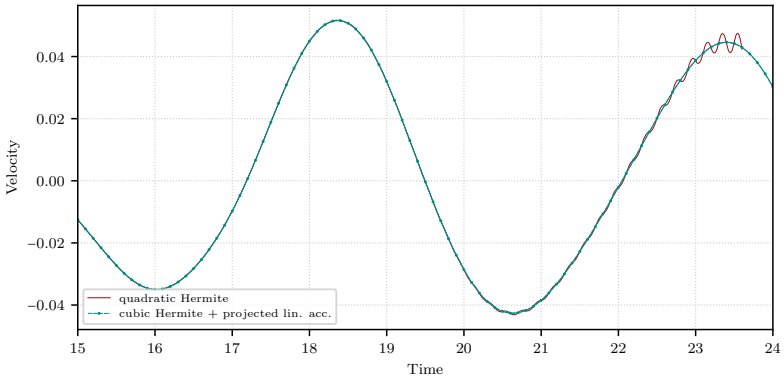


Figure 5.48: Driven cavity with flexible bottom: comparison of the exactly linear and almost linear fluid acceleration assumption in the fluid subcycling scheme using Hermite interpolation

in the fluid acceleration and interface force graphs, is observed. The results of the simulation using the fluid subcycling scheme with linear interpolation of velocities presented in Fig. 5.50 show a similar behavior.

The numerical artifacts in the fluid interface traction are considerably more pronounced compared to the previous test cases due to the fact that the pressure in the fluid domain is much more sensitive to the interface acceleration in the current example.

The results of the structure subcycling scheme based on the discrete displacement continuity condition, presented in Fig. 5.51, confirm that this subcycling scheme leads to unstable results for most of the FSI problems.

5.4.4 Flexible restrictor flap in converging channel

This example, taken from [166, 167], consists of a flexible restrictor flap in a converging channel. Due to the symmetry, only one half of the channel is simulated. The problem's geometry, dimensions, and boundary conditions are given in section 4.3.4.

The current FSI example is challenging due to the low ratio of structural to fluid density. Therefore, the numerical simulations with very small time steps in the solid domain lead to very high-frequency acceleration fluctuations, especially at the tip of the flap. Here, we will focus on using the fluid subcycling scheme to alleviate this issue. The simulation with fluid subcycling is carried out with the time step sizes $\Delta t^s = 0.5$ and $\Delta t^f = 0.005$ in the solid and fluid domains, respectively, leading to a time step ratio of $N^{s,f} = 100$. For comparison, a simulation with matching time steps of $\Delta t^s = \Delta t^f = 0.005$ is also carried out. Both fluid and solid problems are discretized in time using the G-α scheme for first-order systems with the high-frequency numerical

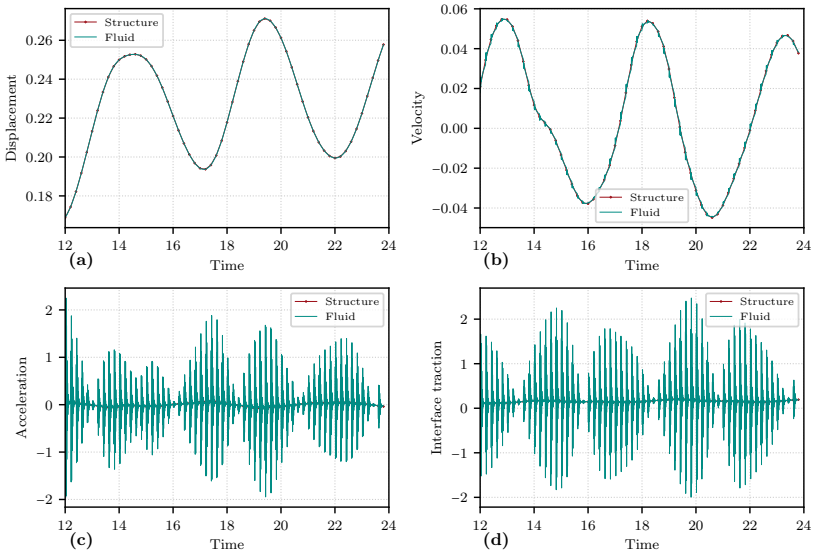


Figure 5.49: Driven cavity with flexible bottom: fluid subcycling results using cubic interpolation of displacement values

damping determined by $\rho_\infty^s = \rho_\infty^f = 0.5$.

The interface quantities at a reference point located at the tip of the flap are plotted in Fig. 5.52 for the simulation with fluid subcycling. The plots in Fig. 5.52 reveal that there is little to no drift in the interface constraints of all kinematic quantities. The time step ratio of $N^{s,f} = 100$ is used in the current simulation. The fluid time step size could be further decreased (thus increasing the time step ratio) without affecting the numerical results. One motivation for subcycling in the current example could be enhancing the convergence of the nonlinear iterations required for solving the fluid problem at every time step (due to using a smaller time step size).

In Fig. 5.53, the acceleration value from the simulation with fluid subcycling is compared to that from the simulation with matching fine time steps. The acceleration graph from the simulation with matching time steps possesses high-frequency fluctuations. In the simulation with fluid subcycling, quite large time steps are used in the solid domain, leading to the disappearance of the acceleration fluctuations at the reference point.

The comparisons of velocity and displacement values from the two simulations are presented in Figs. 5.54 and 5.55, respectively. The graphs in Figs. 5.54 and 5.55 indicate that the results from the simulation with fluid subcycling are as accurate as those from the reference simulation with matching time step sizes.

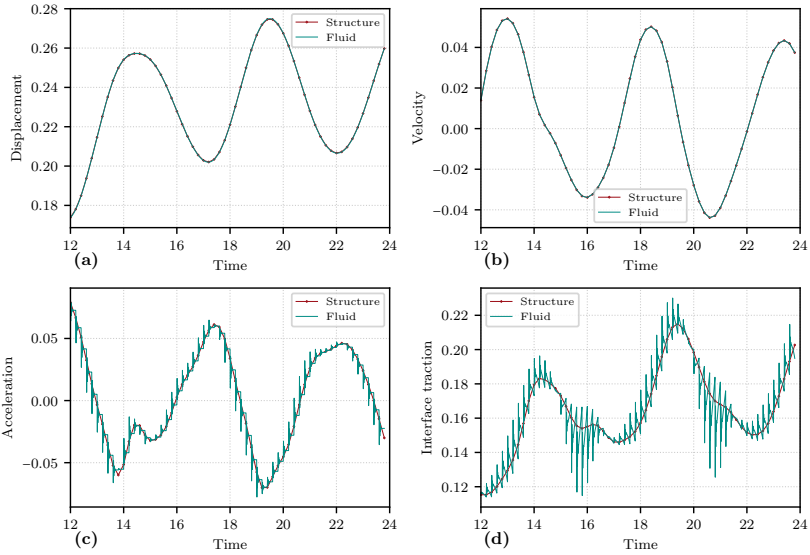


Figure 5.50: Driven cavity with flexible bottom: fluid subcycling results using linear interpolation of velocity values

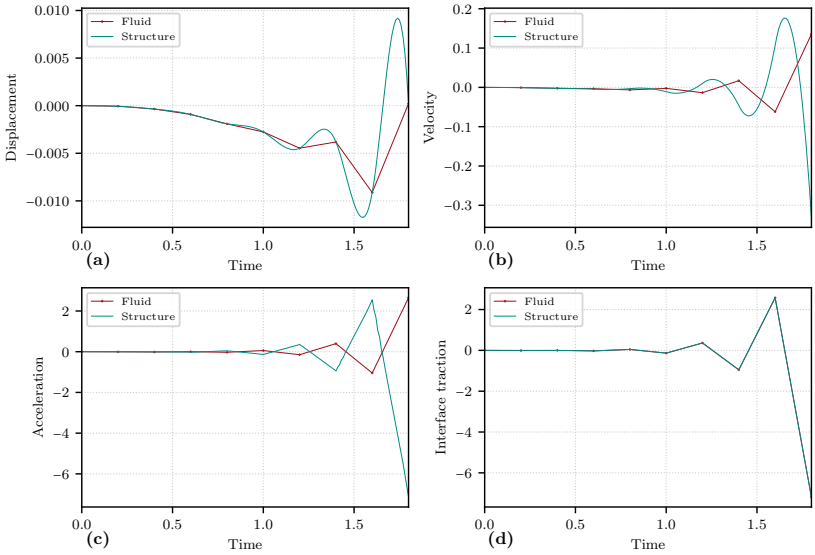


Figure 5.51: Driven cavity with flexible bottom: structure subcycling results based on satisfying the discrete displacement continuity condition

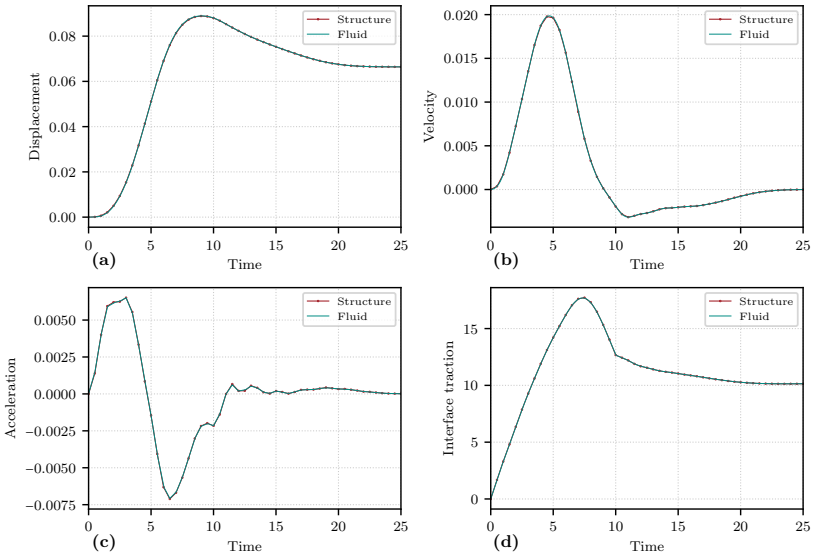


Figure 5.52: Mok benchmark: fluid subcycling results using Hermite interpolation of velocity values and almost linear fluid acceleration assumption

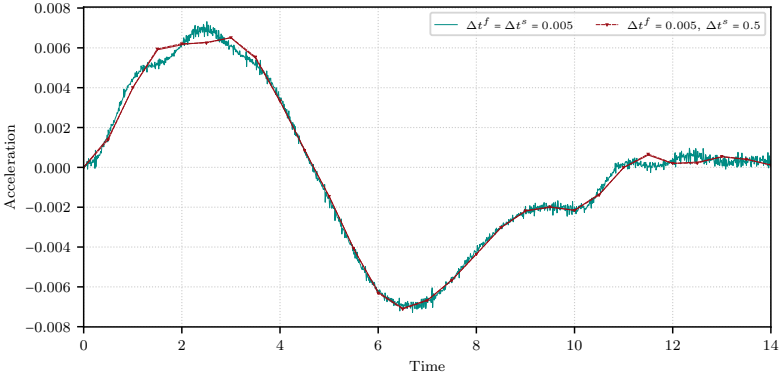


Figure 5.53: Mok benchmark: comparison of acceleration values from fluid subcycling scheme to those from simulation with fine matching time steps. For the subcycling scheme, both the fine and coarse time level values are plotted using solid and dashed lines, respectively

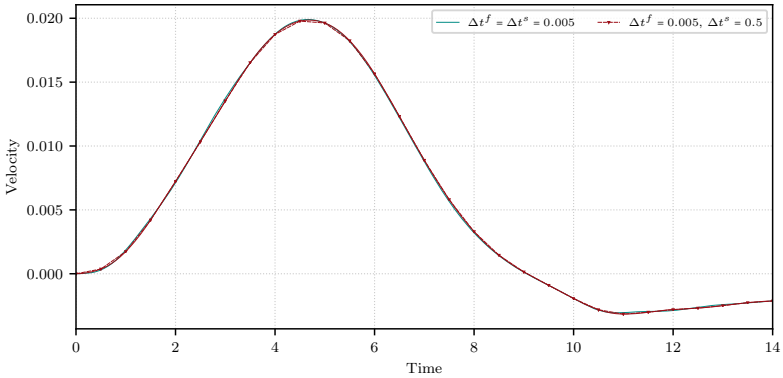


Figure 5.54: Mok benchmark: comparison of velocity values from fluid subcycling scheme to those from simulation with fine matching time steps. For the subcycling scheme, both the fine and coarse time level values are plotted using solid and dashed lines, respectively

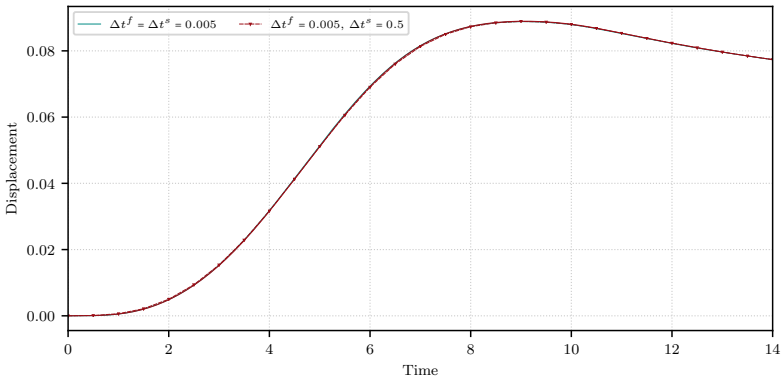


Figure 5.55: Mok benchmark: comparison of displacement values from fluid subcycling scheme to those from simulation with fine matching time steps. For the subcycling scheme, both the fine and coarse time level values are plotted using solid and dashed lines, respectively

Chapter 6

Fluid-structure interaction simulation of an omega-shaped Coriolis mass flow meter

6.1 Introduction

The FSI model of the omega-shaped CMF was developed in collaboration with Rheonik Messtechnik GmbH and supported by Zentrales Innovationsprogramm Mittelstand (ZIM) within the research project ZF4593301RE8. The details of the FSI model and the comparison of numerical and experimental results are published in [168]. Some figures and parts of the text in this chapter are taken in an adjusted form from [168]. The simulations in this section are carried out in the Kratos Multiphysics framework [141, 142, 169].

In this chapter, some of the methods developed in the present work are applied to an industrial example of FSI simulation of an omega-shaped Coriolis mass flow meter (CMF). In particular, the numerical results using the EG-α time integration scheme (see chapter 2) and the enhanced interface quasi-Newton methods (see chapters 3, and 4) are presented. The study presented in [168] shows that the fluid and structural solvers require very similar time step sizes for accurately capturing the physical behavior of the CMF. Therefore, the multi-time-step coupling algorithms developed in chapter 5 are not used in this example.

CMFs measure the mass flow rate in pipes using a principle based on induced Coriolis forces on the fluid particles. The structure of CMF undergoes a harmonic oscillatory motion induced by two drive actuators. Due to this oscillating motion, the fluid particles passing through the CMF experience a relative velocity with respect to the CMF's structure in the plane of oscillation. Consequently, a Coriolis force is induced on the fluid particles. The resulting reaction force influences the motion of the CMF's structure. Therefore, strongly coupled fluid-structure interaction phenomenon plays a crucial role in the operation of CMFs.

CMFs are generally comprised of tube(s), drive actuator(s), two motion detection sensors and a control unit. A harmonic load is applied by the drive actuator on the tube and the structure is brought to forced oscillation. The aim is to bring the CMF's structure to resonance at one of its specific mode shapes in order to achieve a desired magnitude of motion, while minimizing energy consumption. The oscillating motion is then captured by motion detection sensors at two different locations on the tube structure. When there is no fluid in the CMF's tubes, the captured motion signals by the two motion detection sensors have the same phase. When there is a stationary fluid in the tubes, the response of the system is different than previous operational mode, but the two captured motion signals still have the same phase. Finally, when there is a flow passing through the CMF a phase-shift is induced between the captured signals as a result of the Coriolis forces driven by the interaction between the flow and the oscillating structure. The CMFs are designed such that the induced phase-shift is linearly proportional to the mass flow rate in the pipeline [168].

6.2 The FSI model

In omega-shaped CMFs, the symmetrical torsional mode shape is excited by the drive actuators and the Coriolis forces excite a symmetrical bending mode shape. As depicted schematically in Fig. 6.1a, two counteracting actuators excite the symmetrical torsional mode shape and each omega tube oscillates around its oscillation axis (see Fig. 6.1b). The plane, on which each omega tube (together with its oscillation axis) lies, is called the plane of oscillation. The two motion detection sensors, which are located symmetrically with respect to the oscillation axes, capture the motion signals. With flow passing through the CMF's tubes, the oscillatory torsional excitation results in Coriolis forces distributed over the length of the tubes. These Coriolis forces act in the direction perpendicular to the planes of oscillation. The resultant of the torsional motion (induced by the drive actuators) and the bending motion (induced by the Coriolis forces) leads to a phase-shift between the signals captured by the two motion detection sensors.

The omega-shaped CMF's structure is mainly comprised of two symmetric tubes. In general, the motion of the structure can be non-symmetric under different loading conditions. However, under operational conditions, the

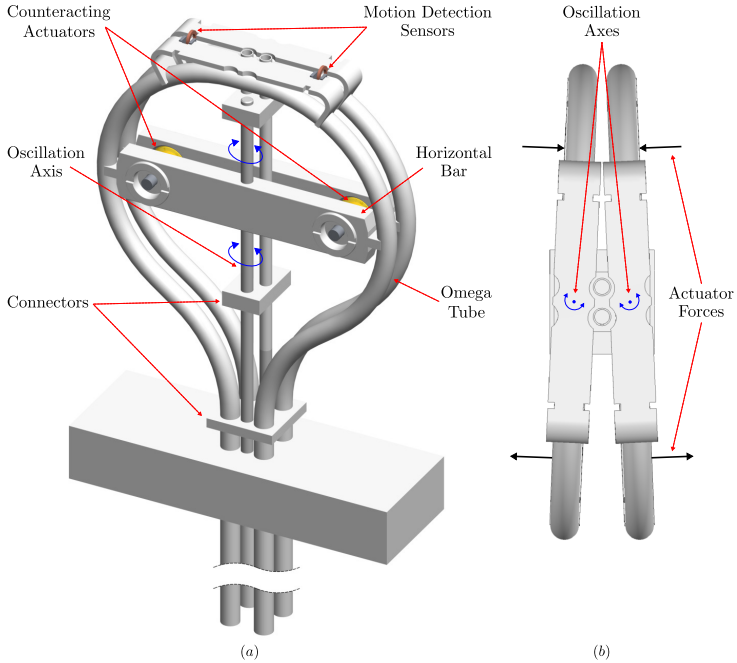


Figure 6.1: Illustration of an omega-shaped CMF composed of omega tubes, counteracting actuators, motion detection sensors, oscillation axes, horizontal bars and connectors and b: top view of the CMF with symmetrical torsional motion [168]

symmetric tubes are excited in symmetric torsional and bending mode shapes. Therefore, in order to reduce the computational costs, only one side of the structure is modeled by applying symmetry BC on the symmetry plane shown in Fig. 6.2.

The torsional mode shape of the structure is excited by applying two dynamic forces with the same amplitude but in opposite directions at the positions where the actuators are located (see Fig. 6.2). The phase-shift can be found by a forced vibration at the resonance. However, there are two issues with this approach. First, the eigenfrequency must be determined in order to have a forced vibration at the resonance. This can be done by analyzing the free vibration response to an impulse. Secondly, to calculate the phase-shift, the simulation must be carried out for a long enough time in order to get only the harmonic particular solution without the transient homogeneous solution. Therefore, in the present work, in order to reduce the computational costs, the impulsive forces are applied and the phase-shift is calculated through analyzing the free vibration response. The impulse forces are modeled by a

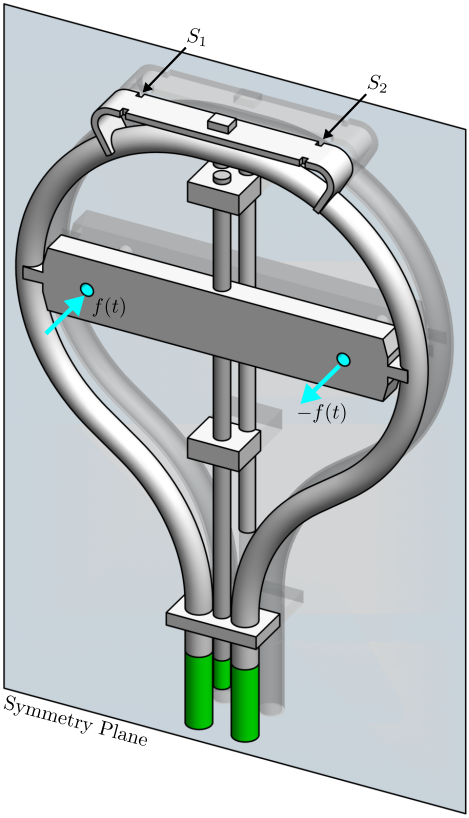


Figure 6.2: One half of the CMF model simulated due to the symmetry in the geometry and operational loading and boundary conditions. The green sections indicate the clamped portions of the CMF's structure [168]

Dirac delta function and applied as distributed loads on the CMF's structure in the position of drive actuators.

In the following, the setup of the FSI problem is concisely presented. For more details on the parameters and numerical setup of the model, the interested readers are referred to [168]. A schematic FSI problem is depicted in Fig. 6.3. The structural domain is discretized using linear tetrahedral elements. This type of element is known to behave stiffer than hexahedral elements or tetrahedral elements with quadratic shape functions. Therefore, in order to have reliable results higher mesh resolutions are usually needed. In addition, to accurately capture the motion of the structure, especially in torsional and bending modes, the mesh refinement is focused on the omega-shaped tube, the oscillation axis and the connectors. The computational

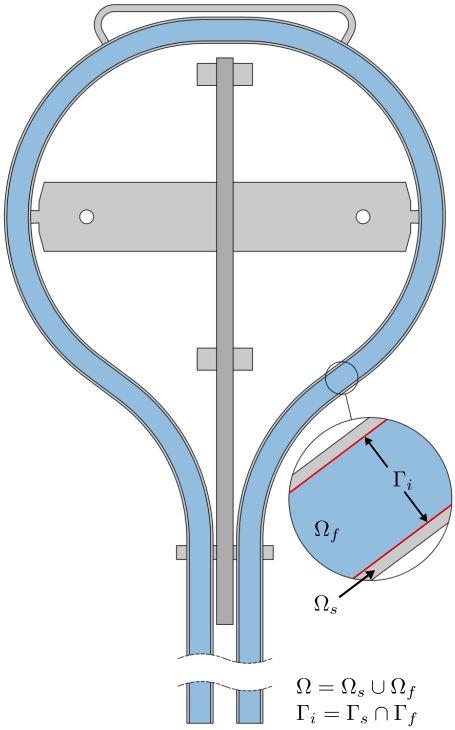


Figure 6.3: The FSI domain divided into the structural and fluid domains [168]

mesh of the structural domain is depicted in Fig. 6.4. The mesh of the omega-shaped tube is structured in radial direction and along the tube. The oscillation axis and connectors have a semi-structured mesh while, the mesh of the rest of the structure is unstructured.

Modeling the structural damping is crucial in correctly capturing the dynamic behavior of the CMF. The structural damping is modeled using Rayleigh damping. The corresponding Rayleigh damping coefficients are calculated from the measured damping ratios of torsional and bending frequencies from the experimental data.

In the fluid domain, spatially constant velocity profiles are assigned to the inlet as a Dirichlet BC. The inlet velocity is ramped up in time to the nominal value in order to initialize the flow simulation. On the boundaries a no-slip BC is defined and the relative outlet pressure is set to 0. In order to prevent the flow inside the CMF to be influenced by the inlet and outlet (e.g. flow development in the inlet and back-flow in the outlet), the fluid domain

Figure 6.4: The mesh of the structural domain

is extended beyond the CMF's structure. The fluid domain is discretized using linear tetrahedral elements. The fluid grid is structured in flow stream direction from inlet to outlet and is semi-structured in radial direction. The computational mesh of the fluid domain is depicted in Fig. 6.5.

The fully developed flow velocity and the pressure profiles in the omega tube are depicted in Fig. 6.6.

The spatial discretizations of the structural and fluid domains are in general non-conforming on the FSI interface because the grid size requirements are different for the solution of each domain. Due to different discretizations, even the interface geometry can be non-matching with gaps and overlaps between boundaries of fluid and structural grids. Since the displacements and tractions must be transferred through the interface, data mapping technique must be employed to perform this data exchange on the FSI interface. In the present work, the nearest element interpolation method is used. In this method, e.g.

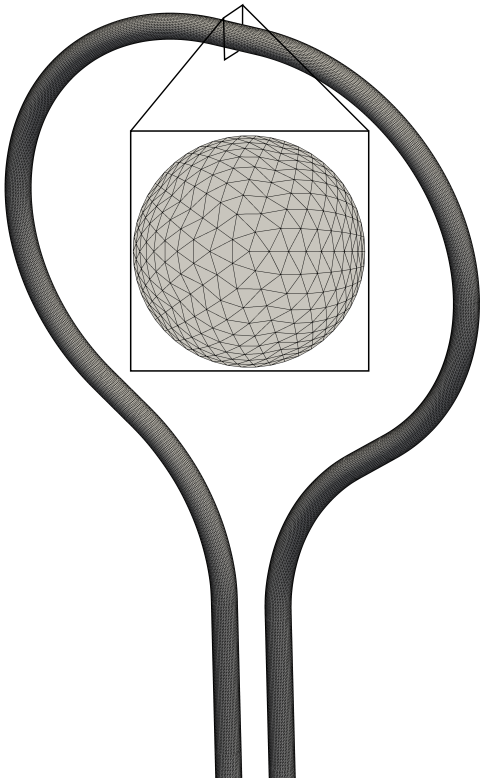


Figure 6.5: The mesh of the fluid domain

for transferring displacement from the structural domain (source) to the fluid domain (target), each node of the fluid interface is projected onto the nearest element on the structural interface and the unknown displacement at the projected point is interpolated from the element's nodes and is assigned to the fluid node.

The deformation of the CMF's structure in the FSI simulation is depicted in Fig. 6.7.

In the following, a comparison will be carried out between the methods developed in the present work and the numerical methods used in [168] for FSI simulation of the omega-shaped CMF. In particular, the advantages of the EG-α time integration scheme (see chapter 2) and the enhanced interface quasi-Newton methods (see chapters 3, and 4) are highlighted for the present industrial FSI example.

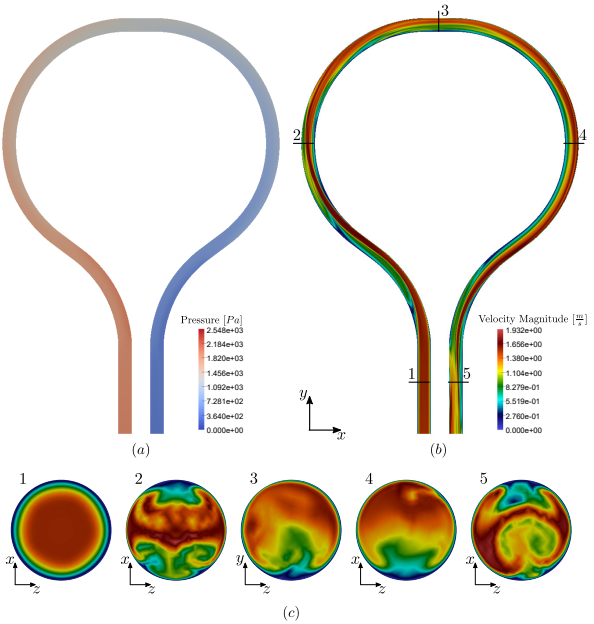


Figure 6.6: The flow velocity and the pressure profiles in the Omega tube. (a) The pressure profile; (b) the velocity profile; (c) the velocity profile at five slices marked in b

The measured mass flow rate in the CMF is (by design) linearly proportional to the induced phase-shift between the outputs of the two sensors. Therefore, the temporal accuracy is of utmost importance in the dynamic simulations of CMFs since the numerically calculated phase-shift is affected by the numerical dissipation introduced by the time integration scheme. The EG-α scheme (as well as other members of the G-α family of schemes) allows for desired amount of high-frequency damping controlled by ρ_∞. In the current example, EG-α scheme is used in both the fluid and structural solvers with $\rho_\infty^f = \rho_\infty^s = 0.5$. The results from the simulation with EG-α scheme are compared with those from a simulation with BDF2 scheme in the fluid solver and Bossak scheme (with $\alpha_m = -0.33$) in the structural solver. In Fig. 6.8, a comparison is carried out between the two simulations with regard to the displacement response at sampling point S_1. The graphs in Fig. 6.8 reveal that the EG-α scheme leads to lower numerical dissipation both in terms of the amplitude decay and period elongation, which is important for the calculation of the phase-shift.

Another area of focus in this work is enhancing the robustness and efficiency of convergence acceleration techniques for FSI problems. The IQN-ILS algorithm with automatic determination of time history presented in chapter 4 is applied to the present example and the results are compared to the

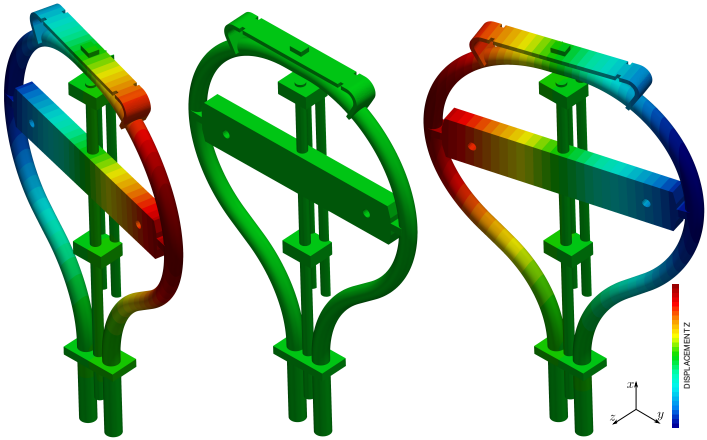


Figure 6.7: The CMF at rest together with the deformed states at the two extremes of deformation. The deformations are magnified 10 times for the sake of clarity

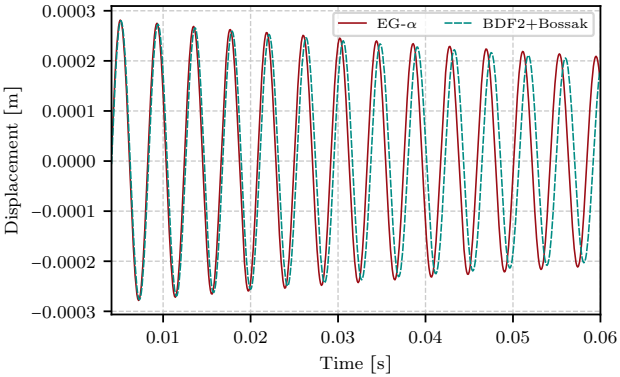


Figure 6.8: The response of the CMF at sampling point S_1 to impulse. Comparison between the EG-α scheme and the combination of BDF2 scheme in fluid and Bossak scheme in the structural domain

Aitken relaxation (with initial relaxation factor 0.5) used in [168] for the same tolerance values. The average number of iterations per time step is **5.81** using the Aitken relation and **3.05** using the IQN-ILS with the automatic column selection strategy presented in chapter 4. The efficiency improvement through the reduced number of FSI iterations is substantial in the current example due to the high evaluation costs of the fluid and structural solvers with detailed high-fidelity models.

Chapter 7

Conclusion and outlook

7.1 Summary of the contributions

The present work contributes to enhancing the robustness, accuracy, and efficiency of multiphysics and coupled simulations, in particular fluid-structure interaction problems, by proposing methods and solution strategies for the time discretization and convergence acceleration of partitioned strongly coupled problems. A Summary of the main contributions and findings in each chapter is presented in the following.

Chapter 2 presents the EG-α time integration scheme, an end-of-step-equilibrium form of the G-α scheme, for FSI problems. The time discretization of the fluid, structural, and mesh motion problems using the EG-α is detailed and analyzed. Using this scheme, the equilibrium equation of the problem is satisfied exactly at discrete time points instead of the classical averaged form. The benefits of the EG-α scheme are demonstrated for the stabilized finite element formulations for incompressible Navier-Stokes equations on fixed and moving domains.

- The incompressibility condition is satisfied precisely at the discrete time points. Therefore, the errors in the incompressibility condition are not propagated in time, and the velocity field remains divergence-free throughout the simulation.

- The EG-α allows the consistent and straightforward temporal discretization of stabilized methods for incompressible Navier-Stokes equations. When using dynamic subgrid scales in the context of the variational multiscale method, an elegant and simple blend of the EG-α scheme for

the discretization of the large-scale problem and the backward Euler scheme for the fine-scale problem is viable, which keeps the overall second-order temporal accuracy of the large-scale problem intact.

- In the fluid problems on moving domains, the spatial integrals are naturally evaluated on the most recent configuration of the moving domain, which reduces the implementation's complexity while maintaining temporal stability and accuracy.

- The second-order temporal accuracy for velocity, pressure, acceleration, pressure time derivative and mesh velocity variables are retained using the EG-α scheme.

In addition to the derivation of the EG-α scheme, this chapter presents a comprehensive analysis of different variants of the generalized-α scheme for the incompressible Navier-Stokes equations and compares them in the case of discretization on fixed and moving domains.

A variant of the EG-α scheme for first-order systems is presented for the temporal discretization of the structure in FSI problems. The proposed variant of the scheme avoids doubling the number of unknowns in the problem's linear system of equations, which is common in the state-space (displacement-velocity) formulations. The presented scheme is consistent with the time integration scheme used in the fluid problem.

The consistent temporal discretization of the FSI problem using the EG-α scheme is presented, which avoids the need for temporal interpolation of interface values between the fluid and solid domains in order to maintain the second-order accuracy for the whole FSI problem.

Chapter 3 addresses some previously open issues regarding the efficiency and robustness of interface quasi-Newton methods for the convergence acceleration of the partitioned coupled problems by bridging the gap between the state-of-the-art knowledge in interface quasi-Newton methods and some of the lesser-known techniques in dense linear algebra.

Algorithmic enhancements are suggested for solving the least squares problem arising from the interface quasi-Newton methods using modified Gram-Schmidt or Householder QR factorization.

The use of column scaling is proposed in the solution of the least squares problem associated with the interface quasi-Newton methods. By scaling the columns of the matrix involved in the least squares problem such that all columns have equal and unit norms, the adverse effect of very dispersed column norms is removed from the matrix. It is demonstrated that the quasi-Newton matrices have better conditioning when column scaling is carried out, even without applying any filtering. Column scaling enhances the robustness of filtering in the quasi-Newton methods as well. It is demonstrated that without column scaling, filtering removes (very often) the columns with the lowest norms, including the columns added in the most recent iterations. For some choices of filtering tolerance, filtering without column scaling leads to

a lack of convergence even after a substantial number of coupling iterations. When using column scaling in conjunction with filtering, the convergence rate of the quasi-Newton methods is less sensitive to the filtering tolerance.

Another aspect of the filtering process elaborated in the current work is the choice of filtering criterion and its relation to the reciprocal condition number of the matrix used in the least squares solution by QR factorization. Based on the information available from the R-factor, different filtering criteria are provided and numerically compared. Furthermore, it is shown that filtering based on the diagonal entries of the R-factor leads to overestimating the exact minimum singular value of the R-factor (sometimes by a few orders of magnitude). This fact should be taken into account when choosing a tolerance for filtering based on the diagonal entries of the R-factor.

Chapter 4 presents a strategy for automatically determining the time step history in interface quasi-Newton methods. A major question concerning the reuse of information from the previous time steps in the IQN-ILS method is the optimal combination of values for the maximum time step history and the filtering tolerance. This issue is tackled in the present work.

A strategy for combining filtering techniques and automatic determination of the number of previous time steps in IQN-ILS method is proposed by combining the basic solution via QR factorization with column pivoting, with the closest column and threshold pivoting, cyclic permutation, age-based column scaling, and restricted column pivoting techniques. The advantage of the proposed strategy is that it achieves a good performance (in terms of the number of coupling iterations) for various FSI problems with a set of default parameters. Therefore, the need for trial and error for determining the optimal time step history in IQN-ILS method is reduced. It is still possible to tune the method for different problems if one wishes to by changing the parameters. However, by combining the filtering and determination of the time step history, the number of parameters directly defined by the user is reduced to one. The various numerical examples presented in this work suggest that the performance of the proposed strategy is not very sensitive to the user-defined parameter ϵ, and a default value leads, in most cases, to a good performance compared to other variants of interface quasi-Newton methods.

Using an incremental condition estimator for the accurate estimation of minimum and maximum singular values in the least squares problem of interface quasi-Newton methods are proposed. This technique is beneficial for filtering techniques based on the QR factorization with or without column pivoting.

Chapter 5 proposes multi-time-step coupling schemes for FSI problems. The schemes are based on partitioned strongly coupled fluid-structure interaction algorithms and Dirichlet-Neumann decomposition. For the convergence acceleration of the iterative coupling procedure, state-of-the-art quasi-Newton methods are successfully used without the need for any adaptation due to the multi-time-step nature of the considered problems. Those features of the

proposed schemes allow a straightforward extension of the existing coupling environments to include multi-time-step capabilities.

Both fluid and structure subcycling schemes are proposed and analyzed. The schemes are based on satisfying the discrete velocity continuity across the FSI interface. It is shown that the approaches based on satisfying the discrete displacement continuity suffer from loss of accuracy in the case of fluid subcycling and lack stability in the case of structure subcycling in strongly coupled FSI problems. In contrast, the fluid and structure subcycling schemes based on satisfying the discrete velocity continuity are stable and accurate even for very high time step ratios.

For the interpolation of kinematic values in the fluid subcycling scheme, a Hermite interpolation of velocity values together with an almost linear acceleration profile is proposed. The assumed acceleration profile is generated using a weighted average of the fluid's linear acceleration value and the structure's actual acceleration value. Consequently, the interpolated velocity profile is almost quadratic. Furthermore, a cubic Hermite interpolation of displacement values is proposed for the cases where the drift in the displacement constraint is unacceptably large. However, the numerical examples show that the drift in the displacement constraint remains small for most combinations of physical and reasonable algorithmic parameters.

Another important aspect of the multi-time-step FSI algorithms addressed here is the transferring of interface forces between different time levels. We make the assumption that the interface forces have a linear profile within a coarse and fine time step. It is shown that this assumption is reasonable for a wide range of time step sizes and algorithmic parameters of the considered family of time integration schemes. Based on this assumption, an averaging formula is deduced for transferring the interface forces from the fine to coarse time scale in the fluid subcycling scheme.

7.2 Outlook

It was previously mentioned that the presented EG-α time integration scheme is very useful for discretizing unsteady INS equations using unfitted methods, including embedded methods based on cutting the elements in a background mesh using a boundary representation of the objects in the domain. One of the challenges in such techniques is that the variables' domain of definition changes when proceeding from one time step to the next. A solution proposed in the literature is the use of so-called extension operators, which extend the solutions from the previous time step to the current domain. Such operators must be consistent with the time integration scheme in order to allow preserving the original order of accuracy of the scheme in the unfitted techniques. A natural extension of the present work is to apply the EG-α time integration scheme to the problems discretized using unfitted methods and derive the required consistent extension operators. Another logical step in the development of EG-α schemes is deriving consistent explicit forms of

the scheme, which are useful for discretizing multiphysics problems with fast dynamics, e.g., the simulation of impact in FSI problems.

With regard to the quasi-Newton convergence acceleration techniques, a further development is the extension of the proposed automatic column selection strategies to allow solving the least squares problem using QR factorization based on modified Gram-Schmidt, which allows a more efficient solution of the least squares problems in very large systems compared to the Householder QR factorization. An interesting idea worth investigating is augmenting the interface quasi-Newton methods with ideas borrowed from the Machine learning and surrogate modeling techniques in order to further accelerate the convergence of multiphysics simulations by predicting better initial guesses for the iterates or enhanced column selection strategies.

The multi-time-step FSI algorithms developed in the present work are based on the Dirichlet-Neumann decomposition of the problem. While simple to implement and quite effective when combined with state-of-the-art convergence acceleration techniques, the approaches based on the Dirichlet-Neumann decomposition have limitations in certain simulations. In recent years, coupling approaches based on Robin-Neumann decomposition have gained traction and are demonstrated to be efficient for many coupled problems. The extension of the present multi-time-step methods to the coupling techniques based on Robin-Neumann decomposition is the subject of future works. The multi-time-step coupling algorithms are also of interest in FSI simulations with separate reduced order models for the fluid and structure with different requirements on their time step sizes. Therefore, extending and applying the proposed methods to the model order reduction techniques for FSI problems is another topic of further research. Finally, the robust and efficient implementation of the proposed algorithms, including efficient checkpointing and restart mechanisms, transferring of the data between different time scales, and incorporation in the existing software frameworks for simulations with matching time steps are potential subjects for future works.

Bibliography

[1] V. John, G. Matthies, and J. Rang. "A comparison of time-discretization/linearization approaches for the incompressible Navier-Stokes equations". In: *Computer Methods in Applied Mechanics and Engineering* 195.44-47 (2006), pp. 5995–6010. DOI: 10.1016/j.cma.2005.10.007.

[2] P. Gresho and R. Sani. "Incompressible flow and the finite element method. Volume 2: Incompressible flow and finite element". In: (1998).

[3] S. Turek. *Efficient Solvers for Incompressible Flow Problems*. Vol. 6. Lecture Notes in Computational Science and Engineering 9. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 1689–1699. ISBN: 978-3-642-63573-1. DOI: 10.1007/978-3-642-58393-3.

[4] J. G. Heywood and R. Rannacher. "Finite-element approximation of the nonstationary navier-stokes problem part IV. Error analysis for second-order time discretization". In: *SIAM Journal on Numerical Analysis* 27.2 (1990), pp. 353–384. DOI: 10.1137/0727022.

[5] W. Dettmer and D. Perić. "An analysis of the time integration algorithms for the finite element solutions of incompressible Navier - Stokes equations based on a stabilised formulation". In: *Computer Methods in Applied Mechanics and Engineering* 192.9-10 (2003), pp. 1177–1226. DOI: 10.1016/S0045-7825(02)00603-5.

[6] R. Codina, J. Principe, O. Guasch, and S. Badia. "Time dependent subscales in the stabilized finite element approximation of incompressible flow problems". In: *Computer Methods in Applied Mechanics and Engineering* 196.21-24 (2007), pp. 2413–2430. DOI: 10.1016/j.cma.2007.01.002.

[7] M. Bristeau, R. Glowinski, and J. Periaux. "Numerical methods for the navier-stokes equations. Applications to the simulation of compressible and incompressible viscous flows". In: *Computer Physics Reports* 6.1 (1987), pp. 73–187. DOI: https://doi.org/10.1016/0167-7977(87)90011-6.

[8] R. Glowinski. “Finite element methods for incompressible viscous flow”. In: *Numerical Methods for Fluids (Part 3)*. Vol. 9. Handbook of Numerical Analysis. Elsevier, 2003, pp. 3–1176. DOI: https://doi.org/10.1016/S1570-8659(03)09003-3.

[9] V. John and J. Rang. “Adaptive time step control for the incompressible Navier-Stokes equations”. In: *Computer Methods in Applied Mechanics and Engineering* 199.9-12 (2010), pp. 514–524. DOI: 10.1016/j.cma.2009.10.005.

[10] J. Rang. “Pressure corrected implicit θ-schemes for the incompressible Navier-Stokes equations”. In: *Applied Mathematics and Computation* 201.1-2 (2008), pp. 747–761. DOI: 10.1016/j.amc.2008.01.010.

[11] B. Sanderse and B. Koren. “Runge-Kutta methods for the incompressible Navier-Stokes equations”. In: *21st AIAA Computational Fluid Dynamics Conference* (2013), pp. 1–16. DOI: 10.2514/6.2013-3085.

[12] K. E. Jansen, C. H. Whiting, and G. M. Hulbert. “A generalized-α method for integrating the filtered {N}avier-{S}tokes equations with a stabilized finite element method”. In: *Computer Methods in Applied Mechanics and Engineering* 190.May 2000 (2000), pp. 305–319. DOI: 10.1016/S0045-7825(00)00203-6.

[13] Y. Bazilevs, V. M. Calo, J. A. Cottrell, T. J. Hughes, A. Reali, and G. Scovazzi. “Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows”. In: *Computer Methods in Applied Mechanics and Engineering* 197.1-4 (2007), pp. 173–201. DOI: 10.1016/j.cma.2007.07.016.

[14] V. Gravemeier, M. Kronbichler, M. W. Gee, and W. A. Wall. “An algebraic variational multiscale-multigrid method for large-eddy simulation: Generalized-α time integration, Fourier analysis and application to turbulent flow past a square-section cylinder”. In: *Computational Mechanics* 47.2 (2011), pp. 217–233. DOI: 10.1007/s00466-010-0541-x.

[15] A. Lovrić, W. G. Dettmer, C. Kadapa, and D. Perić. “A new family of projection schemes for the incompressible Navier–Stokes equations with control of high-frequency damping”. In: *Computer Methods in Applied Mechanics and Engineering* 339 (2018), pp. 160–183. DOI: 10.1016/j.cma.2018.05.006.

[16] J. Liu, I. S. Lan, O. Z. Tikenogullari, and A. L. Marsden. “A note on the accuracy of the generalized-α scheme for the incompressible Navier-Stokes equations”. In: *International Journal for Numerical Methods in Engineering* 122.2 (Jan. 2021), pp. 638–651. DOI: 10.1002/nme.6550.

[17] O. A. Ladyzhenskaya. *The mathematical theory of viscous incompressible flow*. Vol. 2. Gordon and Breach New York, 1969.

[18] I. Babuška. “Error-bounds for finite element method”. In: *Numerische Mathematik* 16.4 (1971), pp. 322–333.

[19] I. Babuška. “The finite element method with Lagrangian multipliers”. In: *Numerische Mathematik* 20.3 (1973), pp. 179–192.

[20] F. Brezzi. “On the existence, uniqueness and approximation of saddle-point problems arising from Lagrangian multipliers”. In: *Publications mathématiques et informatique de Rennes* S4 (1974), pp. 1–26.

[21] C. Taylor and P. Hood. “A numerical solution of the Navier-Stokes equations using the finite element technique”. In: *Computers & Fluids* 1.1 (1973), pp. 73–100.

[22] D. Boffi, F. Brezzi, M. Fortin, et al. *Mixed finite element methods and applications*. Vol. 44. Springer, 2013.

[23] V. John et al. *Finite element methods for incompressible flow problems*. Springer, 2016.

[24] T. J. Hughes and A. N. Brooks. “A multidimentional upwind scheme with no crosswind diffusion”. In: *Finite Element Methods for Convection Dominated Flows, AMD 34* (1979).

[25] A. N. Brooks and T. J. Hughes. “Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations”. In: *Computer methods in applied mechanics and engineering* 32.1-3 (1982), pp. 199–259.

[26] T. J. Hughes, L. P. Franca, and M. Balestra. “A new finite element formulation for computational fluid dynamics: V. Circumventing the Babuška-Brezzi condition: A stable Petrov-Galerkin formulation of the Stokes problem accommodating equal-order interpolations”. In: *Computer Methods in Applied Mechanics and Engineering* 59.1 (1986), pp. 85–99.

[27] L. P. Franca and T. J. Hughes. “Two classes of mixed finite element methods”. In: *Computer Methods in Applied Mechanics and Engineering* 69.1 (1988), pp. 89–129.

[28] P. Hansbo and A. Szepessy. “A velocity-pressure streamline diffusion finite element method for the incompressible Navier-Stokes equations”. In: *Computer Methods in Applied Mechanics and Engineering* 84.2 (1990), pp. 175–192.

[29] T. E. Tezduyar, S. Mittal, S. Ray, and R. Shih. “Incompressible flow computations with stabilized bilinear and linear equal-order-interpolation velocity-pressure elements”. In: *Computer Methods in Applied Mechanics and Engineering* 95.2 (1992), pp. 221–242.

[30] L. Tobiska and R. Verfürth. "Analysis of a streamline diffusion finite element method for the stokes and navier-stokes equations". In: *SIAM Journal on Numerical Analysis* 33.1 (1996), pp. 107–127. DOI: 10.1137/0733007.

[31] T. E. Tezduyar and Y. Osawa. "Finite element stabilization parameters computed from element matrices and vectors". In: *Computer Methods in Applied Mechanics and Engineering* 190.3-4 (2000), pp. 411–430.

[32] T. J. Hughes, L. P. Franca, and G. M. Hulbert. "A new finite element formulation for computational fluid dynamics: VIII. The Galerkin/least-squares method for advective-diffusive equations". In: *Computer methods in applied mechanics and engineering* 73.2 (1989), pp. 173–189.

[33] T. J. Hughes. "Multiscale phenomena: Green's functions, the Dirichlet-to-Neumann formulation, subgrid scale models, bubbles and the origins of stabilized methods". In: *Computer Methods in Applied Mechanics and Engineering* 127.1-4 (1995), pp. 387–401. DOI: 10.1016/0045-7825(95)00844-9.

[34] T. J. Hughes, G. R. Feijóo, L. Mazzei, and J. B. Quincy. "The variational multiscale method - A paradigm for computational mechanics". In: *Computer Methods in Applied Mechanics and Engineering* 166.1-2 (1998), pp. 3–24. DOI: 10.1016/S0045-7825(98)00079-6.

[35] J. Cotela. "Applications of turbulence modeling in civil engineering". PhD thesis. Universitat Politècnica de Catalunya, 2016.

[36] R. Codina. "Stabilization of incompressibility and convection through orthogonal sub-scales in finite element methods". In: *Computer Methods in Applied Mechanics and Engineering* 190.13-14 (2000), pp. 1579–1599. DOI: 10.1016/S0045-7825(00)00254-1.

[37] R. Codina. "Stabilized finite element approximation of transient incompressible flows using orthogonal subscales". In: *Computer Methods in Applied Mechanics and Engineering* 191.39-40 (2002), pp. 4295–4321. DOI: 10.1016/S0045-7825(02)00337-7.

[38] T. J. R. Hughes, G. Scovazzi, and L. P. Franca. *Multiscale and Stabilized Methods*. 2017, pp. 1–64. ISBN: 9781119176817. DOI: 10.1002/9781119176817.ecm2051.

[39] R. Codina, S. Badia, J. Baiges, and J. Principe. "Variational Multiscale Methods in Computational Fluid Dynamics". In: *Encyclopedia of Computational Mechanics Second Edition* (2017), pp. 1–28. DOI: 10.1002/9781119176817.ecm2117.

[40] T. J. Hughes, L. Mazzei, and K. E. Jansen. "Large eddy simulation and the variational multiscale method". In: *Computing and visualization in science* 3.1 (2000), pp. 47–59.

[41] T. J. Hughes, A. A. Oberai, and L. Mazzei. “Large eddy simulation of turbulent channel flows by the variational multiscale method”. In: *Physics of fluids* 13.6 (2001), pp. 1784–1799.

[42] T. J. Hughes and G. Sangalli. “Variational multiscale analysis: The fine-scale green’s function, projection, optimization, localization, and stabilized methods”. In: *SIAM Journal on Numerical Analysis* 45.2 (2007), pp. 539–557. DOI: 10.1137/050645646.

[43] R. Codina. “A stabilized finite element method for generalized stationary incompressible flows”. In: *Computer Methods in Applied Mechanics and Engineering* 190.20-21 (2001), pp. 2681–2706.

[44] M. Avila. “Nonlinear subgrid finite element models for low Mach number flows coupled with radiative heat transfer”. Dissertation. UPC, Escola Tècnica Superior d’Enginyers de Camins, Canals i Ports de Barcelona, 2012.

[45] G. Lube and G. Rapin. “Residual-based stabilized higher-order FEM for advection-dominated problems”. In: *Computer methods in applied mechanics and engineering* 195.33-36 (2006), pp. 4124–4138.

[46] R. Codina. “Analysis of a stabilized finite element approximation of the Oseen equations using orthogonal subscales”. In: *Applied Numerical Mathematics* 58.3 (2008), pp. 264–283. DOI: 10.1016/j.apnum.2006.11.011.

[47] J. Chung and G. M. Hulbert. “A Time Integration Algorithm for Structural Dynamics With Improved Numerical Dissipation: The Generalized-α Method”. In: *Journal of Applied Mechanics* 60.2 (1993), p. 371. DOI: 10.1115/1.2900803.

[48] S. Erlicher, L. Bonaventura, and O. S. Bursi. “The analysis of the Generalized-α method for non-linear dynamic problems”. In: *Computational Mechanics* 28.2 (2002), pp. 83–104. DOI: 10.1007/s00466-001-0273-z.

[49] T. Wick. “Adaptive Finite Element Simulation of Fluid-Structure Interaction with Application to Heart-Valve Dynamics”. Dissertation. University of Heidelberg, 2011. DOI: 10.11588/heidok.00012992.

[50] T. Richter and T. Wick. “On Time Discretizations of Fluid-Structure Interactions”. In: *Multiple Shooting and Time Domain Decomposition Methods*. Ed. by T. Carraro, M. Geiger, S. Körkel, and R. Rannacher. Cham: Springer International Publishing, 2015, pp. 377–400. ISBN: 978-3-319-23321-5.

[51] E. H. Van Brummelen, K. G. Van Der Zee, V. V. Garg, and S. Prudhomme. “Flux evaluation in primal and dual boundary-coupled problems”. In: *Journal of Applied Mechanics, Transactions ASME* 79.1 (2012), pp. 1–8. DOI: 10.1115/1.4005187.

[52] T. J. Hughes, G. Engel, L. Mazzei, and M. G. Larson. "The Continuous Galerkin Method Is Locally Conservative". In: *Journal of Computational Physics* 163.2 (2000), pp. 467–488. DOI: 10.1006/jcph.2000.6577.

[53] F. Brezzi, T. J. R. Hughes, and E. Süli. "Variational approximation of flux in conforming finite element methods for elliptic partial differential equations: a model problem". In: *Rendiconti Lincei. Matematica e Applicazioni* 12.3 (2001), pp. 159–166.

[54] Y. Bazilevs and T. J. Hughes. "Weak imposition of Dirichlet boundary conditions in fluid mechanics". In: *Computers and Fluids* 36.1 (2007), pp. 12–26. DOI: 10.1016/j.compfluid.2005.07.012.

[55] J. Hoffman. "Computation of Mean Drag for Bluff Body Problems Using Adaptive DNS/LES". In: *SIAM Journal on Scientific Computing* 27.1 (Jan. 2005), pp. 184–207. DOI: 10.1137/040614463.

[56] V. John. "Reference values for drag and lift of a two-dimensional time-dependent flow around a cylinder". In: *International Journal for Numerical Methods in Fluids* 44.7 (2004), pp. 777–788. DOI: 10.1002/fld.679.

[57] V. John and G. Matthies. "Higher-order finite element discretizations in a benchmark problem for incompressible flows". In: *International Journal for Numerical Methods in Fluids* 37.8 (2001), pp. 885–903. DOI: 10.1002/fld.195.

[58] M. Giles, M. G. Larson, J. M. Levenstam, and E. Suli. *Adaptive error control for finite element approximations of the lift and drag coefficients in viscous flow.* Tech. rep. NA-76/06. Oxford University Computing Laboratory, 1997.

[59] G. F. Carey, S. S. Chow, and M. K. Seager. "Approximate boundary-flux calculations". In: *Computer Methods in Applied Mechanics and Engineering* 50.2 (1985), pp. 107–120. DOI: 10.1016/0045-7825(85)90085-4.

[60] T. J. Hughes and G. N. Wells. "Conservation properties for the Galerkin and stabilised forms of the advection-diffusion and incompressible Navier-Stokes equations". In: *Computer Methods in Applied Mechanics and Engineering* 194.9-11 (2005), pp. 1141–1159. DOI: 10.1016/j.cma.2004.06.034.

[61] M. M. Joosten, W. G. Dettmer, and D. Perić. "On the temporal stability and accuracy of coupled problems with reference to fluid-structure interaction". In: *International Journal for Numerical Methods in Fluids* 64.10-12 (Dec. 2010), pp. 1363–1378. DOI: 10.1002/fld.2333. eprint: fld.1 (DOI: 10.1002).

[62] M. Arnold and O. Brüls. "Convergence of the generalized-α scheme for constrained mechanical systems". In: *Multibody System Dynamics* 18.2 (2007), pp. 185–202. DOI: 10.1007/s11044-007-9084-0.

[63] C. Förster, W. A. Wall, and E. Ramm. "On the geometric conservation law in transient flow calculations on deforming domains". In: *International Journal for Numerical Methods in Fluids* 50.12 (Apr. 2006), pp. 1369–1379. DOI: 10.1002/fld.1093.

[64] P. Gamnitzer. "Residual-based variational multiscale methods for turbulent flows and fluid-structure interaction". Dissertation. München: Technische Universität München, 2010.

[65] C. Kadapa, W. G. Dettmer, and D. Perić. "Accurate iteration-free mixed-stabilised formulation for laminar incompressible Navier–Stokes: Applications to fluid–structure interaction". In: *Journal of Fluids and Structures* 97.January (Aug. 2020), p. 103077. DOI: 10.1016/j.jfluidstructs.2020.103077.

[66] P. Gamnitzer, V. Gravemeier, and W. A. Wall. "Time-dependent subgrid scales in residual-based large eddy simulation of turbulent channel flow". In: *Computer Methods in Applied Mechanics and Engineering* 199.13-16 (2010), pp. 819–827. DOI: 10.1016/j.cma.2009.07.009.

[67] R. Codina and J. Principe. "Dynamic subscales in the finite element approximation of thermally coupled incompressible flows". In: *International Journal for Numerical Methods in Fluids* 54.6-8 (June 2007), pp. 707–730. DOI: 10.1002/fld.1481.

[68] J. Donea, S. Giuliani, and J. P. Halleux. "An arbitrary lagrangian-eulerian finite element method for transient dynamic fluid-structure interactions". In: *Computer Methods in Applied Mechanics and Engineering* 33.1-3 (1982), pp. 689–723. DOI: 10.1016/0045-7825(82)90128-1.

[69] T. J. Hughes, W. K. Liu, and T. K. Zimmermann. "Lagrangian-Eulerian finite element formulation for incompressible viscous flows". In: *Computer Methods in Applied Mechanics and Engineering* 29.3 (1981), pp. 329–349. DOI: 10.1016/0045-7825(81)90049-9.

[70] J. Donea, A. Huerta, J.-P. Ponthot, and A. Rodríguez-Ferran. "Arbitrary Lagrangian–Eulerian Methods". In: *Encyclopedia of Computational Mechanics Second Edition*. John Wiley & Sons, Ltd, 2017, pp. 1–23. ISBN: 9781119176817. DOI: https://doi.org/10.1002/9781119176817.ecm2009.

[71] F. Nobile. "Numerical approximation of fluid-structure interaction problems with application to haemodynamics". Lausanne, 2001, p. 203. DOI: 10.5075/epfl-thesis-2458.

[72] W. G. Dettmer and D. Perić. "A fully implicit computational strategy for strongly coupled fluid-solid interaction". In: *Archives of Computational Methods in Engineering* 14.3 (2007), pp. 205–247. DOI: 10.1007/s11831-007-9006-6.

[73] R. Codina, G. Houzeaux, H. Coppola-Owen, and J. Baiges. "The fixed-mesh ALE approach for the numerical approximation of flows in moving domains". In: *Journal of Computational Physics* 228.5 (2009), pp. 1591–1611. DOI: 10.1016/j.jcp.2008.11.004.

[74] J. Hron and S. Turek. "A Monolithic FEM/Multigrid Solver for an ALE Formulation of Fluid-Structure Interaction with Applications in Biomechanics". In: *Fluid-Structure Interaction* 53 (2006), pp. 146–170. DOI: 10.1007/3-540-34596-5_7. arXiv: arXiv:1011.1669v3.

[75] T. Wick. "Variational-Monolithic ALE Fluid-Structure Interaction: Comparison of Computational Cost and Mesh Regularity Using Different Mesh Motion Techniques". In: *Modeling, Simulation and Optimization of Complex Processes HPSC 2015*. Vol. 327. Cham: Springer International Publishing, 2017, pp. 261–275. DOI: 10.1007/978-3-319-67168-0_21.

[76] T. Richter. *Fluid-structure Interactions: Models, Analysis and Finite Elements - Lecture Notes in Computational Science and Engineering*. Vol. 118. 2017, pp. 3–413. ISBN: 978-3-319-63969-7. DOI: 10.1007/978-3-319-63970-3.

[77] T. Wick. "Fluid-structure interactions using different mesh motion techniques". In: *Computers and Structures* 89.13-14 (2011), pp. 1456–1467. DOI: 10.1016/j.compstruc.2011.02.019.

[78] K. Stein, T. Tezduyar, and R. Benney. "Mesh moving techniques for fluid-structure interactions with large displacements". In: *Journal of Applied Mechanics, Transactions ASME* 70.1 (2003), pp. 58–63. DOI: 10.1115/1.1530635.

[79] J. Liu and A. L. Marsden. "A unified continuum and variational multiscale formulation for fluids, solids, and fluid–structure interaction". In: *Computer Methods in Applied Mechanics and Engineering* 337.1 (2018), pp. 549–597. DOI: 10.1016/j.cma.2018.03.045. arXiv: 1711.01322.

[80] J. Majdalani. "Exact Navier-Stokes solution for pulsatory viscous channel flow with arbitrary pressure gradient". In: *Journal of Propulsion and Power* 24.6 (2008), pp. 1412–1423. DOI: 10.2514/1.37815.

[81] J. R. Womersley. "Method for the calculation of velocity, rate of flow and viscous drag in arteries when the pressure gradient is known". In: *The Journal of Physiology* 127.3 (Mar. 1955), pp. 553–563. DOI: 10.1113/jphysiol.1955.sp005276.

[82] J. G. Heywood, R. Rannacher, and S. Turek. "Artificial boundaries and flux and pressure conditions for the incompressible Navier-Stokes equations". In: *International Journal for Numerical Methods in Fluids* 22.5 (1996), pp. 325–352. DOI: 10.1002/(SICI)1097-0363(19960315)22:5<325::AID-FLD307>3.0.CO;2-Y.

[83] R. Rannacher. “Methods for Numerical Flow Simulation”. In: *Hemodynamical Flows* 37 (2008), pp. 275–332. DOI: 10.1007/978-3-7643-7806-6_4.

[84] G. Taylor. “LXXV. On the decay of vortices in a viscous fluid”. In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 46.274 (Oct. 1923), pp. 671–674. DOI: 10.1080/14786442308634295.

[85] C. R. Ethier and D. A. Steinman. “Exact fully 3D Navier–Stokes solutions for benchmarking”. In: *International Journal for Numerical Methods in Fluids* 19.5 (1994), pp. 369–375. DOI: 10.1002/fld.1650190502.

[86] A. J. Chorin. “Numerical Solution of the Navier-Stokes Equations”. In: *Mathematics of Computation* 22.104 (Oct. 1968), p. 745. DOI: 10.2307/2004575.

[87] J. Kim and P. Moin. “Application of a fractional-step method to incompressible Navier-Stokes equations”. In: *Journal of Computational Physics* 59.2 (1985), pp. 308–323. DOI: 10.1016/0021-9991(85)90148-2.

[88] M. Schäfer, S. Turek, F. Durst, E. Krause, and R. Rannacher. “Benchmark Computations of Laminar Flow Around a Cylinder”. In: *AIAA Journal*. Vol. 35. 9. 1996, pp. 547–566. DOI: 10.1007/978-3-322-89849-4_39.

[89] S. Turek and J. Hron. “Proposal for Numerical Benchmarking of Fluid-Structure Interaction between an Elastic Object and Laminar Incompressible Flow”. In: *Fluid-Structure Interaction*. Vol. 53. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 371–385. ISBN: 9783540345954. DOI: 10.1007/3-540-34596-5_15.

[90] A. A. Mohamad. “BENCHMARK SOLUTION FOR UNSTEADY STATE CFD PROBLEMS”. In: *Numerical Heat Transfer, Part A: Applications* 34.6 (Oct. 1998), pp. 653–672. DOI: 10.1080/10407789808914008.

[91] U. Küttler and W. A. Wall. “Fixed-point fluid-structure interaction solvers with dynamic relaxation”. In: *Computational Mechanics* 43.1 (2008), pp. 61–72. DOI: 10.1007/s00466-008-0255-5.

[92] J. Degroote. “Partitioned Simulation of Fluid-Structure Interaction”. In: *Archives of Computational Methods in Engineering* 20.3 (Sept. 2013), pp. 185–238. DOI: 10.1007/s11831-013-9085-5.

[93] H.-J. Bungartz, F. Lindner, M. Mehl, and B. Uekermann. “A plug-and-play coupling approach for parallel multi-field simulations”. In: *Computational Mechanics* 55.6 (June 2015), pp. 1119–1129. DOI: 10.1007/s00466-014-1113-2.

[94] J. Alonso and A. Jameson. "Fully-implicit time-marching aeroelastic solutions". In: *32nd Aerospace Sciences Meeting and Exhibit*. Reston, Virigina: American Institute of Aeronautics and Astronautics, Jan. 1994. DOI: 10.2514/6.1994-56.

[95] R. Codina and M. Cervera. "Block-iterative algorithms for nonlinear coupled problems". In: *Advanced computational methods in structural mechanics*. Barcelona: International Centre for Numerical Methods in Engineering (CIMNE), 1996, pp. 115–134.

[96] P. Le Tallec and J. Mouro. "Fluid structure interaction with large structural displacements". In: *Computer Methods in Applied Mechanics and Engineering* 190.24-25 (Mar. 2001), pp. 3039–3067. DOI: 10.1016/S0045-7825(00)00381-9.

[97] H. G. Matthies and J. Steindorf. "Partitioned strong coupling algorithms for fluid-structure interaction". In: *Computers and Structures* 81.8-11 (2003), pp. 805–812. DOI: 10.1016/S0045-7949(02)00409-1.

[98] C. Wood, A. J. Gil, O. Hassan, and J. Bonet. "Partitioned block-Gauss-Seidel coupling for dynamic fluid-structure interaction". In: *Computers and Structures* 88.23-24 (2010), pp. 1367–1382. DOI: 10.1016/j.compstruc.2008.08.005.

[99] P. Causin, J. F. Gerbeau, and F. Nobile. "Added-mass effect in the design of partitioned algorithms for fluid-structure problems". In: *Computer Methods in Applied Mechanics and Engineering* 194.42-44 (2005), pp. 4506–4527. DOI: 10.1016/j.cma.2004.12.005.

[100] C. Förster, W. A. Wall, and E. Ramm. "Artificial added mass instabilities in sequential staggered coupling of nonlinear structures and incompressible viscous flows". In: *Computer Methods in Applied Mechanics and Engineering* 196.7 (2007), pp. 1278–1293. DOI: 10.1016/j.cma.2006.09.002.

[101] J. Degroote, P. Bruggeman, R. Haelterman, and J. Vierendeels. "Stability of a coupling technique for partitioned solvers in FSI applications". In: *Computers and Structures* 86.23-24 (2008), pp. 2224–2234. DOI: 10.1016/j.compstruc.2008.05.005.

[102] S. R. Idelsohn, F. Del Pin, R. Rossi, and E. Oñate. "Fluid-structure interaction problems with strong added-mass effect". In: *International Journal for Numerical Methods in Engineering* 80.10 (Dec. 2009), pp. 1261–1294. DOI: 10.1002/nme.2659. arXiv: 1201.4903.

[103] D. P. Mok and W. A. Wall. "Partitioned analysis schemes for the transient interaction of incompressible flows and nonlinear flexible structures". In: *Trends in Computationl Structural Mechanics*. Ed. by W. A. Wall, K.-U. Bletzinger, and K. Schweizerhof. 2001, pp. 689–698.

[104] B. M. Irons and R. C. Tuck. "A version of the Aitken accelerator for computer iteration". In: *International Journal for Numerical Methods in Engineering* 1.3 (July 1969), pp. 275–277. DOI: 10.1002/nme.1620010306.

[105] C. Michler, E. H. van Brummelen, and R. de Borst. "An interface Newton-Krylov solver for fluid-structure interaction". In: *International Journal for Numerical Methods in Fluids* 47.10-11 (Apr. 2005), pp. 1189–1195. DOI: 10.1002/fld.850.

[106] E. H. van Brummelen, C. Michler, and R. de Borst. *Interface-GMRES(R) Acceleration of Subiteration for Fluid-Structure-Interaction Problems*. Tech. rep. January. 2005.

[107] C. Michler and H. V. Brummelen. "Validation of the Interface-Gmres(R) Solution Method for Fluid-Structure Interactions". In: *ECCOMAS CFD 2006: Proceedings of the European Conference on Computational Fluid Dynamics*. 2006, pp. 1–15.

[108] C. Michler, H. van Brummelen, and R. de Borst. "An investigation of Interface-GMRES(R) for fluid–structure interaction problems with flutter and divergence". In: *Computational Mechanics* 47.1 (Jan. 2011), pp. 17–29. DOI: 10.1007/s00466-010-0519-8.

[109] U. Küttler and W. A. Wall. "Vector Extrapolation for Strong Coupling Fluid-Structure Interaction Solvers". In: *Journal of Applied Mechanics* 76.2 (Mar. 2009), p. 021205. DOI: 10.1115/1.3057468.

[110] J. Vierendeels, L. Lanoye, J. Degroote, and P. Verdonck. "Implicit coupling of partitioned fluid–structure interaction problems with reduced order models". In: *Computers & Structures* 85.11-14 (June 2007), pp. 970–976. DOI: 10.1016/j.compstruc.2006.11.006.

[111] J. Degroote, K. J. Bathe, and J. Vierendeels. "Performance of a new partitioned procedure versus a monolithic procedure in fluid-structure interaction". In: *Computers and Structures* 87.11-12 (2009), pp. 793–801. DOI: 10.1016/j.compstruc.2008.11.013.

[112] R. Haelterman. "Analytical study of the Least Squares Quasi-Newton method for interaction problems". eng. PhD thesis. Ghent University, 2009, pp. XXI, 224.

[113] J. Degroote, R. Haelterman, S. Annerel, P. Bruggeman, and J. Vierendeels. "Performance of partitioned procedures in fluid-structure interaction". In: *Computers and Structures* 88.7-8 (2010), pp. 446–457. DOI: 10.1016/j.compstruc.2009.12.006.

[114] J. Degroote. "Development of algorithms for the partitioned simulation of strongly coupled fluid-structure interaction problems". eng. PhD thesis. Ghent University, 2010, pp. XXXVII, 267. ISBN: 9789085783442.

[115] R. Haelterman, A. E. Bogaers, K. Scheufele, B. Uekermann, and M. Mehl. "Improving the performance of the partitioned QN-ILS procedure for fluid-structure interaction problems: Filtering". In: *Computers and Structures* 171 (2016), pp. 9–17. DOI: 10.1016/j.compstruc.2016.04.001.

[116] F. Lindner, M. Mehl, K. Scheufele, and B. Uekermann. "A comparison of various quasi-Newton schemes for partitioned fluid-structure interaction". In: *COUPLED PROBLEMS 2015 - Proceedings of the 6th International Conference on Coupled Problems in Science and Engineering*. April. 2015, pp. 477–488. ISBN: 9788494392832.

[117] B. W. Uekermann. "Partitioned Fluid-Structure Interaction on Massively Parallel Systems". Dissertation. München: Technische Universität München, 201.

[118] D. Blom, F. Lindner, M. Mehl, K. Scheufele, B. Uekermann, and A. van Zuijlen. "A Review on Fast Quasi-Newton and Accelerated Fixed-Point Iterations for Partitioned Fluid–Structure Interaction Simulation". In: August. 2016, pp. 257–269.

[119] K. Scheufele and M. Mehl. "Robust Multisecant Quasi-Newton Variants for Parallel Fluid-Structure Simulations—and Other Multiphysics Applications". In: *SIAM Journal on Scientific Computing* 39.5 (Jan. 2017), S404–S433. DOI: 10.1137/16M1082020.

[120] A. Bogaers, S. Kok, B. Reddy, and T. Franz. "Quasi-Newton methods for implicit black-box FSI coupling". In: *Computer Methods in Applied Mechanics and Engineering* 279 (Sept. 2014), pp. 113–132. DOI: 10.1016/j.cma.2014.06.033.

[121] A. E. Bogaers, S. Kok, B. D. Reddy, and T. Franz. "An evaluation of quasi-Newton methods for application to FSI problems involving free surface flow and solid body contact". In: *Computers and Structures* 173 (2016), pp. 71–83. DOI: 10.1016/j.compstruc.2016.05.018.

[122] T. Spenke, N. Hosters, and M. Behr. "A multi-vector interface quasi-Newton method with linear complexity for partitioned fluid–structure interaction". In: *Computer Methods in Applied Mechanics and Engineering* 361 (Apr. 2020), p. 112810. DOI: 10.1016/j.cma.2019.112810. arXiv: 2001.07947.

[123] H.-J. Bungartz, F. Lindner, M. Mehl, K. Scheufele, A. Shukaev, and B. Uekermann. "Partitioned Fluid–Structure–Acoustics Interaction on Distributed Data: Coupling via preCICE". In: *Lecture Notes in Computational Science and Engineering*. Vol. 113. 2016, pp. 239–266. ISBN: 9783319405261.

[124] M. Mehl, B. Uekermann, H. Bijl, D. Blom, B. Gatzhammer, and A. Van Zuijlen. "Parallel coupling numerics for partitioned fluid-structure interaction simulations". In: *Computers and Mathematics with Applications* 71.4 (2016), pp. 869–891. DOI: 10.1016/j.camwa.2015.12.025.

[125] A. Santiago, M. Zavala-Aké, R. Borell, G. Houzeaux, and M. Vázquez. "HPC compact quasi-Newton algorithm for interface problems". In: *Journal of Fluids and Structures* 96 (2020), p. 103009. DOI: 10.1016/j.jfluidstructs.2020.103009.

[126] R. Haelterman, A. Bogaers, and J. Degroote. "A Comparison of Different Quasi-Newton Acceleration Methods for Partitioned Multi-Physics Codes". In: *Transactions on Engineering Technologies*. February. Singapore: Springer Singapore, 2018, pp. 135–152. ISBN: 978-94-017-9803-7.

[127] Å. Björck. *Numerical Methods for Least Squares Problems*. Philadelphia: Society for Industrial and Applied Mathematics, Jan. 1996. ISBN: 978-0-89871-360-2. DOI: 10.1137/1.9781611971484.

[128] G. H. Golub and C. F. Van Loan. *Matrix Computations*. 4th. Baltimore: The Johns Hopkins University Press, 2013. ISBN: 978-1-4214-0794-4.

[129] C. H. Bischof and G. Quintana-Ortí. "Algorithm 782: Codes for Rank-Revealing QR Factorizations of Dense Matrices". In: *ACM Transactions on Mathematical Software* 24.2 (1998), pp. 254–257. DOI: 10.1145/290200.287638.

[130] G. W. Stewart. *The Gram-Schmidt Algorithm and its Variations*. Tech. rep. Maryland: Institute for Advanced Computer Studies, University of Maryland, 2004.

[131] Å. Björck. *Numerical Methods in Matrix Computations*. Vol. 59. Texts in Applied Mathematics. Cham: Springer International Publishing, 2015. ISBN: 978-3-319-05088-1. DOI: 10.1007/978-3-319-05089-8.

[132] J. W. Daniel, W. B. Gragg, L. Kaufman, and G. W. Stewart. "Reorthogonalization and Stable Algorithms for Updating the Gram-Schmidt QR Factorization". In: *Mathematics of Computation* 30.136 (Oct. 1976), p. 772. DOI: 10.2307/2005398.

[133] S. Hammarling and C. Lucas. *Updating the QR factorization and the least squares problem*. Tech. rep. Manchester Institute for Mathematical Sciences, The University of Manchester, 2008, p. 73.

[134] C. L. Lawson and R. J. Hanson. *Solving Least Squares Problems*. Philadelphia: Society for Industrial and Applied Mathematics, Jan. 1995. ISBN: 978-0-89871-356-5. DOI: 10.1137/1.9781611971217.

[135] A. van der Sluis. "Condition numbers and equilibration of matrices". In: *Numerische Mathematik* 14.1 (Dec. 1969), pp. 14–23. DOI: 10.1007/BF02165096.

[136] H. Y. Zha. "A Componentwise Perturbation Analysis of the QR Decomposition". In: *SIAM Journal on Matrix Analysis and Applications* 14.4 (Oct. 1993), pp. 1124–1131. DOI: 10.1137/0614076.

[137] A. K. Cline, C. B. Moler, G. W. Stewart, and J. H. Wilkinson. "An Estimate for the Condition Number of a Matrix". In: *SIAM Journal on Numerical Analysis* 16.2 (Apr. 1979), pp. 368–375. DOI: 10.1137/0716029.

[138] L. V. Foster. "The Probability of Large Diagonal Elements in the QR Factorization". In: *SIAM Journal on Scientific and Statistical Computing* 11.3 (May 1990), pp. 531–544. DOI: 10.1137/0911030.

[139] C. T. Pan and P. T. P. Tang. "Bounds on Singular Values Revealed by QR Factorizations". In: *BIT Numerical Mathematics* 39.4 (1999), pp. 740–756. DOI: 10.1023/A:1022395308695.

[140] C. H. Bischof and G. Quintana-Ortí. "Computing Rank-Revealing QR Factorizations of Dense Matrices". In: *ACM Transactions on Mathematical Software* 24.2 (1998), pp. 226–253. DOI: 10.1145/290200.287637.

[141] P. Dadvand, R. Rossi, and E. Oñate. "An Object-oriented Environment for Developing Finite Element Codes for Multi-disciplinary Applications". In: *Archives of Computational Methods in Engineering* 17.3 (Sept. 2010), pp. 253–297. DOI: 10.1007/s11831-010-9045-2.

[142] P. Dadvand, R. Rossi, M. Gil, X. Martorell, J. Cotela, E. Juanpere, S. Idelsohn, and E. Oñate. "Migration of a generic multi-physics framework to HPC environments". In: *Computers & Fluids* 80 (July 2013), pp. 301–309. DOI: 10.1016/j.compfluid.2012.02.004.

[143] L. Formaggia, J. F. Gerbeau, F. Nobile, and A. Quarteroni. "On the coupling of 3D and 1D Navier-Stokes equations for flow problems in compliant vessels". In: *Computer Methods in Applied Mechanics and Engineering* 191.6-7 (2001), pp. 561–582. DOI: 10.1016/S0045-7825(01)00302-4.

[144] F. Nobile. "Numerical approximation of fluid-structure interaction problems with application to haemodynamics". Dissertation. Lausanne, 2001, p. 203. DOI: 10.5075/epfl-thesis-2458.

[145] J.-F. Gerbeau and M. Vidrascu. "A Quasi-Newton Algorithm Based on a Reduced Model for Fluid-Structure Interaction Problems in Blood Flows". In: *ESAIM: Mathematical Modelling and Numerical Analysis* 37.4 (2003), pp. 631–647. DOI: 10.1051/m2an:2003049. eprint: 0608007 (math).

[146] M. A. Fernández and M. Moubachir. "A Newton method using exact jacobians for solving fluid-structure coupling". In: *Computers and Structures* 83.2-3 (2005), pp. 127–142. DOI: 10.1016/j.compstruc.2004.04.021.

[147] T. J. Hughes. "Multiscale phenomena: Green's functions, the Dirichlet-to-Neumann formulation, subgrid scale models, bubbles and the origins of stabilized methods". In: *Computer Methods in Applied Mechanics and Engineering* 127.1 (1995), pp. 387–401. DOI: https://doi.org/10.1016/0045-7825(95)00844-9.

[148] T. J. Hughes, G. R. Feijóo, L. Mazzei, and J.-B. Quincy. "The variational multiscale method—a paradigm for computational mechanics". In: *Computer Methods in Applied Mechanics and Engineering* 166.1 (1998). Advances in Stabilized Methods in Computational Mechanics, pp. 3–24. DOI: https://doi.org/10.1016/S0045-7825(98)00079-6.

[149] W. L. Wood, M. Bossak, and O. C. Zienkiewicz. "An alpha modification of Newmark's method". In: *International Journal for Numerical Methods in Engineering* 15.10 (1980), pp. 1562–1566. DOI: https://doi.org/10.1002/nme.1620151011.

[150] W. A. Wal and E. Ramm. "Fluid–structure interaction based upon a stabilized (ALE) finite element method". In: *4th World Congress on Computational Mechanics: New Trends and Applications*. Ed. by S. Idelsohn and E. Onate. CIMNE. Barcelona, 1998, pp. 1–20.

[151] W. Wall. "Fluid-Struktur-Interaktion mit stabilisierten Finiten Elementen". Dissertation. University of Stuttgart, 1999, pp. 1–248. DOI: http://dx.doi.org/10.18419/opus-127.

[152] T. Wang, R. Wüchner, S. Sicklinger, and K. U. Bletzinger. "Assessment and improvement of mapping algorithms for non-matching meshes and geometries in computational FSI". In: *Computational Mechanics* 57.5 (2016), pp. 793–816. DOI: 10.1007/s00466-016-1262-6.

[153] R. Zorrilla, R. Rossi, R. Wüchner, and E. Oñate. "An embedded Finite Element framework for the resolution of strongly coupled Fluid–Structure Interaction problems. Application to volumetric and membrane-like structures". In: *Computer Methods in Applied Mechanics and Engineering* 368 (2020), p. 113179. DOI: 10.1016/j.cma.2020.113179.

[154] Z. Drmač and Z. Bujanović. "On the Failure of Rank-Revealing QR Factorization Software – A Case Study". In: *ACM Transactions on Mathematical Software* 35.2 (July 2008), pp. 1–28. DOI: 10.1145/1377612.1377616.

[155] P. Businger and G. H. Golub. "Linear least squares solutions by householder transformations". In: *Numerische Mathematik* 7.3 (June 1965), pp. 269–276. DOI: 10.1007/BF01436084.

[156] C. H. Bischof and P. C. Hansen. "Structure-Preserving and Rank-Revealing QR-Factorizations". In: *SIAM Journal on Scientific and Statistical Computing* 12.6 (Nov. 1991), pp. 1332–1350. DOI: 10.1137/0912073.

[157] C. H. Bischof and P. C. Hansen. “A block algorithm for computing rank-revealing QR factorizations”. In: *Numerical Algorithms* 2.3 (Oct. 1992), pp. 371–391. DOI: 10.1007/BF02139475.

[158] J. R. Bischof. “A block QR factorization algorithm using restricted pivoting”. In: *Proceedings of the 1989 ACM/IEEE conference on Supercomputing - Supercomputing ’89*. New York, New York, USA: ACM Press, 1989, pp. 248–256. DOI: 10.1145/76263.76290.

[159] C. H. Bischof. “A Parallel QR Factorization Algorithm with Controlled Local Pivoting”. In: *SIAM Journal on Scientific and Statistical Computing* 12.1 (Jan. 1991), pp. 36–57. DOI: 10.1137/0912002.

[160] C. H. Bischof. “Incremental Condition Estimation”. In: *SIAM Journal on Matrix Analysis and Applications* 11.2 (Apr. 1990), pp. 312–322. DOI: 10.1137/0611021.

[161] C. Bischof and P. Tang. *Robust incremental condition estimation*. Tech. rep. Argonne, IL: Argonne National Laboratory (ANL), Mar. 1991, pp. 1–17. DOI: 10.2172/10133022.

[162] N. J. Higham. “A Survey of Condition Number Estimation for Triangular Matrices”. In: *SIAM Review* 29.4 (Dec. 1987), pp. 575–596. DOI: 10.1137/1029112.

[163] A. K. Cline, A. R. Conn, and C. F. Van Loan. “Generalizing the LINPACK condition estimator”. In: *Numerical Analysis*. Ed. by J. P. Hennart. Berlin, Heidelberg: Springer Berlin Heidelberg, 1982, pp. 73–83. ISBN: 978-3-540-38986-6. DOI: 10.1007/BFb0092961.

[164] E. Anderson et al. *LAPACK Users’ Guide*. Third. Philadelphia, PA: Society for Industrial and Applied Mathematics, 1999. ISBN: 0-89871-447-8 (paperback).

[165] J. Valdés, E. Oñate, and J. Canet. *Nonlinear analysis of orthotropic membrane and shell structures including fluid-structure interaction*. October. 2007, p. 225. ISBN: 9788496736375.

[166] D. P. Mok. “Partitionierte Lösungsansätze in der Strukturdynamik und der Fluid-Struktur-Interaktion”. Dissertation. Universität Stuttgart, 2001. ISBN: 3000079742. DOI: http://dx.doi.org/10.18419/opus-147.

[167] D. P. Mok, W. A. Wall, and E. Ramm. “Accelerated iterative substructuring schemes for instationary fluid-structure interaction”. In: *Computational Fluid and Solid Mechanics*. Elsevier, 2001, pp. 1325–1328. DOI: 10.1016/B978-008043944-0/50907-0.

[168] A. K. Birjandi, S. Shayegan, K. U. Bletzinger, and R. Wüchner. "Development of a high-fidelity partitioned Fluid–Structure Interaction model of an Omega-shaped Coriolis Mass Meter and comparison with experimental data". In: *Journal of Fluids and Structures* 110 (2022), p. 103510. DOI: 10.1016/j.jfluidstructs.2022.103510.

[169] V. M. Ferrándiz et al. *KratosMultiphysics/Kratos: Release 9.2*. Version v9.2. Sept. 2022. DOI: 10.5281/zenodo.7084882.

[170] S. Piperno, C. Farhat, and B. Larrouturou. "Partitioned procedures for the transient solution of coupled aroelastic problems Part I: Model problem, theory and two-dimensional application". In: *Computer Methods in Applied Mechanics and Engineering* 124.1-2 (1995), pp. 79–112. DOI: 10.1016/0045-7825(95)92707-9.

[171] C. Farhat, M. Lesoinne, and N. Maman. "Mixed explicit/implicit time integration of coupled aeroelastic problems: Three-field formulation, geometric conservation and distributed solution". In: *International Journal for Numerical Methods in Fluids* 21.10 (1995), pp. 807–835. DOI: 10.1002/fld.1650211004.

[172] S. Piperno and C. Farhat. *Design and Evaluation of Staggered Partitioned Procedures for Fluid-Structure Interaction Simulations*. Tech. rep. RR-3241. INRIA, Sept. 1997.

[173] C. Farhat, M. Lesoinne, P. Stern, and S. Lantéri. "High performance solution of three-dimensional nonlinear aeroelastic problems via parallel partitioned algorithms: Methodology and preliminary results". In: *Advances in Engineering Software* 28.1 (1997), pp. 43–61. DOI: 10.1016/S0965-9978(96)00028-2.

[174] S. Piperno. "Explicit/implicit fluid/structure staggered procedures with a structural predictor and fluid subcycling for 2D inviscid aeroelastic simulations". In: *International Journal for Numerical Methods in Fluids* 25.10 (Nov. 1997), pp. 1207–1226. DOI: 10.1002/(SICI)1097-0363(19971130)25:10<1207::AID-FLD616>3.0.CO;2-R.

[175] S. Piperno and C. Farhat. "Design of efficient partitioned procedures for the transient solution of aeroelastic problems". In: *Revue Europeenne des Elements* 9.6-7 (2000), pp. 655–680. DOI: 10.1080/12506559.2000.10511480.

[176] S. Piperno and C. Farhat. "Partitioned procedures for the transient solution of coupled aeroelastic problems - part II: Energy transfer analysis and three-dimensional applications". In: *Computer Methods in Applied Mechanics and Engineering* 190.24-25 (2001), pp. 3147–3170. DOI: 10.1016/S0045-7825(00)00386-8.

[177] L. De Moerloose, L. Taelman, P. Segers, J. Vierendeels, and J. Degroote. "Analysis of several subcycling schemes in partitioned simulations of strongly coupled fluid-structure interaction". In: *International Journal for Numerical Methods in Fluids* (2018), pp. 1–16. DOI: 10.1002/fld.4688.

[178] B. Rüth, B. Uekermann, M. Mehl, P. Birken, A. Monge, and H. J. Bungartz. "Quasi-Newton waveform iteration for partitioned surface-coupled multiphysics applications". In: *International Journal for Numerical Methods in Engineering* 122.19 (2021), pp. 5236–5257. DOI: 10.1002/nme.6443.

[179] M. J. Gander. *Waveform Relaxation- in Book: Encyclopedia of Applied and Computational Mathematics*. Ed. by B. Engquist. 2. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015. ISBN: 978-3-540-70528-4. DOI: 10.1007/978-3-540-70529-1. arXiv: 1209.1711.

[180] R. Haelterman, J. Degroote, D. Van Heule, and J. Vierendeels. "The Quasi-Newton Least Squares Method: A New and Fast Secant Method Analyzed for Linear Systems". In: *SIAM Journal on Numerical Analysis* 47.3 (2009), pp. 2347–2368. DOI: 10.1137/070710469.

[181] M. Soszyńska and T. Richter. "Adaptive time-step control for a monolithic multirate scheme coupling the heat and wave equation". In: *BIT Numerical Mathematics* 61.4 (2021), pp. 1367–1396. DOI: 10.1007/s10543-021-00854-3. arXiv: 2007.05372.

[182] L. Zhang. "A Multirate Approach for Fluid-Structure Interaction Computation with Decoupled Methods". In: *Communications in Computational Physics* 27.4 (June 2020), pp. 1014–1031. DOI: 10.4208/cicp.OA-2018-0305. arXiv: 1810.11933.

[183] M. Bukač, I. Yotov, and P. Zunino. "An operator splitting approach for the interaction between a fluid and a multilayered poroelastic structure". In: *Numerical Methods for Partial Differential Equations* 31.4 (2015), pp. 1054–1100. DOI: 10.1002/num.21936.

[184] M. Bukac and B. Muha. "Stability and convergence analysis of the extensions of the kinematically coupled scheme for the fluid-structure interaction". In: *SIAM Journal on Numerical Analysis* 54.5 (2016), pp. 3032–3061. DOI: 10.1137/16M1055396.

[185] J. Nunez-Ramirez, J. C. Marongiu, M. Brun, and A. Combescure. "A partitioned approach for the coupling of SPH and FE methods for transient nonlinear FSI problems with incompatible time-steps". In: *International Journal for Numerical Methods in Engineering* 109.10 (2017), pp. 1391–1417. DOI: 10.1002/nme.5331.

[186] A. Gravouil, A. Combescure, and M. Brun. "Heterogeneous asynchronous time integrators for computational structural dynamics". In: *International Journal for Numerical Methods in Engineering* 102.3-4 (Apr. 2015), pp. 202–232. DOI: 10.1002/nme.4818.

[187] M. Brun, A. Gravouil, A. Combescure, and A. Limam. "Two FETI-based heterogeneous time step coupling methods for Newmark and α-schemes derived from the energy method". In: *Computer Methods in Applied Mechanics and Engineering* 283 (2015), pp. 130–176. DOI: 10.1016/j.cma.2014.09.010.

[188] S. Karimi and K. B. Nakshatrala. "On multi-time-step monolithic coupling algorithms for elastodynamics". In: *Journal of Computational Physics* 273 (2014), pp. 671–705. DOI: 10.1016/j.jcp.2014.05.034. arXiv: 1305.6355.

[189] N. Mahjoubi, A. Gravouil, A. Combescure, and N. Greffet. "A monolithic energy conserving method to couple heterogeneous time integrators with incompatible time steps in structural dynamics". In: *Computer Methods in Applied Mechanics and Engineering* 200.9-12 (2011), pp. 1069–1086. DOI: 10.1016/j.cma.2010.12.004.

[190] N. Mahjoubi and S. Krenk. "Multi-time-step domain coupling method with energy control". In: *International Journal for Numerical Methods in Engineering* 83.13 (Sept. 2010), pp. 1700–1718. DOI: 10.1002/nme.2878. arXiv: 1010.1724.

[191] O. S. Bursi, L. He, A. Bonelli, and P. Pegon. "Novel generalized-alpha methods for interfield parallel integration of heterogeneous structural dynamic systems". In: *Journal of Computational and Applied Mathematics* 234.7 (2010), pp. 2250–2258. DOI: 10.1016/j.cam.2009.08.082.

[192] A. Prakash and K. D. Hjelmstad. "A FETI-based multi-time-step coupling method for Newmark schemes in structural dynamics". In: *International Journal for Numerical Methods in Engineering* 61.13 (2004), pp. 2183–2204. DOI: 10.1002/nme.1136.

[193] A. Combescure and A. Gravouil. "A numerical scheme to couple subdomains with different time-steps for predominantly linear transient analysis". In: *Computer Methods in Applied Mechanics and Engineering* 191.11-12 (2002), pp. 1129–1157. DOI: 10.1016/S0045-7825(01)00190-6.

[194] S. Frei, A. Heinlein, and T. Richter. "On temporal homogenization in the numerical simulation of atherosclerotic plaque growth". In: *PAMM* 21.1 (Dec. 2021), pp. 3–7. DOI: 10.1002/pamm.202100055. arXiv: 2106.09394.

[195] I. Rybak and J. Magiera. "A multiple-time-step technique for coupled free flow and porous medium systems". In: *Journal of Computational Physics* 272 (2014), pp. 327–342. DOI: 10.1016/j.jcp.2014.04.036.

[196] Y. T. Feng, K. Han, and D. R. Owen. "Coupled lattice Boltzmann method and discrete element modelling of particle transport in turbulent fluid flows: Computational issues". In: *International Journal for Numerical Methods in Engineering* 72.9 (2007), pp. 1111–1134. DOI: 10.1002/nme.2114.

[197] D. R. J. Owen, C. R. Leonardi, and Y. T. Feng. "An efficient framework for fluid-structure interaction using the lattice Boltzmann method and immersed moving boundaries". In: *International Journal for Numerical Methods in Engineering* 87.1-5 (July 2011), pp. 66–95. DOI: 10.1002/nme.2985.

[198] M. J. Gander, F. Kwok, and B. C. Mandal. "Dirichlet-Neumann and Neumann-Neumann waveform relaxation algorithms for parabolic problems". English. In: *ETNA, Electron. Trans. Numer. Anal.* 45 (2016), pp. 424–456.

[199] M. J. Gander, F. Kwok, and B. C. Mandal. "Dirichlet-Neumann and Neumann-Neumann Waveform Relaxation for the Wave Equation". In: *Domain Decomposition Methods in Science and Engineering XXII*. Springer International Publishing, 2016, pp. 501–509. ISBN: 978-3-319-18827-0. DOI: 10.1007/978-3-319-18827-0_51.

[200] A. Monge and P. Birken. "A Multirate Neumann–Neumann Waveform Relaxation Method for Heterogeneous Coupled Heat Equations". In: *SIAM Journal on Scientific Computing* 41.5 (Jan. 2019), S86–S105. DOI: 10.1137/18M1187878. arXiv: 1805.04336.

[201] A. Monge and P. Birken. "Towards a Time Adaptive Neumann-Neumann Waveform Relaxation Method for Thermal Fluid-Structure Interaction". In: *Lecture Notes in Computational Science and Engineering* 138 (2020), pp. 466–473. DOI: 10.1007/978-3-030-56750-7_54.

[202] K. Stein, T. E. Tezduyar, and R. Benney. "Automatic mesh update with the solid-extension mesh moving technique". In: *Computer Methods in Applied Mechanics and Engineering* 193.21-22 (2004), pp. 2019–2032. DOI: 10.1016/j.cma.2003.12.046.

[203] T. Richter. "A monolithic geometric multigrid solver for fluid-structure interactions in ALE formulation". In: *International Journal for Numerical Methods in Engineering* 104.5 (Nov. 2015), pp. 372–390. DOI: 10.1002/nme.4943. arXiv: 1010.1724.

[204] L. Failer and T. Wick. "Adaptive time-step control for nonlinear fluid–structure interaction". In: *Journal of Computational Physics* 366 (Aug. 2018), pp. 448–477. DOI: 10.1016/j.jcp.2018.04.021.

[205] C. Kadapa, W. Dettmer, and D. Perić. "On the advantages of using the first-order generalised-alpha scheme for structural dynamic problems". In: *Computers & Structures* 193.September (Dec. 2017), pp. 226–238. DOI: 10.1016/j.compstruc.2017.08.013.

[206] Y. Bazilevs, V. M. Calo, T. J. R. Hughes, and Y. Zhang. "Isogeometric fluid-structure interaction: Theory, algorithms, and computations". In: *Computational Mechanics* 43.1 (2008), pp. 3–37. DOI: 10.1007/s00466-008-0315-x.

[207] M. M. Joosten, W. G. Dettmer, and D. Perić. "Analysis of the block Gauss-Seidel solution procedure for a strongly coupled model problem with reference to fluid-structure interaction". In: *International Journal for Numerical Methods in Engineering* 78.7 (May 2009), pp. 757–778.

[208] W. G. Dettmer and D. Perić. "A new staggered scheme for fluid-structure interaction". In: *International Journal for Numerical Methods in Engineering* 93.1 (Jan. 2013), pp. 1–22. DOI: 10.1002/nme.4370.

[209] W. G. Dettmer, A. Lovrić, C. Kadapa, and D. Perić. "New iterative and staggered solution schemes for incompressible fluid-structure interaction based on Dirichlet-Neumann coupling". In: *International Journal for Numerical Methods in Engineering* June (Aug. 2020), nme.6494. DOI: 10.1002/nme.6494.

[210] M. Mayr, W. A. Wall, and M. W. Gee. "Adaptive time stepping for fluid-structure interaction solvers". In: *Finite Elements in Analysis & Design* submitted.June 2017 (2017), pp. 55–69. DOI: 10.1016/j.finel.2017.12.002.

[211] C. Kadapa. "A second-order accurate non-intrusive staggered scheme for the interaction of ultra-lightweight rigid bodies with fluid flow". In: *Ocean Engineering* 217.August (Dec. 2020), p. 107940. DOI: 10.1016/j.oceaneng.2020.107940.

[212] N. M. Newmark. "A Method of Computation for Structural Dynamics". In: *Journal of the Engineering Mechanics Division* 85.3 (July 1959), pp. 67–94. DOI: 10.1061/JMCEA3.0000098.

[213] H. M. Hilber, T. J. R. Hughes, and R. L. Taylor. "Improved numerical dissipation for time integration algorithms in structural dynamics". In: *Earthquake Engineering & Structural Dynamics* 5.3 (July 1977), pp. 283–292. DOI: 10.1002/eqe.4290050306.

[214] W. L. Wood, M. Bossak, and O. C. Zienkiewicz. "An alpha modification of Newmark's method". In: *International Journal for Numerical Methods in Engineering* 15.10 (1980), pp. 1562–1566. DOI: 10.1002/nme.1620151011.

[215] I. Robertson, L. Li, S. J. Sherwin, and P. W. Bearman. "A numerical study of rotational and transverse galloping rectangular bodies". In: *Journal of Fluids and Structures* 17.5 (2003), pp. 681–699. DOI: 10.1016/S0889-9746(03)00008-2.

Bisherige Titel der Schriftenreihe

Band	Titel
1	Frank Koschnick, *Geometrische Lockingeffekte bei Finiten Elementen und ein allgemeines Konzept zu ihrer Vermeidung*, 2004.
2	Natalia Camprubi, *Design and Analysis in Shape Optimization of Shells*, 2004.
3	Bernhard Thomee, *Physikalisch nichtlineare Berechnung von Stahlfaserbetonkonstruktionen*, 2005.
4	Fernaß Daoud, *Formoptimierung von Freiformschalen - Mathematische Algorithmen und Filtertechniken*, 2005.
5	Manfred Bischoff, *Models and Finite Elements for Thin-walled Structures*, 2005.
6	Alexander Hörmann, *Ermittlung optimierter Stabwerkmodelle auf Basis des Kraftflusses als Anwendung plattformunabhängiger Prozesskopplung*, 2006.
7	Roland Wüchner, *Mechanik und Numerik der Formfindung und Fluid-Struktur-Interaktion von Membrantragwerken*, 2006.
8	Florian Jurecka, *Robust Design Optimization Based on Metamodeling Techniques*, 2007.
9	Johannes Linhard, *Numerisch-mechanische Betrachtung des Entwurfsprozesses von Membrantragwerken*, 2009.
10	Alexander Kupzok, *Modeling the Interaction of Wind and Membrane Structures by Numerical Simulation*, 2009.
11	Bin Yang, *Modified Particle Swarm Optimizers and their Application to Robust Design and Structural Optimization*, 2009.
12	Michael Fleischer, *Absicherung der virtuellen Prozesskette für Folgeoperationen in der Umformtechnik*, 2009.
13	Amphon Jrusjrungkiat, *Nonlinear Analysis of Pneumatic Membranes - From Subgrid to Interface*, 2009.
14	Alexander Michalski, *Simulation leichter Flächentragwerke in einer numerisch generierten atmosphärischen Grenzschicht*, 2010.
15	Matthias Firl, *Optimal Shape Design of Shell Structures*, 2010.
16	Thomas Gallinger, *Effiziente Algorithmen zur partitionierten Lösung stark gekoppelter Probleme der Fluid-Struktur-Wechselwirkung*, 2011.
17	Josef Kiendl, *Isogeometric Analysis and Shape Optimal Design of Shell Structures*, 2011.
18	Joseph Jordan, *Effiziente Simulation großer Mauerwerksstrukturen mit diskreten Rissmodellen*, 2011.

Band	Titel
19	Albrecht von Boetticher, *Flexible Hangmurenbarrieren: Eine numerische Modellierung des Tragwerks, der Hangmure und der Fluid-Struktur-Interaktion*, 2012.
20	Robert Schmidt, *Trimming, Mapping, and Optimization in Isogeometric Analysis of Shell Structures*, 2013.
21	Michael Fischer, *Finite Element Based Simulation, Design and Control of Piezoelectric and Lightweight Smart Structures*, 2013.
22	Falko Hartmut Dieringer, *Numerical Methods for the Design and Analysis for Tensile Structures*, 2014.
23	Rupert Fisch, *Code Verification of Partitioned FSI Environments for Lightweight Structures*, 2014.
24	Stefan Sicklinger, *Stabilized Co-Simulation of Coupled Problems Including Fields and Signals*, 2014.
25	Madjid Hojjat, *Node-based parametrization for shape optimal design*, 2015.
26	Ute Israel, *Optimierung in der Fluid-Struktur-Interaktion - Sensitivitätsanalyse für die Formoptimierung auf Grundlage des partitionierten Verfahrens*, 2015.
27	Electra Stavropoulou, *Sensitivity analysis and regularization for shape optimization of coupled problems*, 2015.
28	Daniel Markus, *Numerical and Experimental Modeling for Shape Optimization of Offshore Structures*, 2015.
29	Pablo Suárez, *Design Process for the Shape Optimization of Pressurized Bulkheads as Components of Aircraft Structures*, 2015.
30	Armin Widhammer, *Variation of Reference Strategy - Generation of Optimized Cutting Patterns for Textile Fabrics*, 2015.
31	Helmut Masching, *Parameter Free Optimization of Shape Adaptive Shell Structures*, 2016.
32	Hao Zhang, *A General Approach for Solving Inverse Problems in Geophysical Systems by Applying Finite Element Method and Metamodel Techniques*, 2016.
33	Tianyang Wang, *Development of Co-Simulation Environment and Mapping Algorithms*, 2016.
34	Michael Breitenberger, *CAD-integrated Design and Analysis of Shell Structures*, 2016.
35	Önay Can, *Functional Adaptation with Hyperkinematics using Natural Element Method: Application for Articular Cartilage*, 2016.

Band	Titel
36	Benedikt Philipp, *Methodological Treatment of Non-linear Structural Behavior in the Design, Analysis and Verification of Lightweight Structures*, 2017.
37	Michael Sean Andre, *Aeroelastic Modeling and Simulation for the Assessment of Wind Effects on a Parabolic Trough Solar Collector*, 2018.
38	Andreas Apostolatos, *Isogeometric Analysis of Thin-Walled Structures on Multipatch Surfaces in Fluid-Structure Interaction*, 2018.
39	Altuğ Emiroğlu, *Multiphysics Simulation and CAD Integrated Shape Optimization in Fluid-Structure Interaction*, 2019.
40	Mehran Saeedi, *Multi-Fidelity Aeroelastic Analysis of Flexible Membrane Wind Turbine Blades*, 2017.
41	Reza Najian Asl, *Shape optimization and sensitivity analysis of fluids, structures, and their interaction using Vertex Morphing parametrization*, 2019.
42	Ahmed Abodonya, *Verification Methodology for Computational Wind Engineering Prediction of Wind Loads on Structures*, 2020.
43	Anna Maria Bauer, *CAD-integrated Isogeometric Analysis and Design of Lightweight Structures*, 2020.
44	Andreas Winterstein, *Modeling and Simulation of Wind-Structure Interaction of Slender Civil Engineering Structures Including Vibration Mitigation Systems*, 2020.
45	Franz-Josef Ertl, *Vertex Morphing for Constrained Shape Optimization of Three-dimensional Solid Structures*, 2020.
46	Daniel Baumgärtner, *On the Grid-based Shape Optimization of Structures with Internal Flow and the Feedback of Shape Changes into a CAD Model*, 2020.
47	Mohamed Khalil, *Combining Physics-based models and machine learning for an Enhanced Structural Health Monitoring*, 2021.
48	Long Chen, *Gradient Descent Akin Method*, 2021.
49	Aditya Ghantasala, *Coupling Procedures for Fluid-Fluid and Fluid-Structure Interaction Problems Based on Domain Decomposition Methods*, 2021.
50	Ann-Kathrin Goldbach, *The Cad-Integrated Design Cycle for Structural Membranes*, 2021.
51	Iñigo Pablo López Canalejo, *A Finite-Element Transonic Potential Flow Solver with an Embedded Wake Approach for Aircraft Conceptual Design*, 2022.
52	Mayu Sakuma, *An Application of Multi-Fidelity Uncertainty Quantification for Computational Wind Engineering*, 2022.

Band	Titel
53	Suneth Warnakulasuriya, *Development of Methods for Finite Element-Based Sensitivity Analysis and Goal-Directed Mesh Refinement Using the Adjoint Approach for Steady and Transient Flows*, 2022.
54	Klaus Bernd Sautter, *Modeling and Simulation of Flexible Protective Structures by Coupling Particle and Finite Element Methods*, 2022.
55	Efthymios Papoutsis, *On the incorporation of industrial constraints in node-based optimization for car body design*, 2023.
56	Thomas Josef Oberbichler, *A modular and efficient implementation of isogeometric analysis for the interactive CAD-integrated design of lightweight structures*, 2023.
57	Tobias Christoph Teschemacher, *CAD-integrated constitutive modeling, analysis, and design of masonry structures*, 2023.