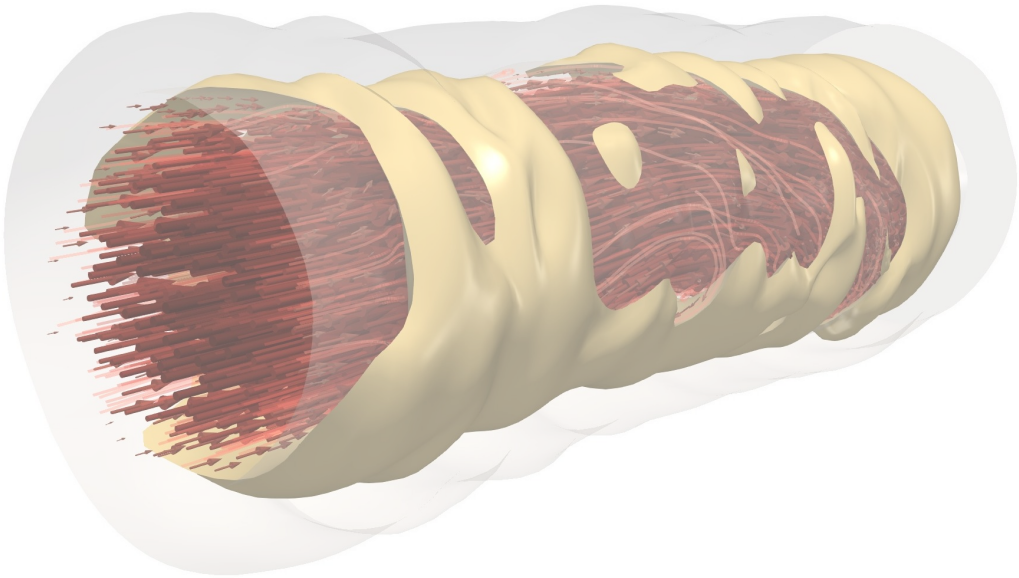
Robust and Parallel Scalable Overlapping Schwarz Preconditioners for Saddle–Point Systems with Applications to Incompressible Fluid Flow and Pharmaco–Mechanical Fluid–Structure Interaction

Lea Anna Pauline Sassmannshausen



Robust and Parallel Scalable Overlapping Schwarz Preconditioners for Saddle–Point Systems with Applications to Incompressible Fluid Flow and Pharmaco–Mechanical Fluid–Structure Interaction



UNIVERSITÄT ZU KÖLN

Inaugural Dissertation

zur

Erlangung des Doktorgrades

der Mathematisch-Naturwissenschaftlichen Fakultät

der Universität zu Köln

vorgelegt von

Lea Anna Pauline Saßmannshausen

Köln, 2026

Gutachter:	Prof. Dr. Axel Klawonn (Universität zu Köln)
	Prof. Dr. Xiao-Chuan Cai (University of Macau)
Tag der Disputation:	23.09.2026

Abstract

Computational modeling provides a promising approach to improving the understanding of cardiovascular diseases and to investigating how antihypertensive medication influences arterial mechanics and plaque stability. Such simulations require the solution of strongly coupled fluid–structure interaction (FSI) problems involving incompressible fluid flow, nonlinear solid mechanics, and deformation–diffusion processes. Their finite element discretization gives rise to large-scale, ill-conditioned nonlinear systems, making robust and parallel scalable iterative solvers and preconditioners indispensable.

This dissertation develops a scalable computational framework for pharmaco–mechanical FSI (FSCI) simulations with particular emphasis on robust and parallel scalable preconditioning techniques. As a prerequisite, a novel `FEDDLib`–`AceGen` interface is developed, enabling the efficient integration of advanced constitutive models into an existing parallel finite element framework. The interface is thoroughly validated and provides the basis for incorporating sophisticated arterial wall models accounting for active material response with a pharmacological coupling. To facilitate the development of an efficient FSCI framework, the fluid, solid, and deformation–diffusion subproblems are first investigated independently, allowing tailored solver and preconditioning strategies to be developed and analyzed for each subproblem before being transferred to the fully coupled formulation.

A particular focus is placed on incompressible fluid flow problems, where monolithic and block two-level overlapping Schwarz preconditioners are compared in a comprehensive numerical study. Different GDSW-type (Generalized Dryja–Smith–Widlund) coarse spaces are combined for the velocity and the pressure within the monolithic approach to obtain a robust and scalable preconditioning strategy. Extensive numerical experiments demonstrate improved robustness compared to the considered block preconditioners for Reynolds numbers up to 3 200, increasing Courant–Friedrichs–Lewy (CFL) numbers, and decreasing viscosity. Furthermore, the numerical investigations provide new insights into the influence of monolithic coarse-space construction on solver performance, highlighting several directions for future research.

The proposed two-level overlapping Schwarz methods are subsequently applied to nonlinear solid mechanics, deformation–diffusion, and fully coupled FSI problems. For the proposed pharmaco–mechanical deformation–diffusion formulation incorporating active smooth muscle response and drug diffusion, robust weak and strong scalability are demonstrated.

Finally, the developed framework is applied to the fully coupled pharmaco–mechanical FSI simulations. The proposed FSCI framework successfully produces the expected qualitative behavior arising from the coupled interaction of blood flow, arterial mechanics, and drug diffusion. Moreover, it is shown that physiologically meaningful boundary conditions are required, while the treatment of highly heterogeneous material parameters remains an important topic for future research.

Zusammenfassung

Die rechnergestützte Modellierung bietet einen vielversprechenden Ansatz, um das Verständnis von kardiovaskulären Erkrankungen zu verbessern und zu untersuchen, wie sich blutdrucksenkende Medikamente auf die Mechanik der Arterienwand und insbesondere auf die Stabilität atherosklerotischer Plaques auswirken. Solche Simulationen erfordern die Lösung stark gekoppelter Fluid–Struktur–Interaktionsprobleme (FSI), die inkompressible Strömung, nichtlineare Festkörpermechanik sowie pharmakologische Effekte auf die Arterienwand umfassen. Letztere werden in dieser Arbeit durch ein Verformungs–Diffusions–Problem beschrieben. Die resultierende Finite-Elemente-Diskretisierung führt zu großskaligen, schlecht konditionierten nichtlinearen Gleichungssystemen, sodass robuste und parallel skalierbare iterative Löser und Vorkonditionierer unverzichtbar sind.

In dieser Dissertation wird ein skalierbares Softwareframework für pharmakomechanische FSI-Simulationen (FSCI) entwickelt, wobei der Schwerpunkt auf robusten und parallel skalierbaren Vorkonditionierern liegt. Hierzu wird ein neu entwickeltes `FEDDLib`–`AceGen`-Interface vorgestellt, das die effiziente Integration fortschrittlicher biochemo–mechanischer Arterienwandmodelle in eine bestehende parallele Finite-Elemente-Umgebung ermöglicht. Das Interface wurde umfassend validiert und bildet die Grundlage für die Einbindung von Arterienwandmodellen, welche die aktive Materialantwort infolge der chemomechanischen Kontraktion glatter Muskelzellen sowie den Einfluss der Wirkstoffdiffusion berücksichtigen.

Zur Entwicklung eines effizienten FSCI-Frameworks werden zunächst die Teilprobleme der inkompressiblen Strömung, der Festkörpermechanik und des Verformungs-Diffusions-Problems getrennt voneinander untersucht. Dadurch können für jedes Teilproblem geeignete Löser- und Vorkonditionierungsstrategien entwickelt und analysiert werden, bevor diese auf die vollständig gekoppelte FSI-Formulierung übertragen werden.

Ein besonderer Schwerpunkt liegt auf der inkompressiblen Strömung, für die monolithische und blockbasierte zweistufige überlappende Schwarz-Vorkonditionierer in einer umfassenden numerischen Studie miteinander verglichen werden. Im monolithischen Ansatz werden verschiedene GDSW (Generalized Dryja–Smith–Widlund) Grobgitterräume für Geschwindigkeits- und Druckvariablen kombiniert, um eine robuste und skalierbare Vorkonditionierungsstrategie zu erhalten. Umfangreiche numerische Experimente zeigen gegenüber den betrachteten Block-Vorkonditionierern eine verbesserte Robustheit hinsichtlich Reynolds-Zahlen bis 3 200, steigender Courant-Friedrichs-Lewy-Zahlen (CFL-Zahlen), sowie abnehmender Viskosität. Darüber hinaus liefern die Untersuchungen neue Erkenntnisse über den Einfluss der Konstruktion monolithischer Grobgitterräume auf die Robustheit des Lösungsverfahrens und zeigen vielversprechende Ansätze für zukünftige Arbeiten auf.

Die zweistufigen überlappenden Schwarz-Verfahren werden anschließend ebenfalls auf die nichtlineare Festkörpermechanik, Verformungs-Diffusions-Probleme sowie vollständig gekoppelte

FSI-Probleme angewendet. Für die betrachtete pharmakomechanische Verformungs-Diffusions-Formulierung, welche die aktive Reaktion glatter Muskelzellen und die Wirkstoffdiffusion berücksichtigt, werden sowohl robuste schwache als auch starke Skalierbarkeit nachgewiesen.

Abschließend wird das entwickelte Verfahren auf vollständig gekoppelte pharmakomechanische FSI-Simulationen angewendet. Das entwickelte FSCI-Framework reproduziert erfolgreich das erwartete qualitative Verhalten, das aus der Kopplung von Blutströmung, arterieller Mechanik und Wirkstoffdiffusion resultiert. Gleichzeitig zeigen die Ergebnisse, dass die Formulierung physiologisch realistischer Randbedingungen sowie der Umgang mit stark heterogenen Materialparametern weiterhin wesentliche Herausforderungen darstellen und Gegenstand zukünftiger Forschung sind.

Acknowledgments

I would like to thank my doctoral advisor, Professor Axel Klawonn, for giving me the opportunity to pursue my PhD and join his research group. I am grateful for the interesting research topics, for the freedom to explore ideas in my own way and for the supportive and enjoyable working environment he created. Without his encouragement, guidance, and support, this work would not have been possible.

I would also like to express my gratitude to Professor Xiao-Chuan Cai for reviewing my thesis and serving on my thesis committee.

I am grateful to our project and collaboration partners Daniel Balzani and Sharan Nurani Ramesh from Ruhr University Bochum, as well as Oliver Rheinbach from TU Bergakademie Freiberg. In particular, I would like to thank Sharan Nurani Ramesh for his close collaboration on the development of the software interface.

I would like to thank all the colleagues who started this journey with me or joined along the way: Adam Wasiak, Natalie Kubicki, Kyrill Ho, Lucas Mager, and Colin Schwanke. I am equally grateful to the wonderful team that was already in place when I joined, including Martin Lanser, Janine Weber-Hamacher, Jascha Knepper, and Sabine Musilack-Erle, as well as to those who joined over the years: Nikita Hust, Simon Klaes, Tommaso Bevilacqua, and Yujie Gong. All of their support, kindness, and willingness to help have been indispensable — not to mention all the memorable and funny moments we shared.

Warm thanks go to my former colleague Alexander Heinlein, whose support, beginning with my master's thesis and continuing throughout the first years of my doctoral studies, has been invaluable.

I am especially grateful to Jascha Knepper. His support, particularly during the final months of this dissertation, has been immeasurable. I would like to thank him for his patience, kindness, and for carefully reviewing this thesis, which he probably read as thoroughly as if it had been his own.

I would also like to thank my family for always providing me with a place of love, encouragement, and support. In particular, I want to express my gratitude to my parents, Monika and Wolfgang, for their unconditional love, unwavering support, and constant encouragement.

I am grateful to my oldest friend, Joana, who has been by my side for more than 20 years and has witnessed every stage of my mathematical journey. Finally, I would like to thank my boyfriend, Dicseng, who unconditionally supported me through everything.

I gratefully acknowledge the scientific support and HPC resources provided by the Erlangen National High Performance Computing Center (NHR@FAU) of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) under the NHR project k105be. NHR funding is provided by federal and Bavarian state authorities. NHR@FAU hardware is partially funded by the German Research Foundation (DFG) – 440719683.

I furthermore thank the ITCC (IT Center University of Cologne) for providing compute resources on the DFG-funded HPC (High Performance Computing) system RAMSES (Research Accelerator for Modeling and Simulation with Enhanced Security) as well as support (DFG funding number: INST 216/512-1 FUGG).

The financial funding from the Deutsche Forschungsgemeinschaft (DFG) through the Priority Program 2311 “Robust coupling of continuum-biomechanical in silico models to establish active biological system models for later use in clinical applications - Co-design of modeling, numerics and usability”, project ID 465228106, is greatly appreciated.

Generative AI tools, in particular OpenAI's ChatGPT, were used during the preparation of this dissertation to assist with language editing and the improvement of text formulations. In addition, ChatGPT was used to generate one illustration, which is explicitly identified and cited in the dissertation. All AI-generated content was critically reviewed, verified, and revised by the author. The author assumes full responsibility for the scientific content, implementation, results, interpretation, and conclusions presented in this dissertation.

Contents

1. Introduction **1**

2. Incompressible Navier–Stokes Equations **6**
2.1. Governing Equations . . . 8
2.2. Finite Element Discretization . . . 8
2.3. Time Discretization . . . 11
2.4. CFL Number . . . 12

3. Solid Mechanics **13**
3.1. Arterial Wall Models . . . 15
3.1.1. Neo-Hookean Model . . . 15
3.1.2. Chemo-Mechanical SMC Model . . . 16
3.1.3. Ψ_A Model . . . 20
3.2. Finite Element Discretization . . . 20
3.3. Time Discretization . . . 22

4. The Deformation–Diffusion Problem **23**
4.1. Simple Deformation–Diffusion Model . . . 23
4.2. Pharmacological Agents . . . 24
4.3. Finite Element Discretization . . . 24
4.3.1. The Diffusion Problem . . . 24
4.3.2. The Coupled Problem . . . 26

5. Fluid–Structure Interaction Problem **28**
5.1. Finite Element and Time Discretization . . . 31

6. Block Preconditioners **35**
6.1. Block Preconditioners for a Navier–Stokes Problem . . . 35
6.1.1. PCD (Pressure Convection–Diffusion) . . . 37
6.1.2. LSC (Least-Squares Commutator) . . . 40
6.1.3. SIMPLE . . . 43
6.1.4. Augmented Lagrangian . . . 43
6.2. FaCSI Block Preconditioner . . . 45

7. Domain Decomposition **49**
7.1. GDSW-Type Coarse Spaces . . . 52
7.1.1. Interface Partition of Unity . . . 53
7.1.2. Extension to Vector-Valued Problems . . . 56

7.1.3. Discontinuous Finite Elements . . . 57
7.2. Block Schwarz Preconditioner . . . 58
7.3. Monolithic Schwarz Preconditioner . . . 59
7.4. Different Aspects of Blockwise and Monolithically Constructed Basis Functions . 60

8. Software Framework **64**
8.1. FEDDLib . . . 64
8.2. AceGen . . . 65
8.3. FEDDLib–AceGen Interface . . . 65
8.4. Trilinos . . . 68
8.4.1. Tpetra Maps . . . 69
8.4.2. Ifpack2 . . . 70
8.4.3. MueLu: Algebraic Multigrid Package . . . 71
8.5. Fast and Robust Overlapping Schwarz . . . 72
8.5.1. Constructing Overlapping Matrices . . . 74
8.5.2. Recycling Strategies . . . 75
8.6. Block Preconditioners in Teko . . . 78
8.6.1. PCD Block Preconditioner . . . 78
8.6.2. LSC Block Preconditioner . . . 80
8.6.3. SIMPLE Block Preconditioner . . . 81
8.6.4. Comparison Methodology . . . 82
8.7. FaCSI Algorithm . . . 82
8.8. Direct Solver . . . 84

9. Test Problems **86**
9.1. Backward-Facing Step . . . 86
9.2. Deformation–Diffusion Cube . . . 88
9.3. Idealized Artery . . . 89
9.4. Realistic Patient-Specific Artery . . . 90
9.5. FSI Curved Artery Benchmark . . . 93
9.6. Boundary Conditions . . . 95
9.6.1. Deformation Dependent Loads . . . 95
9.6.2. Inflow Boundary . . . 95
9.6.3. Outflow Boundary . . . 96
9.6.4. Discussion of Outflow Boundary Conditions . . . 97
9.7. Parameter Sets Passive Material Response . . . 98

10. Numerical Results **101**
10.1. Settings for Solve and Setup . . . 101
10.1.1. Inexact Newton Method . . . 102
10.1.2. Solver Settings . . . 102
10.2. Validation of FEDDLib–AceGen Interface . . . 103
10.3. Preconditioning the Solid Subproblem . . . 107
10.3.1. Neo-Hookean Material Model . . . 107

10.3.2. Ψ_A Material Model . . . 109
10.4. Preconditioning the Deformation–Diffusion Problem . . . 111
10.5. Monolithic and Block Preconditioners for the Fluid Flow Problem . . . 119
10.5.1. Monolithic Preconditioner . . . 121
10.5.2. Block Preconditioners . . . 127
10.5.3. Comparison of Specific Monolithic and Block Preconditioners . . . 138
10.6. Results for FSI Problems . . . 144
10.6.1. Fluid Subproblem . . . 146
10.6.2. Solid Subproblem . . . 148
10.6.3. Discussion on Strong Scaling . . . 151
10.7. Results for FSCI Problems . . . 156
10.7.1. Plaque Segment . . . 157
10.7.2. Further Analysis and Discussion . . . 159

11. Conclusion **163**

A. Appendix **165**
A.1. Parameters . . . 165
A.2. Preconditioning the Solid Problem . . . 166
A.3. Stationary Navier–Stokes Problem . . . 166
A.4. Transient Navier–Stokes Problem . . . 169
A.5. Comparison of Block and Monolithic Preconditioners . . . 170
A.6. Pharmaco–Mechanical FSI . . . 171

Bibliography **173**

List of Figures

3.1. Illustration of the displacement $\mathbf{d}_s$ of a beam with motion $\mathcal{A}(\mathbf{X}, t)$ in the Lagrangian viewpoint. . . . 14
3.2. Smooth muscle cell and cross-bridge cycle. . . . 17

5.1. Description of the motion of the computational fluid domain by the arbitrary Lagrangian–Eulerian (ALE) mapping $\mathcal{A}_f$. . . . 29

7.1. Visualization of three types of (overlapping) interface decompositions in a $2\times2\times2$ domain decomposition. . . . 54
7.2. Basis function for edge interface components for GDSW and RGDSW for 3×3 partition of a square domain. . . . 55
7.3. Coarse basis functions for a 2D linear elasticity problem with Dirichlet boundary condition. . . . 58
7.4. Velocity coarse basis functions $\phi_{\mathbf{u},\mathbf{u}_0}$ for a GDSW interface partition for a 2D lid-driven cavity flow with 3×3 subdomains. . . . 62
7.5. Exemplary coarse basis function for one nodal interface component for RGDSW-type partition with Option 1 and Stokes flow. . . . 63

8.1. Schematic of the `C++` factory design pattern implemented in `AssembleFEFactory` for the assembly of specific problems like elasticity. . . . 66
8.2. Functionality of the interface `AssembleFE` elements. . . . 68
8.3. Functionality of the interface for the FSI and FSCI problems, where the solid subproblem is assembled elementwise through the interface. . . . 68
8.4. Definitions of row-, column-, domain-, and rangemaps for matrix A and input vector x and output vector y in the matrix-vector multiplication $Ax = y$. . . . 70
8.5. Illustration of the different types of rowmaps. . . . 71
8.6. Illustration of the algebraic multigrid method (AMG). . . . 73
8.7. Finite element stencils for the velocity block F_N of the Navier–Stokes system and a P2–P1 discretization. . . . 75
8.8. Algebraic determination of the overlapping matrices for the F_N block of the Navier–Stokes system. . . . 76

9.1. Backward-facing step (BFS) problem. . . . 87
9.2. Number of coarse functions for different numbers of processor cores for the backward-facing step problem. . . . 88
9.3. Structured partition of the unit cube. . . . 89
9.4. Axial pulling of a cube. . . . 90

9.5. Deformation and diffusion results corresponding to the axial pulling of a cube test case. . . . 90
9.6. Idealized arterial segment. . . . 91
9.7. Realistic patient-specific artery with plaque layer, media, and adventitia from [7]. 91
9.8. Boundary conditions of fluid and solid domain of the realistic patient-specific artery. . . . 93
9.9. Boundary conditions imposed for the FSI curved artery benchmark from [8, 10]. 94
9.10. Fiber directions and flow rate for FSI curved artery benchmark. . . . 94

10.1. Structure of different interface validation tests. . . . 104
10.2. Comparison of outflow area and pressure computed in `FEDDLib` with `FEDDLib–AceGen` interface for the FSI curved artery benchmark. . . . 106
10.3. Comparison of deformation and displacement computed in `FEDDLib` with `FEDDLib–AceGen` interface for the FSI curved artery benchmark. . . . 107
10.4. Comparison of flow rate, area and pressure results for the curved artery FSI benchmark with drug diffusion. . . . 108
10.5. Weak scaling results for the neo-Hookean material model for the axial pulling of a cube. . . . 109
10.6. Different fiber angles considered for the axial pulling of a cube. . . . 111
10.7. Comparison of different acceleration values for the active material response and weak scaling. . . . 114
10.8. Average linear iteration count per Newton step over 150 s for different material settings and effects for the axial pulling of a cube using the Ψ_{SMC} model with acceleration of active response. . . . 116
10.9. Average linear iteration count per Newton step over 150 s for different material settings and effects for the axial pulling of a cube using the Ψ_{SMC} model without acceleration of active response. . . . 117
10.10. Measured error between fully implicit (ChI) and explicit (ChE) approach for the deformation–diffusion problem. . . . 117
10.11. Long-term simulation of the deformation–diffusion problem with Ψ_{SMC} model. 118
10.12. Strong scaling study for realistic patient-specific artery. . . . 119
10.13. Comparison of different coarse space combinations for the velocity (first component) and pressure (second component) for a P1–P1 discretization and the monolithic preconditioner. . . . 122
10.14. Comparison of different coarse space combinations for the velocity (first component) and pressure (second component) for a P2–P1 discretization and the monolithic preconditioner. . . . 123
10.15. Comparison of different finite element discretizations for the monolithic preconditioner. . . . 125
10.16. Comparison of total time (setup + solve time) for different finite element discretizations for the monolithic preconditioner. . . . 126
10.17. Comparison of different finite element discretizations for the monolithic preconditioner for an increasing Reynolds number. . . . 127

10.18. Comparison of different finite element discretizations for block-triangular preconditioners. . . . 131
10.19. Sparsity pattern of velocity block (0,0) of linearized Navier–Stokes saddle point system. . . . 132
10.20. Comparison of different components of setup and solve time for the triangular, augmented Lagrangian, and modified augmented Lagrangian block preconditioners. . . . 134
10.21. Weak scaling results for the modified augmented Lagrangian approach using the `MueLu` algebraic multigrid preconditioner. . . . 137
10.22. Comparison of different coarse space combinations for the velocity (first component) and Schur complement components (second component) for P2–P1 discretization for block preconditioners. . . . 139
10.23. Weak scaling comparison of block and monolithic preconditioners. . . . 140
10.24. Comparison of block and monolithic preconditioners for increasing Reynolds number. . . . 141
10.25. Comparison of block and monolithic preconditioners for increasing Reynolds and CFL numbers. . . . 142
10.26. Realistic patient-specific artery for fluid flow problem. . . . 142
10.27. Comparison of block and monolithic preconditioners for the transient realistic artery problem (Figure 10.26) by varying the Reynolds and CFL numbers. . . . 144
10.28. Comparison of strong scaling of block and monolithic preconditioners for transient realistic artery problem (Figure 10.26). . . . 145
10.29. Results of computation of *Test (b)* at $t = 0.56\,\mathrm{s}$ and time step size $\Delta t = 0.001\,\mathrm{s}$ from Figure 10.30 (left). . . . 148
10.30. Settings for test cases concerning the robustness of the solver and preconditioner for the fluid and solid subproblems in FSI. . . . 149
10.31. Results of robustness tests in FSI for the fluid and the solid subproblems. . . . 150
10.32. FSI strong scaling results with individual focus on the fluid and solid subproblems. 150
10.33. Results of inflow and outflow area for FSI problem using realistic patient-specific artery from Section 9.4 with reference fluid pressure $p_{\mathrm{ref}} = 10.66\,kPa$. . . . 151
10.34. Results of deformation of FSI problem using realistic patient-specific artery from Section 9.4 with reference fluid pressure $p_{\mathrm{ref}} = 10.66\,kPa$ using *NH3, Set 1*, and *Set 2*. . . . 152
10.35. Results of displacement of FSI problem using realistic patient-specific artery from Section 9.4 with reference fluid pressure $p_{\mathrm{ref}} = 10.66\,kPa$ and parameter *Set 3*. . . . 152
10.36. Visualization of the unstructured METIS partition of the arterial wall and fluid domain into 416 subdomains . . . 154
10.37. Degrees of freedom per subdomain versus number of subdomains for the fluid and solid subdomain. . . . 154
10.38. Detailed runtime analysis of results in Table 10.27 for *Set 3*. . . . 155
10.39. Results of full FSCI simulation using idealized arterial segment. . . . 157

10.40. Average linear iteration count per Newton step per time step plotted over 2 503 s for long-term FSCI simulations. . . . 158
10.41. Schematic depiction of using local segments of an atherosclerotic artery for simulations. . . . 158
10.42. Strong scaling study for plaque segment. . . . 159
10.43. Arterial segment containing distinct plaque components and linear solver analysis. 160
10.44. Average linear iteration count per Newton step over 100 s for different material settings for the plaque segments in FSCI simulation. . . . 162

A.2. Average linear iteration count per Newton step per time step over 2503 s for long-term SCI simulations. . . . 166
A.1. Results of displacement of SCI problem using realistic patient-specific artery from Section 9.4 with different internal pressure values and different arterial wall models. . . . 166
A.3. Comparison of different coarse space combinations for the velocity (first component) and pressure (second component) for P2–P1 discretization for the monolithic preconditioner. . . . 168
A.4. Comparison of different finite element discretizations for LSC block-triangular preconditioner. . . . 168
A.5. Results of full FSCI simulation using idealized arterial segment and reference pressure 10.66 kPa. . . . 171

List of Tables

6.1. Operators of the different preconditioning strategies. 37
6.2. Boundary conditions used in the PCD preconditioner. 40

7.1. Coarse space dimensions for the scalar Laplace problem and the different GDSW-type coarse spaces. 53

8.1. Recycling strategies. 77
8.2. Comparison of setup and solve times of the monolithic preconditioner of recomputing it during the first k Newton iterations (NR-k). 77
8.3. Stationary simulation with PCD block preconditioner of backward-facing step with P2–P1 discretization, $H/h = 9$, and 1 125 cores. 79
8.4. Approximations of inverses arising in monolithic and block preconditioners used in simulations for fluid flow problems. 82

9.1. Subdomain resolution H/h and number of degrees of freedom of the backward-facing step geometry. 87
9.2. Transient backward-facing step (BFS) problem: CFL number c_{CFL} (see (2.17)) in relation to the element diameter h determined by $H/h = 9$ and the number of subdomains N. The time step size is $\Delta t = 0.05\,s$, the initial velocity $v_{max} = 1\,cm/s$. 88
9.3. Subdomain resolution H/h and number of degrees of freedom of the axial pulling of a cube. 89
9.4. Configuration details for the realistic artery problem for fluid simulations. . . . 93
9.5. Material parameters for the FSI curved artery benchmark problem 94
9.6. Parameter settings regarding the absorbing boundary condition in (9.4) for FSI curved artery benchmark defined in Section 9.5. 97
9.7. Material parameters for the neo-Hookean material model. 98
9.8. Material parameters for the Ψ_A material model. 99
9.9. Material parameters for the Ψ_A material model for the realistic patient-specific artery with parameters for the different material components. 99
9.10. Passive material parameters for the Ψ_{SMC} material model for the arterial segment with distinct plaque regions. 100

10.1. Solver settings for the nonlinear and linear solvers. 103
10.2. Comparison between `FEDDLib` and new interface implementation for a linear elasticity material model. 105
10.3. Comparison between `FEDDLib` and new interface implementation for neo-Hookean material model. 105

10.4. Comparison between AceGen/AceFEM and FEDDLib computations of the simple deformation–diffusion model. . . . 105
10.5. Weak scaling results for the neo-Hookean material model for the axial pulling of a cube. . . . 110
10.6. Average linear iteration count per Newton step for the Ψ_A material model for surface loads of 10.66 kPa and 21.33 kPa. . . . 111
10.7. Solver analysis of the Ψ_{SMC} material model with active material response for acceleration parameter 4 on 2 197 cores. . . . 113
10.8. Details on weak scaling results for the Ψ_{SMC} material model with active material response for acceleration parameter 4 . . . 115
10.9. Number of cores assigned to solve the coarse problem for the results shown in Figure 10.16 for the stationary BFS problem. . . . 121
10.10. Comparison of setup and solve time of the three best coarse space combinations for P1–P1 and P2–P1 discretization on 4 608 cores. . . . 124
10.11. Comparison of boundary conditions in PCD block preconditioner for stationary BFS problem. . . . 128
10.12. Comparison of block preconditioners for different Reynolds numbers for stationary BFS problem. . . . 129
10.13. Comparison of different coarse spaces for PCD block preconditioner with different Reynolds numbers for P2–P1 discretization. . . . 130
10.14. Experimental analysis of optimal value γ for augmented and modified Lagrangian approach. . . . 132
10.15. Comparison of runtime for various preconditioners with the augmented Lagrangian block preconditioner. . . . 133
10.16. Comparison of runtime for various preconditioners with use of RILUK for first level in triangular preconditioner, augmented Lagrangian and modified augmented Lagrangian block preconditioner. . . . 135
10.17. Parameters for the smoothed aggregation algebraic multigrid solver from the Trilinos package MueLu. . . . 136
10.18. Comparison of runtime for modified augmented Lagrangian block preconditioner using MueLu package. . . . 136
10.19. Comparison of average linear iterations for modified augmented Lagrangian block preconditioner using MueLu package and different damping values. . . . 136
10.20. Comparison of different boundary condition strategies for the PCD block preconditioner and a transient simulation. . . . 138
10.21. Comparison of weak scaling between block and monolithic preconditioners. . . . 143
10.22. Number of processor cores assigned to the coarse problem for the results in Figure 10.28. . . . 143
10.23. Reynolds number examples for the FSI test cases . . . 146
10.24. Parameters for the fluid and solid in FSI simulations. . . . 147
10.25. Detailed results for the strong scaling plot shown in Figure 10.32 (left). . . . 147
10.26. Resulting average iteration counts for FSI patient-specific artery with varying reference pressure values for different material models. . . . 153

10.27. Detailed strong scaling results for realistic patient-specific artery with different arterial wall models. 153

A.1. Parameters for the chemical model in the active material response in Ψ_{SMC} from Section 3.1.2. Adopted from [168]. 165
A.2. Parameters for the active chemo-mechanical material response in Ψ_{SMC} from Section 3.1.2. Adopted from [168]. 165
A.3. Detailed strong scaling results for realistic patient-specific artery with different arterial wall models in deformation–diffusion problem. 167
A.4. Comparison of different finite element discretizations for the monolithic preconditioner for an increasing Reynolds number. 169
A.5. Comparison of block and monolithic preconditioners for an increasing Reynolds number. 170

1. Introduction

This work considers fluid–structure interaction problems that incorporate advanced arterial wall models accounting for the influence of pharmaceutical agents on vascular behavior. The resulting mathematical models combine incompressible fluid flow, nonlinear solid mechanics, and deformation–diffusion processes, leading to strongly coupled nonlinear systems of equations. Their finite element discretizations give rise to large sparse and ill-conditioned linear systems, making robust and scalable iterative solvers and preconditioners indispensable for realistic, patient-specific simulations.

This work focuses on the development and analysis of robust and parallel scalable preconditioning strategies for the resulting fluid, solid, deformation–diffusion, and coupled FSI problems. Particular emphasis is placed on incompressible fluid flow through a comprehensive comparison of monolithic and block preconditioning strategies, including the introduction and analysis of newly proposed coarse-space variants. In addition, a newly developed `FEDDLib-AceGen` interface enables the efficient integration of advanced arterial wall models into an existing parallel computational framework.

The motivation for this work originates from the computational modeling of cardiovascular diseases, with a particular focus on coronary artery disease and atherosclerosis. In particular, one objective is to investigate how antihypertensive drugs influence the mechanical behavior of arterial tissue and, consequently, the stress distribution within atherosclerotic plaques. Such simulations may contribute to a better understanding of the influence of pharmaceutical treatment on plaque stability and the risk of plaque rupture. In the long term, computational simulations combined with patient-specific data can support clinical decision making by providing additional insight into quantities that cannot be measured directly. To enable such investigations, two key components are required. First, realistic constitutive models are needed that accurately capture the interaction between arterial tissue and pharmaceutical agents. Second, these models must be integrated into a computational framework that enables robust and scalable simulations of the resulting coupled multiphysics problems. This motivates not only the development and efficient integration of sophisticated constitutive models into an existing finite element framework while maintaining flexibility and parallel scalability, but also the design and analysis of robust preconditioners for the resulting linearized systems.

Fluid–structure interaction (FSI) problems can generally be solved using either partitioned or monolithic coupling schemes. In partitioned approaches, the fluid and solid subproblems are solved separately and coupled through suitable interface conditions, whereas monolithic approaches solve the coupled fluid–structure system simultaneously. A review and comparison of both approaches is given in [29]. In the setting of biomechanics, where blood interacts with comparatively soft arterial tissue and the fluid and structure possess similar densities, partitioned approaches may suffer from the added-mass effect [31]. Under these conditions, monolithic coupling schemes are more flexible

and robust; see the comparison in [29]. Consequently, many state-of-the-art solvers for FSI in biomechanics make use of a monolithic coupling [119, 101, 36, 118, 117, 151, 173, 150, 33, 137, 167].

In this work, the FaCSI preconditioning strategy introduced in [36] is used. The acronym *FaCSI* refers to the Factorized fluid–structure interaction block system, the use of static Condensation, and the application of the SIMPLE preconditioner for the fluid subsystem. The resulting block factorization separates the coupled problem into fluid, structure, and geometry subproblems, allowing the corresponding inverses to be approximated independently. To facilitate a robust FSI framework, the incompressible fluid flow, solid, and deformation–diffusion problems are first investigated separately.

Among the parallel preconditioning techniques for large-scale finite element problems are additive overlapping Schwarz methods [159, 163]. These methods are based on an overlapping decomposition of the computational domain into subdomains, where local problems are solved independently and combined to form a global preconditioner. While one-level overlapping Schwarz methods can be effective, their convergence generally deteriorates as the number of subdomains increases. Consequently, scalable parallel algorithms require a second, global coarse level that communicates global information across the computational domain and yields convergence rates that are largely independent of the number of subdomains. The construction of an effective coarse space is therefore one of the key ingredients of scalable overlapping Schwarz methods. In this work, Generalized Dryja–Smith–Widlund (GDSW)-type coarse spaces [40, 41] are considered. Several approaches have been proposed to reduce the coarse-space dimension of GDSW methods while preserving robustness and improving parallel scalability [43, 44, 45]. These include the reduced-dimension GDSW (RGDSW) coarse space [45]. Furthermore, GDSW*, first introduced in [96] and further developed in [82], represents an intermediate approach between GDSW and RGDSW for three-dimensional problems. The parallel scalability could be further improved by solving the coarse problem inexactly using another GDSW-type preconditioner resulting in a three-level GDSW method; cf. [153, 90, 87, 122]. For highly heterogeneous multiscale problems, adaptive coarse spaces based on local generalized eigenvalue problems have been proposed [67, 161, 81, 93, 116, 1], resulting in methods that are robust with respect to large coefficient contrasts.

Mixed finite element discretizations of the incompressible Navier–Stokes equations give rise to linear systems with saddle-point structure, where the unknowns consist of velocity and pressure variables. After linearization, the resulting systems are generally nonsymmetric and indefinite. Therefore, we employ the Generalized Minimal Residual (GMRES) method [155] as the Krylov solver. Since the arising linear systems are typically ill-conditioned, efficient preconditioners are essential to achieve robust and scalable performance. Comprehensive reviews of numerical methods for saddle-point problems and, in particular, finite element discretizations, iterative solvers, and preconditioners for the incompressible Navier–Stokes equations are given in [16, 55].

One possibility is to construct a monolithic overlapping Schwarz preconditioner for the complete saddle-point system. This approach has been introduced in [109, 110] with Lagrangian coarse spaces and extended to GDSW-type coarse spaces in [77, 78]. Recently, the performance of monolithic overlapping Schwarz preconditioners has been compared with block preconditioning strategies for incompressible fluid flow in [82], building on the developments in [77, 78, 96].

The application of monolithic RGDSW coarse spaces in nonlinear Schwarz preconditioners

has recently been investigated in [108, 76, 95]. Beyond incompressible fluid flow, monolithic overlapping Schwarz preconditioners have also been successfully applied to other coupled multiphysics problems, including chemo-mechanical models [106, 105] and laser beam welding simulations [24, 23, 22].

An alternative to monolithic preconditioners is to exploit the block structure of the saddle-point systems. Block preconditioners are based on an approximation of the block factorization, where the main challenge is the efficient approximation of the Schur complement. For the steady Stokes problem with an inf–sup stable discretization, the Schur complement is spectrally equivalent to a suitably scaled pressure mass matrix; see [54, 114]. Consequently, the pressure mass matrix provides an effective approximation of the Schur complement for low Reynolds number flows. As the Reynolds number increases, however, the pressure mass matrix no longer captures the influence of the convective term sufficiently well, and more sophisticated Schur complement approximations become necessary.

In this work, we consider the Pressure Convection–Diffusion (PCD) [158, 104, 55, 157, 57, 99, 37], Least-Squares Commutator (LSC) [56, 55, 157, 52, 57], augmented Lagrangian [17, 20, 55, 157, 57, 60, 18, 19], and SIMPLE (Semi-Implicit Method for Pressure-Linked Equations) [139, 142, 51, 157, 37] block preconditioners. While all these methods exploit the block structure of the saddle-point system, they differ primarily in the construction of the Schur complement approximation.

For the first time, different types of coarse spaces for the velocity and pressure components of the two-level overlapping Schwarz methods are compared, which improves robustness for the monolithic preconditioner. These results were reported in [82] and are part of this work. By varying Reynolds and CFL numbers in a well-known benchmark problem as well as in a realistic setting, robustness of the monolithic and block overlapping Schwarz approaches is studied and compared.

Similarly, the two-level overlapping Schwarz preconditioners are considered for the solid problem. The constitutive models considered in this work are primarily based on quasi-incompressible and anisotropic materials, which can exhibit an active material response. Studies in [85, 84, 86, 79] show robustness and parallel scalability using the GDSW and RGDSW coarse spaces for linear elasticity problems; for two-level methods used for mixed linear elasticity formulations we refer to [109, 42, 43]. In [84, 85] two-level overlapping Schwarz methods were considered for the solid component in FSI, comparing a linear, a neo-Hookean, and a hyperelastic anisotropic material model. The monolithic two-level overlapping Schwarz preconditioner was applied to a deformation–diffusion type problem in [106, 105, 9].

To the best of our knowledge, this work presents the first application of two-level overlapping Schwarz preconditioners to a material model with an active material response. While monolithic two-level overlapping Schwarz preconditioners have previously been applied to a deformation–diffusion problem in the context of hydrogel swelling [106, 105], the present work considers a fundamentally different application, namely the interaction between pharmaceutical agents and arterial tissue. The active material response is investigated in direct comparison with its passive component, including analysis of the influence of drug diffusion. Preliminary results were reported in [9]; the analysis is extended in the present work.

The FaCSI preconditioning strategy provides a block factorization of the monolithic FSI system, separating the coupled problem into fluid, structural, and geometry subproblems. This

factorization allows the inverse operators associated with the individual subproblems to be approximated independently.

Within the FaCSI framework, the fluid subsystem can be preconditioned either by a block preconditioner or by a monolithic overlapping Schwarz preconditioner. The latter approach was first investigated in [96] and further developed in [107]. Building on these developments, the present work integrates the proposed preconditioning strategies into a complete FSI framework and investigates their robustness for realistic biomechanical applications. In particular, it extends previous work on two-level overlapping Schwarz methods for the structural subsystem of FSI involving linear elasticity, neo-Hookean, and anisotropic hyperelastic material models [84, 85], as well as the first investigations of two-level methods for the fluid subsystem reported in [96].

Besides the development of robust numerical algorithms, an efficient and scalable software framework is essential for solving large-scale multiphysics problems. Such a framework must not only support advanced constitutive models and coupled finite element formulations but also provide scalable implementations of the corresponding linear solvers and preconditioners suitable for high-performance computing. The two-level overlapping Schwarz preconditioners with GDSW-type coarse spaces considered in this work are implemented within the `FROSch` [83, 84] package of `Trilinos` [164]. In numerous publications over the past years, `FROSch` has been employed for a variety of physics problems [84, 77, 91, 83, 89, 92, 183, 90, 76, 82, 121]. The references listed here represent only a selection of the work related to `FROSch`.

In addition to the development of scalable preconditioners, this work addresses the efficient integration of advanced constitutive models into the finite element software `FEDDLib` [61, 96]. Together with the parallel Schwarz implementation provided by `FROSch`, this results in a flexible framework for large-scale fluid–structure interaction simulations involving complex material models. A central software contribution of this work is the development of an interface between `FEDDLib` and `AceGen`, enabling the efficient integration of advanced constitutive models generated with `AceGen` [120] into the existing finite element framework. To ensure the correctness and reliability of the generated finite element routines, the interface is validated through comparisons with previously existing `FEDDLib` implementations, `AceFEM` computations, and benchmark problems.

Based on this motivation, the thesis develops the required computational framework gradually. Following the validation of the interface, the fluid, structural, and deformation–diffusion subproblems are studied independently, enabling the development and validation of efficient implementation, solver, and preconditioning strategies. These building blocks are subsequently combined within a monolithic fluid–structure interaction framework.

Chapter 2 introduces the incompressible Navier–Stokes equations together with their finite element discretization, time discretization, and Newton linearization. Similarly, Chapter 3 presents the governing equations of nonlinear solid mechanics and introduces the arterial wall models considered in this work. The chemo-mechanical smooth muscle cell model summarized in Section 3.1.2 will be part of the doctoral thesis of Sharan Nurani Ramesh, and recent further developments, parameter fitting and implementation in `AceGen` are attributed to him.

Chapter 4 introduces the coupled deformation–diffusion problem. A simple coupling strategy for a neo-Hookean material model is proposed in Section 4.1, while Section 4.2 extends the

formulation by incorporating the influence of pharmacological agents, specifically calcium channel blockers (CCBs), on smooth muscle cell contraction.

Building on the individual subproblems, Chapter 5 derives the fully coupled fluid–structure interaction formulation together with its extension to include the deformation–diffusion problem. The corresponding linearized system forms the basis of the FaCSI block factorization introduced in Section 6.2.

The block preconditioning strategies considered in this work are presented in Chapter 6. Section 6.1 discusses block preconditioners for the Navier–Stokes equations, while Section 6.2 introduces the FaCSI block preconditioner for the coupled FSI problem.

The inverse operators arising in these block preconditioners are approximated using two-level overlapping Schwarz methods, which are described in Chapter 7. After introducing the method using a simplified model problem, we present the considered GDSW-type coarse spaces. The construction of monolithic and block overlapping Schwarz preconditioners is then described in Section 7.3 and Section 7.1.3, respectively.

Chapter 8 describes the software framework used throughout this work. At its core is the finite element library `FEDDLib`, introduced in Section 8.1, which is built upon the `Trilinos` project described in Section 8.4. In addition, the relevant `Trilinos` packages employed in this work are briefly reviewed. The chapter also provides a short overview of `AceGen`, which is used for the implementation of the chemo-mechanical smooth muscle cell model, and presents the newly developed `FEDDLib-AceGen` interface in Section 8.3.

Finally, Chapter 9 summarizes the test problems and computational setups used throughout this work, including the backward-facing step benchmark (Section 9.1) and simulations on a realistic patient-specific arterial geometry (Section 9.4).

The numerical results are presented in Chapter 10 and are organized as follows. First, the `FEDDLib-AceGen` interface is validated using a variety of test problems; see Section 10.2. Based on the developed interface, the solid preconditioner is investigated for different constitutive models in Section 10.3. Subsequently, the coupled deformation–diffusion problem which includes the chemo-mechanical smooth muscle cell activation is studied in Section 10.4, where weak and strong scalability, coupling strategies, and the qualitative behavior of the proposed material models are analyzed. The fluid preconditioners are then assessed in Section 10.5 through an extensive comparison of monolithic and block overlapping Schwarz preconditioners for incompressible flow. The considered test cases include stationary and transient simulations with Reynolds numbers up to 3 200, as well as weak and strong scaling studies. Building on these results, Section 10.6 investigates the fully coupled FSI problem. Robustness is assessed for the realistic patient-specific artery under varying inflow rates and internal pressure values. Finally, Section 10.7 presents the results for the fully coupled pharmaco–mechanical FSI problem. Besides demonstrating the applicability of the proposed computational framework, this section discusses the remaining challenges of the FSCI formulation, providing motivation for future research.

2. The Incompressible Navier–Stokes Equations

In this chapter, the incompressible Navier–Stokes equations are introduced and discretized. They form a system of nonlinear partial differential equations describing the motion of viscous Newtonian fluids. They express the conservation of linear momentum and the conservation of mass. The governing equations are derived from the underlying physical principles of continuum mechanics. In particular, the momentum equation is obtained by applying Newton's second law to an infinitesimal fluid element while accounting for pressure forces, viscous stresses, and external body forces, whereas the continuity equation follows from the conservation of mass. The derivation of the equations follows [55, pp. 1–8]. In continuum mechanics, two complementary descriptions of motion are commonly employed: the Lagrangian and the Eulerian viewpoints. The Lagrangian description follows individual material particles throughout their motion and is therefore employed for solid mechanics; see Chapter 3. In contrast, the Eulerian description considers physical quantities, e.g., the velocity and pressure, at fixed points in space and is the framework for fluid mechanics. Therefore, the incompressible Navier–Stokes equations are formulated in the Eulerian description.

Consider a fluid with density ρ_f occupying the domain Ω_f. The fluid velocity at position $\mathbf{x} = (x_1, x_2, x_3)$ and time t is defined as

$$\mathbf{u} = (u_x, u_y, u_z) := \lim_{\delta t \to 0} \frac{\delta \mathbf{x}}{\delta t}.$$

An arbitrary volume subset $D \subset \Omega_f$ of the fluid domain Ω_f with fixed and closed surface boundary ∂D is considered. Following the rule of conservation of mass, the rate of change of mass in D equals the amount of fluid flowing into and out of D across ∂D. Using the divergence theorem, the boundary integral can be written as a volume integral, which yields

$$0 = \frac{\mathrm{d}}{\mathrm{d}t} \int_D \rho_f \, \mathrm{d}\Omega_f + \int_{\partial D} \rho_f \mathbf{u} \cdot \mathbf{n} \, \mathrm{d}S = \int_D \left(\frac{\partial \rho_f}{\partial t} + \nabla \cdot (\rho_f \mathbf{u}) \right) \mathrm{d}\Omega_f,$$

with outward surface normal vector $\mathbf{n}$. Since D is any arbitrary volume, it follows that

$$\frac{\partial \rho_f}{\partial t} + \nabla \cdot (\rho_f \mathbf{u}) = 0 \qquad \text{in } \Omega_f.$$

For an incompressible and homogeneous fluid with constant density with respect to time and space, we have

$$\nabla \cdot \mathbf{u} := \frac{\partial u_x}{\partial x} + \frac{\partial u_y}{\partial y} + \frac{\partial u_z}{\partial z} = 0 \qquad \text{in } \Omega_f.$$

In order to express the conservation of momentum, the acceleration of the fluid is defined by

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = \lim_{\delta t \to 0} \frac{\delta \mathbf{u}}{\delta t} = \frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u}.$$

The fluid acceleration is the convective derivative of the velocity. Note that the acceleration is nonlinear in $\mathbf{u}$.

First, we will derive the Euler equations for an ideal fluid. An ideal fluid is defined as incompressible, homogeneous, and inviscid. Additionally, the only forces acting on a fluid volume D are due to the pressure p and an external body force $\boldsymbol{f}$, e.g., gravity. Therefore, the total forces are expressed as

$$-\int_{\partial D} p\,\mathbf{n}\,dS + \int_D \rho_f \boldsymbol{f}\ \mathrm{d}\Omega_f.$$

To relate surface and volume integrals, the divergence theorem is applied to transform the pressure contribution into a volume integral

$$-\int_{\partial D} p\,\mathbf{n}\,dS = -\int_D \nabla p\ \mathrm{d}\Omega_f.$$

Newton's second law of motion applied to the fluid in D states that the rate of change of momentum of the fluid in D equals the sum of the external forces

$$\int_D \rho_f \left(\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u}\right)\mathrm{d}\Omega_f = -\int_{\partial D} p\,\mathbf{n}\,\mathrm{d}S + \int_D \rho_f \mathbf{f}\ \mathrm{d}\Omega_f.$$

Substituting the pressure integral identity and rearranging yields

$$\int_D \left[\rho_f \left(\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u}\right) + \nabla p - \rho_f \mathbf{f}\right]\mathrm{d}\Omega_f = 0.$$

Since D is an arbitrary volume, the integrand must vanish pointwise, yielding the Euler equations

$$\begin{aligned} \rho_f \left(\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u}\right) + \nabla p &= \rho_f \mathbf{f} \quad && \text{in } \Omega_f, \\ \nabla\cdot\mathbf{u} &= 0 && \text{in } \Omega_f. \end{aligned}$$

Compared to the Euler equations, the Navier–Stokes equations account for viscous stresses. For a viscous Newtonian fluid, the Cauchy stress tensor is defined as

$$\boldsymbol{\sigma}_f(\mathbf{u}, p) = -p\mathbf{I} + \mu_f(\nabla\mathbf{u} + \nabla\mathbf{u}^T).$$

Substituting the stress tensor into the momentum balance equation gives the Navier–Stokes equations with Cauchy stress tensor $\boldsymbol{\sigma}_f$

$$\begin{aligned} \rho_f\Big(\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u}\Big) - \nabla\cdot\boldsymbol{\sigma}_f(\mathbf{u}, p) &= \rho_f \mathbf{f} \quad \text{in } \Omega_f, \\ \mathrm{div}(\mathbf{u}) &= 0 \quad \text{in } \Omega_f. \end{aligned}$$

For constant viscosity μ_f and incompressible flow, the viscous term reduces to

$$\begin{aligned} -\nabla\cdot(\mu_f(\nabla\mathbf{u} + \nabla\mathbf{u}^T)) &= -\mu_f(\nabla\cdot\nabla\mathbf{u} + \nabla\cdot\nabla\mathbf{u}^T) \\ &= -\mu_f(\Delta\mathbf{u} + \nabla(\nabla\cdot\mathbf{u})) \\ &= -\mu_f\Delta\mathbf{u}. \end{aligned}$$

The derivations in the following sections are taken from [82, Section 2]. Modifications to the notation have been made to improve consistency with the remainder of this dissertation.

2.1. Governing Equations

We denote the fluid domain by $\Omega_f \subset \mathbb{R}^3$. The transient, incompressible Navier–Stokes equations to model a Newtonian fluid are given by

$$\begin{aligned} \rho_f\Big(\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u}\Big) - \mu_f\Delta\mathbf{u} + \nabla p = \rho_f\mathbf{f} \quad &\text{in } \Omega_f, \\ \operatorname{div}(\mathbf{u}) = 0 \quad &\text{in } \Omega_f. \end{aligned} \tag{2.1}$$

with the fluid velocity $\mathbf{u}$, pressure p, density ρ_f, dynamic viscosity μ_f, and volume force $\mathbf{f}$. In the following, we will refer to the kinematic viscosity ν_f, when only viscosity is mentioned. The dynamic viscosity μ_f can be substituted by the kinematic viscosity ν_f by $\mu_f = \nu_f\,\rho_f$. If the set of equations (2.1) reaches a steady state for $t \to \infty$, we have $\partial_t\mathbf{u} = 0$ and obtain the stationary Navier–Stokes equations

$$\begin{aligned} \rho_f(\mathbf{u}\cdot\nabla)\mathbf{u} - \mu_f\Delta\mathbf{u} + \nabla p = \rho_f\mathbf{f} \quad &\text{in } \Omega_f, \\ \operatorname{div}(\mathbf{u}) = 0 \quad &\text{in } \Omega_f. \end{aligned} \tag{2.2}$$

The boundary of Ω_f is denoted by $\partial\Omega_f$. We assume that there are subsets $\partial\Omega_f^{\text{in}}$, $\partial\Omega_f^{\text{out}} \subset \partial\Omega_f$, which denote the inflow and outflow parts of the boundary. We assume that $\partial\Omega_f^{\text{in}}$ and $\partial\Omega_f^{\text{out}}$ have a positive surface measure. In this work, on $\partial\Omega_f^{\text{in}}$ a certain velocity or flow field is prescribed with Dirichlet boundary conditions

$$\mathbf{u} = \mathbf{g} \quad \text{on } \partial\Omega_f^{\text{in}}.$$

Furthermore, on $\partial\Omega_f^{\text{out}}$ a Neumann boundary condition

$$\nu_f\nabla\mathbf{u}\cdot\mathbf{n} - p\mathbf{n} = 0 \quad \text{on } \partial\Omega_f^{\text{out}}$$

is prescribed. Here, for example, one could also implicitly prescribe a pressure value; see Section 9.6 for FSI boundary conditions. For the fluid-only simulation, a zero-traction ("do-nothing") boundary condition is applied.

The other boundaries of the fluid domain are walls $\partial\Omega_f^{\text{wall}} \subset \partial\Omega_f$, which prevent fluid from leaving the computational domain. For purely fluid simulations, we use a no-slip condition, such that

$$\mathbf{u} = 0 \quad \text{on } \partial\Omega_f^{\text{wall}}. \tag{2.3}$$

We have

$$\partial\Omega_f = \partial\Omega_f^{\text{in}} \cup \partial\Omega_f^{\text{out}} \cup \partial\Omega_f^{\text{wall}},$$

and

$$\partial\Omega_f^{\text{in}} \cap \partial\Omega_f^{\text{wall}} = \emptyset, \quad \partial\Omega_f^{\text{out}} \cap \partial\Omega_f^{\text{wall}} = \emptyset, \quad \partial\Omega_f^{\text{in}} \cap \partial\Omega_f^{\text{out}} = \emptyset.$$

2.2. Finite Element Discretization

For the spatial discretization of the incompressible fluid flow problem, we use different pairs of mixed finite elements. We partition the domain with characteristic mesh size h into tetrahedra or

hexahedra. Let $\tau_h := \tau_h(\Omega_f)$ denote the triangulation and $\mathcal{P}_k(T)$ and $\mathcal{Q}_k(T)$ polynomial spaces of linear and bi- or trilinear functions, respectively. For the partition into tetrahedral elements, we use continuous and either piecewise linear or quadratic functions for the velocity space

$$V_{Pk}^h(\Omega_f) = \left\{ \mathbf{v}_h \in \left(C(\Omega_f)\right)^3 \cap \left(H^1(\Omega_f)\right)^3 : \mathbf{v}_h|_T \in \left(\mathcal{P}_k(T)\right)^3 \ \forall T \in \tau_h \right\}, \ k \in \{1, 2\}.$$

Similarly, we can define the velocity space for hexahedral elements,

$$V_{Qk}^h(\Omega_f) = \left\{ \mathbf{v}_h \in \left(C(\Omega_f)\right)^3 \cap \left(H^1(\Omega_f)\right)^3 : \mathbf{v}_h|_T \in \left(\mathcal{Q}_k(T)\right)^3 \ \forall T \in \tau_h \right\}, \ k \in \{1, 2\}.$$

The continuous and piecewise linear pressure spaces for the different partitions are defined as

$$Q_{P1}^h(\Omega_f) = \left\{ q_h \in C(\Omega_f) \cap L^2(\Omega_f) : q_h|_T \in \mathcal{P}_1(T) \ \forall T \in \tau_h \right\},$$
$$Q_{Q1}^h(\Omega_f) = \left\{ q_h \in C(\Omega_f) \cap L^2(\Omega_f) : q_h|_T \in \mathcal{Q}_1(T) \ \forall T \in \tau_h \right\}.$$

Additionally, we consider the combination of Q2–P1-disc., where the pressure space consists of discontinuous piecewise linear functions

$$Q_{P1-disc.}^h(\Omega_f) = \left\{ q_h \in L^2(\Omega_f) : q_h|_T \in \mathcal{P}_1(T) \ \forall T \in \tau_h \right\}.$$

We are focusing on three-dimensional problems for which the P2–P1, Q2–Q1, and Q2–P1-disc. finite element pairs satisfy the inf–sup condition; cf. [25]. In the case of the not inf-sup stable finite element pairs P1–P1 and Q1–Q1, the Bochev–Dohrmann stabilization [39] is used. The pressure stabilization is inserted into the conservation of mass equation and penalizes nonphysical pressure oscillations

$$C = \frac{1}{\nu} \int_{\Omega_f} (p - \rho_0 p)(q - \rho_0 q) \, \mathrm{d}x = 0 \quad \forall q \in L^2(\Omega_f), \tag{2.4}$$

where ρ_0 is an L^2 projection onto the space of discontinuous constant functions $\mathcal{P}_0(\tau_h)$. Thus, the projection can be computed locally on each element $T \in \tau_h(\Omega_f)$ as

$$\rho_0 q_{|_T} = \frac{1}{\int_T \mathrm{d}x} \int_T q \, \mathrm{d}\mathbf{x}. \tag{2.5}$$

If the type of element is unspecified, we write $\mathbf{V}^h$ and Q^h for the velocity and pressure space, respectively. Let $\mathbf{u}_h \in \mathbf{V}^h$ and $p_h \in Q^h$ be the approximations of the solutions $\mathbf{u}$ and p. Then the resulting variational formulation is as follows: Find $\mathbf{u}_h \in \mathbf{V}^h$ and $p_h \in Q^h$ such that

$$\begin{aligned} \nu_f \int_{\Omega_f} \nabla \mathbf{u}_h : \nabla \mathbf{v}_h \, \mathrm{d}\mathbf{x} + \int_{\Omega_f} ((\mathbf{u}_h \cdot \nabla)\mathbf{u}_h) \cdot \mathbf{v}_h \, \mathrm{d}\mathbf{x} - \int_{\Omega_f} p_h (\nabla \cdot \mathbf{v}_h) \, \mathrm{d}\mathbf{x} &= \int_{\Omega_f} \mathbf{f} \cdot \mathbf{v}_h \, \mathrm{d}\mathbf{x} \quad \forall \mathbf{v}_h \in \mathbf{V}^h, \\ \int_{\Omega_f} q_h (\nabla \cdot \mathbf{u}_h) \, \mathrm{d}\mathbf{x} &= 0 \qquad\qquad\quad \forall q_h \in Q^h. \end{aligned} \tag{2.6}$$

We denote by $\boldsymbol{\varphi}_1, \ldots, \boldsymbol{\varphi}_{N_u}$ the finite element basis of $\mathbf{V}^h$ and by $\psi_1, \ldots, \psi_{N_p}$ the basis of Q^h. Then, we can write

$$\mathbf{u}_h(x) = \sum_{i=1}^{N_u} u_i \boldsymbol{\varphi}_i(x) \quad \text{and} \quad p_h(x) = \sum_{i=1}^{N_p} p_i \psi_i(x).$$

We define the following matrix operators corresponding to the weak formulation,

$$\begin{aligned} A &= [a]_{ij}, & a_{ij} &= \int_{\Omega_f} \nabla\boldsymbol{\varphi}_i : \nabla\boldsymbol{\varphi}_j \,\mathrm{d}\mathbf{x}, \\ N(\mathbf{u}_h) &= [n(\mathbf{u}_h)]_{ij}, & n_{ij} &= \int_{\Omega_f} ((\mathbf{u}_h \cdot \nabla)\boldsymbol{\varphi}_j) \cdot \boldsymbol{\varphi}_i \,\mathrm{d}\mathbf{x}, \\ B &= [b]_{ij}, & b_{ij} &= -\int_{\Omega_f} \psi_i (\nabla \cdot \boldsymbol{\varphi}_j) \,\mathrm{d}\mathbf{x}, \end{aligned}$$

and obtain the system of linear equations for the stationary Navier–Stokes equations,

$$\begin{bmatrix} F_O(\mathbf{u}_h) & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u}_h \\ p_h \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ 0 \end{bmatrix}, \tag{2.7}$$

with $F_O(\mathbf{u}_h) = \nu A + N(\mathbf{u}_h)$. The resulting Picard iteration with the iterates $\mathbf{u}_h^k$ and p_h^k reads [55, p. 346]

$$\begin{bmatrix} F_O(\mathbf{u}_h^k) & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u}_h^{k+1} \\ p_h^{k+1} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ 0 \end{bmatrix}, \tag{2.8}$$

defining the discretized Oseen problem with $F_O(\mathbf{u}_h^k) = \nu_f A + N(\mathbf{u}_h^k)$ and $\delta\mathbf{u}_h^k = \mathbf{u}_h^{k+1} - \mathbf{u}_h^k$, $\delta p_h^k = p_h^{k+1} - p_h^k$. We apply the Newton method to solve the nonlinear problem,

$$\begin{aligned} &\text{Solve:} && \mathcal{J}(X^k)\delta X^k = -\mathcal{R}(X^k), \\ &\text{Update:} && X^{k+1} = X^k + \delta X^k, \end{aligned} \tag{2.9}$$

where the δ notation is defined as $\delta X^k := X^{k+1} - X^k$. The Newton residual $\mathcal{R}$ for the stationary case is defined as

$$\mathcal{R}_{\text{stationary}} \begin{pmatrix} \mathbf{u}_h^k \\ p_h^k \end{pmatrix} = \begin{bmatrix} F_O(\mathbf{u}_h^k) & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u}_h^k \\ p_h^k \end{bmatrix} - \begin{bmatrix} \mathbf{f} \\ 0 \end{bmatrix} =: \begin{bmatrix} r_{\mathbf{u}_h}^k \\ r_{p_h}^k \end{bmatrix}. \tag{2.10}$$

We need to linearize (2.7) by adding an additional term $W(\mathbf{u}_h^k)$ to $F_O(\mathbf{u}_h^k)$ from the derivative of the convection term

$$W(\mathbf{u}_h^k) = [w(\mathbf{u}_h^k)]_{ij}, \qquad w_{i,j} = \int_{\Omega_f} ((\boldsymbol{\varphi}_j \cdot \nabla)\mathbf{u}_h^k) \cdot \boldsymbol{\varphi}_i \,\mathrm{d}\mathbf{x},$$

with the velocity solution from the previous kth Newton step $\mathbf{u}_h^k$. This leads to

$$\mathcal{J} \begin{pmatrix} \mathbf{u}_h^k \\ p_h^k \end{pmatrix} \begin{bmatrix} \delta\mathbf{u}_h^k \\ \delta p_h^k \end{bmatrix} = \begin{bmatrix} F_N(\mathbf{u}_h^k) & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} \delta\mathbf{u}_h^k \\ \delta p_h^k \end{bmatrix} = -\mathcal{R}_{\text{stationary}} \begin{pmatrix} \mathbf{u}_h^k \\ p_h^k \end{pmatrix}, \tag{2.11}$$

with $F_N(\mathbf{u}_h^k) = F_O(\mathbf{u}_h^k) + W(\mathbf{u}_h^k)$.

2.3. Time Discretization

For the temporal discretization of the fluid flow problem, the backward differentiation formula [74] is considered. Consider the simple ordinary differential equation $y'(t) = f(t, y(t))$ on a uniform grid $t_n = t_0 + n\Delta t$ with initial value $y(t_0) = y_0$ and uniform step size Δt. The general s-step backward differentiation formula (BDF) is given by

$$\sum_{k=0}^{s} \alpha_k y_{n+k} = \Delta t \beta f(t_{n+s}, y_{n+s}).$$

In this work, the second-order BDF (BDF-2) is employed, which corresponds to the case of $s = 2$. The corresponding BDF-2 scheme is defined as

$$y_{n+2} - \frac{4}{3} y_{n+1} + \frac{1}{3} y_n = \frac{2}{3} \Delta t f_{n+2}. \tag{2.12}$$

We transform the coefficients in (2.12) and shift the index to obtain

$$\frac{3}{2\Delta t} y_{n+1} - \frac{2}{\Delta t} y_n + \frac{1}{2\Delta t} y_{n-1} = f_{n+1}. \tag{2.13}$$

The BDF2 method is an implicit, second-order accurate, and A-stable time integration scheme [74]. For $s = 1$ it is equivalent to the implicit Euler method.

In case of non-uniform time step sizes Δt, the coefficients can be expressed depending on the current time step sizes; see [73, p. 401, eq. 5.14]. Assume $\omega_n = \Delta t_n / \Delta t_{n-1}$ with $\Delta t_n = t_{n+1} - t_n$. Then, the BDF-2 scheme for variable time step size is defined by

$$y_{n+1} - \frac{(1+\omega_n)^2}{1+2\omega_n} y_n + \frac{\omega_n^2}{1+2\omega_n} y_{n-1} = \Delta t_n \frac{1+\omega_n}{1+2\omega_n} f_{n+1} \tag{2.14}$$

Variable time step sizes are employed for the FSCI simulations presented in Section 10.7. After the ramping phase, the time step size is increased to account for the long active material response period.

The BDF-2 scheme is used to discretize the incompressible Navier–Stokes equations (2.1) also in time. For the transient problem with constant time step size Δt the system (2.11) is expanded to include the time discretization:

$$\mathcal{J} \begin{pmatrix} \mathbf{u}_h^k \\ p_h^k \end{pmatrix} \begin{bmatrix} \delta \mathbf{u}_h^{k,n+1} \\ \delta p_h^{k,n+1} \end{bmatrix} = \begin{bmatrix} \frac{3}{2\Delta t_n} M + F_N(\mathbf{u}_h^{k,n+1}) & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} \delta \mathbf{u}_h^{k,n+1} \\ \delta p_h^{k,n+1} \end{bmatrix} = -\mathcal{R}_{\text{transient}} \begin{pmatrix} \mathbf{u}_h^{k,n+1} \\ p_h^{k,n+1} \end{pmatrix}, \tag{2.15}$$

where

$$-\mathcal{R}_{\text{transient}} \begin{pmatrix} \mathbf{u}_h^{k,n+1} \\ p_h^{k,n+1} \end{pmatrix} = -\mathcal{R}_{\text{stationary}} \begin{pmatrix} \mathbf{u}_h^{k,n+1} \\ p_h^{k,n+1} \end{pmatrix} + \begin{bmatrix} \frac{3}{2\Delta t_n} M \mathbf{u}_h^{k,n+1} + \frac{2}{\Delta t} M \mathbf{u}_h^n - \frac{1}{2\Delta t} M \mathbf{u}_h^{n-1} \\ 0 \end{bmatrix},$$

and $\mathbf{u}_h^n$ and $\mathbf{u}_h^{n-1}$ are the solutions to the last two time steps. The velocity mass matrix is defined as

$$M = [m]_{ij}, \qquad m_{ij} = \int_{\Omega_f} \boldsymbol{\varphi}_i \cdot \boldsymbol{\varphi}_j \, \mathrm{d}\mathbf{x}.$$

In case of variable time step sizes, the coefficients are adjusted according to (2.14).

We define the generic form of the saddle-point problem as

$$\mathcal{F} := \begin{bmatrix} F & B^T \\ B & -C \end{bmatrix}, \tag{2.16}$$

which is used to define the block and monolithic preconditioners from Chapter 6 and Sections 7.2 and 7.3, where the fluid component F depends on the underlying problem setting, e.g., stationary or transient, and $C \neq 0$ in case of a discretization that is not inf-sup stable.

2.4. CFL Number

A Courant(-Friedrichs-Lewy)-type number (CFL) is defined in order to indicate how much information travels through a computational grid cell in a unit of time. It relates the velocity of transport, time step, and mesh resolution and is defined as

$$\mathrm{c}_{\mathrm{CFL}}(t) = \frac{u_\infty^h(t)\Delta t}{h}, \qquad u_\infty^h(t) = \max_{x^h \in \tau_h(\Omega_f)} \|\mathbf{u}_h(x^h, t)\|_2, \tag{2.17}$$

for structured meshes, where x^h is a finite element node. In unstructured meshes, the element size and shape can vary. This is taken into account by using

$$\mathrm{c}_{\mathrm{CFL}}(t) = \max_{T \in \tau_h(\Omega_f)} \mathrm{c}_{\mathrm{CFL},T}(t), \qquad \mathrm{c}_{\mathrm{CFL},T}(t) = \frac{u_\infty^h(T,t)\Delta t}{d_I(T)}, \qquad u_\infty^h(T,t) = \max_{x^h \in T} \|\mathbf{u}_h(x^h, t)\|_2,$$

where $T \in \tau_h(\Omega_f)$ is a finite element, and $d_I(T)$ is the diameter of the largest incircle of T.

Furthermore, we will use the average of $\mathrm{c}_{\mathrm{CFL},T}(t)$ over all elements $T \in \tau_h(\Omega_f)$, denoted by $\mathrm{c}_{\mathrm{CFL,avg}}(t)$. By $\mathrm{c}_{\mathrm{CFL}}$ and $\mathrm{c}_{\mathrm{CFL,avg}}$, we refer to the corresponding maximum CFL number over time. In case the CFL number is too large, instability or unphysical behavior may be encountered. Generally, implicit time stepping schemes are more stable with respect to the CFL number. High CFL numbers tend to be more demanding for preconditioners; see, for example, [34, Figure 9] and Figures 10.25 and 10.27.

3. Solid Mechanics

In this chapter, the arterial wall models considered in this work are introduced. To this end, the fundamental concepts of nonlinear solid mechanics are first summarized before the constitutive models are presented. As discussed in the previous chapter, continuum mechanics distinguishes between the Eulerian and the Lagrangian point of view. While the Eulerian description is applied in fluid mechanics, where the motion of the fluid is observed at fixed locations in space, the Lagrangian description follows the motion of material particles and is therefore employed for solid mechanics. Consequently, the governing equations of the arterial wall are formulated in the Lagrangian framework. In fluid–structure interaction (FSI), the fluid and solid domains are described in different reference frames. To couple both descriptions, an Arbitrary Lagrangian–Eulerian (ALE) formulation is introduced, which is discussed in Chapter 5. The following derivations are based on [97, 64].

When describing the deformation of a solid, we need to consider the reference configuration and the current configuration. The reference configuration is the undeformed state of the solid, while the current configuration is the deformed state of the solid at a given time t. We state the following definitions for the solid domain in the reference configuration Ω_s^0 and current configuration Ω_s^t.

The function $\mathcal{A}$ describes the position of each material point $\mathbf{X}$ in the reference configuration Ω_s^0 at time t in the current configuration Ω_s^t and is called the motion

$$\mathcal{A} : \Omega_s^0 \times [0, T] \to \Omega_s^t, \quad (\mathbf{X}, t) \mapsto \mathbf{x} = \mathcal{A}(\mathbf{X}, t). \tag{3.1}$$

Thus, each point $\mathbf{X}$ in the reference configuration Ω_s^0 can be described by $\mathbf{x} = \mathcal{A}(\mathbf{X}, t)$ in the current configuration $\Omega_s^t = \mathcal{A}(\Omega_s^0)$ at time t; see Figure 3.1. The displacement of the material point $\mathbf{X}$ at a given time t is defined as

$$\mathbf{d}_s(\mathbf{X}, t) = \mathbf{x} - \mathbf{X} = \mathcal{A}(\mathbf{X}, t) - \mathbf{X}.$$

The deformation gradient $\mathbf{F}$ is defined as the linear mapping between the reference and current configuration and describes how the material deforms during motion. It is defined as the Jacobian of the motion $\mathcal{A}$ with

$$\mathbf{F} = \nabla_0 \mathcal{A}(\mathbf{X}, t) = \nabla_0 \mathbf{x} = \nabla_0 \mathbf{d}_s + \mathbf{I}.$$

The determinant $J = \det(\mathbf{F})$ of the deformation gradient is a measure of the local volume change during deformation. The velocity and the acceleration of the material point $\mathbf{X}$ at time t are defined as the first and second time derivatives of the motion $\mathcal{A}$, respectively:

$$\dot{\mathbf{d}}_s(\mathbf{X}, t) = \frac{\partial \mathcal{A}(\mathbf{X}, t)}{\partial t}, \qquad \ddot{\mathbf{d}}_s(\mathbf{X}, t) = \frac{\partial^2 \mathcal{A}(\mathbf{X}, t)}{\partial t^2}.$$

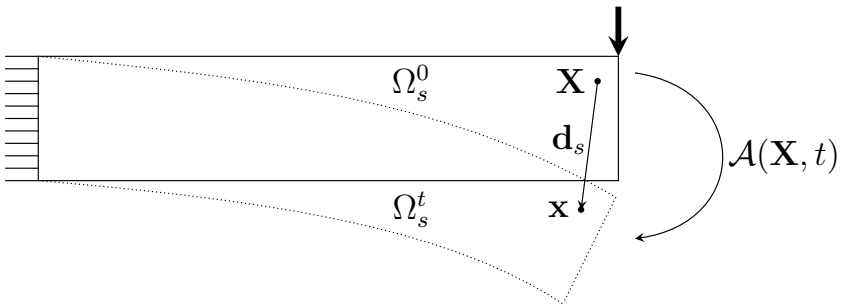


Figure 3.1: Illustration of the displacement $\mathbf{d}_s$ of a beam with motion $\mathcal{A}(\mathbf{X}, t)$ in the Lagrangian viewpoint. The figure is based on [64, Figure 3.5].

Stress is defined as the force per unit area that acts on a material. It characterizes the internal mechanical state of a body and describes its response to external loading. The Cauchy stress tensor $\boldsymbol{\sigma}$ is a second-order tensor describing the internal stresses at a point in the current, deformed configuration Ω_s^t.

In contrast, the second Piola-Kirchhoff stress tensor $\mathbf{S}$ is defined in the reference configuration Ω_s^0 and is related to the Cauchy stress tensor $\boldsymbol{\sigma}$ by the deformation gradient $\mathbf{F}$ as

$$\mathbf{S} = J\ \mathbf{F}^{-1} \cdot \boldsymbol{\sigma} \cdot \mathbf{F}^{-T}.$$

The traction vector $\mathbf{t}$ is defined as the force per unit area acting on a surface with unit outward normal vector $\mathbf{n}$ with $[t] = N/m^2$. It is related to the Cauchy stress tensor $\boldsymbol{\sigma} : \Omega_s^t \to \mathbb{R}^{3\times 3}$ by the equation

$$\mathbf{t} = \boldsymbol{\sigma} \cdot \mathbf{n} \quad \text{on } \partial\Omega_s^t$$

The motion of a solid is governed by the balance of linear momentum. This follows from Newton's second law, which relates the acceleration of the solid to the internal stresses and the external body forces acting on it,

$$\rho_s^t \ddot{\mathbf{d}}_s = \nabla \cdot \boldsymbol{\sigma} + \mathbf{f}_s^t, \qquad \text{in } \Omega_s^t,$$

where ρ_s^t is the current density of the solid, $\ddot{\mathbf{d}}_s$ is the acceleration, and $\mathbf{f}_s^t$ is the body force per unit current volume acting on the solid. In the reference configuration, the balance of linear momentum is expressed as

$$\rho_s \ddot{\mathbf{d}}_s = \nabla_0 \cdot \mathbf{P}\left(\mathbf{d}_s\right) + \mathbf{f}_s \qquad \text{in } \Omega_s^0, \tag{3.2}$$

where $\mathbf{P}$ denotes the first Piola–Kirchhoff stress tensor, ρ_s is the reference density of the solid, and $\mathbf{f}_s$ is the body force per unit reference volume acting on the solid. This formulation serves as the starting point for the finite element discretization.

In the following sections, different material models are considered. First, a neo-Hookean material model is presented, which is an isotropic hyperelastic material; see Section 3.1.1. Then, a chemo-mechanical smooth muscle model is presented in Section 3.1.2.

The strain energy density function Ψ describes how much elastic energy is stored per unit volume of a material as it deforms. For the different constitutive models, different strain energy

density functions are defined. They relate to the balance of linear momentum in the following way

$$\mathbf{P} = \frac{\partial \Psi}{\partial \mathbf{F}},$$

for the first Piola-Kirchhoff stress tensor, or

$$\boldsymbol{\sigma} = \frac{1}{J}\frac{\partial \Psi}{\partial \mathbf{F}}\mathbf{F}^T,$$

for the Cauchy stress tensor. Thus, the strain energy density function does not appear as an additional force term, but determines the stress response of the material, whose divergence represents the internal force contribution in the momentum balance equation.

3.1. Arterial Wall Models

In this section, different arterial wall models are presented. The passive neo-Hookean and Ψ_A material models are introduced in Sections 3.1.1 and 3.1.3, respectively. In Section 3.1.2, the chemo-mechanical smooth muscle cell (Ψ_{SMC}) model is defined, for which the Ψ_A model is augmented by an active smooth muscle cell contraction model, resulting in a chemo-mechanically coupled arterial wall model.

3.1.1. Neo-Hookean Model

We consider neo-Hookean hyperelastic materials, which are characterized by the strain energy density function Ψ, from which the stress-strain response is derived. These materials have a nonlinear response to deformation. Consider right Cauchy-Green deformation tensors $\mathbf{C} = \mathbf{F}^T\mathbf{F}$. We define the three invariants of $\mathbf{C}$ as follows:

$$\begin{aligned} I_1 &= \text{tr}(\mathbf{C}), \\ I_2 &= \frac{1}{2}\left((\text{tr}\mathbf{C})^2 - \text{tr}(\mathbf{C}^2)\right), \\ I_3 &= \det(\mathbf{C}) = J^2. \end{aligned} \tag{3.3}$$

The strain energy density function Ψ can be expressed in terms of the invariants of these tensors, which are scalar quantities that remain unchanged under coordinate transformations. For a standard deformation problem, the following strain energy density function for a compressible neo-Hookean material is used

$$\Psi_{\text{NH}} = \frac{\mu}{2}(I_1 - 3) - \mu \ln(J) + \frac{1}{2}\left(\kappa - \frac{2}{3}\mu\right)(\ln(J))^2,$$

with bulk modulus κ and shear modulus μ. The considered neo-Hookean material model is isotropic, which means that the material properties are the same in all directions.

3.1.2. Chemo-Mechanical SMC Model

Note that the chemo-mechanical smooth muscle cell wall model described in this section is summarized here, but the precise derivation, further development, parameter fitting and implementation in `AceGen` will be part of the doctoral thesis of Sharan Nurani Ramesh; see also [149].

The chemo-mechanical smooth muscle cell model in this section consists of a passive and active material response. The passive response is described by the hyperelastic material model of Balzani et al. [12] (see also Section 3.1.3), whereas the active response is modeled using the chemo-mechanical model proposed by Uhlmann and Balzani [168], which extends the models of Hai and Murphy [72] and Murtada et al. [132] by incorporating stretch-dependent calcium-dependent and calcium-independent contraction mechanisms. Since the focus lies on the stretch-dependent chemo-mechanical response of the arterial wall, other activation mechanisms, such as those induced by the sympathetic or endocrine systems, are neglected. The description of the model is based on [9] and [149]. We adopt an invariant-based framework for the constitutive model with two discrete fiber directions $\mathbf{a}^{(f)}$, where the strain energy density Ψ_{SMC} is additively decomposed as

$$\Psi_{\text{SMC}} = \Psi_{\text{p,isot}} + \sum_{f=1}^{2} \Psi_{\text{p,ti}}^{(f)} + \sum_{f=1}^{2} \Psi_{\text{a}}^{(f)}, \tag{3.4}$$

since it is assumed that there is a weak interaction between the fiber families. $\Psi_{\text{p,isot}}$ represents the influence of the isotropic elastin matrix, $\Psi_{\text{p,ti}}^{(f)}$ represents the transversely isotropic part and models the influence of the passive collagen fibers, whereas $\Psi_{\text{a}}^{(f)}$ represents the response of the active smooth muscle cells. In addition to the invariants in (3.3), the mixed invariants of the right Cauchy–Green tensor $\mathbf{C}$ are defined

$$I_4^{(f)} = \mathbf{C} : \mathbf{M}^{(f)}, \quad I_5^{(f)} = \mathbf{C}^2 : \mathbf{M}^{(f)}, \tag{3.5}$$

where $\mathbf{M}^{(f)} = \mathbf{a}^{(f)} \otimes \mathbf{a}^{(f)}$ are the structural tensors. Using the formulation by Balzani *et al.* [12] we have the passive components

$$\begin{aligned} \Psi_{\text{p,iso}} &= \alpha_1 \left(I_1 I_3^{-1/3} - 3 \right) + \alpha_2 \left(I_3^{\alpha_3} + I_3^{-\alpha_3} - 2 \right), \\ \Psi_{\text{p,ti}}^{(f)} &= \alpha_4 \langle K_3^{(f)} - 2 \rangle^{\alpha_5}, \end{aligned} \tag{3.6}$$

where α_1, α_2, α_3, α_4 and α_5 are material parameters and $K_3^{(f)} = I_1 I_4^{(f)} - I_5^{(f)}$. If we neglect the active contribution $\Psi_{\text{a}}^{(f)}$, we can derive the Ψ_A model from [10]; see Section 3.1.3.

We give some intuition on the meaning of the parameters α_i, where $i = 1, \ldots, 5$. The parameter α_1 controls the stiffness of the isotropic elastin matrix response, the parameters α_2 and α_3 control the volumetric stiffness and compressibility. A larger α_2 means stronger resistance against volumetric changes, i.e., stronger near-incompressibility. The parameter α_4 controls the stiffness contribution of the fiber-related transversely anisotropic part. A larger α_4 means stronger resistance associated with the fiber families. Finally, α_5 controls the nonlinear steepness of the fiber response.

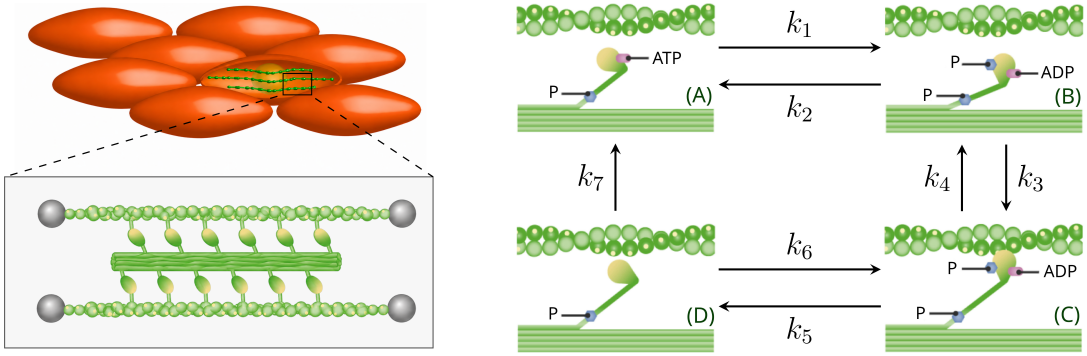


Figure 3.2: Left: Spindle-shaped smooth muscle cell. The smooth muscle cell consists of passive and active components. The active component is formed by the contractile unit: two thin (actin) filaments and one thick (myosin) filament connected by cross-bridges, as illustrated in the zoom window at the bottom. The surrounding matrix represents the passive component. The image is based on [133, Figure 3]. *The illustration was created by the author with the assistance of OpenAI's ChatGPT image generation and subsequently edited.* **Right**: Cross-bridge cycle based on [126] and [133, Figure 2]. MLCK activates myosin by phosphorylation. Once activated, the myosin heads repeatedly attach to actin, pull on the actin filament using energy from ATP, and detach again. This cyclic interaction causes actin and myosin filaments to slide past each other, which brings the dense bodies closer together and results in smooth muscle contraction.

Stretch-Dependent Chemical Kinetics Model for Smooth Muscle Cells

The smooth muscle cell (SMC) cytoskeleton is composed of a three-dimensional network of actin and myosin filaments connected at cytoplasmic dense bodies; see Figure 3.2. SMC contraction results from the relative sliding of myosin and actin filaments. This contraction is regulated by biochemical processes inside the cell, with the intracellular free calcium concentration, ($[Ca^{2+}]_i$), acting as the main activator. The myosin filaments contain distinct head regions that interact with actin to form attached cross-bridges. The formation between a myosin and an actin filament during muscle contraction is called a cross-bridge. Two opposing biochemical processes control smooth muscle activation and relaxation. First, an increase in intracellular calcium concentration leads to the formation of a calcium-calmodulin complex, which activates the enzyme myosin light chain kinase (MLCK). Activated MLCK phosphorylates the myosin regulatory light chain, thereby enabling actin-myosin cross-bridge cycling and active contraction. The opposite process is mediated by myosin light chain phosphatase (MLCP), which dephosphorylates myosin and promotes relaxation. These processes are described by the four-state cross-bridge model proposed by Hai and Murphy [72], where cross-bridges exist in four distinct states:

(A) Free dephosphorylated cross-bridge (inactive),

(B) Free phosphorylated cross-bridge (ready to attach),

(C) Attached phosphorylated cross-bridge (actively generating force),

(D) Attached dephosphorylated cross-bridge (latch state),

which can also be seen in Figure 3.2 (right). The kinetic model is expressed by the following system of ordinary differential equations

$$\begin{bmatrix} \dot{n}_{\mathrm{A}} \\ \dot{n}_{\mathrm{B}} \\ \dot{n}_{\mathrm{C}} \\ \dot{n}_{\mathrm{D}} \end{bmatrix} = \begin{bmatrix} -k_1 & k_2 & 0 & k_7 \\ k_1 & -k_2 - k_3 & k_4 & 0 \\ 0 & k_3 & -k_4 - k_5 & k_6 \\ 0 & 0 & k_5 & -k_6 - k_7 \end{bmatrix} \begin{bmatrix} n_{\mathrm{A}} \\ n_{\mathrm{B}} \\ n_{\mathrm{C}} \\ n_{\mathrm{D}} \end{bmatrix}, \tag{3.7}$$

where n_{A}, n_{B}, n_{C} and n_{D} are the fractions of cross-bridges in the four aforementioned states. Their change is described by seven rate constants $k_1 - k_7$. For example, the rates k_1 and k_6 define the rate of phosphorylation of free cross-bridges by MLCK. They represent the transition from A to B, and B to A, respectively. The rates of cross-bridge attachment k_3 and detachment k_4 and k_7 are assumed to be constant. We have two stretch-dependent contraction mechanisms: a stretch-dependent increase in calcium concentration and a stretch-dependent reduction in MLCP activity. This means that a larger stretch λ causes higher intracellular calcium concentration, which causes larger phosphorylation rates k_1 and k_6, which leads to more attached cross-bridges and in turn stronger active contraction. At the same time, stretch can reduce the dephosphorylation rates k_2 and k_5 through calcium sensitization by the following mechanism. The stretch λ causes a reduced MLCP activity, which leads to smaller dephosphorylation rates k_2 and k_5, with cross-bridges remaining attached for longer, which in turn leads to sustained contraction in the latch state. The rates k_1 and k_6 are described as functions depending on the intracellular calcium concentration

$$k_{1/6} = \eta \frac{\left[\mathrm{Ca}^{2+}\right]_{\mathrm{c}}^2}{\left[\mathrm{Ca}^{2+}\right]_{\mathrm{c}}^2 + (\mathrm{Ca}_{50})^2}. \tag{3.8}$$

This was suggested first by Murtada *et al.* [132]. For low calcium concentration, the phosphorylation rates are small. With increasing calcium concentration, k_1 and k_6 increase, but the material parameter η prescribes an upper bound, i.e., the maximum rate of phosphorylation. The parameter Ca_{50} describes the calcium level at which half of the maximum activation is reached. To model the first contraction mechanism, as proposed by Uhlmann and Balzani [168], the intracellular calcium concentration is assumed to depend on the stretch of the smooth muscle cell and therefore is modeled as a stretch-dependent function

$$\left[\mathrm{Ca}^{2+}\right]_{\mathrm{c}} = \gamma_1 \langle \lambda - \bar{\lambda}_c \rangle^2. \tag{3.9}$$

The parameter $\lambda = (I_4)^{1/2}$ is the stretch along the SMC longitudinal direction, and γ_1 a material parameter. The Macaulay bracket $\langle \cdot \rangle$ ensures that the calcium increases only if the current stretch λ exceeds the adaptive reference stretch $\bar{\lambda}_c$.

In [168] the authors modeled the reduction in intracellular calcium concentration due to various factors like calcium sequestration and calcium outflow by an evolution equation for $\bar{\lambda}_c$; see [9, Equation 8]. This variable evolves over time and models the reduction of calcium. The target calcium concentration $[\mathrm{Ca}^{2+}]_{\mathrm{tar}}$ is also stretch-dependent

$$[\mathrm{Ca}^{2+}]_{\mathrm{tar}} = \gamma_3 \frac{\lambda^2}{\lambda^2 + \lambda_{50,\mathrm{c}}^2}, \tag{3.10}$$

with γ_3 being a material parameter and $\lambda_{50,\mathrm{c}}$ the half activation stretch.

Besides calcium-dependent activation, smooth muscle cells can also contract through calcium sensitization. This is a calcium-independent contraction mechanism [160], which occurs as a result of the action of the enzymes RhoA and Rho-Kinase (ROK). The main effect is to reduce the activity of MLCP, which is responsible for dephosphorylation. This is modeled by reducing the dephosphorylation rates k_2 and k_5. Uhlmann and Balzani [169] modeled the resulting reduction in the rate of dephosphorylation by evolution equations for $k_{2/5}$; see [9, Equation 10]. The model ensures that these rates do not decrease below prescribed minimum values $\dot{k}_{2/5,\mathrm{min}}$, so that dephosphorylation does not vanish completely. Further, the target stretch-dependent MLCP activity is given as

$$k_{2/5,\mathrm{tar}} = \gamma_6 \left(1 - \frac{\lambda}{\lambda + \lambda_{50,\mathrm{p}}}\right), \tag{3.11}$$

where γ_6 is a material parameter and $\lambda_{50,\mathrm{p}}$ is the half activation stretch. This equation means that the target MLCP activity decreases with increasing stretch. Therefore, when the SMC is stretched, dephosphorylation is reduced, which promotes contraction and force maintenance.

Smooth Muscle Cell Activation Model

The chemical kinetics model describes how many myosin heads are in the different biochemical states. However, for the mechanical problem, this biochemical information has to be translated into an active stress contribution. Now, the biochemical information has to enter the mechanical problem via the $\Psi_a^{(f)}$. Based on the work of Uhlmann and Balzani [169], the active strain energy function is given by

$$\Psi_{\mathrm{a}}^{(f)} = \frac{\mu_{\mathrm{a}}}{2}\left(n_{\mathrm{C}}^{(f)} + n_{\mathrm{D}}^{(f)}\right)(\lambda_{\mathrm{e}}^{(f)} - 1)^2 \ , \ \text{with } \lambda_{\mathrm{e}}^{(f)} = \frac{\lambda^{(f)}}{\lambda_{\mathrm{a}}^{(f)}}. \tag{3.12}$$

The parameter μ_{a} is a stiffness constant; $n_{\mathrm{C}}^{(f)}$ and $n_{\mathrm{D}}^{(f)}$ are the fractions of myosin heads in states C and D, respectively. The stretch $\lambda^{(f)}$ is multiplicatively split into an elastic part $\lambda_e^{(f)}$ and an active part $\lambda_{\mathrm{a}}^{(f)}$. The rate of change of the active strain $\lambda_{\mathrm{a}}^{(f)}$ over time is given by

$$\dot{\lambda}_{\mathrm{a}}^{(f)} = \beta_1 \frac{P_{\mathrm{a}}^{(f)} - P_{\mathrm{c}}^{(f)}}{P_{\mathrm{a}}^{(f)} - \beta_2}, \ \text{with} \ \ P_{\mathrm{a}}^{(f)} = \mu_a \left(n_{\mathrm{C}}^{(f)} + n_{\mathrm{D}}^{(f)}\right)(\lambda_{\mathrm{e}}^{(f)} - 1) \ \text{and} \ P_{\mathrm{c}}^{(f)} = \kappa\, n_{\mathrm{C}}^{(f)} \tag{3.13}$$

where β_1 and β_2 are material parameters, $P_{\mathrm{a}}^{(f)}$ and $P_{\mathrm{c}}^{(f)}$ are the active and driving stresses, respectively, and κ is the maximum driving stress.

The material parameters for the chemical and active chemo-mechanical parts of the model are taken from Uhlmann and Balzani [168]. These parameters were fitted to experimental data from Johnson *et al.* [102], who studied the contraction of middle cerebral rat arteries under increasing intravascular pressure. The material parameters of the active model are listed in Tables A.1 and A.2. The passive material parameters are listed in Section 9.6 and are the only ones being modified throughout this work.

In [149], remodeling of the arterial wall, i.e., growth and reorientation, is described, which is also included in the `FEDDLib-AceGen` interface, but not considered in this work.

3.1.3. Ψ_A Model

The Ψ_A model from [12] and tested in [8, 10] is considered, which is anisotropic and quasi-incompressible. It can be derived from (3.4) by neglecting the active contribution $\Psi_{\mathrm{a}}^{(f)}$. Then, the strain energy density function is defined as

$$\Psi_{\mathrm{A}} = \Psi_{\mathrm{A,iso}} + \sum_{f=1}^{2} \Psi_{\mathrm{A,ti}}^{(f)}.$$

The mixed invariants are considered separately for the two helically arranged fiber families $\mathbf{a}^{(f)}$. As in (3.6), the isotropic part of the strain energy density function is given by

$$\begin{aligned} \Psi_{\mathrm{A,iso}} &= \alpha_1 \left(I_1 I_3^{-1/3} - 3 \right) + \alpha_2 \left(I_3^{\alpha_3} + I_3^{-\alpha_3} - 2 \right), \\ \Psi_{\mathrm{A,ti}}^{(f)} &= \alpha_4 \langle I_1 I_4^{(f)} - I_5^{(f)} - 2 \rangle^{\alpha_5}, \end{aligned} \tag{3.14}$$

where I_4 and I_5 are the invariants related to the anisotropy of the material. In our settings, the two fiber families are defined by

$$a^{(1)} = \left(\sin(\theta) \frac{X_2}{\sqrt{X_1^2 + X_2^2}}, -\sin(\theta) \frac{X_1}{\sqrt{X_1^2 + X_2^2}}, \cos(\theta) \right), \tag{3.15}$$

$$a^{(2)} = \left(-\sin(\theta) \frac{X_2}{\sqrt{X_1^2 + X_2^2}}, \sin(\theta) \frac{X_1}{\sqrt{X_1^2 + X_2^2}}, \cos(\theta) \right), \tag{3.16}$$

with the reference coordinates $\mathbf{X}$. The directions X_1, X_2, X_3 correspond to the x,y, and z-plane. The fiber direction is fitting for a geometry with axial direction in z-direction and circumferential direction in the x-y plane. Hence, both fiber families lie in the circumferential-axial plane and are symmetrically arranged, with two non-orthogonal fiber directions, unless $f_A = 45°$ and with $\theta = f_A$ in radians. The curved artery benchmark Section 9.5 is also considered, where we need to define more specialized fiber directions in order to accommodate the bending of the artery. In that case, the fiber direction is directly tailored to the geometry; see Figure 9.10 (left).

3.2. Finite Element Discretization

In the following, the variational formulation and weak form of the balance of linear momentum are briefly discussed. For more details we refer to [179, Section 3.4]. For the spatial discretization of the deformation problem, described by the balance of linear momentum, we consider P_2 finite elements. The domain with characteristic mesh size h is partitioned into tetrahedra. Let $\tau_h^0 := \tau_h(\Omega_s^0)$ denote the triangulation and $\mathcal{P}_k(T)$ the polynomial spaces of bilinear functions. For the partition into tetrahedral elements, continuous and piecewise quadratic functions are considered

$$\mathbf{V}^h(\Omega_s^0) = \left\{ \mathbf{v}_h \in \left(C(\Omega_s^0)\right)^3 \cap \left(H^1(\Omega_s^0)\right)^3 : \mathbf{v}_h|_T \in \left(\mathcal{P}_2(T)\right)^3 \ \forall T \in \tau_h^0 \right\}.$$

Let $\mathbf{d}_s^h \in \mathbf{V}^h$ be the approximation of the solution $\mathbf{d}_s$. Then the resulting variational formulation is given as follows: Find $\mathbf{d}_s^h \in \mathbf{V}^h$ such that

$$\int_{\Omega_s^0} \rho_s \frac{\partial^2 \mathbf{d}_s^h}{\partial t^2} \cdot \mathbf{v}_h \, \mathrm{d}\mathbf{X} + \int_{\Omega_s^0} \mathbf{P}(\mathbf{d}_s^h) : \nabla_0 \mathbf{v}_h \, \mathrm{d}\mathbf{X} = \int_{\Omega_s^0} \mathbf{f}_s \cdot \mathbf{v}_h \, \mathrm{d}\mathbf{X} + \int_{\Gamma_N^0} \mathbf{T} \cdot \mathbf{v}_h \, \mathrm{d}S, \tag{3.17}$$

for all $\mathbf{v}_h \in \mathbf{V}^h$. Here, on the Neumann boundary $\Gamma_N^0 \subset \partial\Omega_s^0$, $\mathbf{T} = p\mathbf{n}_0$ describes the stresses in the outward normal direction. We apply deformation-dependent loads as described in [179, Chapter 4.2.5], where corresponding pressure values are inserted for p; see also Section 9.6.1. Note that this generates another nonlinear contribution to the tangent matrix, which is included in the structure block. The finite element basis of $\mathbf{V}^h$ is denoted by $\boldsymbol{\varphi}_1, \ldots, \boldsymbol{\varphi}_{N_d}$. Then we can write

$$\mathbf{d}_s^h(\mathbf{X}) = \sum_{i=1}^{N_d} d_i \boldsymbol{\varphi}_i(\mathbf{X}).$$

Here, $\mathbf{X} \in \Omega_s^0$ denotes the reference coordinates. The solid mass matrix is defined by

$$M_s = [m_s]_{ij}, \qquad m_{s,ij} = \int_{\Omega_s^0} \rho_s \boldsymbol{\varphi}_i \cdot \boldsymbol{\varphi}_j \, \mathrm{d}\mathbf{X}.$$

For the stress term, the internal force vector is defined as

$$\mathbf{f}_{\mathrm{int}}(\mathbf{d}_s^h) = [f_{\mathrm{int}}]_i, \qquad f_{\mathrm{int},i} = \int_{\Omega_s^0} \mathbf{P}(\mathbf{d}_s^h) : \nabla_0 \boldsymbol{\varphi}_i \, \mathrm{d}\mathbf{X}.$$

The body-force and traction vectors are defined as

$$\begin{aligned} \mathbf{f}_{\mathrm{body}} &= [f_{\mathrm{body}}]_i, \qquad f_{\mathrm{body},i} = \int_{\Omega_s^0} \mathbf{f}_s \cdot \boldsymbol{\varphi}_i \, \mathrm{d}\mathbf{X}, \\ \mathbf{f}_{\mathrm{tract}} &= [f_{\mathrm{tract}}]_i, \qquad f_{\mathrm{tract},i} = \int_{\Gamma_N^0} \mathbf{T} \cdot \boldsymbol{\varphi}_i \, \mathrm{d}S. \end{aligned} \tag{3.18}$$

With these definitions the semi-discrete problem can be described as

$$M_s \ddot{\mathbf{d}}_s^h + \mathbf{f}_{\mathrm{int}}(\mathbf{d}_s^h) = \mathbf{f}_{\mathrm{body}} + \mathbf{f}_{\mathrm{tract}}. \tag{3.19}$$

The nonlinear Newton residual $\mathcal{R}$ at iteration k can be written as

$$\mathcal{R}\left(\mathbf{d}_s^{h,k}\right) := \mathbf{f}_{\mathrm{body}} + \mathbf{f}_{\mathrm{tract}} - M_s \ddot{\mathbf{d}}_s^{h,k} - \mathbf{f}_{\mathrm{int}}\left(\mathbf{d}_s^{h,k}\right), \tag{3.20}$$

with the solution $\mathbf{d}_s^{h,k}$ of Newton step k. The linearization of (3.20) with the Newton increment $\delta\mathbf{d}_s^{h,k} = \mathbf{d}_s^{h,k+1} - \mathbf{d}_s^{h,k}$ is defined as

$$D_{\mathbf{d}_s^h} \mathcal{R}\left(\mathbf{d}_s^{h,k}\right) \delta\mathbf{d}_s^{h,k} = -\mathcal{R}\left(\mathbf{d}_s^{h,k}\right)$$

or in matrix form defining the Newton solve with the corresponding Jacobian

$$\mathcal{J}\left(\mathbf{d}_s^{h,k}\right) \delta\mathbf{d}_s^{h,k} = \mathcal{S}\left(\mathbf{d}_s^{h,k}\right) \delta\mathbf{d}_s^{h,k} = -\mathcal{R}\left(\mathbf{d}_s^{h,k}\right). \tag{3.21}$$

3.3. Time Discretization

To discretize the semi-discrete system (3.19) in time, we employ the Newmark time integration scheme [135]. The Newmark method is a second-order scheme widely used in structural dynamics to integrate equations of motion of the form

$$M\ddot{\mathbf{d}}_s(t) + C\dot{\mathbf{d}}_s(t) + K\mathbf{d}_s(t) = f(t),$$

where $\mathbf{d}_s$ denotes the structural displacement. For a time step from t_n to $t_{n+1} = t_n + \Delta t$, the displacement and velocity are updated according to

$$\begin{aligned} \mathbf{d}_s^{n+1} &= \mathbf{d}_s^n + \Delta t\, \dot{\mathbf{d}}_s^n + \Delta t^2 \left[\left(\frac{1}{2} - \beta \right) \ddot{\mathbf{d}}_s^n + \beta \ddot{\mathbf{d}}_s^{n+1} \right], \\ \dot{\mathbf{d}}_s^{n+1} &= \dot{\mathbf{d}}_s^n + \Delta t \left[(1-\gamma)\ddot{\mathbf{d}}_s^n + \gamma \ddot{\mathbf{d}}_s^{n+1} \right]. \end{aligned} \tag{3.22}$$

The algorithmic parameters β and γ control the stability, accuracy, and numerical dissipation of the Newmark scheme. A common choice is

$$\beta = \frac{1}{4}, \qquad \gamma = \frac{1}{2},$$

which yields a second-order accurate and unconditionally stable scheme for linear problems [100, Section 9.1.2]. More generally, unconditional stability is obtained for

$$2\beta \geq \gamma \geq \frac{1}{2}.$$

Choosing $\beta = 0$ results in an explicit time integration scheme. Substituting the Newmark relations into the semi-discrete residual (3.20) yields the approximation

$$M_s \ddot{\mathbf{d}}_s^{k,n+1} \approx \frac{1}{\Delta t^2 \beta} M_s \mathbf{d}_s^{k,n+1} - M_s \left(\frac{1}{\Delta t^2 \beta} \mathbf{d}_s^{k,n} + \frac{1}{\Delta t \beta} \dot{\mathbf{d}}_s^{k,n} + \frac{\frac{1}{2} - \beta}{\beta} \ddot{\mathbf{d}}_s^{k,n} \right). \tag{3.23}$$

The resulting nonlinear system is solved for the displacement $\mathbf{d}_s^{n+1}$. Subsequently, the velocity and acceleration are updated according to

$$\begin{aligned} \dot{\mathbf{d}}_s^{n+1} &= \frac{\gamma}{\Delta t \beta} \left(\mathbf{d}_s^{n+1} - \mathbf{d}_s^n \right) + \left(1 - \frac{\gamma}{\beta} \right) \dot{\mathbf{d}}_s^n + \Delta t \left(\frac{\beta - \frac{1}{2}\gamma}{\beta} \right) \ddot{\mathbf{d}}_s^n, \\ \ddot{\mathbf{d}}_s^{n+1} &= \frac{1}{\Delta t^2 \beta} \left(\mathbf{d}_s^{n+1} - \mathbf{d}_s^n \right) - \frac{1}{\Delta t \beta} \dot{\mathbf{d}}_s^n - \frac{\frac{1}{2} - \beta}{\beta} \ddot{\mathbf{d}}_s^n. \end{aligned} \tag{3.24}$$

For the test problem presented in Section 9.3, oscillations in the average number of linear iterations can be observed in the FSI simulations. Using different Newmark parameters to introduce a moderate amount of algorithmic damping improved results. Unlike physical damping, algorithmic damping is introduced solely through the time integration scheme and primarily damps high-frequency numerical modes while leaving the low-frequency response largely unaffected [100, Section 9.1.2]. Algorithmic damping is obtained by choosing $\gamma > \frac{1}{2}$. Therefore, in Section 9.3, a value of $\gamma = 0.55$ is used and to retain unconditional stability, the corresponding value of β is selected according to

$$\beta = \frac{1}{4} \left(\gamma + \frac{1}{2} \right)^2,$$

which yields $\beta = 0.275625$; see also [100, Section 9.1.2].

4. The Deformation–Diffusion Problem

To account for the influence of antihypertensive drugs on the arterial wall, the mechanical model presented in Chapter 3 is extended by a diffusion process. The problem is described by the coupling of the balance of linear momentum equation with a diffusion-reaction equation describing the local drug concentrations c. The set of partial differential equations is given as:

$$\text{Balance of linear momentum:} \qquad \rho_s \frac{\partial^2 \mathbf{d}_s}{\partial t^2} - \nabla_0 \cdot \mathbf{P}\left(\mathbf{d}_s, c\right) = \mathbf{f}_s \qquad \text{in } \Omega_s^0, \tag{4.1}$$

$$\text{Diffusion-reaction:} \qquad \frac{\partial c}{\partial t} - \nabla \cdot \Big(\mathbf{D} \nabla c\Big) - r(c) = 0 \qquad \text{in } \Omega_s^t, \tag{4.2}$$

The function $r(c)$ describes, for example, the metabolism rate resulting in the reduction of drug concentration c. The diffusion tensor is isotropic with $\mathbf{D} = D_0\,\mathbf{I}$ and diffusion rate D_0 in cm^2/s. $\mathbf{P}$ is the first Piola-Kirchhoff stress tensor, and ρ_s the solid density in reference configuration. Here, both processes occur simultaneously, namely the deformation due to blood pressure variations and the diffusion of drugs through the arterial wall.

Two coupling mechanisms are considered. In Section 4.1 the stiffness of a neo-Hookean material model is directly influenced by the current drug concentration. In Section 4.2, antihypertensive drugs are considered, which influence the amount of calcium that enters the cells, thereby leading to a reduced contraction. In particular, that coupling is included in the chemo-mechanical SMC model from Section 3.1.2.

4.1. Simple Deformation–Diffusion Model

A simple coupling between a solid and drug concentration can be achieved by changing the value of the Young modulus depending on the current concentration. This simple deformation–diffusion model is based on a neo-Hookean material (see also Section 3.1.1), where the Young modulus is directly influenced by the concentration c of the drug. The strain energy density function of the considered neo-Hookean material model is defined as

$$W_{\mathrm{NH_{SCI}}}(\mathbf{C}) = \frac{\mu}{2}\Big(I_1\, I_3^{-\frac{1}{3}} - 3\Big) + \frac{\kappa}{2}\Big(\sqrt{I_3} - 1\Big)^2, \tag{4.3}$$

with shear modulus μ, bulk modulus κ, right Cauchy-Green deformation tensor $\mathbf{C} := \mathbf{F}^T\mathbf{F}$, principal invariants of $\mathbf{C}$, $I_1 = \mathrm{tr}(\mathbf{C})$ and $I_3 = \det(\mathbf{C}) = J^2$, and the determinant $J = \det(\mathbf{F})$ of the deformation gradient. The coupling between the concentration c and the Young modulus is expressed via

$$E(c) = \mathrm{E}_0 - (\mathrm{E}_0 - \mathrm{E}_1)\frac{c}{c + 0.25}, \tag{4.4}$$

with initial Young modulus E_0 and the final (decreased) value E_1; cf. [138, Section 4]. In (4.3) μ and κ can be substituted by the Young modulus to directly incorporate (4.4)

$$W_{\mathrm{NH}}(\mathbf{C}, c) = E(c)\left(\frac{(I_1\, I_3^{-\frac{1}{3}} - 3)}{4(1+\nu)} + \frac{(\sqrt{I_3}-1)^2}{6(1-2\nu)}\right). \tag{4.5}$$

Results of the simple deformation–diffusion model are included in the interface validation as a proof of concept in Figure 10.4.

4.2. Pharmacological Agents

In Section 3.1.2, the chemo-mechanical SMC model was described, specifically, how the active material response is modeled. One driving factor is the intracellular calcium concentration which facilitates the contraction of the artery. Calcium enters smooth muscle cells mainly through calcium channels located in the cell membrane. One common class of antihypertensive drugs is calcium channel blockers, which reduce this calcium inflow. We will not further elaborate on the properties and details of these antihypertensive drugs. A discussion can be found in [149]. Blocking calcium from entering the cell leads to a reduced contraction and therefore vasodilation, the widening of the blood vessels. Their effect can be modeled by extending the chemical kinetics model given in Section 3.1.2 by making the target calcium concentration $[\mathrm{Ca}^{2+}]_{\mathrm{tar}}$ and the intracellular calcium concentration $[\mathrm{Ca}^{2+}]_{\mathrm{i}}$ dependent on the concentration c of the drug. To accommodate the effect, the parameters γ_1 and γ_3 in Equations (3.9) and (3.10), which control the calcium response, become functions of the current drug concentration c. With higher concentration c, both parameters reduce. Consequently, by reducing γ_1 and γ_3 the drug reduces both the actual calcium concentration and the target calcium concentration. As a result, the phosphorylation rates k_1 and k_6 decrease, fewer cross-bridges become active, and the smooth muscle cell generates less force, i.e., less contraction, which relaxes and dilates the artery. The corresponding equations of γ_1 and γ_3 are shown in [149, Equation 12].

$$\gamma_1(c) = \gamma_{1,\min} + \frac{\gamma_{1,\max} - \gamma_{1,\min}}{1 + e^{\,p_1(c-c_{50})}}, \qquad \gamma_3(c) = \gamma_{3,\min} + \frac{\gamma_{3,\max} - \gamma_{3,\min}}{1 + e^{\,p_3(c-c_{50})}}, \tag{4.6}$$

Qualitative results regarding the vasodilatory effect can be seen in Figure 10.11, and Figure 10.39.

Note that both coupling approaches, the coupling through the Young modulus and through the influence on the intracellular calcium concentration, result in a nonlinear dependency on the drug concentration c. The diffusion process is also influenced by the current deformation of the solid domain, which is reflected in the change of element volume. Linearizing the deformation–diffusion problem results in two off-diagonal coupling blocks.

4.3. Finite Element Discretization

4.3.1. The Diffusion Problem

The finite element discretization of the deformation problem, i.e., the discretization of the balance of linear momentum equation, was described in Section 3.2. Therefore, only the diffusion problem

is further discretized in this section and the resulting off-diagonal blocks from linearization of the deformation–diffusion problem are derived. For the spatial discretization of the diffusion problem, we consider P_2 finite elements. The domain with characteristic mesh size h is partitioned into tetrahedra. Let $\tau_h := \tau_h(\Omega_s^t)$ denote the triangulation and $\mathcal{P}_k(T)$ the polynomial spaces of bilinear functions. For the partition into tetrahedral elements, continuous and piecewise quadratic functions are considered

$$Q^h(\Omega_s^t) = \Big\{ q_h \in \big(C(\Omega_s)\big) \cap H^1(\Omega_s^t) : q_h|_T \in \mathcal{P}_2(T) \ \forall T \in \tau_h \Big\}.$$

Let $c^h \in Q^h$ be the approximation of the solution c. Then the resulting variational formulation is as follows: Find $c^h \in Q^h$ such that

$$\int_{\Omega_s^t} \frac{\partial c^h}{\partial t} q_h \, \mathrm{d}\mathbf{x} + \int_{\Omega_s^t} \nabla q_h \cdot \mathbf{D} \nabla c^h \, \mathrm{d}\mathbf{x} - \int_{\Omega_s^t} r(c^h) q_h \, \mathrm{d}\mathbf{x} = \int_{\Gamma_N} q_h \, g \, \mathrm{d}s \quad \text{for all } q_h \in Q^h, \tag{4.7}$$

where the right-hand side term represents a diffusion flux prescribed on the Neumann boundary $\Gamma_N \subset \partial\Omega_s$ abbreviated as $g := (\mathbf{D}\nabla c^h) \cdot \mathbf{n}$. Equation (4.7) is transformed to the reference configuration to be in the same reference frame as the weak form of the balance of linear momentum (3.17). Transforming the volume elements to the reference configuration yields $\mathrm{d}\mathbf{x} = J \mathrm{d}\mathbf{X}$, with $J = \det(\mathbf{F})$, of the deformation gradient $\mathbf{F}$. Then applying the chain rule yields $\nabla c^h = \nabla_0 \, c^h \mathbf{F}^{-1}$, and similarly, we obtain the weak form of the diffusion-reaction equation in the reference configuration

$$\int_{\Omega_s^0} J \frac{\partial c^h}{\partial t} \cdot q_h \, \mathrm{d}\mathbf{X} + \int_{\Omega_s^0} J \nabla_0 \, q_h \cdot \mathbf{D}_\mathbf{F} \nabla_0 \, c^h \, \mathrm{d}\mathbf{X} - \int_{\Omega_s^0} J r(c^h) \, q_h \, \mathrm{d}\mathbf{X} = \int_{\partial\Gamma_N^0} J g_\mathbf{F} \, q_h \, \mathrm{d}S, \tag{4.8}$$

with $\mathbf{D}_\mathbf{F} := \mathbf{F}^{-1}\mathbf{D}\mathbf{F}^{-T}$, and $g_\mathbf{F} := \mathbf{D}_\mathbf{F} \nabla_0 \, c^h \cdot \mathbf{n}_0$. We denote by $\psi_1, \ldots, \psi_{N_c}$ the basis of $Q^h(\Omega_s^0)$. Then, we can write

$$c^h(\mathbf{X}) = \sum_{i=1}^{N_c} c_i \psi_i(\mathbf{X}).$$

We define the following matrix operators corresponding to the weak formulation of the diffusion-reaction equation. The diffusion matrix is given by

$$A_c = [a]_{ij}, \qquad a_{ij} = \int_{\Omega_s^0} J \, \nabla_0 \, \psi_i \cdot \mathbf{D}_\mathbf{F} \nabla_0 \psi_j \, \mathrm{d}\mathbf{X},$$

the mass matrix is defined as

$$M_c = [m]_{ij}, \qquad m_{ij} = \int_{\Omega_s^0} J \, \psi_i \, \psi_j \, \mathrm{d}\mathbf{X},$$

and the reaction vector and Neumann boundary condition are given by

$$\mathbf{f}_r = [f]_i, \qquad f_i = \int_{\Omega_s^0} J \, r(c^h) \psi_i \, \mathrm{d}\mathbf{X}, \text{ and}$$

$$\mathbf{g}_D = [g]_i, \qquad g_i = \int_{\Gamma_N^0} J \, g_\mathbf{F} \psi_i \, \mathrm{d}S.$$

With these definitions, the resulting semi-discrete diffusion-reaction equation reads

$$M_c \, \dot{\mathrm{c}}^h + A_c \, \mathrm{c}^h - \mathbf{f}_r\Big(c^h\Big) = \mathbf{g}_D. \tag{4.9}$$

4.3.2. The Coupled Problem

In the following we apply the time discretizations to the semi-discrete equations (4.9) and (3.19). The h index is dropped, for more clarity. Furthermore, the diffusion flux and the reaction function are neglected, as they are not further analyzed in this work. We can extend (4.9) with respect to the time discretization from Section 2.3 using a BDF-1 scheme, i.e., backward Euler, to obtain the fully discrete diffusion equation. We use the time step size Δt, denote the time step with n and insert

$$\dot{c}^{n+1} \approx \frac{c^{n+1} - c^n}{\Delta t}$$

into (4.9) to obtain

$$\frac{1}{\Delta t} M_c \, c^{n+1} + A_c \, c^{n+1} = \frac{1}{\Delta t} M_c \, c^n.$$

For the deformation–diffusion problem, the diffusion process is affected by the deformation of the solid domain through changes in the local element volume. This effect is described by the determinant of the deformation gradient, which changes as the geometry deforms. For nearly incompressible materials, however, this contribution is expected to be small. The coupling is now reflected in the definition of the diffusion residual defined as

$$R_c\Big(\mathbf{d}_s^{k,n+1}, c^{k,n+1}\Big) := \left(\frac{1}{\Delta t} M_c\Big(\mathbf{d}_s^{k,n+1}\Big) + A_c(\mathbf{d}_s^{k,n+1})\right) c^{k,n+1} - \frac{1}{\Delta t} M_c\Big(\mathbf{d}_s^{k,n+1}\Big)\, c^n.$$

The precise influence of the drug concentration on the mechanical problem depends on the constitutive model chosen for the arterial wall. Using either the model from Section 4.1 or Section 4.2 results in a nonlinear coupling between diffusion and deformation. The equation (3.19) is discretized in time with the Newmark scheme, see Section 3.3, resulting in the residual

$$R_{\mathbf{d}_s}\Big(\mathbf{d}_s^{k,n+1}, c^{k,n+1}\Big) := \frac{1}{\Delta t^2 \beta} M_s \mathbf{d}_s^{k,n+1} + \mathbf{f}_{\text{int}}\Big(\mathbf{d}_s^{k,n+1}, c^{k,n+1}\Big) - \mathbf{b}_s^{k,n+1},$$

with

$$\mathbf{b}_s^{n+1} = \mathbf{f}_{\text{body}}^{n+1} + \mathbf{f}_{\text{tract}}^{n+1} + M_s\left(\frac{1}{\Delta t^2 \beta}\Big(\mathbf{d}_s^n + \Delta t \dot{\mathbf{d}}_s^n\Big) + \frac{1-2\beta}{2\beta} \ddot{\mathbf{d}}_s^n\right),$$

which is independent of the current displacement $\mathbf{d}_s^{k,n+1}$ and drug concentration. The coupled deformation–diffusion residual can be written as

$$\mathcal{R}\begin{pmatrix} \mathbf{d}_s^{k,n+1} \\ c^{k,n+1} \end{pmatrix} = \begin{bmatrix} R_{\mathbf{d}_s}(\mathbf{d}_s^{k,n+1}, c^{k,n+1}) \\ R_c \ (\mathbf{d}_s^{k,n+1}, c^{k,n+1}) \end{bmatrix} = \mathbf{0}. \tag{4.10}$$

The linearized Newton system is then given by

$$\mathcal{J}\begin{pmatrix} \mathbf{d}_s^{k,n+1} \\ c^{k,n+1} \end{pmatrix} \begin{bmatrix} \delta \mathbf{d}_s^{k,n+1} \\ \delta c^{k,n+1} \end{bmatrix} = -\mathcal{R}\begin{pmatrix} \mathbf{d}_s^{k,n+1} \\ c^{k,n+1} \end{pmatrix}.$$

The coupled Jacobian has the structure

$$\mathcal{J} = \begin{bmatrix} D_{\mathbf{d}_s} R_{\mathbf{d}_s} & D_c R_{\mathbf{d}_s} \\ D_{\mathbf{d}_s} R_c & D_c R_c \end{bmatrix} =: \begin{bmatrix} \frac{1}{\Delta t^2 \beta} M_s + K^{k,n+1}_{\mathbf{d}_s,\mathbf{d}_s} & K^{k,n+1}_{\mathbf{d}_s,c} \\ K^{k,n+1}_{c,\mathbf{d}_s} & \frac{1}{\Delta t} M_c^{k,n+1} + A_c^{k,n+1} \end{bmatrix}. \tag{4.11}$$

As a result, the deformation and diffusion-reaction problems are fully nonlinearly coupled. For clarity, the dependencies of the respective blocks in (4.11) on $\mathbf{d}_s^{k,n+1}$ and $c^{k,n+1}$ are omitted in the notation. We define

$$\mathcal{S}\Big(\mathbf{d}_s^{k,n+1}, c^{k,n+1}\Big) = \frac{1}{\Delta t^2 \beta} M_s + K^{k,n+1}_{\mathbf{d}_s,\mathbf{d}_s} \quad \text{and} \quad \mathcal{CH}\Big(\mathbf{d}_s^{k,n+1}, c^{k,n+1}\Big) = \frac{1}{\Delta t} M_c^{k,n+1} + A_c^{k,n+1},$$

which correspond to the solid and diffusion subproblems, respectively. When neglecting both velocity and acceleration terms from the balance of momentum equation of the original dynamic problem, we get a quasi-static solid problem, assuming that the displacements develop slowly and smoothly; see for example [144] for some discussion on the subject. The corresponding Newton system reads

$$\begin{bmatrix} K^{k,n+1}_{\mathbf{d}_s,\mathbf{d}_s} & K^{k,n+1}_{\mathbf{d}_s,c} \\ K^{k,n+1}_{c,\mathbf{d}_s} & \frac{1}{\Delta t} M_c^{k,n+1} + A_c^{k,n+1} \end{bmatrix} \begin{bmatrix} \delta \mathbf{d}_s^{k,n+1} \\ \delta c^{k,n+1} \end{bmatrix} = - \begin{pmatrix} \mathbf{f}_{\text{int}}(\mathbf{d}_s^{k,n+1}, c^{k,n+1}) - \mathbf{f}_{\text{body}}^{n+1} - \mathbf{f}_{\text{tract}}^{n+1} \\ R_c \end{pmatrix}. \tag{4.12}$$

To improve the robustness of the Newton method, a load-stepping strategy can be employed. The prescribed load is applied gradually over $\hat{N}$ pseudo-time steps (load steps), rather than being imposed at once. The idea is similar to a ramping phase and helps to avoid convergence difficulties. At load step $\hat{n} = 1, \dots, \hat{N}$, the applied pressure is

$$p_m = \frac{\hat{n}}{\hat{N}} p, \qquad \text{for } \hat{n} = 1, \dots, \hat{N}, \tag{4.13}$$

where p is the prescribed target pressure value.

If we treat the solid and diffusion problems separately, an explicit approach can be defined. Therefore, the diffusion problem is solved at the beginning of each Newton iteration, and the corresponding concentration is used to solve the deformation problem. The approach is denoted as "Chemistry Explicit" (ChE). The ChE approach is defined as follows for the quasi-static solid problem

$$\begin{aligned} &\text{(1) Solve :} && \Big(\frac{1}{\Delta t} M_c + A_c\Big) c^{n+1} &&= \frac{1}{\Delta t} M_c c^n, \\ &\text{(2) Solve :} && K^{k,n+1}_{\mathbf{d}_s,\mathbf{d}_s}\, \delta \mathbf{d}_s^{k,n+1} &&= -\Big(\mathbf{f}_{\text{int}}(\mathbf{d}_s^{k,n+1}, c^{n+1}) - \mathbf{f}_{\text{body}}^{n+1} - \mathbf{f}_{\text{tract}}^{n+1}\Big), \\ &&& \text{with} \quad \delta \mathbf{d}_s^{k,n+1} &&= \mathbf{d}_s^{k+1,n+1} - \mathbf{d}_s^{k,n+1} \quad \text{for } k = 1, 2, \dots . \end{aligned} \tag{4.14}$$

Note that we assume $M_c \approx M_c(\mathbf{d}_s^n)$ and $A_c \approx A_c(\mathbf{d}_s^n)$, since the material is nearly incompressible, e.g., for the neo-Hookean material model the Poisson's ratio is set to $\nu = 0.49$. For the chemo-mechanical SMC model from Section 3.1.2, near-incompressible behavior is enforced through the volumetric part of the strain-energy density function. In the present setting, the parameters $\alpha_2 = 151\,\text{kPa}$ and $\alpha_3 = 2.8$ provide a strong volumetric penalty, such that only small local volume changes are expected; see *Set 1* in Table 9.8.

5. Fluid–Structure Interaction Problem

In this chapter, the coupled fluid–structure interaction (FSI) problem is introduced. To model the interaction between blood flow and the surrounding arterial wall, the incompressible Navier–Stokes equations presented in Chapter 2 are coupled with the solid mechanics model introduced in Chapter 3. Since the incompressible Navier–Stokes equations (2.1) describe Newtonian fluids, blood is modeled as a Newtonian fluid with constant viscosity. The resulting model consists of a coupled system of nonlinear partial differential equations describing the interaction between the fluid and the arterial wall.

The fluid and solid problems are defined on different physical domains. The fluid domain represents the blood flowing through the artery, whereas the surrounding arterial wall is modeled by the structural domain. Both equations are coupled, as the deformation of the arterial wall influences the blood flow, and the fluid applies forces on the wall. The interaction between the fluid and the structure is expressed through the coupling conditions at the fluid–structure interface. These consist of geometric adherence, requiring the fluid domain to follow the structural displacement, kinematic continuity, requiring the fluid and structure to share the same velocity at the interface, and continuity of stresses, requiring the traction forces to be balanced across the interface.

As stated in the previous sections, the governing equations of the fluid and the solid are formulated in different reference frames. The fluid equations are described in an Eulerian reference frame, whereas the structural equations are formulated in a Lagrangian description. To consistently describe the motion of the deforming fluid domain, an Arbitrary Lagrangian–Eulerian (ALE) formulation is employed. Several coupling strategies exist for solving FSI problems, including partitioned and monolithic approaches. In this work, a monolithic formulation is adopted, in which the fluid and solid equations are assembled into a single system and solved simultaneously; see Section 6.2 for the considered solver and preconditioning strategy, and further discussion.

Let $\Omega \subset \mathbb{R}^3$ be the combined fluid and solid domain. The fluid subdomain is denoted as $\Omega_f^0 \subset \Omega$ and the solid subdomain is denoted as $\Omega_s^0 \subset \Omega$, both in the initial or reference configuration at time $t = 0$, respectively. The interface between the fluid and solid subdomains is defined by $\Gamma^0 = \partial\Omega_f^0 \cap \partial\Omega_s^0$ as the intersection between the two domains. For the unsteady problem, the fluid and solid domains Ω_f^0 and Ω_s^0 deform over time. Then, we refer to them as Ω_f^t and Ω_s^t for time t, respectively. The following derivation of the FSI system follows [64, Section 3.5, Subsection 9.1.2] and [96, Section 4, Section 5].

To model the fluid in the moving domain Ω_f^t the arbitrary Lagrangian–Eulerian (ALE) framework is used; see also [46]. We define the ALE mapping $\mathcal{A}_f$ from Ω_f^0 to Ω_f^t by

$$\mathcal{A}_f : \Omega_f^0 \times [0, T] \to \Omega_f^t, \quad \mathcal{A}_f(\mathbf{X}) = \mathbf{X} + \mathbf{d}_f(\mathbf{X}), \quad \forall\, \mathbf{X} \subseteq \Omega_f^0,$$

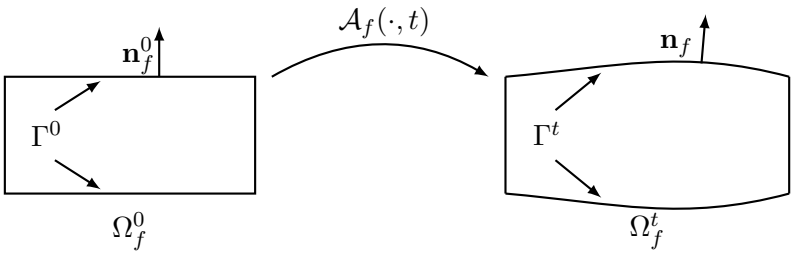


Figure 5.1: Description of the motion of the computational fluid domain by the ALE mapping $\mathcal{A}_f$. The reference configuration Ω_f^0 and the current configuration Ω_f^t. The unit normal vector on the interface changes due to the deformation and is therefore defined in the reference and current configuration as $\mathbf{n}_f^0$ and $\mathbf{n}_f$. Based on [64, Figure 3.17].

which expresses the motion of the fluid domain; see (3.1) and (3.2). The coordinates $\mathbf{X}$ of the reference configuration Ω_f^0 are the ALE coordinates and $\mathbf{d}_f$ is the displacement of the fluid domain.

The displacement of the fluid domain $\mathbf{d}_f$ at the reference configuration Ω_f^0 is determined by a harmonic extension of the moving fluid–structure interface, i.e., by solving the following problem [64, Equation 3.103]

$$\begin{aligned} -\Delta_0 \mathbf{d}_f &= \mathbf{0} \quad \text{in } \Omega_f^0, \\ \mathbf{d}_f &= \mathbf{d}_s \quad \text{on } \Gamma^0, \\ \mathbf{d}_f \cdot \mathbf{n}_f^0 &= \mathbf{0} \quad \text{on } \partial\Omega_f^0 \setminus \Gamma^0, \end{aligned} \tag{5.1}$$

where $\mathbf{n}_f^0$ is the outward normal vector field of the reference fluid domain; $\mathbf{n}_s^0$ is the corresponding field for the solid domain. We refer to (5.1) as the geometry subproblem. In [152, Section 2.5.3] other choices to model the deformation of the fluid domain are described.

The transient incompressible Navier–Stokes equations to model a Newtonian fluid defined on the moving domain Ω_f^t are then written in the ALE formulation

$$\begin{aligned} \rho_f \left(\frac{\partial \mathbf{u}_f}{\partial t}\Big|_{\mathbf{X}} + [(\mathbf{u}_f - \mathbf{w}) \cdot \nabla] \mathbf{u}_f \right) - \nabla \cdot \boldsymbol{\sigma}_f(\mathbf{u}_f, p) &= \mathbf{0} \qquad \text{in } \Omega_f^t \times (0,T], \\ \operatorname{div}(\mathbf{u}_f) &= 0 \qquad \text{in } \Omega_f^t \times (0,T], \end{aligned} \tag{5.2}$$

with fluid density ρ_f, fluid velocity $\mathbf{u}_f$, and pressure p. Note that compared to the equations defined in (2.1), the notation $\boldsymbol{\sigma}_f(\mathbf{u}_f, p) = \mu_f \left(\nabla \mathbf{u}_f + (\nabla \mathbf{u}_f)^T\right) - p\mathbf{I}$ with the Cauchy stress tensor and dynamic viscosity μ_f, is chosen. It is more consistent with respect to coupling condition (3), the continuity of stresses. The velocity of the moving fluid domain is described by

$$\mathbf{w} = \frac{\partial \mathbf{d}_f}{\partial t}\Big|_{\mathbf{X}}.$$

Note that the time derivative is taken with respect to the ALE coordinates, as indicated by $\mathbf{X}$. The coupling between the fluid and the solid domain is expressed in three coupling conditions on the interface Γ^0:

(1) The geometric adherence of the fluid domain and solid domain (geometric condition) which is given by

$$\mathbf{d}_f = \mathbf{d}_s \tag{5.3}$$

(2) the continuity of the velocities (kinematic condition)

$$\frac{\partial \mathbf{d}_s}{\partial t} = \mathbf{u}_f \circ \mathcal{A}_f, \tag{5.4}$$

(3) the continuity of the stresses (dynamic condition)

$$\mathbf{n}_f^0 \cdot J\,(\boldsymbol{\sigma}_f \circ \mathcal{A}_f)\,\mathbf{F}^{-T} = \mathbf{n}_s^0 \cdot \mathbf{P}, \tag{5.5}$$

with ALE deformation gradient $\mathbf{F} = \nabla_0 \mathcal{A}_f$, and $J = \det \mathbf{F}$. Since (5.5) is evaluated at the interface in reference configuration, we can replace $\mathbf{n}_s^0 = -\mathbf{n}_f^0$, with $\mathbf{n}_0$. Then, the fully coupled FSI system in ALE form reads

$$\begin{aligned}
\rho_f\Big(\frac{\partial \mathbf{u}_f}{\partial t}\Big|_{\mathbf{X}} + [(\mathbf{u}_f - \mathbf{w})\cdot\nabla]\mathbf{u}_f\Big) - \nabla\cdot\boldsymbol{\sigma}_f(\mathbf{u}_f, p) &= \mathbf{0} && \text{in } \Omega_f^t \times (0,T],\\
\operatorname{div}(\mathbf{u}_f) &= 0 && \text{in } \Omega_f^t \times (0,T],\\
\rho_s \frac{\partial^2}{\partial t^2}\mathbf{d}_s - \nabla_0\cdot(\mathbf{P}(\mathbf{d}_s)) &= \mathbf{f}_s && \text{in } \Omega_s^0 \times (0,T],\\
\frac{\partial \mathbf{d}_s}{\partial t} &= \mathbf{u}_f \circ \mathcal{A}_f && \text{on } \Gamma^0 \times (0,T],\\
\mathbf{n}_0 \cdot J\,(\boldsymbol{\sigma}_f \circ \mathcal{A}_f)\,\mathbf{F}^{-T} &= -\mathbf{n}_0 \cdot \mathbf{P}, && \text{on } \Gamma^0,\\
\mathbf{d}_f &= \mathbf{d}_s && \text{on } \Gamma^0.
\end{aligned}$$

To also reflect the coupling of deformation–diffusion defined in Chapter 4, the reaction–diffusion equation is added to the system, as well as the coupling to the deformation problem.

$$\begin{aligned}
\rho_f\left(\frac{\partial \mathbf{u}_f}{\partial t}\Big|_{\mathbf{X}} + [(\mathbf{u}_f - \mathbf{w})\cdot\nabla]\mathbf{u}_f\right) - \nabla\cdot\boldsymbol{\sigma}_f(\mathbf{u}_f, p) &= \mathbf{0} && \text{in } \Omega_f^t \times (0,T],\\
\operatorname{div}(\mathbf{u}_f) &= 0 && \text{in } \Omega_f^t \times (0,T],\\
\rho_s \frac{\partial^2}{\partial t^2}\mathbf{d}_s - \nabla_0\cdot(\mathbf{P}(\mathbf{d}_s, c)) &= \mathbf{f}_s && \text{in } \Omega_s^0 \times (0,T],\\
\frac{\partial c}{\partial t} - \nabla\cdot\mathbf{D}\,\nabla c - r(c) &= \mathbf{0} && \text{in } \Omega_s^t \times (0,T],\\
\frac{\partial \mathbf{d}_s}{\partial t} &= \mathbf{u}_f \circ \mathcal{A}_f && \text{on } \Gamma^0 \times (0,T],\\
\mathbf{n}_0 \cdot J\,(\boldsymbol{\sigma}_f \circ \mathcal{A}_f)\,\mathbf{F}^{-T} &= -\mathbf{n}_0 \cdot \mathbf{P}, && \text{on } \Gamma^0,\\
\mathbf{d}_f &= \mathbf{d}_s && \text{on } \Gamma^0.
\end{aligned}$$

5.1. Finite Element and Time Discretization

To construct the finite element formulation, Newton linearization, and time discretization, the derivations from the previous sections are used. For the discretization of the fluid flow problem, $P_2 - P_1$ finite elements are used. The deformation–diffusion subproblem is discretized via $P_2 - P_2$ finite elements; see Sections 2.2, 3.2, and 4.3. In the following, the coupling blocks reflecting the coupling conditions between solid and fluid are defined. The matrix C_1 is a restriction matrix from all fluid degrees of freedom to the velocity interface degrees of freedom and is used to couple the fluid velocity and the solid velocity on the interface

$$\frac{\partial \mathbf{d}_s}{\partial t} = \mathbf{u}_f \circ \mathcal{A}_t.$$

Similarly, the matrix $\widehat{C}_2$ is the restriction matrix of the displacement degrees of freedom to the interface. This results in the following equation in matrix form

$$C_1 \mathbf{u}_f^{n+1} = \frac{1}{\Delta t}\left(\widehat{C}_2 \mathbf{d}_s^{n+1} - \widehat{C}_2 \mathbf{d}_s^{n}\right).$$

Since $\widehat{C}_2$ needs to be scaled with $\frac{1}{\Delta t}$, we define $C_2 = -\frac{1}{\Delta t}\widehat{C}_2$. The coupling of stresses in normal direction is enforced via the Lagrange multiplier λ. The transposed matrices $C_3 = C_1^T$ and $C_4 = C_2^T$ are used to prolong the Lagrange multiplier contributions from the interface to the fluid velocity and solid displacement degrees of freedom, respectively. The geometry problem defined in (5.1) is denoted with $\mathcal{G}$. The coupling matrix C_5 is used to model the geometric coupling condition; cf. (5.3). It restricts the solid displacement to the interface degrees of freedom associated with the fluid mesh displacement. Since the same interface restriction is used as for the displacement coupling, we set $C_5 = \widehat{C}_2$. This results in the algebraic interface equation

$$\mathcal{G}\mathbf{d}_f^{n+1} = -C_5 \mathbf{d}_s^{n+1},$$

where the minus sign is due to the chosen sign convention in the coupling equation. This relation enforces the geometric condition

$$\mathbf{d}_f|_\Gamma = \mathbf{d}_s|_\Gamma.$$

The full system discretized in time then reads

$$\begin{pmatrix} F(\mathbf{u}_f^{n+1}, \mathbf{d}_f^{n+1})\mathbf{u}_f^{n+1} & + & B^T(\mathbf{d}_f^{n+1})p^{n+1} & + & 0 & + & C_3\lambda^{n+1} & + & 0 \\ B(\mathbf{d}_f^{n+1})\mathbf{u}_f^{n+1} & + & 0 & + & 0 & + & 0 & + & 0 \\ 0 & + & 0 & + & S(\mathbf{d}_s^{n+1}) & + & C_4\lambda^{n+1} & + & 0 \\ C_1\mathbf{u}_f^{n+1} & + & 0 & + & C_2\mathbf{d}_s^{n+1} & + & 0 & + & 0 \\ 0 & + & 0 & + & C_5\mathbf{d}_s^{n+1} & + & 0 & + & \mathcal{G}\mathbf{d}_f^{n+1} \end{pmatrix} = \begin{pmatrix} \widehat{\mathbf{b}}_f^n \\ 0 \\ \mathbf{b}_s^n \\ C_2\mathbf{d}_s^n \\ 0 \end{pmatrix}.$$

The fourth block row represents the kinematic coupling condition, whereas the fifth block row enforces the geometric coupling condition for the ALE displacement. The right-hand side vectors

of $\widehat{\mathbf{b}}_f^n$ and $\mathbf{b}_s^n$ are defined as

$$\widehat{\mathbf{b}}_f^n = \frac{2}{\Delta t} M_f \mathbf{u}_f^n - \frac{1}{2\Delta t} M_f \mathbf{u}_f^{n-1}, \qquad \mathbf{b}_s^n = M_s \left(\frac{1}{\Delta t^2 \beta} \left(\mathbf{d}_s^n + \Delta t \dot{\mathbf{d}}_s^n \right) + \frac{1-2\beta}{2\beta} \ddot{\mathbf{d}}_s^n \right).$$

For the fluid problem, the BDF-2 time stepping scheme is applied; cf. Section 2.3. The nonlinear structural contribution after Newmark time discretization is denoted by

$$S\left(\mathbf{d}_s^{n+1}\right) := \frac{1}{\Delta t^2 \beta} M_s \mathbf{d}_s^{n+1} + \mathbf{f}_s^{\text{int}}\left(\mathbf{d}_s^{n+1}\right),$$

where $\mathbf{f}_s^{\text{int}}$ denotes the vector of internal structural forces; cf. (3.23). Here, S denotes a nonlinear vector-valued operator and not a matrix. The corresponding linearization is denoted later by $\mathcal{S}$.

The variational formulation of the fluid problem is extended to accommodate the ALE formulation from (5.2); see (2.6) for the variational formulation of the stationary Navier–Stokes equation. This leads to the following equations

$$\begin{aligned}
\int_{\Omega_f^t} \rho_f \frac{\partial \mathbf{u}_f^h}{\partial t}\Big|_{\mathbf{X}} \cdot \mathbf{v}_f^h \, \mathrm{d}\mathbf{x} + \int_{\Omega_f^t} \mu_f \left(\nabla \mathbf{u}_f^h + \nabla \mathbf{u}_f^{h,T} \right) : \nabla \mathbf{v}_f^h \, \mathrm{d}\mathbf{x} & \\
+ \int_{\Omega_f^t} \rho_f \left((\mathbf{u}_f^h - \mathbf{w}^h) \cdot \nabla \right) \mathbf{u}_f^h \cdot \mathbf{v}_f^h \, \mathrm{d}\mathbf{x} - \int_{\Omega_f^t} p^h \nabla \cdot \mathbf{v}_f^h \, \mathrm{d}\mathbf{x} \quad &= 0 \qquad \forall \mathbf{v}_f^h \in \mathbf{V}_f^h, \\
\int_{\Omega_f^t} q^h \nabla \cdot \mathbf{u}_f^h \, \mathrm{d}\mathbf{x} \quad &= 0 \qquad \forall q^h \in Q_p^h.
\end{aligned}$$

The first term can be transformed to

$$\int_{\Omega_f^t} \rho_f \frac{\partial \mathbf{u}_f^h}{\partial t}\Big|_{\mathbf{X}} \cdot \mathbf{v}_f^h \, \mathrm{d}\mathbf{x} = \frac{d}{dt} \int_{\Omega_f^t} \rho_f \mathbf{u}_f^h \cdot \mathbf{v}_f^h \, \mathrm{d}\mathbf{x} - \int_{\Omega_f^t} \rho_f (\nabla \cdot \mathbf{w}) \mathbf{u}_f^h \cdot \mathbf{v}_f^h \, \mathrm{d}\mathbf{x},$$

compare [64, Subsection 9.1.2]. The following additional matrix operators corresponding to the weak formulation, cf. Section 2.2, are defined

$$\begin{aligned}
A_f = [a]_{ij}, \qquad & a_{ij} = \int_{\Omega_f^t} \mu_f (\nabla \boldsymbol{\varphi}_j + \nabla \boldsymbol{\varphi}_j^T) : \nabla \boldsymbol{\varphi}_i \, \mathrm{d}\mathbf{x}, \\
P_f(\mathbf{w}_f^h) = [p]_{ij}, \qquad & p_{ij} = \int_{\Omega_f^t} \rho_f (\nabla \cdot \mathbf{w}_f^h) \boldsymbol{\varphi}_j \cdot \boldsymbol{\varphi}_i \, \mathrm{d}\mathbf{x}, \\
N_f(\mathbf{u}_f^h, \mathbf{w}_f^h) = [n(\mathbf{u}_f^h, \mathbf{w}_f^h)]_{ij}, \qquad & n_{ij} = \int_{\Omega_f^t} \rho_f (((\mathbf{u}_f^h - \mathbf{w}_f^h) \cdot \nabla) \boldsymbol{\varphi}_j) \cdot \boldsymbol{\varphi}_i \, \mathrm{d}\mathbf{x}.
\end{aligned}$$

The index h is omitted in the following for clarity. Note that the ALE mesh velocity $\mathbf{w}_f^{n+1}$ is obtained from the ALE displacement $\mathbf{d}_f^{n+1}$. Thus, the fluid operator depends on both the fluid unknowns and the current ALE configuration. With the above definitions, the fluid velocity operator is given by

$$F\left(\mathbf{u}_f^{n+1}, \mathbf{d}_f^{n+1}\right) := \frac{2}{\Delta t} M_f^{n+1} + A_f^{n+1} + N_f\left(\mathbf{u}_f^{n+1}, \mathbf{w}_f^{n+1}\right) - P_f\left(\mathbf{w}_f^{n+1}\right).$$

For Newton's method, the fluid operator is linearized at the current iterate $(\mathbf{u}_f^{k,n+1}, p^{k,n+1}, \mathbf{d}_f^{k,n+1})$. The derivative with respect to the fluid velocity contains, in addition to the convection matrix N_f, the contribution arising from the linearization of the convective velocity. This contribution is defined by

$$W_f(\mathbf{u}_f^k) = [w(\mathbf{u}_f^k)]_{ij}, \qquad w_{ij} = \int_{\Omega_f^t} \rho_f((\boldsymbol{\varphi}_j \cdot \nabla)\mathbf{u}_f^k) \cdot \boldsymbol{\varphi}_i \, \mathrm{d}\mathbf{x}.$$

Consequently, the velocity block of the fluid linearization is

$$D_{\mathbf{u}_f} F\left(\mathbf{u}_f^{k,n+1}, \mathbf{d}_f^{k,n+1}\right) = \frac{2}{\Delta t} M_f^{n+1} + A_f^{n+1} + N_f\left(\mathbf{u}_f^{k,n+1}, \mathbf{w}_f^{k,n+1}\right) + W_f\left(\mathbf{u}_f^{k,n+1}\right) - P_f\left(\mathbf{w}_f^{k,n+1}\right).$$

We now collect the nonlinear equations in residual form. At Newton step k and time step $n+1$, the residual is defined by

$$\mathcal{R}(X^{n+1,k}) := \begin{pmatrix} \widehat{\mathbf{b}}_f^n \\ 0 \\ \mathbf{b}_s^n \\ C_2 \mathbf{d}_s^n \\ 0 \end{pmatrix} - \begin{pmatrix} F\left(\mathbf{u}_f^{n+1,k}, \mathbf{d}_f^{n+1,k}\right) \mathbf{u}_f^{n+1,k} + B^T\left(\mathbf{d}_f^{n+1,k}\right) p^{n+1,k} + C_3 \lambda^{n+1,k} \\ B\left(\mathbf{d}_f^{n+1,k}\right) \mathbf{u}_f^{n+1,k} \\ S\left(\mathbf{d}_s^{n+1,k}\right) + C_4 \lambda^{n+1,k} \\ C_1 \mathbf{u}_f^{n+1,k} + C_2 \mathbf{d}_s^{n+1,k} \\ C_5 \mathbf{d}_s^{n+1,k} + \mathcal{G} \mathbf{d}_f^{n+1,k} \end{pmatrix},$$

with $X = (\mathbf{u}_f, p, \mathbf{d}_s, \lambda, \mathbf{d}_f)$. The linearizations of the fluid and solid parts are denoted by $\mathcal{F}$ and $\mathcal{S}$, respectively. More precisely, the fluid block is given by

$$\mathcal{F}\left(\mathbf{u}_f^{k,n+1}, p^{k,n+1}, \mathbf{d}_f^{k,n+1}\right) := \begin{pmatrix} D_{\mathbf{u}_f} F\left(\mathbf{u}_f^{k,n+1}, \mathbf{d}_f^{k,n+1}\right) & B^T\left(\mathbf{d}_f^{k,n+1}\right) \\ B\left(\mathbf{d}_f^{k,n+1}\right) & 0 \end{pmatrix}.$$

The dependence of the fluid residual on the ALE displacement gives rise to shape derivative contributions, which are denoted by $\mathcal{D}$; see [63] for a derivation of the shape derivatives. Therefore, the Jacobian of the fully implicit FSI problem has the block structure

$$\mathcal{J}(\mathbf{X}^{n+1,k}) = \begin{pmatrix} \mathcal{F}(\mathbf{u}_f^{n+1,k}, p^{n+1,k}, \mathbf{d}_f^{n+1,k}) & 0 & C_3 & \mathcal{D} \\ 0 & \mathcal{S}(\mathbf{d}_s^{n+1,k}) & C_4 & 0 \\ C_1 & C_2 & 0 & 0 \\ 0 & C_5 & 0 & \mathcal{G} \end{pmatrix}. \tag{5.6}$$

The resulting system of equations is highly nonlinear due to the convective term in the fluid momentum equation, the dependence on the ALE displacement, and the nonlinear structural contribution. In [33, Section 3.2], convective explicit (CE) treatments of the fluid convection are discussed, as well as a geometry-convective explicit (GCE) approach, where both the convective term and the ALE displacement $\mathbf{d}_f$ are treated explicitly. In this work, we consider the fully implicit formulation with the Jacobian given in (5.6).

For the deformation–diffusion problem introduced in Chapter 4, the structural part of the FSI formulation is replaced by the corresponding deformation–diffusion formulation. Hence, the operator S in the residual and the block $\mathcal{S}$ in the Jacobian are substituted by the residual

and Jacobian of the deformation–diffusion problem defined in (4.10) and (4.11). The resulting pharmaco–mechanical fluid–structure interaction is denoted by “FSCI”. Note that we can also use an explicit coupling of the deformation–diffusion problem as defined in (4.14), which is denoted by “ChE”.

6. Block Preconditioners

In this chapter, block preconditioners for the incompressible fluid flow, modeled by the incompressible Navier–Stokes equations (Chapter 2), and FSI (Chapter 5) are discussed.

Block preconditioners are a common approach to solve the linear systems arising from the discretization of the Navier–Stokes equations. Usually they are based on a block factorization of the system matrix, which involves the Schur complement. Since the assembly of the exact Schur complement is usually too expensive, finding a suitable approximation for the construction of block preconditioners is an integral task. In the following section, we consider different block preconditioning methods, which are based on different Schur complement approximations. The Augmented Lagrangian approach in Section 6.1.4 introduces an additional term to the velocity block, which facilitates a more straightforward Schur complement approximation. Specifically, in Section 6.1, the Pressure Convection–Diffusion (PCD), Least-Squares Commutator (LSC), SIMPLE, and augmented Lagrangian (AL) block preconditioners are presented. For the application of the resulting block preconditioners, the action of the inverses of block matrices is required, which are approximated by two-level overlapping Schwarz methods; see Chapter 7.

In Section 6.2 the FaCSI factorization is described, which defines a block factorization of the FSI system, where the arising block inverses — corresponding to the different physical subproblems — can be approximated with two-level overlapping Schwarz methods; see Chapter 7.

6.1. Block Preconditioners for a Navier–Stokes Problem

The definition of a suitable approximation is highly problem dependent. For a Stokes problem, a different preconditioner can be optimal than for a Navier–Stokes problem. In particular the nonlinearity, which is influenced by the Reynolds number, can play a significant role. Furthermore, the preconditioning approach can vary if the problem is stationary or time-dependent, if Newton's method is used for linearization or a Picard iteration, if it is compressible flow or incompressible, Newtonian or non-Newtonian, and also the choice of the time and space discretization matters.

For a Navier–Stokes problem with kinematic viscosity ν, velocity v, and a characteristic length L (for example, the diameter of a vessel), the Reynolds number is defined as

$$\mathrm{Re} = \frac{vL}{\nu}. \tag{6.1}$$

For steady-state Stokes problems and a stable discretization, the Schur complement can be replaced with the spectrally equivalent, scaled pressure mass matrix $-\frac{1}{\nu}M_p$ [54, 114]. We may, thus, expect that this approach also works well for small Reynolds numbers. For higher Reynolds numbers, the pressure mass matrix cannot account for the dominating advective forces in F. Therefore, for larger advective forces, a different approximation of the Schur complement

should be used. The considered block methods are the Pressure Convection–Diffusion (PCD) [158, 104, 55, 157, 57, 99, 37], Least-Squares Commutator (LSC) [56, 55, 157, 52, 57], and augmented Lagrangian approaches [17, 20, 55, 157, 57, 60, 18, 19], which are all based on a block-triangular factorization of the system matrix, and the SIMPLE (Semi-Implicit Method for Pressure Linked Equations) [139, 142, 51, 157, 37] preconditioner. These methods each pursue a different approach in finding a suitable approximation of the Schur complement.

We introduce a block LDU decomposition of the Navier–Stokes saddle-point problem (2.16),

$$\mathcal{F} = \begin{bmatrix} F & B^T \\ B & -C \end{bmatrix} = \begin{bmatrix} I & 0 \\ BF^{-1} & I \end{bmatrix} \begin{bmatrix} F & 0 \\ 0 & S \end{bmatrix} \begin{bmatrix} I & F^{-1}B^T \\ 0 & I \end{bmatrix}, \tag{6.2}$$

where $S = -C - BF^{-1}B^T$ is the Schur complement. Generally, we seek to replace F and the Schur complement S with suitable approximations $\hat{F}$ and $\hat{S}$ that facilitate the computation of $\hat{F}^{-1}$ and $\hat{S}^{-1}$. Suitable approximations depend on the problem setting. The approximation of F^{-1} will be based on a Schwarz preconditioner (see Chapter 7), and the approximation of S^{-1} depends on the considered Schur complement approximation. The considered block preconditioners contain inverses of matrices that will be further approximated with a Schwarz method. A resulting upper block-triangular preconditioner reads

$$\mathcal{B}_{\text{Tri}}^{-1} = \begin{bmatrix} \hat{F} & B^T \\ 0 & \hat{S} \end{bmatrix}^{-1} = \begin{bmatrix} \hat{F}^{-1} & -\hat{F}^{-1}B^T\hat{S}^{-1} \\ 0 & \hat{S}^{-1} \end{bmatrix}, \tag{6.3}$$

which encapsulates the process of a block-Gaussian elimination.

The PCD and LSC Schur complement approximations are motivated by finding a suitable approximation of F^{-1}, which is used to construct the Schur complement. Here and in the following, for the construction of approximations of F^{-1}, we assume that $F = F_O$, the discrete Oseen operator. In an implementation, however, we will use the general F in (2.16) that includes an additional term from the Newton linearization. For the consideration of the time discretization, see (6.9). The discrete Oseen operator is the discretization of a convection–diffusion differential operator for the velocity,

$$\mathcal{L} := -\nu\Delta + \mathbf{w}_h \cdot \nabla,$$

where $\mathbf{w}_h$ is the approximation to the velocity solution from the previous nonlinear iteration. The discretization of $\mathcal{L}$ corresponds to $F(\mathbf{w}_h) = \nu A + N(\mathbf{w}_h)$. We denote the corresponding differential operator on the pressure space by

$$\mathcal{L}_p := (-\nu\Delta + \mathbf{w}_h \cdot \nabla)_p.$$

Its discretization is given by $F_p(\mathbf{w}_h) = \nu A_p + N_p(\mathbf{w}_h) = [f_{p,ij}(\mathbf{w}_h)]$ with

$$f_{p,ij}(\mathbf{w}_h) = \nu \int_{\Omega_f} \nabla\psi_j \cdot \nabla\psi_i \,\mathrm{d}\mathbf{x} + \int_{\Omega_f} (\mathbf{w}_h \cdot \nabla\psi_j)\psi_i \,\mathrm{d}\mathbf{x}.$$

Using the convection–diffusion operators, we can define a commutator [55, (9.13)]

$$\mathcal{E} := \nabla \cdot (-\nu\Delta + \mathbf{w}_h \cdot \nabla) - (-\nu\Delta + \mathbf{w}_h \cdot \nabla)_p \nabla\cdot \tag{6.4}$$

Table 6.1: Operators appearing in the different preconditioning strategies. These operators are usually further approximated; for example, F^{-1} can be approximated with a Schwarz method. Often the inverse of a mass matrix M is replaced by $H_M^D = \operatorname{diag}(M)^{-1}$, the inverse of the diagonal entries of M. Then, an elaborate strategy (e.g., a Schwarz method) for the approximation of M^{-1} is not necessary.

Operator	Monolithic	PCD	SIMPLE	LSC	LSC_{A_p}	$\mathrm{LSC}_{\mathrm{stab},A_p}$	AL
$\mathcal{F}^{-1}$	✓						
F^{-1}		✓	✓	✓	✓	✓	
A_γ^{-1}							✓
A_p^{-1}		✓			✓	✓	
M_p^{-1}		✓					✓
M_u^{-1}				✓	✓	✓	
$\rho_{\mathrm{eig}}^{-1} D^{-1}$						✓	
$(BM_u^{-1}B^T)^{-1}$				✓			
$(-C - BH_F B^T)^{-1}$			✓				

to find an approximation $\hat{S}$ of the Schur complement S. Note that it is assumed that the differential operators are applied to functions that are sufficiently regular; they are not well defined for the used finite element spaces.

We suppose that the commutator (6.4) is small in some sense in order to define the PCD and LSC preconditioners. The PCD Schur complement approximation is derived from a discretization of the commutator. In turn, to define LSC, the matrix F_p is replaced with a new matrix that makes (6.4) small in a least-squares sense.

Table 6.1 lists the matrices that are required to set up the different preconditioners that are introduced in the following sections; these matrices are usually further approximated via, for example, lumping or a Schwarz method.

The derivations in Sections 6.1.1 to 6.1.3 are taken from [82, Section 3]. Minor modifications to the notation have been made to improve consistency with the remainder of this dissertation.

6.1.1. PCD (Pressure Convection–Diffusion)

We present a derivation following [157] and [55, Chapter 9] of the pressure convection–diffusion preconditioner, which was introduced in [158, 104].

In the velocity space with zero boundary conditions, the matrix representation of the discrete divergence operator is $-M_p^{-1}B$, and $M_u^{-1}B^T$ is the representation of the discrete gradient operator [55, p. 409]. M_u is the velocity mass matrix and M_p the pressure mass matrix:

$$M_u = [m_{ij}^u], \quad m_{ij}^u = \int_{\Omega_f} \boldsymbol{\varphi}_i \cdot \boldsymbol{\varphi}_j \,\mathrm{d}\mathbf{x}, \quad M_p = [m_{ij}^p], \quad m_{ij}^p = \int_{\Omega_f} \psi_i \psi_j \,\mathrm{d}\mathbf{x}.$$

With these representations, we can define a discrete version of the commutator (6.4):

$$\mathcal{E}_h = (M_p^{-1}B)(M_u^{-1}F) - (M_p^{-1}F_p)(M_p^{-1}B). \tag{6.5}$$

Suppose that the commutators $\mathcal{E}$ and $\mathcal{E}_h$ are small in some operator norm (cf. [55, Remark 9.5] and Section 6.1.2). Pre-multiplication of (6.5) with $M_pF_p^{-1}M_p$, post-multiplication with $F^{-1}B^T$, and assuming the commutator is small lead to the Schur complement approximation

$$-BF^{-1}B^T \approx -M_pF_p^{-1}BM_u^{-1}B^T. \tag{6.6}$$

Note that F is not symmetric and that, in this work, the approximation sign is not meant to denote spectral equivalence, but that the difference is approximately zero in some matrix norm. We require the application of the inverse of the Schur complement, that is, also of its approximation $BM_u^{-1}B^T$. This expensive part is replaced with the pressure-Laplace operator A_p, which is spectrally equivalent for an enclosed-flow problem (cf. [55, p. 366] and [157]):

$$\begin{aligned} q_h^T A_p p_h = \langle \nabla p_h, \nabla q_h \rangle_{L_2} &\approx \langle M_u^{-1}B^T p_h, M_u^{-1}B^T q_h \rangle_{M_u} \\ &= \langle B^T p_h, M_u^{-1}B^T q_h \rangle_{l_2} \\ &= q_h^T B M_u^{-1} B^T p_h. \end{aligned}$$

This gives us

$$S = -BF^{-1}B^T \approx -M_pF_p^{-1}A_p =: S_{\mathrm{PCD}}. \tag{6.7}$$

Then, we can define the PCD block-triangular preconditioner

$$\mathcal{B}_{\mathrm{PCD}}^{-1} = \begin{bmatrix} F & B^T \\ 0 & S_{\mathrm{PCD}} \end{bmatrix}^{-1}.$$

As in [158] we can extend the operator F_p to also reflect the contribution of the BDF-2 time discretization for a transient Navier–Stokes problem:

$$F_p(\mathbf{w}_h) = \frac{3}{2\Delta t}M_p + N_p(\mathbf{w}_h) + \nu A_p. \tag{6.8}$$

The motivation is similar to before; the commutator is extended to

$$\mathcal{E} = \nabla \cdot \Big(\frac{\partial}{\partial t} - \nu\Delta + \mathbf{w}_h \cdot \nabla\Big) - \Big(\frac{\partial}{\partial t} - \nu\Delta + \mathbf{w}_h \cdot \nabla\Big)_p \nabla \cdot \tag{6.9}$$

to include a time derivative.

Note that the definition of S_{PCD} is based on the assumption of a small error between the operators $\mathcal{L}$ and $\mathcal{L}_p$. This is only directly applicable to the fixed-point method, as Newton's method introduces an additional term for the linearization. Consequently for Newton's method, we cannot necessarily assume the error as defined above to be small. Nonetheless, it is observed that Newton's method combined with PCD preconditioning yields good results; see for example [55, Chapters 9.3, 9.4].

Stabilized Discretizations

For elements which are not inf-sup stable, the PCD preconditioner also extends to stabilized approximations (e.g., P1–P1); see [55, Chapter 9]. A small commutator $\mathcal{E}$ (see (6.4)) implies

$$\nabla \cdot (-\nu\Delta + \mathbf{w}_h \cdot \nabla)^{-1}\nabla \approx (-\nu\Delta + \mathbf{w}_h \cdot \nabla)_p^{-1}\nabla \cdot \nabla. \tag{6.10}$$

Similarly to before, the left-hand side of (6.10) can be expressed discretely with

$$(-M_p^{-1}B)(M_u^{-1}F)^{-1}(M_u^{-1}B^T) = M_p^{-1}(-BF^{-1}B^T).$$

This is the scaled Schur complement matrix in the case of unstabilized mixed finite element approximations. If our discretization is stabilized, the Schur complement expands to $S = -BF^{-1}B^T - C$, such that we obtain

$$M_p^{-1}(-BF^{-1}B^T - C)$$

for the left-hand side of (6.10). For the right-hand side we get

$$(M_p^{-1}F_p)^{-1}(-M_p^{-1}B)(M_u^{-1}B^T) = F_p^{-1}(-BM_u^{-1}B^T).$$

Thus, for stabilized elements, we have

$$M_p^{-1}(-BF^{-1}B^T - C) \approx F_p^{-1}(-BM_u^{-1}B^T).$$

The matrix $-BM_u^{-1}B^T$ stems from the discretization of $\nabla\cdot\nabla$, but it is not a good representation of a discrete Laplacian in cases where the underlying mixed approximation is not uniformly stable [55, Chapter 9, p. 369]. Furthermore, for inherently unstable discretizations, highly oscillating components are in the null space of B^T, and, consequently, $BM_u^{-1}B^T$ is singular. However, we can fix this by using $-A_p$ as a more direct discretization of Δ, as described in the previous section. As a result, in the stabilized case, we obtain the Schur complement approximation $S \approx -M_pF_p^{-1}A_p = S_{\text{PCD}}$, which is the same as in the stable case; cf. [34, p. 351].

Boundary Conditions

The assembly of the original Schur complement S contains information about the boundary conditions. Similarly, we need to account for this information in the components of S_{PCD}. For a detailed discussion on this topic, we refer to [55, 57], where different boundary conditions are considered. Originally in [104], F_p and A_p were constructed with Neumann, i.e., do-nothing boundary conditions. Then, based on the analysis of a one- and two-dimensional problem, other boundary conditions were derived; see [55, 57].

The derivation of the preconditioner using homogeneous Neumann boundary conditions is based on enclosed-flow problems (e.g., lid-driven cavity). Consequently, we should expect it to perform better in these types of problems compared to nonenclosed-flow problems. In in- and outflow problems, A_p and F_p can be treated using different boundary conditions to improve performance. Note that A_p is not only used in the PCD Schur complement approximation $-M_pF_p^{-1}A_p$ but also for the assembly of $F_p = \frac{3}{2\Delta t}M_p + N_p + \nu A_p$. The setup of the two instances is based on

Table 6.2: Boundary conditions used in the PCD preconditioner. Different strategies for setting homogeneous boundary conditions in A_p and F_p denoted as D: Dirichlet, N: Neumann, and R: Robin. The default strategy (BC–3) is highlighted. Taken from [82, Table 3.2].

	A_p			F_p		
	$\partial\Omega_f^{\text{out}}$	$\partial\Omega_f^{\text{in}}$	$\partial\Omega_f^{\text{wall}}$	$\partial\Omega_f^{\text{out}}$	$\partial\Omega_f^{\text{in}}$	$\partial\Omega_f^{\text{wall}}$
(BC–1)	D	N	N	D	N	N
(BC–2)	D	N	N	D	R	N
(BC–3)	D	N	N	N	R	N

the same Neumann matrix corresponding to A_p but, subsequently, boundary conditions are set separately for A_p in S_{PCD} and for F_p. We will review a few results from [55, 57, 99]. The main idea is to construct F_p (6.8) and A_p as though they come from the corresponding boundary value problem. For the discrete Laplace operator A_p, forcing the discrete pressure p_h to satisfy a homogeneous Dirichlet boundary condition along the outflow boundary of $\partial\Omega_f^N$ is recommended; see [55]. For the $\partial\Omega_f^D = \partial\Omega_f \setminus \partial\Omega_f^N$ part of the boundary of the domain, a Neumann condition $\frac{\partial p_h}{\partial \mathbf{n}} = 0$ should be applied in A_p. For the discrete convection–diffusion operator F_p it is suggested that the Robin condition

$$-\nu \frac{\partial p_h}{\partial \mathbf{n}} + (\mathbf{w}_h \cdot \mathbf{n}) p_h = 0$$

for all boundary edges is appropriate; see, for example, [55, 57]. Note that this includes edges where Dirichlet velocity boundary conditions are applied, such as the inflow boundary or the walls. The use of Dirichlet conditions for F_p along the outlet $\partial\Omega_f^N$ can be beneficial. In [57, Section 7] different boundary conditions for F_p were discussed, and the use of a Robin boundary condition along the inlet and Dirichlet boundary conditions along the outlet showed good results for Reynolds numbers between 10 and 100. For higher Reynolds numbers in the range 200 to 400, results for Robin on the inlet and Neumann boundary conditions on the outlet proved best. In [35], using a similar implementation to ours, the use of scaled Dirichlet conditions was motivated. Other scaling options are mentioned in [57].

Since it is widely emphasized [55, 57, 99, 35] that the boundary conditions applied to PCD can have a substantial effect on the convergence (see also Table 10.11), we will consider the options in Table 6.2 for simulations but keep the recommended one (BC–3) as a default, where A_p has a Dirichlet condition on the outlet and F_p a Robin condition on the inlet.

6.1.2. LSC (Least-Squares Commutator)

The construction of the LSC preconditioner follows [52, 56] and [55, Section 9.2.3]. Similarly to before, we try to define a suitable approximation of the Schur complement by reducing the error of a commutator. Unlike in Section 6.1, however, we do not use the corresponding discrete commutator directly to define a Schur complement approximation with the help of a newly assembled matrix F_p. Instead, F_p is replaced by a matrix that does not require the assembly of an additional matrix (we want to avoid having to assemble the discrete convection–diffusion operator on the pressure) and that results in a small commutator with respect to some norm.

The commutator $\varepsilon = \nabla\cdot(-\nu\Delta + \mathbf{w}_h\cdot\nabla) - (-\nu\Delta + \mathbf{w}_h\cdot\nabla)_p\nabla\cdot$ from Section 6.1 can be regarded as the adjoint commutator to (cf. [55, Remark 9.3])

$$\varepsilon^* := (-\nu\Delta - \mathbf{w}_h\cdot\nabla)\nabla - \nabla(-\nu\Delta - \mathbf{w}_h\cdot\nabla)_p.$$

This time we base the approximation on the minimization of the commutator in the least-squares sense. Specifically, we seek to minimize the operator norm

$$\sup_{p_h\neq 0}\frac{\|\varepsilon^* p_h\|_{L^2(\Omega)}}{\|p_h\|_{L^2(\Omega)}}.$$

Similar to before, we can formulate this in a discrete sense and minimize

$$\sup_{p_h\neq 0}\frac{\|\varepsilon_h^* p_h\|_{M_u}}{\|p_h\|_{M_p}}, \quad \varepsilon_h^* = (M_u^{-1}F)(M_u^{-1}B^T) - (M_u^{-1}B^T)(M_p^{-1}F_p), \tag{6.11}$$

where in this context p_h corresponds to its coefficient vector, $\|\mathbf{v}\|_{M_u} = \langle M_u\mathbf{v},\mathbf{v}\rangle^{\frac{1}{2}}$, and $\|p_h\|_{M_p} = \langle M_p p_h, p_h\rangle^{\frac{1}{2}}$. Instead of using F_p as the discrete convection–diffusion operator on the pressure, we replace it with a matrix that yields a small (not necessarily minimal) value for (6.11). One way to achieve this is to minimize the individual vector norms of the columns of the discrete commutator, that is, by defining the jth column $[F_p]_j$ of F_p to solve the weighted least-squares problem

$$\min_{[F_p]_j}\left\|\left[M_u^{-1}FM_u^{-1}B^T\right]_j - M_u^{-1}B^T M_p^{-1}\,[F_p]_j\right\|_{M_u}.$$

The associated normal equations are

$$M_p^{-1}BM_u^{-1}B^T M_p^{-1}\,[F_p]_j = \left[M_p^{-1}BM_u^{-1}FM_u^{-1}B^T\right]_j,$$

which leads to the following definition of F_p:

$$F_p = M_p(BM_u^{-1}B^T)^{-1}(BM_u^{-1}FM_u^{-1}B^T). \tag{6.12}$$

Similarly to how we have derived the Schur complement approximation (6.6), we can pre-multiply ε_h^* with $BF^{-1}M_u$ and post-multiply it with $F_p^{-1}M_p$. Under the assumption that the commutator is small, we obtain

$$-BF^{-1}B^T \approx -BM_u^{-1}B^T F_p^{-1}M_p.$$

Substituting F_p from (6.12) into this Schur complement approximation gives

$$-BF^{-1}B^T \approx -\left(BM_u^{-1}B^T\right)\left(BM_u^{-1}FM_u^{-1}B^T\right)^{-1}\left(BM_u^{-1}B^T\right).$$

The approximate inverse of the Schur complement for the LSC block preconditioner for a stable discretization reads

$$S_{\mathrm{LSC}}^{-1} := -(BM_u^{-1}B^T)^{-1}(BM_u^{-1}FM_u^{-1}B^T)(BM_u^{-1}B^T)^{-1}. \tag{6.13}$$

We define the LSC block-triangular preconditioner

$$\mathcal{B}_{\mathrm{LSC}}^{-1} = \begin{bmatrix} F & B^T \\ 0 & S_{\mathrm{LSC}} \end{bmatrix}^{-1}.$$

If the mass matrix M_u is replaced with the identity matrix, the resulting Schur complement approximation,

$$S_{\mathrm{BFBt}}^{-1} = -(BB^T)^{-1}(BFB^T)(BB^T)^{-1}, \tag{6.14}$$

is the classical BFBt preconditioner [56], whereas the LSC preconditioner is a scaled BFBt method.

As for the construction of the PCD preconditioner, $BM_u^{-1}B^T$ can be replaced with the pressure-Laplace matrix:

$$S_{\mathrm{LSC}_{A_p}}^{-1} = -A_p^{-1}(BM_u^{-1}FM_u^{-1}B^T)A_p^{-1}. \tag{6.15}$$

We have chosen to use this replacement strategy in our simulations; cf. Section 8.6.2. As before, A_p can be assembled using different boundary conditions [55, Sect. 9.2.2]; see also [57]. In this work, we will only use (BC–2) for LSC variants, that is, a Dirichlet condition at the outflow of the domain.

Stabilization For not inf-sup stable discretizations, the LSC approach requires some modification. As mentioned in Section 6.1.1, the approximation $BM_u^{-1}B^T$ of the discrete Laplacian is singular due to the nontrivial null space of B^T. In [53, Sect. 4.2] (see also [34, Sect. 3.3.3] and [51, Sect. 3.2.2]) a modification was proposed by adding scaled stabilization operators:

$$S_{\mathrm{LSC}_{\mathrm{stab}}}^{-1} = -(BM_u^{-1}B^T + \gamma C)^{-1}(BM_u^{-1}FM_u^{-1}B^T)(BM_u^{-1}B^T + \gamma C)^{-1} - \alpha D^{-1},$$

where, using $\mathrm{diag}(\cdot)$ for the diagonal part and $\rho_{\mathrm{eig}}(\cdot)$ for the spectral radius of a matrix,

$$\gamma := \frac{\rho_{\mathrm{eig}}\left(M_u^{-1}F\right)}{3},$$
$$D := \mathrm{diag}\left(\hat{S}_{\mathrm{diag}}\right), \qquad \hat{S}_{\mathrm{diag}} := -B\,\mathrm{diag}(F)^{-1}B^T - C,$$
$$\alpha := \frac{-1}{\rho_{\mathrm{eig}}\left(\hat{S}_{\mathrm{diag},C=0}D^{-1}\right)}, \qquad \hat{S}_{\mathrm{diag},C=0} := -B\,\mathrm{diag}(F)^{-1}B^T.$$

As in [34, Sect. 3.3.3], we replace $BM_u^{-1}B^T + \gamma C$ with A_p with no special consideration given to the stabilization matrix C:

$$S_{\mathrm{LSC}_{\mathrm{stab},A_p}}^{-1} = -A_p^{-1}(BM_u^{-1}FM_u^{-1}B^T)A_p^{-1} - \alpha D^{-1}.$$

Inverse of Velocity Mass Matrix Similar to a strategy applied in the construction of the SIMPLE preconditioner (see Section 6.1.3), we will approximate the inverse of M_u with one of the following two matrices: Let δ^{ij} denote the Kronecker delta; then

$$H_{M_u}^D := \mathrm{diag}(M_u)^{-1}, \qquad H_{M_u}^{\Sigma} := \delta^{ij}\Big(\sum_{k=1}^{N_u} |(M_u)_{i,k}|\Big)^{-1}.$$

6.1.3. SIMPLE

The SIMPLE (Semi-Implicit Method for Pressure-Linked Equations) preconditioner was originally introduced by Patankar and Spalding in [139] to solve the Navier–Stokes equations, and the SIMPLE block preconditioner is based on it; see also [147, 51].

First, the lower-triangular and the block-diagonal components of the LDU decomposition (6.2) are grouped together in an LU decomposition:

$$\begin{bmatrix} F & 0 \\ B & S \end{bmatrix} \begin{bmatrix} I & F^{-1}B^T \\ 0 & I \end{bmatrix}.$$

Then, the inverse of F in the upper right block is replaced by an approximation H_F, which differs depending on whether SIMPLE or SIMPLEC is used; a SIMPLE variant that is not covered in this work is SIMPLER [127, 47]. The Schur complement S is approximated with $S_{\text{SIMPLE}} = -C - BH_FB^T$. The introduction of an under-relaxation parameter α completes the construction of the preconditioner $\mathcal{B}^{-1}_{\text{SIMPLE}}$:

$$\mathcal{B}_{\text{SIMPLE}} = \begin{bmatrix} F & 0 \\ B & S_{\text{SIMPLE}} \end{bmatrix} \begin{bmatrix} I & \frac{1}{\alpha}H_FB^T \\ 0 & \frac{1}{\alpha}I \end{bmatrix}.$$

H_F is a diagonal matrix and depends on the specific SIMPLE variant. We denote the matrix of the default variant SIMPLE as H_F^D and the one of SIMPLEC as H_F^Σ:

$$H_F^D = \operatorname{diag}(F)^{-1}, \tag{SIMPLE}$$

$$H_F^\Sigma = \delta^{ij}\Big(\sum_{k=1}^{N_u} |F_{i,k}|\Big)^{-1}. \tag{SIMPLEC}$$

If we expand F into its parts

$$F = \frac{3}{2\Delta t}M_u + \nu A + N + W,$$

with velocity mass matrix M_u, discrete Laplacian $-A$, and advective terms $N + W$, the well-conditioned mass matrix dominates F for small time steps Δt. Since M_u is spectrally equivalent to the lumped mass matrix $\operatorname{diag}(M_u)$, approximating F via its diagonal is sufficient for small Δt. The performance of SIMPLE and SIMPLEC can deteriorate for stationary flow problems if Δt is large or if advective forces dominate.

6.1.4. Augmented Lagrangian

The augmented Lagrangian (AL) preconditioner was introduced in [17] as a block preconditioner for saddle-point systems arising from the two-dimensional stationary incompressible Navier–Stokes problem. We consider the finite element discretization from (2.2), resulting in the discretized generic saddle point system in (2.16). The equivalent augmented Lagrangian formulation is given by

$$\begin{bmatrix} F + \gamma B^TW^{-1}B & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u}_h \\ p_h \end{bmatrix} = \begin{bmatrix} \mathbf{f} + \gamma B^TW^{-1}g \\ g \end{bmatrix}. \tag{6.16}$$

We consider the underlying finite element discretization to be stable, which results in a zero matrix in the lower right block entry. The matrix W is symmetric positive definite. In [17, Section 4] it is shown that a good choice for W is the pressure mass matrix M_p. In practice we do not use the inverse of M_p but use the inverse of its main diagonal. In the following, we consider $W = M_p$, as this choice for W is used in this work. The parameter $\gamma > 0$ can be determined experimentally or analytically, e.g., see [19] for an analytical approach using Fourier Analysis. The AL system in (2.7) delivers the same results as the original system in (2.2). In particular, if $B\mathbf{u}_h = 0$ is fulfilled (divergence-free condition) for the computed solution $\mathbf{u}_h$, then the additional term on the right hand side vanishes due to $g = 0$. Simultaneously, the additional term in the velocity block vanishes since $B\mathbf{u}_h = 0$ holds. The Schur complement of (6.16) is given by

$$S = -B(F + \gamma B^T M_p^{-1} B)^{-1} B^T. \tag{6.17}$$

In the following, the Schur complement for (6.20) is derived using the Sherman–Morrison–Woodbury formula [70, Section 2.1.4],

$$(A + UV^T)^{-1} = A^{-1} - A^{-1}U(I + V^T A^{-1} U)^{-1} V^T A^{-1},$$

with $A \in \mathbb{R}^{n\times n}$, and $U, V \in \mathbb{R}^{n\times k}$. First we apply the formula to $(F + \gamma B^T M_p^{-1} B)^{-1}$:

$$(F + \gamma B^T M_p^{-1} B)^{-1} = F^{-1} - F^{-1} B^T (\gamma^{-1} M_p + B F^{-1} B^T)^{-1} B F^{-1}.$$

This is inserted into (6.17), which yields

$$S = -BF^{-1}B^T + BF^{-1}B^T(\gamma^{-1}M_p + BF^{-1}B^T)^{-1}BF^{-1}B^T.$$

The Sherman-Morrison-Woodbury formula can be applied again, which yields

$$-BF^{-1}B^T + BF^{-1}B^T(\gamma^{-1}M_p + BF^{-1}B^T)^{-1}BF^{-1}B^T = -((BF^{-1}B^T)^{-1} + \gamma M_p^{-1})^{-1}.$$

Consequently, the resulting S^{-1} for the augmented Lagrangian formulation is defined as

$$S^{-1} = -(BF^{-1}B^T)^{-1} - \gamma M_p^{-1}. \tag{6.18}$$

In the stationary case, the inverse of the Schur complement is approximated by

$$\hat{S}_{\mathrm{AL}}^{-1} = -(\nu + \gamma) M_p^{-1}. \tag{6.19}$$

We refer to, e.g., [17, Section 4], for a detailed analysis of the preconditioner and derivation of the Schur complement approximation. The preconditioner for the velocity block $A_\gamma = F + \gamma B^T M_p^{-1} B$ is defined as

$$\hat{A}_{\gamma,\mathrm{AL}}^{-1} \approx (F + \gamma B^T M_p^{-1} B)^{-1}.$$

Based on the AL formulation (6.16), we define the augmented Lagrangian formulation for the linearized Navier–Stokes system from (2.11)

$$\begin{bmatrix} F_N(\mathbf{u}_h^k) + \gamma B^T M_p^{-1} B & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} \delta\mathbf{u}_h^k \\ \delta p_h^k \end{bmatrix} = - \begin{bmatrix} r_{\mathbf{u}_h}^k + \gamma B^T M_p^{-1} r_{p_h}^k \\ r_{p_h}^k \end{bmatrix}. \tag{6.20}$$

This modified system has the same discrete solution as (2.11).

Modified Augmented Lagrangian

The modified AL preconditioner was introduced in [18]. It was observed that the main cost of applying the preconditioner was associated with the velocity block $A_\gamma = F + \gamma B^T W^{-1} B$. We refer to Section 8.5.1 and Section 10.5.2 and the discussions therein concerning the sparsity pattern of A_γ. Therefore in [18], the modified augmented Lagrangian (mod AL) was constructed in two dimensions.

The derivation and description follow [20], where the three-dimensional version of the modified augmented Lagrangian preconditioner was considered. In the following we describe the construction based on a three-dimensional problem. Suppose we have $F = diag(F_1, F_2, F_3)$ where F_i corresponds to the degrees of freedom per node. Each block $F_i, i = 1, 2, 3$ is square and of order $n/3$, and $B = (B_1, B_2, B_3)$. Then we can write it by its components

$$\begin{aligned} A_\gamma &= F + \gamma B^T M_p^{-1} B \\ &= \begin{pmatrix} F_1 & 0 & 0 \\ 0 & F_2 & 0 \\ 0 & 0 & F_3 \end{pmatrix} + \gamma \begin{pmatrix} B_1^T \\ B_2^T \\ B_3^T \end{pmatrix} M_p^{-1} \begin{pmatrix} B_1 & B_2 & B_3 \end{pmatrix} \\ &= \begin{pmatrix} F_1 + \gamma B_1^T M_p^{-1} B_1 & \gamma B_1^T M_p^{-1} B_2 & \gamma B_1^T M_p^{-1} B_3 \\ \gamma B_2^T M_p^{-1} B_1 & F_2 + \gamma B_2^T M_p^{-1} B_2 & \gamma B_2^T M_p^{-1} B_3 \\ \gamma B_3^T M_p^{-1} B_1 & \gamma B_3^T M_p^{-1} B_2 & F_3 + \gamma B_3^T M_p^{-1} B_3 \end{pmatrix} \\ &=: \begin{pmatrix} A_{\gamma,11} & A_{\gamma,12} & A_{\gamma,13} \\ A_{\gamma,21} & A_{\gamma,22} & A_{\gamma,23} \\ A_{\gamma,31} & A_{\gamma,32} & A_{\gamma,33} \end{pmatrix}. \end{aligned} \tag{6.21}$$

Instead of approximating the inverse of A_γ, only the upper-triangular part is used

$$\tilde{A}_\gamma = \begin{pmatrix} A_{\gamma,11} & A_{\gamma,12} & A_{\gamma,13} \\ 0 & A_{\gamma,22} & A_{\gamma,23} \\ 0 & 0 & A_{\gamma,33} \end{pmatrix}, \tag{6.22}$$

decreasing the cost of preconditioner application. The modified AL preconditioner can then be defined by the block-triangular matrix

$$\hat{\mathcal{B}}_{\text{mod-AL}} = \begin{pmatrix} \hat{A}_{\gamma,\text{mod-AL}} & B^T \\ 0 & \hat{S}_{\text{AL}} \end{pmatrix}. \tag{6.23}$$

6.2. FaCSI Block Preconditioner for the FSI Problem

FSI problems can generally be solved using either partitioned or monolithic coupling schemes. In partitioned approaches, the fluid and solid subproblems are solved separately and coupled through suitable interface conditions, whereas monolithic approaches solve the coupled fluid–structure system simultaneously. A recent review and comparison of both approaches is given in [29].

Partitioned methods range from loosely coupled fixed-point schemes to strongly coupled iterative algorithms that repeatedly solve the fluid and solid subproblems until the interface conditions are

satisfied. Likewise, monolithic approaches comprise a broad spectrum of methods, ranging from block preconditioners based on segregated fluid and solid solvers to fully coupled formulations equipped with dedicated monolithic preconditioners. The first staggered FSI algorithms were introduced in [62], while one of the early monolithic formulations was proposed by [184]. Since then, considerable effort has been devoted to the development of robust and efficient solvers for monolithic FSI systems. Examples include monolithic algebraic multigrid methods [68], geometric multigrid approaches [151], and block preconditioners based on approximate block factorizations [124, 123].

In the setting of biomechanics, where blood interacts with comparatively soft arterial tissue and the fluid and structure possess similar densities, partitioned approaches may suffer from the added-mass effect [31]. Under these conditions, monolithic coupling schemes are more flexible and robust; see the comparison in [29]. Consequently, many state-of-the-art solvers for FSI in biomechanics make use of monolithic couplings [119, 101, 36, 118, 117, 151, 173, 150, 33, 137, 167]. In this work, the FaCSI preconditioning strategy introduced in [36] is used. The acronym *FaCSI* refers to the Factorized fluid–structure interaction block system, the use of static Condensation, and the application of the SIMPLE preconditioner for the fluid subsystem. The resulting block factorization separates the coupled problem into fluid, structure, and geometry subproblems, allowing the corresponding inverses to be approximated independently. The inverses arising in the FaCSI factorization are approximated by two-level overlapping Schwarz preconditioners; see Chapter 7.

In [84, 85] two-level overlapping Schwarz methods were considered for the solid component in FSI, comparing a linear, a neo-Hookean, and a hyperelastic anisotropic material model. The fluid component in the FaCSI block preconditioner is by definition preconditioned by the SIMPLE block preconditioner. In this work, a monolithic preconditioner for the fluid will additionally be considered and compared; see also [96] for monolithic two-level Schwarz methods for the fluid block. In [8, 10], one-level overlapping Schwarz methods were employed to approximate the arising inverses. Similarly, in [181] restricted additive one-level Schwarz preconditioners were considered for the different subproblems. Monolithic overlapping Schwarz preconditioners for the full FSI system were for example considered in [13, 180].

The following derivation of the FaCSI block preconditioner follows [36] and [96]. The preconditioner is based on an incomplete block factorization of the Jacobian of the FSI system. The starting point is the block structure of the Jacobian of the FSI system from (5.6). First, we reordered the system, such that the solid subproblem $\mathcal{S}$ is in block (0,0), the geometry subproblem $\mathcal{G}$ is in block (1,1), and the fluid subproblem $\mathcal{F}$ is in block (2,2). The coupling blocks denoted by $C_i, i = 1, ..., 5$ and the shape derivatives denoted by $\mathcal{D}$ occupy the off-diagonal blocks of the system

$$\mathcal{J} = \begin{pmatrix} \mathcal{S} & 0 & 0 & C_4 \\ C_5 & \mathcal{G} & 0 & 0 \\ 0 & \mathcal{D} & \mathcal{F} & C_3 \\ C_2 & 0 & C_1 & 0 \end{pmatrix}.$$

In the following step, the different physical problems are decoupled and as such define the

components of the preconditioner

$$\mathcal{B}_{\text{FaCSI}} = \begin{pmatrix} \mathcal{S} & 0 & 0 & C_4 \\ C_5 & \mathcal{G} & 0 & 0 \\ 0 & \mathcal{D} & \mathcal{F} & C_3 \\ C_2 & 0 & C_1 & 0 \end{pmatrix} = \underbrace{\begin{pmatrix} \mathcal{S} & 0 & 0 & 0 \\ 0 & I_\mathcal{G} & 0 & 0 \\ 0 & 0 & I_\mathcal{F} & 0 \\ 0 & 0 & 0 & I_\Gamma \end{pmatrix}}_{\mathcal{B}_\mathcal{S}} \underbrace{\begin{pmatrix} I_\mathcal{S} & 0 & 0 & 0 \\ C_5 & \mathcal{G} & 0 & 0 \\ 0 & 0 & I_\mathcal{F} & 0 \\ 0 & 0 & 0 & I_\Gamma \end{pmatrix}}_{\mathcal{B}_\mathcal{G}} \underbrace{\begin{pmatrix} I_\mathcal{S} & 0 & 0 & 0 \\ 0 & I_\mathcal{G} & 0 & 0 \\ 0 & \mathcal{D} & \mathcal{F} & C_3 \\ C_2 & 0 & C_1 & 0 \end{pmatrix}}_{\mathcal{B}_\mathcal{F}}. \tag{6.24}$$

Note that the C_4 block in the upper part is neglected in the construction of the preconditioner. An approximation of the inverse of $\mathcal{B}_\mathcal{S}$ can be computed, since it already has a block-diagonal structure. The block components $\mathcal{B}_\mathcal{G}$ and $\mathcal{B}_\mathcal{F}$ require further factorization. For the geometry problem this yields

$$\mathcal{B}_\mathcal{G} = \begin{pmatrix} I_\mathcal{S} & 0 & 0 & 0 \\ C_5 & I_\mathcal{G} & 0 & 0 \\ 0 & 0 & I_\mathcal{F} & 0 \\ 0 & 0 & 0 & I_\Gamma \end{pmatrix} \begin{pmatrix} I_\mathcal{S} & 0 & 0 & 0 \\ 0 & \mathcal{G} & 0 & 0 \\ 0 & 0 & I_\mathcal{F} & 0 \\ 0 & 0 & 0 & I_\Gamma \end{pmatrix} = \mathcal{B}_{\mathcal{G}_1} \mathcal{B}_{\mathcal{G}_2},$$

where the inverse of $\mathcal{B}_{\mathcal{G}_2}$ can be approximated similarly to $\mathcal{B}_\mathcal{S}$. The fluid component $\mathcal{B}_\mathcal{F}$ is factorized as

$$\mathcal{B}_\mathcal{F} = \underbrace{\begin{pmatrix} I_\mathcal{S} & 0 & 0 & 0 \\ 0 & I_\mathcal{G} & 0 & 0 \\ 0 & \mathcal{D} & I_\mathcal{F} & 0 \\ 0 & 0 & 0 & I_\Gamma \end{pmatrix}}_{\mathcal{B}_{\mathcal{F}_1}} \underbrace{\begin{pmatrix} I_\mathcal{S} & 0 & 0 & 0 \\ 0 & I_\mathcal{G} & 0 & 0 \\ 0 & 0 & I_\mathcal{F} & 0 \\ C_2 & 0 & 0 & I_\Gamma \end{pmatrix}}_{\mathcal{B}_{\mathcal{F}_2}} \underbrace{\begin{pmatrix} I_\mathcal{S} & 0 & 0 & 0 \\ 0 & I_\mathcal{G} & 0 & 0 \\ 0 & 0 & \mathcal{F} & C_3 \\ 0 & 0 & C_1 & 0 \end{pmatrix}}_{\mathcal{B}_{\mathcal{F}_3}}.$$

The factors $\mathcal{B}_{\mathcal{F}_1}$ and $\mathcal{B}_{\mathcal{F}_2}$ can be exactly inverted. Now, the remaining 2×2 block system in $\mathcal{B}_{\mathcal{F}_3}$ needs to be solved. This block is defined in the following

$$\begin{pmatrix} \mathcal{F} & C_3 \\ C_1 & 0 \end{pmatrix} \begin{pmatrix} x \\ \lambda \end{pmatrix} = \begin{pmatrix} r_\mathcal{F} \\ r_\lambda \end{pmatrix}, \tag{6.25}$$

with residual vectors $r_\mathcal{F}$ and r_λ. We can split the system into inner and interface degrees of freedom:

$$\begin{pmatrix} \mathcal{F}_{II} & \mathcal{F}_{I\Gamma} & 0 \\ \mathcal{F}_{\Gamma I} & \mathcal{F}_{\Gamma\Gamma} & I \\ 0 & I & 0 \end{pmatrix} \begin{pmatrix} x_I \\ x_\Gamma \\ \lambda \end{pmatrix} = \begin{pmatrix} r_{\mathcal{F},I} \\ r_{\mathcal{F},\Gamma} \\ r_\lambda \end{pmatrix},$$

with C_1 restricting the fluid degrees of freedom, e.g., the velocity and pressure degrees of freedom to the velocity interface degrees of freedom, and $C_3 = C_1^T$ doing the opposite. We can write down the equations for the fluid subproblem as

$$\mathcal{F}_{II} x_I + \mathcal{F}_{I\Gamma} x_\Gamma = r_{\mathcal{F},I}, \tag{6.26}$$

$$\mathcal{F}_{\Gamma I} x_I + \mathcal{F}_{\Gamma\Gamma} x_\Gamma + \lambda = r_{\mathcal{F},\Gamma}, \tag{6.27}$$

$$x_\Gamma = r_\lambda. \tag{6.28}$$

We immediately get the solution of $x_\Gamma = r_\lambda$ from (6.28). From (6.26) we get the condensed system for the fluid subproblem

$$\mathcal{F}_{II} x_I = r_{\mathcal{F},I} - \mathcal{F}_{I\Gamma} r_\lambda, \tag{6.29}$$

which needs to be solved. The inverse of $\mathcal{F}_{II}$ in (6.29) can now be approximated with a monolithic or block preconditioner. Finally, we can evaluate λ with (6.27)

$$\lambda = r_{\mathcal{F},\Gamma} - \mathcal{F}_{\Gamma I} x_I - \mathcal{F}_{\Gamma\Gamma} r_\lambda.$$

In practice, we would not construct the preconditioner specifically for $\mathcal{F}_{II}$ but for $\mathcal{F}$. Then, in $\mathcal{F}$ we would set all interface nodes to Dirichlet nodes. Then, we replace the interface values x_Γ with r_λ, and solve the following equation for x_I:

$$\begin{pmatrix} \mathcal{F}_{II} & \mathcal{F}_{I\Gamma} \\ 0 & I \end{pmatrix} \begin{pmatrix} x_I \\ x_\Gamma \end{pmatrix} = \begin{pmatrix} r_{\mathcal{F},I} \\ r_\lambda \end{pmatrix}.$$

Then we use x_I to recover λ in

$$\begin{aligned} \mathcal{F}x + C_3\lambda &= r_{\mathcal{F}} \\ \lambda &= C_1(r_{\mathcal{F}} - \mathcal{F}x) \end{aligned}$$

The complete FaCSI algorithm with respect to the implementation and application of the algorithm can be found in Section 8.7 and Algorithm 1. The additional FSCI coupling is integrated into the algorithm; see also Section 5.1.

7. Domain Decomposition: Overlapping Schwarz Methods

The nonlinear systems arising from the considered finite element formulations are solved using a Newton–Krylov approach. More precisely, the linearized systems arising within Newton's method are solved iteratively using Krylov subspace methods; see Section 10.1.1. Throughout this work, the preconditioned Generalized Minimal Residual (GMRES) method [155] is employed, since the arising Jacobian matrices, for example, from the Navier–Stokes equations (2.16), are generally sparse, nonsymmetric, and indefinite. Several block preconditioning strategies for the Navier–Stokes saddle-point problem were introduced in Section 6.1, while Section 6.2 presented the FaCSI preconditioner for the coupled fluid–structure interaction problem. Both approaches require efficient approximations of the inverse operators associated with the individual subproblems.

Among the parallel preconditioning techniques for large-scale finite element problems are domain decomposition methods. These can be broadly classified into overlapping and nonoverlapping approaches. In this work, we focus on overlapping Schwarz methods [159], which are based on an overlapping decomposition of the computational domain into subdomains. Nonoverlapping domain decomposition methods include the Finite Element Tearing and Interconnecting (FETI) method [58, 59], its extension FETI–DP (Dual–Primal) [115, 130, 111, 112, 113], the Balancing Domain Decomposition (BDD) method [163], and the Balancing Domain Decomposition by Constraints (BDDC) method [32, 38, 129].

While one-level overlapping Schwarz methods can be effective, their convergence generally deteriorates as the number of subdomains increases. Consequently, scalable parallel algorithms require a second, global coarse level that communicates global information across the computational domain and yields convergence rates that are largely independent of the number of subdomains. The construction of an effective coarse space is therefore one of the key ingredients of scalable overlapping Schwarz methods.

In this work, Generalized Dryja–Smith–Widlund (GDSW)-type coarse spaces [40, 41] are considered. Several approaches have been proposed to reduce the coarse-space dimension of GDSW methods while preserving robustness and improving parallel scalability [43, 44, 45]. These include the reduced-dimension GDSW (RGDSW) coarse space [45]. Furthermore, GDSW*, first introduced in [96] and further developed in [82], represents an intermediate approach between GDSW and RGDSW in three dimensions. The parallel scalability could be further improved by solving the coarse problem inexactly using another GDSW-type preconditioner resulting in a three-level GDSW method; cf. [90, 87, 122]. For highly heterogeneous multiscale problems, adaptive coarse spaces based on local generalized eigenvalue problems have been proposed [116, 81], resulting in methods that are robust with respect to large coefficient contrasts. These methods significantly increase the computational cost and are so far only fully robust for

a small set of problems (e.g., diffusion or linear elasticity).

Mixed finite element discretizations of the incompressible Navier–Stokes equations give rise to linear systems with saddle-point structure, where the unknowns consist of velocity and pressure variables. Comprehensive reviews of numerical methods for saddle-point problems and, in particular, finite element discretizations, iterative solvers, and preconditioners for the incompressible Navier–Stokes equations are given in [16, 55].

One possibility is to construct a monolithic overlapping Schwarz preconditioner for the complete saddle-point system. This approach has been introduced in [109, 110] with Lagrangian coarse spaces and extended to GDSW-type coarse spaces in [77, 78]. Recently, the performance of monolithic overlapping Schwarz preconditioners has been compared with block preconditioning strategies for incompressible fluid flow in [82], building on the developments in [77, 78, 96]. The considered overlapping Schwarz preconditioners with GDSW-type coarse spaces are implemented in the `FROSch` package of the `Trilinos` framework; see Section 8.5.

The application of monolithic RGDSW coarse spaces in nonlinear Schwarz preconditioners has recently been investigated in [108, 76, 95]. Beyond incompressible fluid flow, monolithic overlapping Schwarz preconditioners have also been successfully applied to other coupled multiphysics problems, including chemo-mechanical models [106, 105] and laser beam welding simulations [24, 23, 22].

The use of overlapping Schwarz preconditioners for solid mechanics is well established. The robustness and parallel scalability of two-level overlapping Schwarz methods based on GDSW and RGDSW coarse spaces were shown for linear elasticity problems in [84, 86, 79]. In the curved artery benchmark introduced in [8] and later revisited in [10], algebraic one-level Schwarz preconditioners were employed for the FSI subproblems. Furthermore, two-level overlapping Schwarz methods were applied to the structural component of FSI problems in [84, 85], where their performance was investigated for linear elastic, neo-Hookean, and an anisotropic hyperelastic material models.

In this chapter, the two-level overlapping Schwarz preconditioners are introduced based on a simple model problem. In Section 7.1, the GDSW-Type coarse spaces are defined. The extension to block and monolithic preconditioners is derived in Sections 7.2 and 7.3, respectively. Different aspects of blockwise and monolithically constructed basis functions for the Navier–Stokes equations are discussed in Section 7.4. The Sections 7.1 to 7.3 are largely based on [82, Chapter 4].

Overlapping Schwarz Preconditioners The overlapping Schwarz preconditioners are introduced under simplified assumptions, that is, for a linear system

$$Kx = b,$$

resulting from the discretization of a Laplacian model problem on the computational domain Ω using, e.g., piecewise linear finite elements with the finite element space $V^h = V^h(\Omega)$.

In this case, the matrix K is symmetric positive definite, and theoretical results for the convergence are available. We refer to [159, 163] for a more detailed introduction. The following construction is essentially the same for general problems, but is then not based on a robust

theoretical background. Details of the construction to consider for monolithic preconditioning of multiphysics problems will be given in Section 7.3.

Let Ω be decomposed into nonoverlapping subdomains $\{\Omega_i\}_{i=1}^N$ that are conforming with the finite element discretization; that is, the closure of every subdomain is the union of its finite elements. We extend the subdomains by k layers of finite elements, resulting in an overlapping domain decomposition $\Omega = \{\Omega_i'\}_{i=1}^N$ with overlap $\delta = kh$. Note that extending the subdomains by layers of elements is a geometric operation based on the mesh. To only use information available by the matrix K, the subdomains can be algebraically extended by iteratively adding those degrees of freedom that are in the neighborhood of the (overlapping) subdomain, defining neighborhood via the sparsity pattern of K; cf. [83, 79]. For example, if Ω_i is extended by one layer, for all degrees of freedom l in Ω_i, degrees of freedom j for which $K_{l,j} \neq 0$ are added to Ω_i to obtain Ω_i'. This algebraic overlap construction is used in our implementation, which is detailed in Section 8.5.1. We denote the size of the algebraically determined overlap by $\hat{\delta} = k$ for k iterations of the overlap-constructing algorithm; see Section 8.5.1 for details. In this work, we always use $\hat{\delta} = 1$ which corresponds to a geometric overlap of $\delta = 2h$. For all other methods, the initial decomposition corresponding to $\hat{\delta} = 0$ is given by the nonoverlapping domain decomposition. Note that, for the augmented Lagrangian block preconditioner (Section 6.1.4), we construct the overlap differently; this is illustrated in Figure 8.7.

Based on the overlapping domain decomposition, we define a restriction operator $R_i : V^h \to V_i^h$, $i = 1, \dots, N$, to map from the global finite element space V^h to the local finite element space $V_i^h := V^h(\Omega_i')$ on the overlapping subdomains Ω_i'. Moreover, we define the corresponding prolongation operator $R_i^T : V_i^h \to V^h$, which is the transpose of the restriction operator and extends a local function on Ω_i' by zero outside of Ω_i'.

We define a second prolongation $\widetilde{R}_i^T$: Standard additive Schwarz is given by $\widetilde{R}_i^T = R_i^T$. Restricted additive Schwarz [30] uses R_i^T but sets values to zero such that $\sum_{i=1}^N \widetilde{R}_i^T R_i$ is the identity operator. In this work, we define $\widetilde{R}_i^T$ to satisfy the identity-operator property by an inverse multiplicity: A row of R_i^T corresponding to the finite element node x^h is scaled with the inverse of $|\{j \in \{1, \dots, N\} : x^h \in \Omega_j' \setminus \partial\Omega_j'\}|$, where $|\cdot|$ is the cardinality of the set; cf. [30, Remark 2.7] and [78, Sect. 4.1]. Using the operators R_i, R_i^T, and $\widetilde{R}_i^T$, and defining local overlapping stiffness matrices by $K_i := R_i K R_i^T$, $i = 1, \dots, N$, the additive one-level overlapping Schwarz preconditioner is given by

$$M_{\text{OS-1}}^{-1} = \sum_{i=1}^{N} \widetilde{R}_i^T K_i^{-1} R_i. \tag{7.1}$$

Generally, the one-level Schwarz preconditioner (7.1) is not numerically scalable: the number of Krylov iterations will increase with an increasing number of subdomains. In order to define a numerically scalable overlapping Schwarz preconditioner, which yields a convergence rate independent of the number of subdomains, a coarse level can be introduced. With a coarse interpolation operator $\Phi : V_0 \to V^h$, which is still to be specified, the additive two-level overlapping

Schwarz preconditioner reads

$$M_{\text{OS-2}}^{-1} = \Phi K_0^{-1} \Phi^T + \underbrace{\sum_{i=1}^{N} \widetilde{R}_i^T K_i^{-1} R_i}_{=M_{\text{OS-1}}^{-1}}, \tag{7.2}$$

where $K_0 = \Phi^T K \Phi$ is the coarse matrix and K_0^{-1} is the coarse problem to be solved. The columns of Φ form the basis of the coarse space, V_0, and K_0 is a Galerkin projection of K into the coarse space. A natural choice for coarse basis functions are Lagrangian basis functions defined on a coarse triangulation, which yields a scalable preconditioner. However, the definition of a coarse triangulation heavily depends on geometric information and cannot be easily done in a conforming way for complex geometries of Ω. In the next section, the GDSW-type coarse spaces are discussed, which can be constructed algebraically.

7.1. GDSW-Type Coarse Spaces

For this work, we choose a very flexible approach for the construction of a coarse space that was introduced with the Generalized Dryja–Smith–Widlund (GDSW) preconditioner in [40, 41]. The approach allows us to define many different variants, three of which (GDSW [40, 41], RGDSW [45], GDSW*[96, 82]) we introduce and use for simulations. It requires little to no information in addition to the system matrix. In this approach, the nonoverlapping subdomains $\{\Omega_i\}_{i=1}^N$ are chosen as the elements of the coarse triangulation, and the coarse basis functions are defined via extension of trace functions defined on the interface

$$\Gamma = \Big\{ x \in \Big(\Omega_i \cap \Omega_j \Big) \setminus \partial\Omega_D : i \neq j, 1 \leq i, j \leq N \Big\},$$

where $\partial\Omega_D$ is the global Dirichlet boundary; note the comment in Section 7.1.3 for discontinuous finite elements, in which case the interface does not contain any degrees of freedom. More precisely, first, a partition of unity is defined on Γ. The different variants of the coarse spaces differ only in the choice of the partition of unity on the interface. Then the functions are extended into the interior of the subdomains. The extension is defined algebraically using submatrices of K and corresponds to a discrete harmonic extension in the case of a Laplace problem. In this specific case, the extension of a partition of unity on the interface gives a partition of unity on the whole domain Ω. Given interface values Φ_Γ, the full matrix Φ is obtained by solving for all degrees of freedom I that do not correspond to the interface Γ:

$$\Phi = \begin{bmatrix} \Phi_I \\ \Phi_\Gamma \end{bmatrix} = \begin{bmatrix} -K_{II}^{-1} K_{I\Gamma} \Phi_\Gamma \\ \Phi_\Gamma \end{bmatrix}. \tag{7.3}$$

The blocks K_{II} and $K_{I\Gamma}$ are obtained by reordering and partitioning K as follows:

$$K = \begin{bmatrix} K_{II} & K_{I\Gamma} \\ K_{\Gamma I} & K_{\Gamma\Gamma} \end{bmatrix}.$$

Note that K_{II} is block-diagonal, and hence the solver required for K_{II}^{-1} can be dealt with independently for each subdomain. Moreover, as only submatrices of K are required, the

Table 7.1: Coarse space dimensions for the scalar Laplace problem and the different GDSW-type coarse spaces used in this work. Vertices, edges, and faces refer to the respective interface components of the domain decomposition.

Coarse Space	Coarse Space Dimension
GDSW	number of vertices + faces + edges
GDSW*	number of vertices + faces
RGDSW	number of vertices

construction of the coarse space is algebraic. For other problems, like elasticity, additional geometric information may be required; cf., e.g., [42, sect. 4]. The description above outlines the construction for a scalar problem; the extension to vector-valued problems is given in Section 7.1.2.

The coarse spaces used in this work are all based on GDSW-type preconditioners and on extending the interface values into the interior via (7.3); they only differ in the definition of the interface partition of unity Φ_Γ. We consider three coarse space variants: GDSW [40, 41] and reduced-dimension GDSW (RGDSW) [45] ("Option 1", "Option 2.2"). Due to its significantly reduced dimension the RGDSW coarse space heavily improves parallel scalability; cf. [88]. Furthermore, we will introduce an intermediate approach, which we denote as GDSW*; cf. [82, Chapter 4]. It yields smaller coarse spaces than the classical GDSW approach but larger coarse spaces than the RGDSW approach; in two dimensions, GDSW* and RGDSW are identical. Preliminary results for GDSW* based on its implementation in `FROSch` can be found in [96, p. 160], there under the working title *GDSW-Star*. A related approach was taken in the context of adaptive coarse spaces in [116], there under the name R–WB–GDSW.

7.1.1. Interface Partition of Unity of GDSW, GDSW*, and RGDSW Coarse Spaces

The classical GDSW coarse space defines the interface partition of unity based on a nonoverlapping decomposition of the interface degrees of freedom into interface components of faces, edges, and vertices; cf. Figure 7.1 for the three-dimensional interface partitions. In contrast, the RGDSW coarse spaces from [45] employ overlapping interface components, combining each vertex with all its adjacent faces and edges; the resulting decomposition overlaps in the faces and edges. Combining vertices with edges and faces reduces the number of interface components from the number of faces, edges, and vertices to only the number of vertices. In [81], a variant of the RGDSW coarse space is introduced in which the RGDSW interface decomposition is made nonoverlapping without increasing the number of interface components. The new GDSW* coarse space employs a partition of unity of intermediate size, where each vertex is only combined with the adjacent edges, and the faces are kept separate.

The interface partition of unity is obtained by defining one function for each interface component. As a result, the number of coarse basis functions is reduced for GDSW* and even further reduced for RGDSW; cf. Table 7.1 for a summary of the interface components used by the different GDSW variants and Figure 7.1 for a three-dimensional visualization. See also [116, Figure 5.4] for a comparison of the number of interface components for four different types of meshes and domain decompositions; note that, although the precise definition of the interface components differs, their quantity is the same. There, the analogue of GDSW* is denoted as R–WB–GDSW.

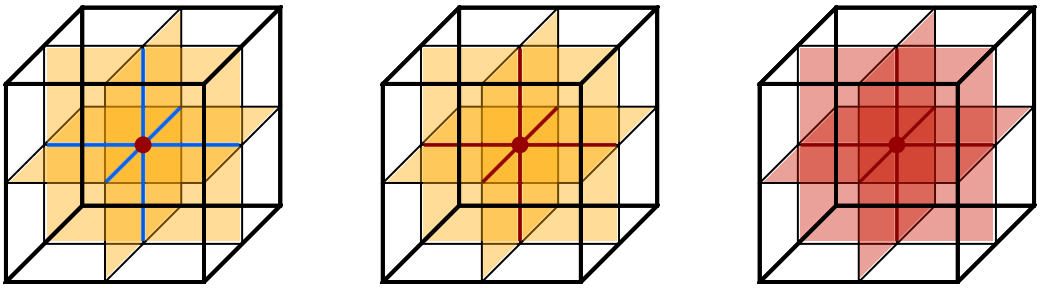


Figure 7.1: Visualization of three types of (overlapping) interface decompositions in a $2 \times 2 \times 2$ domain decomposition. **Left:** GDSW interface components: 6 edges, 1 vertex, and 12 faces. **Center:** GDSW* interface components: 1 vertex-based component (union of vertex and 6 edges), 12 faces. **Right:** RGDSW interface component: 1 vertex-based component (union of vertex, 6 edges, 12 faces).

To formally define the interface partition of unity of the GDSW, GDSW*, and RGDSW coarse spaces, we decompose the interface Γ into components $\gamma_k \subset \Gamma$ such that

$$\Gamma = \bigcup_k \gamma_k.$$

This union can be disjoint or overlapping. Then, we define functions $\Phi_{\Gamma,k}$ on Γ, with $\text{supp}(\Phi_{\Gamma,k}) \subset \gamma_k$, such that

$$\sum_k \Phi_{\Gamma,k} \equiv 1 \text{ on } \Gamma.$$

Then, we gather the functions $\Phi_{\Gamma,k}$ as columns of a matrix

$$\Phi_\Gamma = \begin{pmatrix} \Phi_{\Gamma,1} & \Phi_{\Gamma,2} & \dots \end{pmatrix}, \tag{7.4}$$

which will then be extended using (7.3). The partition of unity functions $\Phi_{\Gamma,k}$ for GDSW, GDSW*, and RGDSW are defined in the following. We introduce the following formal definition of faces, edges, and vertices. In three dimensions:

- A face is a set of nodes that belong to the same two subdomains. Let $\mathcal{F}$ be the set of all faces.
- An edge is a set of at least two nodes that belong to the same (more than two) subdomains. Let $\mathcal{E}$ be the set of all edges.
- A vertex is a single node that belongs to the same (more than two) subdomains. Let $\mathcal{V}$ be the set of all vertices.

These interface components are shown as an example in Figure 7.1 (left); they are all employed in the construction of GDSW coarse spaces. In two dimensions, the definition of an edge is the same as that of a face in 3D, and the definition of the vertex remains the same. In the following, we will introduce the different partitions of unity considered in this work, focusing on the three-dimensional case.

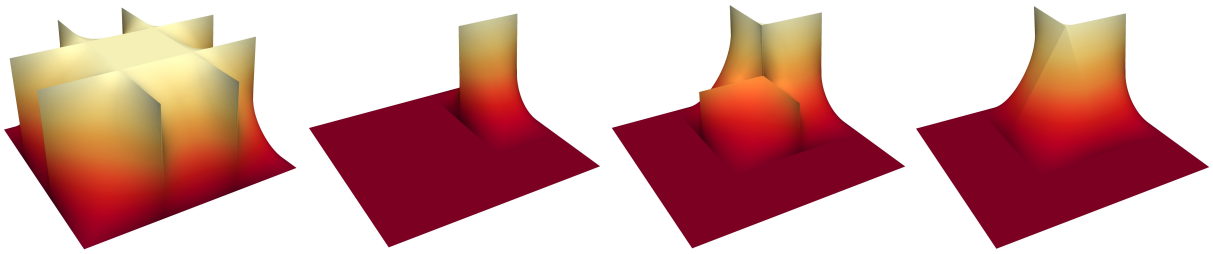


Figure 7.2: Basis function for edge interface components for GDSW and RGDSW for 3×3 partition of a square domain. **Left:** Sum over all basis functions. **Center Left:** GDSW edge function. **Center Right:** RGDSW edge function with Option 1. **Right:** RGDSW edge function with Option 2.2.

GDSW

For the construction of the coarse space, the interface components γ_k are given by faces, edges, and vertices. As a result, we may choose the discrete partition of unity functions as

$$\chi_f, f \in \mathcal{F}, \quad \chi_e, e \in \mathcal{E}, \quad \chi_v, v \in \mathcal{V};$$

here, χ_s is the discrete characteristic function which is one in the finite element nodes contained in the interface component s and zero elsewhere. Then, we have that

$$\sum_{f \in \mathcal{F}} \chi_f + \sum_{e \in \mathcal{E}} \chi_e + \sum_{v \in \mathcal{V}} \chi_v \equiv 1 \text{ on } \Gamma,$$

since the vertices, edges, and faces form a nonoverlapping decomposition of Γ.

RGDSW

Let $\mathcal{V} = \{v_1, \ldots, v_{N_\mathcal{V}}\}$. An interface component γ_k is the union of $v_k \in \mathcal{V}$ and its adjacent faces and edges. Note that, as discussed in [45], the definition is slightly more involved for general domain decompositions, where not every edge and face may have an adjacent vertex; here, for simplicity, we do not discuss these cases.

In [45, 81] various options for defining the corresponding partition of unity functions are given. We further discuss "Option 1" and "Option 2.2". The algebraic variant "Option 1" from [45] is based on a simple multiplicity scaling (cf. [45, eq. (1)]). Therefore, we first define a component multiplicity $\mathcal{M}(n)$ of an interface node n as the number of (RGDSW) interface components γ_k that contain n:

$$\mathcal{M}(n) := |\{k : n \in \gamma_k\}|,$$

where $|\cdot|$ is the cardinality of the set. We can then define a function $\Phi_{\Gamma,k}$ associated with γ_k by using the inverse of the multiplicity:

$$\Phi_{\Gamma,k}(n) := \begin{cases} 1/\mathcal{M}(n), & n \in \gamma_k, \\ 0, & \text{else.} \end{cases} \tag{7.5}$$

Furthermore, the "Option 2.2" variant is considered, which additionally requires geometric information. The interface values are based on the distance to adjacent vertices, i.e., an inverse

distance weighting is considered (cf. [45, eq. (5)]). We define a Euclidean distance function $d_k(n)$ as the distance between an interface node n and the vertex v_k of an interface component γ_k that contains n. Then the function $\Phi_{\Gamma,k}$ is defined by

$$\Phi_{\Gamma,k}(n) := \begin{cases} \frac{1/d_k(n)}{\sum_{\{k:n\in\gamma_k\}} 1/d_k(n)}, & n \in \gamma_k, \\ 0, & \text{else.} \end{cases} \tag{7.6}$$

Both options are visualized in Figure 7.2. The partition of unity property is satisfied as in every interface node $n \in \Gamma$, we have

$$\sum_k \Phi_{\Gamma,k}(n) = 1.$$

GDSW*

Finally, we introduce the partition of unity of the GDSW* coarse space. The definition is straightforward based on the definitions for GDSW and RGDSW. In particular, we define two types of interface components:

- Every vertex $v_k \in \mathcal{V}$ and its adjacent edges. These components overlap with each other.
- Every face $f_k \in \mathcal{F}$. These components do not overlap with the vertex-based interface components or other faces.

Then, we define the partition of unity using the inverse multiplicity scaling (7.5) or inverse distance weighting (7.6), this time defined via the set of GDSW* interface components. Note that, because the faces do not overlap with other interface components, the resulting functions coincide with the discrete characteristic functions χ_f of the faces $v \in \mathcal{F}$. In two dimensions, the partitions of unity of GDSW* and RGDSW are the same, since faces do not exist in that case.

7.1.2. Extension to Vector-Valued Problems

The approaches discussed before are directly applicable to scalar problems, such as problems related to the pressure block in the Navier–Stokes equations. However, the application to vector-valued problems, such as problems related to the velocity block, requires a small modification.

In case of, e.g., a multi-dimensional Laplace operator $\Delta\mathbf{u}$ or a linearized elasticity problem, respectively, the partition of unity functions γ_k are multiplied with a basis of the null space of the corresponding partial differential equation with homogeneous Neumann conditions on the boundary. This ensures that the null space can be represented by the coarse space, which is required for numerical scalability in the case of symmetric positive semidefinite problems; cf. [159, p. 132]. In our case, we use translations for the velocity block, which is motivated by $\Delta\mathbf{u}$ appearing in the diagonal velocity block of the discrete Stokes matrix. For the pressure, we use constant functions; note that these are in the null space of B^T for an enclosed-flow problem.

Specifically, for the two-dimensional case of a velocity coarse function, the scalar partition of unity functions are multiplied with

$$r_1 := \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \; r_2 := \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

and in three dimensions with

$$r_1 := \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \; r_2 := \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \; r_3 := \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix},$$

for each interface node. As a result, the number of coarse space functions associated with the velocity or pressure is the space dimension multiplied with the number of partition of unity functions. This discussion is also compatible with the scalar elliptic case. In particular, for the construction of pressure functions, the node-wise multiplication of the partition of unity functions with the scalar constant function, i.e.,

$$r := \begin{bmatrix} 1 \end{bmatrix},$$

simply yields the partition of unity functions. For elasticity problems, the null space also contains linearized rotations, which are not included in the coarse space for the velocity block. We define the rotations by

$$r_1 := \begin{bmatrix} -x_2 \\ x_1 \end{bmatrix}$$

in two dimensions, and by

$$r_4 := \begin{bmatrix} X_2 \\ -X_1 \\ 0 \end{bmatrix}, \; r_5 := \begin{bmatrix} X_3 \\ 0 \\ -X_1 \end{bmatrix}, \; r_6 := \begin{bmatrix} 0 \\ X_3 \\ -X_2 \end{bmatrix}$$

in three dimensions, where X are the coordinates in reference configuration of Ω. For straight edges in 3D only two linearized rotations are linearly independent. Therefore, only two of the three rotations are included for the edges, resulting in 5 coarse basis functions per edge.

A visualization of the coarse basis functions corresponding to translational and rotational modes for a linear elasticity problem is shown in Figure 7.3.

7.1.3. Discontinuous Finite Elements

For discontinuous finite element spaces, as employed in discontinuous piecewise linear (P1-disc.) pressure spaces, there are no interface nodes and degrees of freedom which are shared between adjacent subdomains. Therefore, the discrete interface is empty in this case, and the construction of GDSW, GDSW*, and RGDSW cannot be applied directly. As motivated in Section 7.1.2, we want to use constant functions for the pressure. Therefore, on each subdomain Ω_i, we employ the single discrete characteristic function χ_{Ω_i} (i.e., a function that is constant on Ω_i and zero elsewhere) as the coarse basis function, yielding a total of N coarse basis functions. If this occurs, the coarse space will be abbreviated with "Vol.", since the characteristic function is prescribed on the entire volume and not just on the interface.

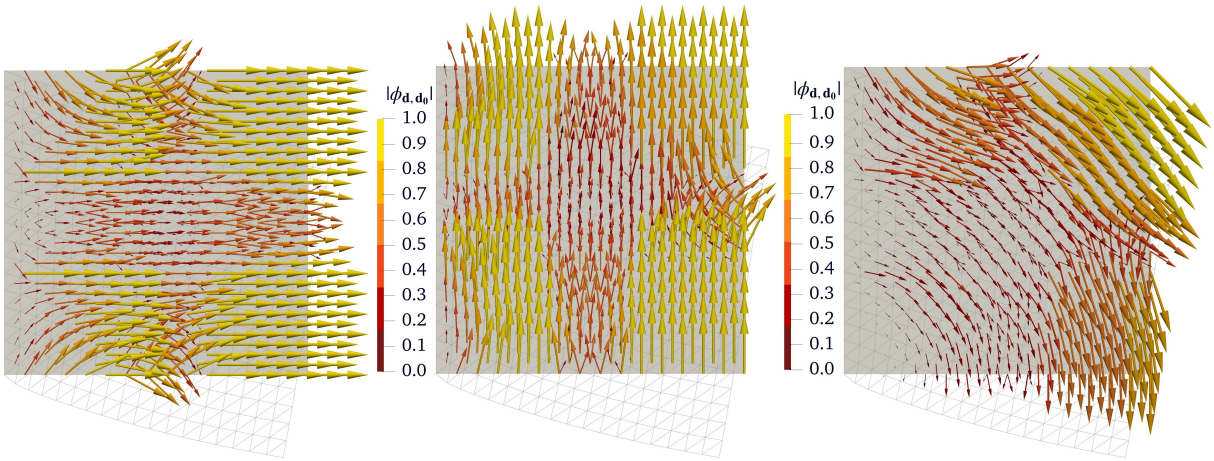


Figure 7.3: Coarse basis functions for a 2D linear elasticity problem with Dirichlet boundary condition on the left edge. Displacement solution shown as gray wire-frame. Depiction for all interface nodes of the 3×3 domain decomposition with RGDSW coarse space. **Left:** Translations in x-direction. **Center:** Translations in y-direction. **Right:** Linearized rotations.

7.2. Block Schwarz Preconditioner

For the presented block preconditioners in Section 6.1, we can use the two-level overlapping Schwarz domain decomposition method from the previous section for the approximation of the blocks F^{-1} and S^{-1}. We define the overlapping subdomain problems separately for the velocity and pressure. The velocity and pressure spaces V^h and Q^h, respectively, are decomposed into local spaces

$$V_i^h = V^h \cap (H_0^1(\Omega_i'))^d \text{ and } Q_i^h = Q^h \cap H_0^1(\Omega_i'),$$

for $i = 1, \dots, N$, on the overlapping subdomains Ω_i'. Note that the Schwarz preconditioner is not used to approximate the Schur complement directly, but the components used in the Schur complement approximations defined in Chapter 6. A precise overview of the approximated operators is given in Table 8.4.

The restriction operators to the overlapping subdomains for the velocity and pressure degrees of freedom, respectively, are defined as

$$R_{u,i} : V^h \longrightarrow V_i^h \quad \text{and} \quad R_{p,i} : Q^h \longrightarrow Q_i^h.$$

Then, the local overlapping problems are defined as

$$F_i = R_{u,i} F R_{u,i}^T \quad \text{and} \quad S_i = R_{p,i} S R_{p,i}^T,$$

where S, or an approximation of S, depends on the underlying strategy. The coarse matrix is constructed similarly to before via a Galerkin product as

$$F_0 = \Phi_u^T F \Phi_u \quad \text{and} \quad S_0 = \Phi_p^T S \Phi_p.$$

Let $\widetilde{R}_{u,i}^T$ and $\widetilde{R}_{p,i}^T$ be the scaled prolongation operators. To approximate the inverses of F and S, we use the additive two-level overlapping Schwarz preconditioners

$$\hat{F}^{-1} = \Phi_u F_0^{-1} \Phi_u^T + \sum_{i=1}^{N} \widetilde{R}_{u,i}^T F_i^{-1} R_{u,i} \quad \text{and} \quad \hat{S}^{-1} = \Phi_p S_0^{-1} \Phi_p^T + \sum_{i=1}^{N} \widetilde{R}_{p,i}^T S_i^{-1} R_{p,i}.$$

7.3. Monolithic Schwarz Preconditioner

For the saddle-point system with system matrix $\mathcal{F}$ defined in (2.16), we compare the block Schwarz preconditioners with monolithic Schwarz preconditioners. This technique has been introduced in [109, 110] with Lagrangian coarse spaces and extended to GDSW-type coarse spaces in [77, 78]. In monolithic Schwarz preconditioners, we construct a single Schwarz domain decomposition preconditioner for the full system $\mathcal{F}$ at once. The monolithic additive two-level Schwarz preconditioner reads:

$$\mathcal{M}_{\text{OS-2}}^{-1} = \phi\, \mathcal{F}_0^{-1}\, \phi^T + \sum_{i=1}^{N} \widetilde{\mathcal{R}}_i^T\, \mathcal{F}_i^{-1}\, \mathcal{R}_i. \tag{7.7}$$

Different from (7.2), we denote all matrices with calligraphic letters to indicate that they are block matrices with the same block structure as $\mathcal{F}$. The local subdomain matrices for (7.7) are given by $\mathcal{F}_i = \mathcal{R}_i \mathcal{F} \mathcal{R}_i^T$ with the restriction operators

$$\mathcal{R}_i = \begin{bmatrix} R_{u,i} & 0 \\ 0 & R_{p,i} \end{bmatrix},$$

where $R_{u,i}$ and $R_{p,i}$ correspond to the restriction operators for velocity $\mathbf{u}$ and pressure p degrees of freedom for Ω_i'. $\widetilde{\mathcal{R}}_i^T$ is the scaled prolongation operator. Then, also the coarse matrix $\mathcal{F}_0 = \phi^T \mathcal{F} \phi$ has a block structure given by the matrix

$$\phi = \begin{bmatrix} \phi_{\mathbf{u},\mathbf{u_0}} & \phi_{\mathbf{u},p_0} \\ \phi_{p,\mathbf{u_0}} & \phi_{p,p_0} \end{bmatrix}, \tag{7.8}$$

which has the coarse basis functions as its columns. $\mathbf{u_0}$ and p_0 indicate the velocity and pressure functions in the coarse space. The interface values of the coarse functions are constructed separately as for the block matrices, and the remaining interface values are always set to zero. Hence, we obtain the structure

$$\phi_\Gamma = \begin{bmatrix} \phi_{\Gamma,\mathbf{u},\mathbf{u_0}} & 0 \\ 0 & \phi_{\Gamma,p,p_0} \end{bmatrix}. \tag{7.9}$$

For a block Schwarz preconditioner, the same $\phi_{\Gamma,\mathbf{u},\mathbf{u_0}}$ and ϕ_{Γ,p,p_0} are used. Only the subsequent extension operator from the interface to the subdomains differs. The extension from the interface to the subdomains is computed monolithically, analogously to (7.3) but with submatrices of $\mathcal{F}$

$$\phi = \begin{bmatrix} \phi_I \\ \phi_\Gamma \end{bmatrix} = \begin{bmatrix} -\mathcal{F}_{II}^{-1} \mathcal{F}_{I,\Gamma} \phi_\Gamma \\ \phi_\Gamma \end{bmatrix}. \tag{7.10}$$

As a result, the coarse basis ϕ also contains components reflecting the coupling between $\mathbf{u}$ and p; cf. (7.8). Since the local saddle-point subdomain matrices $\mathcal{F}_i$ are extracted from $\mathcal{F}$, they possess homogeneous Dirichlet boundary conditions for both velocity and pressure in the case of conforming finite elements.

Pressure Projection Additionally, we enforce a zero mean pressure value on the local overlapping subdomains as in *Version 2* in [109, Sect. 4.1], which is beneficial in some cases, even though the pressure is uniquely determined by the problem setting; see also [77, Sect. 5.1].

Therefore, we introduce a pressure projection in the first level of our two-level overlapping Schwarz preconditioner

$$\mathcal{M}_{\text{OS-2}}^{-1} = \phi\, \mathcal{F}_0^{-1}\, \phi^T + \sum_{i=1}^{N} \widetilde{\mathcal{R}}_i^T\, \overline{\mathcal{P}}_i\, \mathcal{F}_i^{-1}\, \mathcal{R}_i \tag{7.11}$$

with local projection operators $\overline{\mathcal{P}}_i$ corresponding to the overlapping subdomain Ω_i. Here, the projection operators are of the form

$$\overline{\mathcal{P}}_i = \begin{bmatrix} I_{u,i} & 0 \\ 0 & \overline{P}_{p,i} \end{bmatrix}, \quad \overline{P}_{p,i} = I_{p,i} - a_i (a_i^T a_i)^{-1} a_i^T .$$

The vector a_i is defined as $a_i = R_{p,i}\, a_p$ via integrals of the pressure basis functions ψ_j:

$$a = \begin{bmatrix} 0 \\ a_p \end{bmatrix}, \quad a_p = \begin{pmatrix} \int_\Omega \psi_1 \,\mathrm{d}x & \dots & \int_\Omega \psi_{N_p} \,\mathrm{d}x \end{pmatrix}^T .$$

Results in [109, 77] show that this approach, similar to introducing a Lagrange multiplier, can improve the convergence of the iterative linear solver.

For the deformation–diffusion problem, a monolithic overlapping Schwarz preconditioner is also considered. Its construction is analogous to (7.7), with the Jacobian matrix from (4.11) replacing the saddle-point system $\mathcal{F}$. We also refer to [9, Section 4.2] for a description of the monolithic formulation and to [89], where related monolithic Schwarz preconditioners are developed for land ice problems exhibiting a similar block structure. Since the primary objective of the deformation–diffusion problem is to investigate the influence of the material models and to compare implicit and explicit coupling strategies, the monolithic preconditioner is included for completeness but not analyzed in further detail. A detailed investigation of the monolithic approach is instead carried out for the incompressible fluid flow problem; see Section 7.4 for details on the constructed basis functions and Section 10.5.1 for the numerical results.

7.4. Different Aspects of Blockwise and Monolithically Constructed Basis Functions for the Navier–Stokes Equations

The difference in velocity basis functions between the block and monolithic preconditioner is illustrated in Figure 7.4. In both cases the same interface partition of unity is used, resulting in the same interface values for both preconditioning approaches. However, as previously mentioned,

the extension into the interior differs for the block and monolithic Schwarz preconditioners, which results in different coarse basis function values for the interior degrees of freedom. The coarse basis functions which are constructed via the monolithic system visually seem to capture the nature of the underlying problem, the lid-driven cavity flow, better than those constructed for the individual velocity block; compare the left and right columns in Figure 7.4.

Note that when coarse basis reuse (CB) is considered as a recycling strategy (Section 8.5.2), the coarse basis is reused after the initial construction in the first Newton step of the first time step and therefore corresponds to the coarse basis functions for Stokes. Note that in our implementation we start the nonlinear iteration with the zero vector and obtain the Stokes solution as the first iterate.

For the monolithic preconditioner, the off-diagonal blocks from ϕ in (7.8) can be excluded, which corresponds to the coupling between velocity and pressure in the coarse basis functions. Note that, since the extension is still computed monolithically, the resulting coarse basis functions still reflect the coupling between velocity and pressure. The resulting coarse basis then reads

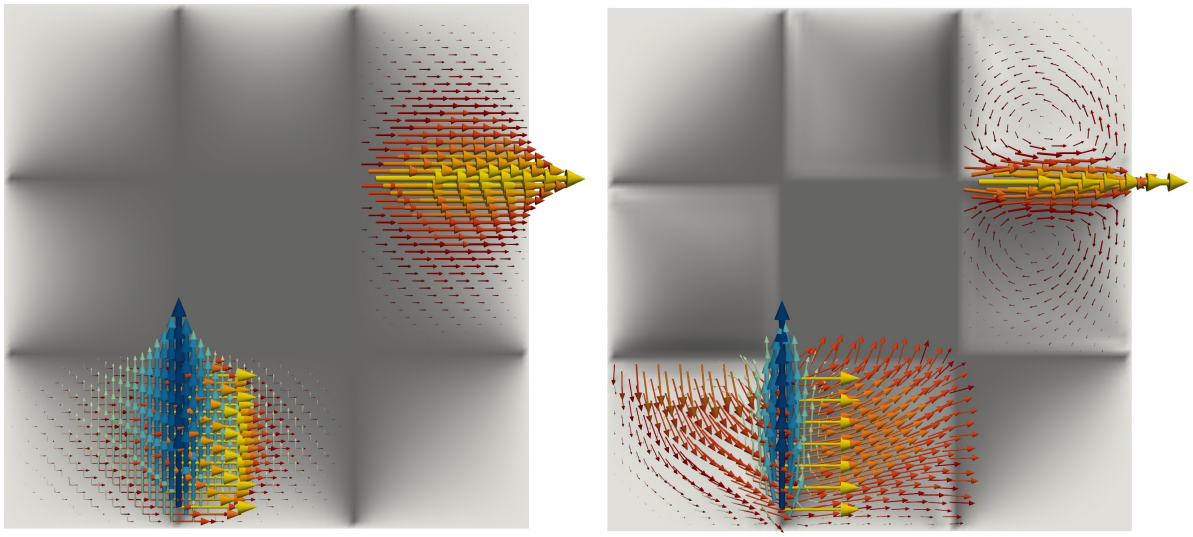
$$\phi = \begin{bmatrix} \phi_{\mathbf{u},\mathbf{u_0}} & 0 \\ 0 & \phi_{p,p_0} \end{bmatrix}. \tag{7.12}$$

This may be visually motivated by the coarse basis functions of the coupling blocks; see Figure 7.5. There, for one RGDSW interface node, the coarse basis functions are illustrated. The block $\phi_{\mathbf{u},p_0}$ couples the pressure coarse basis functions with the velocity degrees of freedom (right), and the block $\phi_{p,\mathbf{u_0}}$ couples the velocity coarse basis functions with the pressure degrees of freedom (center). The numerical results presented in Section 10.5.1, for example in Figure 10.14, show that omitting the off-diagonal blocks leads to a more robust preconditioner. A possible explanation can be obtained by inspecting the corresponding coarse basis functions shown in Figure 7.5. The block $\phi_{p,\mathbf{u}_0}$, which couples the velocity coarse basis functions to the pressure degrees of freedom, exhibits values whose magnitude exceeds one. In contrast, the entries of $\phi\mathbf{u}, p_0$ remain below one. These large coefficients appear to have a stronger influence on the partition of unity and may therefore deteriorate the robustness of the coarse correction.

This observation is further supported by two additional findings. First, the robustness is recovered either by enforcing a zero-mean pressure on each overlapping subdomain through a pressure projection or by omitting the block $\phi_{p,\mathbf{u}_0}$; cf. Figure 10.14 for the results of the different variants. Second, if this block is excluded, the pressure basis functions satisfy a partition of unity on all subdomains that are not connected to a Neumann boundary. This property is lost when the full monolithic coarse basis is employed. Moreover, for the P1–P1 discretization with Bochev–Dohrmann pressure stabilization, the method remains robust (see results in Figure 10.13) even though the corresponding basis functions exhibit a similar qualitative behavior as in Figure 7.5 (center). This suggests that the pressure stabilization compensates for the adverse effects of the coupling in $\phi_{p,\mathbf{u}_0}$. Finally, it should be noted that, unlike the linear Stokes problem, a partition of unity is generally not expected for nonlinear problems. This is illustrated in Figure 7.4, where only the linear Stokes case satisfies the partition of unity for the velocity basis functions. Nevertheless, the recovery of a pressure partition of unity after omitting $\phi_{p,\mathbf{u}_0}$ provides further evidence that this coupling block has a decisive role in the robustness of the preconditioner. A rigorous explanation of these observations, however, remains an open topic for future research.

(a) Block preconditioner for Stokes problem.

(b) Monolithic preconditioner for Stokes problem.



(c) Block preconditioner for Navier–Stokes problem.

(d) Monolithic preconditioner for Navier–Stokes problem.

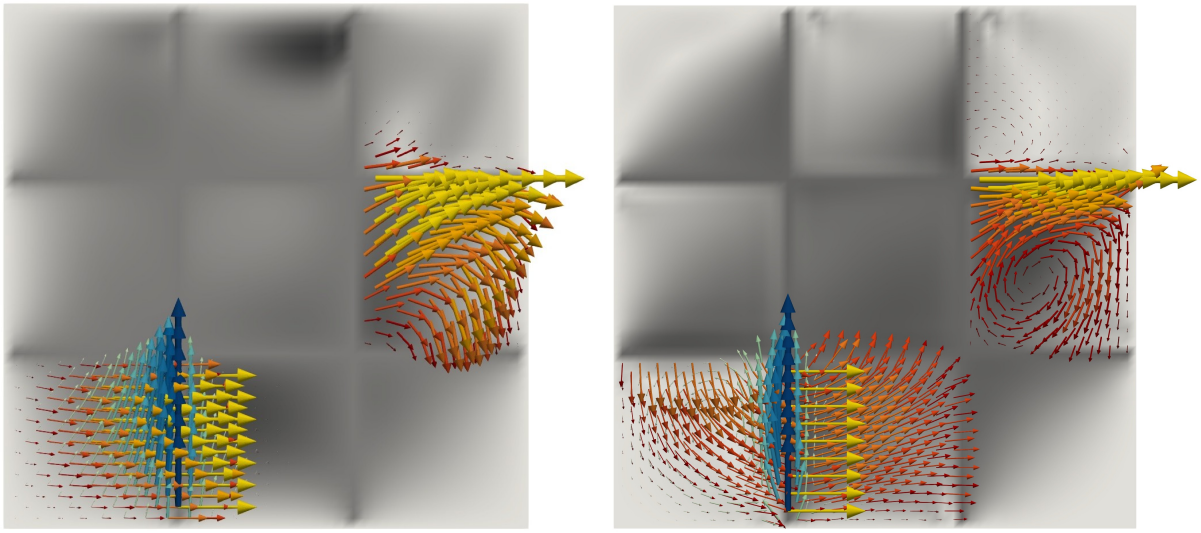


Figure 7.4: Velocity coarse basis functions $\phi_{\mathbf{u},\mathbf{u}_0}$ for a GDSW interface partition for a 2D lid-driven cavity flow with 3×3 subdomains. Coarse basis functions corresponding to two edges illustrated for the translations in x- and y-direction. Translations in x-direction are illustrated by yellow and red shades; for y-direction blue shades are used. The resulting coarse basis functions for the block preconditioning approach are shown in the left column in (a) and (c), and for the monolithic preconditioning approach in the right column in (b) and (d). For the block preconditioner, the coarse basis functions are computed for the F block; see Section 7.2. For the monolithic preconditioner, the extension from interface to interior degrees of freedom is computed for the full saddle-point systems; see (7.10). Therefore, the coarse basis functions contain a coupling between $\mathbf{u}$ and p. In the top row in (a) and (b), a Stokes flow is considered; in the bottom row in (c) and (d), the Navier–Stokes problem is considered with $\mathrm{Re} = 200$. The Navier–Stokes basis functions correspond to the last required Newton step of the stationary simulation. The gray shade in the background illustrates the sum over all basis functions. In Figure 7.1 (left) the sum over all basis functions is shown, which corresponds to the gray shade in (a).

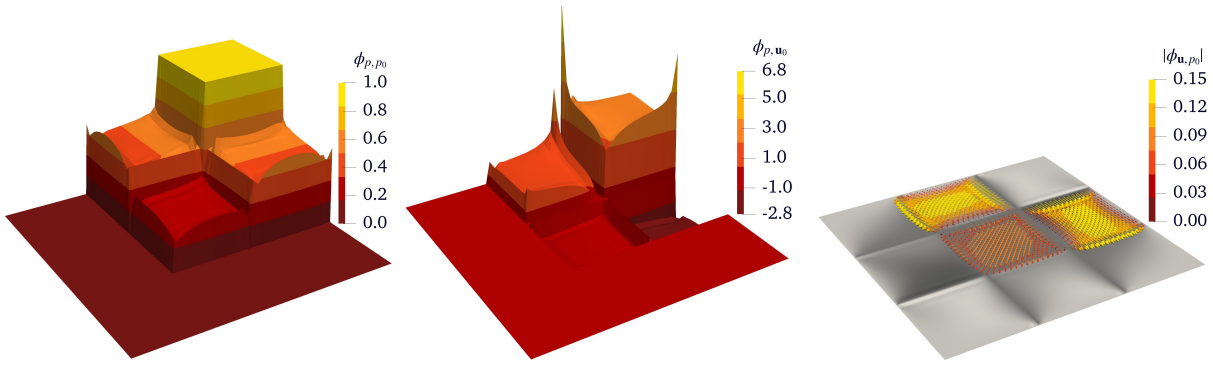


Figure 7.5: Exemplary coarse basis function for one nodal interface component for RGDSW-type partition with Option 1 and Stokes flow. Note that the values $\phi_{p,\mathbf{u_0}}$ and $\phi_{\mathbf{u},p_0}$ are zero on the interface Γ. **Left:** Coarse basis function for ϕ_{p,p_0} block. **Center:** Coarse basis function for the $\phi_{p,\mathbf{u_0}}$ block, which couples the velocity coarse basis functions with the pressure degrees of freedom. **Right:** Coarse basis function for the $\phi_{\mathbf{u},p_0}$ block, which couples the pressure coarse basis functions with the velocity degrees of freedom.

8. Software Framework

The software framework is based on the Trilinos library [164]; in particular, we employ its domain decomposition preconditioning package FROSch (Fast and Robust Overlapping Schwarz) [83] and block-preconditioning package Teko [35, 34] to construct the preconditioners under investigation. The implementation of the model problems and the corresponding finite element discretizations is done in FEDDLib (Finite Element and Domain Decomposition Library) [61, 96], which strongly relies on several Trilinos packages and assembles input directly in the corresponding format to make use of the full capabilities of FROSch and Teko. Moreover, an interface between FEDDLib and AceGen [120] was established to make use of AceGen/AceFEM generated code to construct and include sophisticated material models, e.g., the chemo-mechanical smooth muscle model from Section 3.1.2.

In this chapter, multiple software aspects are considered. First a basic introduction to FEDDLib is given. Then, the developed FEDDLib-AceGen Interface is described, which enables the efficient integration of various different material models into FEDDLib's finite element assembly routines. Then follows a description of Trilinos with the focus on different aspects which will be detailed in the numerical results. This includes the preconditioners used from FROSch, Teko, and a short review of the algebraic multigrid package MueLu [21], direct and approximate solvers, and different aspects of the linear algebra package Tpetra.

8.1. FEDDLib

FEDDLib [61] is a C++-based, object-oriented finite element library built on top of the Trilinos software infrastructure. One of the first descriptions of FEDDLib can be found in [96] by C. Hochmuth who is one of the first FEDDLib developers.

FEDDLib consists of two main packages core and problems. The core package includes the general implementation of the finite element assembly. It provides wrappers and interfaces to the parallel linear algebra classes in Trilinos; in particular, FEDDLib uses Tpetra. Whereas Epetra is the older linear algebra framework from the origins of Trilinos, Tpetra is the current linear algebra framework. There, the maps, multivectors and matrix classes are defined, all based on Tpetra objects. This enables us to directly pass the necessary matrix and vector objects to Trilinos packages; see Section 8.4.1 for a detailed review of maps and distributed matrix-vector multiplications. Furthermore, the interface to the Mathematica code generation package AceGen/AceFEM is located here [120]. The problems package contains implementations of various physical models, which range from simple scalar elliptic equations, such as diffusion problems, to computational fluid dynamics applications, and complex multiphysics systems. In particular, it includes the fluid–structure interaction problems, with the results detailed in Sections 10.6 and 10.7. Further problems can be easily added through the AceGen interface.

In [9] the interface is also discussed and results are shown for a coupled pharmaco–mechanical problem.

Other interfaces to nonlinear, direct and iterative solvers from `Trilinos` are included in the `problems` package. For instance, the package `NOX` is used for solving the nonlinear problems in this work, as it employs (inexact) Newton's method, with globalization strategies; see Section 10.1.1. The unified solver interface `Stratimikos` enables the use of iterative solvers implemented in `Belos` and direct solvers, which are accessible via the `Amesos2` package.

In order to accelerate the convergence of iterative solvers, `Trilinos` provides various preconditioners that are also available to `FEDDLib`; see Section 8.6 for block preconditioners for fluid flow in `Teko` and Section 8.5 for the two-level overlapping Schwarz preconditioners.

8.2. AceGen

`AceGen` [120] is a Mathematica package, which is used for the automated derivation of mathematical expressions arising in numerical methods. It combines the symbolic and algebraic capabilities of Mathematica with a hybrid symbolic–automatic differentiation technique, enabling automatic code generation together with expression optimization, to minimize the computational cost of the resulting code. In addition, `AceGen` supports code generation for several programming languages and finite element software packages — for example, FEAP (Finite Element Analysis Program) [162] — and the Mathematica-based finite element environment AceFEM, which enable the testing of `AceGen` implementations.

In the present work, `AceGen` is employed to generate finite element routines with consistent tangent operators. The residual and tangent operators are derived by automatic differentiation, ensuring that the tangent matrix represents the exact linearization of the discrete residual, which is essential for achieving optimal local convergence properties of Newton's method. Through the `FEDDLib-AceGen` interface, the constitutive models described in Sections 3.1.1 and 3.1.2 are integrated into the software framework. Automatic differentiation is thereby used to derive the consistent tangent matrices, such as the Jacobian in (4.11), required for the solution of the nonlinear problem with Newton's method. In the following, we refer to the generated finite element routines as "`AceGen`-generated", however, they are also AceFEM compatible.

More information and details can be found in the `AceGen` and `AceFEM` manuals in [178, 177, 176]. A detailed description with application examples is given in [120]. For instance, [120, Section 6.3] details three-dimensional hyperelastic solid elements.

8.3. FEDDLib–AceGen Interface

The `FEDDLib-AceGen` interface was developed though a collaboration involving the research groups of Axel Klawonn (University of Cologne) and Daniel Balzani (Ruhr University Bochum), together with Jascha Knepper (University of Cologne), Alexander Heinlein (TU Delft), and Oliver Rheinbach (TU Bergakademie Freiberg). The main development work was carried out by Lea Saßmannshausen and Sharan Nurani Ramesh, with contributions from all other collaborators. In [138] the interface was first used, and further tested in [9]. The following description is part of a publication, which is currently in preparation [11]. The following description contains the

interface details corresponding to FEDDLib. The implementation in AceGen/AceFEM and the development of the corresponding C++ wrapper interface, which directly interacts with FEDDLib, were carried out by Sharan Nurani Ramesh.

The developed interface enables the integration of AceGen-generated finite element code into the external software library FEDDLib by providing the corresponding C++ finite element classes. These classes are capable of computing optimized element-level quantities, such as residual vectors, stiffness matrices, history variables, and post-processing data. To bridge the gap between AceGen's generated code and FEDDLib, the interface employs a layered approach: AceGen's Fortran routines are wrapped in C++. The generated Fortran code is treated as a black box. It is hidden behind a clear C++ API that abstracts away low-level memory management and error-prone operations, enabling safe, high-level interaction while preserving performance. To support these functionalities, the interface supplies a set of routines for managing the underlying data structures required by the AceGen elements. It offers a unified, object-oriented abstraction, thereby greatly simplifying the use of generated finite elements. In principle, this interface also supports the incorporation of a wide range of finite elements from the AceShare library — which hosts numerous community-contributed elements — into C++ finite element codes.

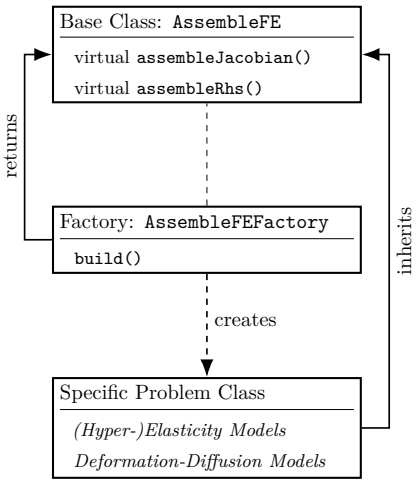


Figure 8.1: Schematic of the C++ factory design pattern implemented in AssembleFEFactory for the assembly of specific problems like elasticity.

AssembleFE Elements

To leverage this interface, FEDDLib's assembly routines have been extended to integrate externally generated AceGen elements. The implementation follows an object-oriented, factory-based design pattern: specific assembly routines are built on top of a generalized base class that defines the interface through a set of pure virtual functions. Through a factory, a specific element — the AssembleFE element— is created for each finite element. This also enables the combination of different constitutive material models, such as neo-Hookean and linear elasticity, or the assignment of different parameters to each AssembleFE object and thus, to each finite element. In Listing 8.1 an example of the AssembleFEFactory is shown and how an AssembleFE of type "LinearElasticity" is built on top of the base class. Then, for each problem type, the derived classes override key functions such as assembleRhs() and assembleJacobian() to implement the appropriate finite element assembly logic; compare Figure 8.1.

Each AssembleFE element holds different data types: static data, which does not change over the course of the simulation, e.g., the dimension, element flag, reference node configuration and parameter list, and dynamic data, which includes the current solution, history variable, the computed right-hand side and Jacobian, and post-processing outputs. History variables are essential for the smooth muscle model described in Section 3.1.2. They include the states of the different materials' past responses, e.g., the evolution of cross-bridge fractions, which develop

```
AssembleFEPtr_Type AssembleFEFactory::
   build(std::string problemType, int flag,
   vec2D_dbl_Type nodesRefConfig, ParameterListPtr_Type parameterList)
{
 AssembleFEPtr_Type assembleFE;
 AssembleFEPtr_Type assembleFESpecific;

 if(problemType == "LinearElasticity"){
   Teuchos::RCP<AssembleFE_LinElas> assembleFESpecific (new
    AssembleFE_LinElas(flag,nodesRefConfig, parameterList));
   assembleFE = assembleFESpecific;
  }
 return assembleFE;
};
```

Listing 8.1: Code snippet showing the factory design pattern implemented in `AssembleFEFactory` for the assembly of a linear elasticity `AssembleFE` object.

over time and describe the four states of the cross-bridge model in (3.7). The element flag allows assigning distinct material parameters to different structural components (e.g., plaque regions).

For practical use, several implementation aspects are relevant: consistent element-specific node ordering may be required to ensure compatibility with global mesh data structures, and history variables, which are essential for realistic material models, are managed locally per element.

Global Assembly

During computation, `FEDDLib` mainly interacts with the base class to retrieve the element residual vector (`getRhs()`) and Jacobian matrix (`getJacobian()`), which are the essential components of the linearized Newton system (4.12). The resulting element-wise system matrices and vectors are assembled into parallel distributed data structures, the sparse matrix-type `Tpetra::CrsMatrix` and vector-type `Tpetra::MultiVector`. These structures are then passed to the `Trilinos` solvers and preconditioners to perform scalable, high-performance computations. This process is visualized in Figure 8.2. Therefore, parallelization is handled outside of the interface by `FEDDLib`'s distributed infrastructure, ensuring the interface remains lightweight, portable, and decoupled from parallel MPI communication.

Figure 8.2 illustrates the class hierarchy of the `FEDDLib`–`AceGen` interface, while Figure 8.3 shows its integration into FSI and FSCI workflows. The incorporation of the interface into FSI and FSCI simulations is based on the modular structure of the coupled formulation; see (5.6). For standard FSI problems, the structural block is represented by the corresponding `AssembleFE` element. In the FSCI setting, this block is replaced by an `AssembleFE` element describing the coupled deformation–diffusion problem.

In Section 10.2, the proposed interface is validated with respect to both performance and correctness. The performance evaluation compares the runtime of the generated assembly routines with that of existing implementations in `FEDDLib`. The correctness of the computed solutions is verified by comparison with existing `FEDDLib` implementations, reference computations performed with `AceGen`/`AceFEM`, and the results of an FSI benchmark problem [8, 10].

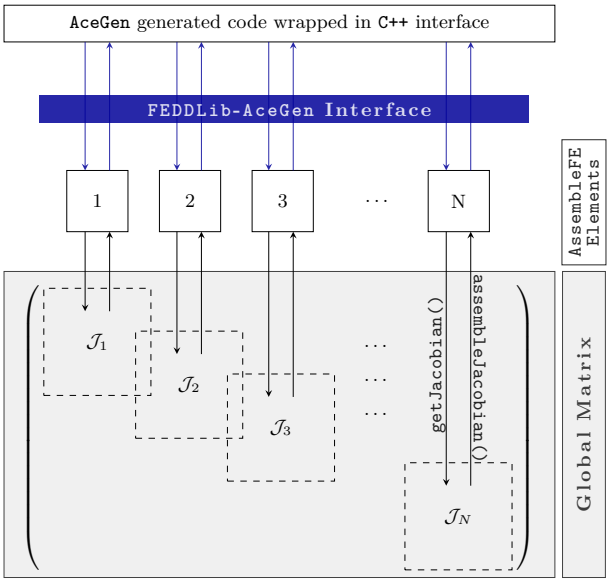


Figure 8.2: Functionality of the interface `AssembleFE` elements. Via `assembleJacobian()` and `getJacobian()`, the global Jacobian matrix is assembled. After the assembly is completed, the `Tpetra` `fillComplete()` function is called to finalize the global matrix. There, also the matrix entries are added that exist on multiple subdomains.

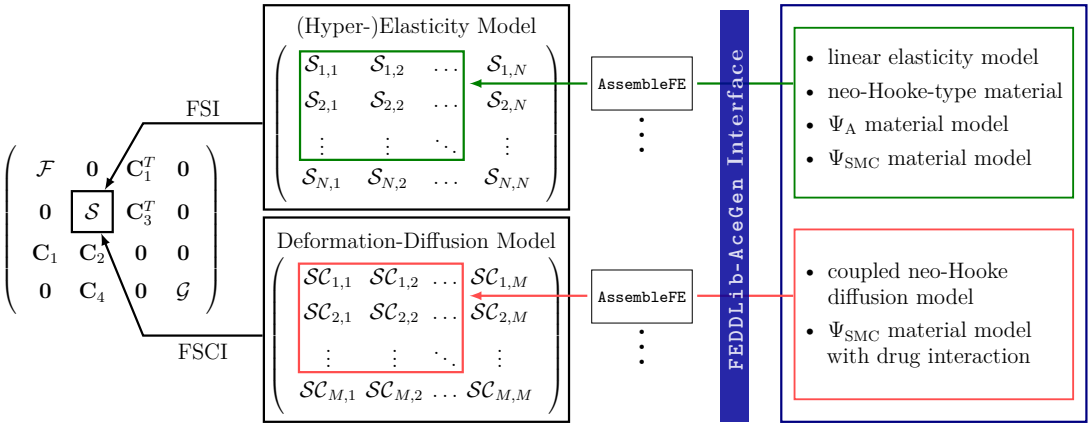


Figure 8.3: Functionality of the interface for the FSI and FSCI problems, where the solid subproblem is assembled elementwise through the interface. The described wall models from Section 3.1 are included in the interface. Furthermore, the deformation–diffusion model from Chapter 4 is also included in the interface.

8.4. Trilinos

The `Trilinos` library [164] is a collection of interoperable software packages for high-performance scientific computing; see also [94]. Since the list of packages is extensive, we will focus on those which are relevant to this work. The main building block for the solvers and preconditioners is the parallel linear algebra package `Tpetra`, which provides parallel vector and matrix classes as well as functions that allow communication of those objects. This results in a software stack of packages

that operate on Tpetra objects, such as direct and iterative solvers, and preconditioners. Tpetra automatically incorporates access to the Kokkos performance portability framework [48, 165] as well as the corresponding kernel library Kokkos Kernels [148]. As a result, functions of Tpetra objects allow for node-level parallelization on CPUs and GPUs via Kokkos' node type template parameter; here, we will only focus on distributed memory parallelization, and hence, we will not consider the effects of node-level parallelization via Kokkos. The older Epetra linear algebra package is currently being deprecated together with the whole Epetra-only stack of packages. Therefore, we employ only the Tpetra stack. The problems package of FEDDLib provides interfaces to the various solver packages used throughout this work. Nonlinear systems are solved using the NOX package, which implements Newton-based algorithms together with globalization techniques such as line search. In particular, NOX is employed as the inexact Newton method described in Section 10.1.1.

The linear systems arising within each Newton iteration are solved through the unified solver interface Stratimikos [14]. Stratimikos provides access to the iterative solvers contained in Belos as well as to direct solvers interfaced through the Amesos2 package. Here, the GMRES method [155] from Belos is employed as the Krylov solver, while the direct solver MUMPS is accessed via Amesos2; see Section 8.8 for further details on MUMPS. Direct solvers are primarily used for the application of the preconditioners.

To accelerate the convergence of iterative solvers, Trilinos provides various preconditioners, including:

- Ifpack2: one-level Schwarz preconditioners, supporting incomplete factorizations on the subdomains; see Section 8.4.2,
- Teko: block preconditioners; see Section 8.6,
- FROSch: multilevel Schwarz preconditioners; Section 8.5, and
- MueLu: algebraic multigrid, as a solver or preconditioner; see Section 8.4.3.

As mentioned before, here, we focus on Teko and FROSch preconditioners; cf. Section 6.1 and chapter 7, respectively, for details on the preconditioners employed in this work. Details on the Teko and FROSch packages are given in Sections 8.5 and 8.6, respectively.

8.4.1. Tpetra Maps

For sparse matrix and vector operations in distributed memory, Tpetra uses different maps to define the distribution of data across processes, i.e., which process holds which rows of a vector or matrix. Specifically, for matrix-vector multiplication the following different maps exist for the matrix A, input vector x, and output vector y in the operation $Ax = y$:

- The domainmap defines the distribution of input vector x, i.e., which rank holds the nth entry of x.
- The rangemap defines where each global row of output vector y is stored after multiplication, i.e., on which rank the nth entry of y is stored.
- The rowmap indicates on which rank each global row of the matrix A is stored.

- The columnmap holds the corresponding global column information for each locally stored row of A, i.e., the nonzero column entries of each locally stored row of A.

For a typical finite element square matrix, the row-, domain-, and rangemaps are identical, because the input and output vectors have the same distribution as the matrix rows; see Figure 8.4 (left). However, for rectangular matrices, the rowmap and domainmap differ, as the input and output vectors have different distributions; see Figure 8.4 (right).

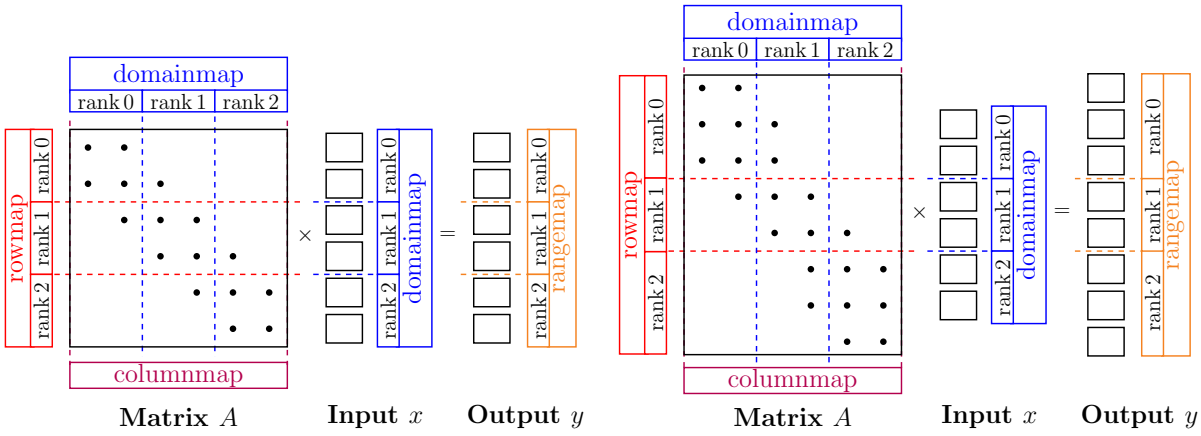


Figure 8.4: Definitions of row-, column-, domain-, and rangemaps for matrix A and input vector x and output vector y in the matrix-vector multiplication $Ax = y$. **Left:** Square matrix A. In this case, the row-, domain-, and rangemaps are the same. For example on rank 1, the rowmap contains [2,3], the domainmap contains [2,3], and the rangemap contains [2,3]. The columnmap contains all column entries referenced by the locally stored rows 2 and 3; therefore it holds the global IDs [1,2,3,4]. **Right:** Rectangular matrix A. In this case, the row- and domainmaps are not the same. For example on rank 0, the rowmap contains [0,1,2], the domainmap contains [0,1], and the rangemap contains [0,1,2]. The columnmap includes all column entries referenced by the locally stored rows resulting in [0,1,2].

Usually, two types of rowmaps are considered: unique and repeated maps. The matrix A is stored using the unique map, where each interface node is assigned to exactly one subdomain; see Figure 8.5 (center). In contrast, the repeated map contains, for each subdomain, all nodes belonging to it, including interface nodes; see Figure 8.5 (left). This representation is useful, for example, for storing subdomain solution values and is used in `FROSch` to construct overlapping matrices; see Figure 8.8.

8.4.2. Ifpack2

The `Ifpack2` package [156] contains incomplete factorization, relaxation, and domain decomposition methods, e.g., algebraic one-level overlapping Schwarz preconditioners. Therefore, it can be applied as an approximate solver, preconditioner, or smoother for algebraic multigrid. It is based on the `Tpetra` linear algebra stack.

In this work, the use of an incomplete factorization is motivated by the augmented Lagrangian preconditioner. As discussed in Section 8.5.1, the additional AL term increases the sparsity pattern of the Navier–Stokes system, resulting in a larger number of nonzero entries; see also Section 10.5.2. Consequently, the computation of an exact LU factorization becomes more

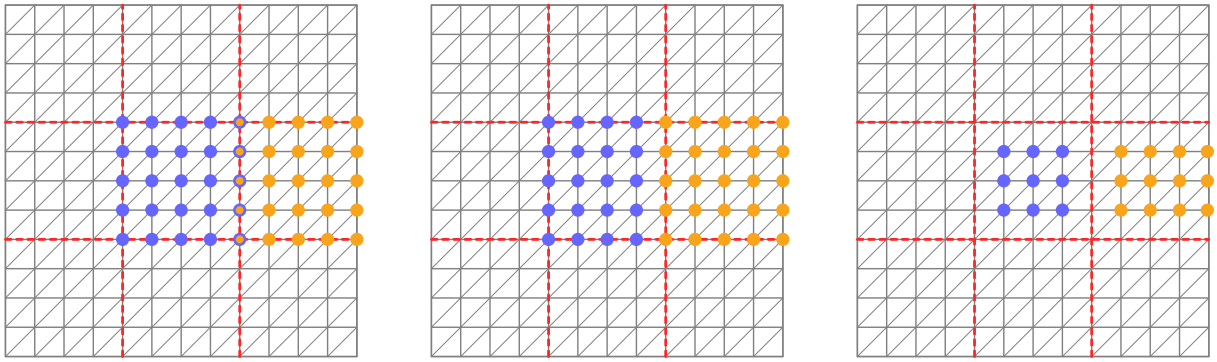


Figure 8.5: Illustration of the different types of rowmaps. **Left:** Repeated Map. All nodes belonging to a subdomain are owned by the same rank, even if they are shared with other subdomains. **Center:** Unique Map. Each node is owned by exactly one rank, even if it belongs to multiple subdomains. **Right:** Map including all nodes on one subdomain but without interface nodes. This map is specifically defined for the augmented Lagrangian approach.

expensive, motivating the use of incomplete factorizations [131, 154]. Specifically, the RILUK (relaxed incomplete LU-factorization with level-of-fill k) method is employed in Section 10.5.2. Additionally, relaxation methods, e.g., Gauss–Seidel, are considered in Table 10.18. For more options available in `Ifpack2` we refer to the user guide [156, Chapter 2].

Instead of computing the exact LU factorization of a sparse matrix, the incomplete factorization constructs approximate lower- and upper-triangular factors, usually denoted by $\tilde{L}$ and $\tilde{U}$. The amount of additional fill-in allowed during the factorization is controlled by the level parameter k. For $k = 0$, the sparsity pattern of the factors is restricted to the original sparsity pattern of the matrix, while larger values of k allow additional nonzero entries and therefore usually lead to a more accurate but more expensive preconditioner. Compared to direct solvers, RILUK is typically much cheaper in terms of memory and setup cost, but it is less robust because the quality of the preconditioner depends on the matrix properties and on the chosen level of fill. In our test cases, we choose $k = 0$ level-of-fill. Corresponding results are shown in Figure 10.20.

8.4.3. MueLu: Algebraic Multigrid Package

We briefly review the smoothed aggregation algebraic multigrid (SA-AMG) method implemented in the `MueLu` package [21] of `Trilinos`. In Section 10.5.2, `MueLu` is employed as a preconditioner for the GMRES method to approximately invert the velocity and Schur complement blocks arising in the augmented Lagrangian preconditioner introduced in Section 6.1.4. A comprehensive short introduction to algebraic multigrid methods is given in [28, Chapter 8]. The smoothed aggregation approach implemented in `MueLu` is based on the work of Vaněk, Mandel, and Brezina [170]. Monolithic AMG methods for coupled systems are not considered here; see, for example, [68, 145].

Unlike geometric multigrid, AMG constructs the multigrid hierarchy solely from the algebraic properties of the system matrix. Consequently, no explicit information about the underlying mesh or geometry is required. Starting from the linear system $A_l u_l = f_l$, the transfer operators and coarse-level matrices are constructed recursively from the matrix graph and its coefficients.

In SA-AMG, the first step is the construction of aggregates, which are groups of degrees of

freedom identified by graph-based aggregation algorithms; see [69, Section 10]. In the present work, the default uncoupled aggregation algorithm of `MueLu` is employed, meaning that each MPI process constructs its aggregates independently. Each aggregate represents one coarse unknown. Based on the aggregates, piecewise constant basis functions are constructed. The corresponding tentative prolongation operator is denoted by P_l^0, where each column represents one coarse basis function. This construction is conceptually similar to the coarse basis functions used in GDSW-type coarse spaces, except that the basis functions are associated with aggregates instead of interface components and aggregates are generally smaller and nonoverlapping. To improve interpolation quality, the tentative prolongation is smoothed according to

$$P_l = \left(I - \omega D^{-1} A_l\right) P_l^0,$$

where D denotes the diagonal of A_l. The coarse-level operator is then obtained using the Galerkin projection

$$A_{l+1} = P_l^T A_l P_l,$$

with P_l^T serving as the restriction operator. During the multigrid cycle, relaxation methods are applied as pre- and post-smoothers to efficiently damp high-frequency error components. The remaining smooth error is reduced by solving the residual equation on the coarse level and interpolating the resulting correction back to the fine level; see [166] for more details. The `MueLu` package provides several smoothing schemes through the `Ifpack2` package, including Jacobi, (symmetric) Gauss–Seidel, and RILUK (relaxed incomplete LU-factorization with level-of-fill k Section 8.4.2). For example, one Gauss–Seidel iteration for the matrix splitting $A_l = D_l + L_l + U_l$ is given by

$$u_l^{k+1} = (D_l + L_l)^{-1}(f_l - U_l u_l^k).$$

In Figure 8.6 one SA-AMG V-cycle is illustrated. The coarse solve is by default solved on all MPI ranks. `MueLu` provides an option to redistribute the coarse problem to a smaller number of MPI ranks, which can improve scalability for large-scale problems.

8.5. Fast and Robust Overlapping Schwarz (FROSch)

The Trilinos package FROSch (Fast and Robust Overlapping Schwarz) [83, 84] provides a highly-scalable, parallel framework for overlapping Schwarz domain decomposition solvers with GDSW-type coarse spaces; the first version of the implementation was described in [84, 75]. In numerous publications over the past years, `FROSch` has been employed for a variety of physics problems [84, 77, 91, 83, 89, 92, 183, 90, 76, 82, 121]. The references listed here represent only a selection of the work related to `FROSch`.

`FROSch` builds on the Schwarz framework [163] to construct preconditioners by combining elementary Schwarz operators, for instance, implementing the additive two-level Schwarz preconditioner by combining the first and second level operators; cf. (7.1) and (7.2). The Schwarz preconditioners are constructed algebraically, that is, based on the sparsity pattern of the parallel-distributed system matrix. The construction of the coarse spaces described in Section 7.1 is mostly algebraic except for elasticity problems, where the null space is spanned by the rigid body modes, and the (linearized) rotations cannot be constructed algebraically; they have to be

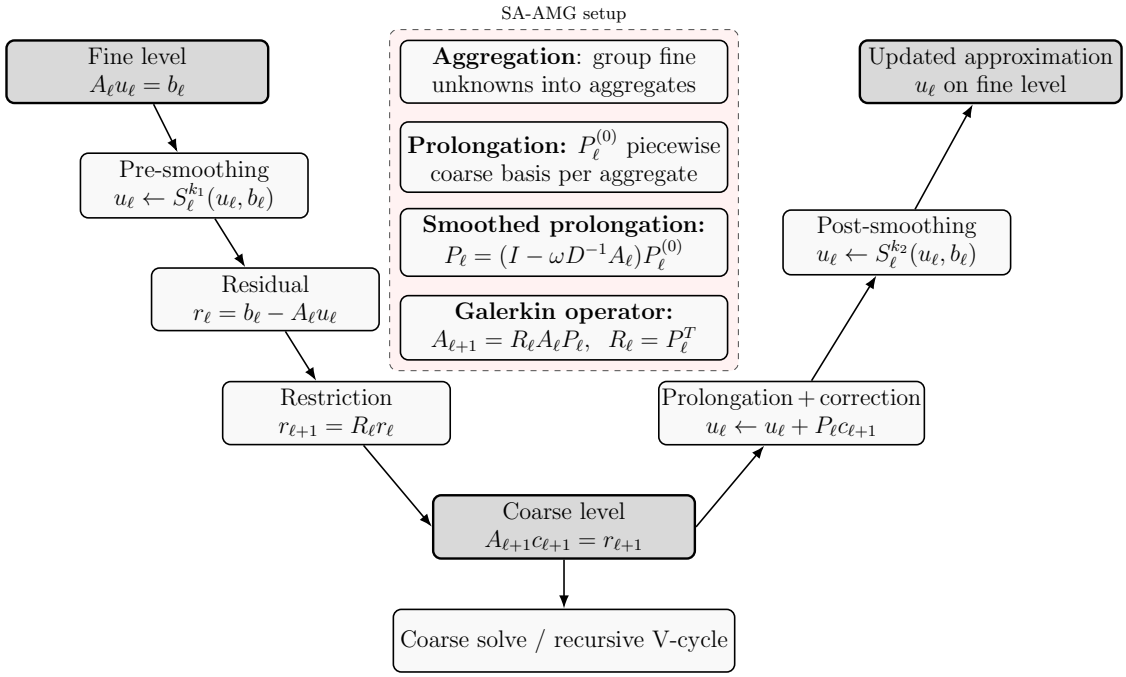


Figure 8.6: Illustration of the algebraic multigrid method (AMG) for the fine level problem $A_l u_l = b_l$. The multigrid hierarchy is constructed based on the algebraic properties of the system matrix, without requiring explicit geometric information of the underlying problem. One multigrid cycle is shown moving from level l to level $l+1$. S_l^k represents the smoothing operator, e.g., Gauss–Seidel, and k the number of smoothing operations. Based on [69, Section 10] and [174, Chapters 3 and 4]

either provided by the user or constructed using coordinates of the finite element nodes. For more details on the algebraic construction of GDSW-type coarse spaces for elasticity problems, see [79]. Furthermore, the Option 2.2 partition of unity function (7.6) for RGDSW requires the distance between a vertex and interface node, which is computed based on the coordinates of the finite element nodes, and needs to be provided by the user.

For the first level, the overlapping subdomains are constructed from nonoverlapping subdomains by extending the subdomains by layers of adjacent degrees of freedom. Adjacency is defined based on nonzero off-diagonal entries in the system matrix; see also Section 8.5.1 for more details. A key feature of `FROSch` is its use of extension-based coarse spaces that avoid explicit geometric information for the problems (cf. Section 7.1), and in particular, the need for coarse triangulations; see Chapter 7 for a detailed discussion of the methodology.

A review of the different coarse spaces is given in Section 7.1. The coarse spaces are based on the GDSW preconditioner [41, 40] and have been adapted for block systems [77]. Moreover, `FROSch` implements the reduced-dimension GDSW variants (RGDSW) [88, 77]. The extension to monolithic GDSW and RGDSW preconditioners was first described in [77], and in [96, 82] for the GDSW* coarse space. Multilevel extensions [153, 92] further enhance scalability, enabling `FROSch` to scale to over 220 000 cores [90]. See also [121, 122] for recent publications using FROSch as coarse solver in multigrid and on monolithic multilevel overlapping Schwarz solvers

for fluid problems. FROSch preconditioners can be easily constructed via Trilinos' unified solver interface Stratimikos using only the parallel-distributed system matrix and a list of parameters defining the specific setup of the preconditioners. FROSch supports both parallel linear algebra frameworks, `Epetra` and `Tpetra`, via the lightweight `Xpetra` interface; due to the recent deprecation of Epetra, its support is merely a legacy feature. If Tpetra is used, the performance portability libraries Kokkos and Kokkos Kernels can be employed, enabling the use on, for instance, GPU architecture [183]. Due to its algebraic implementation, FROSch preconditioners can be constructed recursively, resulting in multilevel Schwarz preconditioners; see [90]. `FROSch` preconditioners are compatible with iterative Krylov solvers from the `Belos` package [15], either directly or through `Stratimikos`. For more details, see [84, 77, 83]. In this work, direct solvers accessed through the `Amesos2` package are used to apply both the first and second levels of the preconditioner; see Section 8.8 for details on MUMPS. In one example, an approximate solver from the `Ifpack2` package is employed for the first level solve instead; see Section 8.4.2.

8.5.1. Constructing Overlapping Matrices

The construction of the first level, i.e., the overlapping subdomain matrices, is algebraic and based on the sparsity pattern of the parallel-distributed system matrix. The construction of overlapping subdomains starts from the repeated map shown in Figure 8.4 (left). This corresponds to an overlap size of $\hat{\delta} = 0$, where adjacent subdomains already share the interface degrees of freedom.

To construct $\hat{\delta} = 1$, one layer-adding operation is performed. To this end, the columnmap is used to determine the degrees of freedom that correspond to nonzero entries of the local matrix rows. For example, Figure 8.7 (left) illustrates the P2 finite element stencil of a velocity degree of freedom (blue), where the coupled velocity degrees of freedom contained in the corresponding columnmap are highlighted in yellow. The resulting overlapping subdomain for $\hat{\delta} = 1$ is shown in Figure 8.8 (c).

The augmented Lagrangian formulation of the Navier–Stokes equation contains the additional term $B^T \hat{M}_p^{-1} B$ added to the velocity block; see Section 6.1.4. For continuous pressure discretizations, the term introduces an additional coupling between velocity degrees of freedom through the pressure degrees of freedom, which enlarges the finite element stencil shown in Figure 8.7 (center). The gray region indicates the pressure degrees of freedom that contribute to these additional couplings. Therefore, a single layer-adding operation produces a larger overlapping subdomain than for the original Navier–Stokes system; see Figure 8.8 (d). Note that $\hat{M}_p$ is the diagonal approximation of the pressure mass matrix M_p.

In order to construct a comparable overlap for the augmented Lagrangian system, we introduced a different map; see Figure 8.4 (right) and Figure 8.7 (b). If the initial overlapping matrix is constructed with this rowmap, the overlap generated for $\hat{\delta} = 1$ is identical to that of the original Navier–Stokes system; see also Figure 8.7 (right). This allows a direct comparison of the original and augmented Lagrangian preconditioners without the additional influence of an enlarged overlap; see Section 10.5.2. Note that this adjustment only produces the identical overlap for a P2–P1 discretization; in case of a P1–P1 discretization the overlap slightly differs.

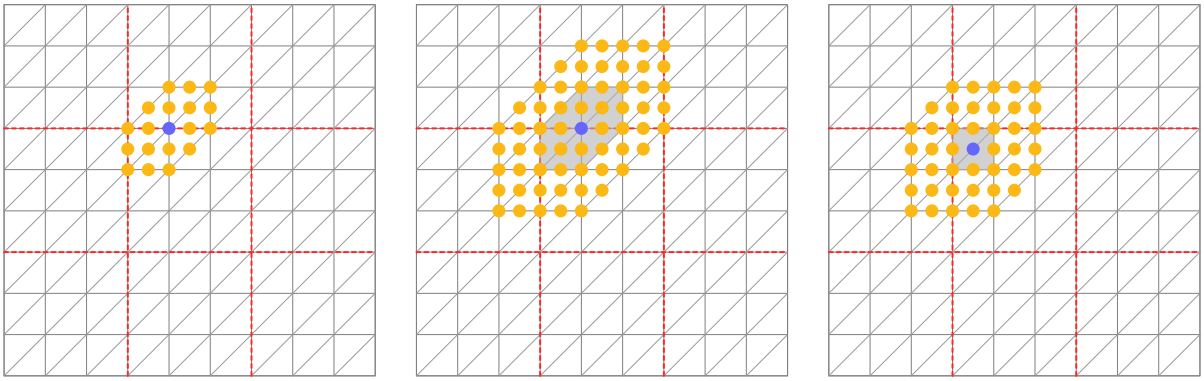


Figure 8.7: Finite element stencils for the velocity block F_N of the Navier–Stokes system and a P2–P1 discretization. The blue node illustrates a velocity degree of freedom. The yellow nodes indicate the velocity degrees of freedom coupled to the blue node, corresponding to the nonzero entries in the associated row (or equivalently, columnmap) of the system matrix. **Left**: Stencil of the standard velocity block. **Center**: Stencil of the augmented Lagrangian formulation. In gray the pressure degrees of freedom are shown, which illustrate the additional coupling introduced by the term $B^T \hat{M}_p^{-1} B$; see (6.20). Note that $\hat{M}_p$ is the diagonal approximation of the pressure mass matrix M_p. The matrix B^T maps the pressure dofs to velocity dofs, while B maps velocity to pressure dofs. The multiplication $B^T \hat{M}_p^{-1} B$ then couples the gray pressure degrees of freedom to all neighboring velocity degrees of freedom associated with them. As a result, additional velocity-velocity couplings are introduced, enlarging the finite element stencil. **Right**: Augmented Lagrangian stencil for a velocity degree of freedom located at a P2 edge node.

8.5.2. Recycling Strategies

To save computing time, information from previous time steps or Newton iterations can be reused. Figure 9.2 shows the dependence of the number of coarse functions on the used coarse spaces. Generally, the higher the number of coarse functions is, the higher the setup costs are. The reuse strategies described in [78], which are included in the software package `FROSch`, are the reuse of the symbolic factorization (SF), of the coarse basis functions (CB), and of the coarse matrix (CM); see Table 8.1. Consider the formulation of the two-level overlapping Schwarz preconditioner

$$M_{\text{OS-2}}^{-1} = \Phi K_0^{-1} \Phi^T + \sum_{i=1}^{N} R_i^T K_i^{-1} R_i, \tag{8.1}$$

with coarse matrix $K_0 = \Phi^T K \Phi$. The reuse or recycling strategies fall into two categories, reusing the symbolic factorization or reusing components of the preconditioner. If these types of reuse are applied, it means that the corresponding components are not recomputed in each Newton iteration or time step, but instead reused after the initial setup in the first Newton iteration (of the first time step).

One part of the setup for a direct solver is the symbolic factorization, where the sparsity pattern of the LU factors is computed, including the fill-in introduced by the factorization, while the numerical entries are not yet evaluated; see Section 8.8 for more details on MUMPS. After it is initially built, the symbolic factorization (SF) can be reused without influencing the solver performance. Generally, the reuse of a symbolic factorization is advantageous with respect to

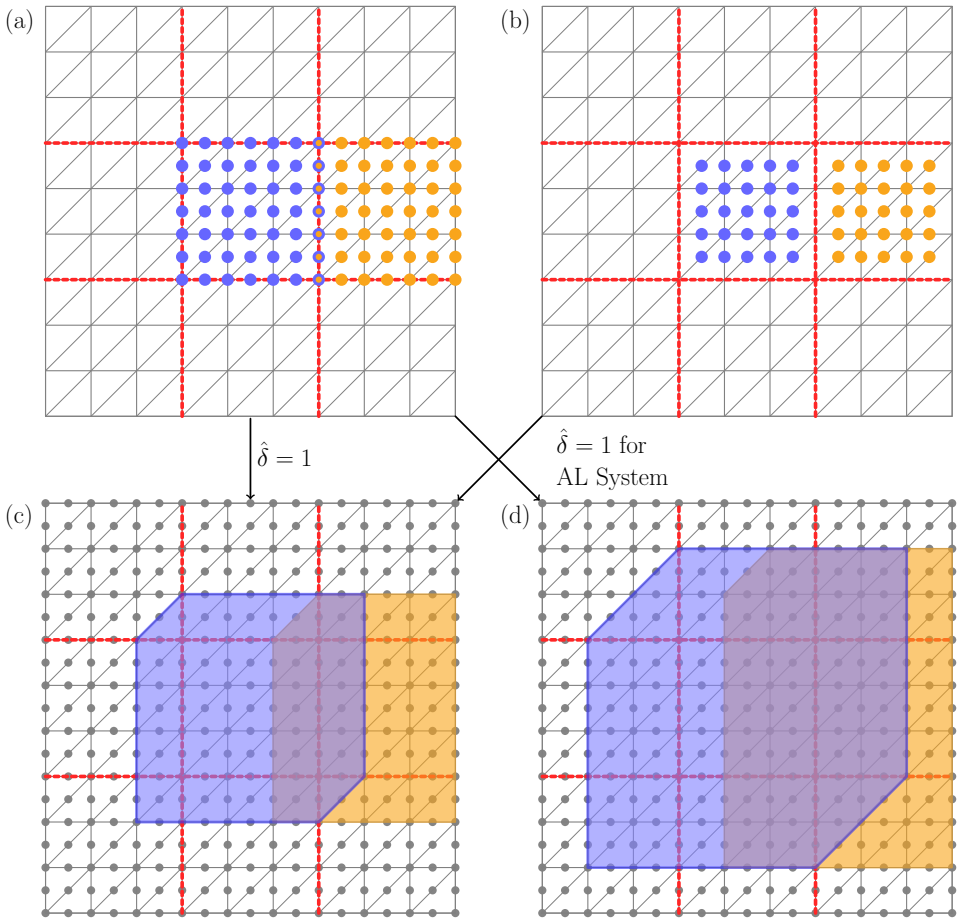


Figure 8.8: Algebraic determination of the overlapping matrices for the F_N block of the Navier–Stokes system with and without the augmented Lagrangian approach; compare (6.20). Visualization of the subdomains and the overlap for a 2D problem with 9 subdomains and P2 elements. Algebraic determination of the overlap in (c) and (d) is based on different map definitions in (a) and (b). The overlap $\hat{\delta}$ determines the number of layer-adding operations of the algebraic overlapping algorithm. Based on the repeated map definition shown in (a), for the AL system the overlap for $\hat{\delta} = 1$ is larger than for the normal system; compare (c) and (d). Using the special map definition from (b) for the AL system, the algebraic overlap construction for $\hat{\delta} = 1$ leads to the same overlap as for the original system. To obtain the minimal overlap with $\hat{\delta} = 0$ the unique map must be used for overlap construction. Note that for P1 elements the overlap for $\hat{\delta} = 1$ is not exactly the same in case (c) for the normal and AL system.

setup time as the nonzero pattern of the system typically stays the same. Therefore, in this work, the symbolic factorization of the overlapping matrices and the coarse matrix is reused.

Furthermore, the coarse basis Φ (CB) and the coarse matrix K_0 (CM) in (8.1) can be reused after the initial setup. If the coarse basis is reused (CB), only the coarse matrix — in our case the numeric factorization — is recomputed in each Newton iteration. Reusing the entire coarse matrix K_0 eliminates the time for symbolic and numeric factorization. Reusing the coarse basis and the coarse matrix are more invasive strategies. They can further decrease the setup time but can also increase the iteration count, as the preconditioner no longer adapts exactly to the new Newton iteration or time step; see for example the results in Table 10.7 for different

Table 8.1: Recycling strategies as described in Section 8.5.2.

Abbrev.	Recycling Strategy
(SF)	reuse of symbolic factorizations of local subdomain matrices and coarse matrix
(CB)	reuse of the coarse basis
(CM)	reuse of the entire coarse matrix

Table 8.2: Comparison of setup and solve times of the monolithic preconditioner of recomputing it during the first k Newton iterations (NR-k). Reuse of symbolic factorization (SF) and coarse basis (CB). Recompute preconditioner (except for (SF) and (CB)) up until k Newton iterations, reuse entire preconditioner based on kth iteration afterwards. Stationary backward-facing step problem Section 9.1 with P2–P1 discretization with $H/h = 9$. Average GMRES iteration count per Newton step. Total number of Newton iterations required to reach convergence is 5. RGDSW–RGDSW coarse space combination for the velocity (first component) and pressure (second component). Taken from [82, Table 6.4].

	k	5	4	3	2
RGDSW–RGDSW	# avg. iter.	52.8	52.8	54.0	78.4
	Setup cost	29.4 s	24.3 s	19.4 s	14.8 s
	Solve cost	35.9 s	34.9 s	35.9 s	56.1 s
	Total	65.3 s	59.2 s	**55.3 s**	70.9 s

reuse strategies for the deformation–diffusion problem from Chapter 4. For the Navier–Stokes problem, reusing the coarse basis (CB) means that the coarse basis functions are computed only for the corresponding Stokes problem; the coarse basis is constructed in the first Newton step, where the initial guess is zero, resulting in the Stokes system. As demonstrated in [78, Section 5.4] and [96, Table 8.14], these basis functions remain a good approximation throughout the Navier–Stokes simulation, resulting in only a marginal increase in the iteration count. Note, however, that we are still solving the correct tangent matrix system in each Newton step since only the preconditioner is affected.

In general, the preconditioner is rebuilt or updated in each Newton iteration. In addition to the previously described reuse strategies, the setup cost can be reduced further by only rebuilding or updating the preconditioner in the first k Newton iterations (of each time step or of the stationary simulation), and then reusing it. This reuse strategy is denoted as (NR-k), defining the number of Newton iterations k after which the preconditioner is reused. This is crucial for the monolithic preconditioner, where the setup cost is higher compared to block-preconditioning strategies; cf. Sections 7.2 and 7.3. In Table 8.2 results for the aforementioned reuse of the full preconditioner after k iterations are shown. Based on the results for the three-dimensional stationary backward-facing step case (Section 9.1), we can save computing time by reusing the complete preconditioner after three Newton steps. For $k = 3$ we achieve the best total time, since the linear iteration count is only slightly increased, while the setup time is small resulting in (NR-3).

8.6. Block Preconditioners in Teko

This section is taken from [82, Section 5.3]. `Teko` is a `Trilinos` package for block-preconditioning multiphysics problems. It can be used to build generic preconditioning strategies for arbitrary problems, the simplest of which is a diagonal preconditioner. Specifically for the Navier–Stokes problem, there are a number of block preconditioners within `Teko`. We will look at the block preconditioners presented in Section 6.1 and different modifications that can be applied. The SIMPLE, SIMPLEC, LSC, and PCD preconditioners are part of the `NS` (Navier–Stokes) namespace within `Teko`. For a detailed description of `Teko` and the Navier–Stokes preconditioners within `Teko`, we refer to [35, 34].

To make components of the Schur complement available to `Teko`, callback functions are defined to pass along different operators that are constructed outside of `Teko` (cf. Table 6.1), in our case in `FEDDLib`. Some callback functions are always optional, e.g., the one for the velocity mass matrix (for which a diagonal matrix based on F is substituted), whereas other operators are required to be passed along, e.g., A_p and F_p for the PCD preconditioner.

For the approximation of the inverses of the fluid matrix, $\hat{F}^{-1}$, and the Schur complement, $\hat{S}^{-1}$, the specification of inverse types is necessary. If no inverse type is specified, the default is to use a direct solver from the `Amesos2` package. Ideally, a less expensive approximation strategy is specified. For our simulations, we will use different two-level overlapping Schwarz preconditioners from the `Trilinos` package `FROSch`; see Table 8.4 for an overview of the used combinations of preconditioning strategies.

The stated recycling strategies are also applied to the block preconditioners; see Section 8.5.2. Additionally, for the LSC and PCD block preconditioners, the approximations of A_p^{-1} and M_p^{-1}, which are used in the Schur complement approximations, are reused after the initial setup. Since both matrices are linear, they do not depend on the current solution and therefore do not require recomputation.

8.6.1. PCD Block Preconditioner

The PCD preconditioner in `Teko` is derived from the abstract class `LU2x2Strategy`. To use it, the parameter `Type` is set to *Block LU2x2* and `Strategy Name` to *NS PCD Strategy*; see Listing 8.2. For the LSC and SIMPLE preconditioners, the `Type` directly refers to the block preconditioner, e.g., it is set to *NS LSC* instead of *LU2x2Strategy*; see Listings 8.3 and 8.4 for the LSC and SIMPLE parameter files.

For PCD, the order of the factors within the Schur complement approximation can be flipped by specifying the parameter `Flip Schur Complement Ordering`:

- $S_{PCD}^{-1} = -M_p^{-1} F_p A_p^{-1}$, specified with *false* (“default”),
- $S_{PCD}^{-1} = -A_p^{-1} F_p M_p^{-1}$, specified with *true* (“flipped”; cf. (6.7)).

The “default” Schur complement ordering corresponds to an earlier version of the Schur complement approximation based on a commutator that uses a gradient operator on the pressure instead of a divergence operator on the velocity; see [57] (2.8) for the “default” and (2.2) for the “flipped” version; see also [55, Remark 9.3].

```
<ParameterList name="PCD">
  <!-- Block LU2x2 is the Type -->
  <Parameter name="Type" type="string" value="Block LU2x2"/>
  <!-- Strategy Name is then NS PCD Strategy -->
  <Parameter name="Strategy Name" type="string" value="NS PCD Strategy"/>
  <ParameterList name="Strategy Settings">
    <Parameter name="PCD Operator" type="bool" value="true"/>
    <Parameter name="Pressure Laplace Operator" type="bool" value="true"/>
    <Parameter name="Pressure Mass Operator" type="bool" value="true"/>
    <!--FROSch preconditioner for approximation of inverses -->
    <Parameter name="Inverse Type" type="string" value="FROSch-Pressure"/> <!-- default -->
    <Parameter name="Inverse F Type" type="string" value="FROSch-Velocity"/>
    <Parameter name="Inverse Laplace Type" type="string" value="FROSch-Pressure"/>
    <Parameter name="Inverse Mass Type" type="string" value="FROSch-Pressure"/>
    <Parameter name="Flip Schur Complement Ordering" type="bool" value="true"/>
  </ParameterList>
</ParameterList>
```

Listing 8.2: Pressure Convection–Diffusion (PCD): Parameter XML file for `Teko` (`Trilinos`).

Table 8.3: Stationary simulation of backward-facing step (see Section 9.1) with P2–P1 discretization, $H/h = 9$, and 1 125 cores. PCD block preconditioner with boundary condition type (BC–2) and RGDSW–RGDSW coarse space. Required number of nonlinear steps in parentheses. See Section 10.1 for parameters of the nonlinear and linear solver. Total time consists of setup and solve time.

	Full LDU	Block-triangular
# avg. iterations	**78** (5)	103 (5)
Setup time	14.0 s	14.4 s
Solve time	77.1 s	56.9 s
Total	91.1 s	**71.3 s**

The Schur complement is constructed and defined as a linear operator. To provide all required operators for the PCD preconditioner, callback functions are defined for M_p, A_p, and F_p. `Teko` then creates approximations for M_p^{-1} and A_p^{-1} according to the parameter file. The inverse of A_p is approximated with an overlapping Schwarz method via `FROSch`. The inverse of M_p will either be approximated via `FROSch` or by using the same approach as in LSC and SIMPLE for M_u and F, respectively; that is, we can use a diagonal approximation of M_p^{-1}:

$$H_{M_p} = \operatorname{diag}(M_p)^{-1},$$
$$H_{M_p}^{i,j} = \delta^{ij} \Big(\sum_{k=1}^{n} |M_p^{i,k}| \Big)^{-1},$$

referred to as *Diagonal* or *AbsRowSum* scaling, respectively. Note that the pressure convection–diffusion operator F_p needs to be updated in each Newton or fixed-point iteration, while A_p and M_p do not change.

The PCD preconditioner introduced in Section 6.1.1 is a block-triangular preconditioner (6.3). In `Teko`, the PCD preconditioner can be applied as a full LDU block preconditioner (based on

(6.2)) or block-triangular preconditioner. The results in Table 8.3 show how the LDU solve can improve iteration count, but the total time is increased as the application of the preconditioner becomes more costly. Consequently, we will use the classic block-triangular approach:

1. $p = \hat{S}_{PCD}^{-1} f_p$,
2. $\mathbf{u} = \hat{F}^{-1}(f_u - B^T p)$.

Here, F^{-1} and S_{PCD}^{-1} have been replaced with their approximations, $\hat{F}^{-1}$ and $\hat{S}_{PCD}^{-1}$, based on a Schwarz method; see Table 8.4 for an overview of the approximations used in preconditioners.

As covered in Section 6.1.1, we need to additionally account for the boundary conditions that would naturally arise from the original Schur complement S. The boundary information is integrated into A_p and F_p by `FEDDLib`. When we set the Dirichlet boundary conditions in the different operators, we zero out the row and keep the diagonal entry; cf. [35, S325]. The use of Robin boundary conditions additionally requires the assembly of a matrix on the surface elements of the respective boundary.

8.6.2. LSC Block Preconditioner

To use the LSC preconditioner in `Teko`, the parameter `Type` is set to *NS LSC*; see Listing 8.3. The LSC preconditioner, just like PCD, is an upper block-triangular preconditioner that is applied in two steps:

1. $p = \hat{S}_{LSC}^{-1} f_p$,
2. $\mathbf{u} = \hat{F}^{-1}(f_u - B^T p)$.

The Schur complement S_{LSC} is defined in (6.13). Similarly to before, F^{-1} and S_{LSC}^{-1} have been replaced with their approximations $\hat{F}^{-1}$ and $\hat{S}_{LSC}^{-1}$ based on a Schwarz method. We have different options to treat the inverse of M_u: M_u can be passed along to `Teko` via a callback function. Inverting M_u is expensive, but it can be well approximated with H_M or $H_M^{i,j}$ (see Section 6.1.2 for the definition). Via the parameter file, H_M can be selected with *Diagonal* and $H_M^{i,j}$ with *AbsRowSum*. If M_u is not provided via a callback function, by default it is replaced with $\mathrm{diag}(F)$. If M_u is replaced with the identity matrix, it leads to the BFBt preconditioner in equation (6.14), presented in [56].

Similarly to the PCD preconditioner, we also have the option to replace $BM_u^{-1}B^T$ for continuous pressure elements with the pressure-Laplacian A_p. The resulting Schur complement approximation we denote by $S_{\text{LSC-A}_p}$; see (6.15). In that case, we need to account for boundary conditions; we use the same method as described in Section 8.6.1 to implement the boundary conditions in A_p and always prescribe a Dirichlet condition on the outlet, which corresponds to the boundary condition strategy (BC–2) of PCD.

The preconditioner is part of the `NS` (Navier–Stokes) sub package of `Teko` as `InvLSCStrategy` or `PresLaplaceLSCStrategy`, where the *Strategy Name* specifies the strategy (see Listing 8.3).

To compute the spectral radius of a matrix M in the stabilized variant, `Teko` uses a block Krylov-Schur method with block size 5 and tolerance 0.05 for the convergence criterion $\|Mx - \lambda x\|_2 / \|\lambda\|_2$, where x is an approximation of an eigenvector and λ of the corresponding eigenvalue. This is carried out in each Newton iteration, when the preconditioner is rebuilt.

```
<ParameterList name="LSC-Pressure-Laplace">
  <Parameter name="Type" type="string" value="NS LSC"/>
  <!-- Basic Inverse, Pressure Laplace -->
  <Parameter name="Strategy Name" type="string" value="Pressure Laplace"/>
  <ParameterList name="Strategy Settings">
    <!--FROSch preconditioner for approximation of inverses -->
    <Parameter name="Inverse Velocity Type" type="string" value="FROSch-Velocity"/>
    <Parameter name="Inverse Pressure Type" type="string" value="FROSch-Pressure"/>
    <!-- Using LDU decomposition for preconditioning structure (true) or upper
    block-triangular (false) -->
    <Parameter name="Use LDU" type="bool" value="false"/>
    <!-- Mass matrix scaling-->
    <Parameter name="Scaling Type" type="string" value="AbsRowSum"/>
    <!-- Use of mass matrix (true) or F -->
    <Parameter name="Use Mass Scaling" type="bool" value="true"/>
    <!-- Assuming stable (true) or stabilized  discretization (false) -->
    <Parameter name="Assume Stable Discretization" type="bool" value="true"/>
  </ParameterList>
</ParameterList>
```

Listing 8.3: Least-Squares Commutator (LSC): Pressure-Laplace Parameter XML file for `Teko` (`Trilinos`).

```
<ParameterList name="SIMPLE">
  <!-- NS SIMPLE is the Type -->
  <Parameter name="Type" type="string" value="NS SIMPLE"/>
  <!--FROSch preconditioner for approximation of inverses -->
  <Parameter name="Inverse Velocity Type" type="string" value="FROSch-Velocity"/>
  <Parameter name="Inverse Pressure Type" type="string" value="FROSch-Pressure"/>
  <!-- Diagonal (SIMPLE), AbsRowSum (SIMPLEC), Lumped, BlkDiag-->
  <Parameter name="Explicit Velocity Inverse Type" type="string" value="AbsRowSum"/>
  <!-- Parameter alpha -->
  <Parameter name="Alpha" type="double" value=".9"/>
</ParameterList>
```

Listing 8.4: SIMPLEC: Parameter XML file for `Teko` (`Trilinos`).

8.6.3. SIMPLE Block Preconditioner

The `SIMPLE` preconditioner is also part of the `NS` sub package of `Teko`. It is selected by setting the parameter `Type` to *NS SIMPLE*; see Listing 8.4. The type used to approximate the inverse of F determines whether it is called SIMPLE or SIMPLEC: The type *Diagonal* corresponds to SIMPLE and *AbsRowSum* to SIMPLEC. The inverses of F and the Schur complement are replaced with approximations $\hat{F}^{-1}$ and $\hat{S}^{-1}_{\text{SIMPLE}}$ based on a Schwarz method. The under-relaxation parameter is always set to $\alpha = 0.9$; cf. [34, p. 350]. Unlike for PCD and LSC, the application of SIMPLE and SIMPLEC requires three steps:

1. $\mathbf{u}^* = \hat{F}^{-1} f_u$,
2. $p = \alpha\, \hat{S}^{-1}_{\text{SIMPLE}}(f_p - B\mathbf{u}^*)$,
3. $\mathbf{u} = \mathbf{u}^* - \frac{1}{\alpha} H_F B^T p$.

Table 8.4: Approximations of $\mathcal{F}^{-1}$ used in simulations. $\mathbf{A_S}[\cdot]$ denotes the application of a Schwarz preconditioner. In the case of a monolithic preconditioner, the entire system $\mathcal{F}^{-1}$ is approximated with a Schwarz method. In all other cases, the matrices F^{-1} and S^{-1} are approximated and then the block preconditioner (6.3) is applied.

	$\hat{S}^{-1}$	$\hat{F}^{-1}$
PCD	$-\mathbf{A_S}\left[A_p^{-1}\right] F_p \mathbf{A_S}\left[M_p^{-1}\right]$	$\mathbf{A_S}[F^{-1}]$
SIMPLEC	$\mathbf{A_S}\left[\left(-C - BH_F^{i,j}B^T\right)^{-1}\right]$	$\mathbf{A_S}[F^{-1}]$
LSC	$-\mathbf{A_S}\left[(BH_M^{i,j}B^T)^{-1}\right]\left(BH_M^{i,j}FH_M^{i,j}B^T\right)\mathbf{A_S}\left[(BH_M^{i,j}B^T)^{-1}\right]$	$\mathbf{A_S}[F^{-1}]$
LSC_{A_p}	$-\mathbf{A_S}\left[A_p^{-1}\right]\left(BH_M^{i,j}FH_M^{i,j}B^T\right)\mathbf{A_S}\left[A_p^{-1}\right]$	$\mathbf{A_S}[F^{-1}]$
$\text{LSC}_{\text{stab},A_p}$	$-\mathbf{A_S}\left[A_p^{-1}\right]\left(BH_M^{i,j}FH_M^{i,j}B^T\right)\mathbf{A_S}\left[A_p^{-1}\right] - \alpha D^{-1}$	$\mathbf{A_S}[F^{-1}]$
Monolithic	$\mathbf{A_S}[\mathcal{F}^{-1}]$	

8.6.4. Comparison Methodology

In order to compare monolithic and block preconditioning for the Navier–Stokes equations, we will evaluate the time to set up the preconditioners, the time taken by the Krylov method (referred to as "solve" time or cost), and the average iteration count per Newton step. The solve time includes the application of the preconditioner. Depending on the preconditioning strategy, the solve cost can vary. The PCD and LSC preconditioners are block-triangular preconditioners and, thus, consist of two steps. On the other hand, the SIMPLE preconditioner contains the application of a backward and forward substitution and consists in total of three application steps. Independent of this, the block preconditioners contain approximations of inverses based on a Schwarz method. The monolithic preconditioner contains only the approximation of the inverse of $\mathcal{F}$ based on a Schwarz method. For the block and monolithic preconditioners, the setup time consists of the approximation of the inverses of F and the Schur complement (which includes approximations of, e.g., A_p^{-1} using a Schwarz method), or of the inverse of $\mathcal{F}$, respectively. Table 6.1 illustrates the different operators that need to be approximated for the different preconditioning strategies. Table 8.4 gives an overview of the combinations of the different preconditioning strategies.

8.7. FaCSI Algorithm

The FaCSI algorithm was implemented in `FEDDLib` through a custom linear operator that extends the `Thyra::PhysicallyBlockedLinearOpBase<SC>` interface; see Section 6.2 for the definition of the block preconditioner and [96, Section 7.5] for further details on the preconditioner implementation. This operator is designed to be compatible with nonlinear solvers such as `NOX` and linear solvers like `GMRES` in `Belos`, enabling its use as a preconditioner within the inexact Newton framework (Section 10.1.1). The operator accepts an input vector `R` and produces

Algorithm 1 FaCSI Preconditioner Application.

Input: Residual Vector $\mathcal{R} = [\mathbf{r}_{\mathbf{d}_s}, \mathbf{r}_{\mathbf{d}_f}, \mathbf{r}_{\mathbf{u}_{f,p}}, r_\lambda]^T$
Output: $\mathbf{Y} = [\mathbf{y}_{\mathbf{d}_s}, \mathbf{y}_{\mathbf{d}_f}, \mathbf{y}_{\mathbf{u}_{f,p}}, y_\lambda]^T$

1: **Solid Block**
- Extract structure block $\mathbf{r}_{\mathbf{d}_s}$ from input vector
- For $\mathcal{S}$: Apply preconditioner $\mathbf{y}_{\mathbf{d}_s} \leftarrow \mathcal{B}_{\mathcal{S}}^{-1}\mathbf{r}_{\mathbf{d}_s}$
- For $\mathcal{SCI}$: Combine structure $\mathbf{r}_{\mathbf{d}_s}$ and diffusion r_c blocks, and apply coupled preconditioner $[\mathbf{y}_{\mathbf{d}_s}, y_c] \leftarrow \mathcal{B}_{\mathcal{SCI}}[\mathbf{r}_{\mathbf{d}_s}, r_c]$

2: **Geometry Block**
- Extract geometry displacement block $\mathbf{r}_{\mathbf{d}_f}$
- Update $\hat{\mathbf{r}}_{\mathbf{d}_f} \leftarrow \mathbf{r}_{\mathbf{d}_f} - C_5\mathbf{y}_{\mathbf{d}_s}$
- Apply preconditioner $\mathbf{y}_{\mathbf{d}_f} \leftarrow \mathcal{B}_{\mathcal{G}}^{-1}\hat{\mathbf{r}}_{\mathbf{d}_f}$

3: **Shape Derivatives**
- Update $\mathbf{r}_{\mathbf{u}_{f,p}} \leftarrow \mathbf{r}_{\mathbf{u}_{f,p}} - \mathcal{D}\mathbf{y}_{\mathbf{d}_f}$

4: **Fluid Block (with condensation)**
- Extract fluid block $\mathbf{r}_{\mathbf{u}_{f,p}}$
- Extract Lagrange multiplier block r_λ
- Update $\hat{r}_\lambda \leftarrow r_\lambda - C_2\mathbf{y}_{\mathbf{d}_s}$
- *Condensation step*
  Eliminate velocity interface constraints in $\mathbf{r}_{\mathbf{u}_{f,p}}$ and replace them with $\hat{r}_\lambda$
  $\hat{\mathbf{r}}_{\mathbf{u}_{f,p}} \leftarrow \mathbf{r}_{\mathbf{u}_{f,p}} - C_1^T C_1\mathbf{r}_{\mathbf{u}_{f,p}} + C_1^T\hat{r}_\lambda$
- Apply either block or monolithic preconditioner $\mathbf{y}_{\mathbf{u}_{f,p}} \leftarrow \mathcal{B}_{\bar{\mathcal{F}}}^{-1}\hat{\mathbf{r}}_{\mathbf{u}_{f,p}}$

5: **Lagrange Multiplier**
- Update $y_\lambda \leftarrow C_1(\mathbf{r}_{\mathbf{u}_{f,p}} - \mathcal{F}\mathbf{y}_{\mathbf{u}_{f,p}})$

an output vector `Y`, representing the application of the preconditioner during the iterative solution process. The preconditioner is built for the block-structured systems arising from fluid–structure interaction (FSI) problems from Chapter 5, with optional support for the coupled deformation–diffusion problem (Chapter 4).

The system is partitioned into physical subproblems, each corresponding to a distinct block in the linearized system; cf. (5.6). Specifically, block 0 represents the fluid velocity $\mathbf{u}_f$, block 1 the fluid pressure p, block 2 the structure or solid displacement $\mathbf{d}_s$, and block 3 the interface Lagrange multipliers λ. Optionally, block 4 may represent the geometry component or ALE displacement $\mathbf{d}_f$, and block 5 the chemistry or concentration field for the deformation–diffusion problems c. Note that the definition of the FaCSI factorization has a different block ordering ((6.24)), which is used in Algorithm 1.

These blocks are coupled through a set of discrete coupling operators; see also Section 5.1

for the different coupling operators. The coupling between the fluid and the velocity interface Lagrange multipliers is represented by C_1 and its transpose $C_3 = C_1^T$. The coupling between the structure and the interface Lagrange multipliers is denoted by C_2 and its transpose $C_4 = C_2^T$. For the implicit treatment of the geometry problem, the coupling C_5 links the geometry displacement to the structure. Each physical subproblem can be preconditioned independently using dedicated preconditioners. The fluid subsystem is preconditioned via $\mathcal{B}_{\mathcal{F}}$, which may be either a monolithic or block preconditioner. The structure subsystem is handled by $\mathcal{B}_{\mathcal{S}}$, while for SCI problems, a coupled deformation–diffusion preconditioner $\mathcal{B}_{\mathcal{SCI}}$ may be used as an alternative. For the geometric implicit approach, the geometry subproblem is preconditioned via $\mathcal{B}_{\mathcal{G}}$; see (6.24). The application of the FaCSI linear operator is shown in Algorithm 1.

8.8. Direct Solver

To solve the local and coarse problems, and to compute the harmonic extensions, sparse direct solvers are used. Sparse direct solvers compute the solution of a linear system $Ax = b$. Therefore, the matrix A is factorized, generally by employing graph models in order to minimize the storage needed and overall work, e.g., into an LU decomposition. After the factorization, the solution is obtained via a forward and backward solve. We will give a short overview of MUMPS (MUltifrontal Massively Parallel Solver) [3, 2], which is the sparse direct solver used in this work; see also [80] for a comparison of sparse direct solvers in `FROSch`. For a comparison and review of different sparse direct solvers we refer to [71, 140]. The MUMPS user guide is available in [4]. The following three main steps are required to solve the system $Ax = b$; see [4, Section 1]:

(1) The analyzing phase includes an ordering based on the symmetrized pattern $A + A^T$, and a symbolic factorization. During the symbolic factorization, a mapping of the multifrontal computational graph, the so-called elimination tree [128], is computed and used to estimate the number of operations and memory necessary for factorization and solution. A good ordering of the unknowns produces as little fill-in as possible, which means as few additional nonzero entries that appear during the numeric factorization. In our case METIS/ParMETIS [103] is used as a package for the ordering algorithm. Then, A_{pre} is the preprocessed matrix.

(2) In the factorization phase, the numeric factorization is performed. Depending on the symmetry of A_{pre} the factorization is either $A_{\text{pre}} = LU$ for an unsymmetric matrix, or $A_{\text{pre}} = LDL^T$ for a symmetric one.

(3) In the solving phase, the forward elimination followed by backward substitution using the stored factors is performed, where x_{pre} and b_{pre} are the transformed solution x and right-hand side b, respectively, associated with the preprocessed matrix A_{pre}. For the forward elimination,

$$Ly = b_{\text{pre}} \text{ or } LDy = b_{\text{pre}},$$

and for backward substitution,

$$Ux_{\text{pre}} = y \text{ or } L^T x_{\text{pre}} = y$$

is computed. Then the solution x_{pre} is post-processed to obtain the solution x of the original system $Ax = b$.

The symbolic and numeric factorizations of the system A are included in the setup time of the preconditioner. The forward and backward solves are part of the solve time. Reusing the symbolic factorization, which determines the sparsity pattern of the matrix factors, can save setup cost, without influencing the preconditioner performance; see also Section 8.5.2. To reuse the numeric factorization can have an effect on the preconditioner performance, and, thus, is not recommended; see, e.g., [96, Section 8.2.4].

9. Test Problems

To evaluate the scalability and robustness of the proposed solvers and preconditioners for FSI applications, the fluid and structural subproblems are first considered independently. Only three-dimensional problems are considered. The well-known backward-facing step benchmark [5, 175] is used to assess the fluid solver, for example, via weak scaling studies; see Section 9.1. This inflow–outflow problem contains a pressure singularity at the step [55, Example 3.1.2], leading to flow separation and recirculation. Therefore, it captures several characteristics that are also present in more realistic flow problems.

For the structural problem, a unit cube under axial loading is used; see Section 9.2. This benchmark is also employed for weak scaling analyses and provides a controlled setting for evaluating the performance of the solid solver and preconditioner independently of fluid effects and for different material models. Moreover, this test case is used for the analysis of the deformation–diffusion problem, consisting of axial loading and simultaneously a diffusion process taking place. A simplified and idealized artery in Section 9.3 is considered to investigate qualitative results regarding the initial motivation of this work, the influence of blood pressure medication on the arterial walls. This test case serves as a proof of concept for the chemo-mechanical smooth muscle cell model (Ψ_{SMC}, Section 3.1.2) and the active material response, in a deformation–diffusion and full FSCI setting. As full FSI test cases, both a patient-specific artery (Section 9.4) and the FSI curved artery benchmark problem (Section 9.5) are examined. The FSI curved artery benchmark is used to validate the results produced by the newly developed `FEDDLib-AceGen` interface; see Section 10.2. The realistic patient-specific artery is used to evaluate the proposed solver and preconditioner configurations under realistic conditions and to assess the robustness of the findings of the isolated test cases for fluid flow and solid deformation.

The initial mesh partition defining the nonoverlapping subdomains is either structured and constructed manually (see for example in Figure 9.3 for construction of the cube), or it is constructed by METIS [103], which results in an unstructured partition of the mesh; cf. Figure 10.36. To analyze numerical and parallel scalability, we employ structured meshes for which the characteristic subdomain size H/h is kept constant, while the number of subdomains increases with the number of processor cores. For perfect scalability, the number of iterations and time-to-solution should stay asymptotically constant with an increase in the number of subdomains and processor cores.

9.1. Backward-Facing Step

The three-dimensional backward-facing step (BFS) test case is used with a structured mesh and a structured mesh partition; see Figure 9.1 (left) and (right) for the solution of a sample case. We test the weak scaling properties of the monolithic and block preconditioners for stationary and

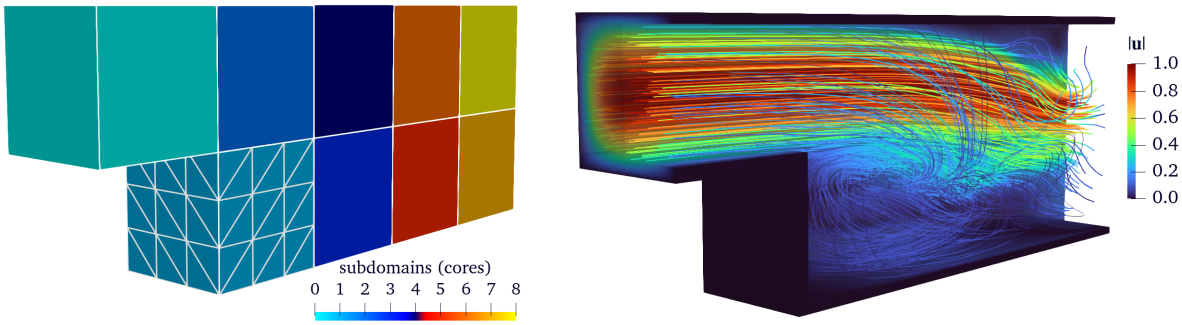


Figure 9.1: Backward-facing step (BFS) problem. **Left:** Backward-facing step geometry with structured mesh partition into 9 subdomains and $H/h = 3$. The BFS geometry is constructed by 9 subcubes, each with length 1 cm. In this graphic, the subdomains correspond to the subcubes. For higher processor core numbers, the subcubes are divided up further. For example, for $243 = 27 \cdot 9$ processor cores, each subcube holds $3 \times 3 \times 3$ subdomains. Taken from [82, Figure 6.1]. **Right:** Velocity solution in cm of transient BFS problem at $t = 10.0\,\mathrm{s}$, using time step size $\Delta t = 0.05\,\mathrm{s}$. Reynolds number is 3 200 with kinematic viscosity $\nu = 6.25 \cdot 10^{-4}\,\mathrm{cm}^2/\mathrm{s}$. P2–P1 discretization with $H/h = 9$, computed on 243 cores.

Table 9.1: Subdomain resolution H/h and number of degrees of freedom of the backward-facing step geometry for different discretizations and a varying number of subdomains or processor cores N, respectively.

		Number of degrees of freedom			
Discretization	H/h	$N = 243$	$N = 1\,125$	$N = 4\,608$	$N = 15\,552$
Q1–Q1 / P1–P1	17	$4.9 \cdot 10^6$	$22 \cdot 10^6$	$92 \cdot 10^6$	$318 \cdot 10^6$
Q2–Q1 / P2–P1	9	$4.5 \cdot 10^6$	$21 \cdot 10^6$	$85 \cdot 10^6$	$285 \cdot 10^6$
Q2–P1-disc.	8	$5.1 \cdot 10^6$	$23 \cdot 10^6$	$95 \cdot 10^6$	$320 \cdot 10^6$

transient flow: see Section 10.5. For the different finite element discretizations and subdomain resolutions, the number of degrees of freedom varies; see Table 9.1. The corresponding coarse space dimensions, shown in Figure 9.2, vary in size depending on the interface partition of unity and number of subdomains.

We prescribe a parabolic inflow profile on the inlet boundary of the domain $\partial\Omega_{\mathrm{in}}$ with a maximum velocity inflow of $v_{\max}$ and inflow height L,

$$V(0, y, z) = \left(16\, v_{\max}\, y\, z\, (L - y)\, (L - z)\right) \Big/ L^4.$$

To define the Reynolds number of the BFS problem, we use the outlet diameter of 2 cm as the characteristic length and $v_{\max}$ as the maximum inflow velocity:

$$\mathrm{Re} = \frac{2 \cdot v_{\max}}{\nu}. \tag{9.1}$$

For $v_{\max} = 1\,\mathrm{cm/s}$ and a kinematic viscosity of $\nu = 0.01\,\mathrm{cm}^2/\mathrm{s}$, the Reynolds number is 200. If not stated otherwise, this is the default configuration for the BFS problem. The CFL number in relation to the mesh resolution is shown in Table 9.2.

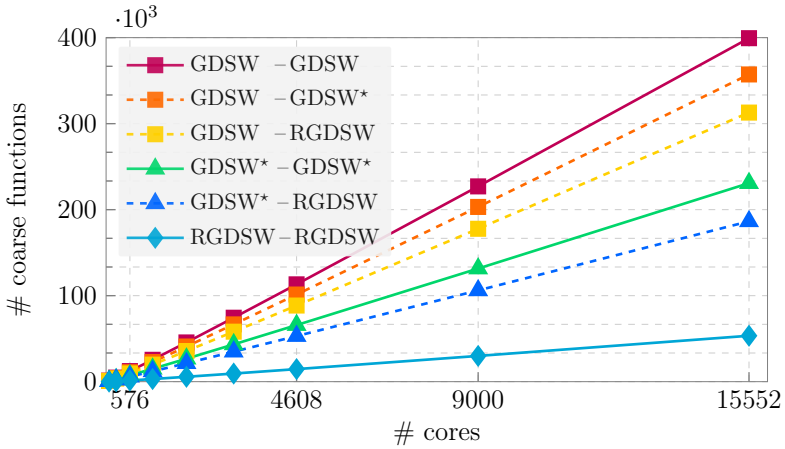


Figure 9.2: Number of coarse functions for different numbers of processor cores (which equals the number of subdomains) for the backward-facing step (BFS) problem. The first name corresponds to the interface partition of unity for the velocity component and the second for the pressure component. Adapted from [82, Figure 6.2].

Table 9.2: Transient backward-facing step (BFS) problem: CFL number c_{CFL} (see (2.17)) in relation to the element diameter h determined by $H/h = 9$ and the number of subdomains N. The time step size is $\Delta t = 0.05\,s$, the initial velocity $v_{\max} = 1\,\mathrm{cm/s}$.

# cores	243	1 125	4 608
h	1/27	1/45	1/72
c_{CFL}	1.35	2.25	3.6

9.2. Deformation–Diffusion Cube

For weak scalability tests of the solid problem, e.g., with a neo-Hookean material model, or the deformation–diffusion problem (Chapter 4), the axial pulling of a cube is considered. A voxel mesh of the unit cube is first created and subsequently refined with 5 tetrahedra per voxel; see Figure 9.3 (left). The mesh of the cube is then partitioned into subcubes as depicted in Figure 9.3 (right).

The test setup is shown in Figure 9.4. The unit cube has a side length of 1 cm. The boundary conditions for the displacement are prescribed as follows: At $(0, 0, 0)$, the cube is fixed in all directions. The three faces adjacent to $(0, 0, 0)$ are fixed in their respective normal directions. The cube is then pulled at the top face, corresponding to the $z = 1$ plane, in z direction and at the right face, corresponding to the $y = 1$ plane, in y direction. For diffusion, a Dirichlet boundary condition emulating drug inflow is prescribed at the back face, corresponding to the $x = 0$ plane. In Table 9.3 the degrees of freedom for different subdomain numbers are shown for the deformation, and the deformation–diffusion problem. The material parameters that are chosen for the corresponding tests are defined in Section 9.6.

In Figure 9.4 (right), one of the loading protocols considered in this work is illustrated. It specifies how the load is applied, consisting of a ramp phase followed by a constant load. In the

Table 9.3: Subdomain resolution H/h and number of degrees of freedom of the axial pulling of a cube geometry for the solid problem. The corresponding finite element discretization is denoted as P2, or for the deformation–diffusion problem, denoted as P2–P2. Varying number of subdomains or processor cores N, respectively.

		Number of degrees of freedom		
Discretization	H/h	$N = 216$	$N = 1\,000$	$N = 4\,096$
P2	8	$2.4 \cdot 10^6$	$11 \cdot 10^6$	$45 \cdot 10^6$
P2–P2	8	$3.2 \cdot 10^6$	$15 \cdot 10^6$	$60 \cdot 10^6$

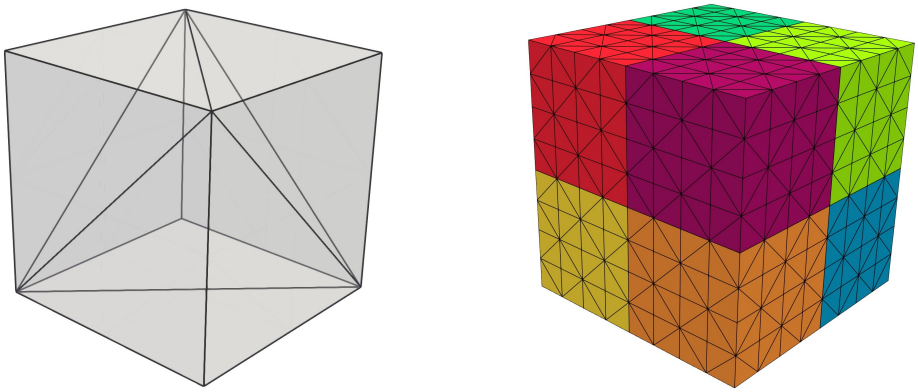


Figure 9.3: Partition of a cube into 5 tetrahedra (left) and corresponding structured partition of the unit cube into 8 subdomains, each containing $5 \cdot 4^3$ tetrahedra (right). The orientation of a partition into 5 tetrahedra needs to be switched from one cube to a neighboring cube such that a checkerboard pattern is obtained. This pattern is visible on the right by looking at the finite element edges. Based on [9, Figure 5]

example shown, the active material response is activated after 1 s, while the diffusion process is initiated after 50 s. This protocol is used for scaling analysis, whilst longer protocols, also containing heartbeats, are used to investigate the active material response and drug influence; cf. Figure 10.11 (left). In Figure 9.5 the deformed cube, the displacement, and the concentration values at $t = 100$ s are illustrated.

9.3. Idealized Artery

The idealized arterial segment in Figure 9.6 is used to illustrate qualitative results of the implemented material model with respect to the effects of blood pressure medication on the active material response. Therefore, the artery is considered for the deformation–diffusion problem as well as the full FSCI problem; see Figure 10.11 and Section 10.7. The segment has a length of 0.2 cm, wall thickness 0.12 cm and interior diameter of 0.18 cm. Wall thickness and interior diameter are chosen in a similar range to that of the realistic patient-specific artery in Section 9.4 and the plaque segment in Figure 10.43. The fluid mesh contains 1899 P1 nodes and 9024 elements, whereas the solid mesh has 3168 P1 nodes and 12 784 elements. This segment is modeled as a homogeneous material, i.e., it does not contain different regions corresponding

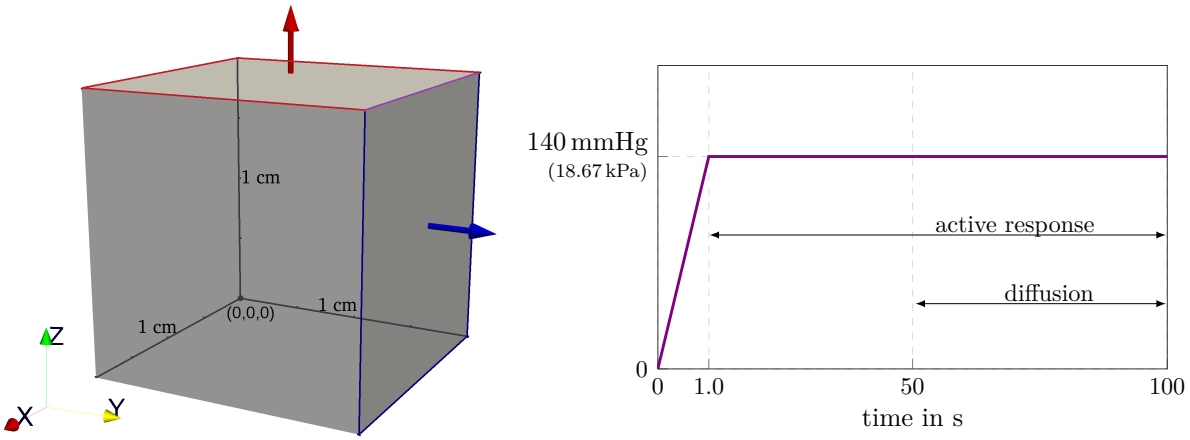


Figure 9.4: Axial pulling of a cube. **Left**: Unit cube with side length 1 cm is displayed. A pressure surface load is applied on the $z = 1$ plane and $y = 1$ plane pulling in normal directions of the respective planes. The boundary conditions for the displacement test are the following: The cube is fixed at (0, 0, 0) in all directions. The three faces adjacent to (0, 0, 0) are fixed in their respective normal directions. **Right**: Loading protocol for weak scaling with diffusion process and active response. Loading from 0 s to 1 s with 20 load steps. Active response from 1 s until the end of 100 s with $\Delta t = 0.25$ s. Diffusion starts at $t = 50$ s.

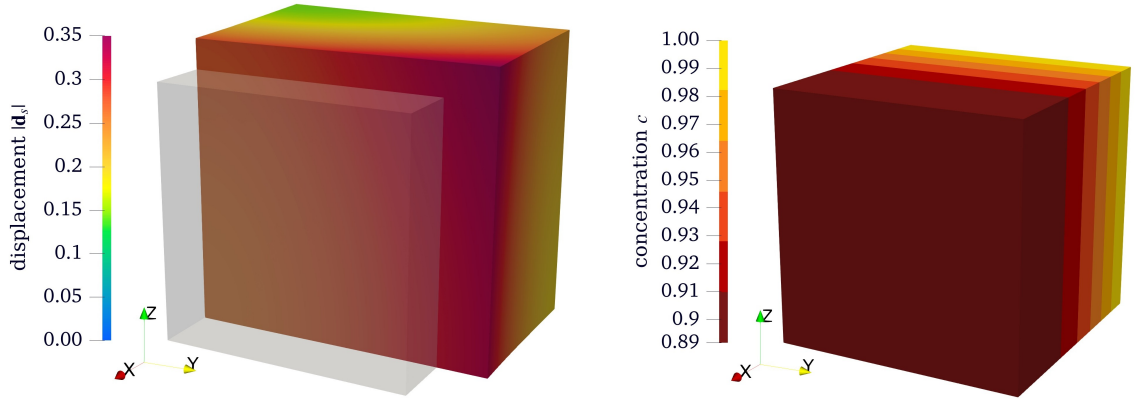


Figure 9.5: Results corresponding to the axial pulling of a cube from the test case in Figure 9.4 with the chemo-mechanical smooth muscle model Ψ_{SMC} from Section 3.1.2. **Left**: Displacement in cm at $t = 100$ s. Deformed cube shown with gray outline of reference configuration. **Right**: Concentration at $t = 100$ s.

to adventitia, media, or plaque. Consequently, the material parameters used correspond to the media, specifically *Set 1* described in Table 9.8.

9.4. Realistic Patient-Specific Artery

This patient-specific artery is used to analyze the solvers' abilities to adapt to a large range of Reynolds numbers and to complex flows. This test case consists of a realistic artery which was constructed in [7] and modified for [82, 107]. The geometry is illustrated in Figure 9.7 and is partitioned into tetrahedra. Initially, the mesh contains 24 017 fluid and 50 800 solid P1

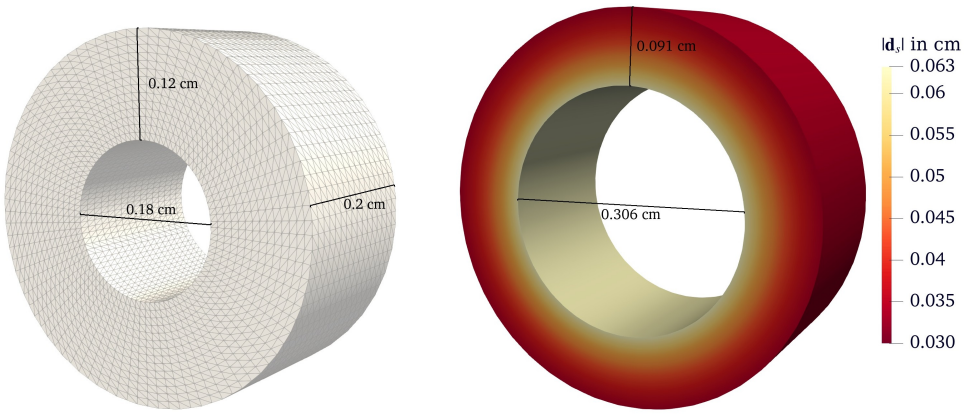


Figure 9.6: Idealized arterial segment. **Left**: Geometry in undeformed state with corresponding measurements of inner diameter 0.18 cm, wall thickness 0.12 cm, and length 0.2 cm. **Right**: Deformed geometry after ramp phase, corresponding to the settings of Figure 10.11 (left) with the chemo-mechanical smooth muscle model Ψ_{SMC} from Section 3.1.2.

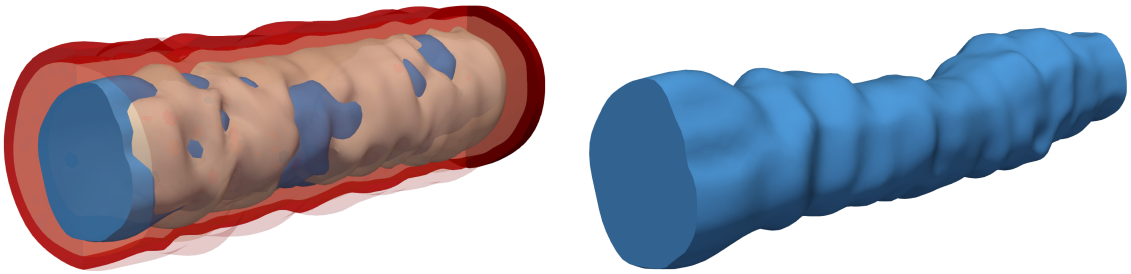


Figure 9.7: Realistic patient-specific artery with plaque layer, media, and adventitia from [7]. **Left**: Full geometry, with solid domain containing layers of adventitia (dark red), media (light red) and plaque (white). Fluid domain is shown in blue. **Right**: Only fluid domain is shown.

nodes, with 114 417 and 262 213 elements respectively. For the strong scaling tests regarding the fluid preconditioner presented in Figures 10.26 and 10.28, the fluid domain was uniformly refined, resulting in 170 000 P1 nodes and 915 336 elements. The applied boundary conditions are illustrated in Figure 9.8. Note that in purely fluid flow simulations, where only the fluid domain is used, the fluid–structure interface is treated as walls, where no-slip boundary conditions are applied. The inflow profile at the inlet $\partial\Omega_{\text{in}}$ of the domain is constructed by solving a Poisson problem on the inlet. Since the inlet is fairly round, this approximately yields a parabolic flow profile. Then, the prescribed fluid velocity or inflow rate follows a time-dependent flow profile; see for example Figure 10.26.

The interior (lumen) diameter of the artery varies between 0.2 cm and 0.34 cm across the domain. The (outer) diameter of this artery is relatively even and approximately between 0.44 cm and 0.47 cm. These values are in the range of mean diameters of renal arteries as stated in [125, Table 2]; see also [141, Sect. 3.1], where a catheter with an interior diameter of 0.2 cm was used to emulate a renal artery. Motivated by this correlation, we prescribe a flow rate and a range for the Reynolds number that is realistic for this kind of artery.

Fluid Problem The proposed blood flow rates in [141, Sect. 3.3] range between $0.1\,\mathrm{cm}^3/\mathrm{s}$ and $4.0\,\mathrm{cm}^3/\mathrm{s}$ and are constant over time. For the (not perfectly round) inlet of our geometry, these flow rate values approximately translate to a maximum inflow velocity ranging from 3.0 to 120.0 cm/s. Since the proposed artery has no fixed diameter, and the velocity field throughout the artery varies, we will estimate the maximum Reynolds number by

$$\mathrm{Re}_{\mathrm{art}}(t) = \frac{D_{\max} \cdot u_{\max}}{\nu}, \quad \mathrm{Re}_{\mathrm{art,max}} = \max_t \mathrm{Re}_{\mathrm{art}}(t), \tag{9.2}$$

where $D_{\max}$ is the maximum diameter, and $u_{\max}$ is the maximum velocity magnitude $|\mathbf{u}|$ attained in a finite element node. For instance, for the fluid simulation a kinematic viscosity of $\nu = 0.03\,\mathrm{cm}^2/\mathrm{s}$ is considered and the maximum measured arterial diameter is 0.34 cm. With a maximum inflow velocity of $v_{\max} = 40\,\mathrm{cm/s}$, which is reached after an initial cosine ramp phase of 0.1 s, the resulting maximum velocity within the artery is $u_{\max} = 112\,\mathrm{cm/s}$; this yields a maximum Reynolds number of $\mathrm{Re}_{\mathrm{art,max}} = 1\,270$. A corresponding solution is shown in Figure 10.26 (left). In Table 9.4, based on a selection of initial flow velocities, the corresponding maximum Reynolds numbers and time steps are given. This corresponds to measurements taken from the renal artery of 10 patients in [182, Table 2]. The peak Reynolds number of the patients ranged between 870 and 1 320, with an average value of 1 145.

The full (uniformly refined) fluid flow problem with a P2–P1 discretization has 4.1 million degrees of freedom. The focus of this test case is on transient flow simulations where the flow profile contains one heartbeat; see Figure 10.26 (right).

Fluid–Structure Interaction (FSI) For FSI simulations, the maximum diameter $D_{\max}$ can vary over time. Therefore, the proposed Reynolds number definition is modified to accommodate the moving fluid domain

$$\mathrm{Re}_{\mathrm{art,FSI}}(t) = \frac{\bar{D}_{\mathrm{inlet}} \cdot u_{\max}}{\nu}, \tag{9.3}$$

where $\bar{D}_{\mathrm{inlet}}$ is an averaged inflow diameter computed via the inflow area

$$\bar{D}_{\mathrm{inlet}} = 2\sqrt{A_{\mathrm{inlet}}/\pi}.$$

Note that the maximum inlet diameter is 0.32 cm in the undeformed state, which is close to the maximum arterial diameter $D_{\max}$ used for the Reynolds number calculations in the artery for the fluid simulation. Since the inlet is not perfectly round, this is only an approximation to the inflow diameter. However, the inflow area is a measure that is computed in our FSI simulations and, therefore, can be used as an indication for the Reynolds number. For the purely fluid simulation, the $\bar{D}_{\mathrm{inlet}}$ definition yields a value of 0.29 cm for inflow area $0.066\,\mathrm{cm}^2$, which is slightly smaller than the maximum inlet diameter $D_{\max} = 0.32\,\mathrm{cm}$. By this definition, the corresponding $\mathrm{Re}_{\mathrm{art,FSI}}(0.8)$ is 1 079.

For the FSI simulation, higher flow rate values are considered compared to sole fluid simulation, ranging from 3 to $6\,\mathrm{cm}^3/\mathrm{s}$. This is motivated by the settings of the curved artery benchmark Section 9.5 and to further analyze the robustness of the fluid solver. The resulting Reynolds number estimates for different material models are given in Table 10.23. At the outlet, the absorbing pressure boundary condition is applied; see (9.4).

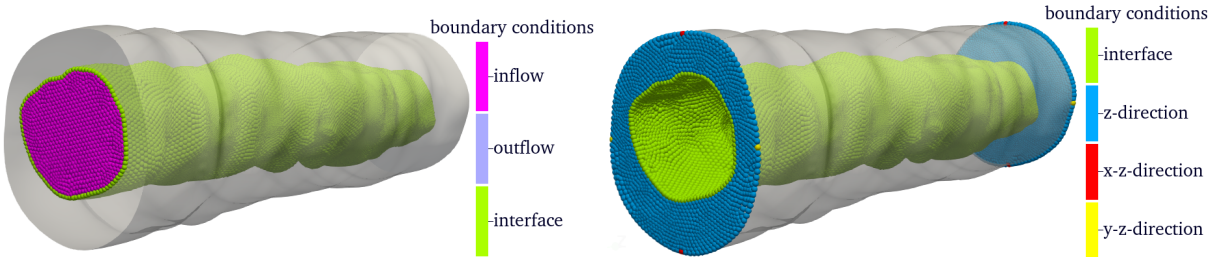


Figure 9.8: Boundary conditions of fluid and solid domain of the realistic patient-specific artery. For the fluid subproblem, a parabolic-like inflow profile is prescribed on the inlet (in pink). On the outflow, the absorbing pressure boundary condition is prescribed. For the solid subproblem, the inflow and outflow planes lying in the z-plane, illustrated in blue, are held in z direction. Additional points on the respective inflow and outflow planes are held in x-z or y-z direction to prevent rigid body motions to ensure that the system is statically determinate. Taken from [107].

Table 9.4: Detailed configuration for the realistic artery problem for fluid simulations. Configurations for the maximum initial inflow velocity, the time step size, and approximate maximum Reynolds number. The maximum Reynolds number for this test is defined as $\mathrm{Re}_{\mathrm{art,max}}$ in (9.2). Taken from [82, Table 6.2].

$v_{\max}$	10 cm/s	20 cm/s	30 cm/s	40 cm/s
Δt	0.005 s	0.0025 s	0.001875 s	0.00125 s
$\mathrm{Re}_{\mathrm{art,max}}$	345	665	970	1 270

9.5. FSI Curved Artery Benchmark

The curved artery benchmark introduced in [8] and later revisited in [10] is considered to validate the `FEDDLib-AceGen` interface (Section 8.3) against an existing benchmark problem. The same geometry, material parameters, and boundary conditions as in the reference studies are employed; see Figure 9.9 and Table 9.5. In [8], several mesh resolutions were investigated to study the mesh convergence behavior of different constitutive models. It was shown that Mesh #3 provides mesh-converged results, that is, a finer mesh discretization provided the same qualitative solution, and therefore this discretization is considered in the present work. The same mesh was also used in [10]. The geometry consists of a curved cylindrical tube with inner and outer radii of 0.15 cm and 0.21 cm, respectively, resulting in a wall thickness of 0.06 cm. Both the radius of curvature and the length of the straight tube sections are equal to 1 cm. The fluid domain is driven by a time-dependent parabolic inflow profile prescribed at the inlet; see Figure 9.10 (right). At the outlet, the absorbing pressure boundary condition (9.4) is applied. The solid is constrained at the inlet in z-direction and at the outlet plane in x-direction. Additional point constraints in the y-z and x-y directions on the inlet and outlet plane are imposed to prevent rigid body motions and to ensure a statically determinate system; see Figure 9.9 (right). For the structural discretization, a P2 finite element formulation is employed. In contrast, [8, 10] used $\bar{F}$ elements for the solid problem. These elements are based on a mixed formulation and are specifically designed to alleviate incompressibility-related locking effects for the Ψ_A constitutive model. For the Ψ_A model, anisotropic fiber directions are additionally considered; see Section 3.1.3. Due

Table 9.5: Material parameters for the FSI curved artery benchmark problem. The parameter sets for the Ψ_A and *NH3*, *NH4* models are adopted from [8, 10]; see Section 3.1 for the definition of the wall models. Note that the Ψ_A model corresponds to the definition of *Set 2* in Table 9.8 and *NH3* corresponds also to Table 9.7.

Ψ_A						*NH3*	NH4
α_1	α_2	α_3	α_4	α_5	f_A	E	E
17.5 kPa	499.8 kPa	2.4	30 001.9 kPa	5.1	43	380 kPa	440 kPa

to the curved geometry, the definition of the fiber orientation field is nontrivial. The definition proposed in (3.15) constructs the fiber angles in the circumferential plane, which corresponds to the x-y-plane in our cases as the axial direction is defined as z-direction. Since the geometry is curved, the definition cannot be applied and therefore a custom definition is used for this geometry, resulting in the fiber directions in Figure 9.10 (left). This results in the geometry-specific fiber arrangement proposed in [8, 10].

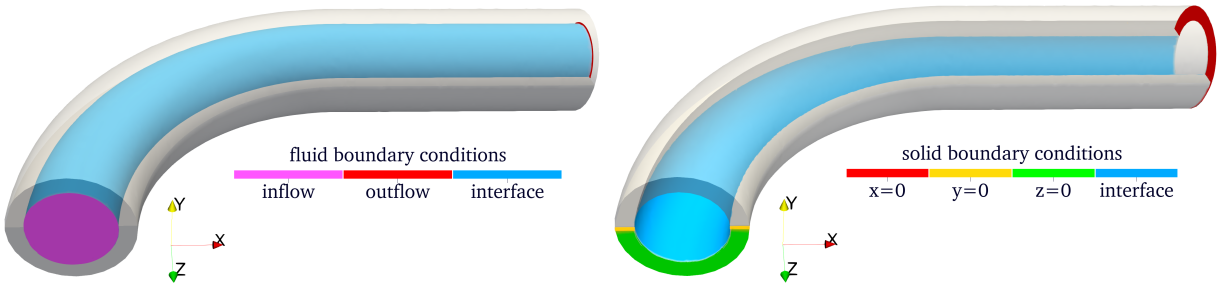


Figure 9.9: Boundary conditions proposed for the FSI curved artery benchmark from [8, 10] for the fluid subproblem (left) and solid subproblem (right). A parabolic inflow profile is prescribed on the inflow area, the absorbing pressure boundary condition is prescribed at the outflow of the fluid domain; cf. (9.4). The solid is constrained at the inlet in z-direction and at the outlet plane in x-direction. Additional point constraints in the y-z and x-y directions, illustrated as yellow strips on the inlet and outlet plane, are imposed to prevent rigid body motions and to ensure a statically determinate system.

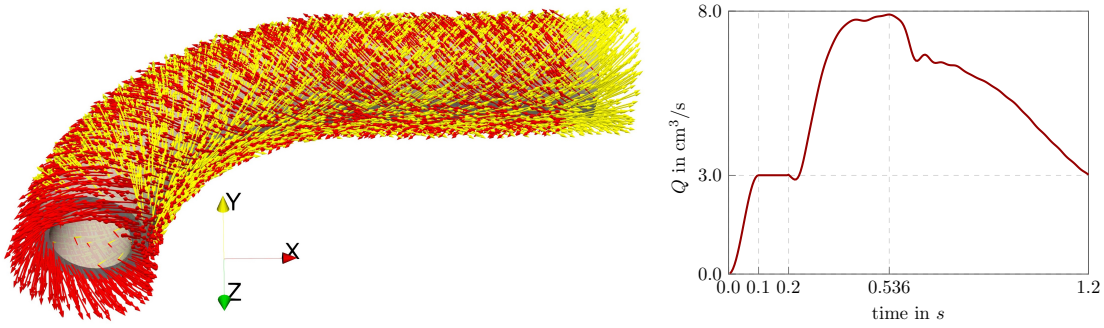


Figure 9.10: Fiber directions (left) and imposed inflow profile over time (right) for the FSI curved artery benchmark from [8, 10]. The two helically arranged fiber families with a 43° angle are used; see Section 3.1.3.

9.6. Boundary Conditions

In this section, specific boundary conditions used in the computation are described. In Section 9.6.1 deformation dependent loads are described, which are applied in the solid simulations to apply pressure loads. Note that surface loads are applied via a linear ramp, while the ramp phase considered in fluid and FSI simulations is based on a cosine ramp; cf. [8, Section 4.2] for an investigation of different inflow ramps. Fluid boundary conditions for the inflow and outflow are described in Sections 9.6.2 and 9.6.3.

9.6.1. Deformation Dependent Loads

The surface load of a boundary element T in the reference configuration is given by

$$\int_{\partial T} p\, \mathbf{n}_0 \cdot \boldsymbol{\varphi}\, \mathrm{d}S,$$

with outward surface unit normal vector $\mathbf{n}_0$ and prescribed surface pressure value p; see the definition of the traction in (3.18). This load is evaluated on the fixed reference geometry. Hence, the normal vector and the surface measure $\mathrm{d}S$ are independent of the deformation. Opposed to that, the deformation-dependent pressure load is evaluated on the current, deformed surface [179, Chapter 4.2.5]. Consider the mapping of the deformation from the reference to the current configuration (see also (3.1))

$$\mathbf{x} = \mathcal{A}(\mathbf{X}, t) = \mathbf{X} + \mathbf{d}_s(\mathbf{X}, t).$$

The corresponding pressure load is then defined by

$$\int_{\partial T_t} p\, \mathbf{n}_t \cdot \boldsymbol{\varphi}_t\, \mathrm{d}s,$$

where $\partial T_t = \mathcal{A}(\partial T)$ represents the deformed boundary. Consequently, both the current normal $\mathbf{n}_t$ and the current surface measure $\mathrm{d}s$ depend on the current nodal positions. Let ξ and η be the natural coordinates of the reference element and let $\mathbf{x}(\xi, \eta, t)$ be a parametrization of the deformed element. For the parametrized surface element the dependence between outward normal and parameterization can be written explicitly as $\mathbf{n}_t\, \mathrm{d}s = \mathbf{x}_{,\xi} \times \mathbf{x}_{,\eta}\, \mathrm{d}\xi\, \mathrm{d}\eta$, which yields

$$\int_{\partial T_t} p_t\, \mathbf{n}_t \cdot \boldsymbol{\varphi}\, \mathrm{d}s = \int_{\partial T} p\, (\mathbf{x}_{,\xi} \times \mathbf{x}_{,\eta}) \cdot \boldsymbol{\varphi}\, \mathrm{d}\xi\, \mathrm{d}\eta.$$

Since $\mathbf{x}$ depends on the nodal displacements, the resulting load vector is deformation-dependent and has to be linearized for a Newton scheme; compare to the derivations in [179, Pages 101 and 143]. The definition and linearization are performed in `AceGen` and the corresponding residual vector and Jacobian contributions are added in through the `FEDDLib-AceGen` interface.

9.6.2. Inflow Boundary

The inflow boundary conditions for the fluid can be prescribed by solving a Laplace problem on the fluid domain, and using the solution on the inlet of the fluid domain to define a parabolic-like

inflow profile. The problem setting is described by

$$\begin{aligned} \Delta u_L &= 0 \quad \text{on } \Omega_f, \\ u_L &= 0 \quad \text{on } \Omega_\Gamma. \end{aligned}$$

Then, all degrees of freedom except on the inlet of the fluid domain are set to zero, and the inlet values are normalized to one, to obtain a normalized inflow profile. The resulting solution vector is used to prescribe the velocity at the inlet with a parabolic-like profile. This is especially useful when the inlet is not round, e.g., for the realistic artery from Figure 9.7. In fluid simulations the interface corresponds to the walls; see (2.3) for the fluid boundary definitions.

9.6.3. Outflow Boundary

In order to properly model pressure behavior in FSI and FSCI in the arterial walls, we have to apply outflow boundary conditions to prescribe pressure values. In this work, we consider the absorbing boundary condition proposed in [136]. This is also motivated by the FSI curved artery benchmark, where this boundary condition was applied [8, 10].

We impose a modified version of the absorbing boundary condition on the outlet of the fluid proposed in [136, Section 6]. The boundary condition is based on a lower dimensional linear elastic model. The goal is to remove or decrease reflections; see for example [8, Section 4.1.5] for further discussion and information. At time step $n+1$ the boundary condition is imposed explicitly by

$$\sigma_f^{n+1} \cdot \mathbf{n}_f|_{\Gamma_{\text{out}}} = p_{\text{out}}^{n+1} \mathbf{n}_f|_{\Gamma_{\text{out}}}$$

with

$$p_{out}^{n+1} = \left(\frac{\sqrt{\rho_f}}{2\sqrt{2}} \frac{Q_{out}^n}{\bar{A}} + \sqrt{\frac{d\,E}{1-\nu^2} \frac{\pi}{A_0} \sqrt{A_0}} \right)^2 - \frac{d\,E}{1-\nu^2} \frac{\pi}{A_0} \sqrt{A_0} + p_{\text{ref}}, \tag{9.4}$$

with Young's modulus E, outflow area A_0 and diameter d of the arterial wall in its reference configuration, Poisson's ratio ν, and reference fluid pressure p_{ref}. The reference pressure value serves as an initial pressure value which is reached after a ramping phase. In our setting $\bar{A}$ is computed from (9.4) by imposing a steady-state condition, that is $p_{out}^{n+1} = p_{\text{steady}} = p_{\text{ref}}$. Therefore, the absorbing boundary condition is modified in the ramping phase, to achieve exactly the reference pressure value. In the then so-called unsteady phase, when heartbeats or active material response occur, the pressure is then influenced by flow rate and outflow area, as $\bar{A}$ is set to A_{out}^n, the current outflow area in time step n. The parameters used for (9.4) are described in Table 9.6; they are specified for the FSI curved artery benchmark from Section 9.5. For the other considered tests, the diameter or thickness of the arterial wall and the reference pressure have to be specified. Using the inflow rate Q_{in}^n instead of Q_{out}^n in (9.4) further stabilizes the boundary condition in the ramp phase. This reduces oscillatory behavior but only has a small effect on the flow rate, since we only consider one inflow and one outflow, resulting in $Q_{\text{in}}^n \approx Q_{\text{out}}^n$.

Table 9.6: Parameter settings regarding the absorbing boundary condition in (9.4) for FSI curved artery benchmark Section 9.5. Young's modulus E, area A_0 and diameter d of the arterial wall in its reference configuration, Poisson's ratio ν, and reference fluid pressure p_{ref}.

E in kPa	A_0 in cm^2	d in cm	Poisson's ratio	p_{ref} in kPa
120	0.07	0.06	0.49	10.66

9.6.4. Discussion of Outflow Boundary Conditions

On the outlet of the fluid domain, a pressure boundary condition is prescribed. This pressure determines the internal pressure acting on the arterial wall and therefore influences the deformation of the artery. A constant outlet pressure can be prescribed. However, in the present work blood flow with heartbeats is considered, such that the outlet pressure should respond to the transient flow conditions. Several approaches have been proposed to prescribe physiologically realistic outlet pressure conditions. These approaches account for quantities such as the vascular resistance, flow rate, outflow area, and other properties of the vascular system outside the computational domain, as in Windkessel models.

In resistance boundary conditions [171], the outflow pressure is based on a linear relationship between the computed outflow rate and the prescribed resistance. For example, in [180, 181] the resistance boundary condition is applied in two- and three-dimensional FSI simulations, where multiple outlets are considered.

The absorbing boundary condition can be interpreted similarly, where instead of a resistance value, wave impedance is considered; see also [171] for a comparison between resistance and impedance. It tells you how much pressure is required to generate a given flow wave without producing reflections. This impedance value is based on a one-dimensional linear elasticity model; a modified version of the boundary condition, which is considered in this work, is based on [136, Section 6]. In contrast to the resistance boundary condition in [171], the considered absorbing boundary condition depends on the current deformed outlet area through the defined wave impedance; see (9.4). Consequently, the prescribed outlet pressure varies not only with the flow rate but also with changes of the outlet area.

In Windkessel models, the outflow pressure is related to the resistance and compliance of the arterial system. There, the compliance describes how much the artery can expand and store blood. A comprehensive review of Windkessel models is given in [172]. The first formulation of the two-element Windkessel model was presented by Frank [65, 66]. A more commonly used model in cardiovascular simulations is the three-element Windkessel model, which additionally includes a proximal resistance representing the characteristic impedance of the large arteries. In [146], fixed, resistance and Windkessel outflow pressure boundary conditions are discussed and compared for three-dimensional Navier–Stokes problems.

For the considered FSI simulations, an inflow rate is prescribed, which follows a fixed flow profile, e.g., as shown in Figure 9.10. Therefore, a resistance type boundary as defined in [171] with a constant resistance would not reflect pressure changes, due to changes of the vessel radius. The absorbing boundary condition accounts for this effect, as the outlet pressure depends on both the flow rate and the current deformed outlet area. For this reason, it provides a suitable outlet

Table 9.7: Material parameters for the neo-Hookean material model. The parameters are adopted from [10, Table 2].

		κ	μ	E
NH3	Media	6 333.3 kPa	127.52 kPa	380 kPa

model for the presented FSI simulations, and can reflect the pressure changes that arise from active material response causing contractions. This result can be seen in Figure 10.39, where the artery contracts over time and therefore an increasing pressure value is computed on the outlet.

The objective of this work is to investigate the parallel performance of the proposed two-level overlapping Schwarz preconditioners. In addition, the influence of different arterial wall models, material parameters, and active arterial wall mechanics on the local pressure and the deformation of the artery is investigated, with respect to the solver performance. Since the response of the downstream vascular system is not modeled explicitly, an absorbing outlet boundary condition that accounts for the current deformed outlet geometry is considered sufficient for the present simulations.

9.7. Parameter Sets for the Passive Material Response

In this work, different material parameter sets are tested for the modeling of the arterial wall. The parameter settings for the neo-Hookean material model are denoted as the *NH3* set, which is adopted from [8, 10]. For the models in Sections 3.1.2 and 3.1.3, different parameter sets are tested for the passive material response; cf. (3.6).

There is only one set of parameters that is used for the active material response which is included in the Ψ_{SMC} models. They are listed in Tables A.1 and A.2. In particular, the Ψ_A and the Ψ_{SMC} models are considered (Sections 3.1.2 and 3.1.3). The Ψ_A model only contains passive material response, which is identical to the passive response in the Ψ_{SMC} model; compare (3.6) and (3.14). If a material is modeled homogeneously, we use parameters corresponding to the media of an artery. The tunica media is the middle layer of the arterial wall and consists mainly of smooth muscle cells which are responsible for the active regulation of the vessel diameter. Note that the active material response is only present in the media, or degenerated media; see the decomposition of an arterial segment shown in Figure 10.43 (left).

The passive material parameters in [9], which were adopted from [168], correspond to the media layer and were used for the chemo-mechanical SMC model Ψ_{SMC}. This parameter set is denoted as *Set 1*. For the curved artery benchmark [8, 10], the considered parameters for the Ψ_A model were fitted in [27] to represent the media as well. The parameter set will be denoted as *Set 2*. For both parameter sets α_1, α_3, and α_5 are in a similar range. For parameter *Set 2*, α_2 is three times higher compared to *Set 1*, and α_4 is 23 000 times higher. This means for *Set 2* that it resists stronger against volumetric changes, i.e., stronger near-incompressibility, and is stiffer with respect to the fiber-related anisotropic part, which is determined by α_4.

For the patient-specific artery from Section 9.4, different layers of the arterial wall are identified, the adventitia, media and plaque. There, the plaque is represented as a homogeneous material. In reality, plaques generally consist of multiple components, such as a lipid pool, a fibrous

Table 9.8: Material parameters for the Ψ_A, i.e., the passive part of the Ψ_{SMC} material model. The parameters from *Set 1* are adopted from [9, Table 2], *Set 2* from [10, Table 3].

		α_1	α_2	α_3	α_4	α_5	f_A
Set 1	Media	11.5 kPa	151.7 kPa	2.8	1.3 kPa	3.1	30
Set 2	Media	17.5 kPa	499.8 kPa	2.4	30 001.9 kPa	5.1	43

Table 9.9: Material parameters for the Ψ_A material model for the realistic patient-specific artery with parameters for the different components defined as *Set 3*. Material parameters adapted from [149, Table 1]. Plaque component is modeled as fibrous cap.

Set 3	α_1	α_2	α_3	α_4	α_5
Adventitia	11.1 kPa	151.7 kPa	2.8	370.4 kPa	4.0
Media	11.5 kPa	151.7 kPa	2.8	1.3 kPa	3.1
Plaque	9.3 kPa	151.7 kPa	2.8	73 360.3 kPa	2.5

cap, and calcifications; cf. Figure 10.43 (left). Thus, the choice of material parameters for the plaque component is not straightforward. Calcifications are typically the stiffest components of atherosclerotic plaques; however, they usually represent only a small part of the overall plaque volume. In contrast, the fibrous cap can cover larger regions of the plaque and separates the plaque from the lumen, i.e., the fluid domain. Modeling the entire plaque as calcifications would significantly increase its stiffness and therefore substantially reduce the wall deformation induced by the fluid pressure. Therefore, the plaque is modeled by the parameters defining the fibrous cap, which is defined in Table 9.9 as *Set 3*; the material parameters are adopted from [149, Table 1]. Since determining accurate or optimal plaque parameters is beyond the scope of this work, and since the primary objective is to analyze solver performance in a more realistic setting, only this one representative case is considered. The deformation of the realistic patient-specific artery for the different parameter settings is illustrated in Figures 10.34 and 10.35.

In Section 10.7.1, an arterial segment is considered, which contains distinct regions for different plaque components, the fibrous cap, calcifications, and the lipid pool. The corresponding parameters and geometry are adapted from [149, Table 1]. The parameters for the passive material response are denoted as *Set 4*. Note that for the test case considered in Section 10.7.1, the active response parameters were slightly adjusted compared to Tables A.1 and A.2; see [149] for details. Note that the parameters for the passive material response for the media in Tables 9.9 and 9.10 correspond to *Set 1*.

Table 9.10: Passive material parameters for the Ψ_{SMC} material model for the arterial segment with distinct plaque regions with parameters for the different components defined as *Set 4*. Material parameters adapted from [149, Table 1].

Set 4	α_1	α_2	α_3	α_4	α_5
Adventitia	11.1 kPa	151.7 kPa	2.8	370.4 kPa	4.0
Media	11.5 kPa	151.7 kPa	2.8	1.3 kPa	3.1
Degenerated Media	13.8 kPa	151.7 kPa	2.8	3.2 kPa	3.1
Fibrous Cap	9.3 kPa	151.7 kPa	2.8	73 360.3 kPa	2.5
Calcification	11 520 kPa	151.7 kPa	2.8	–	–
Lipid Pool	1.3 kPa	151.7 kPa	2.8	–	–

10. Numerical Results

In this chapter, the preconditioners introduced in Chapters 6 and 7 are tested and analyzed using the software framework presented in Chapter 8. The primary focus is on parallel scalability and robustness. To this end, both weak and strong scaling are investigated. In weak scaling, the ratio of the subdomain size to the mesh size, denoted by H/h, is kept constant while the number of processor cores is increased; see Table 9.1, where the number of degrees of freedom for the backward-facing step problem is shown for an increasing number of subdomains. Ideally, the preconditioner and the iterative solver exhibit good weak scalability, such that both the average number of linear iterations per Newton step (and time step) and the time-to-solution, consisting of setup and solve time, remain approximately constant. In strong scaling, the total problem size is fixed while the number of subdomains (or processor cores) is increased. Good strong scalability is achieved if the time-to-solution decreases with an increasing number of processor cores. The numerical results are organized as follows.

First, the newly developed `FEDDLib-AceGen` interface is verified and validated in Section 10.2. The material models implemented through the interface are investigated in Sections 10.3 and 10.4. The proposed overlapping Schwarz preconditioners are evaluated with respect to weak and strong scaling. For the weak scaling study, the axial pulling of the unit cube introduced in Section 9.2 is considered. The strong scaling tests are performed using the patient-specific artery; see Section 9.4. In Section 10.5, monolithic and block overlapping Schwarz preconditioners are compared for the incompressible Navier–Stokes equations. For both approaches, weak scaling is investigated using the backward-facing step problem from Section 9.1. In addition, the robustness of both preconditioning strategies is analyzed for a range of Reynolds and CFL numbers. Strong scaling is again evaluated using the patient-specific artery.

The best-performing preconditioning and solver settings for solid and fluid are subsequently employed for the fluid–structure interaction problem presented in Section 10.6. Moreover, the idealized arterial segment introduced in Section 9.3 is used to demonstrate the active material response and its interaction with the drug concentration for both the deformation–diffusion problem and the pharmaco–mechanical FSI (FSCI) problem discussed in Section 10.7.

The parallel results were obtained on the Fritz supercomputer at Friedrich-Alexander-Universität Erlangen-Nürnberg. Fritz has 992 compute nodes, each with two Intel Xeon Platinum 8360Y Ice Lake processors and 256 GB of DDR4 RAM.

10.1. Settings for Solve and Setup of Preconditioner

This section summarizes the general settings used for the solution of the nonlinear systems and the construction of the preconditioners. Unless stated otherwise, these settings are employed throughout the numerical experiments.

10.1.1. Inexact Newton Method

The governing equations considered in this work are nonlinear and are therefore solved using Newton's method. More precisely, the linear systems arising in each Newton iteration are solved approximately. The Newton method with inexact solutions of the linearized system is called the inexact Newton method; see [49]. If the inexact solution is determined by a Krylov subspace method, it is called a Newton–Krylov method. As a Krylov subspace method, we employ the generalized minimal residual (GMRES) method [155]. With an initial guess for δX^k, we use GMRES to solve

$$\left\| \mathcal{J}(X^k)\delta X^k + \mathcal{R}(X^k) \right\| \leq \eta_k \|\mathcal{R}(X^k)\|, \quad \eta_k \in [0, 1), \tag{10.1}$$

and update the Newton iterate via $X^{k+1} = X^k + \delta X^k$. The linear solver tolerance η_k can be constant throughout the Newton iterations or change between consecutive iterations. In the second case, η_k is referred to as a forcing term [50], which can change depending on the underlying forcing term strategy. We use the *Choice 2* forcing term from [143],

$$\eta_k = \gamma \left(\frac{\|\mathcal{R}(X^k)\|}{\|\mathcal{R}(X^{k-1})\|} \right)^{\alpha}, \tag{10.2}$$

with $\gamma \in [0, 1]$ and $\alpha \in (1, 2]$. To make use of this strategy, we select a $\eta_{\text{init}} = \eta_0 \in [0, 1)$ for the first iteration. We impose a lower and upper bound for the forcing term by

$$\max \left\{ \gamma \eta_{k-1}^{\alpha}, \eta_{\min} \right\} \leq \eta_k \leq \eta_{\max}, \tag{10.3}$$

with prescribed bounds $\eta_{\max}$ and $\eta_{\min}$. This forcing term strategy is referred to as *Type 2* in this work, which corresponds to the `NOX` parameter name. The constants are set to $\alpha = 1.5$ and $\gamma = 0.9$.

For the Newton method in `NOX`, we employ a line search with backtracking as the globalization strategy; see [96, Section 5.4] for more details and [49] for a full derivation of the method and globalization techniques. We consider the starting step length of 1 and the maximum number of steps is set to 5. The other required parameters are left at their respective default values; see the `NOX` documentation [164]. The initial guess for Newton's method is $X^0 = 0$. For time-dependent problems this refers only to the first time step.

10.1.2. Solver Settings

In Table 10.1 the default solver settings are shown. With respect to the adaptive forcing term presented in Section 10.1.1, the minimum and maximum tolerances prescribed for the GMRES solver can be adjusted. The choice of these bounds requires balancing the cost of the linear and nonlinear solves. Increasing the tolerances reduces the computational effort required to solve the linear systems and, consequently, the average number of GMRES iterations per Newton step. However, the resulting lower accuracy of the linear solve may increase the number of Newton iterations required for convergence.

Throughout this work, the default parameter values for the *Type 2* adaptive forcing term are used, as they provide a good compromise between the cost of the linear and nonlinear

Table 10.1: Solver settings for the nonlinear and linear solvers.

Newton	update tol. 10^{-8}	relative tol. 10^{-8}	max. iter. 20
GMRES	constant forcing term	10^{-6}	max. iter. 1 000
	Type 2 forcing term	$\eta_{\min} = 10^{-8}$, $\eta_{\text{init}} = \eta_{\max} = 10^{-3}$	max. iter. 1 000

iterations. In particular, they reduce the computational effort of the linear solver without noticeably increasing the number of Newton iterations. For a comparison of different forcing term parameters for the solid problem, see Table 10.7.

Newton's method terminates when the relative residual $\| \mathcal{R}(X^{k+1})\|_2 / \| \mathcal{R}(X^0)\|_2$ or the Newton update $\|X^{k+1} - X^k\|_2$ reaches a certain tolerance, respectively.

GMRES is used without restarts. The maximum dimension of the Krylov subspace is set to 1 000. The preconditioners are always applied as right preconditioners. We choose an algebraically determined overlap of $\hat{\delta} = 1$ for the construction of the local overlapping stiffness matrices; see Section 8.5.1 for a review of the algebraic construction in `FROSch`.

The problems of the first level and the coarse problem of the Schwarz preconditioner are solved sequentially; for example, the N subdomain problems of the first level are solved on N processor cores, and subsequently a subset of these cores is used to solve the coarse problem in parallel. We do not employ a hybrid Schwarz approach but use a fully additive one; see Chapter 7. As discussed in Section 8.5.2, components of the preconditioner can be reused after they were constructed in the first Newton iteration. In all simulations, the symbolic factorizations of the local subdomain matrices and the coarse matrix are reused. As explained in Section 8.8, reusing the symbolic factorization reduces the setup time without affecting the performance of the preconditioner.

10.2. Validation of FEDDLib–AceGen Interface

The `FEDDLib-AceGen` interface was developed though a collaboration involving the research groups of Axel Klawonn (University of Cologne) and Daniel Balzani (Ruhr University Bochum), together with Jascha Knepper (University of Cologne), Alexander Heinlein (TU Delft), and Oliver Rheinbach (TU Bergakademie Freiberg). The main development work was carried out by Lea Saßmannshausen and Sharan Nurani Ramesh, with contributions from all other collaborators. In [138] the interface was first used, and further tested in [9]. The following description is part of a publication, which is currently in preparation [11].

In order to integrate the chemo-mechanical smooth muscle cell arterial wall model described in Section 3.1.2 into FSI and FSCI simulations, an `FEDDLib-AceGen` interface was established. In the following section the interface is reviewed with respect to different aspects. The different test cases are shown in Figure 10.1. For test cases (1) - (3), the deformation cube — the axial pulling of a unit cube — from Section 9.2 is considered, with surface load of 16 kPa (120 mmHg) and Young's modulus 380 kPa. In (1), the interface is compared to a manual `FEDDLib` implementation for a linear elasticity problem; see Table 10.3. In (2), the interface results are validated against an earlier `FEDDLib` implementation of the neo-Hookean material model, which was also generated

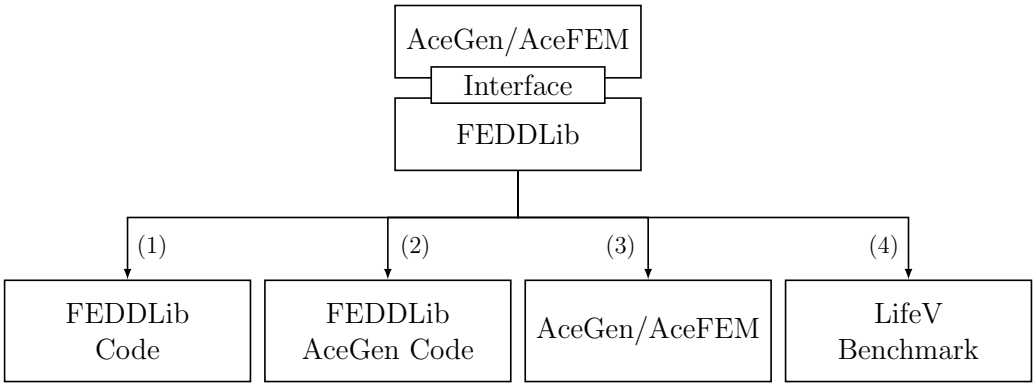


Figure 10.1: Structure of different interface validation tests. The `FEDDLib-AceGen` interface is validated against four different pipelines. (1) `FEDDLib` with its own implementations (linear elasticity). (2) `FEDDLib` with integrated several-years old `AceGen` code; see [96] for results using the routines. (3) Results generated with `AceFEM`. (4) Results from [8, 10] for a benchmark computed with LifeV; see Section 9.5.

using `AceGen`; results using the earlier neo-Hookean implementation are shown in [96]. Although both computations rely on `AceGen`-generated code, the routines used by the `FEDDLib-AceGen` interface and by the integrated `FEDDLib` implementation were generated independently, at different times, and employ different assembly functions. The results for the neo-Hookean material model are shown in Table 10.3. There, the error between the two resulting solutions is compared, and the assembly routines' runtime is evaluated. For (3), the results of the `FEDDLib-AceGen` interface are compared to results from an `AceFEM` computation; see Table 10.4. Finally in (4), the results using the interface for a full FSI implementation are compared to the curved artery benchmark from [8] and [10]. In order to compare the results, the absolute and relative errors are defined as

$$\begin{aligned} \varepsilon_{\text{abs}}(\mathbf{d}_s) &= \|\mathbf{d}_{s,2} - \mathbf{d}_{s,1}\|_{l^\infty}, \ \varepsilon_{\text{abs}}(c) = |c_2 - c_1|_{l^\infty}, \\ \varepsilon_{\text{rel}}(\mathbf{d}_s) &= \frac{\|\mathbf{d}_{s,2} - \mathbf{d}_{s,1}\|_{l^\infty}}{\|\mathbf{d}_{s,1}\|_{l^\infty}}. \end{aligned} \tag{10.4}$$

For test (1), the errors between the respective implementations range from $1.1 \cdot 10^{-13}$ to $3.0 \cdot 10^{-11}$ and are therefore close to machine precision; compare Table 10.2. For test (2), the errors between the respective implementations are in the order of machine precision; see Table 10.3.

In Table 10.3 additionally the runtimes of the respective assembly routines are compared. Both deliver comparable results. Note that the increased assembly time observed for larger numbers of processor cores is mainly associated with the creation of the final communication pattern. For this step, the `fillComplete()` function of `Trilinos` is called. This operation completes the graph structure, combines matrix entries that have been inserted by multiple processors during the finite element assembly, and establishes the communication patterns required for distributed sparse matrix-vector products. The cost of this finalization step increases with both the number of processor cores and the number of degrees of freedom, and can therefore become increasingly costly in large-scale parallel simulations.

In test (3), the interface results are compared against the `AceFEM` simulation — based on the same `AceGen` code that is included in the interface — with the test case shown in Figure 9.4. Therefore, the simple deformation–diffusion model introduced in Section 4.1 is considered. Al-

Table 10.2: Comparison between the original `FEDDLib` implementation and new interface implementation for a linear elasticity material model. The full solution vector is compared with respect to maximum absolute error $\varepsilon_{\text{abs}}(\mathbf{d}_s)$ and relative error $\varepsilon_{\text{rel}}(\mathbf{d}_s)$ between the solutions. The test setup corresponds to Figure 9.4 with applied surface force of 16 kPa (120 mmHg). The cube is based on a structured mesh partition with $H/h = 8$. We consider different numbers of cores to verify the implementation.

N (dofs)	$\varepsilon_{\text{abs}}(\mathbf{d}_s)$	$\varepsilon_{\text{rel}}(\mathbf{d}_s)$
216 ($2.4{\cdot}10^6$)	$1.1{\cdot}10^{-13}$	$2.8{\cdot}10^{-12}$
1 000 ($1.1{\cdot}10^7$)	$3.3{\cdot}10^{-13}$	$8.0{\cdot}10^{-12}$
4 096 ($4.4{\cdot}10^7$)	$1.2{\cdot}10^{-12}$	$3.0{\cdot}10^{-11}$

Table 10.3: Comparison between the previously existing `FEDDLib` implementation and new interface implementation for neo-Hookean material model. The full solution vector is compared with respect to maximum absolute error $\varepsilon_{\text{abs}}(\mathbf{d}_s)$ and relative error $\varepsilon_{\text{rel}}(\mathbf{d}_s)$ between the solutions. The test setup corresponds to Figure 9.4 with applied surface force of 16 kPa (120 mmHg). The cube is based on a structured mesh partition with $H/h = 8$. We consider different numbers of cores to verify the implementation. 4 Newton iterations are required to reach relative tolerance of 10^{-8}. To determine the assembly runtime we averaged 3 consecutive runs.

N (dofs)	time for assembly (`FEDDLib`–`AceGen`)	time for assembly (`FEDDLib`)	$\varepsilon_{\text{abs}}(\mathbf{d}_s)$	$\varepsilon_{\text{rel}}(\mathbf{d}_s)$
64 ($7.3{\cdot}10^5$)	2.1 s	2.3 s	$2.8{\cdot}10^{-16}$	$6.6{\cdot}10^{-15}$
125 ($1.4{\cdot}10^6$)	2.3 s	2.3 s	$2.9{\cdot}10^{-16}$	$7.0{\cdot}10^{-15}$
216 ($2.4{\cdot}10^6$)	2.3 s	2.8 s	$4.5{\cdot}10^{-16}$	$1.1{\cdot}10^{-14}$
512 ($5.7{\cdot}10^6$)	2.3 s	2.4 s	$5.6{\cdot}10^{-16}$	$1.3{\cdot}10^{-14}$
1 000 ($1.1{\cdot}10^7$)	3.2 s	3.5 s	$5.9{\cdot}10^{-16}$	$1.4{\cdot}10^{-14}$
2 197 ($2.4{\cdot}10^7$)	4.1 s	3.9 s	$6.7{\cdot}10^{-16}$	$1.6{\cdot}10^{-14}$
4 096 ($4.4{\cdot}10^7$)	5.5 s	6.0 s	$1.1{\cdot}10^{-15}$	$2.6{\cdot}10^{-14}$

Table 10.4: Comparison of maximum absolute error $\varepsilon_{\text{abs}}(\mathbf{d}_s)$ and relative error $\varepsilon_{\text{rel}}(\mathbf{d}_s)$ over all load steps of the displacement solution $\mathbf{d}_s$ at point (1,1,1). The solution was compared between `AceGen`/`AceFEM` and `FEDDLib` computations of the simple deformation–diffusion model in Section 4.1, without diffusion. The test setup corresponds to Figure 9.4 with applied surface force of 16 kPa (120 mmHg). The considered cube consists of 189 degrees of freedom.

error over all load steps	$\varepsilon_{\text{abs}}(\mathbf{d}_s(1,1,1))$	$\varepsilon_{\text{rel}}(\mathbf{d}_s(1,1,1))$
	$1.0{\cdot}10^{-15}$	$2.4{\cdot}10^{-13}$

though no actual diffusion process is included in this test, the results show that the implementation of the coupled problem produces the same results as AceFEM. For this validation, the results at point (1,1,1) are compared for an incremental loading of the desired surface force 16 kPa (120 mmHg) over 20 load steps. The results are shown in Table 10.4.

Finally, we compare the interface results with the benchmark problem of a curved artery

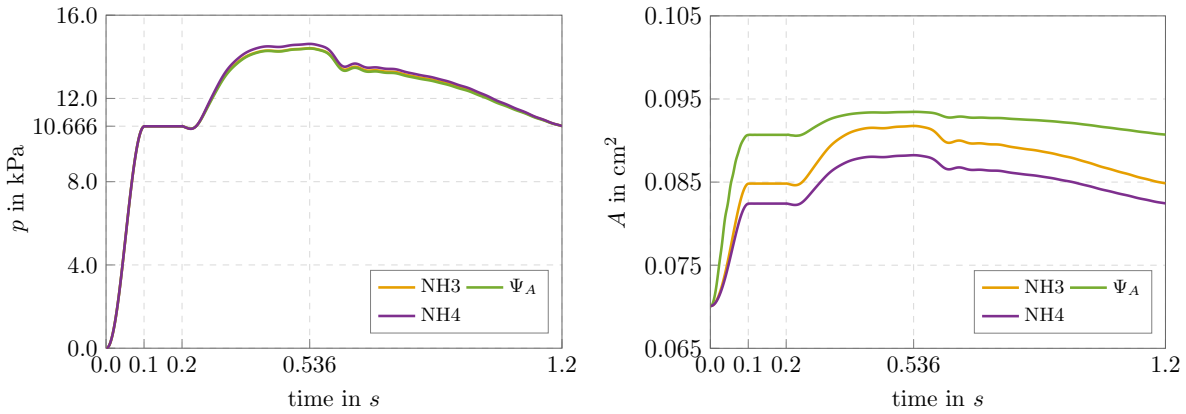


Figure 10.2: Comparison of results computed in `FEDDLib` with `FEDDLib`–`AceGen` interface for the FSI curved artery benchmark [8, 10]. Outflow pressure (left) and outflow area (right) for the material models and parameters described in Table 9.5. Flow rate increases in ramp phase until $t = 0.1\,\mathrm{s}$, is then kept constant, and one heartbeat is computed from $0.2\,\mathrm{s}$ to $1.2\,\mathrm{s}$; see Figure 9.10. Axis scaling and color scheme are chosen to match the results from [10, Figure 5]

presented in [8] and [10]. In particular, we consider the *NH3* and *NH4* models, which are based on a neo-Hookean constitutive law (see Section 3.1.1), as well as the Ψ_A model introduced in Section 3.1.3. The benchmark setup is described in Section 9.5. The absorbing boundary condition from (9.4) is applied as the pressure outflow condition. At the inlet, a parabolic velocity profile with time-dependent flow rate is prescribed; compare Figure 9.10 (right). For the Ψ_A formulation, the original sources employed a $\bar{F}$ element discretization for the solid problem. In the present work, however, a P2 element discretization is used. As reported in [8, Figure 21], both the $\bar{F}$ and P2 discretizations yield nearly identical results for the outflow pressure and cross-sectional area across different mesh resolutions. The obtained results are compared with those reported in [10, Figures 5 and 8]. In Figure 10.2, the outflow pressure and cross-sectional area are shown for the different structural models. The computed results agree closely with the reference results from [10, Figure 5]. Slightly larger outflow areas are observed for all three models, which may be attributed to the different time integration scheme used for the solid subproblem; we consider a Newmark scheme (Section 3.3), whereas for the benchmark a BDF-2 scheme is used [74]. Furthermore, different monolithic coupling schemes are considered. Here, we consider a fully implicit method, whereas in [8, 10] a convective explicit (CE) approach is used; see for example [33, Section 3.2] for a comparison of the methods. Nevertheless, we consider the benchmark results to be reproduced accurately.

Furthermore, Figure 10.3 presents the displacement fields for the Ψ_A and NH4 models. The displacement scaling is chosen consistently with [10, Figure 8]. The deformation patterns show very good agreement with the benchmark results.

Proof of Concept

The results in Figure 10.4 show preliminary results of the FSCI (FSI including a deformation–diffusion coupling) simulation for the simple deformation–diffusion model presented in Section 4.1. The curved artery benchmark model is used with the neo-Hookean material model and the *NH3*

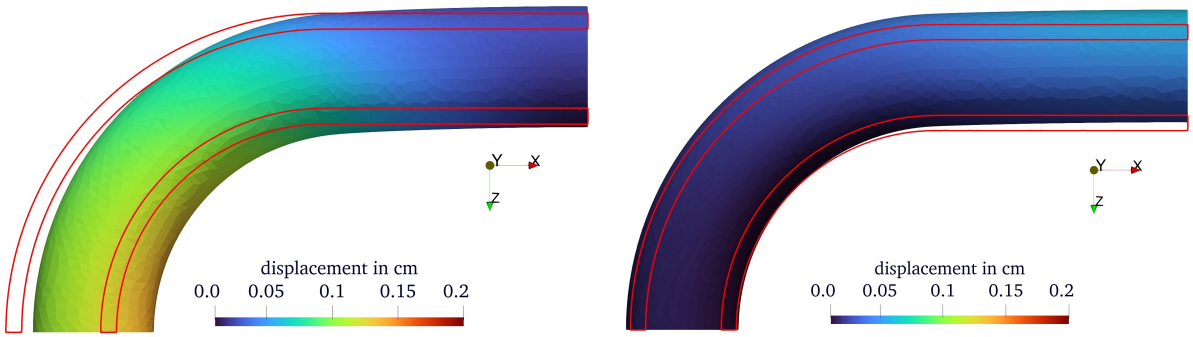


Figure 10.3: Comparison of results computed in `FEDDLib` with `FEDDLib`–`AceGen` interface for the FSI curved artery benchmark [8, 10]. Norm of the displacement and deformation for Ψ_A (left) and NH4 (right) model at $t = 0.536\,\mathrm{s}$ for the material models and parameters described in Table 9.5. The scaling is chosen to match the results in [10, Figure 8]. The red lines indicate the reference configuration of the solid domain.

parameter set; see Table 9.5. This serves as a proof of concept, where the FSCI framework is demonstrated using a simplified model for the relation between drug concentration and material properties. Based on the prescribed inflow rate Figure 10.4 (top) 5 consecutive heartbeats are modeled. The diffusion of a drug starts at $t = 1.2\,\mathrm{s}$. With increasing concentration the arterial wall becomes softer; compare to the relation between Young's modulus and concentration in (4.4). Due to the softer material, the deformation increases, thus, reducing the overall internal pressure; compare Figure 10.4 (middle) for area and Figure 10.4 (bottom) for the corresponding pressure values.

10.3. Preconditioning the Solid Subproblem

The following section contains analysis of the solver and preconditioner performance and robustness for the deformation, i.e., the solid problem. First, a neo-Hookean material model is considered in Section 10.3.1; then the findings are applied to the Ψ_A model (Section 3.1.3) in Section 10.3.2. Note that only a P2 finite element discretization is considered.

Pressure values are reported in kPa and, for reference, in mmHg (given in parentheses). Throughout this work, the conversion $1\,\mathrm{kPa} \approx 7.5\,\mathrm{mmHg}$ is used, and the corresponding values in mmHg are rounded to the nearest integer.

Newton's method terminates when the relative residual $\|\mathcal{R}(X^{k+1})\|_2 / \|\mathcal{R}(X^0)\|_2$ reaches a tolerance of 10^{-8} or the Newton update $\|X^{k+1} - X^k\|_2$ reaches a tolerance of 10^{-6}, respectively.

10.3.1. Neo-Hookean Material Model

In this section, results for the neo-Hookean material model (Section 3.1.1) are presented for the axial pulling of a cube introduced in Section 9.2, where the surface loading is illustrated in Figure 9.4 (left). A load of $21.33\,\mathrm{kPa}$ ($160\,\mathrm{mmHg}$) is applied. This corresponds to the maximum pressure values considered in the deformation–diffusion problem; see for example Figure 10.11.

The weak scaling results are summarized in Table 10.5 and Figure 10.5 using a Young modulus

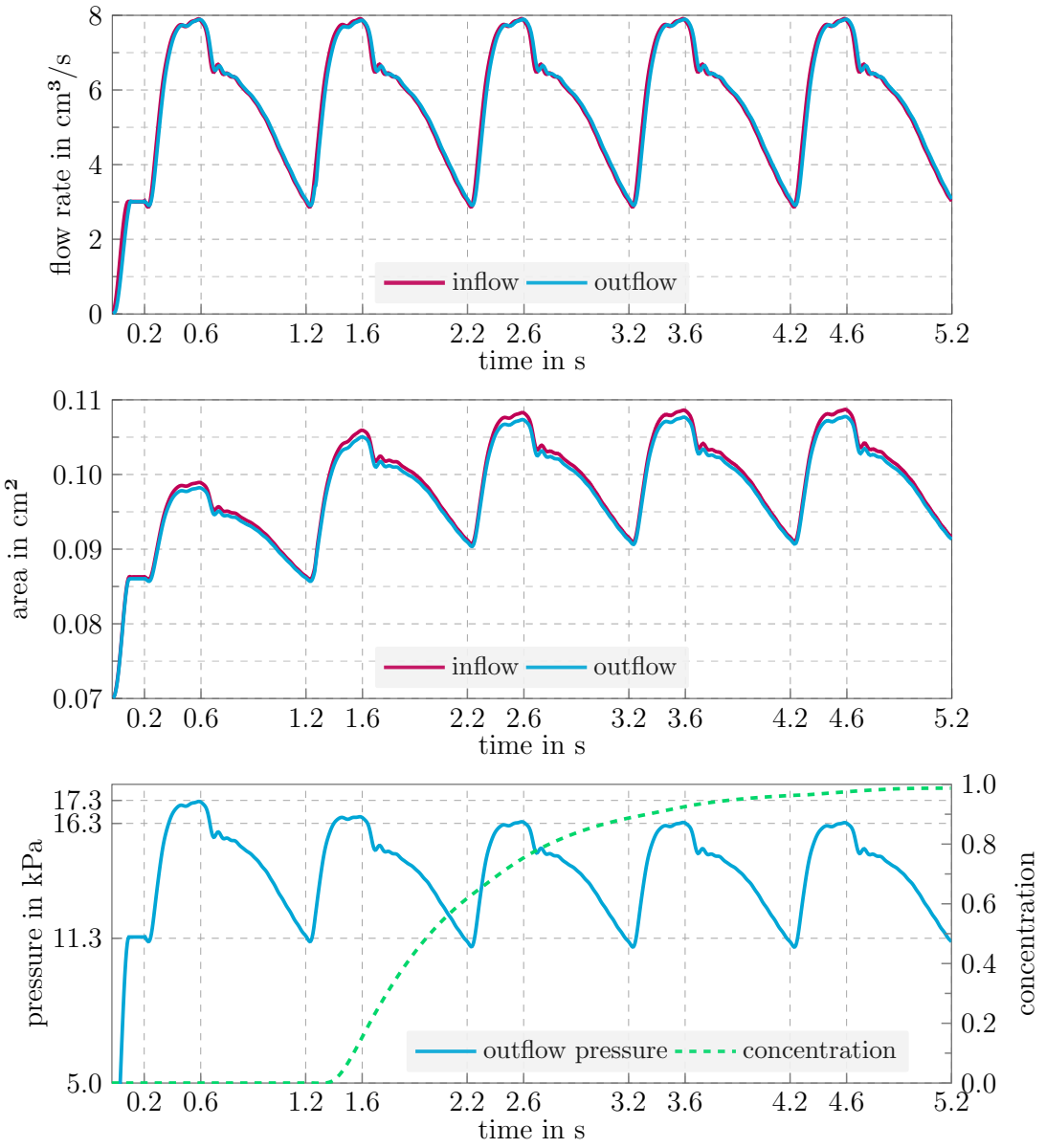


Figure 10.4: Comparison of flow rate, area and pressure results for the FSI curved artery benchmark [8, 10] with drug diffusion. After the first heartbeat the concentration of a drug increases at $t = 1.2\,\mathrm{s}$. Note that compared to the results in Figure 10.2, a higher reference pressure value is considered in (9.4) with $p_{\mathrm{ref}} = 11.3\,\mathrm{kPa}$ (85 mmHg). **Top**: Flow rate prescribed on the inlet of the fluid domain. A ramp phase is executed until $t = 0.1\,\mathrm{s}$. Five consecutive heartbeats are applied. The heartbeat is modeled after [134]. **Middle**: Inflow and outflow area measured at the inlet and outlet of the fluid domain. **Bottom**: Pressure measured at the outflow of the fluid domain. Due to the increasing drug concentration the pressure decreases from 17.3 kPa (130 mmHg) to 16.3 kPa (122 mmHg) at the peak of the heartbeat.

of 380 kPa and Poisson's ratio of 0.49; cf. *NH3* in Table 9.7. The GDSW coarse space, and its reduced dimension variants GDSW* and RGDSW are considered. In Figure 10.5 (left), the different coarse space configurations are compared. The RGDSW coarse space combined

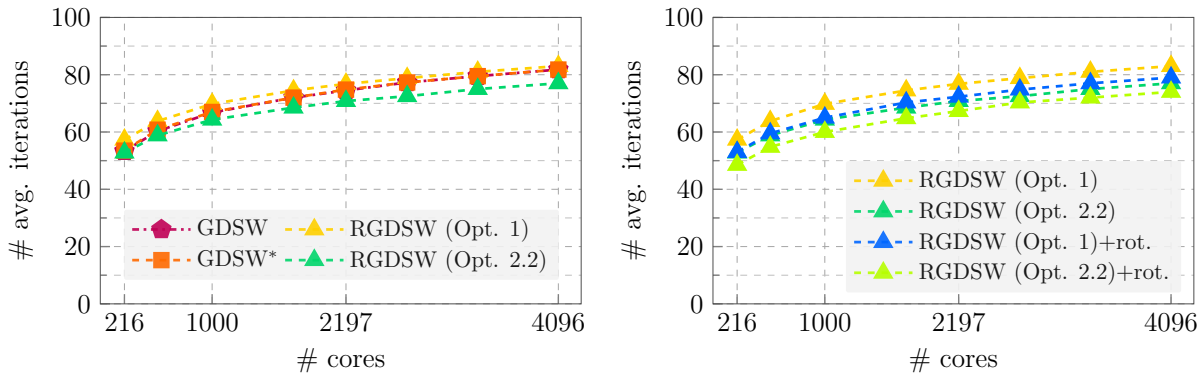


Figure 10.5: Weak scaling results for the neo-Hookean material model for the axial pulling of a cube from Section 9.2 with surface load $p = 21.33\,\text{kPa}$ (160 mmHg). $H/h = 8$. Average linear iteration count per Newton step. Four Newton iterations are required. No reuse of the full preconditioner after k Newton iterations. No (CB) and (CM) reuse. Young's modulus is set to 380 kPa. **Left**: Comparison of different coarse spaces. **Right**: Comparison of the RGDSW coarse space with Option 1, and Option 2.2, and with and without rotational modes in the coarse space.

with Option 2.2 yields the lowest average iteration count, although all coarse spaces provide a comparable overall performance; see (7.5) and (7.6) for the scaling options in RGDSW which correspond to Option 1 and Option 2.2. Similar to the results in [9, Table 5], where some of the considered coarse spaces were compared for the same test setup, we observe that the larger coarse space in GDSW leads to higher setup and solve times, especially for larger processor core sizes; see Table 10.5. The GDSW* coarse space is smaller in size and delivers approximately the same iteration counts as GDSW. Therefore, the timings improve for GDSW* compared to GDSW. However, the RGDSW coarse space leads to the best timings; see Table 10.5. A more detailed review of the RGDSW coarse space is illustrated in Figure 10.5 (right). In particular, the influence of including rotational modes in the coarse space is considered. The additional rotations reduce the iteration count for both Option 1 and Option 2.2, with the combination of Option 2.2 and rotations providing the best overall performance; see also Table 10.5 for the timings. It should be noted, however, that the inclusion of rotational modes increases the dimension of the coarse space which may lead to higher setup and solve costs, particularly for computations on a large number of cores; see Table 10.8; cf. [79] for similar comparison using linear elasticity.

Results for a one-level overlapping Schwarz preconditioner are also included in Table 10.5 as a reference a value. As expected for the one-level method, we do not observe weak scaling, where the number of average linear iterations increases from 184.5 on 216 processor cores to 534.8 on 4 096 processor cores, respectively.

10.3.2. Ψ_A Material Model

Now we consider the Ψ_A material model which was described in Section 3.1.3. The material model is anisotropic and quasi-incompressible with two fiber families, which are exponentially stiffening.

We consider the material settings adopted from [9] and [8, 10], see Table 9.8, which are denoted as *Set 1* and *Set 2*, respectively. Note that the Ψ_A model is extended by the active material

Table 10.5: Weak scaling results for the neo-Hookean material model for the axial pulling of a cube from Section 9.2 with surface load $p = 21.33\,\text{kPa}$ (160 mmHg). $H/h = 8$. Four Newton iterations are required. No reuse of the full preconditioner after k Newton iterations. No (CB) and (CM) reuse. Young's modulus is set to 380 kPa. In RGDSW coarse space Option 2.2 is used.

	N	216	512	1 000	2 197	4 096
1-Level	# avg. iter.	184.5	257.3	329.5	434.5	534.8
	setup	5.5 s	7.4 s	7.3 s	8.5 s	8.4 s
	solve	28.6 s	53.6 s	74.64 s	110.4 s	156.5 s
GDSW	# avg. iter.	53.3	60.3	66.8	74.5	81.8
	setup	9.7 s	11.1 s	15.9 s	21.7 s	29.6 s
	solve	10.2 s	11.9 s	17.3 s	25.2 s	32.5 s
GDSW*	# avg. iter.	53.5	60.8	67.0	74.8	81.8
	setup	9.1 s	9.4 s	12.2 s	15.1 s	16.6 s
	solve	9.8 s	10.8 s	16.9 s	18.0 s	23.7 s
RGDSW w/o Rot	# avg. iter.	52.8	58.8	64.3	70.8	77.0
	setup	8.8 s	9.1 s	10.8 s	12.3 s	12.7 s
	solve	9.2 s	10.0 s	13.6 s	15.3 s	22.1 s
RGDSW w Rot	# avg. iter.	48.5	54.8	60.0	67.3	74.0
	setup	9.2 s	10.9 s	13.0 s	14.2 s	15.7 s
	solve	8.6 s	9.8 s	12.7 s	16.2 s	21.3 s

response to form the smooth muscle model; see Section 3.1.2. Therefore, we first analyze the Ψ_A model and then extend it by the active material response.

For the analysis of the solver performance, we consider again the deformation problem from Section 9.2 and Figure 9.4 (left). Here, the two surface load values of 10.66 kPa and 21.33 kPa are considered, which correspond to approximately 80 mmHg and 160 mmHg, respectively. We use a load-stepping approach to gradually apply the force over 10 load steps, to ensure convergence of the nonlinear solver; see (4.13). Different fiber angles are tested. For a fiber angle of 45°, the fiber families are oriented orthogonally, for 0° the fibers are oriented in the same z-direction, and for 90° the fibers are oriented in the x-y-plane; see Figure 10.6. The results in Table 10.6 indicate that the fiber angle has only a minor influence on the performance of the preconditioner. It should be noted that fiber angles of 0° and 90° represent extreme configurations. These cases are not intended to be physiologically motivated, but are included only to assess the robustness of the preconditioner under limiting conditions. The fiber angle 43°, however, is physically motivated and based on parameter *Set 2*. For parameter *Set 1*, the 90° configuration fails to converge under the prescribed settings, when a surface force of 21.33 kPa is applied. More precisely, Newton diverged at load step 0.06 s. This might be attributed to the fact that the parameter *Set 1* model is softer with respect to the fiber-related anisotropic part compared to parameter *Set 2* and

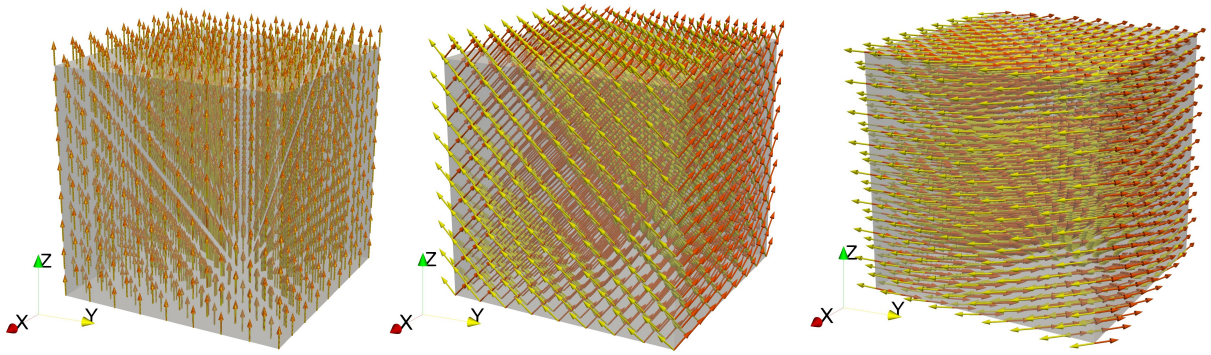


Figure 10.6: Different fiber angles considered for the axial pulling of a cube from Section 9.2. The considered fiber angles are 0° (left), 43° (center) and 90° (right), resulting in the two different fiber families depicted in yellow and red.

Table 10.6: Average linear iteration count per Newton step for the Ψ_A material model for surface loads of 10.66 kPa (80 mmHg) and 21.33 kPa (160 mmHg). 216 subdomains with H/h = 8. RGDSW with Option 2.2. Results including rotations in coarse space are denoted in parentheses. Adaptive forcing term. No Newton reuse. Average of 5 Newton iterations per load step for 21.33 kPa, and 4 Newton iterations for 10.66 kPa.

	f_A	0°	43°	45°	90°
Set 1	10.66 kPa	101.6 (94.3)	94.4 (88.1)	95.0 (88.6)	97.9 (91.7)
	21.33 kPa	93.4 (85.0)	84.1 (77.1)	82.9 (76.0)	- (-)
Set 2	10.66 kPa	120.2 (113.3)	114.5 (107.2)	112.6 (107.5)	103.1 (97.7)
	21.33 kPa	116.6 (107.8)	114.5 (107.8)	113.7 (107.4)	91.0 (86.0)

more load steps may be required. Again, the use of rotations for the coarse space shows overall improved results; see the number in parentheses in Table 10.6.

The results in Table 10.6 show only minor differences in the average iteration count per Newton step for the different surface loads. Therefore, it is justifiable to use a surface load of 18.66 kPa (140 mmHg) for a weak scaling analysis; a mean pressure value of 18.67 kPa (140 mmHg) is considered in the more realistic test setup Figure 10.11, which includes several heartbeats. We consider the parameters from Table 9.8 *Set 1*. The results are included in Figure 10.7 (right) and show a comparison between the Ψ_A model and the Ψ_{SMC} model which contains the active material response. The material response starts at $t = 1$ s, after the loading phase. We see similar weak scaling behavior as for the neo-Hookean material model with overall higher average linear iteration count; see Figure 10.5. For instance, the neo-Hookean material model requires 77 average iterations per Newton step, compared to 111 for the Ψ_A model.

10.4. Preconditioning the Deformation–Diffusion Problem

The deformation–diffusion problem considered in this section is characterized by a material based on the chemo-mechanical smooth muscle cell model from Section 3.1.2 and a diffusion process

that models the increasing concentration of drugs. The chemo-mechanical smooth muscle cell model will be denoted as Ψ_{SMC}. Note that the Ψ_A model we consider in the previous section represents the passive material response of the Ψ_{SMC} model. The active material parameters of this section are chosen according to [9, Tables 2–4], which are based on the work of [168]. The passive material parameters in this section are set according to *Set 1* in Table 9.8. Note that we consider a quasi-static solid problem, where the dynamic problem is neglected. This corresponds to solving the problem described in (4.11).

The first aspect that is investigated with respect to the solver and preconditioner is the active material response, i.e., the smooth muscle cell contraction. To this end, the loading protocol shown in Figure 9.4 (right) up to a final time of $T = 50\,\text{s}$ is considered. The loading phase consists of 10 load steps, while afterwards a time step size of $\Delta t = 0.25\,\text{s}$ is chosen. Weak scaling results for this setup using both GDSW and RGDSW coarse spaces were previously reported in [9, Tables 5–8]. In particular, the study focused on different recycling strategies for the coarse problem, where reusing only the symbolic factorization of the coarse matrix yielded the best overall performance; see also Section 8.5 for a review of the different recycling strategies. In the following, results regarding the RGDSW coarse space are reevaluated, that is, different recycling strategies, adaptive forcing term values, and Newton reuse (NR-k) are considered. Therefore, the RGDSW coarse space is used with Option 2.2, which delivered the best results; see Table 10.5 and Figure 10.5.

Usually, the active material response is a process that takes time to properly develop, since initially the cross-bridge fractions are set to $n_{\text{A}} = 1$ and $n_{\text{B}} = n_{\text{C}} = n_{\text{D}} = 0$. Consequently, no active material response is present. To accelerate that process, certain parameters are multiplied by a constant. For instance, increasing η in (3.8) leads to an increased calcium concentration, which causes stronger active contraction. For the detailed analysis in Table 10.7 an acceleration value of 4 is used. Displacement results of different acceleration values are illustrated in Figure 10.7 (left) which also include the average required linear iterations per Newton step per load step. The results suggest only a moderate influence on the average linear iteration count when varying the acceleration values. If not stated otherwise, the accelerated active response is set to 1, which means the active response is not artificially accelerated. Table 10.7 presents a detailed analysis of the setup and solve phases for computations performed on 2 197 cores. Different reuse strategies for the preconditioner components are investigated and compared. To this end, two different aspects are considered. First, the complete preconditioner can be reused after k Newton steps in each time or load step, e.g., if $k = 1$ the full preconditioner is reused after the first Newton step for the remaining necessary steps denoted as (NR-1). Since, on average, only two Newton iterations are required during the active response phase, i.e., for $t > 1\,\text{s}$, reusing the preconditioner for one additional Newton step is reasonable. This corresponds to (NR-1). However, this strategy is more critical during the loading phase, where the nonlinear solution changes more significantly. In this regime, reconstructing the preconditioner at every Newton iteration remains preferable. Therefore, in the loading phase from 0 s to 1 s the preconditioner is reconstructed in each Newton step; see also Section 10.1.2. Second, when the preconditioner is constructed, different components can be reused, e.g., reusing the coarse basis after it is constructed in the first Newton step of the first time step; see Table 8.1.

Furthermore, the initial parameter η_{init} of the adaptive forcing term strategy could be increased.

Table 10.7: Solver analysis of the Ψ_{SMC} material model with active material response for acceleration parameter 4 on 2 197 cores. Surface load of $p = 18.66\,\text{kPa}$ (140 mmHg) is applied corresponding to the loading protocol in Figure 9.4 with final time of $T = 50\,\text{s}$. The loading phase consists of 10 load steps, while afterwards a time step size of $\Delta t = 0.25\,\text{s}$ is chosen. $H/h = 8$. 206 time steps. Use of RGDSW coarse space without rotations and Option 2.2. Reusing the preconditioner after the first Newton step in each load step (after the loading phase) is denoted as (NR-1). The best solver setting is marked in bold, resulting in the lowest total time (setup + solve time). On average, two Newton iterations are required.

Reuse	SF	CB	CM	CB CM	SF	SF (NR-1)	SF (NR-1)
η_{init}	0.001	0.001	0.001	0.001	0.01	0.01	0.1
# avg. iter	116.1	176.5	155.0	225.9	103.3	103.3	**97.2**
setup	948 s	670 s	864 s	587 s	956 s	556 s	**557 s**
setup 1st Level	504 s	505 s	506 s	505 s	505 s	289 s	288 s
setup 2nd Level	394 s	120 s	349 s	1.8 s	401 s	237 s	237 s
solve	2 713 s	4 150 s	3 627 s	5456 s	2 368 s	2 390 s	**2 298 s**
Orthogonalization	407 s	726 s	600 s	1 108 s	319 s	326 s	324 s
1st Level apply	1 822 s	2 733 s	2 405 s	3 476 s	1 632 s	1 634 s	1 553 s
2nd Level apply	340 s	475 s	433 s	600 s	289 s	300 s	296 s

The default value is set to $\eta_{\text{init}} = 10^{-3}$; see Section 10.1.1. After the loading phase, the solution changes only marginally due to the active response, such that typically only one or two Newton iterations are required. Consequently, employing a higher linear solver tolerance, e.g., for $\eta_{\text{init}} = 0.01$, still ensures convergence of the Newton scheme while simultaneously reducing the number of required linear iterations. This is supported by the results in Table 10.7, where the average iteration count was reduced from 116 to 97.

Similar observations to those reported in [9] can be made. More invasive reuse strategies, such as reusing the coarse basis or the coarse matrix, reduce the setup costs. However, these approaches also increase the average iteration count, leading to higher solve times that outweigh the savings during setup.

In summary, increasing the parameter η_{init} in the adaptive forcing term strategy decreases the average number of linear iterations and consequently reduces the overall solve time without increasing the number of required Newton iterations; see the last three columns of Table 10.7. Furthermore, reusing the preconditioner after the first Newton iteration $k = 1$ (NR-1) following the loading phase yields additional reductions in setup cost. In combination, these modifications lead to an overall improvement in total runtime when we compare the first and last column in Table 10.7.

In Figure 10.7 (left) the displacement solutions for different acceleration values are displayed. The average iteration counts per Newton step are also noted, with average required Newton iterations in parentheses.

The improved parameter settings, see Table 10.7 (last column), are now applied to the full

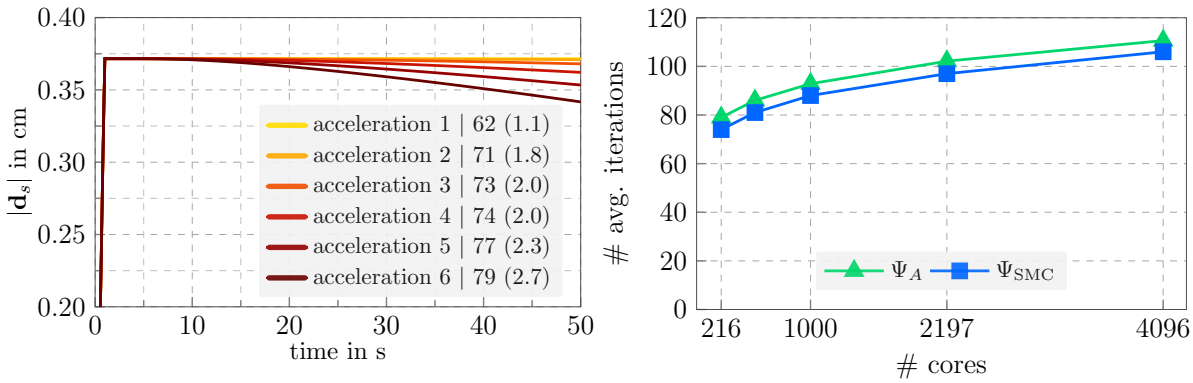


Figure 10.7: Model problem from Figure 9.4. Surface load of 18.66 kPa (140 mmHg). Loading from 0 s to 1 s with 10 load steps. Active response from 1 s until the end of 50 s with $\Delta t = 0.25$ s. Compare to the protocol of Figure 9.4 (right). Rotational modes for the coarse space of the solid preconditioner are not considered. **Left**: Displacement measured at point (1,1,1) for different acceleration multipliers for the active material response. Average linear iteration count per Newton step per time step given in corresponding labels with average Newton iteration count in parentheses. **Right**: Weak scaling for Ψ_A model, which corresponds to the loading phase, from 0 until 1 s, of the considered test case (left). The weak scaling of the Ψ_{SMC} corresponds to the full protocol from 0 to 50 s with 206 time steps, as the model includes active material response. Acceleration of the active response with multiplier 4.

weak scaling study presented in Table 10.8 and Figure 10.7 (right) with active acceleration parameter 4. With respect to the average iteration count, good weak scalability is observed. The first rows in Table 10.8 correspond to computations without rotational modes in the coarse space and illustrate weak scaling behavior especially in terms of setup time. The increase in solve time for increasing number of processor cores is mainly attributed to the application of the preconditioner; the coarse space size increases with increasing number of subdomains, making the application of the second level more expensive. For instance, for 2 197 subdomains, the coarse space size dimension is 5 184, and 10 125 for 4 096 subdomains. This adds approximately 150 s to the solve time. The remaining increase in solve time is attributed to the orthogonalization — due to the increase in average linear iteration count per Newton step — and the first level solve.

Including rotational modes in the coarse space yields only marginal reductions in the average iteration count, while simultaneously increasing both setup and solve times, e.g., by increasing the coarse space dimension. For the comparatively simple test case of axial pulling of a cube, the inclusion of rotations may not be necessary. However, for more complex geometries and test cases, e.g., considering a tube or realistic arteries, rotations in the coarse space may become more relevant and could potentially improve the robustness of the solver.

Now, the diffusion component is added to the deformation problem. In that case, at $t > 50$ s the diffusion of a drug through the solid material is modeled; see Figure 9.4 (right). Two coupling strategies are considered, corresponding to the formulations in (4.11) and (4.14), respectively. The first strategy employs a fully implicit coupling (ChI), resulting in a single monolithic system, which is preconditioned by a monolithic overlapping Schwarz method. For the solid component, the RGDSW coarse space with Option 2.2 is used, whereas the diffusion component employs RGDSW with Option 1; see (7.5) and (7.6). These choices are based on the numerical

Table 10.8: Weak scaling results for the Ψ_{SMC} material model with active material response for acceleration parameter 4. Surface load of $p = 18.66\,\text{kPa}$ (140 mmHg) is applied corresponding to the loading protocol in Figure 9.4 with final time of $T = 50\,\text{s}$. The loading phase consists of 10 load steps, while afterwards a time step size of $\Delta t = 0.25\,\text{s}$ is chosen. $H/h = 8$. 206 time steps. Reuse strategy (SF) and reusing the preconditioner (NR-1). Initial forcing term value $\eta_{\text{init}} = 0.1$. Use of RGDSW coarse space with and without rotations and Option 2.2.

	N	216	512	1 000	2 197	4 096
RGDSW w/o Rot	# avg. iter.	74	81	88	97	106
	setup	415 s	457 s	517 s	557 s	569 s
	solve	1 159 s	1 494 s	1 901 s	2 298 s	2 692 s
RGDSW w Rot	# avg. iter.	68	77	85	95	104
	setup	443 s	495 s	578 s	645 s	695 s
	solve	1 078 s	1 442 s	1 922 s	2 413 s	2 897 s

investigations presented in Table 10.5 and Figures 10.5 and 10.7 and are therefore adopted throughout this section without considering further variants.

The second strategy follows a segregated coupling (ChE), in which the diffusion problem is solved first in each time step and the resulting concentration field is then used in the subsequent deformation solve. Since the two subproblems are treated independently, each can be preconditioned separately. For the diffusion problem, RGDSW with Option 1 is employed with (CB) and (CM), while the deformation problem again uses the previously identified optimal configuration; see [83] for weak scaling results using GDSW and RGDSW coarse spaces for a three-dimensional Poisson problem. Results in Figure 10.12 (right) show that the preconditioner for the diffusion problem is very robust, with average linear iteration count per Newton and time step of 6.

The full protocol of Figure 9.4 (right) is considered with an extended final time $t = 150\,\text{s}$. We use the findings from the previous section, i.e., the settings from the last column in Table 10.7, for both the implicit and explicit approach.

In Figure 10.8, the average linear iteration count per Newton step is compared for different test cases. First, the model only includes active material response. Then, the diffusion component is added at $t > 50\,\text{s}$ (denoted by *+ diffusion*), and then results are compared for the fully implicit and the explicit approaches (denoted by *ChE*). Note that for the implicit approach (denoted by *ChI*), the number of Newton iterations required to reach convergence increases from two to three — compared to the (ChE) approach — when the diffusion process sets in. A lower $\eta_{\max}$ tolerance, e.g., $\eta_{\max} = 0.001$, would lead again to two average Newton iterations required in the implicit case, but would increase the linear iteration count. However, the jump in required linear average iterations per Newton step — when diffusion sets in — is still observable, and thus seems independent of the required Newton iterations. Furthermore, we observe a different behavior in [9, Figure 4], where the average linear count per Newton step decreases as the diffusion process sets in. In that test setting, no accelerated active response is used, and the forcing term is constant with $\eta = 10^{-6}$. Therefore, we additionally consider the aspect, where the active response is not accelerated and compare a constant forcing term to the *Type 2* setting we

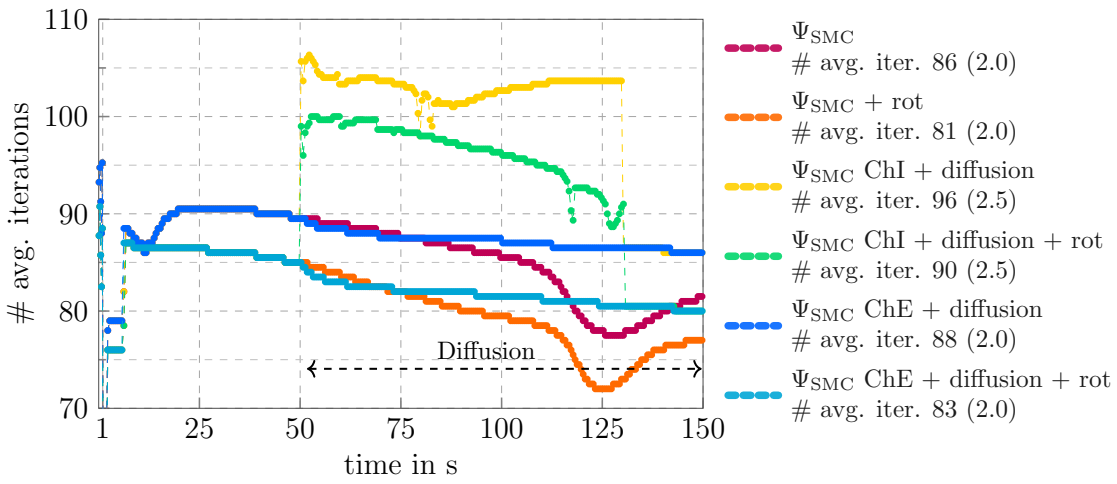


Figure 10.8: Average linear iteration count per Newton step over 150 s for different material settings and effects for the axial pulling of a cube using the Ψ_{SMC} model; see Figure 9.4 with surface load 18.67 kPa (140 mmHg) until $t = 150\,s$. Acceleration of active response with value 4. Computation on 1 000 cores with $H/h = 8$ and (NR-1). Initial forcing term value $\eta_{\text{init}} = 0.1$. RGDSW coarse space with Option 2.2. Diffusion coefficient $D = 0.02\,\text{cm}^2/\text{s}$.

have considered so far; see Figure 10.9. Now, the results correspond to the behavior in [9, Figure 4]. Note that, with active response accelerated and constant forcing term, the average number of linear iterations also increases for the ChI variant, when the diffusion sets in, comparably to the results in Figure 10.8.

The ChE approach proves to be more robust than the ChI approach when the diffusion sets in, with less variation in the average iteration count. Additionally, fewer average Newton iterations are required. Note that it takes approximately 12 iterations to solve the diffusion problem in each time step. For the ChI approach, we observe a jump in average iterations, when the diffusion sets in. When the diffusion process is finished, i.e., full saturation is reached with $c = 1$, the average iteration count reverts back to the level observed for the ChE approach; see Figure 10.8. Including rotational modes in the coarse space improves the average linear iteration count, observable as an offset value in Figure 10.8, improving the robustness of the results.

In Figure 10.10, the relative and absolute errors between the fully implicit (ChI) and explicit (ChE) approaches are shown. Additionally, the level of concentration is depicted on the right axis. The error values were measured at point (1,1,1). All error values are below $5{\cdot}10^{-4}$. For time step $t = 100\,\text{s}$, the maximum relative error over all nodes is $8.6{\cdot}10^{-4}$ for the displacement and the absolute error over all nodes is $8.6{\cdot}10^{-5}$ for the concentration. Note that using smaller time step sizes can reduce the error further.

In the following, we evaluate the qualitative results and practical application to simulate the influence of drug presence on the active material response. Therefore, the test case of the simplified arterial segment is used; see Figure 9.6. In Figure 10.11 (left), the full protocol for this test is shown. In Figure 10.11 (right) the inner radius of the geometry is illustrated and how it changes over time due to the active response. The corresponding deformation of the geometry after the ramp phase is illustrated in Figure 9.6 (right).

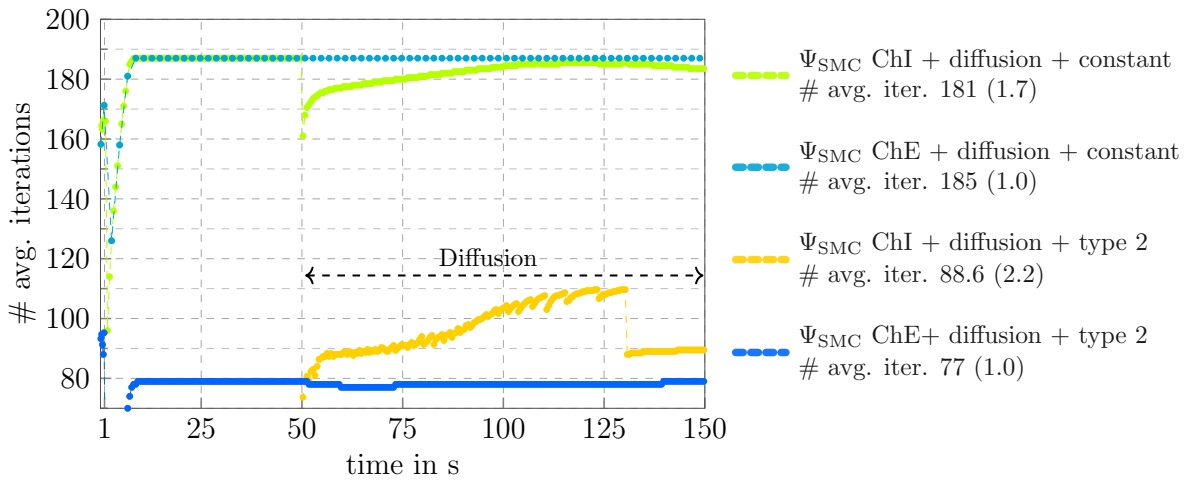


Figure 10.9: Average linear iteration count per Newton step over 150 s for different material settings and effects for the axial pulling of a cube using the Ψ_{SMC} model; see Figure 9.4 with surface load 18.67 kPa (140 mmHg) until $t = 150\,s$. No acceleration of active response. Computation on 1 000 cores with $H/h = 8$ and (NR-1). Comparison between ChI and ChE for constant and *Type 2* forcing term. Diffusion coefficient $D = 0.02\,\text{cm}^2/\text{s}$. Reusing the preconditioner after the first Newton step in each load step is denoted as $(k = 1)$. RGDSW coarse space with Option 2.2.

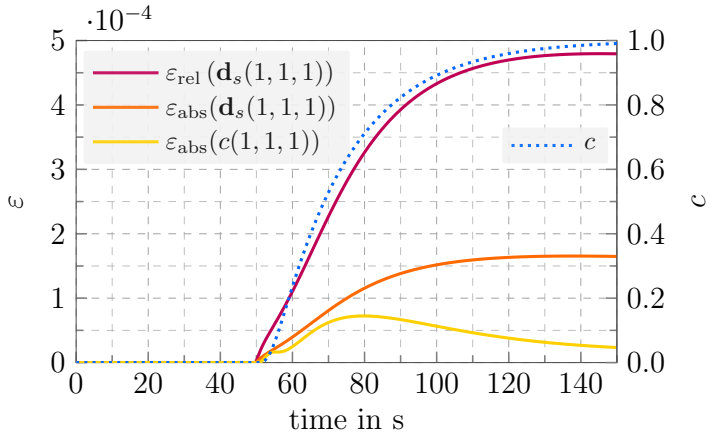


Figure 10.10: Relative and absolute error between fully implicit (ChI) and explicit (ChE) approach (see (4.14)) for the deformation–diffusion problem with Ψ_{SMC} model measured at point (1,1,1) for test problem from Figure 9.4 with surface load 18.67 kPa (140 mmHg). The relative error ε_{rel}, absolute error ε_{abs}; see (10.4). Concentration measured at (1,1,1) displayed on right axis. Computation on 125 cores with $H/h = 8$.

Between 0 and 1 s, a linear increase in pressure to 18.67 kPa (140 mmHg) is considered and held constant until $t = 2\,500$ s. Then 50 consecutive heartbeats are modeled; compare Figure 10.11 (left). The following discussion is similar to the one for [9, Figure 1], where a similar setup was used with a different geometry. First, the cross-bridge fractions are initialized with $n_{\text{A}} = 1$ and $n_{\text{B}} = n_{\text{C}} = n_{\text{D}} = 0$. Consequently, no active material response is initially present. At $t = 1$ s, the active material response is activated and the evolution equations for the internal variables are solved. This leads to a gradual increase in n_{C} and n_{D}, and thus to the development of active contraction. The material response shows an initially slow contraction followed by a rapid contraction phase associated with an increase in the fraction of myosin heads in state (C); cf. Figure 3.2 (right). The drug diffusion is initiated at $t = 1\,500$ s to represent the intake of antihypertensive drugs with a diffusion coefficient $D = 6 \cdot 10^{-5}\,\text{cm}^2/\text{s}$. The inflow is modeled by

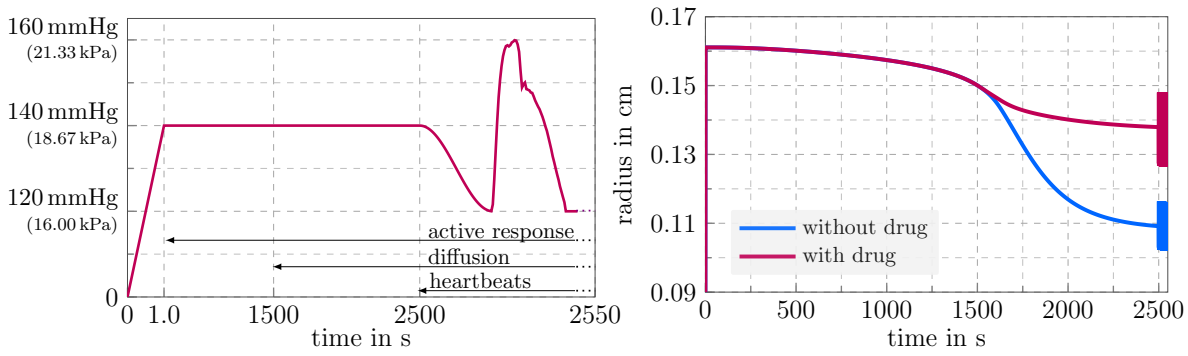


Figure 10.11: Long-term simulation of the deformation–diffusion problem with test case described in Section 9.3 and parameter *Set 1* from Table 9.8 for Ψ_{SMC} model. Corresponding solution of displacement after ramp phase is shown in Figure 9.6 (right). The diffusion coefficient is set to $D = 6 \cdot 10^{-5}\,\text{cm}^2/\text{s}$. **Left**: Loading protocol for deformation–diffusion test. Between 0 and 1 s, a linear increase in pressure to 18.67 kPa (140 mmHg) is considered and held constant until $t = 2\,500\,\text{s}$. Then 50 consecutive heartbeats are modeled. For the loading phase, 20 load steps are used, corresponding to $\Delta t = 0.05\,\text{s}$. Then for $t > 1\,\text{s}$, a time step size $\Delta t = 0.25\,\text{s}$ is used. The heartbeat and the corresponding flow rate were constructed in [8] based on inflow pressure data from [134]. **Right**: Inner radius of idealized artery over time. Note that the heartbeats modeled after 2 500 s are not resolved by the time scale in this visualization. Average linear iteration count per Newton step per time step is shown in Figure A.2.

switching the concentration Dirichlet boundary condition from zero to one at $t = 1\,500\,\text{s}$ on the outer wall of the geometry. This final part of the simulation is used to investigate the response of the material model to a time-varying intravascular pressure, as it occurs in real arteries due to heartbeats, and show that qualitative aspects can be reproduced.

In Figure A.2 the average linear iteration count per Newton step per time step is shown over the 2 550 s of simulations. Similar results compared to Figure 10.8 are shown. The linear iteration count for the (ChE) and (ChI) approach is identical in most of the considered time frame, except in a small time frame between approximately 1 900 s and 2 050 s, where the diffusion process is almost completed; full saturation is reached at 2 100 s.

As a final test case, the patient-specific artery is considered. The simulation begins with a linear ramping phase from $t = 0\,\text{s}$ to $t = 1\,\text{s}$, discretized by 200 load steps, corresponding to a time step size of $\Delta t = 0.005\,\text{s}$. This is followed by a steady phase from $t = 1\,\text{s}$ to $t = 100\,\text{s}$, during which the active material response develops. In this phase, a larger time step size of $\Delta t = 0.2\,\text{s}$ is employed. Drug diffusion is modeled by activating a Dirichlet boundary condition on the outer arterial wall. Consequently, the drug is assumed to diffuse from the exterior of the arterial wall towards its interior. At $t = 50\,\text{s}$, the diffusion process is activated using a diffusion coefficient of $D_0 = 0.0006\,\text{cm}^2/\text{s}$. One heartbeat is simulated starting at $t = 100\,\text{s}$.

During the steady phase, a constant internal pressure of 16 kPa (140 mmHg) is prescribed. Subsequently, an increasing pressure is applied to simulate one heartbeat, reaching a peak pressure of 18.67 kPa (160 mmHg); see Figure 10.11 (left) for the same pressure values on a larger time scale. A deformation-dependent surface load is employed; see Section 9.6.1.

Compared to the test case presented in Figure 10.8, a lower diffusion coefficient is used, since the arterial wall thickness is only approximately 0.1 cm to 0.2 cm; see Section 9.4. Thus, the

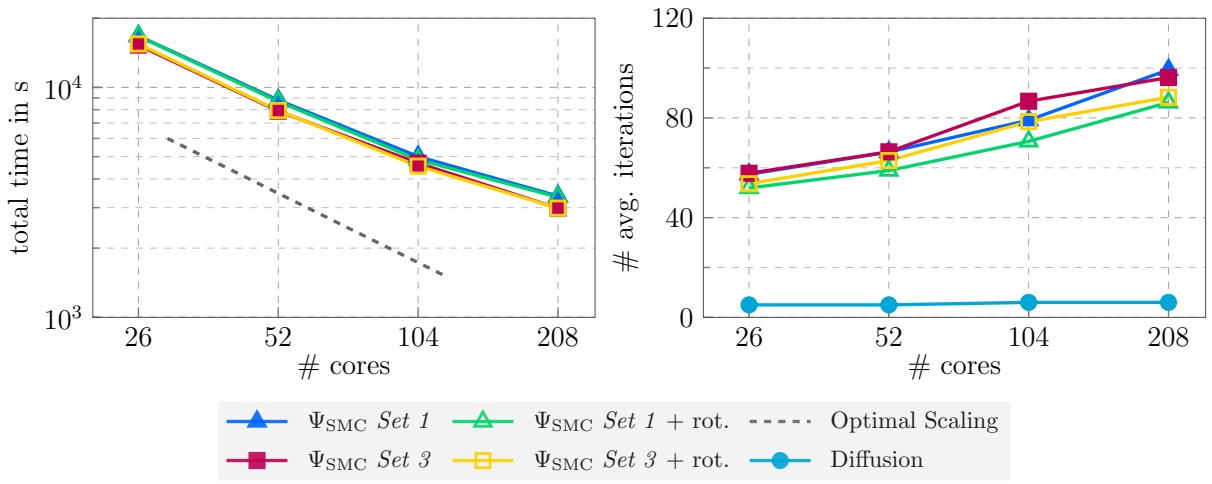


Figure 10.12: Strong scaling study for realistic patient-specific artery. The Ψ_{SMC} model from Section 3.1.2 with parameter *Set 1* and *Set 3*, i.e., homogeneous and heterogeneous material compositions are compared using the chemistry explicit approach (ChE); see Tables 9.8 and 9.9. Simulation of 101 s with one heartbeat at 100 s. 200 load steps in ramp phase. The diffusion coefficient is set to $D = 6 \cdot 10^{-4}\,\text{cm}^2/\text{s}$. Results of corresponding deformation in Figure A.1. **Left**: Total time consisting of setup and solve time. **Right**: Average iteration per Newton step per time step. Acceleration of active response by factor 4.

concentration reaches its maximum value faster than for the cube that was previously considered. The homogeneous material model is defined by *Set 1* from Table 9.8. In addition, a heterogeneous model is considered, in which distinct material parameters are assigned to the adventitia, media, and plaque, where the plaque is modeled as the fibrous cap, using *Set 3* from Table 9.9.

The strong scaling results are shown in Figure 10.12. On the left, the total time is illustrated, consisting of setup and solve time, while on the right the average number of linear iterations over all time steps is shown. Overall, good strong scaling behavior is observed.

Furthermore, for 104 cores, an additional simulation using a one-level overlapping Schwarz preconditioner was performed. The resulting average linear iteration count was 268, which is more than three times higher than that of the corresponding two-level method. This indicates that, for this class of problems, a second coarse level is essential to achieve good parallel scalability. This observation is consistent with the results presented in Table 10.5, where the one-level method was also included for comparison.

10.5. Comparison of Monolithic and Block Preconditioners for the Fluid Flow Problem

In this section, the preconditioning and solving methods for the fluid flow problem described in Chapter 2 are examined. Therefore, the block preconditioning techniques from Section 6.1 combined with two-level overlapping Schwarz methods (Section 7.2) are compared to the monolithic preconditioning approach described in Section 7.3. To this end, two three-dimensional settings are considered: the well-known backward-facing step (Section 9.1) and a realistic geometry of an artery (Section 9.4) from [7]. We show results for stationary and transient simulations with

volume force $\mathbf{f} = 0$.

In [96, Section 8.3.2], a preliminary comparison of the SIMPLE and LSC block preconditioners with a monolithic two-level overlapping Schwarz preconditioner was presented. This study is extended to include the PCD and augmented Lagrangian block preconditioners, a range of Reynolds numbers, and transient flow problems.

Parts of this section are taken from [82, Section 6]. Modifications to the figure styles have been made to improve consistency with the remainder of this dissertation. Results regarding the augmented Lagrangian block preconditioner were not part of [82]

For the monolithic as well as the block preconditioners, different coarse spaces can be applied to the velocity and pressure components. The coarse spaces GDSW, GDSW*, and RGDSW (see Section 7.1) are used. This offers the possibility to combine different coarse space combinations to improve convergence. Similarly to the notation of a discretization, for example, P2–P1, where the first term stands for the velocity and the second for the pressure discretization, we refer to the coarse spaces as, for example, GDSW*–RGDSW. To shorten the notation, sometimes the abbreviations G for GDSW, R for RGDSW, and G* for GDSW* are used in figures.

Newton's method is used with GMRES along with an adaptive forcing term η_k as described in Section 10.1.1. The initial forcing term is set to $\eta_{\text{init}} = 10^{-3}$, the minimum forcing term to $\eta_{\min} = 10^{-8}$, the maximum forcing term to $\eta_{\max} = 10^{-3}$; cf. (10.3). In each nonlinear iteration, the GMRES iteration is started with zero as the initial guess. Newton's method terminates when the relative residual $\|\mathcal{R}(X^{k+1})\|_2 / \|\mathcal{R}(X^0)\|_2$ or the Newton update $\|X^{k+1} - X^k\|_2$ reaches a tolerance of 10^{-8}.

For the stationary tests we will reuse the preconditioner in the Newton method after iteration $k = 3$ resulting in (NR-3). Results in Section 8.5.2 and Table 8.2 illustrated that for the stationary Navier–Stokes problem the (NR-3) configuration yielded the best results. In transient simulations it is sufficient to update the preconditioner only in the first Newton step of each time step (NR-1), probably since the required average number of Newton iterations per time step is approximately three.

In [78, Section 5.4] the GDSW and RGDSW coarse spaces were evaluated and compared for three-dimensional Navier–Stokes problems. In particular, reusing the coarse basis functions produced the lowest total time consisting of solve and setup time and only slightly increased the linear average iterations. Similar results are observed in Figure 10.25, where the reuse of coarse basis had only a very minor effect on the average iteration counts. Therefore, for the fluid flow problems, the reuse of the coarse basis (CB) is considered. Note that when coarse basis reuse (CB) is used, the coarse basis is reused after the initial construction in the first Newton step of the first time step and therefore corresponds to the coarse basis functions for Stokes. This is caused by the zero initial Newton guess.

Regarding the construction of the coarse matrix for the monolithic two-level overlapping Schwarz preconditioner, we can additionally choose between using the unaltered coarse basis (7.8) (*full* ϕ) or excluding the off-diagonal blocks:

$$\phi \leftarrow \begin{bmatrix} \phi_{\mathbf{u},\mathbf{u_0}} & 0 \\ 0 & \phi_{p,p_0} \end{bmatrix}.$$

This can have an effect on the scaling behavior of the preconditioner and will be further discussed

Table 10.9: Number of cores assigned to solve the coarse problem for the results shown in Figure 10.16 for the stationary BFS problem. Specifically, this corresponds to the P2–P1 discretization with the GDSW⋆–RGDSW coarse space combination. For RGDSW–RGDSW fewer cores are used due to the lower number of coarse functions; compare with Figure 9.2.

# cores	243	576	1 125	1 944	3 087	4 608	9 000	15 552
# cores for coarse solve	4	4	6	6	8	12	16	16

in the following sections; see also Section 7.4.

The results are structured as follows. In Section 10.5.1, the monolithic preconditioner is tested for variations of the coarse spaces. Subsequently, in Section 10.5.2, results of block preconditioners are shown, again for different coarse spaces. Both are considered first for the stationary (2.2), then for the transient (2.1) Navier–Stokes problem. Finally, in Section 10.5.3, for specific choices of coarse spaces and modifications, the performance of monolithic and block preconditioners is compared.

10.5.1. Monolithic Preconditioner

The monolithic overlapping two-level Schwarz preconditioner is used for the saddle-point problem (2.16) as described in Section 7.3. This technique has been introduced in [109, 110] with Lagrangian coarse spaces and extended to GDSW-type coarse spaces in [77, 78]. In [78], monolithic overlapping Schwarz preconditioners with GDSW and RGDSW coarse spaces were investigated for three-dimensional incompressible fluid flow problems. The numerical results presented in [78, Table 1] demonstrate that the RGDSW coarse space using Option 1 yields better performance than Option 2.2. Furthermore, the weak scaling results reported in [78, Table 2] show that the one-level monolithic preconditioner does not scale well for the stationary problem, highlighting the importance of a second coarse level. Further improvements with respect to the compute time can be achieved by using additional cores to solve the coarse problem; cf. [78, Section 4.4]. The number of processor cores assigned to the coarse solve depends on the coarse space dimension (shown in Figure 9.2 for the backward-facing step problem) and ranges between 2 and 16; see Table 10.9 for details in case of the BFS problem.

Stationary Navier–Stokes Problem

Different finite element discretizations are considered for the stationary BFS problem. For the analysis and comparison of the different coarse space combinations, and to show results of the GDSW⋆ coarse space, the P1–P1 discretization with the Bochev–Dohrmann stabilization and the inf-sup stable P2–P1 discretization are considered. As discussed in Section 7.4 the off-diagonal blocks in the coarse basis ϕ in (7.8) can be excluded, which corresponds to the coupling between velocity and pressure in the coarse basis functions. Further results show that excluding the off-diagonal blocks proves to be more robust. In Figure 10.13 we observe that P1–P1 elements yield similar results for all coarse space combinations, independent of using the full ϕ or excluding the off-diagonal blocks to construct the coarse matrix. Weak scaling can be observed for up to 15 552 cores; see also Figures 10.15 and 10.16 for results of RGDSW–RGDSW up to 15 552 cores.

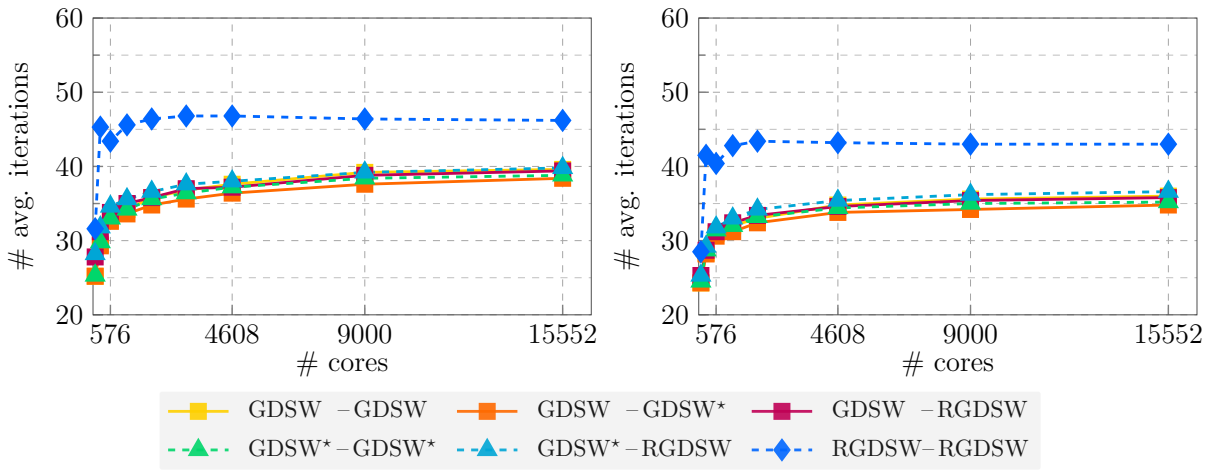


Figure 10.13: Comparison of different coarse space combinations for the velocity (first component) and pressure (second component) for a P1–P1 discretization and the monolithic preconditioner. Use of the Bochev–Dohrmann stabilization. Average GMRES iteration count per Newton step. Total number of Newton iterations required to reach convergence is 5. Weak scaling test. Stationary BFS problem with $H/h = 17$. $\mathrm{Re} = 200$. Pressure projection is not used. **Left:** Excluding off-diagonal blocks in ϕ to build the coarse matrix. **Right:** Using full ϕ to build the coarse matrix. Adapted from [82, Figure 6.4].

For the inf-sup stable discretizations, generally a pressure projection is applied in the first level of the preconditioner for the two-level overlapping Schwarz method; see (7.11). In Figure 10.14 the comparison with and without pressure projection is shown; compare the top to the bottom row. For the stable P2–P1 element, in contrast to the P1–P1 case, we observe a substantial influence for some combinations of coarse spaces and using the full or decoupled ϕ in Figure 10.14. The results indicate that the combination of using the full ϕ without pressure projection fails to converge; see Figure 10.14 (bottom right). The results in Figure 10.14 (bottom left) are comparable to the ones reported in [96, Section 8.3.2].

We focus first on the results in Figure 10.14 (left column), where the off-diagonal blocks of ϕ are excluded. It stands out that, if RGDSW for the pressure is combined with a larger coarse space like GDSW or GDSW* for the velocity, we achieve the lowest average GMRES iteration count. If a smaller coarse space for the velocity is used than for the pressure, the algorithm does not reach the GMRES convergence criterion (results are not shown here). The comparison to an inf-sup condition of a finite element discretization seems natural. On the other hand, the "equal-order" approach RGDSW–RGDSW is scalable with a low number of iterations, albeit not the lowest one. The cause for the correlation of the iteration count and the relative size of the velocity and pressure coarse spaces may also be independent of an inf-sup condition; further research is required.

For the results using the full ϕ in Figure 10.14 (right column), we observe similarly good scaling results but for all coarse space combinations. If the full ϕ is used to construct the coarse matrix, it seems imperative that also the pressure projection is applied in the first level; if the pressure projection is omitted, we lose scalability for all combinations in Figure 10.14 (bottom right).

On the other hand, the analogue to Figure 10.14 (top left) is Figure 10.14 (bottom left),

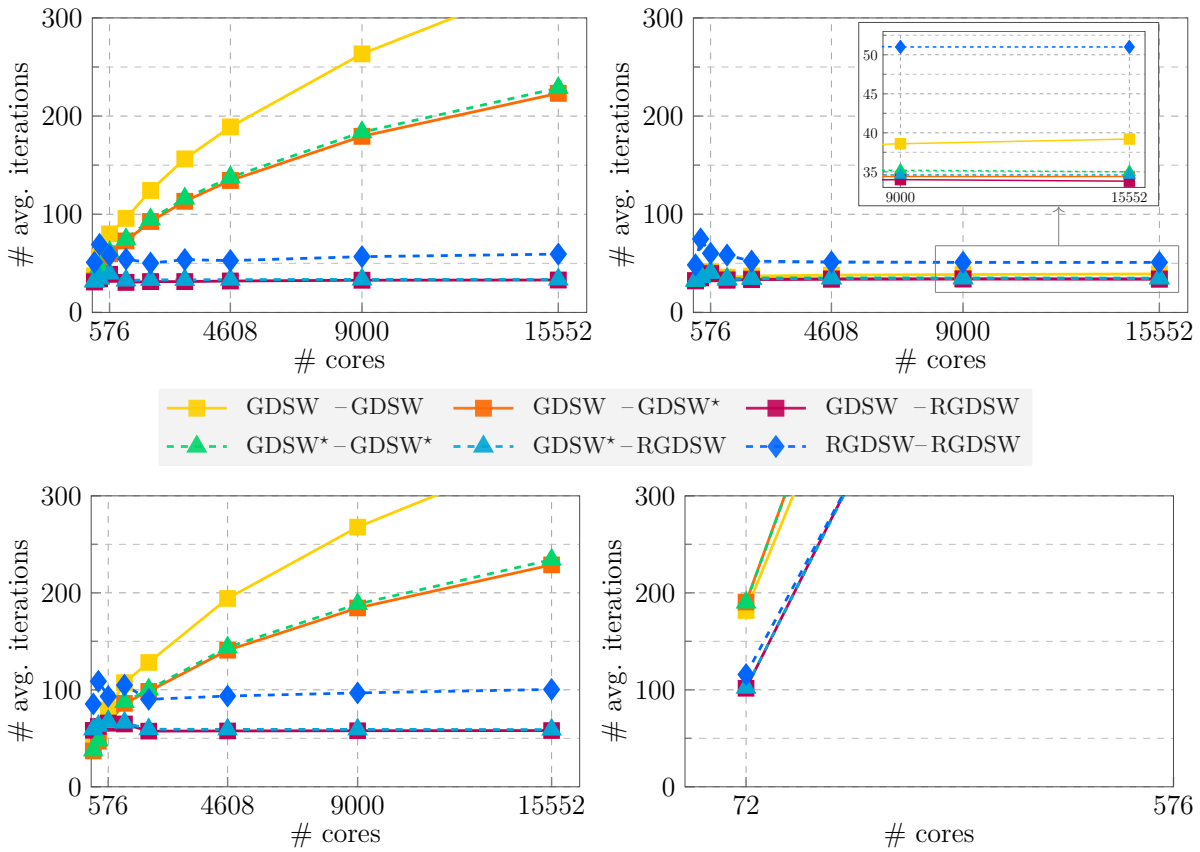


Figure 10.14: Comparison of different coarse space combinations for the velocity (first component) and pressure (second component) for a P2–P1 discretization and the monolithic preconditioner. Average GMRES iteration count per Newton step. Total number of Newton iterations required to reach convergence is 5. Weak scaling test. Stationary BFS problem with $H/h = 9$. Re $= 200$. See Figure A.3 in the appendix for results of a lid-driven cavity problem.
Top Left: Excluding off-diagonal blocks in ϕ to build the coarse matrix and pressure projection included in the first level. **Top Right:** Using full ϕ to build the coarse matrix and pressure projection included in the first level. **Bottom Left:** Excluding off-diagonal blocks in ϕ to build the coarse matrix; omitting pressure projection. **Bottom Right:** Using full ϕ to build the coarse matrix; omitting pressure projection. Adapted from [82, Figure 6.5, Figure A.2].

which shows the same general scaling behavior; that is, the pressure projection does not seem to affect the case in which ϕ is decoupled. Independent of the use of the decoupled or full ϕ, GDSW*–RGDSW is the best performing configuration for the P2–P1 finite element discretization if the pressure projection is used; cf. [82]. It remains to be thoroughly investigated, however, to determine how the various settings and coarse spaces impact each other. Figure A.3 in Appendix A.3 qualitatively shows the same results as Figure 10.14 for a lid-driven cavity problem, indicating that the results are not specific to the backward-facing step problem.

Table 10.10 shows details corresponding to Figure 10.13 (right) (P1–P1 discretization) and Figure 10.14 (right) (P2–P1 discretization) for a selection of coarse space combinations. Even though the average iteration count for the RGDSW–RGDSW combination is highest for P1–P1,

Table 10.10: Comparison of setup and solve time of the three best coarse space combinations for P1–P1 and P2–P1 discretization on 4 608 cores. Stationary BFS problem with configurations according to Table 9.1. Total number of Newton iterations required to reach convergence is 5. Use of Bochev–Dohrmann stabilization for P1–P1 elements. Use of full ϕ. Use of pressure projection for P2–P1. Re = 200. Taken from [82, Table 6.5].

		GDSW–RGDSW	GDSW⋆–RGDSW	RGDSW–RGDSW
	# avg. iterations	34.6	35.4	43.2
P1–P1	Setup time	26.7 s	17.5 s	12.8 s
	Solve time	21.3 s	18.4 s	22.3 s
	Total	48.0 s	35.9 s	**35.1 s**
	# avg. iterations	34.4	34.6	51.4
P2–P1	Setup time	47.6 s	23.7 s	20.2 s
	Solve time	27.3 s	22.5 s	32.8 s
	Total	74.9 s	**46.2 s**	53.0 s

it offers the lowest total cost due to its low setup cost. Notably, the setup cost for GDSW is higher than for GDSW⋆. The lowest setup cost offers the RGDSW coarse space. In Table 10.10 we observe how GDSW⋆–RGDSW performs best with respect to the total time (setup+solve) for the P2–P1 discretization.

For a comparison of the considered different discretizations, P1–P1, Q1–Q1, P2–P1, Q2–Q1, and Q2–P1-disc., we focus on GDSW⋆–RGDSW and RGDSW–RGDSW (see Figure 10.15), based on the previous findings.

Due to the discontinuous pressure in Q2–P1-disc., there does not exist an interface for the pressure component. As a result, all coarse spaces define the same space on the pressure: the span of constant functions on each subdomain; see Section 7.1.3. As mentioned at the beginning of this section, if the pressure coarse space is too large in comparison to the velocity coarse space, the algorithm does not converge. We observe the same for Q2–P1-disc. if RGDSW is used for the velocity. The explanation may be the same as before, since, for example in case of 243 subdomains, the RGDSW pressure coarse space for a continuous P1 pressure space has 124 coarse functions (vertices of the domain decomposition), but any GDSW-type coarse space for P1-discontinuous pressure functions has 243 coarse functions. Thus, RGDSW–RGDSW for Q2–P1 has a smaller pressure coarse space than for Q2–P1-disc. Consequently, the Q2–P1-disc. test results are based on the GDSW⋆ coarse space for the velocity; cf. [82].

The different configurations all yield similarly good weak scaling results for up to 15 552 processor cores. The stabilized P1–P1 and Q1–Q1 discretizations produce slightly higher average GMRES iteration counts than their stable counterparts. Excluding the off-diagonal blocks in ϕ or using it fully has only a minor impact on the results (compare Figure 10.15 (left) and (right)). Corresponding to Figure 10.15 (left), Figure 10.16 shows the total time, consisting of setup and solve time. Again, all the different configurations yield similarly good weak scaling results with respect to the total time for up to 15 552 processor cores. Included are timings for P2–P1 and RGDSW–RGDSW, a combination that was not included in Figure 10.15 but shown previously in

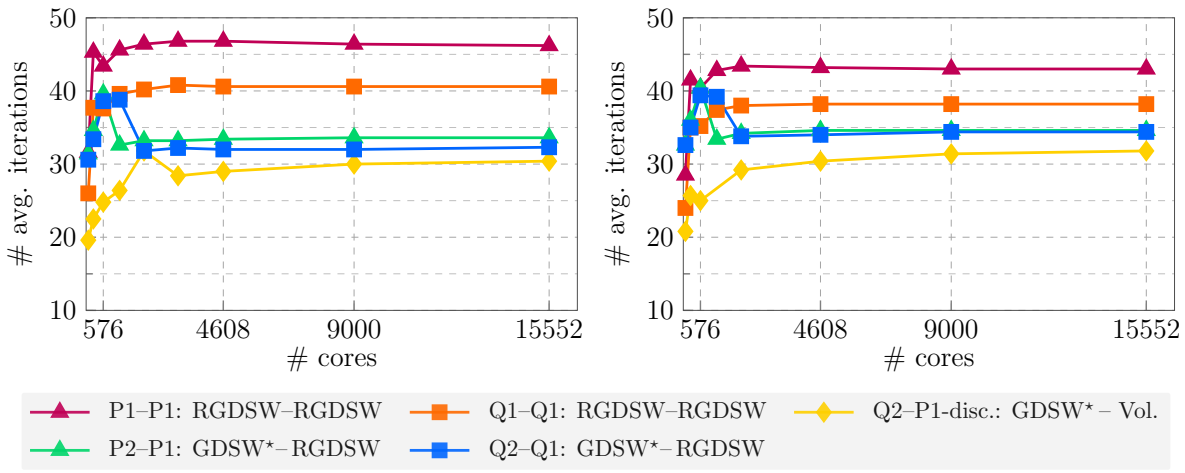


Figure 10.15: Comparison of different finite element discretizations for the monolithic preconditioner. Combinations of different coarse spaces for the velocity (first component) and pressure (second component). Average GMRES iteration count per Newton step. Total number of Newton iterations required to reach convergence is 5. Weak scaling test. Stationary BFS problem with configurations according to Table 9.1. $\mathrm{Re} = 200$. Use of Bochev–Dohrmann stabilization for P1–P1 and Q1–Q1 elements. Pressure projection included in the first level for P2–P1 and Q2–Q1 discretization. **Left:** Excluding off-diagonal blocks in ϕ to build the coarse matrix. **Right:** Using full ϕ to build the coarse matrix. Adapted from [82, Figure 6.6].

Figure 10.14; the results show that for a smaller number of subdomains, the number of iterations and the total time using GDSW* for the velocity are smaller than if RGDSW is used. However, the timings show that this changes for a larger number of subdomains; cf. [82].

Transient Navier–Stokes Problem

For the transient Navier–Stokes problem, again the backward-facing step problem is considered. Therefore, we simulate up to $t = 10.0\,\mathrm{s}$ and choose a time step size of $\Delta t = 0.05\,\mathrm{s}$, giving 200 time steps in total (after which an almost stationary solution is reached). In Figure 9.1 (right) the solution for Reynolds number 3 200 is illustrated. The corresponding definition of the Reynolds number for the BFS problem is shown in (9.1).

The primary objective is to evaluate how the coarse space combinations from the previous section perform in the transient case and how they are influenced by a varying Reynolds number. Specifically, the coarse space combinations that performed best in the stationary case are considered. We omit the pressure projection in the transient case, as it only slightly influences linear convergence but increases the compute time. Furthermore, we focus on the construction of the coarse matrix where the off-diagonal blocks of ϕ are omitted.

In Figure 10.17 we investigate how the average iteration count of the linear solver (over all Newton and time steps) depends on the Reynolds number; see Table A.4 for details. The Reynolds number can be increased in two ways: either by decreasing the viscosity ν (see Figure 10.17 (left)) or by increasing the maximum velocity (see Figure 10.17 (right)). In both cases, the solver and preconditioner combinations are very robust with respect to the Reynolds number.

Both approaches of adjusting the Reynolds number yield similar results for the monolithic

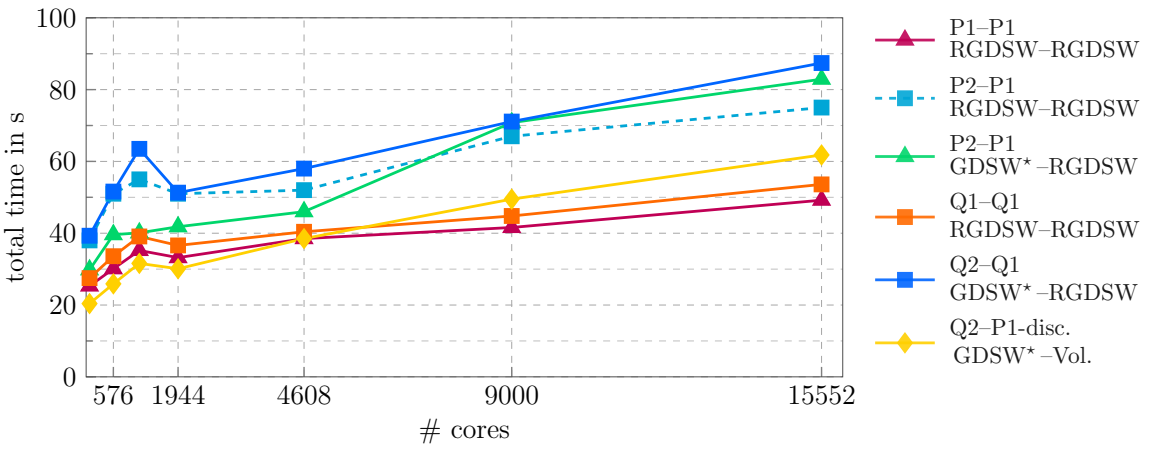


Figure 10.16: Comparison of total time (setup + solve time) for different finite element discretizations for the monolithic preconditioner. Combinations of different coarse spaces for the velocity (first component) and pressure (second component). Average GMRES iteration count per Newton step. Total number of Newton iterations required to reach convergence is 5. Weak scaling test; see Table 10.9 for the number of cores used to solve the coarse problem based on GDSW⋆–RGDSW. Stationary BFS problem with configurations according to Table 9.1. Re = 200. Use of Bochev–Dohrmann stabilization for P1–P1 and Q1–Q1 elements. Pressure projection included in the first level for P2–P1 and Q2–Q1 discretization. Use of full ϕ. Adapted from [82, Figure 6.7].

preconditioner if the off-diagonal blocks of ϕ are omitted; this is not the case if the full ϕ is used or for the block preconditioners PCD and SIMPLEC, as we will see in Section 10.5.3; cf. [82]. When the CFL number increases (Figure 10.17 (right)), the approach using the full ϕ is not robust. However, rebuilding the coarse basis instead of reusing it will improve these results such that the number of iterations is comparable to the other coarse space variants. Nonetheless, since reusing the coarse basis is computationally more efficient and omitting the off-diagonal blocks proved to be more robust, we will continue to use that strategy. It may be, however, that in different circumstances rebuilding the coarse basis ϕ in each Newton step and not discarding the off-diagonal blocks is advantageous; this needs to be further investigated, [82].

Furthermore, we briefly investigate the one-level monolithic overlapping Schwarz preconditioner for this test case. For transient simulations, a one-level method may provide robust performance if the time step is sufficiently small, allowing information to propagate gradually across neighboring subdomains; cf. [78, Section 5.5]. This behavior may be analogous to the influence of the CFL number on information transport. The corresponding results are shown in Figure 10.17 for the P2–P1 discretization. As expected, the average number of linear iterations is influenced by the CFL number. While the two-level variants generally exhibit lower iteration counts and overall robustness, the one-level monolithic Schwarz preconditioner appears to be more robust than the block preconditioning approach for the two highest considered CFL numbers; compare with the results in Figure 10.25. However, the considered number of subdomains is relatively small, and the one-level method loses parallel scalability for an increasing number of subdomains, and is therefore not considered further.

The insight that the CFL number is the relevant metric in the considered case of varying the

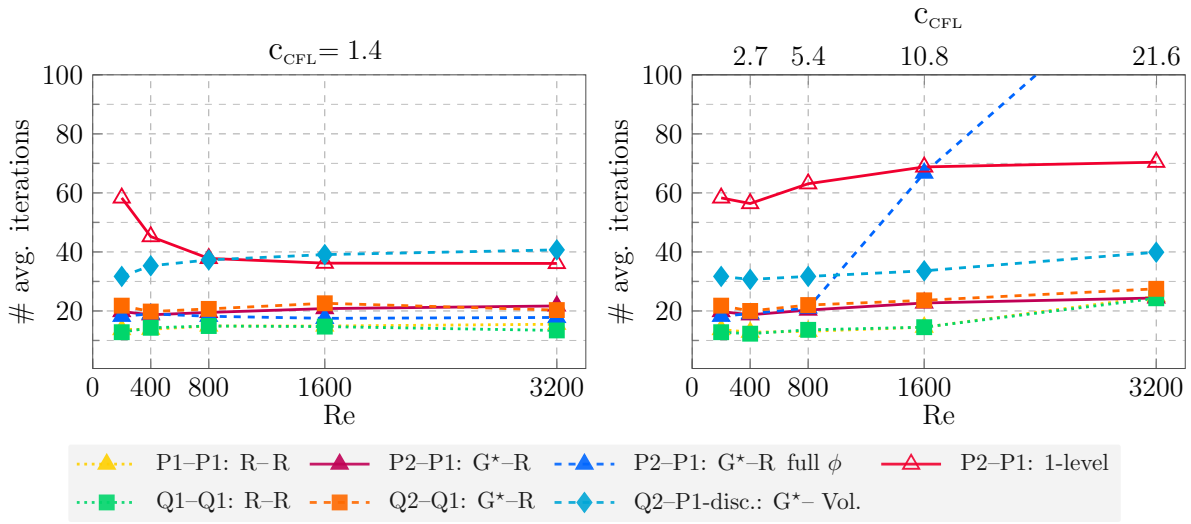


Figure 10.17: Comparison of different finite element discretizations for the monolithic preconditioner for an increasing Reynolds number. For the definition of the Reynolds number, see (9.1). Average GMRES iteration count per Newton step. Transient BFS problem with configurations according to Table 9.1. Time step size is constant, $\Delta t = 0.05\,\mathrm{s}$, simulation until $t = 10\,\mathrm{s}$, which amounts to 200 time steps in total. Use of Bochev–Dohrmann stabilization for P1–P1 and Q1–Q1 elements. 243 processor cores. Combinations of different coarse spaces for the velocity (first component) and pressure (second component). G: GDSW. G*: GDSW*. R: RGDSW. No use of pressure projection. Exclude off-diagonal blocks in ϕ except for one case (see legend). For detailed results, see Table A.4. **Left:** Increasing Reynolds number with constant $v_{\max} = 1\,\mathrm{cm/s}$. The viscosity ν is decreased from $0.01\,\mathrm{cm}^2/\mathrm{s}$ for $\mathrm{Re} = 200$ to $\nu = 6.25 \times 10^{-4}\,\mathrm{cm}^2/\mathrm{s}$ for $\mathrm{Re} = 3\,200$. The CFL number is 1.4. Average number of Newton iterations is between 3.0 and 3.1. **Right:** Increasing Reynolds number with constant $\nu = 0.01\,\mathrm{cm}^2/\mathrm{s}$. The velocity $v_{\max}$ is increased from $1\,\mathrm{cm/s}$ for $\mathrm{Re} = 200$ to $16\,\mathrm{cm/s}$ for $\mathrm{Re} = 3\,200$. The CFL number increases with higher velocity. Average number of Newton iterations is between 3.0 and 4.1. Adapted from [82, Figure 6.8] and extended for one-level results.

Reynolds number in different ways is also reflected by considering the dimensionless form of the Navier–Stokes equations. There, $\mathbf{u}$ is transformed to $\mathbf{u}^* = \mathbf{u}/u_{\mathrm{ref}}$ via a reference velocity (for example, the maximum velocity at the inlet). The relevant influence on the solver by means of changing the Reynolds number via the velocity or the viscosity is then found in the transformed time step $\Delta s = u_{\mathrm{ref}} \Delta t / L$, where L is a characteristic length scale (see [55, p. 334 and p. 412] for details). This is comparable to a CFL number of the type $u_{\mathrm{ref}} \Delta t / \Delta x$, since L and Δx are fixed; cf. [82].

We remark that using a high CFL number may be inadvisable, even for stable time discretizations, depending on the desired accuracy of the numerical solution. It is, however, not a focus of this work to address this but to analyze the robustness of the solvers with respect to different parameters.

10.5.2. Block Preconditioners

We conduct similar tests for the block preconditioners as for the monolithic preconditioner. First, the stationary BFS problem is used to analyze the different block preconditioners with respect

Table 10.11: Comparison of boundary conditions in PCD block preconditioner for stationary BFS problem. For different boundary conditions applicable to PCD, see Table 6.2. P2–P1 discretization with $H/h = 9$, 243 cores, and Re = 200. RGDSW–RGDSW coarse space combination for the velocity and Schur complement components. Average GMRES iteration count per Newton step. Taken from [82, Table 6.7].

Boundary condition strategy	(BC–1)	(BC–2)	(BC–3)
# avg. iterations	> 500	87.7	93.7

to weak scaling. Then, weak scaling and the robustness with respect to an increasing Reynolds number are analyzed for the transient BFS problem. The arising inverses $\hat{F}^{-1}$ and $\hat{S}^{-1}$ of the block preconditioners in Section 6.1 are approximated with different Schwarz preconditioners. In Section 7.2 the application of the two-level overlapping Schwarz preconditioners to block systems is shown. Table 8.4 illustrates which operators specifically are approximated with the overlapping Schwarz methods. In the stationary case, we will approximate the inverse of M_p arising in the PCD preconditioner with the diagonal approximation that is denoted *AbsRowSum*; see Section 8.6.1. This was preferable to approximating the inverse of M_p with a Schwarz method. For the transient case though, approximating M_p with a two-level overlapping Schwarz method proved to be best. The inverse of M_u arising in the LSC-type preconditioners is also approximated via the *AbsRowSum* diagonal scaling; see Section 8.6.2.

A description of the recycling strategies used for the block preconditioners is given in Section 8.6. Depending on the specific block preconditioner, different operators can be reused throughout the simulation.

Stationary Navier–Stokes Problem

Similar to Section 10.5.1 we use the backward-facing step in three dimensions; see Section 9.1. As shown in Table 6.2, different boundary conditions can be applied in the PCD block preconditioner. For the stationary case, using the boundary condition strategy (BC–2) proved to be slightly better than the default (BC–3); see Table 10.11. Consequently (BC–2) will be used for the PCD preconditioner for the stationary BFS problem, but we will revert to the default (BC–3) for transient simulations.

In Table 10.12 the available block preconditioners from `Teko` and a diagonal block preconditioner implemented in `FEDDLib` for the Reynolds numbers 20 and 200 are compared. The diagonal block preconditioner approximates the Schur complement with $-\frac{1}{\nu}M_p$; the inverse of the approximate Schur complement and the inverse of F^{-1} are then approximated with the Schwarz method. The BFBt method is defined by (6.14), as it is a variant of the LSC block preconditioner. Note that the SIMPLE preconditioner is known to not perform well in the stationary case; see, for example, [34, Section 4.2], [52, Section 3], or [157, Section 8.3]. Therefore, it is only briefly considered in Table 10.12.

Only the PCD block preconditioner delivers moderate results. The results for LSC may improve by using an additional scaling, like the ones introduced in [57, 26]; however, we did not see an improvement with the scaling in [26] for our setting, and an implementation of the scaling matrix from [57] is not available to us – see [57] for a derivation of a scaling matrix in a

Table 10.12: Comparison of block preconditioners for different Reynolds numbers for stationary BFS problem. P2–P1 discretization with $H/h = 9$, 1 125 cores. (BC–2) is applied in PCD and LSC_{A_p}. RGDSW–RGDSW coarse spaces for the velocity (first component) and pressure (second component). Average GMRES iteration count per Newton step and number of Newton steps in parentheses. Taken from [82, Table 6.6].

Preconditioner	Diagonal	BFBt	LSC	LSC_{A_p}	PCD	SIMPLEC
Re = 20	127 (4)	180 (4)	169 (4)	154 (4)	82 (4)	278 (4)
Re = 200	266 (5)	292 (5)	257 (5)	207 (5)	103 (5)	316 (5)

two-dimensional setting. Consequently, and particularly later when we compare monolithic with block preconditioners, we will focus mainly on the PCD preconditioner and at times show results based on the SIMPLEC preconditioner for reference.

We select the RGDSW–RGDSW coarse space combination as the default strategy: Table 10.13 shows that different coarse space combinations deliver similar results if the Reynolds number is varied by changing the viscosity. Furthermore, the setup and solve times are comparable; mixing different coarse spaces for PCD, for example GDSW*–RGDSW, does not improve the average iteration count as in the monolithic case. Note that RGDSW has the lowest setup cost, since it has the smallest coarse space dimension, which becomes visible especially for a higher number of processor cores; cf. Figure 10.23 (right).

Due to the decoupling of the subproblems via the block-preconditioning approach, the choice of the coarse space for the two-level overlapping Schwarz method is then secondary as indicated by the results in Figure 10.22 and Table 10.13. These results also show that the PCD block preconditioner is robust once the boundary conditions are correctly specified (cf. Section 6.1.1).

In Figure 10.18 weak scaling results up to 15 552 processor cores are shown for the block-triangular preconditioners PCD and LSC. Most results are based on the application of a two-level Schwarz method to approximate $\hat{F}^{-1}$. For reference, we also show a combination that uses a one-level Schwarz approximation for $\hat{S}^{-1}$. We test the inf-sup stable P2–P1 and Q2–Q1 and the stabilized P1–P1 and Q1–Q1 elements but omit the Q2–P1-disc. element, since the implementation of the discrete pressure Laplace operator A_p cannot be used for discontinuous elements. In that case, a different type of operator is required for PCD and LSC_{A_p}; its implementation is not available to us, and its definition is not straightforward; see [55, p. 368–370].

For PCD we observe weak scalability for all discretizations in Figure 10.18 (left); cf. [82]. Using only a one-level overlapping Schwarz preconditioner for the Schur complement proves to be inadequate; the average iteration count exceeds 400.

The results using LSC are less clear; see Figure 10.18 (right) and Figure A.4. $\text{LSC}_{\text{stab},A_p}$, LSC_{stab}, and PCD show comparably good scaling results for the stabilized Q1–Q1 and P1–P1 elements; see Figures 10.18 and A.4. However, unstabilized LSC and LSC_{A_p} for stable discretizations do not scale well; for a P2–P1 discretization, both exceed 300 average GMRES iterations. Using Q2–Q1 elements, the convergence for LSC_{A_p} is significantly better than for P2–P1 but still significantly worse than for equal-order elements. Both discretizations show the h sensitivity of the LSC preconditioner; compare, for example, with [26, Table 4]. Only the stabilized LSC variants are close to reaching an asymptotically constant number of GMRES

Table 10.13: Comparison of different coarse spaces for PCD block preconditioner with different Reynolds numbers for P2–P1 discretization. Combinations of different coarse spaces for the velocity (first component) and Schur complement components (second component). Stationary BFS problem with $H/h = 9$. (BC–2) is applied in PCD. 1 125 cores. Reynolds number is increased by decreasing the viscosity. For the definition of the Reynolds number, see (9.1). Average GMRES iteration count per Newton step and number of Newton steps in parentheses. Total time consists of setup and solve time. Taken from [82, Table A.1].

Coarse space	Velocity Pressure	GDSW GDSW	GDSW⋆ GDSW⋆	GDSW⋆ RGDSW	RGDSW RGDSW
Re = 20	# avg. iterations	76 (4)	79 (4)	76 (4)	82 (4)
	Setup time	15.0 s	13.7 s	15.0 s	12.9 s
	Solve time	35.3 s	35.1 s	33.8 s	35.2 s
	Total	50.3 s	48.8 s	48.8 s	48.1 s
Re = 200	# avg. iterations	98 (5)	100 (5)	102 (5)	103 (5)
	Setup time	15.5 s	14.2 s	13.9 s	12.9 s
	Solve time	57.7 s	56.7 s	57.2 s	56.9 s
	Total	73.2 s	70.9 s	70.1 s	69.8 s
Re = 400	# avg. iterations	163 (6)	167 (6)	174 (6)	173 (6)
	Setup time	15.9 s	13.8 s	13.7 s	13.0 s
	Solve time	122.3 s	120.2 s	125.0 s	119.8 s
	Total	138.2 s	134.0 s	138.7 s	132.8 s

iterations. See also [157] (e.g., Table 11) for some related results of the LSC preconditioner and the stationary BFS problem or [26] for a comparison of PCD and LSC preconditioners for a lid-driven cavity problem using Q2–Q1 elements.

Augmented Lagrangian Preconditioners

As discussed before, when ν is relatively large, leading to a moderate Reynolds number, the Schur complement S is well conditioned and can be well approximated by the pressure mass matrix M_p. Block preconditioners from the previous section can deliver suitable approximations for the Schur complement; see the PCD block preconditioner in Table 10.12 and Figure 10.18. In practice, the augmented Lagrangian (Section 6.1.4) approaches can lead to iteration counts that are almost independent of mesh size and only mildly dependent on the viscosity, or Reynolds number, for a good choice of γ, i.e., they are both scalable and Reynolds-robust; see [17, 19, 20].

One drawback of the augmented Lagrangian approach is that the (0,0) velocity block ($F + \gamma B^T M_p^{-1} B$) has an increased sparsity pattern and a larger stencil (see Figure 10.19) for a continuous pressure discretization, which can lead to increased computational cost. This aspect is investigated in the following. To this end, we compare the performance of the augmented Lagrangian preconditioner to the PCD block-triangular preconditioner, which was the best performing block preconditioner in the current setting for the stationary Navier–Stokes problem.

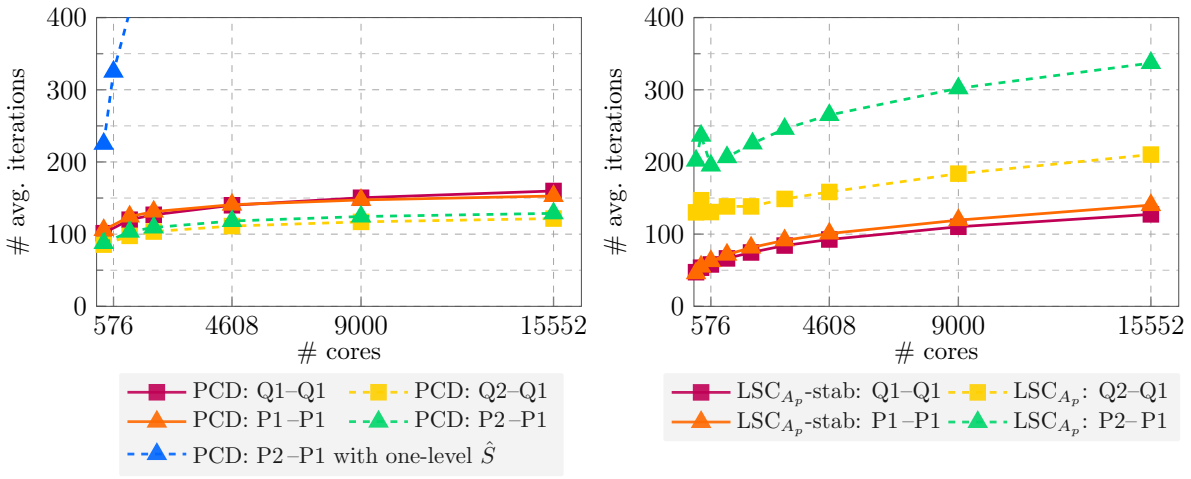


Figure 10.18: Comparison of different finite element discretizations for block-triangular preconditioners. Use of RGDSW–RGDSW (R–R) coarse space for the velocity (first component) and for Schur complement components (second component) for both PCD and LSC (except for one result using PCD; see legend). (BC–2) boundary condition strategy used for PCD and LSC. Average GMRES iteration count per Newton step. Total number of Newton iterations required to reach convergence is 5. Weak scaling test. Stationary BFS problem with configurations according to Table 9.1. Re = 200. Use of Bochev–Dohrmann stabilization for P1–P1 and Q1–Q1 elements. No use of pressure projection. **Left:** PCD block preconditioner. One-level Schwarz approximation for $\hat{S}^{-1}$ denoted as one-lvl $\hat{S}$. **Right:** LSC$_{A_p}$ block preconditioner; see Section 6.1.2. Adapted from [82, Figure 6.9].

We consider the three-dimensional stationary BFS problem on 72 subdomains with $H/h = 9$ and P2–P1 discretization, and no reuse of the preconditioner during the Newton iterations. Furthermore, a constant forcing term for GMRES, which is 10^{-6}, is used. Compared to the classical AL approach, the modified AL preconditioner in (6.23) exhibits a sparser matrix structure, as the lower block-triangular part is omitted. For this test case, the number of nonzero entries for the velocity block A_γ is $8.68{\cdot}10^9$ for the classic AL approach, and $5.79{\cdot}10^9$ in $\hat{A}_\gamma$ for the modified AL approach; see (6.21) and (6.22). This is approximately 6 and 4 times as many nonzero entries compared to the classical Navier–Stokes (0,0) block, which has $1.38{\cdot}10^9$; see also 10.15.

Due to the increased finite element stencil caused by the additional term $B^T M_p^{-1} B$, the map definition used for the initial overlapping matrix, i.e., at $\hat{\delta} = 1$, was adjusted; see Section 8.5.1 for the corresponding analysis and discussion. Therefore, the modified map definition shown in Figure 8.8 (b) is used for the AL system. This ensures that the overlap is identical to that of the original Navier–Stokes system, allowing a fair comparison between the AL and PCD preconditioners without introducing the additional influence of a larger overlap.

First, we will experimentally determine a good value for γ for a variation of Reynolds numbers, and the original and modified AL preconditioner. In [19, Table 2.2] experimentally determined values of γ are compared to values derived by Fourier Analysis. For the problem in [19, Table 2.2] the optimal γ varied depending on viscosity, ranging from 0.45 to 0.017. The results indicate that for decreasing viscosity, the optimal γ also decreases. In Table 10.14 a good value for γ is determined for the AL and modified AL approach. The Reynolds number is increased by

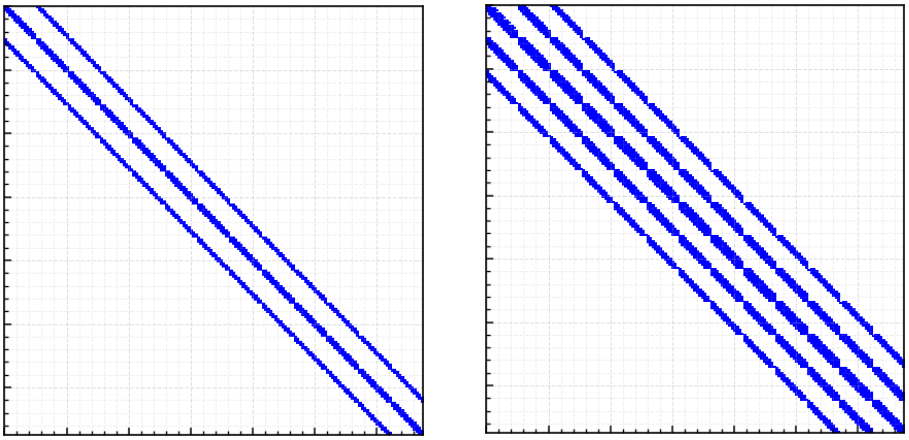


Figure 10.19: Sparsity pattern of velocity block (0,0) of linearized Navier–Stokes saddle point system. **Left:** Standard Navier–Stokes approach with 4324 nonzero entries in F. **Right:** Augmented Lagrangian approach with 10 756 nonzero entries in $F + \gamma B^T M_p^{-1} B$.

decreasing the viscosity. The values range from $\nu = 0.01$ to $\nu = 0.002$ which leads to a Reynolds number of Re = 200 to Re = 1 000. A $\gamma = 0.01$ value appears to deliver the best average iteration count for the considered problem for both approaches. Some configurations were not computed for an increasing Reynolds number, when a lower Reynolds number already delivered high iteration counts. This is indicated by "–".

In Table 10.15, the results of the AL and modified AL approach, for the best experimentally determined $\gamma = 0.01$, are compared to the triangular and PCD block preconditioner, and the monolithic preconditioner. For Reynolds numbers Re = 200 and Re = 1 000 the average iteration count of the augmented Lagrangian approaches is more robust compared to the triangular and PCD block preconditioners.

Table 10.14: Experimental analysis of optimal value γ for augmented and modified Lagrangian approach; see Section 6.1.4. Average linear iteration count per Newton step. Required Newton steps are in parentheses in the left column. The results for the modified AL approach are in parentheses. Three-dimensional BFS problem with Reynolds number ranging from 200 to 1 000 by decreasing ν. 72 subdomains with H/h=9 and P2–P1 discretization. No reuse of full preconditioner during the Newton iterations. Overlap $\hat{\delta} = 1$. Some configurations were not computed for increasing Reynolds number, when lower Reynolds number already delivered high iteration counts. This is indicated by "–".

ν \ γ	0.2	0.1	0.05	0.025	0.01	0.001
0.01 (5)	133 (247)	110 (164)	92 (116)	80 (91)	79 (84)	147 (150)
0.005 (6)	155 (–)	129 (271)	107 (172)	89 (118)	87 (97)	212 (221)
0.0025 (7)	179 (–)	149 (–)	123 (272)	105 (170)	97 (116)	—
0.002 (7)	—	156 (–)	130 (–)	111 (195)	99 (123)	—

When we employ the augmented Lagrangian approach, one important difference in the Navier–Stokes saddle point system is the enlarged finite element stencil and sparsity pattern; see

Table 10.15: Comparison of runtime for various preconditioners with the augmented Lagrangian preconditioner; see Section 6.1.4. Average linear iteration count per Newton step. Results of triangular preconditioner, AL and modified AL approach, and PCD. Additionally the monolithic results are shown. 72 cores, for 3D BFS problem, P2–P1 discretization, $H/h = 9$, $\gamma = 0.01$ for $\mathrm{Re} = 200$ and $\mathrm{Re} = 1\,000$. The overlap is $\hat{\delta} = 1$. Constant forcing term and no use of (NR-k).

Prec.	$\mathcal{B}_{\mathrm{Triag}}$	$\mathcal{B}_{\mathrm{AL}}$	$\mathcal{B}_{\mathrm{ModAL}}$	$\mathcal{B}_{\mathrm{PCD}}$	$\mathcal{B}_{\mathrm{Monolithic}}$
# nnz entries F	$1.38{\cdot}10^9$	$8.68{\cdot}10^9$	$5.79{\cdot}10^9$	$1.38{\cdot}10^9$	–
Re=200, Newton iterations: 5					
# avg. iter.	230	79	84	108	52
setup time	14 s	161 s	145 s	14 s	18 s
solve time	83 s	103 s	110 s	37 s	18 s
Re=1 000, Newton iterations: 7					
# avg. iter.	732	99	123	200	69
setup time	18 s	214 s	191 s	19 s	24 s
solve time	497 s	182 s	221 s	103 s	34 s

Figure 10.19. Even though we get a stable approach with respect to the Reynolds number, for a good choice of γ, the setup and solve costs are still higher compared to the other considered approaches; see Table 10.15. Figure 10.20 shows that there is potential to decrease the runtime of the augmented Lagrangian approach. In particular, the symbolic factorization, which is included in the initial setup, and the numeric factorization result in higher setup time than those of the classic triangular preconditioner. This analysis of certain components of the setup and solve time indicates that the increased number of nonzeros has a significant influence on the symbolic and numeric factorization of the direct solver; see Figure 10.20 light green, light red and light blue.

Instead of employing a direct solver for the first level of the overlapping Schwarz preconditioner, an approximate solver can be used. To this end, the `Ifpack2` package is considered (Section 8.4.2) with the RILUK approximate solver. Instead of computing an exact LU factorization, RILUK constructs an incomplete factorization. We consider a fill-in level of zero, restricting the sparsity pattern of the approximate $\hat{L}$ and $\hat{U}$ to the original system. Three block preconditioners are directly compared: the triangular (green), augmented Lagrangian (blue), and modified augmented Lagrangian (red). The comparison in Figure 10.20 demonstrates how the approximate first level solver can improve the runtime. The darker shades correspond to the incomplete LU factorization, the lighter shades to the direct solver MUMPS. For the modified augmented Lagrangian preconditioner, using an approximate solver for the first level can improve the total time — setup plus solve time — from 252 s to 104 s. Even though the triangular preconditioner has a higher average iteration count, which is denoted in the parentheses, the total time is still lower with 49 s.

We revisit the comparison from Table 10.15, now employing RILUK as first level approximate solver for the triangular, AL, and modified AL block preconditioners. In addition, we make use of the adaptive forcing term with $\eta_{\mathrm{init}} = \eta_{\mathrm{max}} = 10^{-3}$ and constant forcing term for $\mathrm{Re} = 1\,000$.

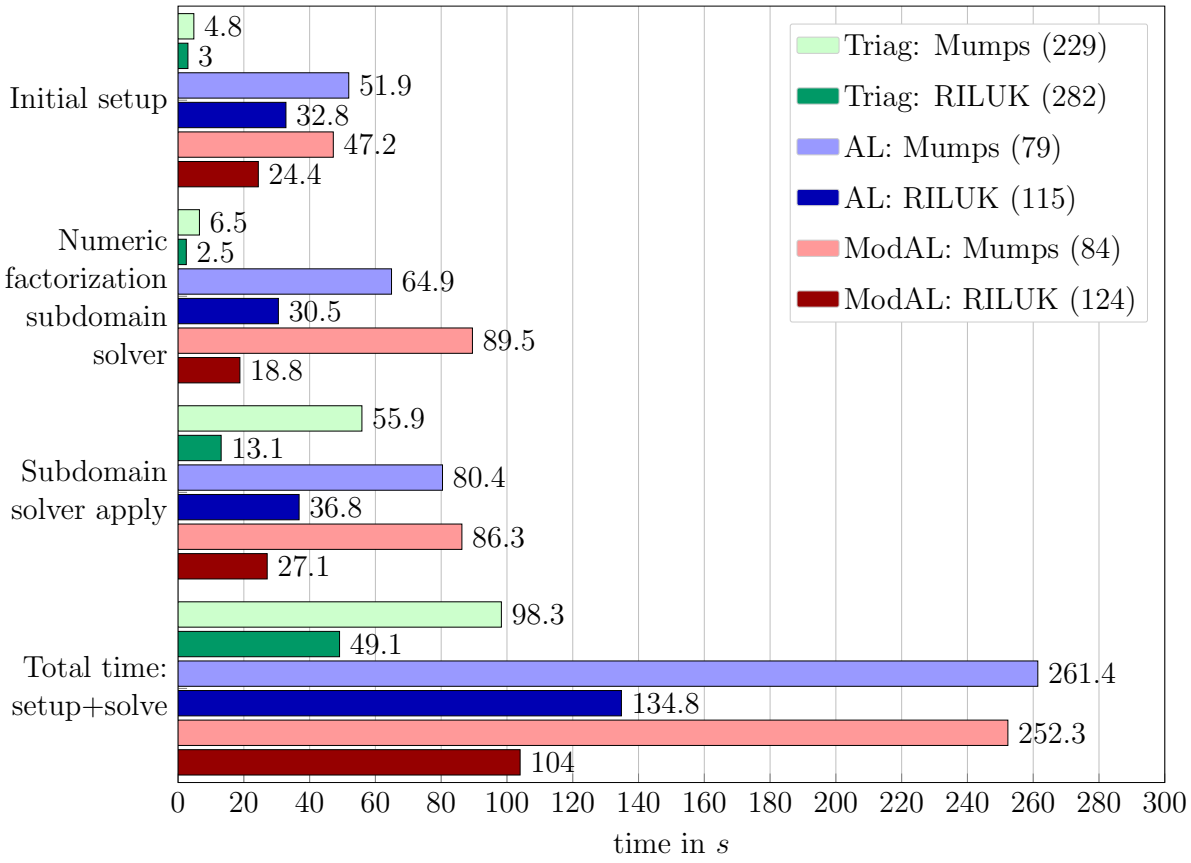


Figure 10.20: Comparison of different components of setup and solve time for the triangular, augmented Lagrangian, and modified augmented Lagrangian block preconditioners. Three-dimensional BFS test with 72 subdomains, H/h = 9, P2–P1 discretization, and Re = 200. The configuration of the respective preconditioners differs in the used subdomain solver. As a direct solver, MUMPS is used. As an approximate solver RILUK is chosen from the `Ifpack2` package; see 8.4.2. The level of fill-ins is 0, and dropping threshold is 10^{-5}. Average linear iteration count per Newton step is denoted in parentheses. No reuse of the full preconditioner during the Newton iteration was considered, and constant forcing term of 10^{-6} was applied.

Furthermore, the full preconditioner is reused after $k = 3$ Newton iterations in the Re = 200 case, and $k = 5$ iterations in the Re = 1 000 case. The corresponding results are shown in Table 10.16. Although the runtime is significantly reduced, the use of an approximate first-level solver leads to an increased number of linear iterations; see the average iteration counts displayed in Figure 10.20 in parentheses. This is especially noticeable for the Re = 1 000 case. Based on these results, the augmented Lagrangian block preconditioner is not included in the further comparisons with the monolithic preconditioner. Additional tests are performed in which the augmented Lagrangian block preconditioner is combined with algebraic multigrid methods (AMG).

Table 10.16: Comparison of runtime for various preconditioners with use of RILUK (Section 8.4.2) for first level in triangular preconditioner, augmented Lagrangian and modified augmented Lagrangian block preconditioner. Average linear iteration count per Newton Step. Results of triangular preconditioner with and without AL approach, and PCD, for 72 cores, for 3D BFS problem, P2–P1 discretization, $H/h = 9$, $\gamma = 0.01$ for $\mathrm{Re} = 200$ and $\mathrm{Re} = 1\,000$. The overlap is $\hat{\delta} = 1$. Use of an adaptive forcing term with $\eta_{\text{init}} = \eta_{\max} = 10^{-3}$ and constant forcing term for $\mathrm{Re} = 1\,000$. Furthermore, the full preconditioner is reused after $k = 3$ Newton iterations in the $\mathrm{Re} = 200$, and $k = 5$ iterations in the $\mathrm{Re} = 1\,000$ case.

Prec.	$\mathcal{B}_{\text{Triag}}$	$\mathcal{B}_{\text{AL}}$	$\mathcal{B}_{\text{ModAL}}$	$\mathcal{B}_{\text{PCD}}$	$\mathcal{B}_{\text{Monolithic}}$
# nnz entries F	$1.38{\cdot}10^9$	$8.68{\cdot}10^9$	$5.79{\cdot}10^9$	$1.38{\cdot}10^9$	–
$\mathrm{Re} = 200$, Newton iterations: 6					
# avg. iter.	176	65	71	74	31
setup time	5.4 s	55.9 s	40.8 s	9.3 s	11.7 s
solve time	27.7 s	37.8 s	32.2 s	29.2 s	10.9 s
$\mathrm{Re} = 1\,000$, Newton iterations: 7					
# avg. iter.	852	213	235	225	78
setup time	5.3 s	66.1 s	48.2 s	11.8 s	14.4 s
solve time	352.1 s	154.9 s	134.9 s	117.4 s	38.2 s

Modified Augmented Lagrangian Combined with Multigrid

In many sources in the literature, the augmented Lagrangian approach is combined with multigrid methods instead of using overlapping Schwarz methods to approximate the arising inverses. For instance, in [18, 20] the `ML` package from `Trilinos` is employed in combination with the AL approach. Through the `Teko` block preconditioning package (Section 8.6), it is also possible to use the more recent algebraic multigrid package version of `ML`, called `MueLu`, which provides support of the `Tpetra` linear algebra stack. The package is based on smoothed aggregation algebraic multigrid (SA-AMG) methods; see Section 8.4.3.

The augmented Lagrangian preconditioner in combination with the AMG approach is briefly investigated in the following. Most SA-AMG parameters are retained at their (`MueLu`) default values, e.g., the minimum and maximum aggregation size, while only a small number of key settings are modified; see Table 10.17. In [20] the multigrid setup is proposed to be simple, due to the augmented Lagrangian formulation. Therefore, simple smoothers should suffice to achieve sufficient results; see also Table 10.18 where different smoothers are compared. We consider the settings listed in Table 10.17 which are based on [20, Table 1] and [69]. Choosing a maximum coarse problem size of 256 results in a three-level hierarchy (level 0 plus two constructed coarser levels), with the coarsest problem containing approximately 200 rows. Aggregation values, e.g., the minimum and maximum aggregation values, are kept at default values; see Table 10.17. For smoothing the symmetric Gauss–Seidel smoother is used in [20]. Additionally, the standard Gauss–Seidel and RILUK smoothers are considered here; see Section 8.4.2 for RILUK. The standard Gauss–Seidel is less expensive, compared to the symmetric version, yet, the symmetric version damps error modes generally better; the symmetric version consists of

Table 10.17: Parameters for the smoothed aggregation algebraic multigrid solver from the Trilinos package `MueLu`. Parameters based on [20, Table 1] and [69].

max. levels	coarse solver	max. coarse size	smoother	sweeps	damping	min. aggreg.	max. aggreg.	dropping threshold
10	pardisomkl	256	{(S)GS, RILUK}	{1,2,3}	0.67	2	9	0.0

Table 10.18: Comparison of runtime for modified augmented Lagrangian block preconditioner using `MueLu` package. Three-dimensional BFS problem on 72 cores, with P2–P1 discretization, $H/h = 9$, and $\gamma = 0.01$ for Re=200. Reuse of preconditioner after $k = 3$ Newton iterations (NR-3). (*) RILUK is only used for presmoothing and Gauss–Seidel for postsmoothing.

Smoother	`Symmetric Gauss-Seidel`			`Gauss-Seidel`			`RILUK`(*)
Sweeps	3	2	1	3	2	1	1
# avg. iter.	50	53	61	55	60	77	50
setup time	4.6 s	4.5 s	4.6 s	4.4 s	4.5 s	4.4 s	19.1 s
solve time	95.8 s	72.6 s	51.0 s	57.1 s	48.4 s	43.8 s	33.5 s
total time	100.4 s	77.1 s	55.6 s	61.5 s	52.9 s	**48.2 s**	52.6 s

Table 10.19: Comparison of average linear iterations for modified augmented Lagrangian block preconditioner using `MueLu` package and different damping values. Three-dimensional BFS problem on 72 cores, with P2–P1 discretization, $H/h = 9$, and $\gamma = 0.01$ for Re=200. Reuse of preconditioner after $k = 3$ Newton iterations (NR-3). Gauss–Seidel smoother with one sweep.

damping	0.5	0.6	0.67	0.7	0.75	0.8	0.85	0.9
# avg. iter.	92	82	77	74	72	69	67	82

one forward and one backward sweep [166]. The incomplete LU factorization used through RILUK is the most expensive smoother considered. The results in Table 10.18 illustrate that the use of AMG to approximate the arising inverses can significantly reduce the setup cost for the modified augmented Lagrangian approach compared to the two-level overlapping Schwarz methods; compare to Table 10.16. The solve time is in a similar range as before. Additionally, in Table 10.19 different damping parameters are tested for the Gauss–Seidel smoother with one sweep. The damping can have an effect on the average GMRES iteration count. The total runtime, e.g., for damping parameter 0.85 decreases to 42.2 s with 4.6 s for the setup and 37.6 s for the solve time compared to Table 10.18.

Lastly, the weak scaling behavior of the considered AMG method is evaluated; see Figure 10.21. The chosen aggregation parameters result in a relatively weak coarsening strategy, where consequently a three-level hierarchy for all considered processor counts was chosen. Note that forcing

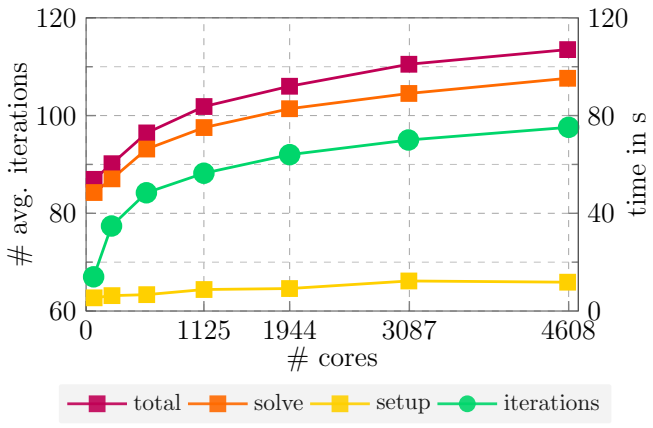


Figure 10.21: Weak scaling results for the modified augmented Lagrangian approach using the `MueLu` algebraic multigrid preconditioner. Stationary three-dimensional BFS problem with degrees of freedom according to Table 9.1 with P2–P1 finite element discretization and $H/h = 9$. $\mathrm{Re} = 200$. (NR-3). Average linear iteration count per Newton step on the left axis. Timings for setup, solve and total time in s on the right axis.

the construction of an additional level yields virtually no further coarsening. In particular, the coarsening ratio between the last two levels is only 1.01, indicating that the aggregation process has essentially reached its limit. Consequently, an additional multigrid level increases both the setup and application costs of the hierarchy without providing a meaningful reduction of the coarse problem. For example, for the case of 4 608 processor cores, the coarsest level consists of 13 824 rows, corresponding to an average of only three rows per processor during the coarse solve. Note that forcing more levels does not decrease the dimension of the coarsest level, due to the aggregation strategy.

Therefore, communication overhead can dominate the computational costs of the coarse-level solve. To address this issue, repartitioning strategies can be employed. The underlying idea is similar to the strategy used for the coarse problem in the two-level overlapping Schwarz method at large core counts where the coarse problem is distributed from one rank to a designated number; see Table 10.9. Conversely, instead of solving the coarse problem on all available processor cores, the coarsest AMG level is redistributed to a significantly smaller subset of processors. Repartitioning is applied to the coarsest level, i.e., Level 2, using a minimum of 250 and a target of 750 rows per processor. For the problem with 4 608 processor cores, this results in 18 processors executing the coarse solve.

Now, the results in Figure 10.21, where the left axis indicates the average linear iteration count per Newton step, and the right axis the setup, solve and total time, are comparable to the results of, e.g., PCD in Figure 10.23. Note that, for the AMG method, other settings and modifications could be used to further improve results, but that is beyond the focus and scope of this work.

Transient Navier–Stokes Problem

As the PCD preconditioner gave consistently good results in the previous section, we focus our analysis on it for the transient backward-facing step problem but also show results for SIMPLEC as a reference. For the augmented Lagrangian approach, the combination with two-level overlapping Schwarz methods delivered robust results, see, e.g., Table 10.15, but due to the high setup and solve cost compared to the other block preconditioning approaches, it will not be considered for the transient simulations.

We iterate for 50 time steps with the step size $\Delta t = 0.02\,\mathrm{s}$. As stated in Section 6.1.1, different

boundary conditions applied to the Schur complement components in the PCD preconditioner can affect the performance. Unlike for the stationary case in Table 10.11, where strategy (BC–2) worked best and (BC–1) failed to converge in a decent number of iterations, for the transient case, all three strategies perform equally well; see Table 10.20 in the appendix. Thus, we will revert to the default setting of (BC–3) for transient simulations.

Table 10.20: Comparison of different boundary condition strategies for the PCD block preconditioner and a transient simulation. Boundary conditions in PCD are defined according to Section 6.1.1. Transient BFS problem with P2–P1 discretization. Average iteration count per Newton step over all time steps, number of Newton steps in parentheses. GDSW–GDSW coarse space for the velocity (first component) and pressure (second component). Number of processor cores is 243. Time step $\Delta t = 0.02\,\mathrm{s}$, simulation until $t = 1.0\,\mathrm{s}$, which amounts to 50 time steps. (*) Results corresponding to Figure 10.22 (left). Taken from [82, Table B.2]

Boundary condition strategy	(BC–1)	(BC–2)	(BC–3) (*)
# avg. iterations	33.5 (3.1)	33.4 (3.1)	33.4 (3.1)

We analyze the weak scaling properties of the PCD and SIMPLEC block preconditioners with Figure 10.22. Similar to the stationary problem, using a one-level method for the pressure component is not feasible due to much higher iteration counts. The other variants of PCD perform similarly to the stationary case; using both two-level methods for the velocity and pressure inverse approximations. Both methods show weak scaling; PCD is weakly scalable and offers a lower average iteration count compared to SIMPLEC.

10.5.3. Comparison of Specific Monolithic and Block Preconditioners

This subsection is taken from [82, Section 6.4]. Modifications to the figure styles have been made to improve consistency with the remainder of this dissertation.

For the comparison of monolithic and block preconditioners, we address the configurations that gave the most robust results in the previous sections; for the monolithic preconditioner we consider the GDSW*–RGDSW combination, for the block preconditioners RGDSW for both pressure and velocity component. Furthermore, we focus mostly on PCD as a block preconditioner but show a number of SIMPLEC results for reference. For the weak scaling behavior, the backward-facing step problem is again considered; for a strong scaling study, the realistic patient-specific artery is used; see Figure 9.7 for the fluid domain. Additionally, the preconditioners' robustness is studied by increasing the Reynolds number. This is especially relevant for physically realistic scenarios. In [37] similar aspects relevant to hemodynamics simulations were analyzed; among other tests, the authors compared the SIMPLE and PCD block preconditioners for varying viscosities and Reynolds numbers.

Backward-Facing Step

Stationary Simulation In Figure 10.23 the average iteration count and total time for the PCD and monolithic preconditioners are shown. Since the average iteration count of the monolithic

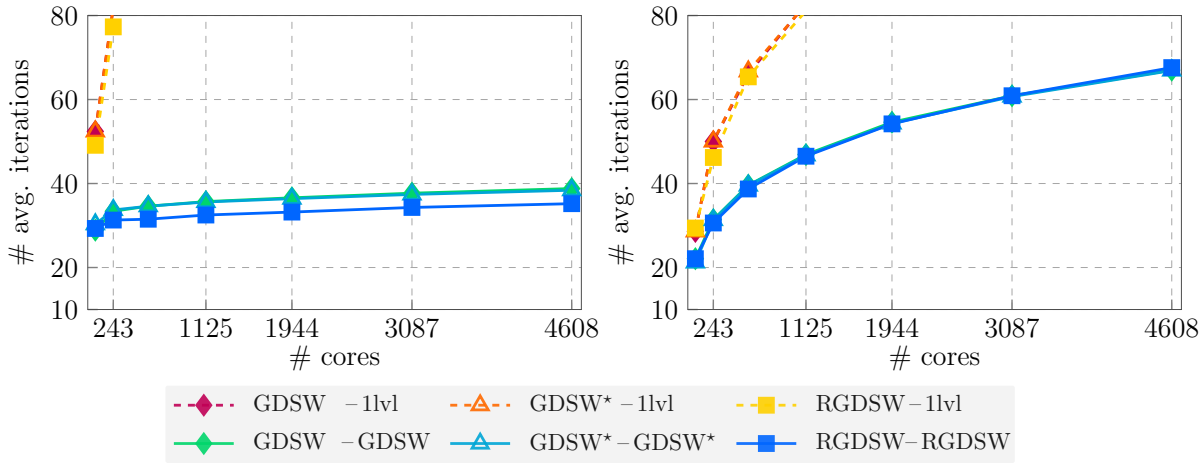


Figure 10.22: Comparison of different coarse space combinations for the velocity (first component) and Schur complement components (second component) for P2–P1 discretization for block preconditioners. Weak scaling test for different combinations of one- and two-level approaches. Average GMRES iteration count per Newton step over all time steps. Average number of Newton iterations per time step required to reach convergence is 3. Transient BFS problem with $H/h = 9$. Time step $\Delta t = 0.02\,\mathrm{s}$, simulation until $t = 1.0\,\mathrm{s}$, which amounts to 50 time steps. $\mathrm{Re} = 200$. The CFL number varies due to the increasing mesh resolution and constant time step size. The maximum CFL number is 1.44 for 4 608 cores. **Left:** Pressure convection–diffusion (PCD) block preconditioner with (BC–3) boundary condition strategy for the Schur complement setup. **Right:** SIMPLEC block preconditioner. Adapted from [82, Table 6.10].

preconditioner is significantly lower, the total time is much lower as well. Despite a slightly lower number of iterations, the RGDSW–RGDSW combination for PCD performs better in case of the P2–P1 discretization and a large number of processor cores compared to the GDSW*-GDSW*coarse space combination, which can be attributed to the smaller cost associated with the coarse problem; cf. [82].

It stands out that stabilized P1–P1 elements lead to smaller compute times compared to P2–P1, despite a larger iteration count and a slightly larger number of degrees of freedom (cf. Table 9.1). This is probably due to a more favorable sparsity pattern. However, since a P2–P1 discretization should give a better finite element approximation, one cannot infer from the results that P1–P1 elements are preferable.

Transient Simulation The transient backward-facing step simulation consists of 200 time steps with $\Delta t = 0.05\,\mathrm{s}$ until $t = 10\,\mathrm{s}$ is reached. We restrict ourselves to a P2–P1 discretization and the monolithic preconditioner with the GDSW*–RGDSW or RGDSW–RGDSW coarse space combination; PCD and SIMPLEC always use RGDSW–RGDSW. For the comparison of monolithic and block preconditioners, we will consider three aspects: Firstly, we will compare the robustness with respect to an increasing Reynolds number. The Reynolds number is varied by adjusting the viscosity. As we have seen before, changing the velocity instead is more challenging, as this also changes the CFL number. In Figure 10.24 (left), the average linear iteration count is shown for an increasing Reynolds number. Detailed iteration counts and timings are included in the appendix in Table A.5. The PCD and monolithic preconditioners are robust with respect

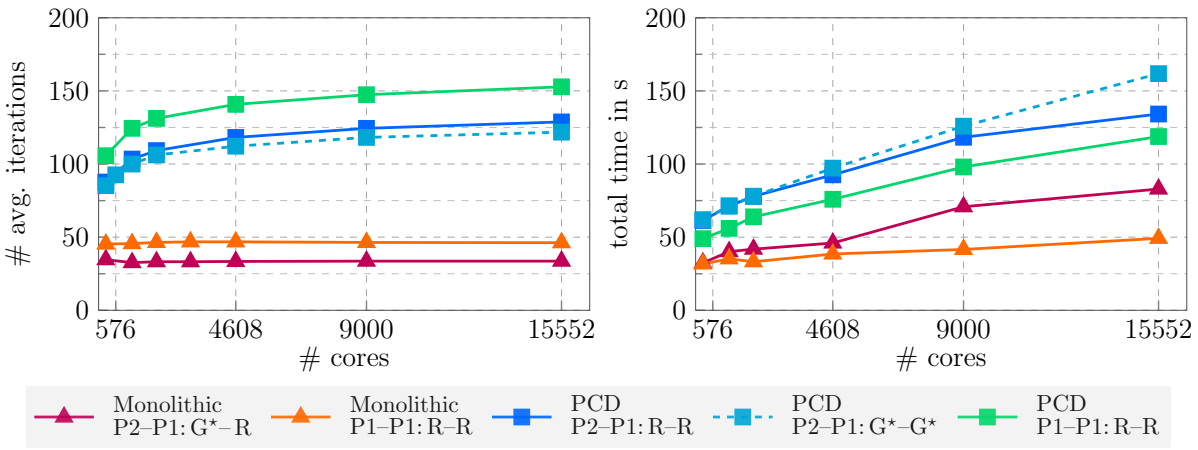


Figure 10.23: Weak scaling comparison of block and monolithic preconditioners. Use of GDSW⋆ (G⋆) and RGDSW (R) coarse spaces for the velocity (first component) and pressure (second component). Stationary BFS problem with configurations according to Table 9.1. Re = 200. Use of Bochev–Dohrmann stabilization for P1–P1 elements. Pressure projection included in the first level for monolithic preconditioner with P2–P1 discretization. Exclude off-diagonal blocks in ϕ for monolithic preconditioner. Boundary condition strategy for PCD is (BC–3). **Left:** Average GMRES iteration count per Newton step. Total number of Newton iterations required to reach convergence is 5. **Right:** Total time in seconds consisting of setup and solve time. Adapted from [82, Table 6.11].

to the increase of the Reynolds number, but also SIMPLEC shows only a small increase in the number of iterations. Despite using a larger coarse space for the velocity component, the monolithic preconditioner requires the least amount of CPU time; see Table A.5 and Figure 10.24 (right).

Secondly, we will consider robustness with respect to an increasing CFL number. In Figure 10.24 the CFL number remained constant. In Figure 10.25 we raise the Reynolds number by increasing the maximum inflow velocity, which increases the CFL number, since the time step size remains constant. For the monolithic preconditioner, the average iteration count per Newton step over all time steps is still almost constant over the range 1.35 to 21.6 of CFL numbers. PCD and SIMPLEC, however, show a large increase of the average iteration count with the increasing CFL number. A sensitivity of PCD and SIMPLEC with respect to the CFL number was also observed in [34, Figure 9]. Not reusing the coarse basis has almost no impact on the results; compare Figure 10.25 (left) with (right). This justifies the reuse of the coarse basis throughout the simulation process to save setup time.

Thirdly, we test the weak scaling behavior; see Table 10.21. Due to the increasing number of processor cores, the mesh resolution increases and in turn also the CFL number. For 4 608 processor cores, the CFL number is still moderate with $c_{\text{CFL}} = 3.6$. Based on the results in Figure 10.25, this should not impact the average iteration count by much. All preconditioners, except for SIMPLEC, perform well. The results for 4 608 cores and SIMPLEC were not computed due to the high computational cost.

Due to the low number of iterations achieved with the monolithic preconditioner and GDSW⋆–RGDSW, this combination delivers by far the lowest solve time. Consequently, with respect to

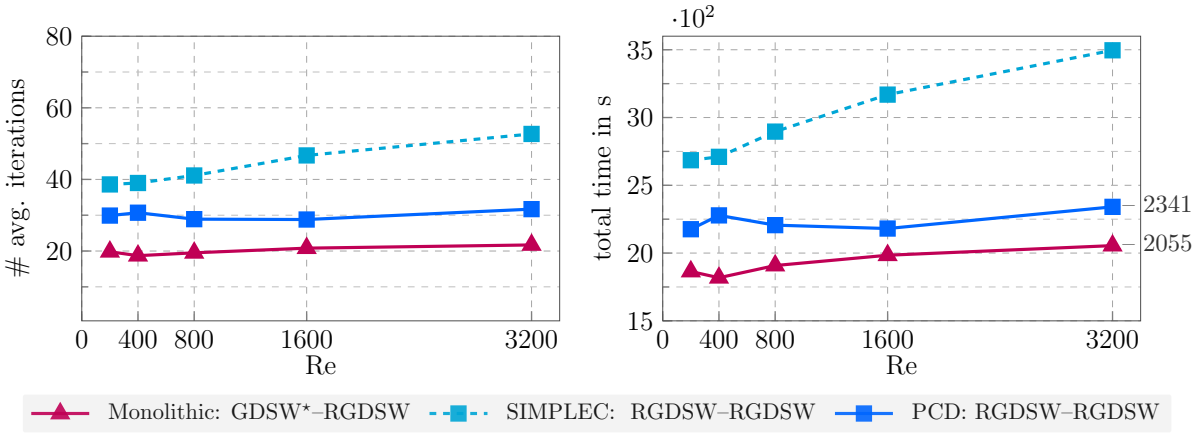


Figure 10.24: Comparison of block and monolithic preconditioners for increasing Reynolds number. Use of GDSW⋆ and RGDSW coarse spaces for the velocity (first component) and pressure (second component). Transient BFS problem with P2–P1 discretization and $H/h = 9$. Increasing Reynolds number due to decreasing viscosity. No pressure projection is used. Exclude off-diagonal blocks in ϕ for monolithic preconditioner. Computation of 200 time steps with time step size $\Delta t = 0.05\,\mathrm{s}$, simulation until $t = 10\,\mathrm{s}$. CFL number is constant and equal to 1.35. Computations on 243 cores. Boundary condition strategy in PCD is (BC–3). See Table A.5 for detailed numbers corresponding to the figure. **Left:** Average GMRES iteration count per Newton step per time step. Average number of Newton iterations per time step required to reach convergence is 3. **Right:** Total time in seconds consisting of setup and solve time. Adapted from [82, Table 6.12].

total time, it performed best; cf. [82].

Realistic Patient-Specific Artery

To test the preconditioners for a range of inflow velocities, we adjust the time step size accordingly to obtain a fairly constant CFL number; see Table 9.4. In Figure 10.27 (left) we can observe the response of the preconditioning strategies to the maximum Reynolds number defined in (9.2). The PCD and monolithic preconditioners deliver robust results. If we take a closer look at the average number of iterations per Newton step for the different time steps, the monolithic approach is very robust to the velocity changes, since the iteration count over time is almost constant in Figure 10.27 (right). The graph of the PCD preconditioner is similar to the graph of the CFL number; that is, it shows that its performance deteriorates with a larger CFL number.

Finally, we show strong scaling results for the artery in Figure 10.28; the mesh has 4.1 million degrees of freedom. Between 52 and 416 processor cores are used, of which 2 to 8 are used to solve the coarse problem; see Table 10.22. The unstructured mesh partition is constructed with METIS. We use an initial velocity of 40 cm/s but increase the time step with respect to Table 9.4 to $\Delta t = 0.002\,\mathrm{s}$ to reach results for 52 processor cores within a reasonable time frame. We test the monolithic preconditioner with GDSW⋆–RGDSW and the PCD block preconditioner with RGDSW–RGDSW. The results show strong scaling of both the PCD and the monolithic preconditioner. However, the monolithic preconditioner is more robust and requires significantly fewer iterations to converge, which results in a substantially lower computing time.

Based on the results in this work and the considered inflow–outflow problems, choosing the

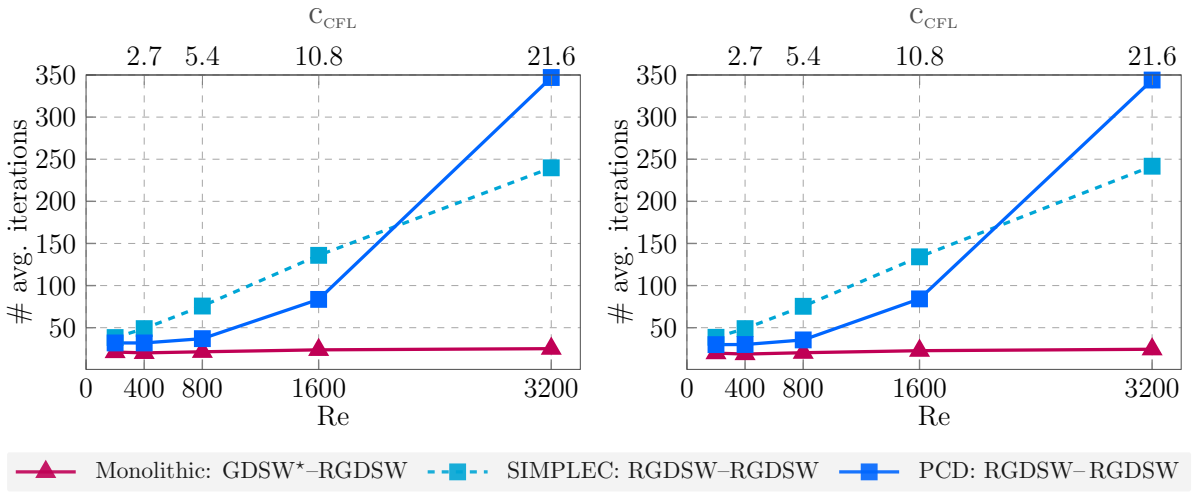


Figure 10.25: Comparison of block and monolithic preconditioners for increasing Reynolds and CFL numbers. Use of GDSW⋆ and RGDSW coarse spaces for the velocity (first component) and pressure (second component). Transient BFS problem with P2–P1 discretization and $H/h = 9$. Increasing Reynolds number due to increasing velocity. No pressure projection is used. Exclude off-diagonal blocks in ϕ for monolithic preconditioner. Constant time step size $\Delta t = 0.05\,\mathrm{s}$, simulation until $t = 10\,\mathrm{s}$. CFL condition increases with increasing velocity. Boundary condition strategy of PCD is (BC–3). Average GMRES iteration count per Newton step per time step. Computations of all methods with reuse of coarse basis (CB) (**Left**) and without reuse (**Right**); see Section 8.5.2. Note that the two plots look identical, since the reuse of the coarse basis (CB) has a very minor effect on the iteration count in this test setting. Taken from [82, Table 6.13].

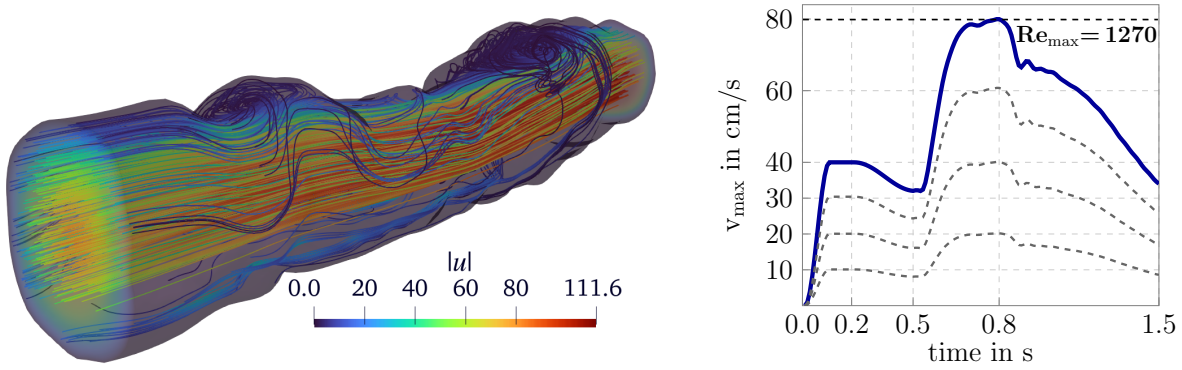


Figure 10.26: Realistic patient-specific artery for fluid flow problem Section 9.4. Geometry taken from [7]. **Left:** Magnitude of solution $\mathbf{u}_h$ at $t = 0.8\,\mathrm{s}$ with initial maximum inflow velocity $v_{max} = 40\,\mathrm{cm/s}$. **Right:** Inflow profile over time of the realistic artery problem. Ramp phase until $t = 0.1\,\mathrm{s}$ and subsequent initial maximum inflow velocity $v_{max} = 40\,\mathrm{cm/s}$. Maximum Reynolds number calculated based on maximum velocity and on approximate maximum diameter of artery; see (9.2). The heartbeat and the corresponding flow rate were constructed in [8] based on inflow pressure data from [134]. Taken from [82, Figure 6.3].

monolithic two-level overlapping Schwarz preconditioner with the GDSW⋆–RGDSW coarse space combination provides a consistently robust performance with a low compute time; cf. [82]. The PCD block preconditioner can achieve similarly reliable results, provided the boundary conditions

Table 10.21: Transient BFS problem with P2–P1 discretization and $H/h = 9$. $\mathrm{Re} = 3\,200$ with $\nu = 6.25 \times 10^{-4}\,\mathrm{cm}^2/\mathrm{s}$ and inflow velocity $v_{\max} = 1\,\mathrm{cm/s}$. Average number of Newton iterations per time step required to reach convergence is 3. Due to the increasing number of subdomains, the mesh resolution increases and in turn also the CFL number c_{CFL}. No pressure projection is used. 200 time steps with $\Delta t = 0.05\,\mathrm{s}$ and final time $T = 10\,\mathrm{s}$. PCD boundary condition strategy is (BC–3). SIMPLEC and PCD use the RGDSW–RGDSW coarse space combination. Exclude off-diagonal blocks in ϕ in monolithic preconditioner. Detailed results for 243 cores and varying Reynolds numbers are given in Table A.5. The results for 4 608 cores and SIMPLEC were not computed due to the high computational cost. In the case of 4 608 processor cores – to achieve the best results – 8 cores are used to solve the coarse problems for PCD, 12 for the monolithic preconditioner and RGDSW–RGDSW, and 16 for the monolithic preconditioner and GDSW⋆–RGDSW. Taken from [82, Table 6.8].

			Block preconditioner		Monolithic preconditioner	
# cores	c_{CFL}		SIMPLEC	PCD	RGDSW–RGDSW	GDSW⋆–RGDSW
243	1.4	# avg. iter.	52.7	31.7	35.0	21.7
		Setup time	595 s	580 s	785 s	787 s
		Solve time	2 901 s	1 761 s	2 008 s	1 268 s
		Total	3 496 s	2 341 s	2 793 s	**2 055 s**
1 125	2.3	# avg. iter.	96.8	41.5	38.4	20.5
		Setup time	709 s	684 s	904 s	935 s
		Solve time	6 882 s	2 924 s	2 788 s	1 512 s
		Total	7 591 s	3 608 s	3 692 s	**2 447 s**
4 608	3.6	# avg. iter.	—	43.7	43.0	19.0
		Setup time	—	745 s	994 s	1 179 s
		Solve time	—	3 567 s	3 642 s	1 664 s
		Total	—	4 312 s	4 636 s	**2 843 s**

Table 10.22: Number of processor cores assigned to the coarse problem for the results in Figure 10.28. Number of coarse functions in parentheses. Coarse spaces: The monolithic preconditioner uses GDSW⋆–RGDSW and the PCD block preconditioner RGDSW–RGDSW.

# cores	52	104	208	416
PCD	2 (404)	3 (1 084)	4 (2 771)	8 (6 057)
Monolithic	2 (1 013)	4 (2 518)	6 (5 912)	12 (12 975)

of the preconditioner are chosen appropriately and the CFL number is not too high; cf., e.g., Figure 10.27.

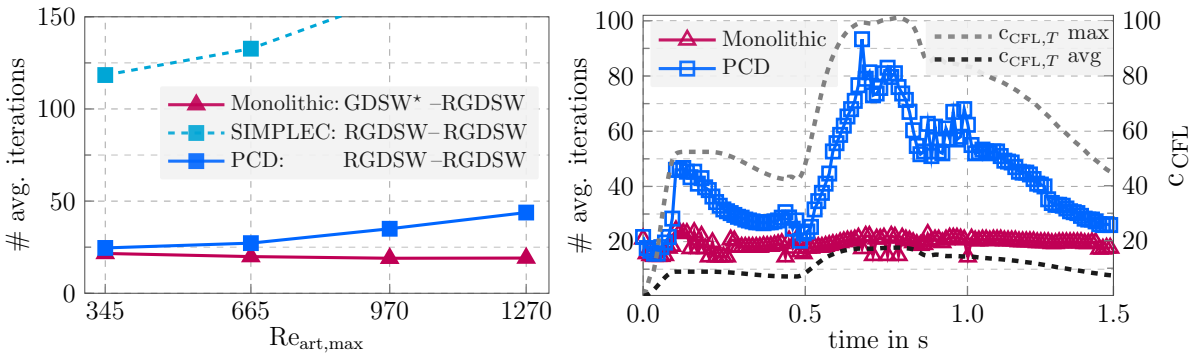


Figure 10.27: Comparison of block and monolithic preconditioners for the transient realistic artery problem (Figure 10.26) by varying the Reynolds and CFL numbers. Transient problem with P2–P1 discretization and 4.1 million degrees of freedom. Average number of Newton iterations per time step required to reach convergence is 3. Computations on 416 processor cores. No pressure projection is used. Use of GDSW⋆ and RGDSW coarse spaces for the velocity (first component) and pressure (second component). Excluding off-diagonal blocks of ϕ in monolithic preconditioner. Boundary condition strategy of PCD is (BC–3). **Left:** Increasing Reynolds number due to increasing initial velocity $v_{\max}$ from 10 to 40 cm/s; see Table 9.4. **Right:** Configuration with $\mathrm{Re}_{\mathrm{art,max}} = 1\,270$; see Table 9.4. Average GMRES iteration count per Newton step for each time step. Value is plotted every tenth time step. The maximum elementwise CFL number $c_{\mathrm{CFL}}(t)$ is shown (light gray); its maximum over all time steps is $c_{\mathrm{CFL}} = 100$. The average of elementwise CFL numbers $c_{\mathrm{CFL,avg}}(t)$ is depicted as well (dark gray); its average over time is 11. Due to the high average GMRES iteration count, the SIMPLEC preconditioner is excluded from the figure. Adapted from [82, Figure 6.14].

10.6. Results for Fluid–Structure Interaction Problems

To solve fluid–structure interaction (FSI) problems, monolithic and segregated approaches are both commonly used, where segregation means that fluid and structure problems are solved separately; see [29, 6] for a summary of the different methods and the references therein. For the monolithic approach, the FaCSI block preconditioner [36] can be used; see Section 6.2. The inverses arising in the FaCSI factorization are approximated by two-level overlapping Schwarz preconditioners. The application of the FaCSI preconditioner is shown in Algorithm 1.

In the previous Sections 10.3 and 10.5, two-level overlapping Schwarz preconditioners were investigated separately for the fluid and solid subproblems in order to identify efficient and robust solver and preconditioner configurations. Their performance was analyzed with respect to weak and strong scalability, as well as their robustness under increasingly realistic hemodynamic conditions, i.e., higher Reynolds numbers for the fluid problem and larger pressure loads for the solid problem.

Building on these results, the optimized solver and preconditioner configurations are now applied to the fully coupled fluid–structure interaction problem. Therefore, the geometry of a realistic artery is used; the test problem is described in Section 9.4. The goal is to assess whether the scalability and robustness we observed for the separate subproblems are retained in the FSI simulations. Consequently, the primary focus is on the performance and robustness of the proposed solver and preconditioner configurations rather than on the computation of highly accurate finite element solutions. In particular, the robustness tests consider parameter

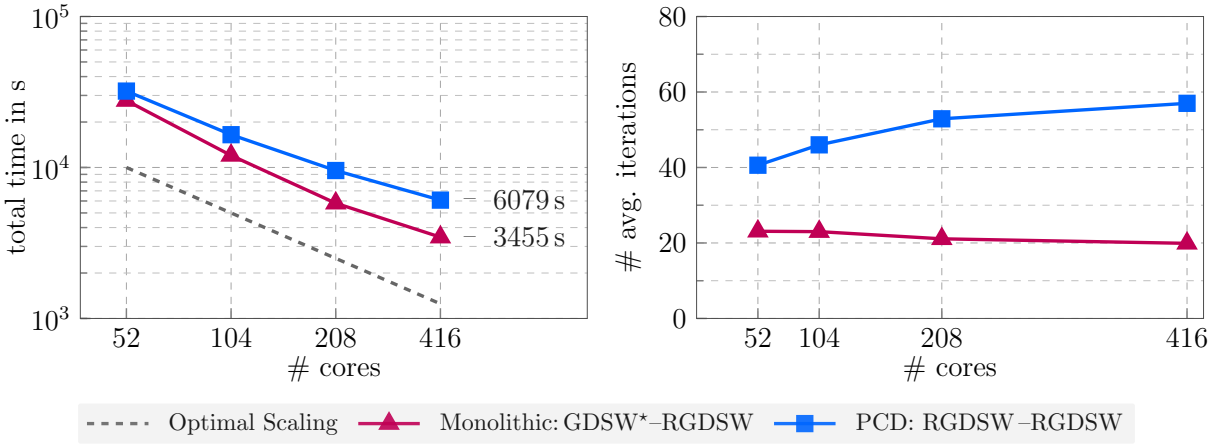


Figure 10.28: Use of GDSW⋆ and RGDSW coarse spaces for the velocity (first component) and pressure (second component). Transient problem with P2–P1 discretization and 4.1 million degrees of freedom. Average number of Newton iterations per time step required to reach convergence is 3. Computations on 52 to 416 processor cores; see Table 10.22 for the number of cores used to solve the coarse problem. Time step size $\Delta t = 0.002\,\mathrm{s}$, initial maximum velocity of $40\,\mathrm{cm/s}$, resulting in $\mathrm{Re}_{\mathrm{art,max}} = 1\,270$. The maximum CFL number is $c_{\mathrm{CFL}} = 160$. For the average elementwise CFL number, we have $c_{\mathrm{CFL,avg}}(t) = 28$. The average of $c_{\mathrm{CFL,avg}}(t)$ over all time steps is 18. No pressure projection is used. Excluding off-diagonal blocks in ϕ. Boundary condition strategy of PCD is (BC–3). **Left**: Total time consisting of setup and solve time. **Right:** Comparison of average GMRES iteration count per Newton step per time step for different numbers of processor cores and different preconditioning techniques. Adapted from [82, Figure 6.15].

settings that are intentionally beyond the physically relevant range, such as elevated Reynolds numbers. While such cases would require a significantly finer spatial and temporal discretization to obtain an accurate finite element solution, they provide a challenging benchmark for assessing the robustness of the solver and preconditioner.

In Section 10.6.1, the fluid subproblem within the FSI framework is investigated. In particular, the monolithic preconditioner is compared to the SIMPLE block preconditioner. Similar to the tests described in Figure 10.26, different inflow rates are considered; see Figure 10.30 (left). The resulting velocity and displacement fields are shown in Figure 10.29. Note that in FSI simulations the characteristic length D_{max} in $\mathrm{Re}_{\mathrm{art,max}}$ (9.2) changes due to the displacement of the fluid domain, which in turn leads to a higher Reynolds number. Therefore in (9.3) an alternative Reynolds number for the considered FSI simulation was defined. The resulting Reynolds numbers $\mathrm{Re}_{\mathrm{art,FSI}}(0.56)$ for the four test settings, consisting of different material models and parameters, in Figure 10.33 are shown in Table 10.23.

In Section 10.6.2 the solid subproblem in FSI is further investigated. Two different models are considered: a neo-Hookean material model and the Ψ_A model, which is anisotropic and quasi-incompressible with two fiber families. Both models were previously analyzed in Section 10.3. Different reference pressure values p_{ref} in the absorbing outlet boundary condition (9.4) are tested; see Figure 10.30 (right). The resulting deformations and displacement values at $t = 0.56\,s$ are shown in Figure 10.34.

Finally, the investigations of both the fluid and solid subproblems are concluded by strong

Table 10.23: Reynolds number examples for the FSI test cases using *NH3* and *Set 1-3*; see Tables 9.7 to 9.9. Values computed at $t = 0.56\,\mathrm{s}$. The Reynolds number is defined in (9.3) with $\nu = 0.0291\,\mathrm{cm}^2/\mathrm{s}$.

Parameter sets	$\lvert u_{\max}\rvert$ in cm/s	A_{inlet} in cm^2	$\bar{D}_{\text{inlet}}$ in cm	$\mathrm{Re}_{\text{art,FSI}}(0.56)$
NH3	179.2	0.087	0.33	2 032
Set 1	71.3	0.204	0.51	1 249
Set 2	142.3	0.092	0.34	1 674
Set 3	136.6	0.120	0.39	1 835

scaling studies covering half a heartbeat; see Figure 10.32. The full FSI system has approximately 2.2 million degrees of freedom.

The following settings are used for both tests. For both subproblems, it is feasible to reuse the full preconditioner after the first Newton step in each time step, that is, $k = 1$. The adaptive forcing term is again considered, with $\eta_{\text{init}} = \eta_{\max} = 10^{-3}$ as it was used for the fluid flow simulations. Newton's method terminates when either the relative residual $\|\mathcal{R}(X^{k+1})\|_2/\|\mathcal{R}(X^0)\|_2$ reaches a tolerance of 10^{-8} or the Newton update $\|X^{k+1} - X^k\|_2$ reaches a tolerance of 10^{-6}, respectively.

The preconditioner $\mathcal{B}_{\mathcal{G}}^{-1}$ in (6.24) used for the geometry problem (5.1) is also a two-level overlapping Schwarz preconditioner with RGDSW coarse space with Option 1 and with (CB) and (CM) reuse. We will not analyze the preconditioner settings for the geometry problem any further, but, for instance, compared to the results in Table 10.27 for *Set 1* on 208 cores, the average iteration count did not improve by omitting (CB) reuse, but the setup time increased.

10.6.1. Fluid Subproblem

Parts of this chapter are based on [107]. We compare the SIMPLE block preconditioner with a monolithic preconditioner for the fluid subproblem in FaCSI (Section 6.2) and analyze whether the findings from Figure 10.27 and Figure 10.28 can be applied to the fluid subproblem in the context of an FSI problem. There, the monolithic preconditioner proved more robust with respect to increasing Reynolds number, compared to the block variants. Let us note that, in this section, the denominations "monolithic" and "block" preconditioners refer only to the fluid subproblem and not to the complete coupled FSI problem. Besides reusing the preconditioner after the first Newton iteration of each time step, the coarse basis reuse (CB) is additionally used; see Section 8.5.2. For preconditioning the solid problem, in this section, the RGDSW coarse space is used with Option 2.2, and rotational modes in coarse space.

The coarse space combination GDSW* and RGDSW is used for the velocity and pressure components in the monolithic preconditioner of the fluid problem. In case of the SIMPLEC variant of the SIMPLE block preconditioner, the RGDSW coarse space is used for the fluid and Schur complement approximations.

First, we compare results for different flow rates, where the flow rates vary between 2 and $6\,\mathrm{cm}^3/\mathrm{s}$. The test contains a ramp and a steady phase, each 0.1 s long; cf. *Test (a)* in Figure 10.30 (left). Secondly, for a strong scaling test, we also include the flow profile of half a heartbeat until

Table 10.24: Parameters for the fluid and solid. The reference fluid pressure p_{ref} is kept constant after an initial ramp phase. For the flow rate at the end of the initial ramp phase, Q, see Figure 10.30. Material parameter set for *NH3* and *Set 1-3* according to Tables 9.7 to 9.9.

ν in cm^2/s	p_{ref} in kPa	ρ_f in kg/cm^3	ρ_s in kg/cm^3
0.0291	10.66	$1.03\,10^{-3}$	$1.0\,10^{-3}$

Table 10.25: Detailed results for the strong scaling plot shown in Figure 10.32 (left). Setup time consists of construction of preconditioner and solve time consists of linear and nonlinear solve. Simulation of 560 time steps with $\Delta t = 0.001\,\text{s}$ and an initial flow rate $3\,\text{cm}^3/\text{s}$. 2.29 $\cdot 10^6$ dofs. Use of rotational modes in coarse space of the solid preconditioner. Average Newton iteration count over all time steps in parentheses.

Prec.	Monolithic			SIMPLEC		
N	# avg. iter.	setup	solve	# avg. iter.	setup	solve
52	21.4 (8.5)	3 047 s	9 725 s	59.9 (8.1)	2 816 s	26 220 s
104	24.3 (8.6)	1 554 s	6 505 s	75.6 (8.2)	1 380 s	19 750 s
208	25.6 (8.8)	965 s	4 546 s	84.5 (8.2)	819 s	14 200 s
416	29.6 (9.1)	890 s	4 411 s	109.1 (8.2)	647 s	14 470 s

0.56 s, which is at the peak of the heartbeat flow; this is denoted as *Test (b)* in Figure 10.30 (left). The solid part of the FSI simulation is modeled with the *NH3* model, which was defined in Table 9.5, i.e., we consider a neo-Hookean material model with Young's modulus 380 kPa; cf. *NH3* in Figure 10.34 for the resulting pressure, and inflow and outflow areas. In Section 10.6.2, additionally, the Ψ_A model is considered; see Section 3.1.3.

The set of parameters for this FSI simulation, using a neo-Hookean material model for the solid component, is shown in Table 10.24.

In Figure 10.31 (left), results for the comparison of block and monolithic preconditioners are shown. The figure shows the average iteration count per Newton step per time step to solve the full FSI system for increasing inflow rates Q, which also increases the Reynolds number. Based on the definition of the Reynolds number for the realistic artery defined in (9.2), the $\text{Re}_{\text{art,max}}$ for inflow rate $6\,\text{cm}^3/s$ is approximately 2 040. The iteration count for the monolithic case is lower compared to the SIMPLEC variant and appears more robust with an increasing inflow rate. Overall, both preconditioning approaches show robust behavior, whilst the monolithic approach has a lower overall average iteration count below 31 compared to SIMPLEC with above 85. In Table 10.25 and Figure 10.32 (left) the strong scaling results are illustrated. Since the monolithic preconditioner for the fluid component performs better than its block counterpart, the overall iteration count to solve the FSI problem is lower. The SIMPLEC block preconditioner offers lower setup times compared to the monolithic approach, but results in significantly longer times for the iterative solvers; see Table 10.25, cf. [107].

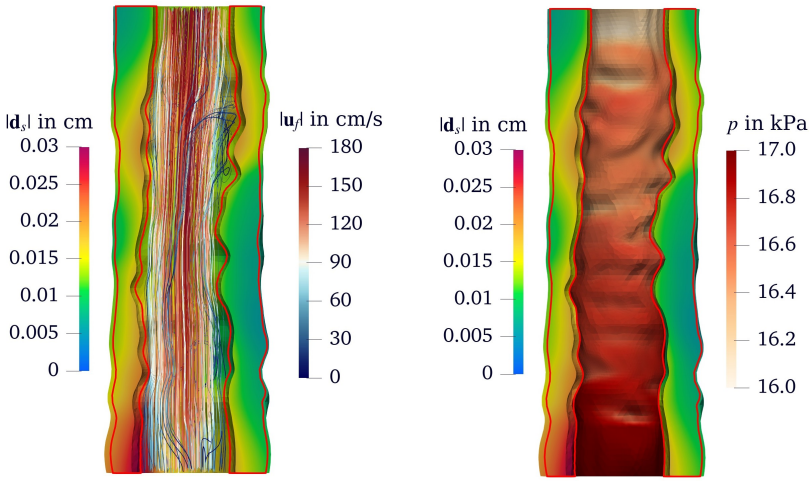


Figure 10.29: Results of computation of *Test (b)* at $t = 0.56\,\mathrm{s}$ and time step size $\Delta t = 0.001\,\mathrm{s}$ from Figure 10.30 (left). The red outline illustrates the undeformed artery. **Left:** Displacement $|\mathbf{d}_s|$ of solid domain in cm and velocity of the fluid $|\mathbf{u}_f|$ in cm/s. **Right:** Displacement $|\mathbf{d}_s|$ of solid domain in cm and pressure of the fluid p in kPa. Based on the definition of the Reynolds number for the realistic artery defined in (9.2), the $\mathrm{Re}_{\mathrm{art,FSI}}(0.56)$ for this test case is approximately 2 032.

10.6.2. Solid Subproblem

After investigating the fluid preconditioner within the FSI setting, we continue with the preconditioning of the structural subproblem. Based on the results of the previous section, the monolithic preconditioner with the GDSW*–RGDSW coarse space is used for the fluid subproblem.

While a neo-Hookean material model was used in the previous section, this section additionally includes the Ψ_A model; see Section 3.1.3. The model is either applied homogeneously for the different material components — the adventitia, media, and plaque — or heterogeneously, where each component has different material parameters. For preconditioning the solid subproblem, the RGDSW coarse space with Option 2.2 is employed. Neither the coarse basis (CB) nor the coarse matrix (CM) is reused. In addition, the influence of including rotational modes in the coarse space is examined.

To assess the robustness of the solid preconditioner, the reference pressure in the absorbing boundary condition is varied, thereby inducing different levels of arterial wall deformation; see Table 10.26. For the first tests, the material model is considered to be homogeneous. To this end, the same material parameters are chosen for all arterial wall components, that is, the plaque, media, and adventitia. The material parameters are chosen according to the *NH3* material for the neo-Hookean material, and for Ψ_A based on parameters *Set 1* and *Set 2*; see Tables 9.5 and 9.8. In Figure 10.33 the inflow and outflow area, and the outflow pressure of the fluid are measured for different material models for a reference pressure value of 10.66 kPa. The corresponding results for the displacement and deformation of the arterial wall at $t = 0.56\,s$ are shown in Figure 10.34.

For parameter *Set 2*, similar to the comparison in Figure 10.2, the displacement is generally higher than for the neo-Hookean model *NH3*, but the impact of the heartbeat is lower. For parameter *Set 1*, the tissue is even softer leading to greater deformation in general, and larger inflow and outflow areas compared to *Set 2*; see also the short discussion of the corresponding

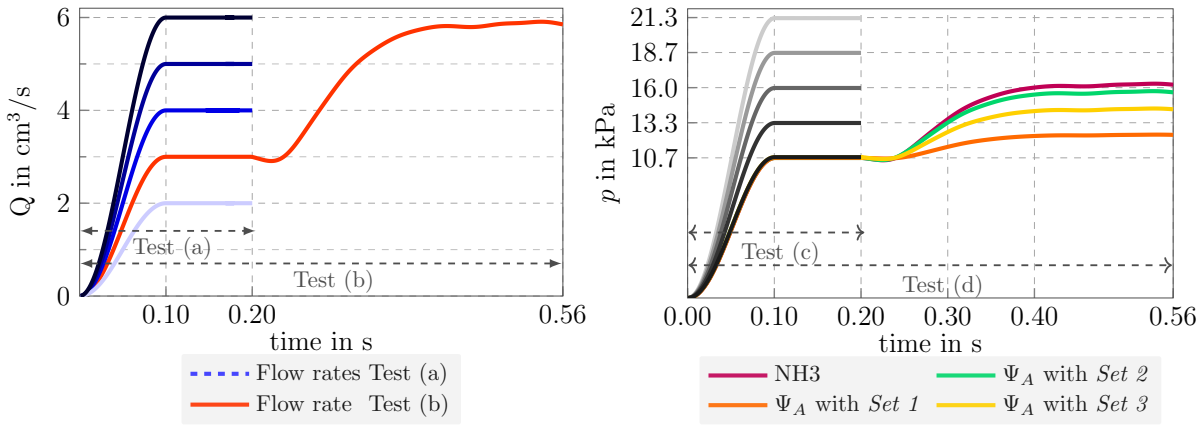


Figure 10.30: Testing the robustness of the solver and preconditioner for the fluid (left) and solid (right) by different test cases. **Left:** Prescribed inflow rate of fluid domain. We consider a range from 2 to $6\,\mathrm{cm}^3/\mathrm{s}$ for Test (a). For the steady phase up to 0.2 s the pressure is not influenced by the flow rate due to the absorbing boundary condition. Time step size $\Delta t = 0.001\,\mathrm{s}$. **Right:** Prescribed reference pressure p_{ref} in the absorbing boundary conditions. The steady phase up to 0.2 s is denoted as *Test (c)*. Time step size $\Delta t = 0.0005\,\mathrm{s}$. For the simulation of half a heartbeat until $t = 0.56\,\mathrm{s}$, the reference pressure value is set to $p_{\mathrm{ref}} = 10.66\,\mathrm{kPa}$ (80 mmHg) and the resulting pressure values are denoted as *Test (d)*. Material parameter set for *NH3* and *Set 1-3* according to Tables 9.7 to 9.9. The heartbeat and the corresponding flow rate were constructed in [8] based on inflow pressure data from [134].

parameter sets in Table 9.8.

Table 10.26 shows the results for different reference pressure values and material models. For the considered configuration, in particular for the time step size $\Delta t = 0.0005\,\mathrm{s}$, the Ψ_A model with parameter *Set 1* fails to converge for pressure loads above 10.67 kPa. More precisely, during the ramping phase, the nonlinear solver does not converge. This may be attributed to the softer material, as the inflow and outflow areas are, for example, approximately twice as large as those for parameter *Set 2*; see Table 10.26. However, convergence can be recovered by reducing the time step size, for example to $\Delta t = 0.00025\,\mathrm{s}$. The results in Figure 10.31 (right) and Table 10.26 illustrate only a marginal increase in average iteration count for increasing reference pressure values p_{ref} for the *NH3* model, and the Ψ_A model with parameter *Set 2*.

For the next step, the arterial wall is treated as heterogeneous material, since it contains distinct regions of adventitia, media, and plaque. Therefore, the material parameters defined in Table 9.9 are considered; see Section 9.6 for a corresponding discussion on the parameters.

We consider a reference pressure of $p_{\mathrm{ref}} = 10.66\,\mathrm{kPa}$. Since the arterial wall contains stiffer components for this test setup, the displacement is expected to be lower than for the homogeneous setup in Figure 10.34 (left), where the model was homogeneously modeled as media; see *Set 1* in Table 9.8. The inflow and outflow area results are also included in Figure 10.33. The corresponding displacement of the arterial wall is illustrated in Figure 10.35 (left), which is lower compared to before; see Figure 10.33 for the inflow and outflow area in comparison to the other models. This is caused by the modeling of the plaque. The von Mises stresses are shown in Figure 10.35 (center). Especially in the plaque regions higher stresses are measured. Note that this is just an indication of the overall stress distribution. Values can also vary due to

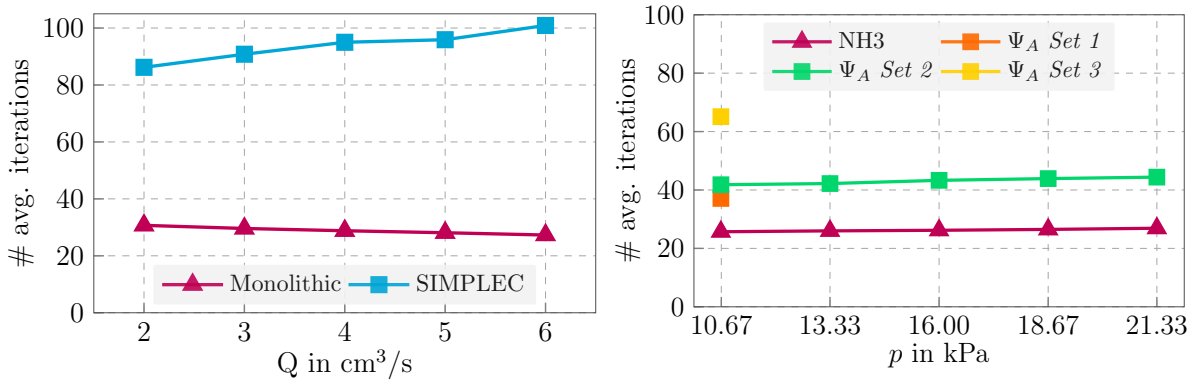


Figure 10.31: Results of robustness tests in FSI for the fluid and the solid subproblems. **Left:** Comparison of monolithic and block preconditioners. Resulting average linear iteration count per Newton step per time step for varying inflow flow rate using Test (a) in Figure 10.30 (left) for 200 time steps. On average 6 Newton iterations per time step are needed for $Q = 2\,\mathrm{cm}^3/\mathrm{s}$ and 8 for $Q = 6\,\mathrm{cm}^3/\mathrm{s}$. Note that the average Newton iteration count varies for monolithic and block approaches for $Q = 2, 3, 4\,\mathrm{cm}^3/\mathrm{s}$ by approximately 0.5. For $Q = 5, 6\,\mathrm{cm}^3/\mathrm{s}$ it is the same average Newton iteration value. Time step size $\Delta t = 0.001\,\mathrm{s}$. **Right:** Comparison of different arterial wall models. Resulting average iteration count per Newton step per time step for varying reference pressure values using *Test (c)* in Figure 10.30 (right) for 400 time steps. Computations on 208 cores. Time step size $\Delta t = 0.0005\,\mathrm{s}$ and inflow rate $Q = 3\,\mathrm{cm}^3/\mathrm{s}$. For the considered configuration, in particular for the time step size $\Delta t = 0.0005\,\mathrm{s}$, the Ψ_A model with parameter *Set 1* and *Set 3* fails to converge for pressure loads above 10.67 kPa. More precisely, during the ramping phase, the nonlinear solver does not converge. However, convergence could be recovered by reducing the time step size, for example to $\Delta t = 0.00025\,\mathrm{s}$.

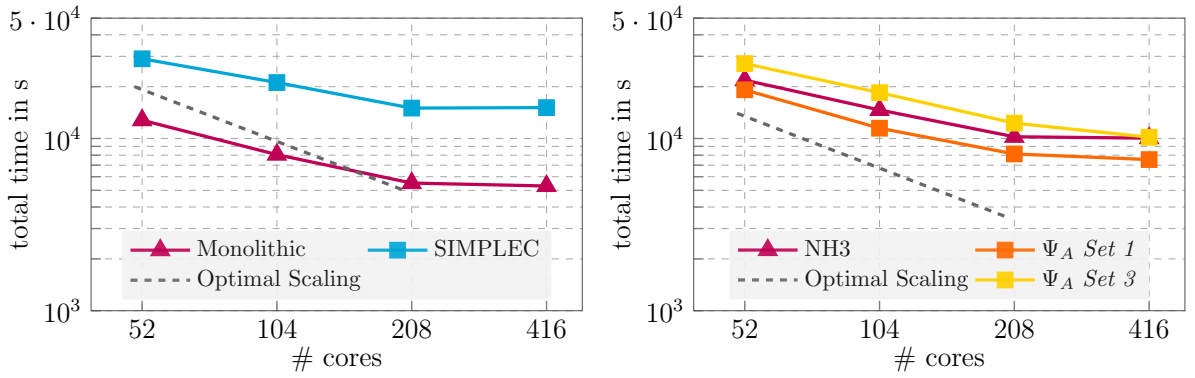


Figure 10.32: FSI strong scaling results with individual focus on the fluid and solid subproblems. The full FSI system has 2 230 418 degrees of freedom. **Left:** Comparison of monolithic and block preconditioners. Strong scaling using Test (b) in Figure 10.30 for 560 time steps with $\Delta t = 0.001\,\mathrm{s}$. Total time consists of setup and solve time. On average 8 Newton iterations per time step are needed. Details are shown in Table 10.25. **Right:** Comparison of different material models. Strong scaling using Test (d) in Figure 10.30 for 1 120 time steps with $\Delta t = 0.0005\,\mathrm{s}$. The full FSI system has 2 230 418 degrees of freedom. Total time consists of setup and solve time. Detailed timings are shown in Table 10.27.

different mesh resolutions in the geometry, and realistic stress distributions are not considered here. Note that, however, the `FEDDLib`–`AceGen` interface contains functionality to include growth

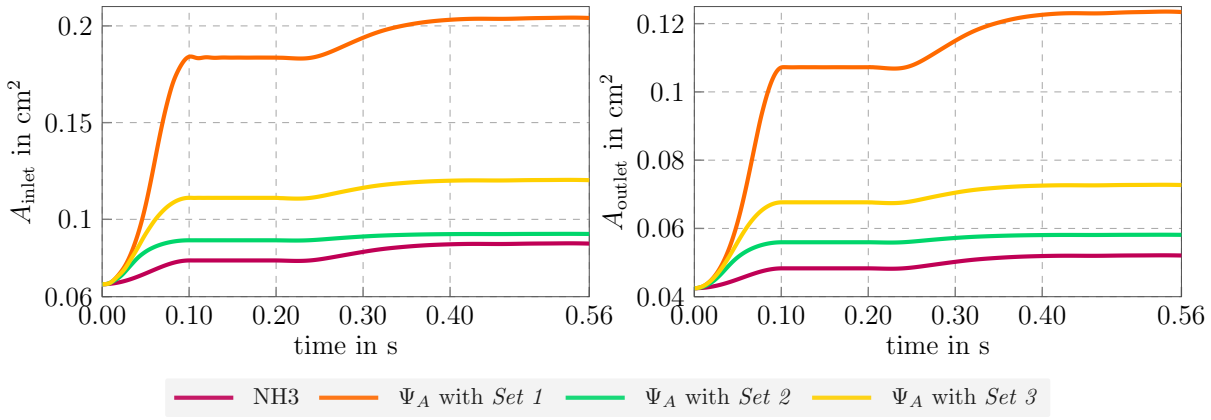


Figure 10.33: Results for FSI problem using realistic patient-specific artery from Section 9.4 with reference fluid pressure $p_{\text{ref}} = 10.66\,kPa$. The *NH3* model and Ψ_A model are compared with different parameter sets. *Set 1* and *Set 2* are shown in Table 9.8; *Set 3* is defined in Table 9.9. **Left:** Measured inflow area. **Right:** Measured outflow area.

and remodeling in the artery to account for, e.g., residual stresses; see [149, Section 2.1.3].

In Table 10.27 a strong scaling comparison is shown for the *NH3* model, and the Ψ_A model with homogeneous and heterogeneous material, i.e., parameter *Set 1* and *Set 3*. This is also included in Figure 10.32 (right). For the three model configurations we observe in general good strong scaling behavior. Note that here, the rotational modes are included in the coarse space of the solid preconditioner. For parameter *Set 1* and *Set 3* including rotations improved the overall results; see also Table 10.26. For example in the results shown in Table 10.27 for *Set 3*, if rotations are omitted in case of 208 cores, the average linear iteration count per Newton step increased to 55.2, with a total time (consisting of setup and solve) of 13 534 s. Even though the setup cost reduced by 200 s, the solve time increased — the subdomain solve — due to the increasing number of linear iterations.

10.6.3. Discussion on Strong Scaling

In Figure 10.32 strong scaling results for the comparison of fluid preconditioners (left) and comparison of different material models are shown. The detailed timings and average iteration counts are shown in Tables 10.25 and 10.27. For the monolithic fluid preconditioner, and the different material models, we see good scaling with respect to the average iteration count. Note that the results for the *NH3* model in Table 10.27 correspond to the results for the monolithic preconditioner in Table 10.25 just using a smaller time step, i.e., $\Delta t = 0.0005\,\text{s}$ instead of $\Delta t = 0.001\,\text{s}$.

In this section, we want to detail some aspects, as to why we lose scalability for core numbers higher than 208. First of all, the FSI problem contains 2.2 million degrees of freedom. For 416 cores, the limit of the strong scalability is reached. There, each core contains approx. 5 300 degrees of freedom, which can be considered as small.

There are several properties that influence the performance of the two-level additive Schwarz method. For example, if the size of local problems decreases, generally, the number of iterations

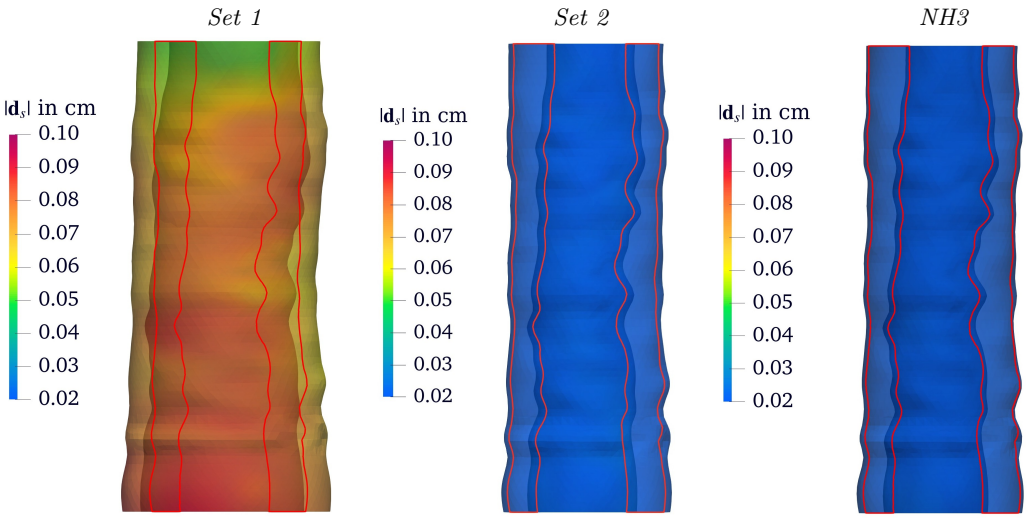


Figure 10.34: Results of deformation of FSI problem using realistic patient-specific artery from Section 9.4 with reference fluid pressure $p_{\text{ref}} = 10.66\,kPa$. Time step is $t = 0.56\,\text{s}$. The *NH3* model and Ψ_A model from Table 9.5 are considered. Material parameter set for *NH3* and *Set 1-3* according to Tables 9.7 and 9.8. In red, the outline of the undeformed artery is shown. Inflow on the bottom, outflow at the top. **Left** *Set 1*. **Center:** *Set 2*. **Right:** *NH3*.

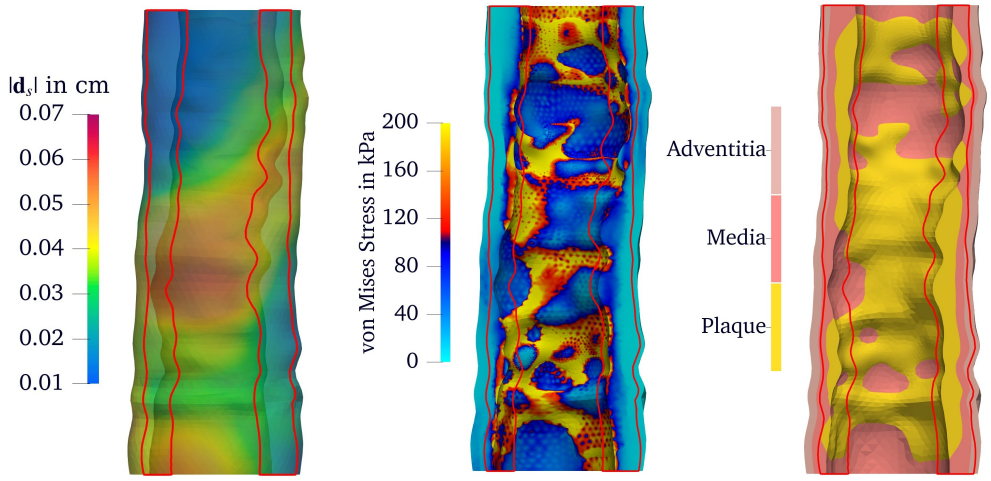


Figure 10.35: Results of displacement of FSI problem using realistic patient-specific artery from Section 9.4 with reference fluid pressure $p_{\text{ref}} = 10.66\,kPa$. Time step is $t = 0.56\,\text{s}$. The Ψ_A model from [9] with heterogeneously modeled components according to *Set 3*; see Table 9.9. 1 120 time steps. In red, the outline of the undeformed artery is shown. Inflow on the bottom, outflow at the top. **Left** Norm of displacement in cm and deformation of solid domain. **Center:** Von Mises stress in kPa. **Right:** Different components of arterial wall.

increases, which is also evident from the results in Table 10.25 concerning the SIMPLEC block preconditioner. Results regarding the monolithic preconditioner, in the presented studies, yield only a moderate increase in average iteration counts. In general, a larger number of iterations not only requires more applications of the system matrix in GMRES but also of the preconditioner.

Table 10.26: Resulting average iteration counts for FSI patient-specific artery with varying reference pressure values for different material models. Average iteration count per Newton step per time step, with values where rotational modes are used in the solid preconditioner in parentheses. Computations on 208 cores. Time step size $\Delta t = 0.0005\,\text{s}$. The reference fluid pressure p is kept constant after an initial ramp phase. Simulation up to $t = 0.2\,\text{s}$. Constant flow rate of $3\,\text{cm}^3/\text{s}$. Inflow and outflow areas are also stated after the ramp phase. The initial inflow and outflow areas are $0.066\,\text{cm}^2$ and $0.042\,\text{cm}^2$. Time step size $\Delta t = 0.0005\,\text{s}$. *Set 1* and *Set 2* are shown in Table 9.8; *Set 3* is defined in Table 9.9.

p_{ref}		10.67 kPa (80 mmHg)	13.33 kPa (100 mmHg)	16.00 kPa (120 mmHg)	18.67 kPa (140 mmHg)	21.33 kPa (160 mmHg)
Material model						
NH3	Inflow area	0.079	0.082	0.086	0.091	0.096
	Outflow area	0.048	0.050	0.052	0.054	0.056
	# avg. iter.	25.7 (22.5)	26.0 (22.7)	26.2 (22.9)	26.5 (23.1)	26.9 (23.3)
Ψ_A *Set 1*	Inflow area	0.184				
	Outflow area	0.107				
	# avg. iter.	37.1 (33.4)	—	—	—	—
Ψ_A *Set 2*	Inflow area	0.089	0.091	0.092	0.093	0.094
	Outflow area	0.056	0.057	0.058	0.059	0.060
	# avg. iter.	41.8 (40.0)	42.2 (38.3)	43.3 (38.9)	43.9 (38.8)	44.4 (39.5)
Ψ_A *Set 3*	Inflow area	0.111				
	Outflow area	0.068				
	# avg. iter.	65.1 (55.9)	—	—	—	—

Table 10.27: Strong scaling results for realistic patient-specific artery with different wall models. Setup time consists of construction of preconditioner and solve time consists of linear and nonlinear solve. Initial flow rate $3\,\text{cm}^3/\text{s}$. Time step size $\Delta t = 0.0005\,\text{s}$ leading to 1 120 time steps. Reference pressure 10.66 kPa. Rotational modes in the coarse space of the solid preconditioner.

	NH3			Ψ_A Set 1			Ψ_A Set 3		
N	# avg. iter.	setup	solve	# avg. iter.	setup	solve	# avg. iter.	setup	solve
52	19.1 (6.2)	6 041 s	12 850 s	24.5 (4.8)	6 084 s	13 090 s	38.2 (5.1)	6 042 s	21 190 s
104	20.6 (6.6)	3 073 s	8 700 s	26.8 (4.9)	3 068 s	8 411 s	45.4 (5.2)	3 100 s	15 370 s
208	22.1 (7.1)	1 911 s	6 579 s	28.9 (5.2)	1 909 s	6 225 s	47.8 (5.2)	1 914 s	10 370 s
416	25.9 (7.9)	1 745 s	6 758 s	33.1 (5.3)	1 743 s	5 802 s	48.0 (5.3)	1 761 s	8 413 s

Moreover, the complexity of the orthogonalization routine within GMRES increases quadratically with the number of iterations. Since the average iteration count only increases marginally, the loss

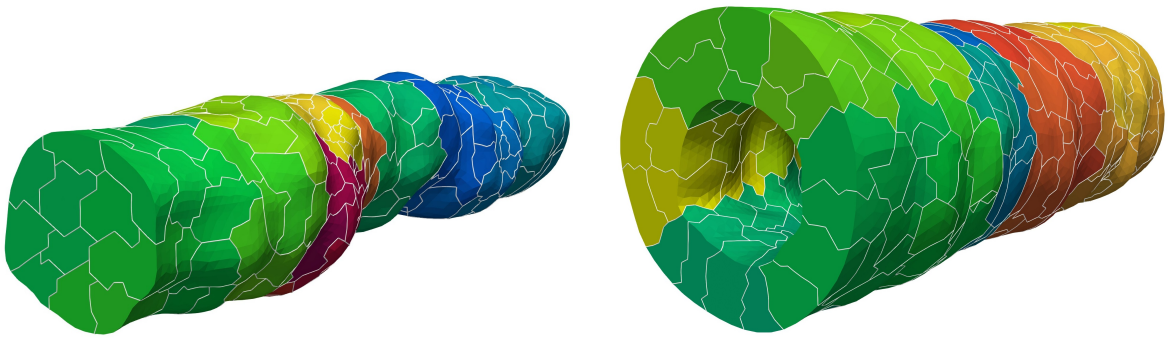


Figure 10.36: Visualization of the unstructured METIS partition of the fluid domain (left) and arterial wall (right) into 416 subdomains. Outflow area of the domain is shown.

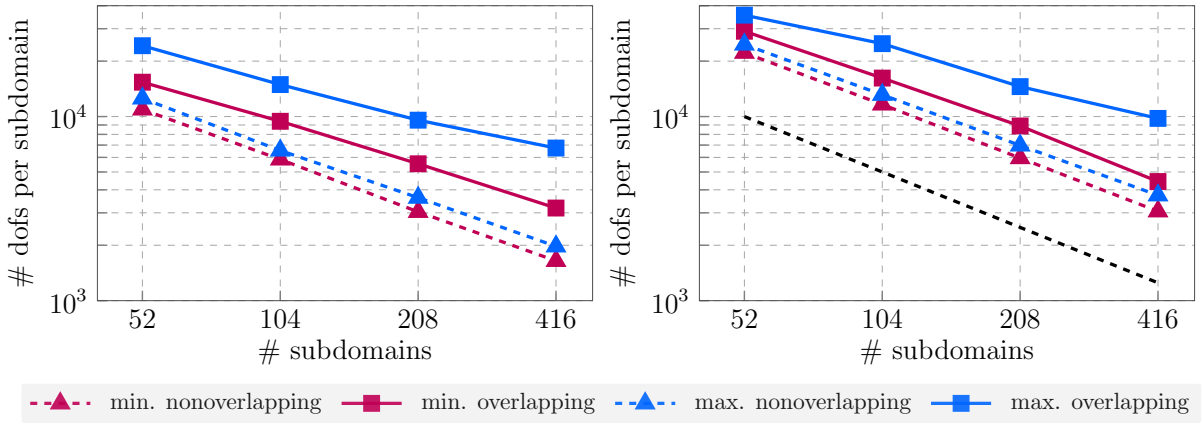


Figure 10.37: Degrees of freedom per subdomain versus number of subdomains. Minimum and maximum number of degrees of freedom per nonoverlapping and overlapping subdomain, respectively. Unstructured mesh partition with METIS. **Left**: Fluid domain. **Right**: Solid domain with 1 131 789 degrees of freedom. For 52 cores, the number of dofs in all overlapping subdomain problems is 1 670 280. For 416 cores, the number of dofs in all overlapping subdomain problems is 3 081 888.

of scaling is rooted elsewhere. In Figure 10.38, an analysis of the different components regarding setup and solve time is shown corresponding to Table 10.27 for *Set 3*. For the components of the setup time (Figure 10.38 (right)), we generally see good scaling behavior. The computation of the second level becomes more expensive for 416 cores, which is accounted for by the increasing coarse space dimension. For the components of the solve (Figure 10.38 (left)), we observe good scaling for the first level solve, i.e., the forward and backward solve component; see Section 8.8 for details on the direct solver MUMPS. Also the orthogonalization delivers moderate scaling results, since the average iteration count per Newton step only increases marginally. The solve of the second level, i.e., coarse problem, increases with increasing number of cores. Although the coarse problem is solved in parallel, the increase in coarse space dimension causes an increase in solve time. This component negatively influences the scaling of the solve component.

Additionally, we discuss how the mesh partition can influence the performance. For instance, the partition of the solid mesh into 208 subdomains is almost two-dimensional in the sense that

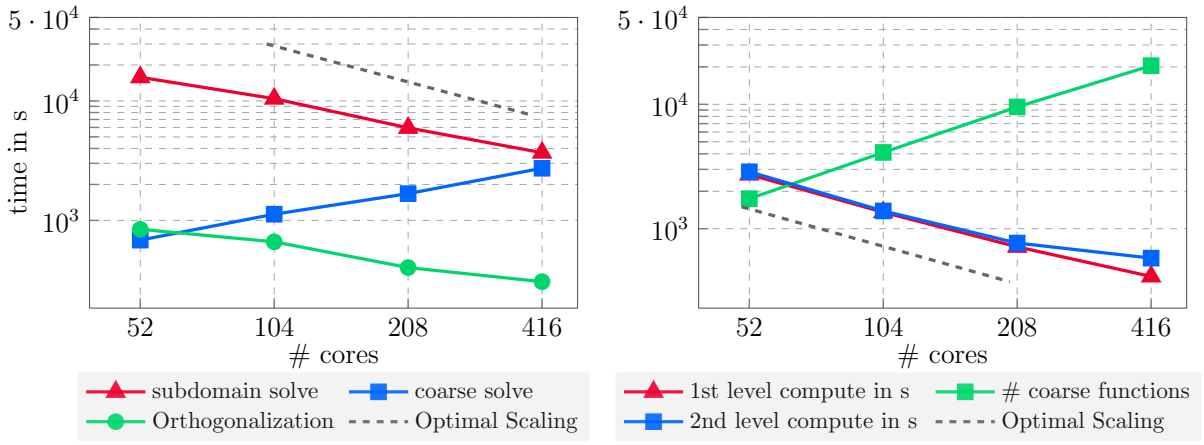


Figure 10.38: Detailed runtime analysis of results in Table 10.27 for *Set 3*. **Left**: Components of solve process, which mainly consist of the subdomain solve, the coarse solve, and orthogonalization in the GMRES iterative solver. The respective solves are the forward and backward solve performed by the direct solver; see Section 8.8. **Right**: Components of setup process, which is mainly the computation of the numeric factorization of the subdomain and coarse problem.

there is usually only a single subdomain in the radial direction. This effect is diminished with a partition into 416 subdomains (see Figure 10.36 (right)), where, for example, on the outflow part of the domain subdomains are partitioned in radial direction. The additional coupling can influence the performance, e.g., by increasing the coarse space dimension and requiring more communication, and weaken strong scalability. See for example Figure 10.38 (right), where the number of coarse functions is denoted for the increasing number of subdomains.

As we can see in Figure 10.37, the METIS-computed mesh partition for the nonoverlapping subdomains differs only slightly with respect to the minimum (red dashed line) and maximum (red dashed line) subdomain size. However, locally, we do not solve nonoverlapping but overlapping subdomain problems. Figure 10.37 shows that the minimum (blue solid line) and maximum (blue solid line) overlapping subdomain sizes diverge with an increasing number of subdomains. A reason may be the shape of the subdomains. Even if the nonoverlapping subdomains are identical in size, a different shape can significantly influence the size of the corresponding overlapping subdomain. For example, the ratio of nodes in the overlapping to the nodes in the nonoverlapping subdomain is large for a thin beam and small for a cube or sphere. In general, the first level solve, i.e., the solve regarding the overlapping problems, scales well with increasing number of cores, since the subproblems can all be solved in parallel. Although, in our case, the solve of the first level scales well, a loss in scalability can be caused by the loss in scaling of the overlapping subdomain size; see Figure 10.37 (right) where the maximum overlapping subdomain size tends to scale less with increasing number of subdomains. The overlapping subdomains with maximum number of degrees of freedom dominate the solve time of the first level, which is based on a load balancing problem.

10.7. Results for Pharmaco–Mechanical Fluid–Structure Interaction Problems

This section presents the numerical results for the pharmaco–mechanical fluid–structure interaction (FSCI) problem. To this end, the idealized artery introduced in Section 9.3 is considered to qualitatively validate the behavior observed for the deformation–diffusion problem in Figure 10.11 (right). In contrast to the deformation–diffusion setting, the internal pressure is not prescribed directly. Instead, it is determined by the prescribed inflow rate and the vessel deformation, through the absorbing outlet boundary condition; see (9.4) and the discussion in Section 9.6.4. The reference pressure is set to 18.67 kPa (140 mmHg). The prescribed inflow rate, together with the resulting internal pressure, outflow area, and outflow radius, is shown in Figure 10.39.

For this simplified test case, the time step size can be increased after the ramp phase. The ramp phase is discretized by 50 load steps, corresponding to a time step size of $\Delta t = 0.02\,\mathrm{s}$. Subsequently, for $t > 1\,\mathrm{s}$, the time step size is increased to $\Delta t = 0.25\,\mathrm{s}$ until the end of the simulation at $t = 2\,500\,\mathrm{s}$.

Overall, the FSCI formulation exhibits the same qualitative contractile behavior as observed for the deformation–diffusion problem. Furthermore, the inner radius measured after the ramp phase is nearly identical, amounting to 0.1607 cm and 0.1611 cm for the FSCI and deformation–diffusion simulations, respectively. Note that the observed pressure values — even for the reduced values due to the pharmacological effects — are comparable to a hypertension crisis, with maximum measurements above 180 mmHg (21.33 kPa). Results corresponding to a lower reference pressure value of 10.66 kPa (80 mmHg) are shown in the appendix in Figure A.5.

As the artery contracts, the outflow area decreases. Through the absorbing outlet boundary condition, this reduction in area leads to an increase in the internal pressure. Consequently, the pressure opposes the active contraction, resulting in a slightly larger vessel radius than in the deformation–diffusion simulation, where the internal pressure follows a fixed protocol and therefore remains unaffected by the vessel deformation; cf. Figure 10.11. Once the drug diffuses through the arterial wall, the active contraction is reduced, leading to a slower decrease in vessel radius and correspondingly a stabilized pressure value; see Figure 10.39 (bottom left).

The present example primarily serves as a proof of concept for the fully coupled FSCI framework; compare also the validation results for the simplified deformation–diffusion model presented in Section 10.2. The relation between flow rate, outflow area, and pressure imposed by the absorbing outlet boundary condition has not been calibrated to physiological measurements. Likewise, the material parameters were originally fitted for the deformation–diffusion problem in [149] and may require further calibration for fully coupled FSCI simulations. Nevertheless, the results demonstrate that the proposed framework is capable of capturing the interaction between drug-induced changes in smooth muscle activation, vessel deformation, and the resulting variation in blood pressure.

In Figure 10.40, the average iteration count per Newton step is illustrated for each time step. The results for the implicit and explicit coupling of the deformation–diffusion problem are almost identical. After the diffusion process is initiated, both results almost fully overlap. A similar behavior was observed in Figure A.2, where the average linear iteration count per time step was plotted for the standalone deformation–diffusion problem, with a similar test setup.

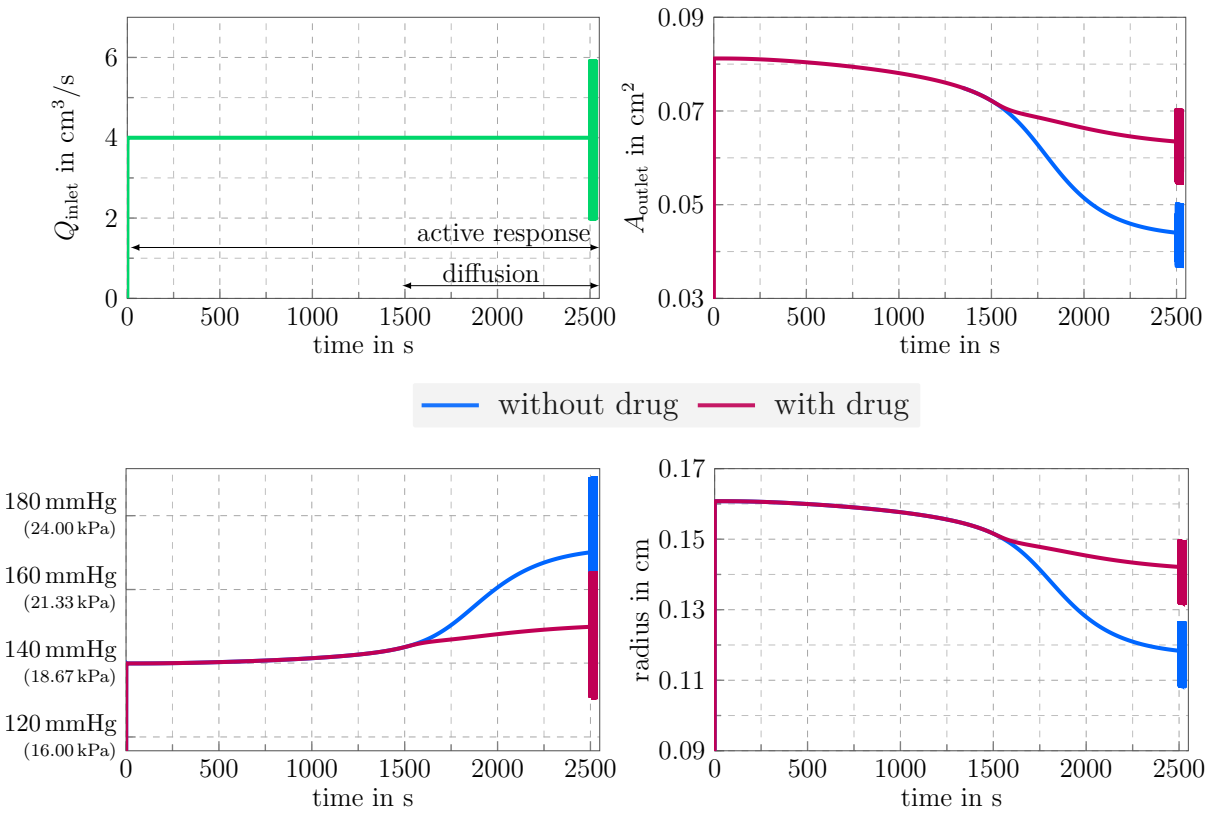


Figure 10.39: Results of full FSCI simulation using idealized arterial segment; cf. Section 9.3. Simulation spans 2 550 s with 50 heartbeats starting at 2 500 s. The solid part is modeled as media with parameters corresponding to *Set 1* in Table 9.8. **Top Left:** Prescribed inflow rate Q. After the cosine ramp phase, the flow rate is set to $4\,\text{cm}^3/s$. Based on variations of the inflow rate, the heartbeats are simulated, where the inflow rate ranges between 2 and $6\,\text{cm}^3/s$. The heartbeat and the corresponding flow rate were constructed in [8] based on inflow pressure data from [134]. **Top Right:** Outflow area A_{outlet}. Due to the active material response, the artery contracts and the inflow and outflow areas decrease over time. **Bottom Left:** Outflow pressure p. Due to the decreasing outflow area, the pressure increases, as described by the absorbing boundary condition; see (9.4). **Bottom Right:** Radius of outflow area. The same behavior as for the outflow area can be observed. Compare also to the results in Figure 10.11 (right) for the deformation–diffusion problem.

10.7.1. Plaque Segment

The previous sections demonstrated that the proposed framework is able to efficiently solve the deformation–diffusion problem and the coupled FSI problem for a broad range of benchmark configurations. In particular, including the chemo-mechanical smooth muscle cell response has only had a marginal influence on the linear solver performance, and the proposed two-level overlapping Schwarz preconditioners were shown to be robust with respect to different material models, flow rates, and internal pressure levels. Furthermore, the long-term simulations in Figure 10.40 indicate robust linear solver behavior over extended simulation times.

In particular, in Figure 10.8, the average linear iteration count over time of the deformation–diffusion problem was compared for the (ChI) and (ChE) approach from (4.11) and (4.14),

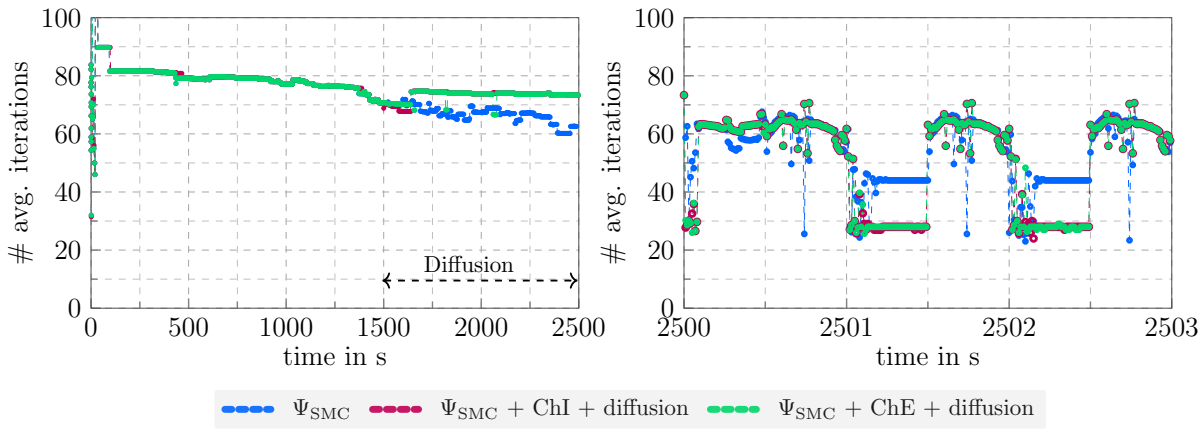


Figure 10.40: Average linear iteration count per Newton step per time step over 2 503 s. Computation on 52 cores. Corresponding solutions for outflow area, radius and pressure are shown in Figure 10.39. The fully implicit FSCI system contains 177 441 degrees of freedom. RGDSW coarse space with Option 2.2. Diffusion coefficient $D = 0.00006\,\text{cm}^2/\text{s}$. **Top**: Full range from 0 s to 2 500 s. **Bottom**: First 3 heartbeats are shown.

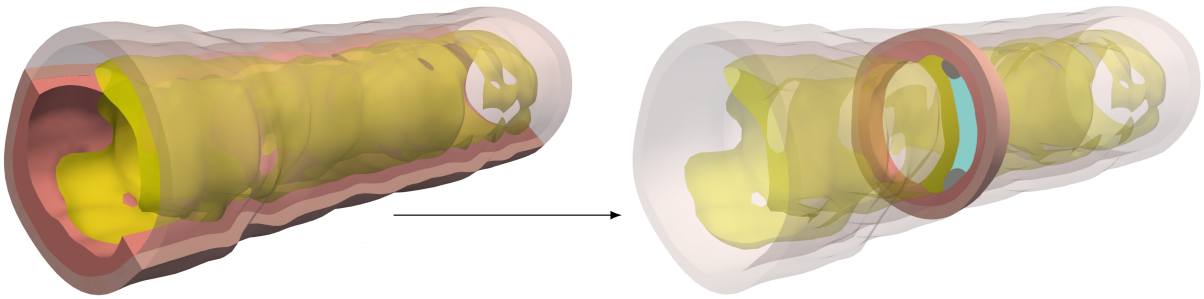


Figure 10.41: Schematic depiction of using the local segments of an atherosclerotic artery for simulations. The full patient-specific artery is taken from [7]. The arterial segment was obtained from Holzapfel et al. [98], and then reconstructed and preprocessed by Sharan Nurani Ramesh; see also [149, Section 2.3.1].

respectively, with active response and diffusion. We observed in Figure 10.7 that the weak scaling behavior of the Ψ_{SMC} model including the active material response is almost identical to only considering the passive material response Ψ_A. That is, approximately five average linear iterations less are required for the Ψ_{SMC} model; cf. Figure 10.7 (right). For the passive material model Ψ_A in Section 10.6, the strong scaling and robustness for variations in flow rate and internal pressure were tested for different material models and preconditioner settings. The proposed two-level overlapping Schwarz preconditioners delivered robust results with respect to the linear solve in all considered cases.

Modeling the realistic patient-specific artery from Section 9.4 with plaque components combined with the smooth muscle cell contraction is computationally expensive. The results in Figure 10.35 show that 1 120 time steps are required to reach the final time $t = 0.56$ s, with maximum time step size 0.0005 s to ensure convergence, which is required due to the nonlinear complexity of the problem and the material model. Compared to the idealized arterial segment considered in Figure 10.39, and the tests considering only the solid part of the realistic patient-specific

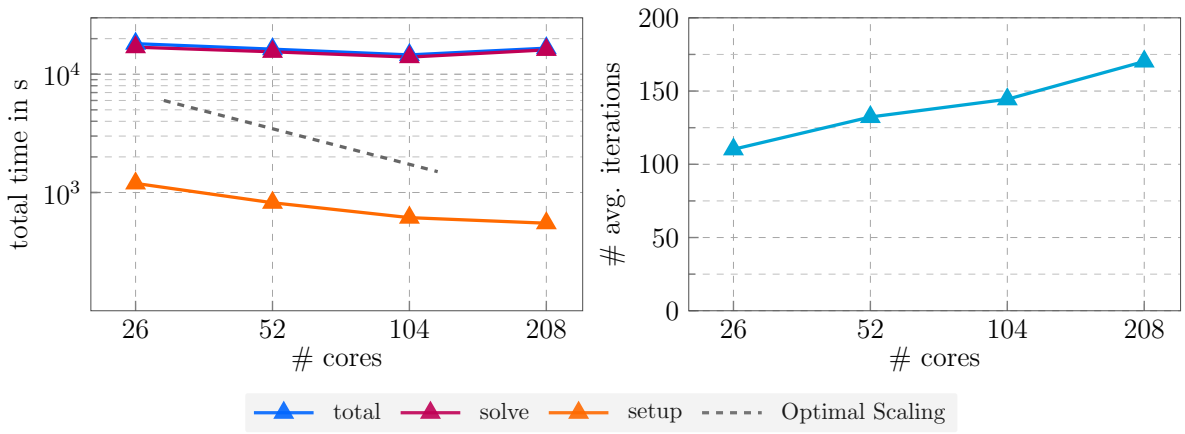


Figure 10.42: Strong scaling for plaque segment. The Ψ_{SMC} model from Section 3.1.2 with parameter *Set 4* is used with explicit approach (ChE); see Table 9.10. Simulation of 100 s. No diffusion is considered. **Left**: Total time consisting of setup and solve time. **Right**: Average iteration per Newton step per time step.

artery for the deformation–diffusion problem in Figure 10.12, the realistic patient-specific artery is not robust with respect to an increase in time step size. This potentially makes simulations of over 1 000 s unfeasible, when results for, e.g., patient-specific atherosclerotic arteries are to be analyzed to predict plaque rupture.

Instead, the use of local segments might be sufficient. A schematic image is shown in Figure 10.41. The arterial segment was obtained from Holzapfel et al. [98], and then reconstructed and processed by Sharan Nurani Ramesh; see also [149, Section 2.3.1]. The resulting FSCI system of the considered arterial segment contains only one tenth of the degrees of freedom compared to the full patient-specific artery. The depicted plaque segment contains six distinct regions, which mainly differ in stiffness and anisotropy; see Figure 10.43 (left). The corresponding parameters are shown in Table 9.10 and denoted as *Set 4*.

To analyze the solver performance, the following test setup is considered. The reference pressure is set to $p_{\text{ref}} = 18.67\,\text{kPa}\,(160\,\text{mmHg})$. The inflow rate is constant with $Q = 3\,\text{cm}^3/\text{s}$ after a cosine ramp phase. The time step size in the ramping phase is $\Delta t = 0.002\,\text{s}$; afterwards it is increased to $\Delta t = 0.05\,\text{s}$ until the final time of 100 s is reached. We do not consider a diffusion process. The active material response is accelerated to 4; compare to the results in Figure 10.8, where different solver setups were considered.

Contrary to the previously considered test cases, the plaque segment does not exhibit strong scaling; see Figure 10.42. This indicates that additional effects dominate the convergence behavior of the linear solver.

10.7.2. Further Analysis and Discussion

Compared to the parameter *Set 3* defined in Table 9.9, the parameter set *Set 4* exhibits stronger material heterogeneities. In particular, the stiffness and anisotropy parameters of the calcification and fibrous cap differ from those of the lipid pool by up to three and five orders of magnitude, respectively.

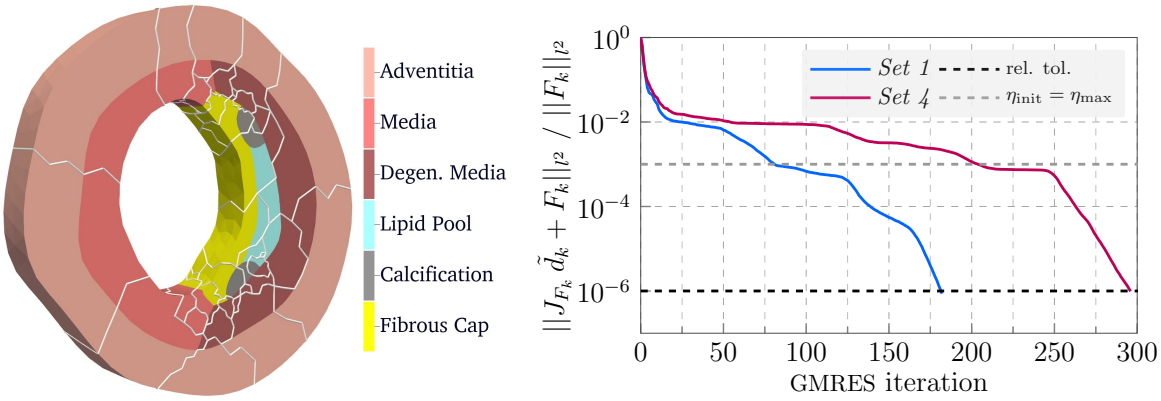


Figure 10.43: Arterial segment containing distinct plaque components and linear solver analysis. **Left:** Idealized arterial segment containing plaque. The segment is decomposed into different layers, containing specific regions for the plaque components. The white outline represents the unstructured partition of the mesh in 52 subdomains. The results of an FSCI simulation of the idealized arterial segment show a deterioration in the linear solver performance. **Right**: Relative GMRES residual at $t = 10\,s$ for 52 subdomains. We observe slow convergence for the first 250 iterations of the considered Newton step for the solid material modeled with Set 4. The material parameters are chosen according to Table 9.8 for *Set 1*, where the full geometry is modeled as media, and Table 9.10 for *Set 4*, for modeling the segment with distinct parameters for the different components.

Such large coefficient contrasts may impair the convergence of the iterative solver, especially when subdomain interfaces intersect regions with strongly varying material properties. The domain decomposition shown in Figure 10.43 indicates that this occurs for many subdomains adjacent to the fibrous cap. Consequently, the large material contrasts are expected to influence both the iterative solver and the overlapping Schwarz preconditioner.

In Figure 10.43 (right), the relative GMRES residual is plotted and compared for two cases at $t = 10\,s$. The material is modeled completely as media, where all components are modeled with the parameters in *Set 1*, or the material parameters for the different regions are chosen as in *Set 4* Table 9.10. There, a constant forcing term of 10^{-6} is used to consistently get the same achieved relative GMRES tolerance in each Newton iteration. Additionally, the value of $\eta_{\text{init}} = \eta_{\max}$ is denoted, which is used by default. For *Set 4* slow convergence is observed up to 250 iterations. The default $\eta_{\text{init}} = 10^{-3}$ for the first Newton iterations is reached after 200 iterations. For the *Set 1* parameters, the problem converges faster, and η_{init} is reached after 75 iterations. This also corresponds to the results shown in Figure 10.44 for *Set 1*, where the average linear iteration count per Newton step per time step is approximately 80.

To quantify the influence of the coefficient jumps, we consider different material compositions. Therefore, we define the following jump coefficients to measure the material heterogeneity

$$\kappa_{\text{jump}}(\text{calc}) = \frac{\alpha_{1,\text{calc}}}{\alpha_{1,\text{lipid}}} = \frac{11\,520}{9.3} \approx 1\,200 \tag{10.5}$$

and

$$\kappa_{\text{jump}}(\text{fibCap}) = \alpha_{4,\text{fibCap}} = 73\,360 \tag{10.6}$$

based on the parameter setting in Table 9.10 for α_1 and α_4. Since the lipid pool has no anisotropic part, the parameter α_4 is 0. Therefore, $\kappa_{\text{jump}}(\text{fibCap})$ uses the absolute value of α_4. Note that the parameter α_4 for the media is 1.3 kPa, which is also low compared to the fibrous cap.

In Figure 10.44, the influence of the material parameters on the linear solver performance is investigated. Two studies are performed. In Figure 10.44 (top), the isotropic stiffness parameter α_1 of the calcifications is varied while all remaining material parameters are fixed. In Figure 10.44 (bottom), the anisotropic stiffness parameter α_4 of the fibrous cap is varied. The average linear iteration count per Newton step and time step is shown for the different coefficient jumps defined in (10.5) and (10.6). As a reference, the average linear iteration count for a homogeneous plaque model is also reported, where all plaque components are assigned the material parameters of *Set 1* given in Table 9.8.

In the first study, α_1 of the calcifications is reduced from 11 520 kPa to 11.52 kPa, thereby decreasing the coefficient jump measure $\kappa_{\text{jump}}(\text{calc})$ between the calcifications and the lipid pool from approximately 1 200 to 1. This reduction results in only a moderate improvement of the solver performance. In fact, for the lowest values of α_1, the average linear iteration count even increases compared with the original parameter setting.

This observation suggests that the stiffness contrast between the calcifications and the lipid pool is not the dominant factor governing the convergence behavior. Furthermore, reducing α_1 increases the stiffness contrast between the isotropic calcifications and the anisotropic fibrous cap. This observation is consistent with the blue line and magenta-red line in Figure 10.44 (top), where the results obtained for $\alpha_1 = 11\,520$ kPa and $\alpha_1 = 11.52$ kPa are considerably closer than those corresponding to the intermediate parameter values.

The second study, shown in Figure 10.44 (bottom), therefore investigates the influence of the fibrous cap by reducing α_4 from 73 360 kPa to 733.60 kPa. In contrast to the variation of α_1, a substantial reduction in the average linear iteration count is observed. For example, at $t = 50$ s, the corresponding average iteration counts are 160, 108, and 88, for $\kappa_{\text{jump}}(\text{fibCap}) = 73\,360$, $\kappa_{\text{jump}}(\text{fibCap}) = 73$, and *Set 1*, respectively. These results indicate that the high anisotropic stiffness of the fibrous cap, and thus the large material contrast between the fibrous cap and the surrounding tissue, is the primary factor responsible for the deterioration of the linear solver performance.

Finally, we note that heterogeneous diffusion coefficients are also assigned to the different plaque components in [149]; see [149, Table 2]. Their influence is not considered in the present analysis, which focuses solely on the mechanical material parameters.

To improve the performance for the heterogeneous material parameters described by *Set 4*, adaptive coarse spaces could be considered. For symmetric positive definite problems, adaptive coarse spaces based on local generalized eigenvalue problems have been shown to provide robustness with respect to large coefficient contrasts; see, e.g., [161, 81, 93, 116, 1]. However, for unsymmetric and indefinite problems, a heuristically motivated construction may be useful and needs to be further investigated.

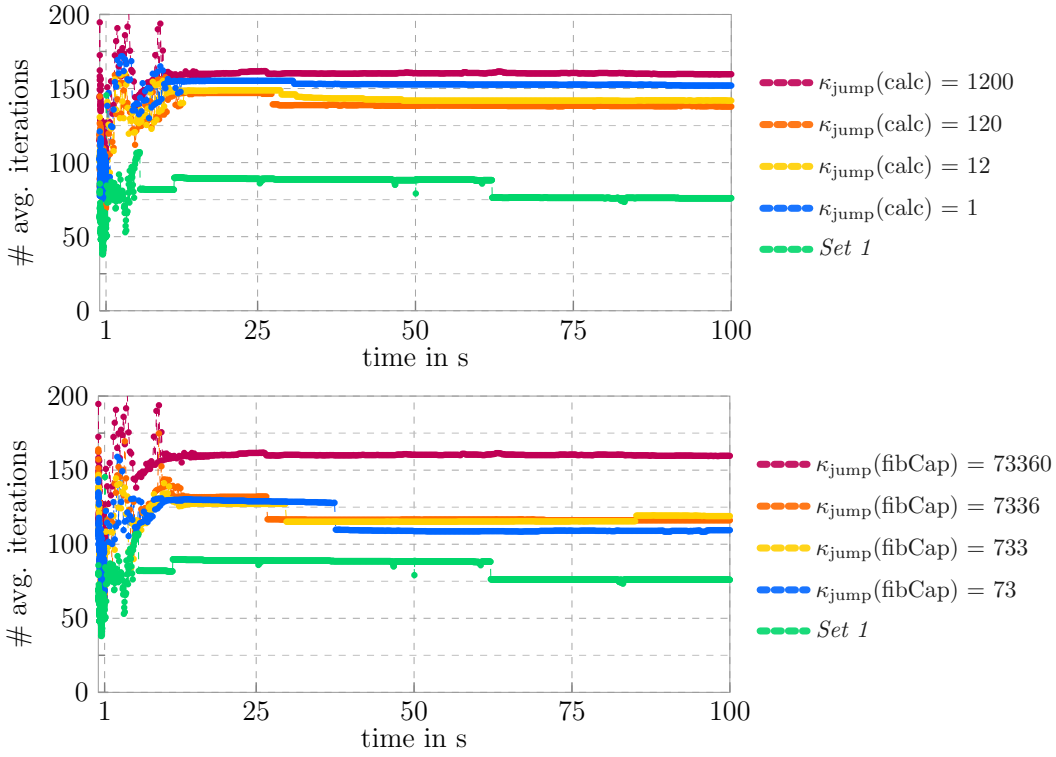


Figure 10.44: Average linear iteration count per Newton step over 100 s for different material settings for the plaque segments. Acceleration of active response with value 4. Computation on 52 cores. RGDSW coarse space with Option 2.2 using rotations in the coarse space. **Top**: Comparison of results for different coefficient jumps between lipid pool and calcification. Therefore, the calcification stiffness is reduced from 11 520 kPa to 11.520 kPa. As a reference, the segment is modeled homogeneously as media. **Bottom**: Comparison of results for different coefficient jumps between lipid pool and fibrous cap. Therefore, the calcification stiffness is reduced from 73 360 kPa to 73.360 kPa. As a reference, the segment is modeled homogeneously as media.

11. Conclusion

The proposed two-level overlapping Schwarz preconditioners for nonlinear solid mechanics exhibited robust and scalable performance under a wide range of conditions. Building on previous investigations for the neo-Hookean and Ψ_A materials, it was shown that both the passive arterial wall model Ψ_A and the active smooth muscle cell model Ψ_{SMC} exhibit comparable scaling behavior. Weak scalability was demonstrated for the fully coupled deformation–diffusion problem incorporating both active material response and drug diffusion. Furthermore, a comparison between fully implicit and explicit coupling strategies showed similar solver performance, while the relative difference between the computed solutions remained on the order of 10^{-4}.

The fluid preconditioners were assessed using both the backward-facing step benchmark and a realistic patient-specific artery under stationary and transient flow conditions. Reynolds numbers between 200 and 3 200, together with a broad range of CFL numbers, were considered to evaluate robustness. The results show that the monolithic preconditioner employing the newly proposed GDSW* coarse space for the velocity together with an RGDSW coarse space for the pressure consistently outperforms the considered block preconditioners PCD, LSC, and SIMPLEC; cf. [82]. Further analysis showed why the augmented Lagrangian preconditioner is not well suited for the combination with overlapping Schwarz methods in the considered test cases. Weak scalability was demonstrated for both the monolithic and PCD preconditioners up to 15 552 processor cores using different finite element discretizations for the stationary Navier–Stokes equations. For realistic arterial simulations, the monolithic approach remained the most robust with respect to variations in viscosity and Reynolds number. Although the setup of the monolithic preconditioner is more expensive compared to block preconditioners, the reduced number of linear iterations generally results in lower overall computation times, particularly for simulations with large CFL numbers; cf. [82].

The results observed for the individual fluid and structural subproblems transferred to the fully coupled FSI formulation. In particular, the monolithic fluid preconditioner consistently performed better than the SIMPLE-based FaCSI variant, yielding a lower iteration count and improved robustness for high inflow rates; cf. [107]. With a focus on the solid subproblem the robustness of the solver was furthermore demonstrated for varying pressure levels and different structural material models. Results for the FSI problem using the Ψ_A model for different parameter sets and both homogeneous and heterogeneous compositions suggest that similar results can be achieved with the Ψ_{SMC} model, based on the observations from the deformation–diffusion problem.

The developed `FEDDLib`–`AceGen` interface was validated by a variety of tests. The results demonstrate both the correctness of the generated finite element routines and the parallel scalability of the newly integrated assembly procedures. In particular, the implementation was verified by a comparison with the curved artery FSI benchmark.

For the fully coupled pharmaco–mechanical FSCI problem, the proposed framework successfully

reproduced the expected qualitative behavior of the drug–tissue interaction. For the simplified arterial segment, the coupled solver together with the proposed preconditioning strategy proved robust, providing a proof of concept for the complete computational framework.

Future Work Throughout this dissertation, several aspects have already been identified as warranting further investigation. The presented framework provides a basis for large-scale pharmaco–mechanical FSI simulations; however, several modeling and numerical challenges remain open.

The numerical results indicate that the construction of monolithic coarse basis functions can have a significant influence on the robustness of the proposed preconditioners. This is illustrated by the comparison presented in Figure 10.14. Different design choices, such as the inclusion of off-diagonal blocks, the application of pressure projections, and the selection of the underlying coarse spaces, can considerably affect both robustness and parallel scalability. The observations summarized in Section 7.4 provide several indications but do not yet yield a theoretical understanding. A systematic investigation of these effects therefore remains an interesting direction for future research.

Although the proposed methods exhibit good parallel scalability, further improvements are possible. In particular, the use of parallel coarse solvers could reduce the coarse-grid bottleneck, as already observed in [78]. Furthermore, the influence of the mesh partitioning on the coarse space dimension and the overall solver performance deserve further investigation. In particular, a comparison of structured and unstructured mesh partitions may provide further insights into the construction of more efficient coarse spaces; see Section 10.6.3.

The numerical experiments indicate that highly heterogeneous material parameters may deteriorate the performance of the linear solver and the overlapping Schwarz preconditioner; see Section 10.7.2. Developing suitable coarse spaces for these coupled problems therefore represents another field of further research.

The definition of physiologically meaningful boundary conditions for the considered pharmaco–mechanical FSI problem remains an open challenge; see also Section 9.6.4. In particular, the boundary conditions should account for the interaction between blood pressure, the actively changing vessel diameter caused by smooth muscle contraction, and the influence of antihypertensive medication. The degree of contraction is influenced by the vessel stretch through the active material response; see for example (3.9). The stretch in turn is affected by the current internal pressure. These aspects need to be considered in developing boundary conditions, which remains an important topic for future work.

A. Appendix

A.1. Parameters

Table A.1: Parameters for the chemical model in the active material response in Ψ_{SMC} from Section 3.1.2. Adopted from [168].

Parameter	k_3	k_4	k_7	Ca_{50}	γ_2	γ_3	$\lambda_{50,\mathrm{c}}$	$\bar{\lambda}_{\mathrm{c,start}}$	$\lambda_{\mathrm{a,start}}$
Value	$0.134\,\mathrm{s}^{-1}$	$0.00166\,\mathrm{s}^{-1}$	$0.000066\,\mathrm{s}^{-1}$	$0.4\,\mu\mathrm{M}$	$50\,\mu\mathrm{M}^{-1}$	$0.9\,\mu\mathrm{M}$	1.2	1.0	1.0
Parameter	γ_4	ζ_1	γ_5	ζ_2	$\Delta\bar{\lambda}_{\mathrm{p,min}}$	γ_6	$\lambda_{50,\mathrm{p}}$	$\bar{\lambda}_{\mathrm{p,start}}$	$\lambda_{\mathrm{e,start}}$
Value	200	100 s	50 s	1 000	−0.00001	$1.5\,\mathrm{s}^{-1}$	1.0	1.0	1.0

Table A.2: Parameters for the active chemo-mechanical material response in Ψ_{SMC} from Section 3.1.2. Adopted from [168].

Parameter	η	γ_1	$\dot{\bar{\lambda}}_{\mathrm{c,max}}$	$\dot{\bar{\lambda}}_{\mathrm{c,min}}$	$\dot{k}_{2/5,\mathrm{max}}$	$\dot{k}_{2/5,\mathrm{min}}$
Value	$0.1624\,\mathrm{s}^{-1}$	$0.5131\,\mu\mathrm{M}$	$0.0443\,\mathrm{s}^{-1}$	$-0.0443\,\mathrm{s}^{-1}$	$0.0009735\,\mathrm{s}^{-2}$	$-0.0010694\,\mathrm{s}^{-2}$
Parameter	$\dot{\bar{\lambda}}_{\mathrm{p,max}}$	$\dot{\bar{\lambda}}_{\mathrm{p,min}}$	μ_{a}	κ	β_1	$k_{2/5,\mathrm{start}}$
Value	$0.0000699\,\mathrm{s}^{-1}$	$-0.0002323\,\mathrm{s}^{-1}$	11.857 kPa	148.262 kPa	$0.001006\,\mathrm{s}^{-1}$	$1.82758\,\mathrm{s}^{-1}$

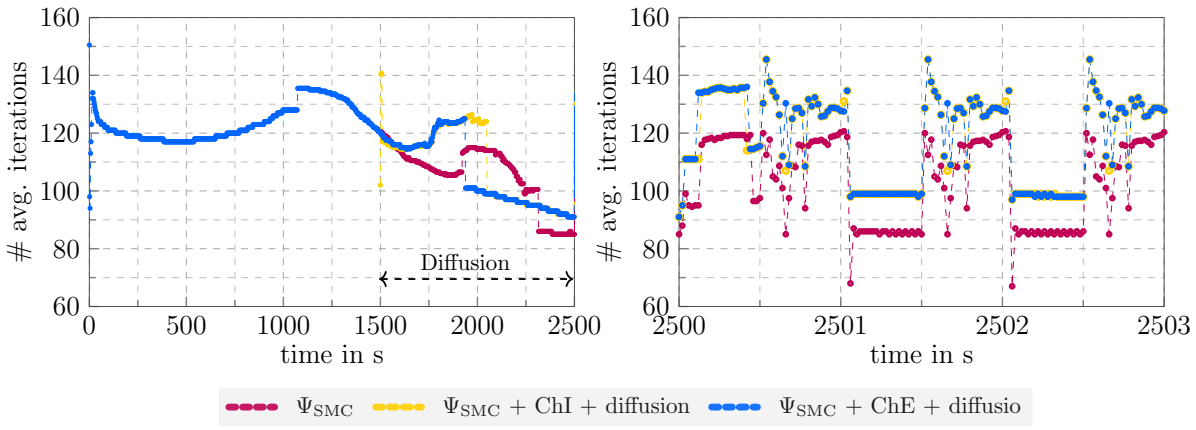


Figure A.2: Average linear iteration count per Newton step per time step over 2503 s. Computation on 52 cores. Corresponding solution and pressure protocol are shown in Figure 10.11. The fully implicit system contains 89 970 degrees of freedom. RGDSW coarse space with Option 2.2. Diffusion coefficient $D = 0.00006\,\mathrm{cm}^2/\mathrm{s}$. **Top**: Full range from 0 s to 2 500 s. **Bottom**: First 3 heartbeats are shown.

A.2. Preconditioning the Solid Problem

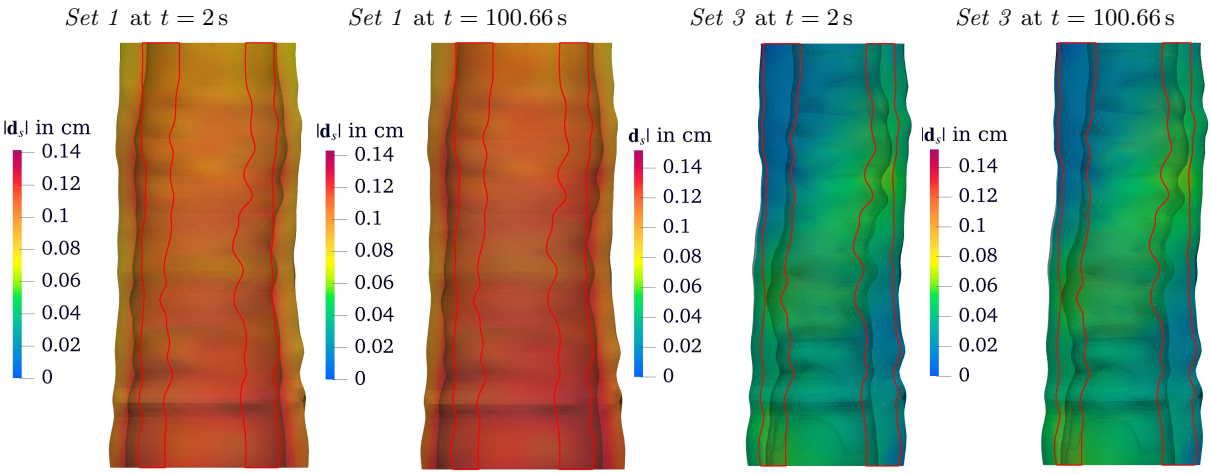


Figure A.1: Results of displacement and deformation of SCI problem using realistic patient-specific artery from Section 9.4 with different internal pressure values and different arterial wall models. At $t = 2\,\mathrm{s}$ a pressure load of $p = 16.00\,\mathrm{kPa}$ is applied, and at $t = 100.66\,\mathrm{s}$ (120 mmHg) a pressure load of $p = 18.67\,\mathrm{kPa}$ (140 mmHg) is reached. Results correspond to strong scaling study in Figure 10.12. The Ψ_{SMC} model from Section 3.1.2 with parameter *Set 1* and *Set 3*, i.e., homogeneous and heterogeneous material composition; see Tables 9.8 and 9.9. In red, the outline of the undeformed artery is shown.

A.3. Stationary Navier–Stokes Problem

In Figure A.3 results for a stationary, three-dimensional lid-driven cavity problem are shown to ensure that the results seen before in Figure 10.14 are not particularly problem specific: The

Table A.3: Strong scaling results for realistic patient-specific artery with different wall models in deformation–diffusion problem. See Section 9.4 for the test case. Setup time consists of construction of preconditioner and solve time consists of linear and nonlinear solve. Average linear iteration count per Newton step per time step. The Ψ_{SMC} model from Section 3.1.2 with parameter *Set 1* and *Set 3*, i.e., homogeneous and heterogeneous material composition; see Tables 9.8 and 9.9. Simulation of 101 s with one heartbeat at 100 s. In total 795 time steps. The diffusion coefficient is set to $D = 6 \cdot 10^{-4}\,\text{cm}^2/\text{s}$. Results of corresponding deformation in Figure A.1. Explicit coupling. Rotational modes in the coarse space denoted as *w rot* and without as *w/o rot* in the solid preconditioner. Corresponding scaling plot in Figure 10.12.

	Ψ_A Set 1					
	w/o rot.			w rot		
N	# avg. iter	setup	solve	# avg. iter	setup	solve
26	57.4 (2.3)	8 776 s	7 971 s	51.9 (2.3)	9 189 s	7 488 s
52	66.3 (2.3)	3 855 s	4 950 s	58.9 (2.3)	4 020 s	4 667 s
104	79.1 (2.3)	1 725 s	3 279 s	70.6 (2.3)	1 842 s	3 002 s
208	99.3 (2.3)	908 s	2 462 s	86.2 (2.3)	1 055 s	2 277 s
	Ψ_A Set 3					
	w/o rot.			w rot		
N	# avg. iter	setup	solve	# avg. iter	setup	solve
26	57.8 (1.8)	8 781 s	6 489 s	53.6 (1.8)	8 870 s	6 307 s
52	66.4 (1.8)	3 849 s	4 035 s	62.9 (1.8)	4 104 s	3 830 s
104	86.7 (1.8)	1 717 s	2 968 s	78.5 (1.8)	1 870 s	2 676 s
208	96.2 (1.8)	909 s	2 075 s	88.3 (1.8)	1 036 s	1 951 s

results for the monolithic preconditioner, different combinations of coarse spaces, and the use of the pressure projection (7.11) qualitatively show the same behavior as before for the stationary backward-facing step problem.

The setup uses the classic formulation, where the entire top part of a unit cube is moved at constant velocity in one direction; see, for example, [77, Sect. 2.1]. We use a structured mesh of the unit cube $[0,1]^3$, decomposed into cubic subdomains. The subdomain resolution is $H/h = 9$, giving for a P2–P1 discretization 4.1 million degrees of freedom if $6^3 = 216$ subdomains are used and 286.7 million degrees of freedom if $25^3 = 15\,625$ subdomains are used. The kinematic viscosity is $\nu = 0.005$, giving $\text{Re} = 200$. The lid includes its boundary points (classic lid-driven cavity formulation). The Dirichlet condition for the velocity at the lid is $(1, 0, 0)$ in each node and zero elsewhere.

It was not necessary to ensure uniqueness of the pressure by fixing a pressure node or introducing a projection into GMRES; note that prescribing a zero mean integral via a Lagrange multiplier can be computationally inefficient in a parallel setting, as the associated vector in the system

matrix is dense and couples globally across all subdomains.

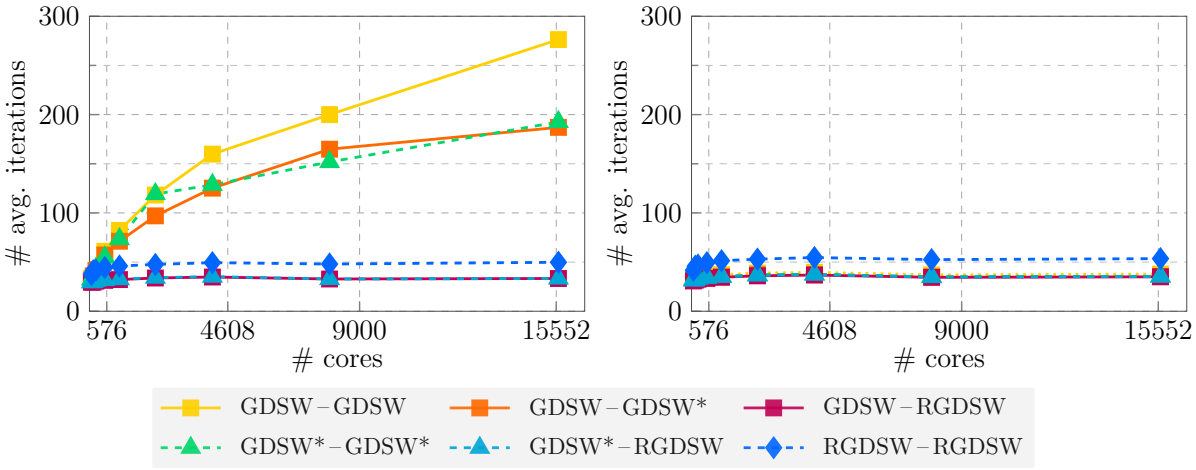


Figure A.3: Comparison of different coarse space combinations for the velocity (first component) and pressure (second component) for P2–P1 discretization for the monolithic preconditioner. Average GMRES iteration count per Newton step. Weak scaling test. Stationary lid-driven cavity problem with $H/h = 9$; see Appendix A.3 for a description. Re = 200. Using pressure projection in first level of monolithic preconditioner. See Figure 10.14 for corresponding BFS problem. **Left:** Excluding off-diagonal blocks in ϕ to build the coarse matrix. **Right:** Using full ϕ to build the coarse matrix. Adapted from [82, Figure A.1].

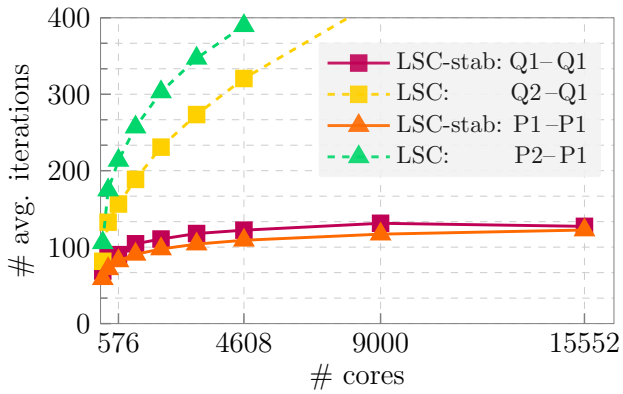


Figure A.4: Use of RGDSW–RGDSW coarse space for the velocity (first component) and for Schur complement components (second component). Average GMRES iteration count per Newton step. Total number of Newton iterations required to reach convergence is 5. Weak scaling test. Stationary BFS problem with configurations according to Table 9.1. Re = 200. Use of Bochev–Dohrmann stabilization for P1–P1 and Q1–Q1 elements. No use of pressure projection. LSC block preconditioner; see Section 6.1.2. Adapted from [82, Figure A.3].

A.4. Transient Navier–Stokes Problem

Table A.4: Comparison of different finite element discretizations for the monolithic preconditioner for an increasing Reynolds number. Average GMRES iteration count per Newton step over all time steps. Transient BFS problem with configurations according to Table 9.1. Time step size is constant, $\Delta t = 0.05\,\mathrm{s}$, simulation until $t = 10\,\mathrm{s}$, which amounts to 200 time steps in total. Use of Bochev–Dohrmann stabilization for P1–P1 and Q1–Q1 elements. 243 processor cores. Combinations of different coarse spaces for the velocity (first component) and pressure (second component). No use of pressure projection. Exclude off-diagonal blocks in ϕ. Results corresponding to Figure 10.17. Taken from [82, Table B.1].

Discretization			P1–P1	P2–P1	Q1–Q1	Q2–Q1	Q2–P1-disc.
Coarse space		Velocity	RGDSW	GDSW⋆	RGDSW	GDSW⋆	GDSW⋆
		Pressure	RGDSW	RGDSW	RGDSW	RGDSW	GDSW⋆
Re	ν	$v_{\max}$	# avg. iter. per Newton step over all time steps				
200	0.01	1	13.7	19.8	12.8	21.8	31.7
400	0.005	1	13.8	18.7	14.3	19.8	35.3
800	0.0025	1	14.9	19.5	14.9	20.7	37.3
1 600	0.00125	1	14.9	20.8	14.7	22.6	39.1
3 200	0.000625	1	15.5	21.7	13.4	20.3	40.7
200	0.01	1	13.7	19.8	12.8	21.8	31.7
400	0.01	2	12.8	18.7	12.3	20.0	30.7
800	0.01	4	13.1	20.3	13.6	22.0	31.7
1 600	0.01	8	14.5	22.7	14.5	23.6	33.6
3 200	0.01	16	24.9	24.4	24.3	27.5	39.9

A.5. Comparison of Block and Monolithic Preconditioners for Varying Reynolds Numbers

Table A.5: Comparison of block and monolithic preconditioners for an increasing Reynolds number. Average GMRES iteration count per Newton step per time step. Total time in seconds consisting of setup and solve time. Transient BFS problem with P2–P1 discretization and $H/h = 9$. Increasing Reynolds number due to decreasing viscosity. Computation of 200 time steps with time step size $\Delta t = 0.05\,\text{s}$, simulation until $t = 10\,\text{s}$. CFL number is constant and equal to 1.35. Computations on 243 cores. Boundary condition strategy in PCD is (BC–3). Use of GDSW* and RGDSW coarse spaces. Results corresponding to Figure 10.24. Taken from [82, Table C.1].

	Re	200	400	800	1 600	3 200
	ν	$1.0{\cdot}10^{-2}$	$5.0{\cdot}10^{-3}$	$2.5{\cdot}10^{-3}$	$1.3{\cdot}10^{-3}$	$6.25{\cdot}10^{-4}$
SIMPLEC RGDSW–RGDSW	# avg. iter.	38.6 (3.0)	39.0 (3.0)	41.1 (3.0)	46.7 (3.0)	52.7 (3.0)
	Setup time	596 s	594 s	598 s	599 s	595 s
	Solve time	2 089 s	2 116 s	2 298 s	2 570 s	2 901 s
	Total	2 685 s	2 710 s	2 896 s	3 169 s	3 496 s
PCD RGDSW–RGDSW	# avg. iter.	29.9 (2.8)	30.7 (3.0)	28.9 (3.0)	28.8 (3.0)	31.7 (3.0)
	Setup time	584 s	578 s	577 s	581 s	580 s
	Solve time	1 592 s	1 700 s	1 629 s	1 600 s	1 761 s
	Total	2 176 s	2 278 s	2 206 s	2 181 s	2 341 s
Monolithic RGDSW–RGDSW	# avg. iter.	26.9 (2.8)	29.2 (3.0)	30.5 (3.0)	32.7 (3.0)	35.0 (3.0)
	Setup time	795 s	789 s	789 s	785 s	785 s
	Solve time	1 458 s	1 671 s	1 747 s	1 873 s	2 008 s
	Total	2 253 s	2 460 s	2 536 s	2 658 s	2 793 s
Monolithic GDSW*–RGDSW	# avg. iter.	19.8 (2.8)	18.7 (2.8)	19.5 (2.9)	20.8 (3.0)	21.7 (3.1)
	Setup time	790 s	783 s	786 s	782 s	787 s
	Solve time	1 074 s	1 035 s	1 122 s	1 203 s	1 268 s
	Total	**1 864 s**	**1 818 s**	**1 908 s**	**1 984 s**	**2 055 s**

A.6. Pharmaco–Mechanical FSI

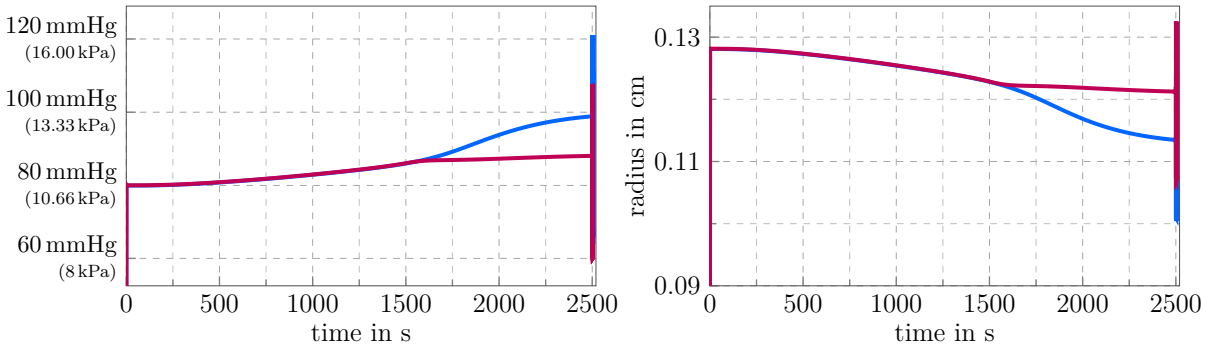


Figure A.5: Results of full FSCI simulation using idealized arterial segment; cf. Section 9.3. Reference pressure is set to 10.66 kPa (80 mmHg) in absorbing boundary condition. Simulation spans 2 505 s with 5 heartbeats starting at 2 500 s. The solid part is modeled as media with parameters corresponding to *Set 1* in Table 9.8. **Bottom Left:** Outflow pressure p. Due to the decreasing outflow area, the pressure increases, as described by the absorbing boundary condition; see (9.4). **Bottom Right:** Radius of outflow area. The same behavior as for the outflow area can be observed. Compare also to the results in Figure 10.11 (right) for the deformation–diffusion problem.

Bibliography

[1] H. Al Daas, P. Jolivet, F. Nataf, and P.-H. Tournier, *A robust algebraic two-level Schwarz preconditioner for sparse matrices*, SIAM Journal on Scientific Computing, 47 (2025), pp. A2378–A2402, https://doi.org/10.1137/24M1629882.

[2] P. R. Amestoy, A. Buttari, J.-Y. L'Excellent, and T. Mary, *Performance and Scalability of the Block Low-Rank Multifrontal Factorization on Multicore Architectures*, ACM Transactions on Mathematical Software, 45 (2019), pp. 2:1–2:26, https://doi.org/10.1145/3242094.

[3] P. R. Amestoy, I. S. Duff, J. Koster, and J.-Y. L'Excellent, *A fully asynchronous multifrontal solver using distributed dynamic scheduling*, SIAM Journal on Matrix Analysis and Applications, 23 (2001), pp. 15–41, https://doi.org/10.1137/S0895479899358194.

[4] P. R. Amestoy, A. Guermouche, J.-Y. L'Excellent, and S. Pralet, *MUMPS Users' Guide*, MUMPS Team, ENSEEIHT-IRIT, Toulouse, France and LIP-ENS Lyon, France, 2023. Available at https://mumps-solver.org.

[5] B. F. Armaly, F. Durst, J. C. F. Pereira, and B. Schönung, *Experimental and theoretical investigation of backward-facing step flow*, Journal of Fluid Mechanics, 127 (1983), pp. 473–496, https://doi.org/10.1017/S0022112083002839.

[6] S. Averweg, A. Schwarz, C. Schwarz, and J. Schröder, *A monolithic fluid–structure interaction approach using mixed LSFEM with high-order time integration*, Computer Methods in Applied Mechanics and Engineering, 423 (2024), p. 116783, https://doi.org/10.1016/j.cma.2024.116783.

[7] D. Balzani, D. Böse, D. Brands, R. Erbel, A. Klawonn, O. Rheinbach, and J. Schröder, *Parallel simulation of patient-specific atherosclerotic arteries for the enhancement of intravascular ultrasound diagnostics*, Engineering Computations, 29 (2012), pp. 888–906, https://doi.org/10.1108/02644401211271645.

[8] D. Balzani, S. Deparis, S. Fausten, D. Forti, A. Heinlein, A. Klawonn, A. Quarteroni, O. Rheinbach, and J. Schröder, *Numerical modeling of fluid-structure interaction in arteries with anisotropic polyconvex hyperelastic and anisotropic viscoelastic material models at finite strains*, International Journal for Numerical Methods in Biomedical Engineering, 32 (2016), p. e02756, https://doi.org/10.1002/cnm.2756.

[9] D. Balzani, A. Heinlein, A. Klawonn, J. Knepper, S. Nurani Ramesh, O. Rheinbach, L. Saßmannshausen, and K. Uhlmann, *A computational framework for pharmaco-mechanical inter-*

actions in arterial walls using parallel monolithic domain decomposition methods, GAMM-Mitteilungen, 47 (2024), p. e202370002, https://doi.org/10.1002/gamm.202370002.

[10] D. Balzani, A. Heinlein, A. Klawonn, O. Rheinbach, and J. Schröder, *Comparison of arterial wall models in fluid–structure interaction simulations*, Computational Mechanics, 72 (2023), pp. 1–17, https://doi.org/10.1007/s00466-023-02321-y.

[11] D. Balzani, A. Klawonn, J. Knepper, S. N. Ramesh, O. Rheinbach, and L. Saßmannshausen, *An AceGen-FEDDLib software interface*, In Preparation, (2026).

[12] D. Balzani, P. Neff, J. Schröder, and G. A. Holzapfel, *A polyconvex framework for soft biological tissues. adjustment to experimental data*, International Journal of Solids and Structures, 43 (2006), pp. 6052–6070, https://doi.org/10.1016/j.ijsolstr.2005.07.048.

[13] A. Barker and X.-C. Cai, *Scalable parallel methods for monolithic coupling in fluid–structure interaction with application to blood flow modeling*, Journal of Computational Physics, 229 (2010), pp. 642–659, https://doi.org/10.1016/j.jcp.2009.10.001.

[14] R. A. Bartlett, *Stratimikos wrapper package*, Tech. Report STRATIMIKOS; 001978MLTPL00, Sandia National Laboratories (SNL), Albuquerque, NM, and Livermore, CA (United States), Aug. 2006, https://www.osti.gov/biblio/1245170.

[15] E. Bavier, M. Hoemmen, S. Rajamanickam, and H. Thornquist, *Amesos2 and Belos: Direct and iterative solvers for large sparse linear systems*, Scientific Programming, 20 (2012), pp. 241–255, https://doi.org/10.3233/SPR-2012-0352.

[16] M. Benzi, G. Golub, and J. Liesen, *Numerical solution of saddle point problems*, Acta Numerica, 14 (2005), pp. 1–137, https://doi.org/10.1017/S0962492904000212.

[17] M. Benzi and M. A. Olshanskii, *An augmented Lagrangian-based approach to the Oseen problem*, SIAM Journal on Scientific Computing, 28 (2006), pp. 2095–2113, https://doi.org/10.1137/050646421.

[18] M. Benzi, M. A. Olshanskii, and Z. Wang, *Modified augmented Lagrangian preconditioners for the incompressible Navier–Stokes equations*, International Journal for Numerical Methods in Fluids, 66 (2011), pp. 486–508, https://doi.org/10.1002/fld.2267.

[19] M. Benzi and Z. Wang, *Analysis of augmented Lagrangian-based preconditioners for the steady incompressible Navier–Stokes equations*, SIAM Journal on Scientific Computing, 33 (2011), pp. 2761–2784, https://doi.org/10.1137/100797989.

[20] M. Benzi and Z. Wang, *A parallel implementation of the modified augmented Lagrangian preconditioner for the incompressible Navier–Stokes equations*, Numerical Algorithms, 64 (2013), pp. 73–84, https://doi.org/10.1007/s11075-012-9655-x.

[21] L. Berger-Vergiat, C. A. Glusa, J. J. Hu, M. Mayr, A. Prokopenko, C. M. Siefert, R. S. Tuminaro, and T. A. Wiesner, *MueLu user's guide*, Tech. Report SAND2019-0537, Sandia

National Laboratories, Albuquerque, NM, and Livermore, CA, 2019, https://doi.org/10.2172/1491860.

[22] T. Bevilacqua, A. Gumenyuk, N. Habibi, P. Hartwig, A. Klawonn, M. Lanser, M. Rethmeier, L. Scheunemann, and J. Schröder, *Large-scale thermo-mechanical simulation of laser beam welding using high-performance computing: A qualitative reproduction of experimental results*, Results in Engineering, 29 (2026), p. 108827, https://doi.org/10.1016/j.rineng.2025.108827.

[23] T. Bevilacqua, A. Klawonn, and M. Lanser, *Highly scalable two-level monolithic overlapping Schwarz preconditioners for thermo-elastoplastic laser beam welding problems*, Computational Mechanics, 77 (2026), pp. 485–499, https://doi.org/10.1007/s00466-025-02674-6.

[24] T. Bevilacqua, A. Klawonn, M. Lanser, and A. Wasiak, *Monolithic overlapping Schwarz preconditioners for nonlinear finite element simulations of laser beam welding processes*, in Computational Mechanics and Applied Mathematics: Perspectives from Young Scholars, F. Marmo, S. Cuomo, and A. Cutolo, eds., Cham, 2025, Springer, pp. 123–133, https://doi.org/10.1007/978-3-031-76591-9_12.

[25] D. Boffi, F. Brezzi, and M. Fortin, *Mixed Finite Element Methods and Applications*, Springer, Berlin, Heidelberg, Germany, 2013, https://doi.org/10.1007/978-3-642-36519-5.

[26] N. Bootland, A. Bentley, C. Kees, and A. Wathen, *Preconditioners for two-phase incompressible Navier–Stokes flow*, SIAM Journal on Scientific Computing, 41 (2019), pp. B843–B869, https://doi.org/10.1137/17M1153674.

[27] D. Brands, A. Klawonn, O. Rheinbach, and J. Schröder, *Modelling and convergence in arterial wall simulations using a parallel FETI solution strategy*, Computer Methods in Biomechanics and Biomedical Engineering, 11 (2008), pp. 569–583, https://doi.org/10.1080/10255840801949801.

[28] W. L. Briggs, V. E. Henson, and S. F. McCormick, *A Multigrid Tutorial*, Society for Industrial and Applied Mathematics, Philadelphia, 2 ed., 2000, https://doi.org/10.1137/1.9780898719505.

[29] M. Bucelli, L. Dede, A. Quarteroni, and C. Vergara, *Partitioned and monolithic algorithms for the numerical solution of cardiac fluid-structure interaction*, Communications in Computational Physics, 32 (2022), pp. 1217–1256, https://doi.org/10.4208/cicp.OA-2021-0243.

[30] X.-C. Cai and M. Sarkis, *A restricted additive Schwarz preconditioner for general sparse linear systems*, SIAM Journal on Scientific Computing, 21 (1999), pp. 792–797, https://doi.org/10.1137/S106482759732678X.

[31] P. Causin, J. Gerbeau, and F. Nobile, *Added-mass effect in the design of partitioned algorithms for fluid–structure problems*, Computer Methods in Applied Mechanics and Engineering, 194 (2005), pp. 4506–4527, https://doi.org/10.1016/j.cma.2004.12.005.

[32] J.-M. Cros, *A preconditioner for the Schur complement domain decomposition method*, in Domain Decomposition Methods in Science and Engineering, National Autonomous University of Mexico (UNAM), Mexico City, Mexico, 2003, pp. 373–380.

[33] P. Crosetto, S. Deparis, G. Fourestey, and A. Quarteroni, *Parallel algorithms for fluid-structure interaction problems in haemodynamics*, SIAM Journal on Scientific Computing, 33 (2011), pp. 1598–1622, https://doi.org/10.1137/090772836.

[34] E. C. Cyr, J. N. Shadid, and R. S. Tuminaro, *Stabilization and scalable block preconditioning for the Navier–Stokes equations*, Journal of Computational Physics, 231 (2012), pp. 345–363, https://doi.org/10.1016/j.jcp.2011.09.001.

[35] E. C. Cyr, J. N. Shadid, and R. S. Tuminaro, *Teko: A block preconditioning capability with concrete example applications in Navier-Stokes and MHD*, SIAM Journal on Scientific Computing, 38 (2016), pp. S307–S331, https://doi.org/10.1137/15M1017946.

[36] S. Deparis, D. Forti, G. Grandperrin, and A. Quarteroni, *FaCSI: A block parallel preconditioner for fluid–structure interaction in hemodynamics*, Journal of Computational Physics, 327 (2016), pp. 700–718, https://doi.org/10.1016/j.jcp.2016.10.005.

[37] S. Deparis, G. Grandperrin, and A. Quarteroni, *Parallel preconditioners for the unsteady Navier–Stokes equations and applications to hemodynamics simulations*, Computers & Fluids, 92 (2014), pp. 253–273, https://doi.org/10.1016/j.compfluid.2013.10.034.

[38] C. R. Dohrmann, *A preconditioner for substructuring based on constrained energy minimization*, SIAM Journal on Scientific Computing, 25 (2003), pp. 246–258, https://doi.org/10.1137/S1064827502412887.

[39] C. R. Dohrmann and P. B. Bochev, *A stabilized finite element method for the Stokes problem based on polynomial pressure projections*, International Journal for Numerical Methods in Fluids, 46 (2004), pp. 183–201, https://doi.org/10.1002/fld.752.

[40] C. R. Dohrmann, A. Klawonn, and O. B. Widlund, *Domain decomposition for less regular subdomains: Overlapping Schwarz in two dimensions*, SIAM Journal on Numerical Analysis, 46 (2008), pp. 2153–2168, https://doi.org/10.1137/070685841.

[41] C. R. Dohrmann, A. Klawonn, and O. B. Widlund, *A family of energy minimizing coarse spaces for overlapping Schwarz preconditioners*, in Domain Decomposition Methods in Science and Engineering XVII, vol. 60 of Lecture Notes in Computational Science and Engineering, Springer, 2008, pp. 247–254, https://doi.org/10.1007/978-3-540-75199-1_28.

[42] C. R. Dohrmann and O. B. Widlund, *An overlapping Schwarz algorithm for almost incompressible elasticity*, SIAM Journal on Numerical Analysis, 47 (2009), pp. 2897–2923, https://doi.org/10.1137/080724320.

[43] C. R. Dohrmann and O. B. Widlund, *Hybrid domain decomposition algorithms for compressible and almost incompressible elasticity*, International Journal for Numerical Methods in Engineering, 82 (2010), pp. 157–183, https://doi.org/10.1002/nme.2761.

[44] C. R. Dohrmann and O. B. Widlund, *Lower dimensional coarse spaces for domain decomposition*, in Domain Decomposition Methods in Science and Engineering XXI, 2014, pp. 527–535, https://doi.org/10.1007/978-3-319-05789-7_50.

[45] C. R. Dohrmann and O. B. Widlund, *On the design of small coarse spaces for domain decomposition algorithms*, SIAM Journal on Scientific Computing, 39 (2017), pp. A1466–A1488, https://doi.org/10.1137/17M1114272.

[46] J. Donea, A. Huerta, J.-P. Ponthot, and A. Rodríguez-Ferran, *Arbitrary Lagrangian–Eulerian methods*, in Encyclopedia of Computational Mechanics, vol. 1, John Wiley & Sons, 2004, https://doi.org/10.1002/0470091355.ecm009.

[47] J. P. V. Doormaal and G. D. Raithby, *Enhancements of the simple method for predicting incompressible fluid flows*, Numerical Heat Transfer, 7 (1984), pp. 147–163, https://doi.org/10.1080/01495728408961817.

[48] H. C. Edwards, C. R. Trott, and D. Sunderland, *Kokkos: Enabling manycore performance portability through polymorphic memory access patterns*, Journal of Parallel and Distributed Computing, 74 (2014), pp. 3202–3216, https://doi.org/10.1016/j.jpdc.2014.07.003.

[49] S. C. Eisenstat and H. F. Walker, *Globally convergent inexact Newton methods*, SIAM Journal on Optimization, 4 (1994), pp. 393–422, https://doi.org/10.1137/0804022.

[50] S. C. Eisenstat and H. F. Walker, *Choosing the forcing terms in an inexact Newton method*, SIAM Journal on Scientific Computing, 17 (1996), pp. 16–32, https://doi.org/10.1137/0917003.

[51] H. Elman, V. Howle, J. Shadid, R. Shuttleworth, and R. Tuminaro, *A taxonomy and comparison of parallel block multi-level preconditioners for the incompressible Navier–Stokes equations*, Journal of Computational Physics, 227 (2008), pp. 1790–1808, https://doi.org/10.1016/j.jcp.2007.09.026.

[52] H. Elman, V. E. Howle, J. Shadid, R. Shuttleworth, and R. Tuminaro, *Block preconditioners based on approximate commutators*, SIAM Journal on Scientific Computing, 27 (2006), pp. 1651–1668, https://doi.org/10.1137/040608817.

[53] H. Elman, V. E. Howle, J. Shadid, D. Silvester, and R. Tuminaro, *Least squares preconditioners for stabilized discretizations of the Navier–Stokes equations*, SIAM Journal on Scientific Computing, 30 (2008), pp. 290–311, https://doi.org/10.1137/060655742.

[54] H. Elman and D. Silvester, *Fast nonsymmetric iterations and preconditioning for Navier–Stokes equations*, SIAM Journal on Scientific Computing, 17 (1996), pp. 33–46, https://doi.org/10.1137/0917004.

[55] H. Elman, D. Silvester, and A. Wathen, *Finite Elements and Fast Iterative Solvers: with Applications in Incompressible Fluid Dynamics*, Oxford University Press, Oxford, United Kingdom, 2014, https://doi.org/10.1093/acprof:oso/9780199678792.001.0001.

[56] H. C. Elman, *Preconditioning for the steady-state Navier–Stokes equations with low viscosity*, SIAM Journal on Scientific Computing, 20 (1999), pp. 1299–1316, https://doi.org/10.1137/S1064827596312547.

[57] H. C. Elman and R. S. Tuminaro, *Boundary conditions in approximate commutator preconditioners for the Navier-Stokes equations*, Electronic Transactions on Numerical Analysis, 35 (2009), pp. 257–280.

[58] C. Farhat, *A Lagrange multiplier based on divide and conquer finite element algorithm*, Journal of Computational and System Engineering, 2 (1991), pp. 149–156, https://doi.org/10.1016/0956-0521(91)90015-w.

[59] C. Farhat and F.-X. Roux, *A method of finite element tearing and interconnecting and its parallel solution algorithm*, International Journal for Numerical Methods in Engineering, 32 (1991), pp. 1205–1227, https://doi.org/10.1002/nme.1620320604.

[60] P. E. Farrell, L. Mitchell, and F. Wechsung, *An augmented Lagrangian preconditioner for the 3D stationary incompressible Navier-Stokes equations at high Reynolds number*, SIAM Journal on Scientific Computing, 41 (2019), pp. A3073–A3096, https://doi.org/10.1137/18M1219370.

[61] FEDDLib Team, *FEDDLib (Finite Element and Domain Decomposition Library)*. GitHub repository, 2026, https://github.com/FEDDLib/FEDDLib.

[62] C. A. Felippa and K. C. Park, *Staggered transient analysis procedures for coupled mechanical systems: Formulation*, Computer Methods in Applied Mechanics and Engineering, 24 (1980), pp. 61–111, https://doi.org/10.1016/0045-7825(80)90040-7.

[63] M. Á. Fernández and M. Moubachir, *A newton method using exact jacobians for solving fluid–structure coupling*, Computers & Structures, 83 (2005), pp. 127–142, https://doi.org/10.1016/j.compstruc.2004.08.020.

[64] L. Formaggia, A. Quarteroni, and A. Veneziani, eds., *Cardiovascular Mathematics: Modeling and Simulation of the Circulatory System*, vol. 1 of MS&A—Modeling, Simulation and Applications, Springer, Milan, 2009, https://doi.org/10.1007/978-88-470-1152-6.

[65] O. Frank, *Die Grundform des arteriellen Pulses. Erste Abhandlung: Mathematische Analyse*, Zeitschrift für Biologie, 37 (1899), pp. 483–526.

[66] O. Frank, *The basic shape of the arterial pulse. first treatise: Mathematical analysis*, Journal of Molecular and Cellular Cardiology, 22 (1990), pp. 255–277, https://doi.org/10.1016/0022-2828(90)91460-O. English translation of the original 1899 publication.

[67] J. Galvis and Y. Efendiev, *Domain decomposition preconditioners for multiscale flows in high contrast media: Reduced dimension coarse spaces*, Multiscale Modeling & Simulation, 8 (2010), pp. 1621–1644, https://doi.org/10.1137/100790112.

[68] M. W. Gee, U. Küttler, and W. A. Wall, *Truly monolithic algebraic multigrid for fluid–structure interaction*, International Journal for Numerical Methods in Engineering, 85 (2011), pp. 987–1016, https://doi.org/10.1002/nme.3001.

[69] M. W. Gee, C. M. Siefert, J. J. Hu, R. S. Tuminaro, and M. G. Sala, *ML 5.0 smoothed aggregation user's guide*, Tech. Report SAND2006-2649, Sandia National Laboratories, 2006.

[70] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, 4th ed., 2013, https://doi.org/10.1137/1.9781421407944.

[71] N. I. M. Gould, Y. Hu, and J. A. Scott, *A numerical evaluation of sparse direct solvers for the solution of large sparse symmetric linear systems of equations*, ACM Transactions on Mathematical Software, 33 (2007), pp. 10:1–10:32, https://doi.org/10.1145/1236463.1236465.

[72] C.-M. Hai and R. A. Murphy, *Cross-bridge phosphorylation and regulation of latch state in smooth muscle*, American Journal of Physiology—Cell Physiology, 254 (1988), pp. C99–C106, https://doi.org/10.1152/ajpcell.1988.254.1.C99.

[73] E. Hairer, S. P. Nørsett, and G. Wanner, *Solving Ordinary Differential Equations I: Nonstiff Problems*, vol. 8 of Springer Series in Computational Mathematics, Springer, Berlin and Heidelberg, 2 ed., 1993, https://doi.org/10.1007/978-3-540-78862-1.

[74] E. Hairer and G. Wanner, *Solving Ordinary Differential Equations II*, Springer, Berlin, Heidelberg, Germany, 1996, https://doi.org/10.1007/978-3-642-05221-7.

[75] A. Heinlein, *Parallel overlapping Schwarz preconditioners and multiscale discretizations with applications to fluid-structure interaction and highly heterogeneous problems*, PhD Thesis, Universität zu Köln, 2016.

[76] A. Heinlein, K. Ho, A. Klawonn, and M. Lanser, *Nonlinear two-level Schwarz methods: A parallel implementation in FROSch*, 2024, https://arxiv.org/abs/2410.24138.

[77] A. Heinlein, C. Hochmuth, and A. Klawonn, *Monolithic overlapping Schwarz domain decomposition methods with GDSW coarse spaces for incompressible fluid flow problems*, SIAM Journal on Scientific Computing, 41 (2019), pp. C291–C316, https://doi.org/10.1137/18M1184047.

[78] A. Heinlein, C. Hochmuth, and A. Klawonn, *Reduced dimension GDSW coarse spaces for monolithic Schwarz domain decomposition methods for incompressible fluid flow problems*, International Journal for Numerical Methods in Engineering, 121 (2019), pp. 1101–1119, https://doi.org/10.1002/nme.6258.

[79] A. Heinlein, C. Hochmuth, and A. Klawonn, *Fully algebraic two-level overlapping Schwarz preconditioners for elasticity problems*, in Numerical Mathematics and Advanced Applications ENUMATH 2019, vol. 139 of Lecture Notes in Computational Science and Engineering, Springer, 2021, pp. 531–539, https://doi.org/10.1007/978-3-030-55874-1_52.

[80] A. Heinlein, B. Kiefer, S. Prüger, O. Rheinbach, and F. Röver, *A comparison of direct solvers in FROSch applied to chemo-mechanics*, 2023, https://arxiv.org/abs/2310.12659.

[81] A. Heinlein, A. Klawonn, J. Knepper, O. Rheinbach, and O. B. Widlund, *Adaptive GDSW coarse spaces of reduced dimension for overlapping Schwarz methods*, SIAM Journal on Scientific Computing, 44 (2022), pp. A1176–A1204, https://doi.org/10.1137/20M1364540.

[82] A. Heinlein, A. Klawonn, J. Knepper, and L. Saßmannshausen, *Monolithic and block overlapping Schwarz preconditioners for the incompressible Navier-Stokes equations*, Electronic Transactions on Numerical Analysis, 65 (2026), pp. 156–206, https://doi.org/10.1553/etna_vol65s156.

[83] A. Heinlein, A. Klawonn, S. Rajamanickam, and O. Rheinbach, *FROSch: A fast and robust overlapping Schwarz domain decomposition preconditioner based on Xpetra in Trilinos*, in Domain Decomposition Methods in Science and Engineering XXV, vol. 138 of Lecture Notes in Computational Science and Engineering, Springer, 2020, pp. 176–184, https://doi.org/10.1007/978-3-030-56750-7_19.

[84] A. Heinlein, A. Klawonn, and O. Rheinbach, *A parallel implementation of a two-level overlapping Schwarz method with energy-minimizing coarse space based on Trilinos*, SIAM Journal on Scientific Computing, 38 (2016), pp. C713–C747, https://doi.org/10.1137/16M1062843.

[85] A. Heinlein, A. Klawonn, and O. Rheinbach, *Parallel two-level overlapping Schwarz methods in fluid-structure interaction*, in Numerical Mathematics and Advanced Applications ENUMATH 2015, B. Karasozen, M. Manguoglu, M. Tezer-Sezgin, S. Göktepe, and O. Ugur, eds., Cham, 2016, Springer, pp. 521–530, https://doi.org/10.1007/978-3-319-39929-4_50.

[86] A. Heinlein, A. Klawonn, and O. Rheinbach, *Parallel overlapping Schwarz with an energy-minimizing coarse space*, in Domain Decomposition Methods in Science and Engineering XXIII, C.-O. Lee, X.-C. Cai, D. E. Keyes, H. H. Kim, A. Klawonn, E.-J. Park, and O. B. Widlund, eds., Cham, 2017, Springer, pp. 353–360, https://doi.org/10.1007/978-3-319-52389-7_36.

[87] A. Heinlein, A. Klawonn, O. Rheinbach, and F. Röver, *A three-level extension for fast and robust overlapping Schwarz (FROSch) preconditioners with reduced dimensional coarse space*, in Domain Decomposition Methods in Science and Engineering XXVI, Lecture Notes in Computational Science and Engineering, Cham, 2023, Springer, pp. 505–513, https://doi.org/10.1007/978-3-030-95025-5_54.

[88] A. Heinlein, A. Klawonn, O. Rheinbach, and O. B. Widlund, *Improving the parallel performance of overlapping Schwarz methods by using a smaller energy minimizing coarse space*, in Domain Decomposition Methods in Science and Engineering XXIV, vol. 125 of Lecture Notes in Computational Science and Engineering, Cham, 2019, Springer, pp. 383–392, https://doi.org/10.1007/978-3-319-93873-8_36.

[89] A. Heinlein, M. Perego, and S. Rajamanickam, *FROSch preconditioners for land ice simulations of Greenland and Antarctica*, SIAM Journal on Scientific Computing, 44 (2022), pp. B339–B367, https://doi.org/10.1137/21M1395260.

[90] A. Heinlein, O. Rheinbach, and F. Röver, *Parallel scalability of three-level FROSch preconditioners to 220000 cores using the Theta supercomputer*, SIAM Journal on Scientific Computing, 45 (2023), pp. S173–S198, https://doi.org/10.1137/21M1431205. Special Section: 2021 Copper Mountain Conference.

[91] A. Heinlein, O. Rheinbach, F. Röver, S. Sandfeld, and D. Steinberger, *Applying the FROSch overlapping Schwarz preconditioner to dislocation mechanics in Deal.II*, Proceedings in Applied Mathematics and Mechanics, 19 (2019), p. e201900337, https://doi.org/10.1002/pamm.201900337.

[92] A. Heinlein, O. Rheinbach, and F. Röver, *A multilevel extension of the GDSW overlapping Schwarz preconditioner in two dimensions*, Computational Methods in Applied Mathematics, 23 (2023), pp. 953–968, https://doi.org/10.1515/cmam-2022-0168.

[93] A. Heinlein and K. Smetana, *A fully algebraic and robust two-level Schwarz method based on optimal local approximation spaces*, 2022, https://doi.org/10.48550/arXiv.2207.05559.

[94] M. A. Heroux, R. A. Bartlett, V. E. Howle, R. J. Hoekstra, J. J. Hu, T. G. Kolda, R. B. Lehoucq, K. R. Long, R. P. Pawlowski, E. T. Phipps, A. G. Salinger, H. K. Thornquist, R. S. Tuminaro, J. M. Willenbring, A. Williams, and K. S. Stanley, *An overview of the Trilinos project*, ACM Transactions on Mathematical Software, 31 (2005), pp. 397–423, https://doi.org/10.1145/1089014.1089021.

[95] K. Ho, A. Klawonn, and M. Lanser, *Two-level nonlinear Schwarz methods - a parallel implementation with application to nonlinear elasticity and incompressible flow problems*, 2026, https://arxiv.org/abs/2603.24542.

[96] C. Hochmuth, *Parallel Overlapping Schwarz Preconditioners for Incompressible Fluid Flow and Fluid-Structure Interaction Problems*, dissertation, University of Cologne, Cologne, Germany, 2020, https://nbn-resolving.de/urn:nbn:de:hbz:38-113450.

[97] G. A. Holzapfel, *Nonlinear solid mechanics: A continuum approach for engineering science*, Meccanica, 37 (2002), pp. 489–490, https://doi.org/10.1023/A:1020843529530. Book review.

[98] G. A. Holzapfel, C. A. J. Schulze-Bauer, and M. Stadler, *Mechanics of angioplasty: Wall, balloon and stent*, in Mechanics in Biology, American Society of Mechanical Engineers, 2000, pp. 141–156, https://doi.org/10.1115/IMECE2000-1927.

[99] V. E. Howle, J. Schroder, and R. Tuminaro, *The effect of boundary conditions within Pressure Convection–Diffusion preconditioners*, Tech. Report SAND2006-4466, Sandia National Laboratories, Albuquerque, NM, and Livermore, CA, July 2006.

[100] T. J. R. Hughes, *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*, Dover Publications, Mineola, NY, 2000.

[101] D. Jodlbauer, U. Langer, and T. Wick, *Parallel block-preconditioned monolithic solvers for fluid-structure interaction problems*, International Journal for Numerical Methods in Engineering, 117 (2019), pp. 623–643, https://doi.org/10.1002/nme.5970.

[102] R. P. Johnson, A. F. El-Yazbi, K. Takeya, E. J. Walsh, M. P. Walsh, and W. C. Cole, *Ca^{2+} sensitization via phosphorylation of myosin phosphatase targeting subunit at Threonine-855 by Rho kinase contributes to the arterial myogenic response*, The Journal of Physiology, 587 (2009), pp. 2537–2553, https://doi.org/10.1113/jphysiol.2008.168252.

[103] G. Karypis and V. Kumar, *A fast and high quality multilevel scheme for partitioning irregular graphs*, SIAM Journal on Scientific Computing, 20 (1998), pp. 359–392, https://doi.org/10.1137/S1064827595287997.

[104] D. Kay, D. Loghin, and A. Wathen, *A preconditioner for the steady-state Navier–Stokes equations*, SIAM Journal on Scientific Computing, 24 (2002), pp. 237–256, https://doi.org/10.1137/S106482759935808X.

[105] B. Kiefer, S. Prüger, O. Rheinbach, and F. Röver, *Monolithic parallel overlapping Schwarz methods in fully-coupled nonlinear chemo-mechanics problems*, Computational Mechanics, 71 (2023), pp. 765–788, https://doi.org/10.1007/s00466-022-02254-y.

[106] B. Kiefer, S. Prüger, O. Rheinbach, F. Röver, and S. Roth, *Variational settings and domain decomposition based solution schemes for a coupled deformation-diffusion problem*, Proceedings in Applied Mathematics and Mechanics, 21 (2021), p. e202100163, https://doi.org/10.1002/pamm.202100163.

[107] A. Klawonn, J. Knepper, and L. Saßmannshausen, *Preconditioning a fluid–structure interaction problem using monolithic and block domain decomposition methods for the fluid*, in Domain Decomposition Methods in Science and Engineering XXIX, P. F. Antonietti, M. Botti, S. C. Brenner, G. Ciaramella, H. H. Kim, U. Langer, and C. Vergara, eds., Lecture Notes in Computational Science and Engineering, Cham, 2026, Springer, https://arxiv.org/abs/2512.01887. To appear. Preprint: arXiv:2512.01887.

[108] A. Klawonn and M. Lanser, *Nonlinear monolithic two-level Schwarz methods for the Navier–Stokes equations*, in Domain Decomposition Methods in Science and Engineering XXVIII, P. E. Bjørstad, X.-C. Cai, V. Dolean, D. E. Keyes, R. Kornhuber, and J. Xu, eds., vol. 155 of Lecture Notes in Computational Science and Engineering, Cham, 2026, Springer, https://arxiv.org/abs/2409.03041. To appear. Preprint: arXiv:2409.03041.

[109] A. Klawonn and L. F. Pavarino, *Overlapping Schwarz methods for mixed linear elasticity and Stokes problems*, Computer Methods in Applied Mechanics and Engineering, 165 (1998), pp. 233–245, https://doi.org/10.1016/S0045-7825(98)00059-0.

[110] A. Klawonn and L. F. Pavarino, *A comparison of overlapping Schwarz methods and block preconditioners for saddle point problems*, Numerical Linear Algebra with Applications, 7 (2000), pp. 1–25, https://doi.org/10.1002/(SICI)1099-1506(200001/02)7:1<1::AID-NLA183>3.3.CO;2-A.

[111] A. Klawonn and O. Rheinbach, *A parallel implementation of dual-primal FETI methods for three-dimensional linear elasticity using a transformation of basis*, SIAM Journal on Scientific Computing, 28 (2006), pp. 1886–1906, https://doi.org/10.1137/050624364.

[112] A. Klawonn and O. Rheinbach, *Inexact FETI-DP methods*, International Journal for Numerical Methods in Engineering, 69 (2007), pp. 284–307, https://doi.org/10.1002/nme.1758.

[113] A. Klawonn and O. Rheinbach, *Highly scalable parallel domain decomposition methods with an application to biomechanics*, Zeitschrift für Angewandte Mathematik und Mechanik, 90 (2010), pp. 5–32, https://doi.org/10.1002/zamm.200900329.

[114] A. Klawonn and G. Starke, *Block triangular preconditioners for nonsymmetric saddle point problems: field-of-values analysis*, Numerische Mathematik, 81 (1999), pp. 577–594, https://doi.org/10.1007/s002110050405.

[115] A. Klawonn, O. B. Widlund, and M. Dryja, *Dual-primal FETI methods for three-dimensional elliptic problems with heterogeneous coefficients*, SIAM Journal on Numerical Analysis, 40 (2002), pp. 159–179, https://doi.org/10.1137/S0036142901388081.

[116] J. Knepper, *Adaptive Coarse Spaces for the Overlapping Schwarz Method and Multiscale Elliptic Problems*, dissertation, University of Cologne, Cologne, Germany, 2022, https://nbn-resolving.de/urn:nbn:de:hbz:38-620024.

[117] F. Kong and X.-C. Cai, *Scalability study of an implicit solver for coupled fluid–structure interaction problems on unstructured meshes in 3D*, The International Journal of High Performance Computing Applications, 32 (2018), pp. 207–219, https://doi.org/10.1177/1094342016646437.

[118] F. Kong, V. Kheyfets, E. Finol, and X.-C. Cai, *An efficient parallel simulation of unsteady blood flows in patient-specific pulmonary artery*, International Journal for Numerical Methods in Biomedical Engineering, 34 (2017), p. e2952, https://doi.org/10.1002/cnm.2952.

[119] F. Kong, V. Kheyfets, E. Finol, and X.-C. Cai, *Simulation of unsteady blood flows in a patient-specific compliant pulmonary artery with a highly parallel monolithically coupled fluid-structure interaction algorithm*, International Journal for Numerical Methods in Biomedical Engineering, 35 (2019), p. e3208, https://doi.org/10.1002/cnm.3208.

[120] J. Korelc and P. Wriggers, *Automation of Finite Element Methods*, Springer, Cham, Switzerland, 2016, https://doi.org/10.1007/978-3-319-39005-5.

[121] S. Köhler and O. Rheinbach, *Four-level overlapping Schwarz as multigrid coarse solver for incompressible non-Newtonian flow in complex geometries*, 2026, https://arxiv.org/abs/2606.01433.

[122] S. Köhler and O. Rheinbach, *Monolithic multi-level overlapping Schwarz solvers for fluid problems*, 2026, https://arxiv.org/abs/2508.04356.

[123] U. Langer and H. Yang, *Robust and efficient monolithic fluid–structure-interaction solvers*, International Journal for Numerical Methods in Engineering, 108 (2016), pp. 303–325, https://doi.org/10.1002/nme.5214.

[124] U. Langer and H. Yang, *Numerical simulation of fluid–structure interaction problems with hyperelastic models: A monolithic approach*, Mathematics and Computers in Simulation, 145 (2018), pp. 186–208, https://doi.org/10.1016/j.matcom.2016.07.008. Special issue associated with the 5th IMACS Conference on Mathematical Modelling and Computational Methods in Applied Sciences and Engineering.

[125] L. Lauder, S. Ewen, A. R. Tzafriri, E. R. Edelman, T. F. Lüscher, P. J. Blankestijn, O. Dörr, M. P. Schlaich, F. Sharif, M. Voskuil, T. Zeller, C. Ukena, B. Scheller, M. Böhm, and F. Mahfoud, *Renal artery anatomy assessed by quantitative analysis of selective renal angiography in 1,000 patients with hypertension*, EuroIntervention, 14 (2018), pp. 121–128, https://doi.org/10.4244/EIJ-D-18-00112.

[126] Lecturio, *Skelettmuskulatur: Muskelphysiologie*, 2023, https://www.lecturio.de/artikel/medizin/muskelphysiologie-der-skelettmuskulatur/. Zugriff am 03.06.2026.

[127] Z.-Z. Liang and G.-F. Zhang, *SIMPLE-like preconditioners for saddle point problems from the steady Navier–Stokes equations*, Journal of Computational and Applied Mathematics, 302 (2016), pp. 211–223, https://doi.org/10.1016/j.cam.2016.02.012.

[128] J. W. Liu, *The role of elimination trees in sparse factorization*, SIAM Journal on Matrix Analysis and Applications, 11 (1990), pp. 134–172, https://doi.org/10.1137/0611010.

[129] J. Mandel and C. R. Dohrmann, *Convergence of a balancing domain decomposition by constraints and energy minimization*, Numerical Linear Algebra with Applications, 10 (2003), pp. 639–659, https://doi.org/10.1002/nla.341.

[130] J. Mandel and R. Tezaur, *On the convergence of a dual-primal substructuring method*, Numerische Mathematik, 88 (2001), pp. 543–558, https://doi.org/10.1007/s211-001-8014-1.

[131] J. A. Meijerink and H. A. van der Vorst, *An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix*, Mathematics of Computation, 31 (1977), pp. 148–162, https://doi.org/10.1090/S0025-5718-1977-0438681-4.

[132] S. C. Murtada, A. Arner, and G. A. Holzapfel, *Experiments and mechanochemical modeling of smooth muscle contraction: Significance of filament overlap*, Journal of Theoretical Biology, 297 (2012), pp. 176–186, https://doi.org/10.1016/j.jtbi.2011.11.012.

[133] S.-I. Murtada, M. Kroon, and G. A. Holzapfel, *A calcium-driven mechanochemical model for prediction of force generation in smooth muscle*, Biomechanics and Modeling in Mechanobiology, 9 (2010), pp. 749–762, https://doi.org/10.1007/s10237-010-0211-0.

[134] National Institute of Science and Technology in Medicine Assisted by Scientific Computing (Brazil), *HeMoLab*, 2014, http://hemolab.lncc.br/adan-web. Accessed November 2014.

[135] N. M. Newmark, *A method of computation for structural dynamics*, Transactions of the American Society of Civil Engineers, 127 (1962), pp. 1406–1433, https://doi.org/10.1061/TACEAT.0008448.

[136] F. Nobile and C. Vergara, *An effective fluid-structure interaction formulation for vascular dynamics by generalized Robin conditions*, SIAM Journal on Scientific Computing, 30 (2008), pp. 731–763, https://doi.org/10.1137/060678439.

[137] D. Nordsletten, M. McCormick, P. J. Kilner, P. Hunter, D. Kay, and N. P. Smith, *Fluid-solid coupling for the investigation of diastolic and systolic human left ventricular function*, International Journal for Numerical Methods in Biomedical Engineering, 27 (2011), pp. 1017–1039, https://doi.org/10.1002/cnm.1405.

[138] S. Nurani Ramesh, K. Uhlmann, L. Saßmannshausen, O. Rheinbach, A. Klawonn, A. Heinlein, and D. Balzani, *First steps towards modeling the interaction of cardiovascular agents and smooth muscle activation in arterial walls*, Proceedings in Applied Mathematics and Mechanics, 22 (2023), p. e202200133, https://doi.org/10.1002/pamm.202200133.

[139] S. Patankar and D. Spalding, *A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows*, International Journal of Heat and Mass Transfer, 15 (1972), pp. 1787–1806, https://doi.org/10.1016/0017-9310(72)90054-3.

[140] D. M. Pedroso, *Caveats of three direct linear solvers for finite element analyses*, International Journal for Numerical Methods in Engineering, 125 (2024), p. e7545, https://doi.org/10.1002/nme.7545.

[141] X. Peng, Q. Ye, J. Xiao, H. Shen, X. Long, and Y. Pei, *Relationship between blood flow velocity and blood suppression inversion time for optimal visualization of renal artery in spatial labeling with multiple inversion pulses: In vitro study*, Iranian Journal of Radiology, 15 (2018), p. e65502, https://doi.org/10.5812/iranjradiol.65502.

[142] M. Pernice and M. D. Tocci, *A multigrid-preconditioned Newton–Krylov method for the incompressible Navier–Stokes equations*, SIAM Journal on Scientific Computing, 23 (2001), pp. 398–418, https://doi.org/10.1137/S1064827500372250.

[143] M. Pernice and H. F. Walker, *NITSOL: A Newton iterative solver for nonlinear systems*, SIAM Journal on Scientific Computing, 19 (1998), pp. 302–318, https://doi.org/10.1137/S1064827596303843.

[144] J. Pina, A. Costa, and J. Appleton, *Formal solution of quasi-static problems*, International Journal of Non-Linear Mechanics, 45 (2010), pp. 525–534, https://doi.org/10.1016/j.ijnonlinmec.2010.02.002.

[145] A. Prokopenko and R. S. Tuminaro, *An algebraic multigrid method for Q_2–Q_1 mixed discretizations of the Navier–Stokes equations*, Numerical Linear Algebra with Applications, 24 (2017), p. e2109, https://doi.org/10.1002/nla.2109.

[146] F. Qi, Y. Liu, R. Chen, and X. Cai, *Computational hemodynamic analysis of renal blood flow and the impact of outflow boundary conditions*, Biomechanics and Modeling in Mechanobiology, 25 (2026), https://doi.org/10.1007/s10237-026-02065-7.

[147] A. Quarteroni, F. Saleri, and A. Veneziani, *Factorization methods for the numerical approximation of Navier–Stokes equations*, Computer Methods in Applied Mechanics and Engineering, 188 (2000), pp. 505–526, https://doi.org/10.1016/S0045-7825(99)00192-9.

[148] S. Rajamanickam, S. Acer, L. Berger-Vergiat, V. Dang, N. Ellingwood, E. Harvey, B. Kelley, C. R. Trott, J. Wilke, and I. Yamazaki, *Kokkos Kernels: Performance portable sparse/dense linear algebra and graph kernels*. arXiv, 2021, https://doi.org/10.48550/arXiv.2103.11991.

[149] S. N. Ramesh, L. Saßmannshausen, A. Klawonn, and D. Balzani, *Pharmaco-mechanical computational simulation of atherosclerotic arteries: effects of calcium channel blockers on local stress distributions*, 2026, https://arxiv.org/abs/2611.21320.

[150] M. Razzaq, H. Damanik, J. Hron, A. Ouazzi, and S. Turek, *FEM multigrid techniques for fluid–structure interaction with application to hemodynamics*, Applied Numerical Mathematics, 62 (2012), pp. 1156–1170, https://doi.org/10.1016/j.apnum.2010.12.010.

[151] T. Richter, *A monolithic geometric multigrid solver for fluid-structure interactions in ALE formulation*, International Journal for Numerical Methods in Engineering, 104 (2015), pp. 372–390, https://doi.org/10.1002/nme.4943.

[152] T. Richter, *Fluid-Structure Interactions: Models, Analysis and Finite Elements*, vol. 118 of Lecture Notes in Computational Science and Engineering, Springer, Cham, Switzerland, 2017, https://doi.org/10.1007/978-3-319-63970-3.

[153] F. Röver, *Multi-level Extensions for the Fast and Robust Overlapping Schwarz Preconditioners*, PhD thesis, TU Bergakademie Freiberg, Freiberg, Germany, 2022, https://nbn-resolving.org/urn:nbn:de:bsz:105-qucosa2-854707.

[154] Y. Saad, *Iterative Methods for Sparse Linear Systems*, Society for Industrial and Applied Mathematics, second ed., 2003, https://doi.org/10.1137/1.9780898718003.

[155] Y. Saad and M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM Journal on Scientific and Statistical Computing, 7 (1986), pp. 856–869, https://doi.org/10.1137/0907058.

[156] Sandia National Laboratories, *Ifpack2 user's guide 1.0*, Tech. Report SAND2016-5338, Sandia National Laboratories, 2016, https://www.osti.gov/servlets/purl/1259544. The definitive user manual for the Ifpack2 package in Trilinos.

[157] A. Segal, M. Ur Rehman, and C. Vuik, *Preconditioners for incompressible Navier–Stokes solvers*, Numerical Mathematics: Theory, Methods and Applications, 3 (2010), pp. 245–275, https://doi.org/10.4208/nmtma.2010.33.1.

[158] D. Silvester, H. Elman, D. Kay, and A. Wathen, *Efficient preconditioning of the linearized Navier–Stokes equations for incompressible flow*, Journal of Computational and Applied Mathematics, 128 (2001), pp. 261–279, https://doi.org/10.1016/S0377-0427(00)00515-X.

[159] B. F. Smith, P. E. Bjørstad, and W. D. Gropp, *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*, Cambridge University Press, Cambridge, 1996.

[160] A. P. Somlyo and A. V. Somlyo, *Ca^{2+} sensitivity of smooth muscle and nonmuscle myosin II: Modulated by G proteins, kinases, and myosin phosphatase*, Physiological Reviews, 83 (2003), pp. 1325–1358, https://doi.org/10.1152/physrev.00023.2003.

[161] N. Spillane, V. Dolean, P. Hauret, F. Nataf, C. Pechstein, and R. Scheichl, *Abstract robust coarse spaces for systems of PDEs via generalized eigenproblems in the overlaps*, Numerische Mathematik, 126 (2014), pp. 741–770, https://doi.org/10.1007/s00211-013-0576-y.

[162] R. L. Taylor, *FEAP: Finite element analysis program*, 2014, http://projects.ce.berkeley.edu/feap/.

[163] A. Toselli and O. Widlund, *Domain Decomposition Methods—Algorithms and Theory*, vol. 34 of Springer Series in Computational Mathematics, Springer, Berlin, 2005, https://doi.org/10.1007/b137868.

[164] Trilinos Project Team, *Trilinos*. GitHub repository, 2026, https://github.com/trilinos/Trilinos.

[165] C. Trott, L. Berger-Vergiat, D. Poliakoff, S. Rajamanickam, D. Lebrun-Grandie, J. Madsen, N. Al Awar, M. Gligoric, G. Shipman, and G. Womeldorff, *The Kokkos ecosystem: Comprehensive performance portability for high performance computing*, Computing in Science & Engineering, 23 (2021), pp. 10–18, https://doi.org/10.1109/MCSE.2021.3098509.

[166] U. Trottenberg, C. W. Oosterlee, and A. Schüller, *Multigrid*, Academic Press, San Diego, CA, 2001.

[167] S. Turek, J. Hron, M. Mádlík, M. Razzaq, H. Wobker, and J. F. Acker, *Numerical simulation and benchmarking of a monolithic multigrid solver for fluid-structure interaction problems with application to hemodynamics*, in Fluid Structure Interaction II, Springer, 2010, pp. 193–220, https://doi.org/10.1007/978-3-642-14206-2_8.

[168] K. Uhlmann and D. Balzani, *Chemo-mechanical modeling of smooth muscle cell activation for the simulation of arterial walls under changing blood pressure*, Biomechanics and Modeling in Mechanobiology, 22 (2023), pp. 1049–1065, https://doi.org/10.1007/s10237-023-01700-x.

[169] K. Uhlmann, A. Zahn, and D. Balzani, *Simulation of arterial walls: Growth, fiber reorientation, and active response*, in Solid (Bio)mechanics: Challenges of the Next Decade, G. Sommer, K. Li, D. C. Haspinger, and R. W. Ogden, eds., vol. 24 of Studies in Mechanobiology, Tissue Engineering and Biomaterials, Springer, Cham, 2022, pp. 181–209, https://doi.org/10.1007/978-3-030-92339-6_8.

[170] P. Vaněk, J. Mandel, and M. Brezina, *Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems*, Computing, 56 (1996), pp. 179–196, https://doi.org/10.1007/BF02238511.

[171] I. E. Vignon-Clementel, C. Alberto Figueroa, K. E. Jansen, and C. A. Taylor, *Outflow boundary conditions for three-dimensional finite element modeling of blood flow and pressure in arteries*, Computer Methods in Applied Mechanics and Engineering, 195 (2006), pp. 3776–3796, https://doi.org/10.1016/j.cma.2005.04.014. Absorbing Boundary Conditions.

[172] N. Westerhof, J.-W. Lankhaar, and B. E. Westerhof, *The arterial windkessel*, Medical & Biological Engineering & Computing, 47 (2009), pp. 131–141, https://doi.org/10.1007/s11517-008-0359-2.

[173] T. Wick, *Solving monolithic fluid-structure interaction problems in Arbitrary Lagrangian-Eulerian coordinates with the deal.II library*, Archive of Numerical Software, 1 (2013), pp. 1–19.

[174] T. Wiesner, *Flexible Aggregation-based Algebraic Multigrid Methods for Contact and Flow Problems*, dissertation, Technical University of Munich, 2015, https://mediatum.ub.tum.de/1229321.

[175] P. T. Williams and A. J. Baker, *Numerical simulations of laminar flow over a 3D backward-facing step*, International Journal for Numerical Methods in Fluids, 24 (1997), pp. 1159–1183, https://doi.org/10.1002/(SICI)1097-0363(19970615)24:11<1159::AID-FLD534>3.0.CO;2-R.

[176] Wolfram Research, *AceFEM*, 2025. Finite element method environment for Mathematica.

[177] Wolfram Research, *AceFEM manual*. https://www.wolfram.com/products/applications/acefem/AceFEMManual.pdf, n.d. Technical documentation for the AceFEM finite element environment.

[178] Wolfram Research, *AceGen user's manual*. https://www.wolfram.com/products/applications/acegen/manual.pdf, n.d. Automatic code generation system for continuum mechanics.

[179] P. Wriggers, *Nonlinear Finite Element Methods*, Springer, Berlin and Heidelberg, 2008, https://doi.org/10.1007/978-3-540-71001-1.

[180] Y. Wu and X.-C. Cai, *A parallel two-level method for simulating blood flows in branching arteries with the resistive boundary condition*, Computers & Fluids, 45 (2011), pp. 92–102, https://doi.org/10.1016/j.compfluid.2010.11.015. 22nd International Conference on Parallel Computational Fluid Dynamics (ParCFD 2010).

[181] Y. Wu and X.-C. Cai, *A fully implicit domain decomposition based ALE framework for three-dimensional fluid–structure interaction with application in blood flow computation*, Journal of Computational Physics, 258 (2014), pp. 524–537, https://doi.org/10.1016/j.jcp.2013.10.046.

[182] T. Yamamoto, Y. Ogasawara, A. Kimura, H. Tanaka, O. Hiramatsu, K. Tsujioka, M. J. Lever, K. H. Parker, C. J. Jones, C. G. Caro, and F. Kajiya, *Blood velocity profiles in the human renal artery by Doppler ultrasound and their relationship to atherosclerosis*, Arteriosclerosis, Thrombosis, and Vascular Biology, 16 (1996), pp. 172–177, https://doi.org/10.1161/01.ATV.16.1.172.

[183] I. Yamazaki, A. Heinlein, and S. Rajamanickam, *An experimental study of two-level Schwarz domain-decomposition preconditioners on GPUs*, in 2023 IEEE International Parallel and Distributed Processing Symposium (IPDPS), IEEE Computer Society, 2023, pp. 680–689, https://doi.org/10.1109/IPDPS54959.2023.00073.

[184] O. C. Zienkiewicz and A. H. C. Chan, *Coupled problems and their numerical solution*, in Advances in Computational Nonlinear Mechanics, Springer, 1989, pp. 139–176, https://doi.org/10.1007/978-3-7091-2828-2_4.

Eidesstattliche Erklärung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Dissertation selbstständig und ohne die Benutzung anderer als der angegebenen Hilfsmittel und Literatur angefertigt habe. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten und nicht veröffentlichten Werken dem Wortlaut oder dem Sinn nach entnommen wurden, sind als solche kenntlich gemacht. Ich versichere an Eides statt, dass diese Dissertation noch keiner anderen Fakultät oder Universität zur Prüfung vorgelegen hat; dass sie - abgesehen von unten angegebenen Teilpublikationen und eingebundenen Artikeln und Manuskripten - noch nicht veröffentlicht worden ist sowie, dass ich eine Veröffentlichung der Dissertation vor Abschluss des Prüfungsverfahrens der Promotion nicht vornehmen werde. Die Bestimmungen der Promotionsordnung der Mathematisch-Naturwissenschaftlichen Fakultät der Universität zu Köln sind mir bekannt. Darüber hinaus erkläre ich hiermit, dass ich die Ordnung zur Sicherung guter wissenschaftlicher Praxis und zum Umgang mit wissenschaftlichem Fehlverhalten der Universität zu Köln gelesen und sie bei der Durchführung der der Dissertation zugrundeliegenden Arbeiten und der schriftlich oder elektronisch verfassten Dissertation beachtet habe, und verpflichte mich hiermit, die dort genannten Vorgaben bei allen wissenschaftlichen Tätigkeiten zu beachten und umzusetzen.

Teilpublikationen

- Sharan Nurani Ramesh, Lea Saßmannshausen, Axel Klawonn, and Daniel Balzani, Pharmaco–mechanical computational simulation of atherosclerotic arteries: effects of calcium channel blockers on local stress distributions, 2026, https://arxiv.org/abs/2611.21320
- Alexander Heinlein, Axel Klawonn, Jascha Knepper, and Lea Saßmannshausen. *Monolithic and block overlapping Schwarz preconditioners for the incompressible Navier-Stokes equations*. Electron. Trans. Numer. Anal., 65:156–206, 2026
- Axel Klawonn, Jascha Knepper, and Lea Saßmannshausen. *Preconditioning a fluid–structure interaction problem using monolithic and block domain decomposition methods for the fluid*. accepted for publication in the proceedings of the DD29 2025 conference, Springer, May 2025
- Daniel Balzani, Alexander Heinlein, Axel Klawonn, Jascha Knepper, Sharan Nurani Ramesh, Oliver Rheinbach, Lea Saßmannshausen, and Klemens Uhlmann. *A computational framework for pharmaco-mechanical interactions in arterial walls using parallel monolithic domain decomposition methods*. GAMM-Mitteilungen, 47(1):e202370002, 2024
- Sharan Nurani Ramesh, Klemens Uhlmann, Lea Saßmannshausen, Oliver Rheinbach, Axel Klawonn, Alexander Heinlein, and Daniel Balzani, First steps towards modeling the interaction of cardiovascular agents and smooth muscle activation in arterial walls, PAMM, 22, p. e202200133, 2023

Ort, Datum	Unterschrift