Efficient numerical and algorithmic realization of a time-simultaneous Pressure-Schur-Complement solver for the incompressible Navier-Stokes equations in FEAT3

Dissertation
zur Erlangung des akademischen Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

Der Fakultät für Mathematik der
Technischen Universität Dortmund
vorgelegt von

Mirco Arndt

im September 2024

Dissertation:

Efficient numerical and algorithmic realization of a time-simultaneous Pressure-Schur-Complement solver for the incompressible Navier-Stokes equations in FEAT3

Fakultät für Mathematik
Technische Universität Dortmund

Erstgutachter: Prof. Dr. Stefan Turek
Zweitgutachter: PD. Dr. Andriy Sokolov

Tag der mündlichen Prüfung: 13.03.2025

ABSTRACT

The aim of this thesis is to develop, implement and validate a fast time-simultaneous solver for computational fluid dynamic (CFD) problems, which is optimized on high-performance computing (HPC) architectures to efficiently exploit numerous cores through a parallelization in space and time. The sequential-in-time and parallel-in-space projection method approach, which decouples velocity and pressure, serves as the basis for the development of the new solver approach. The Crank-Nicolson scheme is used as time stepping scheme. Since the structure and settings of the new solver approach are not clear, especially with respect to a convective term such as in the Navier-Stokes equations, fundamental investigations must first be carried out.

In order to explore the necessary requirements and characteristics of the new solver approach, the first step is to look at the theory for general convergence criteria. From the theory it was determined that a divergence-free velocity is a necessary condition for convergence. Two different divergence-free approaches have been implemented and tested for the new time-simultaneous solver: The augmented Lagrangian method and a newly developed update similar to the projection method update, which only approximately guarantees a divergence-free velocity. Both approaches lead to a reduced number of iterations until convergence, but in terms of computational time, the approximate divergence-free update is significantly more efficient and leads to a considerable reduction in computational time. With the newly developed approximate update approach, thousands of time steps of the Navier-Stokes equations can be computed parallel-in-space and -time.

In the time-simultaneous case, the choice of the right-hand side for the pressure Poisson problem is not clear and has a significant importance. Numerical results verify that a divergence-free velocity component from the previous time step with a non-divergence-free component from the current time step ensures the most optimal convergence behavior.

By ensuring a approximate divergence-free velocity, even in the case of the incompressible Navier-Stokes equations, it is possible to compute large time intervals time-simultaneously. A multigrid in time approach leads to a faster achievement of a divergence-free velocity for large time-intervals, since the approximate divergence-free velocity is not lost as quickly with larger time steps. It has been investigated that the multigrid in time approach only achieves a speed up in combination with a multigrid in space solver.

To compute a large number of time steps, especially in 3D, the memory usage must be taken into account. An embedding of the solver in a memory-optimized local recoupling approach using FGMRES has been investigated, which can further enforce convergence. A local recoupling with FGMRES is not recommended because the performance in terms of the computational time is insufficient. To enforce convergence, a global recoupling seems to be the better approach. Although it ensures fewer iterations and can speed up the computation, it uses too much memory. Embedding the solver in a recoupling approach is not necessary and can be neglected, as it also requires additional programming effort, such as new data types.

Another question was whether a fast existing solver for the pressure Poisson problem using prehandling could be integrated in the solver. The fast solver needs the Q_2/Q_1 finite element with the true Laplacian matrix. The theory already provides that the Q_2/Q_1 finite element is more prone to problems than the $Q_2/P_1^{\mathtt{disc}}$ finite element. Therefore, it is not recommended to integrate the solver, as in this case a recoupling approach using GMRES is necessary to achieve convergence.

In conclusion, the newly developed solver is decoupled from velocity and pressure, as well as from time and space. However, to achieve convergence, a divergence-free velocity must be ensured at least approximately at each time point. A time-simultaneous solver is much faster than a sequential-in-time solver, which can only compute parallel-in-space. To have direct access to the different cores on a computer, an implementation in C++ (FEAT3) is recommended.

Key words: computation fluid dynamics (CFD), high performance computing (HPC), finite element method (FEM), time simultaneous, time parallel, multigrid in space, multigrid in time, augmented Lagrangian, approximate divergence-free update, GMRES, FGMRES, projection method, PP, TSPP, Stokes equations, incompressible Navier-Stokes equations, Crank-Nicolson (CN).

Dedicated to

my wifey
Abida,

my grandparents
Christa & Lothar

&

in memory of my parents
Petra & Manfred.

ACKNOWLEDGMENT

First and foremost, I am extremely grateful to my supervisor, Prof. Dr. Stefan Turek, for giving me the opportunity to be one of his PhD students. Thank you for the invaluable advice and continuous support during my stay at LSIII. You inspired me with your passion and enthusiasm for fast and efficient solvers for fluid simulations. I am fortunate to have had the opportunity to learn from your expertise and to expand my knowledge under your mentorship. Many thanks for your guidance, unwavering support, constructive feedback, patience and immense encouragement throughout my PhD journey.

I am truly grateful to PD Dr. Andriy Sokolov, for his role as a reviewer of my thesis. The support, valuable feedback and insightful discussion throughout the process are genuinely appreciated.

Heartfelt thanks goes to Dirk Ribbrock and Peter Zajac for their constant support with C++ (FEAT3), other technical and numerical challenges. Thank you for always being available to discuss my questions and concerns.

Special thanks are extended to Dr. Christoph Lohmann for his assistance in identifying a hard-to-find coding error during the initial phase of my PhD.

Sincere gratitude goes to Lydia Wambach, Alexander Westermann and my wifey Abida for the proofreading, insightful comments and suggestions for improving in my thesis. Thank you all for your cooperation and time.

I would also like to express my gratitude to my office mate, Falko Ruppenthal, for fostering a friendly atmosphere and engaging in fruitful discussions. My thanks also goes to Dr. Malte Schuh for his moral support. I deeply appreciate the knowledge and experience I gained during my time at LSIII, TU Dortmund University.

A special thanks to all my beloved family and friends, especially my late mother, Petra, whose lifelong sacrifices, countless efforts and inspiration have shaped me into who I am today. I am also immensely grateful to my grandparents, Christa and Lothar, and my loving wifey, Abida, for their prayers, patience and understanding during these challenging years. Without their support and encouragement, I could not have reached this milestone in my life.

Dortmund, 27.09.2024 Mirco Arndt

Contents

Chapter 1

Introduction

Contents

Fluids are an essential part of our lives. Flow of fluids, such as liquids, gases and plasmas can be found in a wide range of applications. The physical behavior of fluids is described in fluid mechanics. A typical fluid mechanics problem involves basic fluid properties including flow velocity, pressure and density as a function of time and space. Fluid mechanics problems occur in industries in various ways, such as

- heating, ventilation and air conditioning technology,
- simulation of the flow over a vehicle,
- interaction of propellers or rotors with the aircraft fuselage,
- determination of the mass flow rate of crude oil in pipelines.

For most complex engineering problems, it is not possible to solve the equations of fluid mechanics analytically. However, it is possible to obtain approximate solutions on the computer, which is also known as computational fluid dynamics (CFD). CFD is attractive to industry because it is less expensive than physical testing. Anyhow, it should be noted that complex flow simulations are challenging as well as requiring a lot of computational time. It is therefore advantageous to develop fast solvers.

High-performance computing (HPC) is increasingly important as a tool for rapid computation, modeling, and simulation of advanced CFD problems. Modern HPC architectures provide great possibilities for parallel computation. Current trends suggest that the number of processor counts will increase even further. Solvers for CFD problems are studied for many years, which solve in parallel-in-space. Unfortunately, these solvers cannot provide full HPC performance. Therefore, they are also many space-time parallel solvers, but the most widely recognized solvers follow the parareal approach from Lions et al., mentioned in [48]. In this work, a fast parallel-in-space solver is analyzed and taken as a framework for a parallel space-time solver. Accordingly, the core of the new approach is not based on parareal in order to obtain better exploitation of the HPC architecture.

Benchmarks (flow-around-a-cylinder)

In this work, the example of flow in pipes is used as the central example model for fluid mechanics.
The earliest numerical solution for the flow past cylinders at low speed published in 1933 and can be found in [76]. Figure 1.1 shows the flow behavior around a cylinder at a faster speed. A Reynolds number of 100 results in vortices behind the cylinder. From these flow behaviors, benchmarks for the flow-around-a-cylinder can be developed to obtain validated numerical solutions.

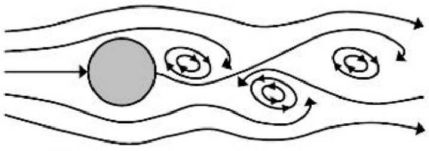


Figure 1.1: Flow behavior around a cylinder at Reynolds number 100.

1.1 General overview

The well known flow-around-a-cylinder benchmark, which is described in [68], can be used as a representative benchmark. The development of it was done in 1995 as part of the «Flow simulation on high-performance computers» project, which was sponsored by the German Research Association (DFG). The benchmark simulates the time-periodic behavior of a fluid in a pipe with a circular obstacle, shown in figure 1.2. The simulation is performed over a fixed time interval $[0, 8]$, with zero as the initial condition for the velocity at start time 0. Up to time 4, the inflow velocity of the fluid is increased until Reynolds number 100 is reached. After that, the inflow velocity is reduced again to zero.

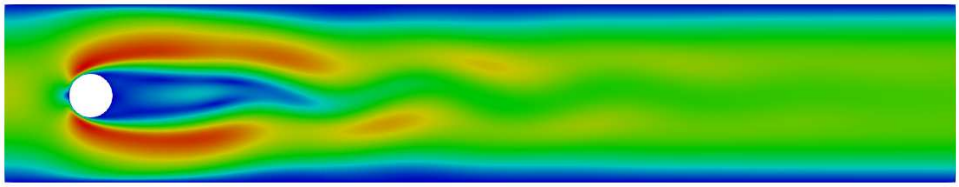


Figure 1.2: Velocity profile of the flow-around-a-cylinder for Reynolds number 100. The velocity flows into the pipe on the left side as inflow and the right side is the outflow.

Analysis of a fast parallel-in-space and sequential-in-time solver

The incompressible Navier-Stokes equations are solved in the flow-around-a-cylinder benchmark using a fast parallel-in-space solver, a so-called PP-solver that uses the projection method. The **P**rojection solution by **P**rojection solver (PP-solver), explained in section 3.1.2 and in more detail in section 3.2.2, can quickly solve the problem parallel-in-space but sequential-in-time. Sequential-in-time means one time step after the other. The projection method solves the system, consisting of pressure and velocity, decoupled from each other, which yields to an approximate solution.

The drag and lift values describe the forces, which occur during the benchmark. The force in the direction of the flow velocity is described by drag and the component of the force normal to the velocity represented by lift. Figure 1.4, figure 1.5 and figure 1.6 show the drag and lift values to validate the numerical solution. The reference solution is plotted in black.[1] It can be seen that especially for a small time step size the reference solution could be reproduced. However, for large time step sizes such as $k = 0.01$, the projection method provides a solution that is not a good approximation. This can be seen particularly well in figure 1.6 for the lift curve. Moreover, it can be seen in figure 1.3 that the divergence-free velocity, i.e., the incompressibility of the fluid, is preserved.

[1] Reference solution available as download at
https://wwwold.mathematik.tu-dortmund.de/~featflow/media/dfg_bench3_2d/draglift_q2_cn_lv1-6_dt4.zip

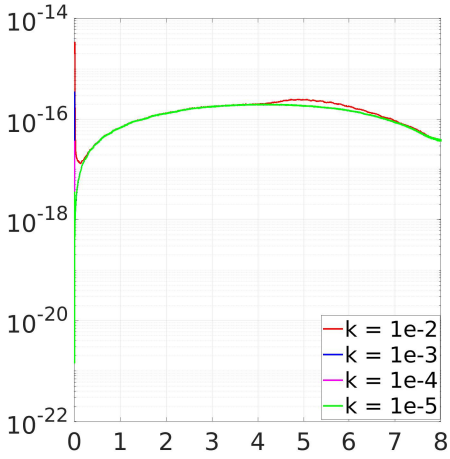


Figure 1.3: Divergence-defect of the velocity plotted over the entire time interval $[0, 8]$. k denotes the time step size of the solver.

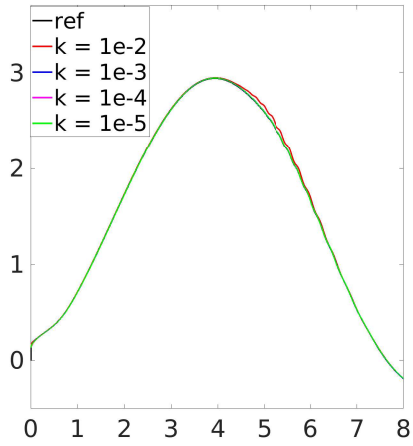
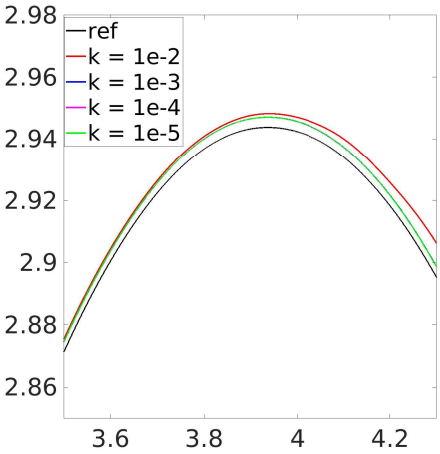


Figure 1.4: Drag curve plotted over the entire time interval $[0, 8]$. The reference solution is shown in black and k denotes the time step size of the solver.



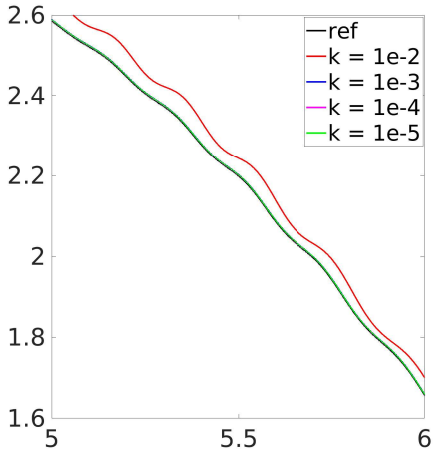
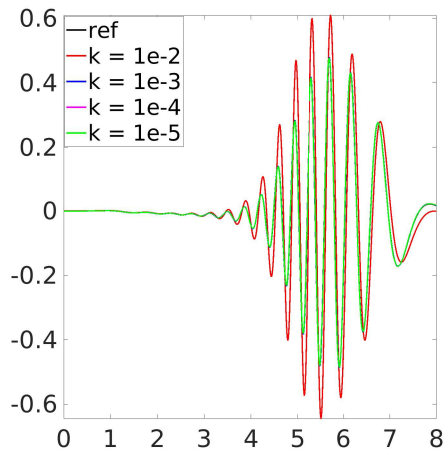


Figure 1.5: Drag curve zoomed in the time interval $[3.5, 4.3]$ (left) and in the time interval $[5, 6]$ (right). The reference solution is shown in black and k denotes the time step size of the solver.



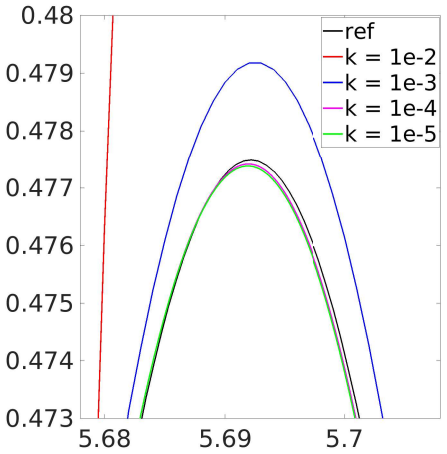


Figure 1.6: Lift curve plotted over the entire time interval $[0, 8]$ (left). Zoom in the time interval $[5.68, 5.71]$ (right). The reference solution is shown in black and k denotes the time step size.

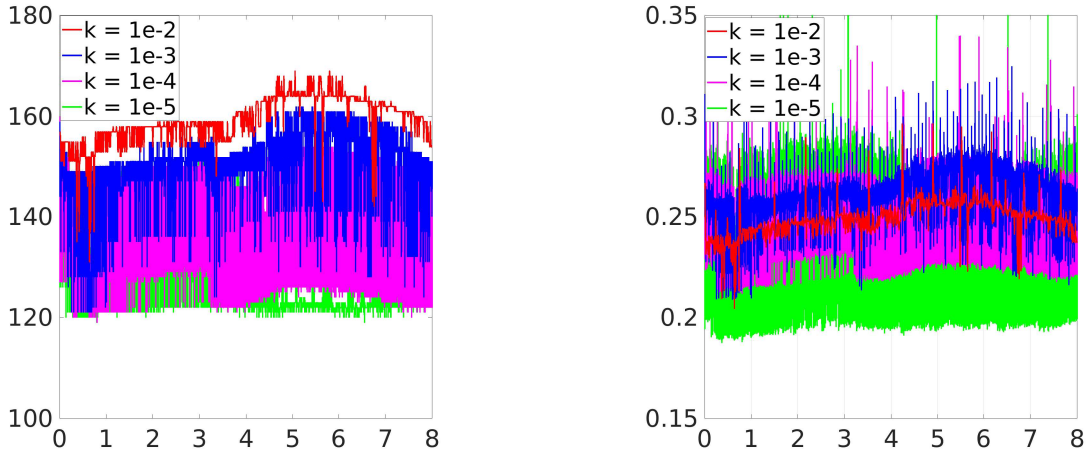


Figure 1.7: Effort for different time step sizes k for the pressure Poisson problem with initial guess zero. Iteration numbers (left) and computation time in seconds (right). The x-axis represents the time interval [0,8].

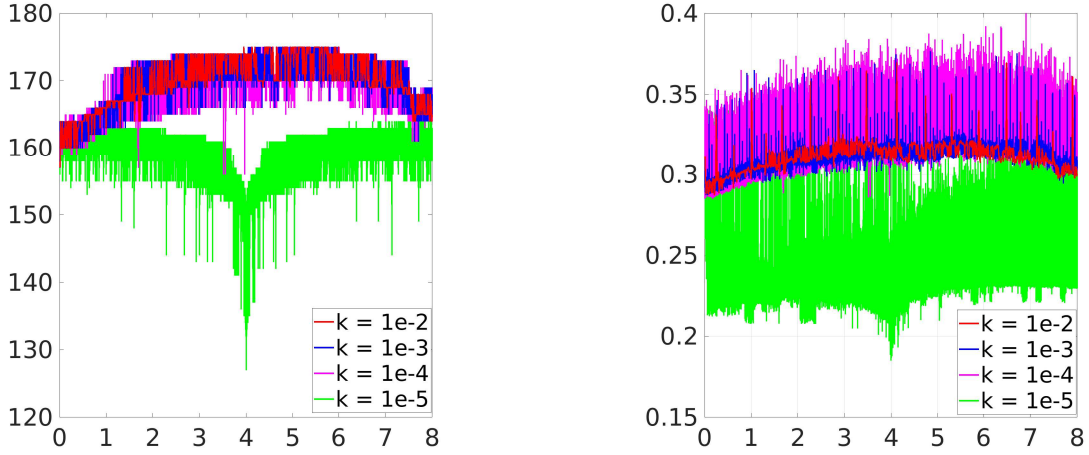


Figure 1.8: Effort for different time step sizes k for the pressure Poisson problem with pressure value from the last iterated as initial guess. Iteration numbers (left) and computation time (right). The x-axis represents the time interval [0,8].

The projection method, i.e. the PP-solver, is constructed in such a way that two systems of equations are solved separately: A pressure Poisson problem and a velocity problem.

Figure 1.7 and figure 1.8 show the iteration numbers and the computation time for the pressure Poisson solver for the entire time interval $[0, 8]$. It can be seen that the pressure Poisson problem is a difficult, so-called ill-conditioned problem. On average, the better configuration requires around 140 iterations and therefore about 0.25 seconds per time step to solve this problem.[2] The iteration numbers become minimally smaller at a smaller time step, but this has almost no effect on the computation time.

Figure 1.8 shows that using the last pressure iteration as initial guess for the pressure Poisson solver results in more iterations and thus relatively more computation time, regardless of the time step size used. The specific solver configuration can be found in section 3.2.3.

The nonlinearity of the velocity problem from the Navier-Stokes equations is treated using the fixed-point iteration mentioned in section 3.2.2. In figure 1.9 and figure 1.10 it becomes very clear that a smaller time step requires significantly fewer iterations and thus less computation time.

[2] At level 3 (8320 elements) and parallel-in-space with 16 cores using Q_2/P_1^{disc} element.

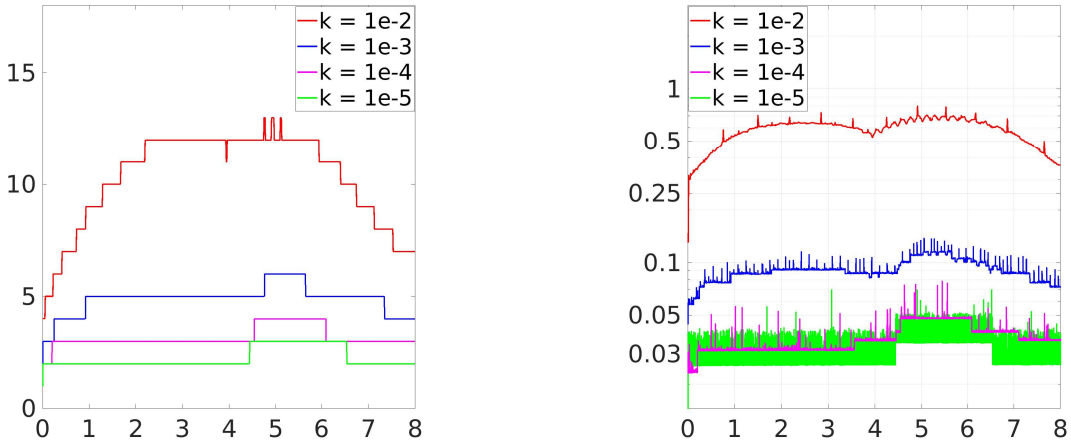


Figure 1.9: Different time step sizes k for the velocity problem. Number of fixed-point iterations (left) and computation time per time step in seconds (right).

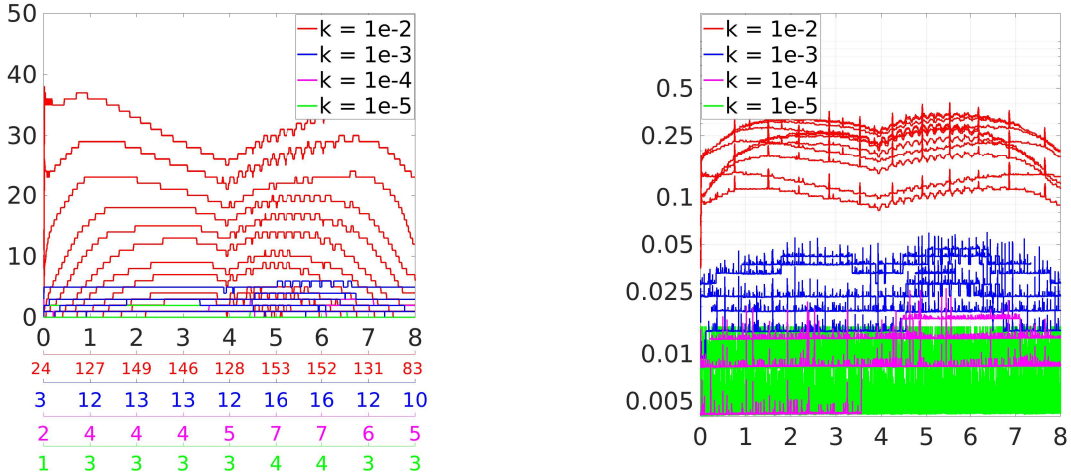


Figure 1.10: Analysis of the linear system solver within the fixed-point iteration. Iteration numbers for each fixed-point iteration (left) and corresponding computation time in seconds (right).

Figure 1.9 and Figure 1.10 show in each case the cost and effort for the entire time interval $[0, 8]$.

Figure 1.10 analyzes the iteration curves per fixed-point iteration. On the lower additional x-axis the total iteration numbers of the linear system solver for the respective time step are added up, where the color of the axis represents the corresponding time step k. It can be observed that the solver has problems for a large time step size such as $k = 0.01$. In this case, one fixed-point iteration takes about 0.25 seconds, which is almost the same time as solving the pressure Poisson problem. Smaller time steps require less than 0.025 seconds of computation, which is roughly only 10% of the required time compared to pressure Poisson solver.

For completeness, the overall convergence rate is shown in the figure 1.11 and figure 1.12 for the entire time interval $[0, 8]$, i.e., the final defect divided by the start defect to the power of 1 divided by the number of iterations.

The convergence rate of the pressure Poisson solver is in the range of 10^{-1}, independent of the chosen time step size k.

The convergence rate of the velocity solver decreases significantly when a smaller time step size is selected.

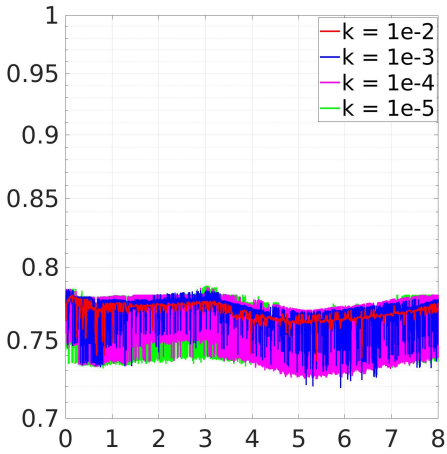


Figure 1.11: Convergence rate of the pressure Poisson solver.

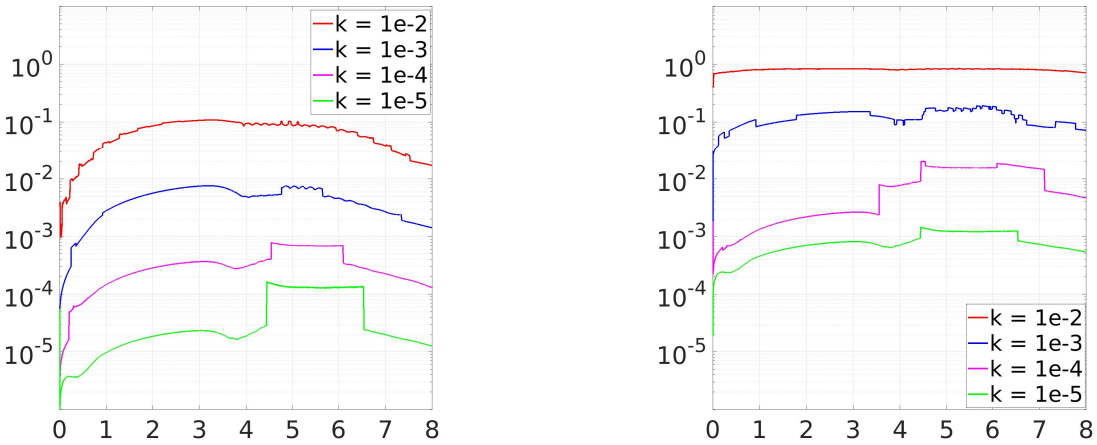


Figure 1.12: Convergence rate of the fixed-point solver (left), where the number of fixed-point iterations was taken as the iteration count and convergence rate of the velocity solver (right), where the number of iterations is the total iteration number of the linear system solver.

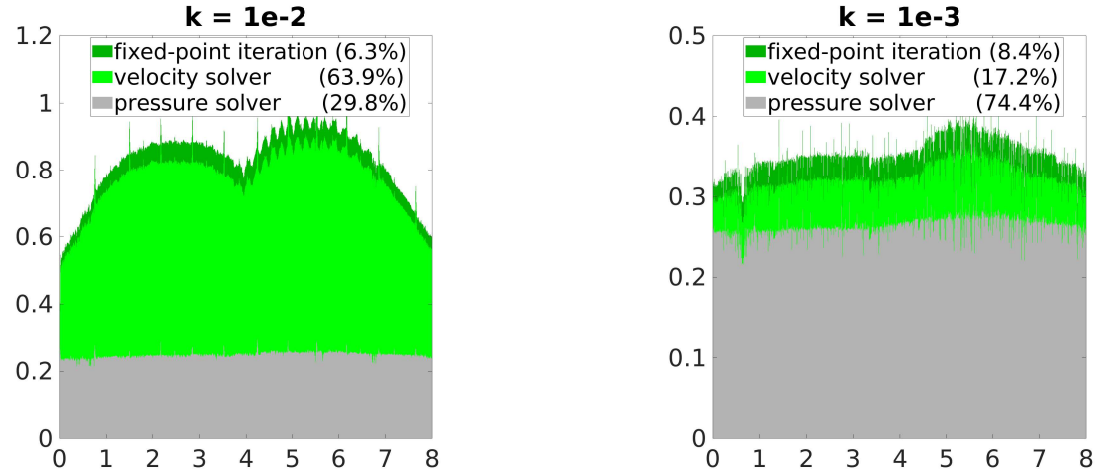


Figure 1.13: Ratio of computation time between velocity and pressure Poisson solver in percentages for time step size $k = 1e-2$ (left) and $k = 1e-3$ (right) over the entire time interval [0,8].

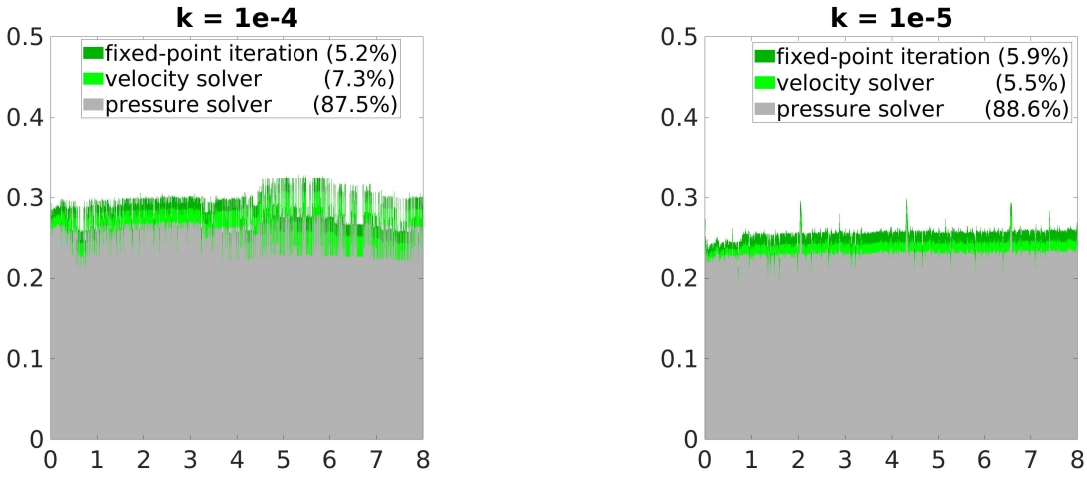


Figure 1.14: Ratio of computation time between velocity and pressure Poisson solver in percentages over the entire time interval [0,8].

To get a better idea of the relation between the velocity solver and the pressure Poisson solver, the composition of the computation time is shown in figure 1.13 and figure 1.14. In general, solving the pressure Poisson problem requires more than 80% of the computation time.

In the dark green region "fixed-point iteration" contains the assembly of the transport matrix as well as the vector additions and multiplications within the fixed-point iteration. The time required is around 6% of the total runtime. In case of a small time step, the assembly of the transport matrix even exceeds the computation time of the velocity problem.

Fast parallel-in-space and simultaneous-in-time solvers

This work focuses on the above mentioned projection method, the so called sequential-in-time PP-solver, as the basis for creating a fast time-simultaneous solver that more efficiently exploits modern HPC architecture. The idea is to decouple the pressure Poisson solver from time in order to solve the most time consuming part fully parallel in time.

The main idea for the development of a high-speed CFD solver came from [78]. Many researchers of the LSIII chair at TU-Dortmund University, are working on different research areas of time-simultaneous CFD solver:

- There is a new fast solver approach for Poisson problems using the Q_1 element with prehandling and lower precision, presented in [63].
  Therefore, a comparison between the commonly used Q_2/P_1^{disc} element and the Q_2/Q_1 element is a part of this thesis. By default, the PP-solver uses an artificial Laplace operator for the pressure Poisson problem, but the fast solver from [63] requires the true Laplace operator. It should already be mentioned that the sequential-in-time PP-solver does not work with the real Laplace matrix and leads to divergence after the first iterations.
- A time-simultaneous approach for the velocity problem, i.e., for non-stationary nonlinear PDEs, can be found in [26], [27] and [49], which is not part of this work.
- Adaptive error estimators as well as other time-simultaneous topics are also being researched.

First results of the time decoupled approach are computed in MATLAB[3] and presented in [51] and [50]. In general, MATLAB is only suitable for less development time and data visualization, but not for HPC computation. The disadvantages of MATLAB are described briefly below.

[3] Available at https://www.mathworks.com/products/matlab.html

Hardware and Software setup

The Finite-Element-Analysis-Toolbox, developed by members of the LSIII chair at TU-Dortmund University, is an advanced open-source software in C++, freely available under GPL.[4] The software is characterized by an optimal interaction of methodology, algorithms and implementation.

All simulations were performed with the FEAT3 software, with the following configurations:[5]

Operating-System:	Red Hat Enterprise Linux 9.4,
Compiler:	CMake 3.28.3,
	GCC 13.2.0,
MPI:	Open MPI 4.1.6.

FEAT3 provides the opportunity to systematically exploit modern HPC architecture so that a very high computational, numerical and hence energy efficiency can be obtained. Compared to vectorized MATLAB code, there are important reasons why an implementation in C++ can provide a 10 to 100 times improvement in performance.

i. MATLAB gets a large part of its performance from the vectorization of operations. The main problem is that CFD usually uses unstructured grids (and thus matrices), which significantly limits the possibilities of vectorization. The unstructured code parts would need to be outsourced to an external backend library in C/C++ to increase the overall performance of the code. In MATLAB calling external libraries costs additional time.

ii. Memory in MATLAB is dynamically allocated and freed. The following example illustrates this problem. For a small matrix multiplication

$$\mathbf{A} = \mathbf{B} \cdot \mathbf{C} + \mathbf{D} \cdot \mathbf{E} + \mathbf{F} \cdot \mathbf{G},$$

two temporary matrices are required to be created in MATLAB. In C++, the memory is allocated beforehand and temporarily matrix creation is not required. The performance problem from MATLAB becomes particularly apparent when memory can be allocated outside of a 1000 times loop in C++.

iii. In case of parallel processing, MATLAB is a multi-process model and C++ is a multi-thread model. Therefore, to parallelize many small tasks, C++ provides a linear gain up to many threads, but in MATLAB there can be a negative performance gain.

iv. In C++, it is possible for an experienced programmer to completely avoid *L2 cache* miss and even *L1 cache* miss, hence pushing CPU to its theoretical throughput limit. Performance of MATLAB can lag behind C++ by a factor of 10 times due to this reason alone. [6]

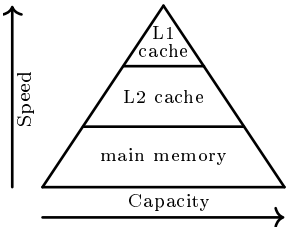
However, the development time in C++ is also higher by a factor of 10 compared to MATLAB.

The computation can be performed on a supercomputer, such as the Linux-HPC-Cluster at the TU-Dortmund University - LiDO3.

[4] Available for preliminary testing at GitHub: https://github.com/tudo-math-ls3/feat3.

[5] MATLAB and ParaView are used for post-processing, i.e. visualization of all data.

[6] Modern computer systems often use at least two levels of caches. The second-level *L2 cache* is larger, and therefore slower, than the first-level *L1 cache*. The processor first looks for the data in the *L1 cache*. If the *L1 cache* misses, the processor looks in the *L2 cache*. If the *L2 cache* misses, the processor fetches the data from main memory. Many modern computer systems add even more levels of cache to the memory hierarchy, because accessing main memory is comparatively slow. For more details see [39].



1.2 Outline

Chapter 2 presents the modeling of the incompressible Navier-Stokes equations, which are used to obtain many well known CFD problems. The condition for well-posedness are analyzed, which highlights difficulties with the discrete gradient and divergence space. It is shown that the main criterion for convergence is the achievement of a divergence-free velocity. The Q_2/P_1^{disc} finite element pair provides a superior approximation of the discrete gradient and divergence space compared to the Taylor-Hood element pair. The discretization, interpolation and consistency errors are calculated for both element pairs. It is shown that the Q_2/P_1^{disc} element is more advantageous compared to the Q_2/Q_1 element. Using the finite element method (FEM) for space discretization and the θ-scheme for time discretization, an algebraic problem is generated, which can be solved using computational techniques.

Chapter 3 contains an efficient and robust solver for the CFD problems, which are discussed in chapter 2. As a first step, the projection method, i.e. the PP-solver, is introduced, that solves the saddle-point problem decoupled in only one (outer) iteration. The idea of the solver is that the resulting error from the solution is not larger than the total error in the discretization, which are discussed in chapter 2. Based on the PP-solver, a global-in-time solver is presented that can perform several outer iterations. The pressure Poisson problems, which take the most computing time, are solved completely parallel-in-time and -space.

A large amount of memory is required to compute a higher number of time steps parallel-in-time.[7] Therefore, a visualization of the memory consumption is provided, that illustrates the necessity of a local approach, which assemble smaller matrices of one time step instead of the large system including all time steps. The local approach is able to compute quickly a larger number of blocked time steps parallel-in-time and -space.

In contrast to the PP-solver, no divergence-free update is performed in the time-simultaneous case. This means that the well-posedness from chapter 2 is not guaranteed. At the end, suggestions, such as the augmented Lagrangian method, are made to get a direct divergence-free global-in-time solver. In addition a recoupling approach is presented, which penalizes the divergence-defect more strictly and can also be considered as smoother in a multigrid in time method.

Chapter 4 analyzes the update behavior of the new developed time-simultaneous solver using the Crank-Nicolson time stepping scheme. The investigation of the pressure update leads to the conclusion, that a divergence-free velocity results in accelerated convergence. Further simulations show the importance of updating a divergence-free velocity after computing the pressure Poisson problem. The posterior divergence-free optimization using the augmented Lagrangian method from chapter 3, is analyzed for different parameters with the focus on the correlation between computing time and convergence behavior in order to achieve a best possible acceleration. In addition, the performance of recoupled approach in the multigrid in time approach is analyzed, which ensure a better accuracy as the sequential-in-time PP-solver after only one (outer) iteration.

Since the performance of the augmented Lagrangian approach turns out being too poor, a much faster divergence-free update similar to the PP-algorithm is presented and tested, which also allows to perform thousands of iterations parallel-in-time and -space in case of Navier-Stokes equations.

Chapter 5 concentrates on the flow-around-a-cylinder benchmark in long-time computations with many thousands of time steps. The parallel performance in space and time is tested in 3D for the flow-around-a-cylinder grid.

[7] Modern operating systems, like Red Hat Enterprise Linux 9 or Windows 11 Enterprise, support up to maximum of 6TB main memory.

1.3 Notation

symbol	description
Ω, $\partial\Omega$	domain, boundary of the domain
T_h	triangulation of the domain
T	element from the triangulation
E	edge or face of an element of the triangulation
∂_t	time derivative
$\otimes$, $\cdot$	tensor product, inner product
$(\cdot,\cdot)$, $\langle\cdot\rangle$	L^2-inner product, dual pairing of the Hilbert space
$\|\cdot\|_m$, $\|\cdot\|_m$	m-Hilbert norm with $p := 2$, semi norm
$[\cdot]$	jump term
	dimensions
d	dimension of the domain (2 or 3)
d_V, d_V^{loc}	global and local degrees of freedom for the velocity
d_Q, d_Q^{loc}	global and local degrees of freedom for the pressure
	spaces
H^m, H_h^1	Hilbert space, discrete Hilbert space
H_0^1	Hilbert space with Dirichlet zero boundary condition
H^{-1}	dual space of H_0^1
$H(div)$	divergence-free finite element space
L^p	Lebesgue space
Q, Q^*, Q_h	continuous, dual and discrete Hilbert space for the pressure
Q_2/P_1^{disc}	finite element pair
Q_2/Q_1	Taylor-Hood element
S_h	spurious mode
V, V^*, V_h	continuous, dual and discrete Hilbert space for the velocity
Z, Z_h	continuous and discrete divergence-free space ($\ker(B^*)$, $\ker(B_h^*)$)
	bi-linear form
$a(\cdot,\cdot)$	bi-linear between the velocity spaces
$b(\cdot,\cdot)$	bi-linear for the divergence and gradient space
$t(\cdot,\cdot)$	bi-linear form between two Hilbert spaces
	matrices / operators / tensors
σ	Cauchy stress tensor
ε	deformation tensor
$\Pi_{V,h}$, $\Pi_{Q,h}$	Fortin operator for velocity and pressure
Φ, Ψ	nodal basis for velocity and pressure
A	linear operator from $V \to V^*$
∇, B, $\mathbf{B}$	gradient, gradient operator, gradient matrix
$\nabla\cdot$, B^*, $\mathbf{B}^\top$	divergence, divergence operator, divergence matrix
C	preconditioner
$\mathbf{I}$	unit matrix
$\mathbf{I}_{\texttt{lvl}}^{\texttt{lvl-1}}$, $\mathbf{I}_{\texttt{lvl-1}}^{\texttt{lvl}}$	prolongation and restriction matrix in space
$\hat{\mathbf{I}}_{\texttt{lvl}}^{\texttt{lvl-1}}$, $\hat{\mathbf{I}}_{\texttt{lvl-1}}^{\texttt{lvl}}$	prolongation and restriction matrix in time
$\mathbf{K}$	convection matrix
$\mathbf{L}$	Laplace matrix
$\mathbf{M}$, $\mathbf{M}_l$, $\mathbf{M}_p$	mass matrix, lumped mass matrix, lumped pressure mass matrix
$\mathbf{P}$	pressure Poisson matrix
$\mathbf{S}$, $\overline{\mathbf{S}}$	Schur matrix, explicit Schur matrix

symbol	description
	elements / test functions / vectors
κ_h, κ_h^v, $\kappa_h^{p_1}$,$\kappa_h^{p_2}$	consistency error
φ	test function
ϕ_i, ψ_i	element from the nodal basis Φ resp. Ψ
f	RHS (divergence-free)
$\mathbf{f}_p$	RHS for pressure Poisson equation
g, $\boldsymbol{g}$, $\mathbf{g}$	weight of the fluid (acceleration of gravity)
p, p_h, $\mathbf{p}$	continuous pressure, discrete pressure, pressure vector
q, q_h	test function
$\mathbf{q}$	solution vector from pressure Poisson equation
$\overline{u}$, u, u_h, $\mathbf{u}$, $\tilde{\mathbf{u}}$	flow independent convection term (constant), continuous velocity, discrete velocity, velocity vector divergence-free, velocity vector not divergence-free
v, v_h	test function
z, z_h	element from Z, Z_h the continuous and discrete divergence-free space ($\ker(B^*)$, $\ker(B_h^*)$)
	constants / scalars
α_D, α_R	diffusive and reactive
β, β_a, β_b	continuous inf-sub constant
β_h, $\beta_{a,h}$, $\beta_{b,h}$	discrete inf-sub constant
γ, γ_L, γ_P	scaling parameter for augmented Lagrangian, Laplace update, pressure Poisson update
$\theta \in [0,1]$	$\theta = 1$: (implicit) backward Euler, $\theta = 0.5$: Crank-Nicolson
λ	proportionality constant (for the Cauchy stress tensor)
ν, μ	kinematic viscosity, dynamic viscosity
ρ	density
n	outward unit normal
L	characteristic length
Re	Reynolds number
	indices
`lvl`	current level
h	grid diameter
k	time step size
K	blocked time steps
l	index for time step
t	time
t_0	staring time
T	end time

Chapter 2

Introduction to incompressible flow problems

Contents

The fundamental basis of computational fluid dynamics (CFD) problems is the Navier-Stokes equations. These equations can be simplified by removing terms, which leads to the Stokes-Oseen or Stokes equations. The incompressible Navier-Stokes equations are a set of partial differential equations written as:

$$\frac{\partial u}{\partial t} - \nu \Delta u + u \cdot \nabla u + \nabla p = g \text{ in } \Omega \times (t_0, T], \quad (2.1a)$$

$$\nabla \cdot u = 0 \text{ in } \Omega \times (t_0, T], \quad (2.1b)$$

where $\Omega \subset \mathbb{R}^d, d = 2, 3$ is a compact domain with Lipschitz boundary, $(t_0, T]$ represents a time interval, $u : \Omega \times (t_0, T] \to \mathbb{R}^d$ is the velocity of the Newtonian fluid, $p : \Omega \times (t_0, T] \to \mathbb{R}$ denotes the pressure of the incompressible fluid and ν is the kinematic viscosity.

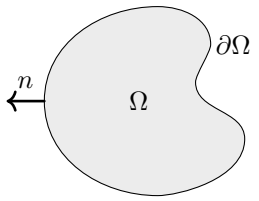


Figure 2.1: Illustration of a domain $\Omega \subset \mathbb{R}^2$ with Lipschitz boundary $\partial\Omega$.

The initial and boundary conditions are discussed in section 2.4.

There is a coupling between the pressure and the velocity, due to the incompressibility condition, which is written in Eq. (2.1b). To solve the Navier-Stokes equations is a challenging problem because of its convective part, i.e., $u \cdot \nabla u$ in Eq. (2.1a), as it creates a nonlinearity.

2.1 Modeling & fundamentals of functional analysis

The Navier-Stokes equations are derived from the conservation of mass and the conservation of momentum using the Reynolds transport theorem, which is also known as the Leibniz-Reynolds transport theorem.

Theorem 2.1: Reynolds transport

Suppose $\Omega(t)$ is a time dependent domain with boundary $\partial\Omega(t)$.
Let $\varphi : \Omega(t) \times (0, \infty) \to \mathbb{R}$ denotes a continuously differentiable vector field and u is the continuously differentiable velocity field in this domain. The outward unit vector denoted by n. Then the Reynolds transport theorem is written in the following form:

$$\frac{d}{dt}\int_{\Omega(t)} \varphi \, dx = \int_{\Omega(t)} \frac{\partial \varphi}{\partial t} \, dx + \int_{\partial\Omega(t)} (\varphi u) \cdot n \, ds. \tag{2.2}$$

Detailed proof of this theorem can be found in [71] page 416 - 418.

2.1.1 Continuity equation

The conservation of mass results from the Reynolds transport theorem 2.1. After applying the well known Gauss's divergence theorem, mentioned in [71], in Eq. (2.2), it reads

$$\frac{d}{dt}\int_{\Omega(t)} \varphi \, dx = \int_{\Omega(t)} \left(\frac{\partial \varphi}{\partial t} + \nabla \cdot (\varphi u) \right) \, dx. \tag{2.3}$$

Using the mass formulation $\varphi := \rho$, as the density of the fluid, the Eq. (2.3) is written in the form of

$$\int_{\Omega(t)} \left(\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho u) \right) \, dx = 0, \tag{2.4}$$

which implies

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho u) = 0. \tag{2.5}$$

The continuity equation Eq. (2.5) is set to zero, since the masses in the system does not change over the time. Considering a constant density in the conservation of mass principle Eq. (2.5) leads to its incompressible form:

$$\nabla \cdot u = 0. \tag{2.6}$$

2.1.2 Derivation of the Navier-Stokes equations

The law of conservation of momentum is derived from the Newton's second law of motion that is

$$F = ma, \tag{2.7}$$

where F describes the force, m is the mass of the body and a denotes the acceleration.
Mathematically, it is written as

$$ma = \frac{d}{dt}\int_{\Omega(t)} \rho u \, dx. \tag{2.8}$$

Applying Reynolds transport theorem 2.1 in Eq. (2.8) leads to

$$\frac{d}{dt}\int_{\Omega(t)} \rho u \; dx = \int_{\Omega(t)} \frac{\partial(\rho u)}{\partial t} \; dx + \int_{\partial\Omega(t)} (\rho u \otimes u) \cdot n \; ds. \tag{2.9}$$

The first term on the right-hand-side describes the rate of change of the fluid inside the region. The second term on the right-hand-side describes the inflow and the outflow rate of the pulse through the surface in terms of the mass flow.

Making use of the divergence theorem yields:

$$\frac{d}{dt}\int_{\Omega(t)} \rho u \; dx = \int_{\Omega(t)} \left(\frac{\partial(\rho u)}{\partial t} + \nabla \cdot (\rho u \otimes u) \right) \; dx. \tag{2.10}$$

Expanding Eq. (2.10) results in

$$\frac{\partial(\rho u)}{\partial t} + \nabla \cdot (\rho u \otimes u) = \rho \left(\frac{\partial u}{\partial t} + u \cdot \nabla u \right) + u \left(\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho u) \right) \tag{2.11a}$$

$$= \rho \left(\frac{\partial u}{\partial t} + u \cdot \nabla u \right). \tag{2.11b}$$

The second term on the right-hand-side of Eq. (2.11a) is equal to zero due to the constant density and also as a result of the incompressibility condition mentioned in Eq. (2.6).

Modeling of the forces

The forces F in Eq. (2.7) can be divided into two categories: body forces $\int_{\Omega(t)} \rho g \; dx$, which are caused by external force fields and affect the entire volume, and surface forces $\int_{\partial\Omega(t)} b_n \; ds$, that are caused by the environment through direct contact and affect only a portion of the volume:

$$F = \int_{\Omega(t)} \rho g \; dx + \int_{\partial\Omega(t)} b_n \; ds, \tag{2.12}$$

where g denotes the acceleration of gravity, which means the specific weight of the fluid.

The forces b_n of the fluid stresses are characterized due to the (symmetric) Cauchy stress tensor $\sigma : \Omega \to \mathbb{R}^d$, with the dimension $d = 2, 3$. After applying the divergence theorem into the surface stresses in Eq. (2.12) leads to

$$F = \int_{\Omega(t)} \rho g \; dx + \int_{\partial\Omega(t)} b_n \; ds = \int_{\Omega(t)} \rho g \; dx + \int_{\partial\Omega(t)} \sigma n \; ds \tag{2.13a}$$

$$= \int_{\Omega(t)} \rho g \; dx + \int_{\Omega(t)} \nabla \cdot \sigma \; dx. \tag{2.13b}$$

Combining Eq. (2.11b) with Eq. (2.13b) yields to the conservation of momentum

$$\rho \left(\frac{\partial u}{\partial t} + u \cdot \nabla u \right) = \rho g + \nabla \cdot \sigma. \tag{2.14}$$

For Newtonian viscous fluids the stress tensor is investigated by [22] and it is mathematically written as

$$\sigma := (\lambda \nabla \cdot u - P) I + 2\mu\varepsilon(\nabla u), \tag{2.15}$$

where P is the pressure, $\lambda \approx -\frac{2}{3}\mu$ shows the Lamé parameters and ε denotes the (symmetric)

deformation tensor, given by

$$\varepsilon(\nabla u) := \frac{1}{2}(\nabla u + (\nabla u)^\top + \nabla u\,(\nabla u)^\top) \tag{2.16a}$$

$$\approx \frac{1}{2}(\nabla u + (\nabla u)^\top). \tag{2.16b}$$

Generally, only linear deformations arise in fluid mechanics, hence the second-order term $\nabla u\,(\nabla u)^\top$ in Eq. (2.16a) can be omitted.

In case of constant viscosity and incompressible fluids mentioned in Eq. (2.6), the Cauchy stress tensor Eq. (2.15) simplifies to

$$\sigma = -PI + 2\mu\varepsilon(\nabla u). \tag{2.17}$$

It follows with the linearized deformation tensor in Eq. (2.16b)

$$\rho\left(\frac{\partial u}{\partial t} + u \cdot \nabla u\right) = \rho g + \nabla \cdot \sigma = \rho g - \nabla P + 2\mu\nabla \cdot \varepsilon(\nabla u) \tag{2.18a}$$

$$= \rho g - \nabla P + \mu\Delta u + \mu\nabla(\nabla \cdot u) \tag{2.18b}$$

$$= \rho g - \nabla P + \mu\Delta u. \tag{2.18c}$$

The term $\mu\nabla(\nabla \cdot u)$ in Eq. (2.18b) vanished due to the incompressibility condition.

After applying the Dirichlet zero-boundary conditions the boundary integrals are zero. In case of natural boundary conditions, the boundary integrals do not vanish. More explanations of the importance of boundary conditions are provided in section 2.4.

Dividing Eq. (2.18) by the density ρ and using the normalized pressure $p := \frac{P}{\rho}$ leads to the incompressible Navier-Stokes equations:

Definition 2.1: The incompressible Navier-Stokes equations

The instationary Navier-Stokes equations are written as

$$\frac{\partial u}{\partial t} - \nu\Delta u + u \cdot \nabla u + \nabla p = g \text{ in } \Omega \times (t_0, T], \tag{2.19a}$$

$$(-)\nabla \cdot u = 0 \text{ in } \Omega \times (t_0, T], \tag{2.19b}$$

and the stationary case is given by

$$-\nu\Delta u + u \cdot \nabla u + \nabla p = g \text{ in } \Omega, \tag{2.20a}$$

$$(-)\nabla \cdot u = 0 \text{ in } \Omega. \tag{2.20b}$$

where to find the velocity field $u : \Omega \times (t_0, T] \to \mathbb{R}^d$ and the pressure $p : \Omega \times (t_0, T] \to \mathbb{R}$. The domain $\Omega \subset \mathbb{R}^d$, $d = 2, 3$ is needed to be bounded and the boundary must be sufficiently regular, such as compact with Lipschitz boundary.[a] The force g and the boundary values should be given and as well as initial condition at the beginning of the time interval for the velocity and pressure, if needed.

[a] The Sobolev embedding theorem 2.2 require that the boundary is sufficiently regular. Accordingly, the Navier-Stokes equations and variational problems in section 2.2 are defined on domains with Lipschitz boundary.

For further simplification, the additional minus in the incompressible condition Eq. (2.19b) and Eq. (2.20b) is used. This has no physical effect on the equation and is only used to get later in the weak formulation in definition 2.5 no change of sign in the pressure matrix.

The resulting system is non-linear and asymmetrical. It can be solved numerically, if boundary and initial conditions are specified.

Details on mathematical modeling of the Navier-Stokes equations can be found in [3].

By replacing the non-linearity $u \cdot \nabla u$ in Eq. (2.19a) with a flow-independent convection velocity $\overline{u} \cdot \nabla u$ yields to the Stokes-Oseen equations:

Definition 2.2: The incompressible Stokes-Oseen equations

The Stokes-Oseen equations for the instationary case are written as

$$\frac{\partial u}{\partial t} - \nu \Delta u + \overline{u} \cdot \nabla u + \nabla p = g \text{ in } \Omega \times (t_0, T], \tag{2.21a}$$

$$(-)\nabla \cdot u = 0 \text{ in } \Omega \times (t_0, T], \tag{2.21b}$$

with a constant $\overline{u} \neq u$. The form of the stationary case is

$$-\nu \Delta u + \overline{u} \cdot \nabla u + \nabla p = g \text{ in } \Omega, \tag{2.22a}$$

$$(-)\nabla \cdot u = 0 \text{ in } \Omega, \tag{2.22b}$$

with the same requirements as for the Navier-Stokes equations in definition 2.1.

Omitting the convective term leads to the Stokes equations. The Stokes equations describe highly viscous fluid, i.e., the kinematic viscosity ν has a higher order of magnitude, so that the convective term can be omitted.

Definition 2.3: The incompressible Stokes equations

The instationary Stokes equations are given by

$$\frac{\partial u}{\partial t} - \nu \Delta u + \nabla p = g \text{ in } \Omega \times (t_0, T], \tag{2.23a}$$

$$(-)\nabla \cdot u = 0 \text{ in } \Omega \times (t_0, T]. \tag{2.23b}$$

The stationary case is

$$-\nu \Delta u + \nabla p = g \text{ in } \Omega, \tag{2.24a}$$

$$(-)\nabla \cdot u = 0 \text{ in } \Omega, \tag{2.24b}$$

with the same requirements as for the Navier-Stokes equations in definition 2.1.

2.1.3 Overview of function spaces and embedding theorems

Some preliminaries are required to make function spaces in which the velocity u and the pressure p are defined. In the case of weak formulation, function spaces are being sought:

- where the boundary conditions must be included appropriately.
- where the derivatives up to m-th order should be included.
- where the derivatives must be integrable to the p-th power. In general, $p = 2$ is sufficient.
- which must be complete, i.e., every Cauchy sequence will have a limit in the space, which is necessary to make statements of convergence.

Banach space

The above mentioned points are fulfilled by the Banach spaces. The Banach spaces are defined in [71] as a complete normalized vector space by

$$W^{m,p}(\Omega) = \{\varphi : \varphi \in L^p(\Omega), D^\alpha \varphi \in L^p(\Omega) \text{ for all } |\alpha| \leq m\}, \tag{2.25}$$

with α as multi-index and the associated norm

$$\|\varphi\|_{m,p} := \|\varphi\|_{W^{m,p}(\Omega)} = \left(\sum_{|\alpha| \leq m} \|D^\alpha \varphi\|^p_{L^p(\Omega)} \right)^{\frac{1}{p}}, \tag{2.26}$$

with $1 \leq p < \infty$, as well as in order to make a statement of the boundary conditions, the semi norm is defined by

$$|\varphi|_{m,p} := \left(\sum_{|\alpha| = m} \|D^\alpha \varphi\|^p_{L^p(\Omega)} \right)^{\frac{1}{p}}. \tag{2.27}$$

Due to embedding theorems the Banach space $W^{m,p}(\Omega)$ can be identified with the $L^q(\Omega)$ space and the $C(\Omega)$ space, which is necessary in many cases. For instance, it allows to make statements about the existence of solutions and to justify why the interpolation is allowed. These embeddings depend on the dimension d of the domain $\Omega \subset \mathbb{R}^d$ and are mentioned below in theorem 2.2.

Theorem 2.2: General Sobolev's inequalities

Let the domain $\Omega \subset \mathbb{R}^d$ be bounded, open and the boundary $\partial\Omega$ is in C^1. Therefore,

- for $mp < d$ the embedding

$$W^{m,p}(\Omega) \hookrightarrow L^q(\Omega) \text{ is continuous for } q \leq \frac{dp}{d - mp}, \tag{2.28}$$

  and the continuity of the embedding leads to the estimate

$$\|u\|_{L^q(\Omega)} \leq C_p \|u\|_{m,p}, \tag{2.29}$$

  with $u \in W^{m,p}(\Omega)$ and C_p as a constant depending on m, p and d in Ω.

- for $mp = d$ the embedding

$$W^{m,p}(\Omega) \hookrightarrow L^q(\Omega) \text{ is continuous for } q < \infty, \tag{2.30}$$

  and in the special case $p = 1$ and $m = d$ also continuous for $q = \infty$.

- for $mp > d$ it follows the embedding

$$W^{m,p}(\Omega) \hookrightarrow C^\alpha(\overline{\Omega}), \tag{2.31}$$

  and the continuity of the embedding leads to the estimate

$$\|u\|_{C^\alpha(\overline{\Omega})} \leq C_\alpha \|u\|_{m,p}, \tag{2.32}$$

  with $u \in W^{m,p}(\Omega)$ and C_α as a constant depending upon m, p, d, α and Ω.

The statement from theorem 2.2 reads, every bounded sequence in $W^{m,p}(\Omega)$ has a (strong) convergent subsequence in $L^q(\Omega)$. The compactness follows from the Rellich-Kondrachov theorem 2.3, written below.

Theorem 2.3: Rellich-Kondrachov

The injections mentioned in Eq. (2.28) are compact for $m = 1$ with $q \in \left[1, \frac{dp}{d-p}\right)$.

Proofs of theorem 2.2 and theorem 2.3 as well as information about embedding and compactness are explained in [29] chapter 5 resp. [16] in chapter 9.3.

Hilbert space

A Banach space equipped with a complete inner product space is called Hilbert space. The Hilbert spaces are denoted by $H^{m,p}(\Omega)$ and the dual space of a Hilbert space H is defined by H^*. Hilbert spaces have the advantage of being reflexive spaces. This means, that the natural embedding in its dual space is an isomorphism. Therefore a Hilbert space can be identified with the dual space of its dual space, i.e., $H = H^{**}$.

Bessel potential spaces

In case of $p = 2$,

$$H^m(\Omega) := H^{m,2}(\Omega) = W^{m,2}(\Omega), \tag{2.33}$$

applies, a fact which is proven generally by Meyers-Serrin theorem in [54]. These spaces are so called Bessel potential spaces with the identified norms, written as

$$\|\cdot\|_m := \|\cdot\|_{m,2} \text{ and } |\cdot|_m := |\cdot|_{m,2}. \tag{2.34}$$

In practice, only the two and three dimensional cases are relevant for CFD problems. The embedding rates in the two dimensional case apply due to the Sobolev embedding theorem 2.2 read as

$$H^1(\Omega) \hookrightarrow L^p(\Omega) \text{ is continuous and compact for } 1 \leq p \leq \infty, \tag{2.35}$$

and in the three dimensional case is given by

$$H^1(\Omega) \hookrightarrow L^p(\Omega) \text{ is continuous for } 1 \leq p \leq 6 \text{ and compact for } 1 \leq p < 6, \tag{2.36}$$

with the estimate

$$\|u\|_{L^p(\Omega)} \leq C_p \|u\|_1. \tag{2.37}$$

The space

$$H^{-1}(\Omega) : = \left(H_0^1(\Omega)\right)^* \equiv \{\mu : H_0^1(\Omega) \to \mathbb{R} \text{ linear and continuous}\} \tag{2.38}$$

is defined as the dual space of $H_0^1(\Omega)$, with

$$H_0^1(\Omega) = \{\varphi : \varphi \in H^1(\Omega), \varphi = 0 \text{ on } \partial\Omega\} \tag{2.39}$$

as Hilbert space with zero boundary condition.

2.2 Numerical treatment of saddle point problems

This section focuses on the numerical treatment of CFD problems for Hilbert spaces V and Q. The structure of the numerical treatment can be described in the following steps:

i. Continuous variational formulation in Hilbert spaces V and Q (see section 2.2.1).

ii. Discrete variational formulation in finite dimensional subspaces $V_n \subset V$ and $Q_n \subset Q$ (see section 2.2.3).

iii. Choose bases of V_n and Q_n (see section 2.2.4 and section 2.3), that leads to an algebraic problem.

The well-posedness of the continuous problem is briefly discussed in section 2.2.2. The peculiarity of discrete divergence-free spaces are shortly described in section 2.2.3.

The variational formulation opens a way to generate an algebraic system, discussed in section 2.3, that can be solved on a computer. Furthermore, the variational formulation also leads to the weak formulation.

Theory of saddle point problems are explained in [52], [2] and [14].

2.2.1 Variational formulation (weak formulation)

Consider a continuous bilinear form $t : H_1 \times H_2 \to \mathbb{R}$, where the Hilbert space H_1 also called the trial space and the Hilbert space H_2 is called the test space. According to Riez's representation theorem, mentioned in [71], for a given right side $l \in H_2^*$, there exists a unique representing element $u \in H_1$, for the problem:

Find $u \in H_1$ for a given $l \in H_2^*$ such that

$$t(u, \varphi) = \langle l, \varphi \rangle \text{ for all } \varphi \in H_2, \tag{2.40}$$

where $\langle \cdot, \cdot \rangle$ denotes the dual pairing of the Hilbert space H_2.

Theorem 2.4: Variation problem

For all $l \in H_2^*$ exists a unique element $u \in H_1$, which fulfills

$$t(u, \varphi) = \langle l, \varphi \rangle \text{ for all } \varphi \in H_2. \tag{2.41}$$

The $u \in H_1$ is characterized as a minimum of the energy functional

$$J(\varphi) := \frac{1}{2} t(\varphi, \varphi) - \langle l, \varphi \rangle, \tag{2.42}$$

with not more than one minimum solution

$$J(u) = \min_{\varphi \in H_2} J(\varphi). \tag{2.43}$$

A proof of theorem 2.4 and further details can be found in [14].

According to theorem 2.4, the solution of the CFD problems can be characterized by the minimization property and is unique. Furthermore, incompressible CFD problems can be formulated as a saddle point problem.

Definition 2.4: Abstract saddle point formulation

Let V, Q be Hilbert spaces, $a : V \times V \to \mathbb{R}$, $b : V \times Q \to \mathbb{R}$ and $c : Q \times Q \to \mathbb{R}$ are bilinear forms and $g : V \to \mathbb{R}$ as well as $f : Q \to \mathbb{R}$ are linear continuous functionals.

Find $(u, p) \in V \times Q$ such that

$$a(u, v) + b(v, p) = \langle g, v \rangle \quad \text{for all } v \in V, \tag{2.44a}$$
$$b(u, q) - c(p, q) = \langle f, q \rangle \quad \text{for all } q \in Q. \tag{2.44b}$$

The abstract saddle point formulation, as given in the definition 2.4, can be interpreted as the variation problem from theorem 2.4, which ensures that the solution will not have more than one minimum solution. Using the definition $H_1 = H_2 = V \times Q$ with

$$t((v_1, q_1), (v_2, q_2)) := a(v_1, v_2) + b(v_2, q_1) + b(v_1, q_2) - c(q_1, q_2), \tag{2.45a}$$
$$\langle l, (v, q) \rangle := \langle g, v \rangle + \langle f, q \rangle. \tag{2.45b}$$

proves the equivalence by

$$t((u, p), (v, 0)) = \langle l, (v, 0) \rangle, \tag{2.46a}$$
$$t((u, p), (0, q)) = \langle l, (0, q) \rangle. \tag{2.46b}$$

Setting $q_2 = q = 0$ in Eqs. (2.45) eliminates the terms $c(\cdot, q_2)$, $b(\cdot, q_2)$ and $\langle f, q \rangle$. This results in the equality of Eq. (2.46a) and Eq. (2.44a). While setting $v_2 = v = 0$ in Eqs. (2.45) leads to the equivalence of Eq. (2.46b) and Eq. (2.44b), as it causes $a(\cdot, v_2) = 0$, $b(v_2, \cdot) = 0$ and $\langle g, v \rangle = 0$.

A precise definition of the bilinear forms is provided in definition 2.5 using the weak formulation.

Theorem 2.5: Sufficient condition for a saddle point

A pair $(u, p) \in V \times Q$ is said saddle-point if and only if

$$\inf_{v \in V} \sup_{q \in Q} t(v, q) = \sup_{q \in Q} t(u, q) = t(u, p) = \inf_{v \in V} t(v, p) = \sup_{q \in Q} \inf_{v \in V} t(v, q). \tag{2.47}$$

A proof and also a necessary condition can be examined in [2] section 2.4. The necessary condition for saddle point argue with the Lagrangian functional.

Weak formulation

The weak formulation is used for solving and analyzing the Navier-Stokes equations mentioned in definition 2.1. It follows by multiplying Eqs. (2.19) with test functions $v \in V$, $q \in Q$ and integration over the domain Ω. By using the L^2-inner product $(\cdot, \cdot) := (\cdot, \cdot)_{L^2(\Omega)}$ it reads

$$\left(\frac{\partial u}{\partial t}, v\right) + (u \cdot \nabla u, v) - (\nu \Delta u, v) + (\nabla p, v) = \langle g, v \rangle \text{ for all } v \in V, \tag{2.48a}$$
$$(q, \nabla \cdot u) = \langle f, q \rangle \text{ for all } q \in Q. \tag{2.48b}$$

In the case of incompressibility, the functional f in Eq. (2.48b) is set to 0.

Integration by parts is used to get a weaker condition for the velocity and pressure, written below:

$$(-\nu \Delta u, v) = (\nu \nabla u, \nabla v) - (\nu \nabla u \cdot n, v)_{L^2(\partial\Omega)}, \tag{2.49a}$$
$$(\nabla p, v) = -(p, \nabla \cdot v) + (p, v \cdot n)_{L^2(\partial\Omega)}. \tag{2.49b}$$

The weaker condition for the pressure is necessary, to lower the requirements for the function and the solution space. This allows even piece-wise constant pressure functions only in $L^2_0(\Omega)$, i.e., L^2 space with zero boundary conditions. In section 2.4 it is shown that the partial integration is needed for stable natural boundary conditions.

Definition 2.5: Weak incompressible Navier-Stokes equations

Let the domain $\Omega \subset \mathbb{R}^d$, $d = 2, 3$, be compact with Lipschitz boundary.
Find a pair $(u, p) \in V \times Q$ with the bilinear forms

$$a(u, v) := \left(\frac{\partial u}{\partial t}, v\right) + (u \cdot \nabla u, v) + \nu(\nabla u, \nabla v), \tag{2.50a}$$

$$b(q, v) := -(q, \nabla \cdot v), \tag{2.50b}$$

for given right-hand-sides $g \in V^*$ such that

$$a(u, v) + b(p, v) = \langle g, v\rangle \text{ for all } v \in V, \tag{2.51a}$$

$$b(q, u) = 0 \quad \text{for all } q \in Q, \tag{2.51b}$$

with given boundary and initial conditions.

The weak form weakens the differentiability requirements and satisfy the boundary condition requirements on the Dirichlet part of the domain. The solution is an approximate, since the requirements on the boundary conditions have been weakened.

Operator formulation

Let introduce the linear operator $A : V \to V^*$ by

$$\langle Au, v\rangle = a(u, v) \text{ for all } u, v \in V, \tag{2.52}$$

and further two linear operators $B : Q \to V^*$ and $B^* : V \to Q^*$ by

$$\langle q, B^*u\rangle = \langle Bq, u\rangle = b(q, u) \text{ for all } u, q \in Q, \tag{2.53}$$

where B represents the gradient space and B^* the divergence space. The operator equation reads as for $g \in V^*$ and $f \in Q^*$, find a pair $(u, p) \in V \times Q$ such that

$$Au + Bp = g \text{ in } V^*, \tag{2.54a}$$

$$B^*u = f \text{ in } Q^*. \tag{2.54b}$$

In the context to FEM, these spaces are finite dimensional. Therefore, the operators A and B in Eqs. (2.54) can be represented by a matrix $\mathbf{A}$ and $\mathbf{B}$ and the functionals $(g, f) \in V^* \times Q^*$ with vectors. In section 2.3.1 the matrix formulation is derived by using the Finite-Element-Method. The linear system of equations is well-posed, i.e., it has a unique solution for all right-hand-sides, specially if the system matrix

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^\top & 0 \end{bmatrix} \tag{2.55}$$

is non-singular.

The upcoming section briefly discusses the inf-sup conditions that guarantee a non-singular system matrix.

2.2.2 Well-posedness of the variation problem (inf-sup conditions)

The problem consisting of Eq. (2.54a) and Eq. (2.54b) is well-posed according to Hadamard [38] if for each pair $(g, f) \in V^* \times Q^*$ there exists a unique solution $(u, p) \in V \times Q$ and the corresponding defined solution operator is continuous. The solution's behavior changes continuously with the initial conditions.

The first inf-sup condition for Hilbert spaces was investigated by Nečas in [55] and reads as:

Theorem 2.6: Banach-Nečas (inf-sup condition)

Let $t : H_1 \times H_2 \to \mathbb{R}$ be a continuous bilinear form. The Eqs. (2.54) are well-posed if and only if

i. There exists a constant β such that

$$\beta := \inf_{v \in H_1} \sup_{\varphi \in H_2} \frac{t(v, \varphi)}{\|v\|_{H_1} \|\varphi\|_{H_2}} > 0. \tag{2.56}$$

The condition is well known as inf-sup condition.

ii. For all $\varphi \in H_2$ and for all $v \in H_1$ holds

$$t(v, \varphi) = 0 \text{ implies } \varphi = 0. \tag{2.57}$$

Furthermore, the following a priori estimate applies:

$$\beta \|u\|_{H_1} \leq \|l\|_{H_2^*} = \sup_{\varphi \in H_2} \frac{\langle l, \varphi \rangle}{\|\varphi\|_{H_2}}. \tag{2.58}$$

The theorem is also well known as Banach-Nečas Babuška (BNB) theorem or Babuška-Lax-Milgram (BLM) theorem. A proof can be found in [2] chapter 2.1.

Theorem 2.7: Well-posed variation problem

The variational problem Eq. (2.41) is well-posed if and only if the dual problem is well-posed. If both problems are well-posed, applies:

$$\inf_{v \in H_1} \sup_{\varphi \in H_2} \frac{t(v, \varphi)}{\|v\|_{H_1} \|\varphi\|_{H_2}} = \inf_{\varphi \in H_2} \sup_{v \in H_1} \frac{t(v, \varphi)}{\|v\|_{H_1} \|\varphi\|_{H_2}}. \tag{2.59}$$

The proof of the above theorem 2.7 is explained in [32], chapter. 8.4.

In case of CFD problems, the necessary condition $\dim H_1 = \dim H_2 = V \times Q$ is fulfilled. If theorem 2.7 holds, it follows that the operator B^* is injective and the operator B is surjective. In the finite dimensional case there is a bijective Matrix $\mathbf{B}$, by the fundamental theorem of linear algebra.

Inf-sup conditions of the saddle point system

The saddle point problem, which is made of Eq. (2.54a) and Eq. (2.54b), is well-posed, if the bilinear form $t : H_1 \times H_2 \to \mathbb{R}$ is continuous. Therefore, all associated operators A, B and B^* with the corresponding bilinear forms $a(\cdot, \cdot)$ and $b(\cdot, \cdot)$ are also continuous, i.e.,

$$a(u, v) \leq \|a\| \|u\|_V \|v\|_V, \text{ for all } u, v \in V, \tag{2.60a}$$
$$b(q, v) \leq \|b\| \|v\|_V \|q\|_Q, \text{ for all } v \in V, q \in Q, \tag{2.60b}$$

with a constant $\|a\|$ and $\|b\|$. In each case an inf-sup condition must be fulfilled according to Banach-Nečas theorem 2.6. So there exist best possible constants β_a and β_b , which satisfy the estimates

$$0 < \beta_a \leq \inf_{u \in V} \sup_{v \in V} \frac{a(u,v)}{\|u\|_V \|v\|_V}, \tag{2.61a}$$

$$0 < \beta_b \leq \inf_{q \in Q} \sup_{v \in V} \frac{b(q,v)}{\|v\|_V \|q\|_Q}. \tag{2.61b}$$

Multiplying Eq. (2.61a) by $\|u\|_V$ and Eq. (2.61b) by $\|q\|_Q$ leads to

$$\beta_a \|u\|_V \leq \sup_{v \in V} \frac{a(u,v)}{\|v\|_V}, \tag{2.62a}$$

$$\beta_b \|q\|_Q \leq \sup_{v \in V} \frac{b(q,u)}{\|v\|_V}. \tag{2.62b}$$

The inf-sup conditions guarantee the uniqueness of the pressure.

The conditions Eqs. (2.62) are well known in the literature as the Brezzi condition. In the discrete case also as Babuška-Brezzi condition as well as the Ladyzhenskaya-Babuška-Brezzi condition (LBB condition). More informations about the interrelationships of the inf-sup conditions are investigated in [66].

Brezzi theory (existence and uniqueness)

The approach from Brezzi about the existence and uniqueness of the solution can be found in [17]. He analyzed a sufficient condition for well-posedness is that the velocity must fulfill the divergence-free condition. Brezzi's theory can be summarized in three steps:

i. Determine a solution u_0 from Eq. (2.54b), written as

$$B^* u_0 = f. \tag{2.63}$$

The prerequisite is that the divergence operator B^* is surjective, which means theorem 2.7 must be fulfilled. The consequence results in the matrix $\mathbf{B}^\top$ being of full rank.

For solving Eq. (2.63), the singular value decomposition can be used.

ii. Consider $u - u_0 \in Z$ as an element in the divergence-free space

$$Z := \ker(B^*) := \{v \in V : (q, \nabla \cdot v) = 0 \text{ for all } q \in Q\} \tag{2.64a}$$

$$= \{v \in V : b(q,v) = 0 \text{ for all } q \in Q\}. \tag{2.64b}$$

By adding the term Au_0 in Eq. (2.54a) leads to

$$A(u - u_0) + Bp = g - Au_0. \tag{2.65}$$

Multiplying with a test function $v \in \ker(Z)$ and applying the inner product gives

$$(A(u - u_0), v) + (Bp, v) = (g - Au_0, v). \tag{2.66}$$

Due to the condition $v \in \ker(Z)$, Eq. (2.66) leads to

$$(Bp, v) = (p, B^* v) = 0. \tag{2.67}$$

The requirement is $A|_Z$ need to be bijective.

iii. The equation

$$Bp = g - Au \tag{2.68}$$

is solvable only if $g - Au$ is orthogonal to Z.

Theorem 2.8: Brezzi (sufficient condition for well-posedness)

Let the bilinear form $a(\cdot,\cdot)$ and $b(\cdot,\cdot)$ be continuous.

The saddle-point problem consisting of Eq.(2.54a) and Eq. (2.54b) is well-posed, if and only if

i. There is a constant $\beta_a > 0$, such that the bilinear form $a(\cdot,\cdot)$ is coercive on $Z \times Z$, i.e.,

$$\beta_a \|v\|_V^2 \leq a(v,v) \text{ for all } v \in Z. \tag{2.69}$$

ii. The bilinear form $b(\cdot,\cdot)$ satisfies the inf-sup condition in Eq. (2.62b).

Brezzi's theorem follows from Eqs. (2.62) with $u = v \in Z$.

2.2.3 Conforming discretization (discrete inf-sup condition)

The FEM uses a general technique to build finite dimensional subspaces of the Hilbert spaces V and Q in order to apply the Galerkin method to the variational problem. The saddle point problem, as defined in Eqs. (2.54), is formulated in the discrete case with the subspaces $V_h \subset V$ and $Q_h \subset Q$ and given by:

Find $u_h \in V_h$ and $p_h \in Q_h$ such that

$$a(u_h, v_h) + b(p_h, v_h) = \langle g, v_h \rangle \text{ for all } v_h \in V_h, \tag{2.70a}$$
$$b(q_h, u_h) = \langle f, q_h \rangle \text{ for all } q_h \in Q_h. \tag{2.70b}$$

The discrete version of Brezzi's theorem 2.8 is written as below:

Theorem 2.9: Discrete Brezzi (sufficient condition for well-posedness)

Let the bilinear form $a(\cdot,\cdot)$ and $b(\cdot,\cdot)$ be continuous.

The saddle-point problem consisting of Eq. (2.70a) and Eq. (2.70b) is wells-posed, if and only if

i. The bilinear form $a(\cdot,\cdot)$ is coercive on $Z_h \times Z_h$, i.e.,

$$\beta_{a,h} \|z_h\|_V^2 \leq a(z_h, z_h) \text{ for all } z_h \in Z_h. \tag{2.71}$$

ii. The inf-sup condition is fulfilled for the pressure, i.e.,

$$\beta_{b,h} \|q_h\|_{Q/\ker(B_h)} = \beta_{b,h} \inf_{\tilde{q}_h \in \ker(B_h)} \|q_h + \tilde{q}_h\|_Q \leq \sup_{v_h \in V_h} \frac{b(q_h, v_h)}{\|v_h\|_V}. \tag{2.72}$$

The associated discrete gradient space is denoted by $B_h : Q_h \to V_h^*$ and the discrete divergence space is represented by $B_h^* : V_h \to Q_h^*$. In general the discrete divergence-free space is not a subset

from the continuous divergence-free space, which is further discussed in section 2.3.3. Mathematically, it holds

$$Z_h := \ker(B_h^*) \not\subseteq \ker(B^*) =: Z. \tag{2.73}$$

The discrete inf-sup conditions is written as:

Theorem 2.10: Babuška (discrete inf sup conditions)

Let the bilinear form $a(\cdot,\cdot)$ and $b(\cdot,\cdot)$ be continuous.

Eqs. (2.70) are well-posed, if and only if there exists a constant $\beta_{a,h}$ and $\beta_{b,h}$, such that

$$\beta_{a,h} \leq \inf_{u_h \in V_h} \sup_{v_h \in V_h} \frac{a(u_h, v_h)}{\|v_h\|_V \|u_h\|_V}, \tag{2.74a}$$

$$\beta_{b,h} \|q_h\|_{Q/\ker(B_h)} \leq \sup_{v_h \in V_h} \frac{b(q_h, v_h)}{\|v_h\|_V}. \tag{2.74b}$$

Furthermore, the following error estimate applies:

$$\|u - u_h\|_V + \|p - p_h\|_{Q/\ker(B_h)} \leq c \left(\inf_{v_h \in V_h} \|u - u_h\|_V + \inf_{q_h \in Q_h} \|p - q_h\|_Q \right), \tag{2.75}$$

with a constant c, which is dependent on β_a, β_b, $\beta_{a,h}$ and $\beta_{b,h}$.

The error estimate in Eq. (2.75) is optimal, except the prefactor c.

Approximation error (discrete error estimate)

Brezzi wrote the composition of the constant c from Eq. (2.75) in [30], section II.2.2. The discrete inf-sup conditions leads for the velocity to the error estimate

$$\|u - u_h\|_V \leq \left(1 + \frac{\beta_a}{\beta_{a,h}}\right) \left(1 + \frac{\beta_b}{\beta_{b,h}}\right) \inf_{v_h \in V_h} \|u - v_h\|_V + \frac{\beta_b}{\beta_{a,h}} \inf_{q_h \in Q_h} \|p - q_h\|_Q, \tag{2.76}$$

and the pressure estimate is given by

$$\|p - p_h\|_{Q/\ker(B_h)} \leq \left(1 + \frac{\beta_b}{\beta_{b,h}}\right) \inf_{q_h \in Q_h} \|p - q_h\|_Q + \frac{\beta_a}{\beta_{b,h}} \|u - u_h\|_V \tag{2.77a}$$

$$\begin{aligned} &\leq \left(1 + \frac{\beta_b(\beta_a + \beta_{a,h})}{\beta_{a,h}\beta_{b,h}}\right) \inf_{q_h \in Q_h} \|p - q_h\|_Q \\ &\quad + \left(\frac{\beta_a\beta_b(\beta_a + \beta_{a,h})}{\beta_{a,h}\beta_{b,h}^2} + \frac{\beta_a(\beta_a + \beta_{a,h})}{\beta_{a,h}\beta_{b,h}}\right) \inf_{v_h \in V_h} \|u - v_h\|_V, \end{aligned} \tag{2.77b}$$

where β_a and β_b are the continuity constants from the continuity case in Eqs. (2.61) and $\beta_{a,h}$ and $\beta_{b,h}$ are the discrete constants from Eqs. (2.74) as a gain factor. The velocity estimate Eq. (2.76) was used for obtaining Eq. (2.77b).

Conclusions and remarks:

i. Comparing estimates Eq. (2.76) and Eq. (2.77b) shows that the error estimate for the pressure $\|p - p_h\|_{Q/\ker(B_h)}$ depends on $\frac{1}{\beta_{b,h}^2}$, but the estimate for the velocity $\|u - u_h\|_V$ depends only on $\frac{1}{\beta_{b,h}}$. It is verified in practice, that a small value of $\beta_{b,h}$ has a stronger effect on the pressure approximation p_h than on the velocity u_h.

ii. The error $\|p - p_h\|_Q$ is estimated up to an element of $\ker(B_h)$. In the case $\ker(B_h) \nsubseteq \ker(B)$ certain components of p are not properly approximated. For more detail see section 2.3.3.

In section 2.3, suitable choices of discrete spaces are considered, so that the constant $\frac{\beta}{\beta_h}$ should not degenerate, which means that the prefactor β_h should not go faster to zero as the error will go to zero.

Next, in section 2.2.4, an error estimate will be derived that is independent of the unknown numeric constant β_h.

2.2.4 Fortin interpolation (different types of divergence-free velocities)

For the discrete inf-sup conditions and for the FEM, discussed in section 2.3, a discrete pair $(u_h, p_h) \in V_h \times Q_h$ is needed. One approach to get an appropriate approximation is to use projection operators, so-called Fortin operators, defined as $\Pi_{V,h} : V \to V_h$ and $\Pi_{Q,h} : Q \to Q_h$, which fulfills $u_h = \Pi_{V,h} u \in V_h$ as well as, $p_h = \Pi_{Q,h} p \in Q_h$.

Theorem 2.11: Fortin operator (discrete inf-sup condition)

Let the continuity problem in Eqs. (2.54) be well-posed. Let constants β_a and β_b exists, such that the continuous inf-sup conditions in Eqs. (2.61) are fulfilled. Then the discrete inf-sup conditions in Eqs. (2.74) with $\beta_{a,h}$ and $\beta_{b,h}$ holds exactly if:

For any $u \in V$, there exists a Fortin operator, a bounded linear operator, $\Pi_{V,h} : V \to V_h$, such that

$$b(q_h, u - \Pi_{V,h} u) = 0, \text{ for all } q_h \in Q_h. \tag{2.78}$$

Furthermore, for any $p \in Q$, there exists a Fortin operator $\Pi_{Q,h} : Q \to Q_h$ such that

$$b(p - \Pi_{Q,h} p, v_h) = 0, \text{ for all } v_h \in V_h. \tag{2.79}$$

The literature [30], chapter II.2, contains a proof of theorem 2.11 above and further information, such as that it does not matter whether the continuous problem is solved first or the Fortin interpolated is applied first. It holds figure 2.2:

$$\begin{array}{ccc} V & \xrightarrow{B^\top} & Q \\ \Pi_{V,h} \downarrow & & \downarrow \Pi_{Q,h} \\ V_h & \xrightarrow{B_h^\top} & Q_h \end{array}$$

Figure 2.2: Commutated diagram of the Fortin operator.

Approximation estimate (for the discrete inf-sup constant β_h)

If the spaces V, Q are chosen, the continuous constants β_a as well as β_b can be calculated, as example due to the singular value decomposition (see [30], chapter II.3). The discrete inf-sup constants $\beta_{a,h}$ and $\beta_{b,h}$ are unknown, so an estimate is needed that the discrete problem depends only on the continuous inf-sub constants.

From the discrete inf-sub condition Eq. (2.71) for the velocity, it follows for all $v_h \in V_h$ for the interpolation error:

$$\begin{aligned}
\beta_{a,h}\|v_h - \Pi_{V,h}u\|_V^2 &\leq a(v_h - \Pi_{V,h}u, v_h - \Pi_{V,h}u) && \text{(2.80a)}\\
&= a(v_h - u, v_h - \Pi_{V,h}u) + a(u - \Pi_{V,h}u, v_h - \Pi_{V,h}u) && \text{(2.80b)}\\
&= b(p_h - p, v_h - \Pi_{V,h}u) + a(u - \Pi_{V,h}u, v_h - \Pi_{V,h}u) && \text{(2.80c)}\\
&= b(p_h - \tilde{q}_h, v_h - \Pi_{V,h}u) + b(\tilde{q}_h - p, v_h - \Pi_{V,h}u)\\
&\quad + a(u - \Pi_{V,h}u, v_h - \Pi_{V,h}u) && \text{(2.80d)}\\
&\leq \beta_b\|v_h - \Pi_{V,h}u\|_V\|p_h - \tilde{q}_h\|_Q + \beta_b\|v_h - \Pi_{V,h}u\|_V\|\tilde{q}_h - p\|_Q\\
&\quad + \beta_a\|u - \Pi_{V,h}u\|_V\|v_h - \Pi_{V,h}u\|_V. && \text{(2.80e)}
\end{aligned}$$

In Eq. (2.80c), the error identity $a(v_h - u, v_h) = b(p_h - p, v_h)$ is used. Since in general $Z_h \nsubseteq Z$ holds, i.e., the kernel of B_h is larger than the kernel of B, it is necessary to insert an element $\tilde{q}_h \in \ker(B_h)$ in Eq. (2.80d), because some components of p are not well approximated. In Eq. (2.80e) the continuous inf-sup conditions in Eqs. (2.61) were used.

Dividing Eq. (2.80) by $\|v_h - \Pi_{V,h}u\|_V$ leads to

$$\beta_{a,h}\|v_h - \Pi_{V,h}u\|_V \leq \beta_b\|p_h - \tilde{q}_h\|_Q + \beta_b\|\tilde{q}_h - p\|_Q + \beta_a\|u - \Pi_{V,h}u\|_V. \tag{2.81}$$

From the discrete inf-sup condition in Eq. (2.74b) for the pressure, it follows for $q_h \in Q_h$:

$$\begin{aligned}
\beta_{b,h}\|q_h - \Pi_{Q,h}p\|_Q &\leq \sup_{v_h \in V_h} \frac{b(q_h - \Pi_{Q,h}p, v_h)}{\|v_h\|_V} && \text{(2.82a)}\\
&\leq \sup_{v_h \in V_h} \frac{b(p - \Pi_{Q,h}p.v_h)}{\|v_h\|_V} + \sup_{v_h \in V_h} \frac{b(q_h - p.v_h)}{\|v_h\|_V} && \text{(2.82b)}\\
&= \sup_{v_h \in V_h} \frac{b(p - \Pi_{Q,h}p, v_h)}{\|v_h\|_V} + \sup_{v_h \in V_h} \frac{a(u - u_h, v_h)}{\|v_h\|_V} && \text{(2.82c)}\\
&\leq \beta_b\|p - \Pi_{Q,h}p\|_Q + \beta_a\|u - u_h\|_V. && \text{(2.82d)}
\end{aligned}$$

Approximation estimate for an *exact* divergence-free velocity

There are three different ways to classify a divergence-free velocity, which will be discussed in section 2.3.3. In special case of an *exact* divergence-free velocity, the bilinear form in Eq. (2.80c) leads to

$$b(p_h - p, v_h - \Pi_{V,h}) = (p_h - p, \nabla \cdot (v_h - \Pi_{V,h}u)) = 0. \tag{2.83}$$

The interpolation error for the velocity in Eq. (2.81) simplifies, so that there is no pressure dependency

$$\beta_{a,h}\|v_h - \Pi_{V,h}u\|_V \leq \beta_a\|u - \Pi_{V,h}u\|_V. \tag{2.84}$$

Conclusions and remarks:

Exact divergence-free finite element spaces have according to Eq. (2.84) the advantage that the error estimate of the velocity is not pressure dependent. In practice, using divergence-free finite element approaches leads to a bunch of other problems. One example is the *exact* divergence-free Raviat-Thomas element in [62], which is inf-sup stable, but a non-conforming triangle element. According to Strang's lemma, new error term arises, e.g. a nonconformity error, which does not lead to a real simplification. Another example is the *exact* divergence-free Scott-Vogelius element in [72] and [81], which is a conformal element, but also a triangle element. Especially in the 3D case and by using multigrid techniques, the usage of these finite element spaces is complicated to handle. A simple partitioning of tetrahedral elements onto a coarser grid is also not trivial.

In this section 2.2, the well-posedness and conformal discretization for Hilbert spaces $V_h \subset V$ as well as $Q_h \subset Q$ was treated. In the next section 2.3, the practical application using FEM is described. Finite elements on quadrilaterals are used, that leads to a straightforward implementation in 3D and can be easily partitioned on coarser grids, which is important for the multigrid techniques discussed in section 3.2.3.

With the help of the Fortin operator finite elements can be created. A finite element pair is inf-sup stable, if and only if there exist a Fortin projection operator $\Pi_{V,h}$ for V_h as well as an operator $\Pi_{Q,h}$ for Q_h. Details of the construction of a Fortin operator for the two dimensional Taylor-Hood element can be found in [18].

2.3 Finite element method (FEM)

The common structure of solvers for the saddle point problem consisting of Eq. (2.54a) and Eq. (2.54b) is:

i. To use a discretization in space, like the finite element method as discussed in section 2.3.1.

ii. To choose a finite element basis, which is briefly investigated in section 2.3.2.

iii. To apply a suitable time discretization, which is briefly described in section 2.3.4.

iv. To solve the algebraic problem efficiently, which is discussed in chapter 3.

For more detailed literature on FEM see [13], [67], [14] and [23].
In practice and in this thesis, only L^2 functions appear. The domain and the given right-hand-side g is decent, i.e., always L^2 integrable. It holds:

$$V := (H_0^1(\Omega))^d,\ Q := L_0^2(\Omega),\ g \in (L^2(\Omega))^d. \tag{2.85}$$

2.3.1 Discretization in space

The basic idea is a partition of a given domain $\Omega \subset \mathbb{R}^d$, $d = 2, 3$ into a set of simple sub domains, so-called finite elements, illustrated in figure 2.3.

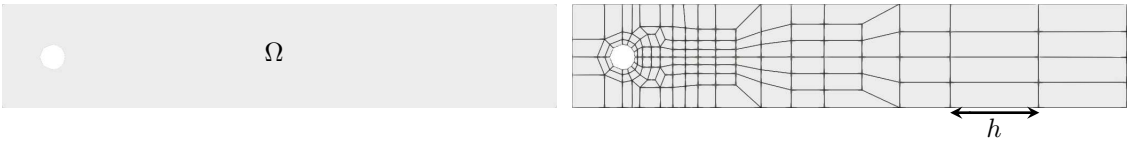


Figure 2.3: Flow-around-cylinder domain (left). A possible quadrilateral decomposition of the flow-around-a-cylinder domain (right).

There are different domain decomposition methods, which are analyzed in [80]. In this thesis, all triangulations are regular, with quadrilateral as finite elements.

Definition 2.6: Regular triangulation

Let Ω be a domain with Lipschitz boundary.

A triangulation $T_h(\Omega)$ is called regular, if

i. $T_h(\Omega) = \{T\}$, where T is a non-degenerate quadrilateral, $T = \overline{T}$,

ii. $\Omega = \bigcup_{T \in T_h(\Omega)} T$,

iii. the intersection of two different elements is either empty, one edge or one face of both elements.

The discretization parameter h stands for the mesh size, denoted by

$$h := \max_{T \in T_h(\Omega)} h_T, \tag{2.86}$$

where $h_T := \operatorname{diam}(T)$ is defined as the diameter of the element T. The idea is that for a fine mesh, in the case $h \to 0$, convergence to the solution of the continuous problem will be achieved.

The finite element method is characterized by a special choice of $V_h \subset V := (H_0^1(\Omega))^d$ and $Q_h \subset Q := L_0^2(\Omega)$. The subspaces V_h and Q_h are called ansatz space and are chosen such that, restricted to each element T, functions in V_h and Q_h are polynomials, i.e.,

$$V_h := \left\{ v_h : v_h \in C^0(\Omega), v_{h|_T} \in \mathbb{P}^k(T), T \in T_h \right\}, \tag{2.87a}$$

$$Q_h := \left\{ q_h : q_h \in C^0(\Omega), q_{h|_T} \in \mathbb{P}^{k-1}(T), T \in T_h \right\}, \tag{2.87b}$$

with k as highest polynomial degree. Like any polynomial, these are uniquely determined by a number of interpolation conditions, which are also referred to as the degrees of freedom of the element. To fulfill the well-posedness from section 2.2, the number of the degrees of freedom from the discrete pressure space Q_h need to be inferior. Mathematically, this means that

$$\dim Q_h =: d_Q \leq d_V := \dim V_h, \tag{2.88}$$

must be satisfied, otherwise the determinant of the saddle point system in Eq. (2.55) is zero. Furthermore, the discrete pressure space need to be large enough, to fulfill the divergence-free condition, i.e. the conservation of mass principle in Eq. (2.6).

To obtain a sparse matrix, it is necessary to construct the basis functions $\phi_1, \ldots, \phi_{d_V^{\text{loc}}}$ and $\psi_1, \ldots, \psi_{d_Q^{\text{loc}}}$ in such a way, that they have only a very local support. The nodal basis is denoted as

$$\Phi := \left\{ \phi_i : \phi_i(x_{V,i}) = \delta_{ik}, i, k = 1, \ldots, d_V^{\text{loc}} \right\}, \tag{2.89a}$$

$$\Psi := \left\{ \psi_i : \psi_i(x_{Q,i}) = \delta_{ik}, i, k = 1, \ldots, d_Q^{\text{loc}} \right\}, \tag{2.89b}$$

where d_V^{loc} and d_Q^{loc} are the local degree of freedom, i.e. the number of grid points in one element T at locations $x_{V,i}$ for the space V_h as well as $x_{Q,i}$ for the space Q_h. The nodal basis is chosen such that a element of the basis are non-zero only in (the neighbourhood of) one of the elements.

For a given basis $\phi_1, \ldots, \phi_{d_V^{\text{loc}}}$ and $\psi_1, \ldots, \psi_{d_Q^{\text{loc}}}$, there exists

$$u_h := \sum_{T \in T_h} \sum_{i=1}^{d_V^{\text{loc}}} u_{\pi_V(i)} \phi_i = \sum_{i=1}^{d_V} u_i \phi_i, \tag{2.90a}$$

$$p_h := \sum_{T \in T_h} \sum_{i=1}^{d_Q^{\text{loc}}} p_{\pi_Q(i)} \psi_i = \sum_{i=1}^{d_Q} p_i \psi_i, \tag{2.90b}$$

with an injective function $\pi_V : d_V^{\text{loc}} \to d_V$ and $\pi_Q : d_Q^{\text{loc}} \to d_Q$ that assigns a unique global number to each local degree of freedom.

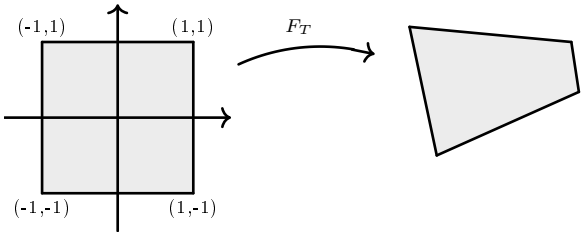


Figure 2.4: Reference element $\hat{T}$ on the domain $[-1,-1] \times [1,1]$ is transformed using the affine transformation $F_T : \mathbb{R}^d \to \mathbb{R}^d$.

The reference element $\hat{T}$ with the base $\hat{\phi}_1, \ldots, \hat{\phi}_{d_V^{\text{loc}}}$ and $\hat{\psi}_1, \ldots, \hat{\psi}_{d_Q^{\text{loc}}}$ can be transformed to any quadrilateral using the affine transformation F_T, which is illustrated in figure 2.4. Mathematically, it reads as:

$$v_h(F_T(\hat{x})) = \sum_{T \in T_h} \sum_{i=1}^{d_V^{\text{loc}}} u_{\pi_V(i)} \hat{\phi}_i(\hat{x}), \tag{2.91a}$$

$$q_h(F_T(\hat{x})) = \sum_{T \in T_h} \sum_{i=1}^{d_Q^{\text{loc}}} p_{\pi_Q(i)} \hat{\psi}_i(\hat{x}). \tag{2.91b}$$

Matrix formulation (setting up a linear equation system):

The discrete saddle point problem resulting from Eqs. (2.70) can be written as an algebraic problem that can be solved on a computer. The test function $v_h \in V_h$ can be replaced by choosing a basis $\phi_1, \ldots, \phi_{d_V^{\text{loc}}}$ and also $q_h \in Q_h$ can be replaced by choosing a basis $\psi_1, \ldots, \psi_{d_Q^{\text{loc}}}$. For all $j = 1, \ldots, d_V^{\text{loc}}$ and $l = 1, \ldots, d_Q^{\text{loc}}$ holds

$$a(u_h, \phi_j) + b(p_h, \phi_j) = \langle g, \phi_j \rangle. \tag{2.92}$$

Using Eqs. (2.90) leads to

$$a\left(\sum_{i=1}^{d_V} u_i \phi_i, \phi_j\right) + b\left(\sum_{i=1}^{d_Q} p_i \psi_i, \phi_j\right) = \langle g, \phi_j \rangle. \tag{2.93}$$

Extracting the sum in front of the bilinear form gives

$$\sum_{i=1}^{d_V} a(\phi_i, \phi_j) u_i + \sum_{i=1}^{d_Q} b(\psi_i, \phi_j) p_i = \langle g, \phi_j \rangle, \tag{2.94}$$

with the weak form of the Navier-Stokes equations definition 2.5

$$\sum_{i=1}^{d_V} a(\phi_i, \phi_j) u_i = \sum_{i=1}^{d_V} \left((\phi_i, \phi_j) \partial_t u_i + (u_h \cdot \nabla \phi_i, \phi_j) u_i + \nu (\nabla \phi_i, \nabla \phi_j) u_i \right) \tag{2.95a}$$

$$= \sum_{i=1}^{d_V} \left(\mathbf{M}_{ij} \partial_t u_i + \mathbf{K}(u_h)_{ij} u_i + \nu \mathbf{L}_{ij} u_i \right), \tag{2.95b}$$

$$\sum_{i=1}^{d_Q} b(\psi_i, \phi_j) p_i = -\sum_{i=1}^{d_Q} (\psi_i, \nabla \cdot \phi_j) p_i \tag{2.95c}$$

$$= \sum_{i=1}^{d_Q} \mathbf{B}_{ij} p_i, \tag{2.95d}$$

where $\mathbf{M}_{ij} := (\phi_i, \phi_j)$ is the mass matrix, $\mathbf{L}_{ij} := (\nabla \phi_i, \nabla \phi_j)$ represents the Laplacian matrix, $\mathbf{B}_{ij} := -(\psi_i, \nabla \cdot \phi_j)$ shows the gradient matrix, as well as $\mathbf{B}^\top$ is the divergence matrix. The transport matrix is defined by $\mathbf{K}(u_h)_{ij} := (u_h \cdot \nabla \phi_i, \phi_j)$, which depends on the solution itself due to the non-linearity $u_h \cdot \nabla u_h$. In case of Stokes-Oseen equations, it leads to a u_h independent transport matrix $\mathbf{K}_{ij} := (\overline{u} \cdot \nabla \phi_i, \phi_j)$.

Next, assemble the right-hand-side

$$\boldsymbol{g}_j := \int_\Omega g\phi_j \approx \sum_{T\in T_h} Q_T(g\phi_j) \tag{2.96}$$

of the linear system of equations. Here, integrals over non-trivial functions occur, which can only be calculated with the help of a numerical quadrature Q_T.

With Vectors $\mathbf{u} \in \mathbb{R}^{d_V}$ and $\mathbf{p} \in \mathbb{R}^{d_Q}$, it follows for the instationary case

$$\begin{bmatrix} \mathbf{M} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{\partial}{\partial t}\mathbf{u} \\ 0 \end{bmatrix} + \begin{bmatrix} \mathbf{K}(\mathbf{u}) + \nu\mathbf{L} & \mathbf{B} \\ \mathbf{B}^\top & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \boldsymbol{g} \\ 0 \end{bmatrix}, \tag{2.97}$$

and for the stationary case

$$\begin{bmatrix} \mathbf{K}(\mathbf{u}) + \nu\mathbf{L} & \mathbf{B} \\ \mathbf{B}^\top & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \boldsymbol{g} \\ 0 \end{bmatrix}. \tag{2.98}$$

2.3.2 Finite elements (quadrilateral elements)

Construction of a finite element space can also be found in [67], chapter 3.

Popular finite elements for incompressible flow problems are the Q_2/P_1^{disc} element, as well as the Q_2/Q_1 element, which is well known as Taylor-Hood element. Both finite element pairs are stable, it means they fulfill the inf-sup condition from section 2.2.

The Q_2/P_1^{disc} element is based on a biquadratic approximation for velocity and a piecewise linear approximation for pressure. The Taylor-Hood element is based on a biquadratic approximation for velocity and a linear approximation for pressure.

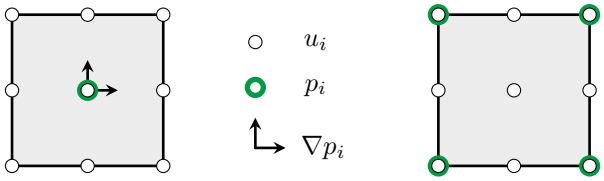


Figure 2.5: DOFs in 2D for the element Q_2/P_1^{disc} (left) and Taylor-Hood, Q_2/Q_1 (right). White circles denote DOFs for horizontal and vertical velocity, respectively. Green circles and arrows denote pressure and pressure gradient DOFs.

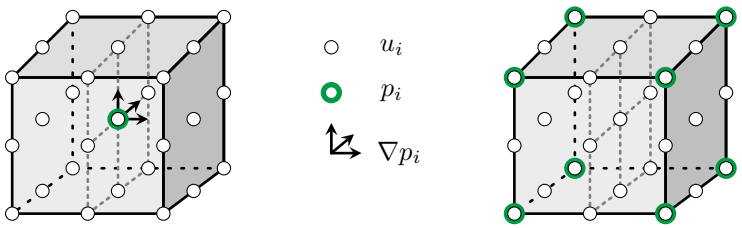


Figure 2.6: DOFs in 3D for the element Q_2/P_1^{disc} (left) and Taylor-Hood, Q_2/Q_1 (right). White circles denote DOFs for horizontal and vertical velocity, respectively. Green circles and arrows denote pressure and pressure gradient DOFs.

Discrete velocity space: Q_2-element

The degrees of freedom (DOFs) for the Q_2 element are shown in figure 2.5 in the 2D case and in figure 2.6 for the 3D case. The 1D basis function on the reference interval are defined by

$$\tilde{\phi}_1(x) := -\frac{1}{2}x(1-x), \quad \tilde{\phi}_2(x) := (1-x)(1+x), \quad \tilde{\phi}_3(x) := \frac{1}{2}(1+x)x. \tag{2.99}$$

at the interpolation points $x_{V,i}$, $i = 1, \ldots, d_V^{\text{loc}}$.

In 2D and 3D, the tensor product of Eq. (2.99) is used to get the corresponding basis function. The dimension of Q_2 in $d = 2$ is $d_V = 9$, which is also the number of the basis functions. The nine basis functions are

$$\phi := \{\tilde{\phi}_i(x)\tilde{\phi}_j(y), \text{ for } 1 \leq i, j \leq 3\}, \tag{2.100}$$

and in $d = 3$ the dimension is $d_V = 27$ with

$$\phi := \{\tilde{\phi}_i(x)\tilde{\phi}_j(y)\tilde{\phi}_l(z), \text{ for } 1 \leq i, j, l \leq 3\}. \tag{2.101}$$

Discrete pressure space: Q_1-element

The DOFs for the Q_1 element are illustrated in figure 2.5 in the 2D case and in figure 2.6 for the 3D case. The 1D local basis functions, which will be used for the tensor product, are given by

$$\tilde{\psi}_1(x) := \frac{1}{2}(1-x), \quad \tilde{\psi}_2(x) := \frac{1}{2}(1+x). \tag{2.102}$$

With these functions, e.g., the basis functions in 2D are computed by $d_Q = 4$ like in Eq. (2.100) and also for $d = 3$ $d_Q = 8$ like in Eq. (2.101).

Discrete pressure space: P_1^{disc}-element

The DOFs for the P_1^{disc} element are illustrated in figure 2.5 in the 2D case and in figure 2.6 for the 3D case. The special feature of this element is that there are no DOFs on the corners and edges.

The P_1^{disc}-element is $H(div)$-conform and therefore fulfills theorem 2.12:

Theorem 2.12: H(div) conform (sufficient condition)

A finite element space $V_h \subset (L^2(\Omega))^d$ is called $H(div)$-conform if and only if

$$[v_h \cdot n]_{|E} = 0 \quad \text{for all } v_h \in V_h \text{ and all faces } E \text{ of } T_h, \tag{2.103}$$

where $[\,\cdot\,]$ denotes the jump term and n the outward normal vector.

The definition of the $H(div)$ space can be found in [22], chapter 2, and reads as

$$H(div) := \left\{v_h : v_h \in V_h \text{ such that } \nabla \cdot v_h \in L^2(\Omega)\right\}. \tag{2.104}$$

with the norm

$$\|v_h\|_{div}^2 := |v_h|_0^2 + |\nabla \cdot v_h|_0^2. \tag{2.105}$$

The advantage of a $H(div)$-conform finite element space is that the error over the edges disappears.

According to Gauss' divergence theorem, for any $p \in H^1(\Omega)$ holds

$$(\nabla p, v_h) = -(p, \nabla \cdot v_h) - (p, v_h \cdot n)_E \tag{2.106a}$$
$$= -(p, \nabla \cdot v_h), \tag{2.106b}$$

Eq. (2.106b) applies, since theorem 2.13 leads to $v_h \cdot n|_E = 0$.

The Q_2/P_1^{disc} is a relatively late discovered element. It was invented around a blackboard at the Banff Conference on Finite Elements in Flow Problems in the year 1979, see [31], chapter 6. Tests in [59] have shown, that the Q_2/P_1^{disc} element being of higher order, yields higher accuracy and higher order of convergence. 'The Q_2/P_1^{disc} element is still the best choice' is the conclusion from [59]. It is also handled as a cure of the instability of the Q_2/Q_1 element, which will be discussed more in detail in upcoming section 2.3.3.

2.3.3 Types of divergence-free functions (consistency error)

In contrast to the Q_2/P_1^{disc} element, the Taylor-Hood element is not $H(div)$ conform. Therefore, the Q_2/Q_1 element has components in the pressure that behave non-physical. The solution p_h of the discrete saddle-point problem in Eq. (2.55) can be changed to $p_h + s_h$, with $s_h \in S_h$, where

$$S_h := \ker B_h \setminus \ker B. \tag{2.107}$$

is known as *spurious mode*, see [22] in chapter 8.10. Locally oscillating parts emerge in the pressure solution of the Taylor-Hood element. The physical interpretation is that there is a violation of mass conservation from Eq. (2.6). Spurious modes correspond to null singular values, see [22] in section 5.6.2. In [31] it is shown that the existence of spurious modes is in many cases strongly mesh dependent.

In theorem 2.13 it is shown that the discrete divergence-free space Z_h may not fulfill the incompressibility equation Eq. (2.6).

Theorem 2.13: Discrete divergence-free function

In general a discrete divergence-free function may not be exactly divergence-free:

$$Z_h := \ker(B_h^*) := \{v_h \in V_h : (q_h, \nabla \cdot v_h) = 0 \text{ for all } q_h \in Q_h\} \tag{2.108a}$$
$$\not\subseteq \{v \in V : (q, \nabla \cdot v) = 0 \text{ for all } q \in Q\} =: \ker(B^*) =: Z. \tag{2.108b}$$

A proof of the above theorem 2.13 can be found in [30] chapter §II.2, and [22] chapter 5.3.

Furthermore, divergence-free functions must be classified more precisely. The following types exist:

i. *Discrete* divergence-free function, which is fulfilled by the Taylor-Hood and the Q_2/P_1^{disc} element pair. By theorem 2.13,

$$(q_h, \nabla \cdot v_h) = 0 \quad \text{for all } q_h \in Q_h \tag{2.109}$$

does hold in the discrete formulation, but this does not satisfy the continuous formulation in general.

ii. *Element-wise* divergence-free function denotes that the divergence-free condition for each element is satisfied by

$$\nabla \cdot v_{h_{|_T}} = 0, \quad \text{for all } T \in T_h. \tag{2.110}$$

iii. The velocity is called *exact* divergence-free, if element-wise the divergence-free condition is fulfilled with the $H(div)$ property, i.e.

$$\nabla \cdot v_h = 0 \quad \text{in } L^2(\Omega). \tag{2.111}$$

The Q_2/P_1^{disc} element is $H(div)$ conform, but does not have the property of being element-wise divergence-free.

The Taylor-Hood element is also not element-wise divergence-free, and in addition does not satisfy the $H(div)$ conformity in theorem 2.12, so that the errors on the edges in Eq. (2.106a) do not vanish.

Numerical results of divergence-free quadrilateral elements were presented in [70] and [79], which allow the use of multigrid methods, especially in 3D. However, divergence-free elements are very effortful. The focus of the numerical results in chapter 4 is on a fast solver that uses the Q_1 element for the pressure. A comparison with the superior Q_2/P_1^{disc} element is sufficient. The divergence-free velocity in the numerical solvers in chapter 3 and chapter 4 can be achieved by an additional divergence-free update instead of using special divergence-free finite elements.

Derivation of the consistency error

Next, the consistency error is evaluated, this means checking whether the approximation actually solves the given problem and not some other problem. To check the consistency error, the approximation is tested with the exact solution:

In [14] chapter III §1, the lemma from Strang is discussed. It reads as

$$\|u - u_h\|_{H_h^1(\Omega)} \leq C \left(\inf_{v_h \in V_h} \|u - v_h\|_{1,h} + \sup_{v_h \in V_h} \frac{1}{\|v_h\|_h} |\kappa_h(v_h)| \right) \tag{2.112}$$

with a constant c independent of h. The first term is the approximation error and the second term is the consistency error. For the saddle point system follows

$$\begin{aligned} &\|u - u_h\|_{H_h^1(\Omega)} + \beta_h \|p - p_h\|_{L^2(\Omega)} \\ &\quad \leq c \left(\inf_{v_h \in V_h} \|u - v_h\|_{1,h} + \inf_{q_h \in Q_h} \|p - q_h\|_{L^2(\Omega)} + \sup_{v_h \in V_h} \frac{1}{\|v_h\|_h} |\kappa_h(v_h)| \right), \end{aligned} \tag{2.113}$$

with $\beta_h := \overline{\beta}(\inf_{T \in T_h} h_T)$.

The consistency error for the exact solution (u, p) and $v_h \in V_h$ is given by

$$\kappa_h(v_h) := (a(u, v_h) - b(p, v_h)) - (a_h(u, v_h) - b_h(p, v_h)), \tag{2.114}$$

where $a_h(u, v_h)$ and $b_h(p, v_h)$ denotes the discrete bilinear form, i.e., the sum over all finite elements T:

$$\kappa_h(v_h) = (-\nu \Delta u + (u \cdot \nabla) u + \nabla p, v_h) - \left(\sum_{T \in T_h} \left(-\nu \Delta u + (u \cdot \nabla) u + \nabla p \right), v_h \right). \tag{2.115}$$

After applying Gauss's divergence theorem for the Laplacian $-\nu \Delta u$ and also for the pressure ∇p, it reads

$$\kappa_h(v_h) = -\nu \sum_E (\partial_n u, [v_h]_E)_E + \sum_E (p, [v_h \cdot n]_E)_E - \sum_{T \in T_h} (p - \Pi_Q p, \nabla \cdot v_h)_T. \tag{2.116}$$

For better overview, the following abbreviations are introduced:

$$\kappa_h^v(v_h) := \sum_E \langle \partial_n u, [v_h]_E \rangle_E, \tag{2.117a}$$

$$\kappa_h^{p_1}(v_h) := \sum_E \langle p, [v_h \cdot n]_E \rangle_E, \quad \kappa_h^{p_2}(v_h) := \sum_{T \in T_h} (p - \Pi_h p, \nabla \cdot v_h)_T. \tag{2.117b}$$

Consequently, a sufficient condition for the convergence is the consistency condition:

$$\lim_{h \to 0} \sup_{v_h \in V_h} \frac{1}{\|v_h\|_h} \kappa_h(v_h) = 0 \tag{2.118}$$

Consistency error with Q_2/Q_1-element

The consistency error for the Taylor-Hood element reads as

$$\begin{aligned} &\|u - u_h\|_{H_h^1(\Omega)} + \beta_h \|p - p_h\|_{L^2(\Omega)} \\ &\quad \leq C \Bigg(\inf_{v_h \in V_h} \|u - v_h\|_{1,h} + \inf_{q_h \in Q_h} \|p - q_h\|_{L^2(\Omega)} \\ &\qquad + \sup_{v_h \in V_h^{div}} \frac{1}{\|v_h\|_1} \left(\kappa_h^v(v_h) + \frac{1}{\nu} \kappa_h^{p_1}(v_h) + \frac{1}{\nu} \kappa_h^{p_2}(v_h) \right) \Bigg) \end{aligned} \tag{2.119}$$

Eq. (2.119) shows that numerical problems occur for $\nu \to 0$. This mean, that for a small viscosity ν or in other words for a high Reynolds number $Re \approx \frac{1}{\nu}$ a stabilization is needed.

The root of the problem is the not conform gradient and divergence space. In section 3.3.3 a stabilization method is presented which is a penalty for a not exact ∇-div space.

Consistency error with $Q_2/P_1^{\mathbf{disc}}$-element

For the Q_2/P_1^{disc} element follows due to the $H(div)$ conformity theorem 2.12 that $\kappa_h^{p_1} = 0$. It holds

$$\|u - u_h\|_{H_h^1(\Omega)} \leq C \left(\inf_{v_h \in V_h} \|u - v_h\|_{1,h} + \sup_{v_h \in V_h^{div}} \frac{1}{\|v_h\|_1} \left(\kappa_h^v(v_h) + \frac{1}{\nu} \kappa_h^{p2}(v_h) \right) \right). \tag{2.120}$$

In addition, less pressure error terms occur in the consistency error compared to the Taylor-Hood element in Eq. (2.119) especially for a high Reynolds number. In the case of Navier-Stokes a stabilization, like SUOG or EO-FEM, for the convective term is needed as a remedy to solve problems with high Reynolds number.

2.3.4 Discretization in time

For the time discretization of the time derivative in the saddle point problem Eq. (2.97), it is common to use the well-known θ-method, which results for the momentum equations in:

$$\mathbf{M} \frac{\mathbf{u}^l - \mathbf{u}^{l-1}}{k} + (\mathbf{K}(\mathbf{u}^l) + \nu \mathbf{L}) \theta \mathbf{u}^l + (\mathbf{K}(\mathbf{u}^{l-1}) + \nu \mathbf{L})(1 - \theta \mathbf{u}^{l-1}) = \theta \boldsymbol{g}^l + (1 - \theta) \boldsymbol{g}^{l-1} \tag{2.121}$$

at a given time l with time-step $k := \Delta t$ and given starting value u_0 and p_0. Taking all known parameters on the right-hand-side and multiplying by the time-step k leads to a new right-hand-side defined by

$$\mathbf{g}^l := k\theta \boldsymbol{g}^l + k(1 - \theta) \boldsymbol{g}^{l-1} + \mathbf{M} \mathbf{u}^{l-1} - k(\mathbf{K}(\mathbf{u}^{l-1}) + \nu \mathbf{L})(1 - \theta) \mathbf{u}^{l-1}, \tag{2.122}$$

and the system matrix defined by

$$\mathbf{S}(\mathbf{u}^l) := \mathbf{M} + k\theta(\mathbf{K}(\mathbf{u}^l) + \nu\mathbf{L}). \tag{2.123}$$

The parameter θ in Eq. (2.122) and in Eq. (2.123) is chosen between 0 and 1. For $\theta = 0.5$ the method yields the Crank-Nicolson method and for $\theta := 1$ holds the backward Euler scheme.

The resulting algebraic system can be solved for each time step l on a computer and reads as

$$\begin{bmatrix} \mathbf{S}(\mathbf{u}^l) & k\mathbf{B} \\ \mathbf{B}^\top & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u}^l \\ \mathbf{p}^l \end{bmatrix} = \begin{bmatrix} \mathbf{g}^l \\ 0 \end{bmatrix}. \tag{2.124}$$

Stability

The backward Euler scheme is of order one. The Crank-Nicolson method has order two and a higher accuracy, but oscillations may occur. The oscillations that occur in the Crank-Nicolson scheme can be explained due to the stability. The Crank-Nicolson method and backward Euler scheme have A-stability, which means that the stability of the time iteration is ensured by

$$\sup_{l\geq 0} |\mathbf{u}^l| < \infty. \tag{2.125}$$

Compared to the Crank-Nicolson method, the backward Euler method has further stabilization properties, which ensures the boundedness of the discrete solution even in the case of inhomogeneous right sides, i.e.,

$$\sup_{l\geq 0} |\mathbf{u}^l| < c \sup_{l\geq 0} |\mathbf{g}^l|, \tag{2.126}$$

with a constant $c > 0$. Furthermore, the backward Euler scheme ensures the (exponential) attenuation of high-frequency solution components and makes the method robust against local disturbances of the data. Therefore, the backward Euler method is also referred to as strong A-stable.

For more information about the time stepping schemes and their stability properties see [60] chapter 5.1.

2.4 Boundary and initial conditions

The most common boundary conditions in fluid problems are inlet boundary conditions defined by $\partial\Omega_{\text{in}}$, outlet boundary conditions denoted by $\partial\Omega_{\text{out}}$ and no-slip boundary conditions denoted by $\partial\Omega_0$, which are illustrated e.g. in figure 2.7. It holds $\partial\Omega := \partial\Omega_{\text{in}} \cup \partial\Omega_0 \cup \partial\Omega_{\text{out}}$.

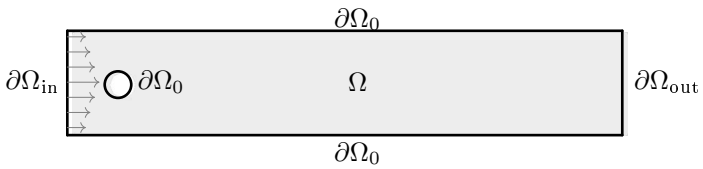


Figure 2.7: Boundary conditions of the flow-around-a-cylinder domain for $d = 2$.

2.4.1 Boundary values (filter)

In the FEAT3 software, the boundary values are not written in the matrices, instead the solution vectors are filtered. Furthermore, the boundary conditions resp. filters only applies to the velocity solution. No filters are set on the pressure.

In case of the flow-around-a-cylinder benchmark, Dirichlet boundary conditions are given on $\partial\Omega_D := \partial\Omega_{\text{in}} \cup \partial\Omega_0$, which means that the corresponding entries of the velocity vectors are set (filtered) as

$$\mathbf{u}_{|\partial\Omega_{\text{in}}} := u_{\text{in}} \quad \text{for the inlet boundary,} \tag{2.127a}$$

$$\mathbf{u}_{|\partial\Omega_0} := 0 \quad \text{for a solid wall,} \tag{2.127b}$$

with a given function u_{in}.

Outlet boundary condition (do nothing)

Due to the partial integration in the weak formulation in Eqs. (2.49) the boundary condition

$$-(\nu\nabla u_h \cdot n, v_h)_{L^2(\partial\Omega_{\text{out}})} + (\Pi_Q p, v_h \cdot n)_{L^2(\partial\Omega_{\text{out}})} = 0 \tag{2.128}$$

need to be fulfilled on the outlet boundary $\partial\Omega_{\text{out}}$.

In this thesis, the most simple outflow boundary condition for the outflow boundary is used, which is know as *do-nothing* or *free-stream* condition. This condition seems most natural, since it leave the solution and the test space free on $\partial\Omega_{\text{out}}$. Also, in most of the cases, a non-physical behavior on the outlet boundary is prevented. A further discussion about the *do-nothing* condition can be found in [61].

2.4.2 Well-posedness (divergence-free velocity)

To get a well-posed problem, in the sense of Hadamard, the saddle point problem Eq. (2.124) need to have suitable (smooth) initial conditions for $\mathbf{u}^0, \mathbf{p}^0$ and boundary conditions on $\partial\Omega$.

Boundary defect in case of a decoupled solver approach

An divergence-free velocity is needed to satisfy the continuity equation in Eq. (2.6) and therefore to fulfill Brezzi's sufficient condition for well-posedness in theorem 2.8. In case of a decoupled solver approach, like the PP-solver in section 3.2, a divergence-free update for the velocity in the form of

$$u = \tilde{u} + k\nabla p \tag{2.129}$$

is performed. The divergence-free velocity update in Eq. (2.129) is fulfilled if p satisfies the condition

$$\nabla p \cdot n = 0 \quad \text{on the boundary } \partial\Omega. \tag{2.130}$$

Since the real pressure does not satisfy the boundary conditions in Eq. (2.130), a defect arises in the corners of the domain, which is shown and analyzed in chapter 4.

Chapter 3

Solver for incompressible flow problems

Contents

The main objective is the development and testing of an *accurate*, *robust* and *efficient* numerical method for the solution of certain flow problems arising in computational fluid dynamics (CFD).

- *Accuracy* is the degree of how close the discrete solution is to the continuous solution. The order of accuracy can be calculated due to measuring of the velocity and pressure error in the L^2- and H^1-norm. The error range indicates the accuracy of the solver. In the case of an accurate solver, the discrete solution corresponds to the analytical solution. The analytical velocity and pressure can be calculated directly for the Stokes problem, which is the reason why the next chapter 4 focuses only on the Stokes equations to analyze the defect of the solver approach presented in this chapter.
- A *robust* method means that it is largely independent to the complexity of the geometry and the shape of the underlying mesh and works well even with disturbed grids, such as strong anisotropies or deformations of the individual grid cells. Ideally. the solver should contain only a few "free" parameters, which the user must first adjust to the problem to be calculated before using the method. The derived techniques shall perform independently of the size of involved time stepping parameters, as well as the initial and boundary values and right-hand-side vectors. The PP-solver tested and developed in [78] fulfills the definition of robustness.
- An *efficient* method is an advanced method that is optimized for high-performance-computing (HPC). It creates a solver which can be used in many parallel processes and is optimized to run at workstations and supercomputers to solve in a fast way CFD problems. In order to get a very efficient solver, a time-simultaneous algorithm is presented in section 3.3 that solves rapidly the most time consuming pressure Poisson problem fully parallel-in-time and -space using the MPI-library and is build up from the PP-solver.

Section 3.1 provides an overview of the most common solver approaches. Additionally, a memory-efficient local approach for a time-simultaneous projection method solver is presented. Section 3.3 presents some general concepts for improving the time-simultaneous solver, such as the augmented Lagrangian method in section 3.3.3 and a multigrid in time method of a recoupled system using GMRES in section 3.3.4, to guarantee a divergence-free velocity and thus to ensure the sufficient condition for the existence and uniqueness of a solution by Brezzi in theorem 2.8.

Numerical results for the Crank-Nicolson time stepping scheme using the Stokes equations can be found in chapter 4 and for the incompressible Navier-Stokes equations and 3D simulations see chapter 5.

3.1 Essential difference between the solvers

The common approaches for solving CFD problems like Eq. (2.124) are analyzed in [78]. There are coupled solvers, which are briefly discussed in section 3.1.1, and decoupled solvers, which are explained in section 3.1.2. A time-simultaneous approach that is build up from the decoupled solver is introduced in section 3.1.3. All solver approaches leads to the "same" solutions, if the velocity is sufficient divergence-free.

3.1.1 Local Multilevel-Pressure-Schur-Complement (local MPSC)

This solver approach belongs to the group of Galerkin schemes. In the literature, this type of solver is also known as coupled solution by coupled solver (CC) as well as local Multilevel-Pressure-Schur-Complement (MPSC). It is a local method since the coupled solver is based on solving exactly on subsets or patches. For a preconditioner a Block-Gauß-Seidel or Block-Jacobi iteration is usually used. The fully coupled system in Eq. (2.124) is treated with a Newton-like solver as an outer nonlinear procedure.

The solver has a fully implicit character, which leads to the most accurate and robust time stepping schemes. Furthermore, it is the only variant, which allows a rigorous a-posterori error control.

The weaknesses of this solver approach are:

- Due to the design of the preconditioners, only large time steps can be performed.
- It takes a long computation time to solve exactly. Therefore, the solver has high costs for one time step.

For further informations, see [78] chapter 2.4.

3.1.2 Global Multilevel-Pressure-Schur-Complement (global MPSC)

The decoupled solver belongs to the group of projection schemes. The idea originally proposed by Chorin and Temam and is sometimes referred to as the Chorin-Temam method [20], [21], [75]. In the literature, the solver is also known as Chorin's projection method, global MPSC as well as projection solution by projection solvers (PP). It is a global method since the solver solves globally on the domain. Accordingly, the preconditioners are also defined globally, see section 3.2.1.

The solver can be viewed as a predictor-corrector strategy aimed at uncoupling viscous diffusion and incompressibility effects. A time step is composed of three sub-steps:

i. In the first sub-step, the pressure is made explicit and a provisional velocity field $\tilde{\mathbf{u}}$ is computed using the momentum equations.

ii. In the second step, the provisional velocity field is projected onto the space of incompressible vector fields by applying the divergence matrix $\mathbf{B}^\top$ onto the velocity field.

iii. The pressure Poisson equation needs to be solved and the pressure is updated.

The solver has an outer decoupling of the velocity and the pressure. Instead of one system, two decoupled problems are solved approximately. Furthermore, only one outer iteration is performed per time step. The main idea is that the errors that occur due to the decoupling process are not greater than the errors that occur in any case, like the approximation errors in Eq. (2.76) and in Eq. (2.77) or the consistency error in Eq. (2.119).

The velocity and pressure problems are treated with multigrid solvers, which are discussed in section 3.2.3. Newton-like methods are used for the momentum equations. It means that in the case of the incompressible Navier-Stokes equations, this solver provides an exact treatment of the non-linearity. To ensure convergence, a divergence-free update of the velocity is performed at the end of each time step, in the same way as described in Eq. (2.129).

Summary

The biggest drawbacks are:

- The high speed of this solver approach is based on the decoupling with only one outer iteration. Due to the one outer iteration, *accuracy* is only ensured for small time steps.
- The resulting solutions satisfy the discrete momentum equations, i.e. Eq. (2.70a), only approximately.
- In the context of an HPC solver, computation can only be performed in parallel on multiple cores in space. This leads to limited exploitation of the increasingly number of cores available. The divergence-free velocity update in Eq. (2.129) requires the computation of the pressure Poisson problem after each time step, which makes a parallel computation in time not possible.

In terms of HPC there are great advantages:

- Due to the construction of the preconditioners, the solver can perform smaller time steps compared to the local MPSC.
- The solver can perform plenty of cheaper time steps. Specifically, as compared to Galerkin schemes mentioned above in section 3.1.1, hundreds of small PP-time steps are in fact a lot quicker than one CC-time step.
- The additional divergence-free update in Eq. (2.129) is needed to satisfy the continuity equation Eq. (2.6) and therefore to fulfill Brezzi's sufficient condition for well-posedness in theorem 2.8. This update guarantees a full divergence-free velocity. Furthermore, it is only a matrix-vector multiplication that does not require much computing time.

The conclusion from [78] is that for fully non-stationary flows with a dominating convective term and on complex domains, this approach is the favourized one.

More details can be found in [78] section 2.3 and [74].

3.1.3 Time-simultaneous global MPSC (global-in-time approach)

This variant has been newly developed and is based on the above mentioned PP-solver. The idea for this solver was first presented at the end of 2019 by Turek in the talk «Massively Parallel & Low Precision Accelerator Hardware as Trends in HPC - How to use it for large scale simulations allowing high computational, numerical and energy efficiency with application to CFD».

This variant contains a higher decoupling than the PP approach due to the additional decoupling from time. Figure 3.1 illustrates the decomposition of time and space, which is organized as follows:

i. First of all, all momentum equations are solved sequential-in-time.[1]

ii. Then, every pressure Poisson equation is solved fully parallel-in-time.[2]

[1] A parallelization using the new developed multigrid in time approach in [49], [26] and [27] or the parareal approach from [48] is possible and necessary for a fast HPC solver.

[2] See appendix 1 for MPI implementation.

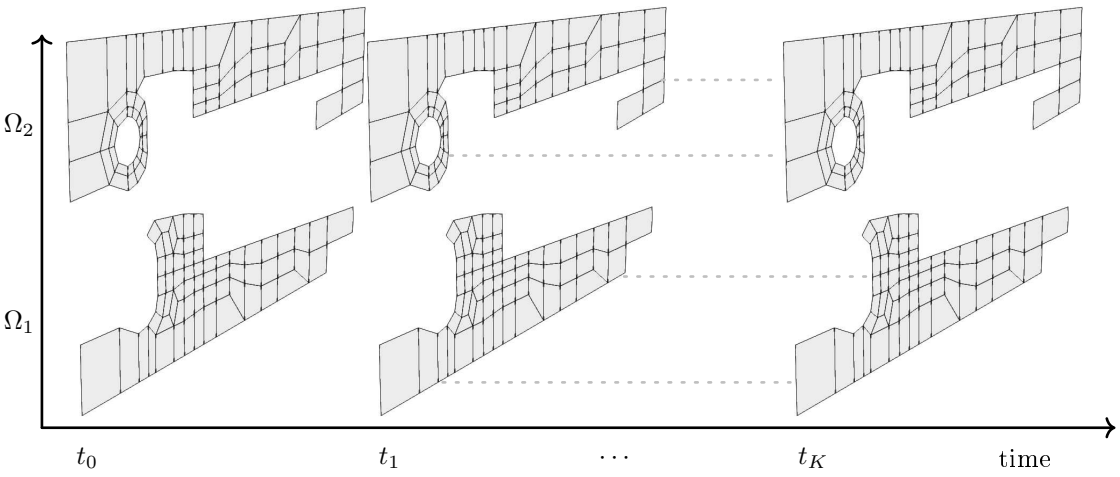


Figure 3.1: Decomposition of space and time.

The decoupled problems are solved in the same fast way like the PP-solver. The most time consuming pressure Poisson equations are completely decoupled from the time and the momentum equations, which allows a rapid treatment of CFD problems by the MPI library. In section 5.3 the parallel performance in time is studied as well as a suggestion for the MPI implementation can be found in the appendix I.

This global-in-time time-simultaneously variant can efficiently handle many cores to rapidly solve CFD problems, but it also requires significantly more memory, which is a problem especially for a wide range of time-steps. The right-hand-side vector of the momentum equations depends on the last pressure iterate, therefore the pressure vectors for each time step must be saved in order to solve the velocity problems. Likewise, the velocity vectors must be stored for each time step to solve the pressure Poisson equations parallel-in-time.

Memory consumption

In order to get a better overview of the memory consumption, the memory requirement per level is calculated in the following. The refinement rank of an finite element grid is called level and denoted by `lvl`. Figure 3.2 shows the asymptotic DOFs per element. Each DOF requires 8 bytes of free space. DOFs shared by elements do not appear in the asymptotics and are therefore not counted. The calculations below are only valid in the asymptotics, i.e. as long as there are not many processes with only a few elements each.

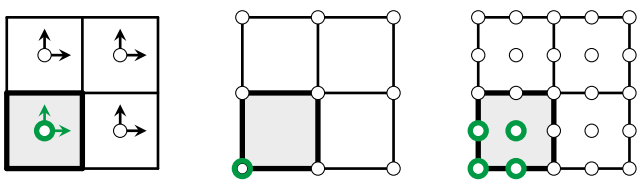


Figure 3.2: Unit square grid on level 1. Highlighted in green are the asymptotically required DOFs per element: P_1^{disc}-element (left), Q_1-element (center) and Q_2-element (right).

In 2D case: On the unit square grid are $4^{\texttt{lvl}}$ elements. The distribution per element is:

- P_1^{disc} element: There are 3 pressure DOFs per element.

- Q_1 element: There is (asymptotically) 1 pressure DOF per element.
- Q_2 element: There are (asymptotically) $2 \times 4 = 8$ velocity DOFs per element.

This leads to in total of $8 + 3 = 11$ DOFs per element in the asymptotics for the Q_2/P_1^{disc} element as well as $8 + 1 = 9$ DOFs for the Q_2/Q_1 element.

Thus, there are $8 \times 11 \times (4^{\texttt{lvl}})$ bytes for the Q_2/P_1^{disc} element and $8 \times 9 \times (4^{\texttt{lvl}})$ bytes for the Q_2/Q_1 element per vector.

In 3D case: On the unit cube grid are $8^{\texttt{lvl}}$ elements. The distribution per element is:

- P_1^{disc} element: There are 4 pressure DOFs per element.
- Q_1 element: There is (asymptotically) 1 pressure DOF.
- Q_2 element: There are (asymptotically) $3 \times 8 = 24$ velocity DOFs.

Asymptotically there are $8 \times 28 \times 8^{\texttt{lvl}}$ bytes for the Q_2/P_1 element or $8 \times 25 \times 8^{\texttt{lvl}}$ bytes for the Q_2/Q_1 element per vector.

In table 3.1, the storage space requirements per time step is listed for the 2D and 3D cases and the levels $\texttt{lvl} = 1, \ldots, 7$. For the 3D case, two different grids were used for the flow-around-a-cylinder benchmark. Figure 3.3 shows the different complexity of the grids. If the cylinder is represented as a cube, only 128 elements are used for the mesh. If the cylinder is to be displayed as a real cylinder, 1772 elements are required for the coarse grid.

d	element	level 1	level 2	level 3	level 4	level 5	level 6	level 7
		unit square grid						
2	Q_2/P_1	0.35 kB	1.41 kB	5.63 kB	22.53 kB	90.11 kB	0.36 MB	1.44 MB
2	Q_2/Q_1	0.29 kB	1.15 kB	4.61 kB	18.43 kB	73.73 kB	0.29 MB	1.18 MB
		unit cube grid						
3	Q_2/P_1	1.79 kB	14.34 kB	0.11 MB	0.92 MB	7.34 MB	58.72 MB	0.47 GB
3	Q_2/Q_1	1.60 kB	12.80 kB	0.10 MB	0.82 MB	6.55 MB	52.43 MB	0.42 GB
		flow-around-a-cylinder grid (130 elements)						
2	Q_2/P_1	45.76 kB	0.18 MB	0.73 MB	2.93 MB	11.71 MB	46.86 MB	0.19 GB
2	Q_2/Q_1	37.44 kB	0.15 MB	0.60 MB	2.40 MB	9.58 MB	38.34 MB	0.15 GB
		flow-around-a-cylinder grid (128 elements)						
3	Q_2/P_1	0.23 MB	1.84 MB	14.68 MB	0.12 GB	0.94 GB	7.52 GB	60.13 GB
3	Q_2/Q_1	0.20 MB	1.64 MB	13.11 MB	0.10 GB	0.84 GB	6.71 GB	53.69 GB
		flow-around-a-cylinder grid (1772 elements)						
3	Q_2/P_1	3.18 MB	25.40 MB	0.20 GB	1.63 GB	13.01 GB	0.10 TB	0.83 TB
3	Q_2/Q_1	2.84 MB	22.68 MB	0.18 GB	1.45 GB	11.61 GB	92.90 GB	0.74 TB

Table 3.1: Additional memory consumption per time step per level.

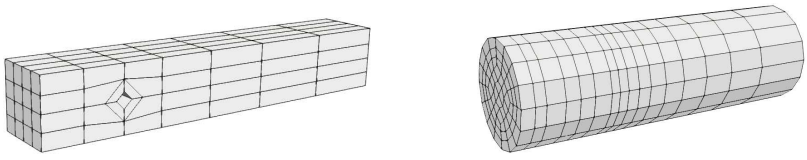


Figure 3.3: Different models for the 3D case for flow-around-a-cylinder on the coarse grid. 128 elements (left) and 1772 elements (right)

Table 3.1 illustrates that especially in 3D and for complex grids, an extremely large amount of memory is required per time step. Independent of the grid or element pair, 300% for 2D and 700% for 3D more memory is required per refinement for storing the velocity and pressure vectors.

Memory for the entire (large) system

In addition to the velocity and pressure vectors listed in table 3.1, further storage is needed for example for the right-hand-side of the momentum and pressure Poisson equations as well as other vectors for saving the results temporarily. Also the corresponding matrices have to be assembled. First, the simplest case is assumed, that for the total number of blocked time steps a (large and sparse) system matrix is assembled for the momentum equations and for the pressure Poisson equation. To get a better understanding of the memory consumption, figure 3.4 and figure 3.5 show approximately the total memory consumption for up to 10000 time steps for the 2D and 3D cases.

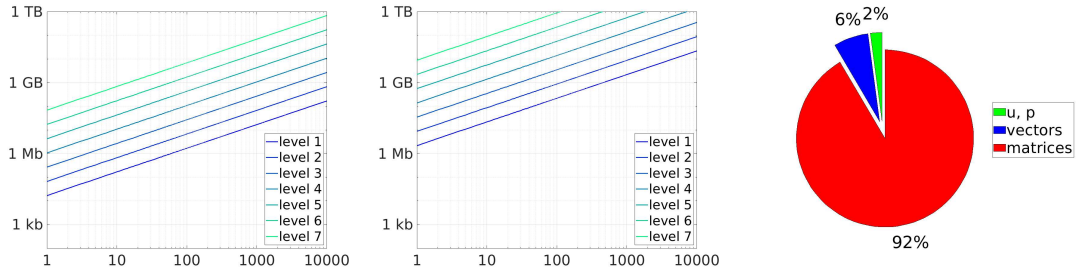


Figure 3.4: 2D memory consumptions for different levels. Visualization of the unit square (left), flow-around-a-cylinder (center), and the memory partitioning (right).

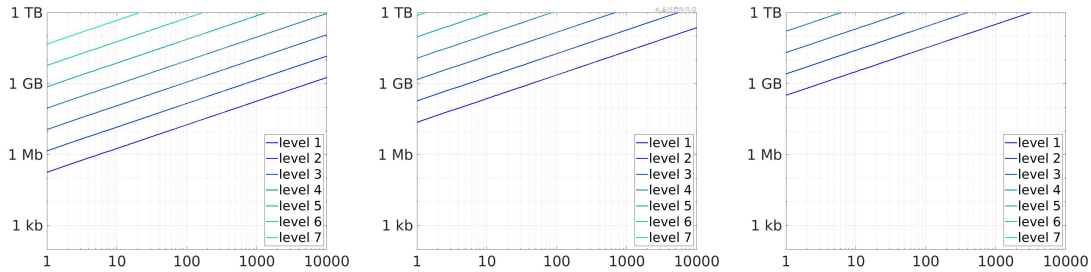


Figure 3.5: 3D memory consumption for different levels. Visualization of the unit cube (left), flow-around-a-cylinder (128 elements) (center), and flow-around-a-cylinder (1772 elements) (right).

It can be seen that the memory space increases linearly over the time. The memory space does not increase further with the number of outer iterations. It should be noted, however, that if the number of time steps is later increased or changed, the (large) system matrix must also be reassembled.

Other time-simultaneous approaches such as the Multigrid Reduction in Time (MGRIT) algorithm from [35] builds up the (large) system matrix and has the disadvantage of having a linear memory gain.

Memory optimization

Since main memory can become an issue, especially for a higher number of blocked time steps, in the 3D case or for complicated grids, a more efficient way must be found. Instead of assembling an expensive (large and dense) system matrix containing all time steps, a more efficient approach is used which assembled matrices containing a single time step. In this case, many small problems are solved. This storage method has also the advantage that additional time steps can be calculated at a later stage.

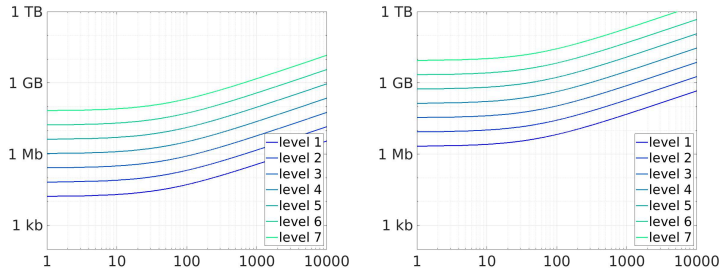


Figure 3.6: 2D memory consumptions for different levels. Visualization of the unit square (left) and flow-around-a-cylinder (right).

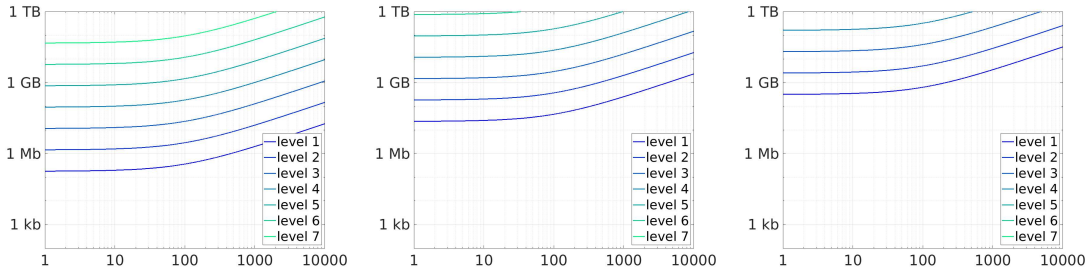


Figure 3.7: 3D memory consumptions for different levels. Visualization of the unit cube (left), flow-around-a-cylinder (128 elements) (center) and flow-around-a-cylinder (1772 elements) (right).

The most memory is required in the first time step, since temporary vectors, such as the right side of the momentum equations or of the pressure Poisson equation, are assembled here. Furthermore, the additional memory for matrices only take place on the first time step.

Comparing figure 3.4 with figure 3.6, it can be seen that a significant amount of memory can be saved. For the memory consumption above, only the fact that the solver works without MPI, with one core, has been taken into account so far. Since the solver is designed for HPC, a useful partitioning for MPI is explained next.

Memory optimization for multiple MPI tasks

Figure 3.8 visualizes the structure of a multicomputer, such as LiDO3. A multicomputer describes a collection of homogeneous independent processors with local memory that communicate over a interconnection network. Assuming the multicomputer has up to a total of $\wp$ cores available, each time step of for the pressure Poisson equation can be computed in parallel on multiple nodes / CPU's, written as $\wp_i$ with $i = 1, \ldots, m$. In addition, each time step is computed in parallel-in-space via domain decomposition on n cores. The solver can be performed on a total number of cores

$$\wp := \{\wp_1, \wp_2, \ldots, \wp_m\} = m \times n. \tag{3.1}$$

The space-time partitioning explained above is advantageous, since no communication outside the local memory space takes place while solving the pressure Poisson equations. This allows an efficient computation because unnecessarily high communication pathways are not required while solving. Only after solving the local vectors it must be synchronized outside the local memory. For an example of an appropriate MPI implementation see appendix I.

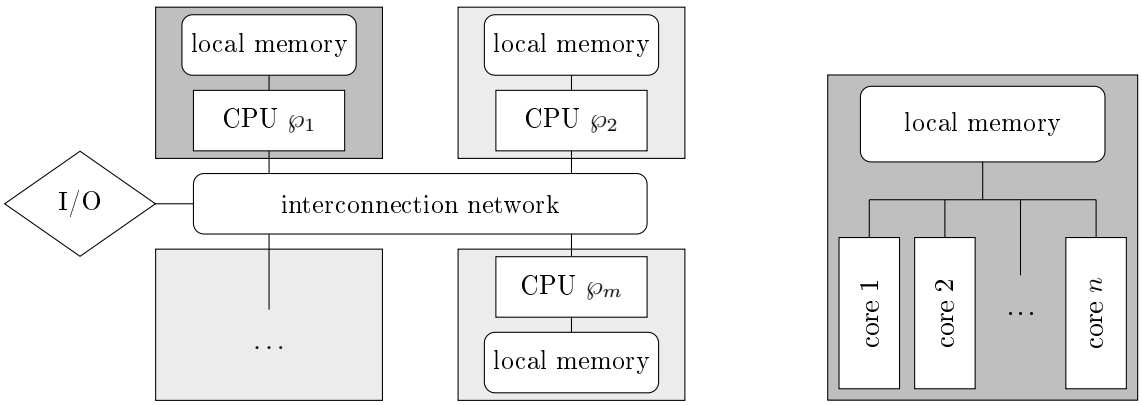


Figure 3.8: Schematic representation of HPC architecture. Illustration of an interconnection network (left). One time step can be solved on one CPU. Exemplary structure of a CPU (right), where the equations are solved in parallel-in-space via domain decomposition on n cores.

Since the pressure Poisson matrix must be provided on each local memory's per CPU $\wp_i$, the total memory requirement is also increased. In addition, the divergence matrix $\mathbf{B}^\top$ and the (local) mass matrix for the pressure $\mathbf{M}_{\text{pl}}$ are need to be assembled on each node / CPU.

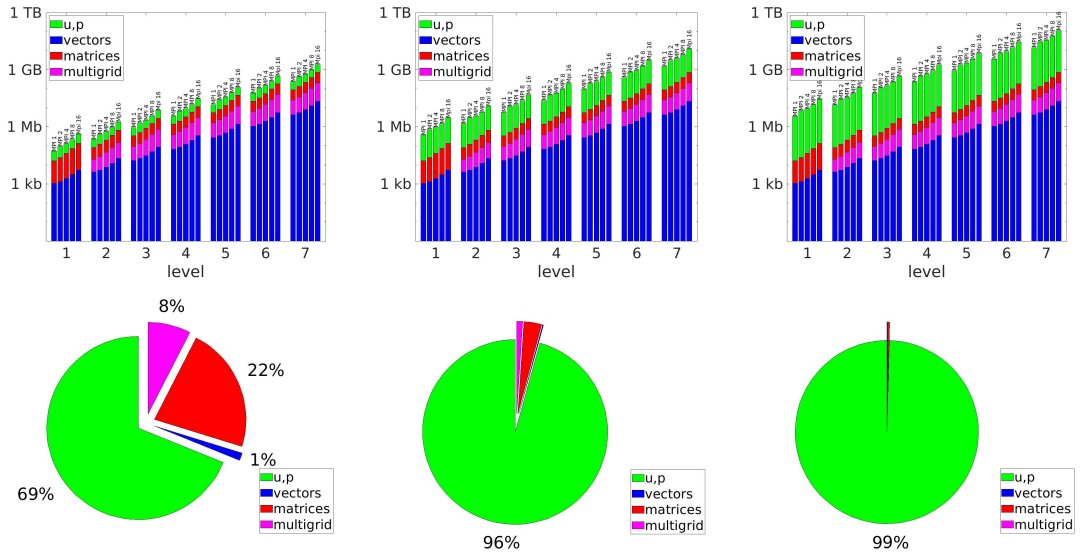


Figure 3.9: 2D Memory consumption for the unit square grid for 16 cores, with various blocked time steps: $K = 100$ (left), $K = 1000$ (center) and $K = 10000$ (right).

Figure 3.9 illustrates the memory efficiency of the variant with a multigrid solver for the Burgers and pressure Poisson equations. Most memory is required for the mandatory velocity and pressure vectors. Further vectors and the matrices require less amount of storage because they only take the size of one time step on each node / CPU. Especially with a high number of time steps, the storage space for the matrices and further vectors is marginal. Measurements are taken with 16 cores and the current real memory is plotted. Here, all cores are used for parallel-in-time computation, i.e. without domain decomposition.

The pressure Poisson matrix $\mathbf{P}$ needs to be assembled only once in a preprocessing step or if the spatial mesh has changed. Also, it is computationally more efficient to store the divergence matrix $\mathbf{B}^\top$ in a separate matrix.

In this work, the momentum equations are solved sequential-in-time, i.e. only one node / CPU is usable for a parallel-in-space computation. Therefore, the momentum matrix only needs to be stored once on the first CPU. An efficient time-simultaneous solver for the momentum problem is not part of this work. For detailed informations on a time-simultaneous velocity solver that exploits many cores on various CPU's, see the work of [49], [26] and [27].

Summary

The time-simultaneous solver approach has the strengths of both PP- and CC-solver:

- The momentum equations and the pressure Poisson equation are solved in the same efficient way as the PP-solver.
- Through the multiple outer iterations, the exact solution can be achieved, so the accuracy is similar to the CC-solver.
- Even large time steps can be performed, which will reach after several outer iterations to the exact solution.
  - As outlined in section 5.2, the use of a coarser time step size enables the computation of larger time intervals in parallel-in-time.
  - The computation time and iteration number is considerably higher because the preconditioners are not exact for a coarser time step size. The optimal preconditioners are derived in section 3.2.1. A performance evaluation for a multigrid in time approach with large time steps k can be found in section 4.4.2.
- The time consuming pressure Poisson equations can be solved fully parallel-in-time, which leads to a major improvement compared to the PP-solver.
- The parareal approach is not used for the most time-consuming pressure Poisson problem, which results in a considerable reduction of synchronization time.

Moreover, the approach is only useful on compute servers and multicomputers due to

- A high main memory consumption.
  - The main memory consumption can be optimized, but still a high storage is required. Also, for performance gains, it makes more sense to temporarily store smaller matrices rather than the entire system matrix, see the end of the introductory section 1.1. Therefore, an implementation in C++, for example in the software FEAT3, is recommended.
- Many processors are needed for an efficient performance of the algorithm.

Furthermore, the challenge is to ensure a divergence-free velocity, which guarantees convergence. Especially in case of the Stokes-Oseen or Navier-Stokes equations, convergence requires a divergence-free velocity approximation at each time step. The divergence-free update in Eq. (2.129) of the PP-solver cannot be used, as this leads to the computation of the time-consuming pressure Poisson problem at each time step, which is opposed to the time-decoupling approach.

- Compared to the PP-algorithm, a more complicated divergence-free update can be applied, using the augmented Lagrangian approach mentioned in section 3.3.3.
- Instead of the time-consuming augmented Lagrangian method, section 4.5 presents the computing of a simple and fast Laplacian problem as an divergence-free update, which can be solved as quickly as the PP divergence-free update, but only guarantees a approximate divergence-free velocity.

3.2 Projection solution by projection solvers (PP-solver)

The fundamental of the projection method, like the PP-solver, is the decoupling of velocity and pressure. According to [78] chapter 2.3, a separation of the velocity and pressure can be achieved by writing the discrete saddle point problem in Eq. (2.124) as

$$\mathbf{S}\mathbf{u} + k\mathbf{B}\mathbf{p} = \mathbf{g}, \tag{3.2a}$$

$$\mathbf{B}^\top \mathbf{u} = 0, \tag{3.2b}$$

and performing a decoupling step for $\mathbf{u}$ and $\mathbf{p}$ as an outer iteration, which reads as

$$\mathbf{u} = \mathbf{S}^{-1}\mathbf{g} - k\mathbf{S}^{-1}\mathbf{B}\mathbf{p}, \tag{3.3a}$$

$$0 = \mathbf{B}^\top \mathbf{S}^{-1}\mathbf{g} - k\mathbf{B}^\top \mathbf{S}^{-1}\mathbf{B}\mathbf{p}. \tag{3.3b}$$

Solving Eq. (3.2a) for $\mathbf{u}$ results in Eq. (3.3a). Multiplying Eq. (3.3a) with the divergence matrix $\mathbf{B}^\top$ and taking advantage of the incompressible equation in Eq. (3.2b) leads to Eq. (3.3b), a scalar equation that contains the pressure. Eq. (3.3b) is also known as pressure Poisson equation and defined as

$$\mathbf{B}^\top \mathbf{S}^{-1}\mathbf{B}\mathbf{p} = \frac{1}{k}\mathbf{B}^\top \mathbf{S}^{-1}\mathbf{g}, \tag{3.4}$$

with the pressure Poisson matrix $\tilde{\mathbf{P}} := \mathbf{B}^\top \mathbf{S}^{-1}\mathbf{B}$ and the right-hand-side defined by $\mathbf{f}_p := \frac{1}{k}\mathbf{B}^\top \mathbf{S}^{-1}\mathbf{g}$. In summary, two decoupled problems need to be solved on account of the PP-algorithm:

i. Solve the momentum equations for the velocity.

   Compute a velocity field without taking incompressibility into account:

   $$\mathbf{S}\tilde{\mathbf{u}}^l = \mathbf{g} - k\mathbf{B}\mathbf{p}^{l-1}. \tag{3.5}$$

   The Burgers equations Eq. (3.5) depends on the pressure from the last time step and needs to be solved for $\tilde{\mathbf{u}}^l$ by Newton-like schemes. In general, the velocity $\tilde{\mathbf{u}}^l$ does not satisfy the incompressible condition in Eq. (3.2b).

ii. Perform a pressure correction, which is a projection back to the subspace of divergence-free vector fields. The pressure correction with a suitable preconditioner C reads as

   $$\mathbf{p}^l = \mathbf{p}^{l-1} + C^{-1}\left(\frac{1}{k}\mathbf{B}^\top \mathbf{S}^{-1}\mathbf{g} - \mathbf{B}^\top \mathbf{S}^{-1}\mathbf{B}\mathbf{p}\right), \tag{3.6}$$

   where the residual of the pressure Poisson equation Eq. (3.4) is in brackets.

Satisfy the saddle point problem

In Eq. (3.5), a generally non divergence-free velocity $\tilde{\mathbf{u}}^l$ is computed through an old iterate $\mathbf{p}^{l-1}$, written as

$$\tilde{\mathbf{u}}^l := \mathbf{u}(\mathbf{p}^{l-1}) = \mathbf{S}^{-1}\mathbf{g} - k\mathbf{S}^{-1}\mathbf{B}\mathbf{p}^{l-1}. \tag{3.7}$$

To fulfill the saddle point problem in Eq. (2.124), a divergence-free velocity approximation $\mathbf{u}^l$ related to the new pressure iterate is required. Suppose there is a certain iterate $\mathbf{p}^L$, which satisfy the Schur complement equations

$$\mathbf{B}^\top \left(\mathbf{S}^{-1}\mathbf{g} - k\mathbf{S}^{-1}\mathbf{B}\mathbf{p}^L\right) = \mathbf{B}^\top \mathbf{u}(\mathbf{p}^L) =: \mathbf{B}^\top \mathbf{u}^L = 0, \tag{3.8}$$

with the velocity approximation defined by

$$\mathbf{u}^L := \mathbf{u}(\mathbf{p}^L) = \mathbf{S}^{-1}\mathbf{g} - k\mathbf{S}^{-1}\mathbf{B}\mathbf{p}^L. \tag{3.9}$$

Than it follows directly that the momentum equations in Eq. (3.9) and the continuity equation in Eq. (3.8), i.e. the saddle point problem in Eq. (2.124), is fulfilled through

$$\begin{bmatrix} \mathbf{S} & k\mathbf{B} \\ \mathbf{B}^\top & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u}^L \\ \mathbf{p}^L \end{bmatrix} = \begin{bmatrix} \mathbf{g} \\ 0 \end{bmatrix}. \tag{3.10}$$

Convergence

Multiplying Eq. (3.6) with the preconditioner C and using the definition of the velocity $\tilde{\mathbf{u}}$ in (3.3a) leads to

$$C\mathbf{p}^l = C\mathbf{p}^{l-1} + \frac{1}{k}\mathbf{B}^\top\left(\mathbf{S}^{-1}\mathbf{g} - k\mathbf{S}^{-1}\mathbf{B}\mathbf{p}^l\right) \tag{3.11a}$$

$$= C\mathbf{p}^{l-1} + \frac{1}{k}\mathbf{B}^\top\tilde{\mathbf{u}}^l. \tag{3.11b}$$

Convergence is achieved if the velocity in Eq. (3.11b) is divergence-free with $\mathbf{B}^\top\tilde{\mathbf{u}} \equiv \mathbf{B}^\top\mathbf{u} = 0$, i.e., Brezzi sufficient condition for well-posedness theorem 2.8 is fulfilled.

Properties of the pressure Poisson equation

For solving the pressure Poisson equation in Eq. (3.4), the matrix has to be invertible, i.e. the inverse needs to be bounded. The condition for a bounded inverse is given by the inf-sup conditions from the previous chapter in section 2.2. In the finite dimensional case, the inf-sup condition becomes a min-max condition, which reads as

$$\beta_h \leq \min_{\mathbf{p}\in\mathbb{R}^{d_Q}} \max_{\mathbf{w}\in\mathbb{R}^{d_V}} \frac{\mathbf{w}^\top\mathbf{S}^{-\frac{1}{2}}\mathbf{B}\mathbf{p}}{\|\mathbf{w}\|_{l^2}\|\mathbf{p}\|_{l^2}}, \tag{3.12}$$

with a fixed constant $\beta_h > 0$ and the Euclidean norm defined as $\|\mathbf{x}\|_{l^2} := \sqrt{\mathbf{x}^\top\mathbf{x}}$.

Multiply Eq. (3.12) by $\|\mathbf{p}\|_{l^2}$ and taking the best choice of $\mathbf{w}$ leads to the inequality

$$\beta_h\|\mathbf{p}\|_{l^2} \leq \|\mathbf{S}^{-\frac{1}{2}}\mathbf{B}\mathbf{p}\|_{l^2}. \tag{3.13}$$

Squaring Eq. (3.13) yields a condition for the pressure Poisson matrix that is

$$\beta_h^2\|\mathbf{p}\|_{l^2}^2 \leq \|\mathbf{S}^{-\frac{1}{2}}\mathbf{B}\mathbf{p}\|_{l^2}^2 = \mathbf{p}^\top\mathbf{B}^\top\mathbf{S}^{-1}\mathbf{B}\mathbf{p}. \tag{3.14}$$

Due to the inf-sup condition, the matrix $\mathbf{B}^\top\mathbf{S}^{-1}\mathbf{B}$ must be bounded below by $\beta_h^2\|\mathbf{p}\|_{l^2}^2$.

The condition in Eq. (3.14) fails if there exists a nonzero pressure $\mathbf{p}$ with $\mathbf{B}\mathbf{p} = 0$, which is the reason why the gradient matrix $\mathbf{B}$ needs to have full column rank. This requirement is fulfilled for the Q_2/P_1^{disc} element. For the Q_2/Q_1 element, the condition is satisfied, but with one exception: In case of a triangulation with squares, the pressure vector of the Q_2/Q_1 element satisfies $\mathbf{B}\mathbf{p} = 0$ with $\mathbf{p} = (1, -1, 1, \ldots, 1, -1)$, see figure 3.10. In the literature this phenomenon is also known as unstable checkerboard mode, see [22] section 8.10.

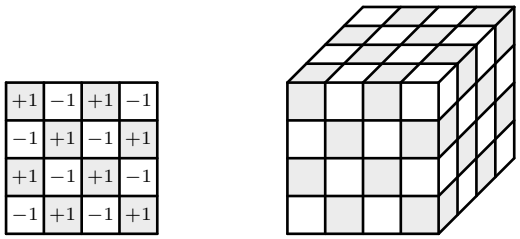


Figure 3.10: Visualization of checkerboard mode for Q_2/Q_1 element: unit square grid (left) and unit cube grid (right).

In order to calculate an accurate value for the pressure norm in Eq. (3.14), the finite element functions must be taken into account. Using the L^2-inner product gives

$$\|p_h\|_{L^2}^2 = (p_h, p_h) = \sum_{i=1}^{d_Q} \sum_{j=1}^{d_Q} p_i p_j (\psi_i, \psi_j) = \mathbf{p}^\top \mathbf{M}_p \mathbf{p}, \tag{3.15}$$

with $\mathbf{M}_{p_{ij}} := (\psi_i, \psi_j)$ as pressure mass matrix.

Using Eq. (3.15) in Eq. (3.14) results in the inequality

$$\beta_h^2 \mathbf{p}^\top \mathbf{M}_p \mathbf{p} \leq \mathbf{p}^\top \mathbf{B}^\top \mathbf{S}^{-1} \mathbf{B} \mathbf{p}, \tag{3.16a}$$

$$\left\| \left(\mathbf{B}^\top \mathbf{S}^{-1} \mathbf{B} \right)^{-1} \right\|_{l^2} \leq \frac{1}{{\beta_h}^2} \left\| \mathbf{M}_p^{-1} \right\|_{l^2}. \tag{3.16b}$$

Eq. (3.16b) provides a bound for the inverse, in other words a condition for an invertible pressure Poisson matrix. This also provides the pressure mass matrix $\mathbf{M}_p$ as a simple preconditioner.

3.2.1 Preconditioning

The idea according to [78] chapter 2.3 is to construct *optimal* preconditioners for the pressure Poisson equation Eq. (3.4). *Optimal* means that the partial preconditioners were direct solvers with respect to the underlying sub-problem. Therefore, the resulting convergence behavior should be independent of outer parameters as well as the underlying mesh.

There are various methods of preconditioning. A factorized preconditioner approach can be found in [78], section 2.3.4. In this thesis, an additive approach is used, which leads to the construction of *optimal* global defined preconditioniers for the limit cases. The matrix $\mathbf{S}$, defined in Eq. (2.123), is made up of three parts. A reactive part defined by the matrix $\mathbf{M}$, a diffusive part denoted by the matrix $\mathbf{L}$ and in the case of the Stokes-Oseen and the Navier-Stokes equations a convective part describes by the matrix $\mathbf{K}$.

i. **Reactive preconditioner for $\mathbf{B}^\top \mathbf{M}^{-1} \mathbf{B}$**

The limit case for a dominant reactive part $\mathbf{M}$ is achieved for a small time step size $k \to 0$, i.e. matrix $\mathbf{S}$ results in

$$\mathbf{S}(\mathbf{u}) \equiv \mathbf{M}. \tag{3.17}$$

In order to get an easily to inverted matrix, it is necessary to "lump" the mass matrix $\mathbf{M}$, i.e., to express it as a diagonal matrix $\mathbf{M}_l$. Three common variants for lumping the mass matrix are described in [82], section 16.2.4.

Lumping leads to the pressure Poisson matrix

$$\mathbf{P} := \mathbf{B}^\top \mathbf{M}_l^{-1} \mathbf{B}, \tag{3.18}$$

which can be viewed as an optimal preconditioner for small time step sizes k, but may become a poor approximation to $\mathbf{B}^\top \mathbf{S}^{-1} \mathbf{B}$ at large time steps, which can be seen in section 4.4.2.

The advantages are an almost exact solver/preconditioner for small time steps, which is highly efficient for multigrid solvers for applying $\mathbf{P}^{-1}$. Also there are very compact matrices $\mathbf{P}$ independent of the spatial discretization.

In section 3.2.2, the condition number of the pressure Poisson matrix will be discussed.

ii. **Diffusive preconditioner for $\mathbf{B}^\top \mathbf{L}^{-1} \mathbf{B}$**

The limit case for a dominant diffusive part $\mathbf{L}$ can be reached for an extremely viscous fluid, like in case of Stokes equations. The inverse discrete Laplacian $\mathbf{L}^{-1}$ and also $\mathbf{B}^\top \mathbf{L}^{-1} \mathbf{B}$ are full matrices. If considering the matrix abstractly, the relation is as follows

$$\nabla \cdot \Delta^{-1} \nabla \backsim I, \tag{3.19}$$

which leads in the finite element context to

$$\mathbf{B}^\top \mathbf{L}^{-1} \mathbf{B} \backsim \mathbf{M}_p, \tag{3.20}$$

with $\mathbf{M}_p$ as the mass matrix for the pressure, which confirms the conclusion from Eqs. (3.16). Similar to the velocity mass matrix, a lumped version can be used.

If $\mathbf{M}_p$ is used as a smoother in a multigrid context, the resulting convergence rates are independent of the mesh size parameter h. However, in contrast to the reactive preconditioner, the diffusive counterpart is not an exact solver for $\mathbf{B}^\top \mathbf{L}^{-1} \mathbf{B}$.

All numerical tests in [78] show, that indeed $\mathbf{M}_p$ is sufficient. It is also absolutely robust against all variations of parameters and the shape of the mesh in the pure Stokes case. Also, the preconditioner leads to an improved convergence rate in the pressure update, which is numerically shown in chapter 4.1.

iii. **Convective preconditioner for $\mathbf{B}^\top \mathbf{K}^{-1} \mathbf{B}$**

In the case of the Stokes equations, there is no convective part. For a small viscosity parameter $\nu \to 0$, a dominant convective part can be obtained in the Stokes-Oseen or Navier-Stokes case.

The inverse transport matrix $\mathbf{K}^{-1}$ and $\mathbf{B}^\top \mathbf{K}^{-1} \mathbf{B}$ are full matrices. There are several approaches shown in [78], but none is promising. There are either poor condition numbers $\mathcal{O}(h^{-1})$ to $\mathcal{O}(h^{-2})$, a preconditioner that is sensitive to mesh anisotropies, or that the entire solution process is almost so expensive as for the original system.

Therefore, stabilization methods or a smaller time step are often used so that a dominant convective part can be avoided.

3.2.2 PP-algorithm (condition number)

The PP-algorithm with the above mentioned preconditioners is written as:

Algorithm 3.1: PP-algorithm

input : initial value $\mathbf{u}^0 = u_0$ ($\mathbf{p}^0 = 0$)
output: sequential-in-time: next iterative $\mathbf{u}^l$, $\mathbf{p}^l$

1 **for** $l \leftarrow 1$ **to** K **do**
2 /* Burgers equations: solve for an intermediate velocity $\tilde{\mathbf{u}}$ */
3 $\tilde{\mathbf{u}}^l \leftarrow \langle$**solve** $\mathbf{S}(\tilde{\mathbf{u}}^l)\tilde{\mathbf{u}}^l = \mathbf{g} - k\mathbf{B}\mathbf{p}^{l-1}\rangle$ // use fixedpoint_solver (for non-linear problems)
4 $\mathbf{f}_p \leftarrow \frac{1}{k}\mathbf{B}^\top \tilde{\mathbf{u}}^l$ // divergence-defect for pressure Poisson equation
5 /* pressure Poisson equation*/
6 $\mathbf{q} \leftarrow \langle$**solve** $\mathbf{P}\mathbf{q} = \mathbf{f}_p\rangle$
7 $\mathbf{p}^l \leftarrow \mathbf{p}^{l-1} + \alpha_R\mathbf{q} + \alpha_D\mathbf{M}_{\mathrm{pl}}^{-1}\mathbf{f}_p$ // update the pressure
8 $\mathbf{u}^l \leftarrow \tilde{\mathbf{u}}^l - k\mathbf{M}_{\mathrm{l}}^{-1}\mathbf{B}\mathbf{q}$ // enforcing divergence-free solution
9 /* non-linearity treatment with fixed-point iteration */
10 **function** fixedpoint_solver
11 $\mathbf{f} \leftarrow \mathbf{g} - k\mathbf{B}\mathbf{p}^{l-1}$
12 **do**
13 $\mathbf{d} \leftarrow \mathbf{f} - \mathbf{S}(\tilde{\mathbf{u}}^l)\tilde{\mathbf{u}}^l$ // compute residual
14 $\mathbf{y} \leftarrow \langle$**solve** $\mathbf{S}(\tilde{\mathbf{u}}^l)\mathbf{y} = \mathbf{d}\rangle$ // solve an auxiliary subproblem
15 $\tilde{\mathbf{u}}^l \leftarrow \tilde{\mathbf{u}}^l + \mathbf{y}$ // update $\tilde{\mathbf{u}}$ via the auxiliary solution
16 **while** $\boldsymbol{d} \leq tol$ // repeat until tolerance reached;

First, suitable start values must be provided. Since the initial pressure value is generally unknown, the zero vector is used. The correct pressure is set when the pressure is updated.

Pressure update

After the first iteration, the pressure update in line 7 sets a proper approximate. In the second part of the pressure update the diffusive preconditioner is applied by a constant $\alpha_D > 0$. According to [78], a choice of $\alpha_R \leq 1$ and $\alpha_D = \theta k\nu$ is suggested for the non-stationary case and $\alpha_R = 0$ and $\alpha_D \leq \nu$ for stationary calculations.

In [46], it is numerically shown that the additional divergence correction significantly improves the pressure approximation in the PP-algorithm. Later, in section 4.1, the diffusive preconditioner is investigated in detail for the new time-simultaneous approach mentioned in the next section.

Divergence-free solution

Figure 3.11 illustrates the velocity update from line 8, an update back to the space of divergence-free functions. Numerical experiments show that the velocity field $\tilde{\mathbf{u}}$ is not guaranteed to be divergence-free. In the PP-algorithm, there exists no outer iteration, which iteratively converges to an exact divergence-free velocity. At the end of every time step, an additional update seems useful to satisfy Brezzi's sufficient condition for well-posedness in theorem 2.8, which can be achieved simply by adding $\mathbf{B}\mathbf{q}$ scaled by $k\mathbf{M}_l^{-1}$.

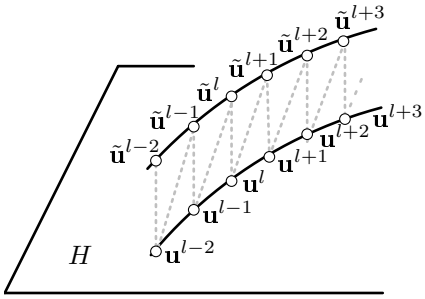


Figure 3.11: Divergence-free update: At time t^{l+1} compute $\tilde{\mathbf{u}}^{l+1}$ without enforcing incompressibility. As a result, $\tilde{\mathbf{u}}^{l+1}$ is observed to project out of H. Project $\tilde{\mathbf{u}}^{l+1}$ onto H to obtain a divergence-free velocity $\mathbf{u}^{l+1}$.

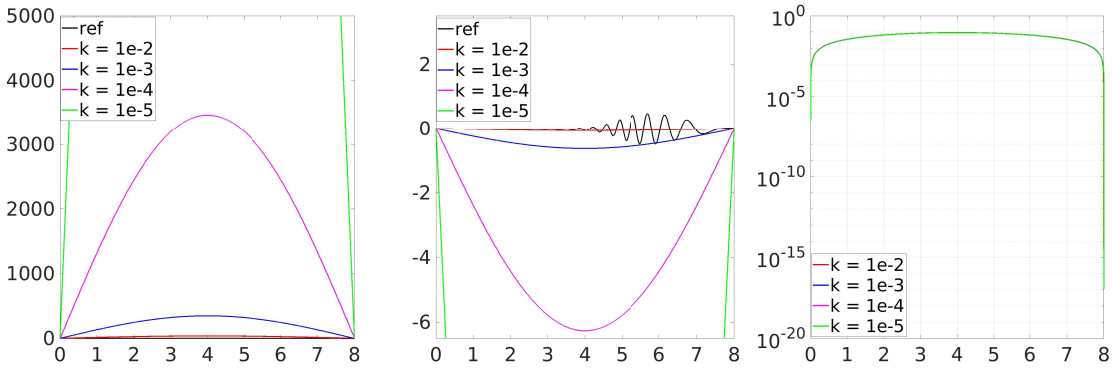


Figure 3.12: Flow-around-a-cylinder benchmark computed by the PP-approach without a divergence-free velocity update showing drag values (left), lift values (center) and the divergence-defect of the velocity (right). The x-axis represents the time interval [0,8].

In figure 3.12, the flow-around-a-cylinder benchmark from the introduction in section 1.1 is computed without enforcing a divergence-free solution. The solver computes a solution, but even for a small time step size k it does not even come close to the reference of the drag and lift values. Compared to figure 1.3, it can be seen that without an additional update for the velocity the divergence-defect is only in the range of 10^{-3}, independent of the selected time step size k. It can be concluded that more than one outer iteration is required if the divergence-free update in line 8 is avoided.

Burgers equations

The Burgers equations are solved in the defect notation. Any existing nonlinearity can be treated due to an outer loop in line 16 to achieve a certain tolerance. However, in the case of pure Stokes equations, there is no nonlinearity, so that the tolerance has been reached in just one outer iteration. In the case of Stokes-Oseen equations, a fixed-point loop can be useful.

Pressure Poisson equation (condition number)

The main part of the PP-algorithm is the pressure Poisson equation in line 6. The convergence behavior is essentially determined by the condition number of the matrix $\mathbf{P}$. Therefore, it is important to derive a bound on the condition number of the pressure Poisson matrix $\mathbf{P}$. Since $\mathbf{P}$ is symmetric, its condition number is

$$\kappa(\mathbf{P}) = \frac{|\lambda_{\max}|}{|\lambda_{\min}|} \tag{3.21}$$

i.e., it is the ratio of the largest to the smallest absolut eigenvalue of $\mathbf{P}$.

An estimate of the eigenvalues from the non-preconditioned matrix $\mathbf{B}^\top \mathbf{S}^{-1}\mathbf{B}$ can be found in [2] Prop 4.47, which is

$$\lambda_{\min} \geq \frac{\beta_{a,h}\beta_{b,h}^2}{c_1} \min\left(\lambda_{\mathbf{M}_p}\right), \tag{3.22a}$$

$$\lambda_{\max} \leq \frac{c_2}{\beta_{a,h}} \max\left(\lambda_{\mathbf{M}_p}\right), \tag{3.22b}$$

with $\beta_{a,h}$ as the coercivity constant and $\beta_{b,h}$ as the inf-sup constant from theorem 2.9 as well as $\lambda_{\mathbf{M}_p}$ as the eigenvalue from the pressure mass matrix. An exact determination of the constants c_1 and c_2 can be found in [2].
Using Eq. (3.21) with the estimations Eqs. (3.22) yields

$$\kappa(\mathbf{P}) \leq \frac{c}{\left(\beta_{a,h}\beta_{b,h}\right)^2}\kappa(\mathbf{M}_p), \tag{3.23}$$

with a constant c depending on c_1 and c_2. According to [2], the condition number in Eq. (3.23) depends on νk. In section 4.4.2, it can be seen that solving the pressure Poisson equation for large time steps results in poor conditioning and therefore requires more iterations. This is caused by the fact that the preconditioner in Eq. (3.18) is no longer exact for large time steps k.
In [2] section 6.2, it is mentioned that for a small νk the preconditioned pressure Poisson matrix in Eq. (3.18) behaves like $\kappa(\mathbf{P}) \sim h^{-2}$, i.e., the condition number of $\mathbf{P}$ behaves like that of the Laplace operator.

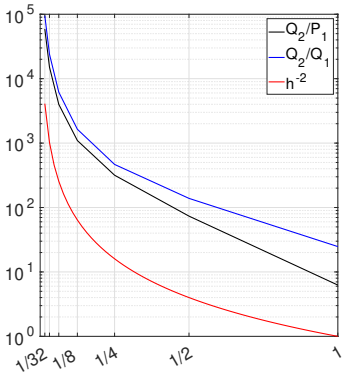


Figure 3.13: Condition number for different levels for the pressure Poisson matrix $\mathbf{P}$ in case of the Q_2/P_1^{disc} element in black, the Q_2/Q_1 element (in blue) as well as the reference function h^{-2} (in red).

In Figure 3.13, the condition number was calculated on the unit square grid. It can be seen that the condition number clearly depends on the grid size. The element Q_2/P_1^{disc} has a slightly better condition number. For both element pairs a $\mathcal{O}(h^{-2})$ behavior can be seen clearly. Therefore, for a small grid size, standard iterative solution methods may not converge well.

3.2.3 Multigrid method (multigrid in space)

The Burgers equations and the pressure Poisson equation are solved with multigrid methods.
For typical iterative numerical solution methods, high-frequency (local) errors in the solution are well-damped, while lower frequency (global) errors are poorly damped. Therefore, the low-frequency errors are difficult to eliminate, which leads to slower solver convergence, especially on fine meshes. The key idea behind multigrid is that effective rates of convergence at all scales can be maintained in a solver by leveraging a sequence of grids at various resolutions.

Classical books about multigrid methods are [15], [37] and [73].

A multigrid technique can solve a system of N equations using as few as $\mathcal{O}(N)$ arithmetic operations, as compared to $\mathcal{O}(N^3)$ for Gaussian elimination and $\mathcal{O}(N^2)$ for forward/backward substitution with a precomputed LU factorization. A key advantage of the multigrid method is that the order of convergence is independent of the mesh size.

The performance of the multigrid solver relies most of all on the efficiency of its components, namely

i. the sparsity of the matrix, i.e. the matrix-vector multiplication

ii. the smoothers on finer levels and the coarse grid solvers,

iii. the grid transfer operators: the prolongation and the restriction operator

Prolongation and restriction

The core steps of the multigrid techniques are to smooth the defect, restrict it to a coarser grid, compute a defect correction, and prolongate it back to update the iteration vector.
In figure 3.14 level `0` and one refinement, level `1`, is shown with the associated prolongation operator $\mathbf{I}_{l-1}^{l}$ and restringation operator $\mathbf{I}_{l}^{l-1}$.

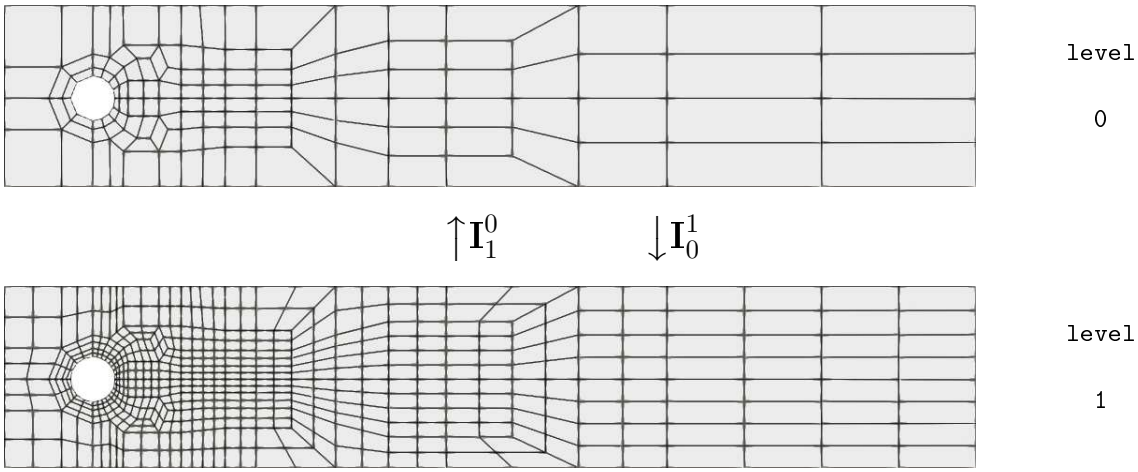


Figure 3.14: Level `0`, the coarse grid, and level `1` of the flow-around-a-cylinder domain with associated prolongation operator $\mathbf{I}_0^1$ and restringation operator $\mathbf{I}_1^0$.

A simplified example of the grid transfer in 2D for the Q_1 element is illustrated in figure 3.15. The restriction and prolongation matrices are very sparse and have almost a rather irregular sparsity pattern.

To determine the prolongation matrix, it is necessary to consider the correspondences of the nodes indices between two grids. Different index mappings lead to different sparsity patterns of the prolongation matrices. An efficient index numbering, which is used in the software FEAT3, can be seen in figure 3.15.

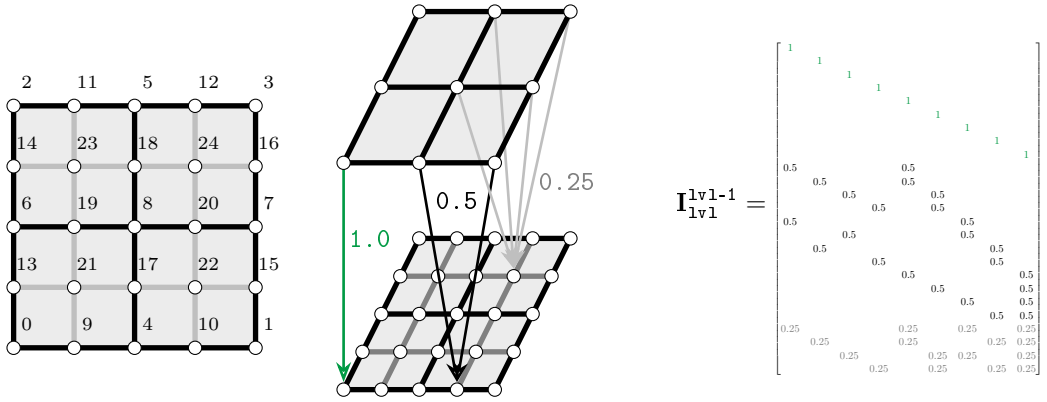


Figure 3.15: Numbering of the DOFs for the Q_1 element in FEAT3 (left).
Visualization of the prolongation operator (center) and associated prolongation matrix (right).

Analogously, the restriction matrix will be constructed. According to [37] section 3.5, the restriction operators are naturally defined as adjoint of prolongation operators.

Multigrid cycles

There are several structures to describe one iteration step in the multigrid method called multigrid cycles. The common cycles are V-cycle and W-cycle, shown in figure 3.16.

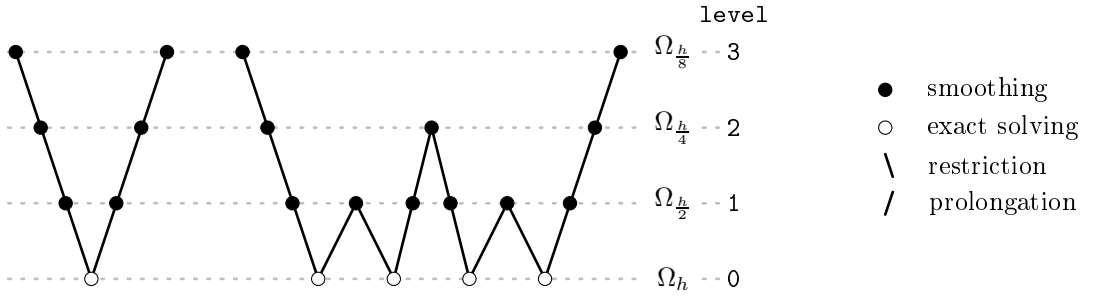


Figure 3.16: Visualization of V-cycle (left) and W-cycle (right) on four grids.

During a multigrid V-cycle, the solution is first approximated using several smoothing iterations with e.g. a damped Jacobi on the finest mesh, called pre-smoothing. The residual is then transferred to the first coarse level, where additional smoothing iterations occur. This restriction followed by smoothing continues recursively until the coarsest mesh level is reached. On the coarsest mesh level, additional smoothing iterations and a direct solver are applied at almost no cost due to the small problem size. A correction for the solution values is transferred to the finer mesh level above. The prolongation and post-smoothing continues until a correction is finally applied to the solution on the finest mesh.

In algorithm 3.2, the structure of the V-cycle of the multigrid method is shown in the form of a psydocode.

Algorithm 3.2: Multigrid (V-cycle)

input : matrix $\mathbf{A}$, right-hand-side $\mathbf{g}$, initial guess $\mathbf{x}$, level `lvl`
output: solution vector $\mathbf{x}$

```
/* multigrid method for the problem Ax = g */
function multigrid_V_cycle(A,g,x,lvl)
    x^lvl ← smoother(g^lvl, A^lvl, x^lvl)                    // pre-smoothing
    r^lvl ← g^lvl − A^lvl x^lvl                              // compute residual
    r^lvl−1 ← I^lvl_lvl−1 r^lvl                              // restriction to coarse grid
    if lvl-1 = 0 then
        x^lvl ← (A^lvl)^−1 r^lvl                             // direct solving on coarsest mesh level
    else
        multigrid_V_cycle(A, r, 0, lvl − 1)                  // recursion
    x^lvl ← x^lvl + I^lvl−1_lvl x^lvl−1                      // prolongation to fine grid
    x^lvl ← smoother(g^lvl, A^lvl, x^lvl)                    // post-smoothing
```

At each level, the matrix $\mathbf{A}^{\texttt{lvl}}$ and the right-hand-side $\mathbf{g}^{\texttt{lvl}}$ have to be assembled. This leads to additional memory consumption, which is described in the previous section in figure 3.9. The memory consumption turns out to be respectively small, especially in the time-simultaneous case in the next section.

Smoother and coarse grid solver

The smoothing is used to dampen the strongly oscillating error modes of the system. The smoother used is

i. a Richardson-multigrid solver for Burgers equations.

ii. a PCG-multigrid solver for the pressure Poisson problem.

Both multigrids use a damped Jacobi preconditioner as pre-smoother, post-smoother as well as for the coarse-grid. In each smoothing step, a fixed number of four smoothing iterations are performed without any convergence control. By default, the V-cycle with a tolerance of 10^{-16} is used for both problems.

Convergence of PCG

For the PCG method, the error can be bounded in terms of the spectral condition number κ of the preconditioned pressure Poisson matrix. The error between the i^{th} iteration approximate $\mathbf{p}^i$ and the exact solution is defined as $e^i := p_h^i - p$. It satisfies according to [6] the formula

$$\|\mathbf{e}^i\|_{\mathbf{P}} \leq 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^i \|\mathbf{e}^0\|_{\mathbf{P}}, \tag{3.24}$$

with $\|\mathbf{e}^i\|_{\mathbf{P}} := \sqrt{\mathbf{e}^i\mathbf{P}\mathbf{e}^i}$.
Eq. (3.24) shows that the convergence is limited by the spectral condition number of the preconditioned pressure Poisson matrix and the initial iteration error. Therefore, providing an appropriate precondition and a good initial iteration value is critical to speed up the iteration process.

3.3 Time-simultaneous PP-solver

Based on the PP-solver, the global-in-time approach is introduced, which can solve the pressure Poisson problems much faster due to the decoupling of the time and further parallelization via the MPI library.
As described in the introduction in the section 3.1.3, the time dependent system

$$\begin{bmatrix} \mathbf{S} & k\mathbf{B} \\ \mathbf{B}^\top & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u}^l \\ \mathbf{p}^l \end{bmatrix} = \begin{bmatrix} \mathbf{g}^l \\ 0 \end{bmatrix} \quad \text{in } \Omega, \text{ for } l = 1, \dots, K \tag{3.25}$$

needs to be decoupled from the time. The system including all time steps is written as

$$\left[\begin{array}{ccc|ccc} \mathbf{S} & & & k\mathbf{B} & & \\ & \ddots & & & \ddots & \\ & & \mathbf{S} & & & k\mathbf{B} \\ \hline \mathbf{B}^\top & & & 0 & & \\ & \ddots & & & \ddots & \\ & & \mathbf{B}^\top & & & 0 \end{array}\right] \left[\begin{array}{c} \mathbf{u}^1 \\ \vdots \\ \mathbf{u}^K \\ \hline \mathbf{p}^1 \\ \vdots \\ \mathbf{p}^K \end{array}\right] = \left[\begin{array}{c} \mathbf{g}^1 \\ \vdots \\ \mathbf{g}^K \\ \hline 0 \\ \vdots \\ 0 \end{array}\right] \quad \text{in } \Omega. \tag{3.26}$$

According to Eq. (2.122), the vector $\mathbf{g}$ depends on the last velocity iterate. More specifically, the entire system reads as

$$\left[\begin{array}{cccc|ccc} \mathbf{S} & & & & k\mathbf{B} & & \\ \overline{\mathbf{S}} & \mathbf{S} & & & & k\mathbf{B} & \\ & \ddots & \ddots & & & & \ddots \\ & & \overline{\mathbf{S}} & \mathbf{S} & & & k\mathbf{B} \\ \hline \mathbf{B}^\top & & & & 0 & & \\ & \mathbf{B}^\top & & & & 0 & \\ & & \ddots & & & & \ddots \\ & & & \mathbf{B}^\top & & & 0 \end{array}\right] \left[\begin{array}{c} \mathbf{u}^1 \\ \mathbf{u}^2 \\ \vdots \\ \mathbf{u}^K \\ \hline \mathbf{p}^1 \\ \mathbf{p}^2 \\ \vdots \\ \mathbf{p}^K \end{array}\right] = \left[\begin{array}{c} \mathbf{g}^1 \\ k(\theta \boldsymbol{g}^2 + (1-\theta)\boldsymbol{g}^1) \\ \vdots \\ k(\theta \boldsymbol{g}^K + (1-\theta)\boldsymbol{g}^{K-1}) \\ \hline 0 \\ 0 \\ \vdots \\ 0 \end{array}\right] \text{in } \Omega, \tag{3.27}$$

with a matrix $\overline{\mathbf{S}} := -\mathbf{M} + k(\mathbf{K} + \nu\mathbf{L})(1-\theta)$, which include the explicit part.
As described in section 3.1.3, the large system Eq. (3.27) becomes very memory-intensive for a large number of blocked time steps and is therefore not implemented in its entirety.

3.3.1 Properties of the time decoupled matrix

The dominant reactive part $\mathbf{M}$ can be obtained in the same way as in Eq. (3.17). In the limiting case $k \to 0$, the system results due to the explicit part in

$$\underline{\underline{\mathbf{S}}} := \begin{bmatrix} \mathbf{M} & & & \\ -\mathbf{M} & \mathbf{M} & & \\ & \ddots & \ddots & \\ & & -\mathbf{M} & \mathbf{M} \end{bmatrix}. \tag{3.28}$$

In the inverse case, this results in an almost full matrix, which reads as

$$\underline{\underline{\mathbf{S}}}^{-1} = \begin{bmatrix} \mathbf{M}^{-1} & & & \\ \mathbf{M}^{-1} & \mathbf{M}^{-1} & & \\ \vdots & & \ddots & \\ \mathbf{M}^{-1} & \cdots\cdots & & \mathbf{M}^{-1} \end{bmatrix}. \tag{3.29}$$

Eq. (3.29) multiplied with the gradient and divergence matrices gives an almost full system of pressure Poisson matrices, written as

$$\underline{\underline{\mathbf{P}}} := \begin{bmatrix} \mathbf{B}^\top\mathbf{M}^{-1}\mathbf{B} & & & \\ \mathbf{B}^\top\mathbf{M}^{-1}\mathbf{B} & \mathbf{B}^\top\mathbf{M}^{-1}\mathbf{B} & & \\ \vdots & & \ddots & \\ \mathbf{B}^\top\mathbf{M}^{-1}\mathbf{B} & \cdots\cdots & & \mathbf{B}^\top\mathbf{M}^{-1}\mathbf{B} \end{bmatrix}. \tag{3.30}$$

The corresponding right-hand-side $\underline{\underline{\mathbf{f}_p}}$ contains $\mathbf{f}_p$ from all time steps. The full system in Eq. (3.30) never needs to be assembled. For solving the system of pressure Poisson's equations

$$\underline{\underline{\mathbf{q}}} = \underline{\underline{\mathbf{P}}}^{-1}\underline{\underline{\mathbf{f}_p}}, \tag{3.31}$$

it is sufficient to observe the inverse of the pressure Poisson system matrix, which is

$$\underline{\underline{\mathbf{P}}}^{-1} = \begin{bmatrix} \left(\mathbf{B}^\top\mathbf{M}^{-1}\mathbf{B}\right)^{-1} & & & \\ -\left(\mathbf{B}^\top\mathbf{M}^{-1}\mathbf{B}\right)^{-1} & \left(\mathbf{B}^\top\mathbf{M}^{-1}\mathbf{B}\right)^{-1} & & \\ & \ddots & \ddots & \\ & & -\left(\mathbf{B}^\top\mathbf{M}^{-1}\mathbf{B}\right)^{-1} & \left(\mathbf{B}^\top\mathbf{M}^{-1}\mathbf{B}\right)^{-1} \end{bmatrix}. \tag{3.32}$$

Instead of considering the whole system Eq. (3.32), it is sufficient to consider many small pressure Poisson problems of the size of only one time step, which can easily be computed in parallel-in-time. The key is to use a modified right-hand-side

$$\mathbf{f}_p := \frac{1}{k}\mathbf{B}^\top\tilde{\mathbf{u}}^l - \frac{1}{k}\mathbf{B}^\top\tilde{\mathbf{u}}^{l-1} \quad \text{for } l = 1, \ldots, K \tag{3.33}$$

for the pressure Poisson problem that contains the already computed divergence-defect of the previous and current time step. Note that a divergence-free update is not yet provided. If the algorithm is accelerated using a divergence-free optimization, it is necessary to use the adjusted right-hand-side from Eq. (3.42). This will be studied in more detail in chapter 4.

3.3.2 Time-simultaneous PP-algorithm

A psydocode for the time-simultaneous case can be found in algorithm 3.3. The algorithm is similar to the PP-algorithm 3.1 from section 3.2.2. A difference can be found in line 15, where the adjusted right-hand-side in Eq. (3.33) is used for the pressure Poisson problem.
A further difference is that there is (so far) no divergence-free update for the velocity, which guarantees convergence of the solver. Compared to the PP-algorithm, an additional outer loop is added in line 19, which leads to a divergence-free velocity in case of convergence. However, it may be useful to directly compute a divergence-free velocity, which leads to a convergence boost for the solver, see section 3.3.3 for an initial concept and section 4.5 for a fast update that is similar to the PP-update, but that only ensures a approximate divergence-free velocity.

The Burgers equations are computed sequential-in-time based on the PP-algorithm, which is not efficient as a large part of the CPUs in line 11 is idle. A better option is to use the parareal implementation, a time-simultaneous or a time-parallel method for the velocity problem, which are being investigated by other members of the chair LSIII at the TU Dortmund University.

The velocity error depends on the previously calculated velocity. The lack of the divergence-free velocity has therefore a strong influence on the number of blocked time steps, which can be observed especially in the convection dominant case of the Navier-Stokes equations.

Algorithm 3.3: Time-simultaneous PP-solver

input : initial value $\mathbf{u}^0 = u_0$ ($\mathbf{p}^0 = 0$)
output: time-simultaneous: solution vector of each time step $\mathbf{U}$, $\mathbf{P}$

```
1  initialize U                                          // setting up velocity starting values
2  initialize P                                          // setting up pressure starting values
3  do
4      if main cores then
5          for l ← 1 to K do
6              /* solve for each time step an intermediate velocity ũ */
7              p^{l-1} ← P(l − 1)                        // get pressure approximate
8              ũ^l ← ⟨solve  Sũ^l = g − kBp^{l-1}⟩
9              U(l) ← ũ^l                                // save velocity in a global vector U
10     else
11         mpi wait

12     do in parallel l ← 1 to K (parallel in space and time)
13         if free cores then
14             mpi sync: ũ^{l-1} ← U(l − 1) , ũ^l ← U(l)   // get velocity from main cores
15             f_p ← (1/k) B^⊤ ũ^l − (1/k) B^⊤ ũ^{l-1}     // derive adjusted right-hand-side
16             q ← ⟨solve  Pq = f_p⟩
17             p^l ← p^l + α_R q + α_D M_pl^{-1} f_p       // update current pressure
18             mpi sync: P(l) ← p^l                        // send pressure to main cores
19 while ( ( B^⊤ u ≤ tol )  &  (‖Su + kBp − g‖ ≤ tol) ) ;
```

The *main cores* in algorithm 3.3 in line 4 represent the first and main CPU. Free CPU's are denoted by *free cores* mentioned above in line 13. In order to keep all CPU's during the pressure Poisson part busy, a simple approach is chosen in which the next CPU is taken in ascending order with the time step. However, as indicated in section 5.3, this approach does not appear to be most effective on many cores with regard to the syncornisation time. The synchronization in line 14 is needed to get access to the velocity vector in the memory of the main CPU and also in line 18 to send the updated pressure vector back to the main CPU.

The allocation of the CPU's can definitely be optimized to save (synchronization) time by taking the next CPU's after every 100 or 500 time steps instead of after each time step. As the synchronization times are relatively small, this further optimization has not yet been considered.

3.3.3 Divergence-free update (augmented Lagrangian)

As mentioned, a divergence-free update is not necessary (for the Stokes equations) due to the outer loop in algorithm 3.3 line 19. However, a divergence-free update provides a major boost to the algorithm, since convergence according to Eq. (3.11b) is achieved due to a divergence-free velocity. In the best case, a divergence-free velocity can be restored precisely, which leads to a PP like solver with only one outer iteration.
The most intuitive way is to rewrite the velocity problem in algorithm 3.3 line 8 as an optimization problem, written as

$$\begin{aligned} \min_{\mathbf{u}\in\mathbb{R}^{d_V}} \quad & \mathbf{S}\mathbf{u} + k\mathbf{B}\mathbf{p} - \mathbf{g} \\ \text{s. t.} \quad & \mathbf{B}^\top \mathbf{u} = 0. \end{aligned} \tag{3.34}$$

For better understanding, the consideration in Eq. (3.34) is made for a single time step. Similarly, the consideration can also be applied to the system in Eq. (3.27).
During every time step, the velocity is minimized under the divergence-free equality constraints $\mathbf{B}^\top\mathbf{u} = 0$. In order to solve Eq. (3.34), the aim is to transfer the minimization under constraints into a minimization problem without constraints. Following techniques can be used:

i. **Quadratic penalty methods**: This method belongs to the simplest category, which are typically smooth unrestricted minimization problems written as

$$\min_{\mathbf{u}\in\mathbb{R}^{d_V}} \mathbf{S}\mathbf{u} + k\mathbf{B}\mathbf{p} - \mathbf{g} + \gamma\left(\mathbf{B}^\top\mathbf{u}\right)^2. \tag{3.35}$$

The penalty parameter $\gamma > 0$ increases the objective function if the divergence-free equality constraint is violated. Under suitable conditions, it can be shown that solutions of Eq. (3.35) converge to solutions of the original problem Eq. (3.34) if γ goes to infinity. Furthermore, the objective function in Eq. (3.35) is continuously differentiable because of the squares.

A disadvantage of the quadratic penalty methods is that the matrices are ill-conditioned when the penalty parameter γ goes to infinity. This leads to slow convergence for Newton-type or CG-type solvers. The reason is based on the fact that the extra penalty term in Eq. (3.35) leads locally to a poor approximation.

ii. **Exact penalty methods**: The advantage of this method is that the penalty parameter remains finite, which can significantly reduce the number of iterations. In the case of the exact penalization, also called l_1-penalization, a problem of the form

$$\min_{\mathbf{u}\in\mathbb{R}^{d_V}} \mathbf{S}\mathbf{u} + k\mathbf{B}\mathbf{p} - \mathbf{g} + \gamma|\mathbf{B}^\top\mathbf{u}|. \tag{3.36}$$

is considered. This objective function is non-smooth because of the absolute value. If the functions $\mathbf{S}\mathbf{u} + k\mathbf{B}\mathbf{p} - \mathbf{g}$ and $\mathbf{B}^\top\mathbf{u}$ are convex, the objective function in Eq. (3.36) is also convex.

The exact choice of the penalty parameter γ is usually not clear.

iii. **The augmented Lagrangian method**: As discussed in [28], it is an unrestricted minimization problem with smooth functionals, where the penalty parameter γ does not go to infinity. The method was originally known as the method of multipliers and first introduced in [41] and [58] and also based on successive minimization of the augmented Lagrangian with respect to u.

In each case in Eq. (3.35) and Eq. (3.36), the last summand, the penalty term, increases the objective function when the divergence-free equality constraint is violated. In [10] proposed an

augmented Lagrangian approach for controlling the Schur complement of Eq. (3.25) [34]. The idea is based on the ∇-div stabilization, where, similar to the above mentioned optimization problems, an additional term that does not change the continuous solution is added.

∇-div stabilization

The continuous form of the stabilization replaces Eq. (2.19a) with

$$\frac{\partial u}{\partial t} - \nu\Delta u + u \cdot \nabla u + \nabla p - \gamma \nabla\nabla \cdot u = g \text{ in } \Omega \times (t_0, T], \tag{3.37}$$

for a penalty parameter $\gamma > 0$. As $\nabla \cdot u = 0$, the solutions of Eq. (2.19a) and Eq. (3.37) are the same.

Due to the additional penalty parameter, a similarity to the quadratic penalty method can be expected, such as ill-conditioned matrices for large γ values.

The general idea of a ∇-div stabilization was introduced in the year 1988 in [47]. The stabilization method improve the solution accuracy for Stokes [57] and Navier-Stokes equations [56]. According to [57], [40] and [43], the ∇-div stabilization significantly improves the pressure robustness of the discretization.

Augmented Lagrangian approach

The idea of writing the added penalty term from Eq. (3.37) in a defect notation scaled by the inverse pressure mass matrix can be found in [33] and is written in the discrete case as

$$\begin{bmatrix} \mathbf{S} + \gamma \mathbf{B}\mathbf{M}_{pl}^{-1}\mathbf{B}^\top & k\mathbf{B} \\ \mathbf{B}^\top & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{g} \\ 0 \end{bmatrix} \text{ in } \Omega \times (t_0, T], \tag{3.38}$$

where $\mathbf{M}_{pl}$ is the lumped pressure mass matrix. Furthermore, the penalty parameter γ also remains finite, which is verified in section 4.3.4. In the discrete case, the right-hand-side remains the same and the system matrix from (3.27) is modified as follows:

$$\left[\begin{array}{cccc|ccc} \mathbf{S} + \gamma \mathbf{B}\mathbf{M}_{pl}^{-1}\mathbf{B}^\top & & & & k\mathbf{B} & & \\ \overline{\mathbf{S}} & \mathbf{S} + \gamma \mathbf{B}\mathbf{M}_{pl}^{-1}\mathbf{B}^\top & & & & k\mathbf{B} & \\ & \ddots & \ddots & & & & \ddots \\ & & \overline{\mathbf{S}} & \mathbf{S} + \gamma \mathbf{B}\mathbf{M}_{pl}^{-1}\mathbf{B}^\top & & & k\mathbf{B} \\ \hline \mathbf{B}^\top & & & & 0 & & \\ & \mathbf{B}^\top & & & & 0 & \\ & & \ddots & & & & \ddots \\ & & & \mathbf{B}^\top & & & 0 \end{array}\right] \text{ in } \Omega \tag{3.39}$$

i. **Divergence-free update for the pressure Poisson problem**

In analogy to Eq. (3.4), the pressure Poisson matrix can be constructed by using the augmented Lagrangian from Eq. (3.38), which reads as

$$\mathbf{B}^\top \left(\mathbf{S} + \gamma^{-1}\mathbf{B}\mathbf{M}_{pl}^{-1}\mathbf{B}^\top\right)^{-1} \mathbf{B} \approx \mathbf{B}^\top \mathbf{S}^{-1}\mathbf{B} + \gamma \mathbf{B}^\top \mathbf{B}^{-\top}\mathbf{M}_{pl}\mathbf{B}^{-1}\mathbf{B} \tag{3.40a}$$

$$\approx \mathbf{B}^\top \mathbf{S}^{-1}\mathbf{B} + \gamma \mathbf{M}_{pl}. \tag{3.40b}$$

Thus, from the inverse pressure Poisson matrix

$$\mathbf{P}^{-1} \approx \left(\mathbf{B}^\top \mathbf{S}^{-1} \mathbf{B}\right)^{-1} + \alpha_D \mathbf{M}_{pl}^{-1}, \tag{3.41}$$

with a constant $\alpha_D := \gamma^{-1}$, the additional pressure update in algorithm 3.3 line 17 can be explained, which contributes to faster convergence by enforcing a divergence-free solution. This additional divergence-free pressure update is analyzed numerically in more detail in section 4.1.2.

ii. **Divergence-free update for the Burgers equations**

According to Eq. (3.11b) and Brezzis sufficient condition for well-posedness theorem 2.8, an additional divergence-free update for the velocity should also accelerate the convergence behavior of the solver.

Based on Eq. (3.41), there can be a relationship between the update parameter α_D and γ. Since an optimal range for the update parameter has already been determined numerically in [78], the inverse range can be adopted for the penalty term γ, that is $\gamma = \alpha_D^{-1} = \frac{1}{\theta k \nu}$. For example, for a small viscosity of $\nu = 0.01$ and a small time step size of $k = 0.01$, a relatively large penalty parameter with $\gamma = 20000$ is achieved. Section 4.3.4 shows that the choice of such a high penalty parameter is not the best option. A good value is more likely to be in the range of 20 and lower.

After section 3.2.1, a small time step is required to have exact preconditioners. Low-Reynolds-number flows result in a small viscosity, which leads to a large penalty parameter γ. With the choice of $\gamma = \alpha_D^{-1}$, the problem $\mathbf{S} + \gamma \mathbf{B}^\top \mathbf{M}_{pl}^{-1} \mathbf{B}$ becomes increasingly ill-conditioned for a small time step size, since $\mathbf{B}^\top \mathbf{B}$ has a null space of dimension $\dim V_h - \dim Q_h$. Section 4.3.4 shows that, the equality $\gamma = \alpha_D^{-1}$ does not hold and that a smaller time step leads to a smaller stabilization parameter.

The fixed-point iteration with a Richardson-multigrid solver from algorithm 3.1 cannot solve this kind of problem. The biconjugate gradient stabilized method (BiCG-stab) was tested, but did not lead to convergence. Compared to the BiCG method, the generalized minimal residual (GMRES) method has a minimization property and sounds the most promising. A brief description of the GMRES solver can be found in the next section 3.3.4. Numerical results for a divergence-free velocity can be found in the section 4.3. An approximated divergence-free approach, which is similar to the PP-update, is analyzed in section 4.5.

Using the augmented Lagrangian approach in the time-simultaneous approach was also discovered by [50]. In [11] it is shown that the convergence order is improved. It also arises in the iterated penalty [30] and artificial compressibility [19] method for the Stokes and Navier-Stokes equations.

Modified algorithm for a directly computed divergence-free velocity

According to section 4.3.2, in algorithm 3.4 two velocity arrays $\tilde{\mathbf{U}}$ in line 9 and $\mathbf{U}$ in line 11 are needed for a divergence-free solver, which contain the computed velocity vectors for all time steps. Since the pressure Poisson problem depends in line 18 on the divergence-defect, it is necessary to compute and store a non-divergence-free velocity as well. Note the choice

$$\mathbf{f}_p = \frac{1}{k} \mathbf{B}^\top \tilde{\mathbf{u}}^l - \frac{1}{k} \mathbf{B}^\top \mathbf{u}^{l-1}, \tag{3.42}$$

where the defect of the last time step $\mathbf{B}^\top \mathbf{u}^{l-1}$ is taken with the (almost) divergence-free velocity $\mathbf{u}^{l-1}$ and a non-divergence-free velocity $\tilde{\mathbf{u}}^l$ is used for the current time step. Section 4.3.1, section 4.3.2 and also section 4.5.3 illustrate the necessity of this choice.

Algorithm 3.4: Time-simultaneous PP-solver using augmented Lagrangian

input : initial value $\mathbf{u}^0 = u_0$ $(\mathbf{p}^0 = 0)$
output: time-simultaneous: solution vector of each time step $\mathbf{U}$, $\mathbf{P}$

1 **initialize** $\mathbf{U}$, $\tilde{\mathbf{U}}$ // setting up velocity starting values
2 **initialize** $\mathbf{P}$ // setting up pressure starting values
3 **do**
4 **if** *main cores* **then**
5 **for** $l \leftarrow 1$ **to** K **do**
6 /* solve for each time step an intermediate velocity $\tilde{\mathbf{u}}$ */
7 $\mathbf{p}^{l-1} \leftarrow \mathbf{P}(l-1)$ // get pressure approximate
8 $\tilde{\mathbf{u}}^l \leftarrow \langle \textbf{solve} \;\; \mathbf{S}\tilde{\mathbf{u}}^l = \mathbf{g} - k\mathbf{B}\mathbf{p}^{l-1} \rangle$ // compute non-divergence-free approximation
9 $\tilde{\mathbf{U}}(l) \leftarrow \tilde{\mathbf{u}}^l$ // save velocity for the pressure Poisson problem
10 $\mathbf{u}^l \leftarrow \langle \textbf{solve} \;\; \mathbf{S}\tilde{\mathbf{u}}^l + \gamma \mathbf{B}\mathbf{M}_{pl}^{-1}\mathbf{B}^\top \tilde{\mathbf{u}}^l = \mathbf{g} - k\mathbf{B}\mathbf{p}^{l-1} \rangle$ // compute divergence-free $\mathbf{u}$
11 $\mathbf{U}(l) \leftarrow \mathbf{u}^l$ // save divergence-free velocity for the pressure Poisson problem
12 **else**
13 **mpi wait**
14 **do in parallel** $l \leftarrow 1$ **to** K (parallel in space and time)
15 **if** *free cores* **then**
16 **mpi sync:** $\mathbf{u}^{l-1} \leftarrow \mathbf{U}(l-1)$, $\tilde{\mathbf{u}}^l \leftarrow \tilde{\mathbf{U}}(l)$ // get velocity from main cores
17 $\mathbf{f}_p \leftarrow \frac{1}{k}\mathbf{B}^\top \tilde{\mathbf{u}}^l - \frac{1}{k}\mathbf{B}^\top \mathbf{u}^{l-1}$ // derive adjusted right-hand-side
18 $\mathbf{q} \leftarrow \langle \textbf{solve} \;\; \mathbf{P}\mathbf{q} = \mathbf{f}_p \rangle$
19 $\mathbf{p}^l \leftarrow \mathbf{p}^l + \alpha_R \mathbf{q} + \alpha_D \mathbf{M}_{\mathrm{pl}}^{-1}\mathbf{f}_p$ // update current pressure
20 **mpi sync:** $\mathbf{P}(l) \leftarrow \mathbf{p}^l$ // send pressure to main cores
21 **while** $\Big(\Big(\boldsymbol{B}^\top \boldsymbol{u} \leq tol \Big) \;\; \& \;\; (\|\boldsymbol{S}\boldsymbol{u} + k\boldsymbol{B}\boldsymbol{p} - \boldsymbol{g}\| \leq tol) \Big)$;

As discussed in chapter 1, the PP-solver computes a solution for the nonlinear momentum equations very quickly compared to the pressure Poisson solver. The computational efficiency is based on the simple divergence-free update in algorithm 3.1 line 8, which is nothing more than a matrix vector multiplication. It can be assumed, if a complete divergence-free velocity were calculated directly, convergence would be achieved with almost one outer iteration. However, a large amount of computation from the outer iterations would be outsourced to the velocity solver. A computing time chart for the GMRES velocity solver can be found in section 4.3.1.

Remark

It is notable that the divergence applied on gradient equals the Laplace operator, but this is not the same as gradient applied on the divergence operator. It holds:

$$\nabla\nabla \cdot u = \Delta u + \nabla \times \nabla \times u \tag{3.43}$$

with $\nabla\times$ as the rotation operator. Since the GMRES solver contains a rotation within the algorithm, it is another reason why it is particularly suited to compute the rotated problem of the ∇-div stabilization with GMRES. The augmented Lagrangian or ∇-div stabilization is therefore the addition of *rotated* artificial diffusion.

Furthermore, in the next chapter in section 4.5 is a different approach presented, which computes a PP-like update, but satisfies a divergence-free velocity only approximately.

3.3.4 Multigrid in time (recoupling with FGMRES)

For the development of a fast time-simultaneous PP-solver, it is necessary to perform only a few, in the best case only one, outer iteration. An idea for decreasing the number of outer iterations is to embed the solver in a multigrid in time implementation. In the best case, a similar or even better accuracy to that of the sequential-in-time PP-solver is gained after one multigrid in time step. The idea of the multigrid in time method is based on [51] which uses the GMRES method. A memory-friendly local FGMRES approach is being used instead of the global GMRES approach from [51].

GMRES (FGMRES)

The GMRES algorithm was developed in 1986 by [65] and is an iterative method for solving non-symmetric linear systems of the form $\mathbf{Ax} = \mathbf{f}$. The method is described in algorithm 3.5 written below and has the property of minimizing the norm of the defect vector $\mathbf{r} = \mathbf{f} - \mathbf{Ax}$ over a Krylov subspace at every step.

Lines 6 to 9 contain Arnoldi's method from [4], which uses the Gram-Schmidt method to compute an l_2-orthonormal basis $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m\}$ of the Krylov subspace $K_m = \text{span}\{\mathbf{v}_1, \mathbf{Av}_1, \ldots, \mathbf{A}^{m-1}\mathbf{v}_1\}$. Thus, the Arnoldi process constructs a $\mathbb{R}^{m\times m}$ Hessenberg matrix $\mathbf{H}_m = \mathbf{V}_m^\top \mathbf{A}\mathbf{V}_m$. The scalars h_{ij} are defined in the algorithm 3.5. Hereby is $\mathbf{V}_m$ a $d_V \times m$ matrix whose columns are the l_2-orthonormal basis $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m\}$.

The m-th iterate minimizes the residual in the Krylov subspace K_m. Since every subspace is contained in the next subspace, the residual does not increase. After d_V iterations the Krylov space K_{d_V} is the whole of $\mathbb{R}^{d_V}$ and hence the GMRES method reach at the exact solution. However, the idea is that after a small number of iterations relative to d_V, the vector x_m is already a good approximation to the exact solution.

Algorithm 3.5: GMRES

input : Matrix $\mathbf{A}$, right-hand-side $\mathbf{f}$ and initial value $\mathbf{x}_0$ // solve linear system $\mathbf{Ax} = \mathbf{f}$
output: Approximation $\mathbf{x}$

1 $\mathbf{r}_0 = \mathbf{f} - \mathbf{Ax}_0$ // set initial defect
2 $\mathbf{v}_0 = \frac{\mathbf{r}_0}{\|\mathbf{r}_0\|}$ // normalize
3 **for** $j \leftarrow 1$ **to** m **do**
4 **apply** preconditioner
5 /* Arnoldi's method */
6 $h_{i,j} \leftarrow (\mathbf{Av}_j, \mathbf{v}_i)$, for $i = 1, \ldots, j$,
7 $\hat{\mathbf{v}}_{j+1} \leftarrow \mathbf{Av}_j - \sum_{i=1}^{j} h_{i,j}\mathbf{v}_i$, // Gram-Schmidt process
8 $h_{j+1,j} \leftarrow \|\hat{\mathbf{v}}_{j+1}\|$,
9 $\mathbf{v}_{j+1} \leftarrow \frac{\hat{\mathbf{v}}_{j+1}}{h_{j+1,j}}$ // normalize
10 $\overline{\mathbf{H}} \leftarrow$ **apply** Givens rotations to $\mathbf{H}$
11 $\beta \leftarrow \sqrt{h_{j,j}^2 + h_{j+1,j}}$
12 /* form the approximate solution: */
13 $\mathbf{y}_m \leftarrow \min_{\mathbf{y}\in\mathbb{R}^m} \|\beta\mathbf{e}_1 - \overline{\mathbf{H}}\mathbf{y}\|$
14 $\mathbf{x}_m \leftarrow \mathbf{x}_0 + \mathbf{V}_m\mathbf{y}_m$ // update solution vector

Based on [65], the following convergence estimate applies to the GMRES method for a positive real matrix $\mathbf{A}$:

$$\|\mathbf{r}_n\| \leq \left(1 - \frac{\lambda_{\min}^2\left(\frac{1}{2}\left(\mathbf{A}^\top + \mathbf{A}\right)\right)}{\lambda_{\max}\left(\mathbf{A}^\top \mathbf{A}\right)}\right), \tag{3.44}$$

where $\lambda_{\min}(\cdot)$ and $\lambda_{\max}(\cdot)$ denote the smallest and largest eigenvalues of the corresponding matrix, respectively.

The flexible-GMRES (FGMRES) solver mentioned in [64] is a variant of the GMRES solver that can handle a wider class of preconditioners. For example, any iterative solver can be used as a preconditioner for FGMRES. The fact that FGMRES uses right preconditioning results in the same convergence criterion as GMRES with right preconditioning. The disadvantage of this solver is that it requires twice as much memory for the same number of iterations as the GMRES method.

FGMRES smoother for the coupled system

According to tests by [51], a GMRES smoother can be applied to the multigrid in time algorithm. This is nothing unusual, as according to [77] GMRES is often used in a multigrid context to solve large linear systems that occur in CFD using the incompressible Stokes or Navier-Stokes equations. Also [24] presents studies about the GMRES and FGMRES method in the space-time framework.

The divergence-defect can be further reduced by a coupled system, which is written in the defect notation

$$\begin{bmatrix} \mathbf{S} & k\mathbf{B} \\ \mathbf{B}^\top & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} 0 \\ -\hat{\mathbf{I}}_{\mathtt{lvl}_t-1}^{\mathtt{lvl}_t} \mathbf{B}^\top \tilde{\mathbf{u}} \end{bmatrix}, \tag{3.45}$$

with $\hat{\mathbf{I}}_{\mathtt{lvl}_t-1}^{\mathtt{lvl}_t}$ as the restriction matrix in time, as discussed in section 3.2.3. Instead of completely solving the system, only a few FGMRES steps are required. The system in Eq. (3.45) can be solved up to machine precision, but this should not be the aim. The focus should be on efficiency and an accuracy that corresponds to the PP-solver in section 3.2.

Furthermore, it should be kept in mind that in Eq. (3.45) only the pressure values are known during the calculation but not the divergence-defect $\mathbf{B}^\top \tilde{\mathbf{u}}$. According to Eq. (3.11b), the divergence-defect of the velocity can be derived out of the given pressure values, written as

$$\mathbf{B}^\top \tilde{\mathbf{u}} = \mathbf{B}^\top \mathbf{S}^{-1}(\mathbf{g} - k\mathbf{B}\mathbf{p}). \tag{3.46}$$

This is no different than from solving the divergence-free problem in algorithm 3.3 line 8 with the addition of applying the divergence matrix.

Prolongation and restriction in time

The coarse time grid uses half of the given time step $\frac{k}{2}$. A coarser time grid size has a much more influence on the computation time of the solver, since the preconditioners from section 3.2.1 become inaccurate. The mapping of the pressure vectors to the coarser grid depends on the discretization in time from section 2.3.4.
For the Crank-Nicolson method, the location of the degrees of freedom in time should be taken into account. The actual pressure is defined at different points in time than the pressure solution vectors. This has direct consequences on the choice of the prolongation and restriction matrix. More information and a detailed derivation of the prolongation and restriction operators can be found in [45].
According to [51], the restriction matrix and the prolongation matrix are written for the two-grid in time as

$$\hat{\mathbf{I}}_{\mathtt{lvl}_t-1}^{\mathtt{lvl}_t} := \begin{bmatrix} 0.5 & 0.5 & & & & & & \\ & & 0.5 & 0.5 & & & & \\ & & & & \ddots & \ddots & & \\ & & & & & & 0.5 & 0.5 \end{bmatrix}, \tag{3.47a}$$

$$\hat{\mathbf{I}}_{\mathtt{lvl}_t}^{\mathtt{lvl}_t-1} := \begin{bmatrix} 0.75 & & & & & \\ 1.25 & & & & & \\ 0.25 & 0.75 & & & & \\ -0.25 & 1.25 & & & & \\ & 0.25 & 0.75 & & & \\ & -0.25 & 1.25 & & & \\ & & \ddots & \ddots & & \\ & & & 0.25 & 0.75 \\ & & & -0.25 & 1.25 \end{bmatrix}. \tag{3.47b}$$

Conclusion and remarks

Advantages of the FGMRES multigrid in time approach are:

i. An exact representation of the incompressible condition by the coupled system in Eq. (3.45). Therefore, the exact numerical solution can be achieved similar to the CC approach mentioned in section 3.1.1. The errors resulting from the decoupling approach are eliminated by the recoupling. Machine precision can already be achieved during the first outer iteration.

ii. The additional divergence-free treatment in Eq. (3.45) leads according to Eq. (3.11b) and Brezzis sufficient condition for well-posedness theorem 2.8 to a faster convergence and thus a better pressure approximation.

iii. As shown in section 4.4, only a few FGMRES iterations are required to continuously improve the pressure approximation. The edges and corners are accurately approximated after just a few FGMRES iterations.

iv. Instead of setting up a large coupled system as in [51], that includes all time steps, only small coupled systems are computed sequential-in-time, which contain one time step. The advantage is that these problems can later be solved in parallel-in-time using the parareal approach. This approach is also more memory-friendly.

Disadvantages are:

i. A coupled system must be assembled with the corresponding prolongation and restriction matrices. This requires new data types, which are time-consuming to implement in the FEAT3 software. A slight increase in memory consumption is also to be expected.

ii. The right-hand-side, i.e. the defect of the divergence-free velocity, must be computed in the coupled system in Eq. (3.45) before starting the FGMRES smoother.

iii. The coupled system ensures a smaller divergence-defect, but only after all velocities have been computed in the first (outer) iteration. In the case of a convective term, this recoupled multigrid in time method is without a divergence-free velocity approach not sufficient.

iv. The system is coupled again, which is a contradiction to the idea of a decoupled approach.

Remark:

Additional penalty of the coupled system, such as through the augmented Lagrangian from section 3.3.3 using

$$\begin{bmatrix} \mathbf{S} + \gamma \mathbf{B} \mathbf{M}_{pl}^{-1} \mathbf{B}^\top & k\mathbf{B} \\ \mathbf{B}^\top & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} 0 \\ -\mathbf{I}_{\mathrm{lvl}_t - 1}^{\mathrm{lvl}_t} \mathbf{B}^\top \mathbf{u} \end{bmatrix}, \quad (3.48)$$

does not bring any improvement and instead causes convergence problems.

Figure 3.17 shows the pressure accuracy for 10 and 20 GMRES iterations of Eq. (3.48). It can be observed that there is nearly no improvement in pressure, but also no deterioration due to the update.

Table 3.2 presents the results for 30 GMRES iterations for each time step. It can be seen that a divergence-free velocity is lost after the first few time steps, which leads to an inaccurate pressure approximation.

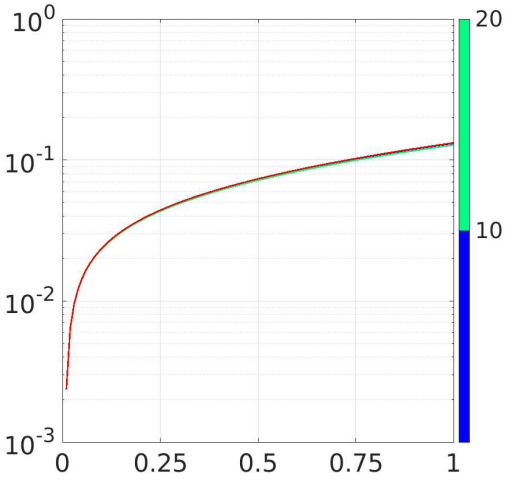


Figure 3.17: Pressure L^2 error during the first (outer) iteration for 10 (blue) and 20 (green) GMRES iterations of Eq. (3.48) at each time step, without the multigrid in time approach. The pressure error before solving the coupled system is shown as reference (red). The x-axis represents the time interval [0,1].

time	0.01	0.02	0.03	0.04	0.05	0.06	0.07
$\|p - p_h\|_{L^2}$	2.392e-03	6.604e-03	1.004e+00	5.934e+04	3.861e+09	3.244e+14	3.271e+19
$\mathbf{B}^\top \mathbf{u}$	1.253e-05	8.598e-05	4.607e-02	3.067e+03	2.070e+08	1.761e+13	1.785e+18

Table 3.2: Pressure L^2 error and divergence-defect for 30 GMRES iterations of Eq. (3.48) during the first (outer) iteration for the first time instants.

Chapter 4

Numerical simulations

Contents

In this chapter, the update behavior of the time-simultaneous solver presented in section 3.3 is numerically investigated in detail. For a direct comparison with the sequential-in-time solver from section 3.2, a reference solution from the PP-solver is provided.

The correct configuration for the solver is determined using diverse non-stationary Poiseuille flows for the Stokes equations. The Stokes equations are the best option for validating the solver because it provides an analytical solution.
Poiseuille flow consists in finding the flow of an incompressible fluid flowing through a circular pipe, as illustrated in figure 4, in which the flow is caused by the pressure gradient. A review of the development and also a stability analysis of Poiseuille flow in a pipe can be found in [25].

Consider the time dependent Stokes equations with three different right-hand-sides

$$\frac{\partial}{\partial t}u - \nu\Delta u + \nabla p = \begin{pmatrix} 2\nu - 2\nu q_i(t) \\ 0 \end{pmatrix}, \quad (4.1)$$

with the parameters

$$q_1(t) := 1 + t, \quad (4.2a)$$
$$q_2(t) := e^t, \quad (4.2b)$$
$$q_3(t) := 1 + \frac{\sin(2\pi t)}{2}. \quad (4.2c)$$

Using a simple parabolic profile with the solution

$$u(x, y, t) := u(y) := \begin{pmatrix} y(1-y) \\ 0 \end{pmatrix} \quad (4.3)$$

leads to the analytically pressure solution

$$p(x, y, t) := p(x, t) := \nu(1 - 2x)q_i(t) + c_i, \quad (4.4)$$

with constants c_i, for $i = 1, 2, 3$.

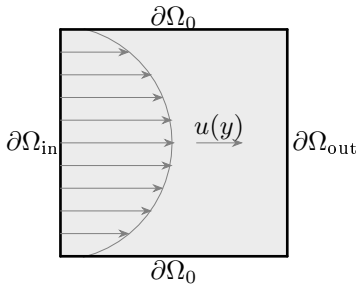


Figure 4.1: Poiseuille flow on the unit square domain, with initial condition

$$u(x, y, 0) = 0, \quad (4.5a)$$
$$p(x, y, 0) = 0, \quad (4.5b)$$

as well as a null Dirichlet condition for the wall defined by $\partial\Omega_0 := 0$, an inflow condition given by $\partial\Omega_{\text{in}} := u(y)$ and $\partial\Omega_{\text{out}}$ as the do nothing condition from section 2.4.

For the simulations, a time step $k = 0.01$ is used. The end time is set to $T = 1$, which results in a total number of $K = 100$ blocked time steps. This test is intended to verify the accuracy of the time-simultaneous solver in order to determine whether machine precision can be achieved even with non-optimal time step sizes. Computations for a large number of blocked time steps and also smaller time-step sizes are performed in the upcoming chapter 5.

Following computations for the Poiseuille flow are executed on refinement level 5.

4.1 Pressure-update

In the configuration mentioned above, the Crank-Nicolson method from section 2.3.4 is taken as the time stepping method, the viscosity is set to $\nu := 0.01$ and the symmetric deformation tensor ε from Eq. (2.16b) is used. In order to avoid the problems mentioned in section 2.3.3 regarding spurious modes that occur with the Q_2/Q_1 element, the update behavior is first tested with the Q_2/P_1^{disc} element.

For large time steps like $k = 0.01$ there is no expectation of achieving high accuracy from the PP-solver, the preconditioners become inaccurate and a good approximation of the solution cannot be guaranteed. The aim of the first test should be to achieve at least the same or better accuracy than the PP-solver in all cases.

4.1.1 Reactive update: $\alpha_R = 1$

Applying only the reactive update, i.e., $\alpha_D = 0$ and $\gamma = 0$ for the divergence-free update mentioned in section 3.3.3 leads to the following numerical results, where in each time step the momentum equations are computed up to the machine precision.

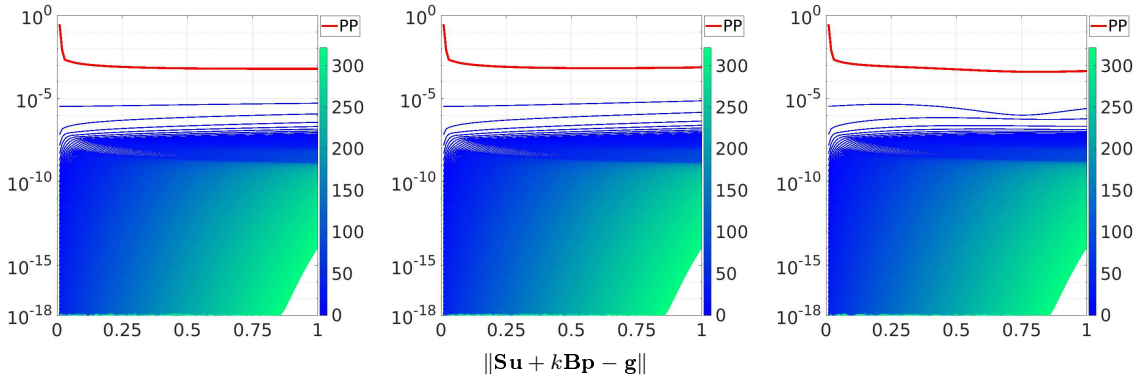


Figure 4.2: Residual for $q_1(t) := 1 + t$ (left), $q_2(t) := e^t$ (center) and $q_3(t) := 1 + \frac{\sin(2\pi t)}{2}$ (right) with the PP reference (in red) over the time interval [0,1].

	Residual												
	PP	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}
$q_1(1)$	1	1	1	1	3	7	47	107	262	282	298	311	322
$q_2(1)$	1	1	1	1	3	8	50	114	262	282	298	311	322
$q_3(1)$	1	1	1	1	2	6	53	203	263	283	298	311	322

Table 4.1: Number of outer iterations for the residual at the endpoint $T = 1$ for $q_i(t)$, $i = 1, 2, 3$. The number of outer iterations required to reach the PP reference is highlighted (in red).

The residuals in figure 4.2 and table 4.1 are defined as the defect of the system without the divergence-free condition, which is

$$\|\mathbf{S}\mathbf{u}^l + k\mathbf{B}\mathbf{p}^l - \mathbf{g}^l\|, \text{ for } l = 1, \ldots, K. \tag{4.6}$$

Each line in figure 4.2 describes an outer iteration of the TSPP-algorithm. The colorbar on the right side indicates the number of outer iterations. Without a divergence-free update, over 300 outer iterations are required to achieve machine precision. Since a relatively large time step of $k = 0.01$ was selected, the accuracy for the PP residual is with the range of 10^{-4} quite poor and achieved after just one TSPP iteration. A better overview of the number of iterations can be seen in table 4.1.

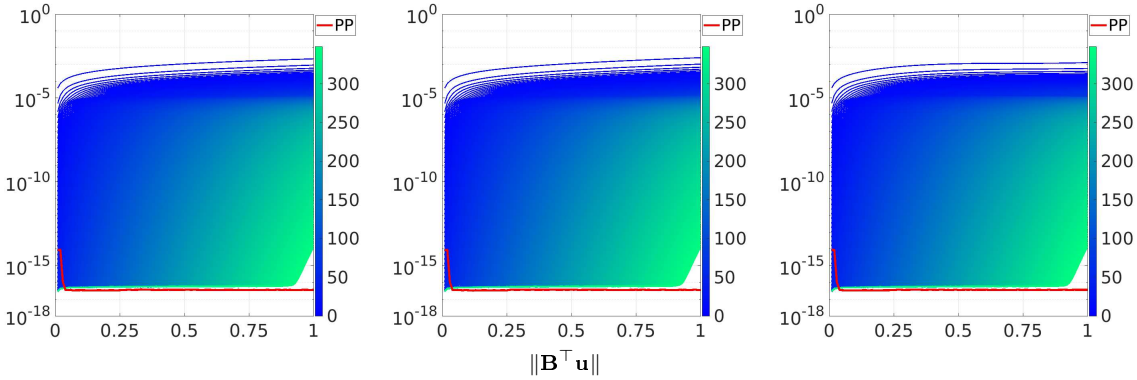


Figure 4.3: Divergence-defect for $q_1(t) := 1 + t$ (left), $q_2(t) := e^t$ (center) and $q_3(t) := 1 + \frac{\sin(2\pi t)}{2}$ (right) with the PP reference (in red) over the time interval [0,1].

	Divergence-defect												
	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}	PP
$q_1(1)$	2	22	118	227	259	278	293	306	318	329	339	348	367
$q_2(1)$	3	23	121	227	259	278	293	306	318	329	339	348	367
$q_3(1)$	2	18	128	229	260	279	294	307	318	329	339	348	367

Table 4.2: Number of iterations for the divergence-defect at the endpoint $T = 1$ for $q_i(t)$, $i = 1, 2, 3$. The number of outer iterations required to reach the PP reference is highlighted (in red).

According to Eq. (3.11b), convergence is only achieved if a divergence-free velocity is guaranteed. In comparison to figure 4.2, the exact opposite is seen in the case of divergence-defect: The PP reference immediately achieves machine precision. As the PP accuracy cannot be fully achieved, table 4.2 assumes 10^{-16} as the lower limit, which is reached after 367 iterations.

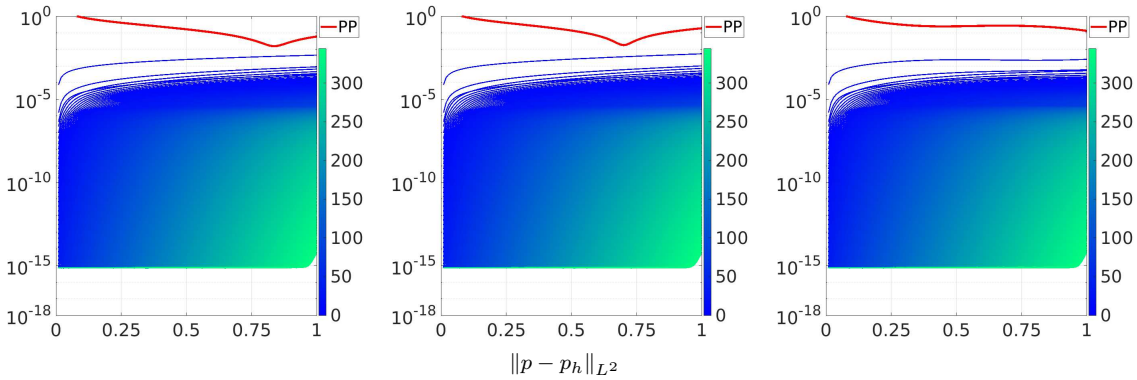


Figure 4.4: Pressure L^2-error for $q_1(t) := 1 + t$ (left), $q_2(t) := e^t$ (center) and $q_3(t) := 1 + \frac{\sin(2\pi t)}{2}$ (right) with the PP reference (in red) over the time interval [0,1].

	$\|p-p_h\|_{L^2}$												
	PP	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}
$q_1(1)$	1	2	9	72	208	252	273	289	302	314	325	335	345
$q_2(1)$	1	3	10	74	208	252	273	289	302	314	325	335	345
$q_3(1)$	1	2	6	73	213	253	273	289	302	314	325	335	345

Table 4.3: Number of iterations for the pressure L^2 error at the endpoint $T = 1$ for $q_i(t)$, $i = 1, 2, 3$. The number of outer iterations required to reach the PP reference is highlighted (in red).

Since the analytical solution is known in the case of Stokes equations, the L^2 error for the pressure is computed and provided in figure 4.4 and table 4.3. This also validates and confirms the correctness of the solver.

It should be emphasized that in figure 4.4 the time-simulated solver already produces a better solution for the pressure in the L^2 error after the first outer iteration for large time steps k.

Figure 4.5 also shows that similarly good results for the L^2 and H^1 error are achieved for the velocity after one outer iteration compared to the PP-solver.

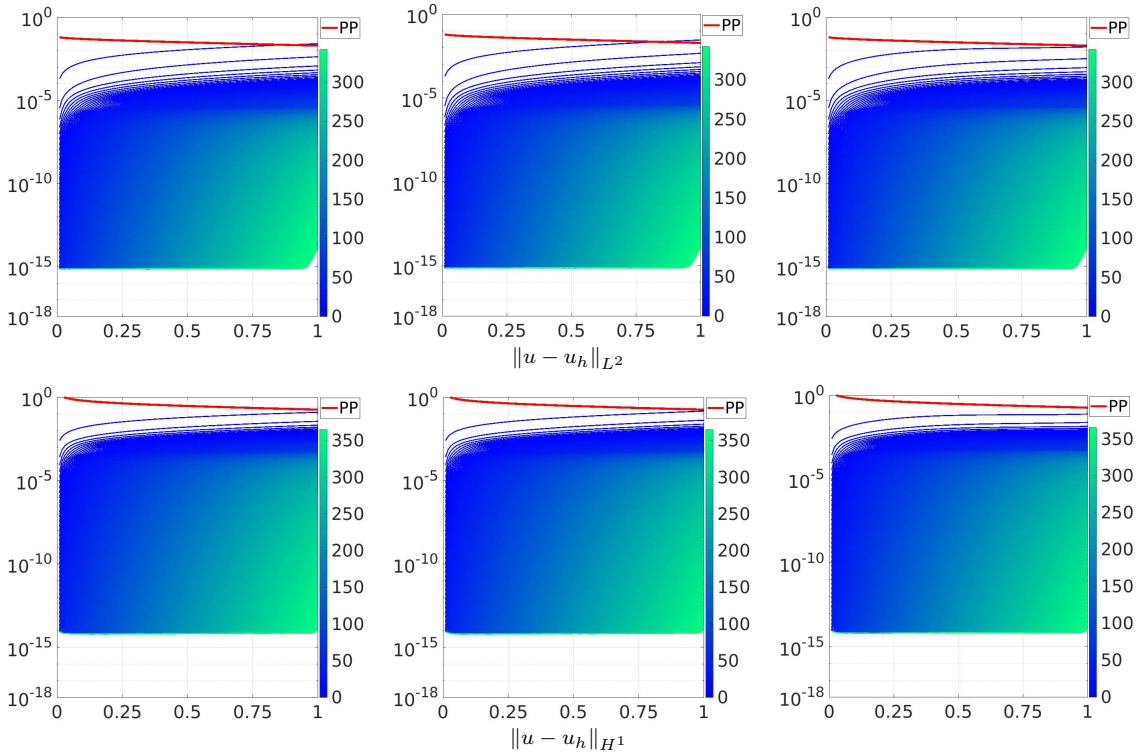


Figure 4.5: Velocity L^2-error (top) and H^1 error (bottom) for $q(t) := 1+t$ (left), $q(t) := e^t$ (center) and $q(t) := 1 + \frac{\sin(2\pi t)}{2}$ (right) with the PP reference (in red) over the time interval [0,1].

	$\|u-u_h\|_{L^2}$												
	PP	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}
$q_1(1)$	2	4	15	69	180	242	267	284	298	311	322	332	342
$q_2(1)$	2	4	16	70	180	242	267	284	298	311	322	332	342
$q_3(1)$	1	3	14	71	187	243	267	284	299	311	322	332	342
	$\|u-u_h\|_{H^1}$												
	PP	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}
$q_1(1)$	1	71	209	253	274	290	303	315	326	336	346	355	365
$q_2(1)$	1	72	209	253	274	290	303	315	326	336	346	355	365
$q_3(1)$	1	72	213	254	274	290	303	315	326	336	346	355	365

Table 4.4: Number of outer iterations for the velocity L^2 error (top) and H^1 error (bottom) at the endpoint $T = 1$ for $q_i(t)$, $i = 1, 2, 3$. The number of outer iterations required to reach the PP reference is highlighted (in red).

4.1.2 Reactive and diffusive update: $\alpha_R = 1, \alpha_D = \theta k \nu$

It has been shown that the PP-solver does not generate good accuracy for comparatively large time step sizes k, but immediately fulfills the divergence-free condition. The divergence-free criterion is according to Eq. (3.11b) not only important for convergence from an algorithmic point of view, but is according to Brezzi's theorem 2.8 also a necessary criterion that must be fulfilled in theory for the existence and uniqueness of the solution.

The influence on the divergence-free update in the pressure space is examined by setting the parameter $\alpha_D := \theta k \nu$.

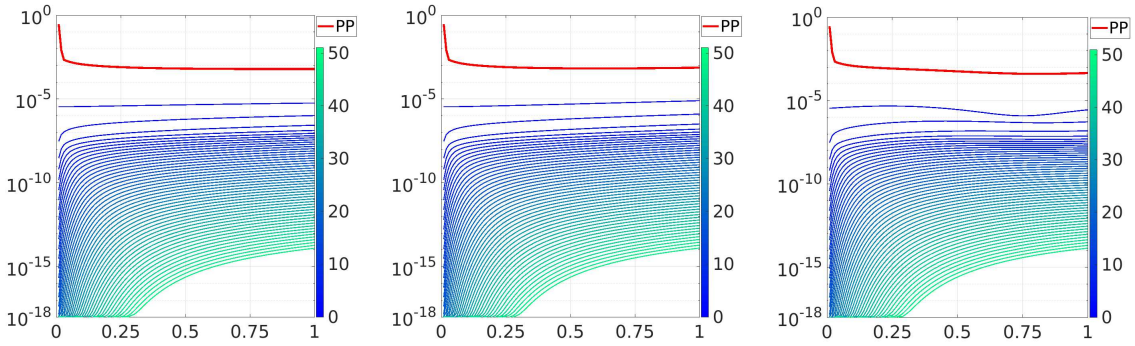


Figure 4.6: Residual for $q_1(t) := 1 + t$ (left), $q_2(t) := e^t$ (center) and $q_3(t) := 1 + \frac{\sin(2\pi t)}{2}$ (right) with the PP reference (in red) over the time interval [0,1].

	Residual												
	PP	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}
$q_1(1)$	1	1	1	1	3	5	13	20	26	33	39	45	51
$q_2(1)$	1	1	1	1	3	6	13	20	27	33	39	45	51
$q_3(1)$	1	1	1	1	2	4	11	19	26	33	39	45	51

Table 4.5: Number of outer iterations for the residual at the endpoint $T = 1$ for $q_i(t)$, $i = 1, 2, 3$. The number of outer iterations required to reach the PP reference is highlighted (in red).

In table 4.5, it can be seen immediately that, compared to table 4.1, approx. 80% less outer iterations are required to obtain the same accuracy. Since the divergence-free update according to algorithm 3.3 only contains a matrix vector multiplication, practically no additional computational effort is required for the update.

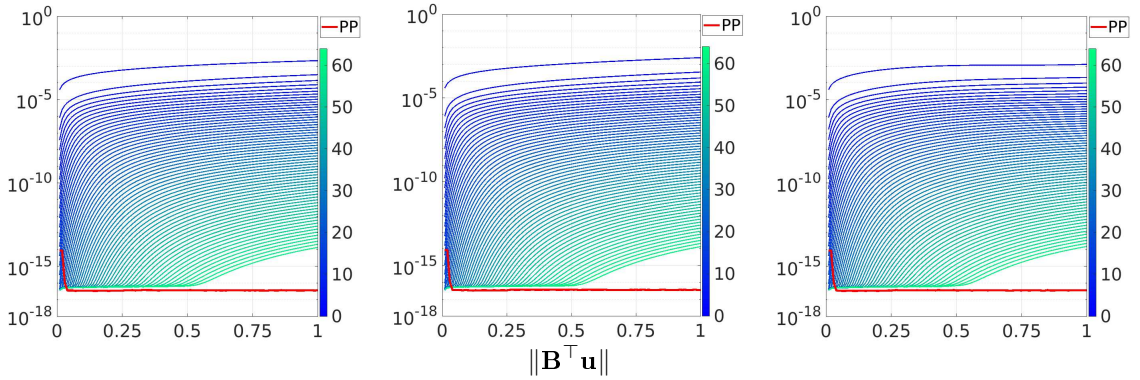


Figure 4.7: Divergence-defect for $q_1(t) := 1 + t$ (left), $q_2(t) := e^t$ (center) and $q_3(t) := 1 + \frac{\sin(2\pi t)}{2}$ (right) with the PP reference (in red) over the time interval [0,1].

	Divergence-defect												
	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}	PP
$q_1(1)$	2	4	10	16	23	29	35	41	47	53	59	64	75
$q_2(1)$	2	4	10	16	23	29	36	42	47	53	59	64	75
$q_3(1)$	2	3	9	16	22	29	35	42	48	53	59	64	75

Table 4.6: Number of iterations for the divergence-defect at the endpoint $T = 1$ for $q_i(t)$, $i = 1, 2, 3$. The number of outer iterations required to reach the PP reference is highlighted (in red).

	$\|p - p_h\|_{L^2}$												
	PP	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}
$q_1(1)$	1	2	4	10	16	23	29	35	41	47	53	59	64
$q_2(1)$	1	2	4	10	16	23	29	35	41	47	53	59	64
$q_3(1)$	1	2	4	9	16	23	29	36	42	48	53	59	64
	$\|u - u_h\|_{L^2}$												
	PP	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}
$q_1(1)$	2	3	5	10	16	22	28	34	40	46	52	57	63
$q_2(1)$	2	3	5	10	16	22	28	34	40	46	52	58	63
$q_3(1)$	1	3	5	9	15	22	28	34	41	46	52	58	63
	$\|u - u_h\|_{H^1}$												
	PP	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}
$q_1(1)$	1	8	14	21	27	34	40	46	51	57	63	68	74
$q_2(1)$	1	8	14	21	27	34	40	46	51	57	63	68	74
$q_3(1)$	1	7	13	20	27	34	40	46	52	57	63	68	74

Table 4.7: Number of outer iterations for the pressure L^2 error (top), the velocity L^2 error (center) and the velocity H^1 error (bottom) at the endpoint $T = 1$ for $q_i(t)$, $i = 1, 2, 3$. The number of outer iterations required to reach the PP reference is highlighted (in red).

Compared with table 4.2 and table 4.6, it shows that approx. 80% of the outer iterations are saved to obtain a divergence-free velocity. A similar behavior can be observed with the errors in table 4.7.

In conclusion, the above observation for the time-simultaneous solver agrees with [46], which described a huge improvement due to the divergence-free update in the pressure space for the sequential-in-time PP-solver.

4.1.3 Over-relaxation of α_D

The simple idea is to achieve even faster convergence with a larger value for α_D. Figure 4.8 and figure 4.9 show the results of an over-relaxated parameter α_D for the function $q_1(t) := 1 + t$.

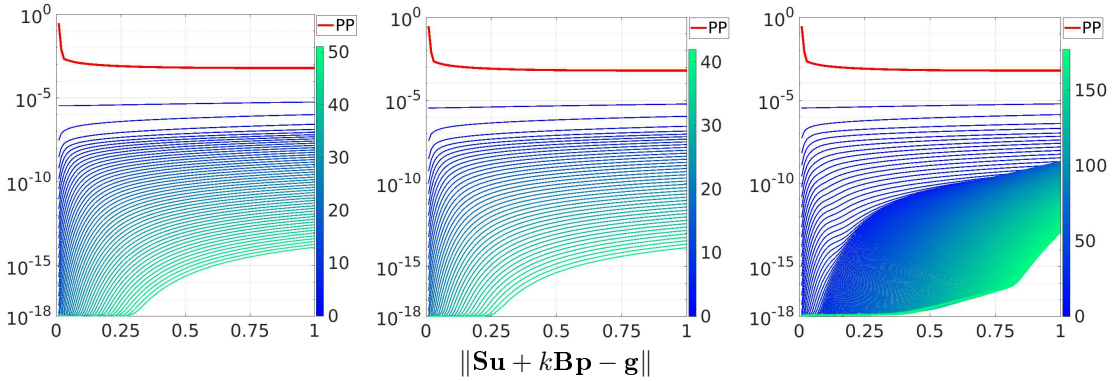


Figure 4.8: Residual for the function $q_1(t) := 1 + t$ with the diffusive update parameter $\alpha_D := \theta k\nu$ (left), $\alpha_D := 1.25\,(\theta k\nu)$ (center) and $\alpha_D := 1.5\,(\theta k\nu)$ (right) over the time interval [0,1].

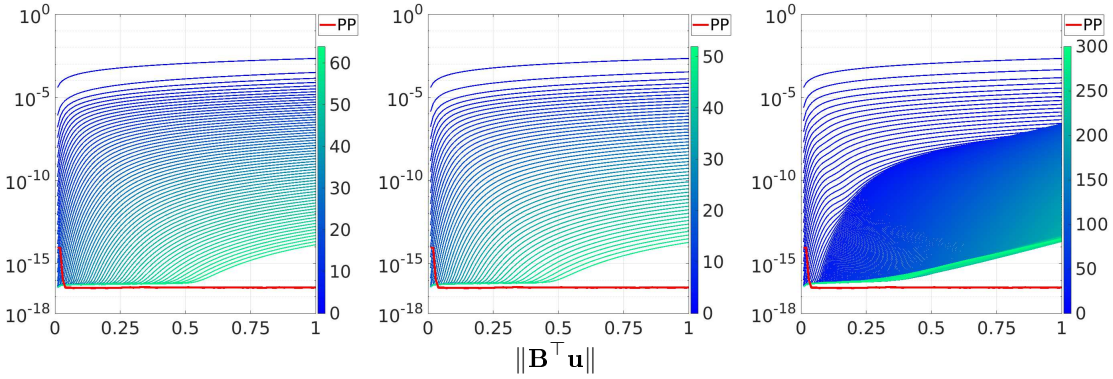


Figure 4.9: Divergence-defect for the function $q_1(t) := 1 + t$ with the diffusive update parameter $\alpha_D := \theta k\nu$ (left), $\alpha_D := 1.25\,(\theta k\nu)$ (center) and $\alpha_D := 1.5\,(\theta k\nu)$ (right) over the time interval [0,1].

Figure 4.8 and figure 4.9 show, a slight over-relaxation with 1.25 can save an additional 15% of outer iterations. In case of an over-relaxation with 1.5, it can be seen on the one hand that convergence problems occur after several (outer) iterations, but a slightly faster convergence can be achieved in the early (outer) iterations.

A similar improvement can be observed for $q_2(t)$ and $q_3(t)$ by a moderate over-relaxation and also for the L^2 and H^1 errors.

An improvement can be seen in the first few outer iterations of the overrelaxing. A detailed view of the divergence-free pressure update vector is shown in figure 4.10. Here it is clear that, especially in the first few iterations, correction only takes place at the inflow (and outflow) boundary. An adaptive over-relaxation strategy starting with a higher parameter of α_D only for the first iterations with $1.5\,(\theta k\nu) \to 1.0\,(\theta k\nu)$ could save some more outer iterations, but this is not further investigated in this work.

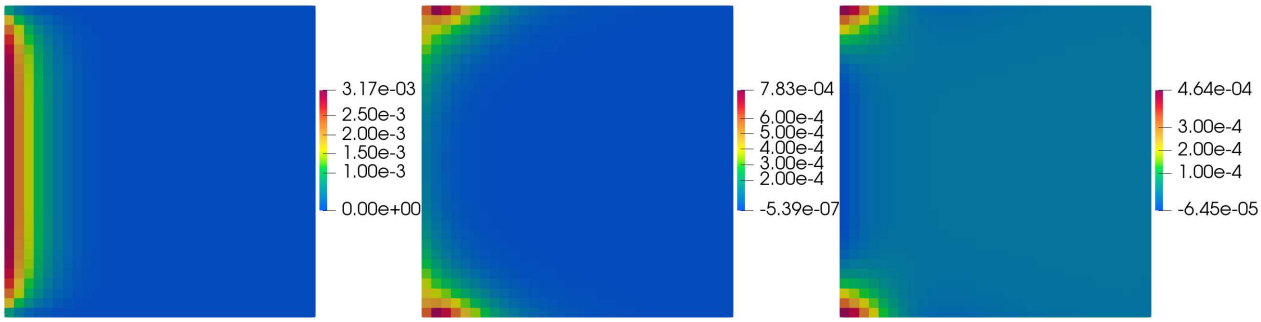


Figure 4.10: Diffusive update vector at the first (left), second (center) and third (right) outer iteration.

A further extension in the pressure space does not bring any further improvement, as the following example illustrates:

$$\mathbf{p}^l = \mathbf{p}^{l-1} + \alpha_R \mathbf{q} + \alpha_D \mathbf{M}_{\text{pl}}^{-1} \mathbf{f}_p \pm k \mathbf{B}^\top \mathbf{M}_l^{-1} \mathbf{B} \mathbf{q} \tag{4.7a}$$

$$= \mathbf{p}^{l-1} + \tilde{\alpha}_R \mathbf{q} + \alpha_D \frac{1}{k} \mathbf{M}_{\text{pl}}^{-1} \mathbf{f}_p. \tag{4.7b}$$

The additional extension of an update $\mathbf{q}$ scaled by a $\mathbf{B}^\top \mathbf{M}_l^{-1} \mathbf{B}$ penalization does not result in any improvement but also no deterioration. It can be seen from the calculation above in Eq. (4.7) that only the parameter α_R is scaled differently.

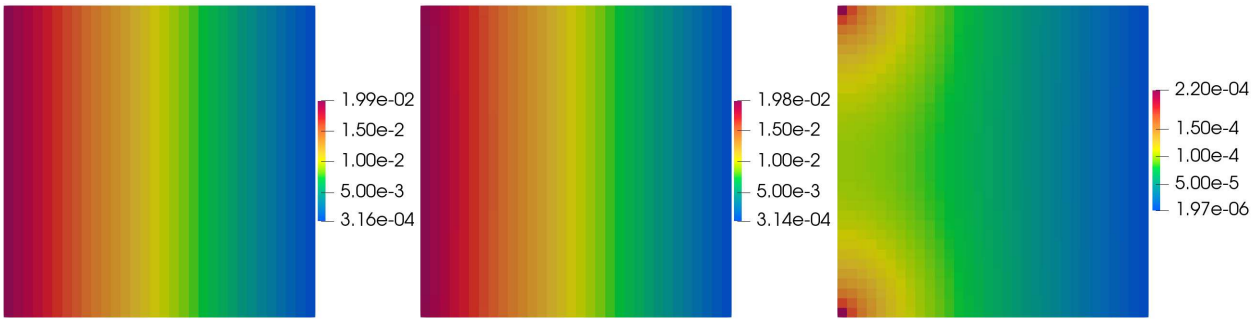


Figure 4.11: Analytical pressure solution (left) compared with the diffusive updated pressure approximation (center) and the absolute error (right) for the first outer iteration for q_1.

Figure 4.11 shows that the reference solution for the pressure is visually achieved after just one outer iteration. The largest error in the pressure approximation is located in the corners, which can be reduced in further outer iterations.

The sequential-in-time PP-solver has a similar error behavior. The largest pressure defect is also located in the corners. Apart from the divergence-defect, the behavior in the first outer iteration of the time-simultaneous solver is similar (even slightly better in some cases).

4.2 Effect of the velocity approximation

4.2.1 Influence of the velocity accuracy on the pressure

The accurate computation of the Burgers equations Eq. (3.5) also has a major impact on the convergence rate of the entire solver.

The common defects and errors are listed in table 4.8, whereby the momentum equations was computed with a relative tolerance of only 10^{-2}. Compared to table 4.5 to table 4.7, the additional (outer) iterations required to achieve the specified tolerance have been highlighted (red, in brackets). It is noticeable that, despite the poorer velocity approximation, there is no visible influence on the velocity errors and the divergence-defect, but only on the pressure error. On average, 10 (outer) iterations more are required if the velocity problem is only roughly approximated and not computed up to machine precision.

	Residual											
	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}
$q_1(1)$	1 (0)	1 (0)	1 (0)	3 (0)	5 (0)	13 (0)	20 (0)	26 (0)	33 (0)	39 (0)	45 (0)	51 (0)
$q_2(1)$	1 (0)	1 (0)	1 (0)	3 (0)	6 (0)	13 (0)	20 (0)	27 (0)	33 (0)	39 (0)	45 (0)	51 (0)
$q_3(1)$	1 (0)	1 (0)	1 (0)	2 (0)	4 (0)	11 (0)	19 (0)	26 (0)	33 (0)	39 (0)	45 (0)	51 (0)
	Divergence-defect											
	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}
$q_1(1)$	2 (0)	4 (0)	10 (0)	16 (0)	23 (0)	29 (0)	35 (0)	41 (0)	47 (0)	53 (0)	59 (0)	65 (+1)
$q_2(1)$	2 (0)	4 (0)	10 (0)	16 (0)	23 (0)	29 (0)	36 (0)	42 (0)	47 (0)	53 (0)	59 (0)	65 (+1)
$q_3(1)$	2 (0)	3 (0)	9 (0)	16 (0)	22 (0)	29 (0)	36 (+1)	42 (0)	48 (0)	53 (0)	59 (0)	65 (+1)
	$\|p - p_h\|_{L^2}$											
	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}
$q_1(1)$	7 (+5)	14 (+10)	20 (+10)	27 (+11)	33 (+10)	39 (+10)	45 (+10)	51 (+10)	57 (+10)	63 (+10)	70 (+11)	77 (+13)
$q_2(1)$	7 (+5)	14 (+10)	20 (+10)	27 (+11)	33 (+10)	39 (+10)	45 (+10)	51 (+10)	57 (+10)	63 (+10)	70 (+11)	82 (+18)
$q_3(1)$	6 (+4)	13 (+9)	20 (+11)	26 (+10)	33 (+10)	39 (+10)	45 (+9)	51 (+9)	57 (+9)	63 (+10)	70 (+11)	77 (+13)
	$\|u - u_h\|_{L^2}$											
	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}
$q_1(1)$	3 (0)	5 (0)	10 (0)	16 (0)	22 (0)	28 (0)	34 (0)	40 (0)	46 (0)	52 (0)	58 (+1)	64 (+1)
$q_2(1)$	3 (0)	5 (0)	10 (0)	16 (0)	22 (0)	28 (0)	34 (0)	40 (0)	46 (0)	52 (0)	58 (0)	64 (+1)
$q_3(1)$	3 (0)	5 (0)	9 (0)	15 (0)	22 (0)	28 (0)	34 (0)	41 (0)	46 (0)	52 (0)	58 (0)	64 (+1)
	$\|u - u_h\|_{H^1}$											
	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}
$q_1(1)$	8 (0)	14 (0)	21 (0)	27 (0)	34 (0)	40 (0)	46 (0)	51 (0)	57 (0)	63 (0)	69 (+1)	77 (+3)
$q_2(1)$	8 (0)	14 (0)	21 (0)	27 (0)	34 (0)	40 (0)	46 (0)	51 (0)	57 (0)	63 (0)	69 (+1)	77 (+3)
$q_3(1)$	7 (0)	13 (0)	20 (0)	27 (0)	34 (0)	40 (0)	46 (0)	52 (0)	57 (0)	63 (0)	70 (+2)	77 (+3)

Table 4.8: Number of outer iterations for the residual, divergence-defect, pressure L^2 error, the velocity L^2 error and the H^1 error with a corresponding increase (red, in brackets) for a roughly approximated Burgers equations using 10^{-2} as relative tolerance at the endpoint $T = 1$ for $q_i(t)$, $i = 1, 2, 3$.

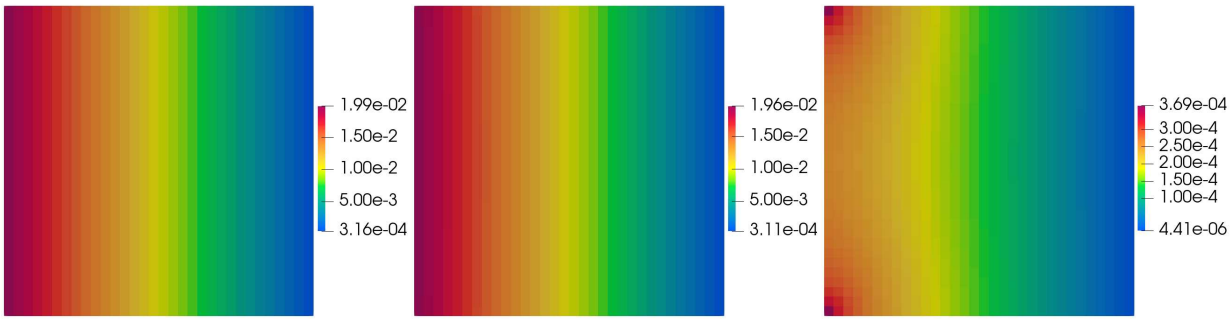


Figure 4.12: Analytical pressure solution (left) compared with the pressure approximation using $\alpha_R = 1$, $\alpha_D = \theta\nu k$ and a relative tolerance of 10^{-2} for the velocity solver (center) and the corresponding absolute pressure error (right) for the first outer iteration for q_1.

Figure 4.12 shows that the largest error is still found in the corners, but compared to figure 4.11, it seems to be more evenly spread from the corners.

For better illustration, the absolute error between the two pressure approximations is compared in figure 4.13, which also shows the more evenly spread error from the corners.

The possibility of a roughly velocity approximation opens up the possibility of successfully using the time-simultaneous approach of [26] or the parareal approaches in the future to accelerate the velocity computation process.

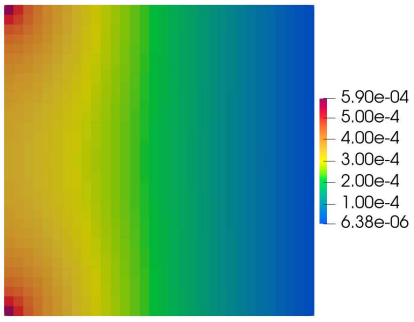


Figure 4.13: Absolute error between the pressure approximation computed with relative velocity tolerance of 10^{-2}, and the pressure approximation computed with relative velocity tolerance of 10^{-16}.

Similar results are also generated for the functions q_2 and q_3.

4.2.2 Influence of the time step size

In case of the PP-solver, as discussed in the introduction in section 1, a higher accuracy is achieved for a smaller time step size k. For the time decoupled solver algorithm 3.3, no convergence boost can be observed. The same number of outer iterations are required regardless of the time step size k. However, the computing time for the momentum solver is noticeable faster with a smaller time step size.

The phenomenon of the same (outer) iteration numbers regardless of the time step size k is examined in more detail in the next section 4.3.

4.3 Divergence-free velocity

One strategy to obtain faster convergence is to compute a divergence-free velocity. In contrast to the diffusive update in the pressure space in section 4.1, the direct calculation of a divergence-free velocity requires more computational effort. An analysis of the computation time can be found at the end in section 4.3.5.

In this section, the augmented Lagrangian method from section 3.3.3 is analyzed. The following tests are performed using the Q_2/P_1^{disc} element. Tests with the Q_2/Q_1 element can be found in section 4.6.

4.3.1 Computing a divergence-free velocity using augmented Lagrangian

In order to have less dependencies in relation to the last time step, the backward Euler time stepping scheme is tested first. First tests are performed with the configuration $\alpha_R = 1$ and $\alpha_D = 0$ for the same test problem from Eq. (4.1). As before, the step size $k = 0.01$ and $K = 100$ blocked time steps are used. The momentum equations are solved by using the optimal penalty parameter $\gamma = \frac{1}{\nu k}$ as described in section 3.3.3 to directly compute a divergence-free velocity. This leads to an almost divergence-free right-hand-side $\mathbf{f}_p$ for the pressure Poisson problem.

Table 4.9 shows on the one hand that the divergence-free velocity is almost fulfilled after the first outer iteration, but on the other hand a high error in pressure approximation occurs. Also after several outer iterations, the pressure approximation does not improve.

it.	$\|\mathbf{Su} + k\mathbf{Bp} - \mathbf{g}\|$	$\|\mathbf{B}^\top \mathbf{u}\|$	$\|p - p_h\|_{L^2}$	$\|u - u_h\|_{L^2}$	$\|u - u_h\|_{H^1}$
1	6.906e-06	7.216e-10	7.373e-01	1.205e-07	5.278e-06
2	6.906e-06	7.215e-10	7.372e-01	1.487e-07	5.685e-06
3	6.905e-06	7.215e-10	7.372e-01	1.192e-07	5.391e-06
4	6.905e-06	7.215e-10	7.372e-01	1.272e-07	5.502e-06
5	6.905e-06	7.215e-10	7.371e-01	1.114e-07	5.209e-06
6	6.905e-06	7.214e-10	7.371e-01	1.209e-07	5.388e-06
7	6.904e-06	7.214e-10	7.371e-01	1.368e-07	5.391e-06
8	6.904e-06	7.214e-10	7.371e-01	1.191e-07	5.564e-06
9	6.904e-06	7.213e-10	7.370e-01	1.228e-07	5.267e-06
10	6.904e-06	7.213e-10	7.370e-01	1.139e-07	5.197e-06

Table 4.9: First 10 outer iterations with the optimal penalty parameter $\gamma = \frac{1}{\nu k} = 10000$ using $\theta = 1$ at endpoint $T = 1$ for the function q_1.

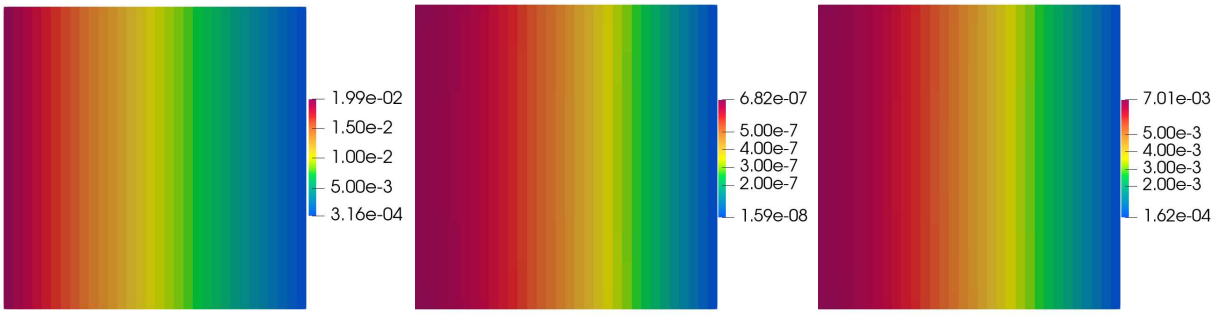


Figure 4.14: Analytical pressure solution (left) compared to the divergence-free velocity computed pressure approximation (center) and a divergence-free velocity derived pressure approximation scaled by γ (right) for the first outer iteration for the function q_1.

Figure 4.14 depicts the analytical pressure and the numerically computed pressure. A comparison with figure 4.11 shows that an inaccurate pressure approximation is computed for a directly derived divergence-free velocity.
As shown in figure 4.14, scaling the right-hand-side of the pressure Poisson problem $\mathbf{f}_p$ by γ provides a better order of magnitude but is still far from the reference solution. In each case, an inaccurate pressure approximation can be seen. The scaling parameter is therefore not trivial.

Computing a divergence-free velocity before solving the pressure Poisson problem seems to be a wrong approach!

4.3.2 Posterior divergence-free optimization using augmented Lagrangian method

According to the above conclusions, the right-hand-side of the pressure Poisson problem $\mathbf{f}_p$ must contain a non-divergence-free velocity. Similar to the PP-algorithm 3.1, a divergence-free update must be performed after solving the pressure Poisson problem. First, a non-divergence-free velocity need to be computed, using a fast Richardson multigrid in space solver with a damped Jacobi preconditioner. This non-divergence-free velocity vector must be stored for the right-hand-side of the pressure Poisson problem. In a next step a posterior divergence-free optimization using e.g. the augmented Lagrangian method can be performed. The divergence-free velocity must be saved, as it is needed for the time step before as described in algorithm 3.4 in section 3.3.3. Since the velocity of the last time step is hidden in the right-hand-side $\mathbf{g}$ of the momentum equations, using a divergence-free velocity for $\mathbf{g}$ leads to faster convergence and ensures convergence even in the case of the Stokes-Oseen or Navier-Stokes equations.

Test case: Reduced right-hand-side for the pressure Poisson problem

It could be assumed that due to the divergence-free optimization, the velocity from the last iteration $\mathbf{B}^\top \mathbf{u}^{l-1} \equiv 0$ is already divergence-free, so that only a shorter right-hand-side in the form of $\mathbf{f}_p = \frac{1}{k}\mathbf{B}^\top \mathbf{u}^l$ for the pressure Poisson problem is sufficient. This idea can lead to achieve an approximation as good as the PP approximation, but is generally wrong.

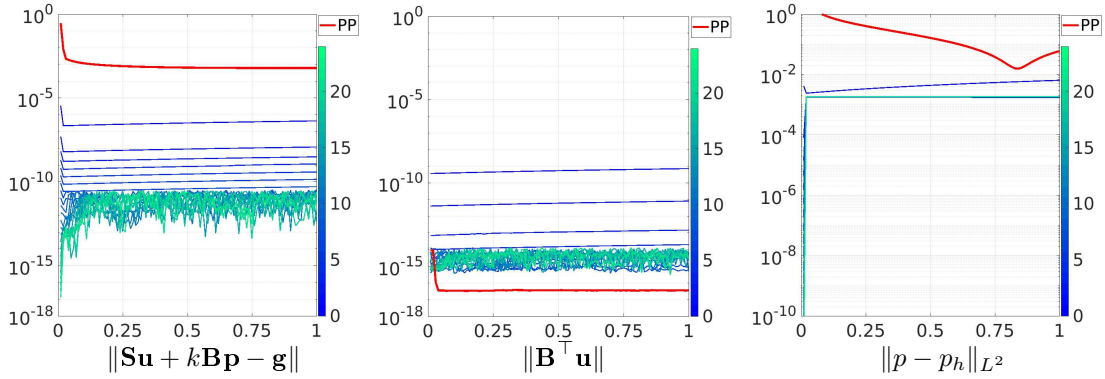


Figure 4.15: Results for a reduced right-hand-side $\mathbf{f}_p = \frac{1}{k}\mathbf{B}^\top \mathbf{u}^l$ using the backward Euler scheme with $\alpha_R = 1$, $\alpha_D = \nu k$ and $\gamma = \alpha_D^{-1}$. The total residual (left), divergence-defect (center) and L^2 pressure error (right) is plotted for the function $q_1(t)$ over the time interval [0,1].

Figure 4.15 shows that the accuracy of the pressure approximation is stagnant for the right-hand-side $\mathbf{f}_p = \frac{1}{k}\mathbf{B}^\top \mathbf{u}^l$ after the second iteration at an accuracy of 10^{-3}. An improvement can only be seen in the first time step, as it does not depend on the previous divergence-defect. The total residual improves after several outer iterations up to the order of 10^{-12} and the velocity is solved almost to machine precision. Therefore, it is necessary to assemble the full divergence-defect, even in the case of (almost) divergence-free velocity.

Test case: Adjusted right-hand-side for the pressure Poisson problem
The following figures show the augmented Lagrangian with $\gamma = \alpha_D^{-1}$, $\alpha_D = \theta\nu k$ and $\alpha_R = 1$ using the full update for the right-hand-side $\mathbf{f}_p$ according to Eq. (3.42) and Crank-Nicolson with $\theta = 0.5$ as the time stepping scheme. As before, the PP reference solution is provided in red.

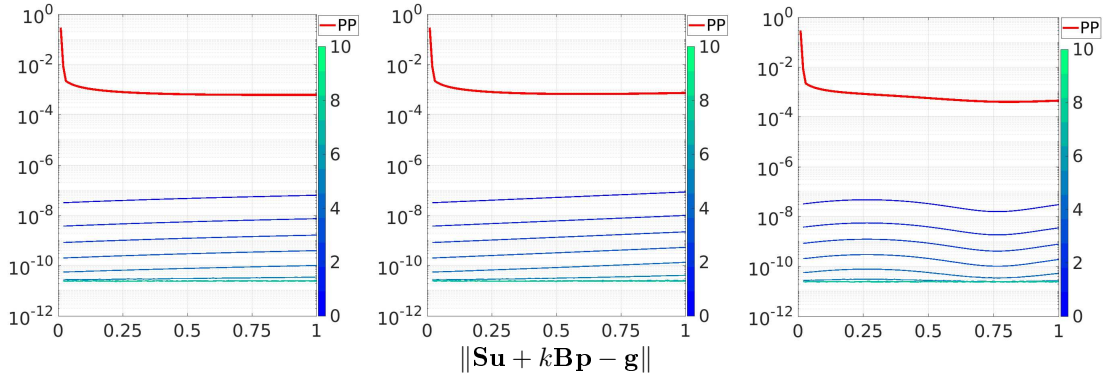


Figure 4.16: Residual for $\gamma = \alpha_D^{-1}$ with $\alpha_D = \theta\nu k$ for function q_1 (left), q_2 (center) and q_3 (right) over the time interval [0,1].

The first thing that stands out in figure 4.16 is the (almost) straight defect line. In the last section 4.1, it was pointed out that, without a divergence-free velocity optimization, a large number of outer iterations are required to break down the defect from the rear time steps. Especially with a large number of blocked time steps, the computation of a divergence-free velocity is necessary to obtain fast convergence due to less (outer) iterations.
Compared to the results from section 4.1.2, approx. 3 additional outer iterations can be saved by computing a divergence-free velocity in order to achieve the same accuracy for the residual in the first outer iteration and approx. 27 additional outer iterations can be saved in order to achieve an accuracy of order 10^{-11}. The accuracy of 10^{-11} is achieved in just 6 outer iterations. Further outer iterations do not provide any improvement.

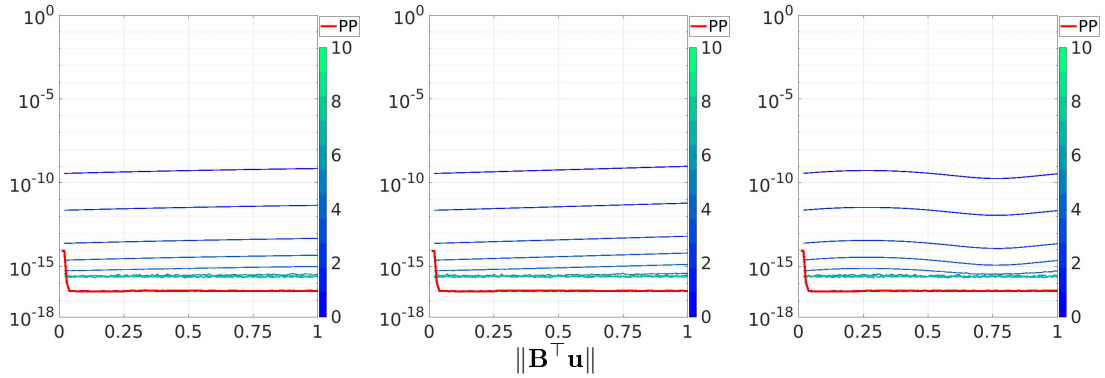


Figure 4.17: Divergence-defect for $\gamma = \alpha_D^{-1}$ with $\alpha_D = \theta\nu k$ for function q_1 (left), q_2 (center) and q_3 (right) over the time interval [0,1].

Figure 4.17 depicts that the accuracy of the PP-solver is still not achieved by the augmented Lagrangian. Compared to the results from the previous section 4.1.2, approx. 35 outer iterations can be saved for the first outer iteration with an optimal penalty parameter of $\gamma = \alpha_D^{-1}$. Furthermore, an accuracy of 10^{-14} is achieved with only 4 outer iterations. This results in a saving of approx. 60 outer iterations.

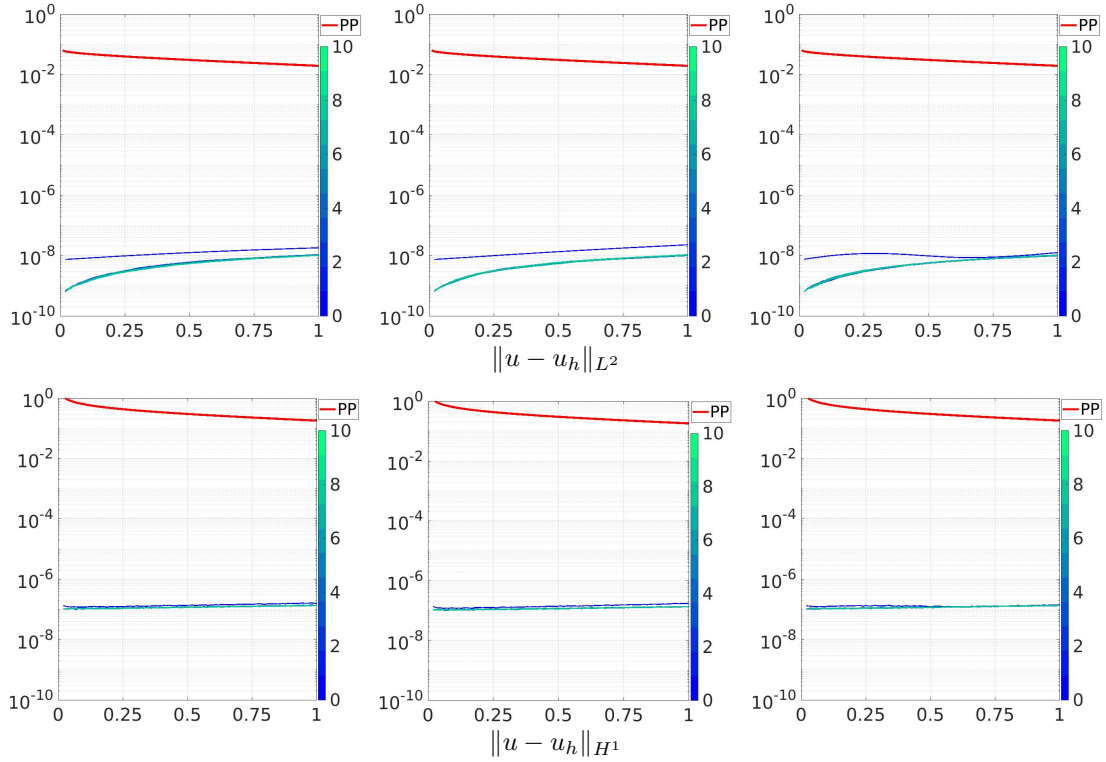


Figure 4.18: Velocity L^2 error (top) and H^1 error (bottom) for $\gamma = \alpha_D^{-1}$ with $\alpha_D = \theta\nu k$ for $q_1(t)$ (left), $q_2(t)$ (center) and $q_3(t)$ (right) over the time interval [0,1].

The L^2 and H^1 errors for the velocity are shown above in figure 4.18. The errors result in a poor approximation for the PP-solver, as the time step size $k = 0.01$ was not chosen small enough and therefore the preconditioners from section 3.2.1 are not exact. The cause is that the velocity of the PP-solver is artificially updated to be divergence-free and a divergence-free velocity is not computed directly. Using the time-simultaneous solver, the L^2 error no longer improves after the second outer iteration. The H^1 error has already stagnated after one iteration at an accuracy of approx. 10^{-7}.

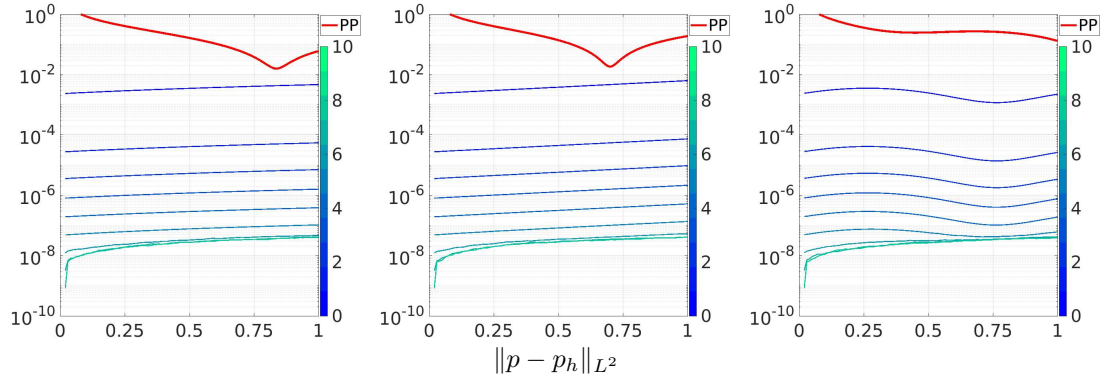
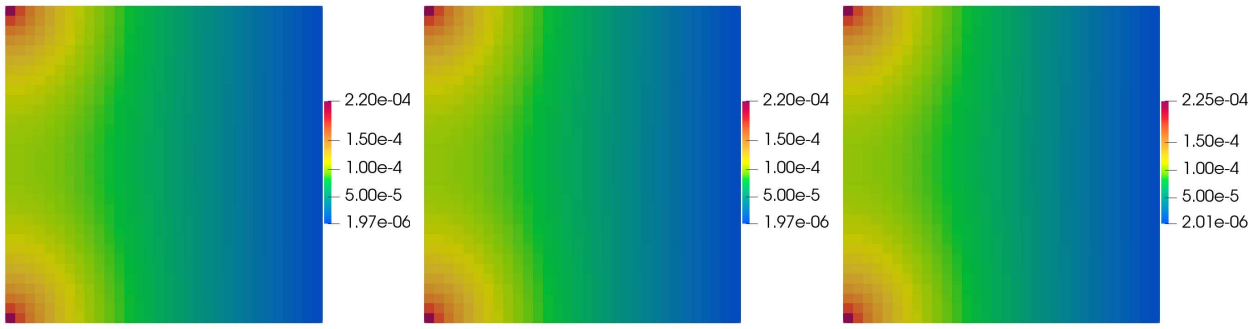


Figure 4.19: Pressure L^2 error for $\gamma = \alpha_D^{-1}$ with $\alpha_D = \theta\nu k$ for $q_1(t)$ (left), $q_2(t)$ (center) and $q_3(t)$ (right) over the time interval [0,1].

The first outer iteration of the pressure approximation looks almost the same as without the augmented Lagrangian approach, but there is a much faster convergence for the further outer iterations. After the 7th iteration at an accuracy of 10^{-8}, the L^2-pressure error stagnates.



$|p - p_h|$

Figure 4.20: Absolute pressure error for $\gamma = \alpha_D^{-1}$ with $\alpha_D = \theta\nu k$ after the first outer iteration for q_1 (left), q_2 (center) and q_3 (right).

Figure 4.20 analyzes the pressure error for the first outer iteration. The absolute pressure error emphasizes that there is no correction in the corners for the decoupled solver after the first outer iteration. The divergence-free time-simultaneous solver therefore has the same pressure behavior as the PP-solver. Similar to the PP-solver, the error in the corners also decreases for smaller time step sizes k.
Section 4.4 utilizes the multigrid in time method from section 3.3.4 to solve the issue of corners in the pressure vector during the first outer iteration by recoupling the system. But first, the influence of the parameter γ is investigated in detail in section 4.3.3 below.

4.3.3 Influence of the parameter γ

In order to find the ideal parameter γ for the time-simultaneous solver, the parameter is over-relaxed and under-relaxed based on the optimally calculated parameter according to section 3.3.3.

Table 4.10 and table 4.11 have the common defects listed for a range of γ values for the function q_1 evaluated at endpoint $T = 1$. Table 4.10 clearly shows that the residual of the entire system is determined most accurately for an under-relaxed parameter γ in the range from $10^{-4}\alpha_D^{-1}$ to $10^{-1}\alpha_D^{-1}$.

	ω										
	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{1}	10^{2}	10^{3}	10^{4}	10^{5}
	$\|\mathbf{Su} + k\mathbf{Bp} - \mathbf{g}\|$										
1	1.04e-07	5.52e-08	6.23e-08	6.31e-08	6.32e-08	6.32e-08	6.32e-08	2.21e-07	1.41e-06	1.91e-06	1.91e-06
2	1.13e-08	7.80e-09	7.43e-09	7.40e-09	7.39e-09	7.39e-09	7.40e-09	1.11e-08	3.38e-08	1.36e-07	1.39e-07
3	2.59e-09	1.74e-09	1.65e-09	1.64e-09	1.64e-09	1.64e-09	1.66e-09	3.68e-09	5.80e-09	1.13e-08	1.15e-08
4	6.27e-10	4.19e-10	3.98e-10	3.96e-10	3.96e-10	3.97e-10	4.66e-10	4.48e-09	4.20e-09	4.38e-09	2.60e-09
5	1.57e-10	1.04e-10	9.93e-11	9.88e-11	9.88e-11	1.02e-10	2.69e-10	4.67e-09	4.53e-09	1.85e-09	3.23e-10
6	4.00e-11	2.65e-11	2.52e-11	2.51e-11	2.52e-11	3.51e-11	2.48e-10	4.79e-09	5.09e-09	2.08e-09	2.20e-09
7	1.03e-11	6.84e-12	6.50e-12	6.47e-12	6.99e-12	2.47e-11	2.46e-10	3.97e-09	4.12e-09	2.95e-09	2.53e-09
8	2.68e-12	1.77e-12	1.69e-12	1.69e-12	3.00e-12	2.41e-11	2.40e-10	4.63e-09	4.51e-09	2.87e-09	1.38e-09
9	7.03e-13	4.63e-13	4.61e-13	5.05e-13	2.56e-12	2.43e-11	2.47e-10	4.45e-09	4.75e-09	2.85e-09	1.21e-09

Table 4.10: Residual for the first 9 outer iterations for a range of over- and under-relaxed $\gamma = \omega\alpha_D^{-1}$ values at the endpoint $T = 1$ for function q_1. Ideal parameters are highlighted (in green).

	ω 10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{1}	10^{2}	10^{3}	10^{4}	10^{5}
	$\|\mathbf{B}^\top \mathbf{u}\|$										
1	6.98e-05	7.18e-06	7.20e-07	7.20e-08	7.20e-09	7.20e-10	7.20e-11	7.19e-12	7.39e-13	7.37e-14	8.87e-15
2	1.05e-06	5.19e-08	4.64e-09	4.58e-10	4.58e-11	4.58e-12	4.57e-13	4.41e-14	1.50e-14	5.87e-15	2.09e-15
3	1.82e-08	5.75e-10	4.94e-11	4.87e-12	4.86e-13	4.86e-14	4.36e-15	1.19e-15	3.15e-16	1.08e-15	1.68e-15
4	8.79e-10	5.14e-11	4.85e-12	4.81e-13	4.81e-14	4.81e-15	9.68e-16	1.37e-15	5.71e-16	8.94e-16	1.66e-15
5	1.64e-10	1.06e-11	1.00e-12	9.95e-14	9.93e-15	1.03e-15	6.89e-16	1.42e-15	5.68e-16	7.14e-16	1.56e-15
6	3.75e-11	2.45e-12	2.32e-13	2.31e-14	2.38e-15	3.50e-16	6.81e-16	1.57e-15	7.51e-16	9.22e-16	1.73e-15
7	9.11e-12	5.94e-13	5.64e-14	5.70e-15	8.06e-16	2.76e-16	7.27e-16	1.46e-15	7.15e-16	8.50e-16	1.58e-15
8	2.26e-12	1.48e-13	1.40e-14	1.74e-15	7.12e-16	2.81e-16	6.52e-16	1.41e-15	6.63e-16	8.84e-16	1.75e-15
9	5.75e-13	3.74e-14	3.90e-15	1.21e-15	6.13e-16	2.77e-16	7.93e-16	1.52e-15	6.64e-16	6.70e-16	1.42e-15
	$\|u - u_h\|_{L^2}$										
1	1.48e-03	1.52e-04	1.52e-05	1.52e-06	1.52e-07	1.82e-08	9.91e-08	5.76e-06	2.59e-04	2.23e-03	2.28e-03
2	2.23e-05	1.09e-06	9.73e-08	9.60e-09	1.10e-09	1.03e-08	9.75e-08	1.76e-06	1.71e-05	1.47e-04	1.49e-04
3	3.60e-07	9.35e-09	7.51e-10	1.29e-10	6.82e-10	1.07e-08	9.80e-08	1.74e-06	1.31e-05	1.20e-05	1.07e-05
4	6.78e-09	1.54e-10	2.14e-11	1.17e-10	6.77e-10	1.08e-08	9.78e-08	1.70e-06	1.26e-05	5.69e-06	3.39e-06
5	2.88e-10	1.35e-11	1.73e-11	1.17e-10	6.11e-10	1.04e-08	9.80e-08	1.66e-06	1.32e-05	6.13e-06	3.32e-06
6	4.17e-11	2.68e-12	1.80e-11	1.15e-10	7.03e-10	1.05e-08	9.24e-08	1.68e-06	1.34e-05	6.29e-06	2.98e-06
7	8.90e-12	9.30e-13	1.88e-11	1.18e-10	6.76e-10	1.05e-08	1.01e-07	1.60e-06	1.31e-05	5.63e-06	2.61e-06
8	2.06e-12	7.24e-13	1.85e-11	1.16e-10	6.74e-10	1.02e-08	9.66e-08	1.65e-06	1.31e-05	5.57e-06	2.23e-06
9	4.95e-13	7.18e-13	2.23e-11	1.19e-10	7.11e-10	1.02e-08	9.42e-08	1.61e-06	1.30e-05	5.56e-06	1.91e-06
	$\|u - u_h\|_{H^1}$										
1	6.10e-03	6.25e-04	6.27e-05	6.27e-06	6.30e-07	1.63e-07	1.50e-06	5.63e-04	9.19e-03	3.87e-02	3.95e-02
2	9.08e-05	4.51e-06	4.04e-07	4.05e-08	1.46e-08	1.35e-07	1.43e-06	2.85e-05	4.07e-04	2.57e-03	2.60e-03
3	1.48e-06	4.34e-08	3.74e-09	1.80e-09	1.29e-08	1.36e-07	1.48e-06	2.12e-05	1.14e-04	1.93e-04	1.88e-04
4	5.02e-08	2.68e-09	4.81e-10	1.64e-09	1.26e-08	1.37e-07	1.45e-06	2.19e-05	1.04e-04	5.76e-05	3.33e-05
5	8.37e-09	5.36e-10	3.73e-10	1.63e-09	1.20e-08	1.37e-07	1.48e-06	2.19e-05	1.06e-04	5.76e-05	3.16e-05
6	1.90e-09	1.24e-10	4.20e-10	1.60e-09	1.31e-08	1.38e-07	1.43e-06	2.23e-05	1.08e-04	5.92e-05	2.88e-05
7	4.58e-10	3.38e-11	4.20e-10	1.64e-09	1.32e-08	1.34e-07	1.47e-06	2.19e-05	1.05e-04	5.46e-05	2.52e-05
8	1.13e-10	1.80e-11	4.22e-10	1.63e-09	1.25e-08	1.34e-07	1.44e-06	2.24e-05	1.05e-04	5.30e-05	2.16e-05
9	2.88e-11	1.66e-11	4.73e-10	1.67e-09	1.25e-08	1.34e-07	1.46e-06	2.11e-05	1.05e-04	5.27e-05	1.85e-05
	$\|p - p_h\|_{L^2}$										
1	1.11e-02	5.34e-03	4.75e-03	4.70e-03	4.69e-03	4.69e-03	4.69e-03	4.69e-03	1.63e-02	5.23e-02	5.29e-02
2	1.93e-04	6.39e-05	5.57e-05	5.49e-05	5.48e-05	5.48e-05	5.55e-05	6.86e-05	8.01e-04	3.61e-03	3.65e-03
3	1.22e-05	7.55e-06	7.15e-06	7.10e-06	7.10e-06	7.10e-06	7.26e-06	2.13e-05	1.08e-04	2.76e-04	2.79e-04
4	2.59e-06	1.69e-06	1.60e-06	1.59e-06	1.59e-06	1.59e-06	1.93e-06	1.90e-05	6.21e-05	2.70e-05	2.29e-05
5	6.21e-07	4.07e-07	3.86e-07	3.84e-07	3.84e-07	3.87e-07	1.14e-06	1.84e-05	5.65e-05	1.84e-05	1.03e-05
6	1.55e-07	1.01e-07	9.61e-08	9.56e-08	9.60e-08	1.04e-07	1.05e-06	1.87e-05	5.91e-05	1.88e-05	9.38e-06
7	3.93e-08	2.57e-08	2.44e-08	2.43e-08	2.57e-08	4.72e-08	1.06e-06	1.89e-05	5.56e-05	1.81e-05	8.09e-06
8	1.01e-08	6.56e-09	6.25e-09	6.38e-09	1.03e-08	4.17e-08	1.05e-06	1.88e-05	5.49e-05	1.71e-05	6.97e-06
9	2.62e-09	1.70e-09	1.64e-09	2.20e-09	8.54e-09	4.00e-08	1.04e-06	1.87e-05	5.57e-05	1.69e-05	5.95e-06

Table 4.11: Various defects and errors for the first 9 outer iterations for a range of over- and under-relaxed $\gamma = \omega\alpha_D^{-1}$ values at the endpoint $T = 1$ for function q_1. Ideal parameters are highlighted (in green).

Table 4.11 confirms the conclusion of the residual from table 4.10. An under-relaxation of γ ensures more accurate results for the L^2 pressure error. In the case of velocity errors, the range for an ideal parameter γ can be defined more precisely (highlighted in green). The most effective error reduction occurs at $10^{-4}\alpha_D^{-1}$ after a few outer iterations and at 10^{-1} for the first outer iterations. The analytically optimally calculated parameter γ with $\omega = 1$ ensures the smallest errors in the first outer iteration, but does not improve with further outer iterations.
A divergence-free velocity is achieved more quickly with a larger parameter γ. It can also be seen that a large choice of $10^{5}\alpha_D^{-1}$ leads to comparatively less accurate results with further outer iterations. To get a better overall view, the defects and errors are plotted in figure 4.21.
For a clear view, only the first and the last outer iterations (1, 7, 8 and 9) are displayed.

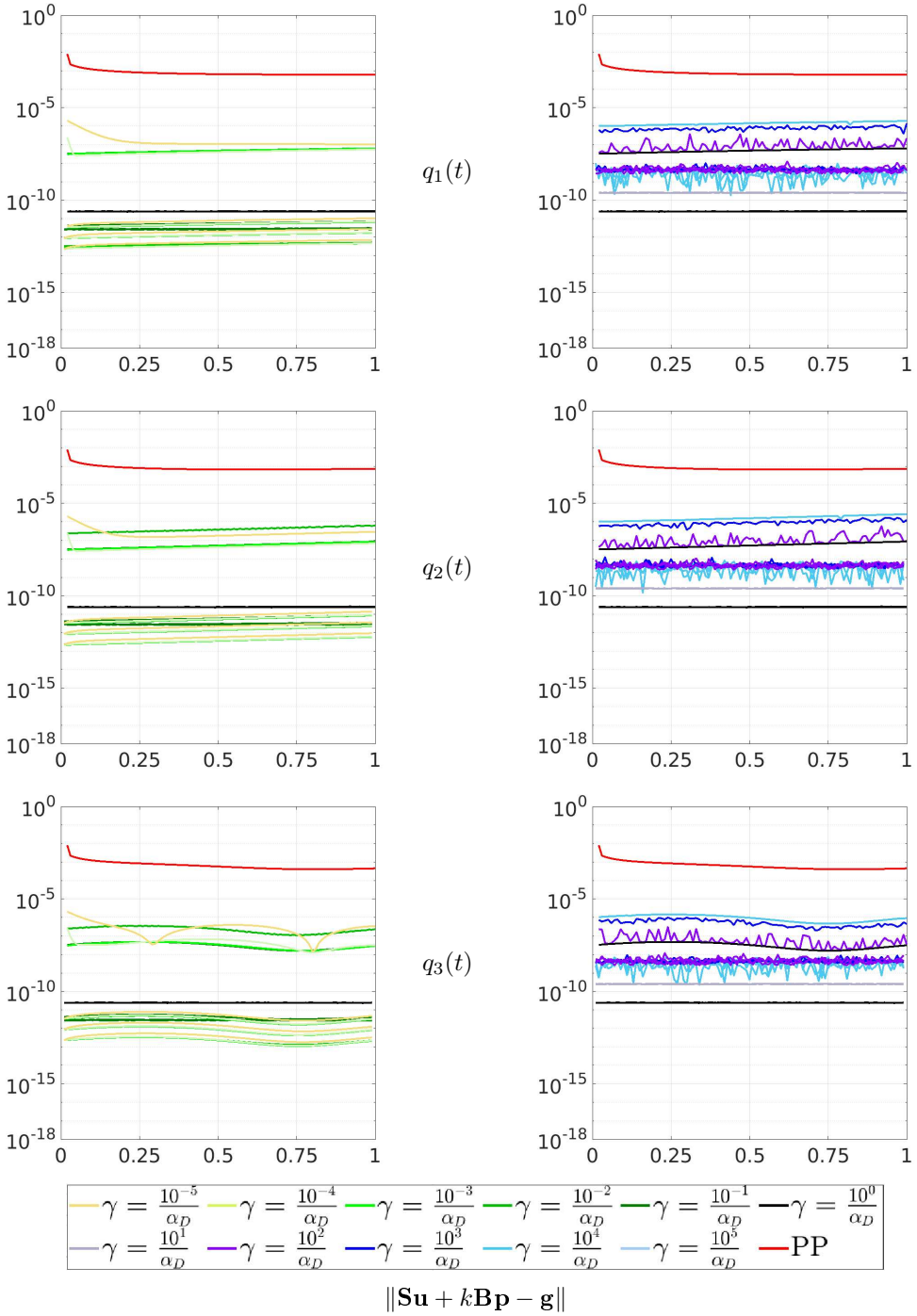


Figure 4.21: Residuals in the case of under-relaxation (left) and over-relaxation (right) for a range of γ values for function $q_1(t)$ (top), $q_2(t)$ (center) and $q_3(t)$ (bottom) over the time interval [0,1].

Figure 4.21 illustrates a problem with the over-relaxation of γ. Oscillations occur, as a high γ value generates a strongly ill-conditioned problem, which is difficult to solve. This problem will be discussed in more detail in the next section 4.3.5. Furthermore, applying a too strong under-relaxation by setting $\gamma = 10^{-5}\alpha_D^{-1}$ can cause that the (almost) straight defect lines disappear.

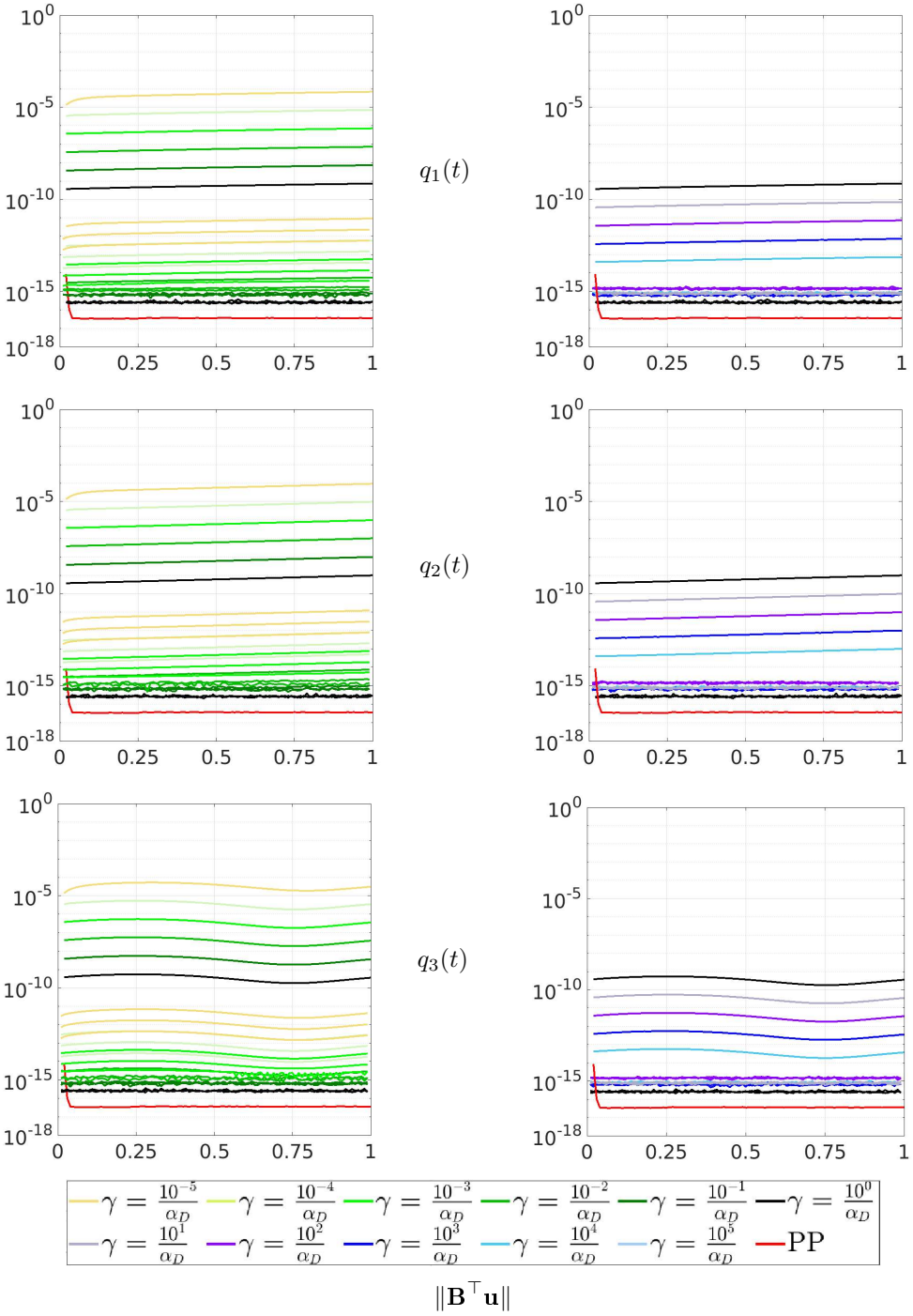


Figure 4.22: Divergence-defect in the case of under-relaxation (left) and over-relaxation (right) for a range of γ values for function $q_1(t)$ (top), $q_2(t)$ (center) and $q_3(t)$ (bottom) over the time interval [0,1].

It is evident that the choice of a high parameter leads to a smaller divergence-defect. The issue related to the pressure error in the corners mentioned in figure 4.20 remains regardless of the divergence-defect in the first outer iteration, and therefore, independent of the parameter choice of γ.

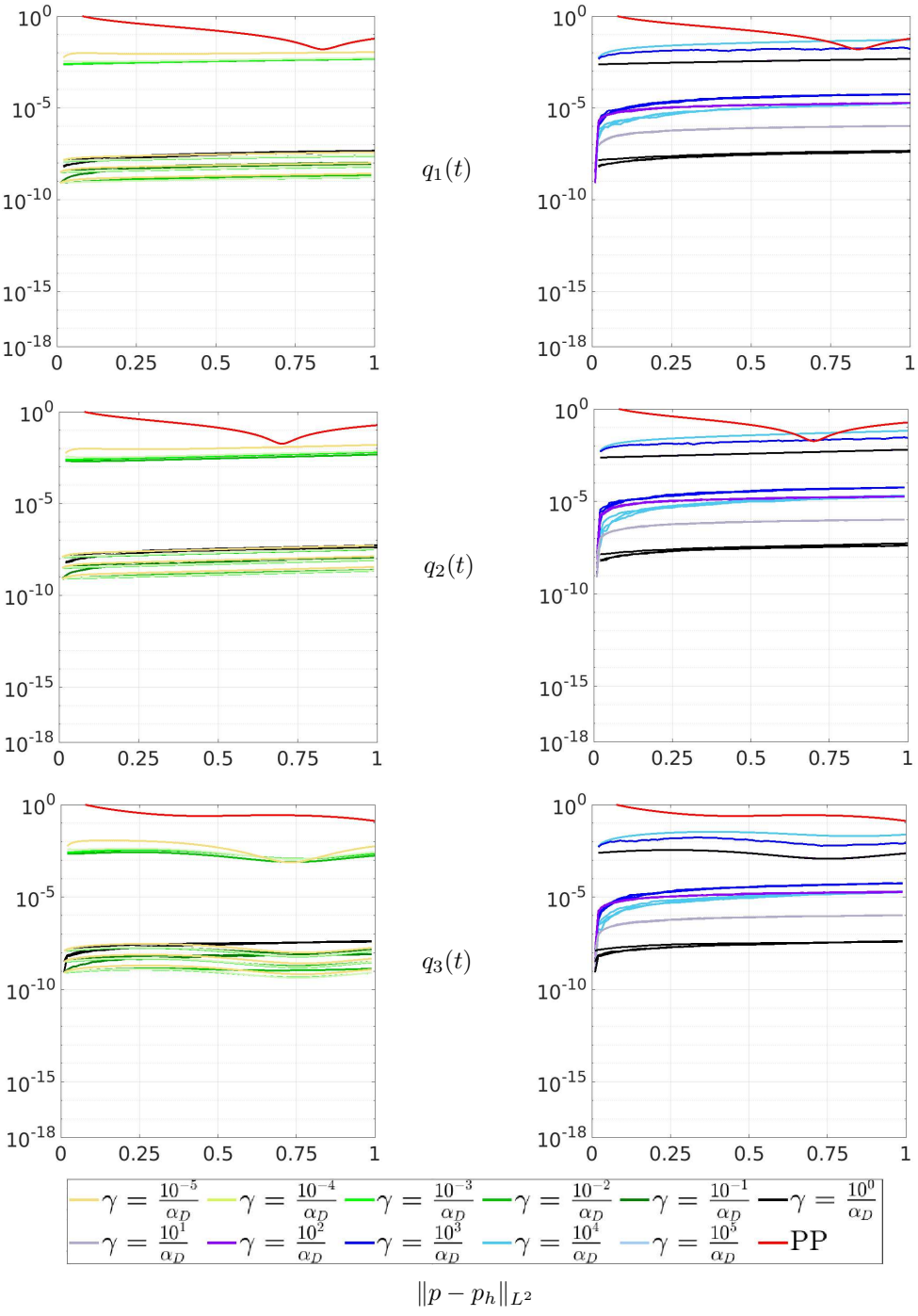


Figure 4.23: Pressure L^2 error in the case of under-relaxation (left) and over-relaxation (right) for a range of γ values for function $q_1(t)$ (top), $q_2(t)$ (center) and $q_3(t)$ (bottom) over the time interval [0,1].

Solving a less ill-conditioned problem with an under-relaxed γ value and then embedding the solver in a multigrid in time from section 3.3.4 is superior, as it is itself defined as a divergence-defect correction and thus further enforces a divergence-free velocity. Additionally, figure 4.23 demonstrates that selecting a large value of γ even results in decreased pressure approximation accuracy.

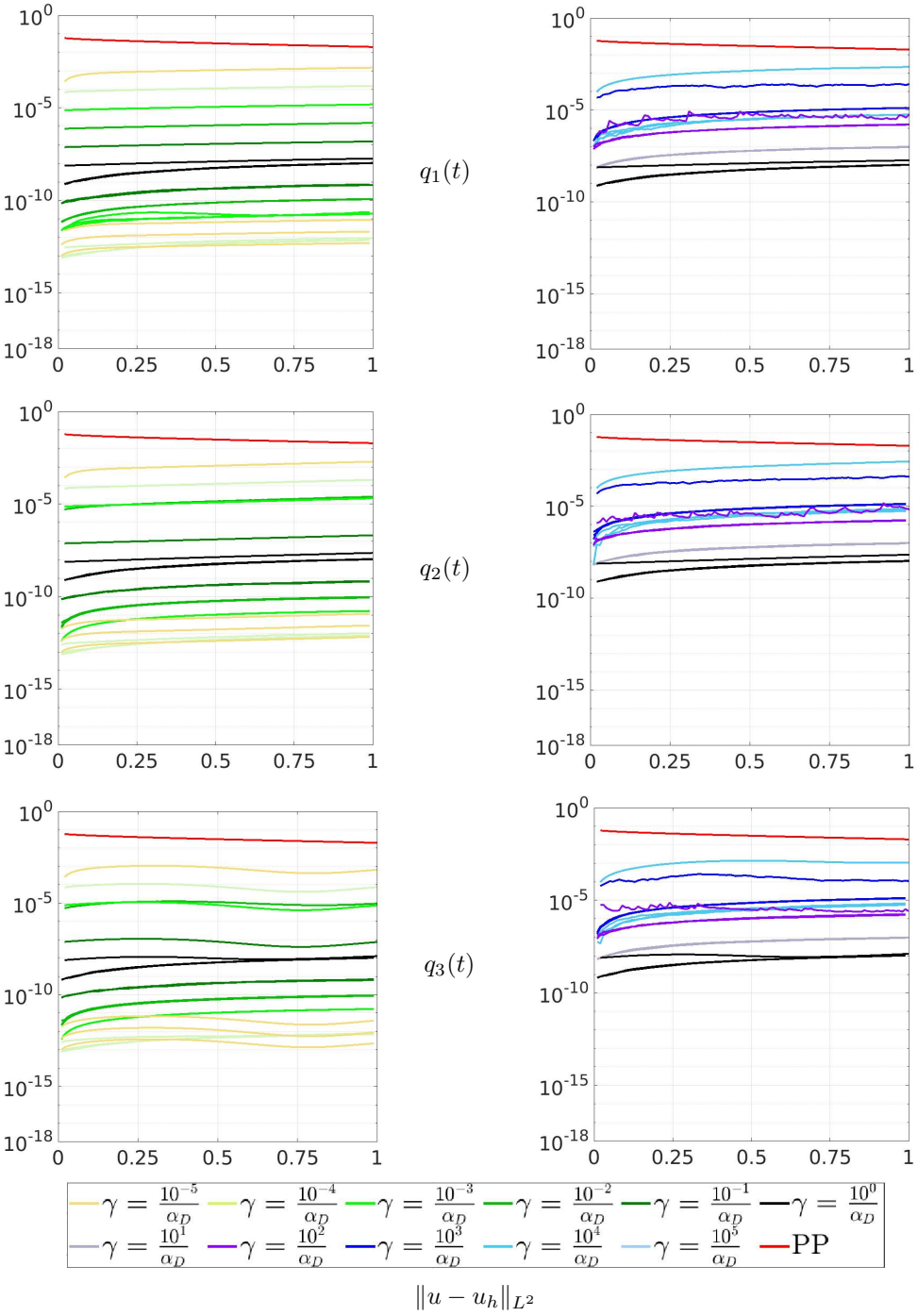


Figure 4.24: Velocity L^2 error in the case of under-relaxation (left) and over-relaxation (right) for a range of γ values for function $q_1(t)$ (top), $q_2(t)$ (center) and $q_3(t)$ (bottom) over the time interval [0,1].

For completeness, the L^2 error for the velocity is shown in figure 4.24. A similar curve is produced by the H^1 error. The importance to obtain an almost straight error curve for the velocity is particularly important in the case of the Stokes Oseen or the Navier-Stokes equations, especially with a large number of blocked time steps K.

4.3.4 An *ideal* parameter γ

In this section, the most suitable stabilization parameter γ is determined. In the following, a range of ν and k values are examined in order to find the *ideal* parameter γ. *Ideal* in the case of the smallest velocity error, since the pressure has a large dependence on the accuracy of the velocity.

		$\|\mathbf{Su} + k\mathbf{Bp} - \mathbf{g}\|$							
		ω							
		10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{1}	10^{2}
$\nu = 1.0$	1	3.24e-04	1.84e-04	6.04e-05	8.58e-05	1.06e-04	1.31e-04	4.45e-04	1.04e-02
	2	1.25e-04	6.97e-05	2.35e-05	4.59e-05	4.52e-05	4.60e-05	7.74e-05	3.22e-04
	3	8.51e-05	4.86e-05	2.96e-05	3.37e-05	3.07e-05	3.05e-05	3.46e-05	7.38e-05
	4	6.70e-05	3.63e-05	3.01e-05	2.52e-05	2.25e-05	2.22e-05	2.34e-05	8.40e-05
	5	5.52e-05	2.75e-05	2.69e-05	1.89e-05	1.66e-05	1.63e-05	1.64e-05	3.93e-05
	6	4.63e-05	2.08e-05	2.24e-05	1.40e-05	1.22e-05	1.20e-05	1.20e-05	2.57e-05
	7	3.98e-05	1.60e-05	1.80e-05	1.04e-05	8.99e-06	8.83e-06	8.76e-06	1.69e-05
	8	3.47e-05	1.23e-05	1.41e-05	7.60e-06	6.55e-06	6.43e-06	6.36e-06	1.14e-05
	9	3.06e-05	9.54e-06	1.08e-05	5.54e-06	4.75e-06	4.66e-06	4.60e-06	7.74e-06
		ω							
		10^{-6}	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{1}
$\nu = 0.1$	1	1.15e-05	1.15e-05	1.27e-06	2.95e-06	3.18e-06	4.26e-06	6.61e-05	1.93e-04
	2	3.74e-06	3.74e-06	1.41e-06	1.01e-06	9.47e-07	9.57e-07	1.39e-06	2.90e-06
	3	1.90e-06	1.90e-06	8.62e-07	5.36e-07	4.99e-07	4.97e-07	7.93e-07	8.42e-07
	4	1.34e-06	1.34e-06	5.02e-07	3.01e-07	2.80e-07	2.78e-07	3.59e-07	4.11e-07
	5	9.61e-07	9.61e-07	2.91e-07	1.71e-07	1.59e-07	1.58e-07	1.86e-07	2.10e-07
	6	6.55e-07	6.55e-07	1.68e-07	9.79e-08	9.09e-08	9.02e-08	1.01e-07	1.11e-07
	7	4.33e-07	4.33e-07	9.79e-08	5.63e-08	5.23e-08	5.18e-08	5.64e-08	6.14e-08
	8	2.78e-07	2.78e-07	5.66e-08	3.22e-08	2.99e-08	2.96e-08	3.17e-08	3.43e-08
	9	1.76e-07	1.76e-07	3.27e-08	1.85e-08	1.71e-08	1.69e-08	1.79e-08	1.92e-08
		ω							
		10^{-7}	10^{-6}	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	10^{0}
$\nu = 0.01$	1	4.35e-06	1.34e-06	1.04e-07	5.52e-08	6.23e-08	6.31e-08	6.32e-08	6.32e-08
	2	7.48e-07	1.06e-07	1.13e-08	7.80e-09	7.43e-09	7.40e-09	7.39e-09	7.39e-09
	3	1.19e-07	1.33e-08	2.59e-09	1.74e-09	1.65e-09	1.64e-09	1.64e-09	1.64e-09
	4	1.93e-08	2.95e-09	6.27e-10	4.19e-10	3.98e-10	3.96e-10	3.96e-10	3.97e-10
	5	4.45e-09	7.71e-10	1.57e-10	1.04e-10	9.93e-11	9.88e-11	9.88e-11	1.02e-10
	6	1.54e-09	2.06e-10	4.00e-11	2.65e-11	2.52e-11	2.51e-11	2.52e-11	3.51e-11
	7	5.99e-10	5.58e-11	1.03e-11	6.84e-12	6.50e-12	6.47e-12	6.99e-12	2.47e-11
	8	2.29e-10	1.51e-11	2.68e-12	1.77e-12	1.69e-12	1.69e-12	3.00e-12	2.41e-11
	9	8.69e-11	4.11e-12	7.03e-13	4.63e-13	4.61e-13	5.05e-13	2.56e-12	2.43e-11
$\nu = 0.001$	1	1.42e-07	1.24e-08	1.01e-09	6.88e-10	2.09e-08	7.59e-10	7.60e-10	7.95e-10
	2	2.73e-09	2.84e-11	1.47e-11	1.40e-11	2.23e-11	1.42e-11	2.90e-11	2.39e-10
	3	6.21e-11	7.89e-13	5.35e-13	5.11e-13	1.17e-12	2.80e-12	2.51e-11	2.41e-10
	4	1.57e-12	3.23e-14	2.18e-14	2.81e-14	1.32e-13	2.90e-12	2.55e-11	2.46e-10
	5	4.04e-14	1.83e-15	2.53e-15	1.94e-14	5.64e-14	2.90e-12	2.56e-11	2.49e-10
	6	1.32e-15	3.73e-16	2.25e-15	1.92e-14	5.46e-14	2.90e-12	2.57e-11	2.42e-10
	7	1.64e-16	9.49e-17	2.20e-15	1.90e-14	5.48e-14	2.90e-12	2.58e-11	2.36e-10
	8	5.17e-17	3.72e-17	2.14e-15	1.89e-14	5.62e-14	2.83e-12	2.47e-11	2.35e-10
	9	2.01e-17	1.47e-17	2.01e-15	1.94e-14	5.47e-14	2.84e-12	2.50e-11	2.40e-10
$\nu = 0.0001$	1	1.27e-09	1.23e-10	1.01e-11	3.92e-09	8.23e-12	2.66e-11	2.51e-10	2.72e-09
	2	4.92e-13	2.07e-14	2.10e-14	2.40e-12	2.80e-12	2.54e-11	2.40e-10	2.74e-09
	3	8.13e-16	2.23e-15	1.36e-14	1.29e-13	2.84e-12	2.60e-11	2.47e-10	2.57e-09
	4	1.04e-16	2.16e-15	1.34e-14	4.29e-14	2.77e-12	2.56e-11	2.44e-10	2.65e-09
	5	3.58e-17	2.18e-15	1.35e-14	4.31e-14	2.79e-12	2.48e-11	2.45e-10	2.81e-09
	6	1.26e-17	2.16e-15	1.34e-14	4.14e-14	2.82e-12	2.54e-11	2.51e-10	2.53e-09
	7	5.00e-18	2.09e-15	1.34e-14	4.10e-14	2.85e-12	2.58e-11	2.51e-10	2.55e-09
	8	2.79e-18	2.14e-15	1.33e-14	5.60e-14	2.77e-12	2.52e-11	2.53e-10	2.54e-09
	9	2.27e-18	2.03e-15	1.34e-14	4.01e-14	2.89e-12	2.53e-11	2.48e-10	2.84e-09

Table 4.12: Residual for the first 9 (outer) iterations for a range of over- and under-relaxed $\gamma = \omega\alpha_D^{-1}$ values at the endpoint $T = 1$ for function q_1. Ideal parameters are highlighted (in green).

		$\|\mathbf{B}^\top\mathbf{u}\|$							
		ω							
		10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{1}	10^{2}
$\nu = 1.0$	1	1.05e-02	4.35e-03	6.72e-04	7.12e-05	7.32e-06	8.84e-07	4.12e-07	7.45e-08
	2	6.69e-03	1.69e-03	1.17e-04	9.74e-06	8.70e-07	8.91e-08	1.50e-08	4.39e-09
	3	4.79e-03	7.13e-04	2.34e-05	1.60e-06	1.30e-07	9.57e-09	1.04e-09	3.75e-10
	4	3.50e-03	3.11e-04	6.09e-06	4.09e-07	3.55e-08	3.60e-09	8.66e-10	1.25e-10
	5	2.57e-03	1.42e-04	2.44e-06	1.83e-07	1.73e-08	1.71e-09	3.66e-10	6.27e-11
	6	1.89e-03	6.81e-05	1.33e-06	1.07e-07	1.03e-08	1.02e-09	2.20e-10	3.35e-11
	7	1.40e-03	3.55e-05	8.35e-07	6.81e-08	6.63e-09	6.53e-10	1.37e-10	1.92e-11
	8	1.04e-03	2.00e-05	5.48e-07	4.51e-08	4.40e-09	4.34e-10	9.02e-11	1.14e-11
	9	7.65e-04	1.22e-05	3.70e-07	3.06e-08	2.98e-09	2.94e-10	5.99e-11	7.09e-12
$\nu = 0.1$	1	3.72e-03	6.58e-04	7.10e-05	7.16e-06	7.24e-07	1.11e-07	1.42e-08	1.57e-09
	2	1.11e-03	5.59e-05	2.88e-06	2.56e-07	2.08e-08	1.06e-09	7.19e-11	5.79e-12
	3	3.49e-04	5.49e-06	1.58e-07	1.29e-08	9.73e-10	8.07e-11	7.06e-12	7.45e-13
	4	1.11e-04	7.68e-07	2.77e-08	2.47e-09	2.32e-10	3.27e-11	2.61e-12	2.67e-13
	5	3.62e-05	2.34e-07	1.16e-08	1.06e-09	1.04e-10	1.25e-11	1.19e-12	1.20e-13
	6	1.29e-05	1.09e-07	5.83e-09	5.36e-10	5.30e-11	6.08e-12	5.90e-13	5.97e-14
	7	5.43e-06	5.74e-08	3.11e-09	2.87e-10	2.84e-11	3.18e-12	3.11e-13	3.16e-14
	8	2.65e-06	3.13e-08	1.70e-09	1.57e-10	1.55e-11	1.72e-12	1.69e-13	1.71e-14
	9	1.44e-06	1.74e-08	9.42e-10	8.68e-11	8.60e-12	9.45e-13	9.27e-14	9.46e-15
$\nu = 0.01$	1	6.98e-05	7.18e-06	7.20e-07	7.20e-08	7.20e-09	7.20e-10	7.20e-11	7.19e-12
	2	1.05e-06	5.19e-08	4.64e-09	4.58e-10	4.58e-11	4.58e-12	4.57e-13	4.41e-14
	3	1.82e-08	5.75e-10	4.94e-11	4.87e-12	4.86e-13	4.86e-14	4.36e-15	1.19e-15
	4	8.79e-10	5.14e-11	4.85e-12	4.81e-13	4.81e-14	4.81e-15	9.68e-16	1.37e-15
	5	1.64e-10	1.06e-11	1.00e-12	9.95e-14	9.93e-15	1.03e-15	6.89e-16	1.42e-15
	6	3.75e-11	2.45e-12	2.32e-13	2.31e-14	2.38e-15	3.50e-16	6.81e-16	1.57e-15
	7	9.11e-12	5.94e-13	5.64e-14	5.70e-15	8.06e-16	2.76e-16	7.27e-16	1.46e-15
	8	2.26e-12	1.48e-13	1.40e-14	1.74e-15	7.12e-16	2.81e-16	6.52e-16	1.41e-15
	9	5.75e-13	3.74e-14	3.90e-15	1.21e-15	6.13e-16	2.77e-16	7.93e-16	1.52e-15
$\nu = 0.001$	1	7.15e-07	7.16e-08	7.23e-09	7.16e-10	7.16e-11	7.16e-12	7.17e-13	8.41e-14
	2	6.85e-10	5.49e-11	5.38e-13	5.37e-13	5.11e-14	5.48e-15	8.83e-16	8.48e-16
	3	9.54e-13	7.16e-14	6.27e-15	3.43e-15	2.34e-15	1.73e-16	3.83e-16	3.69e-16
	4	1.37e-14	3.44e-15	3.97e-16	3.00e-15	2.32e-15	9.11e-17	4.41e-16	4.59e-16
	5	1.80e-15	3.23e-15	1.29e-16	3.09e-15	2.44e-15	1.03e-16	3.91e-16	4.46e-16
	6	2.04e-15	2.33e-15	9.39e-17	3.07e-15	2.43e-15	2.76e-16	3.83e-16	4.76e-16
	7	2.16e-15	2.28e-15	6.73e-17	3.02e-15	2.28e-15	2.04e-16	4.66e-16	4.19e-16
	8	5.01e-16	2.21e-15	2.75e-16	3.10e-15	2.33e-15	1.51e-16	5.79e-16	4.96e-16
	9	2.29e-15	2.15e-15	3.77e-16	3.11e-15	2.30e-15	1.38e-16	4.41e-16	4.35e-16
$\nu = 0.0001$	1	7.16e-09	7.36e-10	7.16e-11	7.16e-12	7.17e-13	7.17e-14	8.31e-15	7.87e-16
	2	5.79e-13	1.63e-13	2.61e-15	5.35e-16	8.55e-16	8.38e-17	1.08e-15	6.39e-16
	3	3.92e-16	6.60e-16	2.97e-15	7.91e-17	8.86e-16	1.15e-16	9.88e-16	6.15e-16
	4	2.73e-16	8.16e-16	2.98e-15	8.59e-17	7.17e-16	1.09e-16	9.72e-16	5.60e-16
	5	1.17e-15	5.64e-16	2.92e-15	5.85e-17	7.73e-16	1.73e-16	9.68e-16	7.67e-16
	6	2.00e-16	7.41e-16	3.01e-15	1.28e-16	7.01e-16	7.77e-17	1.04e-15	7.13e-16
	7	2.24e-16	4.28e-16	3.07e-15	8.30e-17	8.15e-16	8.69e-17	1.19e-15	6.00e-16
	8	1.20e-15	5.83e-16	2.92e-15	1.31e-16	7.33e-16	1.25e-16	9.18e-16	6.90e-16
	9	1.22e-15	5.63e-16	3.01e-15	8.34e-17	8.86e-16	8.03e-17	9.43e-16	7.76e-16

Table 4.13: Divergence-defect for the first 9 (outer) iterations for a range of over- and under-relaxed $\gamma = \omega\alpha_D^{-1}$ values at the endpoint $T = 1$ for function q_1. Ideal parameters are highlighted (in green).

Table 4.12 shows that as ν decreases, ω must also be adjusted accordingly. Therefore, no linear ν dependency can generally be assumed for γ. Since in Eq. (4.1) the right side ν is dependent, a smaller ν results in a smaller change per time step. Consequently, a smaller ν in this case results in a smaller divergence-defect, as seen in table 4.13, and thus to a better and faster convergence, as shown in table 4.12. As expected, a high γ value also ensures a better divergence-free velocity.

		$\|u - u_h\|_{L^2}$							
		ω							
		10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{1}	10^{2}
$\nu = 1.0$	1	2.49e-01	2.14e-01	9.51e-02	1.48e-02	1.57e-03	2.35e-03	5.23e-03	6.92e-03
	2	1.88e-01	1.47e-01	3.73e-02	2.55e-03	2.13e-04	2.87e-04	4.78e-04	4.21e-04
	3	1.48e-01	1.05e-01	1.55e-02	4.88e-04	3.27e-05	3.84e-05	7.02e-05	7.56e-05
	4	1.18e-01	7.65e-02	6.60e-03	1.05e-04	6.03e-06	6.14e-06	1.93e-05	2.78e-05
	5	9.48e-02	5.58e-02	2.88e-03	2.73e-05	1.52e-06	1.78e-06	6.14e-06	1.23e-05
	6	7.60e-02	4.08e-02	1.28e-03	9.18e-06	5.62e-07	6.46e-07	2.76e-06	7.44e-06
	7	3.00e-02	5.91e-04	4.05e-06	2.77e-07	2.67e-07	1.24e-06	5.03e-06	
	8	4.88e-02	2.20e-02	2.80e-04	2.15e-06	1.58e-07	1.31e-07	6.80e-07	3.20e-06
	9	3.87e-02	1.61e-02	1.38e-04	1.27e-06	9.66e-08	6.88e-08	3.83e-07	2.09e-06
		ω							
		10^{-6}	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{1}
$\nu = 0.1$	1	8.03e-02	8.03e-02	1.44e-02	1.55e-03	1.57e-04	2.83e-04	6.30e-04	7.92e-04
	2	2.40e-02	2.40e-02	1.22e-03	6.26e-05	5.59e-06	9.11e-06	8.36e-06	8.74e-06
	3	7.36e-03	7.36e-03	1.13e-04	2.95e-06	2.34e-07	4.41e-07	1.68e-06	1.72e-06
	4	2.24e-03	2.24e-03	1.18e-05	2.02e-07	1.55e-08	8.31e-09	5.38e-07	7.01e-07
	5	6.74e-04	6.74e-04	1.55e-06	3.26e-08	2.76e-09	2.36e-09	2.14e-07	2.94e-07
	6	2.02e-04	2.02e-04	3.28e-07	1.13e-08	1.01e-09	4.01e-09	9.47e-08	1.35e-07
	7	6.28e-05	6.28e-05	1.17e-07	5.23e-09	4.75e-10	1.12e-09	4.83e-08	6.81e-08
	8	2.00e-05	2.00e-05	5.44e-08	2.64e-09	2.42e-10	5.45e-10	2.49e-08	3.60e-08
	9	6.89e-06	6.89e-06	2.79e-08	1.40e-09	1.28e-10	2.81e-10	1.32e-08	1.85e-08
		ω							
		10^{-7}	10^{-6}	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	10^{0}
$\nu = 0.01$	1	2.16e-02	1.07e-02	1.48e-03	1.52e-04	1.52e-05	1.52e-06	1.52e-07	1.82e-08
	2	2.68e-03	8.10e-04	2.23e-05	1.09e-06	9.73e-08	9.60e-09	1.10e-09	1.03e-08
	3	3.28e-04	6.40e-05	3.60e-07	9.35e-09	7.51e-10	1.29e-10	6.82e-10	1.07e-08
	4	4.08e-05	5.09e-06	6.78e-09	1.54e-10	2.14e-11	1.17e-10	6.77e-10	1.08e-08
	5	5.84e-06	4.06e-07	2.88e-10	1.35e-11	1.73e-11	1.17e-10	6.11e-10	1.04e-08
	6	1.11e-06	3.28e-08	4.17e-11	2.68e-12	1.80e-11	1.15e-10	7.03e-10	1.05e-08
	7	2.78e-07	2.87e-09	8.90e-12	9.30e-13	1.88e-11	1.18e-10	6.76e-10	1.05e-08
	8	7.86e-08	3.04e-10	2.06e-12	7.24e-13	1.85e-11	1.16e-10	6.74e-10	1.02e-08
	9	2.42e-08	4.68e-11	4.95e-13	7.18e-13	2.23e-11	1.19e-10	7.11e-10	1.02e-08
$\nu = 0.001$	1	1.11e-03	1.44e-04	1.47e-05	1.47e-06	6.30e-06	1.60e-08	4.29e-08	4.77e-08
	2	2.12e-05	4.17e-07	1.40e-08	1.12e-09	1.32e-08	5.31e-09	4.30e-08	4.67e-08
	3	4.16e-07	1.18e-09	1.59e-11	4.90e-11	2.66e-10	5.29e-09	4.31e-08	4.39e-08
	4	8.03e-09	3.17e-12	2.38e-12	4.89e-11	3.02e-11	5.31e-09	4.31e-08	4.69e-08
	5	1.50e-10	2.93e-13	2.32e-12	4.90e-11	2.14e-11	5.30e-09	4.33e-08	4.35e-08
	6	3.11e-12	5.68e-14	2.43e-12	4.91e-11	2.12e-11	5.26e-09	4.29e-08	4.87e-08
	7	1.25e-13	9.27e-15	2.40e-12	4.96e-11	2.16e-11	5.32e-09	4.37e-08	5.03e-08
	8	1.59e-14	3.87e-15	2.39e-12	4.94e-11	2.15e-11	5.36e-09	4.38e-08	4.34e-08
	9	5.77e-15	1.71e-15	2.45e-12	4.95e-11	2.20e-11	5.32e-09	4.39e-08	4.90e-08
$\nu = 0.0001$	1	1.43e-05	1.45e-06	1.46e-07	4.38e-06	6.21e-09	1.79e-08	2.16e-07	2.48e-06
	2	7.75e-09	1.78e-10	2.61e-11	3.62e-09	6.03e-09	1.76e-08	2.20e-07	2.48e-06
	3	4.69e-12	1.67e-12	2.27e-11	1.51e-10	6.15e-09	1.81e-08	2.19e-07	2.48e-06
	4	1.48e-13	1.79e-12	2.26e-11	9.92e-11	6.09e-09	1.78e-08	2.16e-07	2.49e-06
	5	5.29e-14	1.75e-12	2.27e-11	9.87e-11	6.14e-09	1.76e-08	2.16e-07	2.48e-06
	6	1.87e-14	1.78e-12	2.27e-11	9.93e-11	6.08e-09	1.80e-08	2.21e-07	2.47e-06
	7	6.85e-15	1.80e-12	2.29e-11	9.97e-11	6.18e-09	1.80e-08	2.21e-07	2.47e-06
	8	3.37e-15	1.79e-12	2.28e-11	9.97e-11	6.13e-09	1.78e-08	2.25e-07	2.48e-06
	9	2.38e-15	1.74e-12	2.28e-11	9.93e-11	6.14e-09	1.79e-08	2.20e-07	2.48e-06

Table 4.14: Velocity L^2-error for the first 9 (outer) iterations for a range of over- and under-relaxed $\gamma = \omega\alpha_D^{-1}$ values at the endpoint $T = 1$ for function q_1. Ideal parameters are highlighted (in green).

The same conclusions can be made as in the previous section 4.3.3. Except for $\nu = 0.0001$, an *ideal* stabilization parameter is found for all tested ν values with $\omega = 1$ with $\gamma = \omega\alpha_D^{-1} = \theta\nu k$, which provides a good approximation in the first outer iteration. Unfortunately, the velocity error does not improve with further (outer) iterations. To achieve fast convergence in further (outer) iterations, γ has to be adjusted by scaling it downwards.

		$\|p - p_h\|_{L^2}$							
		ω							
		10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{1}	10^{2}
$\nu = 1.0$	1	5.84e+01	5.23e+01	2.91e+01	1.28e+01	1.00e+01	8.67e+00	6.40e+00	4.68e+00
	2	4.71e+01	3.82e+01	1.23e+01	2.56e+00	1.66e+00	1.31e+00	8.63e-01	6.73e-01
	3	3.81e+01	2.81e+01	5.39e+00	6.73e-01	4.31e-01	3.72e-01	3.18e-01	2.83e-01
	4	3.09e+01	2.07e+01	2.45e+00	2.76e-01	2.01e-01	1.89e-01	1.78e-01	1.64e-01
	5	2.50e+01	1.53e+01	1.17e+00	1.56e-01	1.21e-01	1.17e-01	1.12e-01	1.05e-01
	6	2.02e+01	1.13e+01	6.05e-01	9.99e-02	7.91e-02	7.68e-02	7.48e-02	7.06e-02
	7	1.64e+01	8.42e+00	3.42e-01	6.77e-02	5.40e-02	5.25e-02	5.11e-02	4.95e-02
	8	1.32e+01	6.24e+00	2.09e-01	4.67e-02	3.73e-02	3.64e-02	3.55e-02	3.44e-02
	9	1.06e+01	4.62e+00	1.36e-01	3.26e-02	2.61e-02	2.54e-02	2.48e-02	2.41e-02
		ω							
		10^{-6}	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{1}
$\nu = 0.1$	1	2.31e+00	2.31e+00	6.24e-01	2.98e-01	2.62e-01	2.08e-01	8.95e-02	4.01e-02
	2	7.55e-01	7.55e-01	6.15e-02	1.66e-02	1.35e-02	1.01e-02	6.50e-03	5.47e-03
	3	2.47e-01	2.47e-01	8.94e-03	3.30e-03	2.96e-03	2.82e-03	2.52e-03	2.41e-03
	4	8.19e-02	8.19e-02	3.02e-03	1.52e-03	1.39e-03	1.37e-03	1.28e-03	1.24e-03
	5	2.85e-02	2.85e-02	1.51e-03	8.01e-04	7.37e-04	7.29e-04	6.91e-04	6.75e-04
	6	1.11e-02	1.11e-02	8.24e-04	4.40e-04	4.05e-04	4.01e-04	3.82e-04	3.74e-04
	7	5.19e-03	5.19e-03	4.63e-04	2.47e-04	2.27e-04	2.25e-04	2.16e-04	2.10e-04
	8	2.77e-03	2.77e-03	2.62e-04	1.39e-04	1.28e-04	1.27e-04	1.21e-04	1.18e-04
	9	1.59e-03	1.59e-03	1.49e-04	7.87e-05	7.23e-05	7.16e-05	6.87e-05	6.68e-05
		ω							
		10^{-7}	10^{-6}	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	10^{0}
$\nu = 0.01$	1	1.19e-01	5.62e-02	1.11e-02	5.34e-03	4.75e-03	4.70e-03	4.69e-03	4.69e-03
	2	1.81e-02	4.55e-03	1.93e-04	6.39e-05	5.57e-05	5.49e-05	5.48e-05	5.48e-05
	3	2.69e-03	3.83e-04	1.22e-05	7.55e-06	7.15e-06	7.10e-06	7.10e-06	7.10e-06
	4	4.55e-04	3.53e-05	2.59e-06	1.69e-06	1.60e-06	1.59e-06	1.59e-06	1.59e-06
	5	1.06e-04	4.61e-06	6.21e-07	4.07e-07	3.86e-07	3.84e-07	3.84e-07	3.87e-07
	6	3.24e-05	9.68e-07	1.55e-07	1.01e-07	9.61e-08	9.56e-08	9.60e-08	1.04e-07
	7	1.11e-05	2.46e-07	3.93e-08	2.57e-08	2.44e-08	2.43e-08	2.57e-08	4.72e-08
	8	3.88e-06	6.49e-08	1.01e-08	6.56e-09	6.25e-09	6.38e-09	1.03e-08	4.17e-08
	9	1.38e-06	1.74e-08	2.62e-09	1.70e-09	1.64e-09	2.20e-09	8.54e-09	4.00e-08
$\nu = 0.001$	1	1.41e-03	2.10e-04	7.03e-05	5.62e-05	5.43e-06	5.47e-05	5.48e-05	5.46e-05
	2	2.87e-05	6.04e-07	1.08e-07	8.76e-08	7.30e-08	7.79e-08	1.81e-07	7.97e-08
	3	6.00e-07	3.65e-09	2.17e-09	2.08e-09	2.73e-09	1.50e-08	1.18e-07	8.63e-08
	4	1.25e-08	1.25e-10	8.31e-11	1.61e-10	2.12e-10	1.51e-08	1.18e-07	8.59e-08
	5	2.60e-10	6.12e-12	7.62e-12	1.39e-10	6.22e-11	1.51e-08	1.18e-07	8.72e-08
	6	9.35e-12	6.84e-13	6.86e-12	1.39e-10	5.60e-11	1.51e-08	1.18e-07	9.23e-08
	7	1.39e-12	2.63e-13	7.04e-12	1.40e-10	5.52e-11	1.52e-08	1.19e-07	9.16e-08
	8	5.42e-13	1.01e-13	7.27e-12	1.39e-10	5.45e-11	1.52e-08	1.20e-07	8.09e-08
	9	1.98e-13	4.04e-14	7.10e-12	1.40e-10	5.63e-11	1.52e-08	1.19e-07	9.20e-08
$\nu = 0.0001$	1	3.95e-06	9.00e-07	5.92e-07	1.66e-06	5.62e-07	5.58e-07	4.78e-07	6.55e-07
	2	2.22e-09	1.55e-10	9.30e-11	1.89e-09	3.23e-09	2.38e-09	8.26e-08	1.94e-07
	3	6.50e-12	1.02e-12	7.87e-12	8.41e-11	3.21e-09	2.58e-09	8.30e-08	1.90e-07
	4	1.99e-12	7.48e-13	8.05e-12	4.83e-11	3.20e-09	2.38e-09	8.25e-08	1.92e-07
	5	5.98e-13	6.86e-13	8.02e-12	4.87e-11	3.21e-09	2.38e-09	8.08e-08	1.90e-07
	6	1.78e-13	6.95e-13	8.07e-12	4.89e-11	3.19e-09	2.58e-09	8.29e-08	1.92e-07
	7	2.51e-14	7.24e-13	8.15e-12	4.92e-11	3.23e-09	2.52e-09	8.36e-08	1.81e-07
	8	5.99e-15	7.17e-13	8.05e-12	4.91e-11	3.21e-09	2.58e-09	8.32e-08	1.84e-07
	9	4.65e-14	6.49e-13	8.13e-12	4.90e-11	3.23e-09	2.54e-09	8.25e-08	1.87e-07

Table 4.15: Pressure L^2-error for the first 9 (outer) iterations for a range of over- and under-relaxed $\gamma = \omega\alpha_D^{-1}$ values at the endpoint $T = 1$ for function q_1. Ideal parameters are highlighted (in green).

There is a correlation between the divergence-free velocity accuracy and pressure precision. A higher divergence-free velocity and a smaller velocity error result in an improved pressure accuracy. For $\nu = 1$, the change ratio in the right-hand-side is too high. A time step size of $k = 0.01$ is not sufficient here. In analogy to the PP, the step size must probably be chosen smaller, which is analyzed next.

An *ideal* stabilization parameter range for the smallest velocity error and therefore a better pressure precision can be determined from table 4.14, which is illustrated below in table 4.16.

ν	1.0	0.1	0.01	0.001	0.0001
ω	10^{-1}	10^{-2}	$[10^{-4}, 10^{-3}]$	$[10^{-6}, 10^{-5}]$	$[10^{-7}, 10^{-6}]$
$\gamma = \omega(\theta k \nu)^{-1}$	20	20	$[2, 20]$	$[0.2, 2]$	$[0.2, 2]$

Table 4.16: *Ideal* γ values determined for the second and further (outer) iterations with fixed time step $k = 0.01$ using $\theta = 0.5$, the Crank-Nicolson method.

Time step size behavior

This subsection examines the effects of the selected time step size k. The problem from Eq. (4.1) for the function q_1 with $\nu = 1$ is solved for decreasing time step sizes k. For a smaller time step size k, correspondingly more steps K are solved decoupled in time.

Table 4.17 shows the typical behavior from the sequential-in-time PP-solver mentioned in section 3.2. A smaller time step k ensures faster convergence. However, this does not apply in the case of $\gamma = 0$. Without a divergence-free update, there is no convergence acceleration for a smaller time step size in the case of the Stokes equations.

		$\|\mathbf{Su} + k\mathbf{Bp} - \mathbf{g}\|$								
		ω								
		0	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{1}	10^{2}
$K = 100, k = 0.01$	1	3.55e-05	3.24e-04	1.84e-04	6.04e-05	8.58e-05	1.06e-04	1.31e-04	4.45e-04	1.04e-02
	2	1.38e-05	1.25e-04	6.97e-05	2.35e-05	4.59e-05	4.52e-05	4.60e-05	7.74e-05	3.22e-04
	3	9.26e-06	8.51e-05	4.86e-05	2.96e-05	3.37e-05	3.07e-05	3.05e-05	3.46e-05	7.38e-05
	4	7.28e-06	6.70e-05	3.63e-05	3.01e-05	2.52e-05	2.25e-05	2.22e-05	2.34e-05	8.40e-05
	5	6.03e-06	5.52e-05	2.75e-05	2.69e-05	1.89e-05	1.66e-05	1.63e-05	1.64e-05	3.93e-05
	6	5.09e-06	4.63e-05	2.08e-05	2.24e-05	1.40e-05	1.22e-05	1.20e-05	1.20e-05	2.57e-05
	7	4.41e-06	3.98e-05	1.60e-05	1.80e-05	1.04e-05	8.99e-06	8.83e-06	8.76e-06	1.69e-05
	8	3.87e-06	3.47e-05	1.23e-05	1.41e-05	7.60e-06	6.55e-06	6.43e-06	6.36e-06	1.14e-05
	9	3.45e-06	3.06e-05	9.54e-06	1.08e-05	5.54e-06	4.75e-06	4.66e-06	4.60e-06	7.74e-06
$K = 1000, k = 0.001$	1	3.55e-05	5.64e-06	1.83e-06	3.08e-06	7.70e-06	2.31e-05	2.12e-04	2.26e-04	2.30e-04
	2	1.38e-05	1.27e-06	1.44e-06	1.01e-06	1.07e-06	1.25e-06	3.48e-06	3.46e-06	3.55e-06
	3	9.26e-06	1.49e-06	8.77e-07	5.40e-07	5.17e-07	5.40e-07	8.61e-07	9.79e-07	9.88e-07
	4	7.28e-06	1.43e-06	5.10e-07	3.03e-07	2.83e-07	3.03e-07	4.27e-07	4.29e-07	4.30e-07
	5	6.03e-06	1.12e-06	2.95e-07	1.72e-07	1.60e-07	1.86e-07	2.13e-07	2.15e-07	2.16e-07
	6	5.09e-06	7.94e-07	1.71e-07	9.86e-08	9.14e-08	9.94e-08	1.14e-07	1.15e-07	1.15e-07
	7	4.41e-06	5.35e-07	9.90e-08	5.65e-08	5.23e-08	5.60e-08	6.17e-08	6.26e-08	6.27e-08
	8	3.87e-06	3.49e-07	5.73e-08	3.23e-08	2.99e-08	3.21e-08	3.43e-08	3.47e-08	3.48e-08
	9	3.45e-06	2.23e-07	3.31e-08	1.85e-08	1.71e-08	1.78e-08	1.92e-08	1.94e-08	1.94e-08
$K = 10000, k = 0.0001$	1	3.55e-05	5.43e-08	6.46e-08	5.37e-07	3.43e-06	3.86e-06	3.89e-06	3.89e-06	3.89e-06
	2	1.38e-05	1.14e-08	7.86e-09	7.63e-09	1.45e-08	1.66e-08	1.74e-08	1.74e-08	1.74e-08
	3	9.26e-06	2.61e-09	1.74e-09	1.61e-09	1.70e-09	1.98e-09	1.97e-09	1.98e-09	1.98e-09
	4	7.28e-06	6.31e-10	4.20e-10	4.17e-10	4.43e-10	4.64e-10	4.70e-10	4.70e-10	4.70e-10
	5	6.03e-06	1.58e-10	1.05e-10	1.03e-10	1.07e-10	1.21e-10	1.22e-10	1.22e-10	1.22e-10
	6	5.09e-06	4.03e-11	2.66e-11	2.68e-11	2.92e-11	3.24e-11	3.26e-11	3.26e-11	3.26e-11
	7	4.41e-06	1.04e-11	6.86e-12	6.95e-12	7.92e-12	8.85e-12	8.89e-12	8.89e-12	8.89e-12
	8	3.87e-06	2.71e-12	1.78e-12	1.80e-12	2.19e-12	2.44e-12	2.45e-12	2.45e-12	2.45e-12
	9	3.45e-06	7.12e-13	4.66e-13	4.78e-13	6.21e-13	6.79e-13	6.82e-13	6.82e-13	6.82e-13

Table 4.17: Residual for the first 9 (outer) iterations for a range of over- and under-relaxed $\gamma = \omega\alpha_D^{-1}$ values for $\nu = 1.0$ at endpoint $T = 1$ for function q_1. Ideal parameters are highlighted (in green).

		$\|\mathbf{B}^\top \mathbf{u}\|$								
		ω								
		0	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{1}	10^{2}
$K = 100, k = 0.01$	1	1.36e-02	1.05e-02	4.35e-03	6.72e-04	7.12e-05	7.32e-06	8.84e-07	4.12e-07	7.45e-08
	2	9.02e-03	6.69e-03	1.69e-03	1.17e-04	9.74e-06	8.70e-07	8.91e-08	1.50e-08	4.39e-09
	3	7.07e-03	4.79e-03	7.13e-04	2.34e-05	1.60e-06	1.30e-07	9.57e-09	1.04e-09	3.75e-10
	4	5.72e-03	3.50e-03	3.11e-04	6.09e-06	4.09e-07	3.55e-08	3.60e-09	8.66e-10	1.25e-10
	5	4.66e-03	2.57e-03	1.42e-04	2.44e-06	1.83e-07	1.73e-08	1.71e-09	3.66e-10	6.27e-11
	6	3.79e-03	1.89e-03	6.81e-05	1.33e-06	1.07e-07	1.03e-08	1.02e-09	2.20e-10	3.35e-11
	7	3.08e-03	1.40e-03	3.55e-05	8.35e-07	6.81e-08	6.63e-09	6.53e-10	1.37e-10	1.92e-11
	8	2.50e-03	1.04e-03	2.00e-05	5.48e-07	4.51e-08	4.40e-09	4.34e-10	9.02e-11	1.14e-11
	9	2.01e-03	7.65e-04	1.22e-05	3.70e-07	3.06e-08	2.98e-09	2.94e-10	5.99e-11	7.09e-12
$K = 1000, k = 0.001$	1	1.36e-02	4.38e-03	6.77e-04	7.17e-05	7.70e-06	9.27e-07	1.42e-07	1.76e-08	1.54e-09
	2	9.02e-03	1.48e-03	5.65e-05	2.83e-06	2.02e-07	7.85e-09	8.17e-10	5.74e-11	6.17e-12
	3	7.07e-03	5.30e-04	5.50e-06	1.52e-07	1.09e-08	7.14e-10	6.96e-11	7.51e-12	7.56e-13
	4	5.72e-03	1.94e-04	7.72e-07	2.77e-08	2.49e-09	3.25e-10	3.51e-11	3.37e-12	2.64e-13
	5	4.66e-03	7.26e-05	2.37e-07	1.17e-08	1.10e-09	2.84e-10	1.79e-11	1.38e-12	1.19e-13
	6	3.79e-03	2.80e-05	1.11e-07	5.87e-09	5.54e-10	1.74e-10	9.12e-12	6.75e-13	5.95e-14
	7	3.08e-03	1.13e-05	5.81e-08	3.12e-09	2.95e-10	9.45e-11	4.71e-12	3.51e-13	3.13e-14
	8	2.50e-03	4.94e-06	3.16e-08	1.70e-09	1.60e-10	4.95e-11	2.51e-12	1.88e-13	1.69e-14
	9	2.01e-03	2.37e-06	1.76e-08	9.44e-10	8.82e-11	2.31e-11	1.36e-12	1.02e-13	9.30e-15
$K = 10000, k = 0.0001$	1	1.36e-02	7.17e-05	7.22e-06	7.38e-07	7.72e-08	8.68e-09	8.56e-10	8.59e-11	8.64e-12
	2	9.02e-03	8.42e-07	5.04e-08	2.68e-09	2.08e-10	2.64e-11	2.83e-12	3.02e-13	4.93e-14
	3	7.07e-03	1.30e-08	5.49e-10	4.19e-11	2.46e-12	2.25e-13	2.49e-14	1.85e-15	1.53e-15
	4	5.72e-03	8.48e-10	5.15e-11	5.31e-12	6.55e-13	3.58e-14	3.60e-15	9.38e-16	1.36e-15
	5	4.66e-03	1.65e-10	1.06e-11	9.72e-13	1.11e-13	8.47e-15	9.01e-16	8.93e-16	1.71e-15
	6	3.79e-03	3.78e-11	2.45e-12	2.38e-13	3.16e-14	2.39e-15	3.51e-16	7.50e-16	1.71e-15
	7	3.08e-03	9.17e-12	5.96e-13	5.76e-14	6.13e-15	9.32e-16	2.78e-16	7.33e-16	1.60e-15
	8	2.50e-03	2.29e-12	1.49e-13	1.45e-14	1.85e-15	7.82e-16	2.48e-16	8.28e-16	1.49e-15
	9	2.01e-03	5.82e-13	3.77e-14	4.26e-15	1.65e-15	8.58e-16	3.05e-16	7.99e-16	1.67e-15

Table 4.18: Divergence-defect for the first 9 (outer) iterations for a range of over- and under-relaxed $\gamma = \omega\alpha_D^{-1}$ values for $\nu = 1.0$ at endpoint $T = 1$ for function q_1. Ideal parameters are highlighted (in green).

Table 4.18 and figure 4.25 show that without a posterior divergence-free optimization, the same divergence-defect is computed at all times, independent of the selected time step size k.

Table 4.19 shows that although a smaller defect is computed during the first time step with a smaller time step size k, but this additional advantage is quickly lost again without divergence-free optimization. This illustrates the importance of divergence-free conservation in order to preserve the advantage of the PP-solver.

In the case of a posterior divergence-free optimization, convergence acceleration occurs as the time steps decrease.

	$t = 0.0001$	t = 0.001	$t = 0.01$
$k = 0.01$	-	-	1.51e-03
$k = 0.001$	-	2.73e-04	1.51e-03
$k = 0.0001$	3.85e-05	2.73e-04	1.51e-03

Table 4.19: Divergence-defect at different times t without stabilization with $\gamma = 0$.

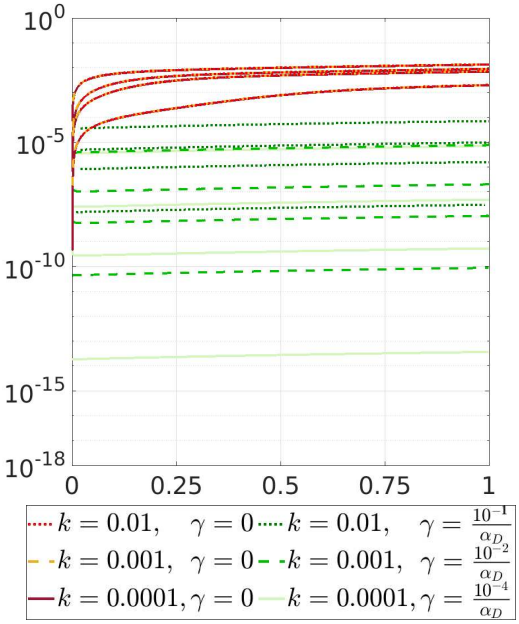


Figure 4.25: Divergence-defect curve for iteration 1 to 3 and 9 for different time steps sizes k, with (in green) and without (in red) stabilization over the time interval [0,1].

		$\|u-u_h\|_{L^2}$								
		ω								
		0	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{1}	10^{2}
$K=100, k=0.01$	1	2.55e-01	2.49e-01	2.14e-01	9.51e-02	1.48e-02	1.57e-03	2.35e-03	5.23e-03	6.92e-03
	2	1.95e-01	1.88e-01	1.47e-01	3.73e-02	2.55e-03	2.13e-04	2.87e-04	4.78e-04	4.21e-04
	3	1.56e-01	1.48e-01	1.05e-01	1.55e-02	4.88e-04	3.27e-05	3.84e-05	7.02e-05	7.56e-05
	4	1.26e-01	1.18e-01	7.65e-02	6.60e-03	1.05e-04	6.03e-06	6.14e-06	1.93e-05	2.78e-05
	5	1.02e-01	9.48e-02	5.58e-02	2.88e-03	2.73e-05	1.52e-06	1.78e-06	6.14e-06	1.23e-05
	6	8.24e-02	7.60e-02	4.08e-02	1.28e-03	9.18e-06	5.62e-07	6.46e-07	2.76e-06	7.44e-06
	7	6.67e-02	6.12e-02	3.00e-02	5.91e-04	4.05e-06	2.77e-07	2.67e-07	1.24e-06	5.03e-06
	8	5.37e-02	4.88e-02	2.20e-02	2.80e-04	2.15e-06	1.58e-07	1.31e-07	6.80e-07	3.20e-06
	9	4.30e-02	3.87e-02	1.61e-02	1.38e-04	1.27e-06	9.66e-08	6.88e-08	3.83e-07	2.09e-06
$K=1000, k=0.001$	1	2.55e-01	9.56e-02	1.49e-02	1.60e-03	5.19e-04	6.60e-04	8.31e-04	8.54e-04	8.57e-04
	2	1.95e-01	3.24e-02	1.24e-03	6.27e-05	1.46e-05	1.05e-05	9.28e-06	9.14e-06	9.25e-06
	3	1.56e-01	1.15e-02	1.14e-04	2.87e-06	8.64e-07	1.08e-06	2.16e-06	1.95e-06	1.96e-06
	4	1.26e-01	4.13e-03	1.17e-05	1.95e-07	1.37e-07	8.35e-07	7.23e-07	6.94e-07	6.95e-07
	5	1.02e-01	1.51e-03	1.55e-06	3.25e-08	3.47e-08	5.38e-07	3.08e-07	2.99e-07	2.99e-07
	6	8.24e-02	5.54e-04	3.34e-07	1.15e-08	1.19e-08	1.64e-07	1.45e-07	1.42e-07	1.42e-07
	7	6.67e-02	2.07e-04	1.20e-07	5.27e-09	4.90e-09	3.93e-08	6.61e-08	7.16e-08	7.18e-08
	8	5.37e-02	7.88e-05	5.58e-08	2.67e-09	2.08e-09	3.28e-08	3.45e-08	3.74e-08	3.75e-08
	9	4.30e-02	3.07e-05	2.87e-08	1.41e-09	9.92e-10	1.31e-08	1.85e-08	2.00e-08	2.00e-08
$K=10000, k=0.0001$	1	2.55e-01	1.58e-03	1.59e-04	4.68e-05	1.11e-04	1.19e-04	1.19e-04	1.19e-04	1.19e-04
	2	1.95e-01	1.84e-05	1.11e-06	9.86e-08	4.28e-07	4.60e-07	4.77e-07	4.77e-07	4.77e-07
	3	1.56e-01	2.46e-07	9.29e-09	1.30e-08	1.95e-08	2.71e-08	2.70e-08	2.71e-08	2.71e-08
	4	1.26e-01	4.56e-09	1.69e-10	4.52e-09	4.83e-09	5.03e-09	5.23e-09	5.22e-09	5.22e-09
	5	1.02e-01	2.58e-10	1.47e-11	7.63e-10	8.88e-10	1.23e-09	1.24e-09	1.24e-09	1.24e-09
	6	8.25e-02	4.22e-11	3.15e-12	1.61e-10	2.24e-10	3.14e-10	3.20e-10	3.20e-10	3.19e-10
	7	6.67e-02	9.12e-12	1.66e-12	2.73e-11	6.74e-11	8.50e-11	8.63e-11	8.58e-11	8.56e-11
	8	5.37e-02	2.12e-12	1.55e-12	1.88e-11	3.06e-11	2.35e-11	2.45e-11	2.38e-11	2.41e-11
	9	4.30e-02	5.15e-13	1.53e-12	2.23e-11	1.50e-11	7.98e-12	9.04e-12	7.65e-12	9.12e-12

Table 4.20: Velocity L^2-error for the first 9 (outer) iterations for a range of over- and under-relaxed $\gamma = \omega\alpha_D^{-1}$ values for $\nu = 1.0$ at endpoint $T = 1$ for function q_1. Ideal parameters are highlighted (in green).

Table 4.20 shows a linear dependency between the *ideal* stabilization parameter γ and the time step size k, which is highlighted in table 4.21.

A smaller time step size leads to a smaller *ideal* stabilization parameter γ. As a result, the momentum equations becomes less ill-conditioned.

Furthermore, less effort is required to maintain a divergence-free velocity. A smaller time step size therefore provides a boost in convergence.

k	0.01	0.001	0.0001
ω	$[10^{-1}, 10^{0}]$	$[10^{-3}, 10^{-2}]$	$[10^{-5}, 10^{-4}]$
$\gamma = \frac{\omega}{\theta k \nu}$	[20,200]	[2,20]	[0.2,2]

Table 4.21: *Ideal* γ values determined for the third and further (outer) iterations with fixed viscosity $\nu = 1.0$ using $\theta = 0.5$ the Crank-Nicolson method.

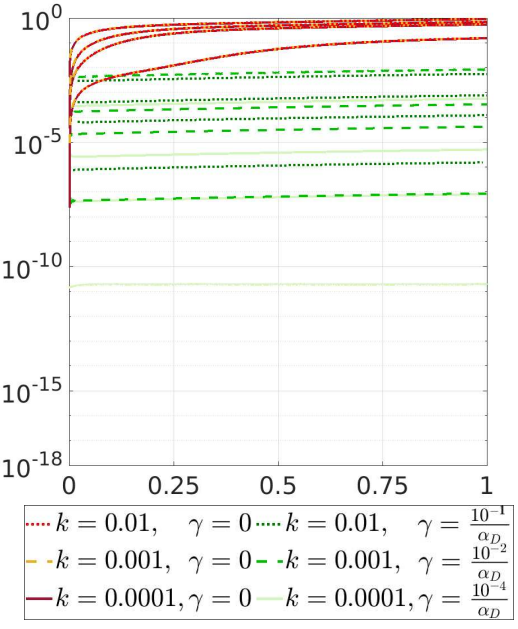


Figure 4.26: Velocity L^2-error curve for iteration 1 to 3 and 9 for different time steps sizes k, with (in green) and without (in red) stabilization over the time interval [0,1].

		$\|p - p_h\|_{L^2}$								
		ω								
		0	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{1}	10^{2}
$K = 100, k = 0.01$	1	5.96e+01	5.84e+01	5.23e+01	2.91e+01	1.28e+01	1.00e+01	8.67e+00	6.40e+00	4.68e+00
	2	4.87e+01	4.71e+01	3.82e+01	1.23e+01	2.56e+00	1.66e+00	1.31e+00	8.63e-01	6.73e-01
	3	3.99e+01	3.81e+01	2.81e+01	5.39e+00	6.73e-01	4.31e-01	3.72e-01	3.18e-01	2.83e-01
	4	3.27e+01	3.09e+01	2.07e+01	2.45e+00	2.76e-01	2.01e-01	1.89e-01	1.78e-01	1.64e-01
	5	2.68e+01	2.50e+01	1.53e+01	1.17e+00	1.56e-01	1.21e-01	1.17e-01	1.12e-01	1.05e-01
	6	2.19e+01	2.02e+01	1.13e+01	6.05e-01	9.99e-02	7.91e-02	7.68e-02	7.48e-02	7.06e-02
	7	1.79e+01	1.64e+01	8.42e+00	3.42e-01	6.77e-02	5.40e-02	5.25e-02	5.11e-02	4.95e-02
	8	1.46e+01	1.32e+01	6.24e+00	2.09e-01	4.67e-02	3.73e-02	3.64e-02	3.55e-02	3.44e-02
	9	1.18e+01	1.06e+01	4.62e+00	1.36e-01	3.26e-02	2.61e-02	2.54e-02	2.48e-02	2.41e-02
$K = 1000, k = 0.001$	1	5.96e+01	2.54e+01	6.19e+00	2.91e+00	1.76e+00	6.54e-01	3.27e-01	3.05e-01	2.99e-01
	2	4.87e+01	9.17e+00	6.04e-01	1.60e-01	8.49e-02	5.56e-02	5.37e-02	5.36e-02	5.35e-02
	3	3.99e+01	3.36e+00	8.91e-02	3.30e-02	2.77e-02	2.40e-02	2.42e-02	2.40e-02	2.39e-02
	4	3.27e+01	1.26e+00	3.06e-02	1.53e-02	1.37e-02	1.27e-02	1.25e-02	1.24e-02	1.24e-02
	5	2.68e+01	4.88e-01	1.54e-02	8.07e-03	7.34e-03	6.93e-03	6.79e-03	6.74e-03	6.73e-03
	6	2.19e+01	1.99e-01	8.38e-03	4.43e-03	4.05e-03	3.85e-03	3.76e-03	3.74e-03	3.73e-03
	7	1.79e+01	8.83e-02	4.69e-03	2.48e-03	2.26e-03	2.22e-03	2.09e-03	2.09e-03	2.09e-03
	8	1.45e+01	4.33e-02	2.65e-03	1.39e-03	1.28e-03	1.25e-03	1.18e-03	1.18e-03	1.18e-03
	9	1.18e+01	2.31e-02	1.51e-03	7.89e-04	7.21e-04	7.07e-04	6.66e-04	6.66e-04	6.66e-04
$K = 10000, k = 0.0001$	1	5.96e+01	8.69e-01	5.12e-01	2.68e-01	1.97e-01	2.39e-01	2.42e-01	2.43e-01	2.43e-01
	2	4.88e+01	1.40e-02	6.06e-03	4.09e-03	2.75e-03	2.76e-03	2.78e-03	2.78e-03	2.78e-03
	3	4.00e+01	1.20e-03	7.56e-04	6.77e-04	5.87e-04	5.82e-04	5.82e-04	5.82e-04	5.82e-04
	4	3.27e+01	2.60e-04	1.69e-04	1.55e-04	1.44e-04	1.45e-04	1.45e-04	1.45e-04	1.45e-04
	5	2.68e+01	6.26e-05	4.08e-05	3.82e-05	3.70e-05	3.76e-05	3.77e-05	3.77e-05	3.77e-05
	6	2.19e+01	1.56e-05	1.01e-05	9.73e-06	9.72e-06	1.00e-05	1.01e-05	1.01e-05	1.01e-05
	7	1.79e+01	3.96e-06	2.57e-06	2.52e-06	2.59e-06	2.72e-06	2.73e-06	2.73e-06	2.73e-06
	8	1.46e+01	1.02e-06	6.61e-07	6.56e-07	7.02e-07	7.48e-07	7.51e-07	7.51e-07	7.51e-07
	9	1.18e+01	2.65e-07	1.71e-07	1.73e-07	1.93e-07	2.07e-07	2.08e-07	2.08e-07	2.08e-07

Table 4.22: Pressure L^2-error for the first 9 (outer) iterations for a range of over- and under-relaxed $\gamma = \omega\alpha_D^{-1}$ values for $\nu = 1.0$ at endpoint $T = 1$ for function q_1. Ideal parameters are highlighted (in green).

If the time step size is too large, such as $k = 0.01$, the divergence-defect is too high, as shown in table 4.18, which leads to a higher L^2-velocity error in table 4.20. The result is a poor pressure approximation, which improves only slowly even after further outer iterations.

A smaller time step size k results in faster convergence due to the reduced divergence-defect. A convergence boost can be seen in the pressure error starting from the second (outer) iteration. During the first (outer) iteration, less improvement can be achieved with a smaller time step size k, since the largest error in the corners, as described in figure 4.20, cannot be reduced by a higher stabilization or a smaller time step size k. In order to obtain a better pressure approximation in the corners during the first iteration, it is necessary to couple the system, which is analyzed in section 4.4.

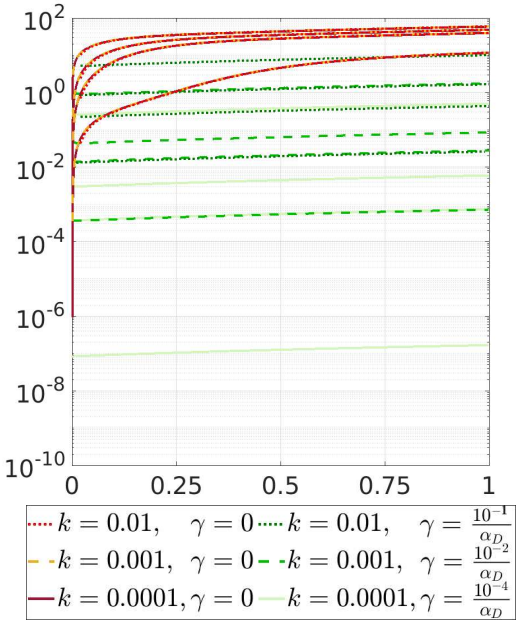


Figure 4.27: Pressure L^2-error curve for iteration 1 to 3 and 9 for different time steps sizes k, with (in green) and without (in red) stabilization over the time interval [0,1].

4.3.5 Analyzing the momentum solver

In contrast to the matrix vector multiplication for the divergence-free update in the pressure space in section 4.1.2, an ill-conditioned system must be solved for the posterior divergence-free velocity optimization. The effect of the choice of the parameter γ on the iteration number and the computation time is examined below for the function $q_1(t)$. Similar results are obtained for other configurations.

The velocity must be computed twice, once as non-divergence-free velocity and once as posterior divergence-free optimization. Both can be computed using the time-simultaneous or parallel-in-time approach from [26].

The non-divergence-free velocity equations from the algorithm 3.4, in line 8, can be computed up to the machine precision, which is shown in figure 4.28. The momentum equations are independent of the parameter γ and have an initial defect in the range of 10^{-5} for each test configuration $q_i(t)$, $i = 1, 2, 3$.

Approx. 13 multigrid iterations are required on average per time step for a (comparatively large) time step size of $k = 0.01$. Figure 4.29 illustrates that it is more efficient than the approx. 50 iterations required without multigrid.

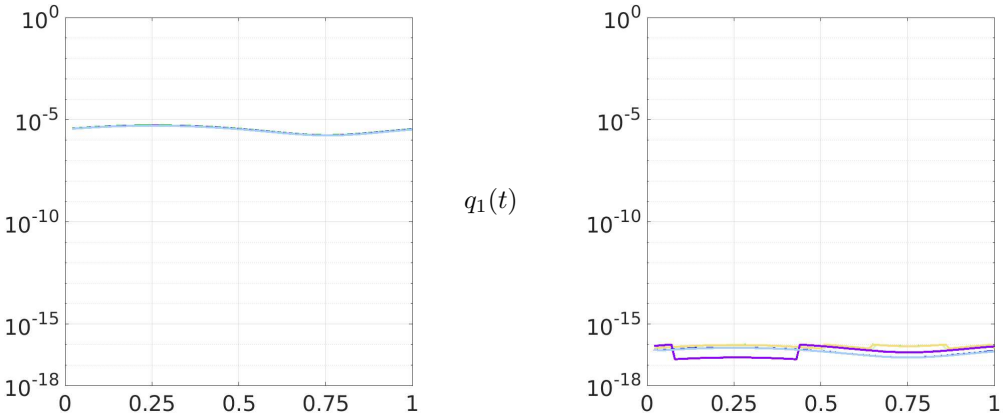


Figure 4.28: Velocity defect at the first (outer) iteration for the non-divergence-free momentum equations before (left) and after (right) computation using time step size $k = 0.01$. The x-axis represents the time interval [0,1].

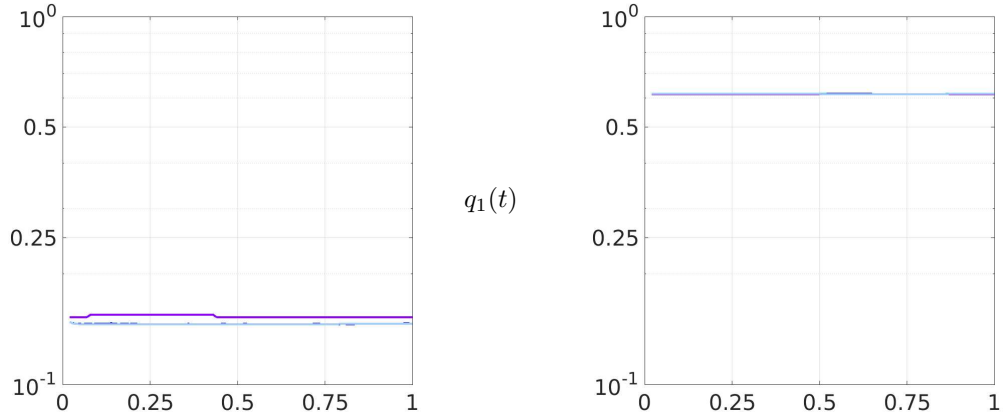


Figure 4.29: Overall convergence rate for the non-divergence-free momentum equations using the multigrid solver (left) and without (right) for time step size $k = 0.01$. The x-axis represents the time interval [0,1].

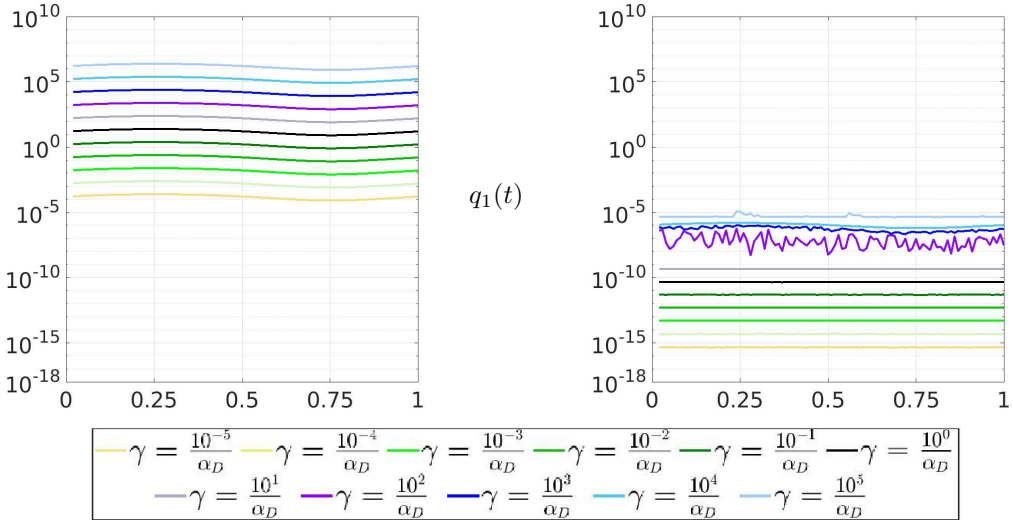


Figure 4.30: Velocity defect at the first (outer) iteration for the γ-stabilized momentum equations before (left) and after (right) solving with a multigrid in space solver for time step size $k = 0.01$ over the time interval [0,1].

Furthermore, the ill-conditioned posterior divergence-free velocity optimization from algorithm 3.4 in line 10 is analyzed. Figure 4.30 shows the accuracy of the posterior divergence-free optimization, which strongly depends on the choice of γ. The stabilization term leads to a high initial defect in the first (outer) iteration, especially for high γ values. Starting from the initial defect, the computer can improve the defect by a factor of 10^{-10}, which is quite impressive but not accurate enough for large γ values. As it can be seen in table 4.11, the inability to achieve machine precision results in a greater L^2- and H^1-error in the velocity space, which leads to a poorer pressure approximation. In all cases, the tolerance was set to machine precision.
The FEAT implementation uses the standard double precison data type (`double`), which has a resolution of 16 digits. A more accurate velocity approximation for high γ values can be achieved only using by a different data type, which provides a higher resolution, but in practice this is not common, see [1].
Since the tolerance of the termination criterion is in general not at machine precision, it is difficult to specify the overall convergence rate. The solver stagnates after a defect correction of approx. 10^{-10} and further iterations do not lead to any improvement. To avoid unnecessary iterations and thus immense computing time, it makes sense to set the termination criterion to

$$\texttt{tol} = 10^{-10+x}, \tag{4.8}$$

scaled by the factor $\omega = 10^x$ with $\gamma = \omega\alpha_D^{-1}$. This ensures that approximately the maximum possible accuracy is achieved without performing too many unnecessary iterations.
The tolerance from Eq. (4.8) generally only applies in the case of under-relaxation with a maximum of $\gamma \leq 10$. Also this tolerance is only valid without the multigrid in space approach. It is not clear how to find a general right tolerance in the multigrid in space approach. This requires further research.

To determine the additional effort for a divergence-free velocity, the iteration numbers and the corresponding computation time are presented in figure 4.31 and figure 4.32 by using Eq. (4.8). If the tolerance was not reached by GMRES, the solver only terminates at the maximum number of 1000 iterations.

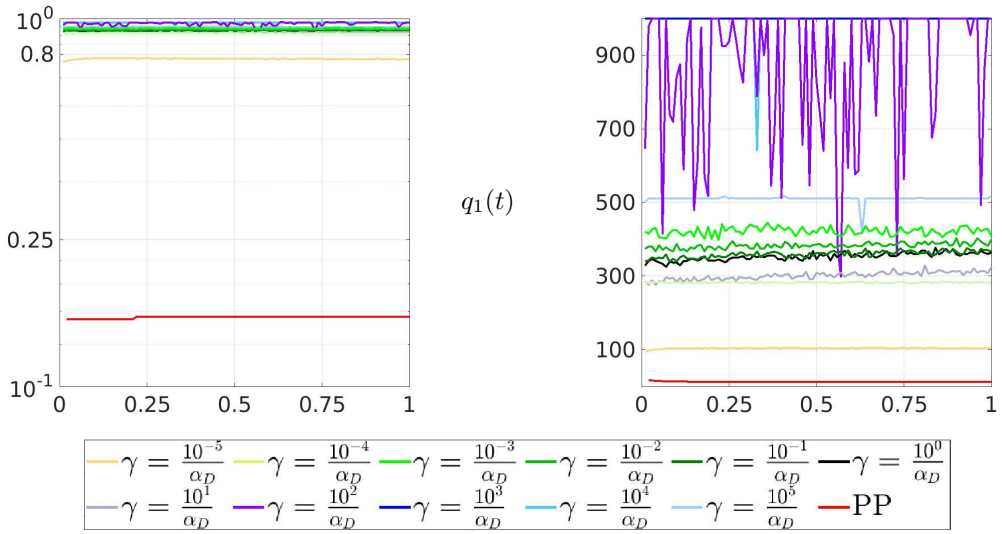


Figure 4.31: Overall convergence rate (left) and number of multigrid iterations (right) for different γ values using time step size $k = 0.01$. The x-axis represents the time interval [0,1].

The non-divergence-free momentum solver is the same solver as used in the PP method: A fast Richardson multigrid solver, which is highlighted in red in figure 4.31 and figure 4.32.

The posterior divergence-free optimization is a highly ill-conditioned problem that can be solved with a GMRES multigrid in space solver. Figure 4.31 shows that the tolerance criterion from Eq. (4.8) is not ideal in the case of an over-relaxated γ, as the iteration numbers may show large oscillations.

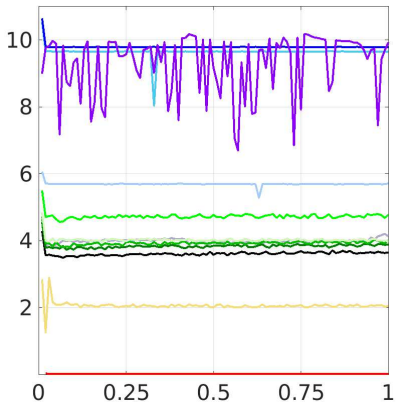


Figure 4.32: Computation time in seconds for different γ values using time step size $k = 0.01$. The x-axis represents the time interval [0,1].

In all observed cases, a higher penalty parameter γ is not advantageous and creates a hard-to-solve problem. The machine precision cannot be achieved, resulting in a poor velocity approximation, which leads to an unprecise pressure approximation. Furthermore, the termination tolerance is not clear, which leads to a high computation time. According to table 4.16 and table 4.21, under-relaxation of γ provides the best convergence results, it is not required to further investigate over-relaxation of γ.

BiCG-stab or the Richardson multigrid solver with the damped Jacobi preconditioner do not lead to convergence in case of the divergence-free momentum equations. There is still a need for further research to determine whether there are better solvers or suitable preconditioners that accelerate the solving process of the ill-conditioned problem. Moreover, the termination condition from Eq. (4.8) of the GMRES multigrid solver is not always valid on the coarser grids, which results in a high computation time, shown in figure 4.32.

The computing time for a fixed time step size $k = 0.01$ is in the range of 4 seconds. Unless γ is scaled much smaller, e.g. with $\omega = 10^{-5}$, machine precision can be achieved and the termination criteria in the multigrid solver works more reliably. For easier comparison, the computations were performed on only one core. The influence of the cores are studies in the upcoming chapter 5.

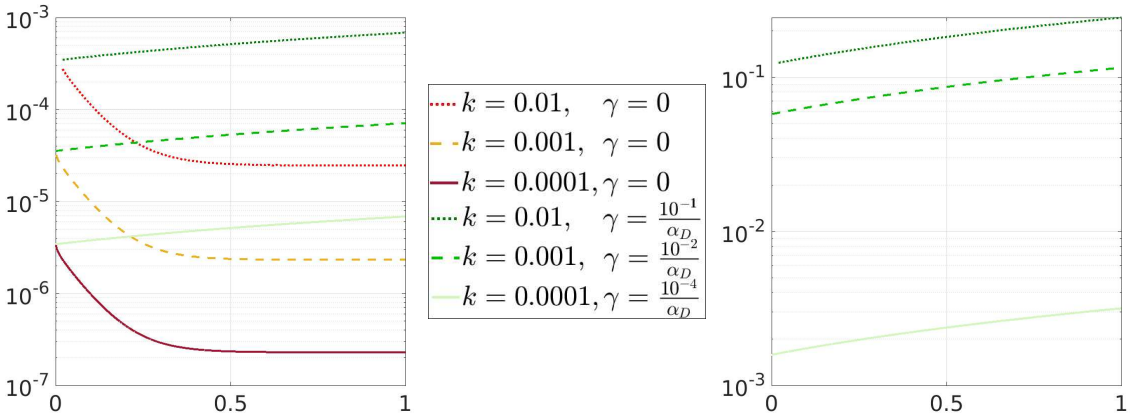


Figure 4.33: Initial defect of the non-divergence-free momentum equations (left) and for the posterior divergence-free optimization (right) for different step sizes k for function q_1 over the time interval [0,1].

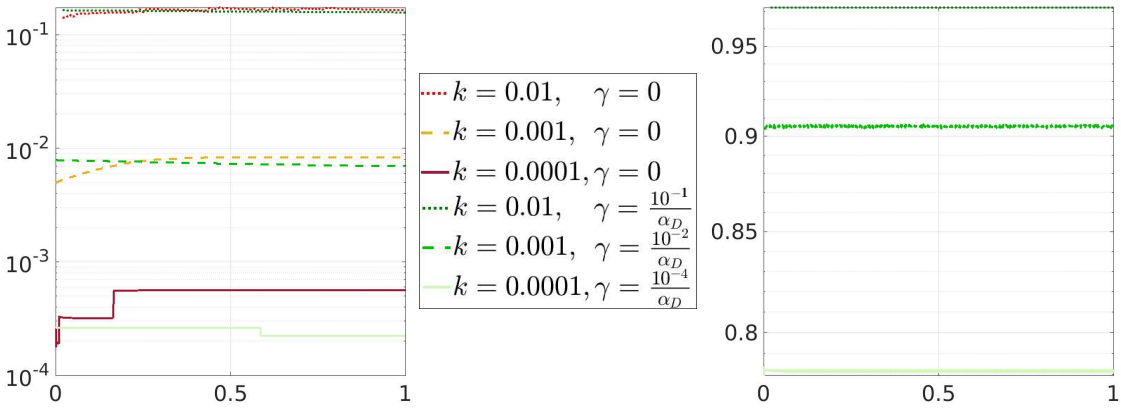


Figure 4.34: Overall convergence rate of the non-divergence-free momentum equations (left) and for the posterior divergence-free optimization (right) for different step sizes k for function q_1 over the time interval [0,1].

A smaller time steps leads to a smaller initial defect in the momentum equations, which can therefore be solved more efficiently. Figure 4.33 demonstrates that in the case of the Stokes equations, the initial defect in the first (outer) iteration decreases linearly with a smaller time step size. In case of the divergence-free momentum equations, a slightly linear increase in the initial defect over time can be observed.

A smaller initial defect resulting from a smaller time step size k leads to a faster achievement of machine precision and avoiding the problem of finding a suitable termination tolerance. As a result, less iterations are required to solve the ill-conditioned problem, as shown in figure 4.34.

Comparison of the computation time (Stokes equations)

The time measurements in table 4.23 are based on the total computing time. This includes the allocation time of the matrices as well as the temporal computation of vectors and synchronization times through MPI. The calculations were performed using 1 core in space and 16 cores in time.

In table 4.23, the divergence-free velocity approach with $\gamma = \alpha_D^{-1}$ is compared with the non-divergence-free velocity solver with $\gamma = 0$. The time step size is set to $k = 0.01$, which solves $K = 100$ decoupled time steps. The pressure Poisson problem is computed parallel-in-time on 16 cores, which incredibly speeds up the solver. Figure 4.35 demonstrate, that the most computing time is now taken by the velocity solvers.

Using multiple cores in space will speed up the computing process, but will not greatly change the ratio between pressure and velocity solver, see results in chapter 5. This shows the necessity of computing the momentum equations simultaneously or in parallel-in-time according to the approach of [26].

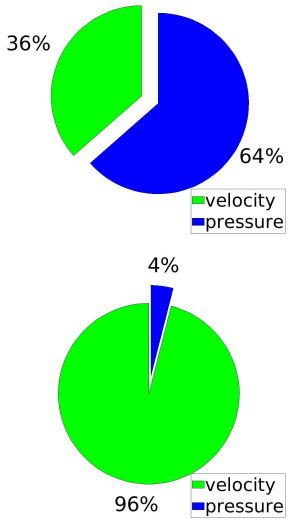


Figure 4.35: Distribution of time among the divergence-free PP-solver (top) and the non-divergence-free TSPP-solver (below).

non-divergence-free velocity solver							divergence-free velocity solver	
time	it.	$\|\mathbf{Su} + k\mathbf{Bp} - \mathbf{g}\|$	$\|\mathbf{B}^\top\mathbf{u}\|$	$\|u - u_h\|_{L^2}$	$\|u - u_h\|_{H^1}$	$\|p - p_h\|_{L^2}$	it.	time
		5.52e-08	7.18e-06	1.52e-04	6.25e-04	5.34e-03	1	7.26 min
1.95 min	11	4.88e-08	5.12e-06	7.88e-05	4.99e-04	4.13e-03		
		7.80e-09	5.19e-08	1.09e-06	4.51e-06	6.39e-05	2	14.46 min
3.49 min	25	5.75e-09	3.76e-08	1.01e-06	3.78e-06	5.36e-05		
		1.74e-09	5.75e-10	9.35e-09	4.34e-08	7.55e-06	3	21.69 min
4.83 min	37	1.52e-09	4.57e-10	8.82e-09	3.28e-08	5.42e-06		
		4.19e-10	5.14e-11	1.54e-10	2.68e-09	1.69e-06	4	28.95 min
5.53 min	43	3.86e-10	4.61e-11	1.38e-10	2.28e-09	1.33e-06		
		1.04e-10	1.06e-11	1.35e-11	5.36e-10	4.07e-07	5	36.22 min
6.04 min	47	9.53e-11	9.66e-12	1.32e-11	4.77e-10	3.27e-07		
		2.65e-11	2.45e-12	2.68e-12	1.24e-10	1.01e-07	6	43.68 min
6.55 min	51	2.29e-11	1.96e-12	2.67e-12	9.68e-11	7.90e-08		
		6.84e-12	5.94e-13	9.30e-13	3.38e-11	2.57e-08	7	50.89 min
6.94 min	54	5.38e-12	5.81e-13	7.86e-13	2.87e-11	1.87e-08		
		1.78e-12	1.48e-13	7.29e-13	1.82e-11	6.59e-09	8	58.10 min
7.40 min	58	1.23e-12	1.11e-13	5.21e-13	1.26e-11	6.25e-09		
		4.65e-13	3.76e-14	7.26e-13	1.73e-11	1.71e-09	9	1.08 hr
7.80 min	61	4.01e-13	3.16e-14	5.21e-13	1.26e-11	1.42e-09		
		1.22e-13	9.68e-15	7.32e-13	1.67e-11	4.45e-10	10	1.21 hr
8.16 min	64	8.75e-14	8.77e-15	5.21e-13	1.26e-11	3.15e-10		

Table 4.23: Comparison of the computation times in the case of the pure Stokes equations. Results are evaluated at endpoint $T = 1$ with $k = 0.01$ and $\nu = 0.01$ from function q_1. The number of (outer) iterations are denoted by it..

Instead of solving the divergence-free momentum equations as best as possible, one idea is to perform only a few GMRES-iterations. Although this leads to a higher number of outer iterations, but can speed up the overall computation process. Without a divergence-free update in the sequential-in-time computed velocity case, around 56 (outer) iterations take almost as long as 1 (outer) iteration of the divergence-free velocity solution approach, but obviously the inefficient choice of the termination criterion is also largely to blame for the much too long computation time.

Furthermore, the posterior divergence-free optimization using the GMRES multigrid solver reacts sensitively after few time steps if too less iterations are performed, similar to the example in table 3.2. Sensitive in the sense that the velocity solver for the non-divergence-free momentum equations may not lead to convergence, especially in the case of a convective term as the Stokes-Oseen equations or Navier-Stokes equations. After only a few iterations, the number of non-linear iterations increases rapidly for the non-divergence-free velocity solver.

4.4 FGMRES defect correction (multigrid in time)

In contrast to the sensitive and highly ill-conditioned posterior divergence-free optimization from section 4.3, the additional divergence-defect correction using FGMRES is more robust. Only a few FGMRES iterations can be performed without convergence problems occurring.

According to figure 4.11 and figure 4.20, the largest pressure error is in the corners. The error in the corners can be corrected using the recoupling approach from section 3.3.4. In the case of the standard PP-solver, the largest pressure error is also located in the corners and was not considered to be a major problem in [78]. Selecting a smaller time step size k leads to a reduced pressure error in the corners. This additional coupled divergence-free correction therefore gives the TSPP-solver CC-solver properties during the first (outer) iteration.

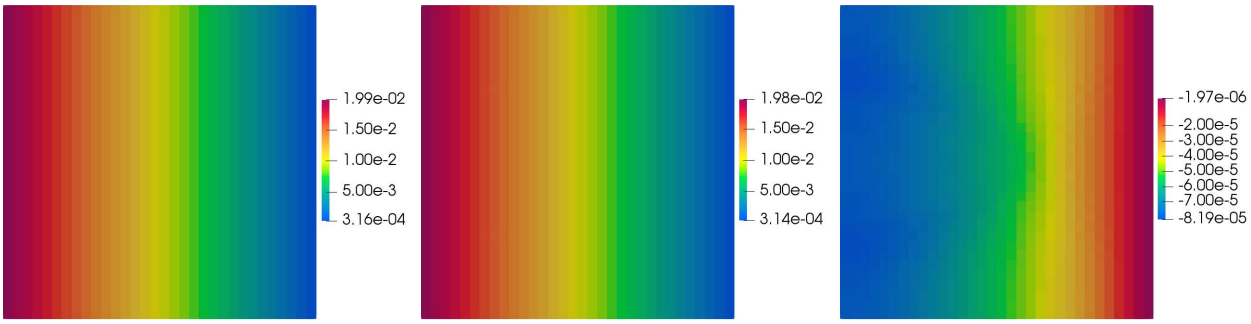


Figure 4.36: Comparison of the analytical pressure solution (left) with the coupled divergence-defect correction pressure approximation (center) and the corresponding absolute error (right) for the first (outer) iteration for q_1 evaluated at endpoint $T = 1$ using 10 FGMRES defect correction iterations at each time step.

Figure 4.36 shows that the pressure approximation does not contain any large errors in the corners. A few FGMRES iterations of the coupled system are sufficient to reduce the pressure error to an order of magnitude of 10^{-6} after just one (outer) iteration.

Since the algorithm is embedded in a multigrid in time, additional computing time is necessary, which is analyzed next.

4.4.1 Analysis of the coupled system

The coupled system in Eq. (3.45) is not computed with a multigrid in space solver, since it is more efficient to perform only a few iterations with the FGMRES algorithm. The results from figure 4.37 were computed on 1 core in space in order to allow better comparability.

Solving the coupled system in Eq. (3.45) is not only associated with the effort of the (F)GMRES solver. The divergence-defect of the velocity field $\mathbf{B}^\top \mathbf{u}$ for the right-hand-side must be recomputed for each time step plus computing time for the actual GMRES algorithm, which can have a significant effect on the total computation time depending on the number of GMRES iterations. According to table 4.24, the FGMRES method requires more computing time, as additional computing time is needed for preconditioning. For a large number of iterations, the additional time is also reduced. Furthermore, it can be seen in the results below that the FGMRES method also achieves higher accuracy. Since the computing time itself does not provide any information, the improvement of the pressure approximation and the associated overall convergence rate are examined in the following.

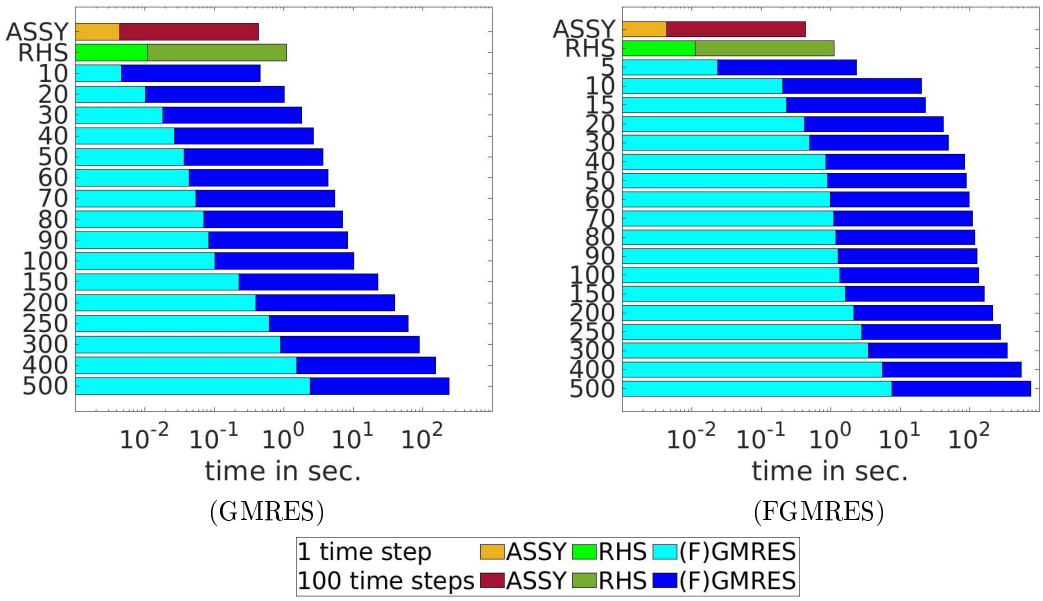


Figure 4.37: Time performance of the coupled system with the GMRES solver (left) and the FGMRES solver (right). The required computation time of one time step for the RHS (light green) and for different (F)GMRES iterations (light blue) compared to the total required computation time for all $K = 100$ time steps (dark green, dark blue) during the first (outer) iteration.

it.	10	20	30	40	50	60	70	80	90
Increase in time	4514%	2721%	2006%	2807%	2187%	1794%	1554%	1305%	1184%

it.	100	150	200	250	300	400	500
Increase in time	1019%	617%	446%	351%	289%	261%	217%

Table 4.24: Overview of the additional time required for the preconditioned FGMRES method in percentage.

In [51], a global approach is taken, which solves a large coupled system that contains all time steps. A different local approach is used here. Small coupled systems are solved sequentially-in-time, which only contain one time step. Similar to the momentum solver, only one CPU is used for the parallel-in-space computing. This means, as illustrated in figure 3.8, that a large number of cores are in idle mode during the computation of the coupled system. A strong acceleration can also be expected if more cores are made accessible, for example using the parareal approach from [48].

For a valid comparison between the GMRES and FGMRES methods, a suitable preconditioner must first be specified. The Uzawa preconditioner from [5], a special preconditioner for saddle-point systems of the form Eq. (3.45), is chosen. The Uzawa diagonal type is chosen, which solves the system

$$\begin{bmatrix} \mathbf{S} & 0 \\ 0 & -\mathbf{B}^\top \mathbf{S}^{-1} \mathbf{B} \end{bmatrix} \begin{bmatrix} \mathbf{u}^l \\ \mathbf{p}^l \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{B}^\top \mathbf{u}^l \end{bmatrix}, \quad \text{for } l = 1, \dots, K. \tag{4.9}$$

The required solution steps of $\mathbf{S}^{-1}$ and $\left(-\mathbf{B}^\top \mathbf{S}^{-1} \mathbf{B}\right)^{-1}$ are performed by two sub-solvers, which are

i. Richardson solver with Jacobi precondition for $\mathbf{S}^{-1}$,

ii. ILU for the approximation of the Schur-complement matrix $\left(-\mathbf{B}^\top \mathbf{S}^{-1} \mathbf{B}\right)^{-1}$.

Both subsolvers are used without the multigrid in space approach.
Figure 4.38 shows the advantage of the FGMRES method. Even after a few FGMRES iterations, a continuous improvement can be seen. If fully iterated, faster convergence can be achieved compared to the GMRES method. The standard GMRES method, on the other hand, shows that a minimum number of iterations must be performed in order to avoid deterioration at larger times.

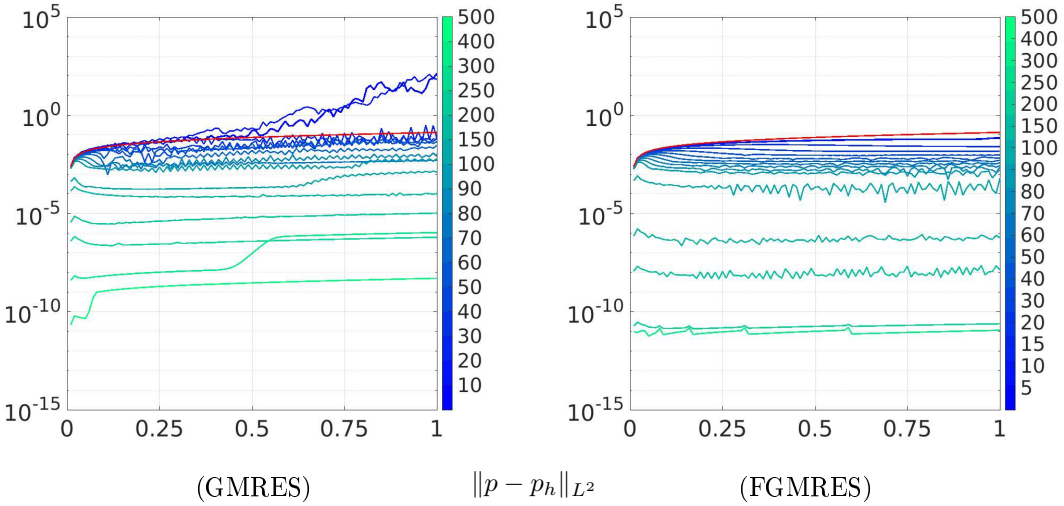


Figure 4.38: Pressure L^2-error after various iterations of the coupled system Eq. (3.45) without preconditioning as GMRES method (left) and with Uzawa preconditioning as FGMRES method (right). The pressure defect before applying the (F)GMRES method is plotted as reference (red). Results for function q_1 with step size $k = 0.01$, viscosity $\nu = 0.01$ and the setting $\alpha_R = 1$, $\alpha_D = \theta\nu k$ and $\gamma = 0$. The x-axis represent the time interval [0,1] and the colorbar the number of performed (F)GMRES iterations.

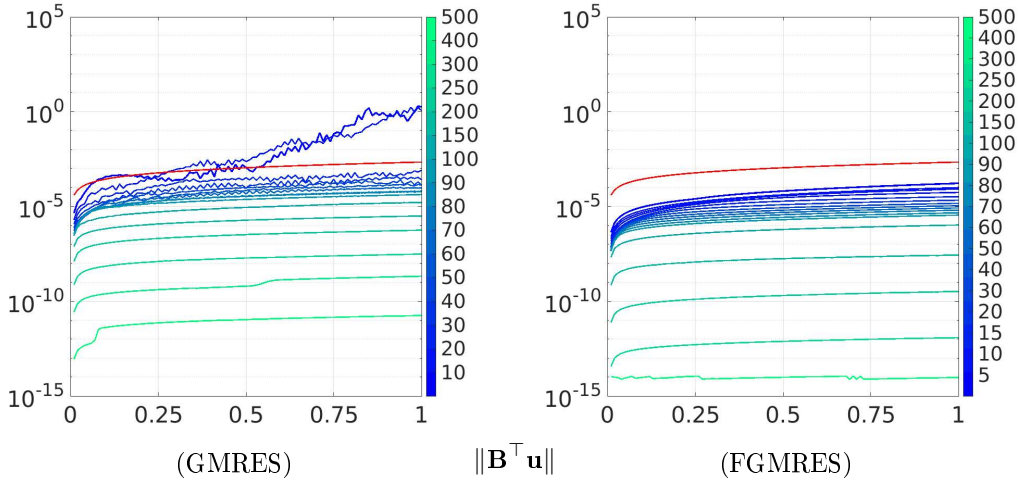


Figure 4.39: Divergence-defect after various iterations of the coupled system Eq. (3.45) without preconditioning as GMRES method (left) and with Uzawa preconditioning as FGMRES method (right). The divergence-defect before applying the (F)GMRES method is plotted as reference (red). The x-axis represent the time interval [0,1] and the colorbar the number of performed (F)GMRES iterations.

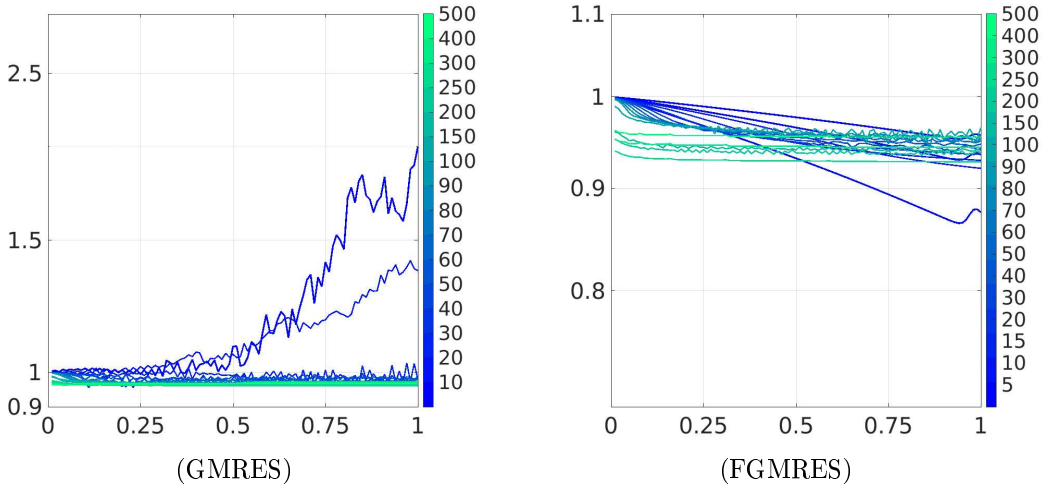


Figure 4.40: Convergence rate for the pressure after various iterations of the coupled system Eq. (3.45) without preconditioning as GMRES method (left) and with Uzawa preconditioning as FGMRES method (right). The x-axis represent the time interval [0,1] and the colorbar the number of performed (F)GMRES iterations.

Figure 4.39 shows that the FGMRES algorithm achieves a divergence-free velocity more quickly. A divergence-free boost is also visible during the first iterations.

In addition, figure 4.40 presents the convergence rate. A poor convergence rate can be observed in both cases, whereby FGMRES performs slightly better. It is therefore not recommended to iterate the coupled problem until the end, but rather to perform a few more outer iterations if a higher accuracy is required.

Furthermore, the PP-solver from section 3.2 is also not recommended for the computing of Eq. (3.45), as the error in the corners cannot be corrected by a decoupled solver during the first (outer) iteration. A full CC approach is neither recommended for the coupled system, as this would take up too much computing time. It is completely sufficient not to solve the coupled system exactly and to perform only a few (F)GMRES iterations.

4.4.2 Timing of the multigrid in time approach

The coupled system Eq. (3.45) is considered as a multigrid in time algorithm. The idea here is to reduce the total computing time of the coupled system. Therefore, no pre-smoothing and post-smoothing steps are performed, i.e. the pressure is only corrected using the coarse time grid. By using the multigrid in time approch from section 3.3.4, the computational effort can be halved at a first view, but it should also be noted that for a larger time step size k more computation time is required for the right-hand-side $\mathbf{B}^\top \mathbf{u}$, since the preconditioners from section 3.2.1 getting less accurate. See figure 4.33, figure 4.34 and also figure 1.9 and figure 1.10 from the introduction.

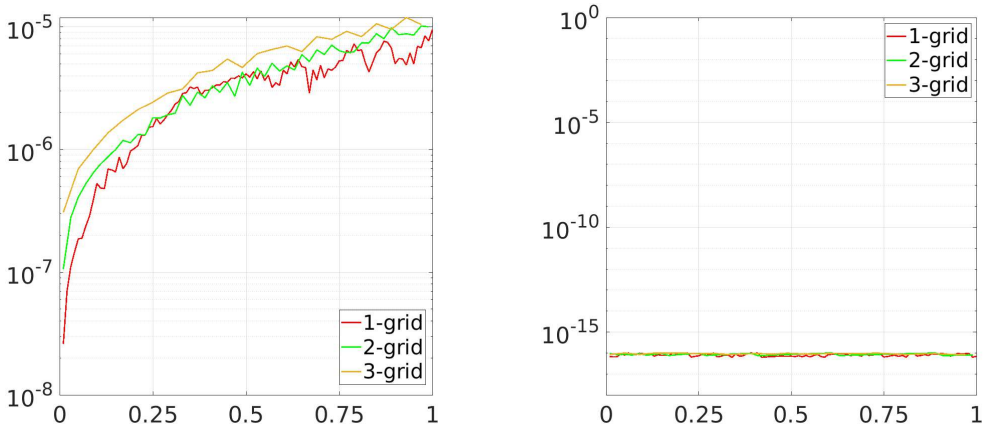


Figure 4.41: Starting defect (left) and end defect (right) of the right-hand-side divergence-defect using $\gamma = 0$ for q_1 over the time interval [0,1].

The convergence rate in figure 4.41 is independent of the choice GMRES or FGMRES, as the divergence-defect represent the right-hand-sides of the coupled systems. Since the divergence-defect must be recomputed for a pre- and post-smoothening, only results on the coarse grid are shown here.
Figure 4.42 and figure 4.43 show the iteration number and convergence curve for the right-hand-sides of the coupled systems. On average, without a multigrid in space approx. twice as many iterations are required for the 2-grid and approx. three times for the 3-grid in time method. For getting a performance gain, it is therefore recommended that the multigrid in time approach should always be used with a multigrid in space solver.

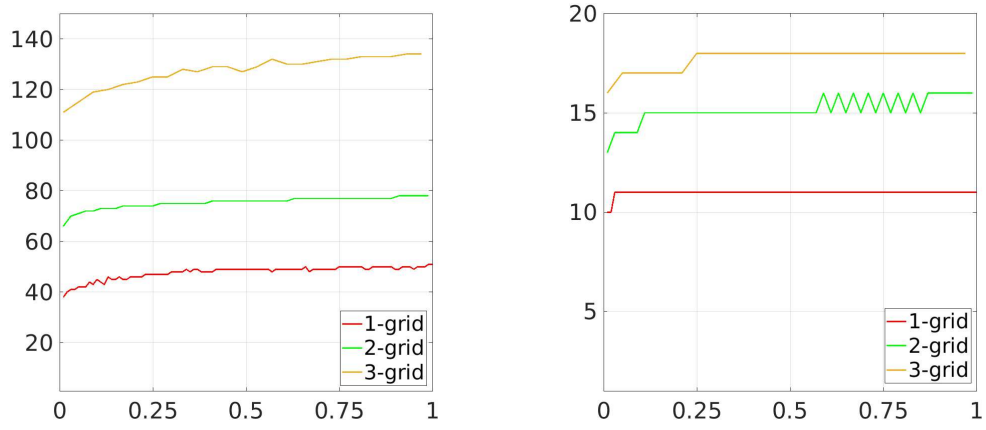


Figure 4.42: Iteration numbers for the right-hand-sides of the coupled systems without a multigrid in space (left) and with a multigrid in space solver (right) using $\gamma = 0$ for q_1 over the time interval [0,1].

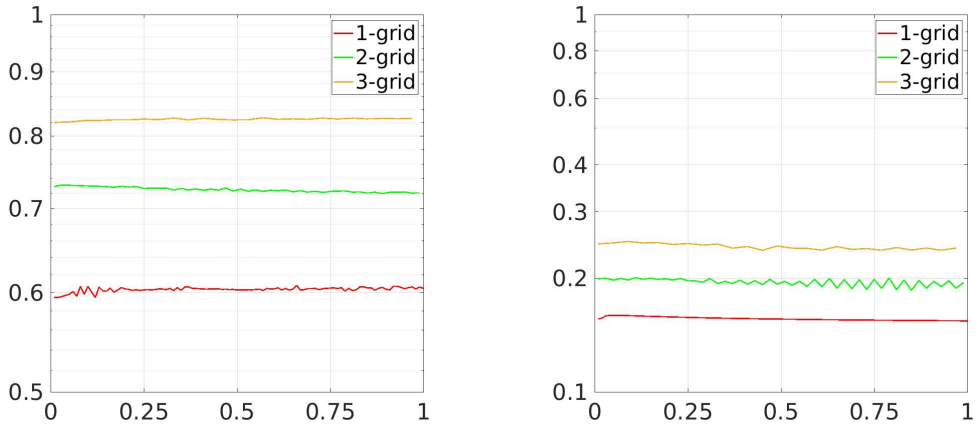
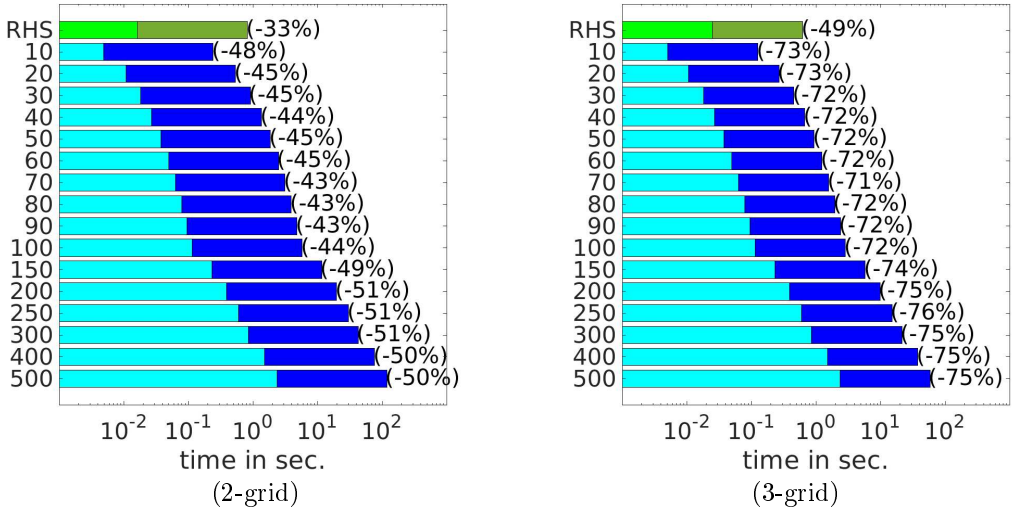


Figure 4.43: Overall convergence rate for the right-hand-sides of the coupled systems without a multigrid in space (left) and with a multigrid in space solver (right) using $\gamma = 0$ for q_1 over the time interval [0,1].

GMRES

Figure 4.44 shows that the same number of GMRES iterations can be performed in half the computing time by using a coarser time level. Using a time grid twice as coarse results in a quarter of the computing time for the GMRES iterations. However, this behavior cannot be observed for the right-hand-side. A coarser time step results in a poorly conditioned momentum problem, which effectively takes longer to solve.



(2-grid) (3-grid)

Figure 4.44: Time comparison for GMRES method for the 2-grid in time (left) and 3-grid in time (right) using $\gamma = 0$ for q_1.

Table 4.25 presents the pressure defect for different GMRES iterations during the first (outer) iteration. For better visualization, the corresponding pressure curve is shown in figure 4.45 with the convergence rate in figure 4.46. Similar improvements can be achieved for the Stokes equations irrespective of the time grid. In the case of a coarser time grid, small oscillations can occur using the local coupled multigrid in time acceleration.
In addition, the convergence rate is still very low even with a coarser time grid.

it.	10	20	30	40	50	60	70	80
1-grid	1.33e+02	6.70e+01	6.69e-02	5.91e-02	6.50e-02	4.44e-02	2.86e-02	5.06e-03
2-grid	7.45e+01	2.06e-01	1.14e-01	5.13e-02	3.15e-02	4.35e-02	1.71e-02	8.10e-03
3-grid	1.66e-01	1.14e-01	8.38e-02	6.23e-02	4.73e-02	6.60e-02	3.96e-02	2.56e-02

it.	90	100	150	200	250	300	400	500
1-grid	7.87e-03	6.09e-03	1.26e-03	1.03e-04	1.04e-05	6.04e-07	1.06e-06	4.96e-09
2-grid	2.48e-02	8.09e-03	1.86e-03	9.57e-05	9.61e-06	8.50e-07	2.16e-06	3.04e-08
3-grid	1.96e-02	7.19e-03	3.65e-03	1.41e-04	1.39e-05	2.37e-06	3.42e-06	5.89e-08

GMRES

Table 4.25: $\|p - p_h\|_{L^2}$-error evaluated at endpoint $T = 1$ for the multigrid in time case for the GMRES method without pre- and post-smoothening for q_1.

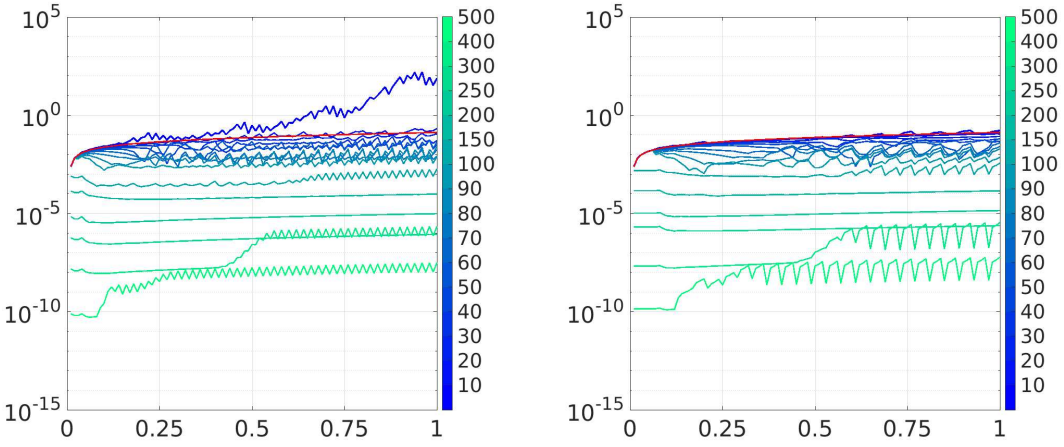


Figure 4.45: $\|p - p_h\|_{L^2}$-error curve for the 2-grid in time (left) and for the 3-grid in time (right) for the GMRES method without pre- and post-smoothening for q_1. The x-axis represent the time interval [0,1] and the colorbar the number of performed GMRES iterations.

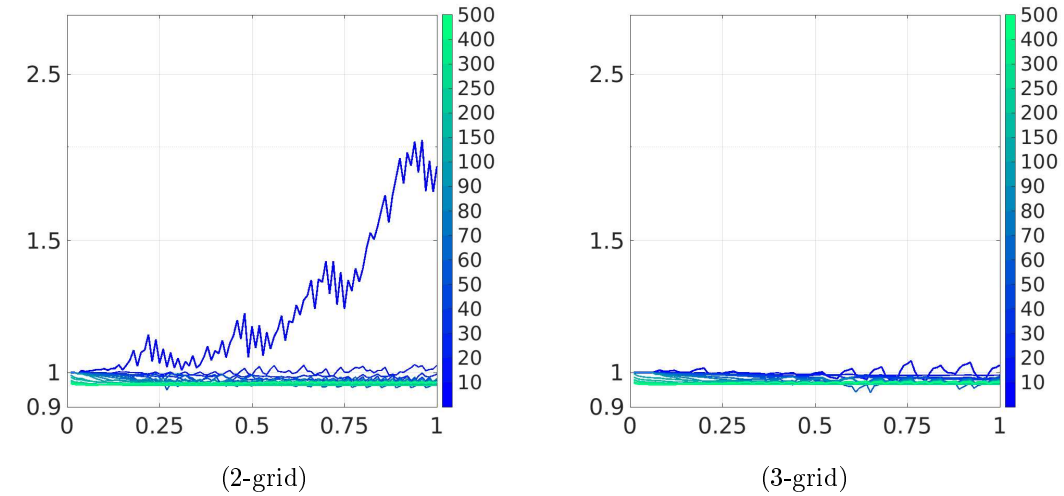


Figure 4.46: Overall convergence rate of the pressure for the 2-grid in time (left) and for the 3-grid in time (right) for the GMRES method without pre- and post-smoothening for q_1. The x-axis represent the time interval [0,1] and the colorbar the number of performed GMRES iterations.

FGMRES

Compared to figure 4.44 of the GMRES approach, no time savings can be observed for the FGMRES solver in figure 4.47. With half as many time steps, even more time is needed than on the fine time grid. The cause is that the preconditioners become worse on a bigger time step size and therefore require more iterations.

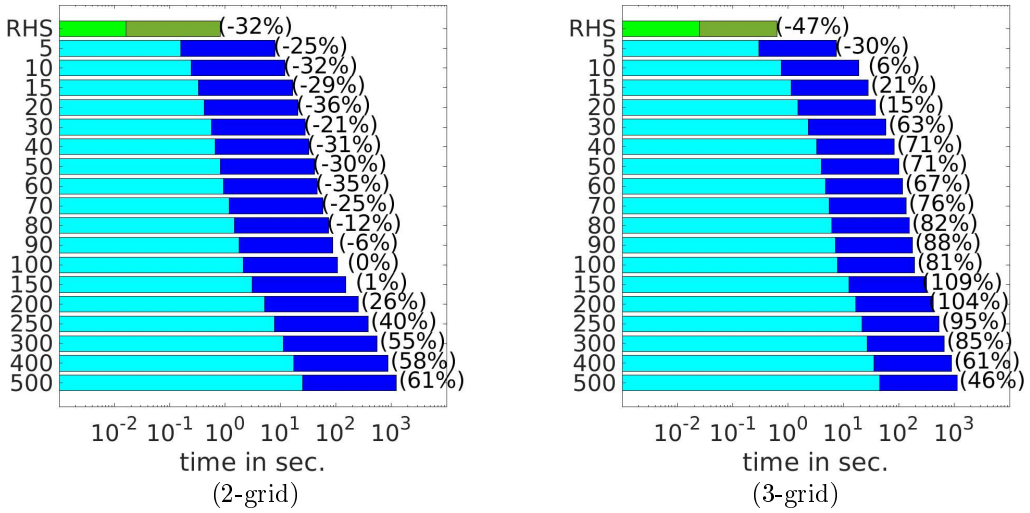


Figure 4.47: Time comparison for FGMRES method for the 2-grid in time (left) and 3-grid in time (right) using $\gamma = 0$ for q_1.

Table 4.26, presents the pressure defect for different FGMRES iterations during the first (outer) iteration. For a better visualization, the corresponding pressure curve is shown in figure 4.48 with the convergence rate in figure 4.49. Similar improvements can be achieved for the Stokes equations irrespective of the time grid. Compared to the GMRES method, no oscillations occur. Moreover, a slightly better convergence rate can be observed.

The better convergence rate is achieved by using a preconditioner. Since, according to figure 4.47, a large part of the computation time is spent on the preconditioner, the FGMRES method is not recommended in case of a multigrid in time approach.

it.	5	10	15	20	30	40	50	60	70
1-grid	1.13e-01	6.80e-02	5.94e-02	2.55e-02	1.48e-02	9.20e-03	6.66e-03	5.02e-03	3.78e-03
2-grid	1.18e-01	8.67e-02	6.84e-02	4.19e-02	2.15e-02	1.37e-02	9.01e-03	6.07e-03	4.07e-03
3-grid	1.24e-01	9.98e-02	8.42e-02	5.88e-02	2.88e-02	1.70e-02	1.02e-02	5.94e-03	3.46e-03

it.	80	90	100	150	200	250	300	400	500
1-grid	2.72e-03	2.29e-03	1.63e-03	6.23e-04	5.69e-07	1.34e-08	2.39e-11	1.11e-11	1.11e-11
2-grid	2.87e-03	1.66e-03	1.16e-03	1.21e-05	2.74e-08	2.74e-11	5.04e-12	5.04e-12	5.09e-12
3-grid	2.07e-03	1.41e-03	6.58e-04	9.44e-08	2.70e-11	2.81e-12	2.81e-12	2.89e-12	2.82e-12

FGMRES

Table 4.26: $\|p - p_h\|_{L^2}$-error evaluated at endpoint $T = 1$ for the multigrid in time case for the FGMRES method without pre- and post-smoothening for q_1.

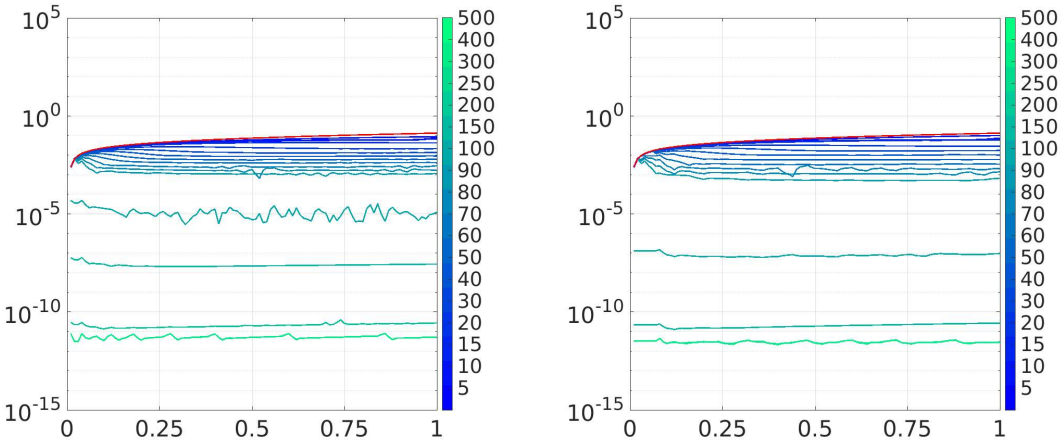


Figure 4.48: $\|p - p_h\|_{L^2}$-error curve for the 2-grid in time (left) and for the 3-grid in time (right) for the FGMRES method without pre- and post-smoothening for q_1. The x-axis represent the time interval [0,1] and the colorbar the number of performed FGMRES iterations.

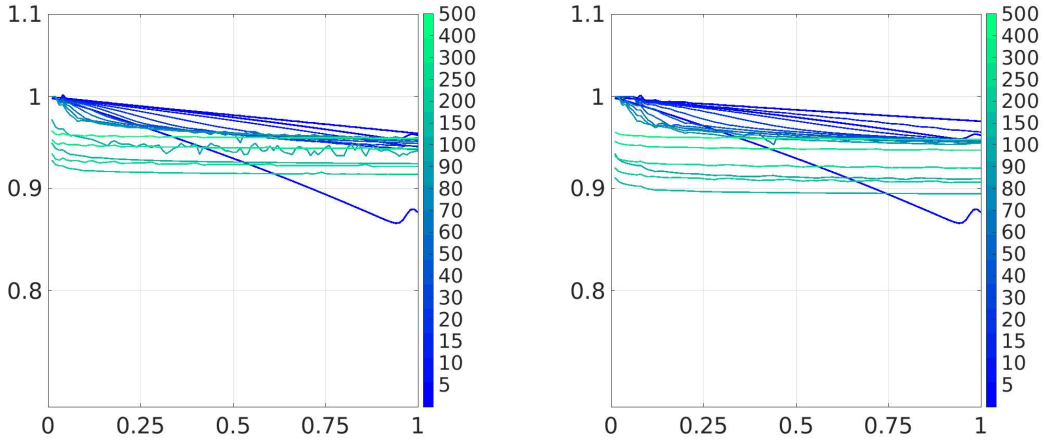


Figure 4.49: Overall convergence rate of the pressure for the 2-grid in time (left) and for the 3-grid in time (right) for the FGMRES method without pre- and post-smoothening for q_1. The x-axis represent the time interval [0,1] and the colorbar the number of performed FGMRES iterations.

4.4.3 Analysis of the coupled system using previous augmented Lagrangian

This section examines the influence of the augmented Lagrangian from section 4.3 for the coupled system Eq. (3.45). The same setting as before is used with the *ideal* stabilization parameter of $\gamma = 0.1$ according to section 4.3.4.

The L^2-pressure error in figure 4.50 shows a similar behavior as in figure 4.38. As before, the standard GMRES method requires a minimum number of iterations to achieve convergence. Also, even after a few FGMRES iterations, an improvement of the pressure error can be seen. Furthermore, compared to figure 4.38, the defect is only resolved up to 10^{-10}.

Figure 4.51 shows that, compared to figure 4.40, the FGMRES solver achieves a better convergence rate of the pressure defect, especially in the first iterations.

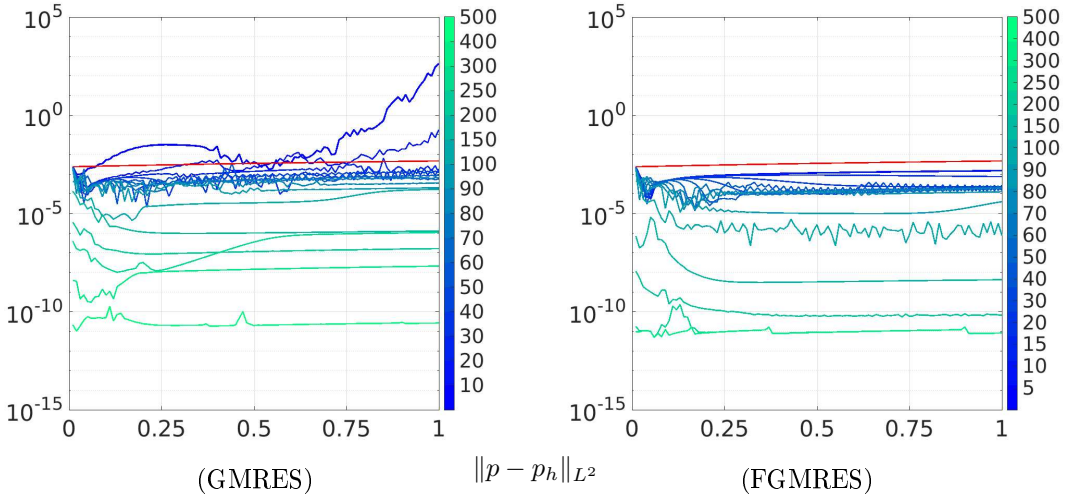


Figure 4.50: Pressure L^2-error after various iterations of the coupled system Eq. (3.45) without preconditioning as GMRES method (left) and with Uzawa preconditioning as FGMRES method (right). The pressure defect before applying the (F)GMRES method is plotted as reference (red). Results for the function q_1 with step size $k = 0.01$ and viscosity $\nu = 0.01$ using the settings $\alpha_R = 1$, $\alpha_D = \theta\nu k$ and $\gamma = 0.1$. The x-axis represent the time interval [0,1] and the colorbar the number of performed (F)GMRES iterations.

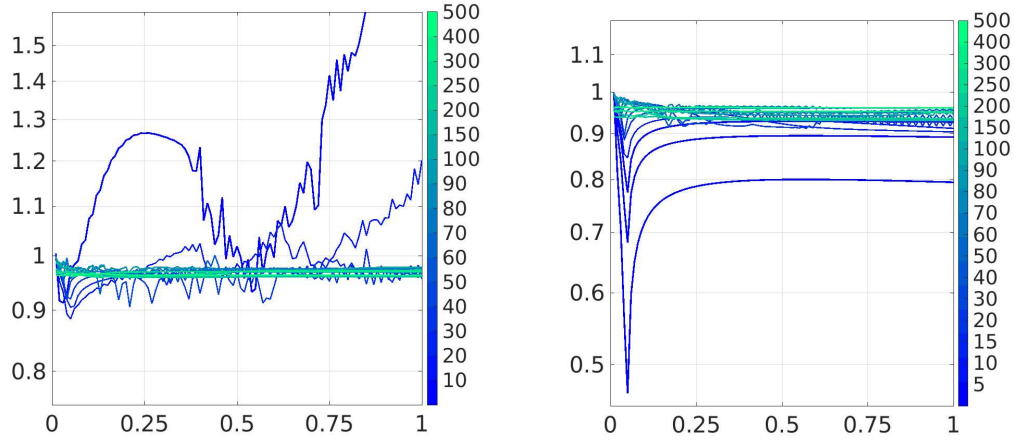


Figure 4.51: Overall convergence rate for the pressure after various iterations of the coupled system Eq. (3.45) without preconditioning as GMRES method (left) and with Uzawa preconditioning as FGMRES method (right). The x-axis represent the time interval [0,1] and the colorbar the number of performed (F)GMRES iterations.

GMRES

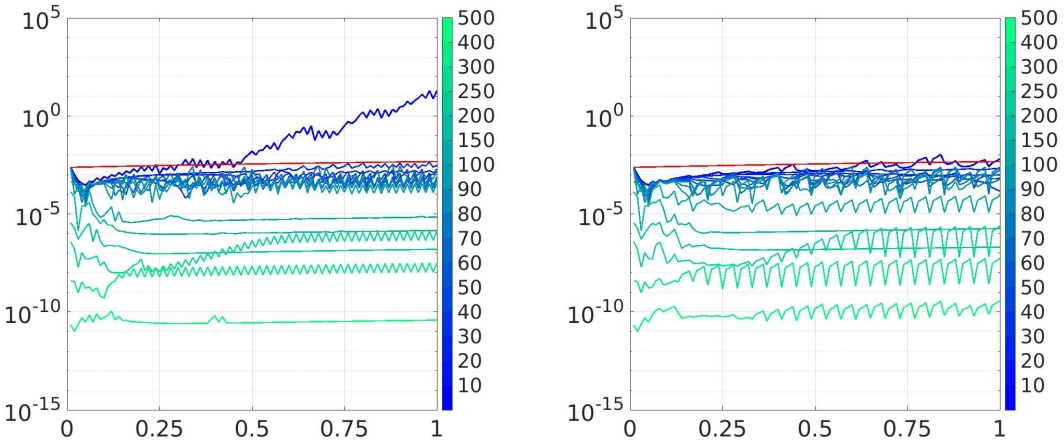
Table 4.27, figure 4.52 and figure 4.53 show for the 2-grid and 3-grid in time the common pressure errors with the corresponding convergence rates. For the multigrid in time method, again small oscillations occur for the local coupled GMRES acceleration. The same behavior is observed as before, where a minimum number of GMRES iterations must be performed to obtain convergence. In addition, the convergence rate remains very low.

it.	10	20	30	40	50	60	70	80
1-grid	4.23e+02	1.86e-01	1.22e-03	1.83e-03	1.18e-03	5.83e-04	8.90e-04	6.48e-04
2-grid	1.92e+01	3.06e-03	1.32e-03	1.99e-03	7.93e-04	6.83e-04	5.68e-04	1.51e-03
3-grid	6.45e-03	2.13e-03	1.47e-04	9.99e-04	6.77e-04	1.67e-03	6.18e-04	2.49e-03

it.	90	100	150	200	250	300	400	500
1-grid	3.44e-04	2.03e-04	1.67e-04	1.28e-06	1.61e-07	1.07e-06	2.07e-08	2.59e-11
2-grid	6.50e-04	3.54e-04	6.88e-06	1.41e-06	1.55e-07	1.33e-06	3.19e-08	3.80e-11
3-grid	1.59e-03	1.87e-03	8.85e-05	1.79e-06	2.02e-07	2.38e-06	5.41e-08	3.72e-10

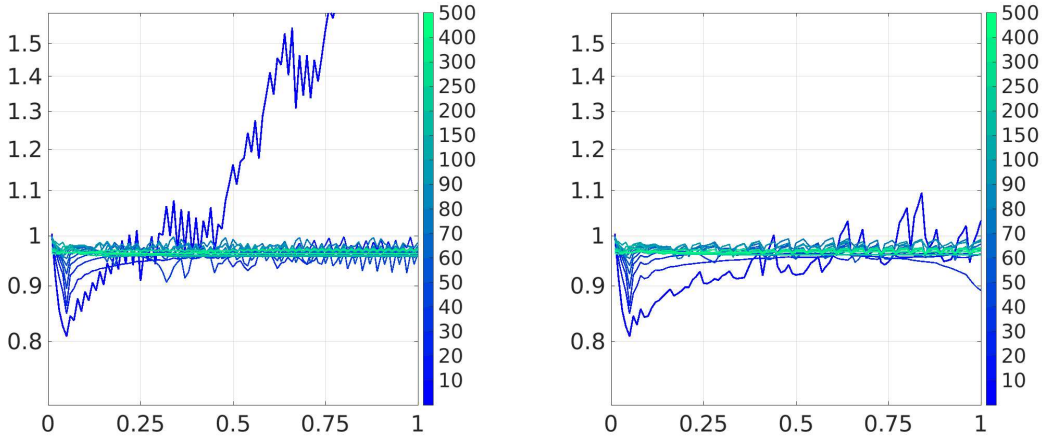
GMRES

Table 4.27: Pressure L^2-error evaluated at endpoint $T = 1$ for the multigrid in time case for the GMRES method without pre- and post-smoothening for q_1



(2-grid) $\|p - p_h\|_{L^2}$ (3-grid)

Figure 4.52: Pressure L^2-error for GMRES using 2-grid in time (left) and 3-grid in time (right). The x-axis represent the time interval [0,1] and the colorbar the number of performed GMRES iterations.



(2-grid) (3-grid)

Figure 4.53: Overall convergence rate of the pressure for GMRES using 2-grid in time (left) and 3-grid in time (right). The x-axis represent the time interval [0,1] and the colorbar the number of performed GMRES iterations.

FGMRES

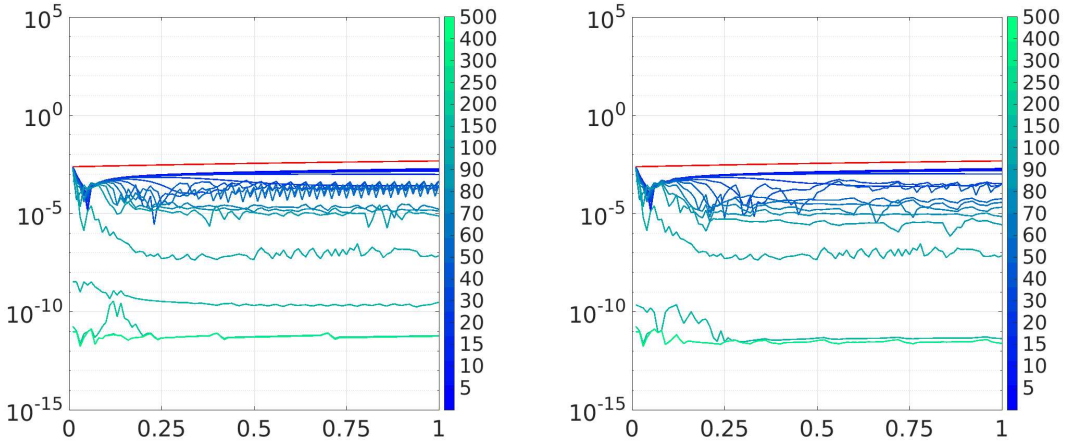
The 2-grid and 3-grid in time has no particular innovations in the FGMRES case either.

it.	5	10	15	20	30	40	50	60	70
1-grid	1.78e-03	1.49e-03	1.23e-03	7.39e-04	2.25e-04	2.69e-04	1.43e-04	2.04e-04	1.80e-04
2-grid	1.83e-03	1.62e-03	1.42e-03	9.58e-04	2.45e-04	4.48e-04	2.82e-04	4.49e-04	2.76e-04
3-grid	1.86e-03	1.68e-03	1.51e-03	9.99e-04	3.36e-04	3.05e-04	3.11e-04	5.57e-05	3.62e-05

it.	80	90	100	150	200	250	300	400	500
1-grid	1.97e-04	1.25e-04	3.96e-05	7.49e-07	4.20e-09	6.61e-11	8.12e-12	8.00e-12	8.00e-12
2-grid	1.62e-05	1.31e-05	7.34e-06	6.93e-08	2.98e-10	5.90e-12	5.56e-12	5.58e-12	5.57e-12
3-grid	2.07e-05	6.87e-06	2.64e-06	1.06e-09	4.24e-12	2.35e-12	2.34e-12	2.33e-12	2.38e-12

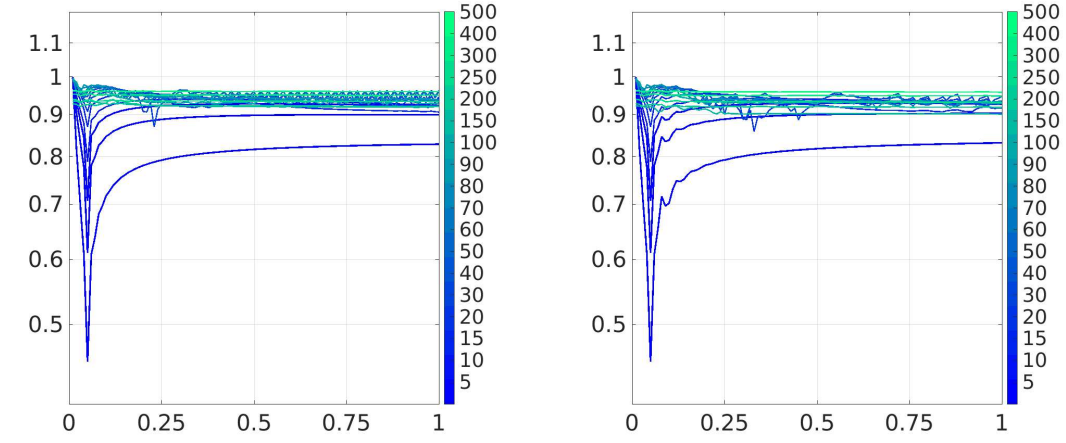
FGMRES

Table 4.28: Pressure L^2-error evaluated at endpoint $T = 1$ for the multigrid in time case for the FGMRES method without pre- and post-smoothening for q_1.



(2-grid) $\|p - p_h\|_{L^2}$ (3-grid)

Figure 4.54: Pressure L^2-error for FGMRES using 2-grid in time (left) and 3-grid in time (right). The x-axis represent the time interval [0,1] and the colorbar the number of performed FGMRES iterations.



(2-grid) (3-grid)

Figure 4.55: Overall convergence rate of the pressure for FGMRES using 2-grid in time (left) and 3-grid in time (right). The x-axis represent the time interval [0,1] and the colorbar the number of performed FGMRES iterations.

4.5 Approximate divergence-free velocity correction

A fast to compute divergence-free update based on the roots of PP-solver is essential, since the augmented Lagrangian method in section 4.3.5 requires too much computing time and yields very poor convergence rates.
In addition, especially for the incompressible Navier-Stokes equations in chapter 5, it is necessary to ensure a divergence-free velocity at each time step in order to guarantee convergence.

4.5.1 Divergence-free Laplace update

The first step in developing a fast update is to analyze the fast divergence-free update of the PP-algorithm. The PP divergence-free optimization is a simple matrix vector multiplication, which is defined as

$$\mathbf{u}^l = \tilde{\mathbf{u}}^l - k\mathbf{M}_l^{-1}\mathbf{B}\mathbf{q} \tag{4.10a}$$

$$\mathbf{u}^l = \tilde{\mathbf{u}}^l - k\mathbf{M}_l^{-1}\mathbf{B}\left(\mathbf{B}^\top\mathbf{M}_l^{-1}\mathbf{B}\right)^{-1}\left(\frac{1}{k}\mathbf{B}^\top\tilde{\mathbf{u}}\right) \tag{4.10b}$$

$$\mathbf{u}^l = \tilde{\mathbf{u}}^l - \mathbf{M}_l^{-1}\mathbf{B}\left(\mathbf{B}^\top\mathbf{M}_l^{-1}\mathbf{B}\right)^{-1}\mathbf{B}^\top\tilde{\mathbf{u}}. \tag{4.10c}$$

The high speed of the divergence-free update lies in the already computed (time-consuming) pressure Poisson vector $\mathbf{q}$. The arising divergence-defect $\mathbf{B}^\top\tilde{\mathbf{u}}$ is simply subtracted from the velocity vector. The update is scaled that the divergence error is completely eliminated:

$$\mathbf{M}_l^{-1}\mathbf{B}\mathbf{B}^{-1}\mathbf{M}_l\left(\mathbf{B}^\top\right)^{-1}\mathbf{B}^\top = \mathbf{M}_l^{-1}\mathbf{M}_l = \mathbf{I}. \tag{4.11}$$

For an exact divergence-free velocity $\mathbf{B}^\top\mathbf{u} = 0$, a zero term is subtracted in Eq. (4.10).

Realization for TSPP

As the TSPP-algorithm computes the (time-consuming) pressure Poisson problem after performing the divergence-free update, an alternative update must be found. Consider

$$\frac{1}{k}\mathbf{B}^\top\tilde{\mathbf{u}} =: \mathbf{d}, \tag{4.12}$$

as the divergence defect vector. Similar to the PP approach, the defect is scaled with $\frac{1}{k}$ to increase the small defect $\mathbf{B}^\top\tilde{\mathbf{u}} \equiv 0$ so that the solver performs more efficiently.
Multiplying Eq. (4.12) with the gradient matrix $\mathbf{B}$ leads to

$$\mathbf{B}\mathbf{B}^\top\tilde{\mathbf{u}} = \mathbf{B}\mathbf{d}, \tag{4.13}$$

which results in the difficult-to-solve problem $\mathbf{B}\mathbf{B}^\top$, respectively $\mathbf{B}\mathbf{M}_{pl}^{-1}\mathbf{B}^\top$ from section 4.3.
If Eq. (4.13) is solved exactly, the exact divergence-free solution is obtained. This will provide a full divergence-free velocity in the first (outer) iteration. Since several outer iterations can be performed, it is sufficient to fulfill equation (4.13) only approximately. Applying Eq. (3.43) leads to

$$(\mathbf{L} + \mathbf{R}\mathbf{R})\tilde{\mathbf{u}} = \mathbf{B}\mathbf{d} \tag{4.14a}$$

$$\mathbf{L}\tilde{\mathbf{u}} \approx \mathbf{B}\mathbf{d}, \tag{4.14b}$$

with $\mathbf{d}$ from Eq. (4.12) as the divergence defect vector. The rotation matrix $\mathbf{R}\mathbf{R}$ in Eq. (4.14b) is omitted to obtain a simple fast to compute Poisson problem in the velocity space.

Based on the PP approach, the update vector in the time-simultaneous case can be defined as

$$\mathbf{u}^l = \tilde{\mathbf{u}}^l - k\mathbf{L}^{-1}\left(\mathbf{B}\left(\frac{1}{k}\mathbf{B}^\top \tilde{\mathbf{u}}\right)\right). \qquad (4.15)$$

The factor k again scales the update vector in the correct order of magnitude.

Breakdown of the computing times

Compared to the pressure Poisson problem, the Laplace update in Eq. (4.15) requires only a fraction of the computing time.
Both Poisson problems are computed using the same solver: A PCG-multigrid in space solver using a damped Jacobi preconditioner as pre-smoother, post-smoother as well as for the coarse grid.

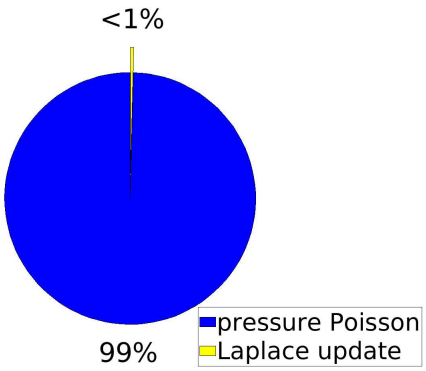


Figure 4.56: Ratio of the computation time required for the Laplace problem and the pressure Poisson update. This result applies to the test configuration $k = 0.01$, $K = 100$ and $\nu = 0.01$ for the function q_1.

non-divergence-free velocity solver									divergence-free velocity solver	
tol.	it.	$k = 0.01$ $K = 100$	%	$k = 0.001$ $K = 1000$	%	$k = 0.0001$ $K = 10000$	%		it.	tol.
	1	10.74 sec		29.34 sec		3.63 min				
		10.81 sec	+0.65	1.40 min	+186	11.34 min	+212		1	
	40	5.18 min		15.42 min		2.01 hr				
10^{-8}		2.49 min	-51.93						19	10^{-8}
				15.80 min	+2.46	2.21 hr	+9.95		14	
	51	6.55 min		22.07 min		2.49 hr				
10^{-10}		3.27 min	-50.08						26	10^{-10}
				23.76 min	+7.65	3.34 hr	+34.15		22	
	63	8.04 min		22.60 min		3.01 hr				
10^{-12}		4.02 min	-50.00						33	10^{-12}
				30.66 min	+35.66	4.33 hr	+43.85		29	
	74	9.44 min		26.02 min		3.49 hr				
10^{-14}		4.89 min	-48.20						41	10^{-14}
				40.56 min	+55.88				39	
						5.89 hr	+66.76		40	

Table 4.29: Time comparison between the non-divergence-free solver from algorithm 3.3 and the divergence-free solver based on the fast to compute Laplace update from Eq. (4.15) and by using 1 core in space and 16 cores in time.

An overview of the total computing time can be found in table 4.29. The computation in the divergence-free case is performed with the adjusted right-hand-side from Eq. (3.42) for the pressure Poisson problem.

In table 4.29, the non-divergence-free solver presents the same behavior as the PP in the introductory chapter 1, as shown in figure 1.12, figure 1.13 and figure 1.14. A smaller time step solves the velocity problem more rapidly, as the preconditioners for smaller time steps become more precise and therefore fewer iterations are required.

In the case of the divergence-free solver, it can be observed that a decrease in the time step size by a factor of 10 also increases the computation time by a factor of 10. The reason is that the Laplace update in Eq. (4.15) requires the same computing time regardless of the time step size. As discussed in chapter 1, the choice of $k = 0.01$ is too large, which means that the momentum

problem requires a lot of computing time. In this case, solving the Laplace update in Eq. (4.15) is faster than solving the momentum equations. With smaller time steps, the solving of the momentum equations requires only a fraction of the computation time, but due to the Laplace update, which stays constant in terms of computing time, it no longer gives a performance gain.

In contrast to the augmented Lagrangian method in table 4.23, the PP based Laplace divergence-free correction can actually save time. It is also noticeable in table 4.29 that only half of the (outer) iterations are required when a posterior divergence-free optimization is performed using the Laplace update in Eq. (4.15).

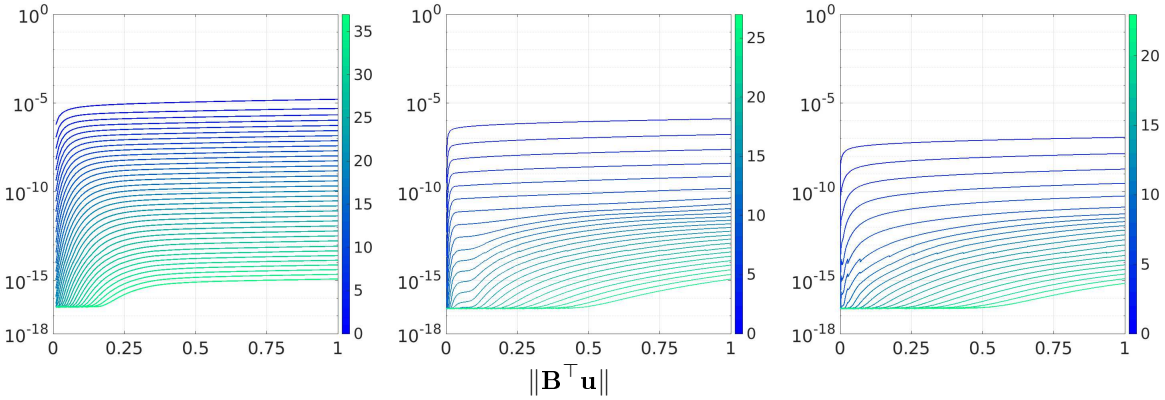


Figure 4.57: Divergence-defect curve for time step size $k = 0.01$, $K = 100$ (left), $k = 0.001$, $K = 1000$ (center) and $k = 0.0001$, $K = 10000$ using the Laplace update Eq. (4.15). The x-axis represent the time interval [0,1] and the colorbar the number of performed (outer) iterations.

The divergence-defect curve in figure 4.57 reveals the convergence properties.
First of all, it should be noted that a smaller time step size k leads to faster convergence, i.e. requires less outer iteration. Also, the start divergence-defect is lower for a smaller time step size. It is particularly noticeable with a shorter time step that a faster convergence takes place at the beginning and the convergence is slower for a higher accuracy. The reason is that only one approximation was used as a divergence-free update in Eq. (4.15).

4.5.2 Improved divergence-free Laplace update

To achieve faster convergence, a more accurate approximation must be used. Since the defect of the pressure Poisson problem, which is computed completely parallel-in-time and -space, is available from the second (outer) iteration, it can be considered whether the divergence-free update of Eq. (4.15) can be scaled appropriately to obtain a better approximation and thus a faster convergence.

i. Use the fast to compute Laplace problem as a starting divergence-free update during first (outer) iteration:

$$\mathbf{u}^{l} = \tilde{\mathbf{u}}^{l} - k\mathbf{L}^{-1}\left(\mathbf{B}\mathbf{d}\right). \tag{4.16}$$

ii. Use at the second (outer) iteration the already fully in space and time computed pressure Poisson results $\mathbf{q}$ as update improvement. Scale appropriate from the second and further (outer) iterations with the already calculated pressure Poisson defect:

$$\mathbf{u}^{l} = \tilde{\mathbf{u}}^{l} - \gamma_L k\mathbf{L}^{-1}\left(\mathbf{B}\mathbf{d}\right) + \gamma_P\left(k\mathbf{M}_l^{-1}\mathbf{B}\mathbf{q}\right), \tag{4.17}$$

with γ_L and γ_P as scaling parameters.

The following tests are performed with the adjusted right-hand-side of Eq. (3.42).

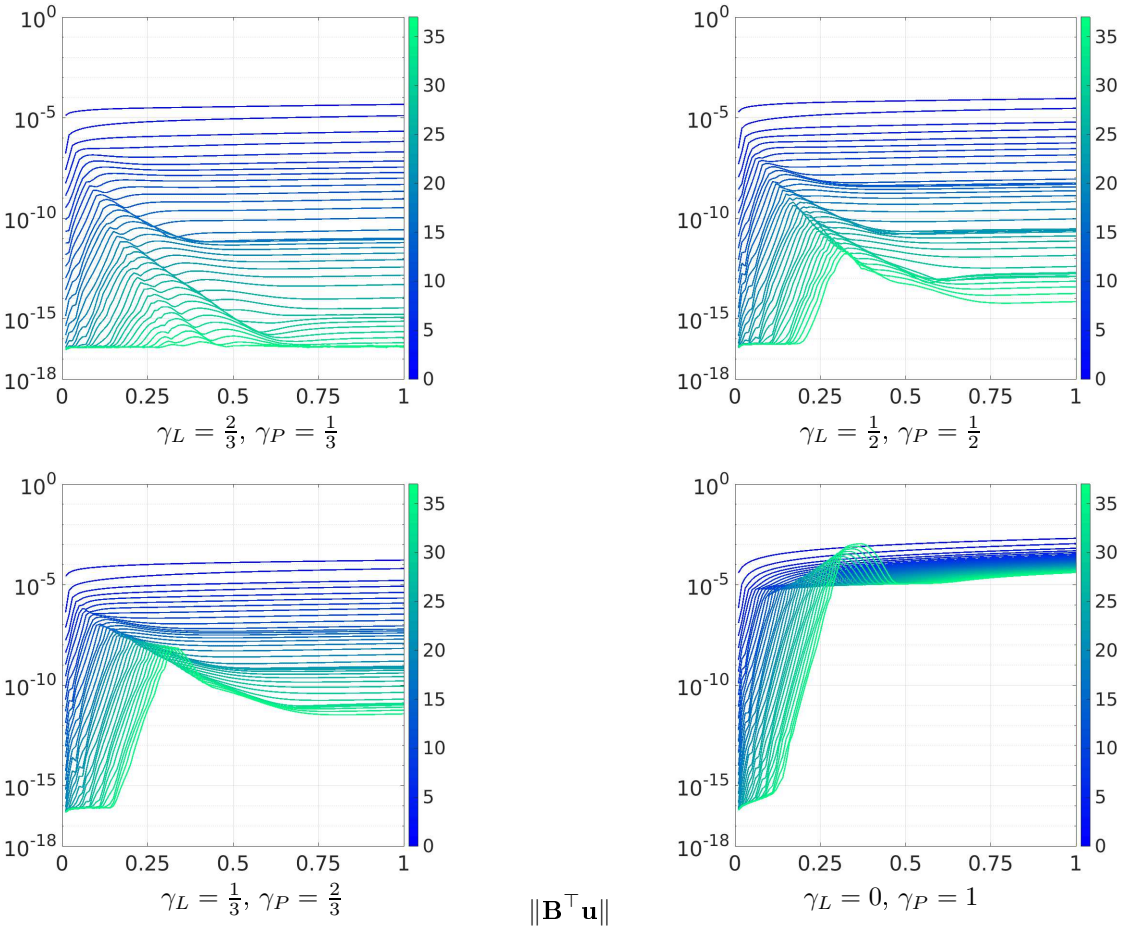


Figure 4.58: Divergence-defect curve of Eq. (4.17) with different weighting of the pressure Poisson update: $\gamma_L = \frac{2}{3}$, $\gamma_P = \frac{1}{3}$ (top, left), $\gamma_L = \gamma_P \frac{1}{2}$ (top, right), $\gamma_L = \frac{1}{3}$, $\gamma_P = \frac{2}{3}$ (bottom, left) and $\gamma_L = 0$, $\gamma_P = 1$ (bottom, right). The x-axis represent the time interval [0,1].

Figure 4.58 shows the results of the improved divergence-free update Eq. (4.17). In the case $\gamma_L = 0$, $\gamma_P = 1$, the Laplace was used as an update during the first (outer) iteration as described in Eq. (4.16).

The optimal scaling seems to be the choice of $\gamma_L = \frac{2}{3}$ and $\gamma_P = \frac{1}{3}$. Figure 4.59 shows the divergence-defect for smaller time step sizes k. It can be seen that the convergence boost is also preserved for smaller time steps.

According to table 4.30, about 5 (outer) iterations can be saved if the pressure Poisson results are used in the update Eq. (4.17). This proves that faster convergence can be achieved with the divergence-free update for the solver even with smaller time steps.
To achieve high accuracy with a small time step size, the divergence-free solver requires more time than the non-divergence-free solver. However, it should be noted that the momentum equations has so far only been computed sequential-in-time. By using a time parallelization from [49] or by applying parareal from [48], a large acceleration boost can be expected.

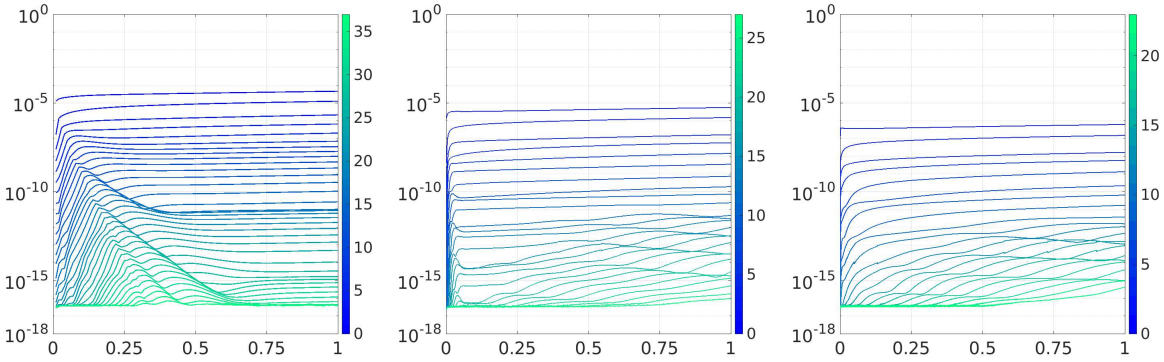


Figure 4.59: Divergence-defect of Eq. (4.17) for different time step size: $k = 0.01$, $K = 100$ (left) $k = 0.001$, $K = 1000$ (center) and $k = 0.0001$, $K = 10000$ (right). The x-axis represent the time interval [0,1].

non-divergence-free velocity solver								**divergence-free velocity solver**	
tol.	it.	$k = 0.01$ $K = 100$	%	$k = 0.001$ $K = 1000$	%	$k = 0.0001$ $K = 10000$	%	it.	tol.
	1	10.74 sec		29.34 sec		3.63 min			
		10.90 sec	+1.49	1.40 min	+186	11.36 min	+213	1	
	40	5.18 min		15.42 min		2.01 hr			
10^{-8}		2.04 min	-60.62					15	10^{-8}
				13.70 min	-11.15			12	
						1.78 hr	-11.44	11	
	51	6.55 min		22.07 min		2.49 hr			
10^{-10}		2.74 min	-58.17					21	10^{-10}
				19.66 min	-10.92	2.78 hr	+11.65	18	
	63	8.04 min		22.60 min		3.01 hr			
10^{-12}		3.40 min	-57.71					27	10^{-12}
				24.58 min	+8.76			23	
						3.77 hr	+25.25	25	
	74	9.44 min		26.02 min		3.49 hr			
10^{-14}		4.17 min	-55.82					34	10^{-14}
				34.41 min	+32.24			33	
						5.19 hr	+48.71	35	

Table 4.30: Time comparison between the non-divergence-free solver from algorithm 3.3 and the divergence-free solver based on the improved update from Eq. (4.17) using $\gamma_L = \frac{2}{3}$ and $\gamma_P = \frac{1}{3}$ for 1 core in space and 16 cores in time.

The following table 4.31 shows the (outer) iteration numbers in detail in the case of $\gamma_L = \frac{2}{3}$ and $\gamma_P = \frac{1}{3}$. Compared to the non-divergence-free solver from algorithm 3.3, more than half of the (outer) iterations can be saved to achieve the same accuracy. Only the pressure defect requires for a small tolerance (red) slightly more (outer) iterations. For the smallest tested time step size $k = 0.0001$, it can be seen that in the range of machine precision one more (outer) iteration is needed for the pressure and velocity error. A convergence boost for smaller time step sizes is therefore not achieved.

	Residual											
	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}
$k = 0.01$	1 (0)	1 (0)	1 (0)	2 (-1)	4 (-1)	5 (-8)	7 (-13)	10 (-16)	12 (-21)	14 (-25)	19 (-26)	21 (-30)
$k = 0.001$	1 (0)	1 (0)	1 (0)	1 (-2)	2 (-3)	4 (-9)	5 (-15)	7 (-19)	8 (-25)	10 (-29)	13 (-32)	16 (-35)
$k = 0.0001$	1 (0)	1 (0)	1 (0)	1 (-2)	1 (-4)	2 (-11)	4 (-16)	5 (-21)	7 (-26)	8 (-31)	10 (-35)	12 (-39)
	Divergence-defect											
	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}
$k = 0.01$	1 (-1)	1 (-3)	3 (-7)	4 (-12)	6 (-17)	9 (-20)	12 (-23)	15 (-26)	17 (-30)	21 (-32)	24 (-35)	26 (-38)
$k = 0.001$	1 (-1)	1 (-3)	1 (-9)	3 (-13)	4 (-19)	6 (-23)	7 (-28)	9 (-32)	11 (-36)	14 (-39)	18 (-41)	21 (-43)
$k = 0.0001$	1 (-1)	1 (-3)	1 (-9)	1 (-15)	3 (-20)	4 (-25)	6 (-29)	7 (-34)	9 (-38)	12 (-41)	15 (-44)	19 (-45)
	$\|p - p_h\|_{L^2}$											
	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}
$k = 0.01$	4 (2)	6 (2)	7 (-3)	9 (-7)	13 (-10)	15 (-14)	17 (-18)	21 (-20)	24 (-23)	27 (-26)	29 (-30)	34 (-30)
$k = 0.001$	4 (2)	6 (2)	7 (-3)	9 (-7)	10 (-13)	11 (-18)	13 (-22)	16 (-25)	19 (-28)	22 (-31)	28 (-31)	33 (-31)
$k = 0.0001$	4 (2)	6 (2)	7 (-3)	9 (-7)	9 (-14)	11 (-18)	14 (-21)	17 (-24)	21 (-26)	25 (-28)	30 (-29)	35 (-29)
	$\|u - u_h\|_{L^2}$											
	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}
$k = 0.01$	2 (-1)	4 (-1)	6 (-4)	7 (-9)	8 (-14)	10 (-18)	14 (-20)	16 (-24)	18 (-28)	22 (-30)	25 (-32)	27 (-36)
$k = 0.001$	3 (0)	4 (-1)	4 (-6)	7 (-9)	8 (-14)	9 (-19)	11 (-23)	15 (-25)	16 (-30)	19 (-33)	22 (-35)	25 (-38)
$k = 0.0001$	3 (0)	4 (-1)	4 (-6)	7 (-9)	8 (-14)	9 (-19)	11 (-23)	14 (-26)	18 (-28)	21 (-31)	23 (-34)	26 (-37)
	$\|u - u_h\|_{H^1}$											
	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}
$k = 0.01$	4 (-4)	6 (-8)	7 (-14)	9 (-18)	13 (-21)	15 (-25)	17 (-29)	21 (-30)	24 (-33)	26 (-37)	28 (-40)	33 (-41)
$k = 0.001$	4 (-4)	6 (-8)	7 (-14)	8 (-19)	10 (-24)	12 (-28)	15 (-31)	18 (-33)	21 (-36)	23 (-40)	27 (-41)	32 (-42)
$k = 0.0001$	4 (-4)	6 (-8)	7 (-14)	8 (-19)	10 (-24)	11 (-29)	14 (-32)	18 (-33)	21 (-36)	24 (-39)	28 (-40)	33 (-41)

Table 4.31: Number of (outer) iterations for the residual, divergence-defect, pressure L^2 error, the velocity L^2 error and the H^1 error with a corresponding increase (red, in brackets) and decrease (green, brackets) at the endpoint $T = 1$ for $q_1(t)$.

Conclusion of the divergence-free update Eq. (4.17):

i. The initial divergence-defect decreases and remains small for smaller time step sizes k.

ii. A smaller time step size means fewer (outer) iterations are required.

iii. Faster convergence during the first (outer) iterations but slower towards machine precision

→ The key advantage of this approach, which is analogous to the PP, is that it is highly efficient for providing an approximate solution.

iv. Fast convergence already even without recoupled multigrid in time approach.

The recoupled multigrid in time correction is possible, but too time-consuming, since the right-hand-side $\mathbf{B}^\top \mathbf{u}$ must be computed and GMRES requires a minimum number of iterations. The local coupled approach, presented in section 3.3.4 and tested in section 4.4 deceleration rather than acceleration the process. According to [51], the global coupled system will probably accelerate the process, which could not be verified here.

4.5.3 Properties of a full non-divergence-free right-hand-side

For completeness, a test with a (not recommended) full non-divergence-free right-hand-side

$$\mathbf{f}_p = \frac{1}{k}\mathbf{B}^\top \tilde{\mathbf{u}}^l - \frac{1}{k}\mathbf{B}^\top \tilde{\mathbf{u}}^{l-1} \tag{4.18}$$

for the pressure Poisson problem is performed in this section.
Figure 4.60 presents the divergence-free curve of Eq. (4.18) for the divergence-free update in Eq. (4.15).

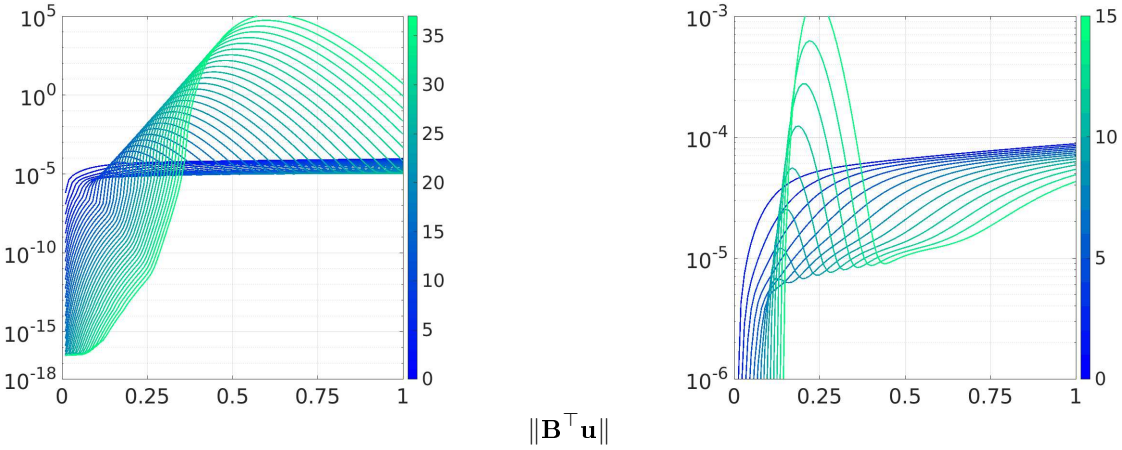


Figure 4.60: Divergence curve with simple divergence-free correction from Eq. (4.15) using Eq. (4.18) as right-hand-side for the pressure Poisson problem. The x-axis represent the time interval [0,1].

Figure 4.60 shows that the first 7 (outer) iterations behave normal, but converge with a really slow rate. Starting from the 8th (outer) iteration, an oscillation occurs, which increases more and more with each (outer) iteration, but still leads to convergence.

Figure 4.61 shows the use of the improved divergence-free update from Eq. (4.17) from section 4.5.2. It can be seen that a slight scaling with $\gamma_P = \frac{1}{3}$ of the already computed pressure Poisson problem stabilizes the divergence-defect curve. Too much scaling with $\gamma_P \geq \frac{1}{2}$ again leads to oscillations. Moreover, the solution become less accurate in each further (outer) iteration if only the pure pressure Poisson update with $\gamma_P = 1$ is used.

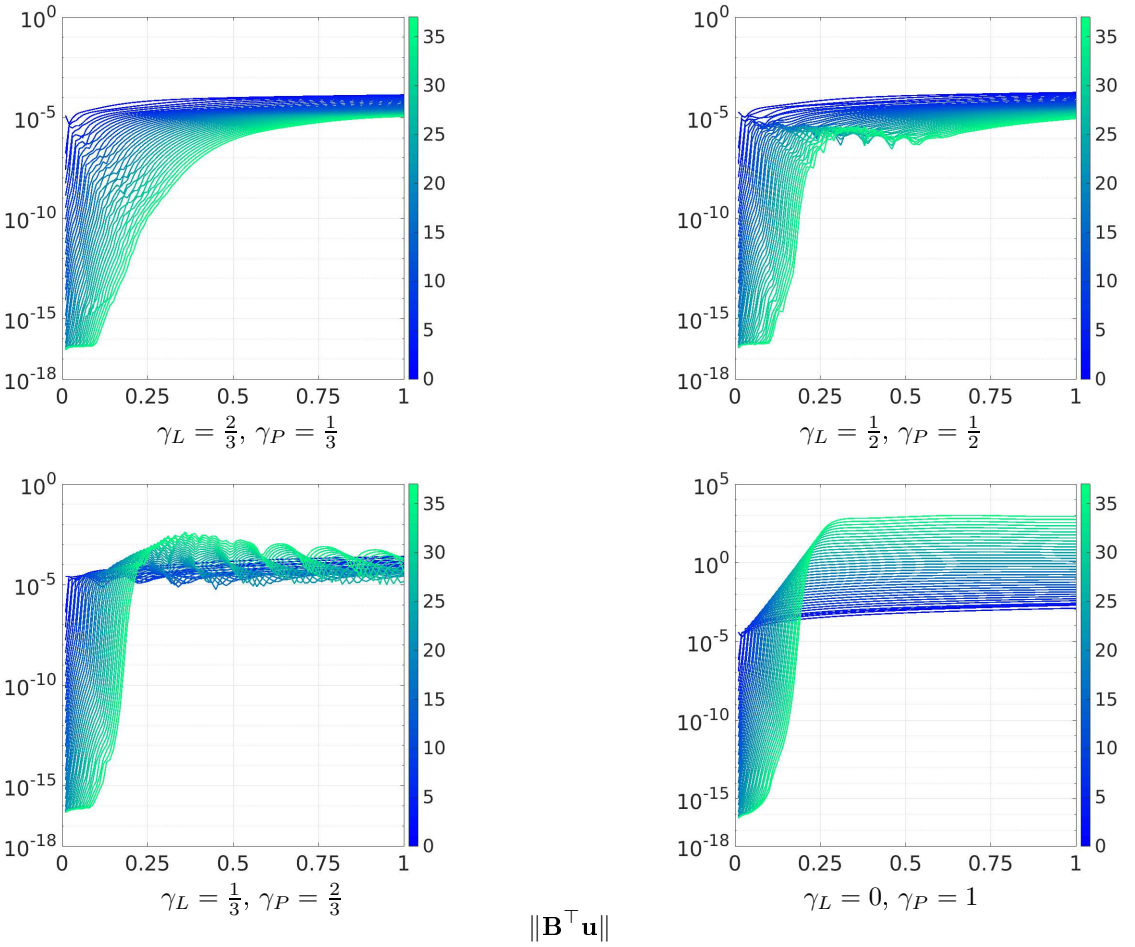


Figure 4.61: Divergence-defect curve of Eq. (4.17) using the right-hand-side Eq. (4.18) with different weighting of the pressure Poisson update: $\gamma_L = \frac{2}{3}$, $\gamma_P = \frac{1}{3}$ (top, left), $\gamma_L = \gamma_P \frac{1}{2}$ (top, right), $\gamma_L = \frac{1}{3}$, $\gamma_P = \frac{2}{3}$ (bottom, left) and $\gamma_L = 0$, $\gamma_P = 1$ (bottom, right). The x-axis represent the time interval [0,1].

Using the recoupling approach with GMRES, analyzed in section 4.4, would make sense in this case, since the convergence rate per (outer) iteration is very low. However, instead of using the coupled multigrid in time method, it is more advantageous to use the correct right-hand-side from Eq. (3.42) for the pressure Poisson problem.

In addition, there are alternative local approaches for a divergence-free velocity in [8], [9].

4.6 Evaluations concerning the Q_2/Q_1 element

This section examines the convergence behavior of the Q_2/Q_1 element. As mentioned in the outline of section 1.2, a fast Poisson solver using the Q_1 element is being developed in [63]. To use this fast solver in the Navier-Stokes context, the convergence properties concerning the Q_1 pressure matrix must first be checked.

4.6.1 Artificial pressure Poisson matrix $\mathbf{B}^\top\mathbf{M}_l^{-1}\mathbf{B}$

First, the Q_2/Q_1 element is tested using the preconditioned matrix

$$\mathbf{P} := \mathbf{B}^\top\mathbf{M}_l^{-1}\mathbf{B} \tag{4.19}$$

for the pressure Poisson problem using the basic algorithm 3.3 with $\alpha_R = 1$ and $\alpha_D = \theta\nu k$, where no posterior divergence-free optimization is performed, i.e. $\gamma = 0$ (also $\gamma_L = 0 and \gamma_P = 0$). Figure 4.62 shows the divergence-defect curve of the sample problem (4.1) for the time step size $k = 0.01$ with $K = 100$.

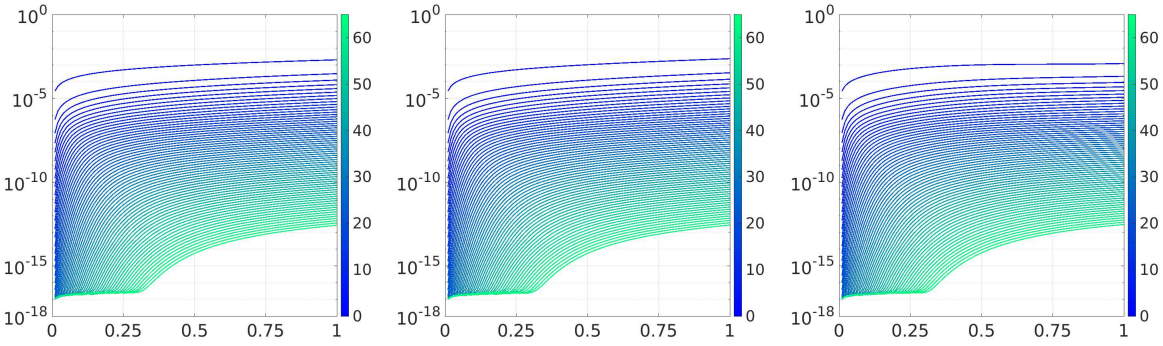


Figure 4.62: Divergence-defect using the Q_2/Q_1 element with Eq. (4.19) for $q_1(t) := 1 + t$ (left), $q_2(t) := e^t$ (center) and $q_3(t) := 1 + \frac{\sin(2\pi t)}{2}$ (right) over the time interval [0,1].

The divergence curve and also the iteration numbers and convergence behavior are similar to that of the Q_2/P_1^{disc} element in section 4.1. Although the computation is based on a quadratic element, no spurious mods and oscillations are noticeable in figure 4.62 in case of the pressure Poisson matrix using the preconditioner from Eq. (3.17).

4.6.2 True pressure Poisson matrix L

The fast Poisson solver from [63] only works with the true Poisson matrix

$$\mathbf{P} := \mathbf{L}. \tag{4.20}$$

Using the same setup from section 4.6.1 leads at a first view to a similar convergence behavior, as shown for the first 10 (outer) iterations in figure 4.63.

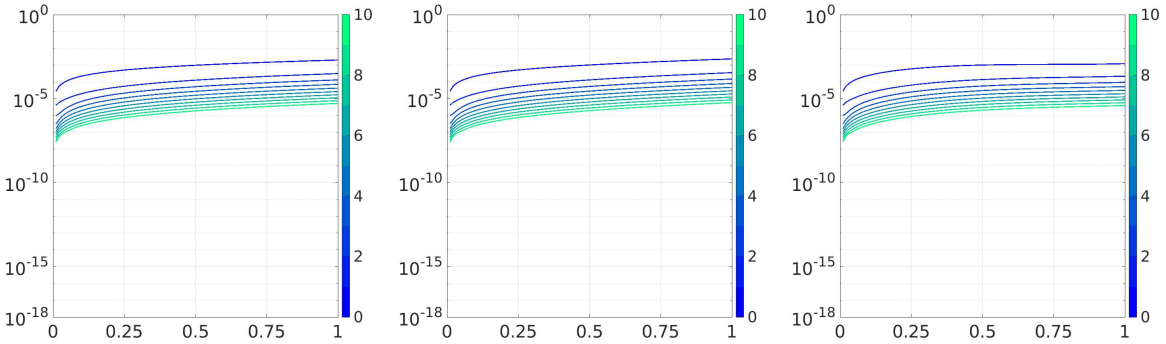


Figure 4.63: Divergence-defect using the Q_2/Q_1 element with Eq. (4.20) for the first 10 (outer) iterations for $q_1(t) := 1+t$ (left), $q_2(t) := e^t$ (center) and $q_3(t) := 1 + \frac{\sin(2\pi t)}{2}$ (right) over the time interval [0,1].

Spurious modes

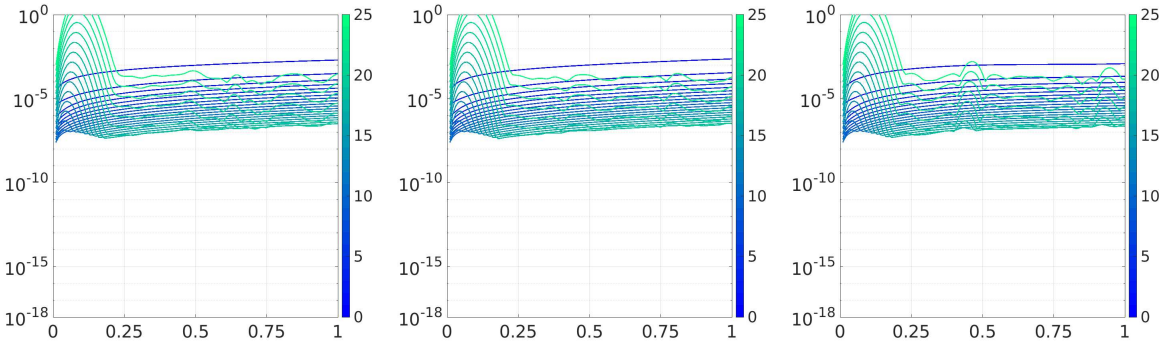


Figure 4.64: Divergence-defect using the Q_2/Q_1 element with Eq. (4.20) for the first 25 (outer) iterations for $q_1(t) := 1+t$ (left), $q_2(t) := e^t$ (center) and $q_3(t) := 1 + \frac{\sin(2\pi t)}{2}$ (right) over the time interval [0,1].

Figure 4.64 shows that slight oscillations begin to occur from the 11th (outer) iteration onwards. Figure 4.65 shows very clearly that each additional (outer) iteration causes a larger defect and there is no convergence any more.

The problem is that the reactive preconditioner $\mathbf{M}_l^{-1}$ from Eq. (3.18) is not available. In order to achieve convergence, it is necessary to embed the coupled system from section 4.4 in the solver. After each (outer) iteration, sufficient GMRES iterations must be performed to ensure convergence.

Note that no divergence-free optimization has been performed so far using $\gamma = 0$ and $\gamma_L = \gamma_P = 0$. The next sections will investigate whether a divergence-free velocity is sufficient to avoid the coupled system with GMRES.

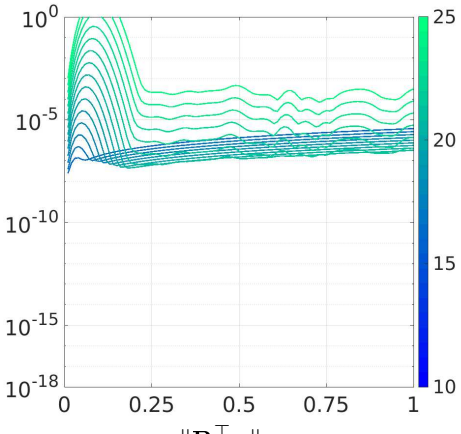


Figure 4.65: Displaying the 10th to 25th (outer) iteration of the divergence-defect using the Q_2/Q_1 element with Eq. (4.20) for $q_1(t)$. The x-axis represent the time interval [0,1].

4.6.3 Augmented Lagrangian for the true pressure Poisson matrix

Since the real Laplace matrix causes oscillations, the next step is to check if a kind of grad-div stabilization, the augmented Lagrangian tested in section 4.3, is sufficient to achieve convergence again.

A special aspect compared to the Q_2/P_1^{disc} element is that in the case of the Q_2/Q_1 element a DOF is shared by the gradient and divergence matrix, as shown in figure 4.66. This leads to an increase of the matrix stencil. In [50], an attempt was made to solve this problem efficiently with the help of the coupled system using GMRES.

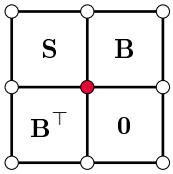


Figure 4.66: Illustration of the shared DOF (red) in the case of the gradient and divergence matrix for $\mathbf{B}\mathbf{M}_{pl}^{-1}\mathbf{B}^\top$.

The results in figure 4.67 and figure 4.68 are made without the embedding of the coupled system. Eq. (3.42) was chosen as the right-hand-side for the pressure Poisson problem.
Figure 4.67 shows that the first 3 (outer) iterations run without problems. Figure 4.68 shows that the divergence-defect deteriorates from the 4th outer iteration onwards. To counteract this problem, sufficient GMRES iterations must be performed at each (outer) iteration with the embedded coupled system.

Using the augmented Lagrangian method alone is not sufficient for convergence.

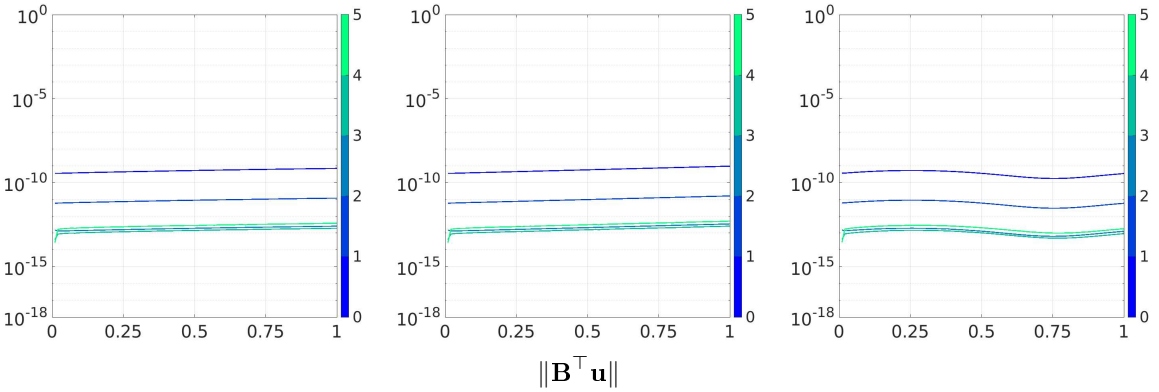


Figure 4.67: Divergence-defect curve for the first 5 (outer) iterations using Eq. (4.20) with augmented Lagrangian for q_1 (left), q_2 (center) and q_3 (right) over the time interval [0,1].

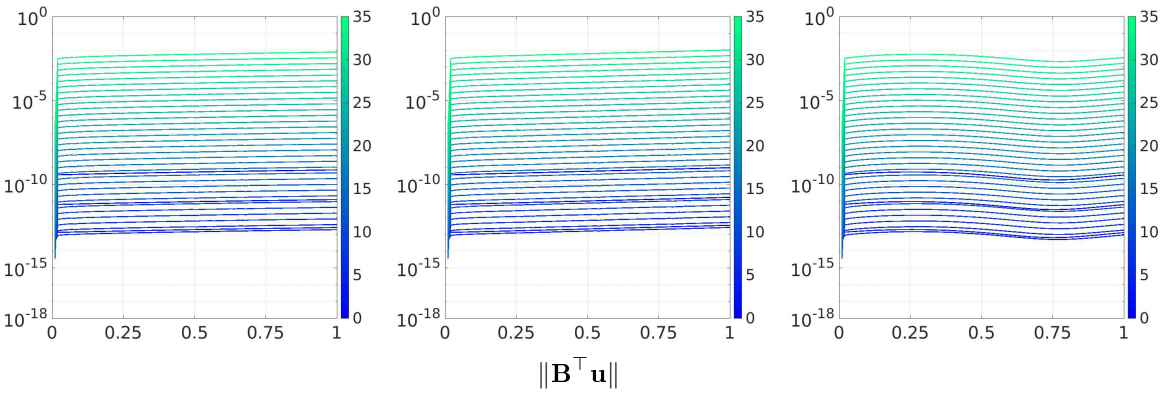


Figure 4.68: Divergence-defect curve for the first 35 (outer) iterations using Eq. (4.20) with augmented Lagrangian for q_1 (left), q_2 (center) and q_3 (right) over the time interval [0,1].

4.6.4 Improved divergence-free Laplace update for the true pressure Poisson matrix

Since the computing time of the augmented Lagrangian method in case of the Q_2/Q_1 element is similar in cost compared to table 4.23, the fast update from section 4.5 is now evaluated. Like before, Eq. (3.42) is used for the right-hand-side of the pressure Poisson problem.

Figure 4.69 presents the convergence behavior for the fast update Eq. (4.15). Oscillations occur after just a few (outer) iterations.

In Figure 4.70, the improved divergence-free update from Eq. (4.17) is tested by setting $\gamma_L := \frac{2}{3}$ and $\gamma_P := \frac{1}{3}$. Oscillations also occur here after a few (outer) iterations, but they are less intense. A better approximation must therefore be found which guarantees a sufficient divergence-free velocity at each time step. The conclusion remains the same as before:

In case of the Laplace matrix for the pressure Poisson problem, an embedding in the coupled system using sufficient GMRES iterations is necessary to ensure convergence.

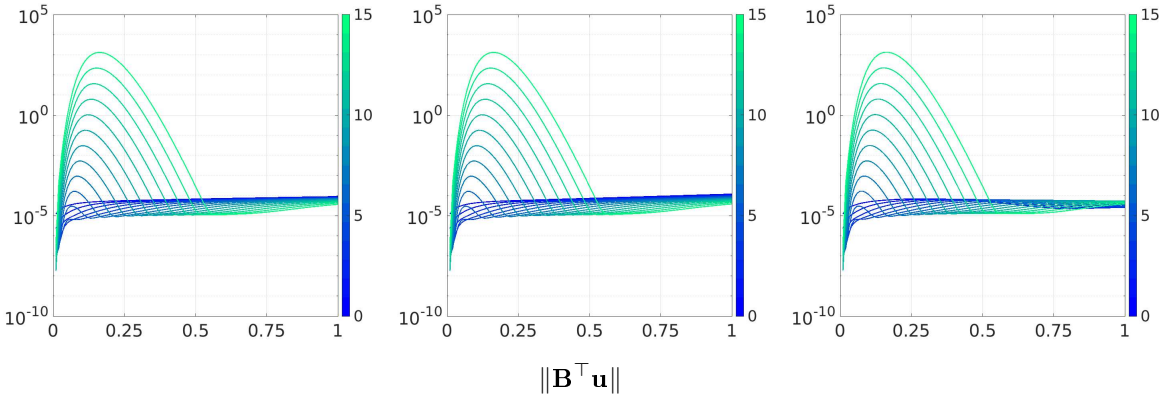


Figure 4.69: Divergence-defect using Eq. (4.20) with the fast update Eq. (4.15) for q_1 (left), q_2 (center) and q_3 (right) over the time interval [0,1].

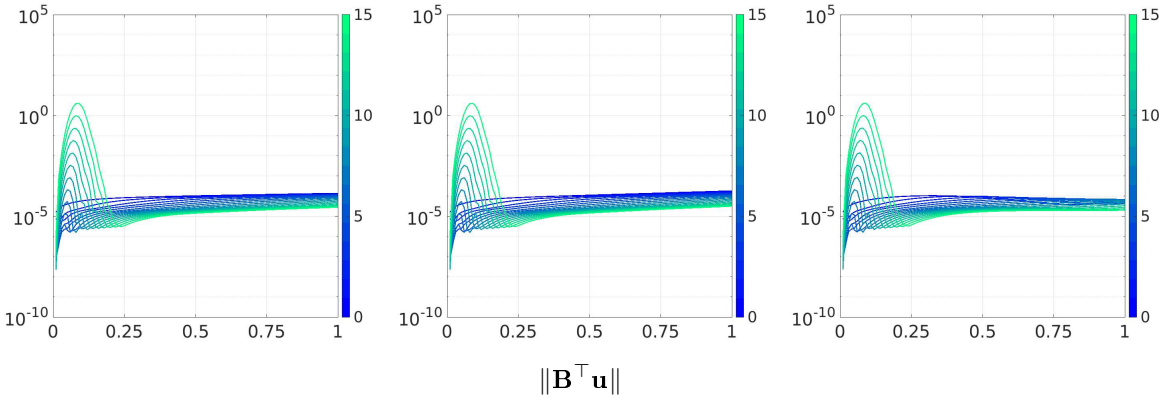


Figure 4.70: Divergence-defect using Eq. (4.20) with the improved divergence-free update from Eq. (4.17) with $\gamma_L = \frac{2}{3}$ and $\gamma_P = \frac{1}{3}$ for q_1 (left), q_2 (center) and q_3 (right) over the time interval [0,1].

Chapter 5

Numerical simulations for the flow-around-a-cylinder benchmark

Contents

Based on the knowledge from the last chapter 4, section 5.1 presents the settings for a parallel-in-space and -time algorithm that is capable for a fast convergence of the convective term in the incompressible Navier-Stokes equations.

In comparison to the previous results, this chapter focuses on the solution of the Navier-Stokes equations on the flow-around-a-cylinder grid. In section 5.2 results for a stationary benchmark are presented. Parallel performance in space and time is examined for the three dimensional case in section 5.3. This involves computing of fluid flow simulations on a complex grid in long-term computations with many thousands of time steps parallel-in-space and -time.

5.1 HPC algorithm

On the basis of the previous section 4.6, the Q_2/P_1^{disc} element is chosen with the artificial pressure Poisson matrix from Eq. (3.18) for the Navier Stokes equations in order to avoid additional issues involving oscillations. Furthermore, the Crank-Nicolson method, as mentioned in section 2.3.4, with $\theta = 0.5$ is used as the time stepping scheme.

To reduce the divergence-defect in the pressure update, the configurations $\alpha_R = 1$ and $\alpha_D = \theta k \nu$ are set, which, as outlined in section 4.1.2, results in accelerated convergence. In order to guarantee a sufficient divergence-free velocity and thus convergence for the Burgers equations at each time step, it follows that the fast PP-like update from Eq. (4.17) is the preferred option. As observed in section 4.5, the setting $\gamma_L = \frac{2}{3}$ and $\gamma_P = \frac{1}{3}$ is selected for enhanced convergence.

The main aim stays to develop a fast PP-like algorithm that exhibits naturally occurring convergence without recoupling. Accordingly, the recoupling of the solver in a multigrid in time using (F)GMRES, as described in section 4.4, is not pursued. The reasons are:

i. An (unknown) minimum number of iterations must be performed by GMRES.

ii. The local approach presented in section 4.4 has a poor convergence rate is too time-consuming. Skipping the (F)GMRES part is faster overall.

   The global multigrid in time approach using GMRES from [51] may turn out to be more advantageous, but at the cost of increased memory consumption, as illustrated in section 3.1.3. This is because a global matrix must be assembled for all time steps.

Figure 5.1 visualize the structure of a time-simultaneously PP-solver that is suitable for the convective term in the Navier-Stokes equations. An embedding of the solver in a multigrid in time is a beneficial approach, as it will be illustrated in the next section 5.2.

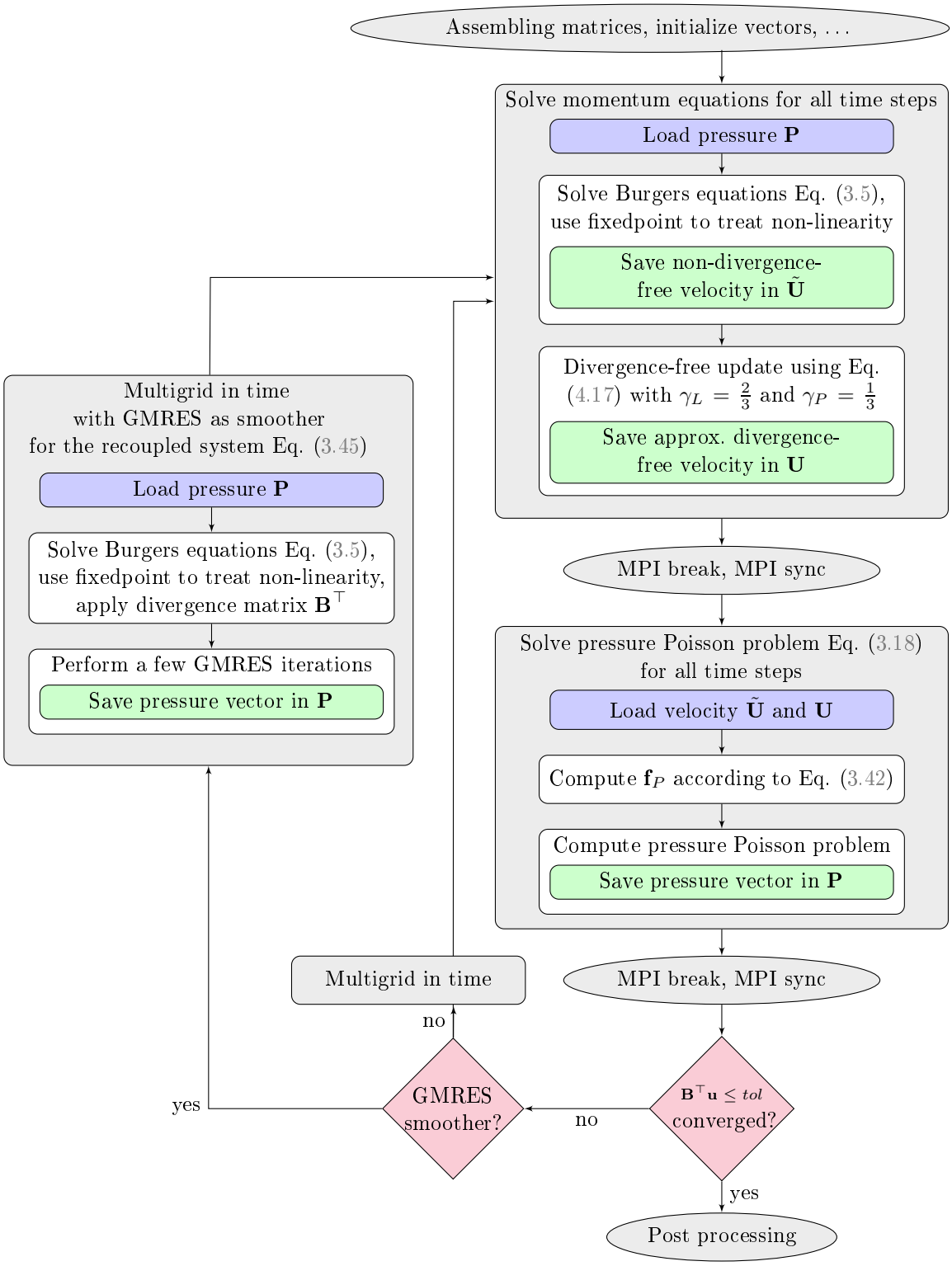


Figure 5.1: Recommended structure for the time-simultaneous PP-solver in the case of a convective term as in the Navier-Stokes equations.

5.2 Benchmark - flow-around-a-cylinder

The flow-around-a-cylinder benchmark uses the incompressible Navier-Stokes equations from definition 2.1 with $\nu := 0.001$. This benchmark is widely used, where numerous research articles providing reference results. Most notably references are [7], [12], [36], [42], [44] and [69].

Figure 2.3 illustrates the chosen discretization of the domain. A characteristic of the flow-around-a-cylinder grid is that the cylinder has its orientation not in the center but shifted slightly downwards. Figure 5.2 shows the grid dimensions for the two dimensional illustration. The domain is defined by

$$\Omega = [0, 2.2] \times [0, 0.41] \setminus B_r(0.2, 0.2), \tag{5.1}$$

with B_r as the cylinder with the radius $r = 0.05$ at the coordinates $[0.2, 0.2]$.

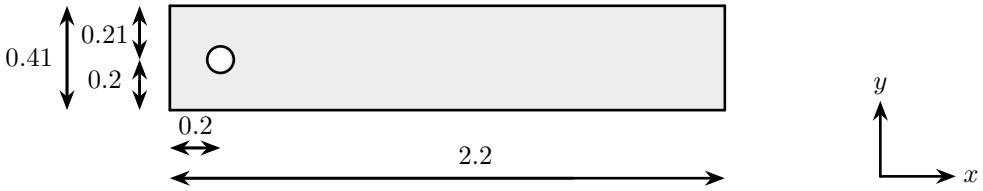


Figure 5.2: Dimensions of the flow-around-a-cylinder grid for two dimensional case. The measurements are in meters. The coordinate system is shown on the right for completeness.

The associated boundary conditions are described in section 2.4.
Considering the stationary benchmark with a parabolic inflow condition

$$u_{\text{in}} := u(0, y) = \left[\frac{1.2y(0.41-y)}{0.41^2}, \quad 0\right]^\top, \tag{5.2}$$

results in a low Reynolds number of $Re = 20$, which leads to stationary velocity and pressure solution. The system uses zero as the initial value for the velocity and pressure, which presents a significant challenge for the time-simultaneous PP-solver.
The space- and time-parallel algorithm is not actually designed for stationary problems. To compute this stationary benchmark, computations are therefore performed up to an end point from which no more changes in the solution can be observed. The endpoint is set at $T = 10$.

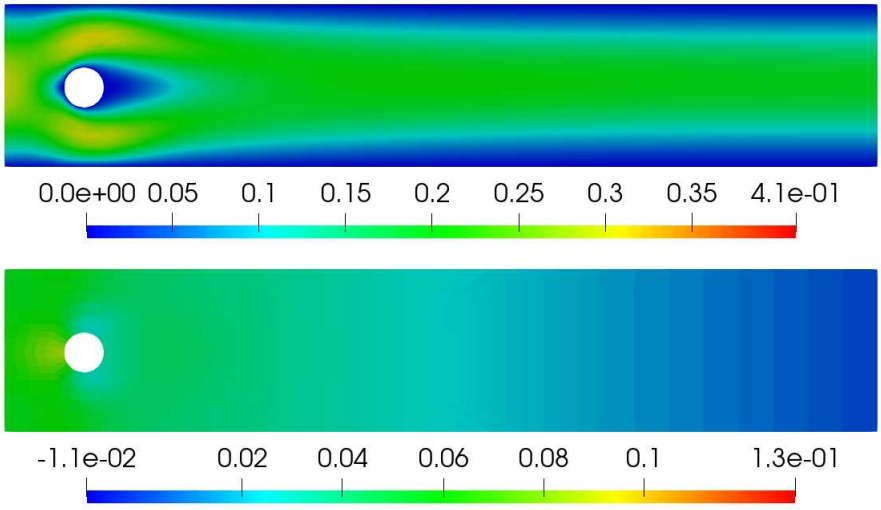


Figure 5.3: Velocity (top) and pressure (bottom) profile on level 2 for the 1st (outer) iteration evaluated at time $t = 10$. The divergence-defect $\|\mathbf{B}^\top \mathbf{u}\|$ is less than 10^{-2}.

Figure 5.3 shows the velocity and pressure curve for the first (outer) iteration. It can be seen that the two profiles are not developed. As a consequence of the absence of a divergence-free velocity, a solution that is considerable apart from the reference solution has been generated.

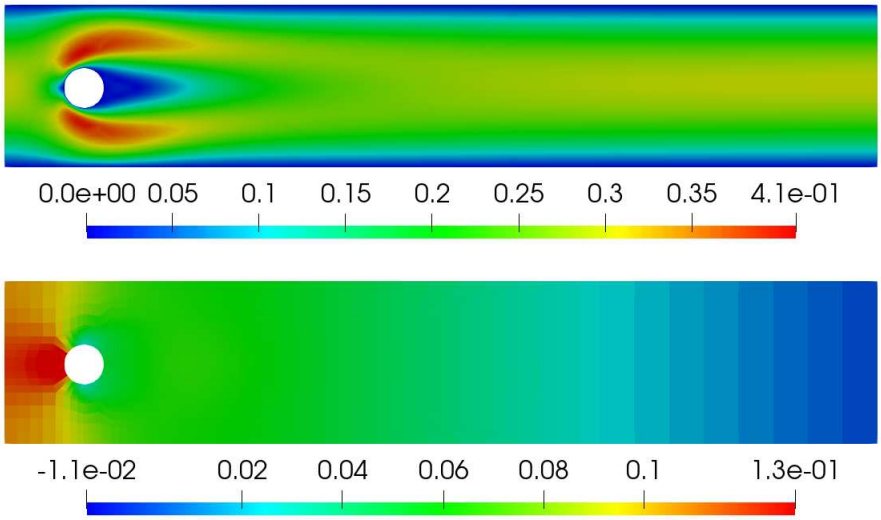


Figure 5.4: Velocity (top) and pressure (bottom) profile on level 2 for the 15. (outer) iteration evaluated at time $t = 10$. The divergence-defect $\|\mathbf{B}^\top \mathbf{u}\|$ is around 10^{-5}.

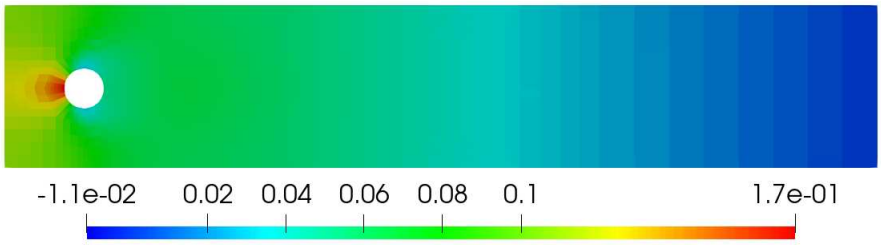


Figure 5.5: Pressure profile from figure 5.4 for scaling the color map outside of the reference color map.

In figure 5.4, after 15 (outer) iterations, the velocity profile already looks similar to that of the reference solution. However, comparing with the reference solution in figure 5.6, the pressure profile has an incorrect pattern, especially around the cylinder, which is clearly shown in figure 5.5. Divergence-defect in the range of 10^{-5} is therefore not sufficient to guarantee a good pressure approximation.

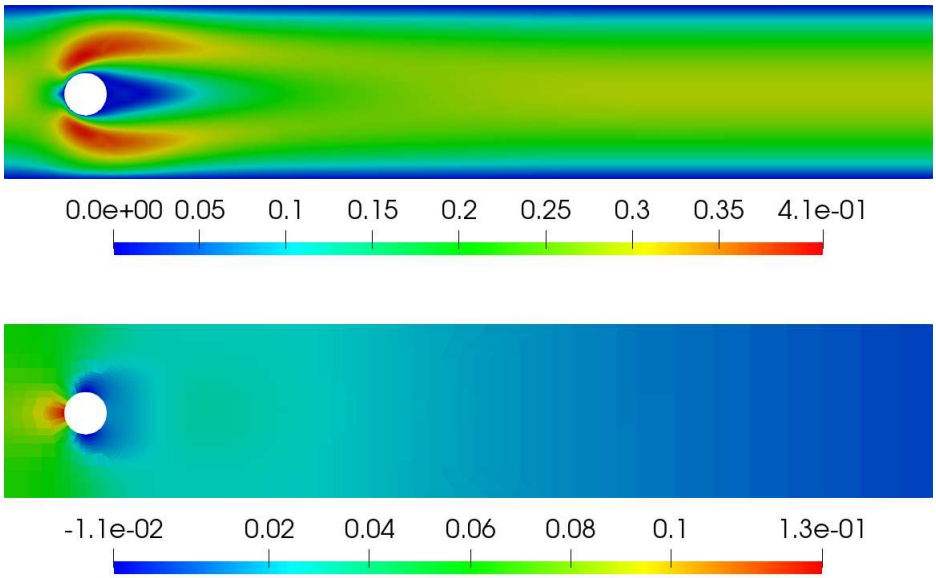


Figure 5.6: Velocity (top) and reference pressure (bottom) profile on level 2 evaluated at time $t = 10$. The divergence-defect $\|\mathbf{B}^\top \mathbf{u}\|$ is at machine precision.

In the following figures, from figure 5.7 to figure 5.9, the drag, lift and pressure difference values are evaluated at time $t = 10$ and displayed for different time step sizes and levels. This is done in order to determine the number of (outer) iterations that are necessary to obtain a good approximation. The number of (outer) iterations is indicated on the x-axis.

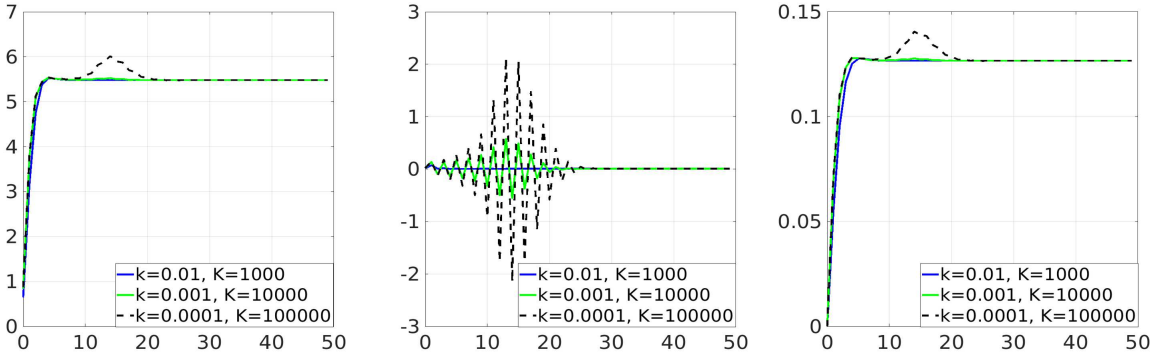


Figure 5.7: Drag (left), lift (center) and pressure difference (right) for the stationary flow-around-a-cylinder benchmark on the coarse grid level 0 using various time step sizes $k \in \{0.01, 0.001, 0.0001\}$ with $K \in \{1000, 10000, 100000\}$ problems computed parallel-in-time (and -space). The values were evaluated at time $t = 10$. The x-axis represents the number of (outer) iterations.

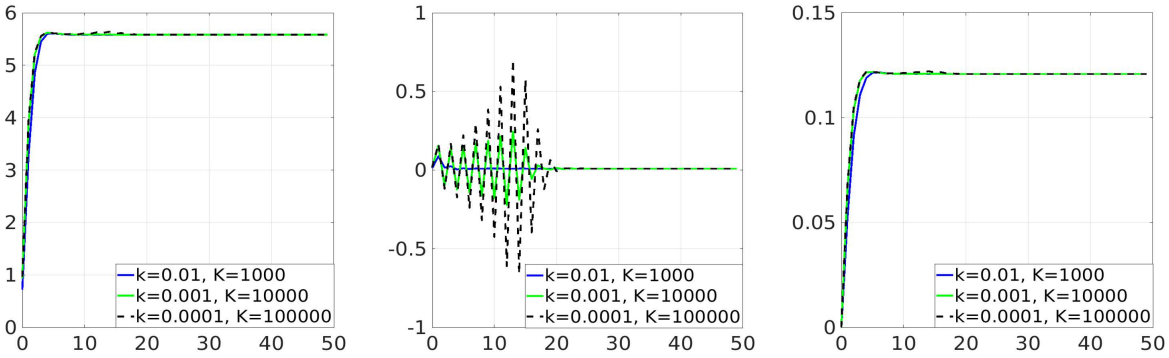


Figure 5.8: Drag (left), lift (center) and pressure difference (right) for the stationary flow-around-a-cylinder benchmark on level 1 using various time step sizes $k \in \{0.01, 0.001, 0.0001\}$ with $K \in \{1000, 10000, 100000\}$ problems computed parallel-in-time (and -space). The values were evaluated at time $t = 10$. The x-axis represents the number of (outer) iterations.

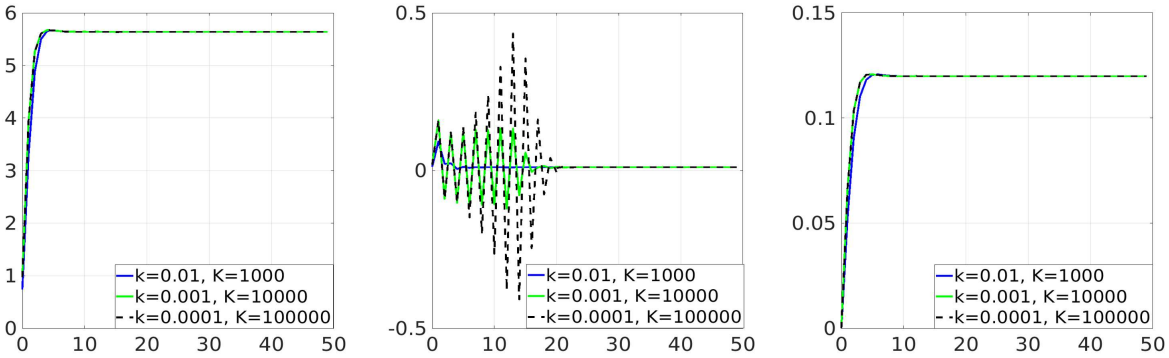


Figure 5.9: Drag (left), lift (center) and pressure difference (right) for the stationary flow-around-a-cylinder benchmark on level 2 using various time step sizes $k \in \{0.01, 0.001, 0.0001\}$ with $K \in \{1000, 10000, 100000\}$ problems computed parallel-in-time (and -space). The values were evaluated at time $t = 10$. The x-axis represents the number of (outer) iterations.

The lift values are the most suitable, as visually larger oscillations occur after each (outer) iteration until the solution converges to the reference value. A smaller time step ensures higher oscillations per (outer) iteration.

There is also a difference between the levels. At higher levels, the occurring oscillations are not as strong. For instance, for $k = 0.0001$, the poorest lift value in the range of ± 2 for level 0, ± 0.6 for level 1 and ± 0.4 for level 2 can be observed. The actual lift reference value is in the range of 0.0106.

Furthermore, it can be deducted that a smaller number of (outer) iterations are necessary to achieve an adequate approximation at a high level. Specially, 30 (outer) iterations are sufficient for level 0, 25 (outer) iterations for level 1 and 22 (outer) iterations for level 2. As illustrated in table 5.1 to table 5.3, this corresponds to a necessary divergence-defect of maximum 10^{-8}.

It is important to note that at level 2 the solver diverges after 22 (outer) iterations for a large time step size with $k = 0.01$, see table 5.3.

The use of insufficiently small time steps, such as $k = 0.01$, can result in convergence issues at a higher level.

$\mathbf{B}^\top\mathbf{u}$												
10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}
$k = 0.01$, $K = 1000$												
1	2	5	8	11	15	19	24	28	31	35	39	42
$k = 0.001$, $K = 10000$												
1	1	(2) 16	21	24	27	29	33	36	38	41	43	45
$k = 0.0001$, $K = 100000$												
1	1	1	(2) 21	26	29	31	34	36	38	41	44	46

Table 5.1: Number of (outer) iterations required to achieve different divergence-defect tolerances on level 0. The defect is evaluated at time $t = 10$ for various the time step sizes $k \in \{0.01, 0.001, 0.0001\}$ with $K \in \{1000, 10000, 100000\}$ problems computed parallel-in-time.

$\mathbf{B}^\top\mathbf{u}$												
10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}
$k = 0.01$, $K = 1000$												
1	2	4	7	11	15	20	24	28	32	35	39	42
$k = 0.001$, $K = 10000$												
1	1	1	16	19	23	25	27	28	30	32	33	35
$k = 0.0001$, $K = 100000$												
1	1	1	1	19	21	25	28	30	31	33	34	36

Table 5.2: Number of (outer) iterations required to achieve different divergence-defect tolerances on level 1. The defect is evaluated at time $t = 10$ for various the time step sizes $k \in \{0.01, 0.001, 0.0001\}$ with $K \in \{1000, 10000, 100000\}$ problems computed parallel-in-time.

$\mathbf{B}^\top\mathbf{u}$												
10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}
$k = 0.01$, $K = 1000$												
1	1	3	6	9	(14) 20	(18)	-	-	-	-	-	-
$k = 0.001$, $K = 10000$												
1	1	1	3	17	20	22	24	26	28	30	31	33
$k = 0.0001$, $K = 100000$												
1	1	1	1	17	19	22	25	27	29	31	32	34

Table 5.3: Number of (outer) iterations required to achieve different divergence-defect tolerances on level 2. The defect is evaluated at time $t = 10$ for various the time step sizes $k \in \{0.01, 0.001, 0.0001\}$ with $K \in \{1000, 10000, 100000\}$ problems computed parallel-in-time.

Regardless of the level, the tables table 5.1 to table 5.3 highlight in green that a smaller time step size k guarantees a smaller starting divergence-defect for the approximate divergence-free update.

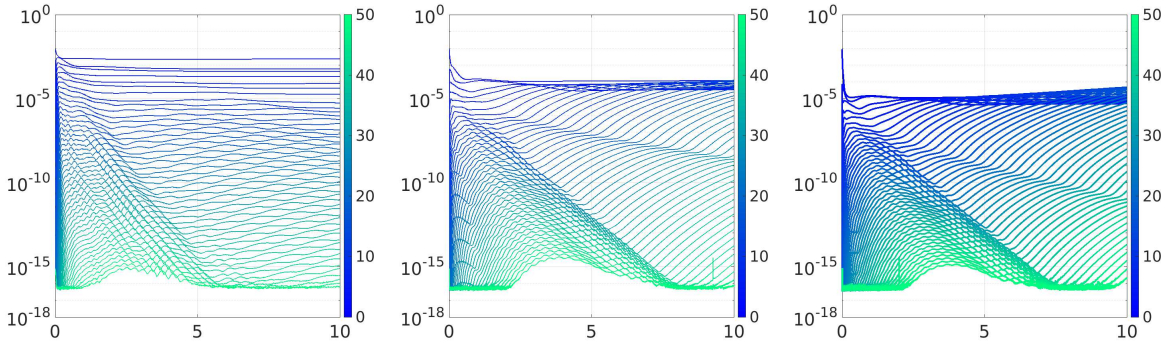
To achieve the sufficient divergence-defect tolerance of 10^{-8}, slightly fewer (outer) iterations are required at higher levels. A larger time step size may reach the tolerance in fewer iterations (highlighted in green), but may cause convergence issues in the worst case (highlighted in red).

Using a multigrid in time can bring acceleration, since a coarser time step provides a faster divergence-free velocity.

Since the divergence-free velocity is only ensured approximately, a computation up to machine precision is not recommended. In this case, smaller steps do not provide the acceleration expected by the PP. For example, for the tolerance 10^{-14}, even more outer iterations are necessary for smaller time step sizes.

The following figures will examine the divergence-defect curve over the time interval in order to gain further insight into its properties. As previously outlined in chapter 4, each line in the figures represents an (outer) iteration of the solver.

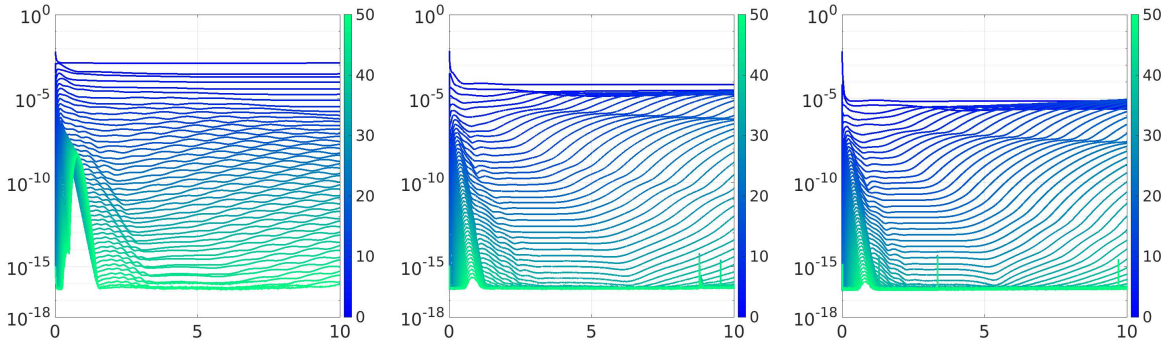
A comparison between level 0 in figure 5.10 and level 1 in figure 5.11 yields that on a finer spatial grid the divergence-defect has difficulties to be reduced, especially for coarser time step sizes at the early time points.



$k = 0.01$, $K = 1000$ $k = 0.001$, $K = 10000$ $k = 0.0001$, $K = 100000$

$\|\mathbf{B}^\top \mathbf{u}\|$

Figure 5.10: Divergence-defect on level 0 using the time step size $k = 0.01$ with $K = 1000$ problems computed parallel-in-time (left), $k = 0.001$ with $K = 10000$ (center) as well as $k = 0.0001$ with $K = 100000$ (right). The x-axis represents the time interval [0,10].

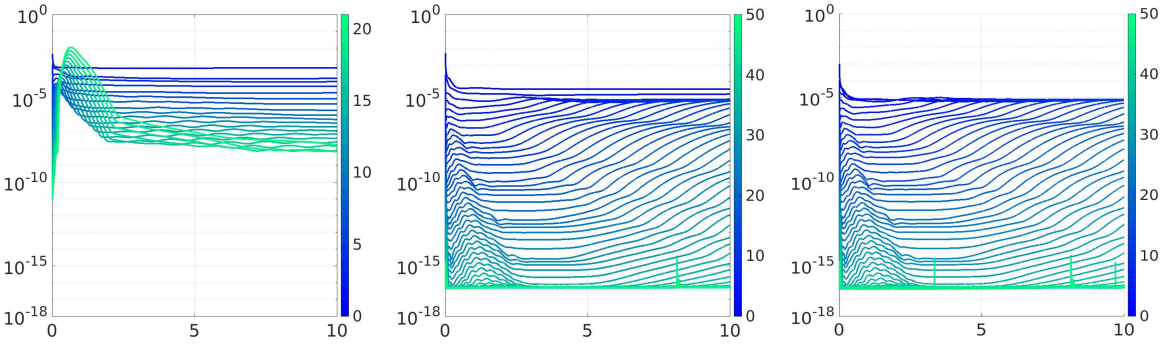


$k = 0.01$, $K = 1000$ $k = 0.001$, $K = 10000$ $k = 0.0001$, $K = 100000$

$\|\mathbf{B}^\top \mathbf{u}\|$

Figure 5.11: Divergence-defect on level 1 using the time step size $k = 0.01$ with $K = 1000$ problems computed parallel-in-time (left), $k = 0.001$ with $K = 10000$ (center) as well as $k = 0.0001$ with $K = 100000$ (right). The x-axis represents the time interval [0,10].

Comparing with the finer spatial grid on level 2 in figure 5.12, it can be seen that the divergence-defect is increased after a few (outer) iterations, if the time step is set too large. Since a divergence-free velocity is no longer ensured, the solver will abort. However, a smaller time step on a finer grid ensures significantly less oscillations.



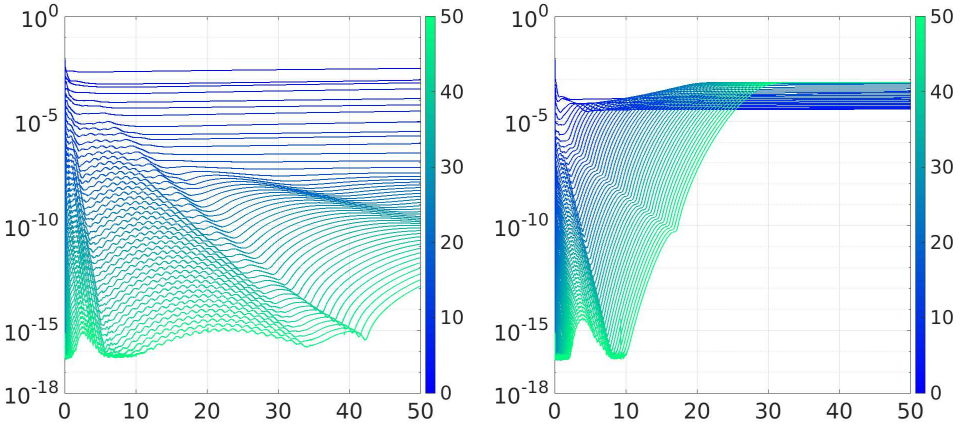
$k = 0.01$, $K = 1000$ $k = 0.001$, $K = 10000$ $k = 0.0001$, $K = 100000$

$\|\mathbf{B}^{\top}\mathbf{u}\|$

Figure 5.12: Divergence-defect on level 2 using the time step size $k = 0.01$ with $K = 1000$ problems computed parallel-in-time (left), $k = 0.001$ with $K = 10000$ (center) as well as $k = 0.0001$ with $K = 100000$ (right). The x-axis represents the time interval [0,10].

Long time-interval computations (stationary flow-around-a-cylinder benchmark)

Further problems with the approximated divergence-free update can be seen in figure 5.13 for large time intervals. A coarse time step size with $k = 0.01$ can still largely guarantee a divergence-free velocity. A smaller step size requires unnecessary number of (outer) iterations for a large number of time-simultaneous computed time steps, since a divergence-free velocity diminishes for larger times.



$k = 0.01$, $K = 5000$ $k = 0.001$, $K = 50000$

$\|\mathbf{B}^{\top}\mathbf{u}\|$

Figure 5.13: Long time computations for time interval [0,50] on level 0 using time step size $k = 0.01$ with $K = 5000$ (left) and $k = 0.001$ with $K = 50000$ (right).

Since the lack of a divergence-free velocity is already observed in the stationary case, it is essential to select a time interval that is not excessively long, particularly in the instationary case, to guarantee a divergence-free velocity and thus convergence.

5.3 Parallel performance in 3D

The 128 element grid in figure 3.3 illustrates the chosen discretization of the domain for the three dimensional case. Figure 5.14 shows the dimensions of the grid.

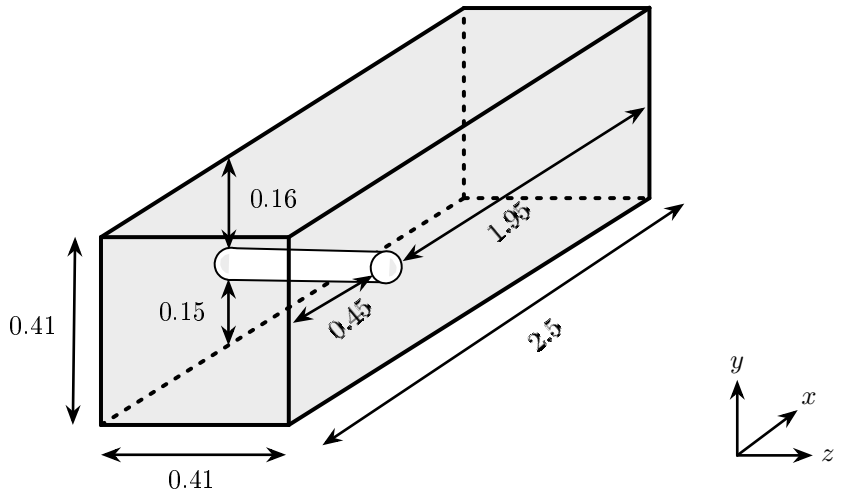


Figure 5.14: Dimensions in 3D for flow-around-a-cylinder grid. The cylinder has a diameter of 0.1. The measurements are in meters. The coordinate system is shown on the right for completeness.

The 3D case is particularly well-suited for performance tests due to the numerous elements involved. However, a proportionately large amount of memory is also necessary, particularly at higher levels and with a large number of time steps to be computed time-simultaneously. To prevent a linear increase in memory space, as illustrated in figure 3.5, and be able to compute a large amount of time steps in parallel-in-time it is recommended to avoid the (global) recoupling approach with GMRES.

Parallel performance for pressure Poisson problem

Since a parallel-in-time velocity solver was not part of this work, only the performance of the pressure Poisson problem is investigated.
Figure 5.15 shows tests on level 0 for many thousands of parallel-in-time and -space computed time-steps. It can be observed that regardless of the number of time steps, a identical behavior can be seen. In both cases, an increase of the computation time from 32 cores to 64 cores can also be observed.

It is not recommended to use many cores for the parallel-in-space computation on a coarse level, particularly when only a few elements are computed on a single core. In such cases, the synchronization time is considerably high, which has a significant impact on the efficiency of the solver.

A parallelization in time shows that using more cores leads to a linear decrease in computation time for the pressure Poisson problems. Regardless of the number of time steps computed in parallel-in-time, an almost identical behavior can be observed.

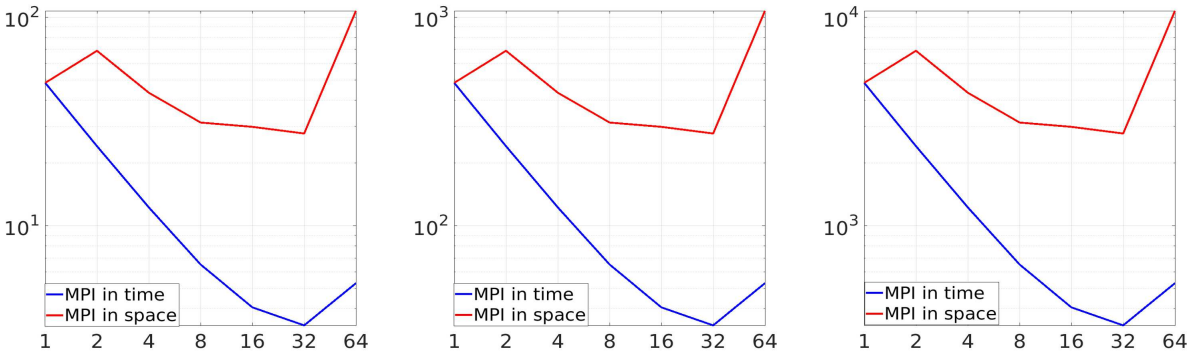


Figure 5.15: Time and space efficiency on level 0 for time step size $k = 0.01$ with $K = 1000$ (left), $k = 0.001$ with $K = 10000$ (center) and $k = 0.0001$ with $K = 100000$ (right). The x-axis represents the number of used cores and the y-axis shows the computation time.

Figure 5.16 shows the parallel performance on a level 2 gird. For the parallel-in time case, similar curves can be observed as for level 0. The parallel-in-space behavior, on the other hand, drops linearly with the cores.

Overall, up to 32 cores, the parallel-in-time computation can be preferred to the parallel-in-space computation, as the computation time is significant less. In case of many cores, the used synchronization must be improved for the parallel-in-time method.

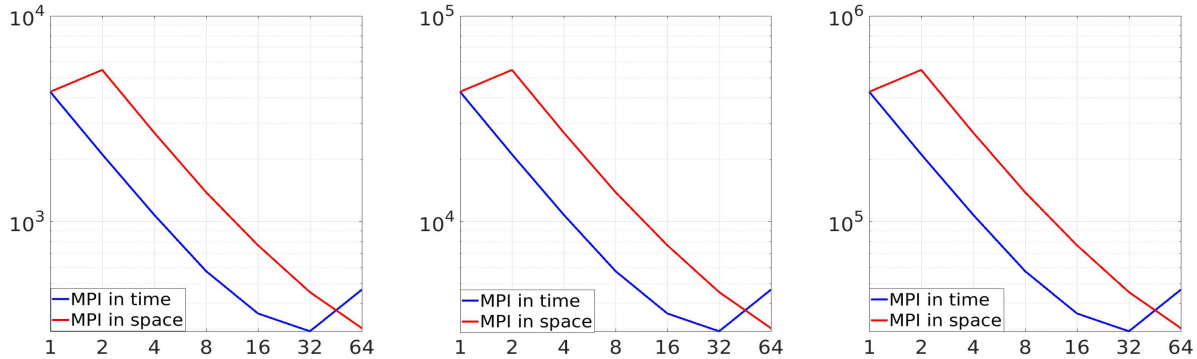


Figure 5.16: Time and space efficiency on level 2 for time step size $k = 0.01$ with $K = 1000$ (left), $k = 0.001$ with $K = 10000$ (center) and $k = 0.0001$ with $K = 100000$ (right). The x-axis represents the number of used cores and the y-axis shows the computation time.

Speed-up and parallel efficiency

Table 5.4 displays the speed up SP and parallel efficiency E for the parallel-in-time and table 5.5 for the parallel-in-space case.

	1	2	4	8	16	32	64
SP	-	2.01	1.97	1.88	1.61	1.22	0.63
E	-	100.6%	98.3 %	94.0%	80.3%	61.0%	31.4%

Table 5.4: Speed-up (SP) and parallel efficiency (E) for different number of cores for the parallel-in-time case. Negative performance is highlighted (red).

	1	2	4	8	16	32	64
				level 0			
SP	-	0.70	1.59	1.39	1.05	1.08	0.26
E	-	35.0%	79.4%	69.5%	52.4%	53.8%	12.9%
				level 2			
SP	-	0.78	2.01	1.95	1.81	1.69	1.5
E	-	39%	100.8%	97.5%	90.6%	84.4%	74.6%

Table 5.5: Speed-up (SP) and parallel efficiency (E) for different number of cores for the parallel-in-space case. Negative performance is highlighted (red).

Pressure behavior

It is interesting that, similar to the two dimensional case in figure 5.4 and figure 5.5, the pressure initially rises during the (outer) iterations as indicated in red in figure 5.17 and then decreases and converges after additional (outer) iterations. The pressure intermediate is incorrect by several orders of magnitude. Figure 5.18 illustrates the corresponding velocity profile. The initial guess for the velocity and pressure is set as zero.

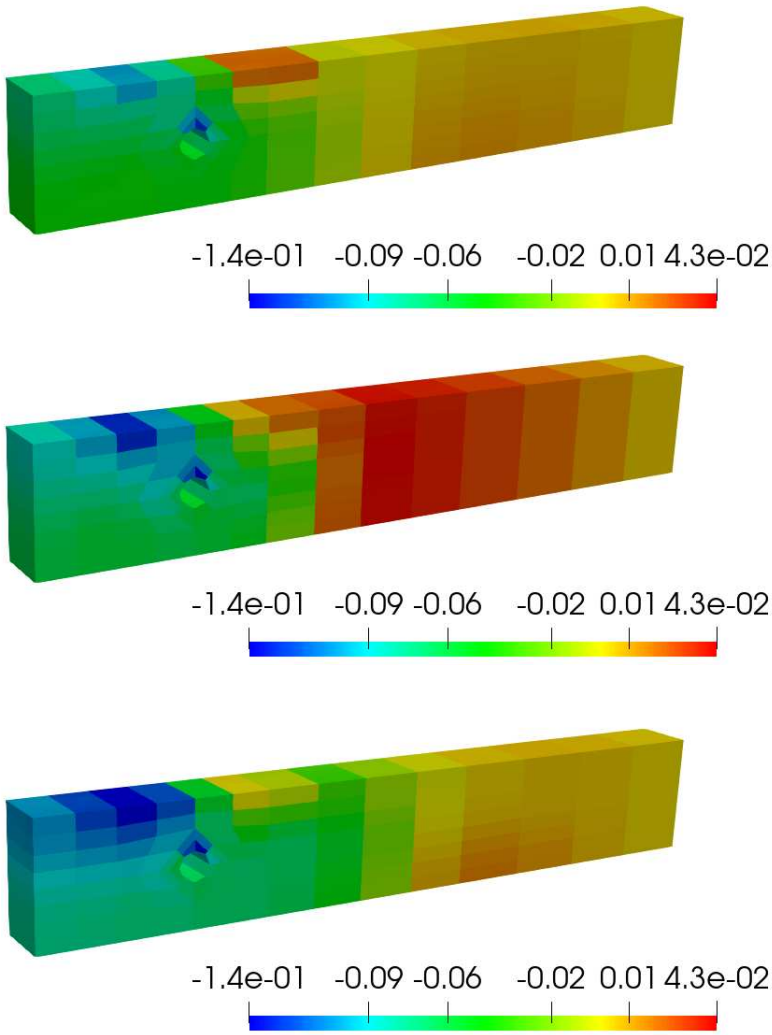


Figure 5.17: Pressure profile for the first (top) and 10. (center) and 40. (bottom) (outer) iteration on level 2.

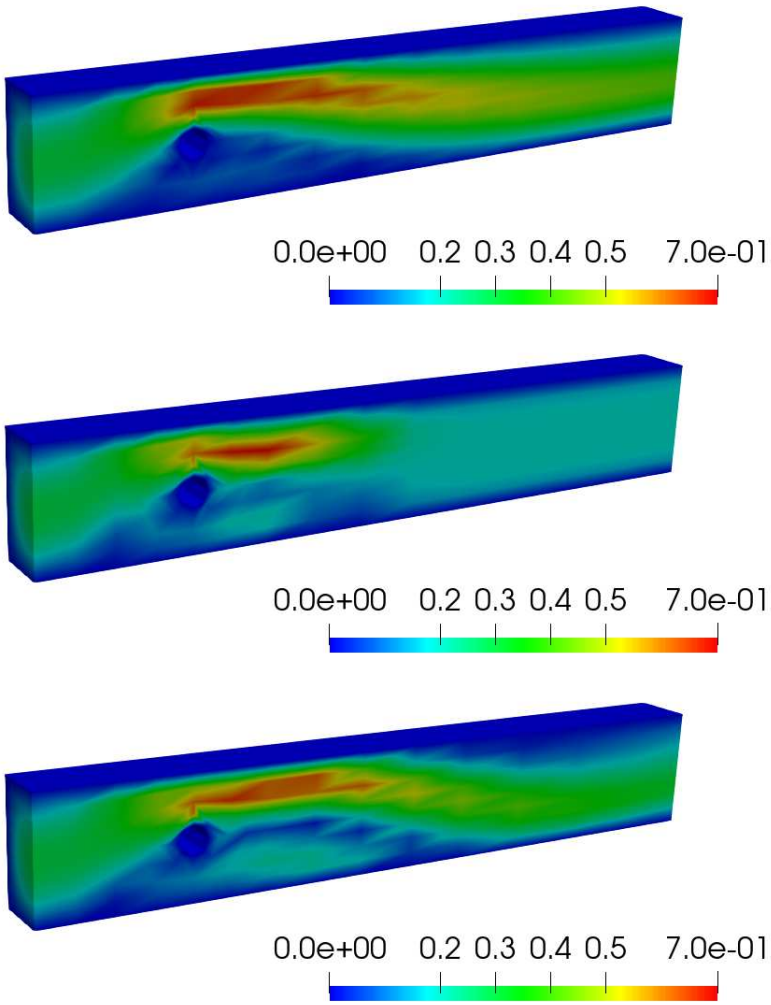


Figure 5.18: Velocity profile for the first (top), 10. (center) and 40. (bottom) (outer) iteration on level 2.

Chapter 6

Conclusions and outlook

Contents

6.1 Conclusion

In this thesis, an efficient numerical and algorithmic implementation of a global-in-time projection method for the incompressible Navier-Stokes equations was developed, tested, and integrated into FEAT3. The Crank-Nicolson scheme was chosen as the time step scheme, which presented along an additional challenge due to its coupling with the previous and current time step.

In chapter 2, the theory was analyzed in detail. The Navier-Stokes equations with their corresponding function spaces were discussed. The primary focus was to explore general criteria for convergence for the continuous case and the discrete case. As a fundamental theorem, the divergence-free condition theorem 2.8 from Brezzi was determined as a sufficient condition for the uniqueness and existence of a solution. Moreover, Eq. (3.11b), in the derivation of the PP-solver in section 3.2, corroborates the necessity of a divergence-free solution to achieve convergence.
Appropriate finite elements were constructed in accordance with the well-posedness, the inf-sup conditions. The Q_2/P_1^{disc} finite element was identified as a preferable option for a Navier-Stokes solver, since $H(div)$ conformity is guaranteed, and therefore the consistency error in section 2.3.3 contain less terms compared to the Taylor-Hood element. In Section 2.3.4, the Crank-Nicolson time-stepping scheme was used to construct a saddle point system Eq. (2.124) that can be solved on a computer.

In chapter 3, the typical sequential-in-time solver approaches for computing the saddle point problem were outlined, and the PP-solver was derived in section 3.2 in detail. A time-simultaneous solver based on the PP-solver, which incorporates various techniques, including the augmented Lagrangian method described in section 3.3.3 and a embedding with FGMRES in section 3.3.4, was presented. These methods were applied to ensure a divergence-free velocity.
The concept of an embedding of a recoupled system with GMRES was initially proposed in [51], in which a global version was presented. However, this approach was identified as having a memory footprint that was exorbitantly costly, especially in three dimensions and with a large number of time steps. To avoid a linear growth in memory as observed in figure 3.4 and figure 3.5, a new local approach was derived, with contains small matrices of the size of one time step.
Furthermore, section 3.1.3 provides an explanation regarding the efficient assembly and computation of the pressure Poisson problems on HPC architectures. This necessitates the usage of a high-level language, such as C++ (FEAT3) to efficiently access the individual cores.

A divergence-free minimization problem is generally constructed in Eq. (3.34). A relationship $\alpha_D^{-1} \approx \gamma$ was derived between the divergence-free pressure update parameter α_D and the divergence-free velocity parameter γ for the augmented Lagrangian method.

As demonstrated in chapter 4, achieving faster a divergence-free velocity results in a reduction in the number of (outer) iterations, as observed in section 4.1.2 for the pressure update.
The augmented Lagrangian method also ensures that only a minimal number of outer iterations are required, as shown in section 4.3.2. However, this approach is extremely ill-conditioned and requires a considerable amount of computation time, as evidenced in section 4.3.5 and was not recommended as an acceleration.
In section 4.5, a return was made to the fundamental principles of the PP-solver, and a novel approximate divergence-free approach for the time-simultaneous PP-solver was developed. This Laplace approach requires less than one percent of the computation time needed for the pressure Poisson problem, thereby providing a significant true acceleration. In section 4.5.2, an enhancement has been implemented that scales the pressure Poisson problem, which has already been computed, from the second iteration onwards. This serves to further accelerate the approximate divergence-free update. As shown in section 4.5.2, a scaling factor of $\gamma_P = 1/3$ for the pressure Poisson problem to $\gamma_L = 2/3$ for the Laplace update provides an optimal approximation of a divergence-free velocity.
A divergence-free right-hand-side for the pressure Poisson equations leads to an incorrect computed pressure in the time-simultaneous PP-solver , as shown in section 4.3.1.
Therefore, tests were conducted to find the optimal right-hand side for the pressure Poisson equations in case of a divergence-free velocity optimization. It was discovered that the most suitable right-hand side is to use a divergence-free velocity from the previous time step and a non-divergence-free velocity from the current time step, as described by Equation (3.42).

In chapter 5, the best settings for the time-simultaneous PP-solver were summarized in section 5.1. It was verified that a multigrid in time approach can be advantageous, especially for larger time intervals, to ensure a approximate divergence-free velocity. It is recommended to avoid embedding with GMRES in cases, where the Reynolds number is within an average range or when the Stokes equations are being used.
The local recoupled multigrid in time approach tested in section 4.4 is not suitable, as it requires too much computation time. In the variant with GMRES, a minimum number of iterations must be performed to ensure convergence. Although the version with FGMRES is more robust, but it requires a significant amount of iterations and computation time due to the preconditioning. In case of Navier-Stokes equations with a high Reynolds number, a global embedding with GMRES is a theoretically suitable approach, for enforcing convergence, but it is important to acknowledge the significant memory space requirements associated with this method.

An important question was whether an integration of a fast solver using the Q_2/Q_1 finite element equipped with the true pressure Poisson matrix can be used in the time-simultaneous PP-solver. Evaluations in section 4.6 show that no convergence occurs in this case. A recoupling with GMRES is necessary in this case, to enforce convergence.

6.2 Outlook

The instationary flow-around-a-cylinder benchmark with Reynolds number 100 needs to be investigated with the approximated improved divergence-free Laplace update of a time-simultaneous solver. First tests indicate that the entire time interval [0.8] cannot be computed simultaneously in time, as the divergence-defect gets increased after a few outer iterations towards the end of the time interval. A smaller time interval must be chosen, which can ensure a better divergence-free velocity. Since the first few (outer) iterations converges for the entire time interval, the recoupling approach with GMRES could also leads to convergence.

Furthermore, following extensions are available for consideration:

- The most important aspect to consider is the integration of a time-simultaneous or time-parallel solver for the Burgers equation.
- Further testing of the multigrid in time method, particularly in the context of the Navier-Stokes equations, is recommended. Testing of different cycles, such as the W-cycle or F-cycle, may facilitate the identification of an optimal convergence behavior.
- It would be beneficial to consider rewriting the augmented Lagrangian approach after [53] as

$$(\mathbf{S}+\gamma\mathbf{B}\mathbf{M}_p^{-1}\mathbf{B}^\top)^{-1} = \mathbf{S}^{-1} - \frac{1}{1+\operatorname{tr}\left(\left(\gamma\mathbf{B}\mathbf{M}_p^{-1}\mathbf{B}^\top\right)\mathbf{S}^{-1}\right)}\mathbf{S}^{-1}\left(\gamma\mathbf{B}\mathbf{M}_p^{-1}\mathbf{B}^\top\right)\mathbf{S}^{-1}, \tag{6.1}$$

  if it is to be a subject of further investigation.
  The computation of $\mathbf{S}^{-1}$ is both rapid and exhibits a tendency to accelerate with a reduction in the time step size k.
- ...

Bibliography

[1] IEEE Std 754-2008. Ieee standard for binary floating-point arithmetic. *ANSI/IEEE Std*, pages 1–20, October 1985.

[2] J. Guermond A. Ern. *Theory and Practice of Finite Elements*. 2004. ISBN 978-1-4419-1918-2.

[3] J. E. Marsden A. J. Chorin. *A Mathematical Introduction to Fluid Mechanics*, volume Thrid edition. 1992. ISBN 3-540-97918-2.

[4] W.E. Arnoldi. The principle of minimized iteration in the solution of the matrix eigenvalue problem. *Quart.Appl.Math*, 9:17–29, 1951.

[5] K. Arrow, L. Hurwicz, and H. Uzawa. Studies in nonlinear programming. *Standford CA Stanford University Press*, 1958.

[6] R. Barrett, M. Berry, T.F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Rominr, and H. Van der Vorst. *Templates for the Solution if Linear Systems: Building Blocks for Iterative Methods*. 1994.

[7] E. Bayraktar, O. Mierka, and S. Turek. Benchmark computations of 3d laminar flow around a cylinder with cfx, openfoam and featflow. *Int. J. of Computational Science and Engineering*, 7:253–266, 07 2012.

[8] J. Bender and D. Koschier. Divergence-free smoothed particle hydrodynamics. *Rwth Aachen*, pages 1–9, 2015.

[9] J. Bender and D. Koschier. Divergence-free sph for incompressible and viscous fluids. *Rwth Aachen*, pages 1–14, 2017.

[10] M. Benzi and M. A. Olshanskii. An augmented lagrangian-based approach to the oseen problem. *SIAM Journal on Scientific Computing*, 28:2095–2113, 2006.

[11] D. Boffi and C. Lovadina. Analysis of new augmented lagrangian formulations for mixed finite element schemes. *Numerische Mathematik*, Numer. Math. 75:405–419, 1997.

[12] M. Braack and T. Richter. Solutions of 3d navier-stokes benchmark problems with adaptive finite elements. *Computers and Fluids*, 35(4):372–392, 2006.

[13] D. Braess. *Numerical Solution of Partial Differential Equations by the Finite Element Method*. Dover Pubn Inc, Dover, 2009.

[14] D. Braess. *Finite Elemente*. Springer, 2013. ISBN 978-3-642-34796-2.

[15] J.H. Bramble. *Multigrid Methods*, volume 294. 1993. ISBN 9780582234352.

[16] H. Brezis. *Functional Analysis, Sobolev Spaces and Partial Differential Equations*. Springer, 2011. ISBN 978-0-387-70913-0.

[17] F. Brezzi. On the existence, uniqueness and approximation of saddle-point problems arising from lagrangian multipliers. *Revue francaise d'automatique, informatique, recherche operationnelle. Analyse numerique*, tome 8(n. 2):p. 129–151, 1974.

[18] L. Chen. A simple construction of a fortin operator for the two dimensional taylor-hood element. *Computers and Mathematics with Applications*, 68:p. 1368–1373, 2014.

[19] A. J. Chorin. A numerical method for solving incompressible viscous flow problems. *Journal of Computational Physics*, 2:12–26, 1967.

[20] A. J. Chorin. Numerical solution of the navier-stokes equations. *Math Comp. 22*, pages 745–762, 1968.

[21] A. J. Chorin. On the convergence of discrete approximations to the navier-stokes equations. *Math Comp. 23*, pages 341–353, 1969.

[22] M. Fortin D. Boffi, F. Brezzi. *Mixed Finite Element Methods and Applications.* Springer, 2013. ISBN 978-3-642-36518-8.

[23] J. Hämäläinen D. Kuzmin. *Finite Element Methods for Computational Fluid Dynamics - A Practical Guide.* 2015.

[24] Federico Danieli, Ben S. Southworth, and Andrew J. Wathen. Space-time block preconditioning for incompressible flow, 2021.

[25] A. Davey and P. G. Drazin. The stability of poiseuille flow in a pipe. *Journal of Fluid Mechanics*, 36(2):209–218, 1969.

[26] J. Dünnebacke, S. Turek, M. Lohmann, A. Sokolov, and P. Zajac. Increased space–parallelism via time–simultaneous Newton–multigrid methods for nonstationary nonlinear PDE problems. Technical report, Fakultät für Mathematik, TU Dortmund, December 2020. Ergebnisberichte des Instituts für Angewandte Mathematik, Nummer 634.

[27] J. Dünnebacke, S. Turek, P. Zajac, and A. Sokolov. A time–simultaneous multigrid method for parabolic evolution equations. Technical report, Fakultät für Mathematik, TU Dortmund, December 2019. Ergebnisberichte des Instituts für Angewandte Mathematik, Nummer 619.

[28] P. Hansbo E. Burman and M.G. Larson. The augmented lagrangian method as a framework for stabilised methods in computational mechanics. *Archives of Computational Methods in Engineering*, 30:2579–2604, January 2023.

[29] L. C. Evans. *Partial Different Equations - Second Edition*, volume v 19. 2010. ISBN 978-0-8218-4974-3.

[30] M. Fortin F. Brezzi. *Mixed and Hybrid Finite Element Methods.* 1991. ISBN-13: 978-1-4612-7824-5.

[31] M. Fortin F. Brezzi, L. Demkowicz and R. G. Duran. Finite elements for the stokes problem. *Chapter in Lecture Notes in Mathematics - Springer Verlag*, pages p. 47–101, January 2008.

[32] J. Stoer F. Jarre. *Optimierung - Einführung in mathematische Theorie und Methoden*, volume 2. Auflage. 2019. ISBN 978-3-662-58854-3.

[33] Patrick E. Farrell, Lawrence Mitchell, and Florian Wechsung. An augmented lagrangian preconditioner for the 3d stationary incompressible navier–stokes equations at high reynolds number. *SIAM Journal on Scientific Computing*, 41(5):A3073–A3096, jan 2019.

[34] M. Fortin and R. Glowinski. Augmented lagrangian methods: Applications to the numerical solution of boundary-value problems. *Studies in Mathematics and Its Applications, Elsevier Science Ltd*, 15, 1983.

[35] S. Friedhoff, R. D. Falgout, T. V. Kolev, S. P. MacLachlan, and J. B. Schroder. A multigrid-in-time algorithm for solving evolution equations in parallel. *Sixteenth Copper Mountain Conference on Multigrid Methods, Copper Mountain, CO, United States*, 2013.

[36] J.L. Guermond, P. Minev, and Jie Shen. An overview of projection methods for incompressible flows. *Computer Methods in Applied Mechanics and Engineering*, 195(44):6011–6045, 2006.

[37] W. Hackbusch. *Multi-grid methods and applications.* Number 4. 1985. ISBN 978-3-642-05722-9.

[38] J. Hadamard. Sur les problemes aux derivees partielles et leur signification physique. *Princeton University Bulletin*, 13:49–52, 1902.

[39] S. L. Harris and D. M. Harris. *Digital Design and Computer Architecture - ARM Edition*. 2016. ISBN 978-0-12-800056-4.

[40] T. Heister and G. Rapin. Efficient augmented lagrangian-type preconditioning for the oseen problem using grad-div stabilization. *International Journal for Numerical Methods in Fluids*, 71:118–134, 2012.

[41] M. R. Hestenes. Multiplier and gradient methods. *Journal of Optimization Theory and Applications*, 4:303–320, 1969.

[42] V. John. Higher order finite element methods and multigrid solvers in a benchmark problem for the 3d navier-stokes equations. *International Journal for Numerical Methods in Fluids*, 40(6):775–798, 2002.

[43] V. John, A. Linke, C. Merdon, M. Neilan, and L. G. Rebholz. On the divergence constraint in mixed finite element methods for incompressible flows. *SIAM Review*, 59:492–544, 2017.

[44] V. John and J. Rang. Adaptive time step control for the incompressible navier-stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 199(9):514–524, 2010.

[45] M. Köster. *A Hierarchical Flow Solver for Optimisation with PDE Constraints*. Phd thesis, Universität Dortmund, Fakultät für Mathematik, Lehrstuhl 3 für Angewandte Mathematik und Numerik, July 2011.

[46] P.D. Minev L-J.P. Timmermans and F.N. van de Vosse. An approximate projection scheme for incompressible flow using spectral elements. *International Journal for numerical methods in Fluids*, 22:673–688, 1996.

[47] T. J.R. Hughes L. P. Franca. Two classes of mixed finite element methods. *Computer Methods in Applied Mechanics and Engineering*, 69(1):89–129, 1988.

[48] J.-L. Lions, Y. Maday, and G. Turinici. A 'parareal' in time discretization of pde's. *Comptes Rendus de l'Academie des Sciences - Series I - Mathematics*, 332:661–668, 2001.

[49] C. Lohmann, J. Dünnebacke, and S. Turek. Fourier analysis of a time–simultaneous two–grid algorithm for the one–dimensional heat equation. Technical report, Fakultät für Mathematik, TU Dortmund, April 2021. Ergebnisberichte des Instituts für Angewandte Mathematik, Nummer 641.

[50] C. Lohmann and S. Turek. Augmented lagrangian acceleration of global–in–time pressure schur complement solvers for incompressible Oseen equations. Technical report, Fakultät für Mathematik, TU Dortmund, December 2023. Ergebnisberichte des Instituts für Angewandte Mathematik, Nummer 669.

[51] C. Lohmann and S. Turek. On the design of global–in–time Newton–multigrid–pressure schur complement solvers for incompressible flow problems. Technical report, Fakultät für Mathematik, TU Dortmund, May 2023. Ergebnisberichte des Instituts für Angewandte Mathematik, Nummer 664.

[52] J. Liesen M. Benzi, G. H. Golub. *Numerical solution of saddle point problems*. 2005. DOI: 10.1017/S0962492904000212.

[53] Kenneth S. Miller. On the inverse of the sum of matrices. *Mathematics Magazine, Published By: Taylor & Francis, Ltd.*, 54(2):67–72, March 1981.

[54] J. Serrin N. G. Meyers. 'h = w'. *Proceedings of the National Academy of Sciences*, 6(51):1055–1056, June 1964.

[55] J. Necas. Sur une methode pour resoudre les equations aux derivees partielles du type elliptique, voisine de la variationnelle. *Ann. Scuola Norm. Sup. Pisa*, 3:305–326, 1962.

[56] M. A. Olshanskii. A low order galerkin finite element method for the navier-stokes equations of steady incompressible flow: a stabilization issue and iterative methods. *Computer methods in applied mechanics and engineering*, 2002. 5515-5536.

[57] M. A. Olshanskii and A. Reusken. Grad-div stabilization for stokes equations. *Mathematics of computation*, 73(248):1699–1718, 19 December 2003.

[58] M. Powell. A method for nonlinear constraints in minimization problems. *In Fletcher, R., editor, Optimization, Academic Press, New York*, pages 283–298, 1969.

[59] B. J. P. Kaus R. S. Lehmann, M. Lukacova-Medvid'ova and A. A. Popov. Comparison of continuous and discountinous galerkin approaches for variable-viscosity stokes flow. *Zeitschrift für Angewandte Mathematik und Mechanik*, 24 August 2015.

[60] R. Rannacher. *Numerik 2: Numerik partieller Differentialgleichungen.* Heidelberg University Publishing, Heidelberg, 2017. ISBN 978-3-946054-38-2.

[61] Rolf Rannacher. *Methods for Numerical Flow Simulation*, pages 275–332. Birkhäuser Basel, Basel, 2008.

[62] P-A. Raviart and J-M. Thomas. A mixed finite element method for 2nd order elliptic problems. In Ilio Galligani and Enrico Magenes, editors, *Mathematical aspects of finite element methods*, volume 606, pages 292–315. 1977.

[63] D. Ruda, S. Turek, D. Ribbrock, and P. Zajac. Very fast finite element poisson solvers on lower precision accelerator hardware: A proof of concept study for nvidia tesla v100. *The International Journal of High Performance Computing Applications*, 36(4):459–474, May 2022.

[64] Y. Saad. A flexible inner-outer preconditioned gmres algorithm. *SIAM Journal on Scientific Computing,*, 14:461–469, 1993.

[65] Yousef Saad and Martin H. Schultz. Gmres: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *Siam Journal on Scientific and Statistical Computing*, 7:856–869, 1986.

[66] N. Satto. Notes on the banach-necas-babuska theorem and kato's minimum modulus of operators. *arXiv:1711.01533v4*, 2018.

[67] L.R. Scott S.C. Brenner. *The Mathematical Theory of Finite Element Methods*, volume Third Edition. 2008. ISBN 978-0-387-75933-3.

[68] M. Schäfer, S. Turek, F. Durst, E. Krause, and R. Rannacher. *Benchmark Computations of Laminar Flow Around a Cylinder*, pages 547–566. Vieweg+Teubner Verlag, Wiesbaden, 1996.

[69] M. Schäfer, S. Turek, F. Durst, E. Krause, and R. Rannacher. Benchmark computations of laminar flow around a cylinder. *Flow simulation with high-performance computers II: DFG priority research programme results 1993-1995*, pages 547–566, 1996.

[70] F. Schieweck. Advantages and construction of h(div)-conforming divergence-free finite element pairs for 3d incompressible flows. *TU Dortmund, Talk*, March 2013.

[71] B. Schweizer. *Partielle Differentialgleichungen : Eine anwendungsorientierte Einführung.* Springer, 2013. ISBN 978-3-642-40637-9.

[72] L. R. Scott and M. Vogelius. Norm estimates for a maximal right inverse of the divergence operator in spaces of piecewise polynomials. *ESAIM: Mathematical Modelling and Numerical Analysis - Modelisation Mathematique et Analyse Numerique*, 19(1):111–143, 1985.

[73] V. V. Shaidurov. *Multigrid Methods for Finite Elements.* 1989. ISBN 978-90-481-4506-5.

[74] A. Sokolov. *Analysis and Numerical Realisation of Discrete Projection Methods for Rotating Incompressible Flows.* Phd thesis, Universität Dortmund, Fakultät für Mathematik, Lehrstuhl 3 für Angewandte Mathematik und Numerik, November 2008.

[75] R. Temam. Sur l'approximation de la solution des equations de navier-stokes par la methode des pas fractionnaires ii. *Arch. Rational Mech. Anal. 33*, pages 337–385, 1969.

[76] A. Thom. The flow past circular cylinders at low speeds. *Royal Society, London*, 17:651–666, 1933.

[77] Stephen Thomas, Arielle Carr, Paul Mullowney, Ruipeng Li, and Kasiawirydowicz. Neumann series in gmres and algebraic multigrid smoothers, 2021.

[78] S. Turek. *Efficient Solvers for Incompressible Flow Problems.* Springer, 1999. ISBN 978-3-642-58393-3.

[79] S. Turek, F. Schieweck, S. Basting, H. Damanik, M. Köster, O. Mierka, A. Ouazzi, and P. Zajac. Numerical studies regarding accuracy and robustness of "divergence-free" finite element discretizations. *Magdeburg, Talk*, 2016.

[80] F. Nataf V. Dolean, P. Jolivet. *An Introduction to Domain Decomposition Methods: Algorithms, Theory, and Parallel Implementation.* 2015. ISBN 978-1-611974-05-8.

[81] M. Vogelius. A right-inverse for the divergence operator in spaces of piecewise polynomials. application to the p-version of the finite element method. *Numerische Mathematik*, 41:19–38, 1983.

[82] J. Zhu, Z. R. L. Taylor, and O. C. Zienkiewicz. The finite element method: its basis and fundamentals. pages 54–102, 2005.

Appendix I

MPI implementation

A possibility for a parallel-in-space and -time implementation in C++ is presented in figure I.1.

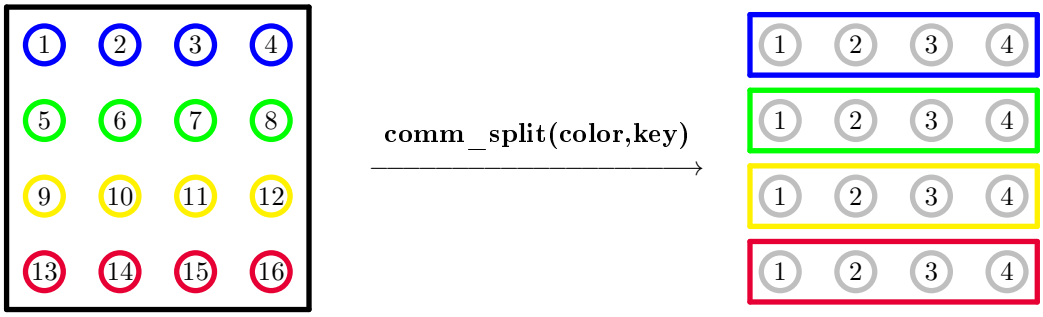


Figure I.1: Split a large communicator (black box) into smaller communicators (blue, green, yellow and red box).

The presented function is based on the C++ command **comm_split**[1], which provides an efficient method for splitting a communicator into subgroups (sub-communicators).

Algorithm 1.1: Split main communicator into smaller communicators

```
1 /* set mpi processes for space */
2 int color = comm.rank() / cfg.comm_space;                    // user given input parameter
   cfg.comm_space (integer)
3 int key = comm.rank();
4 Dist::Comm row_comm(comm.comm_split(color,key));
5 Control::Domain::PartiDomainControl<DomainLevelType> domain(row_comm, true);
6 comm.print("Number of Processes (Space / Time): " + stringify(row_comm.size()) + " /
   " + stringify(comm.size()/cfg.comm_space) +"\n");
```

Algorithm 1.1 shows a segment of the FEAT3 code that has been implemented. New sub-communicators can be created based on the parameters **colors** and **keys**.

The input parameter **color** controls the subset assignment (integer) and the input parameter **key** controls the rank assignment (integer).

To illustrate, with 16 cores available, 4 sub-communicators (**color**) can be allocated for the parallel-in-time computation. Each of the 4 sub-communicators is then capable of performing with 4 cores (**key**) the parallel-in-space computation via domain decomposition.

On each of the 4 sub-communicators, it es necessary to assemble the matrices and vectors that are required for the computation. Use the parameter **key** to order the ranks. This is a crucial aspect of synchronization in space, as the domain decomposition results in various vector sizes across the cores. A synchronization can only occur between **key** i and **key** i from another sub-communicator, for $i = \{1, 2, 3, 4\}$

[1] https://www.mpich.org/static/docs/latest/www3/MPI_Comm_split.html