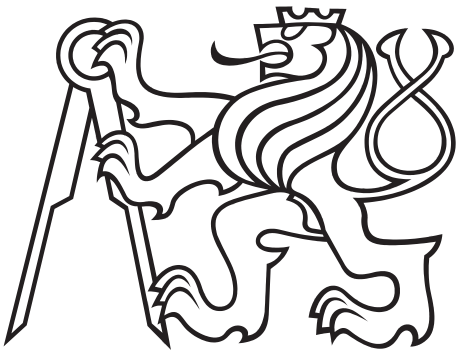
Czech Technical University in Prague
Faculty of Mechanical Engineering



The finite element method in fluids: stabilization and domain decomposition

Ph.D. thesis

Ing. Jakub Šístek

Specialization
Mathematical and Physical Engineering

Advisor
Prof. RNDr. Pavel Burda, CSc.

July 2008

Acknowledgements

At this place, I would like to thank to several people, who have had great influence on my work.

First, I would like to thank to my advisor Prof. Pavel Burda. He introduced me to applied mathematics, has always motivated me with his optimism, and greatly supported me in various ways during all the time we have worked together. He also initiated this research.

I greatly thank to Dr. Jaroslav Novotný. He has been my teacher of using computers and introduced me to the world of parallel computers. I am also very grateful for providing me his frontal solvers, which were useful in many places of the implementation and saved me a lot of time, and for many interesting industral problems for testing the programs.

I would like to thank to Prof. Jan Mandel, who created wonderful conditions for my research during my visit at University of Colorado Denver. He has greatly accelerated the work by frequent consultations on domain decomposition methods and contributed with many helpful ideas.

I thank to Prof. Ivo Marek for his readiness, patience and time he spent with me during many useful consultations on various topics.

I want to thank also to my colleagues and friends Dr. Marta Čertíková and Bedřich Sousedík for pleasant and inspiring collaboration on the subject of domain decomposition.

I greatly thank to my parents, who has kept supporting me during the time of my studies and thus allowed me to fully concentrate on the research, and for the way they have educated me.

Finally, I would like to express my greatest thanks to my wife Hanka and our daughter Anežka, for their tolerance and patience, with which they both accepted my work, and for creating beautiful family background, which is so important for any human efforts.

My research during the Ph.D. studies was supported by the Grant Agency of the Czech Republic under grants 106/05/2731 and 106/08/0403, by the Czech Academy of Sciences under grants IAA2120201/02 and 1ET 400760509, by the U.S. National Science Foundation under grant DMS-0713876, and by HPC-Europa project (RII3-CT-2003-506079) supported by European Community - Research Infrastructure Action under the FP6.

Chýnov, July 13th 2008

Abstrakt

Předkládaná disertační práce se zabývá použítím metody konečných prvků (MKP) na řešení problémů proudění. Je uvažována vazká nestlačitelná tekutina. Její proudění je popsáno soustavou Stokesových či Navierových-Stokesových rovnic.

V práci je dále studována technika stabilizace MKP. Touto metodou je možné úspěšně řešit některá proudění při Reynoldsových číslech výrazně vyšších, než pro jaká jsme schopni nalézt řešení standardní MKP. Za tuto možnost ovšem platíme sníženou přesností výpočtu. V práci je navržena stabilizovaná metoda semiGLS, která je testována na řadě problémů s nestlačitelným prouděním. Dále jsou navrženy dva postupy pro vyhodnocování nepřesnosti, kterou do problému stabilizace zavádí. Pro tento účel jsou úspěšně využity a posteriorní odhady chyby MKP.

Dále jsou diskutovány metody rozkladu oblasti, zejména metoda Balancing Domain Decomposition by Constraints (BDDC). V práci jsou představeny dva algoritmy metody BDDC vhodné pro paralelní počítače. První z nich je založen na standardních součástech MKP programů, druhý využívá globální přístup umožněný navrženou reformulací metody. Oba algoritmy byly implementovány a rozsáhle testovány a vybrané výsledky těchto experimentů jsou prezentovány. Algoritmy jsou testovány na úlohách lineární pružnosti, tedy úlohách se symetrickými pozitivně definitními maticemi. Následně je metoda BDDC úspěšně aplikována na problémy Stokesova proudění.

Abstract

The thesis is devoted to the application of the Finite Element Method (FEM) for solving flow problems. Incompressible viscous fluid is considered. Such flows are governed by the systems of Stokes or Navier-Stokes equations.

In the thesis, techniques of stabilization of the FEM are studied. Using this approach, it is possible to solve problems with considerably higher Reynolds number, than it would be possible using a standard FEM. However, we pay for this possibility by some loss of accuracy. The semiGLS stabilized method is proposed in the thesis and is tested on a number of applications with incompressible flows. Then, two approaches for evaluation of the error induced by stabilization are proposed. For this purpose, a posteriori error estimates of the FEM are successfully applied.

Further, domain decomposition methods are discussed, especially Balancing Domain Decomposition by Constraints (BDDC). Two algorithms of the BDDC method for parallel computers are presented. The first is based on standard parts of FEM software, the second uses a global approach allowed through the proposed reformulation of the method. Both algorithms were implemented and extensively tested and selected results of these experiments are presented. The algorithms are first tested on problems of linear elasticity, which have symmetric positive definite matrices. The BDDC method is then successfully applied to problems of Stokes flow.

Contents

Index of symbols **6**

Abbreviations **8**

1 Introduction **9**

2 Models of incompressible viscous flow **14**
2.1 The Navier-Stokes equations 14
2.2 The Stokes equations 15
2.3 Weak formulation of Stokes and Navier-Stokes problems 15

3 Mixed finite element method for flow problems **18**
3.1 Function spaces for velocity and pressure approximation 18
3.2 Taylor-Hood finite elements 19
3.3 Discretization of steady Stokes and Navier-Stokes equations by FEM 21
3.4 Discretization of unsteady Stokes and Navier-Stokes equations by FEM 22

4 SemiGLS stabilization of the finite element method **24**
4.1 Formulation of the stabilized problem 24
4.2 Newton method for solution of the stabilized problem 26
4.2.1 Functionals for the Newton method and their differentials in steady case . . 26
4.2.2 Matrices for the finite element method in steady case 30
4.2.3 Functionals for the Newton method and their differentials in unsteady case 38
4.2.4 Matrices for the finite element method in unsteady case 40
4.2.5 Computation of stabilization parameter 46
4.3 Accuracy of the stabilized method 48

5 BDDC domain decomposition method **51**
5.1 Introduction to iterative substructuring 51
5.2 Formulation of BDDC 53
5.3 Projected BDDC preconditioner 58
5.4 Generalized change of variables 60

6 Parallel algorithms of the BDDC method **63**
6.1 BDDC by frontal solver 63
6.2 BDDC by multifrontal solver 65
6.3 Details of the algorithm 66
6.3.1 Algorithm of preconditioned conjugate gradient method for BDDC 66
6.3.2 Efficient inverse of transformation matrix 69
6.3.3 Storing the matrices in memory 70

7 Numerical results **73**
7.1 Applications of semiGLS stabilization to Navier-Stokes flow 73
7.1.1 Steady flow of lid driven cavity 73
7.1.2 Steady flow in channel with sudden extension of diameter 74
7.1.3 Flow past NACA 0012 airfoil 79
7.2 Applications of BDDC to linear elasticity 84
7.2.1 Cube problem 84
7.2.2 Steam turbine entry nozzle 88
7.2.3 Shaft with a groove 90
7.3 Numerical experiments with BDDC for steady Stokes flow 92
7.3.1 Lid driven cavity 92
7.3.2 Channel with sudden extension of diameter 92
7.3.3 Channel with sudden reduction of diameter 93

8 Conclusion **96**

Bibliography **98**

Index of symbols

(a, b)	open interval of real numbers
$[a, b]$	closed interval of real numbers
C, C_1, C_2 etc.	generic constants
Ω	domain, a subset of $\mathbb{R}^2$ or $\mathbb{R}^3$
$\partial\Omega$	boundary of domain Ω
$\partial\Omega_g$	part of boundary of domain Ω with Dirichlet boundary condition
$\partial\Omega_{dn}$	part of boundary of domain Ω with natural ('do nothing') boundary condition
Ω_i	subdomain (also called substructure) i, a subset of Ω
$\partial\Omega_i$	boundary of subdomain Ω_i
Γ	interface
T_K	finite element K
h_K	diameter of element T_K
H	characteristic size of subdomain
h	characteristic size of finite element
$C^n(\Omega)$	space of n-times continuously differentiable functions on Ω
$L_2(\Omega)$	Lebesgue space of square integrable functions over Ω
$H^1(\Omega)$	Sobolev space $W^{1,2}(\Omega)$
$H^1_0(\Omega)$	subspace of $H^1(\Omega)$ with zero traces on $\partial\Omega$
V	linear space of vector functions with components in $H^1(\Omega)$ with zero traces on $\partial\Omega_g$
V_g	linear space of vector functions with components in $H^1(\Omega)$ with traces satisfying Dirichlet boundary conditions on $\partial\Omega_g$
$V_{gh} \subset V_g$	finite element functions from V_g
$V_h \subset V$	finite element functions from V
$Q_h \subset L_2(\Omega)/\mathbb{R}$	finite element functions from $L_2(\Omega)/\mathbb{R}$
W_i	space of finite element functions on subdomain Ω_i
W	product space of finite element functions discontinuous across subdomains, $W = W_1 \times \cdots \times W_N$
U	space of finite element functions continuous on Ω
U'	dual space to U
U_Γ	subspace of U of finite element functions with minimal energy on subdomains, defined by values on interface
U'_Γ	dual space to U_Γ
U_i	subspace of U of finite element functions with nonzero values only in interior of subdomain Ω_i

$\widetilde{W}$	subspace of W of functions continuous at all coarse degrees of freedom (unknowns at corners and averages)
$\widetilde{W}^c$	subspace of W of functions continuous at corners
$\widetilde{W}^{avg}$	subspace of $\widetilde{W}^c$ of functions with continuous averages across subdomains
$\widetilde{W}_C$	subspace of $\widetilde{W}$ of functions with minimal energy on subdomains, defined by values in coarse degrees of freedom
$\widetilde{W}_i$	subspace of $\widetilde{W}$ of functions with nonzero values in subdomain Ω_i and zero coarse degrees of freedom
$\|\cdot\|_0$	$L_2(\Omega)$ norm
$\|\cdot\|_1$	$H^1(\Omega)$ norm
$\mathcal{F}(\cdot,\cdot)$, $F_1(\cdot,\cdot)$, $F_2(\cdot,\cdot)$	functionals on $V_{gh} \times Q_h$
$< DF_1(\cdot,\cdot),[\mathbf{h},q] >$ ..	Gateaux differential of functional $F_1(\cdot,\cdot)$ with respect to $\mathbf{h}$ and q
$a(\cdot,\cdot)$	symmetric bilinear form on $U \times U$
$\langle\cdot,\cdot\rangle$	duality pairing
$\widetilde{a}(\cdot,\cdot)$	bilinear form on $\widetilde{W} \times \widetilde{W}$
A	operator associated with $a(\cdot,\cdot)$ by $\langle Au, v\rangle = a(u,v)\ \forall u, v \in U$
$\widetilde{A}$	operator associated with $\widetilde{a}(\cdot,\cdot)$ by $\langle \widetilde{A}u, v\rangle = \widetilde{a}(u,v)\ \forall u, v \in \widetilde{W}$
t	time
τ,δ	stabilization parameters
ϑ	time step
s,$\overline{s}$	scaling parameters
$\mathbf{u}$	vector field of fluid velocity, resp. displacement
p	pressure divided by density
ν	kinematic viscosity of fluid
$\mathbf{f}$	density of volume forces per mass unit
S	Schur complement matrix
G, $\overline{G}$	matrix of global constraints
null(G)	nullspace of G
C_i, $\overline{C}_i$	subdomain matrix of constraints
r	residual
f	right hand side vector
E	operator of projection, $E:\ \widetilde{W} \to U_\Gamma$ or $E:\ \widetilde{W}^{avg} \to U_\Gamma$
T	operator of transformation, $T:\ \widetilde{W}^c \to \widetilde{W}^c$
D_P	operator of weights, $D_P:\ \widetilde{W} \to \widetilde{W}$ or $D_P:\ \widetilde{W}^{avg} \to \widetilde{W}^{avg}$
R	operator of injection, $R:\ U_\Gamma \to \widetilde{W}$ or $R:\ U_\Gamma \to \widetilde{W}^{avg}$
P,$\overline{P}$	operator of projection, $P:\ \widetilde{W}^c \to \widetilde{W}^{avg}$, $\overline{P}:\ \widetilde{W}^c \to \widetilde{W}^{avg}$
∇v	gradient of scalar field v, $\nabla v = \left(\frac{\partial v}{\partial x_1}, \frac{\partial v}{\partial x_2}, \ldots, \frac{\partial v}{\partial x_n}\right)^T$
$\nabla\cdot\mathbf{v}$	divergence of vector field $\mathbf{v}$, $\nabla\cdot\mathbf{v} = \frac{\partial v_1}{\partial x_1} + \frac{\partial v_2}{\partial x_2} + \cdots + \frac{\partial v_n}{\partial x_n}$
$\nabla\mathbf{v}$	componentwise gradient, $\nabla\mathbf{v} = (\nabla v_1, \nabla v_2, \ldots, \nabla v_n)^T$
$\nabla\mathbf{u} : \nabla\mathbf{v}$.....	componentwise scalar product, $\nabla\mathbf{u} : \nabla\mathbf{v} = \nabla u_1 \cdot \nabla v_1 + \nabla u_2 \cdot \nabla v_2 + \cdots + \nabla u_n \cdot \nabla v_n$
Δv	Laplacian of scalar field v, $\Delta v = \frac{\partial^2 v}{\partial x_1^2} + \frac{\partial^2 v}{\partial x_2^2} + \cdots + \frac{\partial^2 v}{\partial x_n^2}$
$\Delta\mathbf{u}$	componentwise Laplacian, $\Delta\mathbf{u} = (\Delta u_1, \Delta u_2, \ldots, \Delta u_n)^T$

Abbreviations

Re	Reynolds number
SPD	symmetric positive definite
PDE	partial differential equations
FEM	finite element method
CFD	computational fluid dynamics
SUPG	Streamline/Upwind Petrov-Galerkin
BB	Babuška-Brezzi condition
GLS	Galerkin/Least Squares
semiGLS	semi Galerkin/Least Squares
DD	domain decomposition
FETI	Finite Element Tearing and Interconnecting
FETI-DP	Finite Element Tearing and Interconnecting, Dual–Primal
BDD	Balancing Domain Decomposition
BDDC	Balancing Domain Decomposition by Constraints
PCG	Preconditioned Conjugate Gradients
MUMPS	Multifrontal Massivelly Parallel sparse direct Solver
AEE	a posteriori error estimator
w.r.t.	with respect to

Chapter 1

Introduction

The overwhelming spread of computers has ignited or accelerated many areas of studies connected somehow to them, among many others, computational mathematics, computational physics, and computational mechanics. The finite element method (FEM) has become one of the most remarkable members of this family, introduced nearly at the same time as computers and still being the leading method for simulations of physical phenomena described by partial differential equations (PDE).

Two main applications of FEM in mechanical engineering have emerged during the years and have kept in the centre of interest – structural analysis of various mechanical parts has been performed by FEM from the beginning of the method, while analysis of various flows of fluids represents slightly younger application. This delay was caused mainly by the generally higher complexity of physical models behind the latter analysis.

Hand in hand with the advance of computers, demands of simulations have grown constantly, allowing successful solution of problems with larger sizes and better resolution.

It has become clear at the end of the last century, that the answer of computer vendors to these growing demands on computers of the twenty-first century will be based on dividing the computational work among several processors, and parallel computers based on this approach have become commonly available (e.g. by processors with multiple cores).

This tendency attracted the attention of numerical matematics to the development of algorithms especially suited for parallel processing. In the context of solving PDE, the family of successful methods generally called *domain decomposition* has rapidly evolved and started to play the leading role of such algorithms.

When application of the FEM in linear structure mechanics is now clear enough for solving common tasks in mechanical engineering, computational fluid dynamics (CFD) still includes amount of open and not well handled problems. One of them is reliable modelling of incompressible flows at high Reynolds numbers, which often appear in engineering practice. Domain decomposition methods together with recent parallel computers with large number of processors open new attractive possibilities for CFD.

The main concern of the presented Ph.D. thesis is modelling of incompressible viscous flows. Ways of stabilization of the standard FEM are studied. Domain decomposition methods are discussed as a promising way of parallelization of problems in mechanics of both solids and fluids.

***

The topic of FEM in CFD is studied in a large number of publications. I mention only the most important monographs here.

GIRAULT AND RAVIART [25] present a pioneering monograph on the subject. A comprehensive theory of mixed finite elements, a successful approach to the solution of problems in fluid mechanics, is presented by BREZZI AND FORTIN [5]. A list of finite elements satisfying the Babuška-Brezzi stability condition is presented in the book. A detailed description of FEM for incompressible fluids is given by GRESHO AND SANI [28]. GLOWINSKI [26] describes the same subject and additionally presents the method of operator splitting. Finally, ELMAN, SILVESTER, AND WATHEN [16] present interesting connections of discretizations of flow problems by FEM and modern iterative solution techniques of linear algebra. CHORIN AND MARSDEN [11] present a nice introduction to mathematical models for fluids. Theoretical questions of existence and uniqueness of solution to Stokes and Navier-Stokes equations are discussed by TEMAM [56].

The idea of stabilizing the FEM is not quite new in comparison to the history of using FEM for flow problems. Several researchers have been involved in this area and already presented a number of techniques and results. Some of them have provided theroretical basis for the problem from the mathematical point of view, others presented often better results but usually with quite unclear background. Stabilization techniques were mainly motivated by two issues, that appear when the standard FEM is applied to flow problems: i) problems with convergence on elements using equal order approximation for both velocities and pressure, ii) problems with convergence at high Reynolds numbers. While the former problem is connected to the violation of Babuška-Brezzi (BB) stability condition, a theoretical result limiting proper choices of velocity and pressure approximations (see e.g. BREZZI AND FORTIN [5] or ELMAN, SILVESTER, AND WATHEN [16]), the latter is caused by the nonlinear advective terms in the equations.

Let us review several publications related to the area. HUGHES, FRANCA, AND BALESTRA [31] presented the stable Petrov-Galerkin formulation of the Stokes problem in 1987. The paper introduced the notion of residual stabilization methods into fluid mechanics and became the main reference of the field. DOUGLAS AND WANG [14] introduced an alternative stabilized method for the Stokes problem in 1989. However, both papers were devoted to the Stokes equations and thus answered only the issue of violating the Babuška-Brezzi condition by using equal order approximation. In the same year, HUGHES, FRANCA, AND HULBERT [32] presented Streamline/Upwind Petrov-Galerkin (SUPG) and Galerkin/Least Squares (GLS) stabilized finite element methods for the advective-diffusive equation. While the former was a generalization of HUGHES, FRANCA, AND BALESTRA [31], the latter represented a novel approach to stabilization terms. This work covered both above-mentioned issues of FEM for fluid mechanics. Their ideas were extended to the Navier-Stokes equations by FRANCA AND FREY [20], however, still missing theoretical background. The convergence theory of GLS for linearized Navier-Stokes equations was presented by FRANCA AND HUGHES [21]. However, all these formulations depended on some *stabilization parameters*, constants, that were necessary to be chosen a priori, but otherwise unclear. FRANCA AND MADUREIRA [23] provided an elegant way of determining these parameters for each element in 1993. While the method has proved to be quite successful in applications, the meaning of the stabilizing terms still remained unclear. An explanation of the meaning of stabilizing terms was presented by HUGHES [30]. It is based on relating the stabilized techniques to multiscale modelling and identifying the stabilized terms with sub-grid scale effects. The similarity to the method of residual free bubbles (e.g. FRANCA ET AL. [22]) is also mentioned. Work of Tezduyar on Pressure Stabilizing Petrov-Galerkin (PSPG) method (e.g. TEZDUYAR AND SATHE [57]) and work of Lube and his co-workers (e.g. GELHARD ET AL. [24]) represent recent research in the field of stabilization of the FEM for fluid dynamics. GLOWINSKI [26] investigates another approach to stabilization. It uses splitting of the Navier-Stokes equations into the Stokes problem and the advective-diffusive equation.

Our *semiGLS* method was presented by ŠÍSTEK [53] and BURDA, NOVOTNÝ, AND ŠÍSTEK [8]

as a modification of the GLS method following HUGHES AND FRANCA, AND HULBERT [32]. It was further analyzed in the aspects of accuracy using a posteriori error estimates in our papers by BURDA, NOVOTNÝ, AND ŠÍSTEK [9, 10]. These results are presented in the thesis.

***

The general idea of *domain decomposition* (DD) methods is simple: divide the physical domain into smaller subsets (called *subdomains* or *substructures*) and transform the problem described by PDE defined on the whole domain to several problems defined on these subsets. In the context of finite elements, this division corresponds to division of the computational mesh into several submeshes.

The role of subdomains has varied during the time. An early use of this idea in FEM computations might be traced back to the time, when the size of discretized problems reached the point where it was impossible to solve them in computer memory by direct methods. Disjoint subdomains, also called *superelements*, were then used to define the *interface*, i.e. the set of nodes common to more than one subdomain, and the problem was solved using *static condensation* of unknowns. Let us note, that already the team of Zlámal applied the idea of condensation of parameters (see [38]). First, unknowns in interior of each subdomain were eliminated, and the *Schur complement* matrix and *reduced right hand side* with respect to the interface was created. These matrices and right hand sides were then assembled to form the global interface problem, much smaller than the original problem and thus solvable on computers of those days. Once the solution on the interface was found, the solution in subdomain interiors was resolved by solving *Dirichlet* problems on each subdomain with prescribed values on the interface.

While this practical trick helped engineers to fit into computer memory with larger problems, a mathematical context was hidden until also the interface problem reached the limits of direct solvers and the transition to iterative methods was necessary. Then, the condition number of the systems got into the centre of interest. It is well known (e.g. BRENNER AND SCOTT [3]), that the condition number of the global problem grows as $O\left(h^{-2}\right)$ for $h \to 0$, where h is the characteristic size of the element in the triangulation. However, mathematical analysis revealed another pleasant aspect of using static condensation – condition number of the resulting Schur complement matrix of the interface problem grows only as $O\left(H^{-1}h^{-1}\right)$, where $H >> h$ is the characteristic size of the substructure (cf. BRENNER [2]). Once the Schur complement systems on the interface started to be solved iteratively, a question of a proper preconditioner was raised, and such methods opened the field of *iterative substructuring*.

Krylov subspace methods, especially preconditioned conjugate gradient method (PCG) introduced by CONCUS, GOLUB, AND O'LEARY [12], gained a lot of popularity and dominate in the field nowadays. When using such methods for solving the problem with Schur complement, only the multiplication of a vector by this matrix is needed. However, one can compute this product without explicit construction of the matrix, which is often too expensive. Instead, a Dirichlet problem on each subdomain is solved in each iteration.

There has been a challenge to construct an optimal preconditioner for the Schur complement problem, that would be scalable and provide resulting condition number of preconditioned operator bounded independently of number of subdomains. Two methods introduced in early 1990s have been particularly successful in fulfilling these goals: the Finite Element Tearing and Interconnecting (FETI) method (FARHAT AND ROUX [19]), and the Balancing Domain Decomposition (BDD) method (MANDEL [43]). While BDD is a preconditioner for the Schur complement system on the interface, where continuity of unknowns on the interface is enforced strongly (*primal* approach), the FETI method enforces the continuity only weakly by *Langrange multipliers* (*dual* approach),

and the Schur complement system on the interface is substituted by dual problem for Lagrange multipliers of the same size. While both methods have got very popular, they have a common drawback limiting their connection to standard FEM software – they both require solution of singular systems on subdomains, a functionality not always available in standard sparse direct solvers used for the realization of these preconditioners.

This necessity was quite recently resolved by newer counterparts of these methods: FETI-DP method (FARHAT ET AL. [17]) and BDDC method (DOHRMANN [13]). The FETI-DP method enforced the continuity of the degrees of freedom at substructure *corners* as in the primal method by representing them by one common variable, while the remaining continuity conditions between the substructures are enforced by Lagrange multipliers. In 2D, the FETI-DP method was proved to have condition number bounded as $O\left(\log^2\left(1+H/h\right)\right)$ by MANDEL AND TEZAUR [50]. However, the method does not converge as well in 3D and generalized continuity constraints, such as averages over *edges* or *faces* of substructures, need to be added as *coarse degrees of freedom* for fast convergence. This fact was first observed experimentally by FARHAT, LESOINNE, AND PIERSON [18], and confirmed theoretically by KLAWONN, WIDLUND, AND DRYJA [37], where the $O\left(\log^2\left(1+H/h\right)\right)$ bound was proved again for 3D considering such constraints.

The Balancing Domain Decomposition by Constraints (BDDC) proposed by DOHRMANN [13] is a primal alternative to the FETI-DP method. The BDDC method imposes the equality of coarse degrees of freedom at corners and of averages by constraints. The bound $O\left(\log^2\left(1+H/h\right)\right)$ for BDDC was first proved by MANDEL AND DOHRMANN [45].

In an important contribution to the theory of these preconditioners, MANDEL, DOHRMANN, AND TEZAUR [46] proved, that the spectra of preconditioned operators of BDDC and FETI-DP are identical (except eigenvalue 1). Simpler proofs of this equality were later presented by LI AND WIDLUND [42], and BRENNER AND SUNG [4]. The first attempt to apply BDDC to the Stokes problem was proposed by LI AND WIDLUND [41]. Although the optimal preconditioning properties of BDDC were recovered, the approach is rather complicated.

So far, only *nonoverlapping* methods were mentioned, characterized by disjoint partitions into subdomains. An important class of methods has evolved for partitions with overlaps, called *overlapping* domain decomposition methods. Although attractive condition number bounds of order $O\left(\left(1+H/\delta\right)^2\right)$, where δ is the size of the overlap, are known for such methods (cf. BRENNER AND SCOTT [3]), these techniques usually suffer from the obvious difficulty of creating the overlap on general unstructured meshes. Since the goal of the thesis is a study of nonoverlapping DD methods, references to overlapping methods are not reviewed.

Nowadays, several monographs devoted to domain decomposition are available. An introduction to domain decomposition methods is presented by SMITH, BJØRSTAD, AND GROPP [55]. QUARTERONI AND VALLI [51] present several interesting applications of domain decomposition to problems of fluid mechanics. A comprehensive summary of theoretical results in the field is presented by TOSELLI AND WIDLUND [58]. Finally, a more practical point of view on domain decomposition methods is presented by KRUIS [39]. A short but interesting chapter on domain decomposition methods was also added to the second edition by BRENNER AND SCOTT [3, Chapter 7].

***

The main goal of this work is to develop a stabilized method suitable for finite elements satisfying the Babuška-Brezzi stability condition and to apply it to flows at high Reynolds numbers. A way of quantifying the loss of accuracy inherited to the stabilized method should be proposed.

A consequent goals of the thesis are the development of an efficient parallel finite element algorithm based on the BDDC method by DOHRMANN [13], suitable for problems with symmetric positive definite matrices (SPD), such as problems of linear elasticity, and investigation of possible extensions of the method to indefinite problems of fluid mechanics.

The rest of the thesis is organized in the following way. Incompressible viscous flow described by the Stokes and the Navier-Stokes system of equations are introduced in Chapter 2. In this chapter, weak formulations for these systems are derived for both steady and unsteady cases. In Chapter 3, FEM formulations based on the mixed methods are derived for the Stokes and the Navier-Stokes problem using Taylor-Hood finite elements. Difficulties accompanying numerical solution, especially the Babuška-Brezzi stability condition are discussed. The *semiGLS* stabilization technique, a modification of the GLS method by HUGHES, FRANCA, AND HULBERT [32], is derived in Chapter 4, the principal part of the thesis. In this Chapter, our previous work on this topic published by BURDA, NOVOTNÝ, AND ŠÍSTEK [8, 9, 10], and ŠÍSTEK [53] is recalled and summarized. Two ways of evaluating the accuracy of the stabilization technique are proposed. A posteriori error estimates for the Navier-Stokes equations are found as a useful tool for this purpose. The stabilization is applied to the incompressible Navier-Stokes equations and several numerical results are presented in Chapter 7. Chapter 5 is devoted to the formulation of the BDDC preconditioner. A reformulation of the algorithm in comparison to the original paper by DOHRMANN [13] is suggested, so that it better fits the desired parallel algorithms with connection to averages over subdomain edges and faces. Two independent parallel algorithms of the preconditioner are described in Chapter 6. In Chapter 7, several numerical tests are presented to verify the algorithm on symmetric positive definite problems arising from linear elasticity, and the applicability is then investigated on several experiments with the Stokes flow. Chapter 8 summarizes main achievements of the presented work and proposes topics for further research.

Chapter 2

Models of incompressible viscous flow

Let Ω be an open bounded domain in $\mathbb{R}^2$ filled with an incompressible viscous fluid, and let $\partial\Omega$ be its boundary. The generic point of $\mathbb{R}^2$ is denoted by $\mathbf{x} = (x_1, x_2)^T$,

2.1 The Navier-Stokes equations

Flow of Newtonian fluids with constant density is modelled by the following *unsteady Navier-Stokes system* of partial differential equations (nonconservative form)

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u} - \nu\Delta\mathbf{u} + \nabla p = \mathbf{f} \quad \text{in } \Omega\times[0,T], \tag{2.1}$$

$$\nabla\cdot\mathbf{u} = 0 \quad \text{in } \Omega\times[0,T], \tag{2.2}$$

where

- t denotes time variable,
- $\mathbf{u} = (u_1, u_2)^T$ denotes the vector of fluid velocity,
- p denotes the pressure divided by the density,
- ν denotes the kinematic viscosity of the fluid supposed to be constant,
- $\mathbf{f}$ denotes the density of volume forces per mass unit.

The system introduced is not sufficient to define a flow since it has an infinity of solutions. To restrict the number of solutions, we have to consider further conditions, such as the initial condition

$$\mathbf{u} = \mathbf{u}_0 \ \text{ in } \Omega,\ t = 0, \tag{2.3}$$

where $\mathbf{u}_0$ is a given flow field satisfying $\nabla\cdot\mathbf{u}_0 = 0$, and the boundary conditions

$$\mathbf{u} = \mathbf{g} \ \text{ on } \partial\Omega_g\times[0,T], \tag{2.4}$$

$$-\nu(\nabla\mathbf{u})\mathbf{n} + p\mathbf{n} = \mathbf{0} \ \text{ on } \partial\Omega_{dn}\times[0,T], \tag{2.5}$$

where

- $\partial\Omega_g$ and $\partial\Omega_{dn}$ are two subsets of $\partial\Omega$ satisfying $\overline{\partial}\Omega = \overline{\partial}\Omega_g \cup \overline{\partial}\Omega_{dn},\ \mu_{\mathbb{R}^1}(\partial\Omega_g \cap \partial\Omega_{dn}) = 0$,
- $\mathbf{n}$ denotes an outer normal vector to the boundary $\partial\Omega$ with unit lenght,

- $\mathbf{g}$ is a given function satisfying $\int_{\partial\Omega} \mathbf{g} \cdot \mathbf{n} \mathrm{d}\partial\Omega = 0$ in the case of $\partial\Omega = \partial\Omega_g$.

If the solution to system (2.1)–(2.2) has a steady limit for $t \to \infty$, the time derivative in (2.1) vanishes in the limit and the state can be modelled by the *steady Navier-Stokes* system

$$(\mathbf{u} \cdot \nabla)\mathbf{u} - \nu\Delta\mathbf{u} + \nabla p = \mathbf{f} \text{ in } \Omega, \tag{2.6}$$
$$\nabla \cdot \mathbf{u} = 0 \text{ in } \Omega, \tag{2.7}$$

and boundary conditions

$$\mathbf{u} = \mathbf{g} \text{ on } \partial\Omega_g, \tag{2.8}$$
$$-\nu(\nabla\mathbf{u})\mathbf{n} + p\mathbf{n} = \mathbf{0} \text{ on } \partial\Omega_{dn}. \tag{2.9}$$

Initial condition is not present, and $\mathbf{u}$, p, $\mathbf{f}$, and $\mathbf{g}$ are no more functions of time.

2.2 The Stokes equations

The Stokes problem is an important simplification of the Navier-Stokes system, that corresponds to linearization of the momentum equation (2.1) around zero velocity. Such simplification is justified for diffusion-dominated flows, such as flows at very low Reynolds number or in regions, where the speed is reduced (e.g. boundary layers). The nonlinear convection term of Navier-Stokes equations is neglected and the *unsteady Stokes problem* reads

$$\frac{\partial\mathbf{u}}{\partial t} - \nu\Delta\mathbf{u} + \nabla p = \mathbf{f} \text{ in } \Omega \times [0,T], \tag{2.10}$$
$$\nabla \cdot \mathbf{u} = 0 \text{ in } \Omega \times [0,T], \tag{2.11}$$
$$\mathbf{u} = \mathbf{u}_0 \text{ in } \Omega,\ t = 0, \tag{2.12}$$
$$\mathbf{u} = \mathbf{g} \text{ on } \partial\Omega_g \times [0,T], \tag{2.13}$$
$$-\nu(\nabla\mathbf{u})\mathbf{n} + p\mathbf{n} = \mathbf{0} \text{ on } \partial\Omega_{dn} \times [0,T]. \tag{2.14}$$

Due to the absence of the nonlinear term in momentum equation (2.10), the problem has a unique steady solution for time $t \to \infty$ [56]. If we are not investigating the transient behaviour of the flow, we could find this state directly by solving the *steady Stokes problem*

$$-\nu\Delta\mathbf{u} + \nabla p = \mathbf{f} \text{ in } \Omega, \tag{2.15}$$
$$\nabla \cdot \mathbf{u} = 0 \text{ in } \Omega, \tag{2.16}$$
$$\mathbf{u} = \mathbf{g} \text{ on } \partial\Omega_g, \tag{2.17}$$
$$-\nu(\nabla\mathbf{u})\mathbf{n} + p\mathbf{n} = \mathbf{0} \text{ on } \partial\Omega_{dn}. \tag{2.18}$$

2.3 Weak formulation of Stokes and Navier-Stokes problems

Although the weak solution to the Navier-Stokes problem is much more difficult to analyze than in the (linear) Stokes case, the standard notion of the weak solution is introduced similarly for both problems. For this reason, the derivation of the weak problem is presented only for the most general case considered, i.e. the time-dependent Navier-Stokes problem, and it is abbreviated for the other problems.

We need to introduce several function spaces for further derivations. Note, that all integrals are considered in the Lebesque sense. Let $L_2(\Omega)$ be the space of square integrable functions on Ω,

and let $L_2(\Omega)/\mathbb{R}$ be the space of functions in $L_2(\Omega)$ ignoring an additive constant. Let $H^1(\Omega)$ and $H_0^1(\Omega)$ be the Sobolev spaces defined as

$$\begin{aligned} H^1(\Omega) &= \left\{ v \mid v \in L_2(\Omega), \frac{\partial v}{\partial x_i} \in L_2(\Omega), i = 1, 2 \right\}, \\ H_0^1(\Omega) &= \left\{ v \mid v \in H^1(\Omega), \mathbf{Tr}\, v = 0 \right\}, \end{aligned}$$

where $\mathbf{Tr}$ is the trace operator $\mathbf{Tr}: H^1(\Omega) \longrightarrow L_2(\partial\Omega_g)$, and derivatives are considered in the weak sense.

The norm of function v in the space $L_2(\Omega)$ is considered as

$$\|v\|^2_{L_2(\Omega)} = \int_\Omega v^2 \mathrm{d}\Omega,$$

and the norm of function v in the Sobolev space $H^1(\Omega)$ is considered as

$$\|v\|^2_{H^1(\Omega)} = \int_\Omega \left(v^2 + \sum_{k=1}^{2} \left(\frac{\partial v}{\partial x_k} \right)^2 \right) \mathrm{d}\Omega.$$

Sometimes, the notation $\|\cdot\|_{L_2(\Omega)}$ is shortened to $\|\cdot\|_0$ and $\|\cdot\|_{H^1(\Omega)}$ to $\|\cdot\|_1$.

Let us define vector function spaces V_g and V by

$$\begin{aligned} V_g &= \left\{ \mathbf{v} = (v_1, v_2)^T \mid \mathbf{v} \in [H^1(\Omega)]^2; \mathbf{Tr}\, v_i = g_i, i = 1, 2 \right\}, \\ V &= \left\{ \mathbf{v} = (v_1, v_2)^T \mid \mathbf{v} \in [H_0^1(\Omega)]^2 \right\}. \end{aligned}$$

Let us note, that the norm of vector function $\mathbf{v}$ in the space V_g and V is then

$$\|\mathbf{v}\|^2_{[H^1(\Omega)]^2} = \sum_{i=1}^{2} \int_\Omega \left(v_i^2 + \sum_{k=1}^{2} \left(\frac{\partial v_i}{\partial x_k} \right)^2 \right) \mathrm{d}\Omega,$$

and the norm of vector function $\mathbf{v}$ in the space $[L_2(\Omega)]^2$ is

$$\|\mathbf{v}\|^2_{[L_2(\Omega)]^2} = \sum_{i=1}^{2} \int_\Omega v_i^2 \mathrm{d}\Omega.$$

Let us first derive the weak formulation of the Navier-Stokes equations (2.1)–(2.5) in the way of mixed methods, i.e. usage of different function spaces of test functions for the momentum equation and for the continuity equation (cf. [5]). To derive it, we suppose for a while, that the functions appearing in the system are sufficiently smooth. Then for any $t \in [0, T]$, we have

$$\int_\Omega \frac{\partial \mathbf{u}}{\partial t} \cdot \mathbf{v} \mathrm{d}\Omega + \int_\Omega (\mathbf{u} \cdot \nabla)\mathbf{u} \cdot \mathbf{v} \mathrm{d}\Omega - \nu \int_\Omega \Delta \mathbf{u} \cdot \mathbf{v} \mathrm{d}\Omega + \int_\Omega \nabla p \cdot \mathbf{v} \mathrm{d}\Omega = \int_\Omega \mathbf{f} \cdot \mathbf{v} \mathrm{d}\Omega, \quad (2.19)$$

$$\int_\Omega \psi \nabla \cdot \mathbf{u} \mathrm{d}\Omega = 0, \quad (2.20)$$

$$\mathbf{u} - \mathbf{u}_g \in V, \quad (2.21)$$

for any $\mathbf{v} \in V$ and $\psi \in L_2(\Omega)$. Here $\mathbf{u}_g \in V_g$ is a representation of the Dirichlet boundary condition $\mathbf{g}$ in (2.4). We assume $\mathbf{g} \in [L_2(\partial\Omega_g)]^2$ and $\mathbf{f} \in [L_2(\Omega)]^2$.

Using Green's formula on the third and the fourth term of equation (2.19), we obtain

$$\begin{aligned}\int_\Omega \frac{\partial \mathbf{u}}{\partial t}\cdot\mathbf{v}\mathrm{d}\Omega + \int_\Omega (\mathbf{u}\cdot\nabla)\mathbf{u}\cdot\mathbf{v}\mathrm{d}\Omega - \nu\int_{\partial\Omega}(\nabla\mathbf{u})\mathbf{v}\cdot\mathbf{n}\mathrm{d}\partial\Omega &+ \nu\int_\Omega \nabla\mathbf{u}:\nabla\mathbf{v}\mathrm{d}\Omega + \\ + \int_{\partial\Omega} p\mathbf{v}\cdot\mathbf{n}\mathrm{d}\partial\Omega - \int_\Omega p\nabla\cdot\mathbf{v}\mathrm{d}\Omega &= \int_\Omega \mathbf{f}\cdot\mathbf{v}\mathrm{d}\Omega, \qquad (2.22)\\ \int_\Omega \psi\nabla\cdot\mathbf{u}\mathrm{d}\Omega &= 0, \qquad (2.23)\\ \mathbf{u}-\mathbf{u}_g &\in V. \qquad (2.24)\end{aligned}$$

The integrals over boundary in (2.22) vanish for considered boundary conditions. Here, $\nabla\mathbf{u}:\nabla\mathbf{v}$ represents the componentwise scalar product $\nabla u_1\cdot\nabla v_1 + \nabla u_2\cdot\nabla v_2$.

Then the *weak formulation of the unsteady Navier-Stokes problem* reads:
Find of $\mathbf{u}(t) = (u_1(t), u_2(t))^T \in V_g$ and $p(t)\in L_2(\Omega)/\mathbb{R}$ satisfying for any $t\in[0,T]$

$$\begin{aligned}\int_\Omega \frac{\partial \mathbf{u}}{\partial t}\cdot\mathbf{v}\mathrm{d}\Omega + \int_\Omega (\mathbf{u}\cdot\nabla)\mathbf{u}\cdot\mathbf{v}\mathrm{d}\Omega + \nu\int_\Omega \nabla\mathbf{u}:\nabla\mathbf{v}\mathrm{d}\Omega - \int_\Omega p\nabla\cdot\mathbf{v}\mathrm{d}\Omega &= \int_\Omega \mathbf{f}\cdot\mathbf{v}\mathrm{d}\Omega, \qquad (2.25)\\ \int_\Omega \psi\nabla\cdot\mathbf{u}\mathrm{d}\Omega &= 0, \qquad (2.26)\\ \mathbf{u}-\mathbf{u}_g &\in V, \qquad (2.27)\end{aligned}$$

for $\mathbf{v}\in V$ and $\psi\in L_2(\Omega)$.

Similarly, the *weak formulation of the steady Navier-Stokes problem* reads:
Find $\mathbf{u} = (u_1, u_2)^T \in V_g$ and $p\in L_2(\Omega)/\mathbb{R}$ satisfying

$$\begin{aligned}\int_\Omega (\mathbf{u}\cdot\nabla)\mathbf{u}\cdot\mathbf{v}\mathrm{d}\Omega + \nu\int_\Omega \nabla\mathbf{u}:\nabla\mathbf{v}\mathrm{d}\Omega - \int_\Omega p\nabla\cdot\mathbf{v}\mathrm{d}\Omega &= \int_\Omega \mathbf{f}\cdot\mathbf{v}\mathrm{d}\Omega, \qquad (2.28)\\ \int_\Omega \psi\nabla\cdot\mathbf{u}\mathrm{d}\Omega &= 0, \qquad (2.29)\\ \mathbf{u}-\mathbf{u}_g &\in V, \qquad (2.30)\end{aligned}$$

for $\mathbf{v}\in V$ and $\psi\in L_2(\Omega)$.

Repeating the same procedure for the unsteady Stokes problem (2.10)–(2.14), and the steady Stokes problem (2.15)–(2.18), respectively, leads to the *weak formulation of the unsteady Stokes problem*:
Find $\mathbf{u}(t) = (u_1(t), u_2(t))^T \in V_g$ and $p(t)\in L_2(\Omega)/\mathbb{R}$ satisfying for any $t\in[0,T]$

$$\begin{aligned}\int_\Omega \frac{\partial \mathbf{u}}{\partial t}\cdot\mathbf{v}\mathrm{d}\Omega + \nu\int_\Omega \nabla\mathbf{u}:\nabla\mathbf{v}\mathrm{d}\Omega - \int_\Omega p\nabla\cdot\mathbf{v}\mathrm{d}\Omega &= \int_\Omega \mathbf{f}\cdot\mathbf{v}\mathrm{d}\Omega, \qquad (2.31)\\ \int_\Omega \psi\nabla\cdot\mathbf{u}\mathrm{d}\Omega &= 0, \qquad (2.32)\\ \mathbf{u}-\mathbf{u}_g &\in V, \qquad (2.33)\end{aligned}$$

for $\mathbf{v}\in V$ and $\psi\in L_2(\Omega)$, and to the *weak formulation of the steady Stokes problem*:
Find $\mathbf{u} = (u_1, u_2)^T \in V_g$ and $p\in L_2(\Omega)/\mathbb{R}$ satisfying

$$\begin{aligned}\nu\int_\Omega \nabla\mathbf{u}:\nabla\mathbf{v}\mathrm{d}\Omega - \int_\Omega p\nabla\cdot\mathbf{v}\mathrm{d}\Omega &= \int_\Omega \mathbf{f}\cdot\mathbf{v}\mathrm{d}\Omega, \qquad (2.34)\\ \int_\Omega \psi\nabla\cdot\mathbf{u}\mathrm{d}\Omega &= 0, \qquad (2.35)\\ \mathbf{u}-\mathbf{u}_g &\in V, \qquad (2.36)\end{aligned}$$

for $\mathbf{v}\in V$ and $\psi\in L_2(\Omega)$.

Chapter 3

Mixed finite element method for flow problems

In this chapter, an oveview of the finite element method used for the solution of the Stokes and Navier-Stokes problem is presented. As in the previous chapter, due to the similarity of finite element spaces used for both problems, the derivations apply to both unless othewise noted.

Let us divide the domain Ω (supposed now polygonal) into N elements T_K of a shape regular triangulation $\mathcal{T}$ such that

$$\begin{aligned} \bigcup_{K=1}^{N} \overline{T}_K &= \overline{\Omega}, \\ \mu_{\mathbb{R}^2}\left(T_K \cap T_L\right) &= 0, K \neq L. \end{aligned}$$

Let $h_K = \mathrm{diam}(T_K)$ denote the largest distance in element T_K.

3.1 Function spaces for velocity and pressure approximation

The choice of function spaces for velocity and pressure approximation is not arbitrary in solving Navier-Stokes as well as Stokes equations by the FEM. It is often useful to choose different polynomial approximation for velocities and for pressure. Although equal order approximation is easier to implement, pressure exhibits instability. Therefore, approximation with different order is more suitable for practical computing, cf. [5, 16].

The following properties of desired solution follow from the weak formulations (2.25)–(2.27), (2.28)–(2.30), (2.31)–(2.33), and (2.34)–(2.36): i) each component of velocity is a square integrable function of $\mathbf{x}$, and at least its first weak derivative by any coordinate exists; ii) pressure is a square integrable function of $\mathbf{x}$.

The polynomial order of approximation of velocity on an element is not independent of approximation of pressure. Babuška and Brezzi derived an inequality limiting these approximations

$$\exists C_B > 0, const.\ \forall q_h \in Q_h\ \exists \mathbf{v}_h \in V_{gh}\ (q_h, \nabla \cdot \mathbf{v}_h)_0 \geq C_B \|q_h\|_0 \|\mathbf{v}_h\|_1, \tag{3.1}$$

where Q_h and V_{gh} are the function spaces for approximation of pressure and velocity. This is an important condition necessary in the proof of uniqueness of pressure for Stokes problem. It has been shown, that severe difficulties have to be overcome when using approximations, which do not satisfy the Babuška-Brezzi (abbreviated to BB) condition, cf. e.g. [5]. There are several

finite elements (in 2D as well as in 3D) which do satisfy the BB-condition. Following list is not complete.

Finite elements satisfying the Babuška-Brezzi condition (cf. [5])

- $P_1^+ P_1$ (mini element)
- $P_2 P_1$ (Taylor-Hood, 1973)
- $P_2^+ P_1$
- $P_2^+ P_{-1}$ (Crouzeix-Raviart)
- $Q_2 Q_1$ (Taylor-Hood)
- $Q_2 P_{-1}$

3.2 Taylor-Hood finite elements

Due to their application in this work, Taylor-Hood finite elements on triangles and quadrilaterals are described more precisely. Values of velocity are aproximated at corner nodes and at centres of edges, while pressure variables are at corner nodes only (Figure 3.1). This corresponds to the following function spaces on element T_K:

- *triangle*

$$\begin{aligned} v_i &\in P_2(T_K),\ i = 1, 2, \quad \text{i.e. polynomial of the second order,} \\ p &\in P_1(T_K) \quad \text{i.e. linear polynomial.} \end{aligned}$$

- *quadrilateral*

$$\begin{aligned} v_i &\in Q_2(T_K),\ i = 1, 2, \quad \text{i.e. polynomial of the second order for each coordinate,} \\ p &\in Q_1(T_K) \quad \text{i.e. bilinear polynomial.} \end{aligned}$$

The approximation leads to the following shape functions written in local coordinate system $\{\xi, \eta\}$:

- *triangle*
  Shape functions for approximation of velocity component:

$$\begin{aligned} N_1 &= 1/2 \cdot (2 - \xi - \eta)(1 - \xi - \eta) \\ N_2 &= 1/2 \cdot \xi(\xi - 1) \\ N_3 &= 1/2 \cdot \eta(\eta - 1) \\ N_4 &= \xi(2 - \xi - \eta) \\ N_5 &= \xi\eta \\ N_6 &= \eta(2 - \xi - \eta) \end{aligned}$$

  Shape functions for approximation of pressure

$$\begin{aligned} M_1 &= 1/2 \cdot (2 - \xi - \eta) \\ M_2 &= 1/2 \cdot \xi \\ M_3 &= 1/2 \cdot \eta \end{aligned}$$

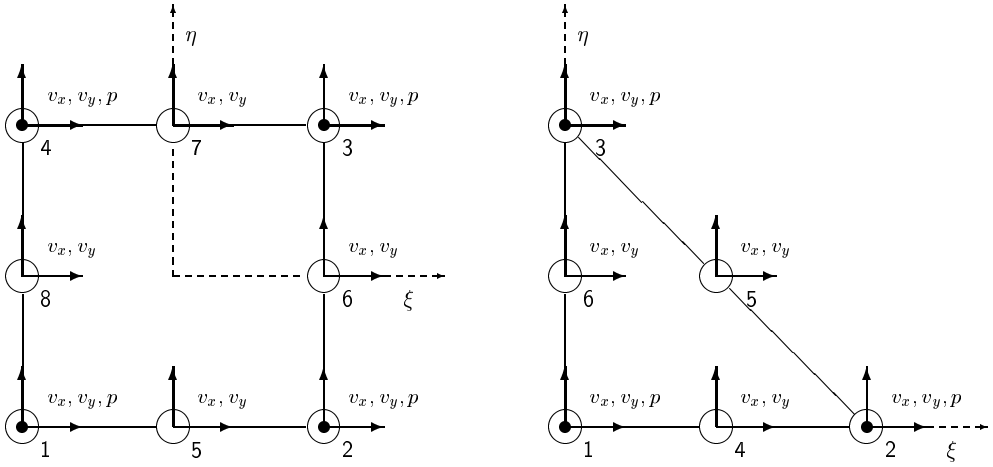


Figure 3.1: Taylor-Hood reference elements

- *quadrilateral*
  Shape functions for approximation of velocity component

$$\begin{aligned}
N_1 &= 1/4 \cdot (1-\xi)(1-\eta)(-\xi-\eta-1) \\
N_2 &= 1/4 \cdot (1+\xi)(1-\eta)(\xi-\eta-1) \\
N_3 &= 1/4 \cdot (1+\xi)(1+\eta)(\xi+\eta-1) \\
N_4 &= 1/4 \cdot (1-\xi)(1+\eta)(-\xi+\eta-1) \\
N_5 &= 1/2 \cdot (1-\xi^2)(1-\eta) \\
N_6 &= 1/2 \cdot (1-\eta^2)(1+\xi) \\
N_7 &= 1/2 \cdot (1-\xi^2)(1+\eta) \\
N_8 &= 1/2 \cdot (1-\eta^2)(1-\xi)
\end{aligned}$$

  Shape functions for approximation of pressure

$$\begin{aligned}
M_1 &= 1/4 \cdot (1-\xi)(1-\eta) \\
M_2 &= 1/4 \cdot (1+\xi)(1-\eta) \\
M_3 &= 1/4 \cdot (1+\xi)(1+\eta) \\
M_4 &= 1/4 \cdot (1-\xi)(1+\eta)
\end{aligned}$$

3.3 Discretization of steady Stokes and Navier-Stokes equations by FEM

For isoparametric finite elements, velocities and pressure on an element T_K are given as

$$\begin{aligned} v_x|_{T_K} &= \sum_{i=1}^{N_u} v_{x_i} N_i, \\ v_y|_{T_K} &= \sum_{i=1}^{N_u} v_{y_i} N_i, \\ p|_{T_K} &= \sum_{i=1}^{N_p} p_i M_i, \end{aligned}$$

where

- v_x denotes the component of velocity in the direction of x-coordinate
- v_y denotes the component of velocity in the direction of y-coordinate
- p denotes pressure
- v_{x_i} is the value of velocity v_x in node i
- v_{y_i} is the value of velocity v_y in node i
- p_i is the value of pressure in node i
- N_u is the number of nodes with value of velocity on element given (in the case of Taylor-Hood elements, $N_u = 8$ for quadrilateral and $N_u = 6$ for triangle)
- N_p is the number of nodes with value of pressure on element given (in the case of Taylor-Hood elements, $N_p = 4$ for quadrilateral, $N_p = 3$ for triangle)

Let us employ the notation

$$R_m(\overline{T_K}) = \begin{cases} P_m(\overline{T_K}), & \text{if } T_K \text{ is a triangle} \\ Q_m(\overline{T_K}), & \text{if } T_K \text{ is a quadrilateral} \end{cases},$$

and let $\mathcal{C}(\overline{\Omega})$ denote the space of continuous functions on $\overline{\Omega}$.

Application of Taylor-Hood finite elements leads to an approximation on the domain Ω satisfying $\mathbf{u}_h \in V_{gh}$ and $p_h \in Q_h$ where

$$\begin{aligned} V_{gh} &= \Big\{ \mathbf{v}_h = (v_{h_1}, v_{h_2})^T \in [\mathcal{C}(\overline{\Omega})]^2;\ v_{h_i}\,|_{T_K} \in R_2(\overline{T_K}),\ K = 1, \ldots, N,\ i = 1, 2,\ \mathbf{v}_h = \mathbf{g}_h \text{ on } \partial\Omega_g \Big\}, \\ Q_h &= \Big\{ \psi_h \in \mathcal{C}(\overline{\Omega});\ \psi_h\,|_{T_K} \in R_1(\overline{T_K}),\ K = 1, \ldots, N \Big\}. \end{aligned}$$

Further, we introduce the space of test functions

$$V_h = \Big\{ \mathbf{v}_h = (v_{h_1}, v_{h_2})^T \in [\mathcal{C}(\overline{\Omega})]^2;\ v_{h_i}\,|_{T_K} \in R_2(\overline{T_K}),\ K = 1, \ldots, N,\ i = 1, 2,\ \mathbf{v}_h = \mathbf{0} \text{ on } \partial\Omega_g \Big\}.$$

Since these function spaces satisfy $V_{gh} \subset V_g$, $V_h \subset V$, and $Q_h \subset L_2(\Omega)/\mathbb{R}$, we are now ready to introduce the *approximate steady Navier-Stokes problem* based on the weak formulation (2.28)–(2.30):

Find $\mathbf{u}_h \in V_{gh}$ and $p_h \in Q_h$ satisfying

$$\begin{aligned}
\int_\Omega (\mathbf{u}_h \cdot \nabla)\mathbf{u}_h \cdot \mathbf{v}_h \mathrm{d}\Omega + \nu \int_\Omega \nabla \mathbf{u}_h : \nabla \mathbf{v}_h \mathrm{d}\Omega - \int_\Omega p_h \nabla \cdot \mathbf{v}_h \mathrm{d}\Omega &= \int_\Omega \mathbf{f} \cdot \mathbf{v}_h \mathrm{d}\Omega, \ \forall \mathbf{v}_h \in V_h, && (3.2)\\
\int_\Omega \psi_h \nabla \cdot \mathbf{u}_h \mathrm{d}\Omega &= 0, \ \forall \psi_h \in Q_h, && (3.3)\\
\mathbf{u}_h - \mathbf{u}_{gh} &\in V_h, && (3.4)
\end{aligned}$$

where $\mathbf{u}_{gh} \in V_{gh}$ is the projection of $\mathbf{u}_g$ onto the space V_{gh}. Similarly, we derive the *approximate steady Stokes problem* from (2.34)–(2.36):
Find $\mathbf{u}_h \in V_{gh}$ and $p_h \in Q_h$ satisfying

$$\begin{aligned}
\nu \int_\Omega \nabla \mathbf{u}_h : \nabla \mathbf{v}_h \mathrm{d}\Omega - \int_\Omega p_h \nabla \cdot \mathbf{v}_h \mathrm{d}\Omega &= \int_\Omega \mathbf{f} \cdot \mathbf{v}_h \mathrm{d}\Omega, \ \forall \mathbf{v}_h \in V_h, && (3.5)\\
\int_\Omega \psi_h \nabla \cdot \mathbf{u}_h \mathrm{d}\Omega &= 0, \ \forall \psi_h \in Q_h, && (3.6)\\
\mathbf{u}_h - \mathbf{u}_{gh} &\in V_h. && (3.7)
\end{aligned}$$

3.4 Discretization of unsteady Stokes and Navier-Stokes equations by FEM

To solve the unsteady Stokes equations (2.31)–(2.33) or Navier-Stokes equations (2.25)–(2.27), we need to discretize the systems both in space and time. Let us follow the concept of the method of lines (MOL) and perform the space semidiscretization first, followed by the discretization in time.

Extending derivations for the steady case in Section 3.3, we first introduce the *semidiscrete unsteady Navier-Stokes problem* based on the weak formulation (2.25)–(2.27):

Find $\mathbf{u}_h(t) \in V_{gh},\ t \in [0,T]$ and $p_h(t) \in Q_h,\ t \in [0,T]$ satisfying

$$\begin{aligned}
\int_\Omega \frac{\partial \mathbf{u}_h}{\partial t} \cdot \mathbf{v}_h \mathrm{d}\Omega + \int_\Omega (\mathbf{u}_h \cdot \nabla)\mathbf{u}_h \cdot \mathbf{v}_h \mathrm{d}\Omega &+ \nu \int_\Omega \nabla \mathbf{u}_h : \nabla \mathbf{v}_h \mathrm{d}\Omega -\\
- \int_\Omega p_h \nabla \cdot \mathbf{v}_h \mathrm{d}\Omega &= \int_\Omega \mathbf{f} \cdot \mathbf{v}_h \mathrm{d}\Omega, \ \forall \mathbf{v}_h \in V_h, && (3.8)\\
\int_\Omega \psi_h \nabla \cdot \mathbf{u}_h \mathrm{d}\Omega &= 0, \ \forall \psi_h \in Q_h, && (3.9)\\
\mathbf{u}_h - \mathbf{u}_{gh} &\in V_h. && (3.10)
\end{aligned}$$

Let us now discretize the problem in time. We consider partition of the time interval $[0,T]$ into M time intervals with $M+1$ time layers. The time step between n-th time layer and $(n+1)$-st time layer is assumed constant and is denoted by ϑ. We employ the implicit Euler method (also known as the backward difference method), i.e. time derivative is substituted as

$$\frac{\partial \mathbf{u}_h}{\partial t} \approx \frac{\mathbf{u}_h^{n+1} - \mathbf{u}_h^n}{\vartheta}.$$

This leads to fully implicit method for seeking $\mathbf{u}_h$ in $(n+1)$-st time layer. The *discrete unsteady Navier-Stokes problem* then reads:

Find $\mathbf{u}_h^{n+1} \in V_{gh}$ and $p_h^{n+1} \in Q_h$ satisfying

$$\begin{aligned}
\frac{1}{\vartheta}\int_\Omega \mathbf{u}_h^{n+1}\cdot\mathbf{v}_h \mathrm{d}\Omega + \int_\Omega (\mathbf{u}_h^{n+1}\cdot\nabla)\mathbf{u}_h^{n+1}\cdot\mathbf{v}_h \mathrm{d}\Omega &+ \nu\int_\Omega \nabla\mathbf{u}_h^{n+1} : \nabla\mathbf{v}_h \mathrm{d}\Omega - \\
-\int_\Omega p_h^{n+1}\nabla\cdot\mathbf{v}_h \mathrm{d}\Omega - \frac{1}{\vartheta}\int_\Omega \mathbf{u}_h^{n}\cdot\mathbf{v}_h \mathrm{d}\Omega &= \int_\Omega \mathbf{f}^{n+1}\cdot\mathbf{v}_h \mathrm{d}\Omega,\ \forall \mathbf{v}_h \in V_h, \quad (3.11)\\
\int_\Omega \psi_h \nabla\cdot\mathbf{u}_h^{n+1}\mathrm{d}\Omega &= 0,\ \forall \psi_h \in Q_h, \quad (3.12)\\
\mathbf{u}_h^{n+1} - \mathbf{u}_{gh}^{n+1} &\in V_h. \quad (3.13)
\end{aligned}$$

Applying now the same derivations to problem (2.31)–(2.33), we can derive the *discrete unsteady Stokes problem*:

Find $\mathbf{u}_h^{n+1} \in V_{gh}$ and $p_h^{n+1} \in Q_h$ satisfying

$$\begin{aligned}
\frac{1}{\vartheta}\int_\Omega \mathbf{u}_h^{n+1}\cdot\mathbf{v}_h \mathrm{d}\Omega + \nu\int_\Omega \nabla\mathbf{u}_h^{n+1} : \nabla\mathbf{v}_h \mathrm{d}\Omega &- \int_\Omega p_h^{n+1}\nabla\cdot\mathbf{v}_h \mathrm{d}\Omega - \\
-\frac{1}{\vartheta}\int_\Omega \mathbf{u}_h^{n}\cdot\mathbf{v}_h \mathrm{d}\Omega &= \int_\Omega \mathbf{f}^{n+1}\cdot\mathbf{v}_h \mathrm{d}\Omega,\ \forall \mathbf{v}_h \in V_h, \quad (3.14)\\
\int_\Omega \psi_h \nabla\cdot\mathbf{u}_h^{n+1}\mathrm{d}\Omega &= 0,\ \forall \psi_h \in Q_h, \quad (3.15)\\
\mathbf{u}_h^{n+1} - \mathbf{u}_{gh}^{n+1} &\in V_h. \quad (3.16)
\end{aligned}$$

Chapter 4

SemiGLS stabilization of the finite element method

In this chapter, the *semiGLS* method of stabilization of FEM is recalled. It was introduced in [53] as a modification of the Galerkin Least-Squares (GLS) technique by Hughes, Franca and their co-workers [20, 21, 23, 31, 32]. The *semiGLS* method was further analyzed in our papers [8] and [10]. Main results are summarized in this chapter.

Throughout the chapter, the Navier-Stokes problem is considered. The purpose of the *semiGLS* stabilization is to extend the applicability of the finite element method of Chapter 3 to flows at high Reynolds numbers, which are of practical interest, but for which the standard Galerkin method fails to converge.

4.1 Formulation of the stabilized problem

Franca and Hughes analyzed in [21] a modification of the GLS method to stabilize the steady *linearized* Navier-Stokes equations given by

$$\begin{aligned} (\nabla \mathbf{u})\mathbf{a} + \nabla p - \nu \Delta \mathbf{u} &= \mathbf{f} \text{ in } \Omega, && (4.1)\\ \nabla \cdot \mathbf{u} &= 0 \text{ in } \Omega, && (4.2)\\ \mathbf{u} &= \mathbf{0} \text{ on } \partial\Omega, && (4.3) \end{aligned}$$

where $\nabla \cdot \mathbf{a} = 0$. They defined the norm on $V_{gh} \times Q_h$ as

$$|||\{\mathbf{u}, p\}|||^2 \equiv \nu \|\nabla \mathbf{u}\|^2_{0,\Omega} + \sum_K \tau \|(\nabla \mathbf{u})\mathbf{a} + \nabla p - \nu \Delta \mathbf{u}\|^2_{0,K} + \sum_K \delta \|\nabla \cdot \mathbf{u}\|^2_{0,K}, \tag{4.4}$$

where δ and τ are certain stabilization parameters, and they prove the following lemma:

Lemma 1 (Stability) *The bilinear form of the linearized problem (4.1)–(4.3) satisfies*

$$B_{GLS}(\mathbf{u}_h, p_h; \mathbf{u}_h, p_h) = |||\{\mathbf{u}_h, p_h\}|||^2.$$

For the Stokes problem, the following estimate is proved in [21]

$$|||\{\mathbf{u}_h, p_h\}|||^2 \geq \frac{\nu}{2} \|\nabla \mathbf{u}\|^2_{0,\Omega} + \frac{1}{2} \sum_K \tau \|\nabla p\|^2_{0,K} + \sum_K \delta \|\nabla \cdot \mathbf{u}\|^2_{0,K}. \tag{4.5}$$

Notice, that the choice $\delta = 0$ also gives stability. It is important for the *semiGLS* method.

Using stability, the following convergence theorem is proved in [21]:

Theorem 1 (Convergence) *Assuming constant viscosity ν, the solution $\{\mathbf{u}_h, p_h\}$ obtained by the GLS method converges to the solution $\{\mathbf{u}, p\}$ of the weak formulation of (4.1)–(4.3) as follows:*

$$\begin{aligned}|||\{\mathbf{u}_h, p_h\} - \{\mathbf{u}, p\}|||^2 &\leq C \sum_K \Bigg[H(Re_K - 1) \left(\sup_{x \in T_K} |a|_q h_K^{2k+1} |\mathbf{u}|^2_{k+1,T_K} + \sup_{x \in T_K} |a|_q^{-1} h_K^{2l+1} |p|^2_{l+1,T_K} \right) \\ &+ H(1 - Re_K) \left(\nu h_K^{2k} |\mathbf{u}|^2_{k+1,T_K} + \nu^{-1} h_K^{2l+2} |p|^2_{l+1,T_K} \right) \Bigg],\end{aligned}$$

where $H(\cdot)$ is the Heaviside function given by

$$H(x - y) = \begin{cases} 0, & x < y \\ 1, & x > y \end{cases},$$

and Re_K is local Reynolds number on element K.

The *semiGLS* method was derived from the variant of GLS presented for the Navier-Stokes equations in [23] with the following modifications: The main difference between the two is the fact, that in *semiGLS*, stabilization of the continuity equation is omitted. The *stabilized unsteady Navier-Stokes problem* is derived from the semidiscrete formulation (3.8)–(3.10) as:

Find $\mathbf{u}_h(t) \in V_{gh},\ t \in [0, T]$ and $p_h(t) \in Q_h,\ t \in [0, T]$ satisfying for any $t \in [0, T]$

$$B_{sGLS}(\mathbf{u}_h, p_h; \mathbf{v}_h, \psi_h) = L_{sGLS}(\mathbf{v}_h, \psi_h), \quad \forall \mathbf{v}_h \in V_h,\ \ \forall \psi_h \in Q_h, \tag{4.6}$$

$$\mathbf{u}_h - \mathbf{u}_{gh} \in V_h, \tag{4.7}$$

where

$$\begin{aligned}B_{sGLS}(\mathbf{u}_h, p_h; \mathbf{v}_h, \psi_h) &\equiv \int_\Omega \frac{\partial \mathbf{u}_h}{\partial t} \cdot \mathbf{v}_h \mathrm{d}\Omega + \int_\Omega (\mathbf{u}_h \cdot \nabla)\mathbf{u}_h \cdot \mathbf{v}_h \mathrm{d}\Omega + \\ &+ \nu \int_\Omega \nabla \mathbf{u}_h : \nabla \mathbf{v}_h \mathrm{d}\Omega - \int_\Omega p_h \nabla \cdot \mathbf{v}_h \mathrm{d}\Omega + \int_\Omega \psi_h \nabla \cdot \mathbf{u}_h \mathrm{d}\Omega + \\ &+ \sum_{K=1}^N \int_{T_K} \left[\frac{\partial \mathbf{u}_h}{\partial t} + (\mathbf{u}_h \cdot \nabla)\mathbf{u}_h - \nu \Delta \mathbf{u}_h + \nabla p_h \right] \cdot \tau \left[(\mathbf{u}_h \cdot \nabla)\mathbf{v}_h - \nu \Delta \mathbf{v}_h + \nabla \psi_h \right] \mathrm{d}\Omega,\end{aligned}$$

$$L_{sGLS}(\mathbf{v}_h, \psi_h) \equiv \int_\Omega \mathbf{f} \cdot \mathbf{v}_h \mathrm{d}\Omega + \sum_{K=1}^N \int_{T_K} \mathbf{f} \cdot \tau \left[(\mathbf{u}_h \cdot \nabla)\mathbf{v}_h - \nu \Delta \mathbf{v}_h + \nabla \psi_h \right] \mathrm{d}\Omega.$$

Here τ is a positive stabilization parameter. In Section 4.2.5, details on its computation are presented.

The *stabilized steady Navier-Stokes problem* is derived analogously as:

Find $\mathbf{u}_h \in V_{gh}$ and $p_h \in Q_h$ satisfying

$$B_{sGLS}(\mathbf{u}_h, p_h; \mathbf{v}_h, \psi_h) = L_{sGLS}(\mathbf{v}_h, \psi_h), \quad \forall \mathbf{v}_h \in V_h,\ \ \forall \psi_h \in Q_h, \tag{4.8}$$

$$\mathbf{u}_h - \mathbf{u}_{gh} \in V_h, \tag{4.9}$$

where

$$
\begin{aligned}
& B_{sGLS}(\mathbf{u}_h, p_h; \mathbf{v}_h, \psi_h) \equiv \int_\Omega (\mathbf{u}_h \cdot \nabla)\mathbf{u}_h \cdot \mathbf{v}_h \mathrm{d}\Omega + \\
+ \quad & \nu \int_\Omega \nabla \mathbf{u}_h : \nabla \mathbf{v}_h \mathrm{d}\Omega - \int_\Omega p_h \nabla \cdot \mathbf{v}_h \mathrm{d}\Omega + \int_\Omega \psi_h \nabla \cdot \mathbf{u}_h \mathrm{d}\Omega + \\
+ \quad & \sum_{K=1}^{N} \int_{T_K} \left[\frac{\partial \mathbf{u}_h}{\partial t} + (\mathbf{u}_h \cdot \nabla)\mathbf{u}_h - \nu \Delta \mathbf{u}_h + \nabla p_h \right] \cdot \tau \left[(\mathbf{u}_h \cdot \nabla)\mathbf{v}_h - \nu \Delta \mathbf{v}_h + \nabla \psi_h \right] \mathrm{d}\Omega,
\end{aligned}
$$

$$
L_{sGLS}(\mathbf{v}_h, \psi_h) \quad \equiv \quad \int_\Omega \mathbf{f} \cdot \mathbf{v}_h \mathrm{d}\Omega + \sum_{K=1}^{N} \int_{T_K} \mathbf{f} \cdot \tau \left[(\mathbf{u}_h \cdot \nabla)\mathbf{v}_h - \nu \Delta \mathbf{v}_h + \nabla \psi_h \right] \mathrm{d}\Omega.
$$

4.2 Newton method for solution of the stabilized problem

In this section, a method for solution of the stabilized problem (4.6)–(4.7) and (4.8)–(4.9) is proposed.

The steady stabilized problem (4.8)–(4.9) is represented by a system of nonlinear equations. If the backward difference is applied to the time derivative in unsteady semiGLS problem (4.6)–(4.7) (cf. Section 3.4), a fully implicit method is derived. This is characterized, again, by solving a system of nonlinear equations in each time layer.

Based on the previous work [53], the Newton method is applied to the solution of the resulting system. In the rest of the section, matrices of the linearized problem solved in each iteration of the Newton method are derived. First, the derivation is presented for the steady stabilized problem (4.8)–(4.9), followed by extension to the unsteady case (4.6)–(4.7).

Notation 1 *Since only finite element functions $\mathbf{u}_h$, $\mathbf{v}_h$, p_h, and ψ_h are considered in this section, index h is omitted in what follows.*

4.2.1 Functionals for the Newton method and their differentials in steady case

Solution of the nonlinear system representing the stabilized problem (4.8)–(4.9) is equivalent to a consequent minimization of two nonlinear functionals $F_1(\mathbf{u}, p)$ and $F_2(\mathbf{u}, p)$ corresponding to

momentum equation and continuity equation, separately. These functionals are defined as

$$
\begin{aligned}
F_1(\mathbf{u},p) &= \int_\Omega (\mathbf{u}\cdot\nabla)\mathbf{u}\cdot\mathbf{v}\mathrm{d}\Omega + \nu\int_\Omega \nabla\mathbf{u}:\nabla\mathbf{v}\mathrm{d}\Omega - \int_\Omega p\nabla\cdot\mathbf{v}\mathrm{d}\Omega - \int_\Omega \mathbf{f}\cdot\mathbf{v}\mathrm{d}\Omega + \\
&+ \sum_{K=1}^{N}\Bigg\{\int_{T_K}\tau(\mathbf{u}\cdot\nabla)\mathbf{u}\cdot(\mathbf{u}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega - \nu\int_{T_K}\tau(\mathbf{u}\cdot\nabla)\mathbf{u}\cdot\Delta\mathbf{v}\mathrm{d}\Omega + \\
&+ \int_{T_K}\tau(\mathbf{u}\cdot\nabla)\mathbf{u}\cdot\nabla\psi\mathrm{d}\Omega - \nu\int_{T_K}\tau\Delta\mathbf{u}\cdot(\mathbf{u}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega + \nu^2\int_{T_K}\tau\Delta\mathbf{u}\cdot\Delta\mathbf{v}\mathrm{d}\Omega - \\
&- \nu\int_{T_K}\tau\Delta\mathbf{u}\cdot\nabla\psi\mathrm{d}\Omega + \int_{T_K}\tau\nabla p\cdot(\mathbf{u}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega - \nu\int_{T_K}\tau\nabla p\cdot\Delta\mathbf{v}\mathrm{d}\Omega + \\
&+ \int_{T_K}\tau\nabla p\cdot\nabla\psi\mathrm{d}\Omega - \int_{T_K}\tau\mathbf{f}\cdot(\mathbf{u}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega + \\
&+ \nu\int_{T_K}\tau\mathbf{f}\cdot\Delta\mathbf{v}\mathrm{d}\Omega - \int_{T_K}\tau\mathbf{f}\cdot\nabla\psi\mathrm{d}\Omega\Bigg\}, \\
F_2(\mathbf{u},p) &= \int_\Omega \psi\nabla\cdot\mathbf{u}\mathrm{d}\Omega.
\end{aligned}
$$

We propose using the Newton method to find the solution, for which we need Fréchet differentials of functionals $F_1(\mathbf{u}, p)$ and $F_2(\mathbf{u}, p)$. We can find them by evaluating Gateaux differentials since we assume that both exist:

$$< DF_1(\mathbf{u}, p), [\mathbf{h}, q] >=$$

$$\begin{aligned}
&= \lim_{t\to 0} \frac{1}{t} \Bigg[\int_\Omega [(\mathbf{u}+t\mathbf{h})\cdot\nabla](\mathbf{u}+t\mathbf{h})\cdot\mathbf{v}\mathrm{d}\Omega + \nu \int_\Omega \nabla(\mathbf{u}+t\mathbf{h}) : \nabla\mathbf{v}\mathrm{d}\Omega - \\
&- \int_\Omega (p+tq)\nabla\cdot\mathbf{v}\mathrm{d}\Omega - \int_\Omega \mathbf{f}\cdot\mathbf{v}\mathrm{d}\Omega + \\
&+ \sum_{K=1}^{N} \Bigg\{ \int_{T_K} \tau[(\mathbf{u}+t\mathbf{h})\cdot\nabla](\mathbf{u}+t\mathbf{h})\cdot[(\mathbf{u}+t\mathbf{h})\cdot\nabla]\mathbf{v}\mathrm{d}\Omega - \\
&- \nu \int_{T_K} \tau[(\mathbf{u}+t\mathbf{h})\cdot\nabla](\mathbf{u}+t\mathbf{h})\cdot\Delta\mathbf{v}\mathrm{d}\Omega + \int_{T_K} \tau[(\mathbf{u}+t\mathbf{h})\cdot\nabla](\mathbf{u}+t\mathbf{h})\cdot\nabla\psi\mathrm{d}\Omega - \\
&- \nu \int_{T_K} \tau\Delta(\mathbf{u}+t\mathbf{h})\cdot[(\mathbf{u}+t\mathbf{h})\cdot\nabla]\mathbf{v}\mathrm{d}\Omega + \nu^2 \int_{T_K} \tau\Delta(\mathbf{u}+t\mathbf{h})\cdot\Delta\mathbf{v}\mathrm{d}\Omega - \\
&- \nu \int_{T_K} \tau\Delta(\mathbf{u}+t\mathbf{h})\cdot\nabla\psi\mathrm{d}\Omega + \int_{T_K} \tau\nabla(p+tq)\cdot[(\mathbf{u}+t\mathbf{h})\cdot\nabla]\mathbf{v}\mathrm{d}\Omega - \\
&- \nu \int_{T_K} \tau\nabla(p+tq)\cdot\Delta\mathbf{v}\mathrm{d}\Omega + \int_{T_K} \tau\nabla(p+tq)\cdot\nabla\psi\mathrm{d}\Omega - \\
&- \int_{T_K} \tau\mathbf{f}\cdot[(\mathbf{u}+t\mathbf{h})\cdot\nabla]\mathbf{v}\mathrm{d}\Omega + \nu \int_{T_K} \tau\mathbf{f}\cdot\Delta\mathbf{v}\mathrm{d}\Omega - \int_{T_K} \tau\mathbf{f}\cdot\nabla\psi\mathrm{d}\Omega \Bigg\} - \\
&- \int_\Omega (\mathbf{u}\cdot\nabla)\mathbf{u}\cdot\mathbf{v}\mathrm{d}\Omega - \nu \int_\Omega \nabla\mathbf{u} : \nabla\mathbf{v}\mathrm{d}\Omega + \int_\Omega p\nabla\cdot\mathbf{v}\mathrm{d}\Omega + \int_\Omega \mathbf{f}\cdot\mathbf{v}\mathrm{d}\Omega - \\
&- \sum_{K=1}^{N} \Bigg\{ \int_{T_K} \tau(\mathbf{u}\cdot\nabla)\mathbf{u}\cdot(\mathbf{u}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega - \nu \int_{T_K} \tau(\mathbf{u}\cdot\nabla)\mathbf{u}\cdot\Delta\mathbf{v}\mathrm{d}\Omega + \\
&+ \int_{T_K} \tau(\mathbf{u}\cdot\nabla)\mathbf{u}\cdot\nabla\psi\mathrm{d}\Omega - \nu \int_{T_K} \tau\Delta\mathbf{u}\cdot(\mathbf{u}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega + \\
&+ \nu^2 \int_{T_K} \tau\Delta\mathbf{u}\cdot\Delta\mathbf{v}\mathrm{d}\Omega - \nu \int_{T_K} \tau\Delta\mathbf{u}\cdot\nabla\psi\mathrm{d}\Omega + \\
&+ \int_{T_K} \tau\nabla p\cdot(\mathbf{u}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega - \nu \int_{T_K} \tau\nabla p\cdot\Delta\mathbf{v}\mathrm{d}\Omega + \\
&+ \int_{T_K} \tau\nabla p\cdot\nabla\psi\mathrm{d}\Omega - \int_{T_K} \tau\mathbf{f}\cdot(\mathbf{u}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega + \\
&+ \nu \int_{T_K} \tau\mathbf{f}\cdot\Delta\mathbf{v}\mathrm{d}\Omega - \int_{T_K} \tau\mathbf{f}\cdot\nabla\psi\mathrm{d}\Omega \Bigg\} \Bigg],
\end{aligned}$$

$$< DF_2(\mathbf{u}, p), [\mathbf{h}, q] >= \lim_{t\to 0} \frac{1}{t} \left[\int_\Omega \psi\nabla\cdot(\mathbf{u}+t\mathbf{h})\mathrm{d}\Omega - \int_\Omega \psi\nabla\cdot\mathbf{u}\mathrm{d}\Omega \right].$$

After letting $t \to 0$, we get the following lemma.

Lemma 2 *Assume all functions sufficiently smooth. Then the Gateaux differentials of functionals* $F_1(\mathbf{u}, p)$ *and* $F_2(\mathbf{u}, p)$ *are*

$$< DF_1(\mathbf{u}, p), [\mathbf{h}, q] >=$$

$$\begin{aligned}
&= \int_\Omega (\mathbf{h}\cdot\nabla)\mathbf{u}\cdot\mathbf{v}\mathrm{d}\Omega + \int_\Omega (\mathbf{u}\cdot\nabla)\mathbf{h}\cdot\mathbf{v}\mathrm{d}\Omega + \nu\int_\Omega \nabla\mathbf{h} : \nabla\mathbf{v}\mathrm{d}\Omega - \int_\Omega q\nabla\cdot\mathbf{v}\mathrm{d}\Omega + \\
&+ \sum_{K=1}^{N}\Bigg\{ \int_{T_K} \tau(\mathbf{h}\cdot\nabla)\mathbf{u}\cdot(\mathbf{u}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega + \int_{T_K} \tau(\mathbf{u}\cdot\nabla)\mathbf{h}\cdot(\mathbf{u}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega + \\
&+ \int_{T_K} \tau(\mathbf{u}\cdot\nabla)\mathbf{u}\cdot(\mathbf{h}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega - \nu\int_{T_K} \tau(\mathbf{h}\cdot\nabla)\mathbf{u}\cdot\Delta\mathbf{v}\mathrm{d}\Omega - \nu\int_{T_K} \tau(\mathbf{u}\cdot\nabla)\mathbf{h}\cdot\Delta\mathbf{v}\mathrm{d}\Omega + \\
&+ \int_{T_K} \tau(\mathbf{h}\cdot\nabla)\mathbf{u}\cdot\nabla\psi\mathrm{d}\Omega + \int_{T_K} \tau(\mathbf{u}\cdot\nabla)\mathbf{h}\cdot\nabla\psi\mathrm{d}\Omega - \nu\int_{T_K} \tau\Delta\mathbf{h}\cdot(\mathbf{u}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega - \\
&- \nu\int_{T_K} \tau\Delta\mathbf{u}\cdot(\mathbf{h}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega + \nu^2\int_{T_K} \tau\Delta\mathbf{h}\cdot\Delta\mathbf{v}\mathrm{d}\Omega - \nu\int_{T_K} \tau\Delta\mathbf{h}\cdot\nabla\psi\mathrm{d}\Omega + \\
&+ \int_{T_K} \tau\nabla q\cdot(\mathbf{u}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega + \int_{T_K} \tau\nabla p\cdot(\mathbf{h}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega - \nu\int_{T_K} \tau\nabla q\cdot\Delta\mathbf{v}\mathrm{d}\Omega + \\
&+ \int_{T_K} \tau\nabla q\cdot\nabla\psi\mathrm{d}\Omega - \int_{T_K} \tau\mathbf{f}\cdot(\mathbf{h}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega\Bigg\},
\end{aligned}$$

and

$$< DF_2(\mathbf{u}, p), [\mathbf{h}, q] >= \int_\Omega \psi\nabla\cdot\mathbf{h}\mathrm{d}\Omega.$$

Let us formally introduce the functional

$$\mathcal{F}(\mathbf{u}, p) = F_1(\mathbf{u}, p) + F_2(\mathbf{u}, p), \tag{4.10}$$

and its differential

$$< D\mathcal{F}(\mathbf{u}, p), [\mathbf{h}, q] >=< DF_1(\mathbf{u}, p), [\mathbf{h}, q] > + < DF_2(\mathbf{u}, p), [\mathbf{h}, q] > . \tag{4.11}$$

The algorithm consists of the following steps

1. solution of the equation system (the z^{th} iteration of the Newton method) for $\mathbf{h}$, q:

$$< D\mathcal{F}(\mathbf{u}, p), [\mathbf{h}, q] >= -\mathcal{F}(\mathbf{u}, p) \tag{4.12}$$

2. correction of the solution $\mathbf{u}$, p:

$$\begin{aligned}
\mathbf{u}^{z+1} &= \mathbf{u}^z + \mathbf{h} \\
p^{z+1} &= p^z + q
\end{aligned}$$

4.2.2 Matrices for the finite element method in steady case

Let us derive the element matrix for a finite element T_K. We substitute

$$\begin{aligned} h_x|_{T_K} &= \sum_{i=1}^{N_u} h_{xi} N_i, \\ h_y|_{T_K} &= \sum_{i=1}^{N_u} h_{y_i} N_i, \\ q|_{T_K} &= \sum_{i=1}^{N_p} q_i M_i, \end{aligned}$$

and reduce all test functions to

$$\begin{aligned} \mathbf{v} &= (N_j, 0),\ \mathbf{v} = (0, N_j), \\ \psi &= M_j, \end{aligned}$$

where

- $N_j,\ j = 1, \ldots, N_u$ are the shape functions for each component of velocity on the element,
- $M_j,\ j = 1, \ldots, N_p$ are the shape functions for pressure on the element.

Then, we derive three equations for each node from the scalar equation (4.12) by taking in turn the following combinations of test functions

$$\begin{aligned} \mathbf{v} &= (N_j, 0), \psi = 0, \\ \mathbf{v} &= (0, N_j), \psi = 0, \\ \mathbf{v} &= (0, 0), \psi = M_j. \end{aligned}$$

These equations read for the node j

$$< D\mathcal{F}(\mathbf{u}, p), [\mathbf{h}, q] >_{j_1} = -\mathcal{F}(\mathbf{u}, p)_{j_1}, \tag{4.13}$$
$$< D\mathcal{F}(\mathbf{u}, p), [\mathbf{h}, q] >_{j_2} = -\mathcal{F}(\mathbf{u}, p)_{j_2}, \tag{4.14}$$
$$< D\mathcal{F}(\mathbf{u}, p), [\mathbf{h}, q] >_{j_3} = -\mathcal{F}(\mathbf{u}, p)_{j_3}, \tag{4.15}$$

where by Lemma 2 we obtain the following terms:

$$< D\mathcal{F}(\mathbf{u},p),[\mathbf{h},q] >_{j_1} =$$

$$
\begin{aligned}
&= \sum_{i=1}^{N_u}\int_\Omega \left(h_{x_i}N_i\frac{\partial u_x}{\partial x}+h_{y_i}N_i\frac{\partial u_x}{\partial y}\right)N_j\mathrm{d}\Omega+\sum_{i=1}^{N_u}\int_\Omega h_{x_i}\left(u_x\frac{\partial N_i}{\partial x}+u_y\frac{\partial N_i}{\partial y}\right)N_j\mathrm{d}\Omega+\\
&+ \sum_{i=1}^{N_u}\nu\int_\Omega h_{x_i}\left(\frac{\partial N_i}{\partial x}\frac{\partial N_j}{\partial x}+\frac{\partial N_i}{\partial y}\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega-\sum_{i=1}^{N_p}\int_\Omega q_iM_i\frac{\partial N_j}{\partial x}\mathrm{d}\Omega+\\
&+ \sum_{i=1}^{N_u}\sum_K\left\{\int_{T_K}\tau\left(h_{x_i}N_i\frac{\partial u_x}{\partial x}+h_{y_i}N_i\frac{\partial u_x}{\partial y}\right)\left(u_x\frac{\partial N_j}{\partial x}+u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega+\right.\\
&+ \int_{T_K}\tau h_{x_i}\left(u_x\frac{\partial N_i}{\partial x}+u_y\frac{\partial N_i}{\partial y}\right)\left(u_x\frac{\partial N_j}{\partial x}+u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega+\\
&+ \int_{T_K}\tau\left(u_x\frac{\partial u_x}{\partial x}+u_y\frac{\partial u_x}{\partial y}\right)\left(h_{x_i}N_i\frac{\partial N_j}{\partial x}+h_{y_i}N_i\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega-\\
&- \nu\int_{T_K}\tau\left(h_{x_i}N_i\frac{\partial u_x}{\partial x}+h_{y_i}N_i\frac{\partial u_x}{\partial y}\right)\left(\frac{\partial^2 N_j}{\partial x^2}+\frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega-\\
&- \nu\int_{T_K}\tau h_{x_i}\left(u_x\frac{\partial N_i}{\partial x}+u_y\frac{\partial N_i}{\partial y}\right)\left(\frac{\partial^2 N_j}{\partial x^2}+\frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega-\\
&- \nu\int_{T_K}\tau h_{x_i}\left(\frac{\partial^2 N_i}{\partial x^2}+\frac{\partial^2 N_i}{\partial y^2}\right)\left(u_x\frac{\partial N_j}{\partial x}+u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega-\\
&- \nu\int_{T_K}\tau\left(\frac{\partial^2 u_x}{\partial x^2}+\frac{\partial^2 u_x}{\partial y^2}\right)\left(h_{x_i}N_i\frac{\partial N_j}{\partial x}+h_{y_i}N_i\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega+\\
&+ \nu^2\int_{T_K}\tau h_{x_i}\left(\frac{\partial^2 N_i}{\partial x^2}+\frac{\partial^2 N_i}{\partial y^2}\right)\left(\frac{\partial^2 N_j}{\partial x^2}+\frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega+\\
&+ \int_{T_K}\tau\frac{\partial p}{\partial x}\left(h_{x_i}N_i\frac{\partial N_j}{\partial x}+h_{y_i}N_i\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega-\\
&- \left.\int_{T_K}\tau f_x\left(h_{x_i}N_i\frac{\partial N_j}{\partial x}+h_{y_i}N_i\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega\right\}+\\
&+ \sum_{i=1}^{N_p}\sum_K\left\{\int_{T_K}\tau q_i\frac{\partial M_i}{\partial x}\left(u_x\frac{\partial N_j}{\partial x}+u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega-\right.\\
&- \left.\nu\int_{T_K}\tau q_i\frac{\partial M_i}{\partial x}\left(\frac{\partial^2 N_j}{\partial x^2}+\frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega\right\},
\end{aligned}
$$

$$< D\mathcal{F}(\mathbf{u},p),[\mathbf{h},q] >_{j_2}=$$

$$
\begin{aligned}
&= \sum_{i=1}^{N_u}\int_\Omega \left(h_{x_i}N_i\frac{\partial u_y}{\partial x}+h_{y_i}N_i\frac{\partial u_y}{\partial y}\right)N_j\mathrm{d}\Omega+\sum_{i=1}^{N_u}\int_\Omega h_{y_i}\left(u_x\frac{\partial N_i}{\partial x}+u_y\frac{\partial N_i}{\partial y}\right)N_j\mathrm{d}\Omega+\\
&+ \sum_{i=1}^{N_u}\nu\int_\Omega h_{y_i}\left(\frac{\partial N_i}{\partial x}\frac{\partial N_j}{\partial x}+\frac{\partial N_i}{\partial y}\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega-\sum_{i=1}^{N_p}\int_\Omega q_iM_i\frac{\partial N_j}{\partial y}\mathrm{d}\Omega+\\
&+ \sum_{i=1}^{N_u}\sum_K\left\{\int_{T_K}\tau\left(h_{x_i}N_i\frac{\partial u_y}{\partial x}+h_{y_i}N_i\frac{\partial u_y}{\partial y}\right)\left(u_x\frac{\partial N_j}{\partial x}+u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega+\right.\\
&+ \int_{T_K}\tau h_{y_i}\left(u_x\frac{\partial N_i}{\partial x}+u_y\frac{\partial N_i}{\partial y}\right)\left(u_x\frac{\partial N_j}{\partial x}+u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega+\\
&+ \int_{T_K}\tau\left(u_x\frac{\partial u_y}{\partial x}+u_y\frac{\partial u_y}{\partial y}\right)\left(h_{x_i}N_i\frac{\partial N_j}{\partial x}+h_{y_i}N_i\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega-\\
&- \nu\int_{T_K}\tau\left(h_{x_i}N_i\frac{\partial u_y}{\partial x}+h_{y_i}N_i\frac{\partial u_y}{\partial y}\right)\left(\frac{\partial^2 N_j}{\partial x^2}+\frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega-\\
&- \nu\int_{T_K}\tau h_{y_i}\left(u_x\frac{\partial N_i}{\partial x}+u_y\frac{\partial N_i}{\partial y}\right)\left(\frac{\partial^2 N_j}{\partial x^2}+\frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega-\\
&- \nu\int_{T_K}\tau h_{y_i}\left(\frac{\partial^2 N_i}{\partial x^2}+\frac{\partial^2 N_i}{\partial y^2}\right)\left(u_x\frac{\partial N_j}{\partial x}+u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega-\\
&- \nu\int_{T_K}\tau\left(\frac{\partial^2 u_y}{\partial x^2}+\frac{\partial^2 u_y}{\partial y^2}\right)\left(h_{x_i}N_i\frac{\partial N_j}{\partial x}+h_{y_i}N_i\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega+\\
&+ \nu^2\int_{T_K}\tau h_{y_i}\left(\frac{\partial^2 N_i}{\partial x^2}+\frac{\partial^2 N_i}{\partial y^2}\right)\left(\frac{\partial^2 N_j}{\partial x^2}+\frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega+\\
&+ \int_{T_K}\tau\frac{\partial p}{\partial y}\left(h_{x_i}N_i\frac{\partial N_j}{\partial x}+h_{y_i}N_i\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega-\\
&- \left.\int_{T_K}\tau f_y\left(h_{x_i}N_i\frac{\partial N_j}{\partial x}+h_{y_i}N_i\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega\right\}+\\
&+ \sum_{i=1}^{N_p}\sum_K\left\{\int_{T_K}\tau q_i\frac{\partial M_i}{\partial y}\left(u_x\frac{\partial N_j}{\partial x}+u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega-\right.\\
&- \left.\nu\int_{T_K}\tau q_i\frac{\partial M_i}{\partial y}\left(\frac{\partial^2 N_j}{\partial x^2}+\frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega\right\},
\end{aligned}
$$

$$< D\mathcal{F}(\mathbf{u},p),[\mathbf{h},q] >_{j_3} =$$

$$\begin{aligned}
&= \sum_{i=1}^{N_u}\int_\Omega \left(h_{x_i}\frac{\partial N_i}{\partial x}+h_{y_i}\frac{\partial N_i}{\partial y}\right)M_j \mathrm{d}\Omega + \\
&+ \sum_{i=1}^{N_u}\sum_K \left\{\int_{T_K}\tau\left(h_{x_i}N_i\frac{\partial u_x}{\partial x}+h_{y_i}N_i\frac{\partial u_x}{\partial y}\right)\frac{\partial M_j}{\partial x}\mathrm{d}\Omega + \right. \\
&+ \int_{T_K}\tau\left(h_{x_i}N_i\frac{\partial u_y}{\partial x}+h_{y_i}N_i\frac{\partial u_y}{\partial y}\right)\frac{\partial M_j}{\partial y}\mathrm{d}\Omega + \\
&+ \int_{T_K}\tau h_{x_i}\left(u_x\frac{\partial N_i}{\partial x}+u_y\frac{\partial N_i}{\partial y}\right)\frac{\partial M_j}{\partial x}\mathrm{d}\Omega + \int_{T_K}\tau h_{y_i}\left(u_x\frac{\partial N_i}{\partial x}+u_y\frac{\partial N_i}{\partial y}\right)\frac{\partial M_j}{\partial y}\mathrm{d}\Omega - \\
&- \left.\nu\int_{T_K}\tau h_{x_i}\left(\frac{\partial^2 N_i}{\partial x^2}+\frac{\partial^2 N_i}{\partial y^2}\right)\frac{\partial M_j}{\partial x}\mathrm{d}\Omega - \nu\int_{T_K}\tau h_{y_i}\left(\frac{\partial^2 N_i}{\partial x^2}+\frac{\partial^2 N_i}{\partial y^2}\right)\frac{\partial M_j}{\partial y}\mathrm{d}\Omega\right\} + \\
&+ \sum_{i=1}^{N_p}\sum_K\int_{T_K}\tau q_i\left(\frac{\partial M_i}{\partial x}\frac{\partial M_j}{\partial x}+\frac{\partial M_i}{\partial y}\frac{\partial M_j}{\partial y}\right)\mathrm{d}\Omega,
\end{aligned}$$

$$\begin{aligned}
\mathcal{F}(\mathbf{u},p)_{j_1} &= \int_\Omega\left(u_x\frac{\partial u_x}{\partial x}+u_y\frac{\partial u_x}{\partial y}\right)N_j\mathrm{d}\Omega + \nu\int_\Omega\left(\frac{\partial u_x}{\partial x}\frac{\partial N_j}{\partial x}+\frac{\partial u_x}{\partial y}\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega - \\
&- \int_\Omega p\frac{\partial N_j}{\partial x}\mathrm{d}\Omega - \int_\Omega f_x N_j\mathrm{d}\Omega + \\
&+ \sum_K\left\{\int_K\tau\left(u_x\frac{\partial u_x}{\partial x}+u_y\frac{\partial u_x}{\partial y}\right)\left(u_x\frac{\partial N_j}{\partial x}+u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega - \right. \\
&- \nu\int_{T_K}\tau\left(u_x\frac{\partial u_x}{\partial x}+u_y\frac{\partial u_x}{\partial y}\right)\left(\frac{\partial^2 N_j}{\partial x^2}+\frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega - \\
&- \nu\int_{T_K}\tau\left(\frac{\partial^2 u_x}{\partial x^2}+\frac{\partial^2 u_x}{\partial y^2}\right)\left(u_x\frac{\partial N_j}{\partial x}+u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega + \\
&+ \nu^2\int_{T_K}\tau\left(\frac{\partial^2 u_x}{\partial x^2}+\frac{\partial^2 u_x}{\partial y^2}\right)\left(\frac{\partial^2 N_j}{\partial x^2}+\frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega + \\
&+ \int_{T_K}\tau\frac{\partial p}{\partial x}\left(u_x\frac{\partial N_j}{\partial x}+u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega - \\
&- \nu\int_{T_K}\tau\frac{\partial p}{\partial x}\left(\frac{\partial^2 N_j}{\partial x^2}+\frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega - \\
&- \int_{T_K}\tau f_x\left(u_x\frac{\partial N_j}{\partial x}+u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega + \\
&+ \left.\nu\int_{T_K}\tau f_x\left(\frac{\partial^2 N_j}{\partial x^2}+\frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega\right\},
\end{aligned}$$

$$
\begin{aligned}
\mathcal{F}(\mathbf{u},p)_{j_2} &= \int_\Omega \left(u_x\frac{\partial u_y}{\partial x}+u_y\frac{\partial u_y}{\partial y}\right)N_j\mathrm{d}\Omega+\nu\int_\Omega\left(\frac{\partial u_y}{\partial x}\frac{\partial N_j}{\partial x}+\frac{\partial u_y}{\partial y}\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega- \\
&- \int_\Omega p\frac{\partial N_j}{\partial y}\mathrm{d}\Omega-\int_\Omega f_y N_j\mathrm{d}\Omega+ \\
&+ \sum_K\left\{\int_{T_K}\tau\left(u_x\frac{\partial u_y}{\partial x}+u_y\frac{\partial u_y}{\partial y}\right)\left(u_x\frac{\partial N_j}{\partial x}+u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega-\right. \\
&- \nu\int_{T_K}\tau\left(u_x\frac{\partial u_y}{\partial x}+u_y\frac{\partial u_y}{\partial y}\right)\left(\frac{\partial^2 N_j}{\partial x^2}+\frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega- \\
&- \nu\int_{T_K}\tau\left(\frac{\partial^2 u_y}{\partial x^2}+\frac{\partial^2 u_y}{\partial y^2}\right)\left(u_x\frac{\partial N_j}{\partial x}+u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega+ \\
&+ \nu^2\int_{T_K}\tau\left(\frac{\partial^2 u_y}{\partial x^2}+\frac{\partial^2 u_y}{\partial y^2}\right)\left(\frac{\partial^2 N_j}{\partial x^2}+\frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega+ \\
&+ \int_{T_K}\tau\frac{\partial p}{\partial y}\left(u_x\frac{\partial N_j}{\partial x}+u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega- \\
&- \nu\int_{T_K}\tau\frac{\partial p}{\partial y}\left(\frac{\partial^2 N_j}{\partial x^2}+\frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega- \\
&- \int_{T_K}\tau f_y\left(u_x\frac{\partial N_j}{\partial x}+u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega+ \\
&+ \left.\nu\int_{T_K}\tau f_y\left(\frac{\partial^2 N_j}{\partial x^2}+\frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega\right\},
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{F}(\mathbf{u},p)_{j_3} &= \int_\Omega\left(\frac{\partial u_x}{\partial x}+\frac{\partial u_y}{\partial y}\right)M_j\mathrm{d}\Omega+ \\
&+ \sum_K\left\{\int_{T_K}\tau\left(u_x\frac{\partial u_x}{\partial x}+u_y\frac{\partial u_x}{\partial y}\right)\frac{\partial M_j}{\partial x}\mathrm{d}\Omega+\right. \\
&+ \int_{T_K}\tau\left(u_x\frac{\partial u_y}{\partial x}+u_y\frac{\partial u_y}{\partial y}\right)\frac{\partial M_j}{\partial y}\mathrm{d}\Omega- \\
&- \nu\int_{T_K}\tau\left(\frac{\partial^2 u_x}{\partial x^2}+\frac{\partial^2 u_x}{\partial y^2}\right)\frac{\partial M_j}{\partial x}\mathrm{d}\Omega- \\
&- \nu\int_{T_K}\tau\left(\frac{\partial^2 u_y}{\partial x^2}+\frac{\partial^2 u_y}{\partial y^2}\right)\frac{\partial M_j}{\partial y}\mathrm{d}\Omega+ \\
&+ \int_{T_K}\tau\frac{\partial p}{\partial x}\frac{\partial M_j}{\partial x}\mathrm{d}\Omega+\int_{T_K}\tau\frac{\partial p}{\partial y}\frac{\partial M_j}{\partial y}\mathrm{d}\Omega- \\
&- \left.\int_{T_K}\tau f_x\frac{\partial M_j}{\partial x}\mathrm{d}\Omega-\int_{T_K}\tau f_y\frac{\partial M_j}{\partial y}\mathrm{d}\Omega\right\}.
\end{aligned}
$$

Now, we can extract the elements of the ji-submatrix $\mathbf{K}_{ji}$ of the element stiffness matrix $\mathbf{K}^e$ (cf. Figure 4.1):

$$\begin{aligned}
K_{ji_{11}}(\mathbf{u},p) &= \int_\Omega N_i \frac{\partial u_x}{\partial x} N_j \mathrm{d}\Omega + \int_\Omega \left(u_x \frac{\partial N_i}{\partial x} + u_y \frac{\partial N_i}{\partial y} \right) N_j \mathrm{d}\Omega + \\
&+ \nu \int_\Omega \left(\frac{\partial N_i}{\partial x}\frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y}\frac{\partial N_j}{\partial y} \right) \mathrm{d}\Omega + \\
&+ \sum_K \left\{ \int_{T_K} \tau N_i \frac{\partial u_x}{\partial x} \left(u_x \frac{\partial N_j}{\partial x} + u_y \frac{\partial N_j}{\partial y} \right) \mathrm{d}\Omega + \right. \\
&+ \int_{T_K} \tau \left(u_x \frac{\partial N_i}{\partial x} + u_y \frac{\partial N_i}{\partial y} \right) \left(u_x \frac{\partial N_j}{\partial x} + u_y \frac{\partial N_j}{\partial y} \right) \mathrm{d}\Omega + \\
&+ \int_{T_K} \tau \left(u_x \frac{\partial u_x}{\partial x} + u_y \frac{\partial u_x}{\partial y} \right) N_i \frac{\partial N_j}{\partial x} \mathrm{d}\Omega - \\
&- \nu \int_{T_K} \tau N_i \frac{\partial u_x}{\partial x} \left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2} \right) \mathrm{d}\Omega - \\
&- \nu \int_{T_K} \tau \left(u_x \frac{\partial N_i}{\partial x} + u_y \frac{\partial N_i}{\partial y} \right) \left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2} \right) \mathrm{d}\Omega - \\
&- \nu \int_{T_K} \tau \left(\frac{\partial^2 N_i}{\partial x^2} + \frac{\partial^2 N_i}{\partial y^2} \right) \left(u_x \frac{\partial N_j}{\partial x} + u_y \frac{\partial N_j}{\partial y} \right) \mathrm{d}\Omega - \\
&- \nu \int_{T_K} \tau \left(\frac{\partial^2 u_x}{\partial x^2} + \frac{\partial^2 u_x}{\partial y^2} \right) N_i \frac{\partial N_j}{\partial x} \mathrm{d}\Omega + \\
&+ \nu^2 \int_{T_K} \tau \left(\frac{\partial^2 N_i}{\partial x^2} + \frac{\partial^2 N_i}{\partial y^2} \right) \left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2} \right) \mathrm{d}\Omega + \\
&+ \left. \int_{T_K} \tau \frac{\partial p}{\partial x} N_i \frac{\partial N_j}{\partial x} \mathrm{d}\Omega - \int_{T_K} \tau f_x N_i \frac{\partial N_j}{\partial x} \mathrm{d}\Omega \right\},
\end{aligned}$$

$$\begin{aligned}
K_{ji_{12}}(\mathbf{u},p) &= \int_\Omega N_i \frac{\partial u_x}{\partial y} N_j \mathrm{d}\Omega + \\
&+ \sum_K \left\{ \int_{T_K} \tau N_i \frac{\partial u_x}{\partial y} \left(u_x \frac{\partial N_j}{\partial x} + u_y \frac{\partial N_j}{\partial y} \right) \mathrm{d}\Omega + \right. \\
&+ \int_{T_K} \tau \left(u_x \frac{\partial u_x}{\partial x} + u_y \frac{\partial u_x}{\partial y} \right) N_i \frac{\partial N_j}{\partial y} \mathrm{d}\Omega - \\
&- \nu \int_{T_K} \tau N_i \frac{\partial u_x}{\partial y} \left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2} \right) \mathrm{d}\Omega - \\
&- \nu \int_{T_K} \tau \left(\frac{\partial^2 u_x}{\partial x^2} + \frac{\partial^2 u_x}{\partial y^2} \right) N_i \frac{\partial N_j}{\partial y} \mathrm{d}\Omega + \\
&+ \left. \int_{T_K} \tau \frac{\partial p}{\partial x} N_i \frac{\partial N_j}{\partial y} \mathrm{d}\Omega - \int_{T_K} \tau f_x N_i \frac{\partial N_j}{\partial y} \mathrm{d}\Omega \right\},
\end{aligned}$$

$$
\begin{aligned}
K_{ji_{13}}(\mathbf{u},p) &= -\int_\Omega M_i \frac{\partial N_j}{\partial x}\mathrm{d}\Omega + \\
&+ \sum_K \left\{ \int_{T_K} \tau \frac{\partial M_i}{\partial x}\left(u_x\frac{\partial N_j}{\partial x}+u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega - \right. \\
&- \left. \nu\int_{T_K}\tau\frac{\partial M_i}{\partial x}\left(\frac{\partial^2 N_j}{\partial x^2}+\frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega \right\},
\end{aligned}
$$

$$
\begin{aligned}
K_{ji_{21}}(\mathbf{u},p) &= \int_\Omega N_i\frac{\partial u_y}{\partial x}N_j\mathrm{d}\Omega + \\
&+ \sum_K \left\{ \int_{T_K}\tau N_i\frac{\partial u_y}{\partial x}\left(u_x\frac{\partial N_j}{\partial x}+u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega + \right. \\
&+ \int_{T_K}\tau\left(u_x\frac{\partial u_y}{\partial x}+u_y\frac{\partial u_y}{\partial y}\right)N_i\frac{\partial N_j}{\partial x}\mathrm{d}\Omega - \\
&- \nu\int_{T_K}\tau N_i\frac{\partial u_y}{\partial x}\left(\frac{\partial^2 N_j}{\partial x^2}+\frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega - \\
&- \nu\int_{T_K}\tau\left(\frac{\partial^2 u_y}{\partial x^2}+\frac{\partial^2 u_y}{\partial y^2}\right)N_i\frac{\partial N_j}{\partial x}\mathrm{d}\Omega + \\
&+ \left. \int_{T_K}\tau\frac{\partial p}{\partial y}N_i\frac{\partial N_j}{\partial x}\mathrm{d}\Omega - \int_{T_K}\tau f_y N_i\frac{\partial N_j}{\partial x}\mathrm{d}\Omega \right\},
\end{aligned}
$$

$$
\begin{aligned}
K_{ji_{22}}(\mathbf{u},p) &= \int_\Omega N_i\frac{\partial u_y}{\partial y}N_j\mathrm{d}\Omega + \int_\Omega\left(u_x\frac{\partial N_i}{\partial x}+u_y\frac{\partial N_i}{\partial y}\right)N_j\mathrm{d}\Omega + \\
&+ \nu\int_\Omega\left(\frac{\partial N_i}{\partial x}\frac{\partial N_j}{\partial x}+\frac{\partial N_i}{\partial y}\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega + \\
&+ \sum_K\left\{\int_{T_K}\tau N_i\frac{\partial u_y}{\partial y}\left(u_x\frac{\partial N_j}{\partial x}+u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega + \right. \\
&+ \int_{T_K}\tau\left(u_x\frac{\partial N_i}{\partial x}+u_y\frac{\partial N_i}{\partial y}\right)\left(u_x\frac{\partial N_j}{\partial x}+u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega + \\
&+ \int_{T_K}\tau\left(u_x\frac{\partial u_y}{\partial x}+u_y\frac{\partial u_y}{\partial y}\right)N_i\frac{\partial N_j}{\partial y}\mathrm{d}\Omega - \\
&- \nu\int_{T_K}\tau N_i\frac{\partial u_y}{\partial y}\left(\frac{\partial^2 N_j}{\partial x^2}+\frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega - \\
&- \nu\int_K\tau\left(u_x\frac{\partial N_i}{\partial x}+u_y\frac{\partial N_i}{\partial y}\right)\left(\frac{\partial^2 N_j}{\partial x^2}+\frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega - \\
&- \nu\int_{T_K}\tau\left(\frac{\partial^2 N_i}{\partial x^2}+\frac{\partial^2 N_i}{\partial y^2}\right)\left(u_x\frac{\partial N_j}{\partial x}+u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega - \\
&- \nu\int_{T_K}\tau\left(\frac{\partial^2 u_y}{\partial x^2}+\frac{\partial^2 u_y}{\partial y^2}\right)N_i\frac{\partial N_j}{\partial y}\mathrm{d}\Omega + \\
&+ \nu^2\int_{T_K}\tau\left(\frac{\partial^2 N_i}{\partial x^2}+\frac{\partial^2 N_i}{\partial y^2}\right)\left(\frac{\partial^2 N_j}{\partial x^2}+\frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega + \\
&+ \left.\int_{T_K}\tau\frac{\partial p}{\partial y}N_i\frac{\partial N_j}{\partial y}\mathrm{d}\Omega - \int_{T_K}\tau f_y N_i\frac{\partial N_j}{\partial y}\mathrm{d}\Omega\right\},
\end{aligned}
$$

$$
\begin{aligned}
K_{ji_{23}}(\mathbf{u},p) &= -\int_\Omega M_i \frac{\partial N_j}{\partial y}\mathrm{d}\Omega + \\
&+ \sum_K \left\{ \int_{T_K} \tau \frac{\partial M_i}{\partial y}\left(u_x\frac{\partial N_j}{\partial x}+u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega - \right. \\
&- \left. \nu\int_{T_K}\tau\frac{\partial M_i}{\partial y}\left(\frac{\partial^2 N_j}{\partial x^2}+\frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega\right\},
\end{aligned}
$$

$$
\begin{aligned}
K_{ji_{31}}(\mathbf{u},p) &= \int_\Omega \frac{\partial N_i}{\partial x} M_j \mathrm{d}\Omega + \\
&+ \sum_K \left\{ \int_{T_K} \tau N_i \frac{\partial u_x}{\partial x}\frac{\partial M_j}{\partial x}\mathrm{d}\Omega + \int_{T_K}\tau N_i\frac{\partial u_y}{\partial x}\frac{\partial M_j}{\partial y}\mathrm{d}\Omega + \right. \\
&+ \left. \int_{T_K}\tau\left(u_x\frac{\partial N_i}{\partial x}+u_y\frac{\partial N_i}{\partial y}\right)\frac{\partial M_j}{\partial x}\mathrm{d}\Omega - \nu\int_{T_K}\tau\left(\frac{\partial^2 N_i}{\partial x^2}+\frac{\partial^2 N_i}{\partial y^2}\right)\frac{\partial M_j}{\partial x}\mathrm{d}\Omega\right\},
\end{aligned}
$$

$$
\begin{aligned}
K_{ji_{32}}(\mathbf{u},p) &= \int_\Omega \frac{\partial N_i}{\partial y} M_j \mathrm{d}\Omega + \\
&+ \sum_K \left\{ \int_{T_K} \tau N_i \frac{\partial u_x}{\partial y}\frac{\partial M_j}{\partial x}\mathrm{d}\Omega + \int_{T_K}\tau N_i\frac{\partial u_y}{\partial y}\frac{\partial M_j}{\partial y}\mathrm{d}\Omega + \right. \\
&+ \left. \int_{T_K}\tau\left(u_x\frac{\partial N_i}{\partial x}+u_y\frac{\partial N_i}{\partial y}\right)\frac{\partial M_j}{\partial y}\mathrm{d}\Omega - \nu\int_{T_K}\tau\left(\frac{\partial^2 N_i}{\partial x^2}+\frac{\partial^2 N_i}{\partial y^2}\right)\frac{\partial M_j}{\partial y}\mathrm{d}\Omega\right\},
\end{aligned}
$$

$$
K_{ji_{33}}(\mathbf{u},p) = \sum_K \int_{T_K}\tau\left(\frac{\partial M_i}{\partial x}\frac{\partial M_j}{\partial x}+\frac{\partial M_i}{\partial y}\frac{\partial M_j}{\partial y}\right)\mathrm{d}\Omega.
$$

Matrix $\mathbf{K}_{ji}$ can be written as

$$
\mathbf{K}_{ji} = \begin{bmatrix} K_{ji_{11}} & K_{ji_{12}} & K_{ji_{13}} \\ K_{ji_{21}} & K_{ji_{22}} & K_{ji_{23}} \\ K_{ji_{31}} & K_{ji_{32}} & K_{ji_{33}} \end{bmatrix}.
$$

Let us define vector of the right hand side as

$$
\mathbf{r}_j = \begin{bmatrix} -\mathcal{F}(\mathbf{u},p)_{j_1} \\ -\mathcal{F}(\mathbf{u},p)_{j_2} \\ -\mathcal{F}(\mathbf{u},p)_{j_3} \end{bmatrix},
$$

and vector of solution as

$$
\mathbf{d}_i = \begin{bmatrix} h_{x_i} \\ h_{y_i} \\ q_i \end{bmatrix}.
$$

This way, we obtain element stiffness matrix $\mathbf{K}^e$ and element vector of the right hand side $\mathbf{r}^e$ (cf. Figure 4.1). After conventional assemblage procedure of stiffness matrix $\mathbf{K}$ and the right hand side $\mathbf{r}$, we solve the system of linear equations

$$
\mathbf{K}\mathbf{d} = \mathbf{r}
$$

in each iteration of the Newton method.

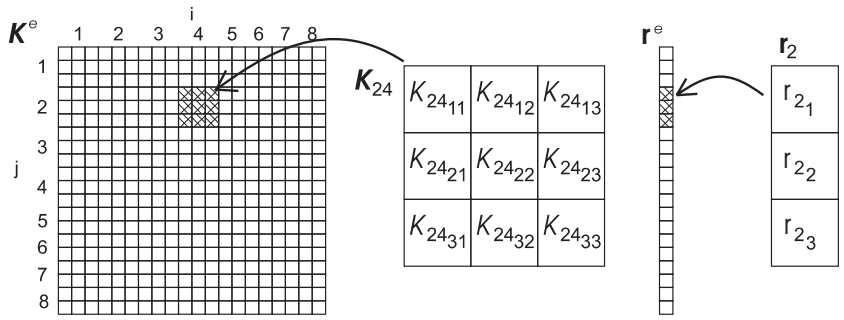


Figure 4.1: Structure of element stiffnes matrix for quadrilateral element

4.2.3 Functionals for the Newton method and their differentials in unsteady case

We extend the theory for the steady case in Section 4.2.1 to the unsteady problem (4.6)–(4.7), again omitting index h in the derivations.

Let us approximate the time derivative in the $(n+1)$-st time layer as

$$\frac{\partial \mathbf{u}}{\partial t} \approx \frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\vartheta},$$

where ϑ is a constant time step.

Functionals for the Newton method for the unsteady case are defined as

$$
\begin{aligned}
F_1(\mathbf{u}^{n+1}, p^{n+1}) &= \frac{1}{\vartheta}\int_\Omega \mathbf{u}^{n+1}\cdot\mathbf{v}\mathrm{d}\Omega + \int_\Omega (\mathbf{u}^{n+1}\cdot\nabla)\mathbf{u}^{n+1}\cdot\mathbf{v}\mathrm{d}\Omega + \\
&+ \nu\int_\Omega \nabla\mathbf{u}^{n+1} : \nabla\mathbf{v}\mathrm{d}\Omega - \int_\Omega p^{n+1}\nabla\cdot\mathbf{v}\mathrm{d}\Omega - \\
&- \int_\Omega \mathbf{f}^{n+1}\cdot\mathbf{v}\mathrm{d}\Omega - \frac{1}{\vartheta}\int_\Omega \mathbf{u}^{n}\cdot\mathbf{v}\mathrm{d}\Omega + \\
&+ \sum_{K=1}^{N}\left\{\frac{1}{\vartheta}\int_{T_K}\tau\mathbf{u}^{n+1}\cdot(\mathbf{u}^{n+1}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega - \frac{\nu}{\vartheta}\int_{T_K}\tau\mathbf{u}^{n+1}\cdot\Delta\mathbf{v}\mathrm{d}\Omega + \right. \\
&+ \frac{1}{\vartheta}\int_{T_K}\tau\mathbf{u}^{n+1}\cdot\nabla\psi\mathrm{d}\Omega + \int_{T_K}\tau(\mathbf{u}^{n+1}\cdot\nabla)\mathbf{u}^{n+1}\cdot(\mathbf{u}^{n+1}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega - \\
&- \nu\int_{T_K}\tau(\mathbf{u}^{n+1}\cdot\nabla)\mathbf{u}^{n+1}\cdot\Delta\mathbf{v}\mathrm{d}\Omega + \int_{T_K}\tau(\mathbf{u}^{n+1}\cdot\nabla)\mathbf{u}^{n+1}\cdot\nabla\psi\mathrm{d}\Omega - \\
&- \nu\int_{T_K}\tau\Delta\mathbf{u}^{n+1}\cdot(\mathbf{u}^{n+1}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega + \nu^2\int_{T_K}\tau\Delta\mathbf{u}^{n+1}\cdot\Delta\mathbf{v}\mathrm{d}\Omega - \\
&- \nu\int_{T_K}\tau\Delta\mathbf{u}^{n+1}\cdot\nabla\psi\mathrm{d}\Omega + \int_{T_K}\tau\nabla p^{n+1}\cdot(\mathbf{u}^{n+1}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega - \\
&- \nu\int_{T_K}\tau\nabla p^{n+1}\cdot\Delta\mathbf{v}\mathrm{d}\Omega + \int_{T_K}\tau\nabla p^{n+1}\cdot\nabla\psi\mathrm{d}\Omega - \\
&- \int_{T_K}\tau\mathbf{f}^{n+1}\cdot(\mathbf{u}^{n+1}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega + \nu\int_{T_K}\tau\mathbf{f}^{n+1}\cdot\Delta\mathbf{v}\mathrm{d}\Omega - \\
&- \int_{T_K}\tau\mathbf{f}^{n+1}\cdot\nabla\psi\mathrm{d}\Omega - \frac{1}{\vartheta}\int_{T_K}\tau\mathbf{u}^{n}\cdot(\mathbf{u}^{n+1}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega + \\
&+ \left.\frac{\nu}{\vartheta}\int_{T_K}\tau\mathbf{u}^{n}\cdot\Delta\mathbf{v}\mathrm{d}\Omega - \frac{1}{\vartheta}\int_{T_K}\tau\mathbf{u}^{n}\cdot\nabla\psi\mathrm{d}\Omega\right\}, \\
F_2(\mathbf{u}^{n+1}, p^{n+1}) &= \int_\Omega \psi\nabla\cdot\mathbf{u}^{n+1}\mathrm{d}\Omega.
\end{aligned}
$$

Notation 2 *For the sake of brevity, another simplification of notation is employed in the following derivations. We omit index $n+1$, and then $\mathbf{u}$ and p denote variables in the $(n+1)$-st time layer. Index of time layer is preserved at variables from other time layers, e.g. $\mathbf{u}^n$.*

Lemma 3 *Assume all functions sufficiently smooth. Then the Gateaux differentials of functionals* $F_1(\mathbf{u}, p)$ *and* $F_2(\mathbf{u}, p)$ *are*

$$< DF_1(\mathbf{u}, p), [\mathbf{h}, q] >=$$

$$\begin{aligned}
&= \frac{1}{\vartheta}\int_\Omega \mathbf{h}\cdot\mathbf{v}\mathrm{d}\Omega + \int_\Omega (\mathbf{h}\cdot\nabla)\mathbf{u}\cdot\mathbf{v}\mathrm{d}\Omega + \int_\Omega (\mathbf{u}\cdot\nabla)\mathbf{h}\cdot\mathbf{v}\mathrm{d}\Omega + \nu\int_\Omega \nabla\mathbf{h}:\nabla\mathbf{v}\mathrm{d}\Omega - \int_\Omega q\nabla\cdot\mathbf{v}\mathrm{d}\Omega + \\
&+ \sum_{K=1}^{N}\left\{ \frac{1}{\vartheta}\int_{T_K}\tau\mathbf{h}\cdot(\mathbf{u}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega + \frac{1}{\vartheta}\int_{T_K}\tau\mathbf{u}\cdot(\mathbf{h}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega - \frac{\nu}{\vartheta}\int_{T_K}\tau\mathbf{h}\cdot\Delta\mathbf{v}\mathrm{d}\Omega + \right. \\
&+ \frac{1}{\vartheta}\int_{T_K}\tau\mathbf{h}\cdot\nabla\psi\mathrm{d}\Omega + \int_{T_K}\tau(\mathbf{h}\cdot\nabla)\mathbf{u}\cdot(\mathbf{u}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega + \int_{T_K}\tau(\mathbf{u}\cdot\nabla)\mathbf{h}\cdot(\mathbf{u}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega + \\
&+ \int_{T_K}\tau(\mathbf{u}\cdot\nabla)\mathbf{u}\cdot(\mathbf{h}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega - \nu\int_{T_K}\tau(\mathbf{h}\cdot\nabla)\mathbf{u}\cdot\Delta\mathbf{v}\mathrm{d}\Omega - \nu\int_{T_K}\tau(\mathbf{u}\cdot\nabla)\mathbf{h}\cdot\Delta\mathbf{v}\mathrm{d}\Omega + \\
&+ \int_{T_K}\tau(\mathbf{h}\cdot\nabla)\mathbf{u}\cdot\nabla\psi\mathrm{d}\Omega + \int_{T_K}\tau(\mathbf{u}\cdot\nabla)\mathbf{h}\cdot\nabla\psi\mathrm{d}\Omega - \nu\int_{T_K}\tau\Delta\mathbf{h}\cdot(\mathbf{u}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega - \\
&- \nu\int_{T_K}\tau\Delta\mathbf{u}\cdot(\mathbf{h}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega + \nu^2\int_{T_K}\tau\Delta\mathbf{h}\cdot\Delta\mathbf{v}\mathrm{d}\Omega - \nu\int_{T_K}\tau\Delta\mathbf{h}\cdot\nabla\psi\mathrm{d}\Omega + \\
&+ \int_{T_K}\tau\nabla q\cdot(\mathbf{u}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega + \int_{T_K}\tau\nabla p\cdot(\mathbf{h}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega - \nu\int_{T_K}\tau\nabla q\cdot\Delta\mathbf{v}\mathrm{d}\Omega + \\
&+ \left.\int_{T_K}\tau\nabla q\cdot\nabla\psi\mathrm{d}\Omega - \int_{T_K}\tau\mathbf{f}\cdot(\mathbf{h}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega - \frac{1}{\vartheta}\int_{T_K}\tau\mathbf{u}^n(\mathbf{h}\cdot\nabla)\mathbf{v}\mathrm{d}\Omega\right\},
\end{aligned}$$

and

$$< DF_2(\mathbf{u}, p), [\mathbf{h}, q] >= \int_\Omega \psi\nabla\cdot\mathbf{h}\mathrm{d}\Omega.$$

As for the steady case, we formally introduce the functional

$$\mathcal{F}(\mathbf{u}, p) = F_1(\mathbf{u}, p) + F_2(\mathbf{u}, p), \tag{4.16}$$

and its differential

$$< D\mathcal{F}(\mathbf{u}, p), [\mathbf{h}, q] >=< DF_1(\mathbf{u}, p), [\mathbf{h}, q] > + < DF_2(\mathbf{u}, p), [\mathbf{h}, q] > . \tag{4.17}$$

4.2.4 Matrices for the finite element method in unsteady case

Let us derive the element matrix for a finite element T_K. We substitute for h_x, h_y, and q restricted to the element as in Section 4.2.2 and use the vector shape functions as test functions.

We obtain elements of the ji-submatrix $\mathbf{K}_{ji}$ of the element stiffness matrix $\mathbf{K}^e$:

$$
\begin{aligned}
K_{ji_{11}}(\mathbf{u},p) &= \frac{1}{\vartheta}\int_\Omega N_i N_j \mathrm{d}\Omega + \int_\Omega N_i \frac{\partial u_x}{\partial x} N_j \mathrm{d}\Omega + \int_\Omega \left(u_x \frac{\partial N_i}{\partial x} + u_y \frac{\partial N_i}{\partial y}\right) N_j \mathrm{d}\Omega + \\
&+ \nu \int_\Omega \left(\frac{\partial N_i}{\partial x}\frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y}\frac{\partial N_j}{\partial y}\right) \mathrm{d}\Omega + \\
&+ \sum_K \left\{ \frac{1}{\vartheta}\int_{T_K} \tau N_i \left(u_x \frac{\partial N_j}{\partial x} + u_y \frac{\partial N_j}{\partial y}\right) \mathrm{d}\Omega + \frac{1}{\vartheta}\int_{T_K} \tau u_x N_i \frac{\partial N_j}{\partial x} \mathrm{d}\Omega - \right. \\
&- \frac{\nu}{\vartheta}\int_{T_K} \tau N_i \left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right) \mathrm{d}\Omega + \\
&+ \int_{T_K} \tau N_i \frac{\partial u_x}{\partial x} \left(u_x \frac{\partial N_j}{\partial x} + u_y \frac{\partial N_j}{\partial y}\right) \mathrm{d}\Omega + \\
&+ \int_{T_K} \tau \left(u_x \frac{\partial N_i}{\partial x} + u_y \frac{\partial N_i}{\partial y}\right) \left(u_x \frac{\partial N_j}{\partial x} + u_y \frac{\partial N_j}{\partial y}\right) \mathrm{d}\Omega + \\
&+ \int_{T_K} \tau \left(u_x \frac{\partial u_x}{\partial x} + u_y \frac{\partial u_x}{\partial y}\right) N_i \frac{\partial N_j}{\partial x} \mathrm{d}\Omega - \\
&- \nu \int_{T_K} \tau N_i \frac{\partial u_x}{\partial x} \left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right) \mathrm{d}\Omega - \\
&- \nu \int_{T_K} \tau \left(u_x \frac{\partial N_i}{\partial x} + u_y \frac{\partial N_i}{\partial y}\right) \left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right) \mathrm{d}\Omega - \\
&- \nu \int_{T_K} \tau \left(\frac{\partial^2 N_i}{\partial x^2} + \frac{\partial^2 N_i}{\partial y^2}\right) \left(u_x \frac{\partial N_j}{\partial x} + u_y \frac{\partial N_j}{\partial y}\right) \mathrm{d}\Omega - \\
&- \nu \int_{T_K} \tau \left(\frac{\partial^2 u_x}{\partial x^2} + \frac{\partial^2 u_x}{\partial y^2}\right) N_i \frac{\partial N_j}{\partial x} \mathrm{d}\Omega + \\
&+ \nu^2 \int_{T_K} \tau \left(\frac{\partial^2 N_i}{\partial x^2} + \frac{\partial^2 N_i}{\partial y^2}\right) \left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right) \mathrm{d}\Omega + \\
&+ \left. \int_{T_K} \tau \frac{\partial p}{\partial x} N_i \frac{\partial N_j}{\partial x} \mathrm{d}\Omega - \int_{T_K} \tau f_x N_i \frac{\partial N_j}{\partial x} \mathrm{d}\Omega - \frac{1}{\vartheta}\int_{T_K} \tau u_x^n N_i \frac{\partial N_j}{\partial x} \mathrm{d}\Omega \right\},
\end{aligned}
$$

$$
\begin{aligned}
K_{ji_{12}}(\mathbf{u},p) &= \int_\Omega N_i \frac{\partial u_x}{\partial y} N_j \mathrm{d}\Omega + \\
&+ \sum_K \left\{ \frac{1}{\vartheta}\int_{T_K} \tau u_x N_i \frac{\partial N_j}{\partial y} \mathrm{d}\Omega + \int_{T_K} \tau N_i \frac{\partial u_x}{\partial y} \left(u_x \frac{\partial N_j}{\partial x} + u_y \frac{\partial N_j}{\partial y}\right) \mathrm{d}\Omega + \right. \\
&+ \int_{T_K} \tau \left(u_x \frac{\partial u_x}{\partial x} + u_y \frac{\partial u_x}{\partial y}\right) N_i \frac{\partial N_j}{\partial y} \mathrm{d}\Omega - \\
&- \nu \int_{T_K} \tau N_i \frac{\partial u_x}{\partial y} \left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right) \mathrm{d}\Omega - \\
&- \nu \int_{T_K} \tau \left(\frac{\partial^2 u_x}{\partial x^2} + \frac{\partial^2 u_x}{\partial y^2}\right) N_i \frac{\partial N_j}{\partial y} \mathrm{d}\Omega + \\
&+ \left. \int_{T_K} \tau \frac{\partial p}{\partial x} N_i \frac{\partial N_j}{\partial y} \mathrm{d}\Omega - \int_{T_K} \tau f_x N_i \frac{\partial N_j}{\partial y} \mathrm{d}\Omega - \frac{1}{\vartheta}\int_{T_K} \tau u_x^n N_i \frac{\partial N_j}{\partial y} \mathrm{d}\Omega \right\},
\end{aligned}
$$

$$
\begin{aligned}
K_{ji_{13}}(\mathbf{u},p) &= -\int_\Omega M_i \frac{\partial N_j}{\partial x}\mathrm{d}\Omega + \sum_K \left\{ \int_{T_K} \tau \frac{\partial M_i}{\partial x}\left(u_x\frac{\partial N_j}{\partial x} + u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega - \right. \\
&- \left. \nu \int_{T_K} \tau \frac{\partial M_i}{\partial x}\left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega \right\},
\end{aligned}
$$

$$
\begin{aligned}
K_{ji_{21}}(\mathbf{u},p) &= \int_\Omega N_i \frac{\partial u_y}{\partial x} N_j \mathrm{d}\Omega + \sum_K \left\{ \frac{1}{\vartheta}\int_{T_K} \tau u_y N_i \frac{\partial N_j}{\partial x}\mathrm{d}\Omega + \right. \\
&+ \int_{T_K} \tau N_i \frac{\partial u_y}{\partial x}\left(u_x\frac{\partial N_j}{\partial x} + u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega + \\
&+ \int_{T_K} \tau \left(u_x\frac{\partial u_y}{\partial x} + u_y\frac{\partial u_y}{\partial y}\right) N_i \frac{\partial N_j}{\partial x}\mathrm{d}\Omega - \\
&- \nu \int_{T_K} \tau N_i \frac{\partial u_y}{\partial x}\left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega - \\
&- \nu \int_{T_K} \tau \left(\frac{\partial^2 u_y}{\partial x^2} + \frac{\partial^2 u_y}{\partial y^2}\right) N_i \frac{\partial N_j}{\partial x}\mathrm{d}\Omega + \\
&+ \left. \int_{T_K} \tau \frac{\partial p}{\partial y} N_i \frac{\partial N_j}{\partial x}\mathrm{d}\Omega - \int_{T_K} \tau f_y N_i \frac{\partial N_j}{\partial x}\mathrm{d}\Omega - \frac{1}{\vartheta}\int_{T_K} \tau u_y^n N_i \frac{\partial N_j}{\partial x}\mathrm{d}\Omega \right\},
\end{aligned}
$$

$$
\begin{aligned}
K_{ji_{22}}(\mathbf{u},p) &= \frac{1}{\vartheta}\int_\Omega N_i N_j \mathrm{d}\Omega + \int_\Omega N_i \frac{\partial u_y}{\partial y} N_j \mathrm{d}\Omega + \int_\Omega \left(u_x\frac{\partial N_i}{\partial x} + u_y\frac{\partial N_i}{\partial y}\right) N_j \mathrm{d}\Omega + \\
&+ \nu \int_\Omega \left(\frac{\partial N_i}{\partial x}\frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y}\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega + \\
&+ \sum_K \left\{ \frac{1}{\vartheta}\int_{T_K} \tau N_i \left(u_x\frac{\partial N_j}{\partial x} + u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega + \frac{1}{\vartheta}\int_{T_K} \tau u_y N_i \frac{\partial N_j}{\partial y}\mathrm{d}\Omega - \right. \\
&- \frac{\nu}{\vartheta}\int_{T_K} \tau N_i \left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega + \int_{T_K} \tau N_i \frac{\partial u_y}{\partial y}\left(u_x\frac{\partial N_j}{\partial x} + u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega + \\
&+ \int_{T_K} \tau \left(u_x\frac{\partial N_i}{\partial x} + u_y\frac{\partial N_i}{\partial y}\right)\left(u_x\frac{\partial N_j}{\partial x} + u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega + \\
&+ \int_{T_K} \tau \left(u_x\frac{\partial u_y}{\partial x} + u_y\frac{\partial u_y}{\partial y}\right) N_i \frac{\partial N_j}{\partial y}\mathrm{d}\Omega - \nu \int_{T_K} \tau N_i \frac{\partial u_y}{\partial y}\left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega - \\
&- \nu \int_K \tau \left(u_x\frac{\partial N_i}{\partial x} + u_y\frac{\partial N_i}{\partial y}\right)\left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega - \\
&- \nu \int_{T_K} \tau \left(\frac{\partial^2 N_i}{\partial x^2} + \frac{\partial^2 N_i}{\partial y^2}\right)\left(u_x\frac{\partial N_j}{\partial x} + u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega - \\
&- \nu \int_{T_K} \tau \left(\frac{\partial^2 u_y}{\partial x^2} + \frac{\partial^2 u_y}{\partial y^2}\right) N_i \frac{\partial N_j}{\partial y}\mathrm{d}\Omega + \\
&+ \nu^2 \int_{T_K} \tau \left(\frac{\partial^2 N_i}{\partial x^2} + \frac{\partial^2 N_i}{\partial y^2}\right)\left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega + \\
&+ \left. \int_{T_K} \tau \frac{\partial p}{\partial y} N_i \frac{\partial N_j}{\partial y}\mathrm{d}\Omega - \int_{T_K} \tau f_y N_i \frac{\partial N_j}{\partial y}\mathrm{d}\Omega - \frac{1}{\vartheta}\int_{T_K} \tau u_y^n N_i \frac{\partial N_j}{\partial y}\mathrm{d}\Omega \right\},
\end{aligned}
$$

$$
\begin{aligned}
K_{ji_{23}}(\mathbf{u},p) &= -\int_\Omega M_i \frac{\partial N_j}{\partial y}\mathrm{d}\Omega + \sum_K \left\{ \int_{T_K} \tau \frac{\partial M_i}{\partial y}\left(u_x\frac{\partial N_j}{\partial x} + u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega - \right. \\
&- \left. \nu \int_{T_K} \tau \frac{\partial M_i}{\partial y}\left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega \right\},
\end{aligned}
$$

$$
\begin{aligned}
K_{ji_{31}}(\mathbf{u},p) &= \int_\Omega \frac{\partial N_i}{\partial x} M_j \mathrm{d}\Omega + \\
&+ \sum_K \left\{ \frac{1}{\vartheta}\int_{T_K} \tau N_i \frac{\partial M_j}{\partial x}\mathrm{d}\Omega + \int_{T_K} \tau N_i \frac{\partial u_x}{\partial x}\frac{\partial M_j}{\partial x}\mathrm{d}\Omega + \int_{T_K} \tau N_i \frac{\partial u_y}{\partial x}\frac{\partial M_j}{\partial y}\mathrm{d}\Omega + \right. \\
&+ \left. \int_{T_K} \tau \left(u_x\frac{\partial N_i}{\partial x} + u_y\frac{\partial N_i}{\partial y}\right)\frac{\partial M_j}{\partial x}\mathrm{d}\Omega - \nu\int_{T_K} \tau \left(\frac{\partial^2 N_i}{\partial x^2} + \frac{\partial^2 N_i}{\partial y^2}\right)\frac{\partial M_j}{\partial x}\mathrm{d}\Omega \right\},
\end{aligned}
$$

$$
\begin{aligned}
K_{ji_{32}}(\mathbf{u},p) &= \int_\Omega \frac{\partial N_i}{\partial y} M_j \mathrm{d}\Omega + \\
&+ \sum_K \left\{ \frac{1}{\vartheta}\int_{T_K} \tau N_i \frac{\partial M_j}{\partial y}\mathrm{d}\Omega + \int_{T_K} \tau N_i \frac{\partial u_x}{\partial y}\frac{\partial M_j}{\partial x}\mathrm{d}\Omega + \int_{T_K} \tau N_i \frac{\partial u_y}{\partial y}\frac{\partial M_j}{\partial y}\mathrm{d}\Omega + \right. \\
&+ \left. \int_{T_K} \tau \left(u_x\frac{\partial N_i}{\partial x} + u_y\frac{\partial N_i}{\partial y}\right)\frac{\partial M_j}{\partial y}\mathrm{d}\Omega - \nu\int_{T_K} \tau \left(\frac{\partial^2 N_i}{\partial x^2} + \frac{\partial^2 N_i}{\partial y^2}\right)\frac{\partial M_j}{\partial y}\mathrm{d}\Omega \right\},
\end{aligned}
$$

$$
K_{ji_{33}}(\mathbf{u},p) = \sum_K \int_{T_K} \tau \left(\frac{\partial M_i}{\partial x}\frac{\partial M_j}{\partial x} + \frac{\partial M_i}{\partial y}\frac{\partial M_j}{\partial y}\right)\mathrm{d}\Omega.
$$

Let us remind matrix $\mathbf{K}_{ji}$

$$
\mathbf{K}_{ji} = \begin{bmatrix} K_{ji_{11}} & K_{ji_{12}} & K_{ji_{13}} \\ K_{ji_{21}} & K_{ji_{22}} & K_{ji_{23}} \\ K_{ji_{31}} & K_{ji_{32}} & K_{ji_{33}} \end{bmatrix},
$$

vector of the right hand side

$$
\mathbf{r}_j = \begin{bmatrix} -\mathcal{F}(\mathbf{u},p)_{j_1} \\ -\mathcal{F}(\mathbf{u},p)_{j_2} \\ -\mathcal{F}(\mathbf{u},p)_{j_3} \end{bmatrix},
$$

and vector of solution

$$
\mathbf{d}_i = \begin{bmatrix} h_{x_i} \\ h_{y_i} \\ q_i \end{bmatrix}.
$$

Elements of the vector of the right hand side are

$$
\begin{aligned}
\mathcal{F}(\mathbf{u},p)_{j_1} &= \frac{1}{\vartheta}\int_\Omega u_x N_j \mathrm{d}\Omega + \int_\Omega \left(u_x\frac{\partial u_x}{\partial x} + u_y\frac{\partial u_x}{\partial y}\right) N_j \mathrm{d}\Omega + \nu\int_\Omega \left(\frac{\partial u_x}{\partial x}\frac{\partial N_j}{\partial x} + \frac{\partial u_x}{\partial y}\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega - \\
&- \int_\Omega p\frac{\partial N_j}{\partial x}\mathrm{d}\Omega - \int_\Omega f_x N_j \mathrm{d}\Omega - \frac{1}{\vartheta}\int_\Omega u_x^n N_j \mathrm{d}\Omega + \\
&+ \sum_K \left\{ \frac{1}{\vartheta}\int_{T_K} \tau u_x \left(u_x\frac{\partial N_j}{\partial x} + u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega - \frac{\nu}{\vartheta}\int_{T_K}\tau u_x\left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega + \right. \\
&+ \int_{T_K}\tau\left(u_x\frac{\partial u_x}{\partial x} + u_y\frac{\partial u_x}{\partial y}\right)\left(u_x\frac{\partial N_j}{\partial x} + u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega - \\
&- \nu\int_{T_K}\tau\left(u_x\frac{\partial u_x}{\partial x} + u_y\frac{\partial u_x}{\partial y}\right)\left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega - \\
&- \nu\int_{T_K}\tau\left(\frac{\partial^2 u_x}{\partial x^2} + \frac{\partial^2 u_x}{\partial y^2}\right)\left(u_x\frac{\partial N_j}{\partial x} + u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega + \\
&+ \nu^2\int_{T_K}\tau\left(\frac{\partial^2 u_x}{\partial x^2} + \frac{\partial^2 u_x}{\partial y^2}\right)\left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega + \\
&+ \int_{T_K}\tau\frac{\partial p}{\partial x}\left(u_x\frac{\partial N_j}{\partial x} + u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega - \nu\int_{T_K}\tau\frac{\partial p}{\partial x}\left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega - \\
&- \int_{T_K}\tau f_x\left(u_x\frac{\partial N_j}{\partial x} + u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega + \nu\int_{T_K}\tau f_x\left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega - \\
&- \left.\frac{1}{\vartheta}\int_{T_K}\tau u_x^n\left(u_x\frac{\partial N_j}{\partial x} + u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega + \frac{\nu}{\vartheta}\int_{T_K}\tau u_x^n\left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega\right\},
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{F}(\mathbf{u},p)_{j_2} &= \frac{1}{\vartheta}\int_\Omega u_y N_j \mathrm{d}\Omega + \int_\Omega \left(u_x\frac{\partial u_y}{\partial x} + u_y\frac{\partial u_y}{\partial y}\right) N_j \mathrm{d}\Omega + \nu\int_\Omega \left(\frac{\partial u_y}{\partial x}\frac{\partial N_j}{\partial x} + \frac{\partial u_y}{\partial y}\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega - \\
&- \int_\Omega p\frac{\partial N_j}{\partial y}\mathrm{d}\Omega - \int_\Omega f_y N_j \mathrm{d}\Omega - \frac{1}{\vartheta}\int_\Omega u_y^n N_j \mathrm{d}\Omega + \\
&+ \sum_K \left\{ \frac{1}{\vartheta}\int_{T_K} \tau u_y \left(u_x\frac{\partial N_j}{\partial x} + u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega - \frac{\nu}{\vartheta}\int_{T_K}\tau u_y\left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega + \right. \\
&+ \int_{T_K}\tau\left(u_x\frac{\partial u_y}{\partial x} + u_y\frac{\partial u_y}{\partial y}\right)\left(u_x\frac{\partial N_j}{\partial x} + u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega - \\
&- \nu\int_{T_K}\tau\left(u_x\frac{\partial u_y}{\partial x} + u_y\frac{\partial u_y}{\partial y}\right)\left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega - \\
&- \nu\int_{T_K}\tau\left(\frac{\partial^2 u_y}{\partial x^2} + \frac{\partial^2 u_y}{\partial y^2}\right)\left(u_x\frac{\partial N_j}{\partial x} + u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega + \\
&+ \nu^2\int_{T_K}\tau\left(\frac{\partial^2 u_y}{\partial x^2} + \frac{\partial^2 u_y}{\partial y^2}\right)\left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega + \\
&+ \int_{T_K}\tau\frac{\partial p}{\partial y}\left(u_x\frac{\partial N_j}{\partial x} + u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega - \nu\int_{T_K}\tau\frac{\partial p}{\partial y}\left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega - \\
&- \int_{T_K}\tau f_y\left(u_x\frac{\partial N_j}{\partial x} + u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega + \nu\int_{T_K}\tau f_y\left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega - \\
&- \left.\frac{1}{\vartheta}\int_{T_K}\tau u_y^n\left(u_x\frac{\partial N_j}{\partial x} + u_y\frac{\partial N_j}{\partial y}\right)\mathrm{d}\Omega + \frac{\nu}{\vartheta}\int_{T_K}\tau u_y^n\left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right)\mathrm{d}\Omega\right\},
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{F}(\mathbf{u},p)_{j_3} &= \int_\Omega\left(\frac{\partial u_x}{\partial x} + \frac{\partial u_y}{\partial y}\right)M_j\mathrm{d}\Omega + \sum_K\left\{\frac{1}{\vartheta}\int_{T_K}\tau\left(u_x\frac{\partial M_j}{\partial x} + u_y\frac{\partial M_j}{\partial y}\right)\mathrm{d}\Omega + \right. \\
&+ \int_{T_K}\tau\left(u_x\frac{\partial u_x}{\partial x} + u_y\frac{\partial u_x}{\partial y}\right)\frac{\partial M_j}{\partial x}\mathrm{d}\Omega + \int_{T_K}\tau\left(u_x\frac{\partial u_y}{\partial x} + u_y\frac{\partial u_y}{\partial y}\right)\frac{\partial M_j}{\partial y}\mathrm{d}\Omega - \\
&- \nu\int_{T_K}\tau\left(\frac{\partial^2 u_x}{\partial x^2} + \frac{\partial^2 u_x}{\partial y^2}\right)\frac{\partial M_j}{\partial x}\mathrm{d}\Omega - \nu\int_{T_K}\tau\left(\frac{\partial^2 u_y}{\partial x^2} + \frac{\partial^2 u_y}{\partial y^2}\right)\frac{\partial M_j}{\partial y}\mathrm{d}\Omega + \\
&+ \int_{T_K}\tau\frac{\partial p}{\partial x}\frac{\partial M_j}{\partial x}\mathrm{d}\Omega + \int_{T_K}\tau\frac{\partial p}{\partial y}\frac{\partial M_j}{\partial y}\mathrm{d}\Omega - \\
&- \int_{T_K}\tau f_x\frac{\partial M_j}{\partial x}\mathrm{d}\Omega - \int_{T_K}\tau f_y\frac{\partial M_j}{\partial y}\mathrm{d}\Omega - \left.\frac{1}{\vartheta}\int_{T_K}\tau\left(u_x^n\frac{\partial M_j}{\partial x} + u_y^n\frac{\partial M_j}{\partial y}\right)\mathrm{d}\Omega\right\}.
\end{aligned}
$$

As in Section 4.2.2, after we obtain element stiffness matrix $\mathbf{K}^e$ and element vector of the right hand side $\mathbf{r}^e$ (cf. Figure 4.1) and perform the assemblage procedure of stiffness matrix $\mathbf{K}$ and the right hand side $\mathbf{r}$, we solve the system of linear equations

$$\mathbf{K}\mathbf{d} = \mathbf{r}$$

in each iteration of the Newton method.

Once we obtain the solution in a particular time layer, we solve the problem in next time layer and use the previous solution as the initial value for the Newton method. This is repeated, until we reach the desired time.

4.2.5 Computation of stabilization parameter

Following the suggestions of Franca and Madureira in [23], we compute the stabilization parameter τ as

$$\tau = \frac{\xi(\mathrm{Re}_K(\mathbf{x}))}{\sqrt{\lambda_K} \mid \mathbf{u}(\mathbf{x}) \mid_2}, \tag{4.18}$$

where

$$\begin{aligned}
\mathrm{Re}_K(\mathbf{x}) &= \frac{\mid \mathbf{u}(\mathbf{x}) \mid_2}{4\sqrt{\lambda_K}\nu}, \\
\xi(\mathrm{Re}_K(\mathbf{x})) &= \begin{cases} \mathrm{Re}_K(\mathbf{x}), & 0 \leq \mathrm{Re}_K(\mathbf{x}) < 1 \\ 1, & \mathrm{Re}_K(\mathbf{x}) \geq 1 \end{cases}, \\
\lambda_K &= \max_{0 \neq \mathbf{v} \in (R_2(T_K)/\mathbb{R})^2} \frac{\|\Delta \mathbf{v}\|^2_{0,T_K}}{\|\nabla \mathbf{v}\|^2_{0,T_K}}, \\
\mid \mathbf{u}(\mathbf{x}) \mid_2 &= \left(\sum_{i=1}^{2} \mid u_i(\mathbf{x}) \mid^2 \right)^{\frac{1}{2}}.
\end{aligned}$$

Parameter λ_K is computed for each element as the largest eigenvalue of the problem

$$(\Delta \mathbf{w}, \Delta \mathbf{v}) = \lambda_K (\nabla \mathbf{w}, \nabla \mathbf{v}), \;\; \forall \mathbf{v} \in (R_2(T_K)/\mathbb{R})^2. \tag{4.19}$$

This is done once, before entering the main computational loop of the Newton method, since λ_K is not a function of velocity and depends only on the computational mesh and shape functions on element K.

Let us focus on computing λ_K more precisely. Problem (4.19) can be written as

$$\int_{T_K} \Delta \mathbf{w} \cdot \Delta \mathbf{v} \mathrm{d}\Omega = \lambda_K \int_{T_K} \nabla \mathbf{w} : \nabla \mathbf{v} \mathrm{d}\Omega, \;\; \forall \mathbf{v} \in (R_2(T_K)/\mathbb{R})^2. \tag{4.20}$$

In the finite element context, similarly to Sections 4.2.2 and 4.2.4, we substitute

$$\begin{aligned}
w_x &= \sum_{i=1}^{N_u} w_{xi} N_i, \\
w_y &= \sum_{i=1}^{N_u} w_{y_i} N_i,
\end{aligned}$$

and use the vector shape functions

$$\begin{aligned}
\mathbf{v} &= (N_j, 0), \\
\mathbf{v} &= (0, N_j)
\end{aligned}$$

as test functions to get two equations from the scalar one (4.20). It leads to

$$\begin{aligned}
\sum_{i=1}^{N_u} \int_{T_K} w_{x_i} \left(\frac{\partial^2 N_i}{\partial x^2} + \frac{\partial^2 N_i}{\partial y^2} \right) \left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2} \right) \mathrm{d}\Omega &= \lambda_K \sum_{i=1}^{N_u} \int_{T_K} w_{x_i} \left(\frac{\partial N_i}{\partial x} \frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y} \frac{\partial N_j}{\partial y} \right) \mathrm{d}\Omega, \\
\sum_{i=1}^{N_u} \int_{T_K} w_{y_i} \left(\frac{\partial^2 N_i}{\partial x^2} + \frac{\partial^2 N_i}{\partial y^2} \right) \left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2} \right) \mathrm{d}\Omega &= \lambda_K \sum_{i=1}^{N_u} \int_{T_K} w_{y_i} \left(\frac{\partial N_i}{\partial x} \frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y} \frac{\partial N_j}{\partial y} \right) \mathrm{d}\Omega.
\end{aligned}$$

Let us create element matrices $\mathbf{A}$ and $\mathbf{B}$ for the purpose of computation of the largest eigenvalue of this problem as

$$\mathbf{A}_{ji} = \begin{bmatrix} \int_{T_K} \left(\frac{\partial^2 N_i}{\partial x^2} + \frac{\partial^2 N_i}{\partial y^2}\right)\left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right) \mathrm{d}\Omega & 0 \\ 0 & \int_{T_K} \left(\frac{\partial^2 N_i}{\partial x^2} + \frac{\partial^2 N_i}{\partial y^2}\right)\left(\frac{\partial^2 N_j}{\partial x^2} + \frac{\partial^2 N_j}{\partial y^2}\right) \mathrm{d}\Omega \end{bmatrix},$$

$$\mathbf{B}_{ji} = \begin{bmatrix} \int_{T_K} \left(\frac{\partial N_i}{\partial x}\frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y}\frac{\partial N_j}{\partial y}\right) \mathrm{d}\Omega & 0 \\ 0 & \int_{T_K} \left(\frac{\partial N_i}{\partial x}\frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y}\frac{\partial N_j}{\partial y}\right) \mathrm{d}\Omega \end{bmatrix}.$$

Now, we need to find the largest eigenvalue of the generalized matrix eigenvalue problem

$$\mathbf{A}\mathbf{w}_K = \lambda_K \mathbf{B}\mathbf{w}_K \tag{4.21}$$

for each element. Here, λ_K is the desired eigenvalue and $\mathbf{w}_K$ is the corresponding eigenvector, which is not used in stabilization.

Recommended method for solving this problem in [23] is the power method. But several difficulties are hidden behind it:

1. The power method is designed for finding of the largest eigenvalue and the corresponding eigenvector of the problem $\mathbf{A}\mathbf{w} = \lambda_K \mathbf{w}$ and not for the generalized problem. We need to transform problem (4.21) to the ordinary problem of eigenvalues. Possible way without necessity of inverting full matrix is sketched. We decompose matrix $\mathbf{B}$ by Choleski's method and find $\mathbf{L}$ such that

   $$\mathbf{B} = \mathbf{L}\mathbf{L}^T,$$

   and $\mathbf{L}$ is lower triangular matrix. Its inversion is simpler, and when we have it, we get

   $$\mathbf{L}^{-1}\mathbf{A}\mathbf{w}_K = \lambda_K \mathbf{L}^T \mathbf{w}_K.$$

   Let us denote $\mathbf{z}_K = \mathbf{L}^T\mathbf{w}_K$ or $\mathbf{w}_K = \mathbf{L}^{-T}\mathbf{z}_K$. After substitution, we have

   $$\mathbf{L}^{-1}\mathbf{A}\mathbf{L}^{-T}\mathbf{z}_K = \lambda_K \mathbf{z}_K.$$

   If we denote $\mathbf{G} = \mathbf{L}^{-1}\mathbf{A}\mathbf{L}^{-T}$, we can solve the ordinary problem of eigenvalues

   $$\mathbf{G}\mathbf{z}_K = \lambda_K \mathbf{z}_K.$$

   It is clear, that applied transformations do not take effect on eigenvalues of the generalized problem.

2. During realizing Choleski's decomposition (as for computing an inverse matrix), we need $\mathbf{B}$ to be nonsingular. But obtained matrix, which is similar to element stiffness matrix without application of boundary conditions, is singular.

   Recommended way to regularize it is to fix corresponding number of degrees of freedom. But since this is done by putting unities on diagonal and zeros on relevant columns and rows of matrix $\mathbf{B}$, this way could affect the largest eigenvalue, if it is less then one.

   We experienced, that more suitable way to regularize the matrix is to 'cut off' two rows and columns from both matrices $\mathbf{A}$ and $\mathbf{B}$. As far as we have tested this way, it has taken no effect on the largest eigenvalue for different omited rows and columns. The only restriction is to omit one for each component of velocity.

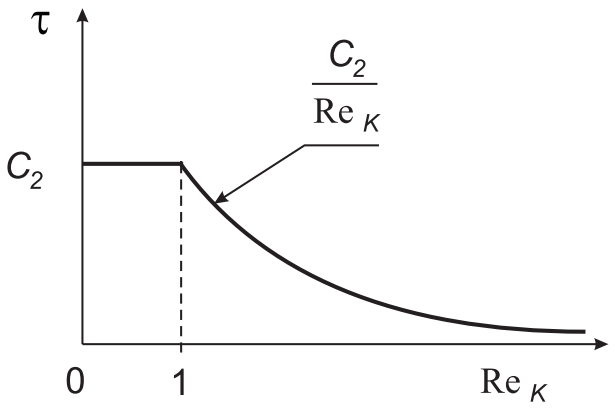


Figure 4.2: Plot of $\tau(\mathrm{Re}_K)$

Let us investigate the dependence of τ on local Reynolds number $\mathrm{Re}_K(\mathbf{x})$ given by (4.18). We can observe that $\mathrm{Re}_K(\mathbf{x})$ is a linear function of $\mid \mathbf{u}(\mathbf{x}) \mid_2$ for constant viscosity on element K, i.e.

$$\mathrm{Re}_K(\mathbf{x}) = C_1 \mid \mathbf{u}(\mathbf{x}) \mid_2, \tag{4.22}$$

where $C_1 = \frac{1}{4\sqrt{\lambda_K}\nu}$.

Substituting (4.22) in (4.18) we get

$$\tau(\mathrm{Re}_K(\mathbf{x}), \mathbf{x}) \;=\; \begin{cases} C_2, & 0 \le \mathrm{Re}_K(\mathbf{x}) < 1 \\ \frac{C_3}{|\mathbf{u}(\mathbf{x})|_2} = \frac{C_2}{\mathrm{Re}_K(\mathbf{x})}, & \mathrm{Re}_K(\mathbf{x}) \ge 1 \end{cases},$$

where $C_2 = \frac{1}{4\lambda_K \nu}$ and $C_3 = \frac{1}{\sqrt{\lambda_K}}$, cf. Figure 4.2.

4.3 Accuracy of the stabilized method

The aspect of accuracy of the stabilized method was mentioned already in [53]. In numerical experiments presented therein, a loss of accuracy was observed. As it is inherited to the stabilized methods of this family, it is always present in such calculations. However, in [8] and [10], two methods for quantification of this loss of accuracy were presented. In this way, it is possible to get an idea of the importance of it, and so to estimate the cost we pay for the stabilized solution at a higher Reynolds number. The evaluation of the loss of accuracy is presented and demonstrated only for the steady Navier-Stokes problem. The extension to the unsteady case could be derived analogously.

One source of the loss of accuracy is the violation of the continuity equation through the non-zero term of element matrix discussed in Sections 4.2.2 and 4.2.4,

$$K_{ji_{33}}(\mathbf{u}, p) \;=\; \sum_K \int_{T_K} \tau \left(\frac{\partial M_i}{\partial x}\frac{\partial M_j}{\partial x} + \frac{\partial M_i}{\partial y}\frac{\partial M_j}{\partial y} \right) \mathrm{d}\Omega.$$

This term introduces dependence on pressure into the continuity equation and affects the presumed incompressibility. Since derivatives of shape functions in $K_{ji_{33}}$ are independent of solution, and since τ is never zero (cf. Figure 4.2), $K_{ji_{33}}$ does not vanish for converged solution. However, Figure 4.2 gives a hope: we can observe, that τ is decreasing for higher local Reynolds number, therefore described perturbation of the continuity equation is also decreasing for higher Re.

A straightforward approach to evaluation of the error by stabilization was presented in [8]. The effect of stabilization was simply evaluated as the difference between discrete solutions obtained with and without stabilization as

$$\delta_\eta = \sqrt{\frac{\sum_{i=1}^{n}(\eta_{sGLS_i} - \eta_{Newton_i})^2}{\sum_{i=1}^{n}\eta_{Newton_i}^2}} \cdot 100 \ \ [\%], \tag{4.23}$$

where η represents in turn u_{h1}, u_{h2} and p_h, n denotes number of nodes with η given, η_{sGLS} denotes the solution obtained by the semiGLS algorithm and η_{Newton} denotes the solution obtained by the Newton method without stabilization.

However, such comparison is possible only at the range of Reynolds numbers, where we are able to find the solution by both stabilized and standard Galerkin method.

For this shortcoming, application of *a posteriori error estimates* was presented in [10]. In this approach to evaluating the achieved accuracy of our solution, we use the following error estimator that represents the relative error on element T_K

$$\mathcal{R}^2(u_{1h}, u_{2h}, p_h, T_K) = \frac{|\Omega|\ \mathcal{E}^2(u_{1h}, u_{2h}, p_h, T_K)}{|T_K|\|(u_{1h}, u_{2h}, p_h)\|^2_{V,\Omega}}, \tag{4.24}$$

based on a posteriori error estimates in the following form derived for Taylor-Hood elements in [6]

$$\|(e_{u_1}, e_{u_2}, e_p)\|^2_{V,T_K} \leq \mathcal{E}^2(u_{1h}, u_{2h}, p_h, T_K), \tag{4.25}$$

where

- (u_1, u_2, p) denotes an exact solution,
- (u_{1h}, u_{2h}, p_h) denotes an approximate solution computed by FEM,
- $(e_{u_1}, e_{u_2}, e_p) = (u_1 - u_{1h}, u_2 - u_{2h}, p - p_h)$ denotes an error of approximate solution,
- $\|u_{1h}, u_{2h}, p_h\|^2_{V,\Omega} = \|u_{1h}, u_{2h}\|^2_{1,\Omega} + \|p_h\|^2_{0,\Omega}$ where,
- $\|u_{1h}, u_{2h}\|_{1,\Omega}$ means the Sobolev $H^1(\Omega)$ norm,
- $\|p_h\|_{0,\Omega}$ means the $L_2(\Omega)$ norm,
- $|\Omega|, |T_K|$ mean the area of the domain Ω, and the element T_K, respectively.

The term on the right hand side of the inequality (4.25) is evaluated as

$$\mathcal{E}^2(u_{1h}, u_{2h}, p_h, T_K) = C\Big[h_K^2 \int_{T_K} \big(r_1^2(u_{1h}, u_{2h}, p_h) + r_2^2(u_{1h}, u_{2h}, p_h)\big)\,\mathrm{d}\Omega + \int_{T_K} r_3^2(u_{1h}, u_{2h}, p_h)\mathrm{d}\Omega\Big],$$

where

$$\begin{aligned}
r_1(u_{1h}, u_{2h}, p_h) &= f_{x_1} - \left(u_{1h}\frac{\partial u_{1h}}{\partial x_1} + u_{2h}\frac{\partial u_{1h}}{\partial x_2}\right) + \nu\left(\frac{\partial^2 u_{1h}}{\partial x_1^2} + \frac{\partial^2 u_{1h}}{\partial x_2^2}\right) - \frac{\partial p_h}{\partial x_1}, \\
r_2(u_{1h}, u_{2h}, p_h) &= f_{x_2} - \left(u_{1h}\frac{\partial u_{2h}}{\partial x_1} + u_{2h}\frac{\partial u_{2h}}{\partial x_2}\right) + \nu\left(\frac{\partial^2 u_{2h}}{\partial x_1^2} + \frac{\partial^2 u_{2h}}{\partial x_2^2}\right) - \frac{\partial p_h}{\partial x_2}, \\
r_3(u_{1h}, u_{2h}, p_h) &= \frac{\partial u_{1h}}{\partial x_1} + \frac{\partial u_{2h}}{\partial x_2}.
\end{aligned}$$

stand for residuals of the system (2.6)–(2.7).

The constant C is a delicate task in a posteriori error estimates. In [6], its derivation was shown for the case of non-stabilized finite element method. In this application, the constant is used in a relative sense: we apply the a posteriori error estimates to show the relative error on finite elements, in order to show the distribution of the error in the solution domain. For the purpose of comparison of this distribution obtained without stabilization and by *semiGLS*, it is important to use the same constant for both solutions.

Chapter 5

BDDC domain decomposition method

In this chapter, the Balancing Domain Decomposition by Constraints (BDDC) method is described. It can be understood as a preconditioner for large systems arising from finite element analysis introduced by Dohrmann [13] in 2003. The theory was developed by Mandel and Dohrmann in [45]. The preconditioner was reformulated by Li and Widlund in [42].

Results of joint work with Jan Mandel and Bedřich Sousedík, and results from our paper [54] are used in this chapter.

First, the abstract formulation of the preconditioner is discussed, followed by some modifications motivated by implementation.

5.1 Introduction to iterative substructuring

Throughout this chapter, let Ω be a bounded domain in $\mathbb{R}^2$ or $\mathbb{R}^3$.

Let U be a finite element space of piecewise polynomial functions v continuous on Ω and U' its dual space. Let $a(\cdot,\cdot)$ be a bilinear form on $U \times U$ and $f \in U'$, and let $\langle \cdot, \cdot \rangle$ denote the duality pairing of U' and U. Consider an abstract variational problem: *Find* $u \in U$ *such that*

$$a(u,v) = \langle f, v \rangle \quad \forall\, v \in U\,. \tag{5.1}$$

For the case of linear elasticity,

$$a(u,v) = \int_\Omega (\lambda(\nabla \cdot \mathbf{u}_h)(\nabla \cdot \mathbf{v}_h) + \frac{1}{2}\mu(\nabla \mathbf{u}_h + \nabla^T \mathbf{u}_h) : (\nabla \mathbf{v}_h + \nabla^T \mathbf{v}_h))\mathrm{d}\Omega, \tag{5.2}$$

$$\langle f, v \rangle = \int_\Omega \mathbf{f} \cdot \mathbf{v}_h \mathrm{d}\Omega. \tag{5.3}$$

Here solution $u = \mathbf{u}_h$ represents the discretized vector field of displacement, λ and μ represent the first and the second Lammé's constant, respectively, and $\mathbf{f}$ represents the external load.

For the case of steady Stokes flow (cf. Chapter 3, problem (3.5))

$$a(u,v) = \nu \int_\Omega \nabla \mathbf{u}_h : \nabla \mathbf{v}_h \mathrm{d}\Omega - \int_\Omega p_h \nabla \cdot \mathbf{v}_h \mathrm{d}\Omega - \int_\Omega \psi_h \nabla \cdot \mathbf{u}_h \mathrm{d}\Omega, \tag{5.4}$$

$$\langle f, v \rangle = \int_\Omega \mathbf{f} \cdot \mathbf{v}_h \mathrm{d}\Omega. \tag{5.5}$$

Solution $u = (\mathbf{u}_h, p_h)$ represents the discretized vector field of velocity and the discretized scalar field of pressure, ν represents the kinematic viscosity of the fluid, and $\mathbf{f}$ represents the external load (see Chapter 2 for details).

For the case of linear elasticity, $a(u, v)$ is a symmetric positive definite bilinear form on $U \times U$, while for the Stokes problem, it is symmetric indefinite.

Let us define operator A associated with the bilinear form $a(\cdot, \cdot)$ as

$$A : U \to U', \quad \langle Av, w\rangle = a(v, w) \quad \forall\, v, w \in U. \tag{5.6}$$

An equivalent formulation of (5.1) is to find a solution $u \in U$ to

$$Au = f. \tag{5.7}$$

Remark 1 *The setting involving dual spaces allows us to make a clear distinction between an approximate solution and its residual, which is in the dual space. It is beneficial to have approximate solutions and residuals in different spaces, because they need to be treated differently.*

In computation, operator A is represented by a stiffness matrix denoted A as well. It is defined as $A = (a_{ij})$, where $a_{ij} = a(\phi_i, \phi_j)$ and $\{\phi_i\}$ is a finite element basis of U. Similarly, u and f are represented in the computation by vectors of discrete values corresponding to their coordinates in the basis of U and U', respectively.

The domain Ω is decomposed into N nonoverlapping subdomains Ω_i, $i = 1, ..., N$ with characteristic size H, which form a conforming triangulation of the domain Ω. Each subdomain is a union of several finite elements of the underlying mesh with characteristic mesh size h, i.e. nodes of the finite elements between subdomains coincide.

Definition 1 *Unknowns common to at least two subdomains are called* boundary unknowns *and the union of all boundary unknowns is called the* interface Γ. *Remaining unknowns are called* interior.

Remark 2 *Note the difference of interface Γ from physical boundary of the domain $\partial\Omega$. According to the definition, nodes in $\partial\Omega$ belong to either interface or interior. Most of them, however, become interior nodes.*

Let us now assume, that $a(u, v)$ is symmetric and positive definite to develop the theory based on the scalar product generated by $a(u, v)$. The first step is the reduction of the problem to the interface. The space U is decomposed as the a-orthogonal direct sum $U = U_1 \oplus \cdots \oplus U_N \oplus U_\Gamma$, where U_i is the space of all functions from U with nonzero values only inside Ω_i (in particular, they are zero on Γ), and U_Γ is the a-orthogonal complement of all spaces U_i; $U_\Gamma = \{v \in U : a(v, w) = 0\ \forall w \in U_i,\ i = 1, \dots N\}$. Functions from U_Γ are fully determined by their values at unknowns on Γ and the *discrete harmonic condition* that they have minimal energy on the interior of every subdomain. The *discrete harmonic condition*, sometimes called the *minimal energy condition*, is often used in domain decomposition literature. It is an analogy to discrete solutions to Laplace equation with homogenous right hand side. In our context, we use the following definition:

Definition 2 *Let b be a vector with zero values at selected unknowns. The solution u to the discrete system $Au = b$ is called discrete harmonic with respect to these unknowns.*

Let us denote the sum of interior solutions as $u_o = \sum_i^N u_i$, $u_i \in U_i$, and let $u_\Gamma \in U_\Gamma$. Then the solution may be rewritten as $u = u_\Gamma + u_o$, and problem (5.7) as

$$A(u_\Gamma + u_o) = f. \tag{5.8}$$

Let us now write the problem in the block form, with the first block corresponding to unknowns in subdomain interiors, and the second block corresponding to unknowns at the interface,

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} u_{\Gamma 1} + u_{o1} \\ u_{\Gamma 2} + u_{o2} \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix}. \tag{5.9}$$

Here u_{o2} is a zero block by definition. Using the minimal energy constraint of functions from U_Γ, problem (5.9) may be split into the sum of the following two problems

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} u_{\Gamma 1} \\ u_{\Gamma 2} \end{bmatrix} = \begin{bmatrix} 0 \\ f_2 - A_{21}A_{11}^{-1}f_1 \end{bmatrix}, \tag{5.10}$$

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} u_{o1} \\ 0 \end{bmatrix} = \begin{bmatrix} f_1 \\ A_{21}A_{11}^{-1}f_1 \end{bmatrix}. \tag{5.11}$$

It follows, that this decomposition is equivalent to solving first the problem

$$A_{11}u_{o1} = f_1, \tag{5.12}$$

which is independent for each subdomain, followed by substitution $u_{o1} = A_{11}^{-1}f_1$ into (5.10) and solving

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} u_{\Gamma 1} \\ u_{\Gamma 2} \end{bmatrix} = \begin{bmatrix} 0 \\ f_2 - A_{21}u_{o1} \end{bmatrix}. \tag{5.13}$$

Once both problems are solved, the solution is obtained as $u = u_\Gamma + u_o$.

Problem (5.13) is equivalent to the solution of

$$Su_{\Gamma 2} = g_2, \tag{5.14}$$

where S is the *Schur complement* with respect to interface, $S = A_{22} - A_{21}A_{11}^{-1}A_{12}$, and g_2 is the *condensed right hand side*, $g_2 = f_2 - A_{21}u_{o1}$, followed by the solution of

$$A_{11}u_{\Gamma 1} = -A_{12}u_{\Gamma 2}. \tag{5.15}$$

Since A_{11} has a block diagonal structure, solution to (5.12) may be found in parallel as $u_{o1} = \sum_i^N u_{o1i}$, where u_{o1i} are solutions of the local problems $A_{11i}u_{o1i} = f_{1i}$ on every Ω_i.

In the following sections, we are interested in an efficient solution of problem (5.13), or equivalently (5.14), by the preconditioned conjugate gradients (PCG) method.

To find solution u_Γ by PCG, we iterate on problem (5.14), but without explicit construction of S. It is circumvented by finding $u_{\Gamma 1}$ in each iteration by solving Dirichlet problem in (5.13), since A_{11} block is already factorized from (5.12).

5.2 Formulation of BDDC

The BDDC method is a particular kind of preconditioner for problem (5.13), or the reduced problem (5.14).

Let W_i be the space of finite element functions on subdomain Ω_i and put

$$W = W_1 \times \cdots \times W_N. \tag{5.16}$$

It is the space, where subdomains are completely disconnected, and functions on them independent of each other. Clearly, $U_\Gamma \subset W$.

The main idea of the BDDC preconditioner in an abstract form [48] is to construct an auxiliary finite dimensional space $\widetilde{W}$ such that

$$U_\Gamma \subset \widetilde{W} \subset W, \tag{5.17}$$

and extend the bilinear form $a(\cdot,\cdot)$ to a form $\widetilde{a}(\cdot,\cdot)$ defined on $\widetilde{W} \times \widetilde{W}$, such that solving the variational problem (5.1) with $\widetilde{a}(\cdot,\cdot)$ in place of $a(\cdot,\cdot)$ is cheaper and can be split into independent computations performed in parallel. Then the solution restricted to U_Γ is used for the preconditioning of (5.13), or (5.14).

More precisely, let

$$E : \widetilde{W} \to U_\Gamma \tag{5.18}$$

be a given projection of $\widetilde{W}$ onto U_Γ, g the right hand side of (5.13), and $r = g - Au_\Gamma \in U_\Gamma'$ the residual in a PCG iteration. Here U_Γ' is the dual space to U_Γ.

Remark 3 *Residual r remains zero at unknowns interior to subdomains for $u_\Gamma \in U_\Gamma$ in each iteration for the discrete harmonic property w.r.t. interior of functions from U_Γ.*

An action of the BDDC preconditioner is described in the following algorithm.

Algorithm 1 *The BDDC preconditioner $M_{BDDC} : U_\Gamma' \to U_\Gamma$ in* the abstract form *is defined as*

$$M_{BDDC} : r \to v = Ew,$$

where $w \in \widetilde{W}$ is obtained as the solution to problem

$$w \in \widetilde{W} : \widetilde{a}(w,z) = (r, Ez) \quad \forall z \in \widetilde{W}. \tag{5.19}$$

In terms of operators,

$$v = E\widetilde{A}^{-1}E^T r, \tag{5.20}$$

where $\widetilde{A}$ is the operator associated with the bilinear form $\widetilde{a}$ by

$$\left\langle \widetilde{A}u, v \right\rangle = \widetilde{a}(u,v) \quad \forall u, v \in \widetilde{W}. \tag{5.21}$$

In computation, $\widetilde{A}$ is represented by a matrix denoted $\widetilde{A}$ as well. It is larger than the original matrix of the problem A, but possess simpler structure in terms of direct solution methods.

All functions from U_Γ are continuous on the domain Ω. In order to design the space $\widetilde{W}$, we relax the continuity on the interface Γ. On Γ, we select *coarse degrees of freedom* and define $\widetilde{W}$ as the space of finite element functions with minimal energy on every subdomain, continuous across Γ only at coarse degrees of freedom. The coarse degrees of freedom can be of two basic types – explicit unknowns (called *coarse unknowns*) at selected nodes (called *corners*), and averages over larger disjoint sets of nodes (subdomain *faces* or *edges*, cf. Definition 3). The continuity condition then means that the values of the corresponding unknowns, or averages, on neighbouring subdomains coincide. The bilinear form $a(\cdot,\cdot)$ is extended to $\widetilde{a}(\cdot,\cdot)$ on $\widetilde{W} \times \widetilde{W}$ by integrating over the subdomains Ω_i separately and adding the results. We need to use the following assumption on selection of corners.

Assumption 1 *It is assumed, that enough corners were chosen for $\widetilde{a}(\cdot,\cdot)$ to fix floating subdomains.*

In the case of elasticity, this means that number of corners is sufficient to prevent relative rigid body motions of any pair of adjacent subdomains. This results in the positive definiteness of $\widetilde{a}(\cdot,\cdot)$. In the case of Stokes problem, satisfying the assumption results in a positive definite block corresponding to velocity unknowns. See Remark 4 for discussion on satisfying Assumption 1.

The projection $E : \widetilde{W} \to U_\Gamma$ is realized as a weighted average of values from different subdomains at unknowns on the interface Γ, thus resulting in functions continuous across the interface, and the solutions of local subdomain problems (5.15) to make the averaged function discrete harmonic. To assure good performance regardless of different stiffnesses of the subdomains [45], the weights may be chosen proportional to the corresponding diagonal entries of the subdomain stiffness matrices.

Definition 3 *The interface Γ may also be classified as a union of three different types of sets:* faces*,* edges*, and* vertices*, according e.g. to [36]. In the case that we have no information about the boundary of the domain, the definition in 3D simplifies to the following:*

- *a* face *contains all nodes shared by two fixed subdomains,*
- *an* edge *contains nodes shared by more than two fixed subdomains,*
- *a* vertex *is a degenerated edge with only one node.*

Let us identify vertices with so called corners (see Remark 4 for more general case). As in [44], we will call any edge and, in the 3D case, a face on the interface Γ a *glob* and it will be identified with the set of degrees of freedom associated with nodes in it. The set of all globs will be denoted by $\mathcal{K}_\mathcal{G}$. Note, that our definition of a glob does not include corners, the set of which will be denoted by $\mathcal{K}_\mathcal{C}$. So, for the interface it holds that

$$\Gamma = \left(\cup_{\mathcal{G}\in\mathcal{K}_\mathcal{G}}\mathcal{G}\right) \cup \mathcal{K}_\mathcal{C}.$$

In the original formulation of the preconditioner [13], space $\widetilde{W}$ is further decomposed as $\widetilde{a}$-orthogonal direct sum $\widetilde{W} = \widetilde{W}_1 \oplus \cdots \oplus \widetilde{W}_N \oplus \widetilde{W}_C$, where $\widetilde{W}_i$ is the space of functions with nonzero values only in Ω_i (i.e. they have zero values at coarse unknowns and they are generally not continuous at other unknowns on Γ) and $\widetilde{W}_C$ is the explicit *coarse space*, defined as the $\widetilde{a}$-orthogonal complement of all spaces $\widetilde{W}_i$; $\widetilde{W}_C = \{v \in \widetilde{W} : \widetilde{a}(v,w) = 0\ \forall w \in \widetilde{W}_i,\ i = 1,\dots N\}$. Functions from $\widetilde{W}_C$ are fully determined by their values at coarse degrees of freedom (where they are continuous) and have minimal energy. Thus, they are generally discontinuous across Γ outside of coarse unknowns. The solution $w \in \widetilde{W}$ from (5.19) is now split accordingly, and the BDDC preconditioner is defined as

Algorithm 2 *The BDDC preconditioner $M_{BDDC} : U'_\Gamma \to U_\Gamma$* with coarse problem *is defined as*

$$M_{BDDC} : r \to u = Ew,$$

where $w \in \widetilde{W}$ is obtained as

$$w = w_C + \sum_{i=1}^{N} w_i.$$

Here the coarse correction *w_C is determined by*

$$w_C \in \widetilde{W}_C : \widetilde{a}(w_C, v) = (r, Ev) \quad \forall v \in \widetilde{W}_C, \tag{5.22}$$

and the subdomain corrections *w_i are determined by*

$$w_i \in \widetilde{W}_i : \widetilde{a}(w_i, v) = (r, Ev) \quad \forall v \in \widetilde{W}_i. \tag{5.23}$$

Remark 4 *Although the set of vertices forms an important subset of corners, the definition of vertices is often not sufficient to detect enough corners for uniqueness of subdomain solutions. To see its shortcomings, we can think of three very simple examples of division into two, four and eight subdomains depicted in Figure 5.1. In the case of two subdomains, whole interface is recognized as a single face. In the second case, the interface is classified into four faces and one edge. In the case of eight subdomains, this classification leads to twelve faces, six edges and a single vertex in the center of the domain. To assure that enough corners will be used for each subdomain, one*

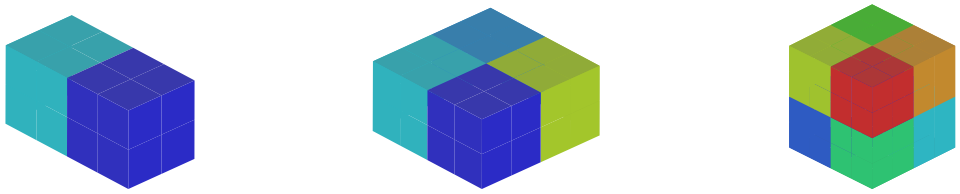


Figure 5.1: Examples of problems with two, four, and eight cubic subdomains

can immediately think of a number of modifications that are possible to generate a better list of corners, edges and faces.

Although several more sophisticated methods were tested for this purpose ([35, 40]), a simple algorithm is applied in most of the practical calculations in this thesis. It is based on addition of prescribed number of random nodes from interface to the set of corners, followed by removal of these nodes from edges and faces. Besides its simplicity, it has been found useful for its flexibility in setting the size of the coarse problem, and thus controlling the efficiency of the preconditioning for a fixed division. A more sophisticated approach was presented in [47]. It is based on adding proper constraints to edges and/or faces.

Let us now rewrite Algorithm 2 in terms of matrices, following [13]. Problem (5.23) is formulated in a saddle point form as

$$\begin{bmatrix} K_i & \overline{C}_i^T \\ \overline{C}_i & 0 \end{bmatrix} \begin{bmatrix} w_i \\ \mu_i \end{bmatrix} = \begin{bmatrix} r_i \\ 0 \end{bmatrix}, \tag{5.24}$$

where K_i denotes the substructure local stiffness matrix, obtained by the subassembly of element matrices only of elements in substructure i, matrix $\overline{C}_i$ represents constraints on subdomain, that enforce zero values of coarse degrees of freedom, μ_i is vector of Lagrange multipliers, and r_i is the weighted residual $E^T r$ restricted to subdomain i.

Matrix K_i is singular for floating subdomains (subdomains not touching Dirichlet boundary conditions), while the augmented matrix of problem (5.24) is regular and may be factorized. Matrix $\overline{C}_i$ contains both constraints enforcing continuity across corners (single point continuity), and constraints enforcing equality of averages over edges and faces of subdomains. The former type corresponds to just one nonzero entry equal to 1 on a row of $\overline{C}_i$, while the latter leads to several nonzero entries on a row. This structure will be exploited in the following section.

Problem (5.24) is solved in each iteration of the PCG method to find the correction from substructure i. However, the matrix of (5.24) is used prior to the whole iterative process to construct the *local subdomain matrix of the coarse problem*. First, the *coarse basis functions* are found independently for each subdomain as the solution to

$$\begin{bmatrix} K_i & \overline{C}_i^T \\ \overline{C}_i & 0 \end{bmatrix} \begin{bmatrix} \psi_i \\ \lambda_i \end{bmatrix} = \begin{bmatrix} 0 \\ I \end{bmatrix}. \tag{5.25}$$

This is a problem with multiple right hand sides, where ψ_i is a matrix of coarse basis functions with several columns, each corresponding to one coarse degree of freedom on i-th subdomain. These functions are given by values equal to 0 at all coarse degrees of freedom except one, where they equal to 1, and they have minimal energy on subdomain outside coarse degrees of freedom. The identity block I has the dimension of the number of constraints on the subdomain.

Once ψ_i is known, the *subdomain coarse matrix* K_{Ci} is constructed as

$$K_{Ci} = \psi_i^T K_i \psi_i. \tag{5.26}$$

Matrices K_{Ci} are then assembled to form the global *coarse matrix* A_C. This procedure is same as the standard process of assembly in finite element solution, with subdomains playing the role of elements, coarse degrees of freedom on subdomain representing degrees of freedom on element, and matrix K_{Ci} representing the element stiffness matrix.

Problem (5.22) is now

$$A_C \mathbf{w}_C = r_C, \tag{5.27}$$

where r_C is the *global coarse residual* obtained by the assembly of the subdomain contributions of the form $r_{Ci} = \psi_i^T r_i$.

The dimension of the coarse solution $\mathbf{w}_C$ is equal to the number of all coarse degrees of freedom. So, to add the correction to subdomain problems, we first have to restrict it to coarse degrees of freedom on each subdomain and to interpolate it to the whole subdomain by $w_{Ci} = \psi_i \mathbf{w}_{Ci}$. By extending w_{Ci} and w_i by zero to other subdomains, these can be summed over the subdomains to form the final vector w. Finally, the preconditioned residual is obtained as $v = Ew$.

It is worth noticing that in the case of no constraints on averages, i.e. using only coarse unknowns for the definition of the coarse space, matrix A_C of problem (5.27) is simply the Schur complement of matrix A with respect to coarse unknowns. This fact was pointed out in [42]. If additional degrees of freedom are added for averages, they correspond to new explicit unknowns in $\mathbf{w}_{Ci}$.

Obviously, several mapping operators among various spaces are needed in the algorithm, defining embedding of subdomains into global problem, local subdomain coarse problem into global coarse problem etc. We have circumvented their mathematical definition by words for the sake of brevity.

In [42], another point of view was presented. It emphasised, that exctracting an explicit coarse space might be redundant, if another suitable solver for (5.20) is available. The following formulation was derived with this fact kept on mind.

First, let us distinguish between two different kinds of coarse degrees of freedom defining space $\widetilde{W}$. The first kind of coarse degrees of freedom is represented by unknowns at corners. Let us denote the space of functions continuous across corners as $\widetilde{W}^c$. The operators on $\widetilde{W}^c$ are obtained by a process called subassembly described in detail by Li and Widlund [42].

The second group represents coarse degrees of freedom associated with satisfying additional constraints on functions: such as equality of their average values across edges or faces. The purpose of this choice is to obtain the desirable polylogarithmic condition number bound

$$\kappa \leq C \left(1 + \log \frac{H}{h}\right)^2. \tag{5.28}$$

We refer to, e.g., monograph [58] for the detailed discussion.

Let us denote the space of functions from $\widetilde{W}^c$ satisfying these additional constraints as $\widetilde{W}^{avg}$. Then, we can rewrite the hierarchy of spaces (5.17), as

$$U_\Gamma \subset \widetilde{W}^{avg} \subset \widetilde{W}^c \subset W. \tag{5.29}$$

This distinction allows us to think of both kinds of coarse degrees of freedom separately, and to handle them differently in an implementation.

Subspace $\widetilde{W}^{avg}$ may be defined as

$$\widetilde{W}^{avg} = \left\{ w \in \widetilde{W}^{c} : Gw = 0 \right\}, \tag{5.30}$$

where G is a constraint matrix. Each row of G represents one constraint and contains coefficients of averages. The constraints are linearly independent, so that G has full rank. As an example, a particular row k of G that corresponds to a constraint that enforces equality of an arithmetic average over an edge between two subdomains may appear as

$$g_k = [0 \dots 0 \quad \underbrace{1\,1\,1\,1}_{\text{edge dof on } \Omega_i} \quad 0 \dots 0 \underbrace{-1\,-1\,-1\,-1}_{\text{edge dof on } \Omega_j} 0 \dots 0] \tag{5.31}$$

Note that, e.g., for elasticity in 3D it is natural to split G into three independent rows, each corresponding to displacements in the direction of a principal axis.

Also we need to be careful in coupling edge averages in 3D that belong to more than just two subdomains and use nonredundant constraint in the same sense as, e.g., [58, Section 6.3.1] so that G has full rank. We note, that using G naturally leads to nonredundant constraints, since for a constraint among m subdomains, we can choose a 'master' subdomain, and generate $m-1$ connections to the remaining subdomains in turn, resulting in $m-1$ constraints in G.

Using the space $\widetilde{W}^{avg}$, the algorithm of the BDDC preconditioner (Algorithm 2) can be reformulated as

Algorithm 3 *The BDDC preconditioner* $M_{BDDC} : U'_\Gamma \to U_\Gamma$ in space $\widetilde{W}^{avg}$ *is defined as*

$$M_{BDDC} : r \to u = Ew, \quad w \in \widetilde{W}^{avg} : \quad \widetilde{a}\,(w, z) = \langle r, Ez \rangle\,, \quad \forall z \in \widetilde{W}^{avg}.$$

Also the projection E is refined as

$$E : \widetilde{W}^{avg} \to U_\Gamma. \tag{5.32}$$

The ways to actually realize Algorithm 3 are discussed in Sections 5.3 and 5.4.

5.3 Projected BDDC preconditioner

Let us now describe in detail, how to actually realize the abstract BDDC preconditioner described in Algorithm 3, i.e. restricted to the subspace $\widetilde{W}^{avg}$. The preconditioner consists of two steps: solving the system

$$\widetilde{A}w = E^T r, \quad \text{subject to } \; Gw = 0, \tag{5.33}$$

followed by the computation of the approximate solution $u \in U_\Gamma$ as $u = Ew$.

A natural way to rewrite this statement is based on Lagrange multipliers

$$\begin{array}{lcll} \widetilde{A}w & + & G^T\mu & = & E^T r, \\ Gw & & & = & 0. \end{array} \tag{5.34}$$

This system might serve for the actual solution. However, it would not lead to an efficient parallel implementation and we only refer to it in the following derivations.

Instead, we propose another way of restricting the action of the preconditioner to space $\widetilde{W}^{avg}$. The idea is to project the system (5.34) onto the right subspace, i.e. nullspace of G, by orthogonal projection operator P defined as

$$P = I - G^T \left(GG^T\right)^{-1} G. \tag{5.35}$$

The projected system has then the form

$$P\widetilde{A}Pw = PE^T r. \tag{5.36}$$

Because $P\widetilde{A}P$ is singular for nontrivial G (null(G) is a proper subspace of $\widetilde{W}^c$), we suggest to solve instead of (5.36) a modified system

$$\left[P\widetilde{A}P + s(I-P)\right] w = PE^T r, \tag{5.37}$$

where $s > 0$ is some scaling constant. Now, the operator $P\widetilde{A}P + s(I-P)$ is regular and the following two lemmas hold.

Lemma 4 *Problems (5.36) and (5.37) have the same solution.*

Proof. Operator P is the projection onto null(G). Since null$(G) \perp$ range(G^T), the complementary operator $I - P$ is a projection onto range(G^T). Therefore

$$\underbrace{P(\widetilde{A}Pw)}_{\text{null}(G)} + \underbrace{s(I-P)w}_{\text{range}(G^T)} = \underbrace{P(E^T r)}_{\text{null}(G)}.$$

Should w be the solution of (5.36), we get a condition

$$s(I-P)w = 0, \tag{5.38}$$

which concludes the proof. As a consequence, from condition (5.38), we see that the solution w satisfies $Pw = w$, i.e., $w \in$ null(G). ■

Lemma 5 *Problems (5.34) and (5.37) have the same solution.*

Proof. From Lemma 4, it is sufficient to show the equivalence of (5.34) and (5.36). First, suppose w is the solution of (5.36). From the last observation in the proof of Lemma 4, we get for solution w of (5.36)

$$Gw = 0,$$

the second equation in (5.34). Since $Pw = w$, system reduces to

$$P(\widetilde{A}w) = P(E^T r).$$

Putting both terms on one side we get

$$P(\widetilde{A}w - E^T r) = 0,$$

which implies that $\widetilde{A}w - E^T r \perp$ null(G), i.e. $\widetilde{A}w - E^T r \in$ range(G^T). If G^T has a full column rank, we get the existence of vector μ that satisfies $E^T r - \widetilde{A}w = G^T \mu$. This can be rewritten as

$$\widetilde{A}w + G^T \mu = E^T r,$$

which is the first equation in (5.34).

Suppose now that w is the solution of (5.34). Since $Gw = 0$, $Pw = w$ and

$$\widetilde{A}w = \widetilde{A}Pw.$$

We can rewrite the first equation of (5.34) as

$$\widetilde{A}Pw + G^T\mu - E^T r = 0.$$

Since this equality holds on the whole space $\widetilde{W^c}$, it holds when projected on its subspace null(G) as well,

$$P(\widetilde{A}Pw + G^T\mu - E^T r) = 0.$$

What remains to show is that $PG^T\mu = 0$. This may be seen from the definition of P in (5.35) as

$$(I - G^T\left(GG^T\right)^{-1}G)G^T\mu = G^T\mu - G^T\underbrace{\left(GG^T\right)^{-1}GG^T}_{I}\mu = G^T\mu - G^T\mu = 0.$$

■

Due to the block structure of G, where each block corresponds to a different glob, and because by definition each degree of freedom belongs to at most one glob, the action of P in (5.36) and (5.37) on the matrix $\widetilde{A}$ can be performed efficiently glob-wise in parallel.

This fact considerably simplifies the explicit construction of the resulting matrix on the left hand side of (5.37), which is necessary in the algorithm. However, the structure of P gives rise to dense off-diagonal blocks in a projected system operator $P\widetilde{A}P$, thus spoiling much of the simpe structure of the corresponding matrix, which is essential for using BDDC as a preconditioner. This unpleasant fact corresponds to the 're-connection' of degrees of freedom that belong to the same glob among subdomains. The next section about *generalized change of variables* presents a satisfactory answer to this problem.

5.4 Generalized change of variables

In this section, a generalization of the change of variables described by Li and Widlund [42] is presented. It allows to prescribe quite general edge or face averages as constraints (in [42], only arithmetic averages are admitted), and also preserve a minimal fill-in of the projected system operator.

The basic idea is to transform the operator $\widetilde{A}$ associated with the bilinear form $\widetilde{a}$ on the space $\widetilde{W^c}$ into a different basis, in which all averages would be represented by explicit degrees of freedom.

The dual coupling by matrix of constraints $\overline{G}$ is then done between these individual degrees of freedom, resulting in a minimum fill-in of the projected system operator (see Section 5.3). We note that, obviously, we could treat the degrees of freedom corresponding to averages after the change of variables as corners, i.e. assemble them as advocated in [42], which would give us no additional fill-in beyond the one caused by the change of variables. However, we do not adopt such approach here, mainly because of reasons related to implementation: different dimensioning of arrays, and loosing the distinction between $\widetilde{W^c}$ and $\widetilde{W^{avg}}$, which is found useful e.g. in the concept of adaptive selection of additional constraints described in [47], where dimensions are, in fact unknown a priori.

The new variables are introduced by transformation

$$\overline{w} = Bw, \tag{5.39}$$

where operator B is represented by an invertible matrix containing weights of the averages intended to constrain. Let us denote the inverse transformation T, so then $T = B^{-1}$ and $w = T\overline{w}$.

Now, using the change of basis we can further modify Algorithm 3 as follows:

Algorithm 4 *The BDDC preconditioner* $\overline{M}_{BDDC} : U'_\Gamma \to U_\Gamma$ with generalized change of variables *is defined as*

$$\overline{M}_{BDDC} : r \to u = ET\overline{w}, \quad w = T\overline{w} \in \widetilde{W}^{avg} : \ \widetilde{a}\left(T\overline{w}, T\overline{z}\right) = \langle r, ET\overline{z}\rangle \ , \ \overline{z} = Bz, \forall z \in \widetilde{W}^{avg}.$$

Note, that in the action of the preconditioner defined by the previous algorithm, transformation B is never used and only its inverse T is necessary.

Algorithm 4 consists of two steps: solving the system

$$T^T \widetilde{A} T\overline{w} = T^T E^T r, \quad \text{subject to } \overline{G}\overline{w} = 0, \tag{5.40}$$

followed by computation of the approximate solution $u \in U_\Gamma$ as

$$u = ET\overline{w}. \tag{5.41}$$

Here, the new operator $\overline{G}$ is defined as

$$\overline{G} = GT, \tag{5.42}$$

and is obtained by transforming the operator G used in (5.33). By the transformation, constraints in $\overline{G}$ simplify to one 1 and one -1 entry on each row. Thus, $\overline{G}$ is much sparser than G. The meaning of $\overline{G}$ is similar to the matrices enforcing continuity across interface in FETI methods (cf. [19]).

The transformation matrix B is constructed separately for each glob and contains weigths of n averages on its first n rows, completed by any block, that leads to an invertible matrix. The choice of this block is arbitrary and driven mainly by the low induced fill-in in transformed matrix. A block of identity is a reasonable choice. See Section 6.3.2 for details of implementation.

For the application of the preconditioner, we need to explicitly construct the matrix of the operator on the left hand side of (5.40). However, the inverse of B and its application to the matrix in (5.40) are performed glob-wise, and thus are quite efficient. This procedure is also described in detail in Section 6.3.2.

For the case of a single arithmetic average on a glob, the block of B corresponding to the glob contains only one row of ones and other ones on diagonal, i.e. for a particular glob, it may look as

$$B_G = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{5.43}$$

with the inverse

$$T_G = B_G^{-1} = \begin{bmatrix} 1 & -1 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{5.44}$$

Remark 5 *The matrix T_G in (5.44) is a particular case of the matrix T_E considered in [42, Section 3.3] for the case of arithmetic averages over globs.*

For the actual realization of the coupled problem (5.40), the projected BDDC of Section 5.3 may be used.

To project the system onto null($\overline{G}$), the orthogonal projection $\overline{P}$ is introduced as

$$\overline{P} = I - \overline{G}^T \left(\overline{G}\, \overline{G}^T \right)^{-1} \overline{G}.$$

The projected system (5.40) has the form

$$\overline{P} T^T \widetilde{A} T \overline{P} \overline{w} = \overline{P} T^T E^T r. \tag{5.45}$$

Again, the operator $\overline{P} T^T \widetilde{A} T \overline{P}$ is singular for nontrivial $\overline{G}$, and it is reasonable to solve instead of (5.45) a modified system

$$\left[\overline{P} T^T \widetilde{A} T \overline{P} + \overline{s}(I - \overline{P}) \right] \overline{w} = \overline{P} T^T E^T r, \tag{5.46}$$

where $\overline{s} > 0$ is some scaling constant. Now, the operator $\overline{P} T^T \widetilde{A} T \overline{P} + \overline{s}(I - \overline{P})$ is regular and also, the solutions of the systems (5.45) and (5.46) are the same (cf. Section 5.3, Lemmas 4 and 5 for the argument).

Finally, let us rewrite the Algorithm 4 in an algebraic form as:

Algorithm 5 *The action of the BDDC preconditioner $\overline{M}_{BDDC} : U'_\Gamma \rightarrow U_\Gamma$ projected onto the space $\widetilde{W}^{avg}$ with the generalized change of variables consists of the two steps: solving the system*

$$\left[\overline{P} T^T \widetilde{A} T \overline{P} + \overline{s}(I - \overline{P}) \right] \overline{w} = \overline{P} T^T E^T r, \tag{5.47}$$

followed by the computation of the approximate solution $u \in U_\Gamma$ as $u = ET\overline{w}$.

Chapter 6

Parallel algorithms of the BDDC method

This chapter is devoted to efficient algorithms of the BDDC method introduced in Chapter 5. The first presented approach uses the frontal algorithm by Irons [33]. It is presented in our paper [54]. The second approach uses a multifrontal method by Duff and Reid [15]. Some tricks useful for the implementation of these algorithms on parallel computers are also included.

6.1 BDDC by frontal solver

In this section, a detailed description of Algorithm 2, i.e. the BDDC preconditioner with explicit construction of the coarse problem, is presented. This approach is based on the fact, that while coarse degrees of freedom introduce new coupling among, otherwise completely separated, subdomains in space W, this coupling is extracted into the new global coarse problem, and thus the independence of subdomain corrections is recovered, and they may be computed in parallel.

We now show how to construct and solve subdomain problem (5.24) and coarse problem (5.27) using the frontal solver. Both continuity at corners and equality of averages over globs are considered, however, they are applied in different ways.

The algorithm has been implemented in Fortran 77 programming language and MPI library is used for parallelization. It is based on standard building blocks of finite element software, and thus require minimal amount of custom coding.

The frontal solver [33] is a direct factorization method, that resolves a square linear system with some of the variables having prescribed fixed values. Equations that correspond to the fixed variables are omitted and the values of these variables are substituted into the solution vector directly. The output of the solver consists of the solution and the resulting imbalance in the equations, called reaction forces. More precisely, consider a block decomposition of the vector of unknowns x with the second block consisting of all fixed variables, and write a system matrix A with the same block decomposition. Then on exit from the frontal solver,

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix} + \begin{bmatrix} 0 \\ r_2 \end{bmatrix}, \tag{6.1}$$

where fixed variable values x_2 and the load vectors f_1 and f_2 are the inputs, while the solution x_1 and the reaction r_2 are the outputs.

Notation 3 *In this section, we drop the subdomain subscript* $_i$.

Let us write subdomain vectors w in the block form with the second block consisting of coarse unknowns, denoted by the subscript $_c$, and the first block consisting of the remaining degrees of freedom, denoted by the subscript $_f$. The vector of the coarse degrees of freedom given by

averages is written as Cw, where each row of C contains the coefficients of the average that makes that degrees of freedom; zeros and ones for arithmetic averages. Then subdomain vectors $w \in \widetilde{W}$ are characterized by $w_c = 0$, $Cw = 0$. Assume that $C = [C_f \; C_c]$, with $C_c = 0$, that is, the averages do not involve single variable coarse unknowns; then $Cw = C_f w_f$. Denote the subdomain local stiffness matrix by K. This matrix is obtained by the subassembly of element matrices only of elements in the subdomain (the global stiffness matrix A may be obtained from matrices K by assembly over subdomains, but it is not needed). The matrix K is singular for floating subdomains (subdomains not touching Dirichlet boundary conditions), but the block K_{ff} is nonsingular if Assumption 1 is satisfied, i.e. enough corners are selected to eliminate rigid body motions.

The local subdomain problems (5.23) are written in the frontal solver form (6.1) as

$$\begin{bmatrix} K_{ff} & K_{fc} & C_f^T \\ K_{cf} & K_{cc} & 0 \\ C_f & 0 & 0 \end{bmatrix} \begin{bmatrix} w_f \\ w_c \\ \mu \end{bmatrix} = \begin{bmatrix} r \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ Rea \\ 0 \end{bmatrix}, \tag{6.2}$$

where $w_c = 0$, r is the part of the residual in the PCG method in the $_f$ block distributed to the subdomain by the operator E^T, and Rea is the reaction. The constraint $w_c = 0$ is enforced by marking the w_c unknowns as fixed, while the remaining constraints $C_f w_f = 0$ are enforced via the Lagrange multiplier μ. Using the fact that $w_c = 0$, we get from (6.2) that

$$K_{ff} w_f = -C_f^T \mu + r, \tag{6.3}$$

$$K_{cf} w_f = Rea, \tag{6.4}$$

$$C_f w_f = 0. \tag{6.5}$$

From (6.3), $w_f = K_{ff}^{-1} \left(-C_f^T \mu + r \right)$. Now substituting w_f into (6.5), we get the dual problem for μ,

$$C_f K_{ff}^{-1} C_f^T \mu = C_f K_{ff}^{-1} r. \tag{6.6}$$

The matrix $C_f K_{ff}^{-1} C_f^T$ is dense but small, with the order equal to the number of averages on the subdomain, and it is constructed by solving the system $K_{ff} U = C_f^T$ with multiple right hand sides by the frontal solver, and then by multiplication $C_f U$. After solving problem (6.6), we substitute for μ in (6.3) and find w_f from (6.3)–(6.4) by the frontal solver, considering both $w_c = 0$ and μ fixed. The factorization in the frontal solver for (6.2) and the factorization of the dual matrix $C_f K_{ff}^{-1} C_f^T$ need to be computed only once.

Note that while the residual in the PCG method applied to the reduced problem is given at the interface only, the right hand side in (6.2) has the dimension of all degrees of freedom on the subdomain. This is corrected naturally by extending the residual to subdomain interiors by zeros, which is required by the condition that the solution w of (6.2) is discrete harmonic inside subdomain. Similarly, only interface values of w_i are used after solution of (6.2) in further PCG computation. Such approach is equivalent to computing with explicit Schur complements. Aware of this, we make no distinction in notation between these vectors given on the whole subdomain and on the corresponding interface.

The coarse problem (5.22) is solved by the frontal solver just like a finite element problem, with the subdomains playing the role of elements. It only remains to specify the basis functions of $\widetilde{W}_C$ on the subdomain and compute the local *subdomain coarse matrix* efficiently. Each coarse basis function is a column vector of values of unknowns on the subdomain and it is associated with one coarse degree of freedom, which has value 1, while all other coarse degrees of freedom have value 0. Denote by ψ^c the matrix whose colums are coarse basis functions associated with the coarse unknowns at corners, and ψ^{avg} the matrix made out of the coarse basis functions associated

with averages. To find the coarse basis functions, we proceed similarly as in (6.2) and write the equations for the coarse basis functions in the frontal solver form, now with multiple right-hand sides,

$$\begin{bmatrix} K_{ff} & K_{fc} & C_f^T \\ K_{cf} & K_{cc} & 0 \\ C_f & 0 & 0 \end{bmatrix} \begin{bmatrix} \psi_f^c & \psi_f^{avg} \\ I & 0 \\ \lambda^c & \lambda^{avg} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & I \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ Rea^c & Rea^{avg} \\ 0 & 0 \end{bmatrix}, \tag{6.7}$$

where Rea^c and Rea^{avg} are matrices of reactions. Let us denote $\psi_f = \begin{bmatrix} \psi_f^c & \psi_f^{avg} \end{bmatrix}$, $\psi_c = \begin{bmatrix} \psi_c^c & \psi_c^{avg} \end{bmatrix} = \begin{bmatrix} I & 0 \end{bmatrix}$, $\lambda = \begin{bmatrix} \lambda^c & \lambda^{avg} \end{bmatrix}$, $Rea = \begin{bmatrix} Rea^c & Rea^{avg} \end{bmatrix}$, and $R = \begin{bmatrix} 0 & I \end{bmatrix}$ with blocks of the same size. Then (6.7) becomes

$$K_{ff}\psi_f + K_{fc}\psi_c = -C_f^T\lambda, \tag{6.8}$$

$$K_{cf}\psi_f + K_{cc}\psi_c = Rea, \tag{6.9}$$

$$C_f\psi_f = R. \tag{6.10}$$

From (6.8), we get $\psi_f = -K_{ff}^{-1}\left(K_{fc}\psi_c + C_f^T\lambda\right)$. Substituting ψ_f into (6.10), we derive the dual problem for Lagrange multipliers

$$C_f K_{ff}^{-1} C_f^T \lambda = -\left(R + C_f K_{ff}^{-1} K_{fc}\psi_c\right), \tag{6.11}$$

which is solved for λ by solving the system (6.11) for multiple right hand sides. Since ψ_c is known, we can use frontal solver to solve (6.8)–(6.9) to find ψ_f.

Finally, we construct the local coarse matrix corresponding to the subdomain as

$$K_C = \psi^T K \psi = \psi^T \begin{bmatrix} -C_f^T\lambda \\ Rea \end{bmatrix} = \begin{bmatrix} \psi_f^T & \psi_c^T \end{bmatrix} \begin{bmatrix} -C_f^T\lambda \\ Rea \end{bmatrix} = -\psi_f^T C_f^T \lambda + \begin{bmatrix} I \\ 0 \end{bmatrix} Rea,$$

where $\psi = \begin{bmatrix} \psi^c & \psi^{avg} \end{bmatrix}$.

At the end of the setup phase, the matrix of coarse problem is factored by frontal solver, using subdomain coarse matrices as input. Note that the factorizations in the subdomain solution and in the computation of the coarse basis functions are the same, and need to be computed only once.

6.2 BDDC by multifrontal solver

As was already mentioned in Section 5.2, the point of view presented in [42] allows the reformulation of the BDDC preconditioner from Algorithm 2 to Algorithm 3.

An efficient parallel solver for problem (5.33) is then the key to a successful implementation. The simple structure of $\widetilde{A}$ compared to A should guarantee, that such solutions are performed much faster, than if we applied the solver to the original problem (5.13) directly, and thus advocate the use of BDDC as the preconditioner in PCG.

A multifrontal method [15] is well-suited for this purpose. We based our implementation on the MUMPS package [1], an interesting open source parallel realization of the multifrontal method.

Our algorithm uses the generalized change of variables (Section 5.4), followed by the projected BDDC described in Section 5.3 and Section 5.4.

Let us describe the procedure more precisely. Let us recall Algorithm 5, where the key part is to find the solution to problem

$$\left[\overline{P}T^T\widetilde{A}T\overline{P} + \overline{s}(I - \overline{P})\right]\overline{w} = \overline{P}T^T E^T r. \tag{6.12}$$

First we use the trick of a virtual renumbering of the mesh, such that it appears as disconnected along the interface, except at corners. Then a standard assembly procedure is used to generate matrix $\widetilde{A}$, stored distributed among processors.

This matrix is then transformed glob-by-glob by matrix T from both sides. Finally, projection P is applied, again, glob-wise to explicitly construct matrix

$$\widetilde{A}^{avg} = \overline{P}T^T\widetilde{A}T\overline{P} + \overline{s}(I - \overline{P}).$$

Scaling coefficient $\bar{t}$ is chosen as the absolute value of the largest diagonal entry of the stiffness matrix.

The MUMPS package is used in the setup phase of the method to factorize matrix $\widetilde{A}^{avg}$. The factors are then reused in each iteration of the PCG method for backsubstitution, thus finding the solution to problem (6.12) for current residual r of PCG.

The algorithm has been implemented in Fortran 90 programming language and MPI library is used for parallelization.

6.3 Details of the algorithm

This section describes several tricks used in the algorithm, that are worth noticing.

6.3.1 Algorithm of preconditioned conjugate gradient method for BDDC

Let us return to the abstract definition of the BDDC preconditioner in the form of Algorithm 1 to show the main idea. Minor technical changes for other algorithms of Section 5.2 would be necessary, however, they are not discussed here.

Since the main part of the BDDC preconditioner is performed in space $\widetilde{W}$, (cf. Section 5.2 for definition), while the PCG method iterates in the continuous space U_Γ, double dimensioning of arrays and matrices seems necessary. However, using both spaces may be circumvented in the iterative process by redefinition of the PCG algorithm into the larger space $\widetilde{W}$.

The starting point is the standard algorithm of preconditioned conjugate gradient method for the solution of (5.7) in the form that can be found, e.g., in [27].

Algorithm 6 *Standard PCG for system* $Au = f$.

1. $u_0 = 0$, $r_0 = f$, $h_0 = Mr_0$, $p_0 = r_0$
2. ***for*** n=$1, 2, \ldots$ ***Do*** *until convergence:,...*
3. $\alpha_n = \left(r_{n-1}^T h_{n-1}\right) / \left(p_{n-1}^T A p_{n-1}\right)$
4. $u_n = u_{n-1} + \alpha_n p_{n-1}$
5. $r_n = r_{n-1} - \alpha_n A p_{n-1}$
6. $h_n = Mr_n$
7. $\beta_n = \left(r_n^T h_n\right) / \left(r_{n-1}^T h_{n-1}\right)$
8. $p_n = h_n + \beta_n p_{n-1}$
9. ***End Do***

At this point, a more detailed description of the connection between spaces U_Γ and $\widetilde{W}$ is needed. Let us recall, that it is realized by the operator E defined by (5.18) and its transpose.

One of possible realizations of E, used also in our implementation, is to split it into two operations as $E = R^T D_p$, where

$$D_p : \widetilde{W} \to \widetilde{W} \tag{6.13}$$

is a weight matrix representing the decomposition of unity on disconnected interface nodes, and

$$R : U_\Gamma \to \widetilde{W} \tag{6.14}$$

is an injection operator. It is realized by a simple copy of designated interface entries into the disconnected counterparts in $\widetilde{W}$. This means, that a vector $v \in U_\Gamma$ injected into $\widetilde{W}$ has some interface entries (except corners) copied in multiple places. Let us give it a special symbol

$$\widetilde{v} \in \widetilde{W} : \widetilde{v} = Rv, \text{ where } v \in U_\Gamma.$$

Its transpose R^T is realized by the sum of all contributions from disconnected interface entries stored into the designated position.

It holds

$$ER = R^T D_p R = I_{U_\Gamma}. \tag{6.15}$$

The construction of D_p is somewhat arbitrary, provided that relation (6.15) is satisfied. The simplest method uses as weigths for an interface node the reciprocal value to the number of subdomains adjacent to the node. This corresponds to usage of the mean value for the resulting continuous function. However, to improve robustness with respect to jumps in coefficients in the model, more sophisticated constructions are necessary [45]. A weighted average, that uses diagonal entries of the system matrix as weights, presents another useful approach. Both options were implemented into our program, and they are chosen according to the problem properties.

The following relation holds for operators A and $\widetilde{A}$

$$A = R^T \widetilde{A} R. \tag{6.16}$$

Let us require, that all quantities evaluated within PCG are computed as if we were iterating with matrix A on vectors from U_Γ. We need to derive some equivalences used in our implementation of PCG.

From (6.16), we get

$$Ap = R^T \widetilde{A} R p = R^T \widetilde{A} \widetilde{p}, \tag{6.17}$$

embedding into $\widetilde{W}$, we get

$$RAp = RR^T \widetilde{A} \widetilde{p},$$

and since $\widetilde{p}^T = p^T R^T$, we also get

$$p^T Ap = p^T R^T \widetilde{A} R p = \widetilde{p}^T \widetilde{A} \widetilde{p}.$$

The action of the preconditioner can be evaluated as follows. From (5.20), we have

$$M = E\widetilde{A}^{-1} E^T = R^T D_p \widetilde{A}^{-1} D_p^T R.$$

Then, we can evaluate

$$h = Mr = R^T D_p \widetilde{A}^{-1} D_p^T R r = R^T D_p \widetilde{A}^{-1} D_p^T \widetilde{r}.$$

Since $\widetilde{h} = Rh$ and $\widetilde{r}^T = r^T R^T$, we get

$$r^T h = r^T R^T D_p \widetilde{A}^{-1} D_p^T \widetilde{r} = \widetilde{r}^T D_p \widetilde{A}^{-1} D_p^T \widetilde{r}.$$

We note that from (6.15), we also have

$$\|r\| = \sqrt{r^T r} = \sqrt{r^T R^T D_P R r} = \sqrt{\widetilde{r}^T D_p \widetilde{r}}.$$

We are ready to present the version of PCG used in the implementation.

Algorithm 7 *Modified PCG embedded into $\widetilde{W}$ for system $Au = f$.*

1. $\widetilde{f} = Rf$, $\widetilde{u_0} = Ru_0$
2. $\widetilde{r}_0 = \widetilde{f} - RR^T \widetilde{A} \widetilde{u_0}$
3. $\widetilde{z}_0 = D_P \widetilde{A}^{-1} D_P \widetilde{r}_0$
4. $\widetilde{h}_0 = RR^T \widetilde{z}_0$
5. $\widetilde{p}_0 = \widetilde{h}_0$
6. ***For*** $n = 1, 2, \ldots$ ***Do*** *until convergence:*
7. $\quad \alpha_n = \left(\widetilde{r}_{n-1}^T \widetilde{z}_{n-1}\right) / \left(\widetilde{p}_{n-1}^T \widetilde{A} \widetilde{p}_{n-1}\right)$
8. $\quad \widetilde{u}_n = \widetilde{u}_{n-1} + \alpha_n \widetilde{p}_{n-1}$
9. $\quad \widetilde{r}_n = \widetilde{r}_{n-1} - \alpha_n RR^T \widetilde{A} \widetilde{p}_{n-1}$
10. $\quad \widetilde{z}_n = D_P \widetilde{A}^{-1} D_P \widetilde{r}_n$
11. $\quad \widetilde{h}_n = RR^T \widetilde{z}_n$
12. $\quad \beta_n = \left(\widetilde{r}_n^T \widetilde{z}_n\right) / \left(\widetilde{r}_{n-1}^T \widetilde{z}_{n-1}\right)$
13. $\quad \widetilde{p}_n = \widetilde{h}_n + \beta_n \widetilde{p}_{n-1}$
14. ***End Do***

The *convergence* for tolerance tol is obtained in the n^{th} iteration if

$$\frac{\|r_n\|}{\|f\|} = \frac{\sqrt{\widetilde{r}_n^T D_P \widetilde{r}_n}}{\sqrt{\widetilde{f}^T D_P \widetilde{f}}} < tol. \tag{6.18}$$

6.3.2 Efficient inverse of transformation matrix

The role of this section is to support the *generalized change of variables* described in Section 5.4. Let us recall, that the change of variables is given by

$$\overline{w} = Bw, \tag{6.19}$$

where operator B is represented by an invertible matrix containing weights of the averages intended to constrain.

We then need to construct an explicit matrix $T^T \widetilde{A} T$ to use it for preconditioning, where $T = B^{-1}$ is the inverse of B.

At the first sight, such construction might be a perfect nightmare for any numerical mathematician – there are two matrix-matrix multiplications and an explicit inverse of a matrix of dimension of space $\widetilde{W}$, i.e. even larger than the original problem.

However, let us have a deeper look at these operations. The first important fact is, that matrix B has very special structure. The change of variables transforms degrees of freedom only within a glob, without coupling several globs together. This fact results in a block diagonal structure of B with size of blocks given by number of degrees of freedom within that glob, and identity in all degrees of freedom outside globs (interiors of subdomains and corners). This means, that the inverse T has the same block diagonal structure and may be found glob by glob, as well as the application of T to $\widetilde{A}$.

Moreover, if equivalence of only one average is requested on every admissible glob (the case of arithmetic averages), only a single line of the block of B corresponding to each glob is not a row of identity matrix, and its inverse may be written directly as in example (5.44), and in [42].

But even in the case of equivalence of several averages on a glob (e.g. in adaptive selection of constraints [47]), we can proceed quite efficiently. Let us now consider change of variables in a block of the global transformation matrix corresponding to only one glob B_G. We have already mentioned that the prescribed averages can be quite general: their number is limited by the number of degrees of freedom on this glob.

Let us denote by B_{AVG} the block of the matrix B_G that prescribes the averages, i.e.

$$B_{AVG} = \begin{bmatrix} v_1^T \\ \vdots \\ v_n^T \end{bmatrix},$$

where $v_1, \ldots, v_n$ are weigths of n averages. Let us assume that the vectors $v_1, \ldots, v_n$ are linearly independent. Then, we can complete B_G by block of identity as

$$B_G = \begin{bmatrix} B_{AVG} \\ 0 \;\; I \end{bmatrix} = \begin{bmatrix} V & W \\ 0 & I \end{bmatrix}, \tag{6.20}$$

to arrive at an invertible matrix. Above, we have schematically split the block $B_{AVG} = \begin{bmatrix} V & W \end{bmatrix}$ for the only reason: the inverse $T_G = B_G^{-1}$ can be found more effectively by inverting only much smaller block V, with the size n, i.e. the number of averages, as

$$T_G = \begin{bmatrix} V^{-1} & -V^{-1}W \\ 0 & I \end{bmatrix}. \tag{6.21}$$

It is not obvious, and in fact, it is often not the case, that block V is readily invertible. However, we may first perform a factorization of the rectangular block B_{AVG} with column permutations as

$$B_{AVG} = LUP,$$

where L is a lower triangular matrix, rectangular matrix U has zeros under its diagonal, and P is the matrix of column permutations. It holds, that

$$\text{null}\,(B_{AVG}) = \text{null}\,(UP)\,, \tag{6.22}$$

which means, that constraints in UP define the same subspace as those in B_{AVG}.

Let us now define matrix B'_G, such that $B_G = B'_G P$. Because for permutation, $P^{-1} = P$,

$$T_G = B_G^{-1} = P(B'_G)^{-1}.$$

We can thus obtain matrix T_G by permutation of the inverse of matrix B'_G, where we repeat the trick with block inverse above, just using block U instead of B_{AVG} as

$$B'_G = \begin{bmatrix} U \\ 0 \;\; I \end{bmatrix} = \begin{bmatrix} V' & W' \\ 0 & I \end{bmatrix}. \tag{6.23}$$

Then

$$T_G = P \begin{bmatrix} (V')^{-1} & -(V')^{-1} W' \\ 0 & I \end{bmatrix},$$

where $(V')^{-1}$ is the inverse of a square triangular matrix of the size of number of averages on glob n.

In the case of linearly dependent rows in B_{AVG}, only the meaningful block of U is used in (6.23) and rows of zeros are excluded. The identity block is then extended in place of these truncated rows. By (6.22), the subspace defined by such constraints remains the same.

Let us now relate the general matrix B_G from (6.20) to the one used for one arithmetic average in (5.43), and extend it to a glob of general size. The general block B_{AVG} is in this case reduced into the first line of the matrix B_G, and so the block V from (6.20) is reduced just to the diagonal entry of 1. It is also remarkable that in this case, the inverse of transformation matrix for glob of any size is explicitly known. It is given as

$$B_G = \begin{bmatrix} 1 & 1 & \dots & 1 \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{bmatrix}, \quad T_G = B_G^{-1} = \begin{bmatrix} 1 & -1 & \dots & -1 \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{bmatrix}. \tag{6.24}$$

This formula may be trivially verified by (6.20), and it was the core of the change of variables presented in [42].

6.3.3 Storing the matrices in memory

When using Algorithm 5, both marices $\widetilde{A}$ and $\left[\overline{P}T^T\widetilde{A}T\overline{P} + \overline{s}(I - \overline{P})\right]$ are necessary simultaneously; the former in the modified PCG algorithm (Algorithm 7 in Section 6.3.1), and the latter in the BDDC preconditioner (cf. Sections 5.4 and 6.2).

Because it is the input format of the MUMPS package (Section 6.2), the coordinate format is used to store the sparse matrix in computer memory. It consists of two integer arrays and one real array of the length of number of nonzero entries in the matrix. For each nonzero entry, they represent the row index, the column index, and the value, respectively. Multiple entries with the same row and column indices are allowed, and sum of such values is considered.

To save the memory space, we would like to distinguish between the entries of original matrix $\widetilde{A}$, and of matrix $\left[\overline{P}T^T\widetilde{A}T\overline{P} + \overline{s}(I - \overline{P})\right]$.

First, we need to construct matrix $T^T\widetilde{A}T$. Since this transformation, in general, does not preserve entries in $\widetilde{A}$, we would have to put zeros into the original matrix, rewriting all the new entries behind the original matrix. However, that is not desired for the iterative process.

This may be circumvented by using matrix

$$\mathcal{T} = T - I \tag{6.25}$$

instead of T, where I stands for identity matrix. With this modification, the transformation of sparse matrix only adds new entries, leaving original entries unchanged. The transformation in the implementation is done by

$$T^T\widetilde{A}T = (I + \mathcal{T}^T)\widetilde{A}(I + \mathcal{T}) = \widetilde{A} + \widetilde{A}\mathcal{T} + \mathcal{T}^T\widetilde{A} + \mathcal{T}^T\widetilde{A}\mathcal{T}. \tag{6.26}$$

This formulation allows to only put new entries in memory behind the original matrix. In case of symmetric matrix $\widetilde{A}$, we can observe, that the third block may be written as $(\widetilde{A}^T\mathcal{T})^T$, which for this case simplifies to $(\widetilde{A}\mathcal{T})^T$, i.e. just transposition of the previous block. If only one triangle of the final matrix is stored, we can proceed by flipping one triangle of $\widetilde{A}\mathcal{T}$ along diagonal and doubling diagonal entries to find both blocks $\widetilde{A}\mathcal{T}$ and $\mathcal{T}^T\widetilde{A}$. However, block $\mathcal{T}^T(\widetilde{A}\mathcal{T})$ must be determined before this flip.

Remark 6 *It is worth emphasizing, that the trick in (6.25) leads to a very simple matrix, if no permutation is applied in (6.3.2) (cf. Section 6.3.2). For example, in the case of arithmetic averages on globs, matrix T_G in (6.24) simplifies to*

$$\mathcal{T}_G = \begin{bmatrix} 0 & -1 & \dots & -1 \\ & 0 & & \\ & & \ddots & \\ & & & 0 \end{bmatrix}. \tag{6.27}$$

Thus, appart of the desired leaving the original entries of matrix unchanged, this approach leads to much more efficient usage of memory and lower time of the whole transformation.

However, the latter advantage is lost, if the identity block of (6.3.2) is scattered by the permutation P in construction of T_G.

After the matrix is transformed, we need to apply projection $\overline{P}$ to get $\left[\overline{P}T^T\widetilde{A}T\overline{P} + \overline{s}(I - \overline{P})\right]$ (cf. Sections 5.3 and 5.4). Let us now denote $\mathcal{A} \equiv T^T\widetilde{A}T$.

The way to proceed from $\mathcal{A}$ to $\left[\overline{P}\mathcal{A}\overline{P} + \overline{s}(I - \overline{P})\right]$ is summarized in the following algorithm.

Algorithm 8 *Projection of matrix $\mathcal{A}$ onto the nullspace of matrix of constraints $\overline{G}$.*

1. *Use QR-algorithm to factor $\overline{G}^T = QR$ and store R.*

   Then $\overline{G}^T\left(\overline{G}\overline{G}^T\right)^{-1}\overline{G} = \overline{G}^TR^{-1}R^{-T}\overline{G} = F^TF$, where F is unknown. Since $F = R^{-T}\overline{G}$, we also have that $R^TF = \overline{G}$.

2. *Solve $R^TF = \overline{G}$ as a system with multiple right-hand sides and store F (as a dense matrix). Substituting $\overline{P} = I - F^TF$ into (5.47), we get*

$$\left[\mathcal{A} - F^TF\mathcal{A} - \mathcal{A}F^TF + F^TF\mathcal{A}F^TF + \bar{t}F^TF\right]u = \left(I - F^TF\right)T^TE^Tr. \tag{6.28}$$

3. *Multiply sparse $\mathcal{A}$ by dense F^T to get dense $\mathcal{A}F^T = H$ and store H (as a dense matrix). The equation (6.28) can now be written as*

$$\left[\mathcal{A} - F^T H^T - HF + F^T FHF + \bar{t}F^T F\right] u = \left(I - F^T F\right) T^T E^T r. \tag{6.29}$$

4. *Use sparse matrix $\mathcal{A}$ and dense H and F to compute the matrices in (6.29).*

Note that, again, the matrix in (6.29) allows the storage of original entries of $\mathcal{A}$ independently of the new entries obtained by the projection.

Let us summarize the final ordering of consecutive parts of entries of the sparse matrix in BDDC, as they are stored in the computer memory:

1. original entries of matrix $\widetilde{A}$,
2. new entries from the generalized change of variables described in (6.26),
3. new entries from projection to nullspace of $\overline{G}$ described in (6.29).

While we iterate only with entries of the first item, we pass entries of all the above-mentioned items to the preconditioner.

Chapter 7

Numerical results

This chapter binds together numerical results obtained with the methods presented in the thesis.

The *semiGLS* method proposed in Chapter 4 has been applied to both steady and unsteady problems of Navier-Stokes flow. The accuracy of the stabilized method was analyzed by methods proposed in Section 4.3. Several results of these experiments are presented in Section 7.1.

Although it was not the main aim of the work, the algorithm of BDDC described in Chapter 6 was first extensively tested on problems of linear elasticity (5.1) – (5.3), and some results are presented in Section 7.2. This problem has symmetric positive definite bilinear form and thus is covered by the theory of BDDC preconditioner.

The applicability of the BDDC preconditioner was then successfully investigated on problem of the Stokes flow, which is not covered by the theory presented in Chapter 5 due to its indefiniteness. These results are presented in Section 7.3.

The reader shoud note, that the *semiGLS* stabilization has not been implemented into the BDDC program yet. This means that results presented in Section 7.1 were obtained simply by a serial direct solver (unsymmetric frontal method). Results of Sections 7.2 and 7.3 were obtained by parallel implementation of BDDC, however, all these results are obtained without stabilization.

7.1 Applications of semiGLS stabilization to Navier-Stokes flow

The *semiGLS* method was tested on several problems for verification and to review its behaviour. Both enclosed flows ($\partial\Omega = \partial\Omega_g$) and inflow/outflow cases ($\partial\Omega_{dn} \neq 0$) are considered. Results obtained by the algorithm are marked as *semiGLS* algorithm results.

7.1.1 Steady flow of lid driven cavity

A popular benchmark problem for testing numerical schemes for viscous flows is the 'lid driven cavity'. Computational domain is of square shape with unit lenght of side. Dirichlet boundary conditions are prescribed on the whole boundary: value of horizontal velocity is prescribed on the upper side, zero boundary conditions on the rest of the boundary representing a wall.

Many solutions of this problem were presented by various authors. Here are some representatives: in [29], solutions for Reynolds numbers 1,000, 3,200, and 5,000 obtained on nonuniform grid of approximately 8,800 elements are presented; in [24], a result for Reynolds number 7,500 on quasi-uniform mesh of 96×96 elements is presented; solutions for Reynolds number 10,000 obtained by several methods on the mesh of 64×64 elements are published in [59].

Solution by the developed algorithm was performed on three uniform meshes of quadrilateral elements – of 32×32 (1,024) elements, of 64×64 (4,096) elements, and of 128×128 (16,384)

elements.

To observe the effect of stabilization, solutions obtained by the Newton method without stabilization together with solutions computed by the semiGLS algorithm for Reynolds number 10,000 on all three meshes are presented in Figures 7.1–7.3. Moreover, we can review the sensitivity to the fineness of the computational mesh in these figures.

Differences of solutions obtained by the semiGLS method from those obtained by the Newton method computed by (4.23) are summarized in Table 7.1. They are quite high for the problem of cavity.

Comparisons of a posteriori error estimates for problems of 32×32, 64×64, and 128×128 elements at Reynolds number 10,000 are presented in Figures 7.4–7.6. In presented plots, AEE is an abbreviation for *a posteriori error estimator* from equation (4.24). Let us note that, as was already suggested at the end of Section 4.3, our error estimator does not yield the exact value of the approximation error. Still it is satisfactory enough to detect elements where the inaccuracy is higher compared with other elements. To present such comparisons, Reynolds number for this experiment was restricted to value, for which we are able to obtain solution also by Galerkin method without stabilization. It does not depreciate an important advantage of this evaluation – we are able to get an idea about the error distribution for any Reynolds number, for which the stabilized method converges. We can observe in these plots, that the inacurracy introduced by the stabilization remains present in the computation even if we are refining the mesh.

mesh	32×32	64×64	128×128
$\delta_{u_{h1}}$ [%]	41.69	39.07	21.42
$\delta_{u_{h2}}$ [%]	70.81	49.12	22.24
δ_{p_h} [%]	197.90	137.10	42.82

Table 7.1: Differences between solutions obtained with and without stabilization

To extend results to higher Reynolds numbers, solution for Re = 100,000 on the mesh of 128×128 is presented in Figure 7.7.

Althought the continuation method was applied to achieve higher Reynolds numbers, we detected limits of convergence of the Newton method for all three meshes. We observed, that on the mesh of 32×32, we were not able to get results above Re ≈ 28,000, on the mesh of 64×64 above Re ≈ 50,000, and on the mesh of 128×128 above Re ≈ 120,000. For comparison, such limit was around Re ≈ 12,500 on the mesh of 32×32 for the method without stabilization.

Another interesting effect was observed during the computations. Since it is known, that stabilized methods are, in general, sensitive to stabilization parameters, we tried to modify the computed parameter τ by a quotient 0.7 to 1.5. This improved the convergence, and we were able to reach higher Reynolds numbers, e.g. Re = 70,000 on the mesh of 64×64 elements.

Finer mesh is able to catch more vortices and provides better resolution, moreover presented experiments showed another important conclusion – suitable refinement of the mesh significantly improves stability of solution, i.e. convergence.

7.1.2 Steady flow in channel with sudden extension of diameter

Steady flow in 2D channel with abruptly extended diameter (Figure 7.8) is another testing problem. This problem is complicated due to singularities of solution in the vicinity of nonconvex internal corners. The aspect of suitable mesh generation for such problems was studied in [7] and [53]. Streamlines are presented in Figure 7.9 for Reynolds number 1,000. For the symmetry of the problem, solution is found only on the upper half of the section. Differences between solutions

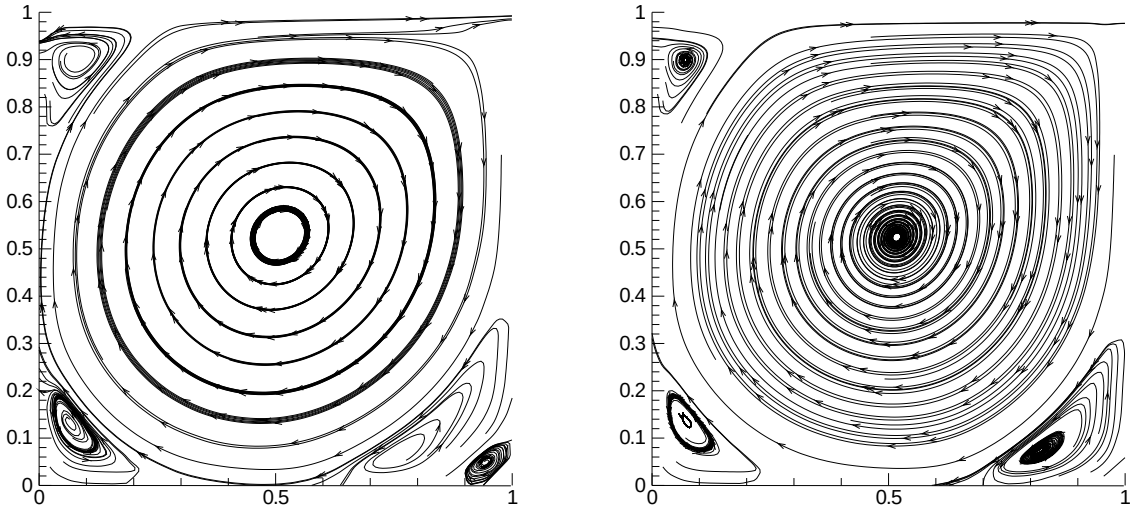


Figure 7.1: Streamlines by the Newton method without stabilization (left) and by the semiGLS algorithm (right), Re = 10,000, mesh 32×32

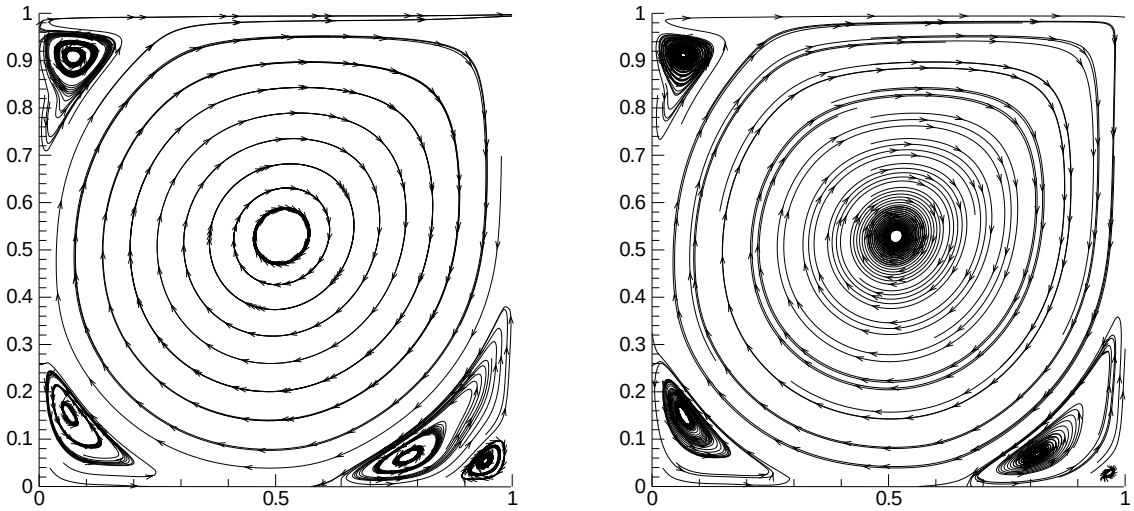


Figure 7.2: Streamlines by the Newton method without stabilization (left) and by the semiGLS algorithm (right), Re = 10,000, mesh 64×64

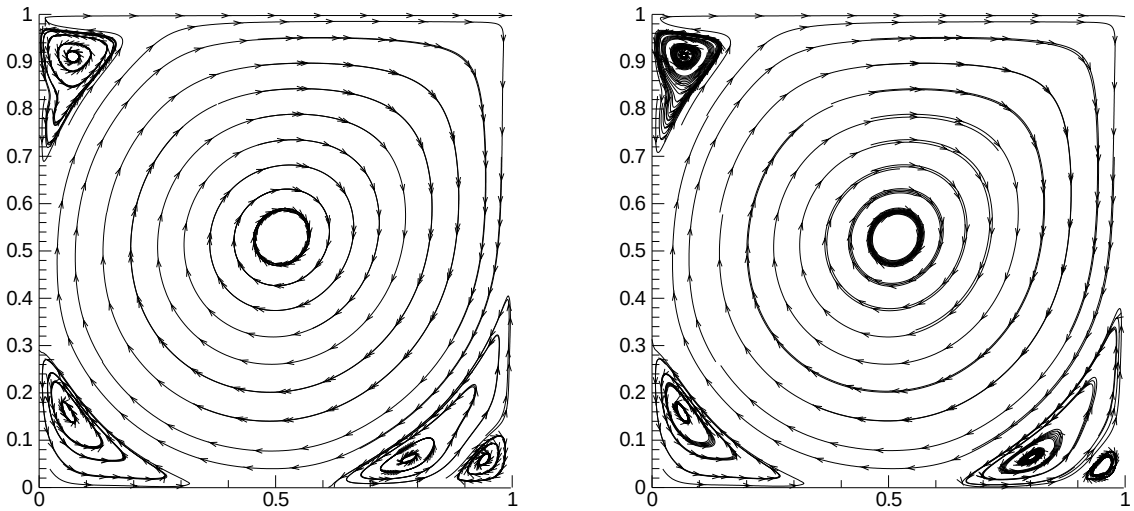


Figure 7.3: Streamlines by the Newton method without stabilization (left) and by the semiGLS algorithm (right), Re = 10,000, mesh 128×128

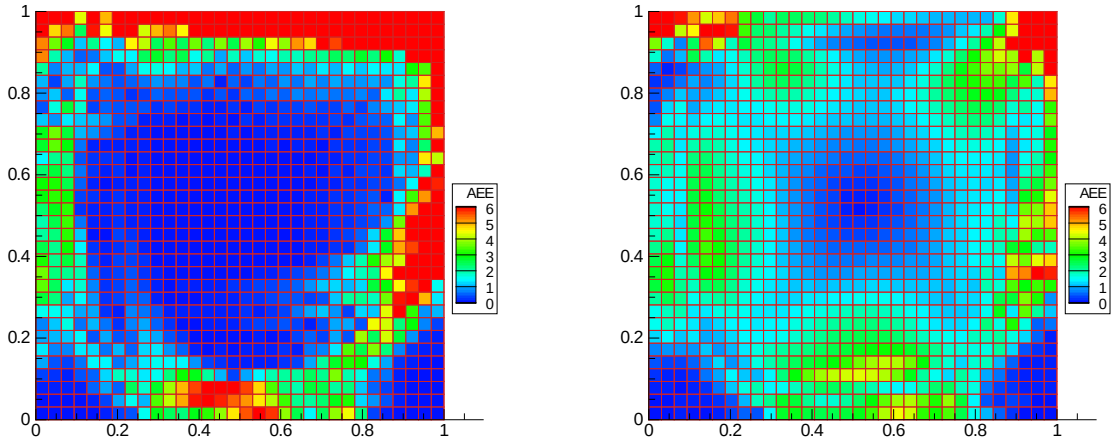


Figure 7.4: A posteriori errors on elements, cavity problem, Re = 10,000, uniform mesh 32×32 without stabilization (left) and by *semiGLS* method (right)

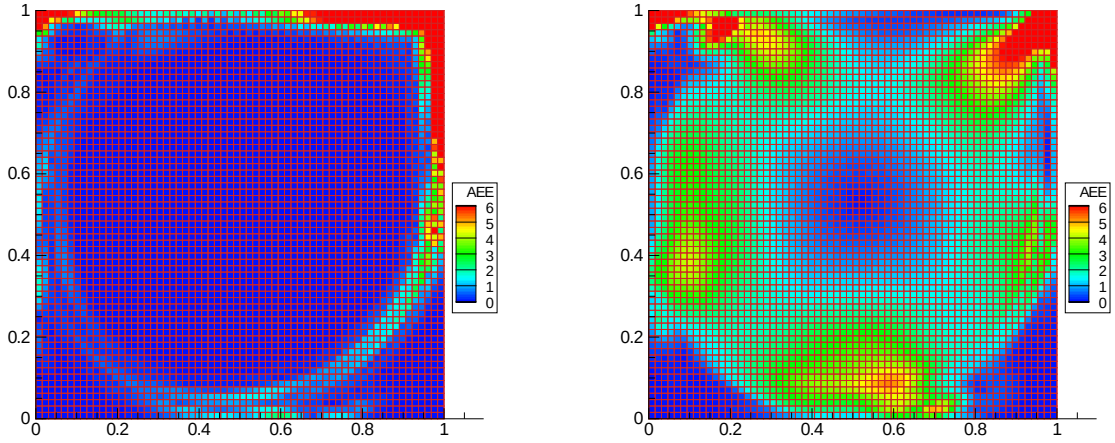


Figure 7.5: A posteriori errors on elements, cavity problem, Re = 10,000, uniform mesh 64×64 without stabilization (left) and by *semiGLS* method (right)

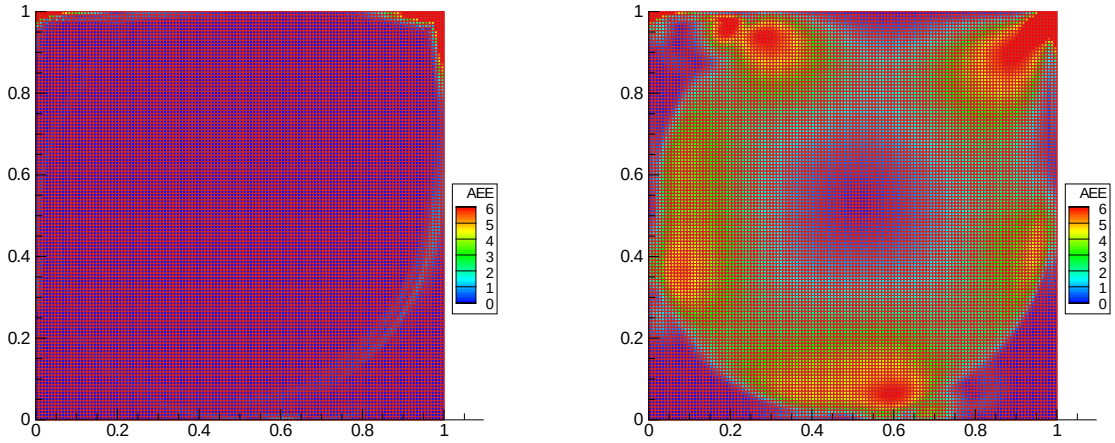


Figure 7.6: A posteriori errors on elements, cavity problem, Re = 10,000, uniform mesh 128×128 without stabilization (left) and by *semiGLS* method (right)

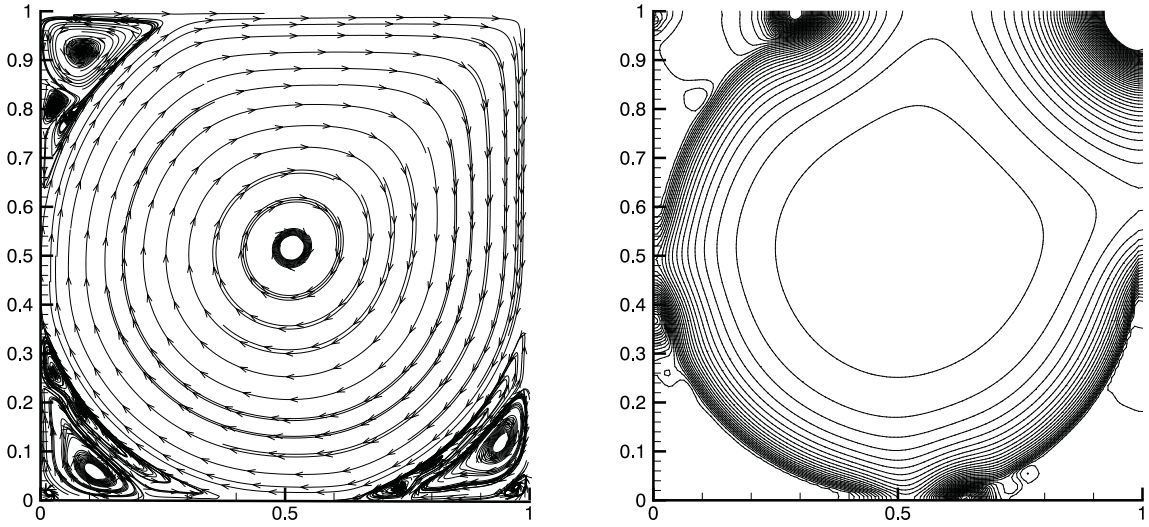


Figure 7.7: Streamlines (left) and pressure contours (right) by the semiGLS algorithm on the mesh of 128×128, Re = 100,000

computed by (4.23) are listed in Table 7.2 and the a posteriori error estimates are presented in Figure 7.10.

Parabolic horizontal velocity distribution is prescribed on the inflow (left) part of the boundary, 'do nothing' boundary condition (2.5) is considered on the outflow (right) part of it, zero velocity is prescribed on the upper part of the boundary representing wall with 'no slip' and symmetry is considered on the lower part of it (see [53] for details). Again, we can observe, that the error of semiGLS is spread to much larger region around the corner singularity than for the case of the solution without stabilization.

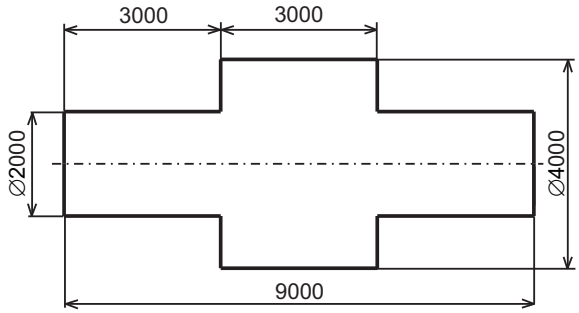


Figure 7.8: Geometry of the channel

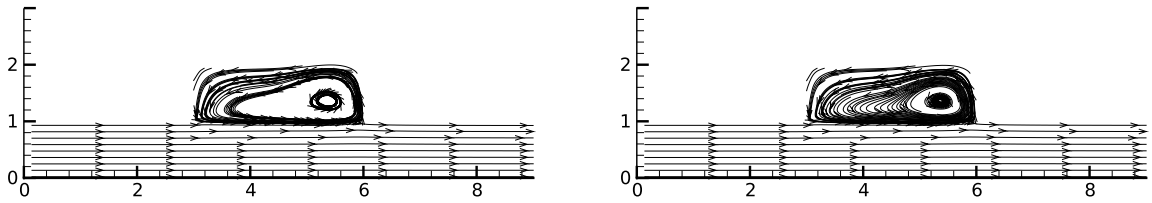


Figure 7.9: Streamlines in the channel by the Newton method without stabilization (left) and streamlines by the semiGLS algorithm (right), Re = 1,000

Additionally, we present streamlines, plots of velocities and pressure for Reynolds number 80,000 in Figures 7.11–7.12.

mesh	channel (Figure 7.8)
$\delta_{u_{h1}}$ [%]	0.0718
$\delta_{u_{h2}}$ [%]	2.7202
δ_{p_h} [%]	0.5139

Table 7.2: Differences between solutions obtained with and without stabilization

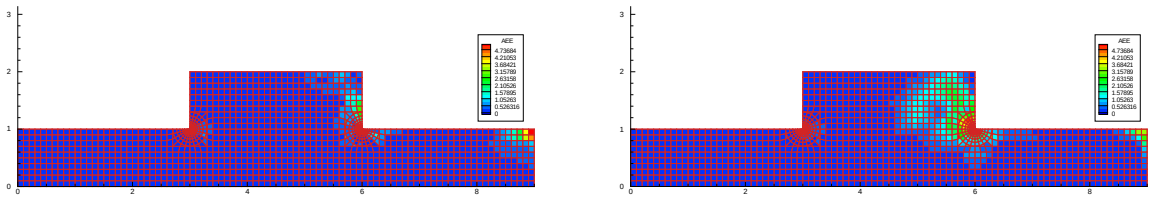


Figure 7.10: A posteriori error estimates in the channel for the Newton method without stabilization (left) and by the *semiGLS* method (right), Re = 1,000

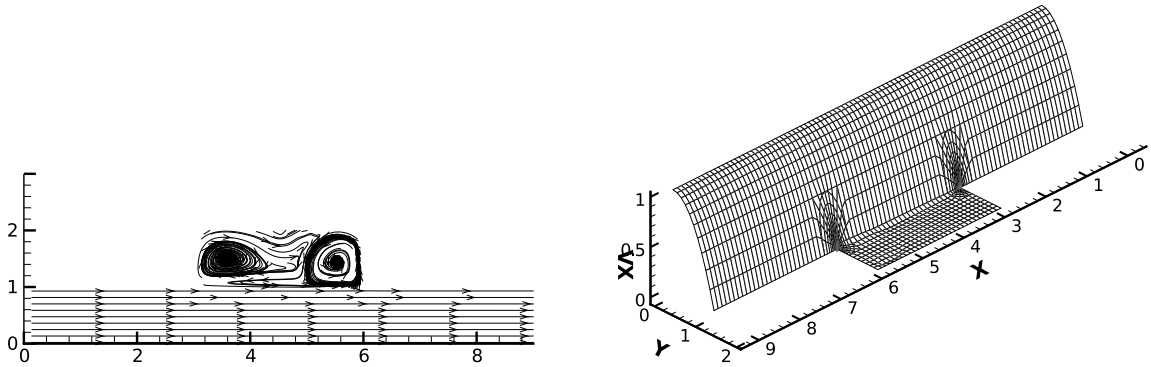


Figure 7.11: Streamlines (left) and plot of velocity u_{h1} (right) by the semiGLS algorithm, Re = 80,000

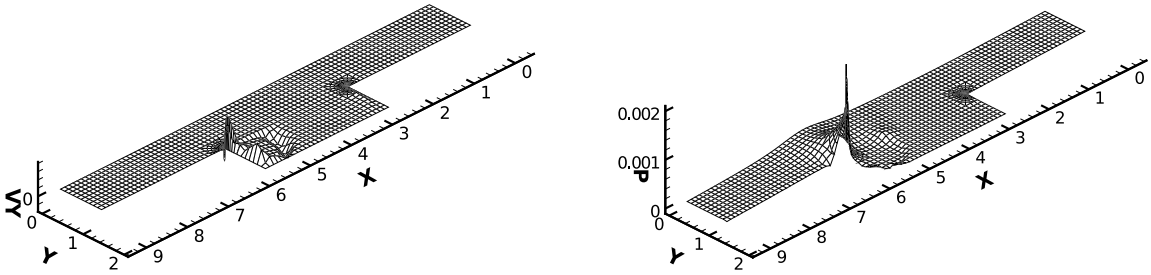


Figure 7.12: Plot of velocity u_{h2} (left) and pressure (right) by the semiGLS algorithm, Re = 80,000

7.1.3 Flow past NACA 0012 airfoil

Flow past NACA 0012 airfoil was investigated as a more practical application. The computational mesh is shown in Figures 7.13 and 7.14. It contains 6,220 elements, 18,478 nodes, and 43,085 degrees of freedom. We consider unit horizontal velocity on the left part of the boundary and 'do nothing' boundary condition (2.5) on the rest of it. For unsteady problem, zero initial condition is considered.

Results of the unsteady problem for angle of incidence of 34 deg and Reynolds number 1,000 obtained by the unconditionaly stable projection FEM were presented by Guermond and Quartapelle in [29]. In Figures 7.15–7.19, these results are compared to ours obtained by the semiGLS algorithm. Streamlines and pressure contours for the problem with Reynolds number 100,000 in several time layers are presented in Figures 7.20–7.23. Time step sizes for simulation for Reynolds number 1,000 are 0.01 and for Reynolds number 100,000 are 0.005 (see [53] for details).

Streamlines of the steady flow at Reynolds number 100 are presented in Figure 7.24. The Reynolds number for computing of the steady state was lowered to correspond to the real behaviour of the system – steady solution is unstable for higher Reynolds numbers where periodic solution plays the role of the stable state of the system. The a posteriori error estimates are presented in Figure 7.25. Again, the region with increased error estimate is larger than for the method without stabilization, indicating a loss of accuracy compared to the classical Galerkin method.

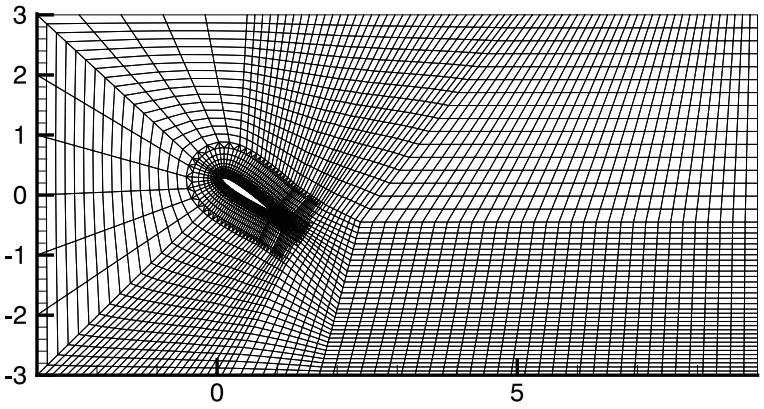


Figure 7.13: Computational mesh for NACA 0012 problem, angle of incidence of 34 °

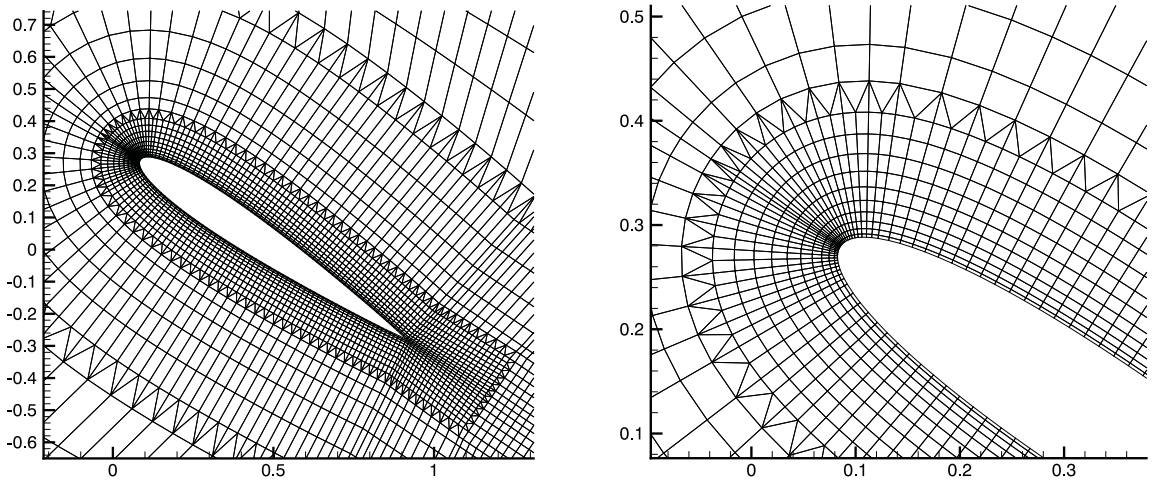


Figure 7.14: Computational mesh for NACA 0012 problem - details

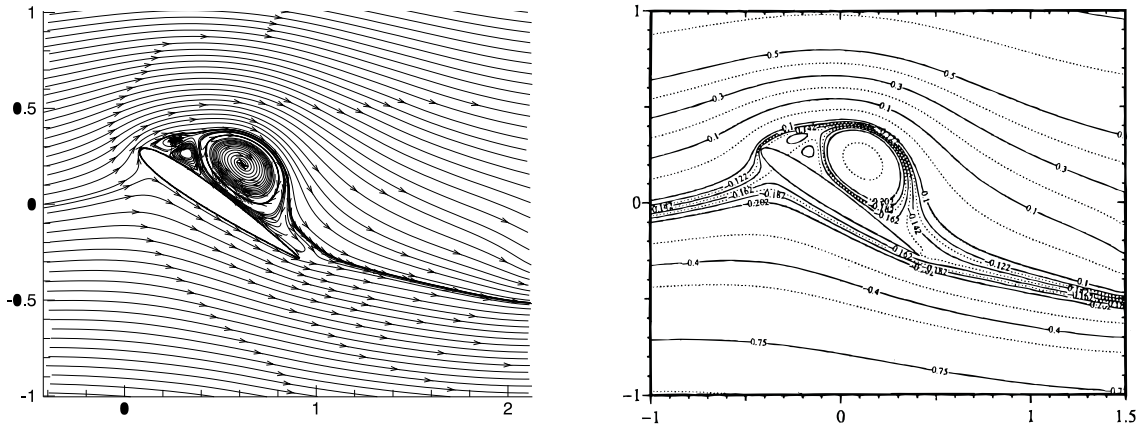


Figure 7.15: Streamlines by the semiGLS algorithm (left) and by [29] (right), $t = 1.6s$, Re = 1,000

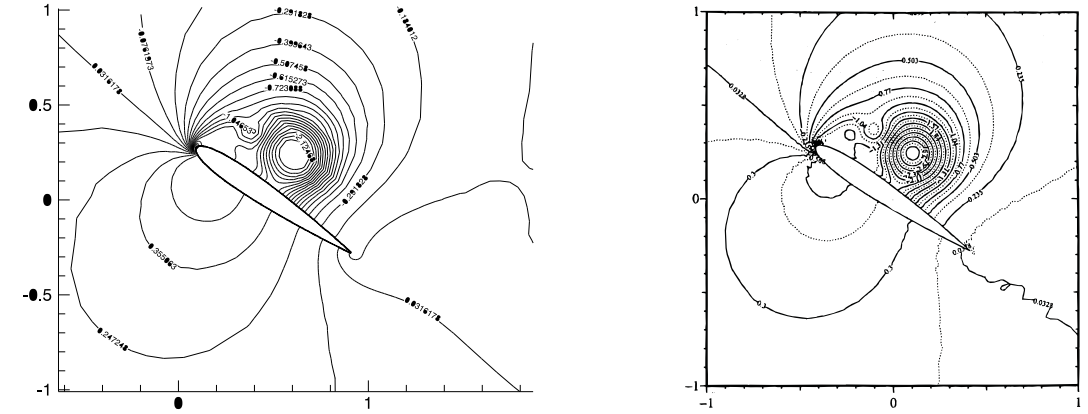


Figure 7.16: Pressure contours by the semiGLS algorithm (left) and by [29] (right), $t = 1.6s$, Re = 1,000

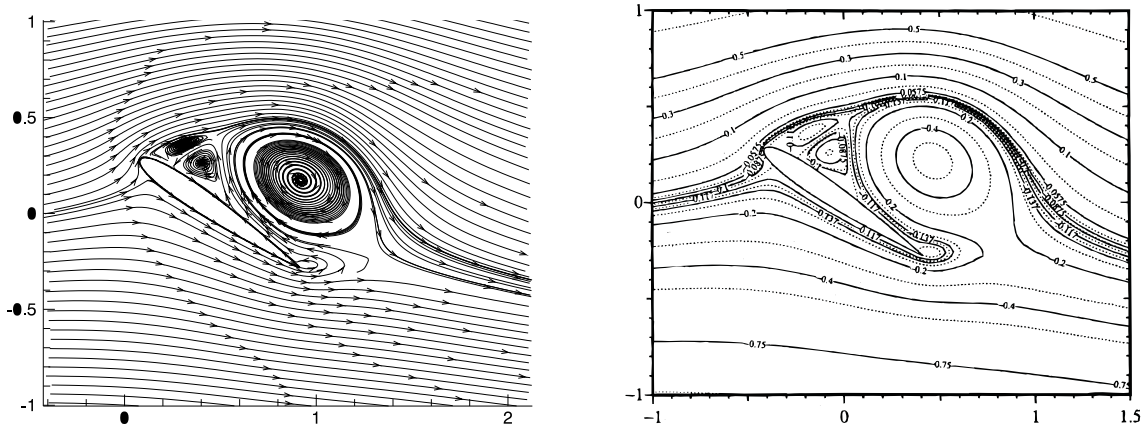


Figure 7.17: Streamlines by the semiGLS algorithm (left) and by [29] (right), $t = 2.6s$, Re = 1,000

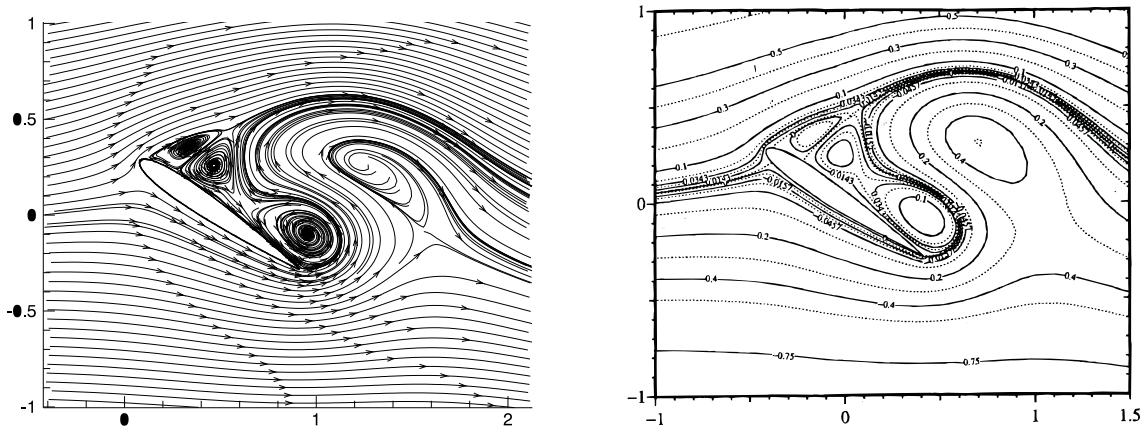


Figure 7.18: Streamlines by the semiGLS algorithm (left) and by [29] (right), $t = 3.6s$, Re = 1,000

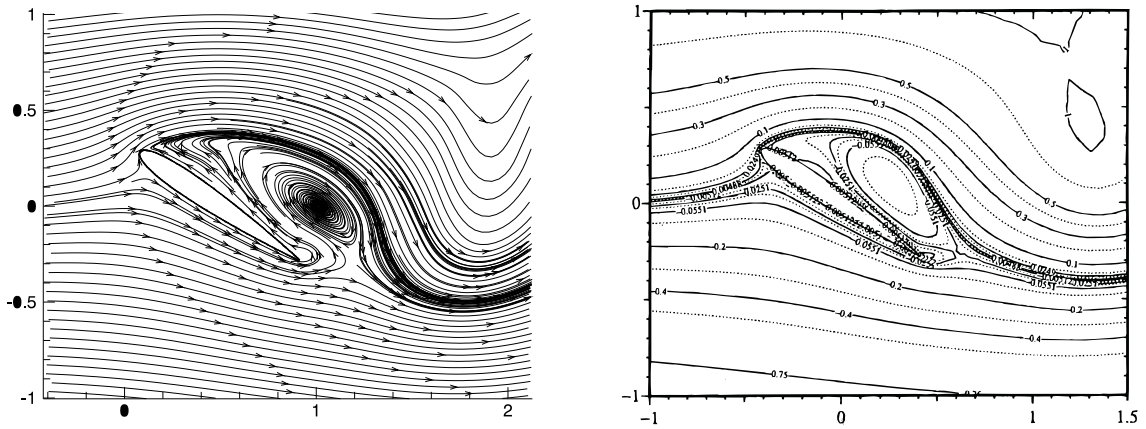


Figure 7.19: Streamlines by the semiGLS algorithm (left) and by [29] (right), $t = 6.0s$, Re = 1,000

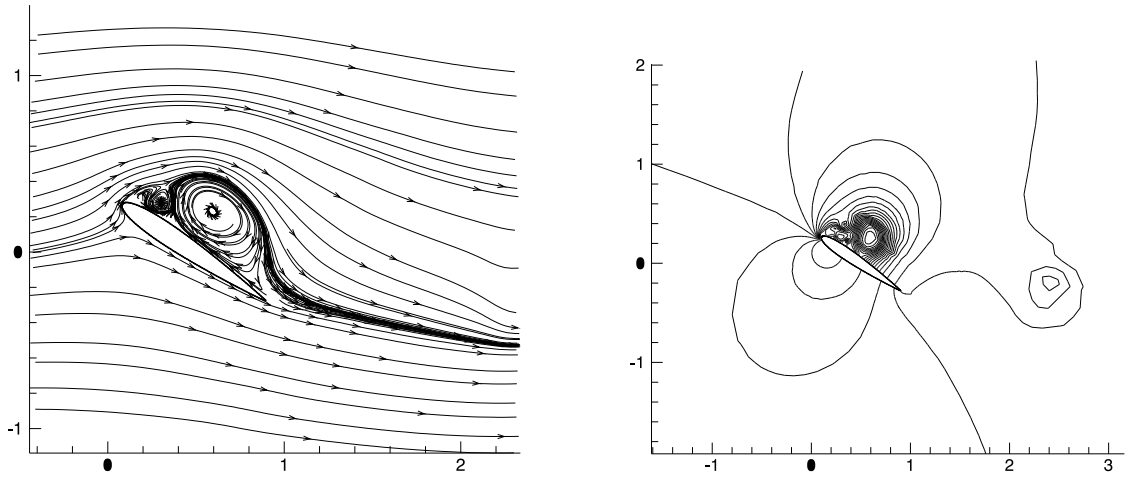


Figure 7.20: Streamlines (left) and pressure contours (right) by the semiGLS algorithm, $t = 1.6s$, Re = 100,000

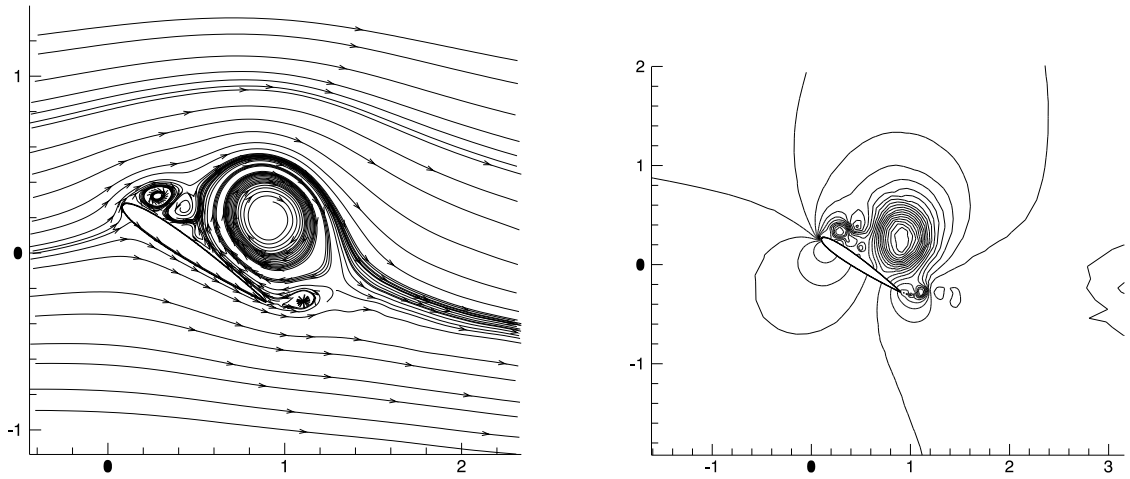


Figure 7.21: Streamlines (left) and pressure contours (right) by the semiGLS algorithm, $t = 2.6s$, Re = 100,000

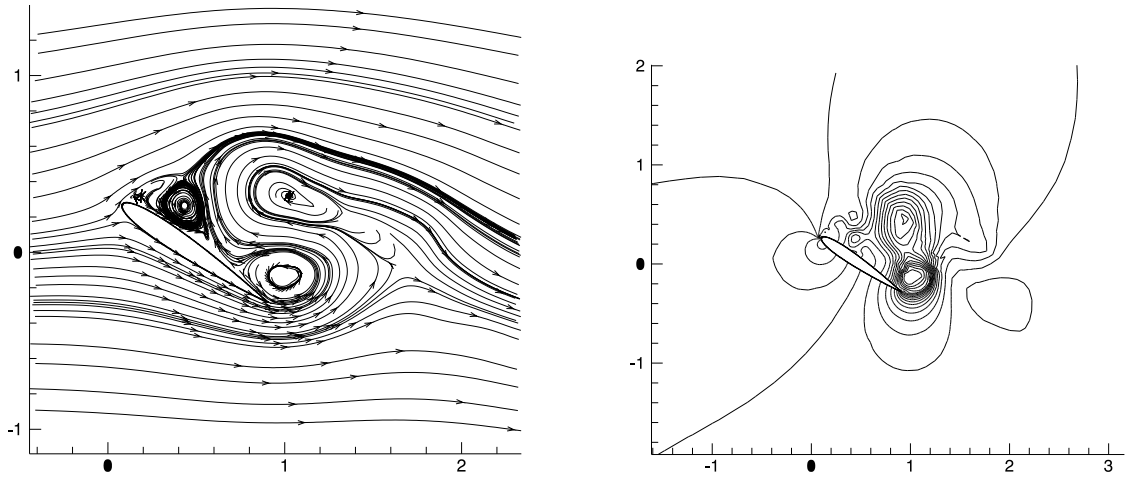


Figure 7.22: Streamlines (left) and pressure contours (right) by the semiGLS algorithm, $t = 3.6s$, Re = 100,000

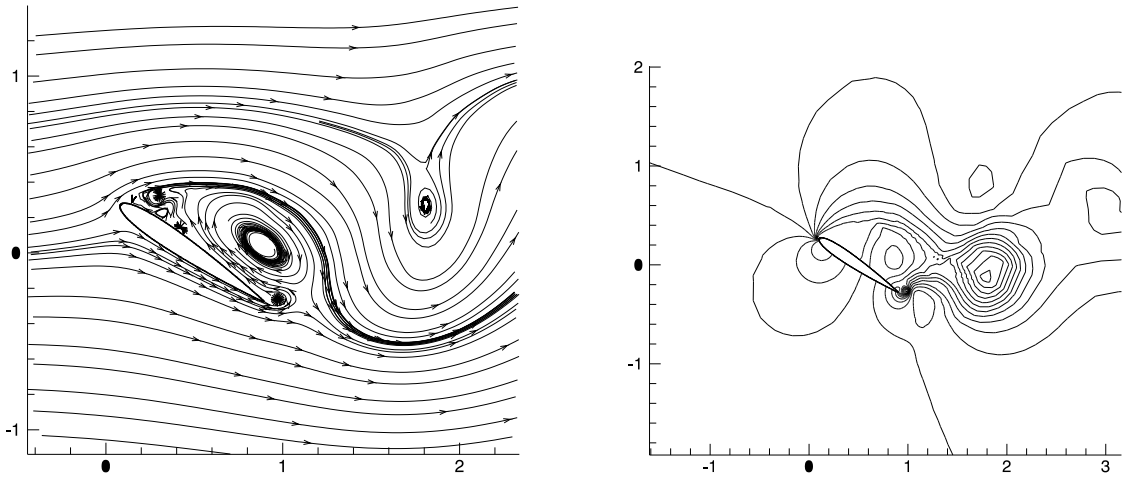


Figure 7.23: Streamlines (left) and pressure contours (right) by the semiGLS algorithm, $t = 6.0s$, Re = 100,000

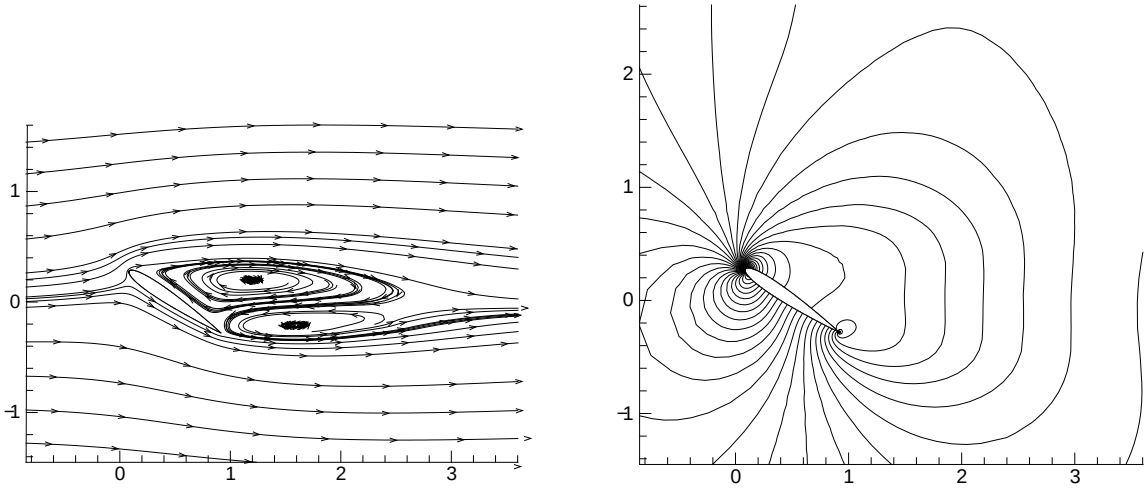


Figure 7.24: Streamlines (left) and pressure contours (right), Re = 100

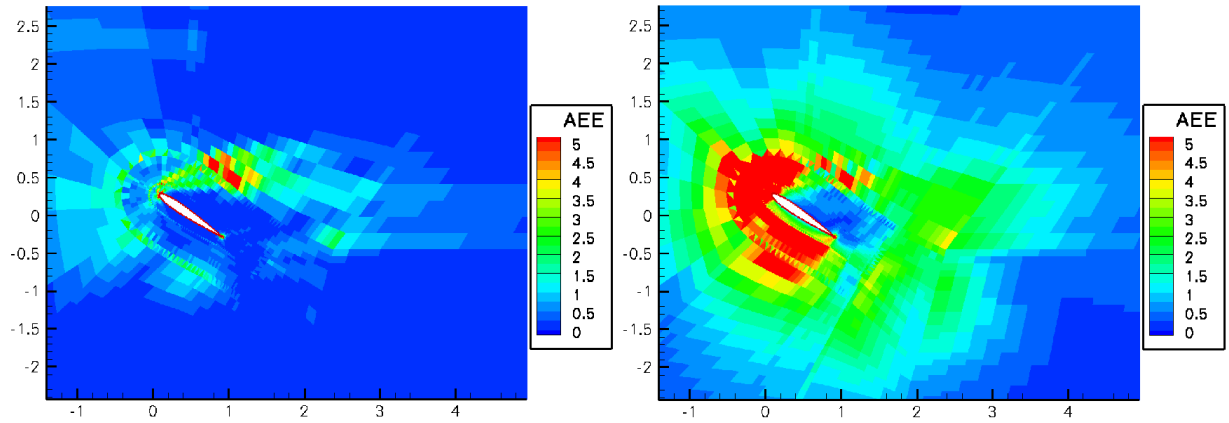


Figure 7.25: A posteriori error on elements for the Newton method without stabilization (left) and by the *semiGLS* method (right), Re = 100

7.2 Applications of BDDC to linear elasticity

This section contains a selection of results obtained by one of the implementations of the BDDC precondtitioner described in Chapter 6. A short description of parallel computers used for these calculations is summarized in Table 7.3 for reference.

hastrman	Compaq Alpha server ES 47,4x Alpha EV7/1000 MHz,OS Tru64	IT CAS, Prague, CR
altix	SGI Altix 3700,32x Intel Itanium,OS Linux	CTU, Prague, CR
rex	SGI Altix 4700,32x Intel Itanium,OS Linux	CTU, Prague, CR
lomond	Sun Fire server E15k,52x UltraSPARC III Cu/1.2 GHz,OS Solaris	EPCC, Edinburgh, UK
HPCx	IBM eServer 575 nodes,2560x IBM POWER5/1.5 GHz,OS AIX	Daresbury, UK
frost	IBM Blue Gene/L,2048x PowerPC-440/700 MHz,OS AIX	NCAR, Boulder, CO

Table 7.3: Parallel computers used for testing

In this section, the model of linear elasticity (5.2)–(5.3) is considered. First, several test problems are computed, followed by computations of several practical problems from mechanical engineering.

7.2.1 Cube problem

The problem of unit cube is a classical problem of domain decomposition methods. In our case, the cube is made of steel – Young's modulus $2.1 \cdot 10^{11}$ Pascal, Poisson's ratio 0.3. It is fixed at one face and loaded by force of 1,000 Newton, acting on one edge opposite to the fixed face in direction parallel to it and pointing outwards of the cube.

Various uniform discretizations of the cube by linear finite elements are used, as well as various divisions into subdomains.

The purpose of the first set of experiments is testing the preconditioner with respect to variable space $\widetilde{W}^{avg}$ (see (5.30), Section 5.2). In presented tables, $\widetilde{W}^{c}$ represents the space of functions continuous across corners, 'edges' means, that equivalence of arithmetic averages across all edges is enforced, and 'faces' stands for equivalence of arithmetic averages across all faces.

We are interested in the quality of preconditioning, represented by number of PCG interates and a condition number estimate [52, Section 6.7.3], but also in the outcome of saving the computational time. Although this is not always the case in domain decomposition literature, we consider this to be the main goal of our efforts. Measurements of time are divided into time of preconditioner setup with factorizations of matrices, and times of the PCG run. The total wall time is also provided. Note, that it contains even some subsidiary operations, that are not included in the previous two rows, so it is not a simple sum of them. All these computation are performed to precision $tol = 10^{-6}$ of the relative residual in the form (6.18).

In Table 7.4, results of this experiment for cube divided into $4^3 = 64$ subdomains and $16^3 = 4,096$ finite elements are presented. Such division results in the ratio of the reference size of subdomain to the reference size of element $H/h = 4$, and in $14,739$ degrees of freedom. The computational mesh and division into subdomains is depicted in Figure 7.26.

In Table 7.5, results for cube divided into 64 subdomains and $32^3 = 32,768$ finite elements are presented. This corresponds to $H/h = 8$ and $107,811$ degrees of freedom. The computational mesh and division into subdomains is depicted in Figure 7.27. Times were obtained by solution of the problem by the implementation based on frontal solver described in Section 6.1 and are presented in seconds.

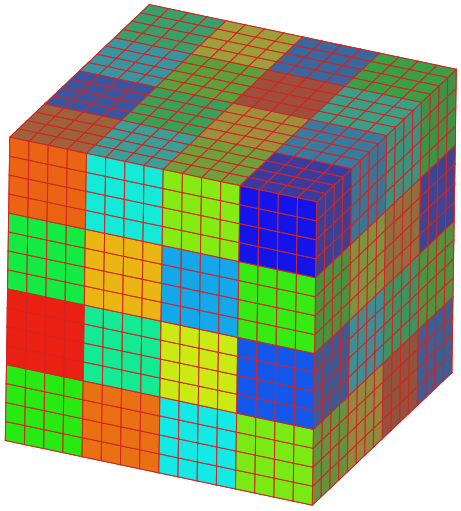


Figure 7.26: Computational mesh for cube problem, 64 subdomains, $H/h = 4$

coarse problem	$\widetilde{W}^c$	$\widetilde{W}^c$ + edges	$\widetilde{W}^c$ + faces	$\widetilde{W}^c$ + edges and faces
iterations	33	17	22	10
cond. number est.	19.1	5.4	9.2	2.3
factorization (sec)	9	9	9	8
PCG iter (sec)	128	68	29	15
total (sec)	140	81	40	26

Table 7.4: Variable $\widetilde{W}^{avg}$ on cube with 64 subdomains, $H/h = 4$, frost – 64 processors, frontal implementation

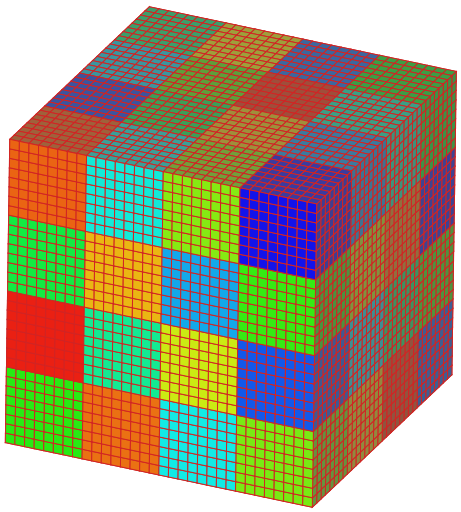


Figure 7.27: Computational mesh for cube problem, 64 subdomains, $H/h = 8$

coarse problem	$\widetilde{W^c}$	$\widetilde{W^c}$ + edges	$\widetilde{W^c}$ + faces	$\widetilde{W^c}$ + edges and faces
iterations	55	20	33	14
cond. number est.	55.1	7.8	27.4	4.2
factorization (sec)	94	157	108	93
PCG iter (sec)	1213	457	225	109
total (sec)	1391	701	420	288

Table 7.5: Variable $\widetilde{W^{avg}}$ on cube with 64 subdomains, $H/h = 8$, frost – 64 processors, frontal implementation

Obtained computational times confirm, that using additional constraints, such as equivalence of arithmetic averages across faces and edges of subdomains, leads to improvement of preconditioning. It also shows, that for the implementation based on frontal solver, decrease in number of iterations can lead to considerable saving of computational time.

The same experiment is now performed with the implementation based on multifrontal solver described in Section 6.2. In the first test, the previous setting is considered, i.e. division into 64 subdomains and $32,768$ finite elements as in Figure 7.27. Note, that in these experiments, the original right hand side f of problem (5.7) is considered, instead of the reduced right hand side of (5.13). This approach saves one extra factorization, that is necessary for finding the right hand side. However, iterations are not performed in the space of functions with minimal energy, and larger number of them is necessary. Results are presented in Table 7.6. Here, an extra column is added – using the MUMPS package for preconditioning of PCG, a natural question arises: how long does the solution by the MUMPS package applied directly to problem (5.7) take. This time is put into the last column of the following tables. Times are in seconds.

A larger experiment was performed on the cube divided into 8 subdomains and $64^3 = 262,144$ elements, $H/h = 32$, resulting in $823,875$ degrees of freedom. Such problem size is closer to the target problem size for such parallel implementations. The computational mesh is depicted in Figure 7.28, and times are presented in Table 7.7.

Next experiment serves for comparison of efficiency of presented approaches to enforcing constraints. Cube of 64 subdomains and $32,768$ finite elements is considered, and equality of averages over all edges and faces is enforced. Results are summarized in Table 7.8, where 'LM' stands for approach using global dual problem for Lagrange multipliers, 'PB' stands for the projected BDDC method described in Section 5.3, and 'PCV' for the generalized change of variables followed by projection described in Section 5.4.

coarse problem	$\widetilde{W^c}$	$\widetilde{W^c}$ + edges	$\widetilde{W^c}$ + faces	$\widetilde{W^c}$ + edges and faces	MUMPS
PCG iterations	103	49	41	24	-
cond. number est.	292.8	76.4	60.5	11.7	-
analysis (sec)	7.5	9.7	26.5	30.9	9.8
factorization (sec)	1.1	1.7	3.2	5.0	25.6
PCG iter (sec)	50.0	23.9	20.7	12.2	-
total (sec)	62.6	47.4	69.8	75.6	39.4

Table 7.6: Variable $\widetilde{W^{avg}}$ on cube with 64 subdomains, $H/h = 8$, frost – 64 processors, multifrontal implementation

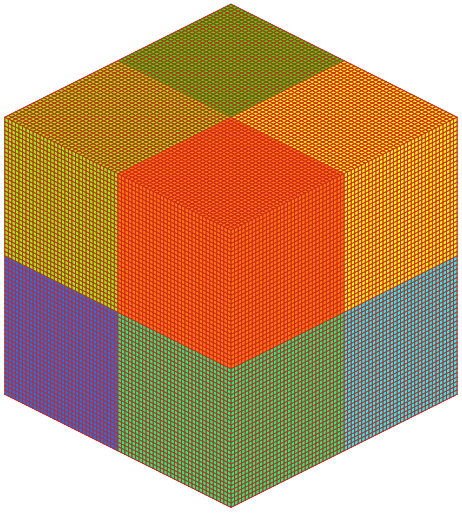


Figure 7.28: Computational mesh for cube problem, 8 subdomains, $H/h = 32$

coarse problem	$\widetilde{W}^c$	$\widetilde{W}^c$ + edges	$\widetilde{W}^c$ + faces	$\widetilde{W}^c$ + edges and faces	MUMPS
PCG iterations	131	75	n/a	n/a	-
cond. number est.	5,941.0	903.1	n/a	n/a	-
analysis (sec)	25.6	23.5	n/a	n/a	27.1
factorization (sec)	1,097.4	1,426.4	n/a	n/a	12,998.0
PCG iter (sec)	743.4	356.2	n/a	n/a	-
total (sec)	1,885.1	1,890.3	n/a	n/a	13,060.6

Table 7.7: Variable $\widetilde{W}^{avg}$ on cube with 8 subdomains, $H/h = 32$, altix – 8 processors, multifrontal implementation

approach	LM	PB	PCV
matrix transformation	-	-	6.5
projection	-	13.6	5.9
analysis	2.9	42.2	12.2
factorization	0.2	41.3	0.6
dual factorization	1,698.2	-	-
PCG iter	316.6	8.4	7.1
total	2,034.9	106.3	33.2

Table 7.8: Comparison of approaches to enforcing constraints – times in seconds

On the largest problem of cube, we can observe several tendencies typical for larger problems. Although the quality of preconditioning is still improved by enforcing additional contraints, this may not lead to savings in total computational time proportionally to the number of iterations. Enforcing additional constraints leads to worse structure of the matrix, which in turn results in more expensive factorization and backsubstitution in problem (5.47) solved by MUMPS in the action of preconditioner. Except of that, corresponding new entries generated by transformation and projection of the matrix in (5.47), or its factors, may not fit into the computer memory, and then we are not able to obtain the solution at all. This behaviour is particularly significant for averages on faces, and it is the case of 'n/a' symbols in Table 7.7.

Presumably, the MUMPS package itself applied to the original problem performs better than BDDC for small problems, while it is less efficient for larger problems.

7.2.2 Steam turbine entry nozzle

The implementation was applied to the analysis of the nozzle box of a ŠKODA steam turbine 28 MW, loaded by steam pressure and temperature.

The body of the turbine nozzle box was divided into $2,696$ isoparametric quadratic finite elements resulting in $40,254$ degrees of freedom. The problem and the computational mesh were provided by Jaroslav Novotný, Institute of Thermomechanics of the Czech Academy of Sciences, and Jan Leština, Vamet Ltd.

Division into 16 subdomains is depicted in Figure 7.29. It was obtained by the METIS package [34]. Such division leads to 37 corners, 14 edges, and 30 faces.

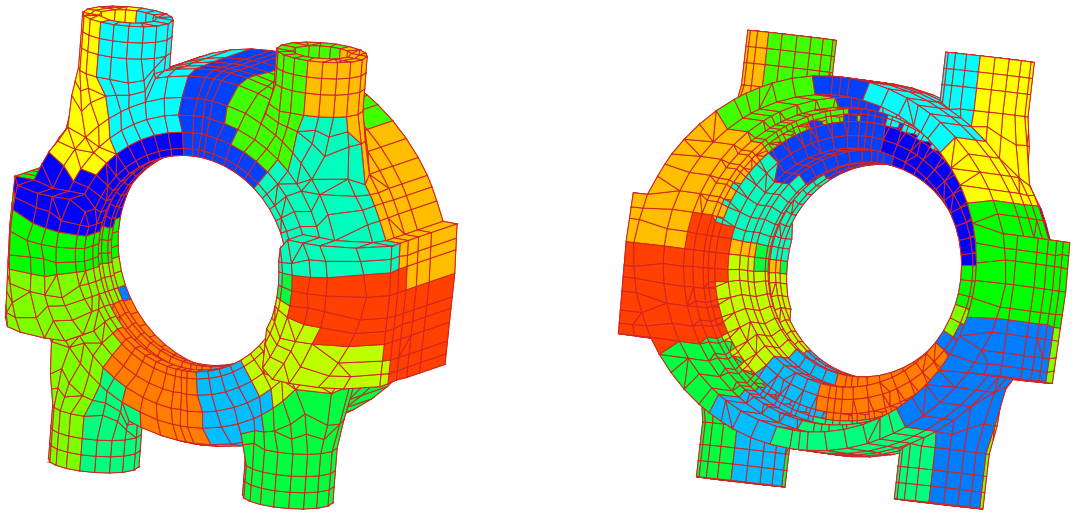


Figure 7.29: Mesh for turbine nozzle, 16 subdomains

Scaling results for variable number of processors of rex computer are presented, obtained by implementation based on frontal solver of Section 6.1. In Table 7.9, only continuity across corners is assumed, without any additional constraints. Continuity of averages across all edges and faces is considered for all computations in Table 7.10.

These results show a reasonable scalability of the implementation, even on an industrial application.

In Table 7.11, the experiment with variable $\widetilde{W}^{avg}$ is presented.

On this problem, we can observe, that the division into subdomains affects the preconditioner through forming the topology of interface. Edge averages are clearly not sufficient for a good preconditioning for this case, while adding face averages considerably improves the preconditioner.

processors	1	2	4	8	16
factorization (sec)	5.2	2.7	1.4	0.8	0.5
PCG iter (sec)	41.2	21.8	11.1	6.3	4.8
total (sec)	48.3	25.5	13.0	7.4	5.5

Table 7.9: Scaling results for turbine nozzle with 16 subdomains, continuity at corners, rex

processors	1	2	4	8	16
factorization (sec)	7.1	3.6	1.9	1.2	0.7
PCG iter (sec)	16.7	9.2	4.7	2.5	1.8
total (sec)	25.7	13.8	7.1	4.0	2.6

Table 7.10: Scaling results for turbine nozzle with 16 subdomains, averages on edges and faces, rex

coarse problem	$\widetilde{W}^c$	$\widetilde{W}^c$ + edges	$\widetilde{W}^c$ + faces	$\widetilde{W}^c$ + edges and faces
PCG iterations	99	78	41	36
cond. number est.	2,933	1,546	164	142
factorization (sec)	0.5	0.6	0.8	0.7
PCG iter (sec)	4.8	3.8	3.1	1.8
total (sec)	5.5	4.6	4.4	2.6

Table 7.11: Variable $\widetilde{W}^{avg}$ on turbine nozzle with 16 subdomains, rex – 16 processors

Improvement of the preconditioner by additional constraints leads to markable savings of time also for this problem.

7.2.3 Shaft with a groove

In this problem, a simplified model of a drilling rod is analyzed. Stress and maximal deflection are investigated to observe the influence of a spiral groove on the mechanics of the drilling rod.

The computational mesh consists of 483,400 linear elements, 505,312 nodes, and 1,515,936 degrees of freedom. It is presented in Figure 7.30. Computation of this problem took 1,470 minutes by serial frontal algorithm on computer 'hastrman'. The problem and the computational mesh were provided by Svatopluk Pták, Institute of Thermomechanics of the Czech Academy of Sciences.

The mesh was initially divided into 512 subdomains, and 10,000 corners were selected. This division is depicted in Figure 7.31.

Scaling results obtained on variable number of processors of HPCx cluster are presented in Table 7.12. These results were computed by an early version of the implementation based on frontal solver (Section 6.1). Note, that no average constraints were allowed in that version and the quality of the preconditioner was controled by number of corners. Also, the coarse problem was solved in serial manner on the master processor. This clearly became the bottleneck of the setup due to its huge size compared to subdomain problems. Only total times in minutes are presented.

Later, the mesh was divided into 16 subdomains, and only 100 corners were selected. The division is depicted in Figure 7.32. This problem was used for comparison of the two implementations desribed in Section 6.1 and Section 6.2, respectively. Results are presented in seconds in Table 7.13.

This comparison reflects all the important properties of both implementations. Implementation, that uses condensed right hand side requires lower number of iterations. However, the approach of the frontal solver, that uses the out-of-core processing, is expensive with respect to time, which amounts to much longer time of factorization, but also slightly affects each iteration. Another source of longer factorization time is the need of two factorizations of the interior unknows. On the other hand, the out-of-core approach would be the only possible way, if we wanted to solve even larger problems, because they would not fit into the computer memory. This would be the limitation of the implementation based on multifrontal solver.

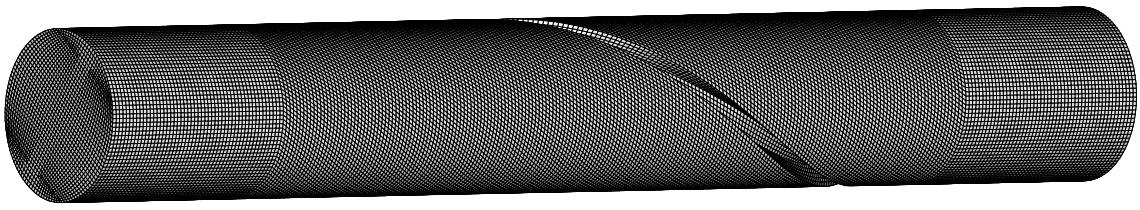


Figure 7.30: Mesh for shaft with a groove

processors	16	32	64
total (minutes)	130	87	72

Table 7.12: Scaling results for shaft with 512 subdomains, continuity at corners, HPCx

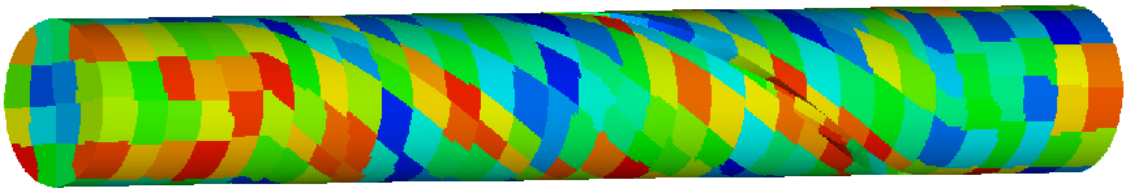


Figure 7.31: Division of the mesh for the shaft into 512 subdomains

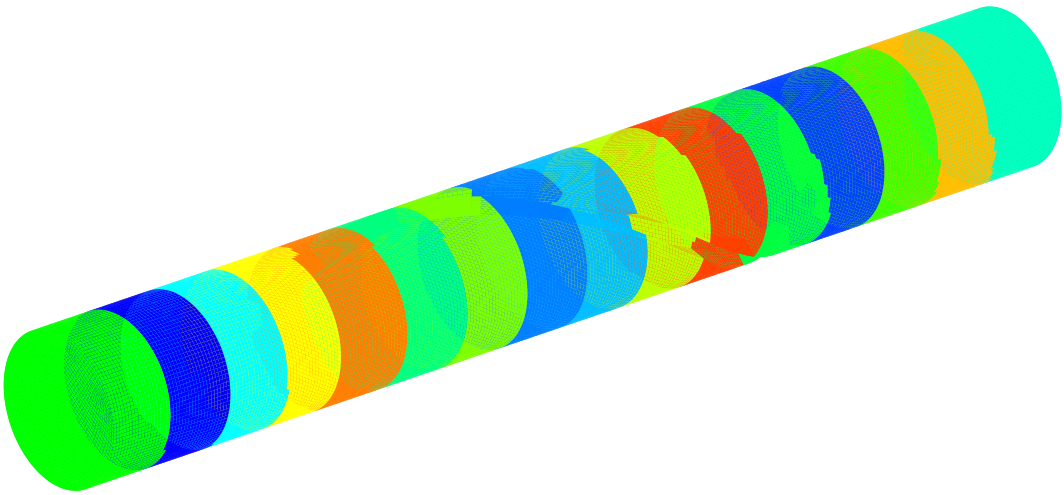


Figure 7.32: Division of the mesh for the shaft into 16 subdomains

implementation	Section 6.1	Section 6.2
PCG iterations	80	256
cond. number est.	335.9	5494.0
analysis (sec)	-	58.2
factorization (sec)	7041.1	82.8
PCG iter (sec)	1433.5	1101.1
total (sec)	9778.9	1264.3

Table 7.13: Comparison of implementation based on frontal solver (left) and multifrontal solver (right), rex – 16 processors

7.3 Numerical experiments with BDDC for steady Stokes flow

The applicability of the preconditioner to the problem of Stokes flow was tested, and results are presented in this section. The system matrix of the Stokes problem is symmetric, but indefinite (cf. e.g. [16, Section 5.5, Section 6]). Note, that for this reason, the theory of Section 5.2 does not cover this case.

However, it was observed, that the BDDC method leads to an indefinite preconditioner, which is able to make the preconditioned operator positive definite, and thus allow the use of PCG method.

An alternative way to assure positive definiteness of the preconditioned operator based on BDDC was presented in [41].

7.3.1 Lid driven cavity

Problem of lid driven cavity was introduced in Section 7.1.1. It was solved using the Navier-Stokes model therein. In this section, we investigate the steady Stokes flow in this domain.

The case of uniform mesh of 128×128 quadratic square elements was chosen. It was divided into 8 subdomains by METIS package (Figure 7.33). We selected 9 corners.

Resulting streamlines and plot of pressure for Reynolds number $10,000$ are presented in Figure 7.34. Streamlines are symmetric for the Stokes problem, unlike in the Navier-Stokes case presented in Section 7.1.1.

Solution of the problem by a serial frontal algorithm took 231 seconds on one processor of the rex computer, compared to 17.2 seconds on 8 processors of the same computer necessary for the solution by BDDC. Due to the corner singularities in pressure, the prescribed accuracy was lowered to $tol = 10^{-3}$ in this case, resulting in 59 PCG iterations.

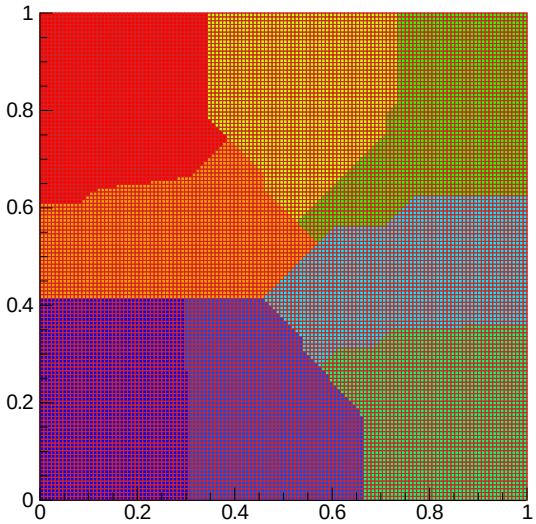


Figure 7.33: Mesh and its division into 8 subdomains for lid driven cavity, 128×128 elements

7.3.2 Channel with sudden extension of diameter

Another geometry introduced already in Section 7.1.2 for the case of Navier-Stokes flow, is the channel with abrupt extension of diameter.

It was divided into 4 subdomains by METIS package (Figure 7.35). We selected 5 corners.

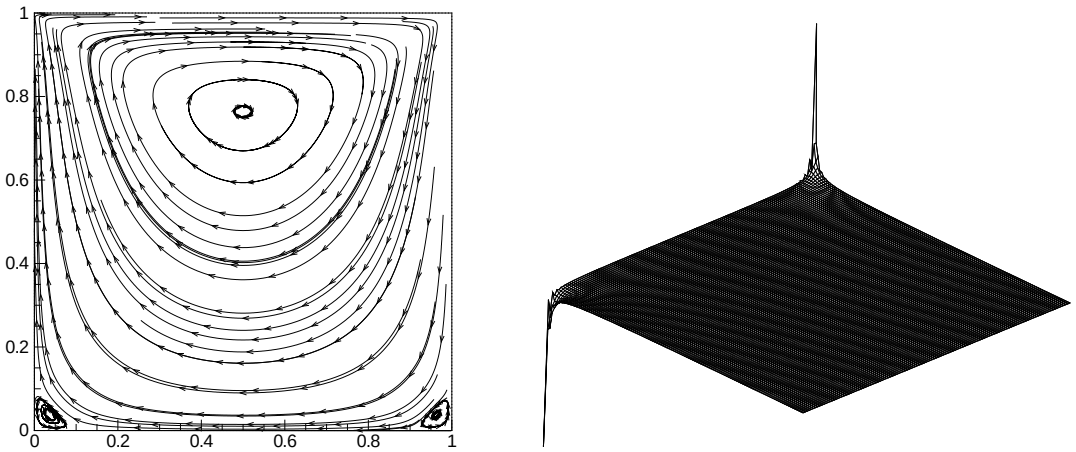


Figure 7.34: Streamlines (left) and pressure (right) for lid driven cavity, 128×128 elements

Resulting streamlines and plot of pressure for Reynolds number $2,000$ are presented in Figures 7.36 and 7.37, respectively. Due to the corner singularities in pressure, the prescribed accuracy was lowered to $tol = 10^{-3}$ in this case, resulting in 26 PCG iterations. Note, that fluid flows from right to left in the plot of pressure in order to show the situation at corners. However, this problem is too small to meaningful comparisons of time.

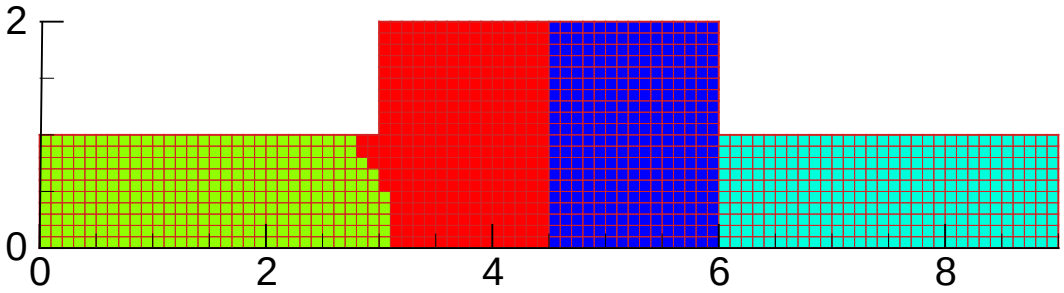


Figure 7.35: Mesh and its division into 4 subdomains for channel with sudden extension of diameter

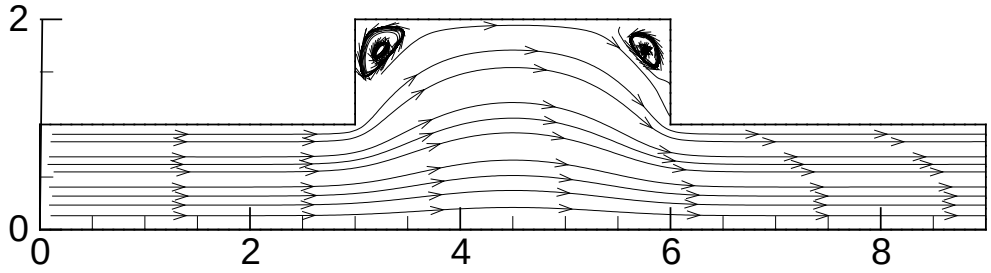


Figure 7.36: Streamlines for channel with sudden extension of diameter, Re = 2,000

7.3.3 Channel with sudden reduction of diameter

In the last example, a geometry with a sudden reduction of diameter is considered. Flow in this geometry described by the Navier-Stokes model was studied in [7], with respect to precise solution

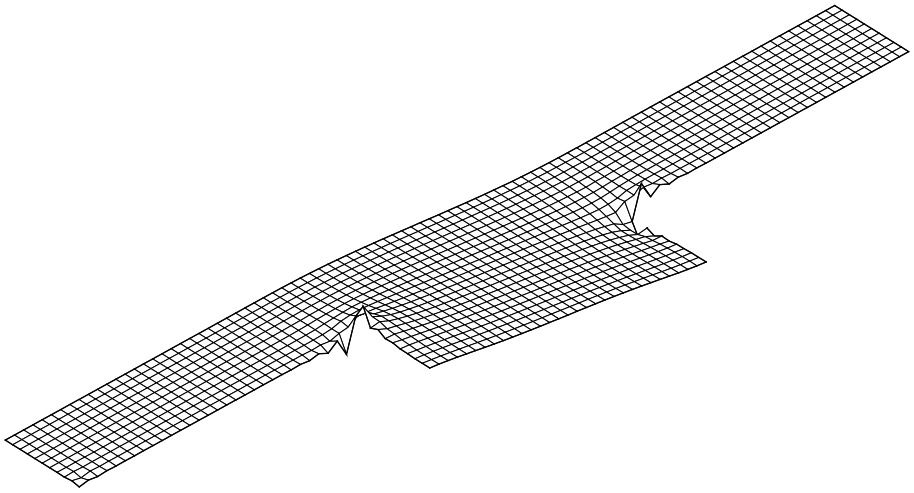


Figure 7.37: Pressure for channel with sudden extension of diameter, Re = 2,000

of corner singularities. The channel geometry is presented in Figure 7.38.

Due to the symmetry of the channel, only the upper part is considered in the computation. Division into 4 subdomains obtained by METIS is presented in Figure 7.39.

Solution obtained by BDDC method at Reynolds number 250 for the Stokes flow is presented in Figures 7.40 and 7.41. Due to the corner singularities in pressure, the prescribed accuracy was lowered to $tol = 10^{-3}$ also for this case, which resulted in 59 PCG iterations. Note, that fluid flows from right to left in the plot of pressure in order to show the situation at corners. Again, comparisons of time are not relevant for such a small problem.

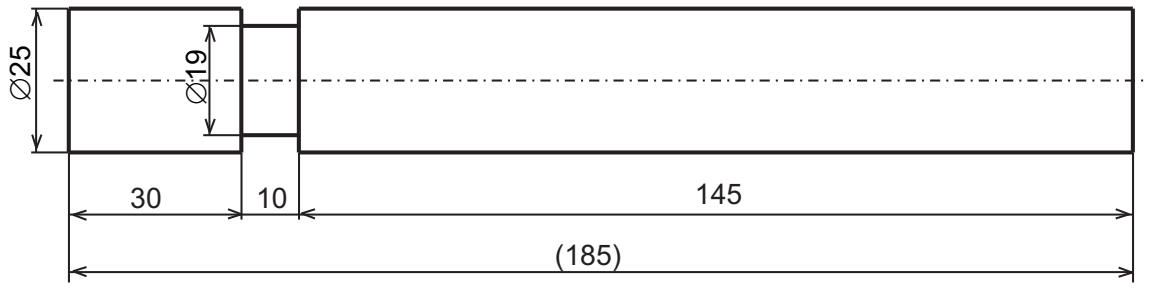


Figure 7.38: Geometry of the channel with abrupt reduction of diameter

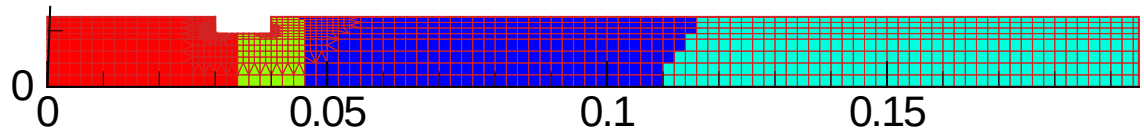


Figure 7.39: Mesh and its division into 4 subdomains for channel with sudden reduction of diameter

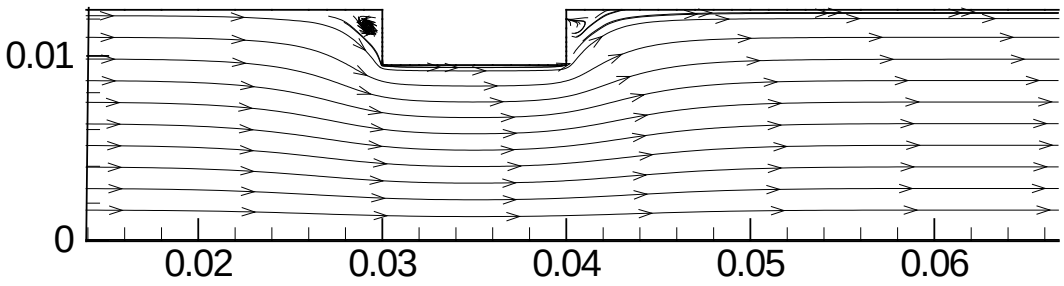


Figure 7.40: Detail of streamlines for channel with sudden reduction of diameter, Re = 250

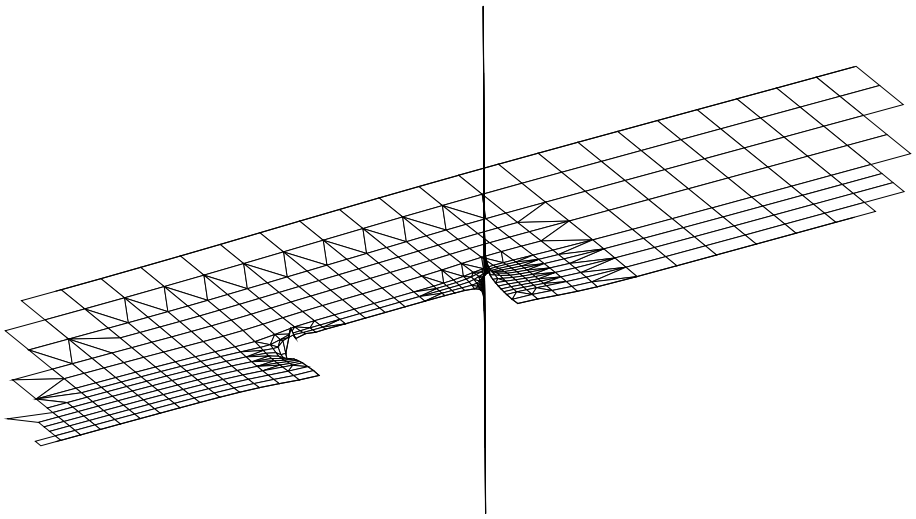


Figure 7.41: Detail of pressure for channel with sudden reduction of diameter, Re = 250

Chapter 8

Conclusion

Despite its importance, analysis of incompressible flow fields in various problems of mechanical engineering is still not completely resolved. This includes the practically important case of flows at high Reynolds numbers. A stabilization represents a way to improve the applicability of the finite element method to larger set of problems of fluid mechanics. Domain decomposition presents a mathematical method suitable for parallel computers, allowing the solution of very large problems in acceptable times. It is an important goal of computational fluid dynamics to combine both methods in new solvers.

In the presented thesis, stabilized finite element method for solving incompressible flows is discussed, together with applications of nonoverlapping domain decomposition methods. The theoretical introduction to models of incompressible flows and to the finite element method for flow problems is presented in Chapters 2 and 3, respectively. In Chapter 4, a new stabilization technique is derived and analyzed. The aspect of accuracy of stabilized FEM is important, but unclear. In an attempt to answer this issue, two methods for evaluation of the accuracy are proposed.

The Balancing Domain Decomposition by Constraints (BDDC) [13] is recalled in Chapter 5 and several reformulations and improvements are introduced. In Chapter 6, two algorithms of the preconditioner suitable for massively parallel computers are presented, including various implementational details. Numerical results are summarized in Chapter 7. Applications of the semiGLS method to steady and unsteady Navier-Stokes flows are presented in Section 7.1. The BDDC method has been tested on problems of linear elasticity (Section 7.2) (which leads to positive definite matrices) and also successfully applied to the Stokes problem (Section 7.3) (which leads to indefinite matrices).

My main original results and achievements are:

- Derivation of *semiGLS*, a new algorithm for stabilization of the FEM for solving incompressible viscous flows. This algorithm is derived as a modification of the earlier *GLS* algorithm from [32]. The *semiGLS* was also presented in journal paper [8].

- Proposing two ways for evaluating the accuracy of the stabilized method: the first based on comparison of discrete solutions with and without stabilization, the second on a posteriori error estimates for the Navier-Stokes equations introduced in [6]. This helps to control the error we introduce to the solution by stabilization, which is crucial for its successful application. This work was also presented in journal papers [7, 8, 9, 10].

- A reformulation of the BDDC algorithm motivated by implementation. Identification and definition of the second intermediate space described in Section 5.2, formulation of projected BDDC described in Section 5.3, and generalization of the change of variables from [42] for

efficient enforcing of additional constraints on averages in BDDC (Section 5.4). This work reflects joint research with Prof. Jan Mandel and Bedřich Sousedík from the Department of Mathematical and Statistical Sciences, University of Colorado Denver, and it is presented in this thesis for the first time.

- Design of two parallel algorithms of the BDDC method and development of corresponding programs. The first proposed algorithm is based on standard tools of finite element software and requires minimal amount of custom coding. It was recently presented in [54]. The second algorithm is more experimental, but easily extensible and was developed mainly to test new formulations of the BDDC method, including projected BDDC and generalized change of variables introduced in this thesis, directly in parallel environment. It was also extended to solutions of the Stokes flow.

- Developed programs were not only carefully tested on many benchmarks problems, but also brought new results in technical applications, when successfully solving several large time-demanding industrial problems.

While main goals of the dissertation were fulfilled, the field offers many new directions for further research. Study and testing of other stabilization techniques still presents an interesting topic. Application of the BDDC method to other types of problems (indefinite and unsymmetric) is desired, although not obvious, and the full connection of the BDDC method and flow described by the Navier-Stokes equations, possibly with stabilization, is still an open problem, which I would like to work on. Extensions of the BDDC implementation to adaptive selection of constraints [47] and multilevel approach [49] are other interesting goals of my research for near future.

Bibliography

[1] AMESTOY, P. R., DUFF, I. S., AND L'EXCELLENT, J.-Y. Multifrontal parallel distributed symmetric and unsymmetric solvers. *Comput. Methods Appl. Mech. Engrg. 184* (2000), 501–520.

[2] BRENNER, S. C. The condition number of the Schur complement in domain decomposition. *Numer. Math. 83*, 2 (1999), 187–203.

[3] BRENNER, S. C., AND SCOTT, L. R. *The mathematical theory of finite element methods*, second ed., vol. 15 of *Texts in Applied Mathematics*. Springer-Verlag, New York, 2002.

[4] BRENNER, S. C., AND SUNG, L.-Y. BDDC and FETI-DP without matrices or vectors. *Comput. Methods Appl. Mech. Engrg. 196*, 8 (2007), 1429–1435.

[5] BREZZI, F., AND FORTIN, M. *Mixed and hybrid finite element methods*, vol. 15 of *Springer Series in Computational Mathematics*. Springer-Verlag, New York, 1991.

[6] BURDA, P., NOVOTNÝ, J., AND SOUSEDÍK, B. A posteriori error estimates applied to flow in a channel with corners. *Math. Comput. Simulation 61*, 3-6 (2003), 375–383.

[7] BURDA, P., NOVOTNÝ, J., AND ŠÍSTEK, J. Precise FEM solution of a corner singularity using an adjusted mesh. *Internat. J. Numer. Methods Fluids 47*, 10–11 (2005), 1285–1292.

[8] BURDA, P., NOVOTNÝ, J., AND ŠÍSTEK, J. On a modification of GLS stabilized FEM for solving incompressible viscous flows. *Internat. J. Numer. Methods Fluids 51*, 9–10 (2006), 1001–1016.

[9] BURDA, P., NOVOTNÝ, J., AND ŠÍSTEK, J. Numerical solution of flow problems by stabilized finite element method and verification of its accuracy using a posteriori error estimates. *Math. Comp. Simul. 76*, 1–3 (2007), 28–33.

[10] BURDA, P., NOVOTNÝ, J., AND ŠÍSTEK, J. Accuracy of SemiGLS stabilization of FEM for solving Navier–Stokes equations and a posteriori error estimates. *Internat. J. Numer. Methods Fluids 56*, 8 (2008), 1167–1173.

[11] CHORIN, A. J., AND MARSDEN, J. E. *A mathematical introduction to fluid mechanics*. Springer-Verlag, New York, 1979.

[12] CONCUS, P., GOLUB, G. H., AND O'LEARY, D. P. A generalized conjugate gradient method for the numerical solution of elliptic PDE. In *Sparse Matrix Computations*, J. R. Bunch and D. J. Rose, Eds. Academic Press, New York, 1976, pp. 309–332.

[13] DOHRMANN, C. R. A preconditioner for substructuring based on constrained energy minimization. *SIAM J. Sci. Comput. 25*, 1 (2003), 246–258.

[14] DOUGLAS, JR., J., AND WANG, J. P. An absolutely stabilized finite element method for the Stokes problem. *Math. Comp. 52*, 186 (1989), 495–508.

[15] DUFF, I. S., AND REID, J. K. The multifrontal solution of indefinite sparse symmetric linear equations. *ACM Trans. Math. Software 9*, 3 (1983), 302–325.

[16] ELMAN, H. C., SILVESTER, D. J., AND WATHEN, A. J. *Finite elements and fast iterative solvers: with applications in incompressible fluid dynamics*. Numerical Mathematics and Scientific Computation. Oxford University Press, New York, 2005.

[17] FARHAT, C., LESOINNE, M., LE TALLEC, P., PIERSON, K., AND RIXEN, D. J. FETI-DP: a dual-primal unified FETI method. I. A faster alternative to the two-level FETI method. *Internat. J. Numer. Methods Engrg. 50*, 7 (2001), 1523–1544.

[18] FARHAT, C., LESOINNE, M., AND PIERSON, K. A scalable dual-primal domain decomposition method. *Numer. Linear Algebra Appl. 7* (2000), 687–714. Preconditioning techniques for large sparse matrix problems in industrial applications (Minneapolis, MN, 1999).

[19] FARHAT, C., AND ROUX, F.-X. A method of finite element tearing and interconnecting and its parallel solution algorithm. *Internat. J. Numer. Methods Engrg. 32* (1991), 1205–1227.

[20] FRANCA, L. P., AND FREY, S. L. Stabilized finite element methods. II. The incompressible Navier-Stokes equations. *Comput. Methods Appl. Mech. Engrg. 99*, 2-3 (1992), 209–233.

[21] FRANCA, L. P., AND HUGHES, T. J. R. Convergence analyses of Galerkin least-squares methods for symmetric advective-diffusive forms of the Stokes and incompressible Navier-Stokes equations. *Comput. Methods Appl. Mech. Engrg. 105*, 2 (1993), 285–298.

[22] FRANCA, L. P., JOHN, V., MATTHIES, G., AND TOBISKA, L. An inf-sup stable and residual-free bubble element for the Oseen equations. *SIAM J. Numer. Anal. 45*, 6 (2007), 2392–2407 (electronic).

[23] FRANCA, L. P., AND MADUREIRA, A. L. Element diameter free stability parameters for stabilized methods applied to fluids. *Comput. Methods Appl. Mech. Engrg. 105*, 3 (1993), 395–403.

[24] GELHARD, T., LUBE, G., OLSHANSKII, M. A., AND STARCKE, J.-H. Stabilized finite element schemes with LBB-stable elements for incompressible flows. *J. Comput. Appl. Math. 177*, 2 (2005), 243–267.

[25] GIRAULT, V., AND RAVIART, P.-A. *Finite element methods for Navier-Stokes equations*, vol. 5 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, 1986. Theory and algorithms.

[26] GLOWINSKI, R. Finite element methods for incompressible viscous flow. In *Handbook of numerical analysis, Vol. IX*, Handb. Numer. Anal., IX. North-Holland, Amsterdam, 2003, pp. 3–1176.

[27] GREENBAUM, A. *Iterative methods for solving linear systems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1997.

[28] GRESHO, P. M., AND SANI, R. L. *Incompressible flow and the finite element method*. John Wiley & Sons Ltd., Chichester, 2000.

[29] GUERMOND, J.-L., AND QUARTAPELLE, L. Calculation of incompressible viscous flows by an unconditionally stable projection FEM. *J. Comput. Phys. 132*, 1 (1997), 12–33.

[30] HUGHES, T. J. R. Multiscale phenomena: Green's functions, the Dirichlet-to-Neumann formulation, subgrid scale models, bubbles and the origins of stabilized methods. *Comput. Methods Appl. Mech. Engrg. 127*, 1-4 (1995), 387–401.

[31] HUGHES, T. J. R., FRANCA, L. P., AND BALESTRA, M. Errata: "A new finite element formulation for computational fluid dynamics. V. Circumventing the Babuška-Brezzi condition: a stable Petrov-Galerkin formulation of the Stokes problem accommodating equal-order interpolations". *Comput. Methods Appl. Mech. Engrg. 62*, 1 (1987), 111.

[32] HUGHES, T. J. R., FRANCA, L. P., AND HULBERT, G. M. A new finite element formulation for computational fluid dynamics. VIII. The Galerkin/least-squares method for advective-diffusive equations. *Comput. Methods Appl. Mech. Engrg. 73*, 2 (1989), 173–189.

[33] IRONS, B. M. A frontal solution scheme for finite element analysis. *Internat. J. Numer. Methods Engrg. 2* (1970), 5–32.

[34] KARYPIS, G., AND KUMAR, V. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM J. Sci. Comput. 20*, 1 (1998), 359–392 (electronic).

[35] KLAWONN, A., AND WIDLUND, O. B. Selecting constraints in dual-primal FETI methods for elasticity in three dimensions. In *Domain Decomposition Methods in Science and Engineering*, R. Kornhuber, R. Hoppe, J. Périaux, O. Pironneau, O. Widlund, and J. Xu, Eds., Lecture Notes in Computational Science and Engineering. Springer, 2004, pp. 67–81. Proceedings of the 15th International Conference on Domain Decomposition Methods in Berlin, Germany, 2003.

[36] KLAWONN, A., AND WIDLUND, O. B. Dual-primal FETI methods for linear elasticity. *Comm. Pure Appl. Math. 59*, 11 (2006), 1523–1572.

[37] KLAWONN, A., WIDLUND, O. B., AND DRYJA, M. Dual-primal FETI methods for three-dimensional elliptic problems with heterogeneous coefficients. *SIAM J. Numer. Anal. 40*, 1 (2002), 159–179.

[38] KOLÁŘ, V., KRATOCHVÍL, J., ŽENÍŠEK, A., AND ZLÁMAL, M. Technical, physical, and mathematical principles of the finite element method. In *Rozpravy ČSAV, 81, 2*. Academia, Praha, 1971.

[39] KRUIS, J. *Domain decomposition methods for distributed computing*. Saxe-Coburg Publications, Kippen, Stirling, Scotland, 2006. ISBN-13: 978-1-874672-23-4, ISBN-10: 1-874672-23-7.

[40] LESOINNE, M. A FETI-DP corner selection algorithm for three-dimensional problems. In *Domain Decomposition Methods in Science and Engineering*, I. Herrera, D. E. Keyes, and O. B. Widlund, Eds. National Autonomous University of Mexico (UNAM), México, 2003, pp. 217–223. 14th International Conference on Domain Decomposition Methods, Cocoyoc, Mexico, January 6–12, 2002. http://www.ddm.org.

[41] LI, J., AND WIDLUND, O. B. BDDC algorithms for incompressible Stokes equations. *SIAM J. Numer. Anal. 44*, 6 (2006), 2432–2455 (electronic).

[42] LI, J., AND WIDLUND, O. B. FETI-DP, BDDC, and block Cholesky methods. *Internat. J. Numer. Methods Engrg. 66*, 2 (2006), 250–271.

[43] MANDEL, J. Balancing domain decomposition. *Comm. Numer. Methods Engrg. 9*, 3 (1993), 233–241.

[44] MANDEL, J., AND BREZINA, M. Balancing domain decomposition for problems with large jumps in coefficients. *Mathematics of Computation 65*, 216 (1996), 1387–1401.

[45] MANDEL, J., AND DOHRMANN, C. R. Convergence of a balancing domain decomposition by constraints and energy minimization. *Numer. Linear Algebra Appl. 10*, 7 (2003), 639–659.

[46] MANDEL, J., DOHRMANN, C. R., AND TEZAUR, R. An algebraic theory for primal and dual substructuring methods by constraints. *Appl. Numer. Math. 54*, 2 (2005), 167–193.

[47] MANDEL, J., AND SOUSEDÍK, B. Adaptive selection of face coarse degrees of freedom in the BDDC and the FETI-DP iterative substructuring methods. *Comput. Methods Appl. Mech. Engrg. 196*, 8 (2007), 1389–1399.

[48] MANDEL, J., AND SOUSEDÍK, B. BDDC and FETI-DP under minimalist assumptions. *Computing 81* (2007), 269–280.

[49] MANDEL, J., SOUSEDÍK, B., AND DOHRMANN, C. R. Multispace and Multilevel BDDC. arXiv:0712.3977, 2007.

[50] MANDEL, J., AND TEZAUR, R. On the convergence of a dual-primal substructuring method. *Numer. Math. 88* (2001), 543–558.

[51] QUARTERONI, A., AND VALLI, A. *Domain decomposition methods for partial differential equations*. Numerical Mathematics and Scientific Computation. Oxford University Press, New York, 1999. Oxford Science Publications.

[52] SAAD, Y. *Iterative methods for sparse linear systems*, second ed. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2003.

[53] ŠÍSTEK, J. Stabilization of finite element method for solving incompressible viscous flows, 2004. Master thesis.

[54] ŠÍSTEK, J., NOVOTNÝ, J., MANDEL, J., ČERTÍKOVÁ, M., AND BURDA, P. BDDC by a frontal solver and the stress computation in a hip joint replacement. arXiv:0802.4295, 2008.

[55] SMITH, B. F., BJØRSTAD, P. E., AND GROPP, W. D. *Domain decomposition*. Cambridge University Press, Cambridge, 1996. Parallel multilevel methods for elliptic partial differential equations.

[56] TEMAM, R. *Navier-Stokes equations*, third ed., vol. 2 of *Studies in Mathematics and its Applications*. North-Holland Publishing Co., Amsterdam, 1984. Theory and numerical analysis, With an appendix by F. Thomasset.

[57] TEZDUYAR, T., AND SATHE, S. Stabilization parameters in SUPG and PSPG formulations. *J. Comput. Appl. Mech. 4*, 1 (2003), 71–88 (electronic).

[58] TOSELLI, A., AND WIDLUND, O. *Domain decomposition methods—algorithms and theory*, vol. 34 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, 2005.

[59] TUREK, S. *Efficient solvers for incompressible flow problems*, vol. 6 of *Lecture Notes in Computational Science and Engineering*. Springer-Verlag, Berlin, 1999. An algorithmic and computational approach, With 1 CD-ROM ("Virtual Album": UNIX/LINUX, Windows, POWERMAC; "FEATFLOW 1.1": UNIX/LINUX).