Harald Carlo Hendrik KOLK

Adaptive Finite Elemente Methode und Finite Volumen Methode zur Lösung der Navier-Stokes Gleichungen

MASTERARBEIT

zur Erlangung des akademischen Grades
eines Diplom-Ingenieur

Masterstudium Technomathematik

Graz University of Technology

Technische Universität Graz

Betreuer:
Prof. Dr. O. Steinbach

Institut für Numerische Mathematik

Graz, im Juni 2014

Vorwort

Die vorliegende Masterarbeit zum Thema "Adaptive Finite Elemente Methode und Finite Volumen Methode zur Lösung der Navier-Stokes Gleichungen" wurde im Rahmen meines Studiums der Technischen Mathematik an der Technischen Universität Graz verfasst.
Ich möchte mich an dieser Stelle herzlich bei Herrn Prof. Steinbach und bei Herrn Dr. Offner, der die AVL List GmbH vertritt, für die Bereitstellung des Themas bedanken. Stets konnte ich mich auf erkenntnisbringende Unterstützung durch Prof. Steinbach verlassen, und das trotz seines ausgefüllten Terminkalenders. Dr. Offner möchte ich dafür danken, dass mir durch seine vermittelnde Tätigkeit zwischen Industrie und Universität meine Masterarbeit in dieser Form möglich wurde und die enge Zusammenarbeit gelingen konnte.
Weiters möchte ich mich bei Herrn Dr. Emans bedanken, der mir seitens der AVL in sämtlichen Fragen bezüglich AVL-Fire, CFD und Implementierung als fachlicher Betreuer zur Seite stand. Mein Dank gilt auch Herrn Dipl.-Ing. John, in dessen Finite Elemente Sourcecode ich meinen Verfeinerungsalgorithmus und meine Fehlerschätzungsmethode einbauen durfte, sowie für seine Unterstützung zur Implementierung.
Insbesondere bedanke ich mich bei meiner Familie und meinen Freunden, auf deren Unterstützung ich während meines gesamten Studiums zählen durfte.

Abstract

Das Ziel dieser Arbeit ist es, zu einer Fehlerschätzungsmethode für adaptive Netzverfeinerung für eine Finite Volumen Methode (FVM) zu gelangen, wodurch Bereiche des Berechnungsgebietes bestimmt werden sollen, in denen eine Verfeinerung des Gitters in größtmöglichem Maße zur Reduzierung des Berechnungsfehlers beiträgt. Die Gleichungen, für die die Methode entwickelt wird, sind die Navier-Stokes Gleichungen für inkompressible Strömungen. Es wird darauf Wert gelegt, dass die Fehlerschätzer in der Praxis relevante "Problembereiche" im Berechnungsgebiet erkennen können, um dort eine Netzverfeinerung zu induzieren. Dazu werden Fehlerschätzer betrachtet, die für die Finite Elemente Methode (FEM) entwickelt und anschließend an eine FVM angepasst wurden, sowie auch Fehlerschätzer, die nur für FVM entwickelt wurden. Insbesondere wurde der Residuenschätzer für FEM in `C++` und für FVM in AVL-Fire implementiert. Abschließend wurden numerische Beispiele mit beiden Implementierungen gerechnet und die Ergebnisse verglichen und analysiert.

The goal of the present work is to find an error estimation method for adaptive mesh refinement for a Finite Volume Method (FVM) which allows us to detect regions of the computational domain where refinement of the mesh seems most promising in terms of error reduction. The equations for which the method is being developed are the time independent Navier-Stokes equations for incompressible flows. An emphasis is placed on error estimators that are capable of detecting practically relevant "problem regions" in the computational domain, in order to induce refinement in those regions. For that purpose error estimators are being considered which have been developed for the Finite Element Method (FEM) and have been adopted to FVM, as well as error estimators which have solely been developed for FVM. Especially the residual error estimator has been implemented for FEM in `C++` and for FVM in AVL-Fire. Finally, numerical examples have been calculated for both implementations and the results compared and analyzed.

Contents

1 Introduction

The aim of this work is to investigate a selection of practically relevant a posteriori error estimation techniques applicable to fluid flow computations, for finite element as well as for finite volume discretizations. Numerical experiments were conducted in order to test the performance of the error estimators. The numerical investigation for finite elements was done using a `C++`-implementation, and for the finite volume method, the commercial code *AVL-Fire*® was used. In order to keep things simple while the main focus lies on the error estimation procedures, only the incompressible Navier-Stokes equations for steady-state calculations are being considered.
The goal when using adaptive mesh refinement is to increase the accuracy of the simulation, i.e. to reduce the approximation error in an effective way, while at the same time keeping the computational effort at a minimum. We do this by increasing the number of control volumina in the computational mesh, also called mesh refinement or h-refinement (in contrary to p-refinement, where the polynomial order of interpolation is increased locally or globally), but only in regions of the mesh where the error is large. In order to identify such regions, an error estimator is required. Desirable properties of such error estimators are reliability and efficiency, which states that the estimator is a meaningful approximation to the real error.
In Chapter 2, an accurate description of the requirements on the computational domain Ω and its decomposition into finite elements is given. The finite volume and finite element methods are both applicable to two and three space dimensions, but for simplicity, the considerations on the finite element method are restricted to two space dimensions and triangles as finite elements.
In Chapter 3 the finite volume method implemented in *AVL-Fire*® to solve the Navier-Stokes equations will be discussed, together with all the other numerical aspects needed to discretize the partial differential equations under consideration. The method allows unstructured grids composed of arbitrary polyhedra. It is based mainly on Ferziger and Perić [19] and Patankar [30]. The starting point will be the modelling equations in integral form, from where the finite volume discretization will be developed. Then methods for the approximation of derivatives will be discussed and the final form of the algebraic equations will be given. Finally, the solution procedure using a SIMPLE-algorithm based approach for the coupling of the velocities and the pressure is being described, which can be found in [19, 30] as well as [18].
The finite element method described in Chapter 4 is based on works by Brezzi and Fortin [13] and Girault and Raviart [21]. More material on computational fluid dynamics and its practical aspects can be found in [20, 41, 44]. At first the derivation of the variational formulation for the Stokes problem is being discussed, followed by different discretization techniques. Special emphasis lies on the Taylor-Hood elements

and the $P1$-$P1$-elements together with a stabilization by local pressure projections (see [15]). Then, the variational formulation of the Stokes problem is extended by a nonlinear convection term to a variational formulation of the Navier-Stokes problem. Furthermore the do-nothing boundary condition is being described. The nonlinear convection term is treated by the Newton-algorithm.
Chapter 5 is focused on the description and analysis of different types of error estimators, both for finite element and finite volume methods. There is the residual error estimator, which is available for both methods (see [23, 43]). The residual can be used to measure how well the original governing equations are satisfied. Then there is an error estimation technique based on the solution of local problems for the finite element method (see [43]). Its basic principles can easily be carried over to the finite volume method (see [23]), which is why it is described only for finite elements. Next, for finite elements there is the Zienkiewicz-Zhu-error estimator based on interpolation techniques for the gradient, such that the norm of the difference between the interpolated and the discrete gradient composes the error estimator (see [46, 47]). Furthermore, for the finite volume method, there is an interpolation technique using third order polynomials, where the solution is being interpolated by a third order polynomial along the direction vector between the centers of two neighbouring cells. The interpolated function is then used to recalculate the flux coefficients in the discretization, which are then used for the computation of the error estimator (see [29]). Finally, there is the neighbour-difference criterion, a simple and easy to implement, yet flexible, criterion. It can be used to discover areas in the computational domain where big changes occur in any quantity like the velocity or pressure, or quantities that can be calculated based on the primary variables (like vorticity).
Among the many available error estimators the residual error estimator was chosen to be implemented in `C++` for a Navier-Stokes solver based on the finite element method in Chapter 4. The residual estimator was also implemented in *AVL-Fire®* . The description of both implementations can be found in Chapter 6.
Numerical results for both methods are presented in Chapter 7.

2 Notations

The computational domain $\Omega \subset \mathbb{R}^d$ $(d = 2, 3)$, a bounded Lipschitz domain with boundary $\Gamma := \partial\Omega$, is being decomposed into elements T. The elements $T \in \mathcal{T}$ themselves are polygonally bounded or polyhedral elements, pairwise disjoint, connected and bounded, such that

$$\overline{\Omega} = \bigcup_{T \in \mathcal{T}} \overline{T}$$

holds. We call this decomposition a mesh $\mathcal{T}$ of Ω, where $\mathcal{T}$ is the set of all elements T. The mesh will consist of triangular elements for two space dimensions and polyhedra for three space dimensions. The boundary Γ can be represented exactly by edges or faces of finite elements if it is a polygonal line or if Ω is a polyhedron. Although the finite element discretization as described in Chapter 4 is derived for two space dimensions, it can analogously be extended to three dimensions using tetrahedra as finite elements, and the same error estimation techniques can be employed. The practical considerations and the implementation of the finite element method were done in two dimensions. For the finite volume method, arbitrary polyhedra can be used as finite elements. In practice, the most common shapes for the elements will be hexahedra, prisms, pyramids and tetrahedra. The terms *cell*, *control volume* and *element* are used synonymously with each other throughout the whole work, as well as the terms *triangulation*, *mesh* and *grid*.

2.1 Basic notations

Definition 2.1. *The set of all nodes of the triangulation is denoted by $\mathcal{N}$, whereas the nodes represent the cornerpoints of the elements $T \in \mathcal{T}$. The set of all boundary nodes is*

$$\mathcal{N}_\Gamma := \{a \in \mathcal{N} \mid a \in \Gamma\}$$

and

$$\mathcal{N}_0 := \mathcal{N} \backslash \mathcal{N}_\Gamma$$

is the set of all interior nodes.

Definition 2.2. *For $d = 2$, the set of all edges of the triangulation is denoted by $\mathcal{E}$. Let $\mathcal{E}_\Gamma$ be the set of all boundary edges, i.e. edges which are part of the boundary Γ. Furthermore, let $\mathcal{E}_0$ be the set of all interior edges, such that $\mathcal{E}_0 = \mathcal{E} \backslash \mathcal{E}_\Gamma$. We denote by $\mathcal{E}_D$ the set of all boundary edges for which Dirichlet boundary conditions are prescribed* *and analogously $\mathcal{E}_N$ the set of all edges with Neumann boundary conditions, such that $\mathcal{E}_\Gamma = \mathcal{E}_D \cup \mathcal{E}_N$. Moreover, the set of all edges of the element T is:*

$$\mathcal{E}_T := \{E \in \mathcal{E} \mid E \subset \partial T\}.$$

For three space dimensions, we can analogously define sets of faces F. The set of interior faces is denoted by $\mathcal{F}_0$. $\mathcal{F}_\Gamma$ is the set of faces on the boundary and $\mathcal{F}$ is the set of all faces.

Definition 2.3 (**Patches**). *We denote by*

$$\omega_a := \{T \in \mathcal{T} \mid a \subseteq \partial T\}$$

the set of all elements containing the node a, also called the node patch of a, see a) in Figure 2.1. Analogously, we define the element patch of T by

$$\omega_T := \left\{L \in \mathcal{T} \mid L = T \text{ or } \overline{T} \cap \overline{L} \in \mathcal{E}\right\}.$$

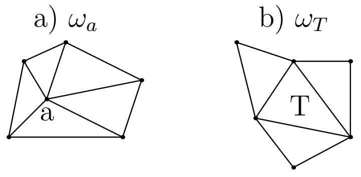


Figure 2.1: Node patch ω_a and element patch ω_T.

2.2 Regular triangulations

In the course of the present work, we assume for two dimensions that the triangulation is regular, a definition which can also be found in [27].

Definition 2.4 (**Regular triangulation**). *Let $\mathcal{T}$ be a triangulation of Ω. Then $\mathcal{T}$ is called a regular triangulation of Ω, if*

1. *for all $K, L \in \mathcal{T}$ either $K = L$ holds or $K \cap L = \emptyset$,*
2. *for all $K, L \in \mathcal{T}$ with $K \neq L$ one of the following cases holds:*
   - $\overline{K} \cap \overline{L} = \emptyset$,
   - $\overline{K} \cap \overline{L} \in \mathcal{N}$,
   - $\overline{K} \cap \overline{L} \in \mathcal{E}$,
3. *each $K \in \mathcal{T}$ has non-empty measure, i.e. meas $K > 0$.*

2.3 Choice of normal vectors

In many places in this work we need to have the normal vector $\mathbf{n}_E$ of an edge E in two dimensions or to a face F in three dimensions. In order to agree on the orientation of the normal vector, we always consider normal vectors in relation to an element T, i.e. $\mathbf{n}_E = \mathbf{n}_{T,E}$. This normal vector $\mathbf{n}_{T,E}$ is pointing out of the element T. In cases, where it is originally not clear which element is meant (for example when considering integrals of the normal derivative over an edge E), the orientation is defined explicitly in the context.

2.4 Diameter h_T and ρ_T

We denote by $d_T := \sup_{x,y\in T} |x-y|$ the diameter of T and by $h_T := |T|^{1/d}$ the local mesh size. Furthermore, r_T is the radius of the largest circle ($d=2$) or largest sphere ($d=3$) which can be inscribed in the finite element T. We call a finite element T shape regular, if the diameter d_T is bounded uniformly by the radius r_T, i.e.

$$d_T \leq c_F r_T \quad \forall T \in \mathcal{T},$$

where the constant c_F does not depend on $\mathcal{T}$. The results in the next chapters depend on the constant c_F, thus we only consider triangulations consisting of shape-regular elements.

3 The Finite Volume Method

In this chapter the finite volume method used to solve the Navier-Stokes equations in three space dimensions will be discussed, together with all the other numerical aspects needed to discretize the equations. The method allows us to use unstructured grids composed of arbitrary polyhedra. The starting point will be the modelling equations for fluid flow in integral form, from where the finite volume discretization will be developed. Finally, the solution procedure using a SIMPLE-algorithm (short for "Semi-Implicit Method for Pressure Linked Equations", see [19, 30]) based approach for the coupling of the velocities and the pressure is being described.

3.1 Finite volume approach

In this work, a cell-centered finite volume method is being discussed. For such methods, the solution is seen as an approximation of the exact solution in the centers of the control volumes $T \in \mathcal{T}$. The discretization method uses the conservation laws of the underlying physical problem in integral form applied over each control volume as starting point for the final algebraic equations. Therefore, when the requirements to the numerical grid, c.f. Chapter 2, are met, the finite volume method will be conservative, which is a desirable property for applications such as computational fluid dynamics.

3.1.1 The transport equation

The equations used to model fluid flow phenomena are transport equations. Such transport equations arise when we want to model the change of a quantity over time, such as the change of total momentum, energy or enthalpy in a domain $\Omega(t)$ which also depends on the time t. In our case, the transported quantity under consideration is the momentum $\rho\mathbf{u}$, where the vector $\mathbf{u}$ with its components u_k is the velocity and ρ is the density of the fluid. Newton's momentum conservation law states that the change of linear momentum in $\omega(t) \subset \Omega(t)$ is equal to all forces acting on $\omega(t)$ and its surface $\partial\omega(t)$, which results in the momentum balance equation for the k-th component u_k of the velocity $\mathbf{u}$

$$\frac{d}{dt}\int_{\omega(t)} \rho(t,x)u_k(t,x)\;dx = \int_{\omega(t)} \rho(t,x)f_k(t,x)\;dx + \int_{\partial\omega(t)} t_k(t,x,\mathbf{n})\;ds_x, \tag{3.1}$$

where the integral on the left hand side of the equation represents the total momentum in coordinate direction k in $\omega(t)$, f_k is the k-th component of the volumetric forces acting on $\omega(t)$ and t_k is the k-th component of the Cauchy stress vector. Applying Reynold's Transport Theorem (here written for an arbitrary differentiable function φ)

$$\frac{d}{dt}\int_{\omega(t)} \varphi(t,\mathbf{x})\,dx = \int_{\omega(t)} \frac{\partial\varphi}{\partial t}\,dx + \int_{\omega(t)} \nabla\cdot(\varphi\mathbf{u})\,dx, \tag{3.2}$$

see [33], integration by parts and using the representation $t_k(t,\mathbf{x},\mathbf{n}) = \sum_{i=1}^{d} T_{ki}n_i$ yields the balance equation for momentum in integral form

$$\int_{\omega(t)} \left\{ \frac{\partial\rho u_k}{\partial t} + \nabla\cdot(\rho u_k\mathbf{u}) - \sum_{i=1}^{d}\frac{\partial}{\partial x_i}T_{ki} - \rho f_k \right\} dx = 0.$$

Considering that the derivation of this equation is valid for arbitrary subdomains $\omega(t)\subset\Omega(t)$, we get the following equation in differential form

$$\frac{\partial\rho u_k}{\partial t} + \nabla\cdot(\rho u_k\mathbf{u}) = \sum_{i=1}^{d}\frac{\partial}{\partial x_i}T_{ki} + \rho f_k.$$

We assume that the fluid is Newtonian, which means that the viscous stresses (i.e. forces suffered by a fluid element from surrounding fluid, which cause it to gradually deform over time) are proportional to the rates of change of the fluid's velocity vector. If the fluid is incompressible, isotropic and its viscosity is constant, then there holds the representation $T = -pI + \mu(\nabla\mathbf{u} + (\nabla\mathbf{u})^\top)$ with p being the pressure and I the identity matrix. A fluid is called isotropic, if its mechanical properties are the same in each direction. For more information on material properties in fluid dynamics, see [10]. The transport equation then assumes the following form:

$$\frac{\partial\rho u_k}{\partial t} + \nabla\cdot(\rho u_k\mathbf{u}) = \sum_{i=1}^{d}\frac{\partial}{\partial x_i}\left(\mu\frac{\partial u_k}{\partial x_i}\right) + \rho f_k - \frac{\partial p}{\partial x_k}. \tag{3.3}$$

3.1.2 Finite volumes

We derive a method to solve equation (3.3). For the rest of the chapter, to avoid confusion with indices, we write ϕ instead of u_k. For simplicity, we choose $\mu = 1$. Since we consider only incompressible flow, the density is constant over time.
Mass cannot be created or destroyed, which is why the total mass M in a control volume T is constant:

$$\frac{d}{dt}M(t) = \frac{d}{dt}\int_T \rho\,dx = \int_T \frac{\partial}{\partial t}\rho + \nabla\cdot(\rho\mathbf{u})\,dx = 0, \tag{3.4}$$

using Reynold's transport theorem (3.2), which leads us to the continuity equation, with $\frac{\partial}{\partial t}\rho = 0$ due to incompressibility:

$$\int_T \nabla\cdot(\rho\mathbf{u})\,dx = 0$$

and using the Gauss' theorem

$$\sum_{j=1}^{n_f} \int_{A_j} \rho \mathbf{u} \cdot \mathbf{n} \, ds_x = 0 \tag{3.5}$$

with n_f being the number of faces A_j of the polyhedral control volume T. The surface integral over ∂T is split into surface integrals over the faces A_j. The volume integral in equation (3.4) has been transformed into surface integrals in (3.5) using the Gauss' theorem

$$\int_T \frac{\partial \varphi}{\partial x_i} \, dx = \int_{\partial T} \varphi n_i \, ds_x,$$

where n_i is the i-th component of the outward pointing normal vector $\mathbf{n}$. In order to approximate surface and volume integrals, the midpoint rule is used. For φ being any scalar valued function, we get

$$\int_{A_j} \varphi \, ds_x \approx |A_j| \varphi_j \quad \text{and} \quad \int_T \varphi \, dx \approx |T| \varphi_T,$$

where $|A_j|$ denotes the surface area of face A_j and $|T|$ the volume of the control volume T. The subscripts j and T denote the value of φ at the center of the face A_j or center of the volume T, respectively.
We approximate the surface integrals over A_j in the continuity equation (3.5) by

$$m_j := \rho_j |A_j| (\mathbf{u} \cdot \mathbf{n})_j \approx \int_{A_j} \rho \mathbf{u} \cdot \mathbf{n} \, ds_x, \tag{3.6}$$

which leads to

$$\sum_{j=1}^{n_f} m_j = 0. \tag{3.7}$$

By integrating equation (3.3) over the control volume T we get

$$\int_T \frac{\partial \rho \phi}{\partial t} + \nabla \cdot (\rho \phi \mathbf{u}) \, dx = \int_T \sum_{i=1}^{d} \frac{\partial^2 \phi}{\partial x_i^2} \, dx + \int_T \left(\rho f_k - \frac{\partial p}{\partial x_k} \right) dx. \tag{3.8}$$

Then the following integral form of the transport equation can be obtained for stationary, i.e. time independent flows (thus the time derivative in (3.8) vanishes):

$$\sum_{j=1}^{n_f} \underbrace{\int_{A_j} \rho \phi \mathbf{u} \cdot \mathbf{n} \, ds_x}_{\text{convection term}} = \sum_{j=1}^{n_f} \underbrace{\int_{A_j} \nabla \phi \cdot \mathbf{n} \, ds_x}_{\text{diffusion term}} + \underbrace{\int_T \left(\rho f_k - \frac{\partial p}{\partial x_k} \right) dx}_{\text{source terms}}. \tag{3.9}$$

With these notations, the general transport equation (3.9) can be approximated as:

$$\sum_{j=1}^{n_f} \underbrace{m_j \phi_j}_{C_j} - \sum_{j=1}^{n_f} \underbrace{|A_j| \left(\nabla \phi \cdot \mathbf{n} \right)_j}_{D_j} = \underbrace{|T| \left(\rho f_k \right)_T}_{S_T^V} - \sum_{j=1}^{n_f} \underbrace{|A_j| \left(p n_k \right)_j}_{S_j^A} \tag{3.10}$$

or

$$\sum_{j=1}^{n_f} C_j - \sum_{j=1}^{n_f} D_j = S_T^V - \sum_{j=1}^{n_f} S_j^A,$$

where the source terms have been transformed according to

$$\int_T \left(\rho f_k - \frac{\partial p}{\partial x_k} \right) \, dx \approx |T| \, (\rho f_k)_T - \sum_{j=1}^{n_f} \int_{A_j} p n_k \, ds_x = |T| \, (\rho f_k)_T - \sum_{j=1}^{n_f} |A_j| p_j n_{k,j}.$$

The convective terms $C_j = m_j \phi_j$ have been approximated through

$$C_j = \int_{A_j} \rho \phi \mathbf{u} \cdot \mathbf{n} \, ds_x \approx \phi_j m_j.$$

3.2 Interpolation schemes

Equation (3.10) still contains derivatives of ϕ and values of ϕ at cell faces. In order to arrive at an algebraic system consisting only of the values of ϕ at the cell centroids, finite difference schemes are employed to replace the derivatives of ϕ, as well as interpolation for values of ϕ at the cell faces. The way in which the values of variables at cell faces are calculated has a huge impact on the accuracy and stability of the numerical method. Those properties were investigated in detail for example in [31]. In this section, two standard finite difference schemes are presented: The central differencing scheme (CDS) and the upwind differencing scheme (UDS).
For the central differencing scheme, the cell face value is approximated by a linear combination of the values of ϕ in the cell centers of two cells T and N_j neighbouring each other over the face A_j, using a factor

$$f_j = \frac{|\mathbf{r}_{N_j} - \mathbf{r}_j|}{|\mathbf{r}_j - \mathbf{r}_T| + |\mathbf{r}_{N_j} - \mathbf{r}_j|},$$

where $\mathbf{r}_T$ and $\mathbf{r}_{N_j}$ are the position vectors of the cell centers of T and its neighbour N_j over the face A_j, see Figure 3.1. The cellcenters are defined as the volume mean

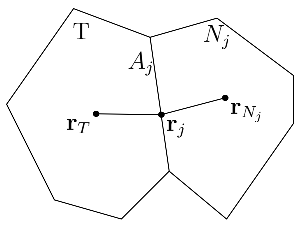


Figure 3.1: A cell T and its neighbour N_j.

over all position vectors inside of the control volume:

$$\mathbf{r}_T := \frac{1}{|T|} \int_T \mathbf{r} \, dx. \tag{3.11}$$

In order to find an alternative expression for (3.11), we first consider the k-th coordinate $x_{T,k}$ of the cellcenter $\mathbf{r}_T$:

$$x_{T,k} = \frac{1}{|T|} \int_T x_k \, dx. \tag{3.12}$$

We use the obvious relation

$$x_k = \frac{1}{3} x_k \nabla \cdot \mathbf{r} = \frac{1}{3} \left[\nabla \cdot (x_k \mathbf{r}) - \mathbf{r} \cdot \nabla x_k \right]$$

and substitute into (3.12):

$$x_{T,k}|T| = \frac{1}{3} \int_T \nabla \cdot (x_k \mathbf{r}) \, dx - \frac{1}{3} \int_T \mathbf{r} \cdot \nabla x_k \, dx. \tag{3.13}$$

Together with

$$\mathbf{r} \cdot \nabla x_k = \mathbf{r} \cdot \mathbf{e}_k = x_k,$$

where $\mathbf{e}_k$ is the k-th unit vector, we get through (3.13)

$$x_{T,k}|T| = \frac{1}{3} \int_T \nabla \cdot (x_k \mathbf{r}) \, dx - \frac{1}{3} \int_T \mathbf{r} \cdot \nabla x_k \, dx = \frac{1}{3} \int_T \nabla \cdot (x_k \mathbf{r}) \, dx - \frac{1}{3} x_{T,k}|T|$$

and finally obtain using Gauss' Theorem

$$4 x_{T,k}|T| = \int_T \nabla \cdot (x_k \mathbf{r}) \, dx = \int_{\partial T} x_k \mathbf{r} \cdot \mathbf{n} \, ds_x.$$

The term $\mathbf{r} \cdot \mathbf{n}$ is constant on the planar polygonal faces of T, thus yielding

$$\int_{\partial T} x_k \mathbf{r} \cdot \mathbf{n} \, ds = \sum_{j=1}^{n_f} \mathbf{r}_j \cdot \mathbf{n}_j \int_{A_j} x_k \, ds_x = \sum_{j=1}^{n_f} \mathbf{r}_j \cdot \mathbf{n}_j x_{T,k} |A_j|$$

and finally a new expression for the cell center (3.11):

$$\mathbf{r}_T = \frac{1}{4|T|} \sum_{j=1}^{n_f} (\mathbf{r}_j \cdot \mathbf{A}_j) \, \mathbf{r}_j \quad \text{with} \quad \mathbf{A}_j = \mathbf{n}_j |A_j|.$$

This formula was first derived by Wang [45]. Furthermore, we compute the cell face centers $\mathbf{r}_j$ of A_j by decomposing the polygon comprising the cell face into triangles, computing the centers of the triangles by taking the average of the vertices of each triangle and then computing the average of the centers, weighted by the area of the triangles.
The value of ϕ at the cell face j center can now be computed by linear combination of the values of ϕ at the centers of adjacent cells T and N_j:

$$\phi_j^{CDS} = f_j \phi_T + (1 - f_j) \phi_{N_j}.$$

The disadvantage of this scheme is that it generates numerical oscillations which possibly leads to unbounded and non-monotonic solutions and thus to instability. Whether oscillations occur when using central differencing, depends on the local Peclet number. For the one-dimensional case, the local Peclet number is given by

$$Pe_T = \frac{\rho u_T \Delta x}{\mu}$$

with Δx being the distance between cell centers of a uniform grid. It can be shown that if $Pe \leq 2$ at every cell, no oscillations will occur for central differencing. This has been investigated in [19, 30]. A more simple version of central differencing is using arithmetic averaging:

$$\phi_j = \frac{1}{2}(\phi_T + \phi_{N_j}).$$

Another possibility is to use the upwind differencing scheme (UDS). It is unconditionally bounded but produces numerical diffusion. For UDS, the value at the upwind nodes is taken as an approximation for the value at the cell face center:

$$\phi_j^{UDS} = \begin{cases} \phi_T & \text{if } m_j \geq 0 \\ \phi_{N_j} & \text{if } m_j < 0, \end{cases} \tag{3.14}$$

with m_j being defined through (3.6).

3.3 Algebraic system

In this section, the set of linear equations as a discrete counterpart for the equation (3.9) will be derived, using equations (3.10) as a starting point. When solving the algebraic system, we will use an iterative strategy to treat the nonlinearities, which means that ϕ is treated implicitly at some of its occurences, i.e. its values from the previous iteration $n-1$ are considered, denoted by the super-index $n-1$.

3.3.1 Convective Fluxes

As discussed in Section 3.1, the convective flux through the cell face A_j is approximated as:

$$C_j \approx \phi_j m_j.$$

In order to express ϕ_j by the values ϕ_T and ϕ_{N_j}, any of the interpolation schemes discussed in Section 3.2 can be applied. The following approach is based on the upwind differencing scheme (3.14):

$$C_j = \max\{m_j, 0\}\,\phi_T - \max\{-m_j, 0\}\,\phi_{N_j}. \tag{3.15}$$

3.3.2 Diffusive fluxes

For the diffusive term D_j in equation (3.10) the derivative of ϕ or the gradient of ϕ at the cell face centers is needed:

$$D_j = \nabla\phi_j \cdot \mathbf{A}_j \approx \int_{A_j} \nabla\phi \cdot \mathbf{n}\, ds_x \quad \text{with} \quad \mathbf{A}_j = |A_j|\mathbf{n}_j.$$

One possibility to express the derivative of ϕ is to use a least squares fit. This method was introduced by Barth and Muzaferija [28]. Linear variation of ϕ in the control volume is assumed. The Taylor series expansion for ϕ then reads:

$$\phi(\mathbf{r}) = \phi_T + \nabla\phi_T \cdot (\mathbf{r} - \mathbf{r}_T).$$

The components of $\nabla\phi_T$ can be computed by minimizing the sum of the squares of the differences $\phi_{N_j} - \phi(\mathbf{r}_{N_j})$. This is achieved by differentiating by the k-th component of the gradient $g_\ell := (\nabla\phi_T)_\ell$ and setting the sum equal to zero for each $\ell = 1, \dots, 3$ (for three space dimensions), which results in a system of 3 equations:

$$\frac{\partial}{\partial g_\ell} \sum_{j=1}^{n_f} \Big(\phi_{N_j} - \phi_T - \nabla\phi_T \cdot \mathbf{d}_j\Big)^2 = -2\sum_{j=1}^{n_f} \Big(\phi_{N_j} - \phi_T - \nabla\phi_T \cdot \mathbf{d}_j\Big)\,(\mathbf{d}_j)_\ell = 0,$$

leading to

$$\sum_{j=1}^{n_f} \Big(\phi_{N_j} - \phi_T\Big)\,\mathbf{d}_j = \sum_{j=1}^{n_f} (\mathbf{d}_j)_\ell\, \nabla\phi_T \cdot \mathbf{d}_j \tag{3.16}$$

where $\mathbf{d}_j := \mathbf{r}_{N_j} - \mathbf{r}_T$ is the distance vector connecting the cell centers $\mathbf{r}_T$ and $\mathbf{r}_{N_j}$ of two neighbouring cells T and N_j. By defining the matrix G and vector h as

$$G_{\ell i} := \sum_{j=1}^{n_f} (\mathbf{d}_j)_\ell\, (\mathbf{d}_j)_i \quad \text{and} \quad h_\ell := \sum_{j=1}^{n_f} \Big(\phi_{N_j} - \phi_T\Big)\,(\mathbf{d}_j)_\ell,$$

equation (3.16) can be rewritten into the following representation for the gradient in the center of T:

$$\nabla\phi_T = G^{-1}\mathbf{h}. \tag{3.17}$$

The gradient at the cell face center follows either by linear interpolation or arithmetic averaging using the interpolation factor f_j from Section 3.1:

$$\overline{\nabla\phi_j} = \begin{cases} f_j\nabla\phi_T + (1 - f_j)\nabla\phi_{N_j} & \text{linear interpolation,} \\ \frac{1}{2}(\nabla\phi_T + \nabla\phi_{N_j}) & \text{arithmetic averaging.} \end{cases} \tag{3.18}$$

The above technique for gradient calculation involves the values of ϕ at neighbouring cells of the neighbour cell N_j. For practical reasons, this must be avoided, but the gradients can still be calculated implicitly. Therefore, an expression containing only ϕ_T and ϕ_{N_j} and the gradient at the previous iteration step $n - 1$ is to be used.
The goal in approximating the diffusive term D_j is to find an expression for $\nabla\phi_j \cdot \mathbf{A}_j$.

The idea is to split up the surface normal $\mathbf{A}_j$ into a part $\mathbf{s}$ which has the direction of the vector $\mathbf{d}_j$ connecting the cell centers of T and N_j and a remainder part $\mathbf{k}$, such that $\mathbf{A}_j = \mathbf{s} + \mathbf{k}$. This principle is also explained in [23]. Since $\mathbf{s}$ and $\mathbf{d}_j$ are parallel, we can simply use the following differencing formula:

$$\mathbf{s} \cdot \nabla \phi_j = |\mathbf{s}| \frac{\phi_{N_j} - \phi_T}{|\mathbf{d}_j|}.$$

For the remainder we set $\mathbf{k} := \mathbf{A}_j - \mathbf{s}$ and use an *over-relaxed approach* with:

$$\mathbf{s} := \frac{\mathbf{d}_j}{\mathbf{d}_j \cdot \mathbf{A}_j} |\mathbf{A}_j|^2$$

which leads to, when using the gradient at the cell face interpolated as above, but calculated with values from the previous iteration:

$$D_j = \nabla \phi_j \cdot \mathbf{A}_j = \nabla \phi_j \cdot (\mathbf{s} + \mathbf{k}) \approx \frac{|\mathbf{A}_j|^2}{\mathbf{A}_j \cdot \mathbf{d}_j} \left(\phi_{N_j} - \phi_T \right) + \overline{\nabla \phi_j}^{n-1} \cdot \left(\mathbf{A}_j - \frac{|\mathbf{A}_j|^2}{\mathbf{A}_j \cdot \mathbf{d}_j} \mathbf{d}_j \right). \tag{3.19}$$

Remark 3.1. *Among other approaches like the orthogonal correction approach or the minimum correction approach, see [23], the over-relaxed approach was chosen for implementation because it yielded the most stable results in practical experiments.*

3.3.3 Boundary and initial conditions

The convective and diffusive fluxes at the boundaries are calculated in the same way as for the inner cell faces, but instead of other differencing schemes, only upwind differencing (3.14) is used for convective fluxes. The values of ϕ are taken from the previous iteration $n-1$:

$$C_b = \max\{m_b, 0\} \phi_T^{n-1} - \max\{-m_b, 0\} \phi_b^{n-1}.$$

The subindex b represents the boundary face b. Writing equation (3.19) for the boundary faces reads:

$$D_b = \frac{|\mathbf{A}_b|^2}{\mathbf{A}_b \cdot \mathbf{d}_b} (\phi_b - \phi_T) + \nabla \phi_T^{n-1} \cdot \left(\mathbf{A}_b - \frac{|\mathbf{A}_b|^2}{\mathbf{A}_b \cdot \mathbf{d}_b} \mathbf{d}_b \right),$$

where $\mathbf{d}_b$ is the difference vector $\mathbf{d}_b := \mathbf{r}_b - \mathbf{r}_T$ between the center $\mathbf{r}_b$ of the face A_b and the cell center $\mathbf{r}_T$ of the associated cell T. The pressure at the boundary is either prescribed, or it can be extrapolated from the inside of the solution domain. Therefore, it is either obtained by $p_b = p_T$ or by the following linear extrapolation formula:

$$p_b = p_T + \nabla p_T \cdot \mathbf{d}_b.$$

Symmetry conditions

Symmetric boundary conditions mean that variables do not change in normal direction to the boundary, resulting in zero diffusive fluxes. Furthermore, velocities in normal direction are zero and therefore yield zero convective fluxes. Symmetry boundary conditions should reflect natural symmetries.

3.3.4 Algebraic equations

Gathering the results from the previous sections in this chapter, the discretization of the transport equation (3.3) leads to a set of linear equations. For every control volume T, these equations can be written in the following compact form:

$$a_T \phi_T = \sum_{j=1}^{n_f} a_j \phi_{N_j} + S_T, \tag{3.20}$$

with n_f being the number of interior faces of the volume T and N_j the neighbouring element of T over the face A_j. The coefficients a_T and a_j are then defined as follows (upwind differencing (3.14) was used for convective fluxes):

$$\begin{aligned} a_T &= \sum_{j=1}^{n_f} a_j + \sum_{b=1}^{n_b} a_b, \\ a_j &= a_j^C + a_j^D = \max\{-m_j, 0\} + \frac{|\mathbf{A}_j|^2}{\mathbf{A}_j \cdot \mathbf{d}_j}. \end{aligned} \tag{3.21}$$

Furthermore, assembling the explicit terms for the sources yields:

$$\begin{aligned} S_T =& \sum_{j=1}^{n_f} \left[\nabla \phi_j \cdot \left(\mathbf{A}_j - \frac{|\mathbf{A}_j|^2}{\mathbf{A}_j \cdot \mathbf{d}_j} \mathbf{d}_j \right) \right] + \\ & \sum_{b=1}^{n_b} \left[a_b \phi_b + \nabla \phi_T \cdot \left(\mathbf{A}_j - \frac{|\mathbf{A}_b|^2}{\mathbf{A}_b \cdot \mathbf{d}_b} \mathbf{d}_b \right) \right] + (\rho f_k)_T - \sum_{j=1}^{n_f} (p n_k)_j \,. \end{aligned} \tag{3.22}$$

Under-relaxation

Instead of using the resultant variable vector ϕ^{new} which is the solution of system (3.20) as the solution at iteration step n, we can also define ϕ^n as a relaxation between ϕ^{n-1} from the previous iteration step and ϕ^{new}:

$$\phi^n := \phi^{n-1} + \alpha_k (\phi^{new} - \phi^{n-1}).$$

For the under-relaxation factor α_k we choose a value between 0 and 1. When applying this under-relaxation technique, system (3.20) still retains the same form, but the diagonal coefficient of the resulting system matrix as well as the source terms have to be redefined:

$$a_T^* = \frac{a_T}{\alpha_k}, \qquad (3.23)$$

$$S_T^* = S_T + \frac{1-\alpha_k}{\alpha_k} a_T \phi_T^{n-1}$$

which leads to the modified system

$$a_T^* \phi_T = \sum_{j=1}^{n_f} a_j \phi_{N_j} + S_T^*.$$

The choice of appropriate relaxation factors α_k is problem dependent. Under relaxation can be interpreted as a fixed point iteration method. For such a method, convergence can be established by making the parameter α_k small enough. Decreasing the under-relaxation factor α_k will also decrease the rate of convergence.

3.4 Numerical accuracy

In the following section, we examine the numerical accuracy of the proposed finite difference and interpolation schemes of the previous sections. Those considerations can also be found in [19]. We explain the underlying principle by considering the one dimensional case. Multidimensional finite differences are handled by treating each coordinate separately, so the results of this section can be adapted to higher dimensions.
Any continuous sufficiently often differentiable function $\phi(x) \in C^p(\Omega)$ can be expressed by a Taylor series expansion around a point $x_0 \in \mathbb{R}$ up to order p:

$$\phi(x) = \phi(x_0) + (x-x_0)\left(\frac{\partial \phi}{\partial x}\right)\bigg|_{x=x_0} + \frac{(x-x_0)^2}{2!}\left(\frac{\partial^2 \phi}{\partial x^2}\right)\bigg|_{x=x_0} + \frac{(x-x_0)^3}{3!}\left(\frac{\partial^3 \phi}{\partial x^3}\right)\bigg|_{x=x_0} + \dots + \frac{(x-x_0)^{(p-1)}}{(p-1)!}\left(\frac{\partial^{(p-1)} \phi}{\partial x^{(p-1)}}\right)\bigg|_{x=x_0} + R, \qquad (3.24)$$

with $x \in \mathbb{R}$ where R stands for the remaining terms which are of order p and higher. There exists a number ξ in the closed interval with bounds x and x_0 such that

$$R = \frac{(x-x_0)^p}{p!} \frac{\partial^p}{\partial x^p} \phi(\xi).$$

Let $x_0 < \ldots < x_N$ be a series of real numbers, representing a grid of a given interval (domain).
The Taylor series expansion will play an important role in the deduction of approximation orders for various interpolation and finite difference schemes. If all the used approximations are second order accurate, then also the final finite volume method is called second order accurate.

3.4.1 Order of approximation for finite difference schemes

The order of approximation for first derivatives plays a role in the approximation of the diffusive fluxes. We can use the Taylor expansion (3.24) to derive an expression for the first derivative of ϕ. For that purpose, it is necessary to assume $\phi \in C^3(\Omega)$. The subindex i denotes evaluation of an expression at the node x_i, i.e. $(\cdots)_i = (\cdots)|_{x=x_i}$.

Uniform grids

We start by considering uniform grids, i.e. the distance Δ_i between two grid points x_{i+1} and x_i equals a fixed distance h for all i. For $p = 3$, we replace x_0 in (3.24) by x_i and x by x_{i+1} and rearrange the terms:

$$\left(\frac{\partial \phi}{\partial x}\right)_i = \frac{\phi_{i+1} - \phi_i}{h} - \frac{h}{2!}\left(\frac{\partial^2 \phi}{\partial x^2}\right)_i - \frac{h^2}{3!}\frac{\partial^3 \phi}{\partial x^3}(\xi), \tag{3.25}$$

for some $\xi \in [x_i, x_{i+1}]$ or when expanding the Taylor series once for x_{i-1} and once for x_{i+1} and subtracting both, with $\eta \in [x_{i-1}, x_i]$ (the terms with the second derivative of ϕ cancel each other out):

$$\left(\frac{\partial \phi}{\partial x}\right)_i = \frac{\phi_{i+1} - \phi_{i-1}}{h} + \frac{h^2}{3}\left(\frac{\partial^3}{\partial x^3}\phi(\xi) + \frac{\partial^3}{\partial x^3}\phi(\eta)\right).$$

We get the following estimate of the error in a grid point x_i:

$$\left|\left(\frac{\partial \phi}{\partial x}\right)_i - \frac{\phi_{i+1} - \phi_{i-1}}{h}\right| = \left|\frac{h^2}{3}\left(\frac{\partial^3}{\partial x^3}\phi(\xi) + \frac{\partial^3}{\partial x^3}\phi(\eta)\right)\right| \leq \frac{h^2}{3!}\max_{\xi\in[x_{i-1},x_{x+1}]}\left|\frac{\partial^3}{\partial x^3}\phi(\xi)\right|$$

under the assumption that ϕ is sufficiently many times differentiable. If we consider the maximum of those errors over all grid points, we get

$$\max_{i\in\{1,\dots,N\}}\left|\left(\frac{\partial \phi}{\partial x}\right)_i - \frac{\phi_{i+1} - \phi_{i-1}}{h}\right| \leq \frac{h^2}{3!}\underbrace{\max_{\xi\in[x_0,x_N]}\left|\frac{\partial^3}{\partial x^3}\phi(\xi)\right|}_{=constant}$$

which means that the maximum error in the grid points decreases with second order as the grid spacing h decreases.
The same as in (3.25) can be done by using the points x_i and x_{i-1}. We arrive at approximations for the first derivative if we leave away the terms on the right side which contain derivatives of higher order:

$$\left(\frac{\partial \phi}{\partial x}\right)_i \approx \frac{\phi_{i+1} - \phi_i}{x_{i+1} - x_i}, \qquad \left(\frac{\partial \phi}{\partial x}\right)_i \approx \frac{\phi_i - \phi_{i-1}}{x_i - x_{i-1}}, \qquad \left(\frac{\partial \phi}{\partial x}\right)_i \approx \frac{\phi_{i+1} - \phi_{i-1}}{x_{i+1} - x_{i-1}}.$$

These are the so called forward- (FDS), backward- (BDS), and central differencing schemes. The terms containing higher order derivatives that were left away are called the truncation errors. The accuracy of the approxmation and the error decay rate are determined by those.
This order of approximation also holds true for non-uniform grids asymptotically, which has been shown in [19].

3.4.2 Order of approximation for interpolation schemes

Linear interpolation

Let's recall the linear interpolation scheme from Section 3.2, rewritten for one dimension. Here, it is sufficient to assume $\phi \in C^2(\Omega)$. We want to interpolate based on points x_i and x_{i+1} to a point $x_{i+1/2}$ that lies between those points:

$$\phi_{i+1/2} = \lambda\phi_{i+1} + (1-\lambda)\phi_i \quad \text{with} \quad \lambda = \frac{x_{i+1/2} - x_i}{x_{i+1} - x_i}. \tag{3.26}$$

In order to see this scheme's order of convergence, we use a Taylor expansion of ϕ_{i+1} around the point x_i to find an expression for the first derivative

$$\left(\frac{\partial\phi}{\partial x}\right)_i = \frac{\phi_{i+1} - \phi_i}{x_{i+1} - x_i} - \frac{x_{i+1} - x_i}{2}\left(\frac{\partial^2\phi}{\partial x^2}\right)_i + R$$

and substitute this in

$$\phi_{i+1/2} = \phi_i + (x_{i+1/2} - x_i)\left(\frac{\partial\phi}{\partial x}\right)_i + \frac{(x_{i+1/2} - x_i)^2}{2}\left(\frac{\partial^2\phi}{\partial x^2}\right)_i + R$$

which finally leads to (after rearrangement of the terms):

$$\phi_{i+1/2} = \lambda\phi_{i+1} + (1-\lambda)\phi_i - \frac{(x_{i+1/2} - x_i)(x_i - x_{i+1/2})}{2}\left(\frac{\partial^2\phi}{\partial x^2}\right)_i + R \tag{3.27}$$

and yields in the case of a uniform grid, analogously to Section 3.4.1 with $x_{i+1/2} - x_i = x_i - x_{i+1/2} = h/2$ and $\xi \in [x_i, x_{i+1}]$:

$$\left|\phi_{i+1/2} - \lambda\phi_{i+1} + (1-\lambda)\phi_i\right| = \frac{h^2}{2}\left|\frac{\partial^2}{\partial x^2}\phi(\xi)\right|,$$

thus the maximum error of the scheme over all grid points scales with h^2:

$$\max_{i\in\{1,\dots,N\}}\left|\phi_{i+1/2} - \lambda\phi_{i+1} + (1-\lambda)\phi_i\right| \leq \frac{h^2}{2}\max_{\xi\in[x_0,x_N]}\left|\frac{\partial^2}{\partial x^2}\phi(\xi)\right|.$$

The first part of expansion (3.27) is exactly the linear interpolation (3.26).

Upwind differencing scheme

Let's recall the upwind differencing scheme (3.14), written for one dimension:

$$\phi_{i+1/2} = \begin{cases} \phi_i & \text{if } \rho_{i+1/2}u_{i+1/2} \geq 0, \\ \phi_{i+1} & \text{if } \rho_{i+1/2}u_{i+1/2} < 0. \end{cases} \tag{3.28}$$

The Taylor expansion around a point x_i reads:

$$\phi_{i+1/2} = \phi_i + (x_{i+1/2} - x_i)\left(\frac{\partial\phi}{\partial x}\right)_i + \frac{(x_{i+1/2} - x_i)^2}{2}\left(\frac{\partial^2\phi}{\partial x^2}\right)_i + R.$$

We see that the UDS approximation (3.28) retains only the first term on the right hand side, thus leaving terms which scale proportionally to the first power of the grid spacing.
The advantage of the upwind differencing scheme is that it is unconditionally stable, see [19], but due to its low order of accuracy it requires very fine grids to achieve accurate solutions.

3.4.3 Approximation of integrals

The approximations of integration for volume integrals through the midpoint rule

$$\int_T \phi\ dx \approx |T|\phi_T$$

and surface integrals over a face A_j

$$\int_{A_j} \phi\ ds_x \approx |A_j|\phi_j,$$

with ϕ_T and ϕ_j being the values of ϕ at cell centers and face centers, respectively, are at least second order accurate. This can be seen by analyzing a one-dimensional situation. We consider integration over an interval $[a, b]$ with $a, b \in \mathbb{R}$. We denote by $x_0 = \frac{a+b}{2}$ the midpoint of the interval and by $h = b - a$ the interval length. Furthermore, we assume $\phi \in C^2(\Omega)$. The Taylor expansion then reads with $\xi \in [a, b]$:

$$\phi(x) = \phi(x_0) + (x - x_0)\frac{\partial}{\partial x}\phi(x_0) + \frac{(x - x_0)^2}{2}\frac{\partial^2}{\partial x^2}\phi(\xi).$$

Then integration yields:

$$\begin{aligned}\int_a^b \phi(x)\ dx &= \int_a^b \phi(x_0) + (x - x_0)\frac{\partial}{\partial x}\phi(x_0) + \frac{(x - x_0)^2}{2}\frac{\partial^2}{\partial x^2}\phi(\xi)\ dx \\ &= h\phi(x_0) + \frac{h^3}{24}\frac{\partial^2}{\partial x^2}\phi(\xi).\end{aligned}$$

Therefore, the error in the integral approximation by the midpoint rule is proportional to the third order of the integration interval length h:

$$\left|\int_a^b \phi(x)\ dx - h\phi(x_0)\right| = \frac{h^3}{24}\max_{\xi\in[a,b]}\left|\frac{\partial^2}{\partial x^2}\phi(\xi)\right|.$$

This result can be carried over analogously to higher dimensions.

3.5 SIMPLE-based solution procedure for incompressible flow

Aside from the velocity components u_k, the other main unknown is the pressure. This means, that a pressure-velocity coupling is needed. This coupling can be done by the SIMPLE algorithm (short for "Semi-Implicit Method for Pressure Linked Equations"), which was introduced by Patankar and Spalding [30]. In the SIMPLE algorithm, a collocated variable arrangement is assumed, which means that the values of unknown physical quantities are considered at the center points of the control volumes instead of points in the faces of the control volumes. This leads to a smaller number of degrees of freedom, but is also the reason why interpolation techniques are required. The following description of the SIMPLE algorithm is based on [17, 18]. This description is correct for staggered grids (see [19]), but is still suitable to explain the principle of the SIMPLE algorithm. We explain the method for staggered grids since they don't require interpolation techniques, thus unnecessary complexity is avoided. The missing interpolation techniques can be found in [34].
So far in this chapter, the pressure appeared in the equations only as a source, but is in fact an unknown and needs to be treated as such. Equation (3.20) with the velocities u_k substituted for ϕ together with the discretized continuity equation (3.7) can be rewritten in matrix form as:

$$\begin{bmatrix} A(\mathbf{u}) & M \\ C & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{b} \\ 0 \end{bmatrix} \tag{3.29}$$

with the velocity vector field $\mathbf{u} = (u_1, \ldots, u_N, v_1, \ldots, v_N, w_1, \ldots, w_N)$ (for three space dimensions), where N is the number of control volumes T of the triangulation $\mathcal{T}$ as well as the pressure field $\mathbf{p} = (p_1, \ldots, p_N)$. Here A denotes the coefficient matrix with coefficients defined through (3.21) or (3.23) acting on the velocity field $\mathbf{u}$, while M's coefficients originate from interpolation of the pressure in (3.22). Instead of treating the pressure as a source, it is now written separately as part of the unknowns. $\mathbf{b}$ is the right hand side containing sources and sinks, and C the matrix which arises through discretization of the continuity equation (3.7).
The SIMPLE algorithm deals with the non-linearity of equation system (3.29) by employing an iterative strategy. A is nonlinear since for the computation of A's coefficients, values of $\mathbf{u}$ are needed. For the calculation of the velocity field at the n-th iteration, $\mathbf{u}^{(n)}$ is split up into a tentative velocity field $\mathbf{u}^*$ and an update part $\mathbf{u}'$. A similar splitting is also applied to the pressure by dividing the pressure $\mathbf{p}^{(n)}$ into a part with values from the previous iteration and and a part $\mathbf{p}'$, called the pressure correction term. This leads to:

$$\mathbf{u}^{(n)} = \mathbf{u}^* + \mathbf{u}' \tag{3.30}$$

$$\mathbf{p}^{(n)} = \mathbf{p}^{(n-1)} + \mathbf{p}', \tag{3.31}$$

and they are defined through the following equations:

$$A(\mathbf{u}^{(n-1)})\mathbf{u}^* + M\mathbf{p}^{(n-1)} = \mathbf{b} \tag{3.32}$$

$$A(\mathbf{u}^{(n-1)})\mathbf{u}^{(n)} + M\mathbf{p}^{(n)} = \mathbf{b}, \tag{3.33}$$

where we use "old" values $\mathbf{u}^{(n-1)}$ from the previous iteration for the computation of the nonlinear operator A. Subtracting equation (3.32) from (3.33) and using (3.30) and (3.31) yields

$$A(\mathbf{u}^{(n-1)})\mathbf{u}' + M\mathbf{p}' = \mathbf{0},$$

which contains only the update and correction parts. In the next step, the operator A is split up into its diagonal part $A_D = \mathrm{diag}(A(\mathbf{u}^{(n-1)}))$ and its remainder

$$A_R := A(\mathbf{u}^{(n-1)}) - A_D.$$

Patankar's idea (see [30]) was to treat the remainder A_R as a zero matrix, such that the diagonal matrix A_D and therefore also A can easily be inverted, which gives

$$\mathbf{u}' = -A_D^{-1} M\mathbf{p}' \tag{3.34}$$

as an expression for the update velocity $\mathbf{u}^*$. Plugging this into the continuity equation

$$C\mathbf{u}^{(n)} = \mathbf{0} \quad \Longrightarrow \quad C\mathbf{u}' = 0 - C\mathbf{u}^*$$

leads to

$$-CA_D^{-1}M\mathbf{p}' = -C\mathbf{u}^*. \tag{3.35}$$

The following steps represent one iteration of the SIMPLE algorithm:

1. Solve the first equation in (3.29) for $\mathbf{u} = \mathbf{u}^*$, i.e. solve $A(\mathbf{u}^{(n-1)})\mathbf{u}^* + Mp^{(n-1)} = \mathbf{b}$.
2. Assemble equation system (3.35) and solve for $\mathbf{p}'$.
3. Compute velocity update $\mathbf{u}'$ by (3.34).
4. Compute $\mathbf{u}^{(n)}$ and $\mathbf{p}^{(n)}$ by equations (3.30) and (3.31).

4 The Finite Element Method

In this chapter the finite element method used to solve the steady state incompressible Navier-Stokes equations in two space dimensions will be discussed by starting with a description of the simpler and linear Stokes equations and then adding corresponding terms in order to reach a discretization of the Navier-Stokes equations. The mixed finite element method that is being described here can be found in [13, 21]. In Subsection 3.1.1 the transport equation

$$\frac{\partial \rho u_k}{\partial t} + \nabla \cdot (\rho u_k \mathbf{u}) = \sum_{i=1}^{d} \frac{\partial}{\partial x_i}\left(\mu \frac{\partial u_k}{\partial x_i}\right) + \rho f_k - \frac{\partial p}{\partial x_k} \tag{4.1}$$

has been derived, which describes the velocity $\mathbf{u}$ and its components u_k. We consider only the stationary (i.e. $\partial/\partial t = 0$) incompressible ($\rho = \textit{constant}$, for simplicity $\rho = 1$) case with $\mu = \nu$. Together with the continuity condition $\nabla \cdot \mathbf{u} = 0$ and $\nabla \cdot (u_k \mathbf{u}) = \mathbf{u} \cdot \nabla u_k + u_k \nabla \cdot \mathbf{u}$, equation (4.1) simplifies to the Navier-Stokes system:

$$\begin{aligned} -\nu \Delta \mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u} + \nabla p &= \mathbf{f} \quad \text{in } \Omega, \\ \nabla \cdot \mathbf{u} &= 0 \quad \text{in } \Omega \end{aligned}$$

with the Dirichlet boundary condition

$$\mathbf{u} = \mathbf{g} \quad \text{on } \Gamma.$$

4.1 The Stokes problem

4.1.1 Variational formulation

A reduced problem which we consider first is the Stokes problem

$$-\nu \Delta \mathbf{u} + \nabla p = \mathbf{f} \quad \text{in} \quad \Omega \subset \mathbb{R}^d, \tag{4.2}$$

$$\nabla \cdot \mathbf{u} = 0 \quad \text{in} \quad \Omega \subset \mathbb{R}^d \tag{4.3}$$

with $d = 2$ or $d = 3$ and with the inhomogeneous boundary condition

$$\mathbf{u} = \mathbf{g} \quad \text{on} \quad \Gamma. \tag{4.4}$$

In general, $\nu(\mathbf{x})$ is a positive function, i.e. $\nu(\mathbf{x}) \geq \nu_0 > 0$, $\forall \mathbf{x} \in \Omega$, with $\nu_0 \in \mathbb{R}_+$, but for simplicity in this work we assume $\nu = \nu_0$.
The Stokes equation (4.2) is used to model the motion of a slowly flowing fluid. The

vectorial unknown $\mathbf{u}$ represents the velocity of the fluid at any given point $\mathbf{x} \in \Omega$, while p represents the pressure.
Integration over Ω of the second equation (4.3) and using integration by parts yields a solvability condition which has to be imposed on the boundary data $\mathbf{g}$:

$$0 = \int_\Omega \nabla \cdot \mathbf{u} \, dx = \int_\Gamma \mathbf{n} \cdot \mathbf{u} \, ds_x = \int_\Gamma \mathbf{n} \cdot \mathbf{g} \, ds_x, \tag{4.5}$$

where $\mathbf{n}$ is the outer unit normal vector. Let $\mathbf{v} \in \left[H^1(\Omega)\right]^d$ be a vectorial test function. We multiply component i of equation (4.2) with the i-th component of $\mathbf{v}$, integrate over Ω and apply integration by parts:

$$\nu \int_\Omega \nabla u_i \cdot \nabla v_i \, dx - \nu \int_\Gamma v_i \left(\nabla u_i \cdot \mathbf{n}\right) \, ds_x + \int_\Omega v_i \partial_i p \, dx = \int_\Omega f_i v_i \, dx. \tag{4.6}$$

Summation over the indices i then gives

$$\nu \int_\Omega \sum_{i=1}^d \nabla u_i \cdot \nabla v_i \, dx - \nu \int_\Gamma \sum_{i=1}^d v_i \left(\nabla u_i \cdot \mathbf{n}\right) \, ds_x + \int_\Omega \sum_{i=1}^d v_i \partial_i p \, dx = \int_\Omega \sum_{i=1}^d f_i v_i \, dx$$

and we get

$$\nu \int_\Omega \nabla \mathbf{u} : \nabla \mathbf{v} \, dx - \nu \int_\Gamma \mathbf{v} \cdot \left(\nabla \mathbf{u} \cdot \mathbf{n}\right) \, ds_x + \int_\Omega \mathbf{v} \cdot \nabla p \, dx = \int_\Omega \mathbf{f} \cdot \mathbf{v} \, dx.$$

Applying integration by parts on the third integral yields

$$\int_\Omega \mathbf{v} \cdot \nabla p \, dx = -\int_\Omega p \nabla \cdot \mathbf{v} \, dx + \int_\Gamma p \mathbf{v} \cdot \mathbf{n} \, ds_x. \tag{4.7}$$

We finally have

$$\nu \int_\Omega \nabla \mathbf{u} : \nabla \mathbf{v} \, dx - \int_\Omega p \nabla \cdot \mathbf{v} \, dx - \nu \int_\Gamma \mathbf{v} \cdot \left(\nabla \mathbf{u} \cdot \mathbf{n}\right) \, ds_x + \int_\Gamma p \mathbf{v} \cdot \mathbf{n} \, ds_x = \int_\Omega \mathbf{f} \cdot \mathbf{v} \, dx. \tag{4.8}$$

If $\mathbf{v}$ is chosen to be a function in $\left[H_0^1(\Omega)\right]^d$, then the two surface integrals in (4.8) over the boundary Γ vanish, which leads to

$$\nu \int_\Omega \nabla \mathbf{u} : \nabla \mathbf{v} \, dx - \int_\Omega p \nabla \cdot \mathbf{v} \, dx = \int_\Omega \mathbf{f} \cdot \mathbf{v} \, dx. \tag{4.9}$$

The weak solution $\mathbf{u}$ needs to be in $\left[H^1(\Omega)\right]^d$ such that the first integral in equation (4.9) exists, while the pressure p needs to be in $L^2(\Omega)$, such that the second integral exists. Considering the first equation in (4.2), it becomes clear that the pressure is unique only up to additive constants $c \in \mathbb{R}$ because of $\nabla(p+c) = \nabla p$. Therefore, a scaling condition must be imposed on the pressure by choosing

$$p \in L_0^2(\Omega) = \left\{ p \in L^2(\Omega) \mid \int_\Omega p(x) \, dx = 0 \right\}. \tag{4.10}$$

Furthermore, in order to incorporate the incompressibility condition (4.3), we multiply equation (4.3) by a test function $q \in L_0^2(\Omega)$ which leads to

$$\int_\Omega q \nabla \cdot \mathbf{u} \, dx = 0.$$

This leads to the follwing mixed variational formulation to find $(\mathbf{u}, p) \in \left[H_g^1(\Omega)\right]^d \times L_0^2(\Omega)$ such that:

$$\begin{aligned} a(\mathbf{u}, \mathbf{v}) - b(\mathbf{v}, p) &= \int_\Omega \mathbf{f} \cdot \mathbf{v} \, dx \\ b(\mathbf{u}, q) &= 0, \end{aligned}$$

is satisfied for all $(\mathbf{v}, q) \in [H_0^1(\Omega)]^d \times L_0^2(\Omega)$, using the bilinear forms $a(\cdot,\cdot) : [H^1(\Omega)]^d \times [H^1(\Omega)]^d \to \mathbb{R}$ and $b(\cdot,\cdot) : [H^1(\Omega)]^d \times L^2(\Omega) \to \mathbb{R}$ which are defined by

$$a(\mathbf{u}, \mathbf{v}) := \nu \int_\Omega \nabla \mathbf{u} : \nabla \mathbf{v} \, dx \quad \text{and} \quad b(\mathbf{v}, q) := \int_\Omega q \nabla \cdot \mathbf{v} \, dx. \tag{4.11}$$

$\left[H_g^1(\Omega)\right]^d$ is the set of all $[H^1(\Omega)]^d$-functions $\mathbf{v}$ satisfying the boundary condition $\mathbf{v} = \mathbf{g}$ on Γ.

Incorporating the Dirichlet boundary condition

The boundary condition $\mathbf{u} = \mathbf{g}$ is usually imposed by choosing an arbitrary but fixed extension $\mathbf{u}_g \in \left[H_g^1(\Omega)\right]^d$ and setting $\mathbf{u} = \mathbf{u}_0 + \mathbf{u}_g$ with $\mathbf{u}_0 \in [H_0^1(\Omega)]^d$. We will use the definitions

$$\begin{aligned} X &:= \left[H_0^1(\Omega)\right]^d, \\ \Pi &:= L_0^2(\Omega) \end{aligned}$$

from here on for shorter notation. This finally leads to the following variational formulation for the Stokes-problem (4.2) which is to seek $(\mathbf{u}_0, p) \in X \times \Pi$, such that:

$$a(\mathbf{u}_0, \mathbf{v}) - b(\mathbf{v}, p) = \int_\Omega \mathbf{f} \cdot \mathbf{v} \, dx - a(\mathbf{u}_g, \mathbf{v}), \tag{4.12}$$

$$b(\mathbf{u}_0, q) = -b(\mathbf{u}_g, q), \tag{4.13}$$

is satisfied for all $(\mathbf{v}, q) \in X \times \Pi$.

Solvability

The linear continuous operator $A : X \to X'$ is defined through the bilinear form $a(\mathbf{u}, \mathbf{v})$:

$$\langle A\mathbf{u}, \mathbf{v} \rangle_{X' \times X} = a(\mathbf{u}, \mathbf{v}) \quad \forall \mathbf{u}, \mathbf{v} \in X.$$

In our case, we have $X = [H_0^1(\Omega)]^d$ and its dual space $X' = [H^{-1}(\Omega)]^d$. Similarly, the operator $B : X \to \Pi'$ and its adjoint operator $B' : \Pi \to X'$ are defined through

$$\langle B\mathbf{v}, q\rangle_{\Pi'\times\Pi} = \langle \mathbf{v}, B'q\rangle_{X\times X'} = b(\mathbf{v}, q) \quad \forall \mathbf{v} \in X, \forall \mathbf{q} \in \Pi.$$

It is then possible to rewrite formulation (4.12)-(4.13) into the following one using operator notation: seek $(\mathbf{u}_0, p) \in X \times \Pi$, such that

$$\langle A\mathbf{u}_0, \mathbf{v}\rangle_{X'\times X} - \langle \mathbf{v}, B'\mathbf{p}\rangle_{X\times X'} = \langle \mathbf{f} - \mathbf{u}_g, \mathbf{v}\rangle_{X'\times X} \tag{4.14}$$
$$\langle B\mathbf{u}_0, q\rangle_{\Pi'\times\Pi} = -\langle B\mathbf{u}_g, q\rangle_{\Pi'\times\Pi} \tag{4.15}$$

is satisfied for all $(\mathbf{v}, q) \in X \times \Pi$.
Solvability of the variational formulation (4.14)-(4.15) follows by the following theorem (see [13, Theorem 1.1] or [38, Theorem 3.11]):

Theorem 4.1. *Let X and Π be Banach spaces and let $A : X \to X'$ and $B : X \to \Pi'$ be bounded operators with boundedness constants c_2^A and c_B, respectively. Further, assume that A is V_0-elliptic,*

$$\langle Au, v\rangle \geq c_1^A \|v\|_X^2 \quad \forall v \in V_0 = kerB = \{\mathbf{v} \in X \mid B\mathbf{v} = 0\},$$

and that the stability condition

$$0 < c_S\|q\|_\Pi \leq \sup_{0\neq v\in X} \frac{\langle Bv, q\rangle_{\Pi'\times\Pi}}{\|v\|_X} \quad \textit{for all } q \in \Pi$$

is satisfied. For $g \in Im_X B$ and $f \in Im_{V_g} A$ there exists a unique solution $(u, p) \in X\times\Pi$ of the variational problem (4.14)-(4.15) satisfying

$$\|u\|_X \leq \frac{1}{c_1^A}\|f\|_{X'} + \left(1 + \frac{c_2^A}{c_1^A}\right) c_B\|g\|_{\Pi'}$$

and

$$\|p\|_\Pi \leq \frac{1}{c_S}\left(1 + \frac{c_2^A}{c_1^A}\right)\left\{\|f\|_{X'} + c_B c_2^A\|g\|_{\Pi'}\right\}.$$

The two bilinear forms $a(\cdot,\cdot)$ and $b(\cdot,\cdot)$ defined by (4.11) are bounded from above with boundedness constants c_2^A and c_B, respectively:

$$a(\mathbf{u}, \mathbf{v}) \leq c_2^A\|\mathbf{u}\|_{H^1(\Omega)}\|\mathbf{v}\|_{H^1(\Omega)} \quad \text{and} \quad b(\mathbf{v}, \mathbf{q}) \leq c_B\|\mathbf{v}\|_{H^1(\Omega)}\|\mathbf{q}\|_{L^2(\Omega)}.$$

In order to show solvability of (4.14)-(4.15) using Theorem 4.1, we also need to prove the ellipticity of $a(\cdot,\cdot)$ and the validity of the stability condition.
The kernel of B is defined as

$$\ker B := \{\mathbf{v} \in X \mid b(\mathbf{v}, q) = 0, \forall q \in \Pi\} \subset X.$$

In order to show ellipticity of A, i.e. ellipticity of the bilinear form $a(\cdot,\cdot)$, we use norm equivalence. Since $\mathbf{v} \in X$, there exists a constant c such that $\|\mathbf{v}\|_{[H^1(\Omega)]^d,\Gamma} \geq$

$c\|\mathbf{v}\|_{[H^1(\Omega)]^d}$ by the Norm Equivalence Theorem of Sobolev, see [38, Theorem 2.6]. Then ellipticity follows:

$$\begin{aligned} a(\mathbf{v},\mathbf{v}) &= \nu \int_\Omega \nabla\mathbf{v} : \nabla\mathbf{v}\, dx = \nu \sum_{i=1}^d |v_i|^2_{H^1(\Omega)} \\ &= \underbrace{\nu \sum_{i=1}^d \left\{ \left[\int_\Gamma v_i\, ds_x\right]^2 + |v_i|^2_{[H^1(\Omega)]^d} \right\}}_{:=\|\mathbf{v}\|^2_{[H^1(\Omega)]^d,\Gamma}} \geq \nu c \|\mathbf{v}\|^2_{[H^1(\Omega)]^d}. \end{aligned}$$

The proof of the stability condition, i.e. that there exists a constant c_S such that

$$c_S \|q\|_{L^2(\Omega)} \leq \sup_{0\neq\mathbf{v}\in X} \frac{\int_\Omega q \operatorname{div}\mathbf{v}\, dx}{\|\mathbf{v}\|_{[H^1(\Omega)]^d}} \quad \text{for all } q \in \Pi, \tag{4.16}$$

can be found in [38, Lemma 4.20].

The scaling condition

A convenient way to incorporate the scaling condition (4.10) into the variational formulation (4.12)-(4.13) without the need to use the space $L^2_0(\Omega)$ is to introduce a scalar Lagrange multiplier $\lambda \in \mathbb{R}$. We define $\Pi := L^2(\Omega)$ and consider an extended saddle point problem to find $\mathbf{u} \in \left[H^1_g(\Omega)\right]^d$ as well as $p \in \Pi$ and $\lambda \in \mathbb{R}$ such that

$$a(\mathbf{u},\mathbf{v}) - \int_\Omega p\nabla\cdot\mathbf{v}\, dx = \langle \mathbf{f},\mathbf{v}\rangle_\Omega, \tag{4.17}$$

$$\int_\Omega q\nabla\cdot\mathbf{u}\, dx + \lambda\int_\Omega q\, dx = 0, \tag{4.18}$$

$$\int_\Omega p\, dx = 0 \tag{4.19}$$

is satisfied for all $\mathbf{v} \in X$ and $q \in \Pi$. Choosing as test function $q \equiv 1$ and using the solvability condition (4.5) yields

$$\lambda|\Omega| = -\int_\Omega \nabla\cdot\mathbf{u}\, dx = -\int_\Gamma \mathbf{n}\cdot\mathbf{g}\, ds_x = 0$$

which leads to $\lambda = 0$. Therefore equation (4.19) can be rewritten as

$$\int_\Omega p\, dx - \lambda = 0.$$

The Lagrange multiplier λ can then be eliminated and finally a modified variational formulation follows which is to find $\mathbf{u}_0 \in X$ and $p \in \Pi$ such that

$$a(\mathbf{u}_0,\mathbf{v}) - b(\mathbf{v},p) = \int_\Omega \mathbf{f}\cdot\mathbf{v}\, dx - a(\mathbf{u}_g,\mathbf{v}) \tag{4.20}$$

$$b(\mathbf{u}_0,q) + \int_\Omega p\, dx \int_\Omega q\, dx = -b(\mathbf{u}_g,q) \tag{4.21}$$

is satisfied for all $\mathbf{v} \in X$ and $q \in \Pi$. Solvability of this formulation is shown in [38] and follows through equivalence with (4.12)-(4.13) by setting $q = 1$.

4.1.2 Discretization

For the approximation of the solution of (4.2) the variational formulation (4.20)-(4.21) is discretized by using

$$X_h = [P_{k_u}]^d \cap X \tag{4.22}$$

and

$$\Pi_h = P_{k_p} \cap \Pi \tag{4.23}$$

as ansatz and test spaces with $X = [H_0^1(\Omega)]^d$ and $\Pi = L^2(\Omega)$, k_u as piecewise polynomial degree for the approximation of the velocity $\mathbf{u}$ and k_p as the polynomial degree for the pressure p. The index h refers to the global mesh size of the triangulation $\mathcal{T}$. The space P_k is defined as the Lagrange finite element space of continuous piecewise polynomials of degree k:

$$P_k := \left\{ u \in C^0(\mathcal{T}) \mid u_{|T} \in \mathcal{P}_k(T),\, \forall T \in \mathcal{T} \right\}.$$

It is defined for a triangulation $\mathcal{T}$ as described in Chapter 2. Furthermore, X_h and Π_h are conforming since $X_h \subset X$ and $\Pi_h \subset \Pi$. The Galerkin variational problem of (4.20)-(4.21) then is to find $(\mathbf{u}_{0,h}, p_h) \in X_h \times \Pi_h$ such that:

$$a(\mathbf{u}_{0,h}, \mathbf{v}_h) - b(\mathbf{v}_h, p_h) = \int_\Omega \mathbf{f} \cdot \mathbf{v}_h \, dx - a(\mathbf{u}_g, \mathbf{v}_h) \tag{4.24}$$

$$b(\mathbf{u}_{0,h}, q_h) = -b(\mathbf{u}_g, q_h), \tag{4.25}$$

is satisfied for all $(\mathbf{v}_h, q_h) \in X_h \times \Pi_h$.
In order to show solvability of the variational formulation (4.24)-(4.25), Theorem 4.1 can be used. Boundedness of the bilinear forms $a(\cdot,\cdot)$ and $b(\cdot,\cdot)$ follows simply by conformity of the spaces Π_h and X_h. Also the ker B-ellipticity on X_h follows with some simple steps if $a(\cdot,\cdot)$ is ker B-elliptic on X. Furthermore, the stability condition needs to hold for the discrete spaces X_h and Π_h such that

$$c_S \|q_h\|_\Pi \leq \sup_{0 \neq \mathbf{v}_h \in X_h} \frac{b(\mathbf{v}_h, q_h)}{\|\mathbf{v}_h\|_X}, \tag{4.26}$$

but this does not follow immediately through conformity. Instead, the discrete stability condition (4.26) can be shown for a pair $X_h \times \Pi_h$ by using Fortin's criterion:

Proposition 4.2. *Suppose that the continuous stability condition (4.16) holds and assume that there exists a uniformly continuous operator $P : X \to X_h$ satisfying*

$$\begin{cases} b(P\mathbf{v} - \mathbf{v}, q_h) = 0, & \forall q_h \in \Pi_h \\ \|P\mathbf{v}\|_X \leq c\|\mathbf{v}\|_X \end{cases}$$

with c independent of h. Then the discrete stability condition (4.26) holds.

The proof of Proposition 4.2 can be found in [13], as well as the proof of the stability for a selection of finite element pairs. Some of those pairs are presented in the next section.

Stable finite element spaces

Two common choices for finite-element spaces yielding unconditionally stable approximations are:

- The Taylor-Hood element: Choose $k_u \geq 2$ in (4.22) for the elementwise polynomial degree for the piecewise continuous approximation of the velocities, and one order less $k_p = k_u - 1$ in (4.23) for the piecewise continuous approximation of the pressure.
- The mini-element: For the mini-element, the space of piecewise linear functions P_1 enriched by the bubble-functions is used as ansatz and test space. The bubble-functions are the elements of the space

$$\mathcal{B}_h := \operatorname{span}\left\{\lambda_{T,1}\lambda_{T,2}\lambda_{T,3} \mid T \in \mathcal{T}_h\right\}, \tag{4.27}$$

  where $\lambda_{T,1}$, $\lambda_{T,2}$ and $\lambda_{T,3}$ are the barycentric coordinates defined on $T \in \mathcal{T}_h$. The ansatz and test spaces X_h and Π_h can then be defined by:

$$X_h := \left[P_1 \cap H_0^1(\Omega) \oplus \mathcal{B}_h\right]^d \quad \text{and} \quad \Pi_h := P_1 \cap \Pi.$$

4.1.3 A priori error estimates

For inf-sup stable and conforming finite element pairs $X_h \times \Pi_h \subset X \times \Pi$, we can conduct an a priori error analysis by standard arguments. We start by using best approximation results. The following results together with proofs can be found in [13] for mixed finite element methods for the Stokes problem (4.2).

Theorem 4.3. *Let $X_h \times \Pi_h \subset X \times \Pi$ be a stable finite element pair, $(\mathbf{u}_h, p_h) \in X_h \times \Pi_h$ with $\mathbf{u}_h := \mathbf{u}_{h,0} + \mathbf{u}_g$ as the solution of (4.24)-(4.25) and h the global mesh size of $\mathcal{T}$. Then there holds*

$$|\mathbf{u} - \mathbf{u}_h|_{H^1(\Omega)} + \|p - p_h\|_{L^2(\Omega)} \leq c\left(\inf_{\mathbf{v}_h \in X_h} |\mathbf{u} - \mathbf{v}_h|_{H^1(\Omega)} + \inf_{q_h \in \Pi_h} \|p - q_h\|_{L^2(\Omega)}\right),$$

where the constant $c > 0$ depends on the inf-sup constant c_S in (4.16). Further, if Ω is a convex domain, there holds

$$\|\mathbf{u} - \mathbf{u}_h\|_{L^2(\Omega)} \leq ch\left(\inf_{\mathbf{v}_h \in X_h} |\mathbf{u} - \mathbf{v}_h|_{H^1(\Omega)} + \inf_{q_h \in \Pi_h} \|p - q_h\|_{L^2(\Omega)}\right).$$

We arrive at an upper bound for the best approximation by substituting $\mathbf{v}_h$ with an arbitrary function in X_h instead of taking the infimum, particularly when substituting with $I_h^m \mathbf{u}_h$, where $I_h^m : C(\overline{\Omega}) \to X_h$ is the Lagrangian interpolation operator into the space of continuous piecewise polynomials of degree m:

$$\begin{aligned}
\inf_{\mathbf{v}_h \in X_h} |\mathbf{u} - \mathbf{v}_h|_{H^1(\Omega)} &\leq |\mathbf{u} - I_h^{m_u}\mathbf{u}|_{H^1(\Omega)} \leq ch^{k_u}|\mathbf{u}|_{H^{k_u+1}(\Omega)}, \\
\inf_{q_h \in \Pi_h} \|p - q_h\|_{L^2(\Omega)} &\leq \|p - I_h^{m_p} p\|_{L^2(\Omega)} \leq h^{k_p+1}|p|_{H^{k_p+1}(\Omega)}.
\end{aligned}$$

We use m_u-th order interpolation for the velocity $\mathbf{u}$ and m_p-th order interpolation for the pressure p. In the last step of each of the above two inequalities, [13, Proposition 3.6] was used, for which we have to assume $\mathbf{u} \in H^{k_u+1}(\Omega)$ or $p \in H^{k_p+1}(\Omega)$, with $m_u \geq k_u \geq 1$ and $m_p \geq k_p \geq 1$. These results then give:

$$\inf_{\mathbf{v}_h \in X_h} |\mathbf{u} - \mathbf{v}_h|_{H^1(\Omega)} + \inf_{q_h \in \Pi_h} \|p - q_h\|_{L^2(\Omega)} \leq c \left(h^{k_u} |\mathbf{u}|_{H^{k_u+1}(\Omega)} + h^{k_p+1} |p|_{H^{k_p+1}(\Omega)} \right).$$

In the case of using Taylor-Hood elements (see Section 4.1.2) and especially in the case $m_u = 2$ and $m_p = 1$ in view of the numerical investigations in Chapter 7, we achieve a final convergence rate of $\mathcal{O}(h^2)$ for the errors $\|p - p_h\|_{L^2(\Omega)}$ and $|\mathbf{u} - \mathbf{u}_h|_{H^1(\Omega)}$ and $\mathcal{O}(h^3)$ for $\|\mathbf{u} - \mathbf{u}_h\|_{L^2(\Omega)}$, assuming that $\mathbf{u} \in H^3(\Omega)$ and $p \in H^2(\Omega)$:

$$|\mathbf{u} - \mathbf{u}_h|_{H^1(\Omega)} + \|p - p_h\|_{L^2(\Omega)} \leq ch^3 \left(|\mathbf{u}|_{H^3(\Omega)} + |p|_{H^2(\Omega)} \right).$$

4.1.4 Stabilization of equal-order approximations

In the case of an appoximation by equal-order polynomials for both the velocity and the pressure, i.e. $k_u = k_p$ in (4.22) and (4.23), the stability condition does not hold. In this subsection, a method to stabilize such choices of spaces $X_h \times \Pi_h$ is given. Although the method works for arbitrary polynomial degrees $k \geq 1$, we restrict ourselves here to $k_u = k_p = 1$ since this is also the degree with which the implementation was tested. In [21] it was shown that the inf-sup-condition is valid if and only if the pressure space coincides with the range of the divergence operator. This is also the case for a discrete inf-sup-condition. The range of the divergence is a polynomial space of one degree less than the pressure space for equal-order approximations, which is why such pairs of spaces would never fulfill the discrete inf-sup-condition.
In the stabilization approach proposed in [15], a local pressure projection term is added.
Let

$$P_m^{DC} = \left\{ q_h \in L^2(\Omega) \mid q_{h|T} \in \mathcal{P}_m(T),\ \forall T \in \mathcal{T} \right\}$$

be the space of elementwise polynomials of degree $m \geq 0$ and

$$X_h = [P_1]^d \cap X \quad \text{and} \quad \Pi_h = P_1 \cap \Pi$$

an equal-order pair of trial and ansatz spaces for velocity and pressure. For a function $q \in L^2(\Omega)$ the projection operator $\rho_m : L^2(\Omega) \to P_m^{DC}$ is defined through:

$$\int_\Omega r_h(\rho_m q - q)\, dx = 0 \quad \forall r_h \in P_m^{DC}. \tag{4.28}$$

This uncouples into local elementwise projection problems, such that

$$\int_T r_h(\rho_m q - q)\, dx = 0 \quad \forall r_h \in P_m^{DC}, \quad \forall T \in \mathcal{T}.$$

Considering that each element $\mathbf{v}_h \in X_h$ is a continuous piecewise linear polynomial, it becomes clear that the divergence of $\mathbf{v}_h$ is a piecewise constant function, i.e.

$$\nabla \cdot \mathbf{v}_h \in P_0^{DC} \quad \forall \mathbf{v}_h \in X_h.$$

Therefore, using the definition (4.28), it follows that

$$\int_\Omega (q_h - \rho_0 q_h) \nabla \cdot \mathbf{v}_h \; dx = 0$$

and consequently

$$b(\mathbf{v}_h, \rho_0 q_h) = b(\mathbf{v}_h, q_h), \tag{4.29}$$

which is why solving the variational formulation (4.24)-(4.25) using the projections $\rho_0 p_h$ and $\rho_0 q_h$ instead of p_h and q_h is the same as solving without using the projections. Introduction of those projections leads to an imbalance between the desired linear order of pressure approximation and represent constant pressure approximation. This imbalance can be penalized by a term

$$c(p_h, q_h) = \frac{1}{\nu} \int_\Omega (p_h - \rho_0 p_h)(q_h - \rho_0 q_h) \; dx.$$

Adding $-c(p_h, q_h)$ to (4.25) and substitution of p_h, q_h by $\rho_0 p_h$, $\rho_0 q_h$ leads to the following variational problem, which is to find $(\mathbf{u}_{0,h}, p_h) \in X_h \times \Pi_h$ such that:

$$\begin{aligned} a(\mathbf{u}_{0,h}, \mathbf{v}_h) - b(\mathbf{v}_h, \rho_0 p_h) &= \int_\Omega \mathbf{f} \cdot \mathbf{v}_h \; dx - a(\mathbf{u}_g, \mathbf{v}_h) \\ b(\mathbf{u}_{0,h}, \rho_0 q_h) - c(p_h, q_h) &= -b(\mathbf{u}_g, \rho_0 q_h), \end{aligned}$$

is satisfied for all $(\mathbf{v}_h, q_h) \in X_h \times \Pi_h$. Finally, using (4.29), this leads to the following variational formulation, which is to seek $(\mathbf{u}_{0,h}, p_h) \in X_h \times \Pi_h$, such that:

$$a(\mathbf{u}_{0,h}, \mathbf{v}_h) - b(\mathbf{v}_h, p_h) = \int_\Omega \mathbf{f} \cdot \mathbf{v}_h \; dx - a(\mathbf{u}_g, \mathbf{v}_h) \tag{4.30}$$

$$b(\mathbf{u}_{0,h}, q_h) - c(p_h, q_h) = -b(\mathbf{u}_g, q_h), \tag{4.31}$$

is satisfied for all $(\mathbf{v}_h, q_h) \in X_h \times \Pi_h$.
Unique solvability of the variational formulation (4.30)-(4.31) can be established by Theorem II.1.2 in [13].
A proof of the stability of the pressure projection method can be found in [12], as well as the following corollary concerning its convergence, thus proving the method's validity as a solution method for the Stokes equation (4.2):

Corollary 4.4. *Let* $(\mathbf{u}, p) \in H_0^1(\Omega) \cap H^2(\Omega) \times L_0^2(\Omega) \cap H^1(\Omega)$ *be a solution of the Stokes problem (4.2)-(4.3) and let* $(\mathbf{u}_{0,h}, p_h)$ *be the solution of the stabilized problem (4.30)-(4.31), where the projection* ρ_0 *satisfies*

$$\|\rho_0 p\|_{L^2(\Omega)} \leq C \|p\|_{L^2(\Omega)} \quad \forall p \in L^2(\Omega)$$

and

$$\|p - \rho_0 p\|_{L^2(\Omega)} \leq Ch\|p\|_{H^1(\Omega)} \quad \forall p \in H^1(\Omega).$$

Then,

$$\|\mathbf{u} - \mathbf{u}_h\|_{H^1(\Omega)} + \|p - p_h\|_{L^2(\Omega)} \leq Ch\left(\|\mathbf{u}\|_{H^2(\Omega)} + \|p\|_{H^1(\Omega)}\right).$$

A proof that the assumptions of Corollary 4.4 are satisfied can be found in [12].

4.2 The Navier-Stokes problem

In this section, a finite element approach based on [20] for the solution of the stationary incompressible Navier-Stokes problem

$$-\nu\Delta\mathbf{u} + \mathbf{u}\cdot\nabla\mathbf{u} + \nabla p = \mathbf{f} \quad \text{in } \Omega, \tag{4.32}$$

$$\nabla\cdot\mathbf{u} = 0 \quad \text{in } \Omega \tag{4.33}$$

with the Dirichlet boundary condition

$$\mathbf{u} = \mathbf{g} \quad \text{on } \Gamma,$$

again underlying the solvability condition (4.5), is being presented.

4.2.1 Variational formulation

The discretization of the nonlinear convection-term $\mathbf{u}\cdot\nabla\mathbf{u}$ appearing in the Navier-Stokes equations leads, proceeding in the same way as in Section 4.1, but considering the convection-term and defining the corresponding trilinear form

$$n(\mathbf{u}, \mathbf{v}, \mathbf{w}) := \int_\Omega (\mathbf{u}\cdot\nabla\mathbf{v})\,\mathbf{w}\,dx,$$

to the following variational problem, which is to find $(\mathbf{u}_0, p) \in X \times \Pi$ such that:

$$a(\mathbf{u}_0, \mathbf{v}) + n(\mathbf{u}_0 + \mathbf{u}_g, \mathbf{u}_0 + \mathbf{u}_g, \mathbf{v}) - b(\mathbf{v}, p) = \tilde{F}(\mathbf{f}, \mathbf{u}_g, \mathbf{v}) \tag{4.34}$$

$$b(\mathbf{u}_0, q) = -b(\mathbf{u}_g, q), \tag{4.35}$$

is satisfied for all $(\mathbf{v}, q) \in X \times \Pi$ with

$$\tilde{F}(\mathbf{f}, \mathbf{u}_g, \mathbf{v}) = \int_\Omega \mathbf{f}\cdot\mathbf{v}\,dx - a(\mathbf{u}_g, \mathbf{v}),$$

where $\mathbf{u}_g \in \left[H^1_g(\Omega)\right]^d$ is an extension of the boundary function $\mathbf{g} \in \left[H^{1/2}(\Gamma)\right]^d$ to the domain Ω. The final solution to the Navier-Stokes problem is represented by $\mathbf{u} = \mathbf{u}_0 + \mathbf{u}_g$. The term $n(\cdot,\cdot,\cdot)$ cannot be treated directly, which is why we employ the iterative Newton-algorithm [21, 44] in order to linearize the Navier-Stokes problem, leading to the modified Stokes problem

$$-\nu\Delta\mathbf{u}^{i+1} + \nabla p^{i+1} + \left[\mathbf{u}^i \cdot \nabla\mathbf{u}^{i+1} + \mathbf{u}^{i+1} \cdot \nabla\mathbf{u}^i\right] = \mathbf{f} + \mathbf{u}^i \cdot \nabla\mathbf{u}^i \quad \text{in } \Omega, \tag{4.36}$$

$$\nabla \cdot \mathbf{u}^{i+1} = 0 \quad \text{in } \Omega, \tag{4.37}$$

$$\mathbf{u}^{i+1} = \mathbf{g} \quad \text{on } \Gamma_D, \tag{4.38}$$

where the superindices i or $i+1$ represent solutions at iterations i or $i+1$, respectively. By setting $\mathbf{u}^{i+1} = \mathbf{u}_0^{i+1} + \mathbf{u}_g$ and $\mathbf{u}^i = \mathbf{u}_0^i + \mathbf{u}_g$ (the extension $\mathbf{u}_g$ does not change during the iterations, because $\mathbf{g}$ also does not change), the terms $\mathbf{u}^i \cdot \nabla\mathbf{u}^{i+1}$ and $\mathbf{u}^{i+1} \cdot \nabla\mathbf{u}^i$ in (4.36) lead to the following terms appearing in the variational formulation:

$$n(\mathbf{u}_0^{i+1} + \mathbf{u}_g, \mathbf{u}_0^i + \mathbf{u}_g, \mathbf{v}) = n(\mathbf{u}_0^{i+1}, \mathbf{u}^i, \mathbf{v}) + n(\mathbf{u}_g, \mathbf{u}^i, \mathbf{v}), \tag{4.39}$$

$$n(\mathbf{u}_0^i + \mathbf{u}_g, \mathbf{u}_0^{i+1} + \mathbf{u}_g, \mathbf{v}) = n(\mathbf{u}^i, \mathbf{u}_0^{i+1}, \mathbf{v}) + n(\mathbf{u}^i, \mathbf{u}_g, \mathbf{v}). \tag{4.40}$$

The terms $n(\mathbf{u}_g, \mathbf{u}^i, \mathbf{v})$ and $n(\mathbf{u}^i, \mathbf{u}_g, \mathbf{v})$ only contain results from the old iteration i, so we subtract them such that they appear on the right hand side F of the variational equations. Incorporating the scaling condition as described in Section 4.1.1 finally leads to the following variational formulation with the scaling condition incorporated, for the $i+1$-th Newton-iteration of the time-independent Navier-Stokes problem for incompressible flow with pure Dirichlet boundary conditions, which is to seek $(\mathbf{u}_0^{i+1}, p^{i+1}) \in X \times \Pi$, such that

$$a(\mathbf{u}_0^{i+1}, \mathbf{v}) + n(\mathbf{u}_0^{i+1}, \mathbf{u}^i, \mathbf{v}) + n(\mathbf{u}^i, \mathbf{u}_0^{i+1}, \mathbf{v}) - b(\mathbf{v}, p) = F(\mathbf{f}, \mathbf{u}^i, \mathbf{v}) \tag{4.41}$$

$$b(\mathbf{u}_0^{i+1}, q) + \int_\Omega p\, dx \int_\Omega q\, dx = -b(\mathbf{u}_g, q), \tag{4.42}$$

is satisfied for all $(\mathbf{v}, q) \in X \times \Pi$ with

$$F(\mathbf{f}, \mathbf{u}^i, \mathbf{v}) = \int_\Omega \mathbf{f} \cdot \mathbf{v}\, dx - a(\mathbf{u}_g, \mathbf{v}) - n(\mathbf{u}_g, \mathbf{u}^i, \mathbf{v}) - n(\mathbf{u}^i, \mathbf{u}_g, \mathbf{v}) + n(\mathbf{u}^i, \mathbf{u}^i, \mathbf{v}). \tag{4.43}$$

Any of the proposed choices for test and ansatz spaces in Section 4.1.2 can be applied to discretize problem (4.41)-(4.42).

Remark 4.1. *Newton's algorithm can be found in [21, 44], it converges quadratically under the assumption that the initial guess $\mathbf{u}^0$ is good enough. If $(\mathbf{u}, p)$ is a solution of (4.32)-(4.33), then the Newton-iteration scheme defines a sequence $(\mathbf{u}^i, p^i)$ that converges to $(\mathbf{u}, p)$. The solution of the Stokes equations can be used as an initial guess $\mathbf{u}^0$. As a termination criterion, a positive real number ϵ has to be chosen. The algorithm stops if $\|\mathbf{u}^{i+1} - \mathbf{u}^i\|_0 \leq \epsilon$.*

Remark 4.2. *For very fast flows (see Reynolds' number [33], which is described in more detail in Chapter 7), the nonlinear term $\mathbf{u} \cdot \nabla\mathbf{u}$ becomes dominant in comparison to the diffusion term $-\nu\Delta\mathbf{u}$, such that the solution of the Stokes equations which* *lack the nonlinear term may not be good enough, i.e. the algorithm diverges. In that case, damped Newton iterations can be employed. That means that the nonlinear terms are multiplied by an increasing sequence* $0 < c_D^{i+1} \leq 1$ *of coefficients. The factor* c_D^0 *is chosen to be zero (then the problem is reduced to the Stokes problem) and* c_D^{i+1} *for consecutive iterations* $i+1$ *is increased for each iteration in such a way that* $c_D^i < c_D^{i+1} \leq 1$*. The damping factor serves to slowly introduce the nonlinearity. However, the final solution is only attained if* $c_D^{i+1} = 1$ *and* $\|\mathbf{u}^{i+1} - \mathbf{u}^i\|_0 \leq \epsilon$*.*

4.2.2 Solvability

Solvability of variational formulations for the Navier-Stokes equations is more difficult to show than for the Stokes equations, where no nonlinear term was present. In most cases, only the existence of a solution can be shown. Investigations on existence and uniqueness of solutions can be found in [21, 40]. The following theorem from [21] is a statement about the existence of a solution of variational formulation (4.34)-(4.35):

Theorem 4.5. *Given* $\mathbf{f} \in [H^{-1}(\Omega)]^d$ *and let* $\mathbf{u}_g \in \left[H_g^1(\Omega)\right]^d$ *be an arbitrary but fixed extension of* $\mathbf{g} \in \left[H^{1/2}(\Gamma)\right]^d$ *satisfying the solvability condition (4.5), such that* $\mathbf{u} = \mathbf{u}_0 + \mathbf{u}_g$*, then there exists at least one pair* $(\mathbf{u}_0, p) \in X \times \Pi$ *which is a solution of (4.34)-(4.35). Furthermore, if for*

$$\mathcal{N} = \sup_{\mathbf{u},\mathbf{v},\mathbf{w}\in V} \frac{n(\mathbf{u},\mathbf{v},\mathbf{w})}{|\mathbf{u}|_{H^1(\Omega)}|\mathbf{v}|_{H^1(\Omega)}|\mathbf{w}|_{H^1(\Omega)}} \quad \text{and} \quad \|\mathbf{f}\|_{V'} = \sup_{\mathbf{v}\in V} \frac{\langle \mathbf{f}, \mathbf{v}\rangle}{|\mathbf{v}|_{H^1(\Omega)}}$$

with $V = \left\{\mathbf{v} \in [H_0^1(\Omega)]^d,\ \nabla\cdot\mathbf{v} = 0\right\}$ *the condition*

$$(\mathcal{N}/\nu^2)\|\mathbf{f}\|_{V'} < 1$$

holds, then problem (4.34)-(4.35) possesses a unique solution $(\mathbf{u}_0, p) \in X \times \Pi$*.*

4.2.3 Linear system

This subsection is about how the variational formulations from the previous sections can be transformed into linear systems of algebraic equations for $d = 2$. We set $\mathbf{u}_h^i = \mathbf{u}_{0,h}^i + \mathbf{u}_g$. Discretizing (4.41)-(4.42) and employing pressure scaling as in (4.20)-(4.21) leads to the following variational formulation, which is to find $(\mathbf{u}_{0,h}^{i+1}, p_h^{i+1}) \in X_h \times \Pi_h$, such that

$$a(\mathbf{u}_{0,h}^{i+1}, \mathbf{v}_h) + n(\mathbf{u}_{0,h}^{i+1}, \mathbf{u}_h^i, \mathbf{v}_h) + n(\mathbf{u}_h^i, \mathbf{u}_{0,h}^{i+1}, \mathbf{v}_h) - b(\mathbf{v}_h, p_h^{i+1}) = F(\mathbf{f}, \mathbf{u}_h^i, \mathbf{v}_h) \quad (4.44)$$

$$b(\mathbf{u}_{0,h}^{i+1}, q_h) + \int_\Omega p_h^{i+1}\, dx \int_\Omega q_h\, dx = -b(\mathbf{u}_g, q_h), \quad (4.45)$$

is satisfied for all $(\mathbf{v}_h, q_h) \in X_h \times \Pi_h$ with the right hand side F defined as in (4.43). We use

$$X_h := [P_{k_u}]^d = \left[\text{span}\left\{\varphi_j^{k_u}\right\}_{j=1}^{D_0}\right]^d = \text{span}\left\{\begin{pmatrix}\varphi_j^{k_u}\\0\end{pmatrix}, \begin{pmatrix}0\\\varphi_j^{k_u}\end{pmatrix}\right\}_{j=1}^{D_0} \subset \left[H_0^1(\Omega)\right]^2,$$

$$\Pi_h := P_{k_p} = \text{span}\left\{\varphi_j^{k_p}\right\}_{j=1}^{D_p} \subset L^2(\Omega)$$

as discrete ansatz and test spaces for the velocity $\mathbf{u}_{0,h}^{i+1}$ and the pressure p_h^{i+1}, where P_k is the space of continuous piecewise polynomials of degree k. The order of approximation for the velocity is k_u and k_p for the pressure. For Taylor-Hood elements, we choose $k_u = 2$ and $k_p = 1$. Here, φ_j^k are the basis functions of k-th order spanning the space P_k.
We denote by D_u the total number of degrees of freedom for the velocity $\mathbf{u}_h^{i+1}$, while D_Γ denotes the number of degrees of freedom with fixed values, determined by the boundary condition. Then D_0 denotes the remaining non-fixed degrees of freedom, such that $D_u = D_0 + D_\Gamma$. The number of degrees of freedom for the pressure is denoted by D_p. In the case of linear approximation ($k = 1$), the degrees of freedom correspond to the nodes $a \in \mathcal{N}$. For quadratic approximations ($k = 2$), the degrees of freedom correspond to the nodes $a \in \mathcal{N}$ and additionally the midpoints of the edges $E \in \mathcal{E}$.
The discrete solutions $\mathbf{u}_h$ and p_h of the $i+1$-th Newton iteration can then be written as

$$\mathbf{u}_h^{i+1}(\mathbf{x}) = \sum_{j=1}^{D_u}\left\{u_j^{i+1}\begin{pmatrix}\varphi_j^{k_u}(\mathbf{x})\\0\end{pmatrix} + v_j^{i+1}\begin{pmatrix}0\\\varphi_j^{k_u}(\mathbf{x})\end{pmatrix}\right\} \quad \text{and} \quad p_h^{i+1}(\mathbf{x}) = \sum_{j=1}^{D_p} p_j^{i+1}\varphi_j^{k_p}(\mathbf{x})$$

with coefficient vectors $\underline{u}^{i+1} \in \mathbb{R}^{D_u}$ and $\underline{v}^{i+1} \in \mathbb{R}^{D_u}$ for the approximation of the velocity in the x- and y- direction, respectively, and the pressure coefficient vector $\underline{p}^{i+1} \in \mathbb{R}^{D_p}$.

Definition of the extension $\mathbf{u}_g$ through interpolation

Let $\mathbf{g} \in \left[H^{1/2}(\Gamma)\right]^d$ be the given Dirichlet datum. By the inverse trace theorem [38, Theorem 2.22] there exists a bounded extension $\mathcal{E}\mathbf{g} \in \left[H^1(\Omega)\right]^d$ and a constant c such that

$$\|\mathcal{E}\mathbf{g}\|_{H^1(\Omega)} \leq c\|\mathbf{g}\|_{H^{1/2}(\Gamma)}$$

and $\mathcal{E}\mathbf{g}|_\Gamma = \mathbf{g}$. We define a piecewise polynomial interpolation (of same order as the approximation of the velocity) on Ω of $\mathcal{E}\mathbf{g}$:

$$\mathbf{u}_g(\mathbf{x}) = (I_h\mathcal{E}\mathbf{g})(\mathbf{x}) \tag{4.46}$$

Then we set the values of the extension $\mathbf{u}_g(\mathbf{x})$ at the Dirichlet boundary simply to the values of the Dirichlet datum such that $\mathbf{u}_g(\mathbf{x}_j)_{|\Gamma} = \mathbf{g}(\mathbf{x}_j)$ for all $j = D_0 + 1, \ldots, D_u$.

Thus the values of the extension are fixed for degrees of freedom corresponding to a Dirichlet boundary condition, and we can consider a splitting of the coefficient vectors $\underline{u}^{i+1} = \underline{u}_0^{i+1} + \underline{u}_g$ and $\underline{v}^{i+1} = \underline{v}_0^{i+1} + \underline{v}_g$ into variable parts and parts fixed by the boundary condition.

Final system

Seeking a solution of the Galerkin variational formulation (4.44)-(4.45) is equivalent to seeking the solution vectors $\underline{s} = (\underline{u}_0^{i+1}, \underline{v}_0^{i+1}, \underline{p}^{i+1}) \in \mathbb{R}^{D_0} \times \mathbb{R}^{D_0} \times \mathbb{R}^{D_p}$ of the linear equation system $M\underline{s} = \underline{r}$:

$$\underbrace{\begin{bmatrix} A + A_1 + M_{00} & M_{10} & -B_1^\top \\ M_{01} & A + A_1 + M_{11} & -B_2^\top \\ B_1 & B_2 & \underline{a} \cdot \underline{a}^\top \end{bmatrix}}_{:=M} \underbrace{\begin{bmatrix} \underline{u}_0^{i+1} \\ \underline{v}_0^{i+1} \\ \underline{p}^{i+1} \end{bmatrix}}_{\underline{s}} = \underbrace{\begin{bmatrix} \underline{F}_1 \\ \underline{F}_2 \\ \underline{b} \end{bmatrix}}_{\underline{r}}, \tag{4.47}$$

with

$$\begin{aligned} A[j,\ell] &:= \int_\Omega \nabla\varphi_j \cdot \nabla\varphi_\ell \, dx, \\ A_1[j,\ell] &:= \int_\Omega \left(\mathbf{u}_h^i \cdot \nabla\varphi_\ell\right) \varphi_j \, dx, \\ M_{00}[j,\ell] &:= \int_\Omega \frac{\partial u_h^i}{\partial x} \varphi_j \varphi_\ell \, dx, \\ M_{01}[j,\ell] &:= \int_\Omega \frac{\partial u_h^i}{\partial y} \varphi_j \varphi_\ell \, dx, \\ M_{10}[j,\ell] &:= \int_\Omega \frac{\partial v_h^i}{\partial x} \varphi_j \varphi_\ell \, dx, \\ M_{11}[j,\ell] &:= \int_\Omega \frac{\partial v_h^i}{\partial y} \varphi_j \varphi_\ell \, dx \end{aligned}$$

for $\ell, j = 1, \ldots, D_0$. The functions u_h^i and v_h^i are the x- and y-components of the velocity $\mathbf{u}_h^i$. Furthermore, we define

$$\begin{aligned} B_1[k,\ell] &:= \int_\Omega \psi_k \frac{\partial\varphi_\ell}{\partial x} \, dx, \\ B_2[k,\ell] &:= \int_\Omega \psi_k \frac{\partial\varphi_\ell}{\partial y} \, dx, \\ \underline{a}[k] &:= \int_\Omega \varphi_k \, dx, \end{aligned}$$

for $\ell = 1, \dots, D_0$ and $k = 1, \dots, D_p$. The right hand sides of system (4.47) are given by

$$\begin{aligned}
F_1[j] &:= \int_\Omega \begin{pmatrix} \varphi_j \\ 0 \end{pmatrix} \cdot \Big[\mathbf{f} - \nabla \left(\mathbf{u}_g\right)_x - \Big(\left(\mathbf{u}_g\right)_x \cdot \nabla \left(\mathbf{u}_h^i\right)_x \Big) \\
&\qquad - \Big(\left(\mathbf{u}_h^i\right)_x \cdot \nabla \left(\mathbf{u}_g\right)_x \Big) + \Big(\left(\mathbf{u}_h^i\right)_x \cdot \nabla \left(\mathbf{u}_h^i\right)_x \Big) \Big] \; dx, \\
F_2[j] &:= \int_\Omega \begin{pmatrix} 0 \\ \varphi_j \end{pmatrix} \cdot \Big[\mathbf{f} - \nabla \left(\mathbf{u}_g\right)_y - \Big(\left(\mathbf{u}_g\right)_y \cdot \nabla \left(\mathbf{u}_h^i\right)_y \Big) \\
&\qquad - \Big(\left(\mathbf{u}_h^i\right)_y \cdot \nabla \left(\mathbf{u}_g\right)_y \Big) + \Big(\left(\mathbf{u}_h^i\right)_y \cdot \nabla \left(\mathbf{u}_h^i\right)_y \Big) \Big] \; dx, \\
b[k] &:= - \int_\Omega \psi_k \nabla \cdot \mathbf{u}_g \; dx,
\end{aligned}$$

with $(\cdot)_x$ and $(\cdot)_y$ denoting the x- and y-components, $j = 1, \dots, D_0$ and $k = 1, \dots, D_p$.

4.2.4 Neumann boundary condition

Another type of boundary condtion is the Neumann boundary condition. For the purpose of description of the boundary conditions, we denote by Γ_D the *Dirichlet-boundary* and by Γ_N the *free-stream-*, *Neumann* or *do-nothing-boundary*, such that:

$$\partial\Omega = \Gamma = \overline{\Gamma_D} \cup \overline{\Gamma_N}, \tag{4.48}$$

under the assumption that the $d-1$-dimensional Lebesgue-measure of $\Gamma_N \cap \Gamma_D$ is zero, which means that Γ_N and Γ_D may intersect each other only at common points in space ($d = 2$) or at common edges/points ($d = 3$).
The Neumann boundary condition which is described in [20, 41] is of use, for example, if we want to simulate the flow of a fluid through a pipe, where the velocity profile at the inlet of the pipe is known, but nothing is known about the behaviour of the flow after leaving the pipe. This basically represents the truncation of a physical domain, e.g. when only the flow in the first half of the pipe is of interest. Then the second half of the pipe can be truncated by introducing a Neumann boundary normal to the pipe axis, so the computational domain Ω can be restricted to the first half, see Figure 4.1, which was taken from [20].

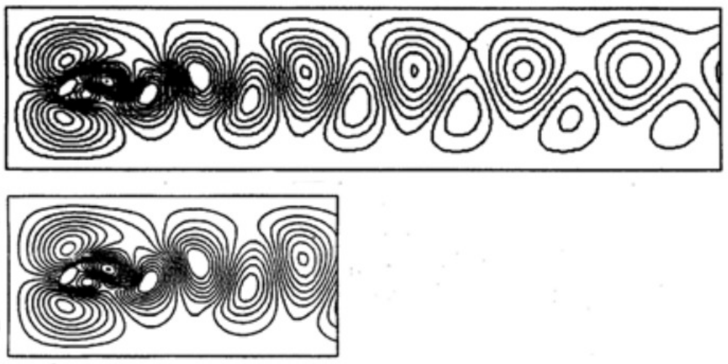


Figure 4.1: Truncated calculation domain, see [20].

So far, we incorporated Dirichlet-boundary conditions by choosing an extension $\mathbf{u}_g$ that fulfills the boundary data $\mathbf{g}$ and then solving for $\mathbf{u}_0 \in [H_0^1(\Omega)]^d$, such that $\mathbf{u} := \mathbf{u}_0 + \mathbf{u}_g$ yields the final solution. In contrary, for the Neumann boundary Γ_N, we do not want to prescribe a specific velocity profile. Therefore, we use the space

$$X := \left\{ \mathbf{v} \in \left[H^1(\Omega) \right]^d \mid \mathbf{v} = 0 \text{ on } \Gamma_D \right\}.$$

Let us recall the momentum balance equation (3.1):

$$\frac{d}{dt} \int_{\omega(t)} \rho(t, \mathbf{x}) u_k(t, \mathbf{x})\, dx = \int_{\omega(t)} \rho(t, \mathbf{x}) f_k(t, \mathbf{x})\, dx + \int_{\partial\omega(t)} t_k(t, \mathbf{x}, \mathbf{n})\, ds_x$$

with the surface stress tensor $T \cdot \mathbf{n} = \mathbf{t}$. Under certain assumptions on material properties, there follows the relation

$$T_{ij} = -pI + \nu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right)$$

and therefore

$$\mathbf{t} = \nu \nabla \mathbf{u} \cdot \mathbf{n} - p\mathbf{n}.$$

The momentum balance equation states that a change in total impulse over time is induced by volumetric forces and surface forces. Definition of a Neumann or do-nothing boundary can be seen as introducing a virtual boundary to a larger domain, such as when a domain is truncated as depicted in Figure 4.1. Therefore, there will be zero surface forces introduced at this virtual boundary, which yields the final Neumann condition

$$\nu \nabla \mathbf{u} \cdot \mathbf{n} - p\mathbf{n} = 0 \quad \text{on } \Gamma_N. \tag{4.49}$$

It is implied when we assume the existence of a sufficiently smooth solution. Cases where the do-nothing condition leads to problems are described in [20, 41].

5 Error Estimators

5.1 Error estimation techniques for finite element discretizations

5.1.1 Residual error estimator

The idea for this error estimator goes back to works on finite elements, c.f. Verfürth [43] or Babuška [3], who derived residual error estimators for problems such as the Poisson problem, linear elasticity, the biharmonic plate bending problem or the nonlinear Navier-Stokes system. In finite element methods, the solution procedure can be interpreted as a minimisation of the weighted residual over the computational domain by using the Galerkin principle, see [39]. The residual error estimator aims to detect local contributions to the weighted residual.

The residual error estimator for the Laplace equation

We begin by introducing the estimator and explaining the basic principles for a simple equation, the Poisson equation. A discrete solution will not necessarily satisfy the governing equations in their strong form, which are:

$$-\Delta u = f \quad \text{in} \quad \Omega \subset \mathbb{R}^d, \tag{5.1}$$

$$u = 0 \quad \text{on} \quad \Gamma. \tag{5.2}$$

The standard weak formulation of (5.1)-(5.2) then is to find $u \in X$ such that

$$\int_\Omega \nabla u \cdot \nabla v \, dx = \int_\Omega fv \, dx \quad \forall v \in X, \tag{5.3}$$

where $X := H_0^1(\Omega)$. Furthermore, we have the finite element discretization of problem (5.1)-(5.2) to find $u_h \in X_h \subset X$, such that

$$\int_\Omega \nabla u_h \cdot \nabla v_h \, dx = \int_\Omega f v_h \, dx \quad \forall v_h \in X_h. \tag{5.4}$$

Let $u \in X$ and $u_h \in X_h$ be the solutions of variational problems (5.3) and (5.4). They fulfill the identity

$$\int_\Omega \nabla(u - u_h) \cdot \nabla v \, dx = \int_\Omega fv \, dx - \int_\Omega \nabla u_h \cdot \nabla v \, dx \quad \forall v \in X, \tag{5.5}$$

while we can also observe

$$\int_\Omega \nabla(u-u_h)\cdot\nabla v_h = 0 \quad \forall v_h \in X_h \tag{5.6}$$

by subtracting equation (5.4) from (5.3). Using a Poincare-Friedrichs inequality

$$\|v\|_{L^2(\Omega)} \le c_\Omega |v|_{H^1(\Omega)} \quad \forall v \in X$$

and consequently

$$\|v\|^2_{H^1(\Omega)} = \|v\|^2_{L^2(\Omega)} + |v|^2_{H^1(\Omega)} \le c^2_\Omega |v|^2_{H^1(\Omega)} + |v|^2_{H^1(\Omega)} = (1+c^2_\Omega)|v|^2_{H^1(\Omega)}$$

we get with Cauchy-Schwarz's inequality and the definition of the $|\cdot|_{H^1(\Omega)}$-norm through the supremum

$$\frac{1}{1+c^2_\Omega}\|v\|_{H^1(\Omega)} \le \sup_{\substack{w\in X \\ \|w\|_{H^1(\Omega)}=1}} \int_\Omega \nabla v\cdot\nabla w\, dx \le \|v\|_{H^1(\Omega)}. \tag{5.7}$$

By substituting v in (5.7) through $u-u_h$ and using (5.5) we get the estimate

$$\begin{aligned} &\sup_{\substack{w\in X \\ \|w\|_{H^1(\Omega)}=1}} \left\{\int_\Omega fw\, dx - \int_\Omega \nabla u_h\cdot\nabla w\, dx\right\} \\ \le& \|u-u_h\|_{H^1(\Omega)} \\ \le&(1+c^2_\Omega) \sup_{\substack{w\in X \\ \|w\|_{H^1(\Omega)}=1}} \left\{\int_\Omega fw\, dx - \int_\Omega \nabla u_h\cdot\nabla w\, dx\right\}. \end{aligned} \tag{5.8}$$

This shows that the error in the full $H^1(\Omega)$-norm is bounded from above and below by the norm of the residual in the dual space of X. In the next steps we estimate the upper bound in (5.8) by quantities which can be computed and therefore used as an estimator for the $H^1(\Omega)$-error. First we apply elementwise integration by parts:

$$\begin{aligned} &\int_\Omega fw\, dx - \int_\Omega \nabla u_h\cdot\nabla w\, dx = \int_\Omega fw\, dx - \sum_{T\in\mathcal{T}}\int_T \nabla u_h\cdot\nabla w\, dx \\ =& \int_\Omega fw\, dx - \sum_{T\in\mathcal{T}}\left\{-\int_T w\Delta u_h\, dx + \int_{\partial T} (\mathbf{n}_T\cdot\nabla u_h)\, w\, ds_x\right\} \\ =& \sum_{T\in\mathcal{T}}\int_T (f+\Delta u_h)w\, dx - \sum_{E\in\mathcal{E}_0}\int_E [\mathbf{n}_E\cdot\nabla u_h]\, w\, ds_x, \end{aligned}$$

where $\mathbf{n}_E$ is the unit normal vector of the edge E while $[\cdot]_E$ is the jump of a function over the edge E. The definition of the orientation of $\mathbf{n}_E$ is in this case irrelevant, but it has to be fixed.
For the next step we use the interpolation operator $I_h^C : X \to X_h$ of Clement [14], for which we have local error estimates available. We define the operator as follows.

Denote by P_1 the space of polynomials of degree at most 1. For a given $\phi \in X$ and node $x \in \mathcal{N}$, let $\pi_x \phi$ be the $L^2(\omega_x)$-projection of ϕ on P_1,

$$\int_{\omega_x} \phi\psi \; dx = \int_{\omega_x} \pi_x \phi \psi \; dx \quad \forall \psi \in P_1,$$

where ω_x is the node patch as defined in definition 2.3. We define $I_h^C \phi$ by

$$\begin{aligned} I_h^C \phi(x) &= (\pi_x \phi)(x) \quad && \forall x \in \mathcal{N}_0 \\ I_h^C \phi(x) &= 0 && \forall x \in \mathcal{N}_\Gamma. \end{aligned}$$

This operator satisfies the following estimates, see [14]:

Lemma 5.1. *Let $T \in \mathcal{T}$ and $E \in \mathcal{E}$. Let $\phi \in X$ be arbitrary. Then there holds:*

$$\begin{aligned} \|\phi - I_h^C \phi\|_{L^2(T)} &\leq c_1^C h_T \|\phi\|_{H^1(\omega_T)} \\ \|\phi - I_h^C \phi\|_{L^2(E)} &\leq c_2^C h_E^{1/2} \|\phi\|_{H^1(\omega_E)}, \end{aligned}$$

where the constants c_1^C and c_2^C depend only on the smallest angle of elements of the triangulation $\mathcal{T}$.

We consider an arbitrary but fixed element $w \in X$. Then, using (5.6) and (5.5) written with $v_h = I_h^C v$ and using Cauchy-Schwarz's inequality, we get

$$\begin{aligned} &\int_\Omega f w \; dx - \int_\Omega \nabla u_h \cdot \nabla w \; dx \\ &\quad = \sum_{T \in \mathcal{T}} \int_T (f + \Delta u_h)(w - I_h^C w) \; dx + \sum_{E \in \mathcal{E}_0} \int_E \left[\mathbf{n}_E \cdot \nabla u_h\right]_E (w - I_h^C w) \; dx \\ &\quad \leq \sum_{T \in \mathcal{T}} c_1^C h_T \|f + \Delta u_h\|_{L^2(T)} \|w\|_{H^1(\omega_T)} + \sum_{E \in \mathcal{E}_0} c_2^C h_E^{1/2} \| \left[\mathbf{n}_E \cdot \nabla u_h\right] \|_{L^2(E)} \|w\|_{H^1(\omega_E)} \\ &\quad \leq \max\left\{c_1^C, c_2^C\right\} \left\{ \sum_{T \in \mathcal{T}} h_T^2 \|f + \Delta u_h\|_{L^2(T)}^2 + \sum_{E \in \mathcal{E}_0} h_E \|[\mathbf{n}_E \cdot \nabla u_h]_E\|_{L^2(E)}^2 \right\}^{1/2} \\ &\qquad\qquad \cdot \left\{ \sum_{T \in \mathcal{T}} \|w\|_{H^1(\omega_T)}^2 + \sum_{E \in \mathcal{E}_0} \|w\|_{H^1(\omega_E)}^2 \right\}^{1/2} \\ &\quad \leq \tilde{c} \|w\|_{H^1(\Omega)} \left\{ \sum_{T \in \mathcal{T}} h_T^2 \|f + \Delta u_h\|_{L^2(T)}^2 + \sum_{E \in \mathcal{E}_0} h_E \|[\mathbf{n}_E \cdot \nabla u_h]_E\|_{L^2(E)}^2 \right\}^{1/2} \end{aligned}$$

and together with (5.8) there follows

$$\|u - u_h\|_{H^1(\Omega)} \leq c \left\{ \sum_{T \in \mathcal{T}} h_T^2 \|f + \Delta u_h\|_{L^2(T)}^2 + \sum_{E \in \mathcal{E}_0} h_E \|[\mathbf{n}_E \cdot \nabla u_h]_E\|_{L^2(E)}^2 \right\}^{1/2}, \tag{5.9}$$

which leads to the definition of the elementwise residual error estimator:

$$\eta_{R,T} := \left\{ h_T^2 \|f + \Delta u_h\|_{L^2(T)}^2 + \frac{1}{2} \sum_{E \in \mathcal{E}_T \cap \mathcal{E}_0} h_E \|[\mathbf{n}_E \cdot \nabla u_h]_E\|_{L^2(E)}^2 \right\}^{1/2}.$$

If appropriate inverse inequalities are available and $f \in [H^{-1}(\Omega)]^d$, equation (5.9) can be estimated by an expression without the scaling factors h_T and h_E:

$$\|u - u_h\|_{H^1(\Omega)} \le c \left\{ \sum_{T \in \mathcal{T}} \|f + \Delta u_h\|^2_{H^{-1}(T)} + \sum_{E \in \mathcal{E}_0} \|[\mathbf{n}_E \cdot \nabla u_h]_E\|^2_{H^{-1/2}(\Gamma)} \right\}^{1/2} .$$

It has been shown that $\eta := (\sum_{T\in\mathcal{T}} \eta^2_{R,T})^{1/2}$ constitutes an upper bound of the global computational error in the $H^1(\Omega)$-norm. Thus, we assume that $\eta_{R,T}$ can be used as bound for the local $H^1(\Omega)$-error. Conversely, it can be shown that $\|u - u_h\|_{H^1(\omega_T)}$ is bounded from below by $\eta_{R,T}$. The proof can be found in [43].

Remark 5.1. *When performing the integration by parts in (4.6) and (4.7) elementwise instead of over the whole domain Ω, one obtains the edgewise contributions in $\eta_{R,T}$. If the integration is performed over the whole domain Ω, the edge-residuals $[\mathbf{n}_E \cdot \nabla u_h]_E$ are implicitly required to be equal to zero, but when approximating with finite elements, this actually cannot be fully satisfied.*

The residual error estimator for the Stokes and Navier-Stokes system

In [43], the residual error estimator is also derived for the Navier-Stokes system. The element residual

$$\mathcal{R}_h(\mathbf{u}_h, p_h) = -\nu\Delta\mathbf{u}_h + \mathbf{u}_h \cdot \nabla\mathbf{u}_h + \nabla p_h - \mathbf{f} \tag{5.10}$$

is defined as a function which measures how well the original governing equations are satisfied by the discrete solution $(\mathbf{u}_h, p_h)$. For the Stokes equations, the residual is defined by leaving the convective term $\mathbf{u}_h \cdot \nabla\mathbf{u}_h$ in (5.10) aside. Furthermore, we define the boundary residual by

$$r_{h|E} := \begin{cases} \left[\nu \frac{\partial \mathbf{u}_h}{\partial \mathbf{n}_E} - \mathbf{n}_E p_h\right]_E, & \text{if } E \in \mathcal{E}_0 \\ 0, & \text{if } E \in \mathcal{E}_D \\ -\nu \frac{\partial \mathbf{u}_h}{\partial \mathbf{n}_E} + \mathbf{n}_E p_h, & \text{if } E \in \mathcal{E}_N, \end{cases} \tag{5.11}$$

with $[\cdot]_E$ being the jump of a function over the edge E, i.e.

$$[\varphi]_E := \varphi_{|T} - \varphi_{|K}, \tag{5.12}$$

where T and $K \in \mathcal{T}$ are two neighbouring cells sharing a common edge $E \in \mathcal{E}_0$. The residual a posteriori error estimator is then given by

$$\eta_{R,T} := \left\{ h_T^2 \left\|\mathcal{R}_{h|T}\right\|^2_{L^2(T)} + \|\nabla \cdot \mathbf{u}_h\|^2_{L^2(T)} + \frac{1}{2} \sum_{E \in \mathcal{E}} h_E \left\|r_{h|E}\right\|^2_{L^2(E)} \right\}^{1/2} . \tag{5.13}$$

Remark 5.2.
- *The second term in (5.13) is the residual of the discrete solution with respect to the continuity equation $\nabla \cdot \mathbf{u} = 0$, i.e. it measures how well the incompressibility constraint is fulfilled.*

- *The pressure jumps in $r_{h|E}$ vanish when using a discretization scheme with continuous pressure approximation.*

In [43], it has been shown that under suitable conditions, the following two results hold for the error estimator $\eta_{R,T}$:

Theorem 5.2 (Global realiability). *There exist constants c_1^* and c_2^* depending only on the shape of the elements T of $\mathcal{T}$, but not on the size of the elements, such that*

$$\left\{|\mathbf{u}-\mathbf{u}_h|^2_{H^1(\Omega)} + \|p-p_h\|^2_{L^2(\Omega)}\right\}^{1/2} \le c_1^* \left\{\sum_{T\in\mathcal{T}} \eta^2_{R,T}\right\}^{1/2} + c_2^* \left\{\sum_{T\in\mathcal{T}} h_T^2 \|\mathbf{f}-\mathbf{f}_{0,T}\|^2_{L^2(T)}\right\}^{1/2} \tag{5.14}$$

holds, where

$$\mathbf{f}_{0,T} := \frac{1}{|T|}\int_T \mathbf{f}\, dx$$

is the integral mean of $\mathbf{f}$ over the control volume T.

Theorem 5.3 (Local efficiency). *There exist constants c_*^1 and c_*^2 depending only on the shape of the elements T of $\mathcal{T}$, but not on the size of the elements, such that*

$$\eta_{R,T} \le c_*^1 \left\{|\mathbf{u}-\mathbf{u}_h|^2_{H^1(\omega_T)} + \|p-p_h\|^2_{L^2(\omega_T)}\right\}^{1/2} + c_*^2 \left\{\sum_{T'\in\omega_T} h^2_{T'} \|\mathbf{f}-\mathbf{f}_{0,T}\|^2_{L^2(T')}\right\}^{1/2} \tag{5.15}$$

holds.

Remark 5.3. *Efficiency and reliability are two properties which each practically relevant a posteriori error estimator should posses. Basically these properties mean that the error indicator scales asymptotically in the same way as the real error.*

Remark 5.4. *An abstract framework to prove efficiency and reliability results for residual based error estimators for general varational formulations can be found in [43]. Using less abstract notation, such results are also shown in [22] for general finite element discretizations and also some type of finite volume discreatizations of the Stokes equations.*

5.1.2 Error estimation based on the solution of local problems

The idea of this error estimator (see [43]) is to construct auxiliary problems for the original discrete problem (e.g. (4.20)-(4.21) or (4.41)-(4.42)). Its basic principle is explained for a simple Poisson problem with Dirichlet-conditions, see (5.1)-(5.2). The auxiliary problems have to be similar to the original problem. They should satisfy the following conditions:

- For information on the local error, the auxiliary problems should involve only small subdomains of Ω.

- The finite element spaces used for their solution should approximate through polynomials of higher order than the original ones.
- As few degrees of freedom as possible are to be preferred in order keep computational work at a minimum.
- To each edge and triangle, there should correspond at least one degree of freedom.
- The solution of all auxiliary problems should not cost more than the assembly of the stiffness matrix of problem (5.1)-(5.2).

These conditions then lead to the following definition: For all $T \in \mathcal{T}$, let

$$\tilde{V}_T := \operatorname{span}\{\beta_{T'}, \beta_E \colon T' \subset \omega_T,\ E \in \mathcal{E}(T) \cap \mathcal{E}\}$$

and define the error estimator

$$\eta_{D,T} := \|\nabla \tilde{v}_T\|_{L^2(\omega_T)},$$

where β_T are the triangle bubble functions (4.27) and β_E are the edge bubble functions. Let $E \in \mathcal{E}$ and $\omega_E = T_1 \cup T_2$ and let $x_1, x_2 \in \mathcal{N}(E)$ be the two vertices of the edge E. Then with $\lambda_{T_i,1}$ and $\lambda_{T_i,2}$ being the barycentric coordinates of the triangle T_i corresponding to the vertices x_1 and x_2, respectively, the edge-bubble-functions are defined by

$$\beta_E := \begin{cases} \lambda_{T_i,1}\lambda_{T_i,2} & \text{on } T_i,\ i = 1,2 \\ 0 & \text{on } \Omega \backslash \omega_E. \end{cases}$$

Supposing a discrete solution u_h of the original problem has been calculated, function $\tilde{v}_T \in \tilde{V}_T$ is then calculated as the unique solution of

$$\int_{\omega_T} \nabla \tilde{v}_T \cdot \nabla w \, dx = \sum_{T' \subset \omega_T} \int_{T'} f_{T'} w \, dx - \int_{\omega_T} \nabla u_h \cdot \nabla w \, dx \quad \forall w \in \tilde{V}_T.$$

Furthermore, $\varphi = u_h + \tilde{v}_T$ can be interpreted as the approximate solution of the following problem:

$$\begin{aligned} -\Delta \varphi &= f && \text{in } \omega_T \\ \varphi &= u_h && \text{on } \partial \omega_T. \end{aligned}$$

In [43] it has been shown that the estimator $\eta_{D,T}$ is in the same way as the residual estimator $\eta_{R,T}$ bounded from above and below by the computational error. Moreover, the both error estimators yield local bounds for each other:

Theorem 5.4. *Let $u \in X$ and $u_h \in X_h$ be the unique solutions of the following variational formulations of the Poisson problem (5.1)-(5.2) with homogeneous Dirichlet-conditions such that*

$$\int_\Omega \nabla u \cdot \nabla v \, dx = \int_\Omega f v \, dx \quad \forall v \in X$$

and

$$\int_\Omega \nabla u_h \cdot \nabla v_h \, dx = \int_\Omega f v_h \, dx \quad \forall v_h \in X_h$$

for some space X and finite element space X_h. There exist constants $c_{\mathcal{T},1}, \ldots, c_{\mathcal{T},4}$ depending only on the smallest angle of the triangulation $\mathcal{T}$, such that the estimates

$$\eta_{D,T} \leq c_{\mathcal{T},1} \left\{ \sum_{T' \subset \omega_T} \eta_{R,T'}^2 \right\}^{1/2},$$

$$\eta_{R,T} \leq c_{\mathcal{T},2} \left\{ \sum_{T' \subset \omega_T} \eta_{D,T'}^2 \right\}^{1/2},$$

$$\eta_{D,T} \leq c_{\mathcal{T},3} \left\{ \|u - u_h\|_{H^1(\omega_T)}^2 + \sum_{T' \subset \omega_T} h_{T'}^2 \|f - f_{T'}\|_{L^2(T')}^2 \right\}^{1/2},$$

$$\|u - u_h\|_{H^1(\Omega)} \leq c_{\mathcal{T},4} \left\{ \sum_{T \in \mathcal{T}} \eta_{D,T}^2 + \sum_{T \in \mathcal{T}} h_T^2 \|f - f_T\|_{L^2(T)}^2 \right\}^{1/2},$$

hold for all $T \in \mathcal{T}$.

Remark 5.5. *Analogously, it is also possible to define an error estimator $\eta_{N,T}$ based on the solution of local Neumann-problems by using a homogeneous Dirichlet-condition on edges E of T which are part of the boundary Γ and the Neumann-condition $\frac{\partial \varphi}{\partial n} = -\frac{1}{2} \left[n_E \cdot \nabla u_h \right]_E$ on internal edges of T. Similar boundedness properties as the ones for $\eta_{D,T}$ can also be shown for $\eta_{N,T}$.*

Remark 5.6. *The estimator $\eta_{D,T}$ was first introduced in [11], and $\eta_{N,T}$ in [7]. A more exhaustive analysis of the error estimation techniques presented in this section and the extension and application to the Stokes and Navier-Stokes problem was done in [9, 8, 42].*

Remark 5.7. *The estimators $\eta_{D,T}$ and $\eta_{N,T}$ pose an alternative to the residual error estimator, but are always more difficult to compute.*

5.1.3 Error estimators based on gradient recovery, the ZZ-estimator

The idea behind the error estimators based on gradient recovery is to construct a continuous approximation of the gradient by postprocessing the gradient of the finite element approximation. So if the aim of the method is to reduce the H^1-error, i.e. if we want to reduce

$$|||e|||^2 := \int_\Omega |\nabla \phi - \nabla \phi_h|^2 \, dx,$$

then the gradient $\nabla \phi$ would have to be replaced by a recovered gradient $G_h(I_h^p \phi_h)$ in order to achieve an a posteriori error estimate, because the exact gradient $\nabla \phi$ is of course not known. I_h^p is the interpolation operator of degree p.
Such methods were investigated in [1, 2]. The description of the methods in the present section is based on those two papers. Those methods do not involve any

information on the kind of problem to be solved, therefore we consider a quantity ϕ, which can be the solution u of the Poisson problem (5.1)-(5.2), the components of the vector-valued velocity $\mathbf{u}$ or the pressure p in the Navier-Stokes problem (4.32)-(4.33). There is one especially noteworthy (because of its numerical robustness) gradient recovery technique: The Zienkiewicz-Zhu-estimator (hence the name ZZ), which was first introduced in [47].

Recovery operators

In this section, a class of recovery operators is described. A recovery operator should satisfy the following conditions:

(R1) *Consistency condition*: If ϕ_h belongs to a finite element space of order $p+1$, then

$$G_h(I_h^p \phi_h) = \nabla \phi_h.$$

(R2) *Localization condition*: The computation of G_h should naturally be inexpensive. This practically means that no global compuations should be necessary, otherwise the whole variational problem could be solved on a finer mesh in the same time. We consider recovery techniques where the recovered gradient at a point x^* depends only on a neighbourhood of x^*, like the element patch ω_T, if $x^* \in T$, $T \in \mathcal{T}$.

(R3) *Boundedness and linearity condition*: If X_h is a finite element space of order p and $G_h : X_h \to X_h \times X_h$ is a linear operator then there exists a constant C independent of h such that

$$\|G_h(\phi)\|_{L^\infty(T)} \le C|\phi|_{W^{1,\infty}(\omega_K)} \quad \forall T \in \mathcal{T}, \forall \phi \in X,$$

with the norm

$$\|\phi\|_{W^{m,\infty}(\Omega)} = \max_{|\alpha|\le m} \|D^\alpha \phi\|_{L^\infty(\Omega)}.$$

If conditions $(R1)$-$(R3)$ are satisfied, then it can be shown (see [2]) that G_h offers a good approximation to the true gradient:

Lemma 5.5. *Suppose that G_h satisfies $(R1)$-$(R3)$ and that $\phi \in H^{p+2}(\Omega)$. Then*

$$\|\nabla \phi - G_h(I_h^p \phi)\|_{L^2(\Omega)} \le C h^{p+1} |\phi|_{H^{p+2}(\Omega)}$$

where $C > 0$ is independent of h and ϕ_h.

The superconvergence property

Lemma 5.5 shows that if an operator G_h satisfies certain conditions, then applying it to $I_h^p \phi$ yields good approximations to the derivatives of ϕ. If the so called *superconvergence phenomenon* is present, then applying G_h to the discrete solution ϕ_h itself also yields good approximations to the derivatives. Normally, for the finite-element method there holds

$$|\nabla\phi - \nabla\phi_h|_{H^1(\Omega)} \leq C(\phi)h^p$$

with a positive constant $C(\phi)$ depending on ϕ. If superconvergence is present, this means that under appropriate regularity assumptions on the triangulation $\mathcal{T}$ and the solution ϕ there holds

$$|\phi_h - I_h^p\phi|_{H^1(\Omega)} \leq C(\phi)h^{p+1}. \tag{5.16}$$

The necessary assumptions needed for superconvergence to occur depend on the used finite element method and were investigated more closely in [24]. Superconvergence being present leads to (see [2]):

Lemma 5.6. *Suppose $\phi \in H^{p+2}(\Omega)$, G_h satisfies $(R1)$-$(R3)$ and (5.16) is valid. Then*

$$\|\nabla\phi - G_h(\phi_h)\|_{L^2(\Omega)} \leq C(\phi)h^{p+1}$$

holds where $C > 0$ is independent of h and ϕ_h.

In general, every error estimator for the local error in a cell $T \in \mathcal{T}$ based on gradient recovery will have the form

$$\eta_T = \|G_h(\phi_h) - \nabla\phi_h\|_{L^2(\Omega)},$$

while the global error estimator is

$$\eta = \left(\sum_{T\in\mathcal{T}} \eta_T^2\right)^{1/2}.$$

A simple way to define the nodal values in nodes a of the recovered gradient is to use a weighted average of $\nabla\phi_h$ over all elements in the node patch ω_a:

$$G_h(\phi_h)(a) = \sum_{T\in\omega_a} \frac{|T|}{|\omega_a|}\nabla\phi_{h|T}.$$

The values of the recovered gradient at an arbitrary point $x \in \Omega$ can then be obtained by piecewise linear interpolation of the nodal values to the interior of the element T to which the point x belongs.

The Zienkiewicz-Zhu patch recovery technique

For the ZZ-estimator we first construct an intermediate recovered gradient G_h^a for each nodepatch ω_a and then perform an averaging to compute the final recovered gradient at the point $\mathbf{x} \in \Omega$:

$$G_h\left[\phi_h\right](\mathbf{x}) = \frac{1}{|\mathcal{N}|}\sum_{a\in\mathcal{N}} G_h^a\left[\phi_h\right](\mathbf{x}).$$

Let $Z(a)$ be a set of points at which the gradient is to be sampled. For quadrilateral elements, this set can be the set of Gauss-Legendre quadrature points. In our case for triangles, we would chose the circumcenters of the triangles

$$Z(a) = \{\mathbf{b} \mid \mathbf{b} \text{ is circumcenter of } T \in \omega_a\}$$

when using piecewise linear basis functions for the approximation of ϕ or the centers of the edges of the triangles in the nodepatch

$$Z(a) = \{\mathbf{b} \mid \mathbf{b} \text{ is center of } E,\, E \in \mathcal{E}(T),\, T \in \omega_a\}$$

for a piecewise quadratic approximation of ϕ. Further examples and explanations can be found in [46].
The function $G_h^a : X_h \to X_h \times X_h$ is then calculated by a least-squares fit to the gradient sampled at the points in $Z(a)$. It has the following form:

$$G_h^a(\mathbf{x}) = \sum_n \boldsymbol{\alpha}_n \varphi_n(\mathbf{x}) \tag{5.17}$$

where $\varphi_n(\mathbf{x})$ is a basis of the finite element space X_h and $\boldsymbol{\alpha}_n$ are constant vectors chosen in such a way that they minimize the expression

$$\sum_{\mathbf{z}\in Z(a)} \{G_h^a(\mathbf{z}) - \nabla\phi_h(\mathbf{z})\}^2 . \tag{5.18}$$

When computing a local L_2-projection G_h^a of $\nabla\phi_h$ on the node patch ω_a, one seeks for constant vectors $\boldsymbol{\alpha}_n$ in (5.17) such that

$$\int_{\omega_a} (G_h^a - \nabla\phi_h) \cdot \varphi_h \, dx = 0$$

holds for all $\varphi_h \in [X_h]^2$, thus

$$\int_{\omega_a} (G_h^a - \nabla\phi_h)^2 \; dx = \|G_h^a - \nabla\phi_h\|^2_{L_2(\omega_a)} = 0$$

holds for $\varphi_h = G_h^a - \nabla\phi_h$. Minimization of (5.18) resembles such a L_2-projection, but only using a small set of sample points $Z(a)$ instead of integrating over ω_a.
In the summation (5.17) we consider only degrees of freedom n which are associated with the elements in the node patch ω_a, such that the recovery procedure fulfills condition $(R2)$. The recovery operator is linear and bounded, therefore condition $(R1)$ is also satisfied. The superconvergence condition (5.16) and condition $(R1)$ can be fulfilled by chosing the sampling points in $Z(a)$ in such a way that superconvergence occurs. The error estimator will then be asymptotically exact, i.e. η will converge to the real computation error $|\nabla\phi - \nabla\phi_h|_{H^1(\Omega)}$ as the meshsize h decreases. This was confirmed with numerical results in [46, 47]. One would expect that the estimator would lose this property if the sampling points cannot be chosen accordingly, but it has been shown in [4, 5] that the estimator still performs satisfactorily in extreme choices of $Z(a)$.

5.2 Error estimation techniques for finite volume discretizations

5.2.1 Residual error estimator

With the residual error estimator for the finite volume method, we try to measure how well the approximate solution fulfills the governing equations in integral form compared to a solution of assumed order of variation p. This method can also be found in [23]. The variable ϕ represents the velocity components u_k.
Each finite volume discretization is accurate only up to p-th order, which means that the function used to evaluate the surface and volume integrals consists of the first p terms of a Taylor series expansion

$$\phi(\mathbf{r}) = \phi_T + (\mathbf{r} - \mathbf{r}_T) \cdot (\nabla\phi)_T + \ldots + \frac{1}{(p-1)!}(\mathbf{r} - \mathbf{r}_T)^{p-1} \underbrace{::}_{p-1} (\underbrace{\nabla\nabla\ldots\nabla\phi}_{p-1})_T, \quad (5.19)$$

and that p-th order accurate face interpolation is used. Thus, the variation of the numerical solution over the control volume is described by (5.19). The finite volume method presented in Chapter 3 is accurate up to second order.
There are two ways to express the value of ϕ at the face A_j with $\mathbf{r}_j$ being the position vector of the face center of A_j, see Figure 5.1:

- Assuming linear variation over the control volume:

$$\phi(\mathbf{r}_j) = \phi_T + (\mathbf{r}_j - \mathbf{r}_T) \cdot \nabla\phi_T. \quad (5.20)$$

  or interpolating from the neighbour N_j of T over the face A_j:

$$\phi(\mathbf{r}_j) = \phi_{N_j} + (\mathbf{r}_j - \mathbf{r}_{N_j}) \cdot \nabla\phi_{N_j}.$$

- Using a linear combination between the values ϕ_T and ϕ_{N_j}:

$$\phi(\mathbf{r}_j) = f_j\phi_T + (1 - f_j)\phi_{N_j},$$

  with

$$f_j = \frac{|\mathbf{r}_{N_j} - \mathbf{r}_j|}{|\mathbf{r}_j - \mathbf{r}_T| + |\mathbf{r}_{N_j} - \mathbf{r}_j|}. \quad (5.21)$$

The representation (5.21) of face values satisfying the governing equations is not consistent with the prescribed linear variation of ϕ used for volume integrals. Therefore, a consistent face value of ϕ for the control volume T should be calculated using (5.20).
The residual can now be defined in the following way for a velocity component $u_k = \phi$:

$$\begin{aligned}
\mathrm{res}_T(\phi) &= \int_T \left[\nabla \cdot (\rho\phi\mathbf{u}) - \nabla \cdot (\nabla\phi) - \rho f_k + \frac{\partial p}{\partial x_k}\right] dx \\
&= \int_{A_j} \mathbf{n}_j \cdot [\rho\phi\mathbf{u} - (\nabla\phi)]\, dx - \int_T \left[\rho f_k - \frac{\partial p}{\partial x_k}\right] dx \\
&\approx \sum_{j=1}^{n_f} \left[m_j - \mathbf{A}_j \cdot (\nabla\phi)_j\right] - |T| \left[(\rho f_k)_T - \left(\frac{\partial p}{\partial x_k}\right)_T\right],
\end{aligned} \quad (5.22)$$

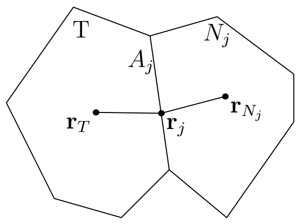


Figure 5.1: Interpolation at cell face A_j.

with

$$(\nabla\phi)_j = \nabla\phi_T$$

where the face value of ϕ is determined through the Taylor expansion instead of the linear combination (5.21):

$$\phi_j = \phi_T + (\mathbf{r}_j - \mathbf{r}_T)\cdot\nabla\phi_T.$$

In order to understand the residual error estimate, we consider a simple 1D-situation. In this case, there are three values available, as depicted in Figure 5.2: two face values, and one at the cell center. The governing equations in their integral form are fulfilled by those values. It is therefore possible to approximate the variation of the exact solution over the cell by passing a parabola (blue line) through the three available points. The residual can then be seen as the difference between the linear variation (red line) of ϕ using the gradient $\nabla\phi_T$ and the parabola (blue line). This implies that the distribution of the error is quadratic, which would mean that the error reduces to the exact error with the fourth power of mesh size.

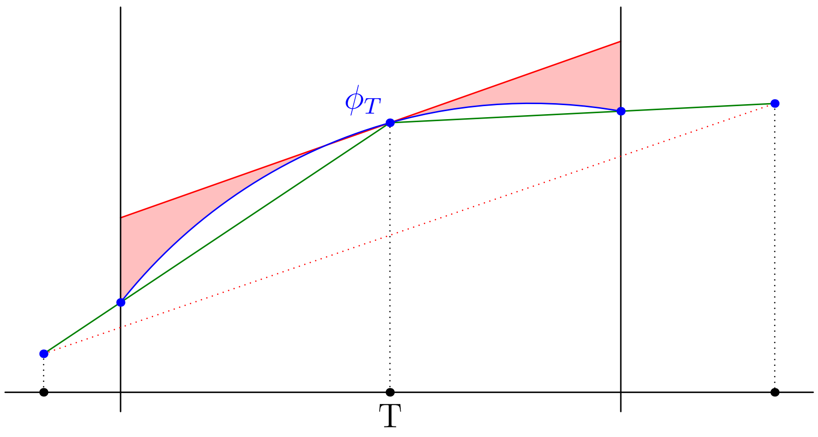


Figure 5.2: Scaling properties of the residual error estimate

Relation to the residual error estimate for the finite element method

If we use Cauchy-Schwarz' inequality on the definition of the residual (5.22) and sum over k, we will find similarity to the residual for finite elements (5.13):

$$\begin{aligned}\sum_{k=1}^{3} \operatorname{res}_T(u_k) &= \sum_{k=1}^{3} \int_T \left[\nabla \cdot (\rho u_k \mathbf{u}) - \Delta u_k - \rho f_k + \frac{\partial p}{\partial x_k}\right] dx \\ &\leq |T|^{1/2} \left(\int_T \left(\nabla \cdot (\rho \mathbf{u}\mathbf{u}) - \Delta \mathbf{u} - \rho \mathbf{f} + \nabla p\right)^2 \, dx\right)^{1/2} \\ &= |T|^{1/2} \|\mathcal{R}_{h|T}\|_{L_2(T)}.\end{aligned}$$

We denote by $\mathcal{R}_{h|T}$ the element residual (5.10) on T. The residual for finite elements additionally contains the term for the edge residual $r_{h|E}$ and the continuity $\|\nabla \cdot \mathbf{u}\|_{L_2(T)}$ over T. The continuity ocer T is already fulfilled for the finite volume method due to conservativity. The definition of an edge residual makes no sense for the finite volume method because the jump $[\cdot]$ is zero since interpolation from both sides yields the same values.

Normalization of the residual estimate

Error estimate (5.22) represents the volume integrated imbalance in approximation methods. By normalization we aim to establish the influence of the imbalance to the local value of ϕ. The terms contributing to the error are the ones containing ϕ and $\nabla\phi$, namely the diffusion and convection terms. A suitable normalization would then involve the characteristic diffusion and convection transport. The diffusion coefficient is known for each face. $|\mathbf{d}|$ is used as the characteristic length and $|\mathbf{A}|$ as the area active in the transport.
The total diffusion transport coefficient can then be calculated as the sum of volume weighted diffusion transport fluxes:

$$F_{diff} = \frac{1}{|T|} \sum_{j=1}^{n_f} \left[|\mathbf{A}_j| \frac{\rho_j}{|\mathbf{d}_j|}\right].$$

The next term of the normalization factor is the convection transport term. For a conservative discretization like the Finite Volume method, the total flux going out of the control volume is equal to the total flux going in. Therefore, the sum of positive mass fluxes m_j over the faces A_j weighted by the volume is used as the convective part of the normalization factor:

$$F_{conv} = \frac{1}{|T|} \sum_{j=1}^{n_f} \max\{m_j, 0\}.$$

The factor now reads:

$$F_{norm} = F_{conv} + F_{diff},$$

and the final form of the residual error estimate is:

$$\eta_{R,T}(\phi) = \frac{res_T(\phi)}{|T|F_{norm}}.$$

The residual error estimator $\eta_{R,T}(\phi)$ has the advantage that it is cheap and easy to compute, and all discretization errors are taken into account. It can also be extended to transport equations of vector or tensor quantities. In this case, the estimator needs to be computed for each vector component of the computed quantity. The magnitude of the resulting residual vector for each control volume can then be taken to gain a scalar value for the estimated error, where u_k are components of the velocity vector field:

$$\eta_{R,T}(\mathbf{u}) = \sqrt{\eta_{R,T}(u_1)^2 + \eta_{R,T}(u_2)^2 + \eta_{R,T}(u_3)^2}$$

5.2.2 Third order polynomial interpolation

For this error estimation technique, which is based on [29], we consider a third order polynomial interpolation of ϕ, but only in the direction of $\mathbf{d}_j$ between a cell T and its neighbour N_j, depending on the parameter ξ. The interpolating polynomial has the following form:

$$\phi^*(\xi) = c_0 + c_1\xi + c_2\xi^2 + c_3\xi^3 \tag{5.23}$$

with coefficients c_i, which can be determined by the conditions

$$\begin{aligned} \xi = 0: &\quad \phi^* = \phi_T, & \xi = |\mathbf{d}_j|: &\quad \phi^* = \phi_{P_j}, \\ \xi = 0: &\quad (\nabla\phi^*)^\xi = (\nabla\phi)^\xi_T, & \xi = |\mathbf{d}_j|: &\quad (\nabla\phi^*)^\xi = (\nabla\phi)^\xi_{T_j} \end{aligned} \tag{5.24}$$

and $(\nabla\phi)^\xi_{T_j}$ being the gradient in the direction $\mathbf{d}_j$, i.e.

$$(\nabla\phi^*)^\xi_{T_j} = (\nabla\phi)_{T_j} \cdot \mathbf{d}_j.$$

The conditions (5.24) are then satisfied by the coefficients

$$\begin{aligned} c_0 &= \phi_T, \\ c_1 &= (\nabla\phi)^\xi_T, \\ c_2 &= 3\frac{\phi_{T_j} - \phi_T}{|\mathbf{d}_j|^2} - \frac{(\nabla\phi)^\xi_{T_j} + 2(\nabla\phi)^\xi_T}{|\mathbf{d}_j|}, \\ c_3 &= -2\frac{\phi_{T_j} - \phi_T}{|\mathbf{d}_j|^3} + \frac{(\nabla\phi)^\xi_{T_j} + (\nabla\phi)^\xi_T}{|\mathbf{d}_j|}. \end{aligned}$$

Expression (5.23) can then be used to express the dependent variable and its gradient by:

$$\begin{aligned} \overline{\phi}_j &= c_0 + c_1\xi_j + c_2\phi_j^2 + c_3\xi_j^3, \\ \overline{\nabla\phi}_j \cdot \frac{\mathbf{d}_j}{|\mathbf{d}_j|} &= c_1 + 2c_2\xi_j + 3c_3\xi_j^2 \end{aligned}$$

with ξ_j being the value of ξ at the cell face j, i.e. at the intersection of the distance vector and the cell face plane, see Figure 5.1. Using those new values of ϕ and its gradient, the diffusion and convection coefficients of Section 3.3 can be recalculated by substituting ϕ and $\nabla\phi$ in the expressions (3.15) and (3.19) for ϕ^* and $\nabla\phi^*$, respectively. This leads to the new coefficients $\overline{D}_j$ and $\overline{C}_j$, which will in general be different than the original ones, D_j and C_j. Summing up the differences between these coefficients for each face bounding a control volume T yields the truncation error $\overline{\tau}$:

$$\overline{\tau}_T = \sum_{j=1}^{n_f} \left[(\overline{C}_j - C_j) + (\overline{D}_j - D_j) \right],$$

and together with an appropriate normalization the error indicator:

$$\eta_{\tau,T} = \frac{\overline{\tau}_T}{a_T \phi_{ref}}.$$

The normalization factor ϕ_{ref} might be set to the value of ϕ at the location where the truncation error is calculated, or it can be a typical value in the computational domain or subdomain. The factor a_T is the coeffcient a_T in the equation system (3.20).

5.2.3 The neighbour-difference criterion

The following error estimator is based on the simple principle of calculating the difference of values of a transported quantity ϕ or the pressure p in a control volume T and its neighbours T_j and taking the maximum of these differences as the error estimator:

$$\eta_{D,T}(\mathbf{u}) = \max_{T_j} \|\mathbf{u}_T - \mathbf{u}_{T_j}\|_2$$

for vectorial quantities like the velocity or

$$\eta_{D,T}(p) = \max_{T_j} \left| p_T - p_{T_j} \right|$$

for the pressure. Analogously, this error estimator can also be computed for every other quantity which is calculated in a postprocessing step based upon the primary variables $\mathbf{u}_h$ and p_h.

6 Implementation

6.1 The adaptive strategy

Algorithms using adaptive mesh refinement in general have the same form, independent of the type of problem to be solved. We are going to explain the separate components of an adaptive algorithm first, and then the algorithm itself.

6.1.1 Termination criteria

Let $\mathcal{T}^{(k)}$ be a sequence of triangulations of Ω, with the initial trinagulation $\mathcal{T}^{(0)}$. Let $N_k := |\mathcal{T}^{(k)}|$ be the number of elements of $\mathcal{T}^{(k)}$ and let N_{max}, k_{max} be natural numbers denoting the maximum number of elements and the maximum number of refinement steps, respectively. Furthermore, let and τ_{max}, ϵ_{max} be positive real numbers denoting a maximum computation time and a maximum computational error. Then we can define a termination criterion by stopping the algorithm as soon as the maximum iterations k_{max}, the maximum number of elements N_{max}, the maximum computation time τ_{max} or the maximum computational error ϵ_{max} is reached. Use of the computational error $\epsilon := \|\mathbf{u} - \mathbf{u}_h\|$ as a criterion is only possible if an exact solution is known, which is sometimes the case when testing the implementation.

6.1.2 Cell selection

Furthermore, it is necessary to choose a rule on how to select the elements T that have to be refined based on the value of the local error estimator η_T. Let $\theta \geq 0$ be the adaptivity parameter. Then we aim to find a set $\mathcal{M} \subset \mathcal{T}$ of marked elements which will be refined such that:

1. *Maximum value rule*: For every $T \in \mathcal{M}$, there holds:
$$\eta_T \geq \theta \max_{T' \in \mathcal{T}} \eta_{T'}.$$
2. *Mean value rule*: For every $T \in \mathcal{M}$, there holds:
$$\eta_T \geq \theta \operatorname*{mean}_{T' \in \mathcal{T}} \eta_{T'}.$$
3. *Absolute value rule*: For every $T \in \mathcal{M}$, there holds:
$$\eta_T \geq \theta.$$

4. *Quantile rule*: Let $T_0, \dots, T_{|\mathcal{T}|}$ be a numbering of elements containing every element T of $\mathcal{T}$ exactly once, such that $\eta_{T_0} \leq \dots \leq \eta_{T_{|\mathcal{T}|}}$. With $p = 1 - \theta$, $g = (|\mathcal{T}| - 1)p + \lfloor (|\mathcal{T}| - 1)p \rfloor$ and the p-th quantile
$$q_p = (1 - g)\eta_{T_{\lfloor (|\mathcal{T}|-1)p+1 \rfloor}} + g\eta_{T_{\lfloor (|\mathcal{T}|-1)p+2 \rfloor}}$$
we define $\mathcal{M}$ such that for every $T \in \mathcal{M}$, there holds:
$$\eta_T \geq q_p.$$

5. *Döfler marking*: This marking strategy was introduced by Dörfler, see [16]. Seek a minimal set $\mathcal{M}$ (minimal in the sense of "least elements"), such that there holds:
$$\sum_{T \in \mathcal{M}} \eta_T^2 \geq \theta \sum_{T \in \mathcal{T}} \eta_T^2.$$

Remark 6.1. *An exemplary value for θ for the maximum value rule would be* 0.5*. This means that every element T for which the estimator η_T is bigger than* 50% *of the maximum is being marked for refinement. The mean value rule is more robust from a statistical point of view and less dependent on the distribution of the estimators η_T. A possible value for this rule could be $\theta = 1.0$. In order to use the absolute value rule, some knowledge about the magnitudes of the error estimators must be available before defining the parameter θ. This makes it useful mainly for experimental cases. The quantile rule chooses the elements with the highest p% of the error estimators for refinement. It offers the greatest control over the number of elements that are being refined in every refinement step.*

6.1.3 The adaptive algorithm

The adaptive algorithm can now be described as follows:

Algorithm 6.1. *Let $\mathcal{T}^{(0)}$ be an initial triangulation of Ω. Further, define an adaptivity parameter $\theta \geq 0$ and choose one of the termination criteria in Section 6.1.1 and one of the cell selection rules in Section 6.1.2. Start with $k := 0$ as the number of refinement steps.*

1. *Compute the discrete solution (u_h^k, p_h^k) of the Stokes or Navier-Stokes problem by the finite element or finite volume method in Chapters 4 and 3.*
2. *Compute the error estimators η_T by using one of the error erstimation techniques described in Chapter 5.*
3. *Compute the set $\mathcal{M}^{(k)}$ based on the chosen cell selection rule.*
4. *Perform refinement of $\mathcal{T}^{(k)}$, yielding the new triangulation $\mathcal{T}^{(k+1)}$.*
5. *Decide based on the termination criterion wether the computation should stop or continue.*
6. *Define $k := k + 1$ and go back to step 1.*

The sequence of discrete solutions $\left\{(\mathbf{u}_h^0, p_h^0), \dots, (\mathbf{u}_h^{k_{end}}, p_h^{k_{end}})\right\}$ is the result of the computation. It belongs to the triangulations $\left\{\mathcal{T}^{(0)}, \dots, \mathcal{T}^{(k_{end})}\right\}$, where k_{end} is the iteration number when the algorithm terminates.

6.2 Implementation of the finite element method

In this section, the implementation of the finite element method as described in Chapter 4 is given. The implementation is done in `C++`.
Figure 6.1 is a visual interpretation of the adaptive solution process 6.1, including the iterative Newton-algorithm which starts by calculating the solution of the Stokes problem (4.2) as the initial guess $\mathbf{u}_h^0$ and performs Newton-iterations as described in Section 4.2 until a prescribed accuracy $\epsilon > 0$ for the iterations is achieved in order to find a solution for the Navier-Stokes problem (4.32)-(4.33).

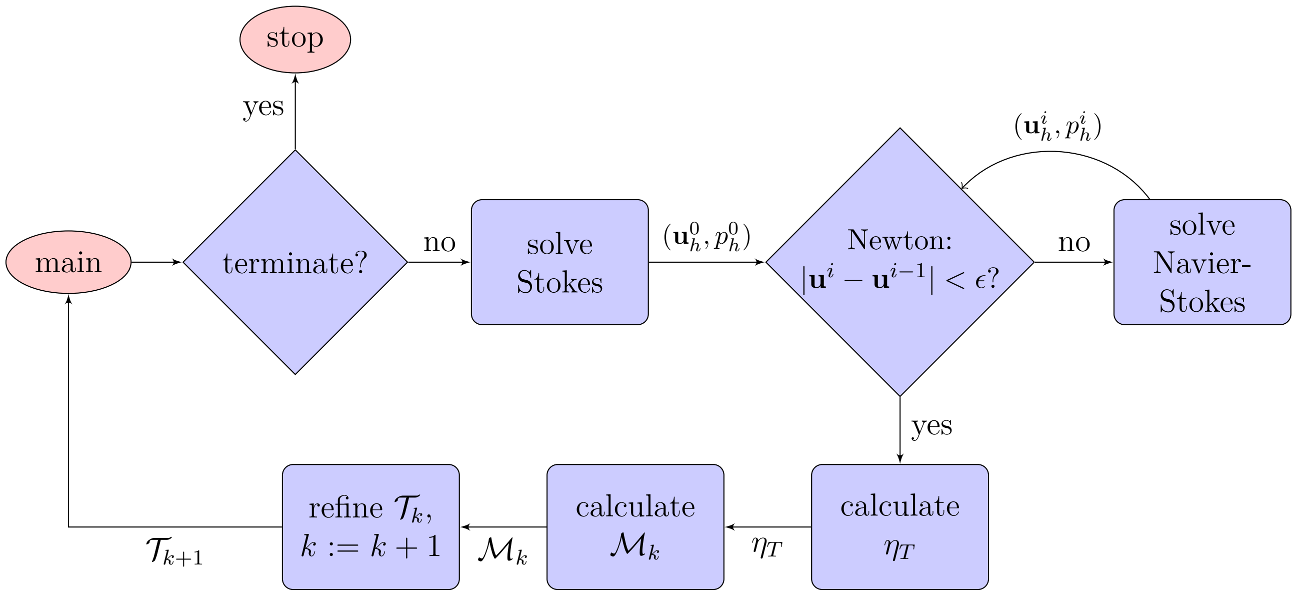

stop
yes
main
terminate?
no
solve Stokes
$(\mathbf{u}_h^0, p_h^0)$
Newton: $|\mathbf{u}^i - \mathbf{u}^{i-1}| < \epsilon$?
$(\mathbf{u}_h^i, p_h^i)$
no
solve Navier-Stokes
yes
calculate η_T
η_T
calculate $\mathcal{M}_k$
$\mathcal{M}_k$
refine $\mathcal{T}_k$, $k := k+1$
$\mathcal{T}_{k+1}$

Figure 6.1: Adaptive solution process, Newton iteration included

6.2.1 Numerical integration

The first step for numerical integration on a triangle T is to introduce a transformation of the integral from coordinates $\mathbf{x} = (x, y) \in T$ into coordinates $\tilde{\mathbf{x}} = (\tilde{x}, \tilde{y})$ on the reference element $\tilde{T}$. The so called reference element $\tilde{T}$ is the unit-triangle $\tilde{T} = \{(\tilde{x}, \tilde{y}) \in \mathbb{R}^2 \mid 0 \leq \tilde{x} \leq 1,\ \ 0 \leq \tilde{y} \leq \tilde{x}\}$ as depicted in Figure 6.2. Let $\mathbf{x}_i = (x_i, y_i)$ be the coordinates of the three nodes i of the triangle T. Then the transformation $\mathbf{x} \Leftrightarrow \tilde{\mathbf{x}}$ is given by:

$$\mathbf{x}(\tilde{\mathbf{x}}) = B_T \tilde{\mathbf{x}} + \mathbf{x}_1, \qquad \tilde{\mathbf{x}}(\mathbf{x}) = B_T^{-1}(\mathbf{x} - \mathbf{x}_1)$$

with the transformation matrix

$$B_T = \begin{pmatrix} x_2 - x_1 & x_3 - x_1 \\ y_2 - y_1 & y_3 - y_1 \end{pmatrix}.$$

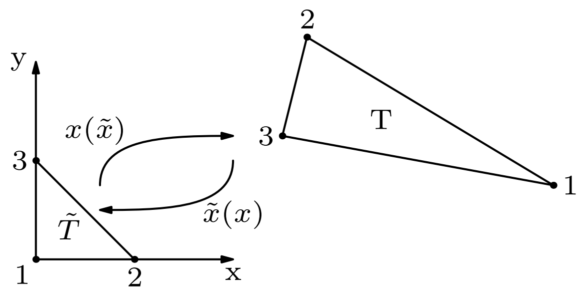


Figure 6.2: Transformation of coordinates between the reference triangle $\tilde{T}$ and the triangle $T \in \mathcal{T}$.

With the determinant of the transformation matrix $|B_T|$, the transformation of the integral is given by:

$$\int_T \varphi(\mathbf{x})\, dx = \frac{|B_T|}{2} \int_{\tilde{T}} \varphi(\mathbf{x}(\tilde{\mathbf{x}}))\, d\tilde{x}.$$

Integrals on the reference element are then approximated by an appropriate quadrature rule:

- *Gauß-quadrature* of N-th order:

  $$\int_a^b f(x)\, dx \approx \sum_{k=1}^{N} f(x)\omega_k,$$

  using the integration points x_k and integration weights ω_k. By Gauss-quadature, one dimensional integrals on the interval $[a, b]$ are approximated exactly if the integrand $f(x)$ is a polynomial of order up to $2N - 1$. More information on Gauss-quadrature can be found in [25]. It can be adopted to multidimensional integrals by simply iterating the quadrature for each coordinate direction and is therefore easy to implement, but requires N^2 integration points where d is the integration dimension. In multidimensional integration using Gauss-quadrature, the highest exponent of either x or y in f if f is a polynomial must be smaller than $2N - 1$ for the formula to be exact. Gauss quadrature is the method used in the implementation.

- *Radon-integration* of N-th order: This quadrature was especially developed for multidimensional integration, see [32]. The quadrature formula

  $$\iint\limits_{\omega} f(x, y)\, dx\, dy \approx \sum_{k=1}^{N} f(x_k, y_k)\omega_k$$

  for integration in two dimensions on connected, bounded domains $\omega \subset \mathbb{R}^2$ yields an exact approximation of the integral if the integrand $f(x, y)$ is a polynomial of order at most p with

  $$N = \frac{(p + 1)(p + 1)}{6}.$$

Radon-integration is better suited for multidimensional integrals since the number of required integration points is smaller than for Gauss-quadrature for multiple dimensions for the same accuracy.

The integration procedure requires that the values of the integrand are known at the points $\mathbf{x}(\tilde{\mathbf{x}}_k)$. If the integrand is a shapefunction like (6.1), then the values are known for points in the reference element, such that a transformation of the integration points $\tilde{\mathbf{x}}$ back again to the global coordinates is not necessary. The linear shape funtions can be given as functions of coordinates in the reference triangle in the following way:

$$\begin{aligned}\tilde{\varphi}_1(\tilde{x},\tilde{y}) &:= 1-\tilde{x}-\tilde{y},\\ \tilde{\varphi}_2(\tilde{x},\tilde{y}) &:= \tilde{x},\\ \tilde{\varphi}_3(\tilde{x},\tilde{y}) &:= \tilde{y},\end{aligned}$$

where the indices 1, 2 and 3 correspond to the degrees of freedom at the cornerpoints of the reference element $(0,0)$, $(1,0)$ and $(0,1)$, respectively. Furthermore, if piecewise quadratic ansatzfunctions are to be used, we have the additional functions

$$\begin{aligned}\tilde{\varphi}_4(\tilde{x},\tilde{y}) &:= 4\tilde{x}(1-\tilde{x}-\tilde{y}),\\ \tilde{\varphi}_5(\tilde{x},\tilde{y}) &:= 4\tilde{y}\tilde{x},\\ \tilde{\varphi}_6(\tilde{x},\tilde{y}) &:= 4\tilde{y}(1-\tilde{x}-\tilde{y})\end{aligned}$$

corresponding to degrees of freedom at the midpoints of the edges of the reference triangle, see Figure 6.3. In Figure 6.4, linear and quadratic shape functions are depicted for a one dimensional situation. The two dimensional case is completely analogous.

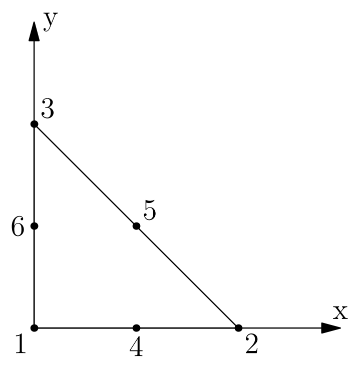


Figure 6.3: Degrees of freedom on the reference triangle: nodal values at $1,2,3$ for linear basis functions and $4,5,6$ at edge mid points for quadratic basis functions.

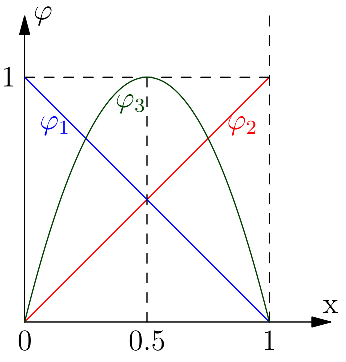


Figure 6.4: The linear shape functions $\varphi_1(x) = 1 - x$ and $\varphi_2(x) = x$ and the quadratic shape function $\varphi_3(x) = 4x(1 - x)$ on the unit interval.

6.2.2 Calculation of element matrices and element vectors, assembly of the system matrix

Similar to other finite element codes, this implementation also employs an elementwise assembly strategy of the required matrices and vectors. The i, j-th element of the matrix A, for example, is defined by

$$A[i, j] := \int_\Omega \nabla\varphi_i \cdot \nabla\varphi_j \, dx.$$

Assume for simplicity that only linear basis functions are considered, i.e. all degrees of freedom correspond to values of the solution in the nodes $a \in \mathcal{N}$ (the extension to quadratic basis functions is done by also considering the midpoints of edges $E \in \mathcal{E}$). The continuous linear basis function φ_a is defined as

$$\varphi_a(\mathbf{x}) := \begin{cases} 1 & \text{in the node } a \\ \textit{linear} & \text{for } \mathbf{x} \in \omega_a \\ 0 & \mathbf{x} \notin \omega_a. \end{cases} \tag{6.1}$$

Therefore, when computing the integral $A[i, j]$, only integration over $\omega_{a(i)} \cap \omega_{a(j)}$ needs to be considered instead of over Ω ($a(i)$ is the node corresponding to the degree of freedom i). This fact is exploited for the elementwise assembly strategy using DOF_T as the set of degrees of freedom corresponding to T:

Algorithm 6.2 (elementwise assembly). $A \in \mathbb{R}^{M_0^u \times M_0^u}$

```
$A[i, j] = 0 \;\; \forall i, j = 1, \ldots, M_0^u$
$A_T \in \mathbb{R}^{|DOF_T| \times |DOF_T|}$
for $T \in \mathcal{T}$ do
    for $i = 1, \ldots, |DOF_T|$ do
        for $j = 1, \ldots, |DOF_T|$ do
            $A_T[i, j] = \int_T \nabla\varphi_{DOF_T(i)} \cdot \nabla\varphi_{DOF_T(j)} \, dx$
        end for
        end for
        for $i = 1, \dots, |DOF_T|$ do
            for $j = 1, \dots, |DOF_T|$ do
                if $DOF_T(i) \in \{1, \dots, M_0^u\}$ then
                    if $DOF_T(j) \in \{1, \dots, M_0^u\}$ then
                        $A[DOF_T(i), DOF_T(j)] += A_T[i, j]$
                    end if
                end if
            end for
        end for
    end for
```

The matrix entry $A_T[i,j]$ is calculated by transformation of the integral to the reference element and application of numerical integration methods as described in Section 6.2.1. The computation of the other matrices in (4.47) works analogously, with the difference that for some of them the velocity $\mathbf{u}_h^i$ at the previous Newton-iteration must be computed first. The pointwise value at $\mathbf{x} \in T$ of this function, using only the degrees of freedom DOF_T corresponding to T, is given by:

$$\mathbf{u}_h^i(\mathbf{x}) = \sum_{j \in \mathrm{DOF}_T} u_j^i \begin{pmatrix} \varphi_j(\mathbf{x}) \\ 0 \end{pmatrix} + v_j^i \begin{pmatrix} 0 \\ \varphi_j(\mathbf{x}) \end{pmatrix}.$$

In the implementation, the local element matrices A^T, M_{ij}^T, A_1, B_1 and B_2 are assembled by the function `calcElMatNavierStokes`. Analogously, the element vectors of the right hand sides RHS_1, RHS_2 and RHS_3 are calculated by `calcElVecNavierStokes`. The function `assembleNavierStokes` assembles the global matrices A, M_{ij},etc. and integrates them into the final system matrix M (see equation (4.47). It returns the final system matrix M and right hand side vector $\underline{r}$. This equation system is then solved by the parallel direct solver *PARDISO*, see [26, 35, 36].
The integrals are integrated by the numerical integration procedure described in Subsection 6.2.1.

6.2.3 Mesh refinement for two space dimensions

The mesh refinement algorithm in use for two space dimensions goes back to [6] and is also described in [43]. The method of choice is the so called red-green-blue refinement (*rgb-refinement*), which offers three possibilities to divide a triangle into new triangles. Those three division strategies are denoted *red*, *green* and *blue*. The advantage of red-green-blue refinement is that with this strategy, hanging nodes can be avoided (i.e. every node in the set $\mathcal{N}$ can only be the endpoint of an edge, but never be part of the interior part of an edge). Furthermore, if a triangle is divided into new triangles, the newly created triangles will always be similar to the original triangle, see Figure 6.5. This ensures that shape regularity in the triangulation is being preserved, as long as the initial triangulation $\mathcal{T}^{(0)}$ is shape regular.
In every one of the three refinement cases, the longest edge, also called the reference

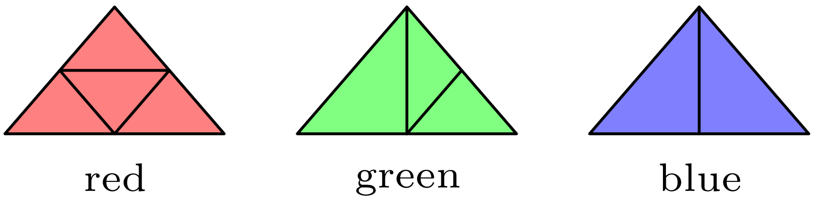


Figure 6.5: The three different possibilities to refine an element T. For green refinement, two variants exist: refining the left upper or the right upper edge

edge, will be divided into two equally long edges.
The initial triangulation and therefore the geometry of the domain Ω is read from a **.msh*-file, which can be produced for example by tools like *NetGen* (see [37]). This file contains node coordinates, elements given by three nodes each and boundary elements (edges) given by two nodes. Each boundary element corresponds to an edge of a triangular element, and for each boundary element, a type of boundary condition is provided (Dirichlet, do-nothing, etc.). We agree on the following necessary conventions (see Figure 6.6):

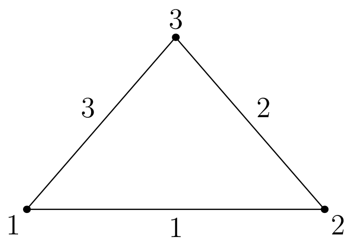


Figure 6.6: Convention by which the nodes and edges are labeled in the data structures which store the triangulation information

- The nodes of an element T are stored in the mathematically positive sense of orientation.
- The first edge of an element T is the first edge after the first node of T in the mathematically positive sense.
- The edges of an element T are labeled in the mathematically positive sense.

Red-Green-Blue refinement

For each refinement step k, we calculate a set $\mathcal{M}_k \subseteq \mathcal{T}_k$ of elements according to a cell selection rule from Subsection 6.1.2. The elements in $\mathcal{M}_k$ have to be refined, but there will be more elements that also have to be refined in order to avoid hanging nodes. Therefore, we first use the following procedure to decide in which way an element $T \in \mathcal{T}_k$ has to be refined (red, green or blue) and store the decisions as marked edges and elements, then use another algorithm (which will not be described in detail) to build up the corresponding data structures representing the new triangulation $\mathcal{T}_{k+1}$.

We use an auxiliary data structure $\mathcal{S}$ to store elements which the algorithm still has to process. The marking is conducted in the following way:

1. All the initially marked elements T in set $\mathcal{M}_k$ are being divided into four new elements, also called *red*-refinement, see Figure 6.5. We mark T and every edge $E \in \mathcal{E}_T$.
2. We perform this and the following steps for all neighbours K of T which have not been marked yet: If a neighbour K of T has not been marked yet, we also mark K for refinement.
3. If the mutual edge $E = T \cap K$ is the longest edge of K, and no other edge of K has been marked, we use *blue*-refinement for K, i.e. K is being marked, but no other edges.
4. If the mutual edge $E = T \cap K$ is the longest edge of K, and exactly one other edge E_2 has been marked, we mark the element K and use *green*-refinement for K.
5. If the mutual edge $E = T \cap K$ is not the longest edge of K, we mark the longest edge and store the neighbour of K over the longest edge in $\mathcal{S}$ in case it is not yet contained in $\mathcal{S}$. If the third edge of K has also been marked, we use *red*-refinement for K.
6. Repeat steps 3 to 5 for all elements T in $\mathcal{S}$, and remove them from $\mathcal{S}$. Do this until $\mathcal{S}$ is empty.

6.3 Adaptive mesh refinement in *AVL-Fire*®

Adaptive mesh refinement is incorporated into the solution process of *AVL-Fire*® in the following way:

Algorithm 6.3. *Define a cell selection rule (see Subsection 6.1.2), a refinement threshold parameter θ, a termination criterion and a refinement intervall $N \in \mathbb{N}$. Set $k := 0$.*

1. *Start with an initial triangulation $\mathcal{T}_0$ and set up the equation system based on the finite volume method described in Chapter 3.*
2. *Calculate iterations of the SIMPLE algorithm in Section 3.5 using triangulation $\mathcal{T}_k$.*
3. *After N iterations, compute an error estimator η_T for all $T \in \mathcal{T}_k$, refine $\mathcal{T}_k$ in order to arrive at a refined triangulation $\mathcal{T}_{k+1}$ according to the chosen cell selection rule and parameter θ. For the first iteration of the SIMPLE algorithm after refinement, interpolation is required to define the velocity field $\mathbf{u}$ on the new mesh based on $\mathbf{u}$ from the last iteration of the SIMPLE algorithm before refinement.*
4. *Set $k := k + 1$ and go to step 1 if the chosen termination criterion has not been met.*

The numerical investigations in *AVL-Fire*® in Chapter 3 were conducted among others using the residual error estimator described in Subsection 5.2.1. We calculate the cell residuals

$$\sum_{j=1}^{n_f} \Big[m_j - \mathbf{A}_j \cdot (\nabla\phi)_j \Big] - |T| \left[(\rho f_k)_T - \left(\frac{\partial p}{\partial x_k} \right)_T \right]$$

and apply a normalization as in Section 5.2.1 in order to arrive at the final residual error estimators:

$$\eta_{R,T}(\phi) = \frac{\mathrm{res}_T(\phi)}{|T| F_{\mathrm{norm}}}.$$

6.3.1 Interpolation of the velocity field after refinement

After the triangulation $\mathcal{T}_k$ has been refined to $\mathcal{T}_{k+1}$, at step 4 in the solution process (described at the beginning of Section 6.3), the values of ϕ correspond to element centers of the old mesh and therefore have to be adopted by interpolation to the new mesh in order to make them usable for the further calculation, see Figure 6.7. The following interpolation techniques are currently in use:

1. *Constant interpolation:* The simplest method. The value of ϕ at the center of the father cell T (the cell before refinement) is taken as the value of ϕ at the center of the child cells T_i (the cells that have been created by refining T).
2. *Taylor expansion:* We use a Taylor-expansion

$$\phi_{T_i} = \phi_T + \nabla\phi_T \cdot \mathbf{d}_i,$$

   where $\mathbf{d}_i$ is the difference vector of the coordinate vectors of the cell centers of T and T_i.

Remark 6.2. *It has to be mentioned that the interpolation process significantly disturbs the convergence of the algorithm, which is because there is no reason why the newly interpolated values should fulfill the continuity or momentum balance equation of the Navier-Stokes system. They can merely be seen as a good initial guesses on a new mesh, based on a solution that fulfills the equations on the old mesh.*

6.3.2 Calculation of the computational error

AVL-Fire® calculates all results in three dimensions. To make the results comparable to the results of the two dimensional finite element method, we make the solution two dimensional by projecting the results onto the x-y-plane. Therefore we assume symmetry boundary conditions in the third dimension, the z-direction. The projection is done in such a way that we take the solution values of cells that are closest to the plane and have positive z-coordinate. In practice, this means that we have to generate the three dimensional meshes in such a way that the element faces are either subsets of

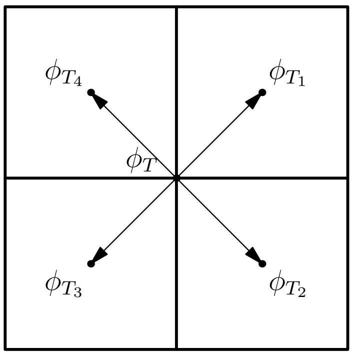


Figure 6.7: Interpolation of values of ϕ to the new triangulation.

the x-y-plane or do not intersect the x-y-plane at all. Furthermore, all element edges have to be parallel to any of the coordinate axis, i.e. only cube shaped elements can be used.
Assume that $\mathcal{T}$ and $\mathcal{T}_{ref}$ are two triangulations that have been projected to the x-y-plane, together with projected solutions $\mathbf{u}_h$ and $\mathbf{u}_{ref}$ for those triangulations, respectively. We assume that for $T \in \mathcal{T}$, either we have $T_r \subset T$ or $T_r \cap T = 0$ for all $T_r \in \mathcal{T}_{ref}$. Since there is no exact solution available for most test cases, we have to compute a reference solution $\mathbf{u}_{ref}$ on a very fine mesh $\mathcal{T}_{ref}$, i.e. for a triangulation of which the global mesh size h is smaller than the local mesh size h_T for all $T \in \mathcal{T}$ by a factor $C \leq 1/2$, such that $h \leq Ch_T$ for all $T \in \mathcal{T}$. The error we want to compare is the L^2-error:

$$\begin{aligned}\|\mathbf{u}_h - \mathbf{u}_{ref}\|^2_{L^2(\Omega)} &= \sum_{T\in\mathcal{T}} \|\mathbf{u}_h - \mathbf{u}_{ref}\|^2_{L^2(T)} = \sum_{T\in\mathcal{T}} \int_T (\mathbf{u}_h - \mathbf{u}_{ref}) \cdot (\mathbf{u}_h - \mathbf{u}_{ref})\, dx \\ &= \sum_{T\in\mathcal{T}} \sum_{\substack{T_r\in\mathcal{T}_{ref},\\ T_r\subset T}} \int_{T_r} (\mathbf{u}_h - \mathbf{u}_{ref}) \cdot (\mathbf{u}_h - \mathbf{u}_{ref})\, dx \\ &= \sum_{T\in\mathcal{T}} \sum_{\substack{T_r\in\mathcal{T}_{ref},\\ T_r\subset T}} |T|((\mathbf{u}_h)_{T_r} - (\mathbf{u}_{ref})_{T_r}) \cdot ((\mathbf{u}_h)_{T_r} - (\mathbf{u}_{ref})_{T_r}).\end{aligned}$$

The last equality holds because $\mathbf{u}_h$ is constant on T (particularly on T_r, since $T_r \subset T$) and $\mathbf{u}_{ref}$ is constant on T_r.

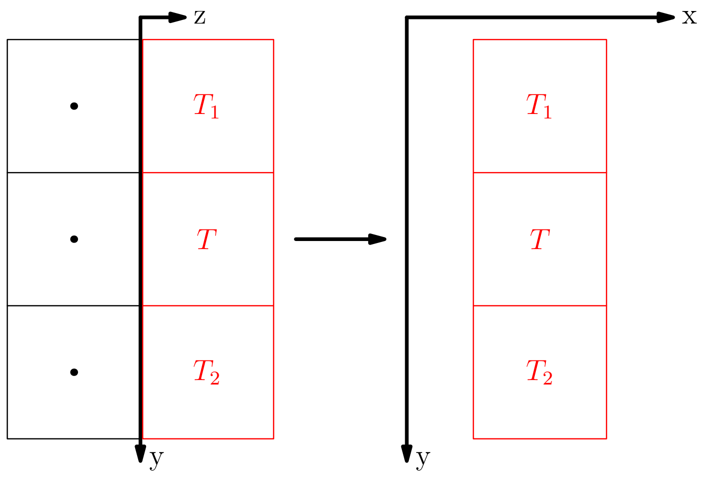


Figure 6.8: Reinterpretation of cube shaped elements as two dimensional square shaped elements in the x-y-plane.

7 Numerical Results

In this chapter we present numerical results for several test cases to demonstrate how the error estimation techniques from Chapter 5 perform under different circumstances. We compare the performance by means of the computational errors $\|u - u_h\|$ and $\|p - p_h\|$ of the velocity and pressure in different norms, the number of elements and degrees of freedom, as well as plots of adaptively generated meshes at several refinement steps during the calculation. Plots of solutions calculated by the finite element method were generated using *ParaView*, Kitware Inc., plots of solutions by the finite volume method by *AVL-Fire*® and plots of the errors were produced using *Python* using the module *matplotlib*.
In order to allow comparison of results, since all calculations in *AVL-Fire*® are performed for three dimensional geometries, while the `C++` implementation calculates the finite element results in two dimensions, we applied symmetry boundary conditions for the third dimension, which is equivalent to having no change of the solution in the third dimension and therefore equivalence to a two dimensional situation.

7.1 Analytical solution

In this example, we compute a flow field where the solution $(\mathbf{u}, p)$ is already known. We choose the unit square as the computational domain, i.e. $\Omega = (0,1)^2$. The solution is a smooth function and fulfills the Dirichlet boundary condition $\mathbf{g} = (0,0)$:

$$\mathbf{u}(x,y) = \begin{bmatrix} \sin(2\pi y)\,[\cos(2\pi x) - 1] \\ \sin(2\pi x)\,[1 - \cos(2\pi y)] \end{bmatrix}$$

with the pressure

$$p(x,y) = \sin(2\pi x)\cos(2\pi y).$$

We choose $\nu = 1$ and substitute this exact solution in the Navier-Stokes equation

$$-\Delta \mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u} + \nabla p = \mathbf{f}$$

in order to get an expression for $\mathbf{f}$. When choosing exact solutions, one has to be careful that the solution fulfills the continuity condition $\nabla \cdot \mathbf{u} = 0$, which is the case here. Furthermore,

$$\int_\Omega p(\mathbf{x})\; dx = 0$$

needs to hold, since the finite element method in Chapter 4 is constructed under that assumption in order to fix the pressure which would otherwise only be unique up to constants.

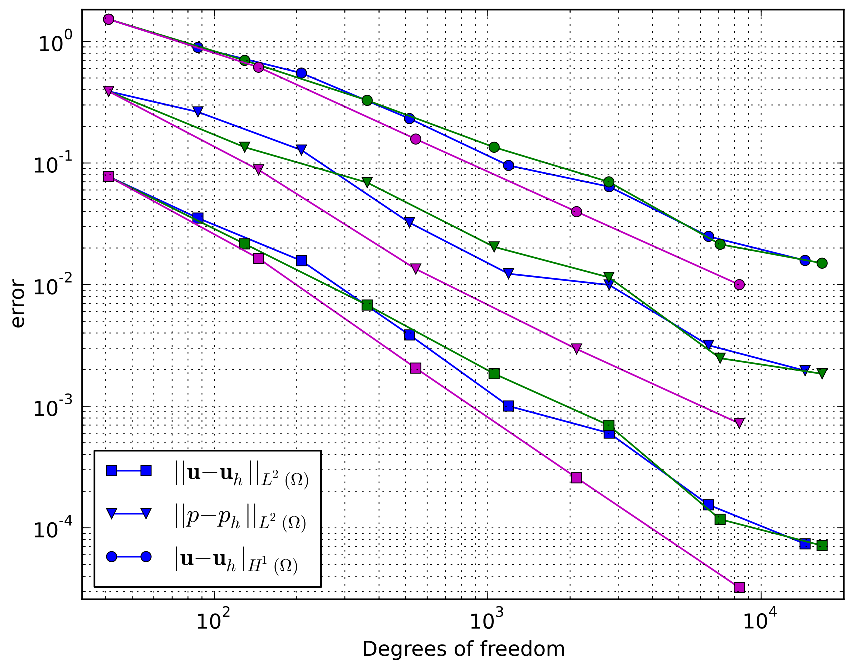


Figure 7.1: Taylor-Hood elements. Comparison of error drop for different cell selection criteria: Quantile criterion with $\theta = 0.35$ (blue), Dörfler marking with $\theta = 0.85$ (green) and uniform refinement (magenta). We compare degrees of freedom versus $\|\mathbf{u} - \mathbf{u}_h\|_{L^2(\Omega)}$ (squares), $\|p - p_h\|_{L^2(\Omega)}$ (triangles) and $|\mathbf{u} - \mathbf{u}_h|_{H^1(\Omega)}$ (circles).

7.1.1 Finite element simulation

We perform the calculation using quadratic-linear Taylor-Hood elements and compare error convergence rates for adaptive refinement with different cell selection criteria, namely the quantile criterion and Dörfler marking. We use the residual error estimator as error estimation technique. Furthermore, we add results using uniform refinement.
In Figure 7.1, for Taylor-Hood elements, we see that uniform refinement (magenta colored line) performs best for all types of error, while the quantile criterion (blue line) performed slightly better than Dörfler marking (green line). In that case, adaptive refinement offers no improvement. The orders of convergence are $\mathcal{O}(h^3)$ for $\|\mathbf{u} - \mathbf{u}_h\|_{L^2(\Omega)}$, $\mathcal{O}(h^2)$ for $\|p - p_h\|_{L^2(\Omega)}$ and $\mathcal{O}(h^2)$ for $|\mathbf{u} - \mathbf{u}_h|_{H^1(\Omega)}$, which is exactly what the results from Section 4.1.3 predict.
Similar results are true for the linear-linear elements $P1$-$P1$, Figure 7.2, although the convergence rate seems to be more stable for those elements. As orders of convergence we get $\mathcal{O}(h^2)$ for $\|\mathbf{u} - \mathbf{u}_h\|_{L^2(\Omega)}$, $\mathcal{O}(h^{1.5})$ for $\|p - p_h\|_{L^2(\Omega)}$ and $\mathcal{O}(h)$ for $|\mathbf{u} - \mathbf{u}_h|_{H^1(\Omega)}$.

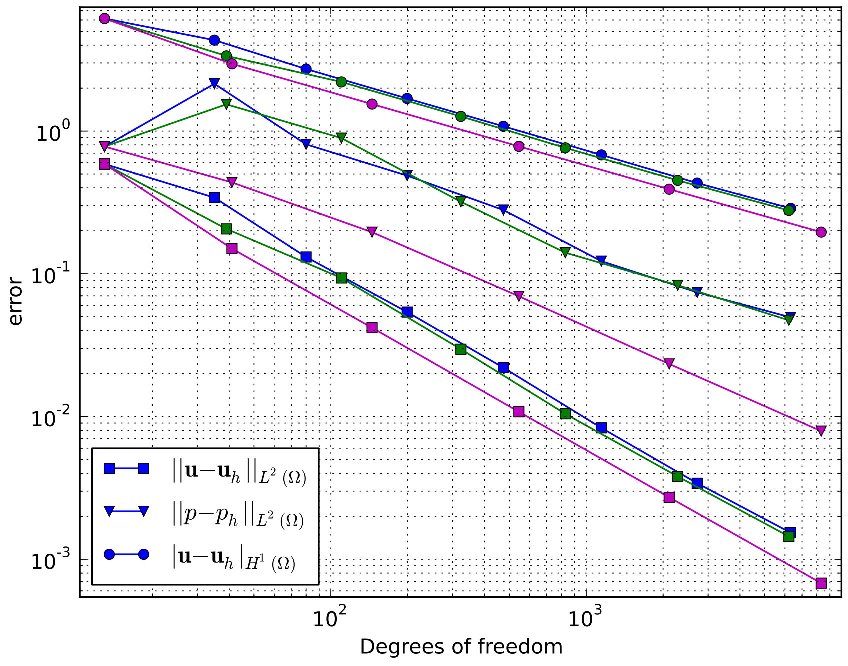


Figure 7.2: $P1$-$P1$ elements. Comparison of error drop for different cell selection criteria: Quantile criterion with $\theta = 0.35$ (blue), Dörfler marking with $\theta = 0.85$ (green) and uniform refinement (magenta). We compare degrees of freedom versus $\|\mathbf{u} - \mathbf{u}_h\|_{L^2(\Omega)}$ (squares), $\|p - p_h\|_{L^2(\Omega)}$ (triangles) and $|\mathbf{u} - \mathbf{u}_h|_{H^1(\Omega)}$ (circles).

7.2 Lid driven cavity flow

In the first example we investigate the flow of a fluid in a square-shaped box. This example is used very often in the literature about computational fluid dynamics to demonstrate certain behaviors of numerical methods. The flow is driven by drawing an imaginary lid over the box at the upper edge Γ_ℓ (red edge, see Figure 7.3) from left to right with velocity U_0. We assume a *no-slip* boundary, which means that because of friction, the fluid very close to the boundary assumes the same velocity as the boundary itself. Therefore, the moving lid at the top of the box introduces momentum into the system and as a consequence generates a vortex rotating clockwise because of mass inertia. We expect the error erstimator to induce refinement primarily in the top part of the box, particularly the corners, as well as the area covering the vortex.
For the setup of the example, we apply the boundary condition $\mathbf{g} = (U_0, 0)$ to the lid Γ_ℓ and $\mathbf{g} = (0, 0)$ on the rest of the boundary $\Gamma \backslash \Gamma_\ell$. We calculate the results for Reynold's number $Re = 400$. Reynold's number is used for comparison of flows with different characteristic velocity U_0 but for shape similar geometries. It is computed by

$$Re = \frac{U_0 L}{\nu},$$

where ν is the kinematic viscosity $\nu := \mu/\rho$ (μ being the dynamic viscosity and ρ the density, which are properties of the fluid) and L the characteristic length. More information on Reynold's number can be found in [19].

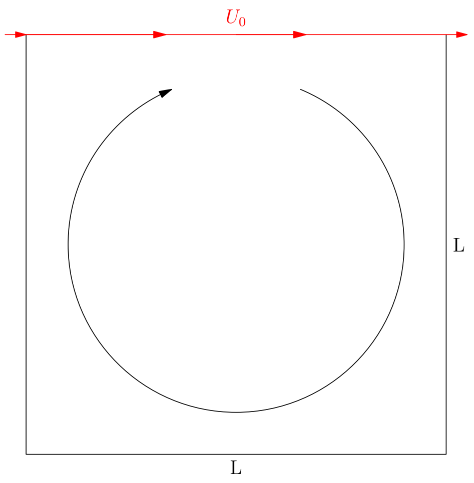
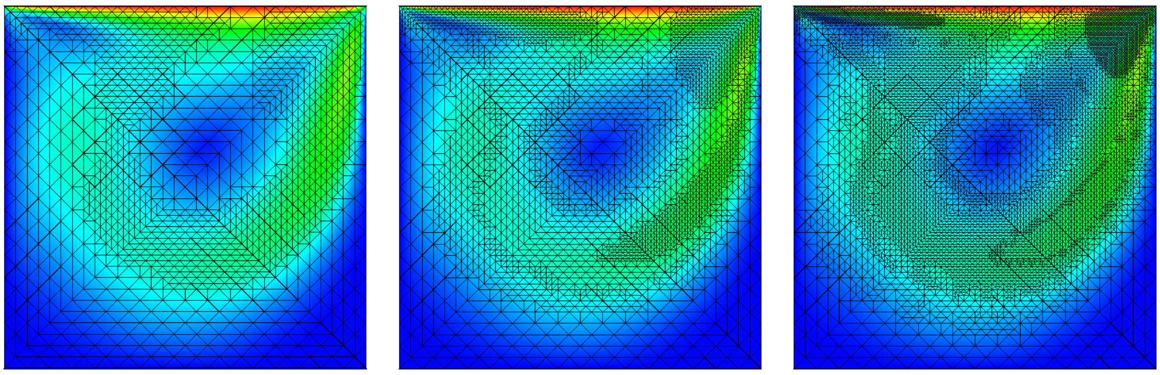


Figure 7.3: Lid driven cavity flow. An imaginary lid (red arrows) on the upper edge of the box is drawn from left to right with constant velocity U_0.

7.2.1 Finite element simulation

Simulation setup

The simulation with the finite element method is carried out on the unit-square $(0, 1)$ as the computational domain Ω. The characteristic length is $L = 1$. Further, we



(a) $L = 1$, $|\mathcal{T}| = 2296$ (b) $L = 2$, $|\mathcal{T}| = 5240$ (c) $L = 3$, $|\mathcal{T}| = 11836$

Figure 7.5: Adaptive mesh refinement using the residual error estimator (5.13) at three refinement steps.

choose $\nu = 1$. Reynold's number then simply corresponds to the velocity of the lid U_0:

$$Re = \frac{U_0 L}{\nu} = U_0 \overset{!}{=} 400.$$

The cell selection criterion in use is the quantile criterion, see Section 6.1.2 with refinement threshold parameter $\theta = 0.35$, thus refining at least 35% of all elements with the largest values for the error estimator.
We plot the meshes that were produced at different refinement steps, using the residual error estimator 5.13. The grid in Figure (7.4) was used as the initial triangulation because it is symmetric, but not without refining it four times uniformly before calculation by dividing each triangle into four shape similar subtriangles. Quadratic-linear Taylor Hood elements were use for this simulation.

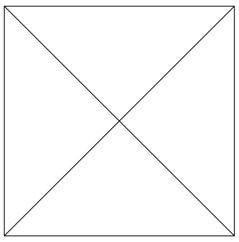


Figure 7.4: Symmetric initial triangulation. For lid driven cavity flow, the mesh was uniformly refined four times before using it for the calculation.

7.2.2 Finite volume simulation

Simulation setup

The flow simulation in *AVL-Fire*® is performed using a cuboid-shaped geometry of dimensions $0.4m \times 0.4m$ (the third dimension is unimportant because of symmetry boundary conditions). With the following material parameters for the fluid

$$\mu = 0.001\,\frac{Ns}{m^2}$$
$$\rho = const. = 1000\,\frac{kg}{m^3},$$

we set the lid velocity U_0 to be

$$U_0 = \frac{\nu Re}{L} = \frac{\frac{\mu}{\rho}400}{0.4\,m} = 0.001\,\frac{m}{s},$$

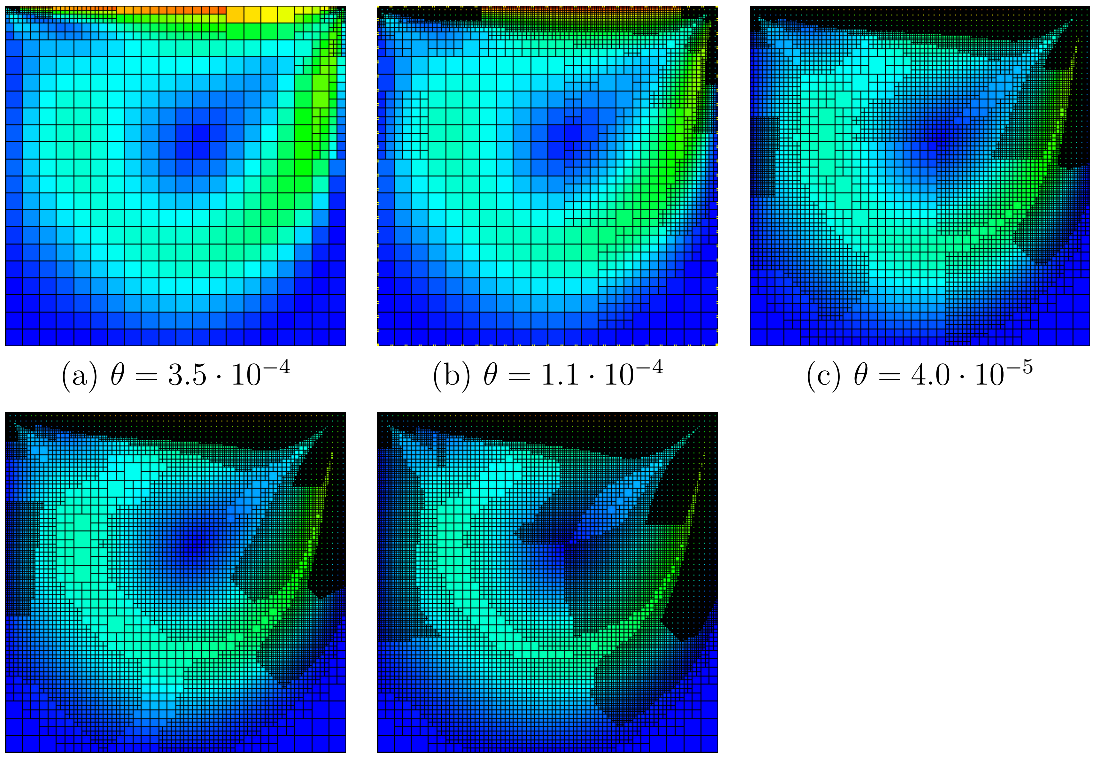
in order to achieve a flow of Reynold's number 400.
For lid driven cavity flow with the finite volume method, meshes generated by different error estimators at several refinement steps were plotted. Furthermore, the results were compared by computing the difference of the solutions in the $L^2(\Omega)$-norm to a reference solution $\mathbf{u}_{ref}$ on a very fine reference mesh $\mathcal{T}_{ref}$. The reference mesh was composed of cube-shaped elements of side length 0.00125 m, while the side length of elements for adaptive refinement was limited to a minimum of 0.0025 m.
As cell selection criterion the maximum value rule was used, see Section 6.1.2. The reasons for this choice are of practical nature, since the maximum value rule offers the greatest control: Once the error estimator for each element is below a threshold θ, no further refinement is conducted and iterations of the SIMPLE-algorithm are carried out until the convergence criterion is met.

Adaptively generated meshes

- Figure 7.6, *neighbour difference criterion, see Section 5.2.3*: The estimator compares values of the difference of the velocity between an element and its neighbours. Refinement in areas of big velocity change, like the vicinity to the lid or the vortex is visible, as well as in the corners at the top of the computational domain.
- Figure 7.7, *third order interpolation, see Section 5.2.2*: Similarly, for third order polynomial interpolation we also get refinement in the same areas, but the estimator seems to cover the whole vortex instead of just areas of bigger change in variables.
- Figure 7.8, *residual error estimator, see Section 5.2.1*: Similar to before. The main difference is here, that also the area close to the right edge of the computational domain is being refined, which is because of a change in the pressure. The residual error estimator is the only estimator that also factors in changes in the pressure.

Comparison of computational errors

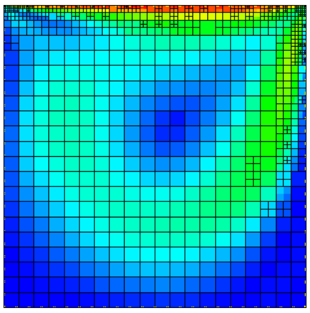
- Figure 7.9: Plot of the number of elements N versus the error $\|\mathbf{u}_h - \mathbf{u}_{ref}\|_{L^2(\Omega)}$. The results are similar, no clear advantage of one method.

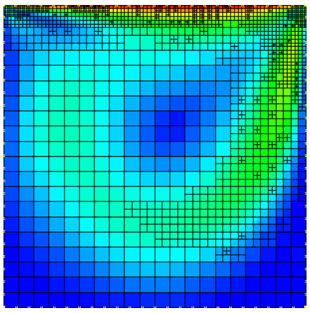


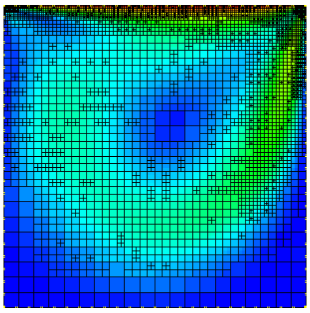
(a) $\theta = 3.5 \cdot 10^{-4}$ (b) $\theta = 1.1 \cdot 10^{-4}$ (c) $\theta = 4.0 \cdot 10^{-5}$

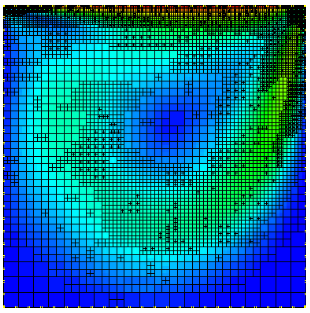
(d) $\theta = 3.5 \cdot 10^{-5}$ (e) $\theta = 2.5 \cdot 10^{-5}$

Figure 7.6: Adaptive mesh refinement using the neighbour difference estimator, Section 5.2.3, measuring the difference in the velocity between neighbouring cells.


(a) $\theta = 2.5 \cdot 10^{-5}$


(b) $\theta = 1.0 \cdot 10^{-5}$


(c) $\theta = 5.0 \cdot 10^{-6}$


(d) $\theta = 2.5 \cdot 10^{-6}$

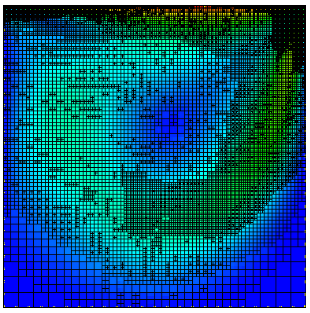
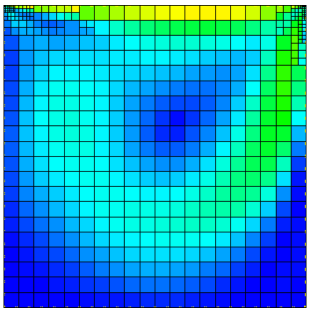
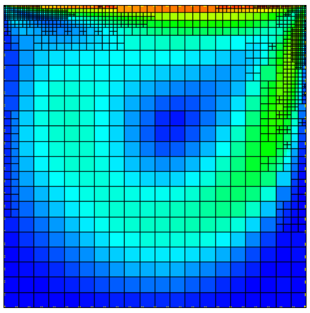
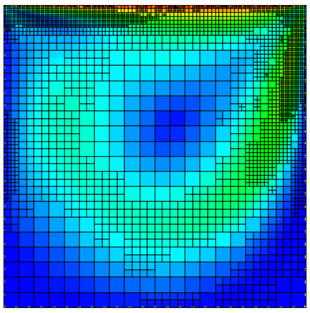
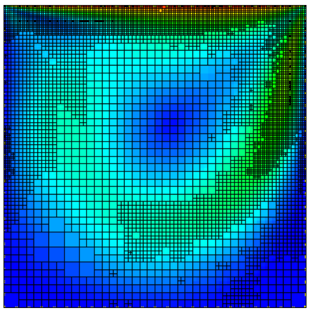

(e) $\theta = 1.0 \cdot 10^{-6}$

Figure 7.7: Adaptively refined meshes using third order interpolation for error estimation.


(a) $\theta = 2.5 \cdot 10^{-5}$


(b) $\theta = 7.5 \cdot 10^{-6}$


(c) $\theta = 2.5 \cdot 10^{-6}$


(d) $\theta = 1.0 \cdot 10^{-6}$

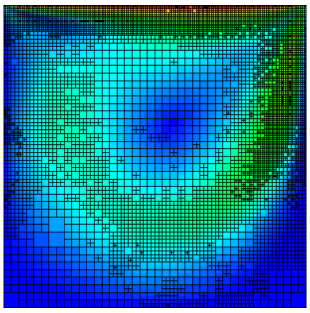

(e) $\theta = 7.5 \cdot 10^{-7}$

Figure 7.8: Adaptively refined meshes using the residual error estimator.

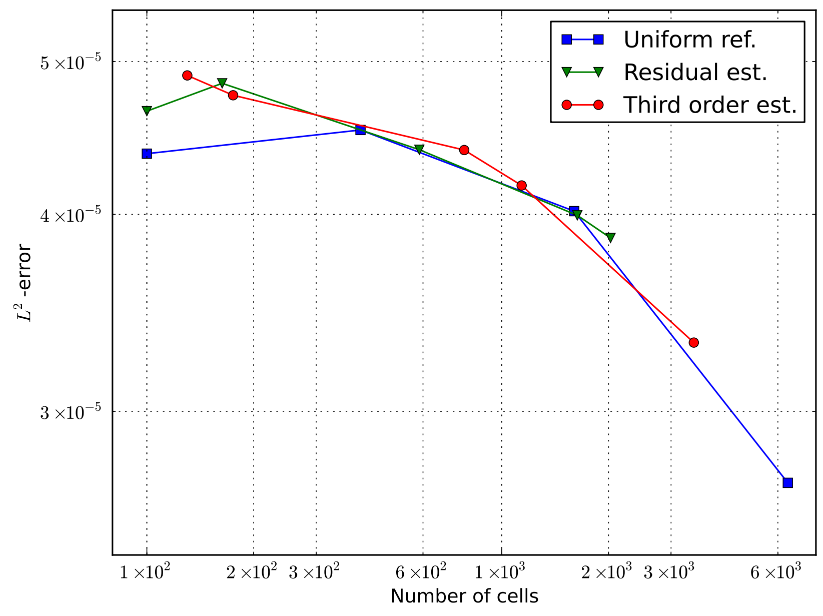


Figure 7.9: Number of elements N versus the error $\|\mathbf{u}_h - \mathbf{u}_{ref}\|_{L^2(\Omega)}$.

- Figure 7.10: Plot of the number of elements N versus the maximum over all elements T of the error $\|\mathbf{u}_h - \mathbf{u}_{ref}\|_{L^2(T)}$. There is apparently no advantage of the adaptive method over uniform refinement.
- Figure 7.11: Plot of the number of elements N versus the mean over all elements T of the error $\|\mathbf{u}_h - \mathbf{u}_{ref}\|_{L^2(T)}$. The residual estimator seems to perform better than uniform refinement, while third order interpolation achieves at least similar results.

Comparison of iteration numbers

The biggest advantage of adaptive methods becomes obvious when comparing numbers of iterations of the SIMPLE-algorithm for different refinement strategies. In Figure 7.12 we can see that the number of necessary SIMPLE-iterations increases much slower for the adaptive strategies. This could be due to the fact that the *AVL-Fire*® -algorithm here uses $\mathbf{u}_0 = \mathbf{0}$ as an initial guess. Since an adaptive strategy starts on a coarse mesh, a developed flow profile is achieved with less iterations than on a fine grid. This flow profile is an initial guess for the next adaptively refined mesh after the solution has been interpolated, which is why subsequent calculations need less iterations to reach a developed flow profile on the new mesh.

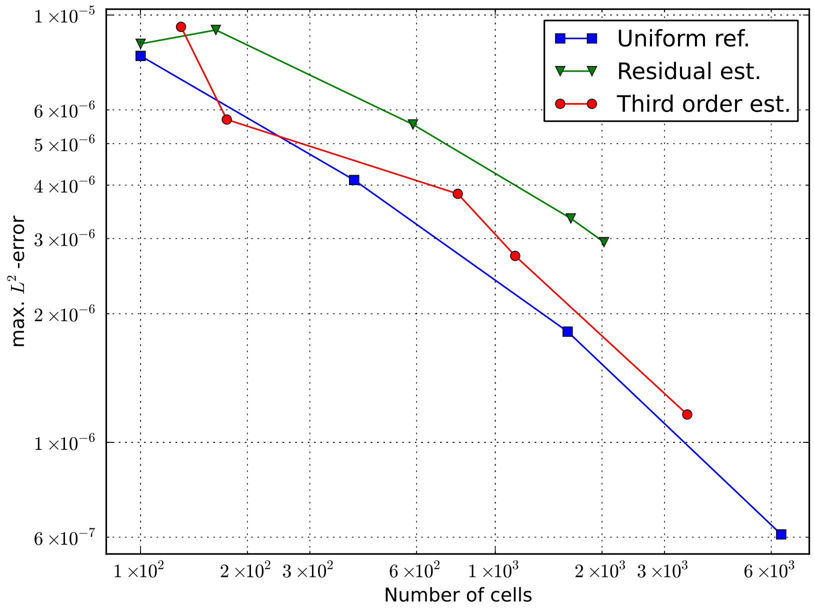


Figure 7.10: Number of elements N versus the maximum over all elements T of the error $\|\mathbf{u}_h - \mathbf{u}_{ref}\|_{L^2(T)}$.

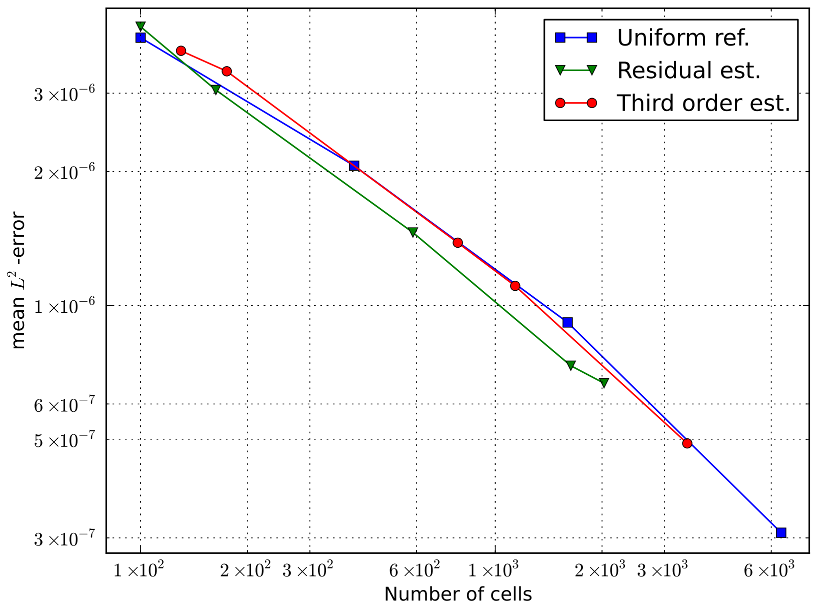


Figure 7.11: Number of elements N versus the mean over all elements T of the error $\|\mathbf{u}_h - \mathbf{u}_{ref}\|_{L^2(T)}$.

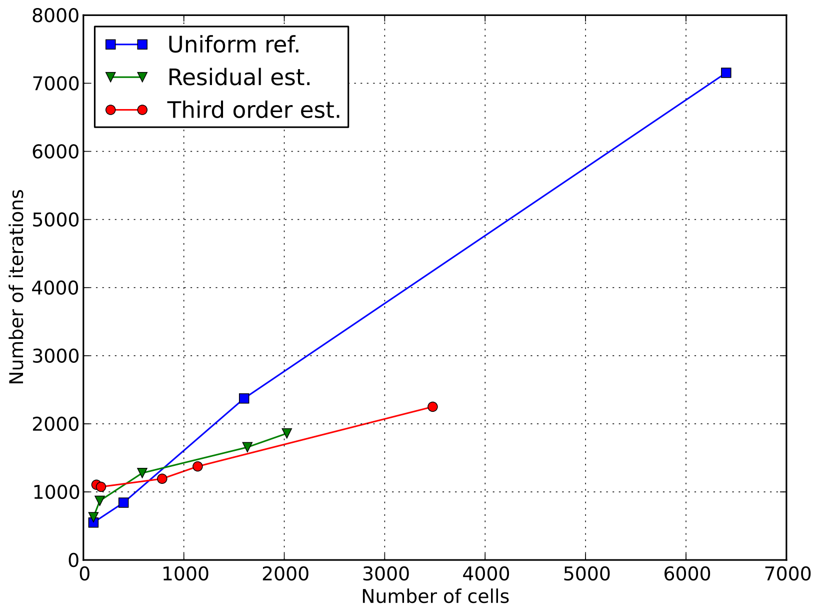


Figure 7.12: Number of elements N versus the number of iterations of the SIMPLE-algorithm needed to meet the same convergence criteria.

7.3 Backward facing step

The next example is the simplest one containing a flow around a corner. It was chosen to investigate how the method behaves at such corner singularities. We define an inflow region at the left side of a box (red edge and arrows) and an outflow region at the right side of the box (green edge and arrows), see Figure 7.13, thus simulating the flow of a fluid through a narrow channel. For the inflow, we choose the flow direction to be normal to the boundary and to be of constant magnitude U_0 along the whole edge. Another approach would be to define a parabolic flow profile. The box itself has length L as its horizontal dimension (x-direction) and height H (y-direction) as its vertical dimension. This means, that on the left side we apply the Dirichlet condition $\mathbf{g} = (U_0, 0)$. At the outflow region, we apply a do-nothing boundary condition. On all other parts of the boundary we define the velocity to be zero, i.e. Dirichlet boundary condition $\mathbf{g} = (0, 0)$. The volumetric forces / sources $\mathbf{f}$ are chosen to be zero.

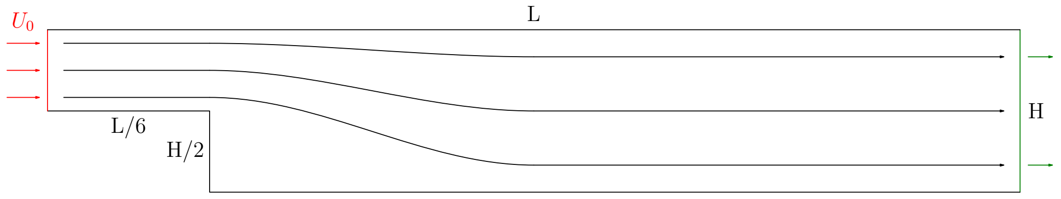


Figure 7.13: The backward facing step. Inflow with velocity of absolute value U_0 on the left (red edge and arrows), outflow region on the right side (green edge and arrows).

For adaptive mesh refinement, we particularly expect our error estimation techniques to introduce refinement on the left part of the channel. There is a uniform flow profile at the inflow edge defined as the boundary condition, therefore the flow has to develop to a more physically natural profile with higher velocities closer to the middle of the channel. There will be a pressure drop around the corner as well as a vortex circulating in clockwise direction in the region directly after and below the corner. Thus, we expect refinement not only in the area where the flow develops, but also around the corner.

7.3.1 Finite element simulation

Simulation setup

The finite element simulation is carried out with the following dimensions for the computational domain:

$$L = 1.2\ m$$
$$H = 0.2\ m.$$

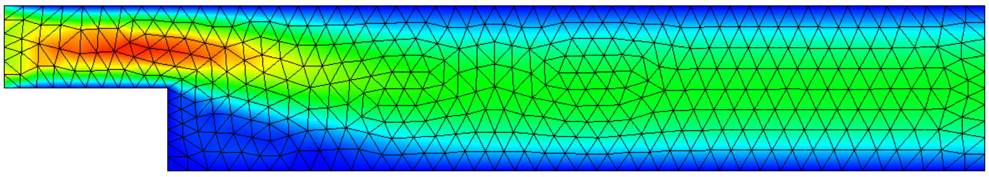
Together with the parameters

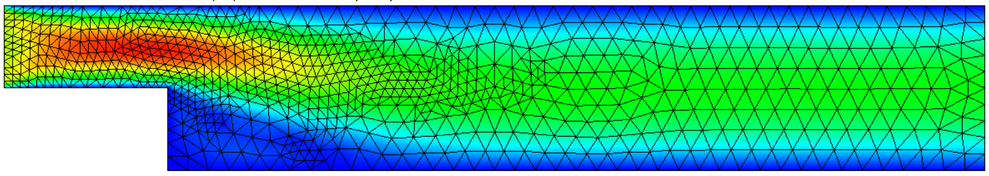
$$\mu = 1\ Ns/m^2$$
$$\rho = 1\ kg/m^3$$

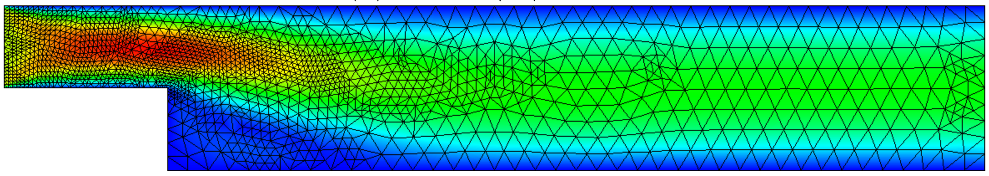
we need to chose $U_0 = 300$ if we want to have Reynold's number $Re = 360$:

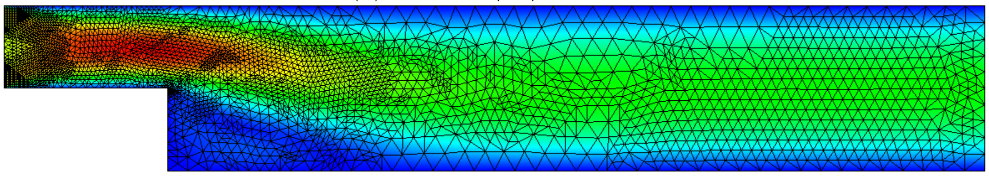
$$U_0 = \frac{\nu Re}{L} = 300.$$

As cell selection criterion we chose the quantile criterion, analogously to the lid driven cavity flow test case, with $\theta = 0.25$ and therefore refining more than 25% of all elements in each refinement step. The initial triangulation in subplot (a) of Figure 7.14 was produced by *NetGen* by choosing 0.05 as the maximum cell size. We use $P1 - P1$ elements for this simulation: They allow us to have more elements for the same number of degrees of freedom, such that it is easier to see where the emphasis for refinement lies.


(a) $L = 0$, $|\mathcal{T}| = 804$. Initial triangulation.


(b) $L = 1$, $|\mathcal{T}| = 1570$


(c) $L = 2$, $|\mathcal{T}| = 3152$


(d) $L = 3$, $|\mathcal{T}| = 6432$

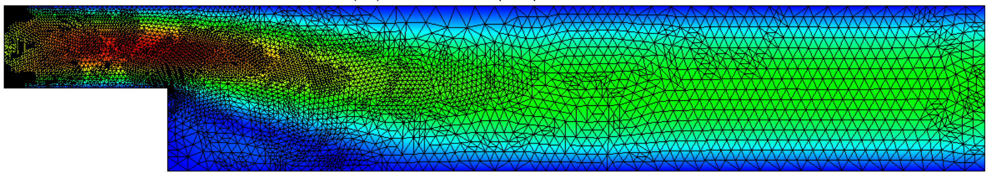

(e) $L = 4$, $|\mathcal{T}| = 13219$

Figure 7.14: Adaptive mesh refinement using the residual error estimator. Refinement directly around the corner is clearly visible, as well as in the low-velocity area (blue) after and below the corner.

7.3.2 Finite volume simulation

Simulation setup

For the simulation in *AVL-Fire®* , we chose the following dimensions for the problem:

$$L = 1.2\ m$$
$$H = 0.2\ m.$$

Together with

$$U_0 = 0.0003\ m/s$$
$$\mu = 0.001\ Ns/m^2$$
$$\rho = 1000\ kg/m^3$$

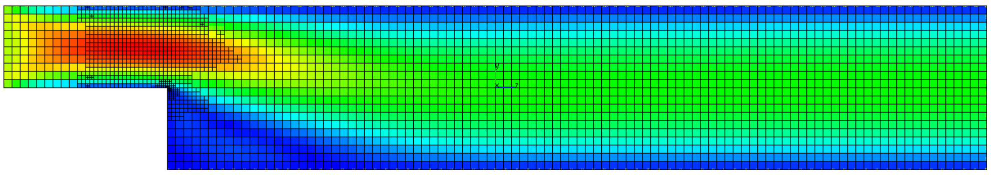
we get Reynold's number

$$Re = \frac{U_0 L}{\nu} = 300\ L = 360.$$

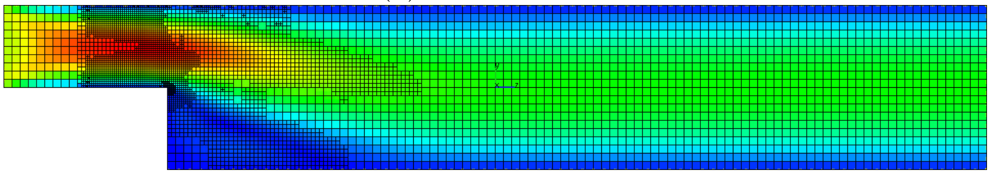
Again, we plot meshes generated by adaptive mesh refinement using the residual error estimator and third order interpolation as error estimation techniques. In contrary to lid driven cavity flow, we do not investigate further the performance of the neighbour difference criterion, since this refinement strategy placed no emphasis on the calculation region behind the corner.
The leftmost part of the computational domain was left out of the scope of the adaptive refinement process, since uniform velocity as boundary condition like in this case does not appear often in practice.

Adaptively generated meshes

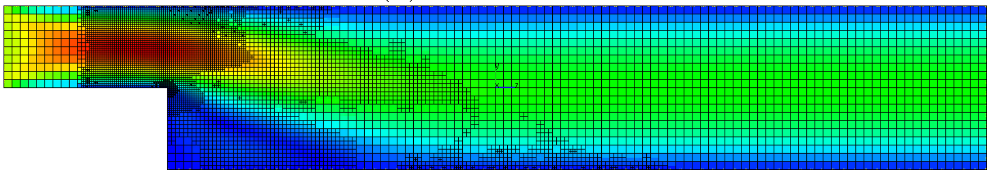
- Figure 7.15, *residual error estimator, see Section 5.2.1*: We see refinement at the left part where the flow develops, as well as refinement around the corner singularity early on. Further refinement is placed in the direction of the flow and in the area below the corner (blue area) where a clockwise flowing vortex of low velocity develops because of a pressure drop. The residual is better capable of capturing this area because it factors in the pressure, while the third order interpolation techniques do not. The generated meshes looks similar to the meshes produced by the finite element simulation, Figure 7.14.
- Figure 7.16, *third order interpolation, see Section 5.2.2*: More focus for refinement is laid in the direction of the flow. The backflow vortex in the area after and below the corner is not captured that well as by the residual estimator.



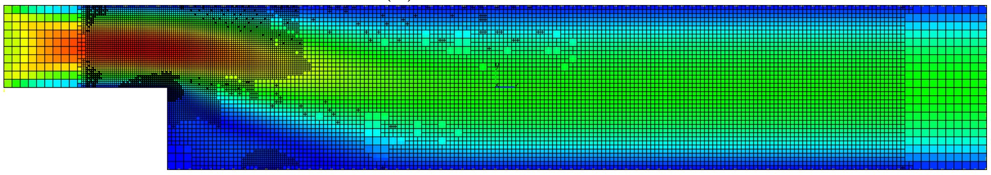
(a) $\theta = 5.0 \cdot 10^{-6}$



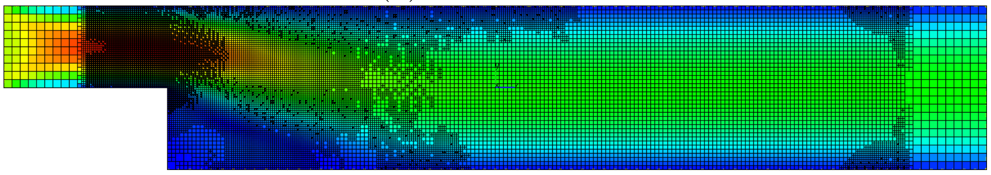
(b) $\theta = 2.5 \cdot 10^{-6}$



(c) $\theta = 1.0 \cdot 10^{-6}$

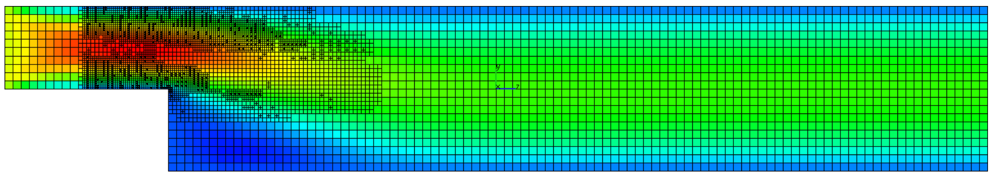


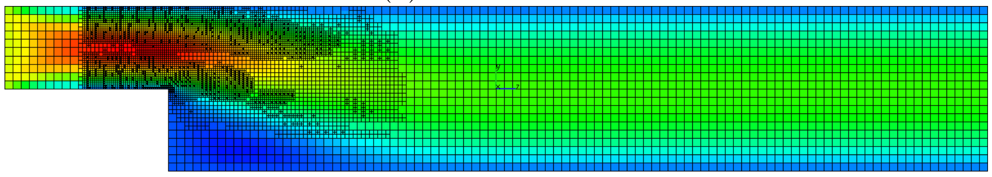
(d) $\theta = 7.5 \cdot 10^{-7}$

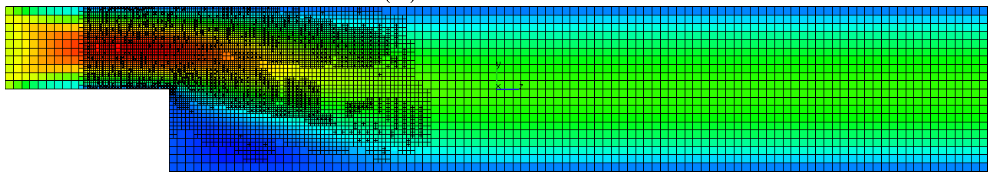


(e) $\theta = 5.0 \cdot 10^{-7}$

Figure 7.15: Adaptive mesh refinement using the residual error estimator. Refinement directly around the corner is clearly visible.


(a) $\theta = 2.5 \cdot 10^{-6}$


(b) $\theta = 1.0 \cdot 10^{-6}$


(c) $\theta = 6.0 \cdot 10^{-7}$

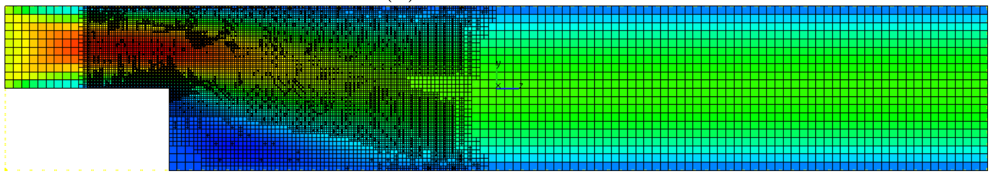

(d) $\theta = 2.0 \cdot 10^{-7}$

Figure 7.16: Adaptive mesh refinement using third order polynomial interpolation (Section 5.2.2). Less emphasis is laid by this estimator on the region behind and below the corner than by the residual estimator.

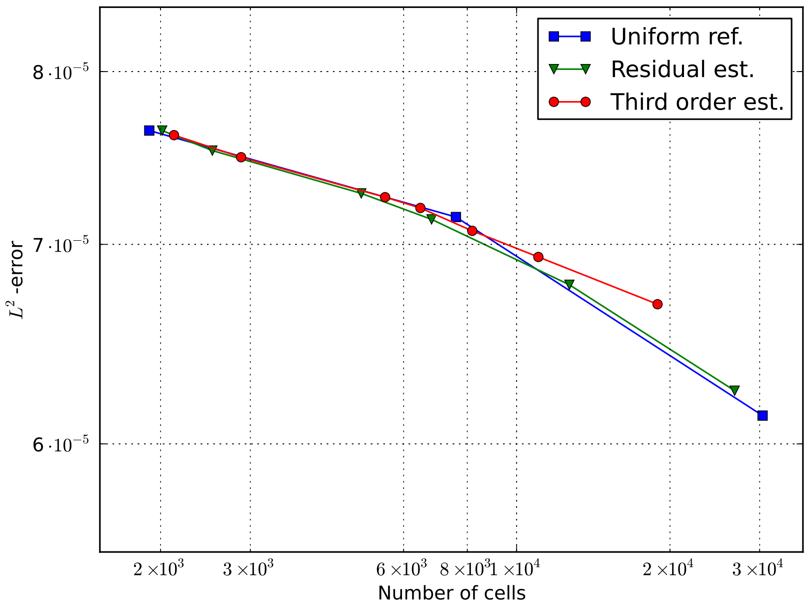


Figure 7.17: Number of elements N versus the error $\|\mathbf{u}_h - \mathbf{u}_{ref}\|_{L^2(\Omega)}$.

Comparison of computational errors

- Figure 7.17: Plot of the number of elements N versus the error $\|\mathbf{u}_h - \mathbf{u}_{ref}\|_{L^2(\Omega)}$. Overall performance in the sense of a drop of $L^2(\Omega)$-error seems to stay the same. Third order interpolation performs worse at later iterations, probably because the right half of the channel was left untouched by refinement which would also contribute to the error.
- Figure 7.18: Plot of the number of elements N versus the maximum over all elements T of the error $\|\mathbf{u}_h - \mathbf{u}_{ref}\|_{L^2(T)}$. Adaptive refinement seems to be performing worse than uniform refinement concerning reduction of the highest $L^2(T)$-error per element T.
- Figure 7.19: Plot of the number of elements N versus the mean over all elements T of the error $\|\mathbf{u}_h - \mathbf{u}_{ref}\|_{L^2(T)}$. The mean $L^2(T)$-error seems to drop faster with adaptive refinement than with uniform refinement.

Comparison of iteration numbers

Even more than in the case of lid driven cavity flow, in the backward facing step case adaptive refinement provides a huge improvement in terms of iteration numbers, as can be seen in Figure 7.20. This again is probably due to multilevel effects, i.e. solutions computed on coarser meshes are better initial guesses for the solution process on finer meshes.

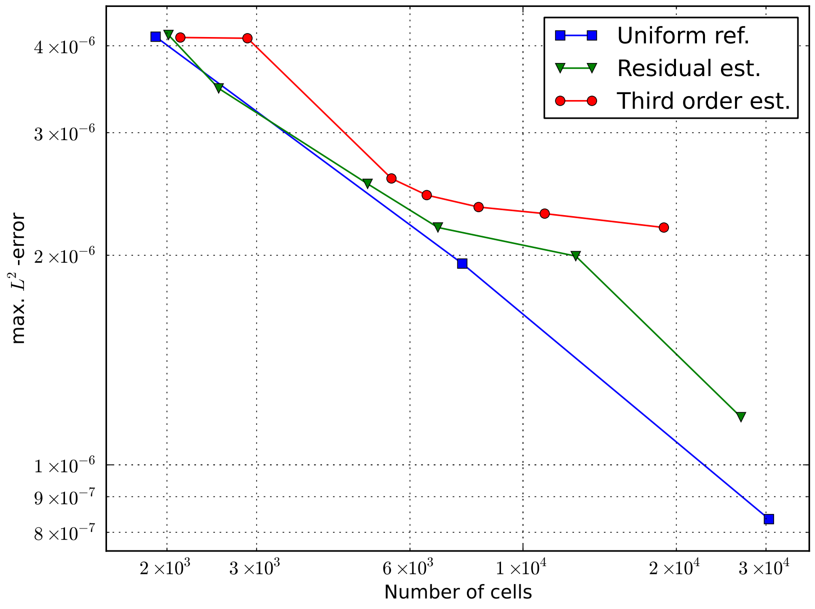


Figure 7.18: Number of elements N versus the maximum over all elements T of the error $\|\mathbf{u}_h - \mathbf{u}_{ref}\|_{L^2(T)}$.

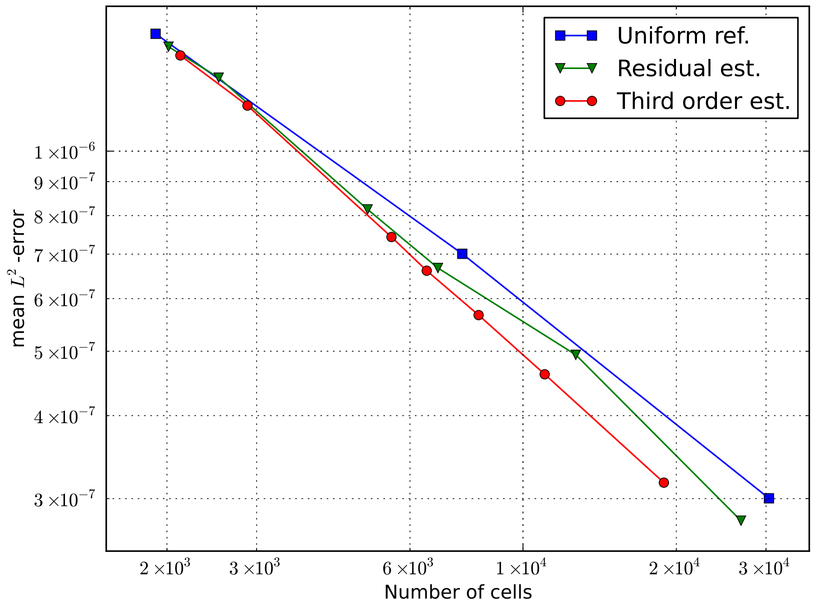


Figure 7.19: Number of elements N versus the mean over all elements T of the error $\|\mathbf{u}_h - \mathbf{u}_{ref}\|_{L^2(T)}$.

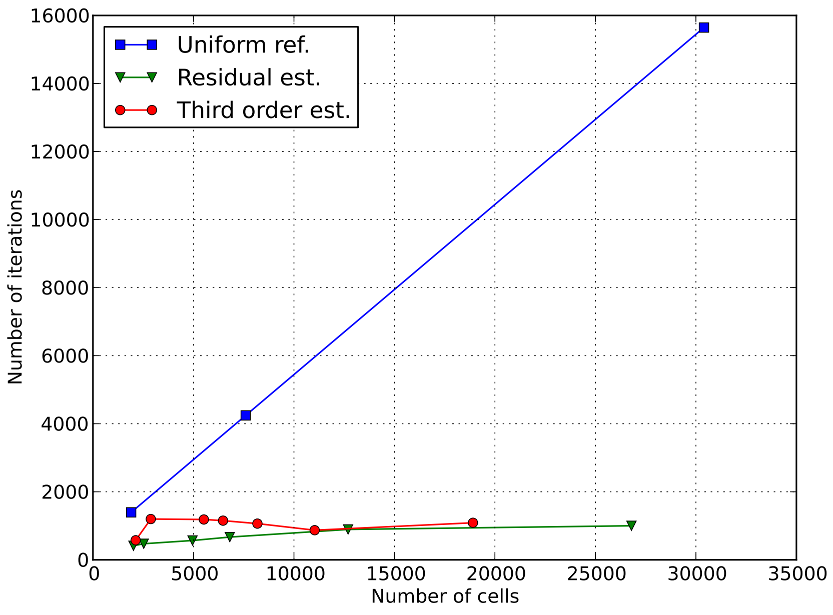


Figure 7.20: Number of elements N versus the number of iterations of the SIMPLE-algorithm needed to meet the same convergence criteria.

8 Conclusions

Based on Chapter 7 we can observe the following effects when using adaptive mesh refinement:
Finite volume simulation: Adaptive mesh refinement offers a great possibility to massively reduce the necessary number of iterations due to multilevel effects, i.e. solutions on coarser meshes are good initial guesses for the further simulation on finer meshes. Furthermore, the residual error estimator is better able to capture physical details of the flow like the small vortex behind and below the corner in the backward facing step test case than the other error estimation techniques, see Section 7.3, which may or may not be desirable depending on the purpose of the simulation. Its main difference compared to other error estimation techniques is that it also factors in the pressure into the error estimation process, which the other presented estimators do not. An improvement in error decline for simulations employing adaptive refinement when comparing to a reference solution on a very fine grid was not observable as opposed to uniform refinement.
Finite element simulation: It was shown that the residual error estimator for finite elements induces refinement similar to the refinement induced by the residual error estimator for the finite volume method. When applying adaptive refinement using the residual error estimator for the solution of an example with a known smooth exact solution, no improvement of convergence rates could be observed. The number of necessary Newton iterations stayed the same regardless of the number of elements and whether uniform or adaptive refinement was used.

Bibliography

[1] M. Ainsworth and A. Craig. A posteriori error estimators in the finite element method. *Numer. Math.*, 60(4):429–463, 1992.

[2] M. Ainsworth and J. T. Oden. A posteriori error estimation in finite element analysis. *Comput. Methods Appl. Mech. Engrg.*, 142(1-2):1–88, 1997.

[3] I. Babuška and W. C. Rheinboldt. Error estimates for adaptive finite element computations. *SIAM J. Numer. Anal.*, 15(4):736–754, 1978.

[4] I. Babuška, T. Strouboulis, and C. S. Upadhyay. A model study of the quality of a posteriori error estimators for finite element solutions of linear elliptic problems, with particular reference to the behavior near the boundary. *Internat. J. Numer. Methods Engrg.*, 40(14):2521–2577, 1997.

[5] I. Babuška, T. Strouboulis, C. S. Upadhyay, S. K. Gangaraj, and K. Copps. Validation of a posteriori error estimators by numerical approach. *Internat. J. Numer. Methods Engrg.*, 37(7):1073–1123, 1994.

[6] R. E. Bank. The efficient implementation of local mesh refinement algorithms. *SIAM J. Numer. Anal.*, 1983.

[7] R. E. Bank and A. Weiser. Some a posteriori error estimators for elliptic partial differential equations. *Math. Comp.*, 44(170):283–301, 1985.

[8] R. E. Bank and B. D. Welfert. A posteriori error estimates for the Stokes equations: a comparison. *Comput. Methods Appl. Mech. Engrg.*, 82(1-3):323–340, 1990. Reliability in computational mechanics (Austin, TX, 1989).

[9] R. E. Bank and B. D. Welfert. A posteriori error estimates for the Stokes problem. *SIAM J. Numer. Anal.*, 28(3):591–623, 1991.

[10] G. K. Batchelor. *An introduction to fluid dynamics.* Cambridge Mathematical Library. Cambridge University Press, Cambridge, paperback edition, 1999.

[11] C. Bernardi, B. Métivet, and R. Verfürth. Analyse numérique d'indicateurs d'erreur, in maillage et adaptation. *P.-L. George ed., Hermes*, pages 251–278, 2001.

[12] P. B. Bochev, C. R. Dohrmann, and M. D. Gunzburger. Stabilization of low-order mixed finite elements for the Stokes equations. *SIAM J. Numer. Anal.*, 44(1):82–101 (electronic), 2006.

[13] F. Brezzi and M. Fortin. *Mixed and hybrid finite elements methods.* Springer series in computational mathematics. Springer-Verlag, 1991.

[14] Ph. Clément. Approximation by finite element functions using local regularization. *ESAIM: Mathematical Modelling and Numerical Analysis - Modélisation Mathématique et Analyse Numérique*, 9(R2):77–84, 1975.

[15] C. R. Dohrmann and P. B. Bochev. A stabilized finite element method for the Stokes problem based on polynomial pressure projections. *Internat. J. Numer. Methods Fluids*, 46(2):183–201, 2004.

[16] W. Dörfler. A convergent adaptive algorithm for poisson's equation. *SIAM J. Numer. Anal.*, 33(3):1106–1124, June 1996.

[17] M. Emans. Benchmarking aggregation AMG for linear systems in CFD simulations of compressible internal flows. *Electron. Trans. Numer. Anal.*, 37:351–366, 2010.

[18] M. Emans. Efficient parallel AMG methods for approximate solutions of linear systems in CFD applications. *SIAM J. Sci. Comput.*, 32(4):2235–2254, 2010.

[19] J. H. Ferziger and M. Perić. *Computational methods for fluid dynamics.* Springer-Verlag, Berlin, revised edition, 1999.

[20] G. P. Galdi, R. Rannacher, A. M. Robertson, and S. Turek. *Hemodynamical flows*, volume 37 of *Oberwolfach Seminars*. Birkhäuser Verlag, Basel, 2008. Modeling, analysis and simulation, Lectures from the seminar held in Oberwolfach, November 20–26, 2005.

[21] V. Girault and P.-A. Raviart. *Finite element methods for Navier-Stokes equations*, volume 5 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, 1986. Theory and algorithms.

[22] A. Hannukainen, R. Stenberg, and M. Vohralík. A unified framework for a posteriori error estimation for the Stokes problem. *Numer. Math.*, 122(4):725–769, 2012.

[23] H. Jasak. *Error analysis and estimation in the Finite Volume method with applications to fluid flows.* PhD thesis, Imperial College, University of London, 1996.

[24] M. Křížek and P. Neittaanmäki. On superconvergence techniques. *Acta Appl. Math.*, 9(3):175–198, 1987.

[25] V. I. Krylov. *Approximate calculation of integrals.* Translated by Arthur H. Stroud. The Macmillan Co., New York, 1962.

[26] A. Kuzmin, M. Luisier, and O. Schenk. Fast methods for computing selected elements of the greens function in massively parallel nanoelectronic device simulations. In F. Wolf, B. Mohr, and D. Mey, editors, *Euro-Par 2013 Parallel Processing*, volume 8097 of *Lecture Notes in Computer Science*, pages 533–544. Springer Berlin Heidelberg, 2013.

[27] K. Morgan. The finite element method for elliptic problems. *International Journal for Numerical Methods in Engineering*, 14(5):786–786, 1979.

[28] S. Muzaferija. *Adaptive Finite Volume Method for Flow Prediction Using Unstructured Meshes and Multigrid Approach.* British Thesis Service. University of London, 1994.

[29] S. Muzaferija and D. Gosman. Finite-volume CFD procedure and adaptive error control strategy for grids of arbitrary topology. *J. Comput. Phys.*, 138(2):766–787, 1997.

[30] S.V. Patankar. *Numerical Heat Transfer and Fluid Flow.* Series in computational and physical processes in mechanics and thermal sciences. Hemisphere Publishing Company, 1980.

[31] M. Peric. *A finite volume method for the prediction of three-dimensional fluid flow in complex ducts.* PhD thesis, University of London, 1985.

[32] J. Radon. Zur mechanischen Kubatur. *Monatsh. Math.*, 52:286–300, 1948.

[33] O. Reynolds. Papers on mechanical and physical subjects. *International Journal of Heat and Mass Transfer*, 12(2):129 – 136, 1969.

[34] C. M. Rhie and W. L. Chow. A numerical study of the turbulent flow past an airfoil with trailing edge separation, 1982.

[35] O. Schenk, M. Bollhöfer, and R. A. Römer. On large-scale diagonalization techniques for the anderson model of localization. *SIAM Rev.*, 50(1):91–112, February 2008.

[36] O. Schenk, A. Wächter, and M. Hagemann. Matching-based preprocessing algorithms to the solution of saddle-point problems in large-scale nonconvex interior-point optimization. *Computational Optimization and Applications*, 36(2-3):321–341, 2007.

[37] J. Schöberl, J. Gerstmayr, and R. Gaisbauer. NETGEN - automatic 3d tetrahedral mesh generator. http://www.hpfem.jku.at/netgen/, May 2003.

[38] O. Steinbach. *Numerische Näherungsverfahren für elliptische Randwertprobleme. Finite Elemente und Randelemente.* Teubner, Stuttgart-Leipzig-Wiesbaden, 2003.

[39] R. L. Taylor and O. C. Zienkiewicz. *Finite Element Method: Volume 1, Fifth Edition (Finite Element Method Ser).* Butterworth-Heinemann, 5 edition, September 2000.

[40] R. Temam. *Navier-Stokes equations. Theory and numerical analysis.* North-Holland Publishing Co., Amsterdam, 1977. Studies in Mathematics and its Applications, Vol. 2.

[41] S. Turek. *Efficient solvers for incompressible flow problems*, volume 6 of *Lecture Notes in Computational Science and Engineering.* Springer-Verlag, Berlin, 1999.

[42] R. Verfürth. A posteriori error estimators for the Stokes equations. *Numer. Math.*, 55(3):309–325, 1989.

[43] R. Verfürth. *A Review of A posteriori error estimation and adaptive mesh-refinement techniques.* Wiley Teubner, 1996.

[44] R. Verfürth. Computational fluid dynamics. University Lecture, 2012.

[45] Z. J. Wang. Improved formulation for geometric properties of arbitrary polyhedra. *AIAA J.*, 37(10):1326–1327, 1999.

[46] O. C. Zienkiewicz and J. Z. Zhu. A simple error estimator and adaptive procedure for practical engineering analysis. *Internat. J. Numer. Methods Engrg.*, 24(2):337–357, 1987.

[47] O. C. Zienkiewicz and J. Z. Zhu. Superconvergence recovery technique and a posteriori error estimators. *Internat. J. Numer. Methods Engrg.*, 30(7):1321–1339, 1990.

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Graz, am ....

(Unterschrift)