INAUGURAL–DISSERTATION

zur
Erlangung der Doktorwürde
der
Naturwissenschaftlich-Mathematischen Gesamtfakultät
der
Ruprecht-Karls-Universität Heidelberg

vorgelegt von
Diplom-Physiker Christian Engwer
aus Bad Kreuznach

Tag der müdlichen Prüfung: 30. Oktober 2009

An Unfitted Discontinuous Galerkin Scheme for Micro-scale Simulations and Numerical Upscaling

Christian Engwer

June 5, 2009

Gutachter: Prof. Dr. Peter Bastian
Prof. Dr. Rolf Rannacher

Acknowledgments

I'd like to take the chance to express my gratitude to all the people who supported and helped me while I was working on this thesis.

First of all my gratitude goes to Prof. Peter Bastian (Universität Heidelberg), for giving me the opportunity to work on this interesting subject and for the possibility to contribute actively to the DUNE project. He helped me with many fruitful discussions, suggestions and valuable comments concerning this work and he always took the time to sit down and discuss, even if he actually had no time at all.

I'd like to thank my colleagues for a great time and good atmosphere. Especially to Markus, Olaf and Sven, with whom I could discuss endlessly over scientific problems and world politics.

This work would not have been possible without the software library DUNE thus my thank goes to all my fellow developers, particularly Oliver, who became a good friend.

Moreover, I my gratitude goes to my proof readers, especially to Sabine, Vera, Dirk and Mariya, who read everything again and again; Dirk for offering me a desk in Heidelberg during my exile at University of Stuttgart, for countless discussions and many coffees; to my housemates and my friends for being the counterbalance to my daily work.

And last but not least I'd like to thank Vera for her patience and her support during the years of my work.

Abstract

The aim of this thesis is the development of a new discretization method for solving partial differential equations on complex shaped domains. Many biological, physical, and chemical applications involve processes on such domains and the numerical treatment of such processes is a challenging task.

The proposed method offers a higher-order discretization where the mesh is not required to resolve the complex shaped boundary. The method combines the Unfitted Finite Element method with a Discontinuous Galerkin discretization. Trial and test functions are defined on a structured grid and their support is restricted according to the domain boundary. Essential boundary conditions are imposed weakly via the Discontinuous Galerkin formulation. Thus, the mesh is not required to resolve the domain boundary but higher-order ansatz functions can still be used. Hence it is possible to vary the size of the ansatz space independently of the geometry.

For an elliptic test problem, stability and convergence properties of the method are analyzed numerically. Even though some assumptions of the underlying Discontinuous Galerkin method regarding the finite element mesh cannot be guaranteed, the method is stable in all tests and converges optimally.

The control over the size of the approximation space is especially attractive for applications like numerical upscaling and multi-scale simulations. In this thesis the method is successfully applied to numerical upscaling of a stationary flow problem and to a time-dependent transport problem, where the complex domains used are artificially generated as well as experimentally measured structures, obtained from micro X-ray CT scans.

Zusammenfassung

Gegenstand dieser Arbeit ist die Entwicklung eines Diskretisierungsverfahrens zur Lösung partieller Differentialgleichungen auf Gebieten mit komplizierten Rändern. In vielen biologischen, physikalischen und chemischen Anwendungen treten Prozesse auf solchen Gebieten auf, deren numerische Behandlung eine Herausforderung darstellt.

Das entwickelte Verfahren erlaubt eine Diskretisierung mit Ansatzfunktionen höherer Ordnung, wobei das Gitter den komplexen Rand nicht auflösen muss. Das Verfahren stellt eine Kombination aus der „Unfitted Finite Element"-Methode und einem Discontinuous-Galerkin-Verfahren dar. Ansatz- und Testfunktionen sind auf einem strukturierten Gitter definiert und ihr Träger wird entsprechend des Gebietsrandes eingeschränkt. Essentielle Randbedingungen werden durch die Discontinuous-Galerkin-Formulierung schwach erzwungen. Dies führt dazu, dass das Gitter den Gebietsrand nicht auflösen muss, aber trotzdem Ansatzfunktionen höherer Ordnung verwendet werden können. Dadurch ist es möglich die Größe des Ansatzraumes unabhängig von der geometrischen Struktur zu variieren.

Anhand eines elliptischen Testproblems werden Stabilitäts- und Konvergenzeigenschaften des Verfahrens numerisch untersucht. Auch wenn verschiedene Voraussetzungen des zugrunde liegenden Discontinous-Galerkin-Verfahrens bezüglich des Finite-Elemente-Gitters nicht garantiert werden können, zeigt sich das Verfahren in allen Tests stabil und konvergiert optimal.

Die Eigenschaft, die Größe des Ansatzraumes frei wählen zu können, macht das Verfahren besonders attraktiv für Anwendungen im Bereich des numerischen Upscalings oder für Mehrskalenverfahren. In dieser Arbeit wird die Methode erfolgreich zum numerischen Upscaling eines stationären Flußproblems sowie zur Lösung eines zeitabhängigen Transportproblems angewandt, teils auf künstlich generierte, teils auf experimentell gemessene Strukturen, welche aus Mikro-CT-Aufnahmen gewonnen sind.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

There exists today a universal language that is spoken and understood almost everywhere: it is Broken English. I am not referring to Pidgin English [...], but to the much more general language that is used by waiters in Hawaii, prostitutes in Paris and ambassadors in Washington, by businessmen from Buenos Aires, by scientists at international meetings and by dirty-postcard peddlers in Greece, in short by honourable people like myself all over the world.

— *Hendrik Casimir*

1.1 Motivation

For nearly 400 years science has been based on the fruitful symbiosis of theory and experiment. Experiments are carried out to verify a theory or to observe new facts and thus to spark ideas for new or better theories.

As the focus of science moves to smaller and smaller systems and the complexity of the processes increases, these processes elude experiments.

Numerical simulations can give insights into these difficult processes. They have been established as a third fundamental way of gaining knowledge besides theory and experiment. Especially in biology, the term 'in silico experiment' is used synonymously to computer simulations [81].

Processes in natural sciences are often observed on different scales. On the small scale processes can be described by fundamental physical laws, but the scientific interest often lies in large scale results. These large scale processes are described by empiric laws, which require the knowledge of parameters, usually obtained by experiments. As experiments are expensive or sometimes not feasible at all, numerical simulations become more and more popular to support parameter estimation.

Such kind of applications can be found in many subjects, e. g. in soil science or cell biology. Examples are solute transport on the pore scale of porous media or reaction diffusion processes on cell membranes. In these contexts parameter estimation using

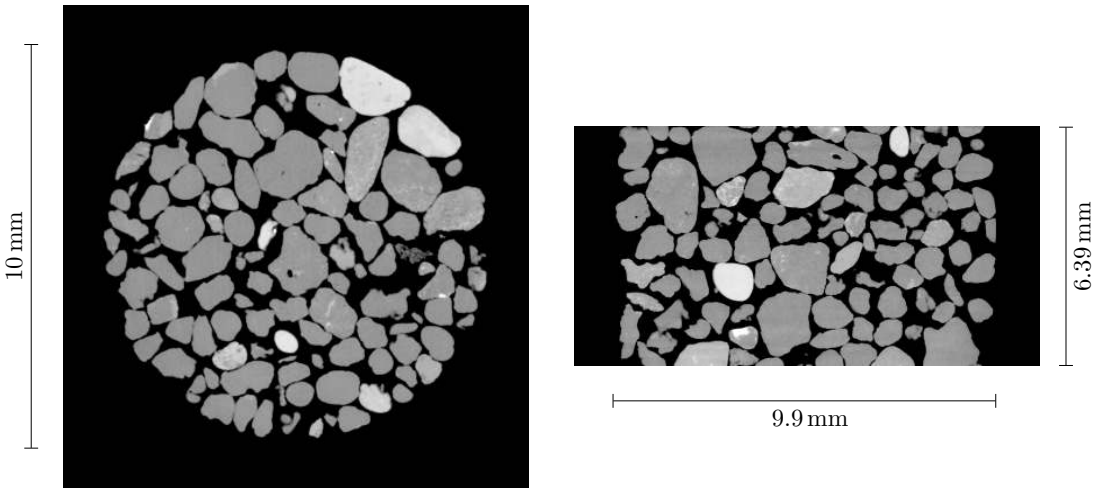
numerical upscaling can be applied. Macroscopic parameters are obtained using small scale simulations and some form of averaging.

Micro-scale simulations, as they are needed during the upscaling process, require a good approximation of the geometrical shape of the microscopic domain to obtain reliable numerical results. Although the numerical software and the methods improve constantly, high resolution simulation of small scale structures are still a challenging task.

Often the geometrical shape is not even stationary. Evolving domains and sometimes even topological changes increase the complexity further.

Such kind of problems arise in many different applications. To give an idea we briefly sketch three different examples from different subjects.

1.1.1 Soil Sciences – Pore-scale processes



(a) xy–slice (b) yz–slice

Figure 1.1: Pore structure of a coarse sand, obtained from micro X-ray computer tomography (H.-J. Vogel, UFZ Halle).

Large scale numerical simulations of fluid flow and transport in porous media are an important tool in geosciences and industry, first and foremost petroleum engineering [22]. These simulations at the continuum scale require the knowledge of effective parameters. Hydraulic parameters (e. g. capillary pressure/saturation curve, relative permeability function) for the macroscopic models are often hard to measure [22, 102].

Nowadays detailed measurements of the pore-scale structure are possible [105, 80], e. g. X-ray tomography (Figure 1.1). As the governing equations on the micro-scale are well known, macroscopic parameters can be obtained directly from the pore-scale geometry by numerical upscaling.

Handling the pore structure in a numerical simulation is not easy. The solid phase forms a complex shaped geometry and a good approximation of this geometrical shape is crucial in order to obtain reliable numerical results. Corners in the approximation of the geometry lead to singularities in the solution. Depending on the equation these will disturb the solution as a whole.

Obtaining a good approximation of the resulting domain usually requires simulations on a very fine grid, while the interest on the other hand lies only in macroscopic parameters, which could be calculated with sufficient precision from a solution on a much coarser mesh.

1.1.2 Bio Mechanics – Elasticity of bones

Mechanical behavior of and load on bones is of interest for medical topics [39] like healing processes, stability of bone implants and osteoporosis.

The mechanical properties of bones are largely determined by their micro-structure [96]. This micro-structure forms a complex three-dimensional network, called spongy or cancellous bone. The local structure differs largely between different bones and even different areas of the same bone [64] as shown in Figure 1.2. These differences in structure strongly influence the isotropy or anisotropy of the elastic modulus, as well as the stiffness and the stability of the bone.

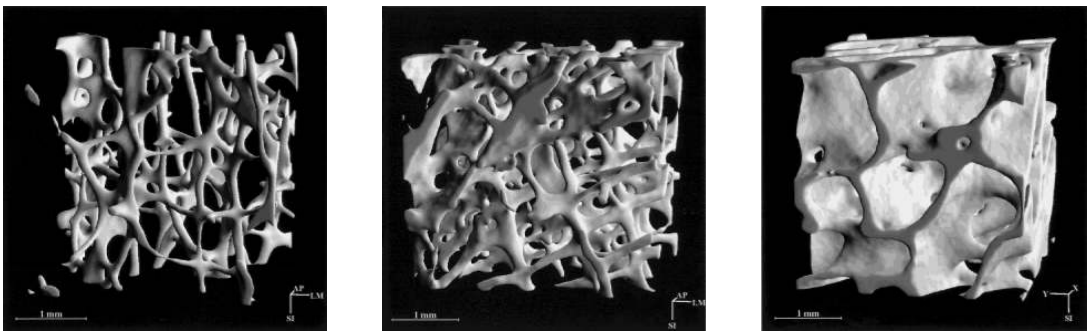


Figure 1.2: Mirco-CT images of cancellous bone of different human bones [64].

Simulation of elasticity on the microscopic network structure can give parameters for the computation of strain and stress in whole bones.

In living bone tissue the pore structure is fluid-saturated. The so-called poroelasticity additionally considers the interaction of the fluid with the elastic structure [39]. Simulations involve the computation of elasticity on the structure coupled with incompressible flow in the pores.

1.1.3 Cell Biology – Endoplasmic Reticulum

The endoplasmic reticulum [2] is part of the cell surrounding the cell nucleus. It forms a complex network of tubules, vesicles, and cisternae, enclosed by a highly folded

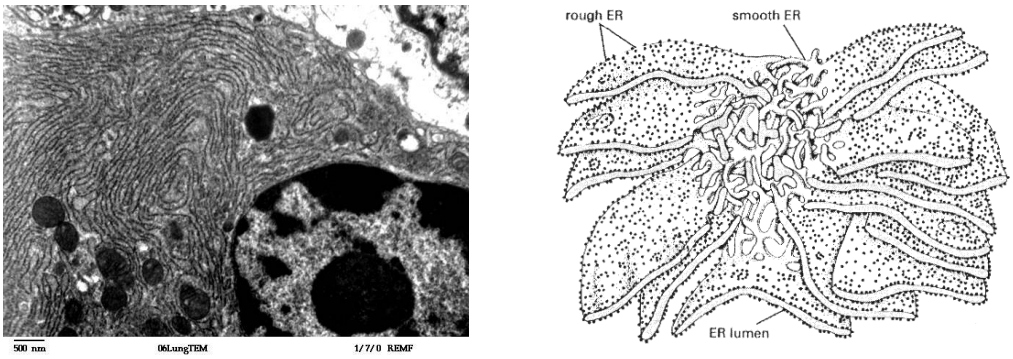


Figure 1.3: The endoplasmic reticulum. Left: microscopy image of nucleus and endoplasmic reticulum (Rippel electron microscope facility, Dartmouth College). Right: sketch of endoplasmic reticulum [62].

membrane. This structure plays an essential role in many cell processes, e. g. during mitosis and DNA replication.

The membrane does change in time. It's temporal behavior does influence the chemical and biological processes. Obversely reaction–diffusion processes on the membrane and in the endoplasmic reticulum are important for its functional behavior [92].

1.2 Objective

We presented different applications where it is necessary to handle complex shaped, possibly time-dependent domains. The goal is to find a numerical method that helps coping with that kind of problems. Applications related to the first example, originating from soil science, must act as a test case for this method. The second and third example will not be treated but should be considered when designing the new method. Extracting the common ground of all these processes we can state the objective as follows.

A complicated geometry gives a partition into sub-domains. On one or more of these sub-domains a partial differential equation (PDE) must be solved. On the interface between these sub-domains boundary conditions or transmission conditions apply. In some applications the partition might even be a manifold. It must be considered that the sub-domains can change in time.

Computation power is a precious source and especially in three-dimensional setups the computational costs quickly rise to a point where even super-computers cannot help anymore. Therefore it is desired to have a discretization that can accurately represent the complicated geometry while keeping the size of the approximation space as low as possible. In particular, this requires the ability to carry out simulations on a relatively coarse mesh.

Classical numerical methods require a grid resolving the geometry. Creating such

grids is a very sophisticated process, especially when it comes to time-dependent geometries. Therefore methods without this requirement are of high interest. Obtaining a good approximation of the resulting domain usually requires simulations on a very fine grid, in order to resolve the complex shape. This is undesired, especially in cases where focuses is on macroscopic parameters, which could be calculated with a sufficient precision from a solution on a coarser mesh. Evolving domains increase the difficult further. Moving mesh methods, as they are often used [42, 33], cannot guarantee shape regularity of the grid throughout time. Remeshing is required. Topological changes cannot be handled at all.

This thesis presents a new approach to the solution of partial differential equations on complex shaped domains. It combines the idea of Unfitted Finite Element [10] with a Discontinuous Galerkin (DG) Finite Element discretization. This *Unfitted Discontinuous Galerkin* method allows the discretization of processes on domains of complicated shape without the need of a geometry resolving grid. It successfully uncouples the construction of the finite element mesh from the geometrical properties. Although it is not used in this work, a combination of this method with adaptive mesh refinement and error estimators would allow to refine the grid where accuracy of the discretization is not sufficient and not where the geometry is of small scale. In contrast to other methods, this method does not require a modification of the formulated problem in order to incorporate the geometrical constraints. The Unfitted Discontinuous Galerkin method incorporates DG methods in their primal formulation.

1.3 Outline

This thesis consists of eight chapters. After this introduction, in Chapter 2, we introduce the concepts of the Discontinuous Galerkin Finite Element method. This method forms the basis of the presented work. We will give a short historical overview and describe the central concepts of the method. Then we discuss different properties of the method and their attraction for simulations in natural sciences. We put emphasis on those aspects that are important for our later work.

Chapter 3 is central to this work. It describes the concepts of the Unfitted Discontinuous Galerkin method. It is a new approach to the solution of partial differential equations on complex shaped domains. We will discuss both mathematical and practical aspects of the method. We compare the new approach to the standard Conforming Finite Element approach and to other methods targeting similar goals.

After discussing implementation issues and the integration into the DUNE framework [19, 20] in Chapter 4, Chapter 5 will get back to the different DG properties discussed in Chapter 2. Simulations will show numerically that the changes introduced in the Unfitted Discontinuous Galerkin approach do not harm the stability, convergence and other properties of the underlying Discontinuous Galerkin method.

In the following two chapters numerical simulations of static and time-dependent problems are presented. These emphasize the flexibility of the new method.

A summary and discussion follows in Chapter 8. An appendix provides a list of abbreviations, notations and additional data required for the implementation of the algorithms described in Chapter 3.

Chapter 2

Discontinuous Galerkin Finite Element Method

Diviser pour régner.
— Louis XI, roi de France.

Discontinuous Galerkin (DG) Finite Element Methods name a class of Finite Element Methods (FEM), which are of special attraction for the simulation of physical problems. Many of them preserve continuous conservation laws in the discrete formulation, e. g. local mass conservation in fluid transport problems. Discontinuous Galerkin methods date back to the early 1970s, although they weren't called like this initially. Their name derives from the fact that trial and test functions are discontinuous across element boundaries.

Discontinuous Galerkin methods show a lot of similarities to Finite Volume (FV) [70] methods in their construction as well as in the discrete representation of conservation laws using a flux formulation. In fact, the simplest DG discretization for hyperbolic problems using piecewise constant basis functions is equivalent to a cell-centered FV discretization. Unlike DG methods, FV methods generally do not exhibit Galerkin orthogonality. Sometimes the FV schemes are also referred to as Integrated Finite Differences.

2.1 Overview

In 1973 Reed and Hill [87] published a method for the solution of hyperbolic equations. It marks the beginning of the development of a wide range of Discontinuous Galerkin Methods for hyperbolic problems.

In the same decade several authors independently proposed DG methods for elliptic and parabolic equations and different variants of the method were developed. These methods were usually called *Interior Penalty (IP) Methods*; see e. g. Douglas Jr and Dupont [46], Baker [9], Wheeler [109], Arnold [3, 4]. They adopt a method of enforcing Dirichlet boundary conditions through penalties, proposed 1971 by Nitsche [78], and

apply these concepts to inter-element boundaries. More recently the development of Discontinuous Galerkin methods for elliptic equations increased again with publications by Bassi and Rebay [12], Oden et al. [79] and Cockburn and Shu [37] (1997 and 1998).

With the work of Hansbo and Hansbo [59], the concepts known from Discontinuous Galerkin Methods started to find application in domain decomposition for non-matching grids.

Finally, in 2002, a unified analysis of Discontinuous Galerkin Methods for elliptic equations was published in [5]. This paper allows to describe and analyze the different DG methods using a unified framework. This enabled recent general work on DG, e. g. general a-posteriori error estimates [73].

2.2 Elliptic Model Problem

In this work we follow the nomenclature introduced in [5].

Let $\Omega \subseteq \mathbb{R}^d$ be a domain of size

$$L = \operatorname{diam}(\Omega) \tag{2.1}$$

Definition 2.1 (Model Problem/Strong formulation): *We consider the elliptic model problem in its* strong formulation

$$-\Delta u = f \qquad \text{in} \quad \Omega, \tag{2.2a}$$
$$u = 0 \qquad \text{on} \quad \partial\Omega, \tag{2.2b}$$

with the unknown $u \in C^2(\Omega) \cap C^0(\bar{\Omega})$ and f being a source term.

Definition 2.2 (Triangulation $\mathcal{T}$): *Given a non-overlapping partition*

$$\mathcal{T} = \{E_0, \ldots, E_{M-1}\} \qquad \text{with} \tag{2.3a}$$
$$E_n \subseteq \Omega \qquad \forall \quad 0 \le n < M, \qquad E_n \text{ open}, \tag{2.3b}$$
$$E_n \cap E_m = \emptyset \qquad \forall \quad 0 \le n < m < M, \tag{2.3c}$$
$$\bar{\Omega} = \bigcup_{n=0}^{M-1} \bar{E}_n, \tag{2.3d}$$

and a finite set of references elements $\{\hat{E}\}$. Assuming further that for each E_n there exists a smooth one-to-one mapping T_{E_n} from a reference element $\hat{E}$

$$E_n = T_{E_n}(\hat{E}), \tag{2.4}$$

we call $\mathcal{T}$ a triangulation *of Ω with mesh size*

$$h = \max\{\operatorname{diam}(E) \mid E \in \mathcal{T}\}\,. \tag{2.5}$$

Definition 2.3 (Internal Skeleton): *The* internal skeleton Γ_{int} *of the partitioning is given by*

$$\Gamma_{int} = \{\gamma_{e,f} = \partial E_e \cap \partial E_f \mid E_e, E_f \in \mathcal{T} \text{ and } E_e \neq E_f \text{ and } |\gamma_{e,f}| > 0\}, \tag{2.6}$$

where $|\gamma|$ *denotes the codimension one volume of* γ.

Definition 2.4 (External Skeleton): *Correspondingly, the external skeleton is denoted by*

$$\Gamma_{ext} = \{\gamma_e = \partial E_e \cap \partial\Omega \mid E_e \in \mathcal{T} \text{ and } |\gamma_e| > 0\}. \tag{2.7}$$

Definition 2.5 (Skeleton): *The* skeleton Γ *of the partitioning is defined as the union*

$$\Gamma = \Gamma_{int} \cup \Gamma_{ext}. \tag{2.8}$$

2.2.1 Enforcing Dirichlet Boundary Conditions through a Penalty Term

In 1971 Nitsche [78] proposed a new method of enforcing Dirichlet boundary conditions for elliptic partial differential equations through penalties. This method forms the foundation for the later development of Discontinuous Galerkin methods. Therefore we want to outline this approach.

Nitsches approach is inspired by [6]. Babuška replaced (2.2b) by an approximate boundary condition

$$u + \frac{1}{\mu}\nabla u \cdot \mathbf{n} = 0 \qquad \text{with} \quad \mu > 0, \tag{2.9}$$

where $\mathbf{n}$ is the outward normal unit vector of $\partial\Omega$. The resulting weak formulation is not consistent with the strong problem, i. e. solutions of the strong problem are not a solution of the weak formulation. A proper definition of consistency follows later.

Variational view

To obtain a consistent and symmetric method Nitsche proposes the following variational formulation:

Find $u_h \in V_h \subset H^1(\Omega)$ such that

$$a(u_h, v) = l(v) \qquad \forall v \in V_h \subset H^1. \tag{2.10a}$$

with the bilinear form a and the linear form l given as

$$a(u_h, v) = \int_\Omega \nabla u_h \cdot \nabla v \, dx - \int_{\partial\Omega} \nabla u_h \cdot \mathbf{n}\, v \, ds - \int_{\partial\Omega} \nabla v \cdot \mathbf{n}\, u_h \, ds + \mu \int_{\partial\Omega} u_h v \, ds \tag{2.10b}$$

and

$$l(v) = \int_\Omega f\, v \, dx. \tag{2.10c}$$

This formulation can be motivated as follows. Equation (2.2a) is tested with $v \in H^1$ and (2.2b) with $v - \frac{1}{\mu}\nabla v \cdot \mathbf{n}$. Combining these equations and scaling the second with μ yields (2.10a).

The term $\int_{\partial\Omega} v\nabla u \cdot \mathbf{n}\, ds$ arises from integration by parts of the left hand side of (2.2a) and thus it is important to ensure consistency of the problem. The weak form of (2.2b) leads to two additional terms in bilinear form

$$S = -\int_{\partial\Omega} u_h \nabla v \cdot \mathbf{n}\, ds \tag{2.11}$$

and

$$J = \mu \int_{\partial\Omega} u_h v\, ds. \tag{2.12}$$

Term (2.11) makes the problem symmetric, where as (2.12) is necessary to guarantee coercivity of the problem.

Nitsche proved that for $\mu = \frac{\eta}{h}$, $\eta > 0$ and sufficiently large, u_h converges optimally in the H^1- and L_2-norm.

Minimization view

The Nitsche Method can also be seen in a minimization context. Equation (2.10a) then reads: Find $u \in V$ such that

$$\begin{aligned} J(u) &= \min(F(v) : v \in V) \qquad \text{with} \\ J(v) &= \frac{1}{2}\, a(v,v) - l(v). \end{aligned} \tag{2.13}$$

The method is stable if $a(v,v)$ is a symmetric and coercive functional. Formally written this stability condition reads as

$$a(v,v) \geq c_s \|v\|^2_{1,\Omega} \qquad \forall v \in V_k\,, \tag{2.14}$$

with a constant $c_s > 0$ [57].

Theorem 2.1: *If a discrete function space fulfills the property*

$$|v|_{1,\partial\Omega} \leq c h^{-\frac{1}{2}} |v|_{1,\Omega} \tag{2.15}$$

with a constant $c > 0$ the bilinear form (2.10b) is stable for $\mu \geq 2\frac{c^2}{h}$.

Proof: Follows directly from the estimate (2.15) and the following inequalities:

- the Cauchy-Schwarz inequality ($|(a,b)| \leq \|a\| \cdot \|b\|$)

$$\left| \int_{\partial\Omega} v\, \nabla v \cdot \mathbf{n}\, ds \right| \leq \|v\|_{0,\partial\Omega} \cdot \|\nabla v\|_{0,\partial\Omega} \quad = \|v\|_{0,\partial\Omega} \cdot |v|_{1,\partial\Omega} \tag{2.16}$$

- and Young's inequality ($2ab \leq \epsilon a^2 + b^2/\epsilon$ with $\epsilon > 0$)

$$\begin{aligned} 2\,\|v\|_{0,\partial\Omega} \cdot |v|_{1,\partial\Omega} &\leq 2\,\|v\|_{0,\partial\Omega} c h^{-\frac{1}{2}} \cdot |v|_{1,\Omega} \\ &\leq 2\,\|v\|^2_{0,\partial\Omega} c^2 h^{-1} + \frac{1}{2}\,|v|^2_{1,\Omega} \end{aligned} \tag{2.17}$$

Using Equations (2.15), (2.16) and (2.17) the bilinear form $a(v, v)$ can be estimated from below.

$$\begin{aligned} a(v,v) &= \int_\Omega (\nabla v)^2\,dx - 2\int_{\partial\Omega} \nabla v \cdot \mathbf{n}\, v\,ds + \mu \int_{\partial\Omega} v^2\,ds \\ &\underset{(2.16)}{\geq} |v|^2_{1,\Omega} - 2\,\|v\|_{0,\partial\Omega} \cdot |v|_{1,\partial\Omega} + \mu \|v\|^2_{0,\partial\Omega} \\ &\underset{(2.17)}{\geq} \frac{1}{2}\,|v|^2_{1,\Omega} + (\mu - 2c^2 h^{-1}) \|v\|^2_{0,\partial\Omega} \end{aligned} \tag{2.18}$$

□

Remark 2.1 (Interpretation of 2.15): The required estimate 2.15 can be obtained using the Poincaré-Friedrich inequality, an inverse estimate and the trace theorem. For a norm $\|v\|_{1,\partial\Omega}$, an inverse estimate gives an estimate with lower order and the trace theorem allows to estimate using a norm on Ω.

$$\|v\|_{1,\partial\Omega} \underset{\text{(inv. e.)}}{\leq} c_1 h^{-\frac{1}{2}} \|v\|_{\frac{1}{2},\partial\Omega} \underset{\text{(tr.th.)}}{\leq} c_2 h^{-\frac{1}{2}} \|v\|_{1,\Omega}\,. \tag{2.19}$$

The Poincaré-Friedrich inequality gives estimates using semi norms

$$|v|_{1,\partial\Omega} \leq c h^{-\frac{1}{2}} |v|_{1,\Omega}\,. \tag{2.20}$$

Remark 2.2: In the minimization problem the term originating from (2.12) guarantees the coercivity of the problem. It penalizes solutions with large u_h at the boundary and thus leads to a solution with $u = 0$ in the continuous limit. This explains the name *penalty term* for (2.12).

Remark 2.3: Note that Theorem 2.1 gives the same relation $\mu \propto \frac{1}{h}$ that was required for the error estimates in the variational formulation, choosing the constant $\eta \geq 2c^2$.

Equivalence of Strong and Weak Formulation

Definition 2.6 (Consistency): *Given the solution $u \in C^2$ of the model problem (2.2), a weak formulation with the bilinear form $a(u, v)$ is* consistent *if u fulfills*

$$a(u,v) = \int_\Omega f\,v\,dx \qquad \forall v \in V_h. \tag{2.21}$$

Theorem 2.2: *The weak formulation (2.10a), as proposed by Nitsche, is consistent with the strong formulation of the model problem (2.2).*

Proof: Given a solution $u \in C^2(\Omega)$ of the model problem (2.2). As $u = 0$ on $\partial\Omega$ the weak formulation (2.10a) reduces to

$$\int_\Omega \nabla u \cdot \nabla v \, dx - \int_{\partial\Omega} \nabla u \cdot \mathbf{n} \, v \, ds = \int_\Omega f \, v \, dx. \tag{2.22}$$

Reverting the integration by parts gives

$$\int_\Omega -\Delta u v \, dx = \int_\Omega f \, v \, dx. \tag{2.23}$$

As $-\Delta u = f$ Equation (2.23) is true and u is also a solution of the weak formulation. □

Theorem 2.3: *Let u be a unique solution u of the weak formulation (2.10a). Assuming that u has sufficient regularity, i. e. $u \in H^1(\Omega) \cap C^2(\Omega)$, u is also a solution of (2.2).*

Proof: As u fulfills (2.10a) for any $v \in H^1$, it is also true for any $v \in C_0^\infty$. Choosing v from C_0^∞ all boundary integrals vanish and we obtain

$$\int_\Omega \nabla u \cdot \nabla v \, dx = \int_\Omega f \, v \, dx \qquad \forall v \in C_0^\infty(\Omega). \tag{2.24}$$

As u and f are continuous on Ω the *Fundamental Lemma of Calculus of Variations* yields

$$-\Delta u = f. \tag{2.25}$$

Thus (2.2a) is fulfilled. Reverting integrations by part in (2.10a) and using $-\Delta u = f$ the weak formulation simplifies to

$$-\int_{\partial\Omega} u \nabla v \cdot \mathbf{n} \, ds + \mu \int_{\partial\Omega} u \, v \, ds = 0 \qquad \forall v \in H^1(\Omega). \tag{2.26}$$

As 2.26 holds for any $v \in H^1$ it also holds for any $v \in H_0^2$ and we obtain

$$-\int_{\partial\Omega} u \nabla v \cdot \mathbf{n} \, ds = 0 \qquad \forall v \in H_0^2(\Omega). \tag{2.27}$$

In [1], Theorem 7.53, it is stated that the normal trace maps surjectively from $H_0^2(\Omega)$ to $H^{\frac{1}{2}}(\partial\Omega)$ so that (2.27) becomes

$$-\int_{\partial\Omega} u \, \tilde{v} \, ds = 0 \qquad \forall \tilde{v} \in H^{\frac{1}{2}}(\partial\Omega). \tag{2.28}$$

Using again the Fundamental Lemma of Calculus of Variations we obtain $u = 0$ on $\partial\Omega$ and (2.2b) is also fulfilled. □

2.2.2 A Unified Framework

In [5] Arnold et al. introduced a unified framework to handle all different DG methods. We start again with the model problem in its strong formulation (2.2), which is a second-order partial differential equation.

Definition 2.7 (Model Problem/Mixed Formulation): *We rewrite the problem (2.2) from its higher-order partial differential equation representation to a system of first-order PDEs with the unknowns u and σ. This first-order system is also called the* mixed formulation*:*

$$\sigma = \nabla u \qquad in \quad \Omega, \tag{2.29a}$$
$$-\nabla \cdot \sigma = f \qquad in \quad \Omega, \tag{2.29b}$$
$$u = 0 \qquad on \quad \partial\Omega. \tag{2.29c}$$

On the triangulation $\mathcal{T}$ (Definition 2.2) we define two function spaces of discontinuous polynomial functions

$$V_h = \left\{ v \in L^2(\Omega) \,:\, v|_E \in P_k(E) \right\} \qquad \text{and} \tag{2.30a}$$
$$\Sigma_h = \left\{ \tau \in [L^2(\Omega)]^d \,:\, \tau|_E \in [P_k(E)]^d \right\}, \tag{2.30b}$$

where P_k denotes the space of polynomial functions of degree k.

Definition 2.8: *The space of polynomial functions of degree k is given as*

$$P_k = \left\{ \psi : \mathbb{R}^d \to \mathbb{R} \;\middle|\; \psi(x) = \sum_{|\alpha| \leq k} c_\alpha x^\alpha \right\} \tag{2.31}$$

where α denotes a multi-index.

Remark 2.4: According to [45] the local basis functions can be chosen independently of the shape of the element and the shape of the elements E_n can be quite arbitrary. In [45] Dolejsí et al. show that star shaped elements (see [85, p. 18]) are sufficient but not necessary for the convergence rate to be independent of the shape of the elements.

Definition 2.9 (Model Problem/Weak Formulation): *Continuing with the problem in its mixed formulation on a sub-domain $K \subseteq \Omega$, we multiply (2.29a) and (2.29b) with test functions $\tau \in \Sigma_h$ and $v \in V_h$ and apply integration by parts. We obtain the problem in its* weak formulation

$$\begin{aligned} \int_K \sigma \cdot \tau \, dx &= -\int_K u \nabla \cdot \tau \, dx + \int_{\partial K} u \, \mathbf{n} \cdot \tau \, ds, \\ \int_K \sigma \cdot \nabla v \, dx &= -\int_K f v \, dx + \int_{\partial K} \sigma \cdot \mathbf{n} \, v \, ds, \end{aligned} \tag{2.32}$$

where $\mathbf{n}$ *is the outward pointing normal unit vector of* ∂K.

We choose $K = E_n \in \mathcal{T}$, $\tau \in \Sigma_h$ and $v \in V_h$. On each element we introduce *numerical fluxes* $\widehat{\sigma}_{E_n}$ and $\widehat{u}_{E_n}$ forming approximations to σ and u on the boundary. The operator $\widehat{\cdot}_{E_n}$ maps

$$\begin{aligned} \widehat{u}_{E_n} &: \qquad H^1(E_n) \to L^2(\partial E_n) \qquad \text{and} \\ \widehat{\sigma}_{E_n} &: \quad H^2(E_n) \times [H^1(E_n)]^d \to [L_2(\partial E_n)]^d. \end{aligned} \tag{2.33}$$

and is constructed such that

$$\int_{\partial E_n} \widehat{u}_{E_n}\, ds = \int_{\partial E_n} u\, ds \qquad \text{and} \qquad \int_{\partial E_n} \widehat{\sigma}_{E_n}\, ds = \int_{\partial E_n} \nabla u\, ds.$$

Globally numerical fluxes $\widehat{u}$ and $\widehat{\sigma}$ are given as the combination of the local operators $\widehat{u} = (\widehat{u}_{E_n})_{E_n \in \mathcal{T}}$, $\widehat{\sigma} = (\widehat{\sigma}_{E_n})_{E_n \in \mathcal{T}}$. The global operator $\widehat{\cdot}$ maps

$$\begin{aligned} \widehat{u} &: \qquad H^1(\mathcal{T}) \to T(\Gamma) \qquad \text{and} \\ \widehat{\sigma} &: \quad H^2(\mathcal{T}) \times [H^1(\mathcal{T})]^d \to [T(\Gamma)]^d. \end{aligned} \tag{2.34}$$

with $T(\Gamma) = \Pi_{E_n \in \mathcal{T}} L_2(\partial E_n)$. Note that functions in $T(\Gamma)$ are double-valued on Γ_{int} and Γ_{ext}.

Definition 2.10 (Model Problem/Flux Formulation): *Together with the* numerical fluxes $\widehat{\sigma}_{E_n}$ *and* $\widehat{u}_{E_n}$ *the problem reads: Find* $u_h \in V_h$ *and* $\sigma_h \in \Sigma_h$ *such that*

$$\begin{aligned} \int_{E_n} \sigma_h \cdot \tau\, dx &= -\int_{E_n} u_h \nabla \cdot \tau\, dx + \int_{\partial E_n} \widehat{u}_{E_n} \mathbf{n}_{E_n} \cdot \tau\, ds, \\ \int_{E_n} \sigma_h \cdot \nabla v\, dx &= -\int_{E_n} f v\, dx + \int_{\partial E_n} \widehat{\sigma}_{E_n} \cdot \mathbf{n}_{E_n} \cdot v\, ds \end{aligned} \tag{2.35}$$

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

Depending on the choice of the numerical fluxes it is possible to construct all different kinds of Discontinuous Galerkin methods. This choice has a strong influence on the stability and accuracy of the scheme as well as on the sparcity and the structure of the resulting stiffness matrix.

Definition 2.11 (Jump): *Following Arnold et al. the* jump *of a piecewise continuous function* x *on the interface between two adjacent elements* E_n *and* E_m *is denoted by the linear operator*

$$[\![\, x \,]\!] = x|_{\partial E_n} \mathbf{n}_{E_n} + x|_{\partial E_m} \mathbf{n}_{E_m}\,. \tag{2.36}$$

On the boundary $\partial\Omega$ *we define* $[\![\, x \,]\!]$ *to be* $[\![\, x \,]\!] = x\mathbf{n}_{E_n}$. *Note that the jump* $[\![\, p \,]\!]$ *of a scalar function* p *is a vector parallel to its normal vector and that the jump* $[\![\, \mathbf{v} \,]\!]$ *of a vector valued function* $\mathbf{v}$ *is a scalar.*

Definition 2.12 (Average): *The* average *of a piecewise continuous function x on the interface is given by the arithmetic mean*

$$\{\, x \,\} = \frac{1}{2}(x|_{\partial E_n} + x|_{\partial E_j}) \,. \tag{2.37}$$

On $\partial\Omega$ the average is defined as $\{\, x \,\} = x$.

Remark 2.5: The jump and average operators fulfill the relation

$$[\![\, xy \,]\!] = [\![\, x \,]\!]\{\, y \,\} + [\![\, y \,]\!]\{\, x \,\} \,. \tag{2.38}$$

Using definition 2.11 and 2.12 and summing over all elements, (2.35) can be rewritten as

$$\begin{aligned} \int_\Omega \sigma_h \cdot \tau \, dx &= -\int_\Omega u_h \nabla \cdot \tau \, dx + \int_\Gamma [\![\, \widehat{u}_{E_n} \tau \,]\!] \, ds, \\ \int_\Omega \sigma_h \cdot \nabla v \, dx &= -\int_\Omega f v \, dx + \int_\Gamma [\![\, \widehat{\sigma}_{E_n} v \,]\!] \, ds \,. \end{aligned} \tag{2.39}$$

In order to express σ_h in terms of u_h, the operators $[\![\, u \,]\!]$ and $\{\, u \,\}$ are extended into the whole domain using suitable lifting operators.

Definition 2.13 (Lifting operators): Lifting operators r *and* l *are defined such that*

$$\int_\Omega r(\psi) \cdot \tau \, dx = -\int_\Gamma \psi \cdot \{\, \tau \,\} \, ds \qquad \text{and} \qquad \int_\Omega l(q) \cdot \tau \, dx = -\int_{\Gamma_{int}} q [\![\, \tau \,]\!] \, ds \,. \tag{2.40}$$

Using the lifting operators and applying integration by parts formula, σ_h can be written as

$$\sigma_h = \nabla u_h - r([\![\, \widehat{u} - u_h \,]\!]) - l(\{\, \widehat{u} - u_h \,\}) \,. \tag{2.41}$$

Substitution of σ_h leads to the problem in its primal formulation.

Definition 2.14 (Model Problem/Primal Formulation): *The discretization of the elliptic model problem (2.2) in the* primal formulation *reads:*

Find $u_h \in V_h$ such that

$$a(u_h, v) = \int_\Omega f v \, dx \qquad \forall v \in V_h \tag{2.42}$$

with the bilinear form

$$\begin{aligned} a(u_h, v) &= \int_\Omega \nabla u_h \cdot \nabla v \, dx + \int_\Gamma [\![\widehat{u} - u_h]\!] \cdot \{ \nabla v \} - [\![v]\!] \cdot \{ \widehat{\sigma} \} \, ds \\ &\quad + \int_{\Gamma_{int}} \{ \widehat{u} - u_h \} [\![\nabla v]\!] - \{ v \} [\![\widehat{\sigma}]\!] \, ds \\ &= \sum_{E_n \in \mathcal{T}} \int_{E_n} \nabla u_h \cdot \nabla v \, dx \\ &\quad + \sum_{\gamma_{e,f} \in \Gamma} \int_{\gamma_{e,f}} [\![\widehat{u} - u_h]\!] \cdot \{ \nabla v \} - [\![v]\!] \cdot \{ \widehat{\sigma} \} \, ds \\ &\quad + \sum_{\gamma_{e,f} \in \Gamma_{int}} \int_{\gamma_{e,f}} \{ \widehat{u} - u_h \} [\![\nabla v]\!] - \{ v \} [\![\widehat{\sigma}]\!] \, ds \, . \end{aligned} \tag{2.43}$$

2.2.3 Selected Schemes

Within this work we want to concentrate on three DG schemes, for numerical computations: the *Symetric Interior Penalty Galerkin Method* (SIPG, see [109]), the *Non-Symetric Interior Penalty Galerkin Method* (NIPG, see [89]) and the *Oden-Babuška-Baumann Scheme* (OBB, see [79]). We will investigate the properties of these specific schemes in detail. All three schemes can be described in the previously introduced framework. Table 2.1 shows the choice of the numerical fluxes for each of these schemes. For an extensive list of DG schemes and their numerical fluxes, see [5].

Method	$\widehat{u}$	$\widehat{\sigma}$
SIPG	$\{ u_h \}$	$\{ \nabla u_h \} - \eta h^{-1} [\![u_h]\!]$
NIPG	$\{ u_h \} + \mathbf{n}_{E_n} \cdot [\![u_h]\!]$	$\{ \nabla u_h \} - \eta h^{-1} [\![u_h]\!]$
OBB	$\{ u_h \} + \mathbf{n}_{E_n} \cdot [\![u_h]\!]$	$\{ \nabla u_h \}$

Table 2.1: Selected DG methods and their numerical fluxes.

For most of our simulations, the OBB scheme will be used because it does not depend on any additional parameters. The correct choice of additional parameters, e. g. the η in the NIPG and SIPG scheme, can become rather intricate in the case of heterogeneous problems.

Remark 2.6 (Local h): For non-uniform grids h will often be chosen locally for each $\gamma \in \Gamma$:

$$h_\gamma \propto \operatorname{diam}(\gamma) \, . \tag{2.44}$$

In two dimensions $h_\gamma = |\gamma|$ is given by the length of the edge γ. In three dimensions γ is a face and $h_\gamma = |\gamma|^{\frac{1}{2}}$ is defined using the area of the face.

A Generalized Bilinear Form

In [17] these three schemes are written as a single scheme with numerical fluxes

$$\widehat{u} = \{ u_h \} + \frac{1-\epsilon}{2} \mathbf{n}_{E_n} \cdot [\![u_h]\!] \qquad \text{and} \qquad \widehat{\sigma} = \{ \nabla u_h \} - \eta h^{-1} [\![u_h]\!], \tag{2.45}$$

the resulting bilinear form reads

$$\begin{aligned} a_\epsilon(u_h, v) = \int_\Omega \nabla u_h \cdot \nabla v \, dx \quad &+ \int_\Gamma \epsilon \{ \nabla v \} \cdot [\![u_h]\!] - \{ \nabla u_h \} \cdot [\![v]\!] \, ds \\ &+ \eta h^{-1} \int_\Gamma [\![u_h]\!] \cdot [\![v]\!] ds. \end{aligned} \tag{2.46}$$

The new parameter $\epsilon = \pm 1$ allows to switch between a symmetric and a non-symmetric bilinear form. Choosing $\epsilon = -1$ yields the classic SIPG method. For $\epsilon = 1$ the NIPG method is obtained and $\epsilon = 1$, $\eta = 0$ shows the OBB scheme.

Definition 2.15: *Using the locally defined h_γ the penalty term $\eta h^{-1} \int_\Gamma [\![u_h]\!] \cdot [\![v]\!] ds$ differs for each edge $\gamma \in \Gamma$ and we introduce the following* abbreviated form

$$\eta \int_\Gamma h_\gamma^{-1} [\![u_h]\!] \cdot [\![v]\!] ds = \sum_{\gamma \in \Gamma} \eta h_\gamma^{-1} \int_\gamma [\![u_h]\!] \cdot [\![v]\!] ds \tag{2.47}$$

2.3 Analysis

Consistency, adjoint consistency and conservativity are important properties of Discontinuous Galerkin methods. In the following we will analyze these properties for the three considered schemes.

2.3.1 Consistency

Using Definition 2.6 we want to check the consistency of the primal formulations of three selected schemes.

Lemma 2.1: *All three considered schemes, SIPG, NIPG, and OBB, are consistent.*

Proof: We consider the generalized notation of Equation (2.46). For a solution u of

(2.2) the jump $[\![u]\!]$ vanishes and the bilinear form reduces to

$$\begin{aligned} a_\epsilon(u,v) &= \int_\Omega \nabla u \cdot \nabla v \; dx \quad + \int_\Gamma \underbrace{\epsilon \{ \nabla v \} \cdot [\![u]\!]}_{=0} - \{ \nabla u \} \cdot [\![v]\!] \; ds \\ &\qquad + \underbrace{\eta h^{-1} \int_\Gamma [\![u]\!][\![v]\!] ds}_{=0} \\ &= \int_\Omega \nabla u \cdot \nabla v \; dx \quad - \int_\Gamma \{ \nabla u \} \cdot [\![v]\!] \; ds \\ &= - \int_\Omega \Delta u \, v \; dx \end{aligned}$$

Together with (2.2) consistency of all three methods is found. □

Remark 2.7 (Galerkin orthogonality): Consistency of the method implies the Galerkin orthogonality

$$a(u - u_h, v) = 0 \qquad \forall v \in V_h. \tag{2.48}$$

2.3.2 Adjoint Consistency

On the analogy of Definition 2.6 (consistency) the property of adjoint consistency is defined.

Definition 2.16 (Adjoint Consistency): *Given the solution $u \in C^2$ of the underlying problem (2.2), the primal formulation is* adjoint consistent *if u fulfills*

$$a(v,u) = \int_\Omega f \, v \, dx \qquad \forall v \in V_h. \tag{2.49}$$

Lemma 2.2: *The SIPG scheme is adjoint consistent.*

Proof: Analyzing adjoint consistency of the bilinear form (2.46) yields

$$a_\epsilon(v,u) = \int_\Omega \nabla v \cdot \nabla u \; dx \quad + \int_\Gamma \epsilon \{ \nabla u \} \cdot [\![v]\!] \; ds$$

For $\epsilon = -1$ we obtain

$$a_\epsilon(v,u) = - \int_\Omega \Delta v \, u \; dx,$$

thus the symmetric SIPG scheme is adjoint consistent. □

Remark 2.8: Only the symmetric SIPG scheme ($\epsilon = -1$) is adjoint consist, the NIPG as well as the OBB scheme ($\epsilon = 1$) do not fulfill (2.49).

2.3.3 Conservation Laws

First we want to discuss conservation of mass. It is a fundamental concept of physics along with the conservation of energy and the conservation of momentum. For mass conserving problems it is desired that this property is also guaranteed by the discretization.

For the elliptic test problem, mass conservation is related to the vector field σ (Gauss theorem):

$$\int_{\Omega} \nabla\sigma \, dx = \int_{\partial\Omega} \sigma \cdot n \, ds \tag{2.50}$$

Lemma 2.3: *A scheme in the form of Definition 2.14 guarantees a conservative discretization of σ if the numerical fluxes $\widehat{\sigma}$ are conservative.*

Proof: Testing with $v = 1$ (i. e. $v = 1$, $[\![v]\!] = 0$) Equation (2.43) becomes

$$\begin{aligned} a(u_h, 1) = &\int_{\Omega} \nabla u_h \cdot 0 \, dx + \int_{\Gamma_{\text{int}}} [\![\widehat{u} - u_h]\!] \cdot 0 - 0 \cdot \{ \widehat{\sigma} \} \, ds \\ &+ \int_{\Gamma_{\text{ext}}} [\![\widehat{u} - u_h]\!] \cdot 0 - 1 \cdot \{ \widehat{\sigma} \} \, ds \\ &+ \int_{\Gamma_{\text{int}}} \{ \widehat{u} - u_h \} [\![0]\!] - 1 [\![\widehat{\sigma}]\!] \, ds \end{aligned} \tag{2.51}$$

Assuming $[\![\widehat{\sigma}]\!] = 0$, i. e. a conservative flux, yields

$$a(u_h, 1) = \int_{\Gamma_{\text{ext}}} -1 \cdot \{ \widehat{\sigma} \} \, ds = \int_{\Omega} f \, dx. \tag{2.52}$$

Substitution of $\int_{\Gamma_{\text{ext}}} \widehat{\sigma} = \int_{\Gamma_{\text{ext}}} \sigma_h$ und $f = -\nabla\sigma_h$ gives

$$\int_{\partial\Omega} \sigma_h \cdot n \, ds = \int_{\Omega} \nabla\sigma_h \, dx. \tag{2.53}$$

Thus the scheme is conservative in σ if the flux $\widehat{\sigma}$ is conservative, i. e. $[\![\widehat{\sigma}]\!] = 0$. □

Remark 2.9: The same relation can be varified for u. The scheme is conservative in u if $[\![\widehat{u}]\!] = 0$. For all three selected schemes $\widehat{\sigma}$ is conservative, but $\widehat{u}$ is only conservative for $\epsilon = 1$, i. e. for the symmetric SIPG scheme.

Remark 2.10 (Adjoint Consistency and Conservation): A scheme is adjoint consistent if the fluxes $\widehat{\sigma}$ and $\widehat{u}$ are conservative.

Proof: The numerical flux operators applied to the test function are denoted by $\widehat{v}$ and $\widehat{\tau}$. Inserting v into the bilinear form testing with u yields

$$\begin{aligned} a(v,u) = & \int_\Omega \nabla v \cdot \nabla u \, dx + \int_\Gamma \underbrace{[\![\widehat{v} - v]\!]}_{=[\![\widehat{v}]\!]} \cdot \{ \nabla u \} - \underbrace{[\![u]\!]}_{=0} \cdot \{ \widehat{\tau} \} \, ds \\ & + \int_{\Gamma_{\text{int}}} \{ \widehat{v} - v \} \underbrace{[\![\nabla u]\!]}_{=0} - \{ u \} [\![\widehat{\tau}]\!] \, ds \\ = & \int_\Omega \nabla v \cdot \nabla u \, dx + \int_\Gamma [\![\widehat{v}]\!] \cdot \{ \nabla u \} \, ds + \int_{\Gamma_{\text{int}}} - \{ u \} [\![\widehat{\tau}]\!] \, ds \, . \end{aligned} \tag{2.54}$$

Conservation of fluxes gives

$$\begin{aligned} a(v,u) = & \int_\Omega \nabla v \cdot \nabla u \, dx + \int_{\Gamma_{\text{ext}}} \widehat{v} \, \mathbf{n} \cdot \nabla u \, ds \\ = & \int_\Omega \nabla u \cdot \nabla v \, dx + \int_{\Gamma_{\text{ext}}} v \, \mathbf{n} \cdot \nabla u \, ds \, . \end{aligned} \tag{2.55}$$

Thus the scheme is adjoint consistent. □

Chapter 3

Unfitted Discontinuous Galerkin Method

un·fit·ted */ʌn'fɪtɪd/ adj.* ***~for sth | ~to do sth*** *(formal) not suitable for something: She felt herself unfitted for marriage*

— Oxford Dictionary

Using conforming finite element methods, Barrett and Elliott [10] presented a discretization method on an unfitted mesh, i. e. the finite element mesh does not resolve the geometry and boundary conditions along the geometry are enforced weakly using Nitsche's method [78]. This method is known as the *Unfitted Finite Element Method*. However the method itself does only allow first order trial and test functions.

Recently, new interest arose in this method due to it's ability to handle interfaces or discontinuities internal to an element. The method is attractive for time-dependent problems with evolving domains, e. g. moving interfaces. It does not require remeshing. In [59], Hansbo and Hansbo propose an improved method and prove optimal convergence rates. An application to crack propagation in solid mechanics is presented in [60].

In our new *Unfitted Discontinuous Galerkin* (UDG) approach we extend the idea of Unfitted Finite Elements by using Discontinuous Galerkin methods (Chapter 2) instead of Nietzsche's method (Section 2.2.1) for the handling of essential boundary conditions. This allows the use of higher-order trial and test functions.

Remark 3.1: For problems described by a conservation equation DG methods are especially attractive. As shown in Section 2.3.3, certain DG formulations are element-wise mass conservative and therefore able to accurately describe fluxes over element boundaries.

3.1 Finite Element Space and Mesh

3.1.1 The Computational Domains

In the following, the computational domains are sub-domains of a given domain Ω.

Definition 3.1 (Sub-domains): *Let $\Omega \subseteq \mathbb{R}^d$ be a domain and $\mathcal{G}$ a disjoint partition of Ω into* sub-domains

$$\mathcal{G}(\Omega) = \left\{\Omega^{(0)}, \ldots, \Omega^{(N-1)}\right\} \tag{3.1}$$

with

$$\begin{aligned} \Omega^{(i)} &\subseteq \Omega && \forall \quad 0 \leq i < N, \\ \Omega^{(i)} \cap \Omega^{(j)} &= \emptyset && \forall \quad 0 \leq i < j < N, \\ \partial\Omega^{(i)} \cap \partial\Omega^{(j)} &= \Gamma^{(i,j)} && \forall \quad 0 \leq i < j < N, \\ \bar{\Omega} &= \bigcup_{i=0}^{N-1} \bar{\Omega}^{(i)}. \end{aligned} \tag{3.2}$$

On each $\Omega^{(i)}$ we consider an elliptic partial differential equation

$$L_i(u_i) = f_i \tag{3.3}$$

with a linear differential operator L_i together with suitable boundary conditions on $\partial\Omega$ and transmission conditions on the interfaces $\Gamma^{(i,j)}$.

Remark 3.2 (Geometric Interpretation): The partition $\mathcal{G}$ is usually based on geometric properties obtained from experiments or previous simulations. The boundaries $\partial\Omega^{(i)}$ may have a complex shape.

3.1.2 The Mesh

In addition to the partitioning $\mathcal{G}$ we consider a triangulation $\mathcal{T}$ of the domain Ω, see Definition 2.2. $\mathcal{T}$ can be chosen independently of $\mathcal{G}$. Reaping benefit from the Discontinuous Galerkin method, the trial and test functions are defined on $\mathcal{T}$ and their support is restricted to $\mathcal{G}$. This approach allows to choose the size of the approximation space independent of the geometric properties.

Definition 3.2 (Fundamental Mesh): *We call $\mathcal{T}(\Omega)$ the* fundamental mesh, *its elements are denoted by $\mathcal{T}(\Omega) = \{E_0, \ldots, E_{M-1}\}$.*

Remark 3.3 (Mesh Size): The triangulation $\mathcal{T}$ is a partition of Ω, where the mesh size h (see Equation 2.5) is not directly determined by the shape of $\mathcal{G}$. Nevertheless, error control on the solution of the partial differential equations (3.3) might require mesh refinement dependent to the shape of $\mathcal{G}$.

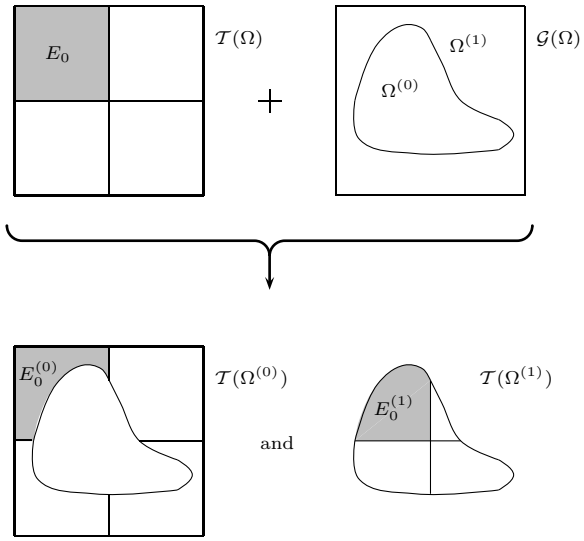


Figure 3.1: Construction of the partitions $\mathcal{T}(\Omega^{(i)})$ of a sub-domain $\Omega^{(i)}$ given the partitions $\mathcal{G}$ and $\mathcal{T}$ of the domain Ω.

Definition 3.3 (Sub-domain Mesh): *For each $\Omega^{(i)} \in \mathcal{G}$ a mesh based on equation (2.3a) is defined (see Figure 3.1):*

$$\mathcal{T}(\Omega^{(i)}) = \left\{ E_n^{(i)} = \Omega^{(i)} \cap E_n \;\middle|\; E_n^{(i)} \neq \emptyset \right\}. \tag{3.4}$$

Note that $E_n^{(i)}$ is always a subset of E_n, therefore we will call E_n the fundamental element of $E_n^{(i)}$.

Remark 3.4: For practical reasons, we require that $E_n^{(i)}$ is path-wise connected, i. e. any two points in $E_n^{(i)}$ can be connected by a curve lying completely within $E_n^{(i)}$ (see [108]).

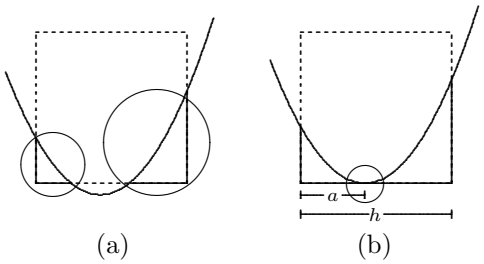


Figure 3.2: Situations with non-connected parts of one $E_n^{(i)}$ can occur. Often these cases vanish for $h < h_{\min}$ small enough (a), but there are situations where they never vanish (b).

Not for every pair of $\mathcal{G}(\Omega)$ and $\mathcal{T}(\Omega)$ this requirement can be guaranteed. As the partition $\mathcal{G}$ and $\mathcal{T}$ are defined independently of each other, situations may occur where $E_n^{(i)} = \Omega^{(i)} \cap \Omega_n$ consists of two or more unconnected parts. Often these cases vanish choosing h small enough (see Figure 3.2 (a)), such that for small h the common estimate for the convergence error applies. An example where a decrease in h cannot resolve the problematic situation of two unconnected parts is shown in Figure 3.2 (b): A sub-domain $\Omega^{(i)}$ has a point of contact with an edge of the grid and the point's offset a along the edge fulfills that $\frac{a}{h}$ is irrational.

Analogous to Definition 2.3 and 2.4, we introduce the following definitions for the sub-domain $\Omega^{(i)}$:

Definition 3.4 (Internal Skeleton of Sub-domain Mesh): *The* internal skeleton $\Gamma_{int}^{(i)}$ *of the sub-domain mesh is given by*

$$\Gamma_{int}^{(i)} = \left\{ \gamma_{e,f} = \partial E_e^{(i)} \cap \partial E_f^{(i)} \;\middle|\; E_e^{(i)}, E_f^{(i)} \in \mathcal{T}(\Omega^{(i)}),\ E_e^{(i)} \neq E_f^{(i)} \text{ and } |\gamma_{e,f}| > 0 \right\}. \tag{3.5}$$

Definition 3.5 (External Skeleton of Sub-domain Mesh): *The* external skeleton $\Gamma_{ext}^{(i)}$ *is denoted by*

$$\Gamma_{ext}^{(i)} = \left\{ \gamma_e = \partial E_e^{(i)} \cap \partial \Omega^{(i)} \;\middle|\; E_e^{(i)} \in \mathcal{T}(\Omega^{(i)}) \text{ and } |\gamma_{e,f}| > 0 \right\}. \tag{3.6}$$

3.1.3 The Basis Functions

Following the common Discontinuous Galerkin approach as described in Chapter 2 we use piecewise polynomial basis functions.

Remark 3.5: Since each element $E_n^{(i)}$ in the outlined finite element mesh $\mathcal{T}(\Omega^{(i)})$ can be shaped arbitrarily, Discontinuous Galerkin methods are particularly attractive (Remark 2.4). For such complicated grids it is hard to use conforming trial and test functions. Since conforming basis functions depend on the shape of the elements, it would be necessary to construct a suitable local basis for each $E_n^{(i)}$ in accordance to local boundary conditions.

We use a DG formulation with a discontinuous, piecewise polynomial approximation. Recalling Definition 2.8, the space of polynomial functions of degree k is given as

$$P_k = \left\{ u : \mathbb{R}^d \to \mathbb{R} \;\middle|\; u(x) = \sum_{|\alpha| \leq k} c_\alpha x^\alpha \right\}, \tag{3.7}$$

with a multi-index α.

The finite element space for the discretization of $L_i u_i = f_i$ on $\Omega^{(i)}$ is defined by

$$V_h^{(i)} = \left\{ v \in L_2(\Omega^{(i)}) \;\middle|\; v|_{E_n^{(i)}} \in P_k \right\}, \tag{3.8}$$

and consists of piecewise polynomials with discontinuities on the internal skeleton $\Gamma_{\text{int}}^{(i)}$.

Definition 3.6 (UDG Local Basis Functions): *The* local basis functions $\varphi_{n,j}^{(i)}$ *are given by polynomials* $\varphi_j \in P_k$ *with their support restricted to* $E_n^{(i)}$*:*

$$\varphi_{n,j}^{(i)} = \begin{cases} \varphi_j & \textit{inside of } E_n^{(i)} \\ 0 & \textit{outside of } E_n^{(i)} \end{cases} . \tag{3.9}$$

Remark 3.6: Since the assembling of the stiffness matrix requires integration over the volume of $E_n^{(i)}$ and over the surface $\partial E_n^{(i)}$, for elements $E_n^{(i)}$ significantly smaller than the fundamental element E_n only a very small part of $\varphi_{n,j}$ is integrated. This means that their matrix entries become very small. Such small matrix entries increase the condition of the matrix and pose numerical problems especially for the linear solver. In order to avoid this difficulty the local basis functions are scaled linearly according to the bounding box of the element $E_n^{(i)}$. This scaling ensures that $\min(\varphi_{n,j}) = 0$ and $\max(\varphi_{n,j}) = 1$ on the bounding box of the element $E_n^{(i)}$.

3.2 Integration using a Local Triangulation

Assembling the local stiffness matrix in a DG approach requires integration over the volume of each element $E_n^{(i)}$ and its surface $\partial E_n^{(i)}$. As stated in Remark 3.5, the mesh elements obtained by the UDG method might exhibit very complicated shapes, thus evaluation of these volume and surface integrals is a very involved process.

For the integration of a function on irregular shaped domains quadrature rules based on interpolation functions are not directly applicable. A common approach is to use Monte-Carlo integration. For Monte-Carlo integration the integration error converges with Err $\propto 1/\sqrt{N}$ [107], where N is the number of sampling points. Thus, the costs for the integration of a polynomial function on a single element using this kind of integration are rather high compared to methods based on interpolation functions, e. g. Gauss quadrature.

As the number of elements with complicated shapes can be very high in UDG meshes, a different approach is chosen here. Integration is based on a local triangulation of $E_n^{(i)}$ (see Figure 3.3). To do so, $E_n^{(i)}$ is subdivided into a disjoint set $\{E_{n,k}^{(i)}\}$ of simple geometric objects, i. e. simplices and hypercubes, with

$$\begin{aligned} E_{n,k}^{(i)} &\subseteq E_n^{(i)} \\ E_{n,k}^{(i)} \cap E_{n,l}^{(i)} &= \emptyset \qquad\qquad \forall \quad k \neq l, \\ \bar{E}_n^{(i)} &= \bigcup_k \bar{E}_{n,k}^{(i)}. \end{aligned} \tag{3.10}$$

For each of these simple geometric objects efficient Gaussian type quadrature rules can be applied. For a comprehensive overview of numerical integration methods, see e. g. [41].

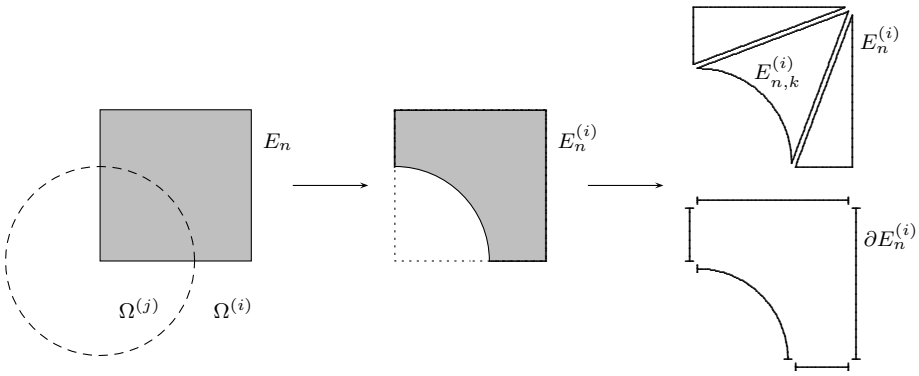


Figure 3.3: Construction of $E^{(i)}$ from its fundamental element E_n and the sub-domain $\Omega^{(i)}$. On the right: the local triangulation of $E_n^{(i)}$ and $\partial E_n^{(i)}$.

This construction implies that the integration of an element $E_n^{(i)}$ with a complex shape requires many integration parts $E_{n,k}^{(i)}$. On the analogy of Equation (2.4), we have a smooth one-to-one mapping $T_{E_{n,k}^{(i)}}$ from a reference element $\hat{E}$ to $E_{n,k}^{(i)}$:

$$E_{n,k}^{(i)} = T_{E_{n,k}^{(i)}}(\hat{E}). \tag{3.11}$$

Remark 3.7: For a good boundary approximation either very fine local triangulations or isoparametric elements [57] must be used. Iso-parametric elements are used in the implementation described in 3.2.2. This technique is used for the simulations and the stability analysis presented in Sections 5.3.2 and 5.4, where boundaries are approximated using quadratic mapping (higher-order mappings have not been implemented yet, but should also be feasible).

Using standard quadrature formulae, $\mathcal{Q} = \{(q_i, w_i)\}$ denotes a set of pairs of integration points and scalar weights on the reference elements $\hat{E}$. The integral over a globally defined function f can be approximated on $E_n^{(i)}$ as

$$\int_{E_n^{(i)}} f dV \approx \sum_k \sum_j f(T_{E_{n,k}^{(i)}}(q_j))\ w_j\ |\det(J_{T_{E_{n,k}^{(i)}}}(q_j))\ |, \tag{3.12}$$

with $J_{T_{E_{n,k}^{(i)}}}$ denoting the Jacobian matrix of the mapping $T_{E_{n,k}^{(i)}}$.

Since basis functions are defined in local coordinates (ξ, η) on $\hat{E}$, the integral over a local basis function $\varphi \in P_k$ is given by (see Figure 3.4)

$$\int_{E_n^{(i)}} \varphi \circ T_{E_n}^{-1} dV \approx \sum_k \sum_j \varphi((T_{E_n}^{-1} \circ T_{E_{n,k}^{(i)}})(q_i))\ w_j\ |\det(J_{T_{E_{n,k}^{(i)}}}(q_j))\ |. \tag{3.13}$$

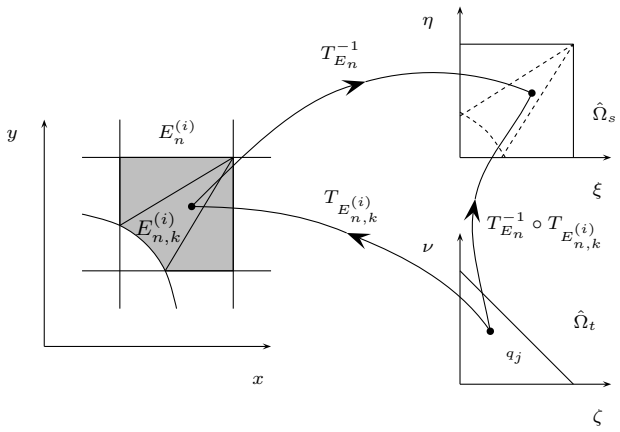


Figure 3.4: Transformations from the reference triangle $\hat{E}_t$ to the reference square $\hat{E}_s$ are realized by concatenating the transformation $T_{E_{n,k}^{(i)}}$ from the reference triangle to global coordinates and $T_{E_n}^{-1}$ onto the reference square.

Remark 3.8: On a structured grid T_{E_n} consists only of a scaling and a translation, hence it is easy to compute the inverse $T_{E_n}^{-1}$. The costs for evaluation of the composite mapping $T_{E_n}^{-1} \circ T_{E_{n,k}^{(i)}}$ are dominated by that of $T_{E_{n,k}^{(i)}}$.

3.2.1 Construction of Local Triangulations

Since the local triangulation is used for integration only, its construction is much simpler than the construction of a conforming Finite Element mesh. For finite element meshes different criteria concerning the size of angles in and the aspect ratio of the elements must be met [7, 68].

Remark 3.9 (Constraint-free Shapes): The local triangulation does not pose any constraints on the angles within or the aspect ratio of the integration parts $E_{n,k}^{(i)}$ as long as $J_{T_{E_{n,k}^{(i)}}}^{-1}$, the inverse of the Jacobian matrix, exists at all quadrature points $(q_i, w_i) \in \mathcal{Q}$.

Remark 3.10 (Locality): The construction of the local triangulation is an operation which is completely local. No information about neighboring elements is required.

The local triangulation algorithm consists of two parts. First the elements E_n are repeatedly bisected to create a set of sub-elements $\{R_{n,k}\}$. In order to keep the implementation simple cuboid shaped sub-elements $R_{n,k}$ are preferred. Then each $R_{n,k}$ is classified according to the way $R_{n,k}$ intersects with the interfaces $\Gamma^{(i,j)}$. Choosing suitable rules to control the bisection the set of classes can be kept small. For each of these classes a predefined triangulation is chosen.

For the UDG method no particular local triangulation is required. Many different approaches for the local triangulation are possible. In the reference implementation two different algorithms are implemented and were used in the numerical experiments in Chapter 5, 6, and 7.

3.2.2 Local Triangulation of Analytically Described Geometries

For analytically described geometries a local triangulation is implemented for domains in two space dimensions. The geometries are given as a list of primitives, e. g. circles and splines, as used in Computer Aided Design (CAD). These are approximated by second-order iso-parametric elements.

Currently the implementation supports only a very limited set of CAD primitives. Every new primitive, as well as the extension to $\mathbb{R}^3$, would require additional bisection rules. We do not regard this approach practical for three space dimensions, as the complexity of the local triangulation code would be nearly as high as the complexity of unstructured mesh generators.

Bisection Rules

Recursive bisection creates a set of sub-rectangles $\{R_{n,k}\}$. This process is controlled by two rules. First, for the shape of each interface $\Gamma^{(i,j)}$ a set of bisection points is selected where the rectangles are split along one or more Cartesian axis (see Figure 3.5). At the end of this Section we give an overview of the requirements which are taken into account for the choice of these *bisection* points. Which points are chosen does not affect the algorithm itself, but it does affect the number of classes how $\Gamma^{(i,j)}$ intersects with $R_{n,k}$.

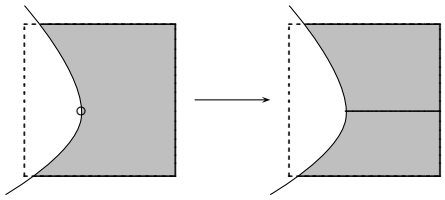


Figure 3.5: Bisection at "special" points, which are determined by the shape of the interface $\Gamma^{(i,j)}$.

The second criterion for bisection is the number of sub-domains intersecting with the sub-rectangles (see Figure 3.6). We continue the bisection until we have only one sub-domain intersecting with each sub-rectangle. In practice, one might get cases where the second criterion forces a very deep subdivision of the element. In such cases one could stop the bisection at a minimal diameter h_{min} of the sub-rectangle. This would require to handle additional special cases or to constrain the shape of $\Omega^{(i)}$.

One can find different strategies to fulfill this second criterion. We have chosen to bisect at the intersection points between the edges of the sub-rectangle and the

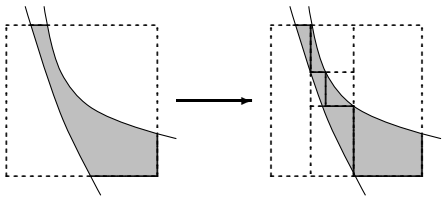
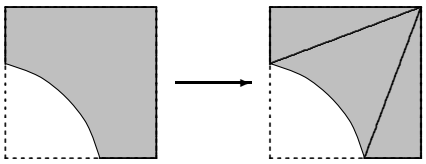


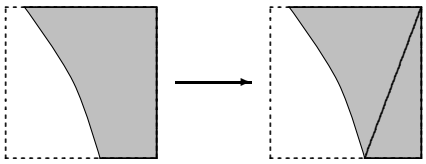
Figure 3.6: Recursive bisection until each rectangle intersects with not more than one interface $\Gamma^{(i,j)}$.

interface $\Gamma^{(i,j)}$.

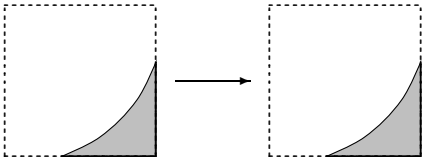
Intersection Classes



(a) one corner missing: connect the intersection points to the corner opposite of the missing one.



(b) two corners missing: cut diagonally so that the resulting diagonal is as short a possible.



(c) three corners missing: nothing to be done at all.

Figure 3.7: Basic intersection-cases for $\mathbb{R}^2$: Choosing suitable rules to control the bisection one obtains three classes how $\Gamma^{(i,j)}$ can intersect with $R_{n,k}$.

Using the outlined rules to control the bisection, we obtain three classes how $\Gamma^{(i,j)}$ can intersect with $R_{n,k}$. For each of these classes we define a triangulation rule that directly implies certain rules to create the triangles (see Figure 3.7). In the case of three-dimensional domains one obtains a lot more classes. For the case of linear subelements one may refer to [84].

Choosing *Bisection* Points

To obtain a small number of classes, certain cases should be avoided. This is achieved during the first part of the recursive bisection. We explicitly choose points to bisect so that we avoid:

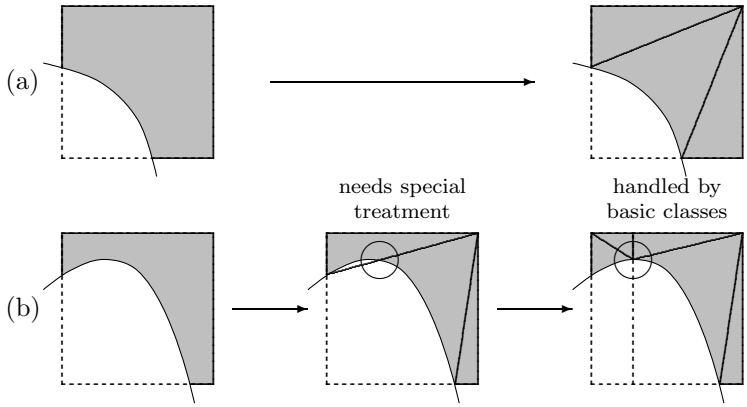


Figure 3.8: Forcing a bisection along the normal vector $\hat{n}$ at points $p \in \Gamma^{(i,j)}$ where $\hat{n}$ is parallel or anti-parallel to $\hat{e}_x$ or $\hat{e}_y$ makes the second case coincide with the same class as the first one. Otherwise one would need a separate treatment to avoid an intersection between the cutting edge and the curvilinear edge.

- Discontinuities in the First Derivative: As it is not possible to find a smooth mapping for an integration part with an edge exhibiting a discontinuity in the first derivative, we require bisection at all points where $\Gamma_n^{(i,j)}$ is not differentiable.

- Multiple Intersections: $E_n^{(i)}$ is not necessarily convex so that a line between two corners of $E_n^{(i)}$ might intersect with the surface $\partial E_n^{(i)}$. This can be avoided by forcing a bisection along the normal vector $\hat{n}$ at points $p \in \Gamma^{(i,j)}$ where $\hat{n}$ is parallel or anti-parallel to $\hat{e}_x$ or $\hat{e}_y$. This makes the second case coincide with the same class as the first one. Otherwise one would need a separate treatment to avoid an intersection between the cutting edge and the curvilinear edge (see Figure 3.8).

3.2.3 Local Triangulation of Implicitly Described Geometries

In applications dealing with biological or environmental systems, detailed measurements of complex geometries, e. g. pore scale structures, are most often obtained using imaging technology. These measurements, e. g. X-ray tomography, yield data on a structured grid. Image processing techniques lead to a data set where the sub-domain boundary $\partial\Omega^{(0)}$ is given by a threshold value.

In order to simplify both the preparation of the data as well as the computations on the domain, an alternative method for the local triangulation is proposed. Instead of reconstructing CAD primitives from the image data this alternative local triangulation algorithm can directly use this experimentally obtained data. An implementation is available for two and three dimensions. For simplicity we describe the algorithm for

two space dimensions. The domain $\Omega^{(0)}$ is described by a scalar function ϕ:

$$\phi(x) = \begin{cases} > 0 & \text{if } x \in \Omega^{(i)} \\ \leq 0 & \text{else} \end{cases} . \tag{3.14}$$

This scalar function can usually be obtained through post processing of image data, e. g. from CT images. The sub-domain boundary $\partial\Omega^{(0)}$ is given as an iso-surface of the scalar function – see Figure 3.9.

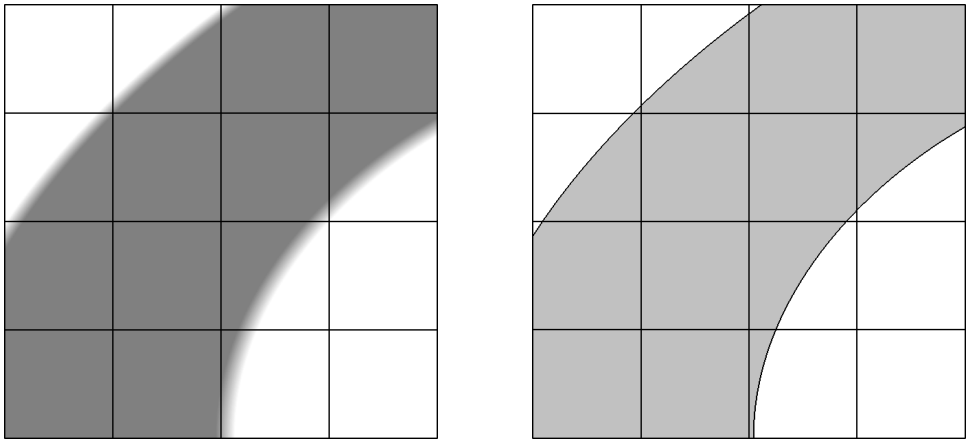


Figure 3.9: A scalar function (left) defines the geometry $\mathcal{G}$. The sub-domain boundary $\Gamma^{(i,j)}$ is given as the iso-surface of value 0.0 (right).

Remark 3.11 (Level set methods – moving geometries): For free-boundary problems and computations on evolving surfaces, level set methods are becoming very popular [47, 52, 93, 95, 100]. In level set methods, a level set function $\phi(x(t), t)$ is computed and the iso-surface $\phi = 0$ represent the position of the domain boundary at time t. Often ϕ is constructed as a signed distance function in order to speed up certain computations, e. g. curvature of the boundary.

Again the sub-domain boundary $\partial\Omega^{(0)}$ is given as an iso-surface of the scalar function. Thus, the UDG method can be very well combined with free-boundary simulations and moving geometries.

Definition 3.7 (Image Grid): *The scalar function ϕ describing the domain boundary is discretized as a piecewise linear function on a fine grid with mesh width h_g. As this data is most often obtained from image data, this grid is called* Image Grid.

Remark 3.12: Often the image data is given on a much finer grid than the one used for computations. In this case it is required that the image grid is a hierarchic refinement of the computation grid.

The local triangulation is based on the *Marching Cubes* algorithm which was developed to give a reconstruction of an iso-surface of a scalar function.

The Marching Cubes Algorithm

The Marching Cubes (MC) algorithm was presented in 1987 by Lorensen and Cline in [72]. It has its origin in computer graphics and generates a set of triangles for the visualization of an iso-surface of a scalar function ϕ.

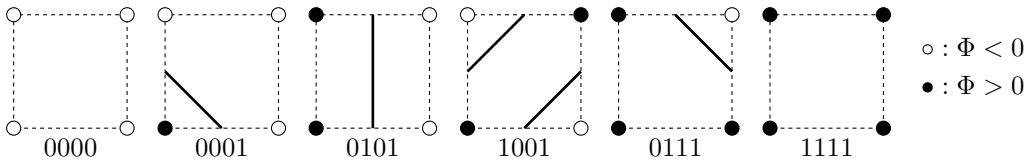


Figure 3.10: The Marching Cube algorithm in $\mathbb{R}^2$ distinguishes six basic cases depending on the value of a scalar function ϕ in the corners. The figures show these six different cases together with their keys in the lookup table.

The algorithm is based on a piecewise linear representation of ϕ on a cubical structured grid and operates locally on the reference cube of each cell. Each vertex of an element in the image grid can have a value below or above the threshold value of the iso-surface, i. e. inside or outside the sub-domain. For a cube element in $\mathbb{R}^2$ this results in 16 different cases. Each of these cases corresponds to one of six basic cases and can be transformed using simple geometric operations, shown in Figure 3.10. To obtain a fast reconstruction algorithm a *Triangle Lookup Table* is used, where the information about the triangles that form the iso-surface is stored for each of the 16 cases. The index for the lookup table is computed by treating the scalar values of each of the vertices as a bit in an integer number. The bit value is chosen as 0 or 1, depending on whether the vertex value is below or above the value of the iso-surface (see the key values in Figure 3.10).

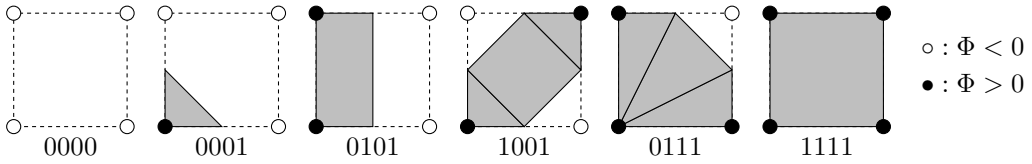


Figure 3.11: Extended Marching Cube algorithm adds a volume reconstruction to all six basic cases of the original MC algorithm.

The original algorithm provides a surface reconstruction of Γ_{ij} but still a volume reconstruction is needed. In order to obtain this, the original algorithm is extended with a *Volume Lookup Table*, providing a list of simple sub-volumes for each case. The reconstructions for the six basic cases are shown in Figure 3.11. In order to keep the list of sub-volumes small the code is not restricted to simplices, but allows different simple geometric objects.

The classic Marching Cubes algorithm cannot guarantee a topologically correct reconstruction. Situations are possible where two topologically different setups lead to

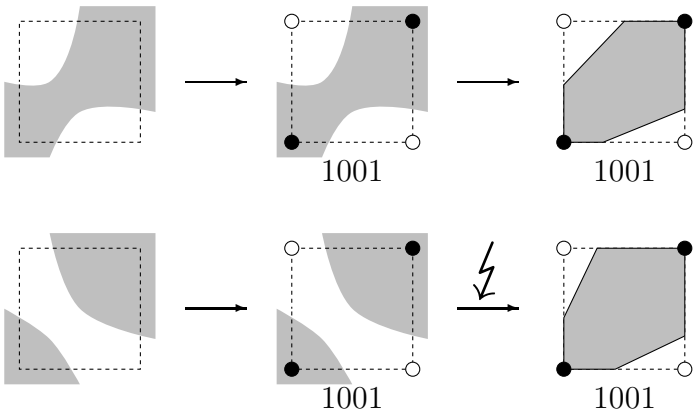


Figure 3.12: The classic Marching Cubes algorithm cannot guarantee a topologically correct reconstruction. Different functions give the same key into the lookup table but only in the first case the reconstruction is correct.

the same key into the lookup table. In Figure 3.10 and Figure 3.11 the case 1001 is ambiguous. This problem is illustrated in Figure 3.12. The sign of the values in the vertices do not yield enough information about the function inside the volume. The two vertices inside the domain can either be connected or not, leading to a completely different surface. The Marching Cubes algorithm arbitrarily chooses one of the two possible reconstructions. Only in one of the two cases the reconstruction is topologically correct.

Topologically Correct Marching Cubes Algorithm

In 1995 Chernyaev [35] proposed a modification of the original marching cubes algorithm that allows a topologically correct surface reconstruction. It is called *Marching Cubes 33*. The idea is to evaluate the value at the weighted cell center, or in 3D at the weighted cell and face centers, of the multi-linear function ϕ. This allows to disambiguate the reconstruction (see Figure 3.2.3).

For an efficient implementation, further lookup tables are used with an index based on the evaluation of the additional node values. An exhaustive description of the algorithm and of the additional lookup tables is provided in [71].

For the local triangulation of implicitly given domains, Volume Lookup Tables for all different cases of the MC33 algorithm are provided. The list of volume triangulations can be found in Appendix B.

The Local Triangulation Algorithm

Using the Marching Cubes algorithm a local triangulation for elements in the computational mesh is constructed.

If an element $E_n^{(i)}$ intersects with the domain boundary, a set of sub-rectangles $\{R_{n,k}\}$

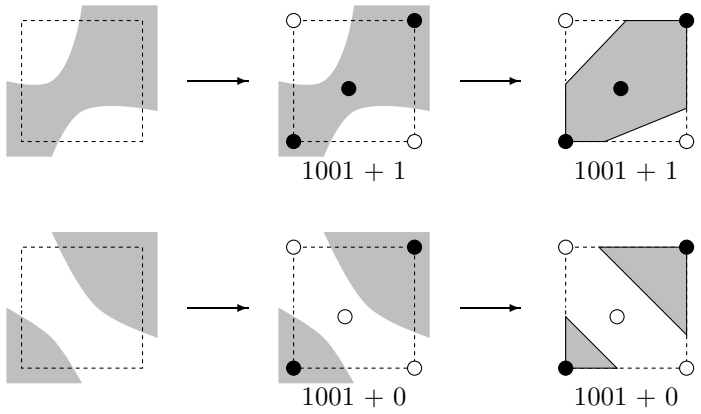


Figure 3.13: To resolve the topological ambiguities of the classic Marching Cube algorithm, the Marching Cubes 33 algorithm tests the function value at additional points. This allows a topologically correct reconstruction of the iso-surface.

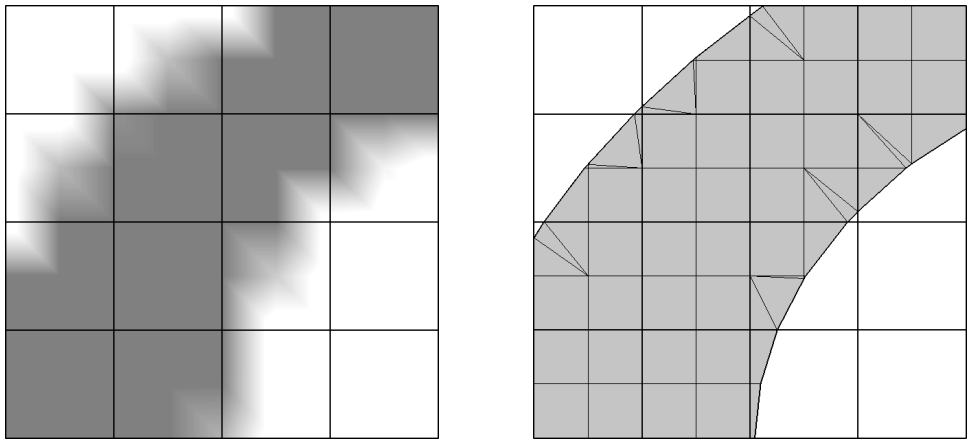


Figure 3.14: A scalar function defines the geometry $\mathcal{G}$. A piecewise linear representation (left) is obtained from some imaging process. The local triangulation (right) is constructed using the extended Marching Cubes algorithm.

is formed by all those elements of the image grid which are a descendant of $E_n^{(i)}$. Note that this is possible as we require that the image grid is a hierarchic refinement of the fundamental grid.

On each rectangle $R_{n,k}$, the scalar function ϕ is multi-linear and the Marching Cubes algorithm gives a reconstruction of the domain boundary, i.e. an iso-surface of ϕ. Using the Marching Cubes 33 algorithm, topologically correct reconstruction also for complicated geometries is possible. Using an extended lookup table a volume reconstruction is obtained as well.

The volume reconstruction is used for the evaluation of volume integrals on $E_n^{(i)}$. Evaluation of boundary integrals is based on the surface reconstruction of the iso-surface. Volume triangulations of the face are necessary when computing the intersections to neighboring cells of the computational mesh.

To sketch an example the scalar function in Figure 3.9 is considered. Figure 3.14 shows the piecewise linear interpolation on a mesh with $h_i = 1/8$, the corresponding reconstruction of the implicitly given domain and the resulting local triangulation for a computational mesh with $h = 1/4$.

3.3 Comparison with other Methods

The problem of disentangling the construction of the finite element space from the geometric properties has been approached in a number of different methods. We will compare the UDG approach to the classic conforming Finite Element method as well as to other methods featuring geometry independent grids.

3.3.1 Conforming Finite Elements

The main difference between Discontinuous Galerkin methods and Conforming Finite Element methods must be seen in the continuity requirement. Conforming Finite Elements require basis functions which are continuous across element boundaries.

The term *"conforming"* is used in two contexts. First it is used for the requirement of conforming, i. e. continuous, trial and test functions [30].

Definition 3.8 (Conforming Finite Elements): *A* conforming Finite Element method *is one in which the discrete space V_h is a subspace of the Sobolev space V of the continuous problem.*

Second it is used to name a certain class of grids.

Definition 3.9 (Conforming mesh): *A triangulation $\mathcal{T}$ is called* conforming *if it fulfills the following requirements.*

1. *$\mathcal{T}$ is a non-overlapping partition of Ω in the sense of Definition 2.2.*
2. *If the intersection $I_{nm} = \bar{E}_n \cap \bar{E}_m, n \neq m$, is non-empty, then I_{nm} is a sub-entity, e. g. point, edge, face and so forth of E_n and E_m.*

Following the standard finite element paradigm, one would create a finite element mesh of the whole domain Ω or a sub-domain $\Omega^{(i)}$ so that the elements resolve the boundaries $\partial\Omega^{(i)}$. Most finite element methods require a conforming triangulation, at least within each $\Omega^{(i)}$. A non-conforming treatment of interfaces $\Gamma^{(i,j)}$ is possible with Mortar Finite Elements [26]. However, constructing a triangulation of good quality is very difficult, especially in three space dimensions. High quality tetrahedral mesh generation has been worked on for many years and still is a non-trivial problem [101]. Moreover, approximation errors of finite element schemes and the convergence behavior of iterative linear solvers depend on the mesh quality. Resolving the shape of $\partial\Omega^{(i)}$ might require very fine grids, resulting in a large number of degrees of freedom. Recent developments concentrate on mesh generation for curved elements [74] which would reduce the number of elements, but these algorithms are still not generally applicable. In Section 5.6 we will present a comparison of the size of the function space necessary to obtain a certain approximation quality using standard finite element methods and using the UDG approach.

3.3.2 Geometry Independent Methods

Generally, one can distinguish two approaches to obtain geometry independent discretization. One idea is to embed $\Omega^{(i)}$ in a larger domain Ω and to find a suitable extension of the PDE outside the domain $\Omega^{(i)}$, so that a solution of the PDE on Ω gives also a solution for the PDE on $\Omega^{(i)}$. The boundary and transmission conditions pose some kind of constraint on the solution. The different methods vary in the way the constraints are imposed. The other idea is to modify the basis functions such that they comply with the boundary conditions.

Embedding Domain Techniques

The techniques of using an embedding domain date back to the work on Embedding Domain methods of Buzbee et al. [32]. Nowadays they are more commonly known as Fictitious Domain methods, which were extensively studied, e. g. by [56]. These methods are based on an arbitrary grid irrespective of the boundaries $\partial\Omega^{(i)}$; usually this will be a structured grid. A conforming Finite Element discretization is applied on the whole domain Ω, neglecting the internal boundaries. The internal boundary conditions on the interfaces $\Gamma^{(i,j)}$ are imposed as constraints on the involved partial differential equations. The resulting problem is solved using the technique of Lagrange multipliers. This method successfully decouples the number of unknowns from the shape of interfaces $\Gamma^{(i,j)}$, but it needs additional degrees of freedom to formulate the constraints. Solving the modified problem is quite expensive, because the Lagrange multiplier technique results in a saddle point problem.

The Immersed Boundary Method [83] and Immersed Interface Method [69] are based on the same idea as the Fictitious Domain method, but the constraints are introduced using virtual forces, contributing to the right hand side, i. e. the source–sink term.

The techniques developed for Fictitious Domain and Immersed Interface methods are also very popular in combination with level set methods [51, 54].

In contrast to these two methods the Finite Cell method [82] does not introduce any additional constraints or forces. The PDE is defined on a larger domain and the material parameters are adjusted to guarantee existence of the solution on the whole domain. The methods work well for elasticity problems, but must be modified for each new equations. It is not clear whether this approach is generally applicable.

Modified Basis Functions

The second class of methods modify the basis functions according to the geometry. A wide range of different methods are available.

In 1987 Barrett and Elliott proposed the idea of *Unfitted Finite Elements*. The basis functions are given on a grid that not only covers, but overlaps $\Omega^{(i)}$. Outside $\Omega^{(i)}$ the function is defined to be 0. These ansatz functions are not elements of H_0^1, essential boundary conditions are imposed using the technique of Nitsche (see Section 2.2.1). Our *Unfitted Discontinuous Galerkin* method (UDG) extends this idea and allows to use higher-order ansatz functions.

Composite Finite Element methods, as introduced in [58], were developed to improve geometric multigrid methods on domains with complicated structures and micro structures. They are based on hierarchic grid constructions, where the finest grid must resolve the geometric structure. Conforming trial and test functions are used on the finest grid. Basis functions for coarser meshes are constructed as linear combinations of the basis functions of the conforming grid. This approach was primarily intended as a fast iterative solver, not a discretization scheme. Furthermore the construction of the coarse grid basis functions can become very expensive, especially for higher-order trial functions. In recent work the method is also used for the construction of a coarse discretizations [86].

In the 1990s Babuška and others started with what is called the *Partition of Unity* method [8] or *Generalized Finite Element* method [99]. This approach does not only weaken the constraints on the Finite Element mesh, it doesn't even use a mesh in the classical sense any more. A finite set of patches is constructed which overlap each other and cover the whole domain. Ansatz functions are defined patch-wise and are 0 outside, similar to DG methods. On each patch a weighting function $w_j(x)$ is given such that they form a partition of unity $\sum w_j(x) = 1$. The basis of the global function space is constructed by multiplying the local basis functions ϕ_i with w_j. The method allows great freedom in the choice of the local basis functions. A priori knowledge of the solution (e. g. discontinuities at interfaces or singularities at re-entrant corners) can be taken into account when choosing the shape functions. Essential boundary conditions are either imposed using Nitsches method, or using Lagrange multipliers. Difficulties are the numerical integration on arbitrarily shaped intersections of the supports of the different ansatz functions and the integration of non-polynomial functions.

The idea of the so-called *Extended Finite Element* method (XFEM) [44, 24] is to

start from classical FEM and enrich the function space by additional basis functions to incorporate discontinuities and other boundary effects. Again the meshes are independent from the geometry, but the enrichment does introduce additional unknowns. XFEM is very popular in the simulation of crack growth [76, 43] where it helps to avoid repeated remeshing which is necessary in classical approaches.

Chapter 4

Implementation

People think that computer science is the art of geniuses but the actual reality is the opposite, just many people doing things that build on each other, like a wall of mini stones.
— Donald E. Knuth

The implementation of the presented work is based on the DUNE framework. In this chapter we will shortly describe the concepts of DUNE, as well as the design of the UDG code. For further details on DUNE we refer to [18, 19, 20] (The DUNE website[1] provides all the documentation, implementation and development details on-line).

4.1 The DUNE Framework

DUNE is an acronym standing for *Distributed and Unified Numerics Environment*. It provides a framework for solving partial differential equations using grid based methods.

The software is designed as a modular system. DUNE provides a set of *core modules* on which further modules and applications are based. Core modules are modules with a sufficiently stable interface and a certain maturity of the code. The implementation of DUNE follows three main principals:

Flexibility: Users should be able to write generic components, which can be reused in many different applications.

Efficiency: Scientific computing has an unlimited demand for computing power. The implementation must avoid big performance losses as the price for a clean interface.

External libraries: Users must be able to incorporate existing code and libraries into their new applications. It must be possible to reuse large functionality of existing finite element packages.

[1] `http://www.dune-project.org/index.html`

The central idea in the DUNE software design is the separation of data structures and algorithms by abstract interfaces. Often numerical codes are designed with a particular application in mind, leading to a tailored data model, which is later hard to extend when implementing new algorithms and applications. The separation of data structures and algorithms offers high flexibility and reduces the code size. Thus it increases maintainability and extendibility of the framework.

High-level interfaces allow the implementation of applications without knowledge of the underlying structures. Such layers of abstraction usually add an overhead, leading to a performance penalty. An efficient implementation of the interface is provided using generic programming techniques, such as static polymorphism and traits [103].

The use of generic programming techniques for the efficient separation of data structures and algorithms is well known from the Standard Template Library (STL) [77], which is part of the C++ standard library. The most important aspect of generic programming with respect to performance is that dynamic polymorphism, realized with virtual functions in C++, is replaced by static (or compile-time) polymorphism. This allows the compiler to inline interface implementation methods and to apply its full range of optimization techniques. As a consequence the abstract interface is effectively eliminated at compile time and *small* methods (consisting of only a few machine instructions) do not impose a performance penalty. Thus interfaces can be defined at any level of program design, e. g., even for the access to individual elements of a vector.

4.2 Design of the *dune-udg* Module

In Chapter 3 the Unfitted Discontinuous Galerkin Method was introduced. The method is implemented in a separate DUNE Module. It is designed for easy incorporation into new DUNE applications and easy implementation of new DG discretizations.

The `dune-udg` module is build ontop (see Figure 4.1) the three dune-core modules `dune-common`, `dune-grid` and `dune-istl`. For the discretization many components are used from the `dune-disc` module, including extensions especially designed for UDG.

The `dune-grid` module is used to handle the fundamental mesh $\mathcal{T}(\Omega)$. For the solution of the linear equation system a range of solvers is available in the `dune-istl` module.

4.2.1 UDGAssembler

Member	Description
`void assemble()`	assemble the global matrix

Table 4.1: Public interface of the `Dune::UDGAssembler` class.

The central component of `dune-udg` is the assembler class `Dune::UDGAssembler`. It

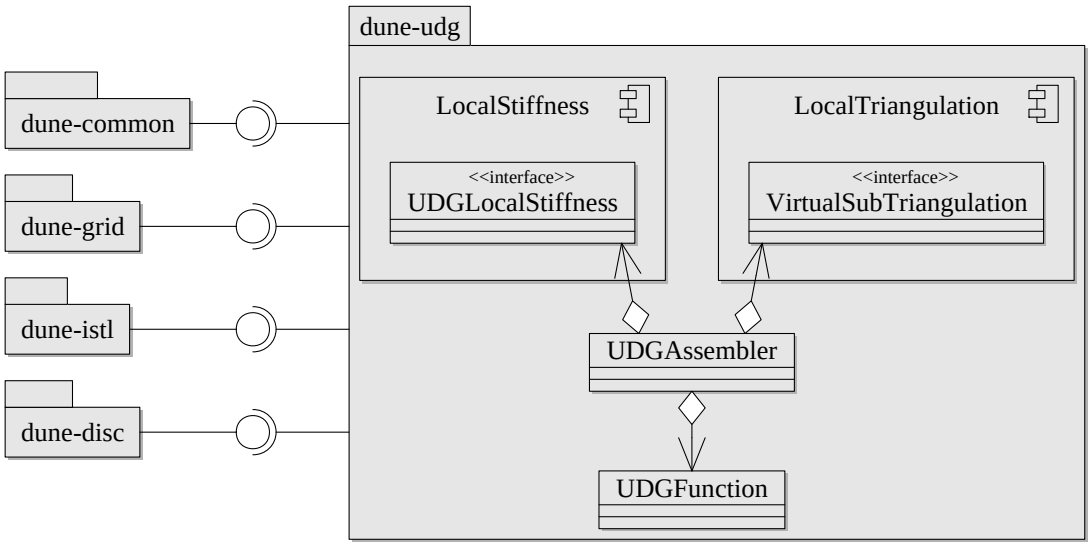


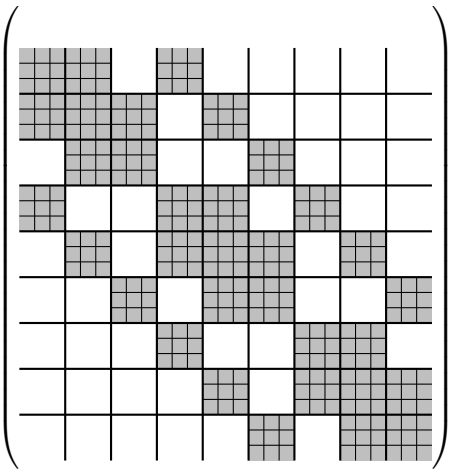
Figure 4.1: Design of the dune-udg module.

assembles the global stiffness matrix into a matrix object of type `Dune::BCRSMatrix`. The triangulation of the fundamental mesh is given as an instance of `Dune::GridView`. For fundamental element E_n the assembler obtains a local triangulation of the element $E_n^{(i)}$ and of the surface $\partial E_n^{(i)}$. The local triangulation is provided by one of the implementations of the `Dune::VirtualSubTriangulation` interface (Figure 4.3).

The assembler computes the local stiffness matrix for a single element using local triangulation. For a single sub-element the bilinear form and the right-hand side are provided by an implementation of the `Dune::UDGLocalStiffness` interface.

The assembled matrix is stored block wise in a compressed row storage format [11] with dense blocks of size n_{dof}^2, where n_{dof} is the number of degrees of freedom associated with one element. In case of a scalar problem this is identical to the number of local basis functions. In case of a vector valued problem, this is the sum of all local scalar basis functions.

Using such a layout allows a memory efficient storage of the matrix, because it reduces the size of the structures storing the data layout. In combination with a block structured iterative linear solver [27], available in `dune-istl`, the data structure improves the cache efficiency and as well as the convergence rates of the iterative solver [104, 14]. The dense blocks are rather small and thus they can easily be inverted directly. Thus the iterative solver only operates on the sparse structure. Especially for Saddle point problems, as we would find them for the incompressible Navier–Stokes equations, the linear solver benefits greatly from a block structure in conjunction with a block-pre-conditioner in contrast to an approaches like the Schur-Complement methods [66, 94].



A

Figure 4.2: Block structure of an assembled matrix. The gray squares ▦ represent dense blocks in the matrix. Each hollow square □ represents one block with only zero entries. This example is for a domain partitioned into 3×3 elements. Each diagonal block corresponds to a grid cell and stores the coupling between the local basis functions. The off-diagonal blocks store the coupling between two adjacent elements.

4.2.2 VirtualSubTriangulation

The `Dune::VirtualSubTriangulation` interface represents the functionality of the local triangulation for a single sub-domain.

The `LocalTriangulation` component (Figure 4.3) offers three different implementations of the `Dune::VirtualSubTriangulation` interface:

`Dune::SubTriangulation2D` Implementation of a two-dimensional sub-domain. The domain boundary is given by a list of primitives, see Section 3.2.2.

`Dune::MarchingCubeSubTriangulation` A sub-domain in two or three dimensions is given by a scalar function. This class implements the local triangulation of implicitly described geometries described in Section 3.2.3.

`Dune::NoSubTriangulation` This implementation assumes $\Omega^{(0)} = \Omega$ and does not perform any local triangulation. It is used for debugging.

All implementations are parameterized by a fundamental mesh and a geometry description. The fundamental mesh is given as a `Dune::GridView` object. Description of the geometry does vary between the different implementations.

The evaluation of volume and surface integrals is implemented using the methods `create_geometries`, `create_edges` and `create_boundaries`, which generate sets of sub-elements.

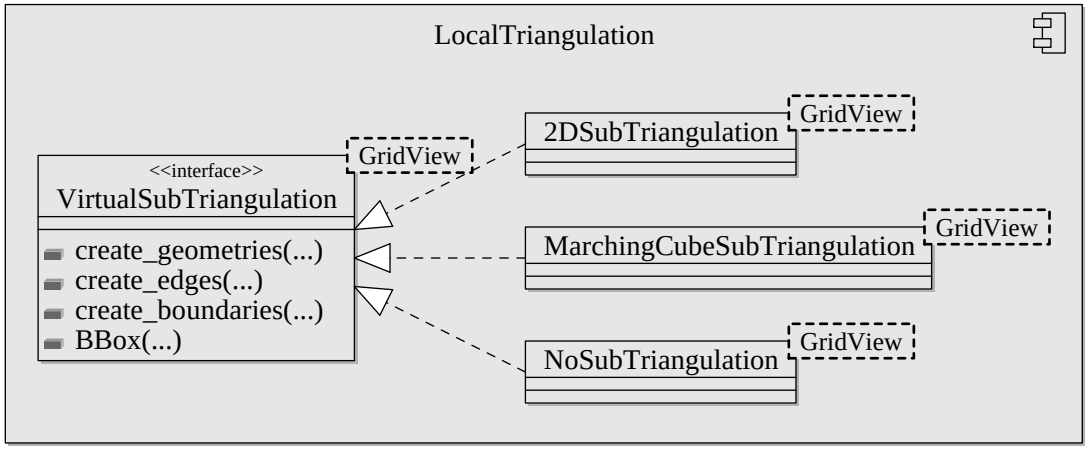


Figure 4.3: The `LocalTriangulation` component.

Member	**Description**
`EntityPointer`	type of the codimension 0 EntityPointer on the fundamental mesh
`SubEntity`	type for a codimension 0 sub-element $E_{n,k}^{(i)}$
`SubIntersection`	type for a codimension 1 sub-element
`GridCuboid`	type for the bounding box of $E_n^{(i)}$
`create_geometries(EntityPointer, std::list<SubEntity>)`	calculate the list of codimension 0 sub-elements of $E_n^{(i)}$
`create_edges(EntityPointer, std::list<SubIntersection>)`	calculate the list of codimension 1 sub-elements of $E_n^{(i)}$, interfacing to an other cell $E_m^{(i)}$
`create_boundaries(EntityPointer, std::list<SubIntersection>)`	calculate the list of codimension 1 sub-elements of $E_n^{(i)}$, lying on the boundary
`GridCuboid BBox(EntityPointer)`	calculate bounding box for a given entity; required for the scaling of local basis functions

Table 4.2: Methods of the `Dune::VirtualSubTriangulation` interface.

Additionally `Dune::VirtualSubTriangulation` exposes a method `BBox` to compute the bounding box of an element. During evaluation of the local basis functions the bounding box is necessary for a scaling, see Remark 3.6.

4.2.3 LocalStiffness

The `LocalStiffness` component provides implementations of different Discontinuous Galerkin discretizations.

`Dune::UDGLocalStiffness` is the interface encapsulating the bilinear form and the right-hand side. The terms in the discretization are sorted into volume, intersection and boundary integrals:

$$\begin{aligned} a(u,v) = &\sum_{E_e^{(0)} \in \mathcal{T}^{(0)}} \int_{E_e^{(0)}} a_{\text{vol}}(u,v)\,dx \\ &+ \sum_{\gamma_{ef} \in \Gamma_{\text{int}}^{(0)}} \int_{\gamma_{ef}} a_{\text{int}}(u,v)\,ds + \sum_{\gamma_e \in \Gamma_{\text{ext}}} \int_{\gamma_e} a_{\text{ext}}(u,v)\,ds \end{aligned} \tag{4.1a}$$

$$\begin{aligned} b(v) = &\sum_{E_e^{(0)} \in \mathcal{T}^{(0)}} \int_{E_e^{(0)}} b_{\text{vol}}(v)\,dx \\ &+ \sum_{\gamma_{ef} \in \Gamma_{\text{int}}^{(0)}} \int_{\gamma_{ef}} b_{\text{int}}(v)\,ds + \sum_{\gamma_e \in \Gamma_{\text{ext}}} \int_{\gamma_e} b_{\text{ext}}(v)\,ds \end{aligned} \tag{4.1b}$$

An implementation of `Dune::UDGLocalStiffness` must provide the methods `assembleVolumeTerm`, `assembleFaceTerm` and `assembleBoundaryTerm` to evaluate these integrals on a sub-element obtained from `Dune::VirtualSubTriangulation`. The types `BoundaryParameters` and `ElementParameters` are implementation specific classes to transport additional information, like boundary conditions, to the assembling stage.

The `LocalStiffness` component implements Discontinuous Galerkin discretizations of several different partial differential equations. Discretizations for four different problems are available, parts of them were implemented within other projects:

`Dune::DGLaplace` Discretization of the stationary heat equation

$$-\nabla(K\nabla u) = f$$

in arbitrary space dimensions. The discretization uses the formulation of [17], see Section 5.1.

`Dune::DGNavierStokes` Discretization of the incompressible stationary Navier–Stokes equations

$$\begin{aligned} -\mu\Delta\mathbf{v} - \rho\mathbf{v}\cdot\nabla\mathbf{v} + \nabla p &= \mathbf{f}\,, \\ \nabla\cdot\mathbf{v} &= 0\,, \end{aligned}$$

Member	Description
`BoundaryParameters`	class providing boundary specific information, e. g. boundary conditions or flux **j** for Neumann boundaries
`ElementParameters`	class providing element specific information, e. g. source terms
`assembleVolumeTerm (ElementParameters, EntityPointer, int, Geometry, LocalGeometry, LocalMatrixBlock, LocalVectorBlock)`	evaluate volume integrals on LocalGeometry and assemble into LocalMatrixBlock and LocalVectorBlock of the element
`assembleFaceTerm (BoundaryParameters, int, Geometry, int, Geometry, SubIntersection, ctype, LocalMatrixBlock, LocalMatrixBlock, LocalMatrixBlock, LocalVectorBlock)`	evaluate intersection integral on SubIntersection and assemble into LocalMatrixBlock and LocalVectorBlock of the adjacent elements
`assembleBoundaryTerm (BoundaryParameters, int, Geometry, SubIntersection, ctype, LocalMatrixBlock, LocalVectorBlock)`	evaluate boundary integral on SubIntersection and assemble into LocalMatrixBlock and LocalVectorBlock of the element
`ElementParameters* createElementParameters (EntityPointer)`	retrieve BoundaryParameters for a given entity of the fundamental mesh
`BoundaryParameters* createBoundaryParameters (SubIntersection)`	retrieve BoundaryParameters for a given SubIntersection

Table 4.3: Methods of the `Dune::UDGLocalStiffness` interface.

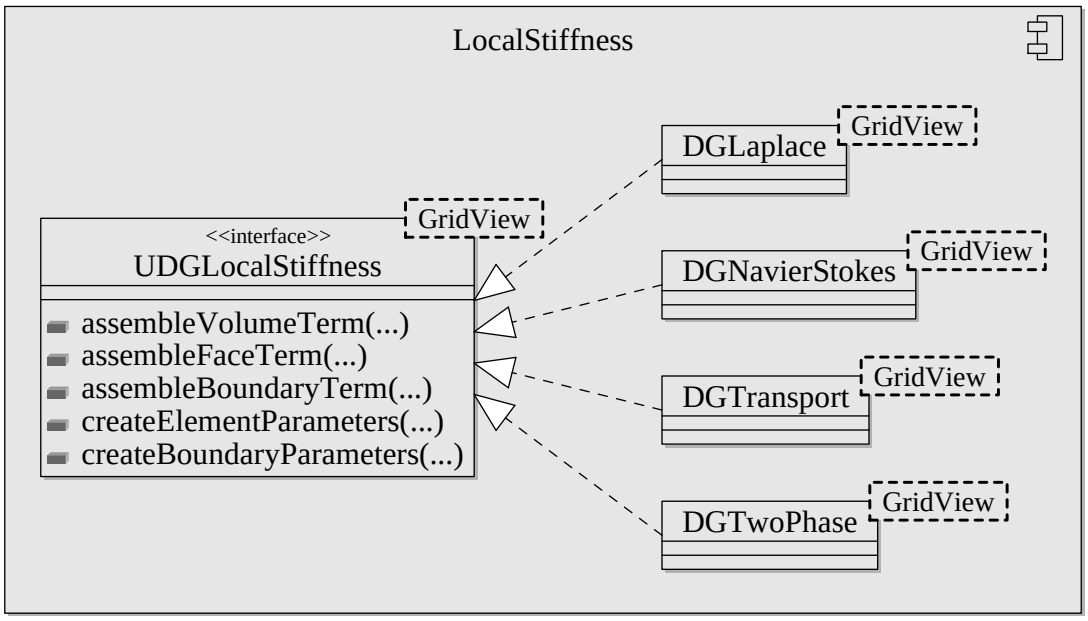


Figure 4.4: The `LocalStiffness` component.

as described in [88]. See [67] and Section 6.1.1.

`Dune::DGTransport` Discretization of the transport equation

$$R\partial_t c + \nabla \cdot \mathbf{u}c - \nabla D(\mathbf{u})\nabla c = q(c)\,.$$

The DG formulation of [13] is used, see Section 7.1 and [50].

`Dune::TwoPhase` Discretization of two immiscible fluids. Interface evolves in time, fluids dynamics are covered by Navier-Stokes equations, surface forces arise from mean curvature. The DG formulation of [88] is used. For details see [61].

Implementations for other DG schemes can easily be added in their primal formulation.

Chapter 5

Stability and Convergence

It is common sense to take a method and try it. If it fails, admit it frankly and try another. But above all, try something.

— Franklin D. Roosevelt

For the *Unfitted Discontinuous Galerkin* method some assumptions with respect to mesh properties of the normal Discontinuous Galerkin method are not guaranteed. In this chapter we want to investigate stability and convergence of the scheme. Even though the UDG method lacks certain guarantees the properties of the classic Discontinuous Galerkin method will be observed.

The test problem for the numerical experiments presented in this section will be the stationary heat equation, a generic elliptic model problem, similar to that defined in Equation (2.2).

We only consider a single complex shaped sub-domain $\Omega^{(0)} \subseteq \Omega \subset \mathbb{R}^d$ with Ω being cuboidal. For all computations in this section the heat conductivity is chosen to be $const = 1$.

The model problem reads

$$-\Delta u = f \qquad \text{on} \qquad \Omega^{(0)} \tag{5.1a}$$

with Dirichlet boundary conditions

$$u = g \qquad \text{on} \qquad \Gamma_{\mathrm{D}} \subseteq \partial\Omega^{(0)} \tag{5.1b}$$

and Neumann boundary conditions

$$\mathbf{j} \cdot \mathbf{n} = F \qquad \text{on} \qquad \Gamma_{\mathrm{N}} = \partial\Omega^{(0)} \setminus \Gamma_{\mathrm{D}}, \tag{5.1c}$$

where K denotes the heat conductivity, $\mathbf{j}$ denotes the flux ∇u, and $\mathbf{n}$ is the normal vector. The right-hand side f is a given function, the boundary conditions are given by the functions g and F.

The discretization of the model problem is based on the generalized bilinear form outlined in Section 2.2.3.

Let $\mathcal{T}(\Omega^{(0)}) = \{E_1^{(0)}, \ldots, E_n^{(0)}\}$ be a non-degenerate quasi-uniform subdivision of $\Omega^{(0)}$, as defined in (2.3a) and (3.4). The numerical fluxes are defined as in Section 2.2.3 Equation (2.45).

The problem to be solved reads: Find $u \in V_k^{(0)}$ such that

$$a_\epsilon(u, v) = l_{\epsilon\eta}(v) \qquad \forall v \in V_k^{(0)}. \tag{5.2}$$

The bilinear form is the one given in Equation (2.46). It is parameterized by ϵ and η. Depending on the parameters we can choose between the non-symmetric Oden-Babuška-Baumann scheme, the NIPG and the symmetric SIPG scheme.

The right hand side of (5.2) is given as the linear form

$$\begin{aligned} l_{\epsilon\eta}(v) = &\int_\Omega f\ v\ dV + \int_{\Gamma_\mathrm{N}} F\ v\ ds \\ &+ \int_{\Gamma_\mathrm{D}} \epsilon\, \nabla v \cdot \mathbf{n}\ g\ ds + \eta \int_{\Gamma_\mathrm{D}} h_\gamma^{-1} v\ g\ ds. \end{aligned} \tag{5.3}$$

5.1 Interpolation Error

Given a domain Ω, a function $u \in H^{k+1}(\Omega)$, and an interpolation operator

$$I : H^{k+1} \to \mathcal{P}_k\,, \tag{5.4}$$

which maps u onto the space of piecewise polynomials $\mathcal{P}_k$ of order k. The H^m norm of the approximation error $u - Iu$ of the interpolated function should be estimated. The approximation error $\|u - Iu\|_{H^m}$ on a single element is transformed onto the reference element and bound from above using the Bramble–Hilbert lemma [31]. The transformation back into the global space yields the desired estimate. In the optimal case this error measured in L_2 or H^1 shows the order of convergence

$$\|u - Iu\|_{L_2} \ \leq\ ch^{k+1}\, |u|_{H^{k+1}} \tag{5.5}$$

$$\|u - Iu\|_{H^1} \ \leq\ ch^{k}\, |u|_{H^{k+1}} \tag{5.6}$$

$$\tag{5.7}$$

These estimates require that the domain satisfies the strong cone property, as defined in [31] and [110, p. 45].

In general the elements $E_n^{(i)}$ might not fulfill this cone property. E.g. consider the sub-domain

$$\Omega^{(i)} = (0, 2)^2 \cap \left\{(x, y)\,|\, y < x^2\right\}. \tag{5.8}$$

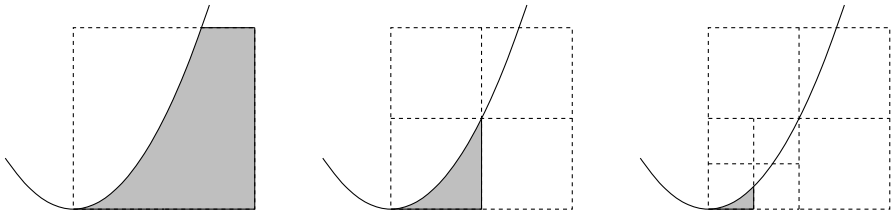


Figure 5.1: Refinement of cusp elements results in anisotropic elements, which do not fulfill the cone property.

In point $(0, 0)$ the edges meet so that the tangents of both edges in this point are equal (see Figure 5.1). Such a point is called *a cusp* and poses particular problems. Not only the cone condition is violated, the element also becomes anisotropic when refining the grid. To our knowledge there exist no estimates of the interpolation error in such a case. Note that this problem can occur only when using quadratic or higher-order transformations. For multi-linear transformations the tangents cannot become parallel.

Estimates for anisotropic linear triangles have been obtained by Babuška and Aziz [7] and they have shown that anisotropy does not pose further problems as long as the largest angle in every element is bounded away from π. In [45], it is shown that full convergence can be obtained for star shaped elements. However, none of these papers applies in this case of anisotropic elements with quadratic transformation.

As an analytic study was not feasible, we took another approach and studied the interpolation error measured in L_2- and H^1-norms for a single cusp element in $\mathbb{R}^2$ and computed the experimental order of convergence.

Definition 5.1 (Experimental Order of Convergence): *Given a function u and an approximation u_h, depending on the mesh size h. For a given norm $\|\cdot\|$, the* Experimental Order of Convergence *(EOC) is given as*

$$EOC_h = \frac{\log(\|u - u_h\|/\|u - u_{\frac{h}{2}}\|)}{\log(2)}. \tag{5.9}$$

To reduce numerical inaccuracies these calculations are done using MAPLE [75], using an accuracy of 30 digits. The function $u = e^{-(x^2+y^2)}$ was chosen as function with full regularity. Using Lagrange interpolation the error measured in both L_2- and in H^1-norm shows optimal convergence rate (see Figure 5.2). When using L_2 projection values close to the cusp have only little influence on the projection. If the error is measured in L_2-norm this is not visible, as errors close to the cusp also give only a small contribution to the total error. In the H^1-norm this fact is visible and we loose one order in the error convergence.

In Section 5.3.2 we will present numerical results supporting the claim that the order of convergence for the DG scheme is independent of the shape of the elements $E_n^{(i)}$.

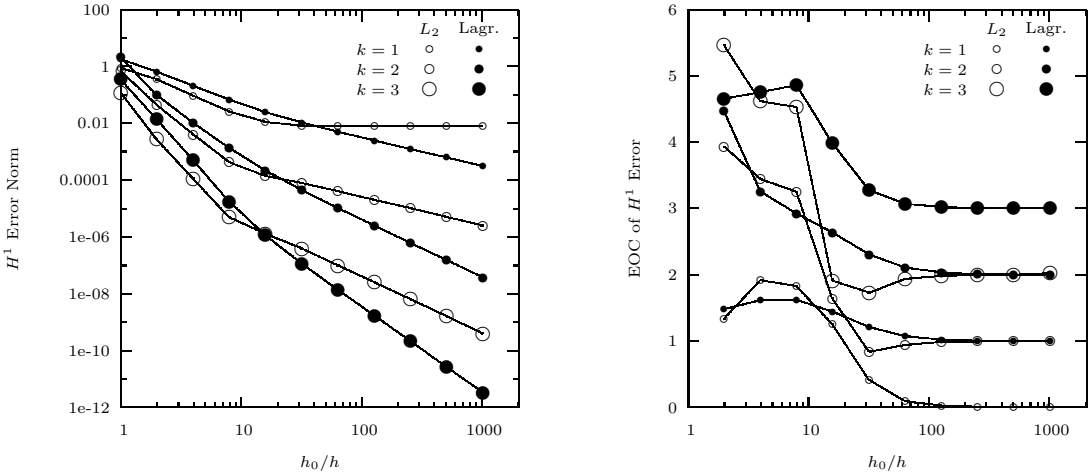


Figure 5.2: H^1-error and its convergence for L_2-projection and Lagrange interpolation on cusp elements.

5.2 Condition Number of the Matrix

The condition of a matrix A gives a measure for the computational error depending on the input error.

When solving $Ax = b$, an error e in b, leads to the error $A^{-1}e$ in the solution $x = A^{-1}b$. The ratio of the relative error $\|A^{-1}e\|/\|x\| = \|A^{-1}e\|/\|A^{-1}b\|$ of the solution to the relative error $\|e\|/\|b\|$ of b can be bound from above using Cauchy–Schwarz inequality

$$\frac{\|A^{-1}e\|/\|A^{-1}b\|}{\|e\|/\|b\|} \leq \|A^{-1}\| \cdot \|A\|, \tag{5.10}$$

where matrix norm $\|A\|$ must be the one associated with the vector norm $\|b\|$.

Definition 5.2 (Condition number): *The condition number κ of a non-singular matrix A is an upper bound for the error propagation [36] with respect to inversion. It is defined by*

$$\kappa(A) = \|A^{-1}\| \cdot \|A\|. \tag{5.11}$$

Common choices for the norm are the L^∞ vector norm with the maximum row sum matrix norm or the L^2 norm with the spectral norm, when the matrix is symmetric and positive definite.

The condition number is also an important property for numerical applications, as the convergence of many iterative linear solvers strongly depends on the condition number. For matrices with high condition number the number of iterations will increase.

As stated in Remark 3.6, the shape functions are defined on the reference element E_n of the fundamental mesh but are scaled according to the bounding box of $E_n^{(i)}$.

In the triangulation $\mathcal{T}(\Omega^{(i)})$, the ratio H/h between two adjacent elements is not bound, H and h are proportional to the size $\text{diam}(E_n^{(i)})$ of a small and a big element.

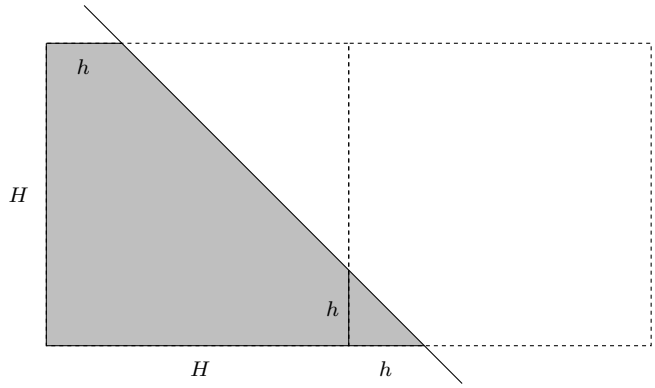


Figure 5.3: Domain Ω consisting of two elements. Intersecting the domain with a plane creates a small and a big element. In 2D the small element has the shape of a triangle, in 3D that of a tetrahedron.

For a domain with two elements, the condition number associated with the maximum row sum norm was computed using MAPLE. The domain consists of two squares in 2D, respectively two cubes in 3D. It is cut by a plane, or line in two space dimensions, so that the small element is a simplex. For two dimensions Figure 5.3 illustrates the construction. The ratio H/h can be changed by changing the size of h, which means moving the cutting plane in normal direction.

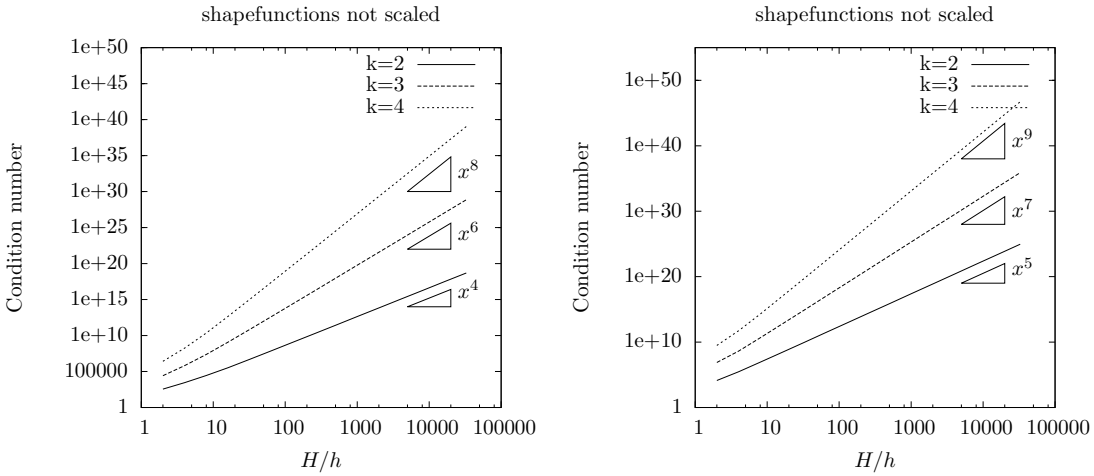


Figure 5.4: Condition number of the stiffness matrix for a domain with two elements and edge length H and h. The local basis functions are not scaled. Results for 2D (left), 3D (right).

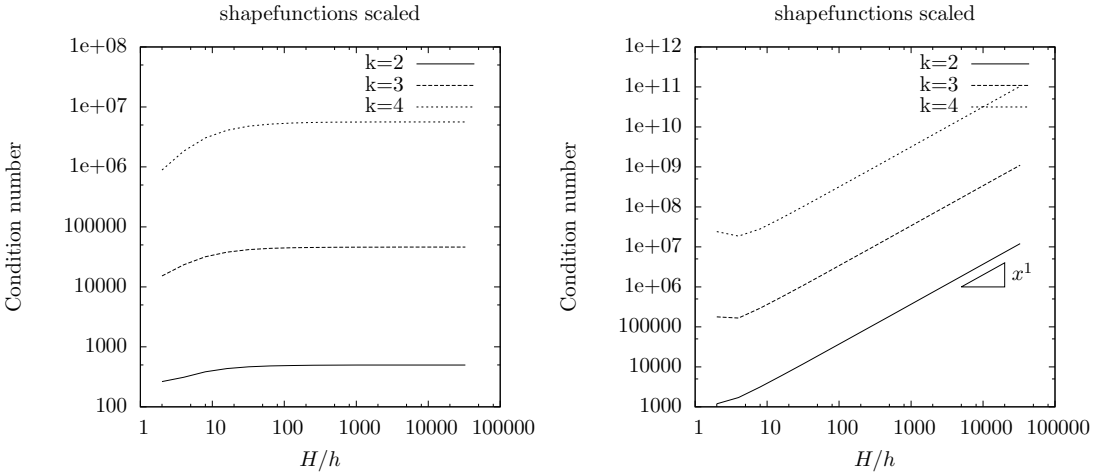


Figure 5.5: Condition number of the stiffness matrix for a domain $\Omega^0 \subset \mathbb{R}^3$, with two elements and edge length H and h. The local basis functions are scaled according to the bounding box. Results for 2D (left), 3D (right).

Computations were carried out in two and three space dimensions, both with scaled local basis functions and with local basis functions not scaled according to the bounding box. The stiffness matrix is assembled for the OBB ($\epsilon = 1, \eta = 0$) discretization of the model problem (2.2), with Neumann boundary conditions on the internal face and Dirichlet boundary everywhere else. The local basis functions are (scaled) monomial functions.

For uniform grids, the condition number is constant in two dimensions and it scales with $1/h$ in three dimensions. In this example, the grids are strongly not uniform and the ratio H/h goes up to 32768. If the local basis functions are not scaled, the log-log plots in Figure 5.4 show that the condition number is not bound and is growing with $(H/h)^{2k}$ in 2D and $(H/h)^{2k+1}$ in 3D.

Figure 5.5 shows the condition number for a discretization with local basis functions which are scaled according to the bounding box. In the 2D example the condition number is growing a at the beginning but quickly reaches its limit and becomes constant. In three space dimensions the condition number is growing linearly regardless of the polynomial degree of the ansatz functions.

The condition number doesn't pose any problems for the two-dimensional case. In three space dimensions the condition number increases with h^{-1}. In practice this will usually pose no problems. When using some Krylov method to solve the linear system and the matrix has only few very small eigenvalues, the convergence is still fast. This is the case if there are only a small number of elements where χ is large. If there is a significant number of very small eigenvalues local refinement can help to limit χ. The current implementation does not feature local refinement.

5.3 Discretization Error in L_2 and H^1 Norm

Error analysis on all three considered methods (OBB, NIPG and SIPG, see 2.2.3) exists. A priori error estimates for the SIPG method are presented in [109]. In [89, 90] the two non-symmetric schemes, OBB and NIPG, are analyzed. The SIPG scheme converges optimally in L_2 and H^1 norm, read $\|u - u_h\|_{L_2} \propto h^{k+1}$ and $\|u - u_h\|_{H^1} \propto h^k$, where k denotes the polynomial degree of the ansatz functions. The NIPG and OBB scheme show optimal convergence in the H^1-norm, but in the L_2-norm one order is lost for k even, for k odd you obtain optimal convergence.

Remark 5.1: Using over-penalization the NIPG scheme can be modified to show optimal convergence rates also in the L_2-norm. This requires the penalty term to be scaled with h^{-2k-1} instead of h^{-1} (see [5]). As this over-penalization increases the condition of the matrix significantly we use scaling with h^{-1}.

It is not possible to directly transfer these proofs to the UDG method because the mesh does not guarantee certain assumptions of the standard Discontinuous Galerkin method, in particular shape regularity and quasi uniformity. In this section we will numerically verify the convergence of the elliptic model problem on an UDG mesh.

Definition 5.1 of *Experimental Order of Convergence* requires the knowledge of the exact solution u, but in general u is not accessible. For some of the following examples no exact solution can be given. For these cases Definition 5.1 must be extended

Definition 5.3 (Experimental Order of Convergence (cont.)): *Given a setup where the exact solution u is not accessible.*

Using the saturation assumption

$$\|u - u_{\frac{h}{2}}\| < \|u - u_h\| \tag{5.12}$$

the exact solution u is approximated by $u_{\tilde{h}}$, a solution computed on a fine mesh with mesh width $\tilde{h}$. The Experimental Order of Convergence *now reads*

$$EOC_{h,\tilde{h}} \quad = \quad \frac{\log(\|u_{\tilde{h}} - u_h\| / \|u_{\tilde{h}} - u_{\frac{h}{2}}\|)}{\log(2)}, \tag{5.13}$$

with $\tilde{h} \ll h$.

5.3.1 Behavior on Non Quasi Uniform Grids

We consider two adjacent elements in a finite element mesh. As in Section 5.2 we consider a triangulation where the size of two adjacent elements in the worst case is $\text{diam}(E^{(i)}_{\text{small}}) = h$ and $\text{diam}(E^{(i)}_{\text{big}}) = H$.

On a normal finite element mesh the ratio $\chi = H/h$ of two adjacent elements is bound. This is one of the constraints for the mesh generator and is usually bound to be $\chi \leq 2$.

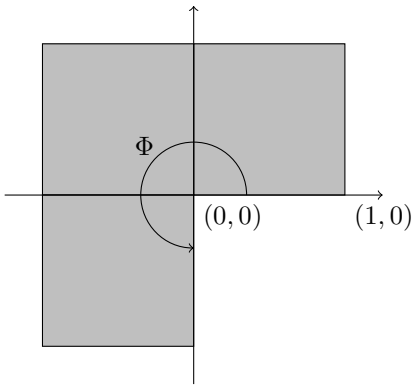


Figure 5.6: Domain with re-entrant corner.

In the UDG triangulation $\mathcal{T}(\Omega^{(i)})$, the ratio χ is not bound any more.

We consider the model problem stated in Equation (5.1) on a domain with a re-entrant corner, Figure 5.6. Boundary conditions are given in polar coordinates with the relation

$$x = r \cdot \cos\theta, \qquad y = r \cdot \sin\theta\,.$$

Only Dirichlet boundary conditions are considered. The boundary values are given by

$$\begin{aligned} u &= \sin\left(\frac{\pi}{\Theta}\theta(x,y)\right) && \text{on} \quad \Gamma_1 = \Gamma \cap (-1,1)^2\,, \\ u &= 0 && \text{on} \quad \Gamma \setminus \Gamma_1\,. \end{aligned}$$

Θ denotes the angle of the re-entrant corner, in this example it is $\Theta = \frac{3}{2}\pi$.

$$u = r^k \sin(k\theta(x,y)) \tag{5.14}$$

is a general solution, choosing $k = \pi/\Theta = 2/3$ fulfills the boundary values.

Computations are carried out for coarse grids with different levels of hanging nodes. The upper left quadrant is locally refined towards the singularity, while all other quadrants are not refined (see Figure 5.7). The resulting values for χ range from 1 to 64. The singularity is not in the ansatz space, thus the error is high close to $(0.5, 0.5)$. These coarse grids are globally refined and the error is measured in the L_2-norm, the refinement does not change χ. In Figure 5.8 results are shown for $k = 2$ and $k = 5$ using the OBB scheme.

The error on the coarse mesh is dominated by the two big refined elements (lower left, upper right). It is independent of the minimal h but depends on $h \cdot \chi$ and stays more or less the same, without respect to the local refinement of the upper left element. The error on the refined grids converges with the expected order of $h^{\frac{4}{3}}$, Table 5.1 shows the EOC after five refinement steps for all values of χ.

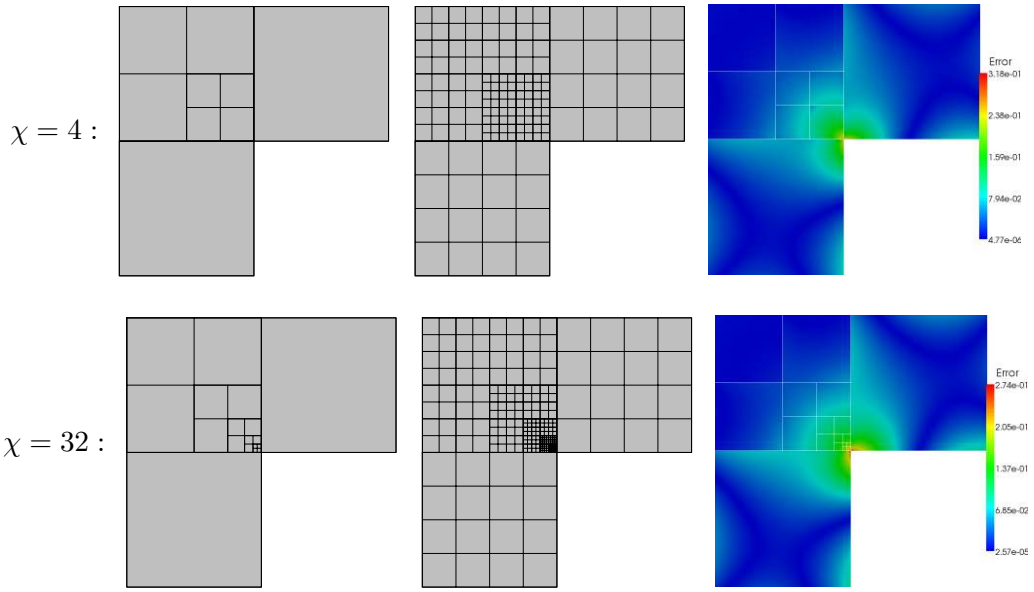


Figure 5.7: Strongly non-uniform grids. Starting from a grid with different levels of hanging nodes (left), simulations are done on globally refined grids (middle). Errors for the coarse grid with ansatz functions $k = 2$ are shown (right).

χ	$k = 2$	$k = 5$
1	1.30	1.37
2	1.27	1.37
4	1.26	1.37
8	1.25	1.37
16	1.24	1.37
32	1.24	1.37
64	1.21	1.37

Table 5.1: Convergence rates of the L_2-error for different ratios χ after 5 levels of refinement.

Due to the low regularity of the problem, the computations do not benefit from the high order basis functions with $k = 5$, see Figure 5.8. The convergence rates do not show any considerable difference related to growing χ. The discretization does not benefit from local refinement if χ is big but it does not harm the computations.

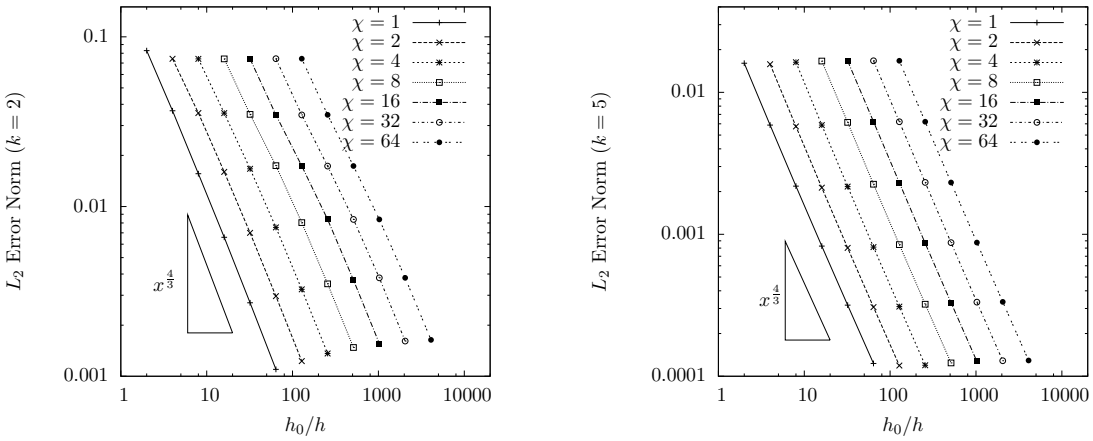


Figure 5.8: Convergence rates of the L_2-error for setups with different ratios χ

5.3.2 Convergence with Degenerated Elements

As described in Section 5.1, the usual estimates for the convergence rates do not hold for cusp elements due to violation of the cone condition. This section will show numerical results which support our claim that within this scheme that kind of degenerated elements have no negative impact on the convergence rate of the discretization error.

This example treats a test problem with full regularity. Equation (5.1) is solved on the unit square with a parabola shaped sub-domain cut out (see Figure 5.9). The domain $\Omega^{(0)}$ doesn't pose any problems, still the construction of the UDG finite element mesh leads degenerated elements as they were considered in Section 5.1.

The exact solution is

$$u_{\text{exact}}(x) = e^{(-\|x-x_0\|^2)} \qquad \text{with} \qquad x_0 = (0.5, 0.5). \tag{5.15}$$

f, g and F are chosen in accordance to the exact solution:

$$\begin{aligned} f &= 2\, e^{-\|x-x_0\|^2} \cdot (2 - \|x-x_0\|^2), \\ g &= e^{(-\|x-x_0\|^2)}, \qquad \text{and} \\ F &= -2\, e^{(-\|x-x_0\|^2)} \cdot (x - x_0). \end{aligned}$$

Two different sets of boundary conditions are considered:

1. Pure Dirichlet boundary conditions

$$\Gamma_{\text{D}} = \partial\Omega^{(0)} = \Gamma_0 + \Gamma_l + \Gamma_r + \Gamma_b + \Gamma_t, \tag{5.16}$$

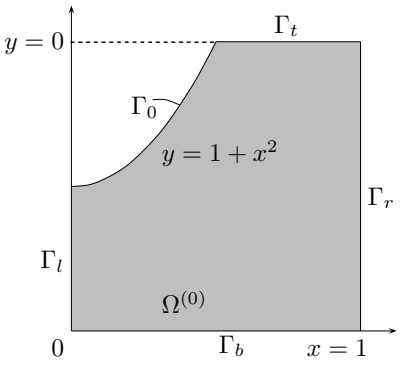


Figure 5.9: Sub-domain $\Omega^{(0)}$ with a parabola-shaped cut out on the unit square.

2. Neumann boundary conditions on the curved and on the left boundary and Dirichlet boundary conditions on the right boundary

$$\Gamma_{\mathrm{N}} = \Gamma_0 + \Gamma_l \qquad \text{and} \qquad \Gamma_{\mathrm{D}} = \Gamma_r + \Gamma_b + \Gamma_t. \tag{5.17}$$

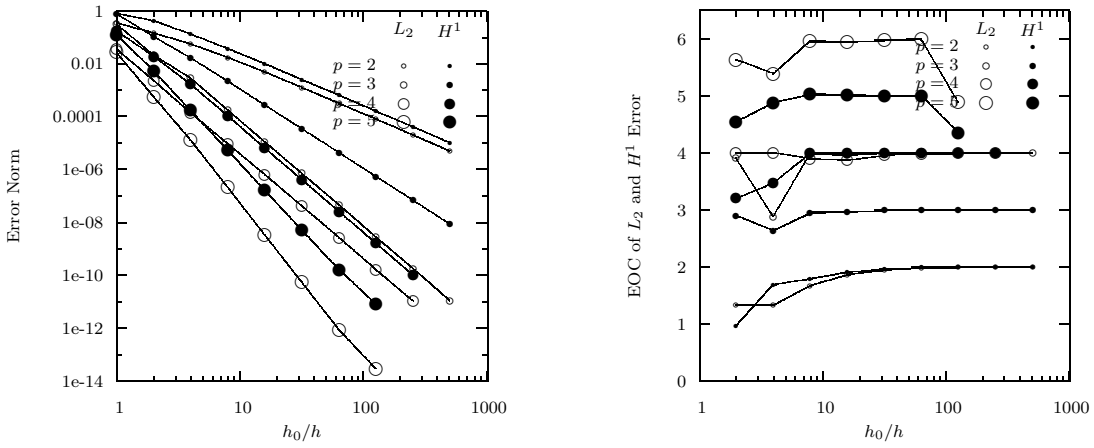


Figure 5.10: Discretization error measured in L_2- and H^1-norm (left) and their convergence for $h \to 0$ (right) for the UDG discretization of the model problem on a domain with a cusp. The plots show the errors for a discretization of $-\nabla(K\nabla p) = f$ with the exact solution $e^{-(x-x_0)^2}$ and Dirichlet boundary conditions using the OBB scheme. Similar results are obtained for Neumann boundary conditions and for the NIPG and SIPG scheme.

The local triangulation of the elements $E_n^{(0)}$ is obtained using the algorithm described in [49]. Iso-parametric elements with second order transformations allow the shape of $\Omega^{(0)}$ to be resolved exactly.

Figure 5.10 shows the L_2- and H^1-error and their convergence for $h \to 0$ for Dirichlet boundary conditions. The graph on the right shows the experimental order of conver-

gence (EOC). The calculations are done for trial functions of polynomial degrees 2–5, with the scheme introduced by Oden, Babuška and Baumann in [79] ($\epsilon = 1, \eta = 0$). Although the cone condition is not fulfilled in the sub-domain $\Omega^{(0)}$ (Figure 5.9), an optimal h-convergence rate in the H^1-norm is obtained. The h-convergence in the L_2-norm also exhibits the predicted behavior $O(h^{p+1})$ for p odd and $O(h^p)$ for p even.

As visible in Tables 5.2 and 5.3 the symmetric ($\epsilon = -1$, $\sigma > 0$) and the non-symmetric ($\epsilon = 1$, $\sigma > 0$) Interior Penalty method also show optimal convergence rates. As the schemes are all stable and give comparable results, we will use the OBB scheme in all following computations.

5.3.3 Convergence for a Problem with No Analytic Solution

The next example investigates a setup where no analytic solution can be given. We consider the stationary heat equation on the unit square. Dirichlet boundary conditions are imposed on the left and right hand boundary and Neumann boundary conditions on the top and bottom boundary. f and g are chosen as

$$f = 0 \qquad \text{and} \qquad g = 1 - x.$$

The result is the stationary temperature distribution in a plate which is heated on the left and cooled on the right boundary. Given a hole in the plate (Figure 5.11), $\nabla u \cdot \mathbf{n}$ on the boundary of the hole, we are not aware of an analytic solution.

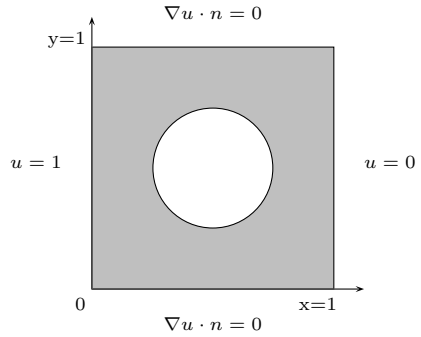


Figure 5.11: Domain with a circular hole. Dirichlet boundary conditions on the left and the right boundary, Neumann boundary conditions on the circle. We are not aware of an analytic solution.

As in the previous simulation, the local triangulation described in Section 3.2.2 is used. Again computations are carried out for trial and test functions of polynomial degrees 2–5 and for a discretization uses the OBB scheme ($\epsilon = 1, \eta = 0$).

Once the grid is sufficiently fine to give a good approximation of the geometry, the predicted h-convergence rate in the L_2- and the H^1-norm is observed, see Figure 5.12. For very coarse grids the convergence rate is lower, but extrapolating the error on fine grids back to $h = L$, one would still expect an error which is bigger than the one obtained in these computations.

L/h	SIPG		NIPG		OBB	
	Err_{L_2}	EOC_{L_2}	Err_{L_2}	EOC_{L_2}	Err_{L_2}	EOC_{L_2}
			$k=2$			
1	1.540×10^{-1}	—	2.013×10^{-1}	—	3.564×10^{-1}	—
2	4.626×10^{-2}	1.73	7.476×10^{-2}	1.43	1.410×10^{-1}	1.34
4	1.006×10^{-2}	2.20	1.786×10^{-2}	2.07	5.634×10^{-2}	1.32
8	1.027×10^{-3}	3.29	4.333×10^{-3}	2.04	1.768×10^{-2}	1.67
16	1.069×10^{-4}	3.26	1.064×10^{-3}	2.03	4.856×10^{-3}	1.86
32	1.100×10^{-5}	3.28	2.644×10^{-4}	2.01	1.259×10^{-3}	1.95
64	1.270×10^{-6}	3.11	6.598×10^{-5}	2.00	3.195×10^{-4}	1.98
128	1.536×10^{-7}	3.05	1.648×10^{-5}	2.00	8.032×10^{-5}	1.99
256	1.893×10^{-8}	3.02	4.119×10^{-6}	2.00	2.013×10^{-5}	2.00
512	2.380×10^{-9}	2.99	1.029×10^{-6}	2.00	5.038×10^{-6}	2.00
			$k=3$			
1	1.302×10^{-1}	—	1.799×10^{-1}	—	3.099×10^{-1}	—
2	9.933×10^{-3}	3.71	1.044×10^{-2}	4.11	2.061×10^{-2}	3.91
4	7.151×10^{-4}	3.80	1.164×10^{-3}	3.17	2.791×10^{-3}	2.88
8	4.011×10^{-5}	4.16	6.720×10^{-5}	4.11	1.769×10^{-4}	3.98
16	2.306×10^{-6}	4.12	4.105×10^{-6}	4.03	1.135×10^{-5}	3.96
32	1.387×10^{-7}	4.06	2.542×10^{-7}	4.01	7.148×10^{-7}	3.99
64	8.539×10^{-9}	4.02	1.579×10^{-8}	4.01	4.473×10^{-8}	4.00
128	5.307×10^{-10}	4.01	9.850×10^{-10}	4.00	2.796×10^{-9}	4.00
256	3.310×10^{-11}	4.00	6.149×10^{-11}	4.00	1.747×10^{-10}	4.00
512	2.125×10^{-12}	3.96	3.848×10^{-12}	4.00	1.092×10^{-11}	4.00
			$k=4$			
1	2.042×10^{-2}	—	2.333×10^{-2}	—	3.568×10^{-2}	—
2	1.702×10^{-3}	3.58	1.985×10^{-3}	3.56	2.239×10^{-3}	3.99
4	6.494×10^{-5}	4.71	9.819×10^{-5}	4.34	1.398×10^{-4}	4.00
8	2.053×10^{-6}	4.98	6.011×10^{-6}	4.03	9.350×10^{-6}	3.90
16	6.374×10^{-8}	5.01	3.924×10^{-7}	3.94	6.365×10^{-7}	3.88
32	1.984×10^{-9}	5.01	2.500×10^{-8}	3.97	4.105×10^{-8}	3.95
64	6.270×10^{-11}	4.98	1.576×10^{-9}	3.99	2.597×10^{-9}	3.98
			$k=5$			
1	1.883×10^{-2}	—	2.065×10^{-2}	—	2.764×10^{-2}	—
2	3.807×10^{-4}	5.63	4.521×10^{-4}	5.51	5.541×10^{-4}	5.64
4	6.707×10^{-6}	5.83	9.297×10^{-6}	5.60	1.324×10^{-5}	5.39
8	1.060×10^{-7}	5.98	1.452×10^{-7}	6.00	2.119×10^{-7}	5.97
16	1.684×10^{-9}	5.98	2.289×10^{-9}	5.99	3.430×10^{-9}	5.95
32	2.652×10^{-11}	5.99	3.608×10^{-11}	5.99	5.448×10^{-11}	5.98

Table 5.2: Comparison of different schemes for simulations on a cusp domain. L_2-error and EOC for SIPG, NIPG and OBB scheme.

L/h	SIPG		NIPG		OBB	
	Err_{H^1}	EOC_{H^1}	Err_{H^1}	EOC_{H^1}	Err_{H^1}	EOC_{H^1}
			$k=2$			
1	5.502×10^{-1}	—	5.776×10^{-1}	—	8.122×10^{-1}	—
2	3.040×10^{-1}	0.86	3.103×10^{-1}	0.90	4.166×10^{-1}	0.96
4	9.236×10^{-2}	1.72	7.649×10^{-2}	2.02	1.294×10^{-1}	1.69
8	2.221×10^{-2}	2.06	1.957×10^{-2}	1.97	3.732×10^{-2}	1.79
16	6.524×10^{-3}	1.77	4.917×10^{-3}	1.99	9.922×10^{-3}	1.91
32	1.200×10^{-3}	2.44	1.232×10^{-3}	2.00	2.542×10^{-3}	1.96
64	2.910×10^{-4}	2.04	3.083×10^{-4}	2.00	6.412×10^{-4}	1.99
128	7.193×10^{-5}	2.02	7.712×10^{-5}	2.00	1.609×10^{-4}	2.00
256	1.790×10^{-5}	2.01	1.929×10^{-5}	2.00	4.027×10^{-5}	2.00
512	4.466×10^{-6}	2.00	4.822×10^{-6}	2.00	1.007×10^{-5}	2.00
			$k=3$			
1	4.952×10^{-1}	—	5.289×10^{-1}	—	7.757×10^{-1}	—
2	7.952×10^{-2}	2.64	7.807×10^{-2}	2.76	1.042×10^{-1}	2.90
4	1.218×10^{-2}	2.71	1.107×10^{-2}	2.82	1.677×10^{-2}	2.63
8	1.432×10^{-3}	3.09	1.382×10^{-3}	3.00	2.174×10^{-3}	2.95
16	1.708×10^{-4}	3.07	1.716×10^{-4}	3.01	2.784×10^{-4}	2.97
32	2.083×10^{-5}	3.04	2.137×10^{-5}	3.01	3.509×10^{-5}	2.99
64	2.578×10^{-6}	3.01	2.667×10^{-6}	3.00	4.397×10^{-6}	3.00
128	3.210×10^{-7}	3.01	3.331×10^{-7}	3.00	5.501×10^{-7}	3.00
256	4.006×10^{-8}	3.00	4.162×10^{-8}	3.00	6.878×10^{-8}	3.00
512	5.005×10^{-9}	3.00	5.201×10^{-9}	3.00	8.598×10^{-9}	3.00
			$k=4$			
1	1.177×10^{-1}	—	1.315×10^{-1}	—	1.704×10^{-1}	—
2	2.612×10^{-2}	2.17	1.739×10^{-2}	2.92	1.856×10^{-2}	3.20
4	1.356×10^{-3}	4.27	1.351×10^{-3}	3.69	1.674×10^{-3}	3.47
8	8.741×10^{-5}	3.96	8.897×10^{-5}	3.92	1.054×10^{-4}	3.99
16	5.434×10^{-6}	4.01	5.621×10^{-6}	3.98	6.598×10^{-6}	4.00
32	3.391×10^{-7}	4.00	3.527×10^{-7}	3.99	4.128×10^{-7}	4.00
64	2.123×10^{-8}	4.00	2.208×10^{-8}	4.00	2.580×10^{-8}	4.00
			$k=5$			
1	1.026×10^{-1}	—	1.045×10^{-1}	—	1.224×10^{-1}	—
2	4.912×10^{-3}	4.38	4.884×10^{-3}	4.42	5.236×10^{-3}	4.55
4	2.264×10^{-4}	4.44	1.568×10^{-4}	4.96	1.776×10^{-4}	4.88
8	5.336×10^{-6}	5.41	4.921×10^{-6}	4.99	5.441×10^{-6}	5.03
16	1.566×10^{-7}	5.09	1.531×10^{-7}	5.01	1.692×10^{-7}	5.01
32	4.922×10^{-9}	4.99	4.772×10^{-9}	5.00	5.275×10^{-9}	5.00

Table 5.3: Comparison of different schemes for simulations on a cusp domain. H^1-error and EOC for SIPG, NIPG and OBB scheme.

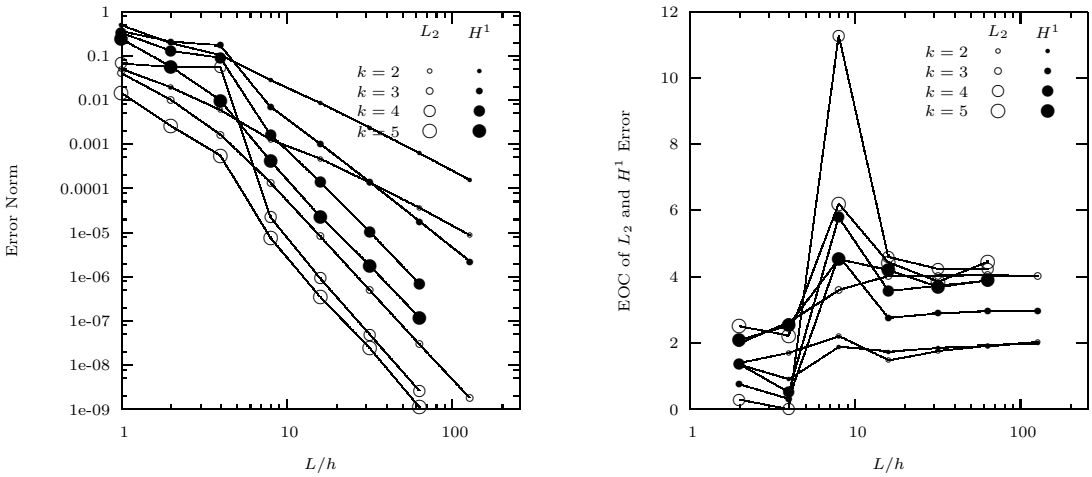


Figure 5.12: Discretization error measured in L_2- and H^1-norm (left) and their convergence for $h \to 0$ (right) for the UDG discretization of the model problem on the domain shown in Figure 5.11. The reference solution is computed for $h = L/256$ and order 5. The plots show the expected convergence rates for a discretization using the OBB scheme.

Remark 5.2: Note that the coarse grid simulations wouldn't be possible using standard finite elements.

5.3.4 Convergence on a Domain with Holes

At last, the convergence on a domain with many holes is investigated. The setup is similar to those used in the following chapter for three-dimensional computations. For the local triangulation, a different algorithm than in the previous computations is used, outlined in Section 3.2.3.

The domain $\Omega^{(0)}$ is implicitly given by a scalar function, as it would be obtained through post processing of image data. Instead of experimental data, an artificial structure is generated, using a sphere-packing algorithm [97]. Figure 5.13 shows the scalar function, the described domain, and a closeup of the resulting local triangulation. Notably the image grid in this example is very fine, such that an appropriate reference solution can be computed.

The boundary conditions are chosen similar to the setup in Section 5.3.3: Dirichlet boundary conditions on the left and on the right boundary, Neumann boundary conditions else.

Figure 5.14 shows the results for different polynomial degrees k and different mesh sizes h.

In order to compare the results, the flux through the inflow boundary

$$j_{\text{inflow}} = \int_{\Gamma_{\text{left}}} \nabla p \, ds \tag{5.18}$$

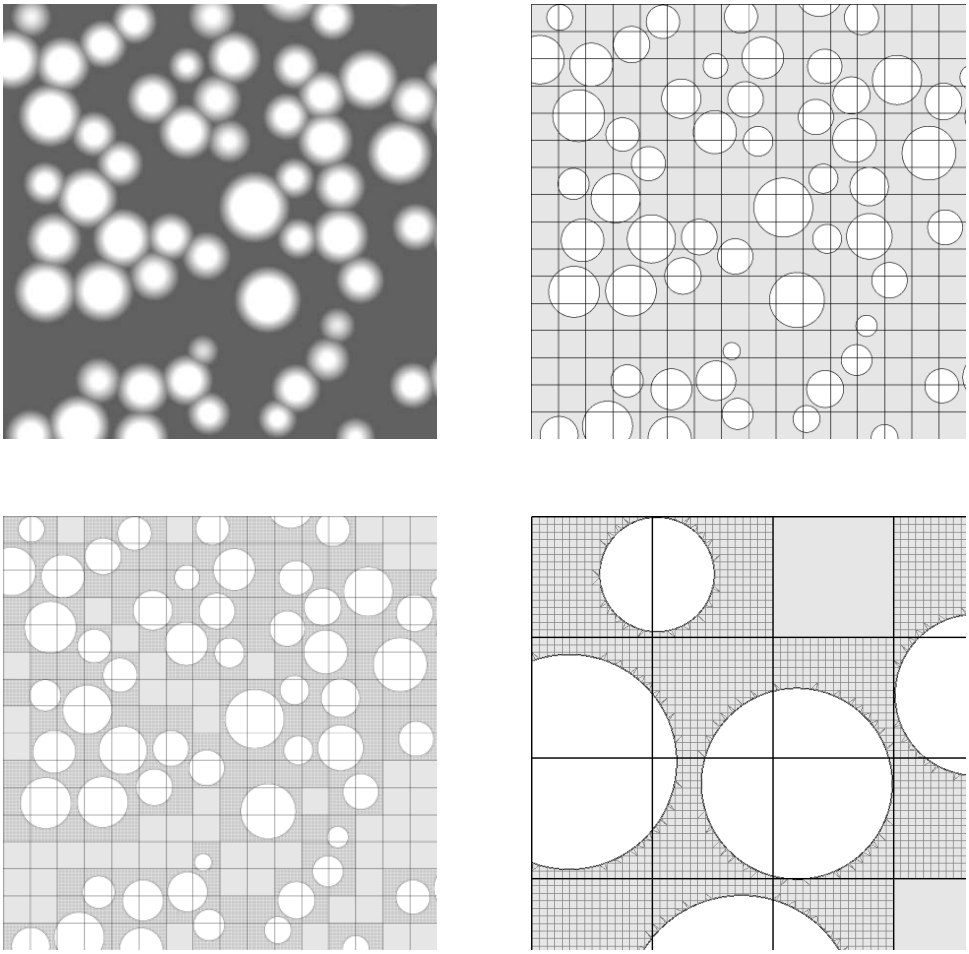


Figure 5.13: A scalar function (upper left) defines a complex geometry of random spheres. The sub-domain boundary $\Gamma^{(i,j)}$ is given as the iso-surface of value 0.0. The fundamental mesh $\mathcal{T}$ intersected with $\mathcal{G}$ gives the finite element mesh (upper right). The local triangulation is constructed by a Marching Cubes based algorithm (lower, closeup in the right).

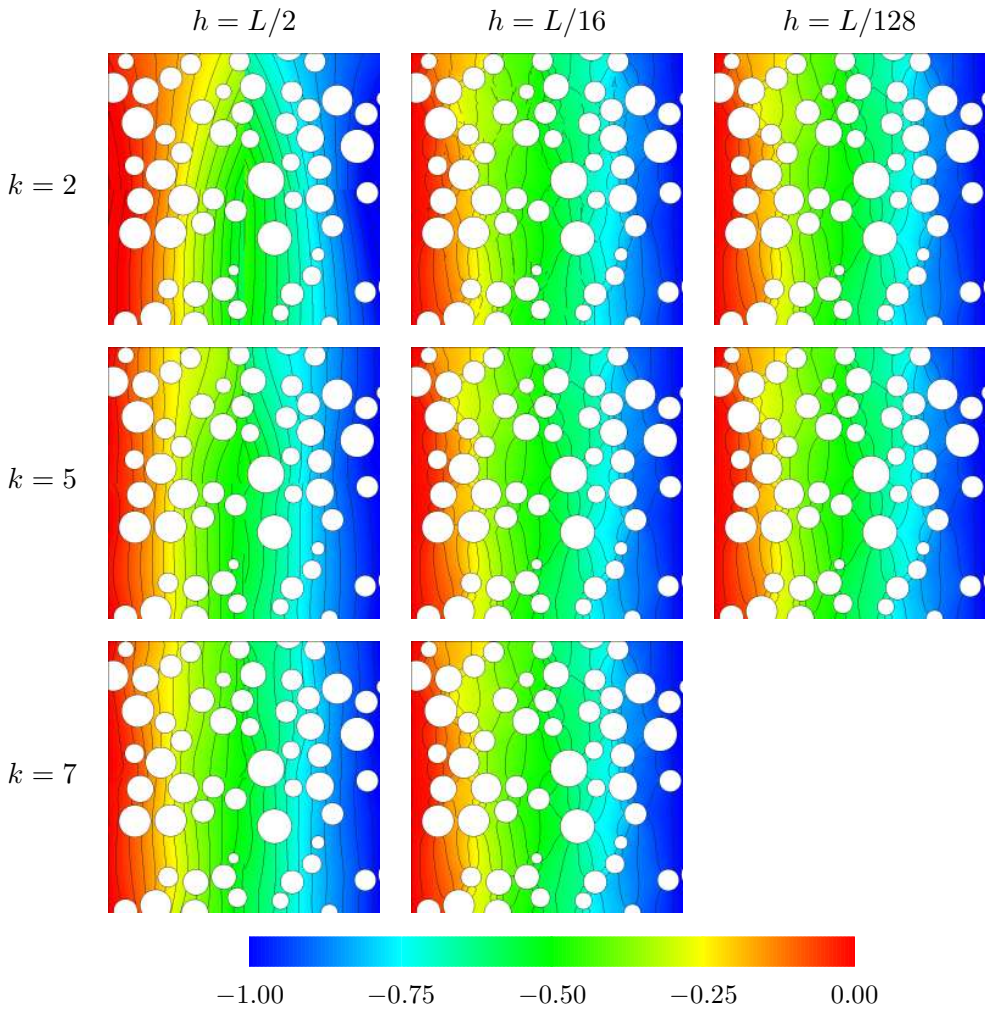


Figure 5.14: Solution and iso-lines. Computations for different polynomial degrees k and different mesh sizes h.

was evaluated. Figure 5.15 shows the flux through the system along a vertical line, the approximation error and the order of convergence for different polynomial degrees. The error is always computed with respect to a reference solution on a grid with h being half the size of the finest computation grid. The domain is given on a grid with $h = L/256$. Since this problem does not show sufficient regularity, the convergence rate is limited, as can be seen in the graphs (Figure 5.15).

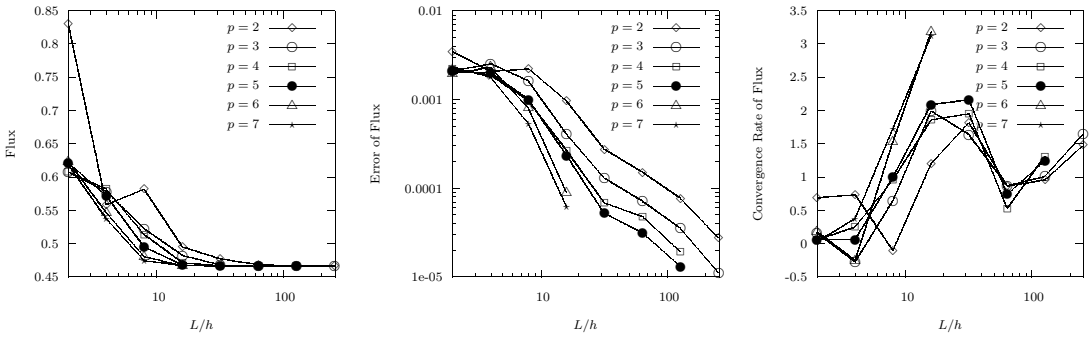


Figure 5.15: Flux through the inflow boundary (left), the approximative error (middle) and the order of convergence (right). Insufficient regularity limits the convergence rate.

5.4 Super-Convergence of the Discontinuities

It is in the nature of Discontinuous Galerkin methods that the solutions in general are not continuous over element boundaries. These jumps $[\![u]\!]$ (Definition 2.11) in the solution must vanish for $h \to 0$.

Theoretical estimates of the convergence rate of jump in the solution along an internal edge are shown in [16]. Furthermore, it is stated that for problems with sufficient regularity super-convergence of the jumps can be observed; numerical examples are given.

We define the L_2-norm along an edge as

$$\|f\|_{L_2,e} = \left(\int_{\gamma_e} f^2 ds \right)^{\frac{1}{2}}. \tag{5.19}$$

and examine the jump in the solution

$$\mathrm{Err}_{\mathrm{jump}} = \sup_e \|[\![u]\!]\|_{L_2,e}. \tag{5.20}$$

For the elliptic test problem on a domain with degenerated elements, the convergence of the jump is investigated. The domain and the boundary conditions are the same as for the computations in Section 5.3.2. Even for this example we can observe super-convergence in the jump.

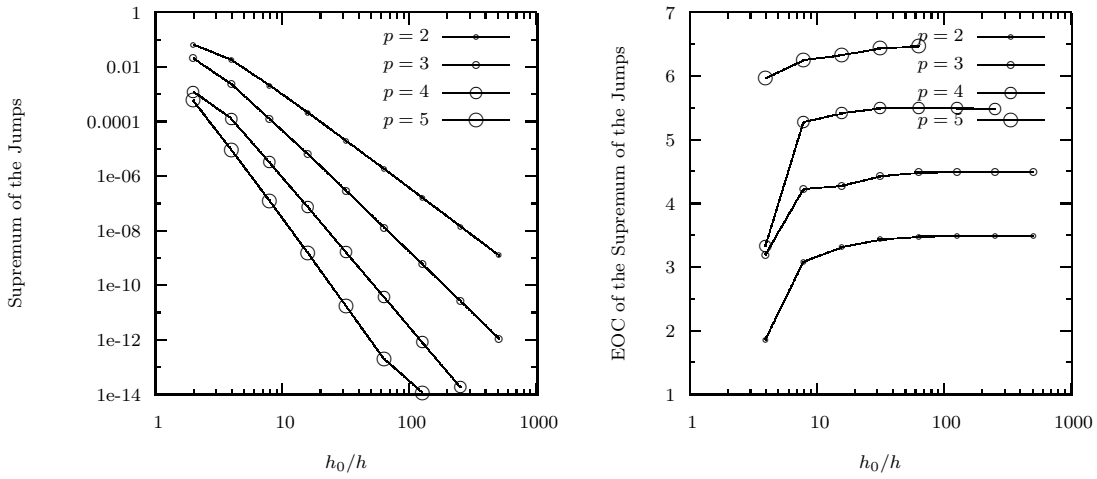


Figure 5.16: Super convergence of $\sup_e \|[\![u]\!]\|_{L_2,e}$ is observed for a problem with sufficient regularity.

5.5 Conservation of mass

Conservation of mass is a fundamental physical concept. It is strongly related to Gauss theorem and basically states that mass cannot vanish. Considering a control volume, the sum over all sink and source terms within the volume must be equal to the sum over all fluxes through the boundary of the volume.

Section 2.3.3 describes the property of mass conservation (2.50) and how it is related to a Discontinuous Galerkin discretization.

In this test we evaluate the global mass error

$$\mathrm{Err}_{\mathrm{mass}} = \int_{\partial\Omega^{(0)}} \mathbf{j} \cdot \mathbf{n} \, ds - \int_{\Omega^{(0)}} f \, dx \tag{5.21}$$

where the flux $\mathbf{j}$ evaluates to

$$\mathbf{j} = F \cdot \mathbf{n} \quad \text{on } \Gamma_N \qquad \text{and} \qquad \mathbf{j} = \nabla u \quad \text{on } \Gamma_D \,. \tag{5.22}$$

The mass conservation is analyzed for three setups from Section 5.3, Dirichlet boundary conditions for the re-entrant corner and the channel flow. All simulations are done using second order ansatz functions. The results (Table 5.4) show that global mass conservation is fulfilled within the error bounds given by the finite accuracy of floating point numbers.

5.6 Efficiency Comparison

Chapter 3 states that the Unfitted Discontinuous Galerkin method has the favorable property that it allows discretizations on very coarse function spaces and still normal

Mesh Size	Re-entrant Corner	Single Sphere	Random Spheres
$H/h = 1$	$7,88 \times 10^{-15}$	$8,93 \times 10^{-17}$	$1,92 \times 10^{-16}$
$H/h = 2$	$2,47 \times 10^{-15}$	$1,45 \times 10^{-16}$	$4,26 \times 10^{-16}$
$H/h = 4$	$8,70 \times 10^{-16}$	$2,05 \times 10^{-16}$	$2,30 \times 10^{-16}$
$H/h = 8$	$4,03 \times 10^{-16}$	$2,87 \times 10^{-16}$	$3,56 \times 10^{-16}$
$H/h = 16$	$1,30 \times 10^{-15}$	$1,21 \times 10^{-16}$	$8,56 \times 10^{-17}$
$H/h = 32$	$3,22 \times 10^{-16}$	$2,36 \times 10^{-16}$	$1,93 \times 10^{-16}$
$H/h = 64$	$7,12 \times 10^{-16}$	$9,37 \times 10^{-16}$	$9,60 \times 10^{-16}$

Table 5.4: Verifying the discrete mass conservation. Global mass error $\mathrm{Err}_{\mathrm{mass}}$ is computed for different setups.

convergence behavior can be observed.

Section 5.3.4 shows computations on a domain with holes. For this domain and a problem with known solution, the UDG method is compared with conforming finite elements with first order Lagrange ansatz functions. Figure 5.18 plots the discretization error in the L_2-norm over the number of degrees of freedom (#DOF). For the conforming discretization as well as for the second order OBB scheme the L_2-error converges with h^2. As the size of the function space is proportional to h^{-2} we expect a linear behavior in the plot.

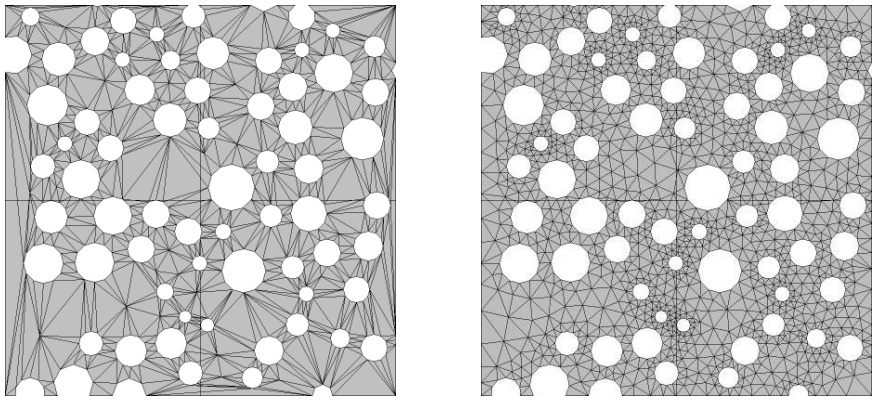


Figure 5.17: Conforming meshes, generated with the Gmsh mesh generator. For coarse meshes the mesh quality is very poor (left), it is not possible to create a mesh with less than 822 nodes. A mesh with 1615 nodes already shows good element quality (right).

For the construction of a conforming finite mesh the Gmsh mesh generator [53] was used (Figure 5.17). It was not possible to create a mesh with less than 822 nodes. For coarse grids, the mesh quality is extremely poor and hence the discretization error diverges. For a mesh with more degrees of freedom the expected linear behavior is obtained. It was to be expected that the conforming discretization is not suitable for

very coarse grids.

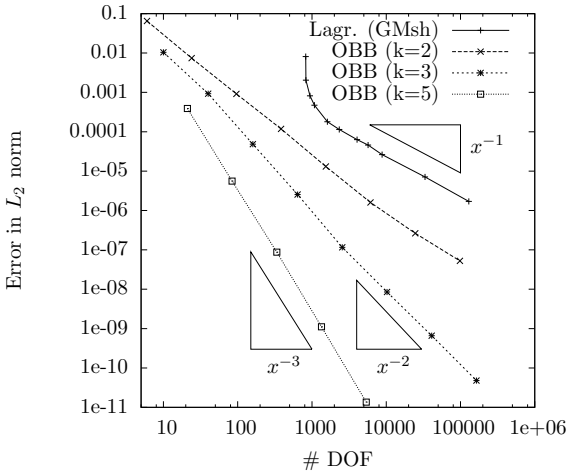


Figure 5.18: Efficiency of the UDG method compared with conforming finite elements. The UDG method allows discretizations with any number of degrees of freedom (DOF). For the same number of DOFs the discretization of the UDG method ($k = 2$) is at least one order of magnitude smaller than using conforming finite elements, the order of convergence is the same. The minimal number of unknown for conforming finite elements is limited by the geometry and close to that limit the discretization error diverges.

For the UDG simulations, the same local triangulation as in Section 5.3.4 is used. The OBB scheme with $k = 2, 3, 5$ is applied. Even for very coarse grids with only one element, the error shows the expected behavior. For second order ansatz, the OBB scheme is only second order accurate. Still it should be noted that also for larger function spaces, the discretization error of the Unfitted Discontinuous Galerkin method is at least one order of magnitude smaller than that of the conforming method for the same number of degrees of freedom.

All simulations with less than 822 degrees of freedom form a kind of numerical upscaling process. The discretization doesn't resolve the microscopic scale anymore, it only yields a macroscopic solution. The accuracy of the upscaled solution can be chosen as needed. Comparing with other multi-scale approaches we are able to quantify the error induced by the upscaling process, provided a suitable a-posteriori error estimator is available.

5.7 Discussion

For an elliptic model problem different the stability and convergence behavior of the Unfitted Discontinuous Galerkin method is investigated numerically.

For setups with different domains, different schemes and different boundary con-

ditions optimal convergence rates in L_2- and H^1-norm are obtained for the UDG approach. For a problem with full regularity the predicted super-convergence of the discontinuities $\sup_e \|[\![u]\!]\|_{L_2,e}$ is observed.

In contrast to normal mesh generation algorithms the ratio χ of the size of two adjacent elements is not bound. It is verified numerically that even big ratios χ do not harm the convergence of the global solution. Also global mass conservation is verified for different setups.

Comparing with a conforming first order Lagrange discretization shows that the UDG method is able to give results on very coarse grids with the appropriate convergence rate. It is also notable that the discretization error of the UDG method is at least one order of magnitude smaller than that of the conforming discretization, compensating the bigger stencil of Discontinuous Galerkin methods.

Chapter 6

Stationary Applications

There are no such things as applied sciences, only applications of science.
— Louis Pasteur

6.1 Numerical Upscaling – Permeability

On the pore scale, groundwater flow is modeled by the incompressible Navier–Stokes equations on a domain with a complex shape. Fluid velocity in groundwater processes is usually slow, therefore the flow can be assumed to be laminar and the non-linear transport term can be neglected. Thus the Navier–Stokes equations reduce to the Stokes equations

$$-\mu\Delta\mathbf{v} + \nabla p = \mathbf{f} \qquad \text{in} \quad \Omega \subset \mathbb{R}^d \tag{6.1a}$$
$$\nabla\cdot\mathbf{v} = 0 \qquad \text{in} \quad \Omega \tag{6.1b}$$
$$\mathbf{v} = 0 \qquad \text{on} \quad \Gamma_0 \subseteq \partial\Omega \tag{6.1c}$$
$$\partial_n\mathbf{v} + (p_0 - p)\mathbf{n} = 0 \qquad \text{on} \quad \Gamma_P = \partial\Omega \setminus \Gamma_0\,, \tag{6.1d}$$

with viscosity μ , velocity $\mathbf{v}$ and pressure p. On the internal boundaries, i. e. on the surface of the grains, no-slip boundary conditions are appropriate. On the boundaries of the macroscopic domain, we consider Dirichlet and natural Neumann boundary conditions for velocity.

On the macroscopic scale, groundwater flow is described by conservation of mass and *Darcy's Law*, which was introduced as an empirical flux law in [40], but can be rigorously derived from Stokes equations using homogenization [25]. The groundwater equations are given by

$$\nabla \cdot \mathbf{u} = 0 \qquad \text{in} \quad \Omega \subset \mathbb{R}^d \tag{6.2a}$$
$$\mathbf{u} = -\tfrac{1}{\mu}\kappa\nabla p \qquad \text{in} \quad \Omega \tag{6.2b}$$
$$p = p_0 \qquad \text{on} \quad \Gamma_D \subseteq \partial\Omega\,, \tag{6.2c}$$

where $\mathbf{u}$ denotes the Darcy velocity, $\mathbf{n}$ is the outward pointing normal vector, p the pressure, μ the viscosity and κ the permeability tensor. In experiments, as well as numerical simulations, κ is often assumed to be a diagonal tensor.

The mean pore velocity $\bar{\mathbf{v}}$ is given by the macroscopic velocity $\mathbf{u}$ and the porosity Φ

$$\bar{\mathbf{v}} = \frac{\mathbf{u}}{\Phi}\,. \tag{6.3}$$

6.1.1 Discretization of Stokes Equations

For the discretization of the Stokes equations we use the Discontinuous Galerkin formulation described in [88], it was implemented in the thesis of S. Kuttanikkad [67]. This discretization guarantees local mass conservation as well as conservation of momentum. Pressure boundary conditions are imposed as described in [63]. The Discontinuous Galerkin discretization of the Stokes equations reads:

Find $\mathbf{v} \in [V_k^{(i)}]^d$, $p \in V_{k-1}^{(i)}$ such that

$$\begin{aligned} \mu\, a(\mathbf{v},\mathbf{z}) + J(\mathbf{v},\mathbf{z}) + b(\mathbf{z},p) &= l(\mathbf{z}) \qquad && \forall\ \mathbf{z} \in [V_k^{(i)}]^d \quad , \\ b(\mathbf{v},q) &= 0 && \forall\ q \in V_{k-1}^{(i)} \quad . \end{aligned} \tag{6.4}$$

where

$$a(\mathbf{u},\mathbf{v}) = \int_\Omega \nabla\mathbf{u} : \nabla\mathbf{v} \;- \int_\Gamma \{\,\nabla\cdot\mathbf{u}\,\}[\![\,\mathbf{v}\,]\!] + \epsilon \int_\Gamma \{\,\nabla\cdot\mathbf{v}\,\}[\![\,\mathbf{u}\,]\!] \tag{6.5}$$

$$b(\mathbf{u},q) = -\int_\Omega q\;\nabla\cdot\mathbf{u} + \int_\Gamma \{\,q\,\}[\![\,\mathbf{u}\,]\!] \tag{6.6}$$

$$l(\mathbf{z}) = -\int_{\Gamma_D} p_0\,\mathbf{v}\cdot\mathbf{n}ds \tag{6.7}$$

and where $[\![\,\cdot\,]\!]$ and $\{\,\cdot\,\}$ denote the component-wise jump and average defined in Equation (2.11) and (2.12).

$J(\mathbf{v},\mathbf{z})$ is an interior penalty term [109], which is of non-physical origin. Dirichlet type boundary conditions (e. g. no-slip and no-flux) are enforced weakly. The penalty term vanishes for $h \to 0$ and penalizes jumps in the solution, in order to enforce a certain continuity of the solution.

$$J(\mathbf{v},\mathbf{z}) = \sum_{\gamma\in\Gamma} \eta h_\gamma^{-1} \int_\gamma [\![\,\mathbf{v}\,]\!][\![\,\mathbf{z}\,]\!]\,ds\,, \tag{6.8}$$

where h_γ is defined as in (2.44), proportional to the size of γ.

6.1.2 Homogenization

Darcy's law was first derived as an empiric law. However, in 1978 Bensoussan et al. presented a rigorous derivation of Darcy's law from Stokes equations [25].

In the following we briefly derive Darcy's law, introducing all concepts and terms necessary to understand the application of numerical upscaling. Together with the homogenized counterpart of the Stokes equations, effective parameters for Darcy's law on a macroscopic domain Ω are obtained. For a detailed description and rigorous proof we refer to [65].

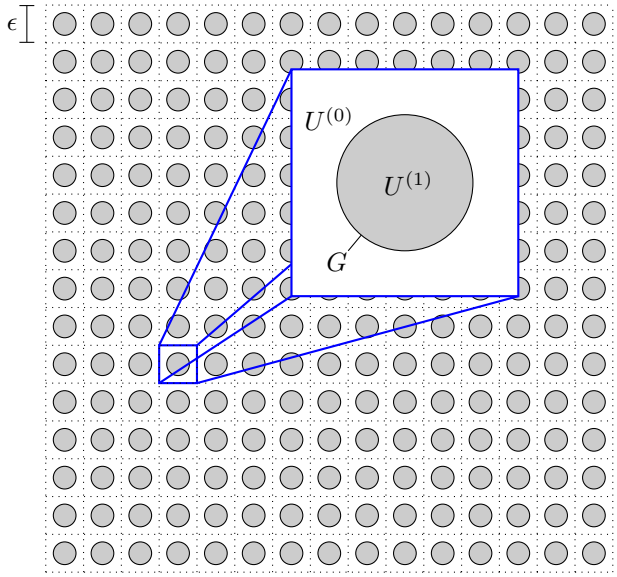


Figure 6.1: A periodic micro-structure with characteristic length ϵ. Each unit cell has sub-domains $U^{(0)}$ (fluid part) and $U^{(1)}$ (solid part), separated by the boundary G.

The pore scale geometry is modeled as a periodic structure with characteristic length ϵ (Figure 6.1). The macroscopic domain Ω now depends on ϵ and hence is denoted by Ω^ϵ. It consists of $N(\epsilon) \propto (\frac{1}{\epsilon})^d$ cells U_j^ϵ of size ϵ^d. Each U_j^ϵ maps to the unit cell U using a scaling by $\frac{1}{\epsilon}$ and a translation. The unit cell U has the microscopic (local) coordinates $y \in (0,1)^d$ and consists of two sub-domains $U^{(0)}$ (the fluid part) and $U^{(1)}$ (the solid part), separated by the boundary G. Using this microscopic partition

of U, a partition of the macroscopic domain Ω is induced as

$$\mathcal{G}^\epsilon(\Omega^\epsilon) = \left\{\Omega^{\epsilon(0)}, \Omega^{\epsilon(1)}\right\}, \tag{6.9}$$

$$\bar{\Omega}^{\epsilon(i)} = \bigcup_{j=0}^{N(\epsilon)} \bar{U^\epsilon}_j^{(i)}, \tag{6.10}$$

$$\Gamma^\epsilon = \bigcup_{j=0}^{N(\epsilon)} {G^\epsilon}_j^{(i)}. \tag{6.11}$$

The central idea of homogenization is to consider a family of functions $\mathbf{v}^\epsilon$ and p^ϵ which depend on the characteristic length ϵ. The upscaling process consists of determining the limit $\mathbf{v} = \lim_{\epsilon \to 0} \mathbf{v}^\epsilon$ which yields the macroscopic equations.

On the macroscopic scale the Stokes problem with $f = 0$ and no-slip boundary conditions on Γ^ϵ are considered:

$$-\epsilon^2 \mu \Delta \mathbf{v}^\epsilon + \nabla p^\epsilon = 0 \qquad \text{in} \quad \Omega^{\epsilon(0)} \tag{6.12a}$$

$$\nabla \cdot \mathbf{v}^\epsilon = 0 \qquad \text{in} \quad \Omega^{\epsilon(0)} \tag{6.12b}$$

$$\mathbf{v}^\epsilon = 0 \qquad \text{on} \quad \Gamma^\epsilon, \tag{6.12c}$$

where $\mathbf{v}^\epsilon$ and p^ϵ are written with a superscript ϵ as they depend on the micro-structure. The velocity v^ϵ is scaled with ϵ^2. This scaling is needed later in the limiting process for $\epsilon \to 0$ (for details see [65]).

To determine the limit $\epsilon \to 0$ a two scale approach is followed. It is assumed that there is an asymptotic expansion of the unknown variable in the form

$$\begin{aligned} p^\epsilon(x,y) &= p_0(x,y) + \epsilon p_1(x,y) + \epsilon^2 p_2(x,y) + \dots \\ \mathbf{v}^\epsilon(x,y) &= \mathbf{v}_0(x,y) + \epsilon \mathbf{v}_1(x,y) + \epsilon^2 \mathbf{v}_2(x,y) + \dots , \end{aligned}$$

where all $\mathbf{v}_i$ and p_i are functions in $\Omega \times U^{(0)}$, periodic in y. Note that the variable x is now defined on whole Ω and it doesn't know about the microscopic structure as this information is only visible on $U^{(0)}$. The gradient and the Laplace operator on $\Omega \times U^{(0)}$ are given as

$$\nabla = \nabla_x + \frac{1}{\epsilon}\nabla_y \qquad \text{and} \qquad \Delta = \Delta_x + \frac{1}{\epsilon^2}\Delta_y + \frac{1}{\epsilon}\nabla_x \nabla_y + \frac{1}{\epsilon}\nabla_y \nabla_x.$$

The resulting Stokes problem reads

$$\begin{aligned} -\epsilon^2 \mu \frac{1}{\epsilon^2}\Delta_y \mathbf{v}_0(x,y) + \frac{1}{\epsilon}\nabla_y p_0(x,y) \\ + \nabla_y p_1(x,y) + \nabla_x p_0(x,y) + O(\epsilon) = 0 \qquad \text{in} \quad \Omega \times U^{(0)} \end{aligned} \tag{6.13a}$$

$$\begin{aligned} \nabla_x \cdot \mathbf{v}_0(x,y) + \frac{1}{\epsilon}\nabla_y \cdot \mathbf{v}_0(x,y) \\ + \nabla_y \cdot \mathbf{v}_1(x,y) + O(\epsilon) = 0 \qquad \text{in} \quad \Omega \times U^{(0)} \end{aligned} \tag{6.13b}$$

$$\mathbf{v}_0(x,y) + O(\epsilon) = 0 \qquad \text{on} \quad \Omega \times G. \tag{6.13c}$$

Comparing coefficients of different order of ϵ yields

$$\nabla_x \cdot \mathbf{v}_0(x,y) + \nabla_y \cdot \mathbf{v}_1(x,y) = 0 \tag{6.14}$$

and

$$\nabla_y p_0(x,y) = 0\,, \tag{6.15}$$

which is equivalent to $p_0(x,y) = p_0(x)$, i. e. p_0 does not depend on the microscopic variable y. We only consider the terms of zero'th order in ϵ of Equation (6.13a) and use the component wise representation of $\nabla_x p_0 = \sum_j \mathbf{e}_j \partial_j p_0$, where $\mathbf{e}_j$ denotes the j'th cartesian basis vector. Using linearity of the problem the following cell problem is formulated using new U-periodic variables $\mathbf{w}$ and π with

$$\mathbf{v}_{0j} = \frac{\mathbf{e}_j \nabla_x p_0}{\mu} \mathbf{w}_j \qquad \text{and} \qquad p_1 = \mathbf{e}_j \nabla_x p_0 \, \pi\,.$$

The final cell problem reads: Find a U-periodic vector field $\mathbf{w}_j$ that solves

$$\begin{aligned} \Delta \mathbf{w}_j &= \nabla \pi_j - \mathbf{e}_j && \text{in} \quad U^{(0)} \\ \nabla \cdot \mathbf{w}_j &= 0 && \text{in} \quad U^{(0)} \\ \mathbf{w}_j &= 0 && \text{on} \quad G. \end{aligned} \tag{6.16}$$

Averaging the vector field over the unit cell yields Darcy's law

$$\mathbf{u} = \int\limits_{U^0} \mathbf{v}_0(x,y) dy = -\frac{1}{\mu} \kappa \nabla p_0,$$

with the permeability tensor given by

$$\kappa_{ij} = \int\limits_{U^{(0)}} \mathbf{w}_{ji} dy. \tag{6.17}$$

In general it is not possible to compute κ analytically. In these cases numerical simulations of (6.16) are used to determine κ.

6.1.3 Analytical Test Problems

For simple periodic porous media an analytic computation of the permeability κ is possible using homogenization techniques.

We consider the case of flow through a simple porous medium, i. e. a periodic simple cubic (SC) array of spheres of equal radii as shown in Figure 6.2. Sangani and Acrivos derived semi-analytic results [91] for periodic arrays of spheres with equal radii. The spheres are arranged in crystal lattice structures as shown in Figure 6.2. They could give the permeability as an expansion series, the coefficients are computed numerically with high accuracy. Three different setups are possible, we will always consider the case of touching spheres:

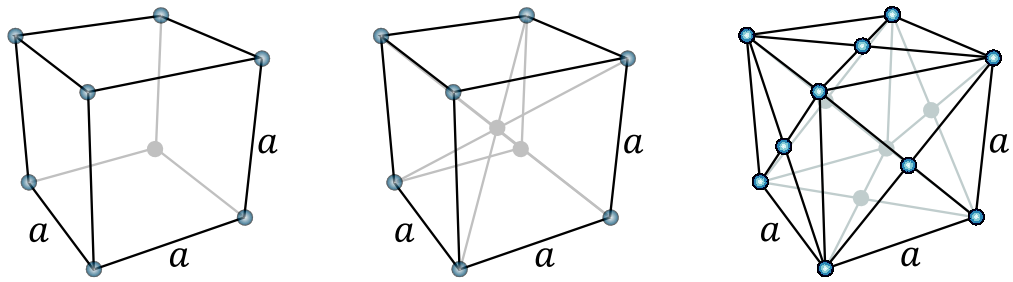


Figure 6.2: Crystal lattice structures. left: Simple Cubic (SC), middle: Body Centered Cubic (BCC), right: Face Centered Cubic (FCC)

Simple cubic array (SC):
eight spheres situated in the corners of a cube,

Body centered cubic array (BCC):
nine spheres situated in the corners and the center of a cube,

Face centered cubic array (FCC):
14 spheres situated in the corners and the face centers of a cube.

In [34] Chapman and Higdon analytically computed permeabilities for all three setups and a wide range of radii are given. These results will be compared with numerical computations for the permeability. In all three examples the problem is isotropic. Thus the permeability tensor κ will also become isotropic and is represented by a scalar. To compute the scalar permeability coefficient κ we choose an arbitrary direction i and compute $\mathbf{w}_{ii}$ following the formulation in (6.16), without loss of generality we choose the x-direction. As in the homogenized formulation the computations are carried out on a periodic domain, on the grain surface no-slip boundary conditions are applied. Figure 6.3 shows the periodic unit cell and the velocity magnitude of the computed solution.

Type	r	Φ_{ana}	Φ_h	$\text{Err}_{\Phi}^{\text{rel.}}$	κ_{ana}	κ_h	$\text{Err}_{\kappa}^{\text{rel.}}$
SC	$\frac{1}{2}$	0.4764	0.478	0.5%	$2.527e-3$	$2.53e-3$	0.03%
BCC	$\frac{1}{4}\sqrt{3}$	0.3198	0.321	0.7%	$4.350e-4$	$4.52e-4$	3.8%
FCC	$\frac{1}{4}\sqrt{2}$	0.2595	0.264	1.5%	$8.68e-5$	$9.06e-5$	4%

Table 6.1: Porosity Φ and permeability κ for SC, BCC, and FCC sphere arrangement. Comparison between numerical and analytic [34] results.

The analytic values are taken from [34], where high precision results for all three setups are presented. Table 6.1 shows the relative error on the porosity Φ which is due to the linear approximation of the spherical shape, and the relative error in the computed permeability κ. The geometry is given on an image grid with $h_g = 1/32$.

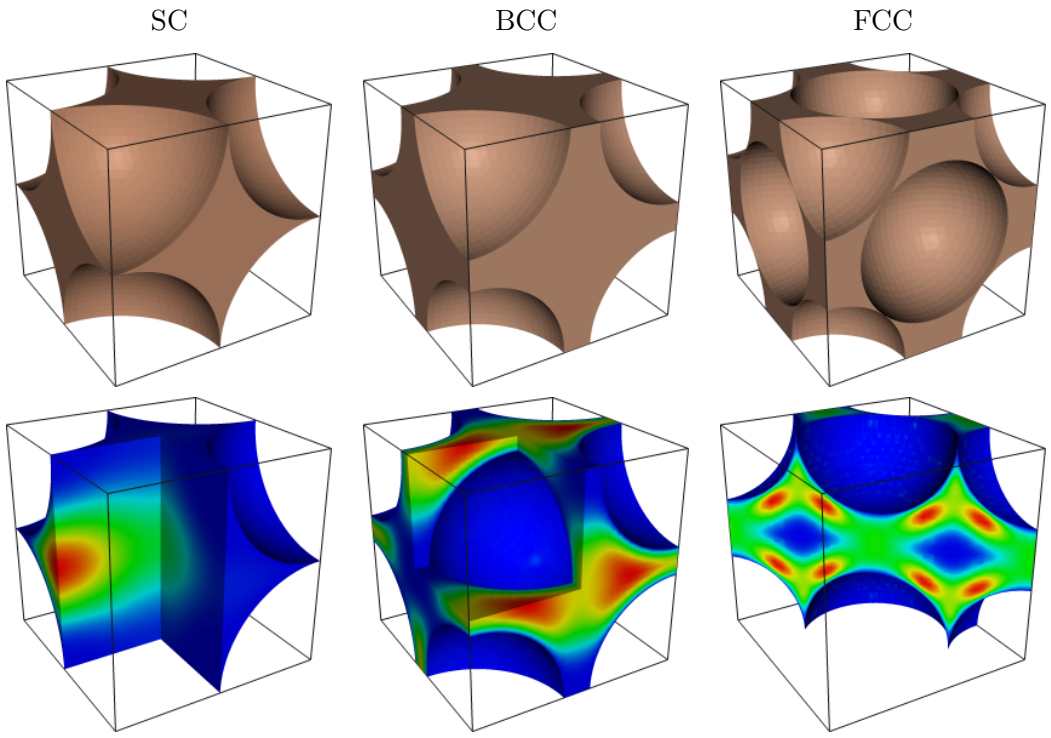
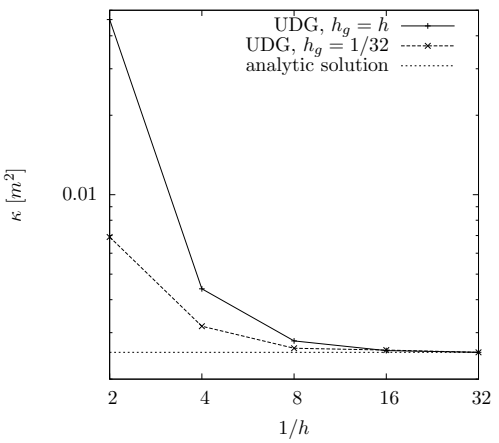
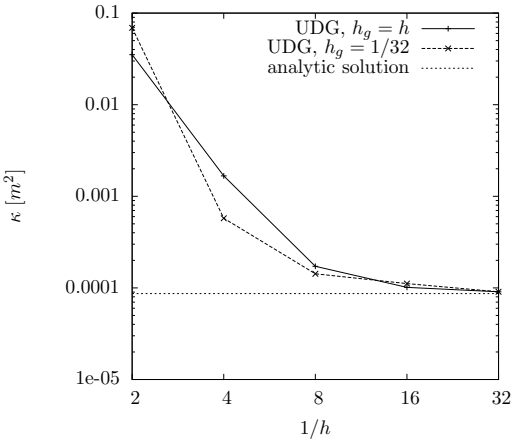


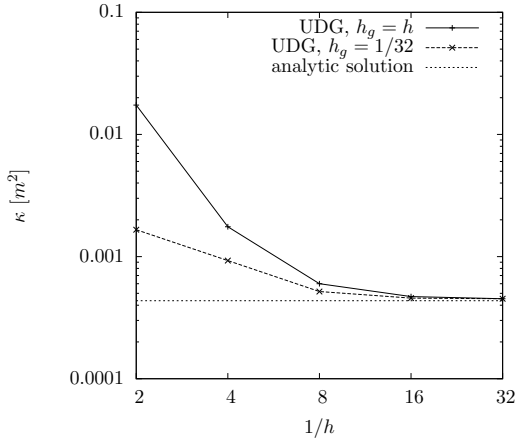
Figure 6.3: Unit cells for SC, BCC and FCC layout (from left to right) on an image grid with $h = 1/32$ and the computed velocity magnitude.



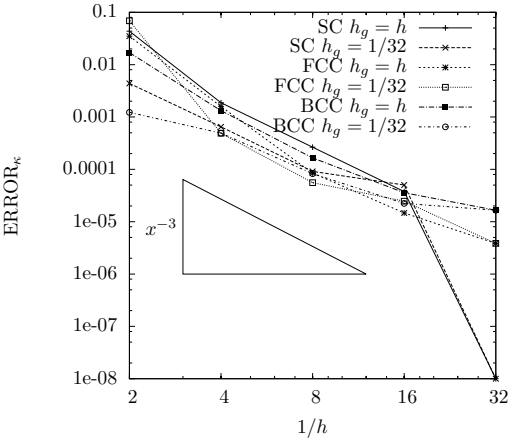
(a) Computed permeability κ_h (SC setup).



(b) Computed permeability κ_h (FCC setup).



(c) Computed permeability κ_h (BCC setup).



(d) Error $|\kappa_h - \kappa_{\mathrm{ana}}|$.

Figure 6.4: Computed permeability for a periodic domain with known analytic solution. Comparing results for different mesh size h of the fundamental grid and different h_g of the image grid.

For the SC setup, Figure 6.4(a) shows convergence of the computed permeability for different mesh sizes h of the fundamental grid and compares these with the analytic solution. First the solution is computed with a setup where the image grid and the fundamental grid are identical, i. e. $h = h_g$. In the second run the geometry is always defined on an image grid with $h_g = 1/32$ and the fundamental mesh is coarse. The error of the solution of the second run is always smaller than that of the more classic approach with $h = h_g$.

6.2 Random Media

In the case of random media the assumption of periodicity required for classical homogenization approaches does not hold any more. Sometimes the technique of stochastic homogenization [55] can be an alternative but it is also not generally applicable. Thus a different approach to determine the permeability must be chosen.

Representative Elementary Volume

An alternative approach for the transition from the microscopic scale to the macroscopic scale is given by an averaging approach, known as the concept of the *Representative Elementary Volume* (REV); see [22]. An REV is the smallest volume which is representative for the homogeneous medium. Quantities on the microscopic scale are replaced by averaged quantities on the macroscopic scale.

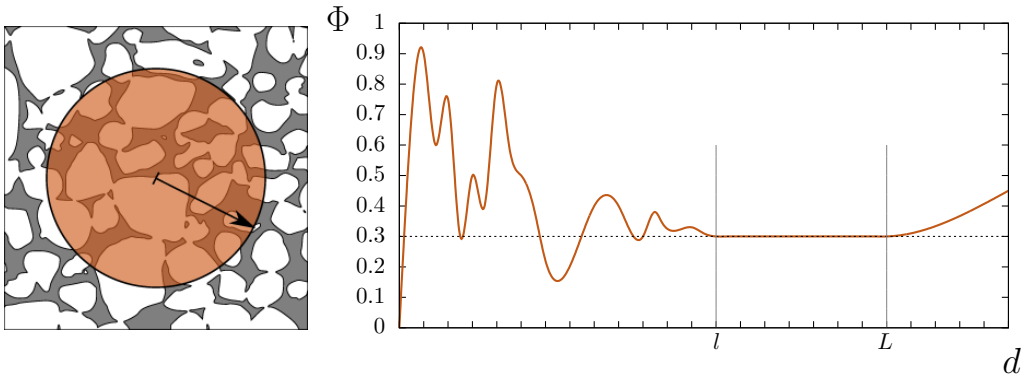


Figure 6.5: Porosity Φ for different diameters d of the averaging volume.

We consider a small sub-domain of a macroscopically homogeneous porous medium, measure a property in a non-destructive way and then enlarge the sub-domain and measure again. The idea of an averaging volume assumes that repeated measurements will (after initial fluctuations) converge to a value which is representative for this porous medium. When we enlarge the sub-domain further the measurements might diverge again due to macroscopic inhomogeneities (Figure 6.5). Let us illustrate this concept for the property of porosity.

On the microscopic scale of a porous medium, a point x is either part of the void space or of the solid matrix. The averaging volume $U(x, d)$ is a sub-domain of the porous medium, centered around the point x with a diameter d. In contrary to homogenization the size of the inhomogeneities is fixed and does not scale with the diameter of U.

Classically, an indicator function

$$\chi(x) = \begin{cases} 1 & \text{if } x \text{ is in a pore,} \\ 0 & \text{if } x \text{ is in the solid phase} \end{cases} \tag{6.18}$$

is defined. For our computations however we will distinguish between two sub-domains $U^{(0)}(x, d)$ and $U^{(1)}(x, d)$. As in Section 6.1.2 we denote the pore space as $U^{(0)}$ and the solid phase as $U^{(1)}$. The porosity of the porous medium with respect to the averaging volume is then given as

$$\begin{aligned} \Phi(x, d) &= \frac{1}{|U(x, d)|} \int\limits_{U(x,d)} \chi(y)\, dy \\ &= \frac{1}{|U(x, d)|} \int\limits_{U^{(0)}(x,d)} 1\, dy \end{aligned} \tag{6.19}$$

A typical graph for the porosity $\Phi(x, d)$ is shown in Figure 6.5. For small d the discontinuities of χ produce large variations in Φ. At a diameter l these variations decrease and the value for Φ stabilizes. For large d large scale inhomogeneities change Φ again.

Definition 6.1 (REV): *Given a range $[l, L]$ with $l \ll L$. If the measured property (e. g. the porosity) does not depend on the diameter d of the averaging volume $\Omega_{avg}(x, d)$ for $d \in [l, L]$ this averaging volume is called a* Representative Elementary Volume. *The diameter d can be chosen anywhere in the range $[l, L]$.*

This averaging approach offers access to macroscopic quantities where periodic homogenization is not applicable. It doesn't require periodicity of the microscopic structure. Still it can be difficult or even impossible to determine the REV for a given medium, and the size of the REV can vary considerably.

6.2.1 Artificially generated Pore Structure

In this example the domain partition $\mathcal{G}$ is given by an artificially generated pore structure. We choose a cubic averaging volume U, filled with 100 randomly packed spheres of uniform radius $r = 0.106$, see Figure 6.6. The sphere distribution was computed using the packing algorithm from [98]. Again the domain is given implicitly as a scalar function with mesh size $h_g = 1/64$.

For this setup we computed a macroscopic porosity $\Phi = 0.554$. Keeping the sphere positions, but varying the radius r changes Φ. In Table 6.2 different values of r and the resulting porosities Φ are shown.

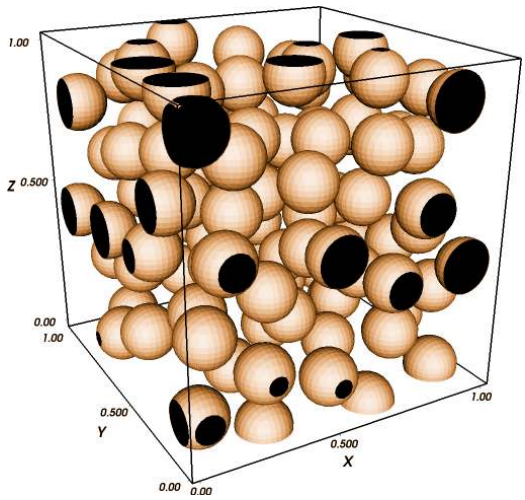


Figure 6.6: An artificially generated pore structure is given as a scalar function on a mesh with mesh size $h_g = 1/64$.

r	0.053	0.063	0.074	0.079	0.084	0.095	0.100	0.106	0.111	0.116
Φ	0.943	0.901	0.844	0.809	0.769	0.674	0.618	0.554	0.486	0.417

Table 6.2: Change of the macroscopic porosity Φ with the radius r of the spheres.

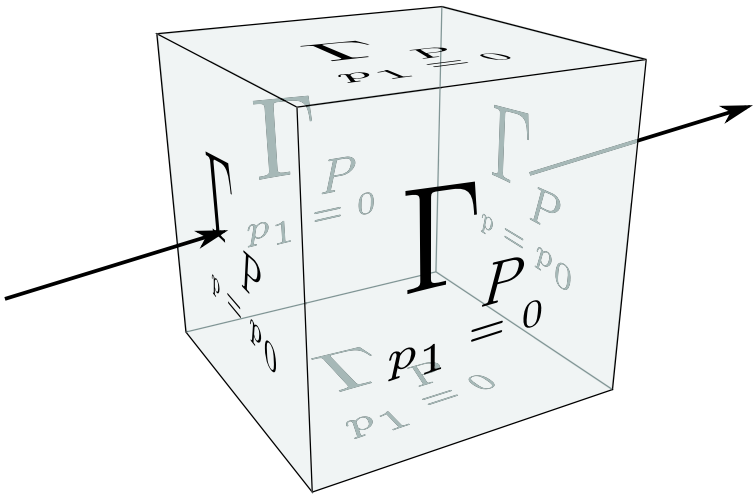


Figure 6.7: A macroscopic pressure gradient is imposed along the x-axis using pressure boundary conditions and no-slip boundary condition otherwise. Direct simulation yields an effective κ.

Effective parameters for Darcy's law can be computed by solving Stokes equations on the pore space domain $U^{(0)}$. If U has the size of an REV the computed value is valid also for other samples and larger scales. Finding the correct size for an REV is a complex question it self, which we do not want to approach now. It is not verified that the domain is an REV, still it is sensible to compute the effective permeability for this particular setup.

As in homogenization the pressure is separated into a global pressure p_0, with $\nabla p_0 =$ const. on U, and local fluctuations p_1. The local fluctuations p_1 are computed for different global pressures gradients $\nabla p_0 = \mathbf{e}_i$ for the different unit vectors $\mathbf{e}_x, \mathbf{e}_y$, and $\mathbf{e}_z$. Equation (6.17) yields the dimensionless entries of the permeability tensor κ for the cell problem. The global pressure gradient ∇p_0 is statically evaluated and contributes to the right hand side. The microscopic pressure vanishes on the inflow and outflow boundary, $p_1 = 0$ is imposed using pressure boundary condition for (6.1). On all inner boundaries $\partial U^{(0)} \setminus \partial U$ no-slip boundary condition is imposed (Figure 6.7).

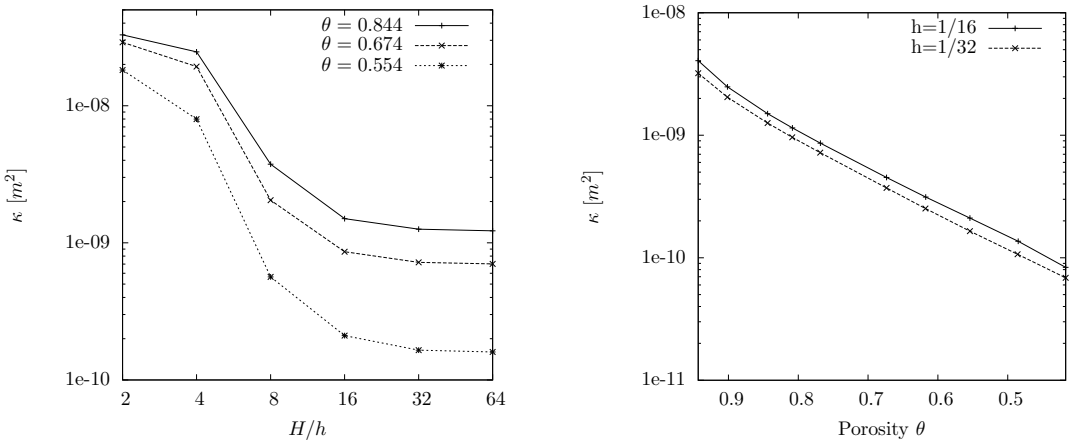
Following the REV approach, the mean velocity

$$\bar{\mathbf{v}} = \int_{U^{(0)}} \mathbf{v}\, dx \cdot |U^{(0)}|^{-1} \tag{6.20}$$

and the macroscopic porosity $\Phi = \frac{|U^{(0)}|}{|U|}$ are computed, where $|U|$ denotes the size of U. Averaging yields the permeability coefficient

$$\kappa_{ij} = -\mu \frac{\bar{\mathbf{v}}\, \Phi}{\mathbf{e}_j} \ . \tag{6.21}$$

The size of the averaging volume is $|U| = 1\ \text{mm}^3$ and the viscosity of water is $\mu_{\text{water}} = 1 \cdot 10^{-3}$ Pa s.



(a) Grid Convergence: κ for different mesh size h of the fundamental grid.

(b) Effective Permeability: κ computed for different porosities.

Figure 6.8: Computed permeability for an artificial random porous media.

Figure 6.8 shows the ability of this method to give reliable numerical results for the macroscopic parameter already for a relatively coarse grid. For three different porosities the permeability κ was computed on different mesh sizes h of the fundamental mesh.

We observe convergence for $h \to h_g$, reasonable results can already be obtained for meshes with $h > h_g$. Figure 6.8(a) shows that computations on a grid with $h = 1/32$ already yield comparable results as with $h = 1/64$. Already the size of the image grid is coarser than that of a conforming triangulation. But the discretization can use even coarser meshes and it can still yield reliable results.

For a mesh size of $h = 1/32$, respectively $h = 1/16$, a sensitivity analysis was carried out. κ was computed for a range of different Φ (see Table 6.2). Although the geometry is defined on a grid with $h_g = 1/64$, computations are carried out on a coarser grid. This reduces the time for the computation significantly without reducing the accuracy. The resulting permeability component κ_{xx} is plotted in Figure 6.8(b). Even on the very coarse grid with $h = 1/16$ the results show qualitatively the same behaviour as on the fine grid, with a systematic error.

6.2.2 Experimentally obtained Pore Structures

Micro X-ray computer tomography (*micro-CT*) provides non-destructive measurement of small three-dimensional structure down to a resolution of about 1 μm. The sample sizes range from a few millimeters to hundred millimeters. The first micro-CT system was developed by Jim Elliott in the early 1980s [48]. He presented reconstructed slices of a small tropical snail, with a pixel size about 50 μm.

Nowadays micro-CT systems are used in many different fields. They are used to obtain detailed measurements of the pore-scale structure [105, 80] and more recently even to obtain time-dependent measurements [21].

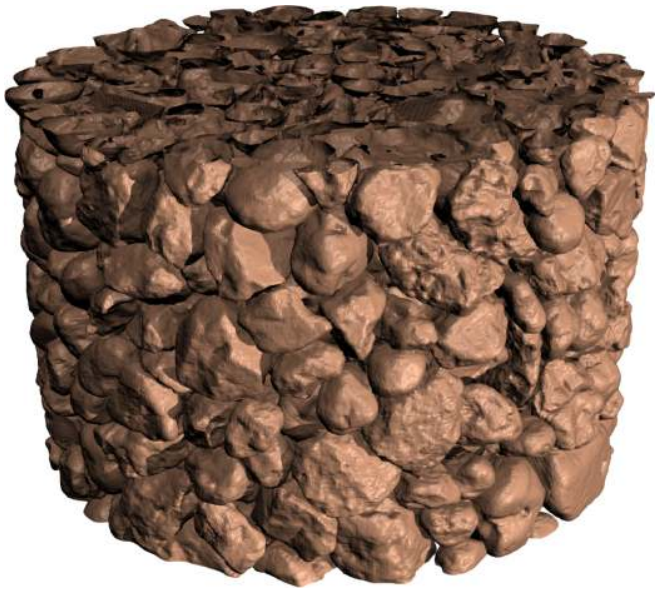


Figure 6.9: Pore structure of a coarse sand (H.-J. Vogel, UFZ Halle). Micro-CT scan of a sample with diameter 10 mm and height 6 mm.

By courtesy of H.-J. Vogel we are able to perform computations on real structures (see Figure 6.9). The micro-CT measurements were done at Department Bodenphysik, Helmholtz Zentrum für Umweltforschung (UFZ Halle). The sample is a cylinder filled with coarse sand. It has a diameter of 10 mm and a height of 6 mm. The data is given as 16bit gray scale values on a $800 \times 828 \times 426$ grid. The current implementation of the UDG method features sequential computations only, hence computations on the whole domain with full resolution were not yet feasible.

Porosity

On a range from $d = 0.015$ mm to 6 mm the porosity of a cube with edge length d was computed. The center of the cube is always at the center of the sample.

Figure 6.10 shows that the limit is not yet reached and that the sample is too small to be an REV. As the size of sample is smaller than that of an REV an effective permeability computed from the microscopic flow field on this domain cannot be directly compared with the macroscopic permeability of the coarse sand. The grain diameter is between approximately 5–10 mm. An REV might have a size which 100 times bigger than the grain size.

Microscopic Flow

The permeability of coarse sand is, according to literature, $10^{-11} - 10^{-9}$ m^2 [22]. Solving the microscopic Stokes problem we were able to compute the macroscopic

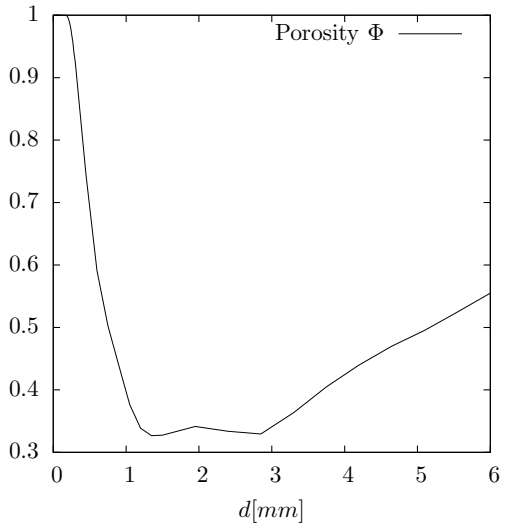


Figure 6.10: Porosity of the micro-CT data. For cubes with different edge length d the porosity is computed. On the whole range no REV can be found.

permeability of this sample.

Stokes equations are solved for different averaging volumes. Figure 6.11 shows the velocity field for a domain of size with edge length $L = 3.84$ mm. The discretization is on a 16^3 fundamental mesh with an image grid of size 64^3 and a resolution of 0.06 mm, this means the original data was coarsened by factor of 4. Second order ansatz functions for the velocity and first order for the pressure are used. The domain and the computed solution are visualized in Figure 6.11.

$L = \text{diam}(\Omega)$	Permeability κ
0.48 mm	$1.1 \cdot 10^{-10}$ m^2
0.96 mm	$5.6 \cdot 10^{-10}$ m^2
1.92 mm	$1.0 \cdot 10^{-9}$ m^2
3.84 mm	$4.3 \cdot 10^{-10}$ m^2

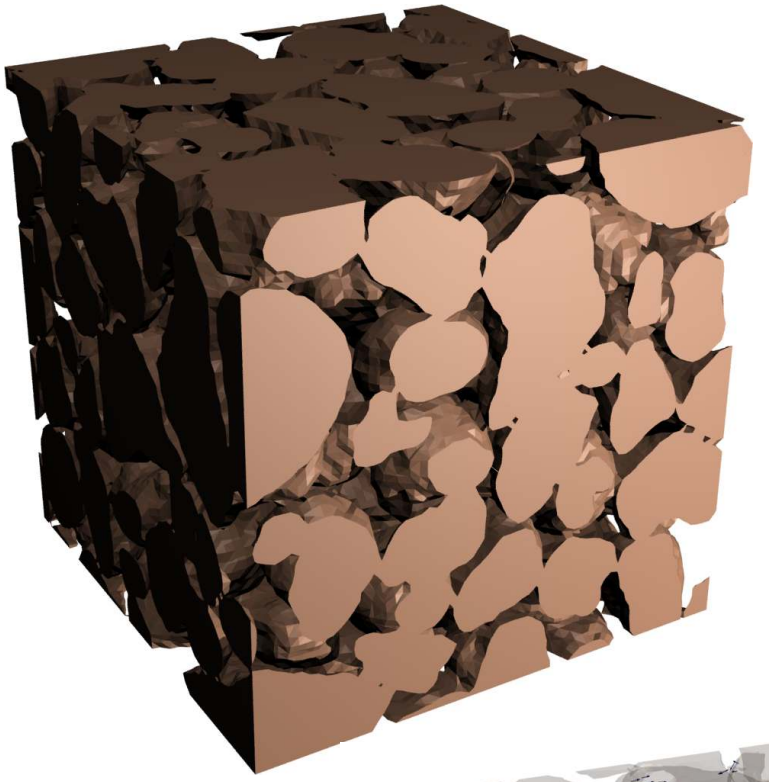
Table 6.3: permeability of a coarse sand, measured for different averaging volumes.

For different sizes L of the averaging volume the effective permeability κ is computed. these values range between 10^{-10} and 10^{-9} m^2 which is in accordance with the predicted values.

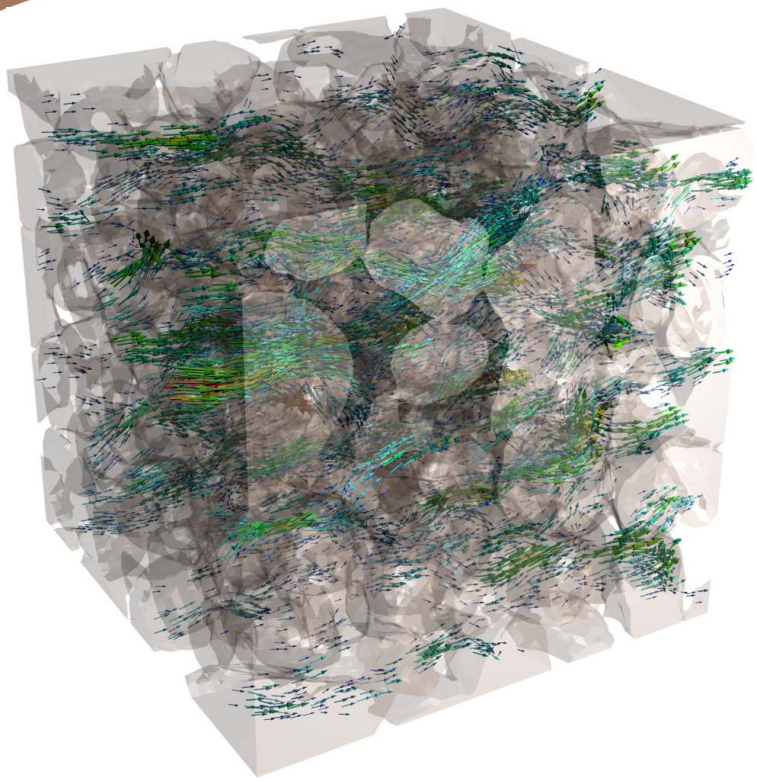
6.3 Discussion

The results show the ability of the UDG method to do reliable simulations on very coarse grids. For different analytic test problems, the UDG method did show h^{k+1} convergence for the macroscopic quantity.

The local triangulation algorithm for implicitly given domains makes it very easy to do computations on measured structures, e. g. micro-CT scans.



Pore scale domain (coarse sand). Selection of size 3.6^3 mm with a resolution of 64^3 voxel. Original data had a resolution of 256^3 voxel.



Velocity field through coarse sand. Pressure drop from left to right and natural boundary conditions on the other faces. Computations are done on a 32^3 mesh with second order ansatz functions for the velocity and first order for the pressure.

Figure 6.11: Microscopic velocity field through porous media. (Visualization using VTK [106] and Blender [28])

When dealing with experimentally obtained data the uncertainty in the measurement is very big. In those situations the accuracy of the discretization should not be too high compared to the uncertainty of the measurement. Otherwise only the computational costs increase, but no additional information can be obtained. Thus, the ability to do simulations on coarse function spaces, as it is featured by the UDG method, is of great interest.

Chapter 7

Time-dependent Applications

πάντα ῥηεῖ
— Heraklit

The simulations presented in this chapter are joint work with J. Fahlke as part of his diploma thesis [50]. The aim is to apply the Unfitted Discontinuous Galerkin method to a time-dependent problem.

7.1 Convection–Diffusion–Reaction Equation

We consider the convection-diffusion-reaction equation as a model problem for parabolic partial differential equations. We will refer to it as the "transport equation"

$$\partial R_t c + \nabla \cdot \mathbf{j} = q(c) \qquad \text{in } \Omega \tag{7.1a}$$
$$\mathbf{j} = \mathbf{u}c - D(\mathbf{u})\nabla c\,, \qquad \text{in } \Omega \tag{7.1b}$$

and we consider Dirichlet, Neumann and outflow boundary conditions

$$c = c_0 \qquad \text{on} \quad \Gamma_D \subseteq \partial\Omega \qquad \text{(Dirichlet BC)} \tag{7.1c}$$
$$\mathbf{j}\cdot\mathbf{n} = F \qquad \text{on} \quad \Gamma_N \subseteq \partial\Omega \setminus \Gamma_D \qquad \text{(Neumann BC)} \tag{7.1d}$$
$$\mathbf{j}\cdot\mathbf{n} = (\mathbf{u}c - D(\mathbf{u})\nabla c)\cdot\mathbf{n} \qquad \text{on} \quad \Gamma_O = \partial\Omega \setminus \Gamma_D \qquad \text{(Outflow BC),} \tag{7.1e}$$

where c denotes the concentration of a solute. Its transport is described via the flux $\mathbf{j}$, whereas reactions are modeled using the source-sink term q, $\mathbf{n}$ denotes the outer normal to Ω. The flux $\mathbf{j}$ is composed of two parts. The first one is diffusion or dispersion, proportional to the gradient ∇c and parametrized by the diffusion-dispersion tensor $D(\mathbf{u})$. The second part is convection which is proportional to a given velocity $\mathbf{u}$. It is assumed that the system is convection dominated. R is the retardation factor, which describes the deferment of the transport compared to the velocity $\mathbf{u}$ caused by processes like adsorption.

Dirichlet boundary conditions specify a concentration c_0 on the boundary, for Neumann boundaries a flux F is prescribed. The special case $F = 0$ of the Neumann boundary condition is known as the "noflux condition." The outflow boundary condition is applied where the velocity is pointing outwards. It is essentially a "do-nothing" boundary condition and allows the solute to leave the domain freely.

7.1.1 Discretization of the Time-dependent Problem

We will now discuss the discretization of the transport equation (7.1); it can be written as

$$\underbrace{\partial_t R c}_{\text{Term a}} + \underbrace{\nabla \cdot \mathbf{u} c}_{\text{Term b}} - \underbrace{\nabla \cdot D(\mathbf{u}) \nabla c}_{\text{Term c}} = \underbrace{q(c)}_{\text{Term d}} . \tag{7.2}$$

Method of lines is applied to handle time and space derivatives. Space discretization is done using the Unfitted Discontinuous Galerkin method. The DG discretization follows the formulation in [13]. For the elliptic part (7.2c) the NIPG scheme chosen, which allows higher-order discretization and fulfills the discrete local mass conservation. This scheme uses a penalty term for stabilization. The hyperbolic part (7.2b) is discretized using a flux based *Upwind* scheme and does not need additional stabilization. For the time derivative the implicit θ scheme is used.

Velocity Field

The velocity $\mathbf{u}$ is in general obtained from previous computations. We compute the velocity using an Unfitted Discontinuous Galerkin Discretization of the Stokes equations, see Section 6.1.1. Second order polynomial test functions are used, the velocity is extended onto the skeleton Γ (see Definition 2.5) using the average operator

$$\mathbf{u}|_\gamma = \{\, \mathbf{u} \,\}, \quad \gamma \in \Gamma. \tag{7.3}$$

Upwinding

The concentration c is given element-wise. On the skeleton a suitable definition of the solution must be chosen.

It is desired that for a smooth velocity field $\mathbf{u}$ the monotonicity of the continuous problem should be preserved by the discrete solution. According to Godunov's theorem monotonicity can only be guaranteed if the time discretization is first order accurate. When using higher order schemes non-physical oscillations should be kept small. This can be achieved using using *upwinding*. Historically, the origin of upwind methods can be traced back to the work of Courant, Isaacson, and Reeves who proposed the CIR method [38].

Definition 7.1 (Upwind solution): *The solution $c|_\gamma$ on $\gamma \in \Gamma$ is given following the*

streamlines. For $\gamma_{ef} \in \Gamma_{int}$ the upwind solution c^* is given as

$$c|_{\gamma_{ef}} = c^*(x) = \begin{cases} c|_e(x) & \text{if } \mathbf{u}|_{\gamma_{ef}} \cdot \mathbf{n}_e > 0 \\ c|_f(x) & \text{else,} \end{cases} \tag{7.4}$$

where e and f denote the two elements sharing γ_{ef}. For $\gamma \in \Gamma_D$ the upwind solution is

$$c|_\gamma = c^* = \begin{cases} c & \text{if } \mathbf{u}|_\gamma \cdot \mathbf{n} > 0 \\ c_0(x) & \text{else.} \end{cases} \tag{7.5}$$

Space Discretization

The space discretized Problem reads: Find c such that

$$\partial_t m(c, v) + a(c, v) + J(c, v) = l(v) \qquad \forall v \in V, \tag{7.6a}$$

with the bilinear forms a and m, the penalty term J, and the linear form l. Contributions to $a(c, v)$ and to $l(v)$ origin in the elliptic (a_i, b_1) as well as the hyperbolic part (a_2, b_2). The time derivative term (7.2a) is denoted by $m(\cdot, \cdot)$.

The elliptic part is discretized using the NIPG scheme. The discretization of the elliptic part (7.2c) yields

$$\begin{aligned} a_1(c, v) = &\int_\Omega D\nabla c \cdot \nabla v \, dx - \int_{\Gamma \setminus \Gamma_N} [\![c]\!] \cdot \{ D\nabla v \} \, ds \\ &+ \int_{\Gamma \setminus \Gamma_N} [\![v]\!] \cdot \{ D\nabla c \} \, ds - \int_{\Gamma_D} v D\nabla c \cdot \mathbf{n} \, ds \end{aligned} \tag{7.7}$$

with the penalty terms

$$J(c, v) = \eta \int_{\Gamma \setminus \Gamma_N} h_\gamma^{-1} [\![c]\!] [\![v]\!] \, ds - \eta \int_{\Gamma_D} h_\gamma^{-1} c_0 \, v \, ds \,. \tag{7.8}$$

The contribution of the elliptic part to the right hand side is

$$l_1(v) = \int_\Omega q v \, dx + \int_{\Gamma_D} c_0 D\nabla v \cdot \mathbf{n} \, ds \,. \tag{7.9}$$

The discretization of the hyperbolic part is obtained straight forward. In Equation (7.2b) partial integration and Gauss theorem is applied, the $c|_\gamma$ and $\mathbf{u}|_\gamma$ are substituted according to (7.4), (7.5), and (7.3):

$$a_2(c, v) = -\int_\Omega \mathbf{u} \, c \, \nabla v \, dx + \int_{\Gamma \setminus \Gamma_N} [\![v]\!] \cdot \{ \mathbf{u} \} \, c^* ds. \tag{7.10}$$

The Neumann boundary condition contributes to the right hand side

$$l_2(v) = -\int_{\Gamma_N} Fv\, ds\,. \tag{7.11}$$

Remark 7.1: In the discrete problem the outflow boundary Γ_O is not handled explicitly. On $\Gamma \setminus \Gamma_N$ the upwind decision does automatically switch to the outflow boundary condition where $\mathbf{u} \cdot \mathbf{n} > 0$.

Time Discretization

The space discretization allows higher-order ansatz functions. We use a second order approximation in space, hence the time discretization should be second order as well. Further we are using the *one-step-θ*-scheme which can be first and second order accurate, see [15].

Definition 7.2 (one-step-θ-scheme): *Given a linear ordinary differential equation*

$$\frac{d}{dt}x = L(t, x), \tag{7.12}$$

and an initial value $x(0) = x_0$, the solution x on the time interval $[0, T]$ is requested. The interval is subdivided into $0 = t^0 < t^1 < \cdots < t^M = T$ with $\Delta t^n = t^{n+1} - t^n$. The approximation of x^{n+1} by the one-step-θ-scheme is given as

$$x^{n+1} - \Delta t^n\,(1-\theta)L(t^{n+1}, x^{n+1}) = x^n + \Delta t^n\,\theta L(t^n, x^n)\,. \tag{7.13}$$

The one-step-θ-scheme is parametrized by the coefficient $\theta \in [0, 1]$. For $\theta = 0$ the explicit Euler scheme is obtained, for $\theta = 1$ the scheme is identical to implicit Euler. The scheme with $\theta = 1/2$ is known as *Crank-Nicolson* and is second order accurate.

Definition 7.3 (Courant number): *The courant number*

$$C = \frac{\mathbf{u}\Delta t}{h} \tag{7.14}$$

measures the traveling distance in a single time step compared with mesh width.

The explicit Euler scheme is only stable if the Courant–Friedrichs–Lewy (*CFL*) condition $C < 1$ is fulfilled. For the UDG scheme the size of a single element is not bound from below. The size h of most elements will be equal to the mesh width of the fundamental mesh, but the size of some elements can be several orders of magnitude smaller. To fulfill the CFL condition the time step Δt becomes very small. This renders the explicit Euler scheme impractical for our simulations.

The implicit Euler and the Crank-Nicolson method are unconditionally stable, the Courant number is not bound.

7.2 Microscopic Solute Transport

In this example microscopic transport of an inert solute is computed. Hence the retardation factor R is 1 and the dispersion tensor D reduces to the scalar molecular diffusion coefficient. Although the implementation is dimension-independent, the presented results confine to the two-dimensional case.

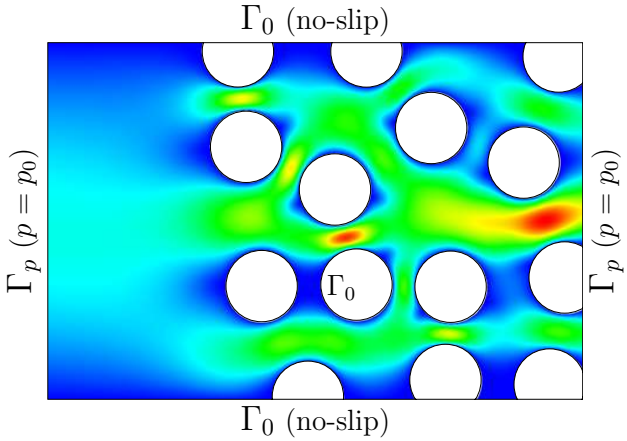


Figure 7.1: Velocity field obtained by solving Stokes equations. Boundary conditions and velocity magnitude.

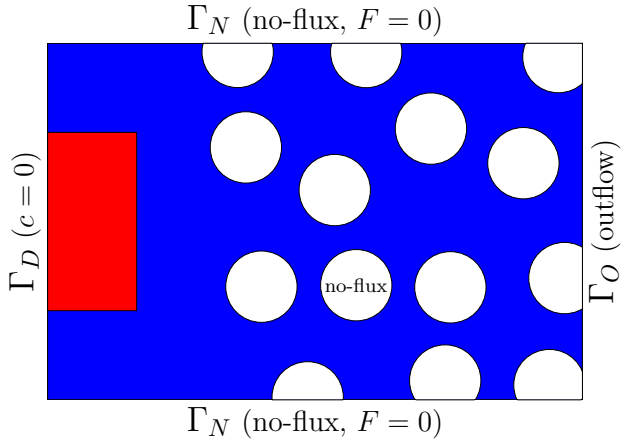


Figure 7.2: Setup for the inital value problem. Boundary conditions and initial concentration.

The concentration of the solvent does strongly depend on the local geometry. The setup is a channel going from left to right with spherical obstacles inside the channel (see Figure 7.2). The velocity field for the convection is given by the solution of the Stokes equation (6.1) on the pore scale. The Stokes equation is discretized using for-

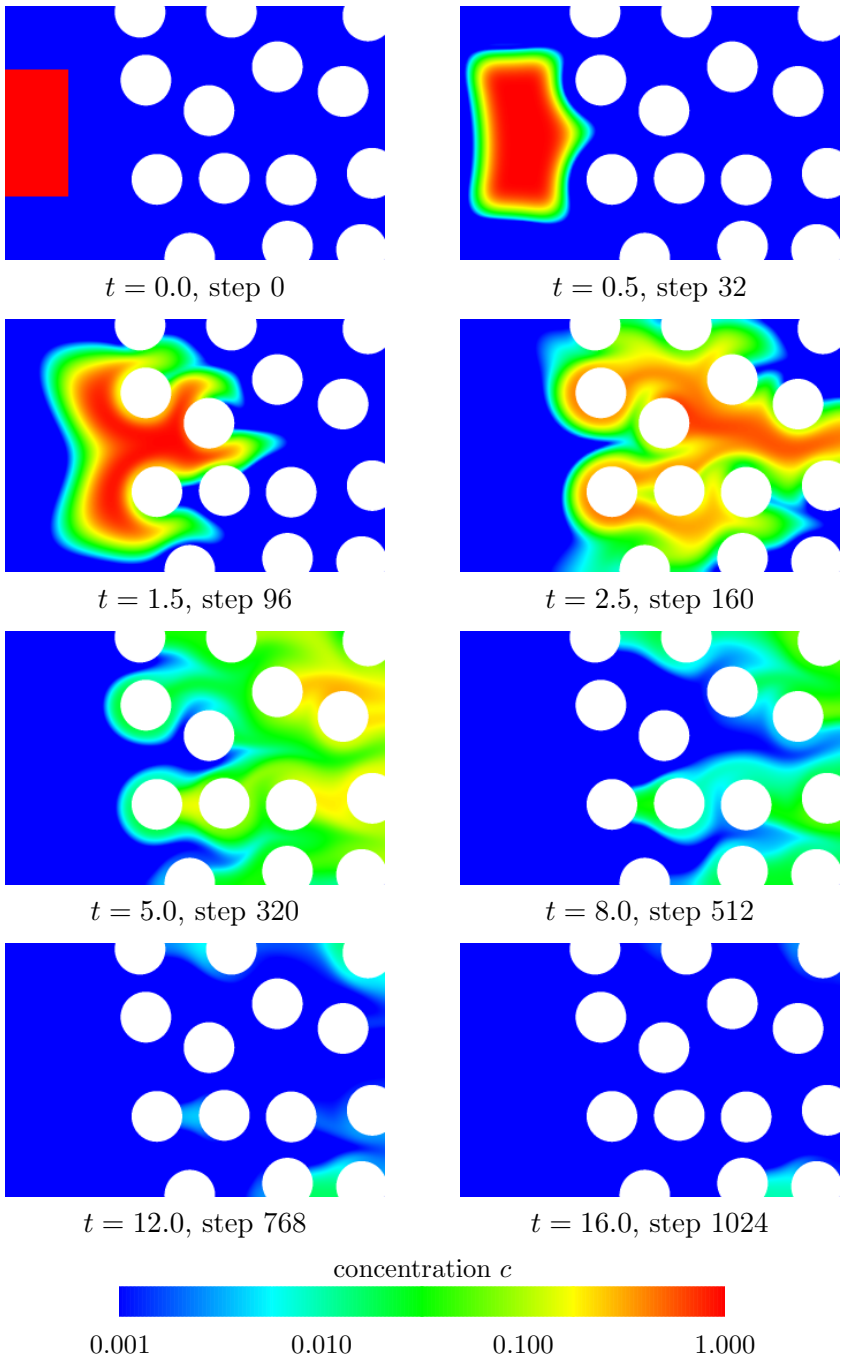


Figure 7.3: Solute transport on a domain with holes, 1024 time steps with $\Delta t = 0.015625$. Computations on a 64×96 mesh, the geometry given implicitly on a twice refined grid (256×384).

mulation (6.4) from the previous chapter. The non-symmetric version without penalty is used. Boundary conditions are no-slip boundary conditions on the upper and lower boundary of the channel and on the surface of the obstacles. On the left and right boundary Neumann boundary conditions with a prescribed pressure are applied. The magnitude of the resulting velocity field is shown in Figure 7.1.

The boundary conditions for the transport equation are no-flux conditions on the upper and lower edge and on all obstacles, outflow boundary condition on the right and Dirichlet boundary condition on the left. These are chosen consistently to the boundary conditions of the velocity field.

The time discretization uses $\theta = 0.5$, i. e. Crank-Nicolson. Although the method is unconditionally stable it must be noted that it is not reasonable to take very large time steps as the approximation error increases. The errors arising from spatial and temporal discretization are balanced for a Courant number of about 1. We decided to choose a time step so that the CFL condition $\mathbf{u}\Delta t < h$ is fulfilled on the fundamental mesh.

The temporal evolution of the plume is visualized in Figure 7.3. The computations are carried out on a 64×96 mesh, while the geometry given as a scalar function on a twice refined grid with 256×384 cells. The maximum concentration in the domain varies across several orders of magnitude throughout the time period.

7.3 Macroscopic Dispersion – Breaktrough curves

A common approach to measure the macroscopic dispersion experimentally are breakthrough curves [22]. The breakthrough curve measures the temporal evolution of the relative flux of the solute.

Definition 7.4 (Relative Flux): *The relative flux of a solute through a plane A is given as*

$$f = \frac{\int\limits_A \mathbf{j}\, da}{\int\limits_A \mathbf{u}\, da}\,. \tag{7.15}$$

Evaluating the flux f for every time step t on the outflow boundary $A = \Gamma_D^{\text{out}}$ yields the breakthrough curve. For the setup of Section 7.2 the breakthrough curve was computed, see Figure 7.4.

For different time steps ($t = 640, 768, 896, 1152, 1408$) the flux through the outflow boundary is computed for different resolutions of the fundamental mesh. The coarsest mesh has a resolution of 2×3. At this resolution it isn't even possible to resolve the initial conditions and hence the results are not usable. A resolution of 16×24 cells is necessary to obtain reasonable results. Comparing the results for different time steps we observe grid convergence.

Figure 7.4 shows the breakthrough curves, the flux relative to the best solution ($f_h / f_{h_{\min}}$) at different time steps, and the relative error ($\|f_{h_{\min}} - f_h\| / f_{h_{\min}}$) at different

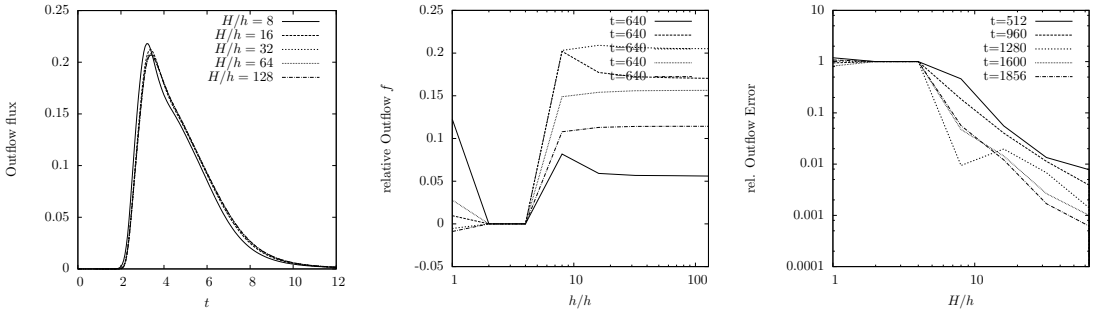


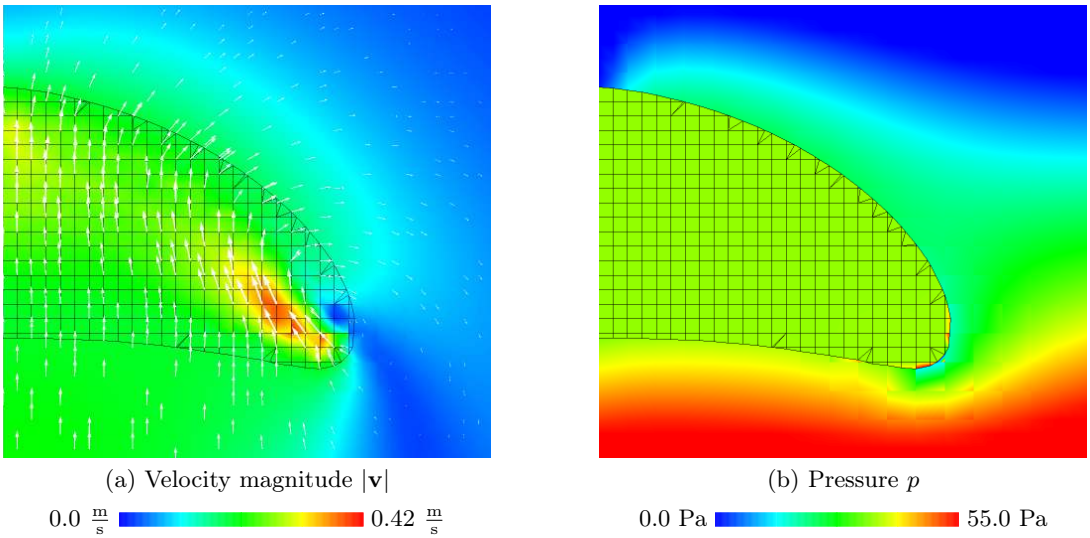
Figure 7.4: Breakthrough curve (left) computed with different fundamental meshes. Grid convergence for different time steps $t = 640, 768, 896, 1152, 1408$. Middle: flux relative to the best solution ($f_h / f_{h_{\min}}$). Right: relative error of flux ($\|f_{h_{\min}} - f_h\| / f_{h_{\min}}$)

time steps.

7.4 Discussion

We have shown the application of the UDG method to the convection–diffusion–reaction equation. Detailed simulations on the pore scale as well as measurements of macroscopic quantities like breakthrough curves are possible. The size h of the smallest element in a UDG mesh is not bounded from below, thus explicit time stepping schemes are not suitable as the maximum time step is determined by the CFL condition $\mathbf{u}\Delta t < h$. The relative solute flux through the outflow boundary was computed and grid convergence for this macroscopic property is observed.

An other class of time-dependent problems are evolving domain problems. In the diploma thesis of Heimann [61] the UDG method is successfully used for the simulation of incompressible viscous two-phase flow with surface tension. The discontinuous formulation allows both an accurate representation of the surface tension induced discontinuities in the pressure field and an explicit reconstruction of the interface. The fluid's interface is tracked numerically with the Level Set method and local integration is done using the algorithm for implicitly described domains (Section 3.2.3). Computational results for two known test problems, oscillating bubble and rising bubble (see Figure 7.5), are compared to experimental and theoretical predictions. The measured oscillation frequency and the velocity of the rising bubble are in good agreement with the predicted values [61].



(a) Velocity magnitude $|\mathbf{v}|$

(b) Pressure p

Figure 7.5: Evolving domain problem: rising bubble. For a given setup the velocity and pressure for a rising bubble are computed [61]. The rate of ascent of the bubble is compared to a reference solution.

Chapter 8

Summary and Discussion

Bei wissenschaftlichen Streitigkeiten nehme man sich in Acht, die Probleme nicht zu vermehren.
— Johann Wolfgang von Goethe

In this work a new approach to simulations on domains with complicated shapes has been developed. The combination of Discontinuous Galerkin methods with an Unfitted Finite Element technique offers a discretization that can accurately represent the complicated geometry, while the dimension of the approximation space remains as low as desired.

The Unfitted Discontinuous Galerkin (UDG) method can easily be applied to Discontinuous Galerkin discretizations of other partial differential equations without changing their primal formulations. The shape of the domain is incorporated during the assembly of the matrix, using appropriate numerical integration techniques. Hence only a modified assembler is required to apply UDG Methods.

In this work the method has been implemented for 2D and 3D, it is based on the DUNE framework [19, 20]. Local triangulation algorithms analytically given domains in 2D and for implicitly given domains in two and three dimensions as been developed. This latter is based on an extension of the well known Marching Cubes algorithm. It offers an efficient way to handle structures obtained from imaging processes as well as time-dependent domains using a level set approach. Furthermore, the algorithm is applicable for the evaluation of integral expressions over implicitly given domains and is not restricted to the UDG method.

The construction of the UDG finite element mesh can lead to degenerated elements.

Numerical experiments show stability of the method even for cases with pathological elements. For an elliptic test problem optimal convergence rates in H^1- and L_2-norm are obtained. Furthermore, super-convergence of the discontinuities was observed. For a locally mass-conservative DG scheme it has been verified numerically that the use of unfitted meshes does not degrade the mass conservation. The application to numerical upscaling has been successfully demonstrated for the example of an effective permeability of a porous medium, using Stokes equations on the pore scale and Darcy's

law on the macroscopic scale. Further work using the UDG method are applications of convection–diffusion problems [50] and two-phase Navier–Stokes flow [61].

Mesh generation for conforming finite elements is improving constantly, still handling complicated domains with small scale structures is an involved process. The construction of a local triangulation has less constraints than the generation of a conforming finite element mesh for the same domain. Hence local triangulation algorithm can be implemented more efficiently. For domains with a simpler boundary mesh generation algorithm work reliable. In these situations the UDG method requires extra effort without much benefit.

Comparison with standard finite elements shows the ability of the method to give reasonable results already on very coarse grids. For the elliptic model problem the discretization error of the DG discretization was always lower than the error of a standard finite element discretization with the same order of convergence and number of unknowns. The computational cost of the matrix assembly is similar to those of an unstructured mesh. However as the fundamental mesh can be rather coarse, the resulting matrix is small.

The local triangulation algorithm for implicitly is fast, but could be optimized further. In particular using, a locally refined image grid for the representation of the level set function describing the domain, as the cost of the assembly is proportional to the size of the image grid. Especially simulations with for a partition $\mathcal{G}$ given by high-resolution image data will benefit from local refinement, as most cells are either completely inside or outside the domain.

In order to take full benefit from this flexibility, it will be necessary to also incorporate local adaptive refinement of the fundamental mesh. On locally refined Cartesian grids with hanging nodes, both local triangulation algorithms can be used without modification and without losing efficiency. Adaptive mesh refinement requires the employment of error estimators. For the computation of macroscopic parameters, the norm of the error of the solution is not of much importance. The interest often lies in an integrated physical quantity; this means that the error estimator should assess the local contributions to the error of the objective functional. Dual weighted a-posteriori error estimators [23] are a promising approach as they allow the optimization of the finite element mesh with respect to an arbitrary error functional. The resolution of the image grid has also a direct influence on the computation time. A coarse image grid increases the model error, while a fine grid increases the computational cost. Braack and Ern [29] worked on the simultaneous estimation of model and discretization error. A similar approach could be followed to simultaneously estimate the discretization- and the geometry-induced error.

A future parallelization of the implementation will the beneficial, especially for time-dependent and three-dimensional problems which have a high demand on computation power. The local triangulation is completely element-local and does not need any information from neighboring processes. Hence the effort for a parallelization is the same as for any other DG method and known parallelization techniques can be used, e. g. domain decomposition methods.

Appendix A

Notation and Symbols

We provide a list of the most important abbreviations technical terms and mathematical symbols. The list contains a short description and a reference to the first appearance, respectively to a detailed explanation.

Abbreviations and Notations

Notation	**Description**	
BCC	Body Centered Cubic array (lattice structure)	73
CAD	Computer Aided Design	28
CFL cond.	Courant–Friedrichs–Lewy condition	90
DG	Discontinuous Galerkin Finite Element Method	7
DUNE	Distributed and Unified Numerics Environment	39
EOC	Experimental Order of Convergence	49
FCC	Face Centered Cubic array (lattice structure)	73
FEM	Finite Element Method	7
FV	Finite Volume Method	7
IP	Interior Panelty Finite Element Method	7
Level set	Scalar function $\phi(x,t)$, describing the temporal evolution of an interface.	30
MC	Marching Cubes algorithm	31
MC33	Marching Cubes 33, topologically correct MC algorithm	33
Micro-CT	Micro X-ray computer tomography	81
NIPG	Non-Symetric Interior Penalty Galerkin Method	16
OBB	Oden-Babuška-Baumann Scheme	16

Notation	Description	
PDE	partial differential equation	4
REV	Representative Elementary Volume	78
SC	Simple Cubic array (lattice structure)	73
SIPG	Symetric Interior Penalty Galerkin Method	16
UDG	Unfitted Discontinuous Galerkin Method	21

Mathematical Symbols

Symbol	Description	
$\{ \cdot \}$	Average of a function, defined on inter-element boundaries.	14
$[\![\cdot]\!]$	Jump of a function, defined on inter-element boundaries.	14
$\lvert \cdot \rvert$	Euclidic norm.	10
$\lVert \cdot \rVert$	Operator norm, or sobolev norm.	10
$\partial\Omega$	Surface of a domain Ω.	8
$\hat{E}$	Reference element.	8
E_i	Element in a mesh.	8
η	Penalty factor for IP or the Nitsche method.	9
G	Boundary of the unit cell U.	71
$\mathcal{G}$	Partition of a domain into sub-domains according to geometric properties.	22
Γ	Skeleton of a partition.	9
γ	Edge/Face in the skeleton of a partition.	8
Γ_{ext}	External skeleton of a partition.	9
Γ_{int}	Internal skeleton of a partition.	8
h	Mesh width.	8
h_g	Mesh width of the image grid.	31
κ	Condition number of a matrix.	50
κ	Permeability of a porous medium.	69
L	Diameter of a domain Ω.	8
μ	Fluid viscosity.	69
$\mathbf{n}$	Outward pointing normal unit vector.	9

Symbol	Description	
Ω	Domain $\Omega \subseteq \mathbb{R}^d$.	8
$\Omega^{(i)}$	Sub-domains, element of a Partition $\mathcal{G}$.	22
Φ	Porosity of a porous medium.	70
ϕ	Scalar function, describing a domain.	30
P_k	Space of polynomial functions of degree k.	13
$\mathcal{T}$	Triangulation (mesh) of a domain.	8
T_{E_i}	Transformation from a reference element to E_i.	8
U	Unit cell for the upscaling process.	71
u	Scalar function.	8
$\mathbf{v}$	Vector valued function.	69

Appendix B

Extended Marching Cubes

The extended Marching Cubes algorithm adds lookup tables for volume triangulation to the classic Marching Cubes lookup tables (see Section 3.2.3).

The MC33 algorithm offers topologically correct reconstruction of the interface. The lookup tables for the surface reconstruction and disambiguation rules are described in [71]. Our implementation uses a different numbering in the unit cube, therefore we show the reference unit cube and the reference simplex (Figure B.1).

Figures B.2 and B.3 show the volume triangulation for the basic cases for simplices and cubes. For the cubes we also note the corresponding disambiguated case number as it is used in [71].

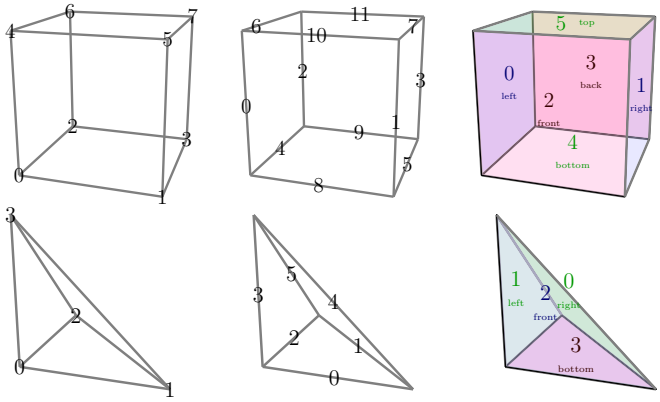


Figure B.1: Reference cube and simplex: Numbering of vertices, edges and faces.

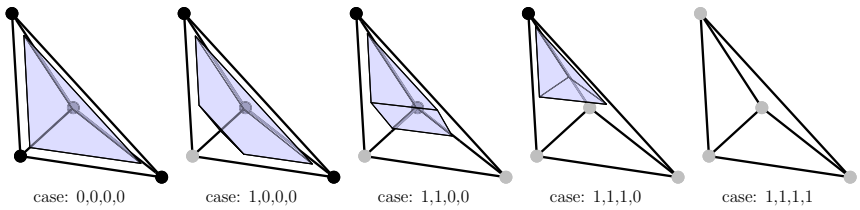


Figure B.2: Basic cases for the triangulation of the 3d simplex.

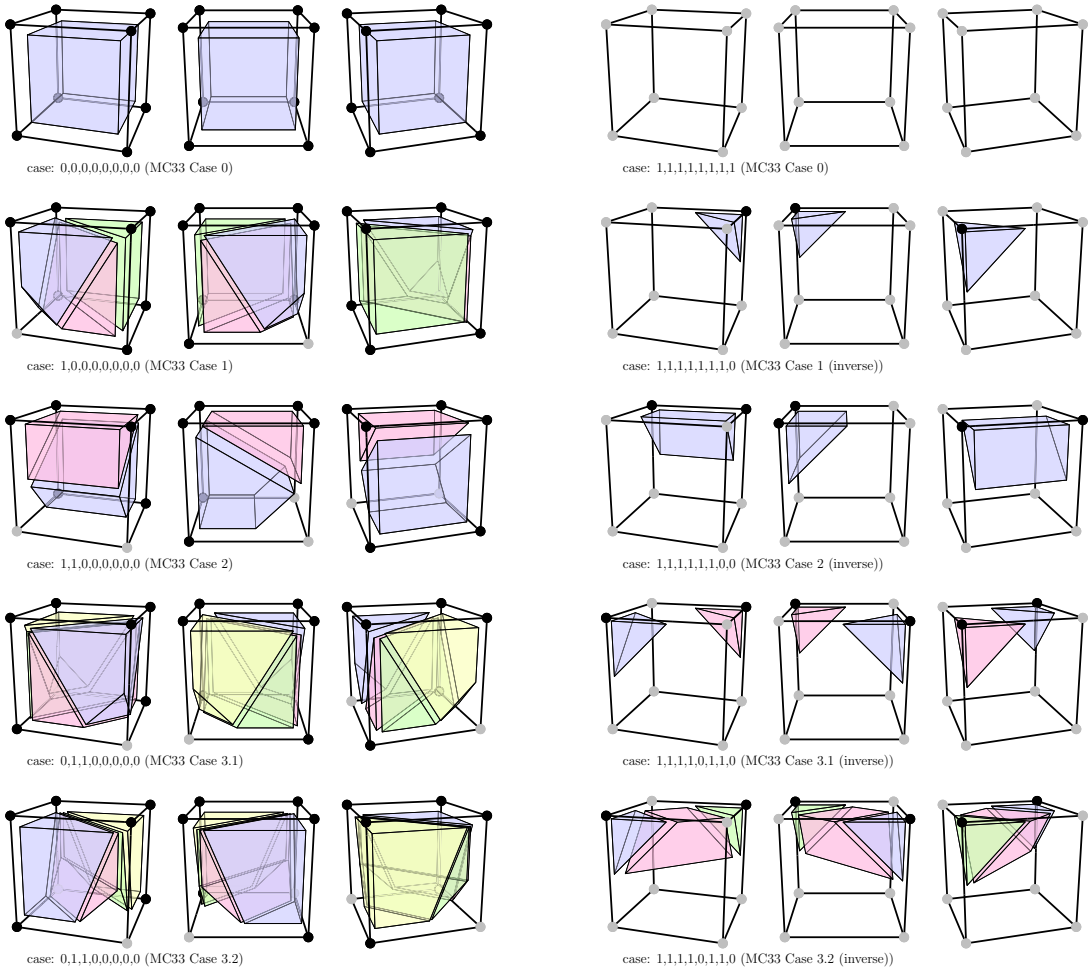


Figure B.3: Basic cases for the triangulation of the 3d cube.

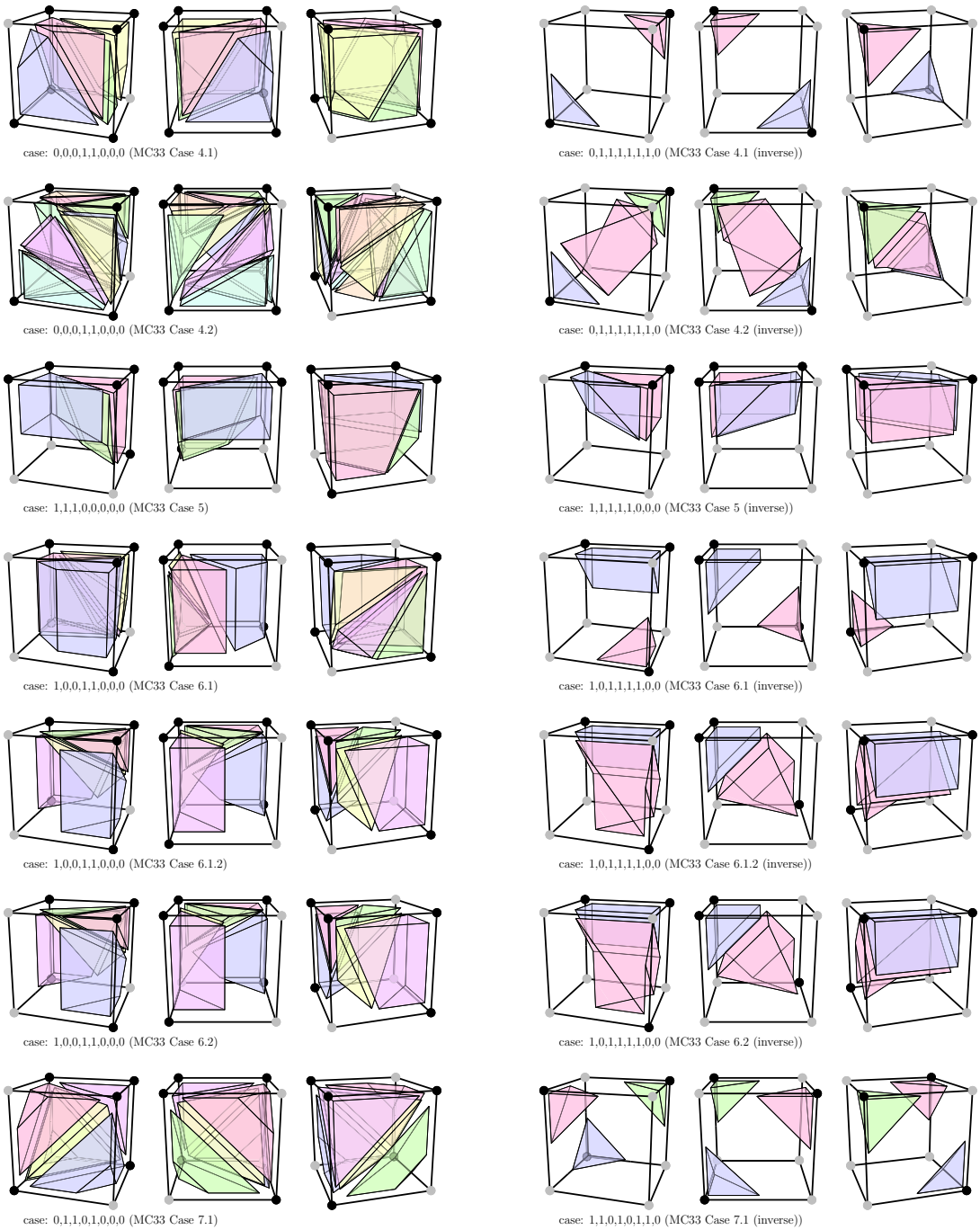


Figure B.3 (continued)

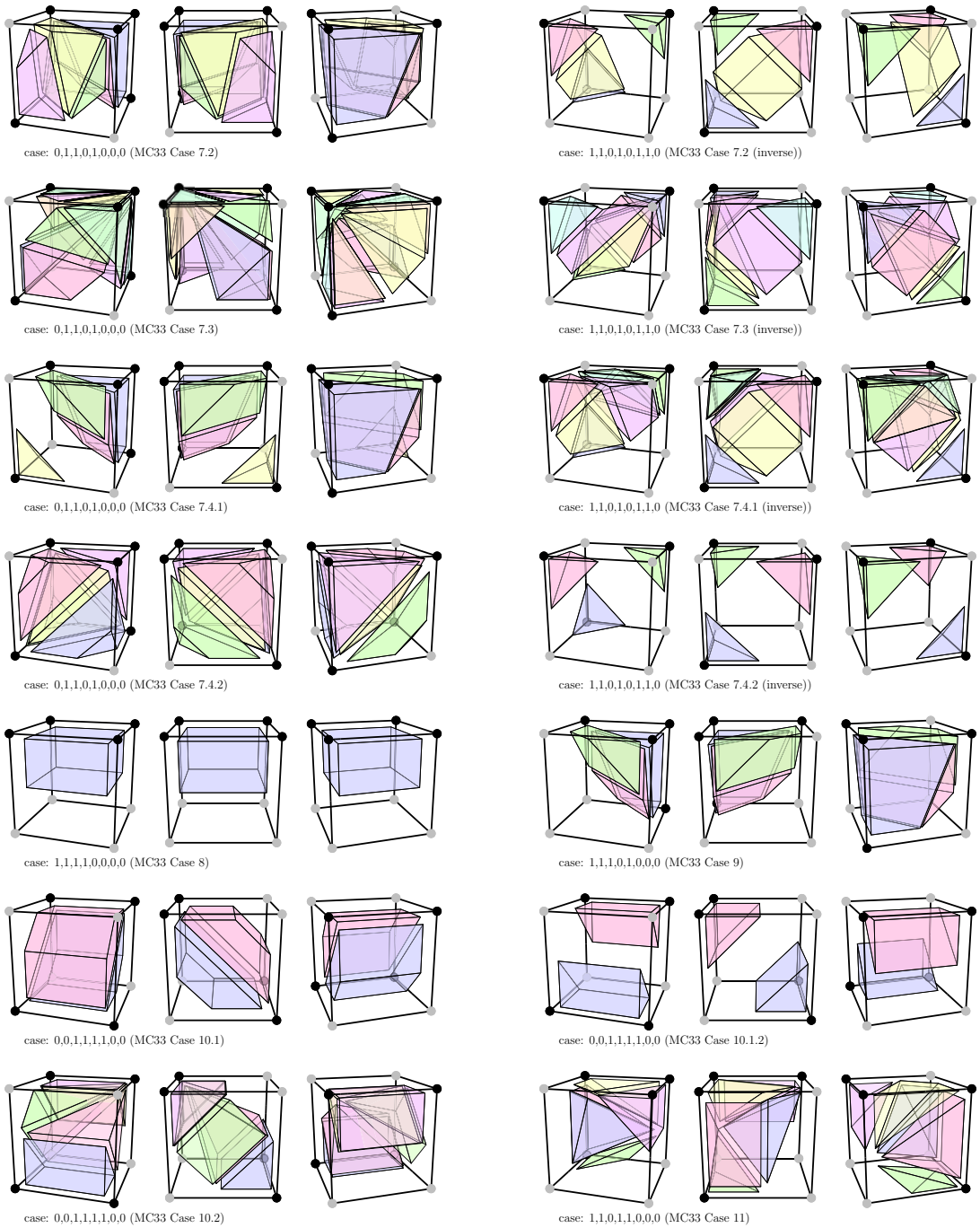


Figure B.3 (continued)

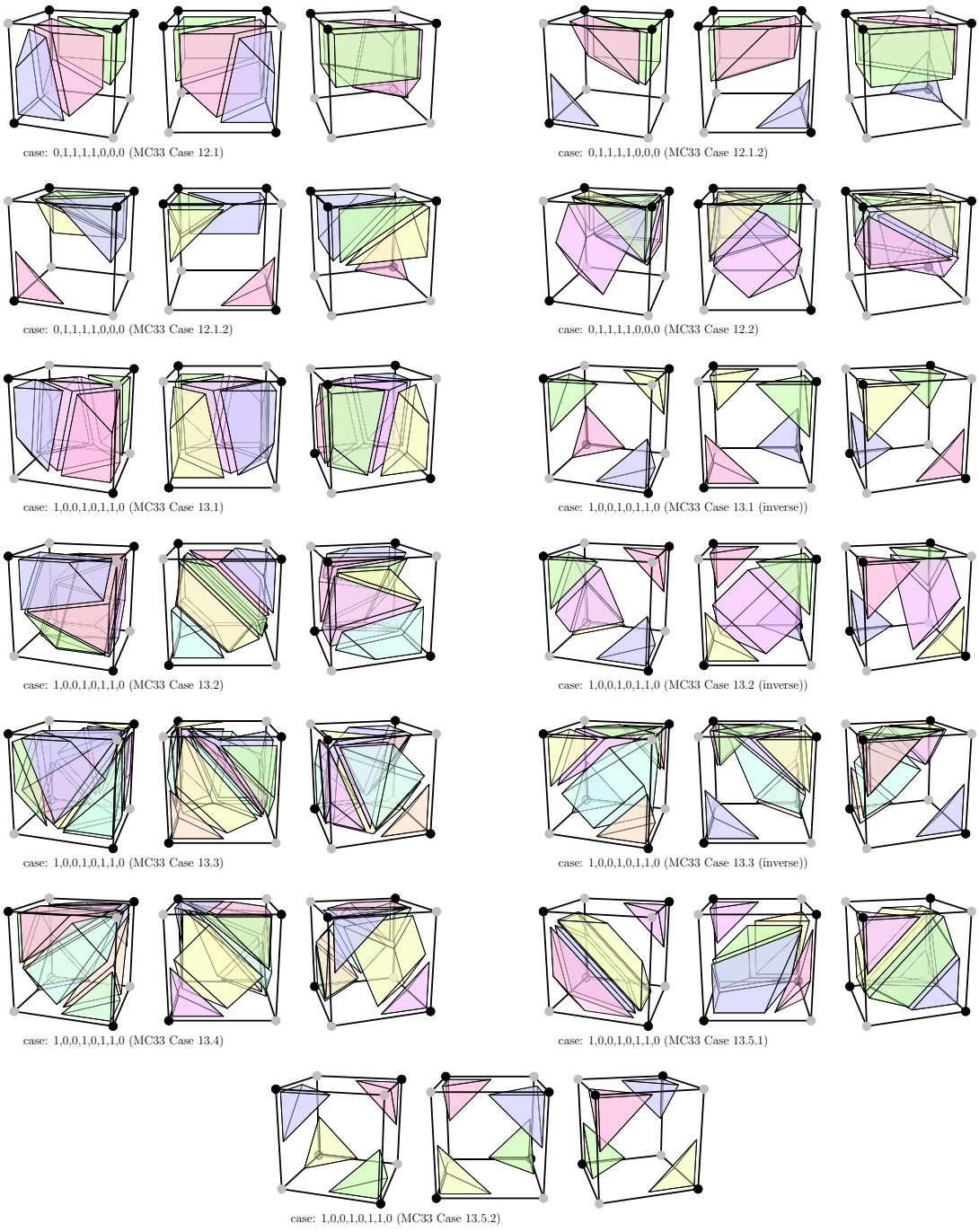


Figure B.3 (continued)

Bibliography

[1] R. A. Adams. *Sobolev spaces*. Academic Press, 1975.

[2] B. Alberts, D. Bray, K. Hopkin, and A. Johnson. *Essential cell biology*. Garland Science, 2003.

[3] D. N. Arnold. *An interior penalty finite element method with discontinuous elements*. PhD thesis, University of Chicago, 1979.

[4] D. N. Arnold. An interior penalty finite element method with discontinuous elements. *SIAM Journal on Numerical Analysis*, 19:742–760, 1982.

[5] D. N. Arnold, F. Brezzi, B. Cockburn, and L. D. Marini. Unified Analysis of Discontinuous Galerkin Methods for Elliptic Problems. *SIAM Journal on Numerical Analysis*, 39(5):1749–1779, 2002.

[6] I. Babuška. Numerical solution of boundary value problems by the perturbed variational principle. Technical Note BN-624, Institute for Fluid Dynamics and Applied Mathematics, Univ. of Maryland, 1969.

[7] I. Babuška and A. K. Aziz. On the Angle Condition in the Finite Element Method. *SIAM Journal on Numerical Analysis*, 13(2):214–226, 1976.

[8] I. Babuška and J. M. Melenk. The partition of unity method. *International Journal for Numerical Methods in Engineering*, 40(4), 1997.

[9] G. A. Baker. Finite Element Methods for Elliptic Equations Using Nonconforming Elements. *Mathematics of Computation*, 31(137):45–59, Jan. 1977.

[10] J. W. Barrett and C. M. Elliott. Fitted and unfitted finite-element methods for elliptic equations with smooth interfaces. *IMA Journal of Numerical Analysis*, 7 (3):283–300, 1987.

[11] R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. V. der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods, 2nd Edition*. SIAM, 1994. URL `http://www.netlib.org/linalg/html_templates/Templates.html`.

[12] F. Bassi and S. Rebay. A High-Order Accurate Discontinuous Finite Element Method for the Numerical Solution of the Compressible Navier-Stokes Equations. *Journal of Computational Physics*, 131(2):267–279, 1997.

[13] P. Bastian. Higher order discontinuous galerkin methods for flow and transport in porous media. In E. Bänsch, editor, *Challenges in Scientific Computing – CISC 2002*, number 35 in LNCSE, pages 1–22, 2003.

[14] P. Bastian and R. Helmig. Efficient fully-coupled solution techniques for two-phase flow in porous media Parallel multigrid solution and large scale computations. *Advances in Water Resources*, 23(3):199–216, 1999.

[15] P. Bastian and S. Lang. Couplex benchmark computations with UG. *Computational Geosciences*, 8(2):125–147, 2004.

[16] P. Bastian and B. Rivière. Superconvergence and H(div) projection for discontinuous Galerkin methods. *International Journal for Numerical Methods in Fluids*, 42(10):1043–1057, 2003.

[17] P. Bastian and B. Rivière. Discontinuous Galerkin Methods for Two-Phase Flow in Porous Media. Technical Report 2004–28, IWR (SFB 359), Universität Heidelberg, 2004.

[18] P. Bastian, M. Droske, C. Engwer, R. Klöfkorn, T. Neubauer, M. Ohlberger, and M. Rumpf. Towards a unified framework for scientific computing. In *Proceedings of the 15th Conference on Domain Decomposition Methods*, Lecture Notes in Computational Science and Engineering, 2004.

[19] P. Bastian, M. Blatt, A. Dedner, C. Engwer, R. K. , M. Ohlberger, and O. Sander. A generic grid interface for parallel and adaptive scientific computing. Part I: Abstract framework. *Computing*, 82(2–3):103–119, 7 2008.

[20] P. Bastian, M. Blatt, A. Dedner, C. Engwer, R. K. , R. Kornhuber, M. Ohlberger, and O. Sander. A generic grid interface for parallel and adaptive scientific computing. Part II: Implementation and tests in DUNE. *Computing*, 82(2–3):121–138, 7 2008.

[21] A. Bayer, H. Vogel, and K. Roth. Direct measurement of the soil water retention curve using X-ray absorption. *Hydrology and Earth System Sciences*, 8(1):2–7, 2004.

[22] J. Bear. *Dynamics of fluids in porous media.* Dover Publications, 1988.

[23] R. Becker and R. Rannacher. An optimal control approach to error estimation and mesh adaptation in finite element methods. *Acta Numerica*, 10:1–102, 2001.

[24] T. Belytschko, N. Moes, S. Usui, and C. Parimi. Arbitrary discontinuities in finite elements. *International Journal for Numerical Methods in Engineering*, 50 (4), 2001.

[25] A. Bensoussan, J. L. Lions, and G. Papanicolaou. *Asymptotic Analysis for Periodic Structures*. North-Holland, 1978.

[26] C. Bernardi, Y. Maday, and A. T. Patera. A new non conforming approach to domain decomposition: The mortar element method. In H. Brezis and J.-L. Lions, editors, *Collège de France Seminar*. Pitman, 1994.

[27] M. Blatt and P. Bastian. The iterative solver template library. In *Applied Parallel Computing. State of the Art in Scientific Computing*, volume 4699 of *Lecture Notes in Computer Science*, page 666. Springer, 2007.

[28] Blender. Free and open source 3D content creation suite, 2009. URL `http://www.blender.org/`.

[29] M. Braack and A. Ern. A posteriori control of modeling errors and discretization errors. *Multiscale Modeling & Simulation*, 1:221, 2003.

[30] D. Braess. *Finite elemente*. Springer, 2003.

[31] J. H. Bramble and S. R. Hilbert. Estimation of Linear Functionals on Sobolev Spaces with Application to Fourier Transformations and Spline Interpolation. *SIAM Journal on Numerical Analysis*, 7(1):112–124, 1970.

[32] B. L. Buzbee, F. W. Dorr, J. A. George, and G. H. Golub. The Direct Solution of the Discrete Poisson Equation on Irregular Regions. *SIAM Journal on Numerical Analysis*, 8(4):722–736, 1971.

[33] E. Bänsch, P. Morin, and R. Nochetto. Finite element methods for surface diffusion. *Free Boundary Problems: Theory and Applications*, page 53, 2004.

[34] A. M. Chapman and J. J. L. Higdon. Oscillatory Stokes flow in periodic porous media. *Physics of Fluids A: Fluid Dynamics*, 4:2099, 1992.

[35] E. V. Chernyaev. Marching cubes 33: Construction of topologically correct isosurfaces. *Institute for High Energy Physics, Moscow, Russia, Report CN/95-17*, 1995.

[36] A. K. Cline, C. B. Moler, G. W. Stewart, and J. H. Wilkinson. An estimate for the condition number of a matrix. *SIAM Journal on Numerical Analysis*, pages 368–375, 1979.

[37] B. Cockburn and C. W. Shu. The local discontinuous Galerkin finite element method for convection-diffusion systems. *SIAM Journal on Numerical Analysis*, 35(6):2440–2463, 1998.

[38] R. Courant, E. Isaacson, and M. Rees. On the solution of nonlinear hyperbolic differential equations. *Comm. Pure Appl. Math*, 5:243–255, 1953.

[39] S. C. Cowin. Bone poroelasticity. *Journal of Biomechanics*, 32(3):217–238, 1999.

[40] H. Darcy. *Les Fontaines de la Ville de Dijon*. Dalmont, Paris, 1856.

[41] P. J. Davis and P. Rabinowitz. *Methods of Numerical Integration: Second Edition*. Dover Publications, November 2007.

[42] K. Deckelnick, G. Dziuk, and C. M. Elliott. Computation of geometric partial differential equations and mean curvature flow. *Acta Numerica*, 14:139–232, 2005.

[43] J. Dolbow, N. Moes, and T. Belytschko. An extended finite element method for modeling crack growth with frictional contact. *Computer Methods in Applied Mechanics and Engineering*, 190(51-52):6825–6846, 2001.

[44] J. E. Dolbow. *An extended finite element method with discontinuous enrichment for applied mechanics*. PhD thesis, Northwestern University, 1999.

[45] V. Dolejsí, M. Feistauer, and V. Sobotíková. Analysis of the discontinuous galerkin method for nonlinear convection-diffusion problems. *Computer Methods in Applied Mechanics and Engineering*, 194(25-26):2709–2733, Jul 2005.

[46] J. Douglas Jr and T. Dupont. Interior penalty procedures for elliptic and parabolic Galerkin methods. *Lecture Notes in Physics*, 58:207–216, 1976.

[47] G. Dziuk and C. M. Elliott. Eulerian finite element method for parabolic PDES on implicit surfaces. *Interfaces and Free Boundaries*, 10:119–138, 2008.

[48] J. C. Elliott and S. D. Dover. X-ray microtomography. *Journal of Microscopy*, 126(2):211, 1982.

[49] C. Engwer and P. Bastian. A Discontinuous Galerkin Method for Simulations in Complex Domains. Technical Report 5707, IWR , Universität Heidelberg, 2005. URL `http://www.ub.uni-heidelberg.de/archiv/5707/`.

[50] J. Fahlke. Pore Scale Simulation of Transport in Porous Media. Master's thesis, Universität Heidelberg, 2008.
URL `http://www.ub.uni-heidelberg.de/archiv/8915/`.

[51] R. P. Fedkiw. Coupling an Eulerian fluid calculation to a Lagrangian solid calculation with the ghost fluid method. *Journal of Computational Physics*, 175(1): 200–224, 2002.

[52] P. Frolkovič and K. Mikula. Flux-based level set method: A finite volume method for evolving interfaces. *Applied numerical mathematics*, 57(4):436–454, 2007.

[53] C. Geuzaine and J. Remacle. Gmsh: a three-dimensional finite element mesh generator with built-in pre-and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 2009. accepted for publication.

[54] F. Gibou, L. Chen, D. Nguyen, and S. Banerjee. A level set based sharp interface method for the multiphase incompressible Navier–Stokes equations with phase change. *Journal of Computational Physics*, 222(2):536–555, 2007.

[55] J. Glimm, H. Kim, D. Sharp, and T. Wallstrom. A stochastic analysis of the scale up problem for flow in porous media. *Computational and Applied Mathematics*, 17:67–80, 1998.

[56] R. Glowinski, T.-W. Pan, and K. Periaux. A fictitious domain method for Dirichlet problem an applications. *Computer Methods in Applied Machanics and Engineering*, 8(4):722–736, 1971.

[57] W. Hackbusch. *Theorie und Numerik elliptischer Differentialgleichungen*. Teubner, 1986.

[58] W. Hackbusch and S. A. Sauter. Composite Finite Elements for the Approximation of PDEs on Domains with complicated Micro-Structures. *Preprint*, 1991.

[59] A. Hansbo and P. Hansbo. An unfitted finite element method, based on Nitsche's method, for elliptic interface problems. *Computer Methods in Applied Mechanics and Engineering*, 191(47-48):5537–5552, 2002.

[60] A. Hansbo and P. Hansbo. A finite element method for the simulation of strong and weak discontinuities in solid mechanics. *Computer Methods in Applied Mechanics and Engineering*, 193(33-35):3523–3540, 2004.

[61] F. Heimann. An Unfitted Discontinuous Galerkin Method for Two-Phase Flow. Master's thesis, Universität Heidelberg, 2009.

[62] A. Helenius. Lecture notes on biochemistry II. Institute of biochemistry, ETHZ, 2001.

[63] J. Heywood, R. Rannacher, and S. Turek. Artificial boundaries and flux and pressure conditions for the incompressible Navier–Stokes equations. *International Journal for Numerical Methods in Fluids*, 22:325–352, 1996.

[64] T. Hildebrand, A. Laib, R. Müller, J. Dequeker, and P. Rüegsegger. Direct Three-Dimensional Morphometric Analysis of Human Cancellous Bone: Microstructural Data from Spine, Femur, Iliac Crest, and Calcaneus. *Journal of Bone and Mineral Research*, 14:1167–1174, 1999.

[65] U. Hornung, editor. *Homogenization and porous media*. Springer-Verlag New York, Inc., New York, NY, USA, 1997.

[66] V. John. A comparison of parallel solvers for the incompressible Navier-Stokes equations. *Computing and Visualization in Science*, 1(4):193–200, 1999.

[67] S. P. Kuttanikkad. *Pore-Scale Direct Numerical Simulation of Flow and Transport In Porous Media*. PhD thesis, Universität Heidelberg, 2009.

[68] M. Křižek. On the maximum angle condition for linear tetrahedral elements. *SIAM Journal on Numerical Analysis*, 29(2):513–520, 1992.

[69] L. Lee and R. J. LeVeque. An immersed interface method for incompressible Navier-Stokes equations. *SIAM Journal on Scientific Computing*, 25:832–856, 2003.

[70] R. J. LeVeque. *Finite volume methods for hyperbolic problems*. Cambridge university press, 2002.

[71] T. Lewiner, H. Lopes, A. W. Vieira, and G. Tavares. Efficient implementation of marching cubes' cases with topological guarantees. *Journal of Graphics Tools*, 8 (2):1–16, 2003.

[72] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 163–169, New York, NY, USA, 1987. ACM Press.

[73] C. Lovadina and L. D. Marini. A-posteriori error estimates for Discontinuous Galerkin approximations of second order elliptic problems. submitted, 2008.

[74] X. J. Luo, M. S. Shephard, R. M. O'Bara, R. Nastasia, and M. W. Beall. Automatic p-version mesh generation for curved domains. *Engineering with Computers*, 20(3):273–285, 2004.

[75] MAPLE. Math and Engineering Software by Maplesoft, 2009. URL `http://www.maplesoft.com/products/maple/`.

[76] N. Moes, J. Dolbow, and T. Belytschko. A finite element method for crack growth without remeshing. *Int. J. Numer. Meth. Engng*, 46:131–150, 1999.

[77] D. R. Musser, G. J. Derge, and A. Saini. *STL Tutorial and Reference Guide*. Addison-Wesley, 2001. ISBN 0-201-37923-6.

[78] J. Nitsche. Über ein Variationsprinzip zur Lösung von Dirichlet-Problemen bei Verwendung von Teilräumen, die keinen Randbedingungen unterworfen sind. *Abh. Math. Sem. Univ. Hamburg*, 36:9–15, 1971.

[79] J. T. Oden, I. Babuška, and C. E. Baumann. A discontinuous *hp*-finite element method for diffusion problems. *Journal of Computational Physics*, 146:491–519, 1998.

[80] H. Okabe and M. J. Blunt. Prediction of permeability for porous media reconstructed using multiple-point statistics. *PHYSICAL REVIEW E Phys Rev E*, 70:066135, 2004.

[81] B. Palsson. The challenges of in silico biology. *Nature Biotechnology*, 18:1147–1150, 2000.

[82] J. Parvizian, A. Düster, and E. Rank. Finite cell method. *Computational Mechanics*, 41(1):121–133, 2007.

[83] C. S. Peskin. Numerical analysis of blood flow in the heart. *Journal of Computational Physics*, 25(3):220–252, November 1977.

[84] C. Pflaum. Subdivision of boundary cells in 3d. Technical report, Mathematische Institute, Universität Würzburg, 2000.

[85] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction*. Springer, 1985.

[86] M. Rech, S. Sauter, and A. Smolianski. Two-scale composite finite element method for Dirichlet problems on complicated domains. *Numerische Mathematik*, 102(4):681–708, 2006.

[87] W. Reed and T. Hill. Triangular mesh methods for the neutron transport equation. Technical Report LA-UR-73-479, Los Alamos National Laboratory, 1973.

[88] B. Rivière and V. Girault. Discontinuous finite element methods for incompressible flows on subdomains with non-matching interfaces. *Computer Methods in Applied Mechanics and Engineering*, 195:3274–3292, 2006.

[89] B. Rivière, M. F. Wheeler, and V. Girault. Improved energy estimates for interior penalty, constrained and discontinuous Galerkin methods for elliptic problems. Part I. *Computational Geosciences*, 3(3):337–360, 1999.

[90] B. Rivière, M. F. Wheeler, and V. Girault. A priori error estimates for finite element methods based on discontinuous approximation spaces for elliptic problems. *SIAM Journal on Numerical Analysis*, pages 902–931, 2002.

[91] A. S. Sangani and A. Acrivos. Slow flow through a periodic array of spheres. *International Journal of Multiphase Flow*, 8:343–360, 1982.

[92] I. F. Sbalzarini, A. Hayer, A. Helenius, and P. Koumoutsakos. Simulations of (an) isotropic diffusion on curved biological surfaces. *Biophysical Journal*, 90(3): 878–885, 2006.

[93] A. Schmidt. Computation of three dimensional dendrites with finite elements. *Journal of Computational Physics*, 125(2):293–312, 1996.

[94] J. Schöberl and W. Zulehner. On Schwarz-type smoothers for saddle point problems. *Numerische Mathematik*, 95(2):377–399, 2003.

[95] J. A. Sethian. *Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science.* Cambridge University Press, 1999.

[96] I. Sevostianov and M. Kachanov. Impact of the porous microstructure on the overall elastic properties of the osteonal cortical bone. *Journal of Biomechanics*, 33(7):881–888, 2000.

[97] M. Skoge, A. Donev, F. H. Stillinger, and S. Torquato. Packing hyperspheres in high-dimensional Euclidean spaces. *Physical Review E*, 74(4):41127, 2006.

[98] M. Skoge, A. Donev, F. H. Stillinger, and S. Torquato. Packing hyperspheres in high-dimensional euclidean spaces. *Physical Review E (Statistical, Nonlinear, and Soft Matter Physics)*, 74(4):041127, 2006.

[99] T. Strouboulis, K. Copps, and I. Babuška. The generalized finite element method. *Computer Methods in Applied Mechanics and Engineering*, 190(32-33): 4081–4193, 2001.

[100] M. Sussman, P. Smereka, and S. Osher. A level set approach for computing solutions to incompressible two-phase flow. *Journal of Computational Physics*, 114(1):146–159, 1994.

[101] S. H. Teng and C. W. Wong. Unstructured mesh generation: Theory, practice, and perspectives. *International Journal of Computational Geometry and Applications*, 10(3):227–266, 2000.

[102] J. C. Van Dam, J. N. M. Stricker, and P. Droogers. Inverse method to determine soil hydraulic functions from multistep outflow experiments. *Soil Science Society of America Journal*, 58(3):647, 1994.

[103] D. Vandevoorde and N. Josuttis. *C++ Templates - The Complete Guide.* Addison-Wesley, 2003.

[104] S. P. Vanka. Block-implicit multigrid solution of Navier-Stokes equations in primitive variables. *J. Comput. Phys.*, 65:138–158, 1986.

[105] H. J. Vogel and K. Roth. Quantitative morphology and network representation of soil pore structure. *Advances in Water Resources*, 24(3-4):233–242, 2001.

[106] VTK. The visualization toolkit, 2009. URL `http://www.vtk.org/`.

[107] S. Weinzierl. Introduction to Monte Carlo Methods. Lecture notes, 2000. URL `http://arxiv.org/abs/hep-ph/0006269/`.

[108] E. W. Weisstein. Connected space. From MathWorld – A Wolfram Web Resource, 2009. URL `http://mathworld.wolfram.com/ConnectedSpace.html`.

[109] M. F. Wheeler. An Elliptic Collocation-Finite Element Method with Interior Penalties. *SIAM Journal on Numerical Analysis*, 15(1):152–161, 1978.

[110] J. Wloka. *Partielle Differentialgleichungen*. Teubner, Stuttgart, 1982.