JOHANNES KEPLER
UNIVERSITÄT LINZ | JKU

TNF

Technisch-Naturwissenschaftliche
Fakultät

Efficient Solution Strategies in Isogeometric Analysis

DISSERTATION

zur Erlangung des akademischen Grades

Doktor

im Doktoratsstudium der

Technischen Wissenschaften

Eingereicht von:
Dipl.-Ing. Stefan K. Kleiss Bakk. techn.

Angefertigt am:
Johann Radon Institute for Computational and Applied Mathematics

Beurteilung:
Dr. Satyendra Tomar (Betreuung)
Univ.-Prof. Dr. Bert Jüttler

Linz, Dezember, 2013

Abstract

Numerical computation of solutions for partial differential equations plays an important role in modern product development and engineering. Also, the use of computer aided design (CAD) software in the design process has become a widespread standard. These two fields are closely connected in the practical development process, but the corresponding techniques have been developed independently over the last decades, and a gap has opened between them. Transferring information between these two fields and processing transferred data to fit the respective requirements can be a very costly procedure in practical applications.

Isogeometric analysis (IGA) aims at closing this gap. By directly using the geometry representation from CAD, the need of transforming geometry data is eliminated. By using underlying non-uniform rational B-splines (NURBS) as ansatz functions for numerical solutions, an initial mesh is obtained automatically, thereby eliminating the need of creating a new mesh of the imported object. Furthermore, one can profit from certain properties of NURBS functions, such as high regularity and NURBS-specific refinement options. In this thesis, two particular aspects, which arise in the course of numerical computations, are addressed in the context of IGA.

In the first part, we consider the situation where a complicated object cannot be represented by a single NURBS geometry mapping, and is thus composed of several subdomains. By applying techniques from finite element tearing and interconnecting methods in isogeometric analysis, we introduce the isogeometric tearing and interconnecting (IETI) method. We discuss requirements for and the realization of C^0-coupling at subdomain interfaces, as well as suitable preconditioners, both for fully-matching settings and for situations with so-called "hanging knots". The latter appear in local refinement methods which are introduced by the IETI method, and which are also discussed in this thesis.

In the second part, we address the issue of quantitative a posteriori error estimation. The presently used error estimation techniques are adapted from classical finite element methods and do not take advantage of IGA-specific properties. Furthermore, these estimators do not provide sharp quantitative error bounds. By applying functional-type a posteriori error estimators in IGA, we

study the realization of error estimators which are fully computable and provide quantitatively sharp error bounds, both from above and below. Furthermore, these estimators also indicate the error distrubution, which can be used for adaptive refinement. In this thesis, we focus upper bounds for the error. We exploit NURBS-specific properties in order to define a time-efficient set-up of the underlying problem, and derive a criterion which indicates the sharpness of the computed bound and the quality of the indicated error distribution.

Kurzfassung

Das numerische Lösen partieller Differentialgleichungen nimmt eine bedeutende Rolle im modernen Produktentwicklungsprozessen ein, wie auch die Verwendung computergestützter Konstruktionsverfahren (computer-aided design, CAD). Während diese beiden Bereiche in der Praxis eng verflochten sind, wurden die ihnen zu Grunde liegenden Methoden im Lauf der vergangenen Jahrzehnte unabhängig voneinander entwickelt. Der Transfer von Daten zwischen den jeweiligen Programmen sowie deren Aufbereitung entsprechend den jeweiligen Anforderungen kann in der Praxis sehr zeitaufwändig sein.

Isogeometric Analysis (IGA) zielt auf einen Brückenschlag zwischen diesen Bereichen ab. Durch die direkte Verwendung der Geometriedarstellung aus CAD-Programmen wird eine Transformation der Geometrie-Daten unnotwendig. Indem Non-Uniform Rational B-Splines (NURBS), die CAD-Darstellungen oft zu Grunde liegen, auch als Ansatzfunktionen für die numerische Lösung verwendet werden, wird automatisch ein erstes, grobes Netz mitgeliefert. Dadurch ist auch der Prozess des Vernetzen der Geometrie nicht mehr notwendig. Weiters ist es möglich, bei numerischen Berechnungen von positiven Eigenschaften von NURBS zu profitieren, zum Beispiel von deren Glattheitseigenschaften oder von NURBS-spezifischen Verfeinerungsmethoden. In dieser Dissertation werden zwei konkrete Themen behandelt, die im Rahmen numerischer Berechnungen auftreten und die hier im Kontext von IGA untersucht werden.

Im ersten Teil der Arbeit widmen wir uns Objekten, die auf Grund ihrer Geometrie nicht mit einer einzigen NURBS-Abbildung dargestellt werden können, sondern über eine Vereinigung von mehreren Teilgebieten definiert sind. Wir wenden die Techniken von FETI-Methoden (Finite Element Tearing and Interconnecting) in IGA an und definieren die Isogeometric Tearing and Interconnecting (IETI) Methode. Wir untersuchen die Voraussetzungen und die Umsetzung einer C^0-stetigen Kopplung an den Übergängen zwischen Teilgebieten, sowie entsprechende Präkonditionierer. Dabei werden auch sogenannte "hängende Knoten" behandelt, die im Rahmen von neuen, durch die IETI Methode ermöglichten lokalen Verfeinerungsmethoden auftreten.

Im zweiten Teil der Arbeit untersuchen wir quantitative a posteriori Fehlerschätzer. Die derzeit in IGA verwendeten a posteriori Fehlerschätzer wurden

von klassichen Finite Elemente Methoden übernommen und nutzen keine der speziellen Eigenschaften von NURBS aus. Weiters liefern diese Fehlerschätzer keinen verlässlichen quantitativen Aussagen. Wir integrieren "Functional-Type" a posteriori Fehlerschätzer in IGA und untersuchen die Umsetzung von Fehlerschätzern, die in dem Sinne berechenbar sind, dass sie keine unbestimmten Konstanten beinhalten, und die quantitativ scharfe obere und untere Fehlerschranken liefern. Weiters zeigen sie die Verteilung des Fehlers an, was für adaptive Netzverfeinerung verwendet werden kann. Wir konzentrieren uns im Rahmen dieser Arbeit auf die Untersuchung oberer Fehlerschranken. Durch die Ausnützung NURBS-spezifischer Eigenschaften erreichen wir, dass das zu Grunde liegende Problem zeiteffizient aufgestellt werden kann. Weiters definieren wir ein Kriterium, das anzeigt, ob die berechnete Fehlerschranke scharf und ob die ermittelte Fehlerverteilung verlässlich ist.

Acknowledgements

First of all, my deep gratitude goes to Dr. Satyendra Tomar for his supervision and guidance during my Ph.D. studies. His scientific advice, his explanations, his patient understanding, and his careful review of reports, papers, and this thesis are highly appreciated. Furthermore, I am thankful for the support and the opportunity to visit so many conferences and workshops. His unconditional loyalty to his Ph.D. students and his way of putting his students' interest above everything else were never taken for granted and still are not taken for granted.

I am thankful to Prof. Dr. Bert Jüttler and Dr. Clemens Pechstein for the countless discussions and their advice, for the knowledge they shared with me, and the time and effort they spent so that I could understand this shared knowledge. I also want to express my gratitude to Prof. Dr. Sergey I. Repin for his helpful discussions and suggestions.

I gratefully acknowledge the nice working atmosphere at the campus of the Johannes Kepler University Linz. I am thankful to my colleagues at the Radon Institute for Computational and Applied Mathematics (RICAM), in particular all current and former members of our group "Computational Methods for Direct Field Problems", and our group leader Prof. Dr. Ulrich Langer. I very much enjoyed the many mathematical and non-mathematical discussions at conferences and at the mensa. Also, I am thankful to all the current and former members of the Institute of Applied Geometry for the friendly and open environment (and the countless coffees and cakes). In particular, I want to thank Monika Bayer for taking care of the hearty atmosphere and for being the anchor that keeps the institute and its quickly changing staff together.

The support from the Austrian Science Fund (FWF) through the project P21516-N18, the European Union through the 7th Framework Programme, project 218536 "EXCITING", and the Austrian Academy of Sciences (ÖAW) are gratefully acknowledged.

Special thanks go to my whole family, for helping me getting where I am today. To my wife Marion I am more thankful than I can possibly put down in words, for giving me energy when mine is running low, for having an open ear for my worries and my successes, for your faith in me, and above all, for your closeness and your love.

Contents

1 Introduction **1**
- 1.1 Motivation . . . 1
- 1.2 Outline . . . 5

2 Preliminaries **7**
- 2.1 Basic notation, function spaces, and norms . . . 7
  - 2.1.1 Computational domain . . . 7
  - 2.1.2 Function spaces and norms . . . 7
- 2.2 Numerical analysis . . . 9
  - 2.2.1 Model problems . . . 9
  - 2.2.2 Variational formulation . . . 10
  - 2.2.3 Existence and uniqueness . . . 11
  - 2.2.4 Galerkin's method . . . 12
  - 2.2.5 A posteriori error estimation and adaptive refinement . . . 13

3 Isogeometric analysis **15**
- 3.1 B-spline basis functions . . . 15
  - 3.1.1 Univariate B-spline basis functions . . . 15
  - 3.1.2 Bivariate B-spline basis functions . . . 21
- 3.2 Isogeometric single-patch discretizations . . . 22
  - 3.2.1 B-spline geometry mappings . . . 22
  - 3.2.2 Refinement of the geometry mapping . . . 25
  - 3.2.3 Non-uniform rational B-splines (NURBS) . . . 26
  - 3.2.4 Isogeometric discretization . . . 27
- 3.3 Further remarks regarding IGA . . . 27
  - 3.3.1 Approximation properties . . . 28
  - 3.3.2 Singular and distorted geometry mappings . . . 28
  - 3.3.3 A posteriori error estimation . . . 29
  - 3.3.4 Local refinement . . . 29

4 IETI - IsogEometric Tearing and Interconnecting **31**
- 4.1 Multi-patch NURBS discretization . . . 31
- 4.2 Solver design . . . 35

4.2.1 Continuity constraints 35
4.2.2 Saddle point formulation 37
4.2.3 Dual-primal formulation 38
4.2.4 Preconditioner 43
4.2.5 Isogeometric tearing and interconnecting algorithm 44
4.3 Refinement options 44
4.3.1 h-refinement on one subdomain 44
4.3.2 Local refinement by substructuring 47
4.3.3 Preconditioning in the presence of hanging knots 50
4.4 Numerical examples 51

5 Functional-type a posteriori error estimates 59
5.1 Guaranteed upper bound of the error 59
5.1.1 Post-processing of u_h 60
5.1.2 Cell-wise interpolation 61
5.1.3 Global minimization 63
5.2 Steps involved in minimizing the majorant 64
5.3 Quality indicator and local error indicator 66
5.4 Efficient computation/implementation 68
5.4.1 Straightforward procedure 69
5.4.2 Alternative cost-efficient procedure 70
5.5 Numerical examples for the upper bound 76
5.6 Guaranteed lower bound of the error 88
5.6.1 Definition and computation 88
5.6.2 Numerical examples for the lower bound 90

6 Summary and discussion 93
6.1 Isogeometric tearing and interconnecting method 93
6.2 Functional-type a posteriori error estimators in IGA 94
6.3 Subjects for further studies 95

Chapter 1

Introduction

The work presented in this thesis was funded and supported by the Austrian Science Fund (FWF) through the project P21516-N18, the European Union through the 7th Framework Programme, project 218536 "EXCITING", and the Austrian Academy of Sciences (ÖAW).

The main results of this thesis, which are presented in Chapters 4 and 5, have been published or are currently under review. The studies of the isogeometric tearing and interconnecting (IETI) method presented in Chapter 4 have been published in [57]. The results for guaranteed and sharp a posteriori error estimates in isogeometric analysis, which are presented in Chapter 5, can be found in the technical report [58].

1.1 Motivation

Numerical computation of solutions for partial differential equations (PDEs) plays an important role in the design process in modern engineering. A large variety of physical processes, for example, heat transfer, structural mechanics, fluid dynamics, or electromagnetics, can be modelled by PDEs. However, in general, it is not possible to find analytic, i.e., exact solutions for practical problems. Instead, approximate solutions are obtained by conducting numerical simulations on computers. These numerical computations are of large importance, in particular if real-life tests are costly in terms of time, effort, or money. Over the past decades many different techniques have been developed, such as, for example, the finite difference method (FDM), the finite element method (FEM), the finite volume method (FVM), and the boundary element method (BEM). The geometry representations used in these methods can differ, since they have been developed to fit the specific needs of the respective method. In this thesis, we will follow the FEM approach.

At the same time, it has become standard to use computer aided design (CAD) software in the design process. CAD provides high functionality and flexibility for the designer, but the geometry representation used in CAD differs greatly from geometry descriptions used in numerical computations. This means that numerical methods cannot directly use geometry data exported from CAD software. Thus, the full

process of numerical computation and quality assurance involves three major tasks.

1. Transformation of the geometry representation from CAD description to a FEM-suitable representation.
2. Computation of a numerical solution for the problem of interest.
3. Evaluation of the quality of the numerical solution.

We will now discuss these three tasks individually in more detail (the following discussion can also be found in similar versions in [57, 58]).

Task 1: Geometry data transformation

The transformation of CAD geometry description to typical FEM geometry representation, i.e., meshing the computational domain, does not only necessitate suitable software, but often also requires manual input. While both FEM and CAD have been under constant development individually, bridging these two fields has become a bottleneck. When complicated objects from practical problems have to be meshed, this manual work can amount in many man-hours and thus also comes at large financial cost.

The concept of *isogeometric analysis (IGA)*, introduced by Hughes et al. in 2005 [45], see also [27], is a concept that establishes a close link between the technologies of CAD and FEM. It takes advantage of the facts that, firstly, it is a common standard in CAD to use spline representations based on non-uniform rational B-splines (NURBS), and that, secondly, these NURBS basis functions have properties which make them suitable as FEM ansatz functions. By directly using the geometry provided from CAD, the need for data transformation is eliminated, and computations can be conducted on the original, unchanged domain. This exact geometry representation can also be preserved throughout refinement. Chapter 3 of this thesis is dedicated to a discussion of the fundamentals of IGA as far as they are relevant for this thesis.

Numerous studies show that IGA can be successfully applied to various problems, such as, e.g., structural mechanics and elasticity [3, 4, 5, 17, 28, 29, 36, 46], electromagnetics [23, 24], fluid dynamics [10, 71], and fluid-structure-interaction [11, 12]. Theoretical issues such as error estimates and convergence rates [8, 14], stability issues [77, 91, 92, 93], and numerical quadrature rules [6, 47] have also been studied thoroughly.

Task 2: Efficient computation of the numerical solution

Efficient implementation of isogeometric methods on single-patch domains has been thoroughly studied in many publications (see the references listed above). Solvers and preconditioners for single-patch IGA have also been studied in [15, 16, 22, 42, 43].

In practical applications, however, situations where it is not possible to represent complicated objects by only one NURBS mapping occur frequently. In such

situations, the computational domain may be composed of several NURBS patches, and even though the overall geometry may be complicated, the structure within one subdomain remains highly regular.

In [13], it was discussed how to merge two-patch geometries with T-splines such that one global domain is obtained. In contrast to this approach, the *isogeometric tearing and interconnecting (IETI)* method (to be pronounced ['jɛtɪ], like Yeti) is proposed in Chapter 4 (see [57]). The multi-patch structure is preserved and the technology from finite element tearing and interconnecting (FETI) methods is applied within the isogeometric framework.

FETI methods are powerful solvers for large-scale finite element systems. They were introduced by Farhat and Roux [41] and belong to the class of iterative substructuring methods (also called non-overlapping domain decomposition methods), see, e.g., [95]. In the FETI-approach, the computational domain is given as one global geometry, which is then subdivided into non-overlapping subdomains. In contrast to this, we assume for the IETI method that the computational domain is already given as a composition of non-overlapping subdomains. In both cases, each subdomain gets its own set of equations derived from the global equation. To ensure the equivalence to the global equation, continuity conditions are introduced at the interfaces between subdomains using Lagrange multipliers.

This tearing and interconnecting method, however, is not only a coupling method but provides a powerful solver design. By (carefully) eliminating the original variables from the resulting saddle point problem, one obtains a system in the Lagrange multipliers (i.e., only on the interface). The solution of the original problem can be easily computed from the solution of this interface problem. For generalizations to boundary element discretizations see, e.g., [60, 61, 73]. For generalizations to spectral element discretizations see, e.g., [51]. For a thorough study of FETI and BETI methods for multiscale problems, see [75] and the monograph [74].

Since the number of Lagrange multipliers is typically large, the interface problem is usually solved iteratively by FETI preconditioned conjugate gradients [86]. Suitable preconditioners have been proposed in [41, 52, 53]. The analyses in [53, 66] show that under suitable conditions the condition number of the preconditioned system is bounded by $C(1+\log(H/h)^\gamma)$, $\gamma \leq 3$, where H is the subdomain diameter and h is the mesh size. This results in quasi-optimal complexity of the overall method. Recently, it has been shown in [15] that this bound also holds true for BDDC preconditioners in IGA.

We mention that the classical FETI method is formulated as a two-grid method, and thus involves the solution of a coarse system. An efficient alternative is the dual-primal FETI (FETI-DP) method, see [39, 54, 55, 67], which is followed in this thesis. Since FETI-DP methods and BDDC methods have the same essential spectrum (except for zeros and ones), it is expected that the above mentioned bound for the condition number of the preconditioned system also holds for the IETI-DP method.

Task 3: Sharp estimation of the unknown error

Once a numerical solution has been computed, it is important to assure the quality of this solution. Since, typically, the exact solution is *not* known, it is not possible to compute the exact error. Hence, it is necessary to find *computable bounds* for the unknown error in order to quantify the accuracy of the numerical solution. Despite the importance of this issue, a posteriori error estimation in isogeometric analysis is still in an infancy stage. To the best of the author's knowledge, the only published results are [33, 49, 97, 98, 99]. In [33, 97], the authors used the a posteriori error estimates based on hierarchical bases [7]. Its reliability and efficiency is subjected to the saturation assumption on the (enlarged) underlying space and the constants in the strengthened Cauchy inequality. As the authors remarked, the first assumption is critical and its validity depends on the considered example. Moreover, an accurate estimation of constants in the strengthened Cauchy inequality requires the solution of generalized minimum eigenvalue problem. As noted in [49], this approach delivers less than satisfactory results. In [49, 98, 99], the authors used the residual-based a posteriori error estimates, which require the computation of constants in Clement-type interpolation operators. Such constants are mesh (element) dependent, often generic/unknown or incomputable for general element shape; and the global constant often over-estimates the local constants, and thus the exact error. This fact has been explicitly stated by the authors in [49] and in [98]. Furthermore, Zienkiewicz-Zhu type a posteriori error estimates are based on post-processing of approximate solutions, and depend on the superconvergence properties of the underlying basis. To the best of our knowledge, superconvergence properties for B-splines (NURBS) functions are not yet known. Summarily, in general situations, the reliability and efficiency of these methods often depend on undetermined constants, which is not suitable for quality assurance purposes.

In Chapter 5 of this thesis, a *functional-type* a posteriori error estimator for isogeometric discretizations is presented (see [58]). These error estimates, which were introduced in [79, 80, 82] and have been studied for various fields (see the monograph [84] and the references therein), provide guaranteed, sharp, and fully computable bounds (without any generic undetermined constants). These estimates are derived on purely functional grounds (based on integral identities or functional analysis) and are thus applicable to any conforming approximation in the respective space. Functional-type a posteriori error estimates provide two-sided bounds of the unknown error, and thus provide a guaranteed interval containing the true error. In this thesis, we mention lower bounds only briefly and focus on the efficient realization of upper bounds of the error in IGA.

For elliptic problems with the weak solution $u \in H^1_0(\Omega)$, these upper bounds involve computing a free function $y \in H(\Omega, \mathrm{div})$. In order to get a sharp estimate, this function y is computed by solving a *global* problem. This could be perceived as a drawback when compared to error estimation techniques which rely on local computations and are thus apparently cheaper. However, we stress that functional-type error estimates do not only provide an *error indicator* which can be used for

cell marking in the course of adaptive refinement, but also *quantify the error in the computed solution* (and thus guarantee the quality of the computed solution). Therefore, the associated cost should be weighed against the importance of these two aspects. To the best of our knowledge, there is no other, particularly cheaper, method available which can fulfill these objectives in general situations.

In Chapter 5, we will elaborate on how such an upper bound of the error can be computed efficiently by a proper set-up of the global problem. Two aspects motivate the application of functional-type error estimates in IGA: Firstly, unlike the standard Lagrange basis functions, NURBS basis functions of degree p are, in general, globally C^{p-1}-continuous. Hence, NURBS basis functions of degree $p \geq 2$ are, in general, at least C^1-continuous, and therefore, their gradients are automatically in $H(\Omega, \mathrm{div})$. Thereby, we avoid constructing complicated functions in $H(\Omega, \mathrm{div})$, in particular for higher degrees (see, e.g., [21, 23, 24, 38]). Secondly, since the considered problem is solved in an isogeometric setting, an efficient implementation of NURBS basis functions is readily available, which can be used to construct the above mentioned function y. Hence, applying the technique of functional-type a posteriori error estimation in a setting that relies only on the use of already available NURBS basis functions is greatly appealing.

1.2 Outline

The remainder of this thesis is organized as follows.

- In Chapter 2, some basic, but important definitions and concepts are recalled, and we specify the considered model problems. This is done for the sake of completeness and in order to fix the used notations.

- In Chapter 3, the main aspects of IGA are recalled, in particular those relevant for the scope of this thesis.

- In Chapter 4, which is based on [57], the isogeometric tearing and interconnecting (IETI) method (see discussion of Task 2 above) is introduced. The coupling at subdomain interfaces and requirements on the discretization are discussed, as well as a suitable preconditioner for the interface problem. New options for local refinement, which are introduced by the IETI method and which rely only on tensor product basis functions, are also elaborated.

- In Chapter 5, which is based on [58], functional-type a posteriori error estimators for IGA are presented (see discussion of Task 3 above). The focus is set on estimating the error from above, which is discussed in the light of isogeometric discretizations. Special NURBS properties are exploited in order to obtain an efficient set-up. The accuracy of the computed bounds and the effort for obtaining these bounds are investigated. Furthermore, a criterion for the quality of the error bound, both with respect to the magnitude and the distribution of

the error, is presented. Functional-type lower error bounds are briefly addressed and a proof-of-concept is presented in a basic, straightforward set-up.

- In Chapter 6, the presented results are summarized and discussed, including the recommendation for further studies.

Numerical examples illustrating the potential of the methods introduced in Chapters 4 and 5 are presented within the respective chapters.

Chapter 2

Preliminaries

In this chapter, some well-known definitions and results are recalled in order to fix the notations used in this thesis, and for later reference. These results and definition can be found in many standard references, such as, for example, the monographs [18, 19, 26, 74, 84, 95].

2.1 Basic notation, function spaces, and norms

2.1.1 Computational domain

Throughout this thesis, let $\Omega \subset \mathbb{R}^2$ be a non-empty, open, bounded and connected Lipschitz domain with boundary $\Gamma = \partial\Omega$. Let Γ be composed of two disjoint sets, namely Γ_0 and Γ_1, i.e.,

$$\Gamma = \Gamma_0 \cup \Gamma_1 \quad \text{and} \quad \Gamma_0 \cap \Gamma_1 = \emptyset.$$

On Γ_0, essential (Dirichlet or displacement) boundary conditions are prescribed. On Γ_1, natural (Neumann or traction) boundary conditions are prescribed.

When a domain Ω is composed of N disjoint subdomains, we denote these subdomains by the superscript index in parenthesis, i.e.,

$$\overline{\Omega} = \bigcup_{k=1}^{N} \overline{\Omega^{(k)}}, \text{ where } \Omega^{(k)} \cap \Omega^{(\ell)} = \emptyset, \text{ for } k \neq \ell, \tag{2.1}$$

and where $\overline{\Omega}$ denotes the closure of Ω. This will be used in particular in Chapter 4.

2.1.2 Function spaces and norms

Let $\Omega \subset \mathbb{R}^2$ be as described above. We denote the space of functions which are continuous over Ω by $C(\Omega)$, and the space of functions which have k continuous derivatives by $C^k(\Omega)$. The L^2-norm of a function v over Ω is defined by

$$\|v\|_0^2 = \int_\Omega |v(x)|^2 \, dx.$$

The space of square-integrable functions over Ω is denoted by

$$L^2(\Omega) = \Big\{ v : \Omega \to \mathbb{R} \Big| \ \|v\|_0 < \infty \Big\}.$$

The L^∞-norm of a function v over Ω is defined by

$$\|v\|_\infty^2 = \operatorname{ess\,sup}_{x\in\Omega} |v(x)|,$$

and the space of Lebesgue-measurable functions with bounded essential supremum is denoted by

$$L^\infty(\Omega) = \Big\{ v : \Omega \to \mathbb{R} \Big| \ v \text{ is Lebesgue-measurable, } \|v\|_\infty < \infty \Big\}.$$

Let $\alpha = (\alpha_1, \ldots, \alpha_d)$ be a multi-index of dimension d with non-negative integer entries α_i, with $|\alpha| = \sum_{i=1}^d \alpha_i$. We define

$$\nabla^\alpha = \left(\frac{\partial^{\alpha_1}}{\partial x_1^{\alpha_1}}, \ldots, \frac{\partial^{\alpha_d}}{\partial x_d^{\alpha_d}} \right).$$

The Sobolev space $H^k(\Omega)$ is defined by

$$H^k(\Omega) = \Big\{ v \in L^2(\Omega) \Big| \ \nabla^\alpha v \in L^2(\Omega), \ \forall \text{ multi-indices } \alpha, |\alpha| \le k \Big\}.$$

For $k = 1$, the H^1-seminorm is defined by

$$|v|_1^2 = \|\nabla v\|_0^2,$$

and the full H^1-norm is defined by

$$\|v\|_1^2 = \|v\|_0^2 + \|\nabla v\|_0^2 = \|v\|_0^2 + |v|_1^2.$$

The space

$$H^1(\Omega) = \Big\{ v \in L^2(\Omega) \Big| \ \nabla v \in (L^2(\Omega))^d \Big\},$$

thus denotes the space of $L^2(\Omega)$-functions with square integrable first derivatives.

The Laplace operator Δ, acting on a scalar function $v : \mathbb{R}^d \to \mathbb{R}$, is defined by

$$\Delta v = \sum_{i=1}^d \frac{\partial^2 v}{\partial x_i^2},$$

the divergence of a vector-valued function $w : \mathbb{R}^d \to \mathbb{R}^d$ is defined by

$$\operatorname{div} w = \sum_{i=1}^d \frac{\partial w_i}{\partial x_i}.$$

The space of functions in $(L^2(\Omega))^d$ with square integrable divergence is denoted by

$$H(\Omega, \mathrm{div}) = \Big\{ w \in (L^2(\Omega))^d \Big| \ \mathrm{div}\, w \in L^2(\Omega) \Big\}.$$

and endowed with the norm

$$\|w\|^2_{\mathrm{div}} = \|w\|^2_0 + \|\,\mathrm{div}\, w\|^2_0.$$

Note that, in general, we do not use a special notation (such as, e.g., the symbol $\vec{\cdot}$) for distinguishing between vectors and scalars. The context will sufficiently clarify whether a variable is scalar, a vector, or a matrix.

Two norms $\|\cdot\|_a$ and $\|\cdot\|_b$ on a space V are *equivalent*, if there exist constants $\underline{\mu}$ and $\overline{\mu}$, such that

$$\underline{\mu}\|v\|_a \leq \|v\|_b \leq \overline{\mu}\|v\|_a, \quad \forall v \in V.$$

When a norm $\|\cdot\|_*$ is taken over a subset $Q \subset \Omega$, we indicate this by writing $\|\cdot\|_{*,Q}$. For example,

$$\|v\|^2_{0,Q} = \int_Q |v(x)|^2 \, dx.$$

When boundary values of L^2-functions are discussed, they are to be understood as traces of these functions. We omit a discussion of the standard trace operator and refer the reader to the references listed at the beginning of this chapter.

The dual of a Banach space V is denoted by V^*, and the duality product on $V^* \times V$ by $\langle \cdot, \cdot \rangle$. The dual norm is given by

$$\|f\|_{V^*} = \sup_{v \in V,\ v \neq 0} \frac{|\langle f, v \rangle|}{\|v\|_V}.$$

2.2 Numerical analysis

In this section, we define the model problems which will be considered in this thesis, and we recall the general steps needed for obtaining a numerical solution for these problems.

2.2.1 Model problems

In the scope of this thesis, we will consider three model problems which are described by the partial differential equations (PDEs) presented below. By $\vec{n}$, we denote the outer unit normal vector to Ω (as mentioned above, we will, in general, omit the symbol $\vec{\cdot}$). Let $V_C = C^2(\Omega) \cap C^1(\Omega \cup \Gamma_1) \cap C(\Omega \cup \Gamma_0)$. Functions u which solve the following PDEs in a strong sense are called *classical solutions* or *strong solutions*.

(I) Scalar diffusion

Find a function $u \in V_C$, $u : \overline{\Omega} \to \mathbb{R}$, such that

$$\left.\begin{aligned} -\operatorname{div}(A\nabla u) &= f \quad \text{in } \Omega, \\ u &= g_0 \quad \text{on } \Gamma_0, \\ A\tfrac{\partial u}{\partial \vec{n}} &= g_1 \quad \text{on } \Gamma_1, \end{aligned}\right\} \tag{2.2}$$

where A denotes the diffusion coefficient, and f denotes the source term. The functions g_0 and g_1 are given Dirichlet and Neumann boundary conditions, respectively.

(II) Linear elasticity

Find the vector-valued displacement field $u \in (V_C)^2$, $u : \overline{\Omega} \to \mathbb{R}^2$, such that

$$\left.\begin{aligned} -\operatorname{div}(\sigma(u)) &= f \quad \text{in } \Omega, \\ u &= g_0 \quad \text{on } \Gamma_0, \\ \sigma(u)\vec{n} &= t_N \quad \text{on } \Gamma_1, \end{aligned}\right\} \tag{2.3}$$

where $\sigma(u) = \mathbb{C}\varepsilon(u)$, $\mathbb{C}$ is the fourth-order stiffness tensor, and $\varepsilon(u) = \frac{1}{2}(\nabla u + \nabla u^T)$ is the linearized strain tensor. The function f denotes body forces, g_0 given boundary displacements, and t_N the surface traction forces.

We will consider only isotropic materials. In this case, we have

$$\sigma(u) = \lambda \operatorname{tr}(\varepsilon(u))I + 2\,\mu\,\varepsilon(u),$$

where λ and μ are Lamé's parameters. These parameters can be represented by the Young's modulus E and Poisson's ratio ν as

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)}, \quad \mu = \frac{E}{2(1+\nu)}.$$

2.2.2 Variational formulation

From the PDEs presented in Section 2.2.1, one can derive the *variational form* or *weak form* in the standard way. Choose a proper function space V and a set of test functions $V_0 \subset V$ which vanish on Γ_0. Let V_g be the set of functions in V which fulfill the essential boundary conditions on Γ_0 (note that, if only homogeneous essential boundary conditions are considered, we have $V_0 = V_g$). Multiply the first equation in (2.2) or (2.3), respectively, with a test function $v \in V_0$. Then, integrating over the computational domain, applying partial integration and using boundary conditions, we obtain the following general *variational formulation* or *weak formulation*.

Find $u \in V_g$, such that

$$a(u, v) = \langle f, v \rangle \quad \forall v \in V_0. \tag{2.4}$$

The bilinear forms $a(\cdot,\cdot)$ and the functionals $\langle f, \cdot \rangle$ are given as follows.

(I) Scalar diffusion:

$$\begin{aligned} a(u,v) &= \int_\Omega A\nabla u \cdot \nabla v \, dx, \\ \langle f, v\rangle &= \int_\Omega f\, v \, dx + \int_{\Gamma_1} g_1\, v \, ds. \end{aligned}$$

(II) Linear elasticity:

$$\begin{aligned} a(u,v) &= \int_\Omega \sigma(u) : \varepsilon(v) \, dx, \\ \langle f, v\rangle &= \int_\Omega f \cdot v \, dx + \int_{\Gamma_1} t_N \cdot v \, ds, \end{aligned}$$

where the product of matrices of size $d \times d$ is defined by $\sigma : \varepsilon = \sum_{i,j=1}^{d} \sigma_{ij}\varepsilon_{ij}$.

2.2.3 Existence and uniqueness

We recall some fundamental results regarding the existence and uniqueness of a solution to the problems presented above.

Definition 2.1. Let V be a normed linear space. A bilinear form $a(\cdot,\cdot) : V \times V \to \mathbb{R}$ is called *coercive* on V, if there exist a constant $\mu_1 > 0$, such that

$$\mu_1 \|v\|^2 \leq |a(v,v)|, \quad \forall v \in V.$$

It is called *bounded*, if there exists a constant $\mu_2 < \infty$, such that

$$|a(v,w)| \leq \mu_2 \|v\|_V \|w\|_V, \quad \forall v, w \in V.$$

In all model problems, we assume that the given data is such that $\langle f, \cdot\rangle$ is a bounded linear functional and that the bilinear form $a(\cdot,\cdot)$ is bounded and coercive. In this case, the *energy norm* is defined by

$$\|v\|_E = \sqrt{a(v,v)}. \tag{2.5}$$

The existence and uniqueness of a solution for problem (2.4) is guaranteed by the following theorem.

Theorem 2.2. (Lax-Milgram) *Let V be a Hilbert space, the bilinear form $a(\cdot,\cdot) : V \times V \to \mathbb{R}$ be bounded and coercive, and $f : V \to \mathbb{R}$ be a bounded linear form, i.e., $f \in V^*$. Then, the variational problem* (2.4) *has a unique solution $u \in V$ and*

$$\tfrac{1}{\mu_2}\|f\|_{V^*} \leq \|u\|_V \leq \tfrac{1}{\mu_1}\|f\|_{V^*}, \tag{2.6}$$

where μ_1 and μ_2 are as in Definition 2.1.

When the bilinear form is symmetric, bounded, and coercive, we can reformulate problem (2.4) as the following minimization problem.

Find u such that

$$u = \arg\min_{v \in V_g} J(v), \quad \text{where} \quad J(v) = \tfrac{1}{2}a(v,v) - \langle f, v\rangle. \tag{2.7}$$

The functional $J(v)$ is called *Ritz energy functional*.

2.2.4 Galerkin's method

Usually, the infinite-dimensional problem (2.4) cannot be solved analytically. To reduce it to a finite-dimensional problem, we apply *Galerkin's method* as follows. Choose a finite-dimensional subspace $V_h \subset V$ and replace the infinite-dimensional spaces V, V_0, and V_g in (2.4) by finite-dimensional subspaces $V_h, V_{0h}, V_{gh} \subset V$, such that $V_{0h} \subset V_0$, and $V_{gh} \subset V_g$. We assume that the prescribed essential boundary conditions are such that there exists a function $\check{g} \in V_h : \ g_0 = \check{g}|_{\Gamma_0}$ (otherwise, we project the data g_0 to V_h). We define

$$\begin{aligned} V_{0h} &= \{v \in V_h : \ v|_{\Gamma_0} = 0\}, \\ V_{gh} &= \check{g} + V_{0h} \ = \ \{v \in V_h : \ v|_{\Gamma_0} = \check{g}\}. \end{aligned}$$

Note that, when homogeneous essential boundary conditions are prescribed, we have $V_{gh} = V_{0h}$. The problem is thus reformulated as follows.

Find $u_h \in V_{gh}$, such that

$$a(u_h, v_h) = \langle f, v_h \rangle \quad \forall v_h \in V_{0h}. \tag{2.8}$$

Let $\Phi = \{\varphi_1, \ldots, \varphi_n\}$ be a basis of V_h (i.e., V_h is n-dimensional), then any function $u_h \in V_h$ can be represented in the form

$$u_h(x) = \sum_{i=1}^{n} u_i \varphi_i(x) \tag{2.9}$$

with real-valued coefficients u_i. These coefficients are referred to as *degrees of freedom (DOF)*. The vector $\mathbf{u} = (u_i, \ldots, u_n)^T$ is called the *coefficient vector*. It uniquely determines the corresponding discrete function, and thus, we identify u_h and $\mathbf{u}$ with each other. We insert the representation (2.9) in (2.8) and use each basis function $\varphi_j \in V_{0h}$, $j = 1, \ldots, n$ as a test function. This results in a total of n conditions which can be written as the linear system of equations

$$\mathbf{K}\mathbf{u} = \mathbf{f}, \tag{2.10}$$

where the *stiffness matrix* $\mathbf{K}$ is given by

$$(\mathbf{K})_{ji} = a(\varphi_i, \varphi_j), \quad i, j = 1, \ldots, n,$$

and the *load vector* $\mathbf{f}$ by

$$(\mathbf{f})_j = \langle f, \varphi_j \rangle, \quad j = 1, \ldots, n.$$

The statement of Theorem 2.2 also holds true for the discretized problem (2.8), since the conditions are fulfilled on the full function space V, and thus also on the subspace $V_h \subset V$.

The entries of the stiffness matrix and the load vector are usually assembled from local contributions, i.e., for each element of the mesh, the underlying integral

is computed locally and the result is added to the corresponding entry in the matrix or vector, respectively.

Note that, in numerical implementations, $a(\cdot,\cdot)$ and $\langle f, \cdot \rangle$ in (2.8) usually cannot be realized exactly, e.g., due to inexact integral computation using quadrature rules, or due to the approximation of boundary conditions. While this could be indicated, e.g., by writing $a_h(\cdot,\cdot)$ and $f_h(\cdot)$, we omit this for the sake of readability.

2.2.5 A posteriori error estimation and adaptive refinement

Since the size of the problem (2.10) depends on the size of the chosen basis Φ, it is desirable to achieve a certain accuracy with as few basis functions as possible. By applying a posteriori error estimation techniques and estimating the distrubution of the (unknown) true error, it is possible to identify areas with high contributions to the global error and apply local refinement only there. For surveys of a posteriori error estimation techniques, see, e.g., [1, 2, 25, 37] and the references therein, as well as the references given at the beginning of this chapter.

Let η denote an estimate of the true error in some norm $\|\cdot\|_*$, i.e.,

$$\eta \approx \|u - u_h\|_*.$$

Let $\mathcal{Q}_h$ denote a mesh of the computational domain. Assume that we can write η as a sum of local, element-wise contributions, e.g.,

$$\eta = \sum_{Q \in \mathcal{Q}_h} \eta_Q \quad \text{or} \quad \eta^2 = \sum_{Q \in \mathcal{Q}_h} \eta_Q^2,$$

and assume that the distribution of the local contributions η_Q estimates the distribution of the true error. Once we have computed local estimates η_Q for each element Q of the mesh, we can compare them and choose a criterion for selecting elements which will be marked for further refinement. Typically, one chooses a threshold Θ and marks all cells Q for refinement, where the local error is above this threshold, i.e., where

$$\eta_Q > \Theta. \tag{2.11}$$

There are several possibilities for determining Θ, some of which we briefly present here. The parameter $\psi \in [0,1]$ which appears in the criteria below determines how many elements will be marked for refinement. Choosing $\psi = 0$ results in global refinement, setting $\psi = 1$ results in no refinement.

2.2.5.1 Fixed percentile of elements

In every step, a fixed percentile of all elements is marked for refinement, i.e., Θ is chosen such that

$$\Theta = (100 \cdot \psi)\text{-percentile of } \{\eta_Q\}_Q. \tag{2.12}$$

The α-percentile of a set $\mathcal{A} = \{a_1, \dots, a_\nu\}$ denotes the value $\bar{a}$ below which α percent of all values a_i fall. For example, if we choose $\psi = 0.8$ in (2.12), then Θ is chosen such that $n_Q > \Theta$ holds for 20% of the elements of the mesh.

When this criterion is used, the DOF increase by a fixed ratio in every step, but if ψ is chosen too large, it might result in the refinement of more elements than necessary.

2.2.5.2 Fixed ratio of the largest local error

This is a very simple to implement criterion where a cell is marked for refinement, if it exceeds a certain fraction of the largest local error, i.e., Θ is chosen as

$$\Theta \;=\; \psi \; \max_{Q}\{\eta_Q\}. \tag{2.13}$$

Here, the number of marked elements can vary. If the error distribution has a very large peak in a small area, and if the parameter ψ is chosen too large, the following situation may arise. If some (possibly large) areas have local errors which are large enough to contribute significantly to the total error, but still small enough to fall below the threshold Θ (which can be high due to the peak in the error distribution), then these areas may remain unmarked, thus slowing down the convergence.

2.2.5.3 Bulk-criterion

As a third criterion we mention the method described in [34]. The marked elements are chosen such that their contributions to the global estimate sum up to a certain fraction of the global estimate, i.e., a set of elements $\mathcal{T} \subset \mathcal{Q}_h$ is marked, such that

$$\sum_{Q\in\mathcal{T}} \eta_Q \geq (1-\psi)\,\eta. \tag{2.14}$$

The convergence of an adaptive scheme based on this marking criterion (and certain conditions) was shown in [34].

Chapter 3

Isogeometric analysis

The motivation for IGA has been discussed in the introduction in Section 1.1, Task 1. With this in mind, the mathematical principle is easily summarized. In IGA, the isoparametric principle is applied by expressing both the geometry and the discrete solution in terms of the same basis functions. In contrast to isoparametric methods of classical FEM, however, we assume that the geometry description is given as an input from CAD, which also already determines the initial, coarsest discretization of the domain.

In this chapter, we recall the definitions and methods of IGA relevant for the scope of this thesis and fix the used notations. The results presented in this chapter can be found, for example, in [28, 45] and the monograph [27]. For details on splines, see [76]. Note that these references will, in general, not be cited explicitly within this chapter, since they provide the basis for most of the statements in this chapter.

3.1 B-spline basis functions

We first define univariate B-spline basis functions and, for simplicity, discuss some important properties and refinement methods for univariate basis functions. The results extend straightforwardly to bivariate tensor product basis functions. Note that this is not a complete discussion of splines (for a complete discussion, the reader is referred to [76]).

3.1.1 Univariate B-spline basis functions

3.1.1.1 Recursive definition

Definition 3.1. A *knot vector* s is defined as a finite, real-valued, monotonically increasing sequence of real numbers, i.e.,

$$s = (s_1, \dots, s_m), \quad s_i \leq s_{i+1}, \quad \forall i \in \{1, \dots, m-1\}.$$

An entry s_i, $i \in \{1, \dots, m\}$, of the knot vector is referred to as *knot*. A knot s_i is called *interior knot*, if $(s_1 < s_i) \wedge (s_i < s_m)$. We say that the *multiplicity* of a knot

s_i is r, if $r = \#\{j \in \{1, \dots, m\} : s_j = s_i\}$ (where $\#$ denotes the cardinality), i.e., if a total of r knots have the same value as s_i. For any $i \in \{1, \dots, m-1\}$, the interval between two consecutive knots (s_i, s_{i+1}) is called *knot span*. It is called *empty knot span*, if $s_i = s_{i+1}$. It is called an *interior knot span*, if both conditions $s_1 < s_{i+1}$ and $s_i < s_m$ hold true.

Definition 3.2. Let p be a non-negative *degree* and let $s = (s_1, \dots, s_m)$ be a knot vector where the multiplicity of any interior knot is at most p. The $n = m - p - 1$ *univariate B-spline basis functions* $B^s_{i,p} : [s_1, s_m] \to \mathbb{R}$, $i = 1, \dots, n$, are defined by the Cox-de Boor recursion formula as follows.

$$B^s_{i,0}(\xi) = \begin{cases} 1 & \text{for} \quad s_i \le \xi < s_{i+1}, \\ 0 & \text{else.} \end{cases} \tag{3.1}$$

$$B^s_{i,p}(\xi) = \frac{\xi - s_i}{s_{i+p} - s_i} B^s_{i,p-1}(\xi) + \frac{s_{i+p+1} - \xi}{s_{i+p+1} - s_{i+1}} B^s_{i+1,p-1}(\xi). \tag{3.2}$$

At the endpoint s_m, the function values are defined by

$$B^s_{i,p}(s_m) = \lim_{\xi \to s_m} B^s_{i,p}(\xi) = \begin{cases} 1, & \text{if } k = n, \\ 0, & \text{else.} \end{cases} \tag{3.3}$$

Whenever a zero denominator appears in (3.2), the corresponding term is considered to be zero.

The derivatives of B-spline basis functions are given by the following formula.

$$\frac{\partial B^s_{i,p}}{\partial \xi}(\xi) = \frac{p}{s_{i+p} - s_i} B^s_{i,p-1}(\xi) - \frac{p}{s_{i+p+1} - s_{i+1}} B^s_{i+1,p-1}(\xi). \tag{3.4}$$

It easily follows from (3.4) and (3.3) that first derivatives at s_m are given by

$$\frac{\partial B^s_{i,p}}{\partial \xi}(s_m) = \begin{cases} \dfrac{p}{s_{m-1} - s_{m-p-1}}, & \text{if } k = n, \\ \dfrac{-p}{s_{m-1} - s_{m-p-1}}, & \text{if } k = n-1, \\ 0, & \text{else.} \end{cases}$$

Definition 3.3. A knot vector s is called *open knot vector*, if the multiplicity of a knot is at most p, except for the first and last knot which have multiplicity $p + 1$.

In Figure 3.1, B-spline basis functions of degree $p = 1$ to $p = 5$ defined on the knot vectors

$$s = (\underbrace{0, \dots, 0}_{p+1\text{-times}}, \tfrac{1}{10}, \tfrac{2}{10}, \tfrac{3}{10}, \tfrac{4}{10}, \tfrac{5}{10}, \tfrac{6}{10}, \tfrac{7}{10}, \tfrac{8}{10}, \tfrac{9}{10}, \underbrace{1, \dots, 1}_{p+1\text{-times}}) \tag{3.5}$$

are presented.

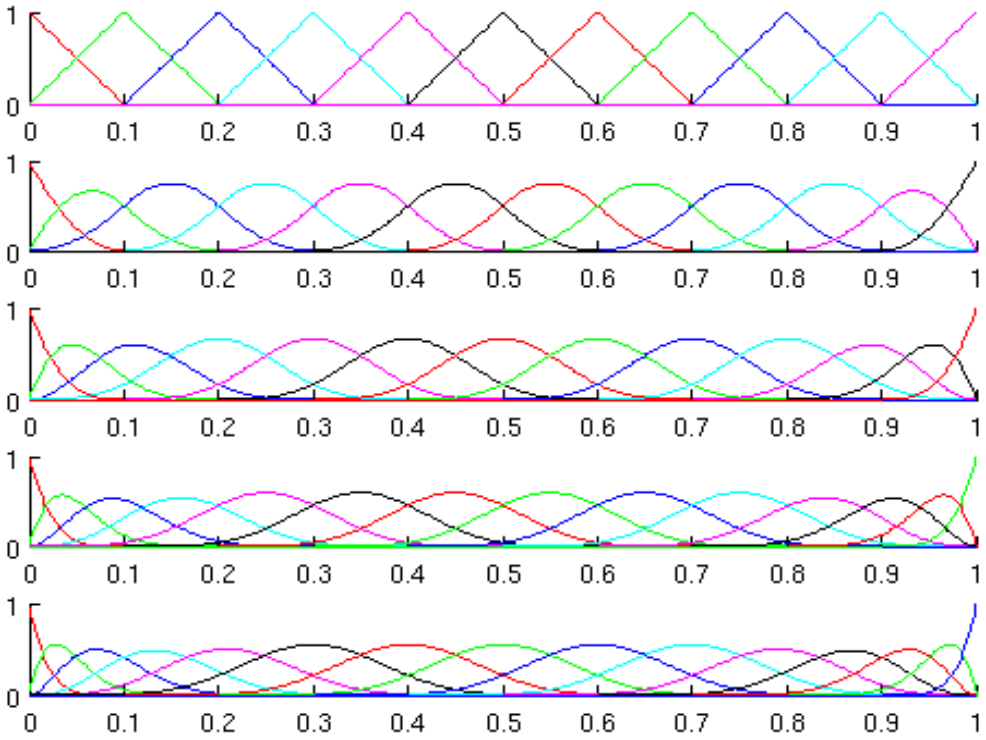


Figure 3.1: B-spline basis functions of degrees $p = 1$ (top) to $p = 5$ (bottom) defined on open knot vectors with uniform spacing of the interior knots as in (3.5).

We explicitly mention two properties of B-spline basis functions defined on open knot vectors. Firstly, the first and the last basis function are interpolatory at the beginning and the end of the parameter interval, respectively, while all other basis functions are zero there (as illustrated in Figure 3.1). This property will be relevant in Section 4.2.3. Secondly, since the number of interior knot spans ν is given by

$$\nu = m - 1 - 2p,$$

the number of basis functions $n = m - p - 1$ can be expressed in terms of interior knot spans by

$$n = \nu + p. \tag{3.6}$$

Assumption 3.4. *Hereinafter, all used knot vectors will be open knot vectors.*

Remark 3.5. *Note that, under the premise of considering only open knot vectors, the knot vector s also uniquely defines p and thus the basis.*

3.1.1.2 Properties of basis functions

We list some properties of univariate B-spline basis functions

1. The support of the basis function $B^s_{i,p}$ is local and is contained in $p+1$ knot spans, namely

$$\operatorname{supp} B^s_{i,p} \subseteq (s_i, s_{i+p+1}), \quad \forall i \in \{1, \dots, n\}.$$

2. On the knot span (s_i, s_{i+1}), only the $p+1$ basis functions with indices $i-p, \dots, i$ are nonzero.

$$B^s_{k,p}|_{(s_i,s_{i+1})} \neq 0 \Leftrightarrow k \in \{i-p, \dots, i\}.$$

3. The basis functions are non-negative,

$$B^s_{i,p}(\xi) \geq 0, \quad \forall i \in \{1, \dots, n\}, \ \forall \xi \in [s_1, s_m].$$

4. The basis functions form a partition of unity,

$$\sum_{i=1}^{n} B^s_{i,p}(\xi) = 1, \quad \forall \xi \in [s_1, s_m].$$

5. Basis functions are piecewise polynomials of degree p and, in general, globally C^{p-1} continuous. In the presence of multiple knots, the continuity reduces according to the multiplicity, i.e., if a knot appears r times, the continuity of a B-splines basis function at that knot is C^{p-r}.

These properties are also illustrated in Figures 3.1 and 3.2, where the B-spline basis functions of degree $p = 3$ with the knot vector

$$s = (0, 0, 0, 0, \tfrac{1}{10}, \tfrac{2}{10}, \tfrac{3}{10}, \tfrac{4}{10}, \tfrac{5}{10}, \tfrac{6}{10}, \tfrac{7}{10}, \tfrac{7}{10}, \tfrac{8}{10}, \tfrac{8}{10}, \tfrac{8}{10}, \tfrac{9}{10}, 1, 1, 1, 1)$$

are plotted. The knots at $\xi = \frac{7}{10}$ and $\xi = \frac{8}{10}$ are repeated. The functions are, in general, C^2-continuous, but the regularity is reduced at the repeated knots (C^1 at $\xi = \frac{7}{10}$ and C^0 at $\xi = \frac{8}{10}$). Also, the support of the basis functions in the vicinity of these repeated knots is smaller.

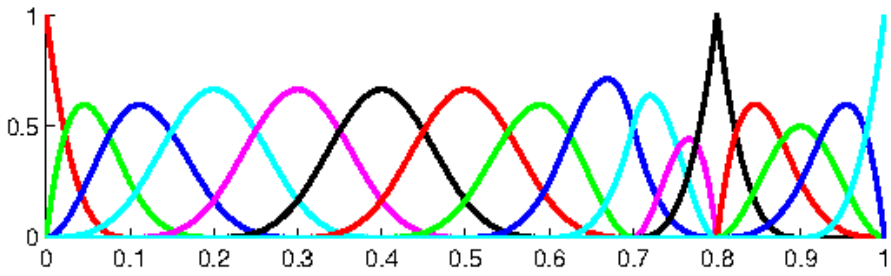


Figure 3.2: B-spline basis functions of degree $p = 3$ defined on the open knot vector $s = (0, 0, 0, 0, \frac{1}{10}, \frac{2}{10}, \frac{3}{10}, \frac{4}{10}, \frac{5}{10}, \frac{6}{10}, \frac{7}{10}, \frac{7}{10}, \frac{8}{10}, \frac{8}{10}, \frac{8}{10}, \frac{9}{10}, 1, 1, 1, 1)$.

Properties 1 and 2 motivate the use of B-spline basis functions as ansatz functions for the discrete solution. The high smoothness described in Property 5 is a special

feature of B-spline basis functions and distinguishes these functions from the standard C^0-continuous FEM basis functions. Properties 3 and 4 are of particular importance in the design process.

As we see from this list of properties, the knot vector plays an important role. It not only defines a subdivision of the interval $[s_1, s_m]$, but also the shape of the basis functions as well as their smoothness.

Assumption 3.6. *For simplicity, we assume for the remainder of this thesis that* $s_1 = 0$ *and* $s_m = 1$*, which can be easily achieved by a suitable scaling.*

3.1.1.3 Function evaluation via the inverted triangular scheme

As briefly mentioned in Section 2.2.4, stiffness matrix and load vector are assembled element-wise. In the one-dimensional setting, the knots provide a decomposition of the interval $[s_1, s_m]$ into "elements", namely the knot spans. As mentioned in Section 3.1.1.2, the basis functions whose support intersect the knot span (s_i, s_{i+1}) are given by the index set $\{i-p, \dots, i\}$. Obviously, a straightforward implementation of the recursive formula (3.2) would be very inefficient. There are efficient alternative procedures for the evaluation of B-spline functions and curves described in [76]. A summary and comparison of various techniques can be found in [48]. For the numerical tests presented in this thesis, the inverted triangular scheme was implemented, which is described in Algorithm 3.1. Note that, in the presented algorithm, the indices of the basis functions range from 1 to $p+1$, i.e., their *local* indices are used. The corresponding global indices are stored in the vector $\mathcal{I}_B$.

Algorithm 3.1 Inverted triangular scheme

Input: Knot vector s, degree p, knot span index i, evaluation points ξ.
Output: Values of local basis functions $\{B^s_{i,p}\}_{i=1}^{p+1}$,
Vector of global indices of local basis functions $\mathcal{I}_B$.

$B^s_{1,0} = 1$
for $q = 1$ **to** p **do**
  $\alpha_L = (\xi - s_i)/(s_{i+q} - s_i)$
  $\alpha_R = (s_{i+q+1} - \xi)/(s_{i+q+1} - s_{i+1})$
  $B^s_{1,q} = \alpha_R B^s_{1,q-1}$
  for $j = 2$ **to** q **do**
    $B^s_{j,q} = \alpha_L B^s_{j-1,q-1} + \alpha_R B^s_{j,q-1}$
  end for
  $B^s_{q+1,q} = \alpha_L B^s_{q,q-1}$
end for
$\mathcal{I}_B = (i-p, \dots, i)^T$
return $\mathcal{I}_B$, $\{B^s_{i,p}\}_{i=1}^{p+1}$.

From the definition (3.2) and from the presented Algorithm 3.1, it is obvious that the computational cost for function evaluation increases with the degree p. This is

not only due to the increased number of basis functions which have support on a given knot span, but also due to the additional recursion levels or loops that need to be computed.

3.1.1.4 Refinement

There are various options for refining a knot vector, and thereby, also the function space spanned by the corresponding B-spline basis functions. Discussions can also be found in [27, 45, 76], a paper with special focus and an extensive discussion on refinement methods in IGA is [28].

We will now denote the original knot span by s^0 and the knot vector after refinement by s^1. Accordingly, the corresponding numbers of basis functions and knot spans will be denoted by the respective upper indices 0 and 1. The number of *non-empty* interior knot spans will be denoted by $\overline{\nu}$.

Knot insertion (h-refinement)

Knot insertion is the analogue of h-refinement in classical FEM analysis. Given the knot vector s^0, the refined knot vector s^1 is obtained by copying s_0 and inserting a new knot in a non-empty knot span (s_i^0, s_{i+1}^0) of the original knot vector. This can be done at the midpoint, but not necessarily. In the case of uniform h-refinement, the number of new basis functions n_1 depends on the number of non-empty interior knot spans $\overline{\nu}^0$ in s^0,

$$n^1 = n^0 + \overline{\nu}^0.$$

Example (uniform knot insertion):
$s^0 = (0,\ 0,\ 0,\ \ 1/4,\ \ 1/2,\ 1/2,\ \ 3/4,\ \ 1,\ 1,\ 1)$
$s^1 = (0,\ 0,\ 0,\ 1/8,\ 1/4,\ 3/8,\ 1/2,\ 1/2,\ 5/8,\ 3/4,\ 7/8,\ 1,\ 1,\ 1)$
$\nu^0 = 5,\ \overline{\nu}^0 = 4,\ p^0 = 2,\ n^0 = 7;\quad \nu^1 = 9,\ \overline{\nu}^1 = 8,\ p^1 = 2,\ n^1 = 11.$

Degree elevation (p-refinement)

By *degree elevation*, we refer to an increase in the polynomial degree of the basis functions while keeping the continuity at the knots (as in p-refinement known from classical FEM). The multiplicity of each knot is increased by one (including the knots at the beginning and the end of the interval). The number of new basis functions n^1 depends on the number of non-empty interior knot spans $\overline{\nu}^0$ in s^0,

$$n^1 = n^0 + \overline{\nu}^0.$$

Example:
$s^0 = (0,\ 0,\ 0,\ \ 1/4,\ \ 1/2,\ 1/2,\ \ 3/4,\ \ 1,\ 1,\ 1)$
$s^1 = (0,\ 0,\ 0,\ 0,\ 1/4,\ 1/4,\ 1/2,\ 1/2,\ 1/2,\ 3/4,\ 3/4,\ 1,\ 1,\ 1,\ 1)$
$\nu^0 = 5,\ \overline{\nu}^0 = 4,\ p^0 = 2,\ n^0 = 7;\quad \nu^1 = 8,\ \overline{\nu}^1 = 4,\ p^1 = 3,\ n^1 = 11.$

Note that this method is also referred to as *order elevation* in the literature.

k-refinement

By *k-refinement*, we refer to increasing both the polynomial degree *and* the smoothness. This is very easily realized by increasing the multiplicity of the first and the last knot by one. With this refinement method, the number of basis functions is only increased by 1, i.e.,

$$n^1 = n^0 + 1. \tag{3.7}$$

Example:
$s^0 = (0,\ 0,\ 0,\quad 1/4,\ 1/2,\ 1/2,\ 3/4,\ 1,\ 1,\ 1)$
$s^1 = (0,\ 0,\ 0,\ 0,\ 1/4,\ 1/2,\ 1/2,\ 3/4,\ 1,\ 1,\ 1,\ 1)$
$\nu^0 = 5,\ p^0 = 2,\ n^0 = 7;\quad \nu^1 = 5,\ p^1 = 3,\ n^1 = 8.$

Note that, when applying k-refinement, the "coarse" functions (defined via s^0) are *not* contained in the span of the "refined" functions (defined via s^1). Also note that k-refinement has no analogue in FEM and is a special property of B-spline basis functions. We will exploit (3.7) in Chapter 5 for an efficient computation of a functional-type a posteriori error estimator. For a detailed study of k-refinement, see [28].

3.1.2 Bivariate B-spline basis functions

Bivariate B-spline basis functions are defined via the tensor product of univariate B-spline basis functions.

Definition 3.7. Let $\{B^s_{i,p}\}_{i=1}^{n_1}$ and $\{B^t_{j,q}\}_{j=1}^{n_2}$ be two families of B-spline basis functions defined by the degrees p and q, and the open knot vectors

$$s = (s_1, \dots, s_{m_1}),\ t = (t_1, \dots, t_{m_2}),$$

respectively, where $m_1 = n_1 + p + 1$ and $m_2 = n_2 + q + 1$. We denote the set of all double-indices (i, j) by

$$\mathcal{I} = \{(i,j) :\ i \in \{1, \dots, n_1\},\ j \in \{1, \dots, n_2\}\}.$$

The *bivariate B-spline basis functions* $B^{(s,t)}_{(i,j),(p,q)} : [s_1, s_{m_1}] \times [t_1, t_{m_2}] \to \mathbb{R}$, $(i,j) \in \mathcal{I}$ are defined by

$$B^{(s,t)}_{(i,j),(p,q)}(\xi_1, \xi_2) = B^s_{i,p}(\xi_1)\, B^t_{j,q}(\xi_2). \tag{3.8}$$

For better readability, we will use the shorter notation

$$B^{(s,t)}_{(i,j),(p,q)} = B_{(i,j)}$$

when the context sufficiently clarifies the knot vectors and degrees. In some places, in particular in Chapter 4, we will collapse the double index (i, j) into one *multi-index* $\boldsymbol{i}$ for better readability. For example, a family of bivariate B-spline basis functions can be described as

$$\left\{B_{(i,j)}\right\}_{(i,j)\in\mathcal{I}} = \{B_{\boldsymbol{i}}\}_{\boldsymbol{i}\in\mathcal{I}}\,. \tag{3.9}$$

The knot vectors s and t define a tensor product mesh which divides the domain,

$$\widehat{\Omega} = (s_1, s_{m_1}) \times (t_1, t_{m_2}),$$

into rectangular elements

$$\widehat{Q}_{(i,j)} = (s_i, s_{i+1}) \times (t_j, t_{j+1}), \tag{3.10}$$

which we refer to as *cells* (cf. Remark 3.12 below). We denote this mesh of $\widehat{\Omega}$ by

$$\widehat{\mathcal{Q}}_h = \Big\{ (s_i, s_{i+1}) \times (t_j, t_{j+1}) \;\Big|\; i \in \{1, \dots, m_1 - 1\},\ j \in \{1, \dots, m_2 - 1\} \Big\}, \tag{3.11}$$

where the parameter h indicates the typical cell-size of the particular mesh (see Figure 3.4 for an illustration).

3.2 Isogeometric single-patch discretizations

3.2.1 B-spline geometry mappings

The B-spline basis functions discussed above are used to define the following geometry mappings.

Definition 3.8. Let $\{B^s_{i,p}\}_{i=1}^n$ be a family of univariate B-spline basis functions. Given a *control polygon* of *control points* $P_i \in \mathbb{R}^2$, $i \in \{1, \dots, n\}$, the *B-spline curve* $C : (s_1, s_m) \to \mathbb{R}^2$ is defined by

$$C(\xi) = \sum_{i=1}^{n} B^s_{i,p}(\xi)\, P_i. \tag{3.12}$$

Definition 3.9. Let $\{B^{(s,t)}_{(i,j),(p,q)}\}_{(i,j)\in\mathcal{I}}$ be a family of bivariate B-spline basis functions and let

$$\widehat{\Omega} = (s_1, s_{m_1}) \times (t_1, t_{m_2}).$$

Given a *control net* of *control points* $P_{(i,j)} \in \mathbb{R}^2$, $(i,j) \in \mathcal{I}$, the two-dimensional *B-spline surface* $G : \widehat{\Omega} \to \mathbb{R}^2$ is defined by

$$G(\xi_1, \xi_2) = \sum_{(i,j)\in\mathcal{I}} B^{(s,t)}_{(i,j),(p,q)}(\xi_1, \xi_2)\, P_{(i,j)}. \tag{3.13}$$

The domain of the mapping G, defined by $\widehat{\Omega}$, is called the *parameter domain*, and its image, defined by $\Omega = G(\widehat{\Omega}) \subset \mathbb{R}^2$, is called the *physical domain*. The *Jacobian* of the geometry mapping is denoted by

$$\nabla G = \nabla_\xi G = \begin{pmatrix} \frac{\partial G_1}{\partial \xi_1} & \frac{\partial G_1}{\partial \xi_2} \\ \frac{\partial G_2}{\partial \xi_1} & \frac{\partial G_2}{\partial \xi_2} \end{pmatrix},$$

where G_k denotes the k-th component of G. The geometry mapping is called *regular*, if $\det \nabla G > 0$ on $\widehat{\Omega}$.

Note that, as a consequence of Assumption 3.6, we have

$$\widehat{\Omega} = (0,1)^2.$$

The use of open knot vectors (see Assumption 3.4) ensures that a B-spline curve is interpolatory at the first and last control point. In case of a B-spline surface, the corners of the parameter domain are mapped to the coordinates of the control points at the corners of the control net. We refer to the image of a corner of the parameter domain as *domain vertex* (or *vertex* for brevity). For later reference, we formulate the following observation as a remark.

Remark 3.10. *At a given domain vertex of a B-spline surface, there is only one basis function which has value* 1 *at this vertex, while all other basis functions are zero there.*

The geometry mapping (3.13) can be interpreted as a linear combination of control points, weighted by the values of the respective B-spline basis functions at a given point of the parameter domain. Due to the local support of the basis functions, the position of any control point only has local influence on the geometry, which allows easy local editing of the physical domain. In Figure 3.3, this is illustrated on a B-spline curve.

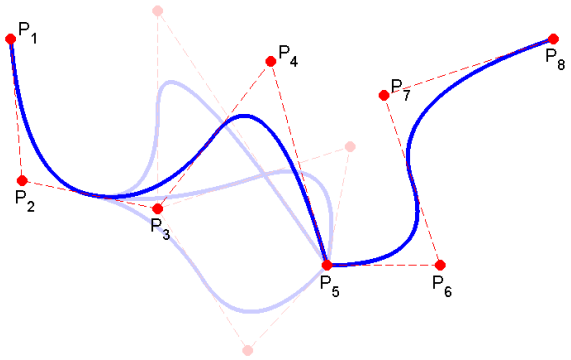


Figure 3.3: Influence of position of control point P_4 on the shape of the plotted B-spline curve with knot vector $s = (0, 0, 0, \frac{1}{5}, \frac{2}{5}, \frac{3}{5}, \frac{3}{5}, \frac{4}{5}, 1, 1, 1)$.

Assumption 3.11. *A key assumption in IGA is that the geometry representation is determined by some preceding design process, i.e., the geometry mapping is already provided as an input to the numerical computation. Furthermore, we assume that geometry mapping is continuous, regular, and bijective (i.e., not self-penetrating), which are natural assumptions for CAD-applications.*

The cells $\widehat{Q}$ of the mesh $\widehat{\mathcal{Q}}_h$ of the parameter domain (see (3.10) and (3.11)) are mapped to Ω by the geometry mapping G, thus defining cells and a mesh on the

physical domain. We denote the cells in the physical domain by

$$Q_{(i,j)} = G(\widehat{Q}_{(i,j)}),\ \widehat{Q}_{(i,j)} \in \widehat{\mathcal{Q}}_h, \tag{3.14}$$

and the mesh by

$$\mathcal{Q}_h = \left\{ Q = G(\widehat{Q}) \,\middle|\, \widehat{Q} \in \widehat{\mathcal{Q}}_h \right\}.$$

See Figure 3.4 for an illustration.

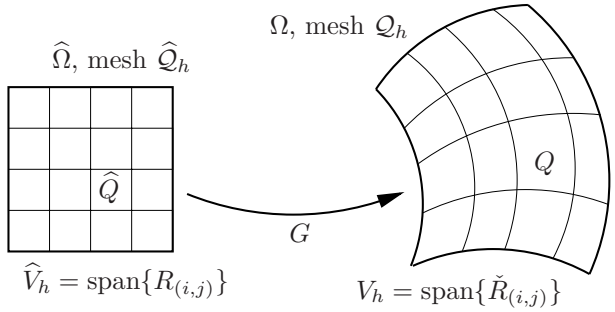


Figure 3.4: Illustration of the used notation.

Remark 3.12. *Note that we use the term* cell *for the rectangular elements of the parameter domain (see* (3.10)*), as well as for their images in the physical domain (see* (3.14)*). We use the term* knot span *only in reference to the knot spans of a (one-dimensional) knot vector (see Definition 3.1).*

In Section 1.1, we have used the term "element" for easier understanding, since this term is well known from classical FEM. Our use of the term "cell" of a mesh (and "knot span" in a one-dimensional setting) is analogous to the "element" in classical FEM. Due to the different nature and construction of the cells in IGA, however, we will use the term "cell" in the remainder of this thesis.

Definition 3.13. A family of meshes $\{\mathcal{Q}_h\}_h$ is called *shape-regular*, if there exists a constant $C_r > 0$, such that

$$\max_{Q \in \mathcal{Q}} \frac{\mathrm{diam}(Q)}{|Q|} < C_r, \quad \forall \mathcal{Q} \in \{\mathcal{Q}_h\}_h,$$

where $|Q|$ denotes the area of Q. It is called *quasi-uniform*, if the ratio of the sizes of two neighbouring elements is uniformly bounded (note that shape-regularity follows from quasi-uniformity).

Assumption 3.14. *In addition to Assumption 3.11, we assume that the considered meshes are quasi-uniform (and thus shape-regular).*

References for discussion of geometry mappings which might not satisfy Assumption 3.14 are mentioned in Section 3.3.2.

3.2.2 Refinement of the geometry mapping

When knot vectors are refined as described in Section 3.1.1.4, the number of basis functions is changed. Since the control points are closely linked to the basis functions, the set of control points has to be adapted properly. Depending on the specific refinement, the coordinates of some control points have to be re-computed in order to ensure that the image of the geometry mapping, i.e., the phyiscal domain, remains unchanged, and that the parameterization is preserved as well.

For simplicity, the descriptions below are given for (univariate) B-spline curves, but the methods straightforwardly extend to the tensor product settings, and thus to B-spline surfaces.

Knot insertion

Let a B-spline curve be defined by the family of B-spline basis functions $\{B^s_{i,p}\}_{i=1}^n$ and the corresponding set of control points $\{P_i\}_{i=1}^n$, where

$$s^0 = (s_1, \dots, s_j, s_{j+1}, \dots, s_m).$$

By inserting the knot $\bar{s}$ in the knot span (s_j, s_{j+1}), we obtain the refined knot vector

$$s^1 = (s_1, \dots, s_j, \bar{s}, s_{j+1}, \dots, s_m).$$

The $n+1$ new control points $\bar{P}_i$, $i = 1, \dots, n+1$, are formed from the original control points as follows:

$$\bar{P}_i = \alpha_i P_i + (1 - \alpha_i) P_{i-1},$$

where

$$\alpha_k = \begin{cases} 1, & 1 \le k \le j-p, \\ \frac{\bar{s}-s_i}{s_{i+p}-s_i}, & j-p < k \le j, \\ 0, & j < k \le n+p+2 = m+1. \end{cases}$$

Degree elevation

It is also possible to apply degree elevation while keeping the curve as well as its parameterization intact. As described in [45], this involves subdividing the curve into several Bézier curves, degree elevation of these segments, and re-combining the segments to one B-spline curve. For details, see [27, 76].

k-refinement

We are not aware of a method that preserves the geometry and the parametrization when k-refinement is performed. When k-refinement is applied in the numerical examples presented herein, the geometry mapping and its Jacobian are computed from the original geometry mapping. Since the original geometry mesh is typically coarse compared to the meshes (on which numerical solutions are computed), this requires only very little additional data to be stored.

3.2.3 Non-uniform rational B-splines (NURBS)

With B-spline geometry mappings, a great variety of shapes can be represented very accurately. However, the *exact* representation of conic sections, such as circles and ellipses, is not possible using only B-splines. For this, we introduce *non-uniform rational B-spline (NURBS) basis functions*.

Definition 3.15. Let $\{B_{i,p}^s\}_{i=1}^n$ be a family of B-spline basis functions, and let $\{w_i\}_{i=1}^n$ be positive *weights*. The *univariate NURBS basis functions* are defined by

$$R_{i,p}^s(\xi) = \frac{w_i B_{i,p}^s(\xi)}{\sum_{j=1}^n w_j B_{j,p}^s(\xi)}. \tag{3.15}$$

Let $\{B_{i,p}^s\}_{i=1}^{n_1}$ and $\{B_{j,q}^t\}_{j=1}^{n_2}$ be two families of B-spline basis functions, and let $\{w_{(i,j)}\}_{(i,j)\in\mathcal{I}}$ be positive *weights*. The *bivariate NURBS basis functions* are defined by

$$R_{(i,j)}(\xi_1,\xi_2) = \frac{w_{(i,j)} B_{(i,j)}(\xi_1,\xi_2)}{\sum_{(k,\ell)\in\mathcal{I}} w_{(k,\ell)} B_{(k.\ell)}(\xi_1,\xi_2)}. \tag{3.16}$$

Definition 3.16. The *NURBS curve* and *NURBS surface* are defined analogously to Definitions 3.8 and 3.9, where the B-spline basis functions $B_{i,p}^s$ and $B_{(i,j)}$ are replaced by the rational basis functions $R_{i,p}^s$ and $R_{(i,j)}$, respectively.

Remark 3.17. *The properties of B-spline basis functions listed in Section 3.1.1.2 are inherited by their rational counterparts, i.e., the properties are also valid for rational basis functions. Furthermore, B-splines can be seen as a special case of NURBS where all weights are equal. Thus, hereinafter, we will only use the term NURBS to refer to B-spline as well as NURBS functions, unless stated otherwise.*

Note that one can write (3.16) in the form

$$R_{(i,j)}(\xi_1,\xi_2) = \frac{w_{(i,j)} B_{(i,j)}(\xi_1,\xi_2)}{R_D(\xi_1,\xi_2)} \tag{3.17}$$

where the denominator is represented by one global function

$$R_D(\xi_1,\xi_2) = \sum_{(k,\ell)\in\mathcal{I}} w_{(k,\ell)} B_{(k.\ell)}(\xi_1,\xi_2).$$

In the literature, this function is also referred to as the *weighting function*.

If a refinement of the basis functions is carried out in such a way that the geometry mapping is not changed (see Section 3.2.2), the denominator R_D also remains unchanged. Hence, it is possible to compute the denominator from the original, coarse discretization, which can be advantageous in the implementation.

3.2.4 Isogeometric discretization

The isoparametric principle is applied by representing the discrete solution u_h in terms of the NURBS basis functions, which, in turn, are determined by the given geometry mapping. These basis functions, which are defined on the parameter domain $\widehat{\Omega}$, are pushed-forward to the physical domain Ω by G, i.e., we define

$$\check{R}_{(i,j)} = R_{(i,j)} \circ G^{-1}. \tag{3.18}$$

In the scalar valued problem (I), the discrete solution u_h is thus represented in the form (2.9) as

$$u_h(x) = \sum_{(i,j)\in\mathcal{I}} u_{(i,j)} \check{R}_{(i,j)}(x), \tag{3.19}$$

and the discrete function space V_h is given by

$$V_h = \operatorname{span}\left\{\check{R}_{(i,j)}\right\}_{(i,j)\in\mathcal{I}} \subset H^1(\Omega). \tag{3.20}$$

In some places, we will also refer to the span of the basis functions on the parameter domain, which we denote by

$$\widehat{V}_h = \operatorname{span}\left\{R_{(i,j)}\right\}_{(i,j)\in\mathcal{I}} \subset H^1(\widehat{\Omega}). \tag{3.21}$$

In problem (II), where u_h is vector-valued, we take the corresponding vector-valued basis functions from $(H^1(\widehat{\Omega}))^2$ and $(H^1(\Omega))^2$, respectively. Hereinafter, however, we will not distinguish between these cases (unless explicitly stated) assuming that the context sufficiently clarifies whether the considered basis functions are scalar or vector valued.

Note that the NURBS basis is not a nodal basis. In IGA, the function value at a given point of the domain is, in general, not determined by only one coefficient, but is the sum of several coefficients $u_{(i,j)}$ multiplied with the values of the respective basis functions at this point.

V_{0h} and V_{gh} are subsets of V_h. Homogeneous essential boundary conditions can be incorporated exactly by setting all DOF to zero, which are associated with basis functions at the boundary. In the case of inhomogeneous essential boundary conditions, we assume that they can be represented by NURBS basis functions (see discussion in Section 2.2.4).

3.3 Further remarks regarding IGA

We briefly mention some further aspects of IGA, which are not discussed in this thesis, but for which some references shall be provided for the sake of completeness.

3.3.1 Approximation properties

The following a priori error estimate was shown in [8]. We will not discuss any technical details, but only mention one of the main results. For details, the reader is referred to [8].

We define the *bent Sobolev space* of order $k \in \mathbb{N}$ as follows.

$$\begin{aligned}\mathcal{H}^k(\Omega) = \Big\{ v \in L^2(\Omega) \Big|\ & v|_Q \in H^k(Q),\ \forall Q \in \mathcal{Q}_h, \text{ and} \\ & \nabla^\alpha(v|_{Q_1}) = \nabla^\alpha(v|_{Q_2}) \text{ on } \partial Q_1 \cap \partial Q_2, \forall \alpha \text{ multi-index with} \\ & |\alpha| \leq \min\{k_{Q_1,Q_2}, k-1\} \forall Q_1, Q_2 \in \mathcal{Q}_h \text{ with } \partial Q_1 \cap \partial Q_2 \neq \emptyset \Big\}.\end{aligned}$$

Note that the notation in [8] differs from the notation used herein. In [8], Q and $\mathcal{Q}_h$ refer to the parameter domain. In this thesis, we use $\widehat{Q}$ and $\widehat{\mathcal{Q}}_h$ referring to the parameter domain, and Q and $\mathcal{Q}_h$ referring to the physical domain. Also note that the original definition of $\mathcal{H}^k$ in [8] is given on the parameter domain.

The space $\mathcal{H}^k(\Omega)$ is a well-defined Hilbert space, endowed with the seminorms

$$|v|^2_{\mathcal{H}^i(\Omega)} = \sum_{Q \in \mathcal{Q}_h} |v|^2_{H^i(Q)}, \quad 0 \leq i \leq k,$$

and the norm

$$\|v\|^2_{\mathcal{H}^k(\Omega)} = \sum_{i=0}^{k} |v|^2_{\mathcal{H}^i(\Omega)}.$$

Theorem 3.18. *Let $\Pi_{V_h} : L^2(\Omega) \to V_h$ be the projector to the NURBS space V_h. Let k and ℓ be integer indices with $0 \leq k \leq \ell \leq p+1$. Then, under suitable conditions (including, e.g, Assumption 3.14), for all $v \in H^\ell(\Omega)$,*

$$\sum_{Q \in \mathcal{Q}_h} |v - \Pi_{V_h} v|^2_{\mathcal{H}^k_h(Q)} \leq C_{\text{shape}} \sum_{Q \in \mathcal{Q}_h} h_Q^{2(\ell-k)} \sum_{i=0}^{\ell} \|\nabla G\|^{2(i-\ell)}_{L^\infty(G^{-1}(Q))} |v|^2_{H^i(Q)}, \tag{3.22}$$

where $|\cdot|_{\mathcal{H}^k_h(Q)}$ is the restriction of the seminorm to the cell Q, and where the constant C_{shape} may depend on the shape of the geometry and the specific parameterization, but not on the meshsize h.

As discussed in [8], this result shows that the same optimal rate of convergence is achieved with NURBS space on the physical domain Ω as with classical finite element spaces of degree p.

3.3.2 Singular and distorted geometry mappings

From (3.18), it is obvious that basis functions on the physical domain are directly influenced by the geometry mapping G. Also in (3.22), the influence of the geometry mapping is indicated by the explicitly appearing Jacobian ∇G in the estimate. Other

dependencies on the specific parameterization of the computational domain, however, are hidden in the constant C_{shape}. In [77], the effect of distorted geometry mappings on the condition number of the stiffness matrix are studied. The bounds presented therein depend on the quality of the mesh, on the parameter domain and on the physical domain, as well as the determinant of the Jacobian.

Singular geometry mappings can arise when non-quadrilateral domain are represented by a single NURBS geometry mapping, or as a result from isogeometric shape optimization methods (see, e.g., [68]). Such singular mappings may lead to discretizations where, on the physical domain, some basis functions are not properly defined or do not fulfill regularity properties. Also, the mesh in the physical domain may become heavily distorted, up to a point where quasi-uniformity is no longer ensured. In [91, 92, 93], the properties of the basis functions in such cases are studied. Furthermore, sufficient conditions for guaranteeing the regularity of basis functions are formulated, and a scheme for modifying the discrete function space to regain regularity are presented.

For detailed discussion, the reader is referred to the above-mentioned references and the references therein.

3.3.3 A posteriori error estimation

For the sake of completeness, we recall the discussion in Section 1.1, Task 3, regarding the state of the art of a posteriori error estimation in IGA (see also Section 2.2.5). As stated there, the only published results on a posteriori error estimation in IGA are, to the best of our knowledge, [33, 49, 97, 98, 99]. The a posteriori error estimators proposed in these references have in common that they contain generic/unknown constants, resulting in possible quantitative over-estimation of the true error. In this thesis, we consider quantitatively sharp error estimators in IGA, which will be discussed in Chapter 5.

3.3.4 Local refinement

Local refinement can be motivated from the design process (in order to obtain a better resolution of sharp, local features) or in the course of adapative mesh refinement (see Section 2.2.5). In any case, it is obvious that local refinement in IGA cannot be realized in a naive and straightforward manner due to the tensor product structure of bivariate NURBS basis functions. Any refinement on one knot vector will be extended throughout the whole domain.

While isogeometric local refinement methods are out of the scope of this thesis, some current developments are mentioned for the sake of completeness.

- Analysis-suitable T-splines [9, 63, 64, 87, 88, 89, 90].
- Truncated Hierarchical B-splines (THB-splines) [44].
- Polynomial splines over hierarchical T-meshes (PHT-splines) [31, 70, 98].

- Splines over locally refined box-partitions [32].

The isogeometric tearing and interconnecting method presented in Chapter 4 provides another, easy to implement alternative, if the global geometry is composed of several subdomains. In Section 4.3, we discuss how refinement on a tensor product mesh can be restricted to one subdomain (out of many) by proper coupling at the subdomain interfaces.

Chapter 4

IETI - IsogEometric Tearing and Interconnecting

In practical applications, the construction and design of geometric objects may require multi-patch parameterizations, i.e., the representation of the domain as a union of several NURBS surfaces or volumes (see, e.g., [69]). While the overall geometry may be complicated, the structure within one subdomain remains highly regular, which opens new perspectives for the design of solvers in numerical simulation. This will be studied in this chapter, and the results have been published in [57].

The model problems considered in this chapter are the scalar diffusion and the linear elasticity problem, i.e., problems of type (I) and (II) in Section 2.2.1.

4.1 Multi-patch NURBS discretization

Recall the following notation from Section 2.1.1. Let the physical domain $\Omega \subset \mathbb{R}^2$ be represented by N single-patch NURBS geometry mappings $G^{(k)}$, $k = 1, \dots, N$, each of which maps the parameter domain $\widehat{\Omega} = (0,1)^2$ to an open physical subdomain

$$\Omega^{(k)} = G^{(k)}(\widehat{\Omega}) \subset \Omega, \ k = 1, \dots, N,$$

such that

$$\overline{\Omega} = \bigcup_{k=1}^{N} \overline{\Omega^{(k)}} \text{ and } \Omega^{(k)} \cap \Omega^{(\ell)} = \emptyset \text{ for } k \neq \ell.$$

We use the superscript (k) to indicate that knot vectors, degrees, NURBS basis functions, index sets, DOF, etc. are associated with a mapping $G^{(k)}$. For example, using the multi-index notation (see (3.9)) in the scalar-valued case, we denote the set of NURBS basis functions of the geometry mapping $G^{(k)}$ by $\{R_{\boldsymbol{i}}^{(k)}\}_{\boldsymbol{i} \in \mathcal{I}^{(k)}}$. For each subdomain $\Omega^{(k)}$, the local function space is defined analogously to (3.18) and (3.20) as

$$V_h^{(k)} = \operatorname{span} \left\{ \ \check{R}_{\boldsymbol{i}}^{(k)} \ \right\}_{\boldsymbol{i} \in \mathcal{I}^{(k)}} \subset H^1(\Omega^{(k)}),$$

where

$$\check{R}_{\boldsymbol{i}}^{(k)} = R_{\boldsymbol{i}}^{(k)} \circ {G^{(k)}}^{-1}. \tag{4.1}$$

We denote the space of functions that are locally in $V_h^{(k)}$ by

$$\Pi V_h = \left\{ v \in L^2(\Omega) : \, v|_{\Omega^{(k)}} \in V_h^{(k)}, \, \forall \, k = 1, \dots, N \right\} \equiv \prod_{k=1}^{N} V_h^{(k)}. \tag{4.2}$$

Functions in ΠV_h are not necessarily continuous across subdomain interfaces (for the formal definition of interfaces, see (4.5) below). We choose the following subsets of continuous functions in ΠV_h:

$$\begin{aligned} V_h &= \Pi V_h \cap C(\Omega), \\ V_{0h} &= \{ v \in V_h : \, v|_{\Gamma_0} = 0 \}, \\ V_{gh} &= \breve{g} + V_{0h}. \end{aligned} \tag{4.3}$$

A function $u_h \in V_h$ is represented subdomain-wise by

$$u_h\big|_{\Omega^{(k)}} = \sum_{\boldsymbol{i} \in \mathcal{I}^{(k)}} u_{\boldsymbol{i}}^{(k)} \, \check{R}_{\boldsymbol{i}}^{(k)}. \tag{4.4}$$

In the case of vector-valued basis functions, the corresponding spaces are defined analogously.

We denote the *interface* of two subdomains $\Omega^{(k)}$ and $\Omega^{(\ell)}$ (see Figure 4.1) by

$$\Gamma^{(k,\ell)} = \partial\Omega^{(k)} \cap \partial\Omega^{(\ell)}. \tag{4.5}$$

We collect the index-tupels of all interfaces that are non-empty

$$\mathcal{C}_\Gamma = \{ (k,\ell) : \, \Gamma^{(k,\ell)} \neq \emptyset \}, \tag{4.6}$$

with $k, \ell \in \{1, \dots, N\}$. For $(k,\ell) \in \mathcal{C}_\Gamma$, we call $\Gamma^{(k,\ell)}$ a *subdomain vertex* (or *vertex* for brevity) if it consists of a single point, otherwise we call it an *edge*. For $(k,\ell) \in \mathcal{C}_\Gamma$, we collect the indices of those basis functions in $\Omega^{(k)}$ whose support intersects the interface $\Gamma^{(k,\ell)}$:

$$\mathcal{B}(k,\ell) \;=\; \left\{ \boldsymbol{i} \in \mathcal{I}^{(k)} : \, \operatorname{supp} \check{R}_{\boldsymbol{i}}^{(k)} \cap \Gamma^{(k,\ell)} \neq \emptyset \right\}. \tag{4.7}$$

If $\boldsymbol{i} \in \mathcal{B}(k,\ell)$, we say that the DOF $u_{\boldsymbol{i}}^{(k)}$ is associated with the interface $\Gamma^{(k,\ell)}$. In case of a subdomain vertex, we have $\#\mathcal{B}(k,\ell) = \#\mathcal{B}(\ell,k) = 1$, where $\#$ indicates the cardinality.

Definition 4.1. Let $\Gamma^{(k,\ell)}$ be an edge. We say that the subdomains $\Omega^{(k)}$ and $\Omega^{(\ell)}$ are *fully matching*, if the following two conditions are fulfilled (see Figure 4.1 for an illustration):

(i) The interface $\Gamma^{(k,\ell)}$ is the image of an entire edge of the respective parameter domains.

(ii) For each index $\boldsymbol{i} \in \mathcal{B}(k,\ell)$, there must be a unique index $\boldsymbol{j} \in \mathcal{B}(\ell,k)$, such that

$$\check{R}^{(k)}_{\boldsymbol{i}}|_{\Gamma^{(k,\ell)}} = \check{R}^{(\ell)}_{\boldsymbol{j}}|_{\Gamma^{(k,\ell)}}. \tag{4.8}$$

This is the case, if the two knot vectors are affinely related and the corresponding weights and degrees are equal.

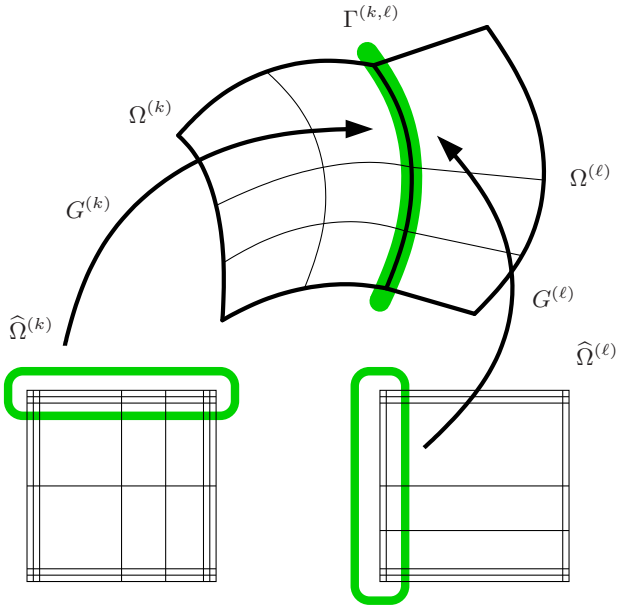


Figure 4.1: Fully matching subdomains $\Omega^{(k)}$ and $\Omega^{(\ell)}$:
All weights equal to 1,
$p^{(k)} = 2$, $s^{(k)} = (0,0,0,0.5,0.75,1,1,1)$,
$q^{(k)} = 2$, $t^{(k)} = (0,0,0,0.5,1,1,1)$,
$p^{(\ell)} = 1$, $s^{(\ell)} = (0,0,1,1)$,
$q^{(\ell)} = 2$, $t^{(\ell)} = (0,0,0,0.25,0.5,1,1,1)$.

For an illustration of two fully matching subdomains, see Figure 4.1: The interface $\Gamma^{(k,\ell)}$ is the image of the entire northern edge of $\widehat{\Omega}^{(k)}$ under the mapping $G^{(k)}$, and the image of the entire western edge of $\widehat{\Omega}^{(\ell)}$ under $G^{(\ell)}$. Furthermore, $p^{(k)} = q^{(\ell)} = 2$. The knot vectors $s^{(k)}$ and $t^{(\ell)}$ are not equal, but due to the way they are mapped to $\Gamma^{(k,\ell)}$, condition (ii) is fulfilled. Hence, $\Omega^{(k)}$ and $\Omega^{(\ell)}$ are fully matching.

The tensor-product structure of the NURBS basis functions is very convenient for collecting/identifying the DOF associated with an interface, i.e., the index-set $\mathcal{B}(k,\ell)$. In particular, in combination with condition (i) for the fully matching case, one only has to know which side (north, east, south or west) of the parameter domain

defines the interface to identify the associated DOF. For example, in Figure 4.1, using the double-index notation, we have

$$\begin{aligned}\mathcal{B}(k,\ell) &= \{(i,j)\}_{\substack{i=1,\dots,5\\ j=4}} \subset \mathcal{I}^{(k)},\\ \mathcal{B}(\ell,k) &= \{(i,j)\}_{\substack{i=1\\ j=1,\dots,5}} \subset \mathcal{I}^{(\ell)}.\end{aligned}$$

Assumption 4.2. *Throughout Chapter 4, we assume that the representation of the geometry as a composition of several subdomains (and thus also the initial discretization) is given. We assume that Assumptions 3.4, 3.6, 3.11, and 3.14 hold for the respective subdomains. Furthermore, we assume that the interfaces on the initial discretization are fully matching.*

Assume that the linear form $\langle f, \cdot\rangle$ can be assembled from contributions $\langle f^{(k)}, \cdot\rangle$ on $\Omega^{(k)}$, and let $a^{(k)}(\cdot,\cdot)$ denote the restriction of $a(\cdot,\cdot)$ to $\Omega^{(k)}$. Then, we can discretize the original minimization problem (2.7) as a sum of local contributions:

Find u_h such that

$$u_h = \operatorname*{arg\,min}_{v_h \in V_{gh}} \sum_{k=1}^{N} \Big(\tfrac{1}{2} a^{(k)}(v_h, v_h) - \langle f^{(k)}, v_h \rangle \Big). \tag{4.9}$$

Condition (ii) for the fully matching setting implies that each DOF $u_{\boldsymbol{i}}^{(k)}$, $\boldsymbol{i} \in \mathcal{B}(k,\ell)$, can be associated with a DOF $u_{\boldsymbol{j}}^{(\ell)}$, $\boldsymbol{j} \in \mathcal{B}(\ell,k)$, such that the corresponding basis functions match as in (4.8). If we identify the DOF corresponding to these matching basis functions, then, together with the remaining DOF, we get a representation of the space V_h in (4.3). Employing a suitable *global* numbering for the identified DOF, we can rewrite (4.9) in the form

$$\mathbf{K}\mathbf{u} = \mathbf{f}, \tag{4.10}$$

as discussed in Section 2.2.4. This global system can be solved by standard sparse *LU*-factorization, see, e.g., [30]. However, it is well known that for large problem size, the memory requirement and the runtime complexity of direct solvers are inefficient. Alternatively, one can use efficient iterative solvers such as conjugate gradient methods with appropriate preconditioners, see, e.g., [86]. In the case of standard FEM discretizations, such preconditioners have been well studied in the literature, see, e.g., [96] for geometric and algebraic multigrid methods, and see [95] for domain decomposition methods.

It is important to note that by assembling the system matrix $\mathbf{K}$ from the subdomain contributions, the structural (subdomain-wise) properties of the problem are lost, which are hard to regain from $\mathbf{K}$ alone. To alleviate this difficulty, in the following section we present a solver (inspired by the FETI methods [35, 39, 40, 41, 72, 95]) which inherently uses the local structure of (4.9). This approach is very suitable for parallelization, and moreover, since it mainly uses solvers on the local subdomains, their tensor product structure can be exploited (e.g. using wavelets or FFT).

4.2 Solver design

The techniques presented in this section follow FETI methods, see [35, 39, 40, 41, 72] and the monographs [74, 95]. In the IETI method, we work in the space

$$\Pi V_h = \prod_{k=1}^{N} V_h^{(k)}$$

as defined in (4.2). Since these functions are, in general, discontinuous across subdomain interfaces, we need to impose the continuity conditions separately. In the following, let $v^{(k)}$ denote the k-th component of a function $v \in \Pi V_h$.

4.2.1 Continuity constraints

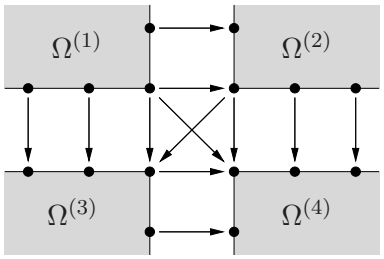
Note that, in the index set $\mathcal{C}_\Gamma$, every interface $\Gamma^{(k,\ell)}$ is represented twice. Therefore, we define the set

$$\mathcal{C} = \{(k,\ell) \in \mathcal{C}_\Gamma : \ k < \ell\}, \tag{4.11}$$

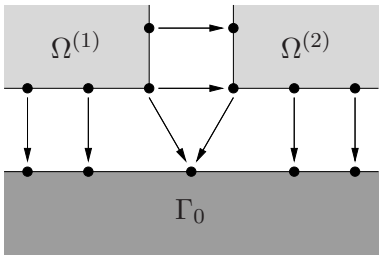
where each interface is represented only once. For each pair $(k,\ell) \in \mathcal{C}$, we say that $\Omega^{(k)}$ is the *master subdomain* and $\Omega^{(\ell)}$ the *slave subdomain*. For all $(k,\ell) \in \mathcal{C}$ and a fixed index $\boldsymbol{i} \in \mathcal{B}(k,\ell)$, we can rewrite (4.8) in the following form:

$$\check{R}_{\boldsymbol{i}}^{(k)}|_{\Gamma^{(k,\ell)}} = \sum_{\boldsymbol{j} \in \mathcal{B}(\ell,k)} z_{\boldsymbol{i},\boldsymbol{j}}^{(k,\ell)} \check{R}_{\boldsymbol{j}}^{(\ell)}|_{\Gamma^{(k,\ell)}}, \tag{4.12}$$

where, for a fixed $\boldsymbol{i} \in \mathcal{B}(k,\ell)$, all coefficients $z_{\boldsymbol{i},\boldsymbol{j}}^{(k,\ell)}$ are zero, except for one coefficient which is 1. Note that the generalization of (4.8) given in (4.12) might seem superfluous in the fully matching setting, but it will be needed in Section 4.3 in the context of cases which are not fully matching. There, we will also adapt the definition of $\mathcal{C}$.



(a) Fully redundant coupling at a subdomain vertex.



(b) Incorporation of essential boundary conditions by coupling to virtual neighbour subdomains.

Figure 4.2: Illustration of fully redundant coupling and all floating setting. Arrows indicate coupling conditions and point from master subdomain to slave subdomain.

We can collect the coefficients $z_{\boldsymbol{i},\boldsymbol{j}}^{(k,\ell)}$ in the permutation matrix

$$\mathbf{Z}^{(k,\ell)} = (z_{\boldsymbol{i},\boldsymbol{j}}^{(k,\ell)})_{\substack{\boldsymbol{i}\in\mathcal{B}(k,\ell)\\ \boldsymbol{j}\in\mathcal{B}(\ell,k)}}. \tag{4.13}$$

In the case of a subdomain vertex, where $\#\mathcal{B}(k,\ell) = 1$, the matrix $\mathbf{Z}^{(i,j)}$ has only one entry which is 1. Figure 4.2(a) illustrates the coupling conditions between subdomains and at subdomain vertices.

To guarantee the continuity of u_h across all interfaces, we impose the following condition on the DOF associated with interfaces, i.e., for all $(k,\ell) \in \mathcal{C}$:

$$u_{\boldsymbol{i}}^{(k)} - \sum_{\boldsymbol{j}\in\mathcal{B}(\ell,k)} z_{\boldsymbol{i},\boldsymbol{j}}^{(k,\ell)} u_{\boldsymbol{j}}^{(\ell)} = 0, \quad \forall \boldsymbol{i} \in \mathcal{B}(k,\ell). \tag{4.14}$$

The incorporation of essential boundary conditions is done in a similar way, see the all-floating BETI method [72] and the total FETI method [35]. We denote the interface between $\partial\Omega^{(k)}$ and Γ_0 by

$$\Gamma^{(k,0)} = \partial\Omega^{(k)} \cap \Gamma_0.$$

Similar to (4.7), we collect the DOF of $u^{(k)}$ that are associated with Γ_0, i.e., with the interface $\Gamma^{(k,0)}$, in the following index set:

$$\mathcal{D}(k) \;=\; \left\{ \boldsymbol{i} \in \mathcal{I}^{(k)} :\; \operatorname{supp} \check{R}_{\boldsymbol{i}}^{(k)} \cap \Gamma^{(k,0)} \neq \emptyset \right\}.$$

Recall from Section 2.2.4 that we assume that there exists a function $\check{g} \in V_h$ with $\check{g}|_{\Gamma_0} = g_0|_{\Gamma_0}$. Hence, we can write

$$g_0\big|_{\Gamma^{(k,0)}} = \sum_{\boldsymbol{i}\in\mathcal{D}(k)} g_{\boldsymbol{i}}^{(k)} \check{R}_{\boldsymbol{i}}^{(k)}\big|_{\Gamma^{(k,0)}}$$

with real-valued coefficients $g_{\boldsymbol{i}}^{(k)}$. We incorporate essential boundary conditions by imposing the following constraints on the DOF that are associated with Γ_0, i.e., for all k such that $\Gamma^{(k,0)} \neq \emptyset$:

$$u_{\boldsymbol{i}}^{(k)} = g_{\boldsymbol{i}}^{(k)} \quad \forall \boldsymbol{i} \in \mathcal{D}(k). \tag{4.15}$$

These constraints can be thought of as continuity between the physical subdomains and a virtual neighbour subdomain (see Figure 4.2(b) for an illustration).

Let J denote the total number of constraints of the form (4.14) and (4.15):

$$J = \sum_{(k,\ell)\in\mathcal{C}} \#\mathcal{B}(k,\ell) + \sum_{k=1}^{N} \#\mathcal{D}(k).$$

We assume a fixed numbering of these constraints and introduce the following notation. For a vector $y \in \mathbb{R}^J$, $y_{\boldsymbol{i}}^{(k,\ell)}$ denotes the component corresponding to the

constraint (4.14), and $y_{\boldsymbol{i}}^{(k,D)}$ denotes the component corresponding to the constraint (4.15). We define the *jump operator* B as follows:

$$\begin{aligned} B:\ & \Pi V_h \to \mathbb{R}^J \\ (Bu)_{\boldsymbol{i}}^{(k,\ell)} =\ & u_{\boldsymbol{i}}^{(k)} - \sum_{\boldsymbol{j}\in\mathcal{B}(\ell,k)} z_{\boldsymbol{i},\boldsymbol{j}}^{(k,\ell)} u_{\boldsymbol{j}}^{(\ell)} \qquad (4.16) \\ (Bu)_{\boldsymbol{i}}^{(k,D)} =\ & u_{\boldsymbol{i}}^{(k)}. \qquad (4.17) \end{aligned}$$

Hence, the conditions for C^0-continuity (4.14) and the incorporation of essential boundary conditions (4.15) read

$$Bu = \mathbf{b}, \tag{4.18}$$

where the entry of the vector $\mathbf{b} \in \mathbb{R}^J$ is zero when corresponding to an interface condition (4.16), and it equals $g_{\boldsymbol{i}}^{(k)}$ when corresponding to an essential boundary condition (4.17). For $\check{g} \in V_h$ with $\check{g}|_{\Gamma_0} = g_0|_{\Gamma_0}$, we have $\mathbf{b} = B\check{g}$. Note that the linear operator B can be represented by a signed Boolean matrix. With (4.18), we obtain the following restricted minimization problem which is equivalent to (4.9):

Find u such that

$$u = \underset{\substack{v\in\Pi V_h \\ Bv=\mathbf{b}}}{\arg\min} \sum_{k=1}^{N} \Big(\tfrac{1}{2} a^{(k)}(v^{(k)}, v^{(k)}) - \langle f^{(k)}, v^{(k)} \rangle \Big). \tag{4.19}$$

Remark 4.3. *We assume that the computational domain is represented as a collection of several patches which are joined with C^0-smoothness along their interfaces. These patches simultaneously serve as the subdomains for the IETI method. The approach can be extended to unstructured meshes, such as T-spline representations [9, 87, 88], which are also used in isogeometric analysis. Similar to the case of classical FETI, a computational domain represented by a T-spline mesh can be split into subdomains. The coupling of the DOF across the interfaces needs to take all test functions into account that do not vanish on the interface. Typically, these DOF form a strip whose width increases with the degree of smoothness of the T-spline representation. This is different both from the classical FETI method for piecewise linear finite elements and from the multi-patch NURBS discretization with C^0-continuity, where these DOF are arranged along lines. Consequently, when extending the framework to unstructured meshes in IGA, a larger number of Lagrange multipliers will be needed as compared with the C^0 case. Moreover, for general T-spline meshes, it is no longer possible to benefit from the simple tensor-product structure of the subdomains.*

4.2.2 Saddle point formulation

Using the local basis of each subdomain space $V_h^{(k)}$, each function $v^{(k)} \in V_h^{(k)}$ is uniquely represented by a vector $\mathbf{v}^{(k)}$. Correspondingly, each function $v \in \Pi V_h$ has a representation as a vector $\mathbf{v}$ of the form

$$\mathbf{v} = (\mathbf{v}^{(1)}, \dots, \mathbf{v}^{(N)})^T \tag{4.20}$$

whose blocks are the local vectors $\mathbf{v}^{(k)}$. Let $\mathbf{K}^{(k)}$ denote the stiffness matrix corresponding to the local bilinear form $a^{(k)}(\cdot,\cdot)$ and define

$$\mathbf{K} = \begin{pmatrix} \mathbf{K}^{(1)} & & 0 \\ & \ddots & \\ 0 & & \mathbf{K}^{(N)} \end{pmatrix}.$$

Analogously, let $\mathbf{f}$ denote the load vector whose blocks $\mathbf{f}^{(k)}$ correspond to the local subdomain load vectors, and let $\mathbf{B}$ be the matrix representation of the jump operator B.

The minimization problem (4.19) is then equivalent to the following saddle point problem:

Find $u \in \Pi V_h$ with the vector representation $\mathbf{u}$ as in (4.20) and Lagrange multipliers $\boldsymbol{\lambda} \in \mathbb{R}^J$, such that

$$\begin{pmatrix} \mathbf{K} & \mathbf{B}^T \\ \mathbf{B} & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \mathbf{f} \\ \mathbf{b} \end{pmatrix}. \tag{4.21}$$

We note that even though $\boldsymbol{\lambda}$ is only unique up to an element from $\ker \mathbf{B}^T$, the solution $\mathbf{u}$ is unique.

The common strategy of FETI-type methods is to reduce (4.21) to an equation that involves only $\boldsymbol{\lambda}$. This is not straightforward, since in the case of our model problems (see Section 2.2.1) the local matrices $\mathbf{K}^{(k)}$ are not invertible. More precisely, in the scalar elliptic case (I) the kernel is spanned by the constant functions, and for the two-dimensional linearized elasticity problem (II), the kernel is spanned by three rigid body modes.

In the classical FETI methods [41] and the total FETI method [35], additional unknowns are introduced that span the kernel. We will, however, follow the dual-primal FETI (FETI-DP) method, see [39] and [95, Sect. 6.4], thus obtaining IETI-DP.

4.2.3 Dual-primal formulation

Recall that we only consider NURBS geometry mappings with open knot vectors. Therefore, at every vertex of the parameter domain $\widehat{\Omega}$, there is exactly one multi-index $\boldsymbol{i}_0 \in \mathcal{I}^{(k)}$, such that $R^{(k)}_{\boldsymbol{i}_0}$ at this vertex is 1, while all other basis functions are zero (see Remark 3.10). Hence, we can distinguish DOF that are associated with the vertices of the parameter domain. Such DOF that are associated with the vertices of a parameter domain are called *primal* DOF. All other DOF are referred to as *non-primal* or *remaining* DOF.

We define the following subset of ΠV_h,

$$\widetilde{W}_h = \{v \in \Pi V_h : \ v \text{ is continuous at all subdomain vertices}\}.$$

To achieve the continuity above, we identify all the primal DOF that are associated with a common point in the physical domain, and we fix a *global numbering* of these

primal DOF. Then, a function $v \in \widetilde{W}_h$ can be uniquely represented by a vector $\widetilde{\mathbf{v}}$ of the form

$$\widetilde{\mathbf{v}} = (\widetilde{\mathbf{v}}_P, \widetilde{\mathbf{v}}_R)^T = (\widetilde{\mathbf{v}}_P, \widetilde{\mathbf{v}}_R^{(1)}, \ldots, \widetilde{\mathbf{v}}_R^{(N)})^T, \tag{4.22}$$

where the subscripts P and R refer to *primal* and *remaining* DOF, respectively. Note that $\mathbf{v}$ in (4.20) and $\widetilde{\mathbf{v}}$ in (4.22) are different vector representations of the same function $v \in \widetilde{W}_h \subset \Pi V_h$.

Let $\widetilde{\mathbf{B}}$ denote the jump operator on $\widetilde{W}_h$ defined by

$$\widetilde{\mathbf{B}}\widetilde{\mathbf{u}} = Bu,$$

where $\widetilde{\mathbf{u}}$ is the representation of u in the form of (4.22). Analogously to (4.22), we can write

$$\widetilde{\mathbf{B}} = (\widetilde{\mathbf{B}}_P, \widetilde{\mathbf{B}}_R) = (\widetilde{\mathbf{B}}_P, \widetilde{\mathbf{B}}_R^{(1)}, \ldots, \widetilde{\mathbf{B}}_R^{(N)}).$$

Remark 4.4. *Note that we can distinguish between primal and remaining DOF in such a straightforward manner only due to the fact that we are using open knot vectors (see the discussion in Section 3.1.1.1 and Remark 3.10). Even though NURBS basis functions are not a nodal basis in general, this special property makes the dual-primal approach simple to implement and thus very appealing.*

4.2.3.1 Setting up the global system

Since the solution $u \in \widetilde{W}_h$, we can replace the space ΠV_h in (4.21) by $\widetilde{W}_h$. In the following, we will derive an equivalent saddle point formulation.

By rearranging the local DOF $\mathbf{u}^{(k)}$ in such a way that the primal DOF come first, the subdomain stiffness matrix $\mathbf{K}^{(k)}$ and the load vector $\mathbf{f}^{(k)}$ take the form

$$\widetilde{\mathbf{K}}^{(k)} = \begin{pmatrix} \widetilde{\mathbf{K}}_{PP}^{(k)} & \widetilde{\mathbf{K}}_{PR}^{(k)} \\ \widetilde{\mathbf{K}}_{RP}^{(k)} & \widetilde{\mathbf{K}}_{RR}^{(k)} \end{pmatrix}, \quad \widetilde{\mathbf{f}}^{(k)} = \begin{pmatrix} \widetilde{\mathbf{f}}_P^{(k)} \\ \widetilde{\mathbf{f}}_R^{(k)} \end{pmatrix}. \tag{4.23}$$

From these local contributions, we obtain the global stiffness matrix $\widetilde{\mathbf{K}}$ and the global load vector $\widetilde{\mathbf{f}}$

$$\widetilde{\mathbf{K}} = \begin{pmatrix} \widetilde{\mathbf{K}}_{PP} & \widetilde{\mathbf{K}}_{PR} \\ \widetilde{\mathbf{K}}_{RP} & \widetilde{\mathbf{K}}_{RR} \end{pmatrix}, \quad \widetilde{\mathbf{f}} = \begin{pmatrix} \widetilde{\mathbf{f}}_P \\ \widetilde{\mathbf{f}}_R \end{pmatrix}. \tag{4.24}$$

Note that, due to the identification of the primal DOF, the components carrying a subscript P in (4.24) are *assembled* from local contributions. $\widetilde{\mathbf{K}}_{RR}$ and $\widetilde{\mathbf{f}}_R$ have the form

$$\widetilde{\mathbf{K}}_{RR} = \begin{pmatrix} \widetilde{\mathbf{K}}_{RR}^{(1)} & & 0 \\ & \ddots & \\ 0 & & \widetilde{\mathbf{K}}_{RR}^{(N)} \end{pmatrix}, \; \widetilde{\mathbf{f}}_{RR} = \begin{pmatrix} \widetilde{\mathbf{f}}_R^{(1)} \\ \vdots \\ \widetilde{\mathbf{f}}_R^{(N)} \end{pmatrix}.$$

The coupling conditions of the form (4.16) can be neglected at the primal DOF, but it has no effect on the algorithm if they are included. Depending on the implementation of the jump operator, one might decide to keep them or not.

With the same steps as before, we arrive at the following saddle point problem which is equivalent to (4.21):

Find $u \in \widetilde{W}_h$, represented by $\widetilde{\mathbf{u}}$ as in (4.22), and Lagrange multipliers $\boldsymbol{\lambda} \in \mathbb{R}^J$, such that

$$\begin{pmatrix} \widetilde{\mathbf{K}} & \widetilde{\mathbf{B}}^T \\ \widetilde{\mathbf{B}} & 0 \end{pmatrix} \begin{pmatrix} \widetilde{\mathbf{u}} \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \widetilde{\mathbf{f}} \\ \mathbf{b} \end{pmatrix}. \tag{4.25}$$

Note that the matrix $\widetilde{\mathbf{K}}$ in (4.25) is singular, because the space $\widetilde{W}_h$ has no restrictions on Γ_0 for our model problems. In the next section, we will incorporate essential boundary conditions at those primal DOF that are associated with Γ_0. This incorporation will lead to a non-singular matrix.

4.2.3.2 Essential boundary conditions

We distinguish between *essential* primal DOF associated with Γ_0, and *floating* primal DOF that are in the interior of Ω or associated with Γ_1. We indicate this with the subscripts d and f, respectively. We assume for simplicity that in the vector $\widetilde{\mathbf{u}}$ the essential primal DOF are listed first, i.e.,

$$\widetilde{\mathbf{u}} = (\widetilde{\mathbf{u}}_d, \widetilde{\mathbf{u}}_f, \widetilde{\mathbf{u}}_R)^T = (\widetilde{\mathbf{u}}_d, \widetilde{\mathbf{u}}_f, \widetilde{\mathbf{u}}_R^{(1)}, \ldots, \widetilde{\mathbf{u}}_R^{(N)})^T \tag{4.26}$$

and

$$\widetilde{\mathbf{K}} = \begin{pmatrix} \widetilde{\mathbf{K}}_{dd} & \widetilde{\mathbf{K}}_{df} & \widetilde{\mathbf{K}}_{dR} \\ \widetilde{\mathbf{K}}_{fd} & \widetilde{\mathbf{K}}_{ff} & \widetilde{\mathbf{K}}_{fR} \\ \widetilde{\mathbf{K}}_{Rd} & \widetilde{\mathbf{K}}_{Rf} & \widetilde{\mathbf{K}}_{RR} \end{pmatrix}, \quad \widetilde{\mathbf{f}} = \begin{pmatrix} \widetilde{\mathbf{f}}_d \\ \widetilde{\mathbf{f}}_f \\ \widetilde{\mathbf{f}}_R \end{pmatrix},$$
$$\widetilde{\mathbf{B}} = \begin{pmatrix} \widetilde{\mathbf{B}}_d & \widetilde{\mathbf{B}}_f & \widetilde{\mathbf{B}}_R \end{pmatrix}.$$

Let $\widetilde{\mathbf{g}}_d$ be the vector whose entries are the values of g_0 at the essential primal DOF. Since $\widetilde{\mathbf{u}}_d = \widetilde{\mathbf{g}}_d$, the saddle point problem (4.25) is equivalent to the following problem:

Find $u \in \widetilde{W}_h$, represented by $\widetilde{\mathbf{u}}$ as in (4.26), and Lagrange multipliers $\boldsymbol{\lambda} \in \mathbb{R}^J$, such that

$$\begin{pmatrix} \overline{\mathbf{K}} & \overline{\mathbf{B}}^T \\ \overline{\mathbf{B}} & 0 \end{pmatrix} \begin{pmatrix} \widetilde{\mathbf{u}} \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \overline{\mathbf{f}} \\ \overline{\mathbf{b}} \end{pmatrix}, \tag{4.27}$$

where

$$\overline{\mathbf{K}} = \begin{pmatrix} \mathbf{I} & 0 & 0 \\ 0 & \widetilde{\mathbf{K}}_{ff} & \widetilde{\mathbf{K}}_{fR} \\ 0 & \widetilde{\mathbf{K}}_{Rf} & \widetilde{\mathbf{K}}_{RR} \end{pmatrix}, \tag{4.28}$$

$$\overline{\mathbf{f}} = \begin{pmatrix} \widetilde{\mathbf{g}}_d \\ \widetilde{\mathbf{f}}_f - \widetilde{\mathbf{K}}_{fd}\widetilde{\mathbf{g}}_d \\ \widetilde{\mathbf{f}}_R - \widetilde{\mathbf{K}}_{Rd}\widetilde{\mathbf{g}}_d \end{pmatrix}, \tag{4.29}$$

$$\overline{\mathbf{B}} = \begin{pmatrix} 0 & \widetilde{\mathbf{B}}_f & \widetilde{\mathbf{B}}_R \end{pmatrix}, \tag{4.30}$$

$$\overline{\mathbf{b}} = \mathbf{b} - \widetilde{\mathbf{B}}_d\widetilde{\mathbf{g}}_d \tag{4.31}$$

and $\mathbf{I}$ is the identity matrix. We see that each entry of $\overline{\mathbf{b}}$ corresponding to a multiplier acting on an essential primal DOF vanishes. Also, these multipliers are superfluous as they do not influence the solution $\widetilde{\mathbf{u}}$ and can be left out completely.

As before, we denote the components of $\overline{\mathbf{K}}$, $\overline{\mathbf{f}}$, and $\overline{\mathbf{B}}$ in (4.28)–(4.30) that correspond to primal and remaining DOF with the subscripts P and R, respectively. Hence,

$$\begin{aligned}
\overline{\mathbf{K}}_{PP} &= \begin{pmatrix} \mathbf{I} & 0 \\ 0 & \widetilde{\mathbf{K}}_{ff} \end{pmatrix}, & \overline{\mathbf{K}}_{PR} &= \begin{pmatrix} 0 \\ \widetilde{\mathbf{K}}_{fR} \end{pmatrix}, \\
\overline{\mathbf{K}}_{RP} &= \begin{pmatrix} 0 & \widetilde{\mathbf{K}}_{Rf} \end{pmatrix}, & \overline{\mathbf{K}}_{RR} &= \widetilde{\mathbf{K}}_{RR}, \\
\overline{\mathbf{f}}_P &= \begin{pmatrix} \widetilde{\mathbf{g}}_d \\ \widetilde{\mathbf{f}}_f - \widetilde{\mathbf{K}}_{fd}\widetilde{\mathbf{g}}_d \end{pmatrix}, & \overline{\mathbf{f}}_R &= \widetilde{\mathbf{f}}_R - \widetilde{\mathbf{K}}_{Rd}\widetilde{\mathbf{g}}_d, \\
\overline{\mathbf{B}}_P &= \begin{pmatrix} 0 & \widetilde{\mathbf{B}}_f \end{pmatrix}, & \overline{\mathbf{B}}_R &= \widetilde{\mathbf{B}}_R.
\end{aligned}$$

Again, the coupling conditions of the form (4.16) and (4.17) can be neglected at all the essential primal DOF, but it has no effect on the algorithm if they remain.

Remark 4.5. *Similar to the construction* (4.28)–(4.31)*, we can also directly incorporate the essential boundary conditions at the remaining non-primal essential DOF. If this is done, the corresponding multipliers can again be left out. This approach would be closer to the original FETI-DP method as proposed in [39].*

4.2.3.3 Dual problem

In our model problems the matrix $\overline{\mathbf{K}}$ in (4.28) is invertible, and the first line of (4.27) yields

$$\widetilde{\mathbf{u}} = \overline{\mathbf{K}}^{-1}(\overline{\mathbf{f}} - \overline{\mathbf{B}}^T\boldsymbol{\lambda}). \tag{4.32}$$

Inserting this identity into the second line of (4.27), we obtain the dual problem

$$\mathbf{F}\boldsymbol{\lambda} \;=\; \mathbf{d} \tag{4.33}$$

with $\mathbf{F} = \overline{\mathbf{B}}\,\overline{\mathbf{K}}^{-1}\overline{\mathbf{B}}^T$ and $\mathbf{d} = \overline{\mathbf{B}}\,\overline{\mathbf{K}}^{-1}\overline{\mathbf{f}} - \overline{\mathbf{b}}$. To realize the application of $\overline{\mathbf{K}}^{-1}$, we use the block factorization

$$\overline{\mathbf{K}}^{-1} = \begin{pmatrix} \mathbf{I} & 0 \\ -\overline{\mathbf{K}}_{RR}^{-1}\overline{\mathbf{K}}_{RP} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \overline{\mathbf{S}}_{PP}^{-1} & 0 \\ 0 & \overline{\mathbf{K}}_{RR}^{-1} \end{pmatrix} \begin{pmatrix} \mathbf{I} & -\overline{\mathbf{K}}_{PR}\overline{\mathbf{K}}_{RR}^{-1} \\ 0 & \mathbf{I} \end{pmatrix} \tag{4.34}$$

where

$$\overline{\mathbf{S}}_{PP} = \overline{\mathbf{K}}_{PP} - \overline{\mathbf{K}}_{PR}\overline{\mathbf{K}}_{RR}^{-1}\overline{\mathbf{K}}_{RP}.$$

Recall that $\overline{\mathbf{K}}_{RR}$ is block diagonal. Hence, applying $\overline{\mathbf{K}}_{RR}^{-1}$ corresponds to solving local problems independently on each subdomain, e.g., by (sparse) LU-factorization [30]. Note that $\overline{\mathbf{K}}_{RR}^{-1}$ appears three times in (4.34), but it has to be applied only twice, because two of the applications of $\overline{\mathbf{K}}_{RR}^{-1}$ are on the same vector.

The matrix $\overline{\mathbf{S}}_{PP}$ can be assembled from local contributions

$$\overline{\mathbf{S}}_{PP}^{(k)} = \overline{\mathbf{K}}_{PP}^{(k)} - \overline{\mathbf{K}}_{PR}^{(k)}(\overline{\mathbf{K}}_{RR}^{(k)})^{-1}\overline{\mathbf{K}}_{RP}^{(k)}.$$

It can be shown that $\overline{\mathbf{S}}_{PP}$ is sparse and that it can be factorized using standard sparse LU-factorization [30]. The size of $\overline{\mathbf{S}}_{PP}$ is determined by the number of primal DOF. Since we only have primal DOF at the subdomain vertices, their number is bounded by $4N$. Typically, the number of subdomains, and therefore the size of $\overline{\mathbf{S}}_{PP}$, is much smaller than the size of $\mathbf{K}_{RR}^{(k)}$, but even in the case of many subdomains $\overline{\mathbf{S}}_{PP}$ is sparse.

We can solve the symmetric and positive definite system (4.33) for $\boldsymbol{\lambda}$ by a CG algorithm. Once we have obtained $\boldsymbol{\lambda}$, we can compute $\widetilde{\mathbf{u}}_P$ as follows. We write (4.32) in the form

$$\begin{pmatrix} \widetilde{\mathbf{u}}_P \\ \widetilde{\mathbf{u}}_R \end{pmatrix} = \overline{\mathbf{K}}^{-1} \begin{pmatrix} \overline{\mathbf{f}}_P - \overline{\mathbf{B}}_P^T\boldsymbol{\lambda} \\ \overline{\mathbf{f}}_R - \overline{\mathbf{B}}_R^T\boldsymbol{\lambda} \end{pmatrix} \tag{4.35}$$

and see that $\widetilde{\mathbf{u}}_P$ is given by the first component of the right-hand side in (4.35). Since

$$\begin{pmatrix} \mathbf{I} & 0 \\ -\overline{\mathbf{K}}_{RR}^{-1}\overline{\mathbf{K}}_{RP} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \overline{\mathbf{S}}_{PP}^{-1} & 0 \\ 0 & \overline{\mathbf{K}}_{RR}^{-1} \end{pmatrix} = \begin{pmatrix} \overline{\mathbf{S}}_{PP}^{-1} & 0 \\ * & * \end{pmatrix},$$

we obtain from (4.34) and (4.35),

$$\begin{aligned} \begin{pmatrix} \widetilde{\mathbf{u}}_P \\ \widetilde{\mathbf{u}}_R \end{pmatrix} &= \overline{\mathbf{K}}^{-1} \begin{pmatrix} \overline{\mathbf{f}}_P - \overline{\mathbf{B}}_P^T\boldsymbol{\lambda} \\ \overline{\mathbf{f}}_R - \overline{\mathbf{B}}_R^T\boldsymbol{\lambda} \end{pmatrix} \\ &= \begin{pmatrix} \overline{\mathbf{S}}_{PP}^{-1} & 0 \\ * & * \end{pmatrix} \begin{pmatrix} \mathbf{I} & -\overline{\mathbf{K}}_{PR}\overline{\mathbf{K}}_{RR}^{-1} \\ 0 & \mathbf{I} \end{pmatrix} \begin{pmatrix} \overline{\mathbf{f}}_P - \overline{\mathbf{B}}_P^T\boldsymbol{\lambda} \\ \overline{\mathbf{f}}_R - \overline{\mathbf{B}}_R^T\boldsymbol{\lambda} \end{pmatrix} \\ &= \begin{pmatrix} \overline{\mathbf{S}}_{PP}^{-1} & 0 \\ * & * \end{pmatrix} \begin{pmatrix} (\overline{\mathbf{f}}_P - \overline{\mathbf{B}}_P^T\boldsymbol{\lambda}) - \overline{\mathbf{K}}_{PR}\overline{\mathbf{K}}_{RR}^{-1}(\overline{\mathbf{f}}_R - \overline{\mathbf{B}}_R^T\boldsymbol{\lambda}) \\ \overline{\mathbf{f}}_R - \overline{\mathbf{B}}_R^T\boldsymbol{\lambda} \end{pmatrix}. \end{aligned} \tag{4.36}$$

The first component in (4.36), and thus $\widetilde{\mathbf{u}}_P$, is given by

$$\widetilde{\mathbf{u}}_P = \overline{\mathbf{S}}_{PP}^{-1}\Big((\overline{\mathbf{f}}_P - \overline{\mathbf{B}}_P^T\boldsymbol{\lambda}) - \overline{\mathbf{K}}_{PR}\overline{\mathbf{K}}_{RR}^{-1}(\overline{\mathbf{f}}_R - \overline{\mathbf{B}}_R^T\boldsymbol{\lambda})\Big). \tag{4.37}$$

We write the first line of (4.27) in more detail,

$$\begin{pmatrix} \overline{\mathbf{K}}_{PP} & \overline{\mathbf{K}}_{PR} \\ \overline{\mathbf{K}}_{RP} & \overline{\mathbf{K}}_{RR} \end{pmatrix} \begin{pmatrix} \widetilde{\mathbf{u}}_P \\ \widetilde{\mathbf{u}}_R \end{pmatrix} + \begin{pmatrix} \overline{\mathbf{B}}_P^T \\ \overline{\mathbf{B}}_R^T \end{pmatrix} \boldsymbol{\lambda} = \begin{pmatrix} \overline{\mathbf{f}}_P \\ \overline{\mathbf{f}}_R \end{pmatrix}, \tag{4.38}$$

and see from the second line of (4.38) that the equality

$$\overline{\mathbf{K}}_{RP}\widetilde{\mathbf{u}}_P + \overline{\mathbf{K}}_{RR}\widetilde{\mathbf{u}}_R + \overline{\mathbf{B}}_R^T\boldsymbol{\lambda} = \overline{\mathbf{f}}_R \tag{4.39}$$

holds. Hence, having computed $\widetilde{\mathbf{u}}_P$ and $\boldsymbol{\lambda}$, the remaining local solutions are then given by

$$\widetilde{\mathbf{u}}_R = \overline{\mathbf{K}}_{RR}^{-1}\Big(\overline{\mathbf{f}}_R - \overline{\mathbf{B}}_R^T\boldsymbol{\lambda} - \overline{\mathbf{K}}_{RP}\widetilde{\mathbf{u}}_P\Big). \tag{4.40}$$

Again, since $\overline{\mathbf{K}}_{RR}$ is block-diagonal, the application of $\overline{\mathbf{K}}_{RR}^{-1}$ corresponds to solving independent local problems on each subdomain.

In [40], the unpreconditioned interface problem (4.33) is discussed for the classical FETI method, and it is shown that the condition number is of order

$$\kappa(\mathbf{F}) = \mathcal{O}(H/h), \tag{4.41}$$

where H and h denote the characteristic subdomain size and the finite element mesh size, respectively. The numerical tests presented in Section 4.4 indicate that the IETI method behaves similarly. In the next section, following [39], we define a preconditioner for the interface problem which will be used for the numerical examples in Section 4.4.

4.2.4 Preconditioner

Our construction follows the scaled Dirichlet preconditioner that was introduced in [40, 53, 85] and extended to the dual-primal formulation in [39, 54]. We indicate *interior* DOF with the subscript I, and the DOF associated with the *boundary* $\partial\Omega^{(k)}$ of a subdomain with the subscript B. Assume that the DOF are now numbered such that the interior DOF are listed first, then the local stiffness matrix $\mathbf{K}^{(i)}$ takes the form

$$\mathbf{K}^{(k)} = \begin{pmatrix} \mathbf{K}_{II}^{(k)} & \mathbf{K}_{IB}^{(k)} \\ \mathbf{K}_{BI}^{(k)} & \mathbf{K}_{BB}^{(k)} \end{pmatrix}. \tag{4.42}$$

The dual-primal Dirichlet preconditioner is defined by

$$\mathbf{M}^{-1} = \sum_{i=1}^{N} \mathbf{D}^{(k)}\,\mathbf{B}^{(k)} \begin{pmatrix} 0 & 0 \\ 0 & \mathbf{S}_{BB}^{(k)} \end{pmatrix} \mathbf{B}^{(k)\,T}\,\mathbf{D}^{(k)}, \tag{4.43}$$

where

$$\mathbf{S}_{BB}^{(k)} = \mathbf{K}_{BB}^{(k)} - \mathbf{K}_{BI}^{(k)}(\mathbf{K}_{II}^{(k)})^{-1}\mathbf{K}_{IB}^{(k)}.$$

Since $\mathbf{K}_{II}^{(k)}$ is the local stiffness matrix of $\Omega^{(k)}$ with all boundary DOF fixed, it can be factorized as easily and cheaply as $\mathbf{K}_{RR}^{(k)}$. The matrix $\mathbf{B}^{(k)}$ in (4.43) is the restriction of $\mathbf{B}$ to the interface conditions associated with $\Omega^{(k)}$. The matrix $\mathbf{D}^{(k)}$ is a scaled diagonal matrix of size $J \times J$, where J is the number of Lagrange multipliers. Its entries are

$$(\mathbf{D}^{(k)})_{ii} = 1/\operatorname{mult}(i),$$

where $\operatorname{mult}(i)$ is the number of subdomains which have interfaces associated with the Lagrange multiplier λ_i. In particular, $\operatorname{mult}(i)$ takes the following values:

$\text{mult}(i) = 1$, if λ_i corresponds to an essential boundary condition.

$\text{mult}(i) = 2$, if λ_i corresponds to a coupling condition that does not involve a subdomain vertex.

$\text{mult}(i) = m_i \geq 2$, if λ_i corresponds to a coupling condition that involves a subdomain vertex, and where m_i denotes the number of subdomains which share this vertex (i.e, $m_i \leq N$).

These scalings can, e.g., be found in [53, 54], where the authors show that certain jumps in the diffusion coefficient (problem (I)) or the Lamé parameters (problem (II)) can be treated robustly.

In [54], it was shown that the condition number of the preconditioned FETI-DP interface problem behaves like

$$\kappa(\mathbf{M}^{-1}\mathbf{F}) = \mathcal{O}((1 + \log(H/h)^2),$$

where H and h are as defined at the end of Section 4.2.3. In [15], it was shown that this bound also holds for isogeometric BDDC preconditioners. Since BDDC and FETI-DP preconditioners have the same essential spectrum when the primal DOF are chosen in the same way (see [20, 65]), it is expected that this bound also holds for the IETI-DP method. This is confirmed by the numerical results presented in Section 4.4.

4.2.5 Isogeometric tearing and interconnecting algorithm

To summarize, the overall IETI-DP algorithm is presented in Algorithm 4.1.

4.3 Refinement options

As briefly mentioned in Section 3.3.4, the tensor-product structure of NURBS basis functions is inconvenient for local refinement. The insertion of a knot affects the whole domain and may introduce superfluous DOF.

However, the IETI-approach introduces some possibilities for restricting the refinement to one or a few subdomains (of many), even when working with tensor-product NURBS basis functions and straightforward knot insertion. We will sketch two such methods: h-refinement on one subdomain and refinement by substructuring (see Figure 4.3). Note that, in both cases, we assume that the initial setting is fully matching.

4.3.1 h-refinement on one subdomain

Although a knot insertion affects the whole parameter domain due to the tensor-product structure of the NURBS basis functions, we can limit the refinement to a single subdomain, as depicted in Figure 4.3(b). Such a local refinement procedure was already proposed in [28] in the context of multi-patch NURBS discretizations.

Algorithm 4.1 IETI-DP

Input: Local function spaces $V_h^{(k)}$, jump operator $\mathbf{B}$, scaling matrices $\mathbf{D}^{(k)}$, index sets of dual, essential primal, floating primal, boundary, and interior DOF.
Output: Coefficient vectors $\widetilde{\mathbf{u}}_P, \widetilde{\mathbf{u}}_R^{(1)}, \dots, \widetilde{\mathbf{u}}_R^{(N)}$.

for each $k = 1, \dots, N$ locally on each subdomain $\Omega^{(k)}$ (in parallel) **do**
  Assemble $\mathbf{K}^{(k)}$ and $\mathbf{f}^{(k)}$ using a fully local numbering of the DOF.
  Partition $\mathbf{K}^{(k)}$ and $\mathbf{f}^{(k)}$ as in (4.24), factorize $\mathbf{K}_{RR}^{(k)}$, compute $\overline{\mathbf{S}}_{PP}^{(k)}$.
  Partition $\mathbf{K}^{(k)}$ as in (4.42), factorize $\mathbf{K}_{II}^{(k)}$.
end for
Assemble and factorize $\overline{\mathbf{S}}_{PP}$, compute $\mathbf{d}$.
Solve $\mathbf{F}\boldsymbol{\lambda} = \mathbf{d}$ by PCG with preconditioner $\mathbf{M}^{-1}$ as in (4.43).
Compute $\widetilde{\mathbf{u}}_P$ as in (4.37),

$$\widetilde{\mathbf{u}}_P = \overline{\mathbf{S}}_{PP}^{-1}\Big(\overline{\mathbf{f}}_P - \overline{\mathbf{B}}_P^T\boldsymbol{\lambda} - \overline{\mathbf{K}}_{PR}\overline{\mathbf{K}}_{RR}^{-1}(\overline{\mathbf{f}}_R - \overline{\mathbf{B}}_R^T\boldsymbol{\lambda})\Big).$$

for each $k = 1, \dots, N$ **do** obtain $\widetilde{\mathbf{u}}_R^{(k)}$ as in (4.40) (in parallel),

$$\widetilde{\mathbf{u}}_R^{(k)} = {\mathbf{K}_{RR}^{(k)}}^{-1}\Big(\mathbf{f}_R^{(k)} - (\overline{\mathbf{B}}_R^{(k)})^T\boldsymbol{\lambda} - \overline{\mathbf{K}}_{RP}^{(k)}\widetilde{\mathbf{u}}_P\Big).$$

end for
return $\widetilde{\mathbf{u}}_P, \widetilde{\mathbf{u}}_R^{(1)}, \dots, \widetilde{\mathbf{u}}_R^{(N)}$.

We assume that on an interface, which is not fully matching, the knot vector on one subdomain (the *fine side*) is a refinement of the knot vector on the other subdomain (the *coarse side*), as in the example in Figure 4.3(b). In Figure 4.4, such a case is illustrated schematically: The knot vector $s^{(k)}$ is obtained from $s^{(\ell)}$ by one step of uniform h-refinement. In contrast to the fully matching case, the numbers of DOF of $V_h^{(k)}$ and $V_h^{(\ell)}$ on the interface $\Gamma^{(k,\ell)}$ are not equal, and condition (ii) in Definition 4.1 for the fully matching case is not fulfilled. In reference to hanging nodes in finite element methods, we call this a *setting with hanging knots* (note that we still assume that the geometry is conforming). As a consequence, formulating a coupling as in (4.14) is not straightforward. In particular, the matrix $\mathbf{Z}^{(i,j)}$ in (4.13) has to be modified accordingly.

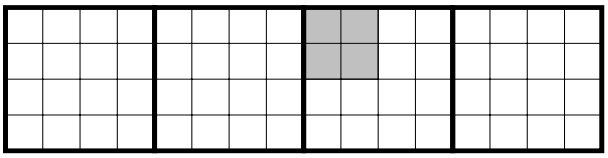
The number of interface conditions on such an interface is determined by the fine side, which is chosen as the master subdomain. Hence, we adapt the definition (4.11) of $\mathcal{C}$ as follows.

If $\Gamma^{(k,\ell)}$ is an interface with hanging knots, we define

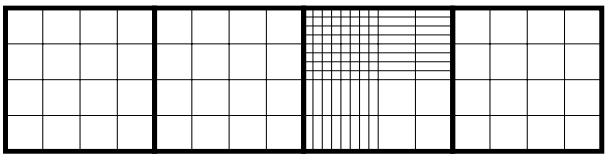
$$\begin{aligned} &(k,\ell) \in \mathcal{C}, \quad \text{if } \#\mathcal{B}(k,\ell) \geq \#\mathcal{B}(\ell,k), \\ &(\ell,k) \in \mathcal{C}, \quad \text{otherwise.} \end{aligned}$$

If $\Gamma^{(k,\ell)}$ is a fully matching interface, we follow the definition of $\mathcal{C}$ in (4.11), i.e.,

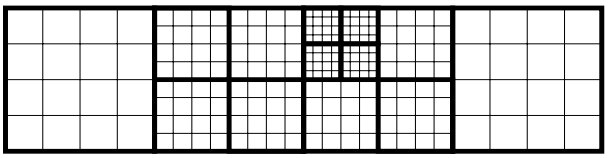
$$\begin{aligned} &(k,\ell) \in \mathcal{C}, \quad \text{if } k < \ell, \\ &(\ell,k) \in \mathcal{C}, \quad \text{otherwise.} \end{aligned}$$



(a) Shaded area marked for refinement.



(b) h-refinement of (a) on one subdomain (see Section 4.3.1).



(c) Refinement of (a) by two steps of 1-level substructuring (see Section 4.3.2).

Figure 4.3: Two options for refining the shaded area in (a).

For example, in Figure 4.4, the master subdomain is $\Omega^{(k)}$, i.e., $(k,\ell) \in \mathcal{C}$.

Without loss of generality, we assume that $s^{(k)}$ is a refinement of $s^{(\ell)}$ and that the weights on the finer side are obtained by the knot insertion algorithm [76]. Hence, on the interface $\Gamma^{(k,\ell)}$, the coarse basis function $\check{R}_{\boldsymbol{j}}^{(\ell)}|_{\Gamma^{(k,\ell)}}$, $\boldsymbol{j} \in \mathcal{B}(\ell,k)$ can be represented *exactly* as a linear combination of fine basis functions $\check{R}_{\boldsymbol{i}}^{(k)}|_{\Gamma^{(k,\ell)}}$, $\boldsymbol{i} \in \mathcal{B}(k,\ell)$. Therefore, for each $\boldsymbol{j} \in \mathcal{B}(\ell,k)$, there exist coefficients $\zeta_{\boldsymbol{j},\boldsymbol{i}}$, such that

$$\check{R}_{\boldsymbol{j}}^{(\ell)}|_{\Gamma^{(k,\ell)}} = \sum_{\boldsymbol{i} \in \mathcal{B}(k,\ell)} \zeta_{\boldsymbol{j},\boldsymbol{i}}\ \check{R}_{\boldsymbol{i}}^{(k)}|_{\Gamma^{(k,\ell)}}. \tag{4.44}$$

The coefficients $\zeta_{\boldsymbol{j},\boldsymbol{i}}$ can be obtained from well-known formulae for the refinement of B-Spline basis functions [76]. We require C^0-continuity of u_h across the interface $\Gamma^{(k,\ell)}$, i.e., we require, using (4.44),

$$\begin{aligned}\sum_{\boldsymbol{i} \in \mathcal{B}(k,\ell)} u_{\boldsymbol{i}}^{(k)} \check{R}_{\boldsymbol{i}}^{(k)}|_{\Gamma^{(k,\ell)}} &= \sum_{\boldsymbol{j} \in \mathcal{B}(\ell,k)} u_{\boldsymbol{j}}^{(\ell)} \check{R}_{\boldsymbol{j}}^{(\ell)}|_{\Gamma^{(k,\ell)}} \\ &= \sum_{\substack{\boldsymbol{j} \in \mathcal{B}(\ell,k) \\ \boldsymbol{i} \in \mathcal{B}(k,\ell)}} u_{\boldsymbol{j}}^{(\ell)} \zeta_{\boldsymbol{j},\boldsymbol{i}}\ \check{R}_{\boldsymbol{i}}^{(k)}|_{\Gamma^{(k,\ell)}}.\end{aligned}$$

By comparing the coefficients, we obtain

$$u_{\boldsymbol{i}}^{(k)} - \sum_{\boldsymbol{j}\in\mathcal{B}(\ell,k)} \zeta_{\boldsymbol{j},\boldsymbol{i}}\; u_{\boldsymbol{j}}^{(\ell)} = 0,$$

i.e., we obtain a continuity constraint in the same form as in (4.14) and (4.16), where the coefficients of the coupling matrix $\mathbf{Z}^{(k,\ell)}$, see (4.13), are given by

$$z_{\boldsymbol{i},\boldsymbol{j}}^{(k,\ell)} = \zeta_{\boldsymbol{j},\boldsymbol{i}}.$$

Remark 4.6. *In [50], the issue of coupling mortar discretizations in the FETI-DP context was addressed. The procedure for formulating the jump operator given therein would result in the same coefficients* $z_{\boldsymbol{i},\boldsymbol{j}}^{(k,\ell)}$ *if it is applied to the setting considered here.*

With the modified coupling matrices, one can then perform the same steps as in Sections 4.2.3.2 to 4.2.5. Here we would like to point out that there are Lagrange multipliers which connect primal essential DOF with a boundary condition as well as multipliers connecting one and the same primal DOF (this is illustrated in Figure 4.2(a), where the redundant coupling conditions between the subdomain vertices actually couple the same primal DOF). Both types of multipliers are superfluous and can be left out. However, in the presence of an interface with hanging knots, there are also multipliers that connect primal and remaining DOF. These multipliers cannot be left out and the corresponding entries in the vector $\overline{\mathbf{b}}$ do not vanish in general.

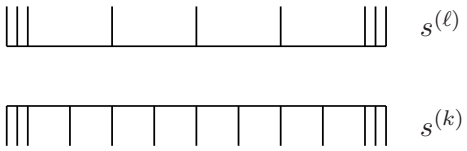


Figure 4.4: Interface with hanging knots: $p^{(k)} = p^{(\ell)} = 2$, $s^{(k)} = \{0,0,0,1/8,2/8,\ldots,7/8,1,1,1\}$ is a refinement of $s^{(\ell)} = \{0,0,0,1/4,1/2,3/4,1,1,1\}$.

4.3.2 Local refinement by substructuring

As an alternative, the number of DOF can be increased locally by subdividing one subdomain into smaller subdomains as illustrated in Figure 4.3(c).

If one wanted to subdivide the NURBS geometry mapping, e.g., in order to be able to edit the geometry locally, one would split the mapping and construct new knot vectors, weights and control nets for the new geometry mappings. This is not necessary in the IETI context, since we are not interested in actually splitting the geometry representation. Instead, we split the parameter domain into smaller subdomains which are mapped to the physical domain by the original (unchanged) coarse geometry mapping.

A very simple method for substructuring the subdomain $\Omega^{(k)}$ is the following: We split the parameter domain $\widehat{\Omega} = (0,1)^2$ into four subdomains

$$\begin{aligned}\widehat{\Omega}^{(k,1)} &= (0,1/2) \times (0,1/2),\\ \widehat{\Omega}^{(k,2)} &= (0,1/2) \times (1/2,1),\\ \widehat{\Omega}^{(k,3)} &= (1/2,1) \times (0,1/2),\\ \widehat{\Omega}^{(k,4)} &= (1/2,1) \times (1/2,1).\end{aligned}$$

We refer to this substructuring method as *cross insertion*. The basis functions of the original parameter domain $\widehat{\Omega}$ are pushed forward to the smaller subdomain $\widehat{\Omega}^{(k,\bar{k})}$ by a linear mapping $G^{(k,\bar{k})} : \widehat{\Omega} \to \widehat{\Omega}^{(k,\bar{k})}$ and then transformed to the physical domain by the original mapping $G^{(k)}$ (see Figure 4.5 for an illustration). The basis functions on $\Omega^{(k,\bar{k})}$ have the form

$$R_{\boldsymbol{i}}^{(k)} \circ \left({G^{(k,\bar{k})}}^{-1} \circ {G^{(k)}}^{-1} \right).$$

The domain decomposition obtained by substructuring is again a setting with hanging knots. The matrix $\mathbf{Z}^{(k,\ell)}$ in the interface condition (4.14) is adapted as described in Section 4.3.1.

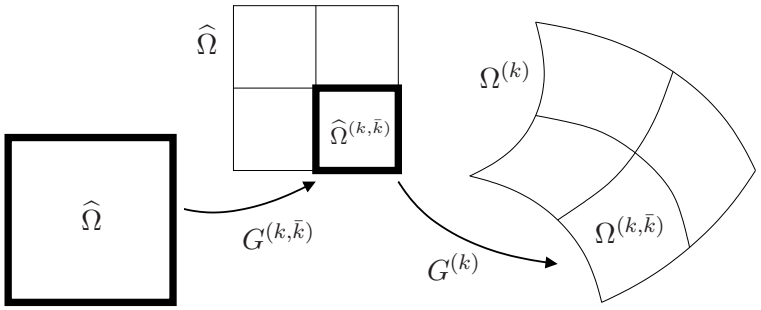
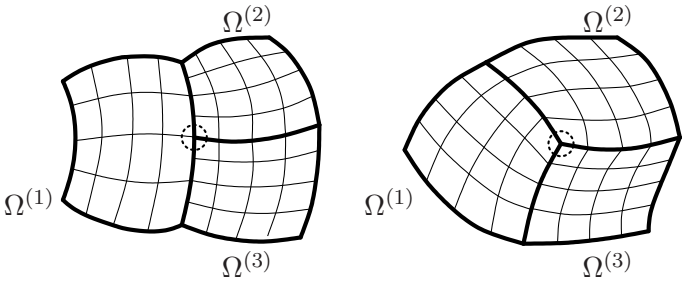


Figure 4.5: Embedding smaller subdomains $\widehat{\Omega}^{(k,\bar{k})}$ into the original parameter domain $\widehat{\Omega}$.

When we refine by substructuring, we introduce situations where the vertex of one subdomain lies on the edge of another subdomain. Such cases are illustrated in Figure 4.3(c) and Figure 4.6(a). We call such a subdomain vertex a *hanging subdomain vertex* (or short *hanging vertex*). Note that not every T-shaped subdomain vertex is a hanging vertex, as illustrated in the example in Figure 4.6(b).

The choice of primal DOF in substructured subdomains, where we have hanging vertices, is not as straightforward as in the fully matching case. In the example of a hanging vertex in Figure 4.6(a), there is (in the scalar case) exactly one DOF on $\Omega^{(2)}$ that is associated with the hanging vertex marked by the dashed circle (cf. Remark 3.10 and the discussion at the beginning of Section 4.2.3). While the same applies to $\Omega^{(3)}$, this is not true on $\Omega^{(1)}$, where we have several NURBS basis functions which are nonzero at the marked hanging vertex. Instead of incorporating a special treatment of hanging vertices, we choose to omit primal DOF at hanging vertices and discuss under which conditions this is possible.

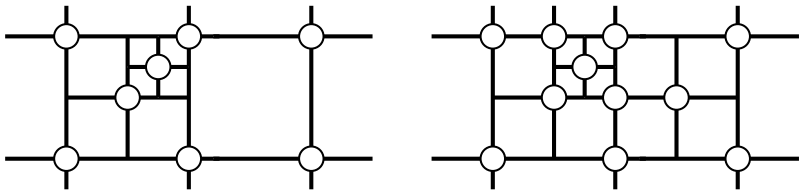


(a) Hanging subdomain vertex marked by dashed circle.

(b) T-shaped, but not hanging subdomain vertex.

Figure 4.6: Examples for hanging and not hanging subdomain vertices.

For the scalar elliptic problem (I), the kernel of the stiffness matrix of a floating subdomain is spanned by the constant function, i.e., the kernel has dimension one. In this case, it is sufficient to have at least one primal DOF on each subdomain. This is easily guaranteed, if we start from a fully matching setting, apply substructuring by cross-insertion as described in Section 4.3.2, and select primal DOF at all subdomain vertices which are not hanging. The example in Figure 4.7(a) shows the positions of primal DOF after two cross insertions.



(a) Cross-insertion results in at least one primal DOF on each subdomain.

(b) 1-level refinement (used for linearized elasticity) results in at least two primal vertices on each subdomain.

Figure 4.7: Subdomains refined by substructuring. Positions of primal vertices marked by ◯.

For the two-dimensional linearized elasticity problem (II), where the kernel is spanned by the three rigid body modes, we need at least three primal DOF per subdomain. We call a subdomain vertex a *primal vertex*, if the two DOF associated with this vertex are primal. Choosing at least two primal vertices per subdomain results in four primal DOF per subdomain, which is one more than necessary for fixing the kernel. However, as illustrated in Figure 4.7(a), this is not guaranteed if we apply substructuring by cross-insertion without additional considerations.

For linearized elasticity problems, we introduce *refinement levels* and we assign refinement level 0 to every subdomain in the initial setting. When a subdomain is split into four smaller subdomains by cross insertion, the levels of the new, smaller

subdomains are increased by 1 (see Figure 4.8 for an illustration). We call the refinement a *1-level substructuring*, if the refinement levels of any two subdomains with an edge as their interface differ by at most 1. If we start from a fully matching setting, apply 1-level substructuring by cross-insertion, and choose all non-hanging vertices as primal vertices, then it is guaranteed that there are at least two primal vertices, i.e., at least four primal DOF, on each subdomain. The example in Figure 4.7(b) illustrates the positions of primal vertices after two such 1-level substructuring steps. Note that, depending on the location of the refined area, 1-level substructuring can effect neighbouring subdomains. This disadvantage is accepted as a trade-off for avoiding an involved treatment of hanging vertices.

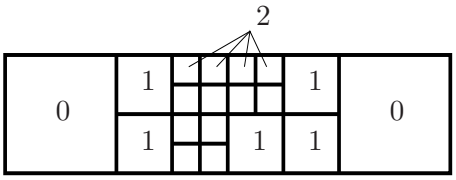


Figure 4.8: Refinement levels of subdomains (initial stetting as in Figure 4.3(a)).

Note that the discretization is only C^0-continuous along subdomain interfaces. By substructuring a subdomain, new interfaces are introduced, thereby changing the discretization.

Remark 4.7. *Note that the refinement options discussed in Section 4.3.1 and Section 4.3.2 can be combined by applying them one after the other. Furthermore, the coupling methods described in this section can be combined with isogeometric local refinement methods such as those mentioned in Section 3.3.4. If such local refinement methods are applied only in the interior of one or many subdomains, obviously, the coupling at the subdomain interfaces is not influenced. If the refined areas extend to subdomain boundaries, and one side is a refinement of the other, C^0 coupling can be done as described in this section.*

4.3.3 Preconditioning in the presence of hanging knots

As mentioned in Section 4.3.1, when we have hanging knots, the coupling matrix $\mathbf{B}^{(k)}$ is not a signed Boolean matrix any more. The Dirichlet preconditioner defined in (4.43) can still be applied with these more complicated coupling matrices. However, as already mentioned in [50] in the context of mortar discretizations, while the asymptotic behaviour of the condition number remains the same, the condition number itself increases. This can also be observed in the numerical tests with the IETI-DP method (see Section 4.4 for the results).

We now adapt the preconditioner for settings with hanging knots by replacing the scaling matrix $\mathbf{D}^{(k)}$ in (4.43) by a modified diagonal matrix $\mathbf{D}_Z^{(k)}$. Its entries are defined as follows:

$\left(\mathbf{D}_Z^{(k)}\right)_{ii} = 1/\operatorname{mult}(i)$, if λ_i corresponds to a fully matching interface. Here, $\operatorname{mult}(i)$ is as defined in Section 4.2.4.

$\left(\mathbf{D}_Z^{(k)}\right)_{ii} = 1$, if λ_i corresponds to an interface with hanging knots and $\Omega^{(k)}$ is the master subdomain.

$\left(\mathbf{D}_Z^{(k)}\right)_{ii} = 0$, otherwise.

The preconditioner $\mathbf{M}_Z^{-1}$ for the case with hanging knots is defined analogous to (4.43) by

$$\mathbf{M}_Z^{-1} = \sum_{k=1}^{N} \mathbf{D}_Z^{(k)} \mathbf{B}^{(k)} \begin{pmatrix} 0 & 0 \\ 0 & \mathbf{S}_{BB}^{(k)} \end{pmatrix} \mathbf{B}^{(k)^T} \mathbf{D}_Z^{(k)}.$$

As it will be reported in Section 4.4, this preconditioner leads to lower condition numbers as compared to $\mathbf{M}^{-1}$ from Section 4.2.4 in settings with hanging knots. Note that we have $\mathbf{M} = \mathbf{M}_Z$ in fully matching settings.

4.4 Numerical examples

In this section, we present three numerical test examples for the IETI-DP method. We test the method and the refinement options presented in Section 4.3, and we study the performance of the proposed preconditioners. The conjugate gradient method is applied to solve the interface problem $\mathbf{F}\boldsymbol{\lambda} = \mathbf{d}$ in (4.33), both without and with the discussed preconditioners. In the following tables, we display the numbers of iterations needed until the stopping criterion

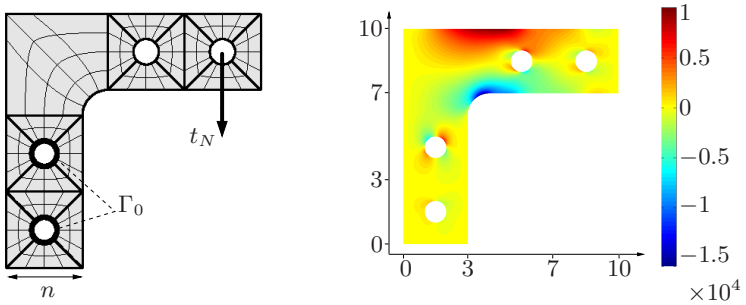
$$\|\mathbf{r}_i\|_{\ell_2} / \|\mathbf{r}_0\|_{\ell_2} < 10^{-8}$$

is fulfilled, where $\mathbf{r}_0$ is the initial residual, $\mathbf{r}_i$ is the residual in the i-th iteration, and $\|\cdot\|_{\ell^2}$ denotes the Euclidean norm. Choosing the zero-vector as initial guess for $\boldsymbol{\lambda}$, we have $\mathbf{r}_0 = \mathbf{d}$. The condition numbers in these tables are computed numerically using the Lanczos method.

Note that the aim of these examples (due to their sizes, they could be solved by direct solvers) is to illustrate the method and its potential. The true computational advantage of the IETI method against direct solvers will become apparent in large problem sizes and/or three-dimensional problems.

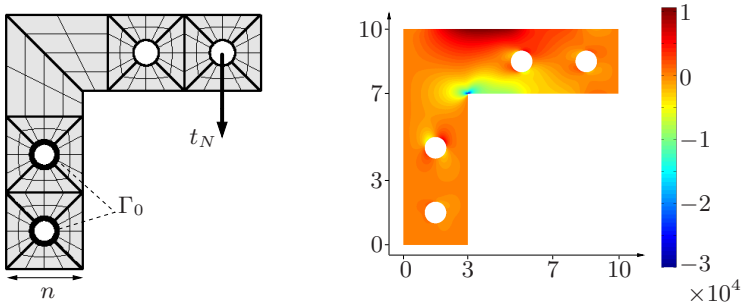
Example 4.1: Bracket under load

Our first example is a linearized elasticity problem (type (II) in Section 2.2.1). We consider the two geometries displayed in Figure 4.9(a) and Figure 4.10(a), where the first one, which is taken from an illustration in [29], has a rounded reentrant corner, and the second one has a sharp reentrant corner. We refer to these geometries as case (A) and case (B), respectively.



(a) Setting in case (A). (b) Stress component σ_{11}.

Figure 4.9: Case (A), bracket with rounded reentrant corner.



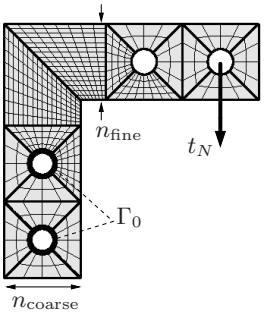
(a) Setting in case (B). (b) Stress component σ_{11}.

Figure 4.10: Case (B), bracket with sharp reentrant corner.

Case	n	$\#\boldsymbol{\lambda}$	condition numbers $\mathbf{F}$	condition numbers $\mathbf{M}^{-1}\mathbf{F}$	(P)CG iterations $\mathbf{F}$	(P)CG iterations $\mathbf{M}^{-1}\mathbf{F}$
(A)	8	464	173.95	39.60	69	42
	16	784	470.09	67.98	103	49
	32	1424	1230.11	105.87	149	58
	64	2704	3074.62	154.10	> 200	67
(B)	8	484	174.17	51.72	70	43
	16	820	509.39	85.71	106	50
	32	1492	1388.58	130.24	150	60
	64	2836	3543.44	185.42	> 200	67

Figure 4.11: Condition numbers and (P)CG iterations for cases (A) and (B) of a bracket under load, n denoting the number of knot spans in the direction indicated by arrows in the above pictures.

In Figure 4.9(a) and 4.10(a), we show the subdomain decomposition and indicate the boundary conditions. We fix the two lower holes by applying homogeneous essential boundary conditions, while a constant downward pointing traction t_N with magnitude 1000N is applied at the walls of the rightmost hole. The remaining boundaries are free of traction. The material parameters are set to $E = 3 \cdot 10^7$kPa and $\nu = 0.3$. Note that the circular holes contained in the domains are represented exactly



(a) Setting in case (C).

			condition numbers			(P)CG iterations		
n_{coarse}	n_{fine}	$\#\boldsymbol{\lambda}$	$\mathbf{F}$	$\mathbf{M}^{-1}\mathbf{F}$	$\mathbf{M}_Z^{-1}\mathbf{F}$	$\mathbf{F}$	$\mathbf{M}^{-1}\mathbf{F}$	$\mathbf{M}_Z^{-1}\mathbf{F}$
4	16	428	469.89	344.72	86.40	71	69	45
8	32	708	1324.01	542.19	130.75	105	89	57
16	64	1268	3419.50	820.16	185.40	148	103	66
32	128	2388	8448.24	1170.29	258.89	> 200	120	75

(b) Condition numbers and (P)CG iterations for case (C).

Figure 4.12: Case (C), bracket with sharp reentrant corner and local h-refinement near the corner. The ratio $n_{\text{fine}}/n_{\text{coarse}} = 4$ is the same for all chosen meshes. The stress component σ_{11} as as in Figure 4.10(b).

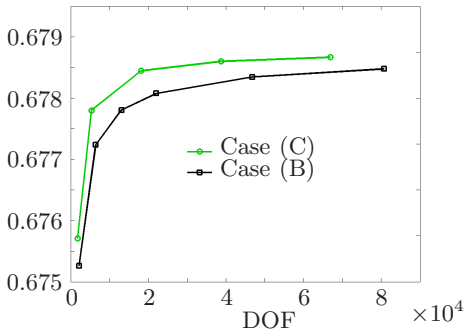


Figure 4.13: Comparison of the energy norms of the discrete solutions $\|u_h\|_E$ in cases (B) and (C).

by NURBS geometry mappings of degree 2.

In Figure 4.9(b) and 4.10(b), the computed stress component σ_{11} is depicted for cases (A) and (B), respectively. Note that the scales are different, and that the scale in Figure 4.10(b) has been cutoff below for a better visibility of the stress distribution. The results illustrate that the IETI-DP method can be applied to non-trivial geometries including holes and consisting of numerous subdomains.

The condition numbers and the (P)CG iteration numbers for these fully matching settings are given in the table in Figure 4.11. The column labeled n shows the number of knot spans in the direction indicated by the small arrows at the bottom in Figure 4.9(a) and 4.10(a). The column $\#\boldsymbol{\lambda}$ shows the number of Lagrange mul-

tipliers, i.e., the size of the interface problem (4.33). The columns labeled $\mathbf{F}$ and $\mathbf{M}^{-1}\mathbf{F}$ display the condition numbers of the interface problems and the (P)CG iteration numbers without preconditioner, and with preconditioner $\mathbf{M}^{-1}$ as defined in Section 4.2.4, respectively. The entry "> 200" indicates that the desired accuracy was not reached after 200 iterations. The results show the expected, moderate growth in the preconditioned case.

In Figure 4.10(b), the peak of the stress component σ_{11} near the reentrant corner in case (B) is visible. As mentioned above, the scale has been cutoff below for better visibility, and the stress component actually exceeds the value of $-3 \cdot 10^4$ (the lower limit of the colorscale). To obtain a better resolution of the peak stress, we introduce case (C), which is indicated in Figure 4.12(a). Here, the subdomains near the corner have a finer discretization than the subdomains which are far from the corner, and we have interfaces with hanging knots. The number of knot spans on the finest and coarsest subdomain discretizations are denoted by n_{fine} and n_{coarse}, respectively. These numbers are measured in the directions indicated by the small arrows in Figure 4.12(a). The ratio $n_{\text{fine}}/n_{\text{coarse}} = 4$ is the same for all chosen meshes. The condition numbers and (P)CG iteration numbers for this setting with hanging knots are presented in Figure 4.12(b). Clearly, the preconditioner $\mathbf{M}_Z^{-1}$ defined in Section 4.3.3 performs better than $\mathbf{M}^{-1}$ from Section 4.2.4. The stress component σ_{11} is not plotted for case (C), since it is the same as in case (B), Figure 4.10(b). The energy norm of the numerical solutions in case (B) and case (C) is compared in Figure 4.13. It shows that for given DOF a faster convergence can be achieved by local h-refinement.

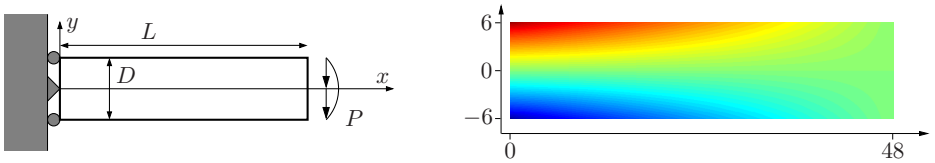
Example 4.2: Bending of a Cantilever

We now consider a linearized elasticity problem (type (II) in Section 2.2.1) on a cantilever of length L and thickness D. It is fixed at $x = 0$ and subject to a parabolic traction at $x = L$ with resultant P as illustrated in Figure 4.14(a). We choose the parameters as follows: $L = 48\text{m}$, $D = 12\text{m}$, $E = 3 \cdot 10^7\text{kPa}$, $\nu = 0.3$, and $P = 1000\text{N}$.

An analytical solution for the displacement field $u = (u_1, u_2)^T$ can be found, e.g., in [56, 70, 94]:

$$
\begin{aligned}
u_1 =& \frac{Py}{6E_0I}\left((6L - 3x)x + (2 + \nu_0)\Big(y^2 - \frac{D^2}{4}\Big)\right),\\
u_2 =& -\frac{P}{6E_0I}\left(3\nu_0 y^2(L - x) + (4 + 5\nu_0)x\frac{D^2}{4} + (3L - x)x^2\right),
\end{aligned}
$$

where $I = D^3/12$ is the moment of inertia of the cross section of the cantilever. When we consider the plane stress problem, we set $E_0 = E$ and $\nu_0 = \nu$. For the plane strain problem, we set $E_0 = E/(1 - \nu^2)$ and $\nu_0 = \nu/(1 - \nu)$. Then, in both



(a) Problem setting. (b) Stress component σ_{11}.

Figure 4.14: Bending of a cantilever, problem setting.

cases, the resulting exact stress components are as follows:

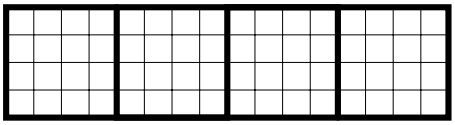
$$\begin{aligned} \sigma_{11} &= \frac{P(L-x)y}{I}, \\ \sigma_{12} &= -\frac{P}{2I}\left(\frac{D^2}{4} - y^2\right), \\ \sigma_{22} &= 0. \end{aligned}$$

The stress component σ_{11} is plotted in Figure 4.14(b).

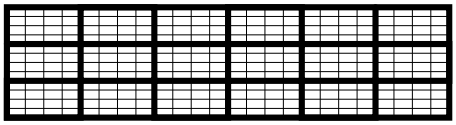
We apply the exact displacement as essential boundary conditions at the boundary $x = 0$, and the exact traction t_N at $x = L$. The remaining boundaries at $y = \pm D/2$ are free of traction.

Note that the exact displacement field $u = (u_1, u_2)^T$ is a cubic polynomial. By using basis functions of degree 3 and due to the simple geometry mappings, the exact solution u is in V_h. Although the exact solution does not have peaks or singularities, we refine randomly chosen subdomains, in order to demonstrate the performance of the IETI-DP method in settings with hanging knots. In Figure 4.15(a), the thick lines indicate the subdomain decomposition, while the thin lines schematically indicate how fine the discretizations of the respective subdomains are, and whether the setting is fully matching or not. The numerical tests confirm that, for all considered discretizations, the computed numerical solution is exact (up to the accuracy to which we solve (4.33) for $\boldsymbol{\lambda}$).

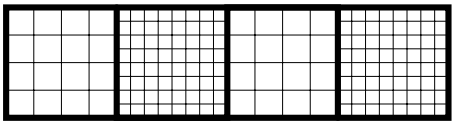
The table presented in Figure 4.15(b) shows that the condition numbers behave as expected. For cases which are not fully matching, the columns labeled n_{coarse} and n_{fine} indicate the number of knot spans in one direction on the coarse and fine discretizations, respectively. The other columns are labeled as in the previous example. In the settings (C), (D), and (E) with hanging knots, it can be observed that the condition number of $\mathbf{M}^{-1}\mathbf{F}$ grows slower than in the unpreconditioned case. However, on coarse discretizations, both the absolute value of the condition number and the number of (P)CG iterations are larger in the preconditioned case than in the unpreconditioned case (cf. the discussion in the beginning of Section 4.3.3). The preconditioner $\mathbf{M}_Z^{-1}$ performs better in all cases with hanging knots.



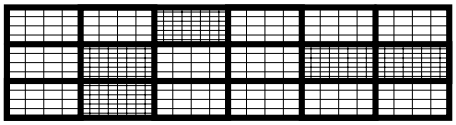
Setting (A), 4 subdomains, fully matching.



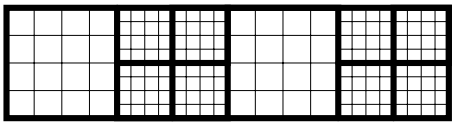
Setting (B), 18 subdomains, fully matching.



Setting (C), 4 subdomains, 2 of which are h-refined.



Setting (D), 18 subdomains, 5 of which are h-refined.



Setting (E), 10 subdomains, with hanging vertices.

(a) Illustration of the tested discretizations.

Setting	n_{coarse}	n, n_{fine}	$\#\boldsymbol{\lambda}$	condition numbers			(P)CG iterations		
				$\mathbf{F}$	$\mathbf{M}^{-1}\mathbf{F}$	$\mathbf{M}_Z^{-1}\mathbf{F}$	$\mathbf{F}$	$\mathbf{M}^{-1}\mathbf{F}$	$\mathbf{M}_Z^{-1}\mathbf{F}$
(A)	*	8	88	33.80	11.81	†	24	15	†
	*	16	152	77.02	14.99	†	33	16	†
	*	32	280	170.81	18.38	†	40	18	†
	*	64	536	378.38	23.49	†	50	20	†
(B)	*	8	700	53.87	13.50	†	48	24	†
	*	16	1100	116.68	16.79	†	67	27	†
	*	32	2140	264.72	20.38	†	85	30	†
	*	64	4060	595.88	24.21	†	120	34	†
(C)	8	16	136	66.03	109.10	28.43	23	31	21
	16	32	248	157.41	146.07	35.53	37	38	22
	32	64	472	358.85	184.14	43.02	48	41	24
	64	128	920	795.88	224.55	51.11	62	47	28
(D)	4	8	580	46.38	74.78	31.02	50	53	35
	8	16	940	102.90	109.54	38.59	68	63	39
	16	32	1660	231.74	142.50	47.01	93	71	43
	32	64	3100	530.70	175.41	55.94	123	77	46
(E)	*	8	338	184.11	351.08	98.71	53	58	37
	*	16	578	387.44	433.00	141.50	69	63	41
	*	32	1058	813.77	532.37	194.07	90	69	46
	*	64	2018	1733.71	647.72	257.28	119	73	52

(b) Condition numbers and (P)CG iterations of the unpreconditioned and the preconditioned interface problem.

*In cases (A), (B), and (E), the number of knot spans is the same on all subdomains.

†In fully matching settings, we have $\mathbf{M}_Z^{-1} = \mathbf{M}^{-1}$.

Figure 4.15: Bending of a cantilever, discussed cases.

Example 4.3: Poisson Problem on Yeti's Footprint

In our third example, we solve the Poisson problem $-\Delta u = f$ (type (I) in Section 2.2.1 with $A = I$) on the physical domain Ω resembling the footprint of a Yeti. The domain is shown in Figure 4.16(a) and consists of 21 subdomains. We set Dirichlet boundary conditions at the big toe, and Neumann boundary conditions everywhere else. The boundary conditions and the right hand side f are determined by the exact solution

$$u(x,y) = \begin{cases} (R - r(x,y))^4 + y/10, & \text{if } r(x,y) < R, \\ y/10, & \text{else,} \end{cases}$$

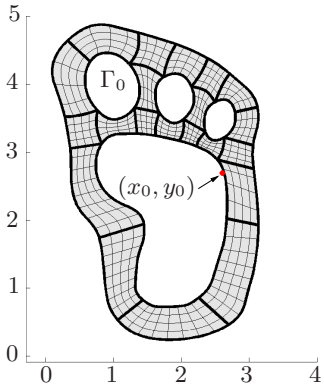
where $r(x,y) = |(x,y) - (x_0,y_0)|$. The solution u is constructed in such a way that it has a peak at (x_0,y_0). We set $R = 1$ and $(x_0,y_0) = (2.6, 2.7)$ (see Figure 4.16(a)).

For simplicity, since we know the exact solution u, we apply adaptive refinement based on the exact error. For each subdomain $\Omega^{(k)}$, we compute the local error in the energy norm $\eta^{(k)} = \|u - u_h\|_{E,\Omega^{(k)}}$. We use a marking criterion similar to (2.13) in Section 2.2.5.2, where subdomains are marked instead of cells. We mark all subdomains for refinement, for which

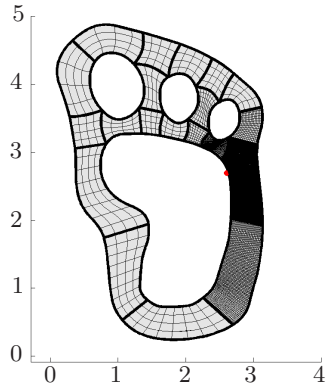
$$\eta^{(k)} \geq 0.1 \max\{\eta^{(\ell)},\ \ell = 1, \dots, N\}$$

holds. In each such refinement step, we apply uniform h-refinement on the marked subdomains. The global mesh after 5 such refinement steps is shown in Figure 4.16(b). In Figure 4.16(c), we compare the error obtained by global uniform refinement and by the described adaptive refinement. As expected, for a given number of DOF a more accurate solution can be achieved by adaptive, local h-refinement, as compared to global refinement in a fully matching setting.

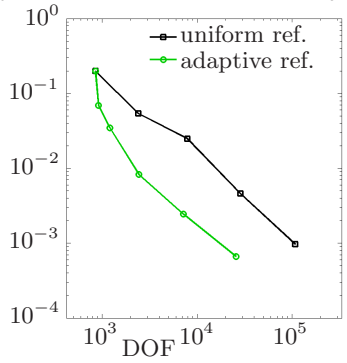
In Figure 4.16(d), the condition numbers and the (P)CG iteration numbers for the fully matching setting are presented. The column labeled "DOF" indicates the global number of DOF. In Figure 4.16(e), the condition numbers and the (P)CG iterations in the adaptive refinement are shown. As in the previous examples, these numbers illustrate the good performance of the preconditioner $\mathbf{M}_Z^{-1}$ in the setting with interfaces, which are not fully matching. Note that applying the preconditioner $\mathbf{M}^{-1}$ results in iteration numbers which are similar to those for the unpreconditioned case (for the considered meshes). In contrast to this, the iteration numbers obtained with $\mathbf{M}_Z^{-1}$ are, for a given size of $\boldsymbol{\lambda}$, comparable to the fully matching case.



(a) Problem setting on Yeti's footprint.



(b) Discretization after 5 refinement steps.



(c) Exact errors in the energy norm $\|u - u_h\|_E$ vs. DOF.

n	DOF	$\#\boldsymbol{\lambda}$	condition numbers $\mathbf{F}$	condition numbers $\mathbf{M}^{-1}\mathbf{F}$	(P)CG iterations $\mathbf{F}$	(P)CG iterations $\mathbf{M}^{-1}\mathbf{F}$
4	852	180	16.26	5.52	29	17
8	2420	292	35.25	7.55	37	20
16	7956	516	81.37	9.90	46	23
32	28628	964	188.51	12.49	59	26
64	108372	1860	431.61	15.36	85	28

(d) Condition numbers and (P)CG iterations, fully matching setting with global refinement.

n	DOF	$\#\boldsymbol{\lambda}$	condition numbers $\mathbf{F}$	condition numbers $\mathbf{M}^{-1}\mathbf{F}$	condition numbers $\mathbf{M}_Z^{-1}\mathbf{F}$	(P)CG iterations $\mathbf{F}$	(P)CG iterations $\mathbf{M}^{-1}\mathbf{F}$	(P)CG iterations $\mathbf{M}_Z^{-1}\mathbf{F}$
0	852	180	16.26	5.52	5.52	29	17	17
1	916	188	22.92	39.14	13.99	29	30	19
2	1204	212	51.94	68.62	19.63	31	33	20
3	2444	268	119.36	111.01	25.94	36	38	22
4	7196	384	272.22	174.72	33.13	44	44	23
5	25756	628	612.87	317.69	41.14	55	53	25

(e) Condition numbers and (P)CG iterations, adaptive refinement.

Figure 4.16: Yeti's footprint with adaptive refinement.

Chapter 5

Functional-type a posteriori error estimates

The theoretical foundation for the functional-type a posteriori error estimator presented in this chapter is well-known and well-studied, see, e.g., [79, 80, 81, 82], and in particular the monograph [84] and the references therein.

The aim of this chapter is to discuss an efficient application of such error estimators in IGA, taking advantage of some properties which are specific to NURBS. The results presented in this chapter can be found in [58]. We mainly focus on guaranteed *upper* bounds of the true error. Guaranteed and sharp functional-type *lower* error bounds are briefly discussed and results are presented in Section 5.6.

Assumption 5.1. *In this chapter, we discuss model problems of type (I) as defined in Section 2.2.1 with Dirichlet boundary conditions (i.e., $\Gamma_0 = \partial\Omega$), and we only consider* single-patch *geometry mappings. We assume that the matrix A is a symmetric positive definite matrix and has a positive inverse A^{-1}, and that there exist constants $c_1, c_2 > 0$ such that*

$$c_1|\xi|^2 \leq A\xi \cdot \xi \leq c_2|\xi|^2, \quad \forall \xi \in \mathbb{R}^2. \tag{5.1}$$

Under these assumptions, the norms

$$\|v\|_A^2 = \int_\Omega Av \cdot v \, dx, \quad \|v\|_{\bar{A}}^2 = \int_\Omega A^{-1}v \cdot v \, dx, \tag{5.2}$$

are equivalent to the L^2-norm $\|v\|_0^2 = \int_\Omega v \cdot v \, dx$, see Section 2.1.2.

5.1 Guaranteed upper bound of the error

The starting point for the proposed method is the following main result which gives an upper bound for the error in the energy norm. It can be found, e.g., in [80, 82, 84].

Theorem 5.2. *Let C_Ω be the constant in the Friedrichs' type inequality $\|v\|_0 \leq C_\Omega \|\nabla v\|_A$, $\forall v \in V_0$. Let u be the exact solution of the model problem (I), and let $u_h \in V_h$ be an approximate solution. Then, the following estimate holds.*

$$\|\nabla u - \nabla u_h\|_A \leq \|A\nabla u_h - y\|_{A^{-1}} + C_\Omega \|\operatorname{div} y + f\|_0, \tag{5.3}$$

where y is an arbitrary vector-valued function in $H(\Omega, \operatorname{div})$.

The constant C_Ω depends only on the domain Ω and the coefficient matrix A (but not on the underlying mesh), see, e.g., [62, 84]. Note that C_Ω can be computed either numerically or, if one can find a domain $\Omega_\square \supset \Omega$, where $\Omega_\square$ is a square domain with side-length ℓ, then $C_\Omega \leq c_2 \frac{\ell}{\pi\sqrt{d}}$, where d is the dimension and c_2 is the constant in (5.1).

Note that, if we choose y via the (unknown) exact solution $y = A\nabla u$, both sides of (5.3) coincide. Hence, the estimate is sharp in the sense that, for any fixed u_h, we can find a function y such that the upper bound is as close to the exact error as desired. The estimate given in Theorem 5.2 is a guaranteed and fully computable upper bound for any conforming approximation $u_h \in V_g$.

In the following, we describe some approaches to construct y and discuss their relative merits.

5.1.1 Post-processing of u_h

It is possible to obtain *good* error indicators by constructing functions y by some post-processing of the discrete solution u_h, see [62, 84] and the references therein. Consider, for example, $A = I$ and that $u_h \in V_h$ has been computed with NURBS basis functions of degree $p \geq 2$ with at least C^1-continuity. Then, since $u_h \in C^{p-1}$, we have $\nabla u_h \in (C^{p-2})^2 \subset H(\Omega, \operatorname{div})$. Choosing $y = \nabla u_h$ will thus result in

$$\|\nabla u - \nabla u_h\|_0 \leq C_\Omega \|\Delta u_h + f\|_0. \tag{5.4}$$

To show the efficiency of the estimator (5.4), we present an illustrative numerical example. This example, referred to as *Example 5.1* in the remainder, is chosen due to a smoothly varying function (without any large gradients) in both directions.

Example 5.1 (Sinus Function on the Unit Square) In this numerical example, the computational domain is the unit square $\Omega = (0,1)^2$ and u_h is piecewise quadratic in both directions, i.e., $p = q = 2$. The coefficient matrix is constant, $A = I$, and the exact solution is given by

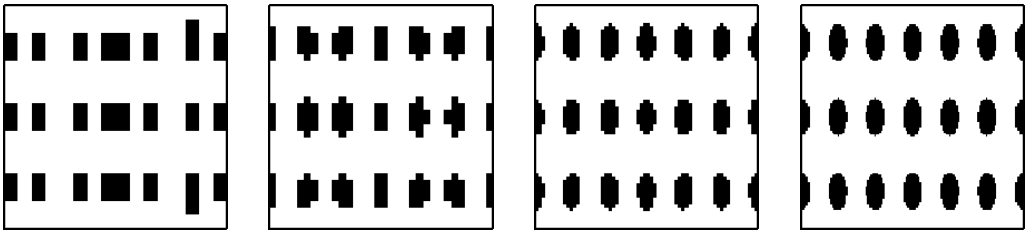
$$u = \sin(6\pi x)\sin(3\pi y).$$

The right-hand-side f and the (homogeneous) boundary conditions g_0 are determined by the prescribed exact solution u.

The local error indicator η_Q is given by the localized version of the bound in (5.4), i.e., by

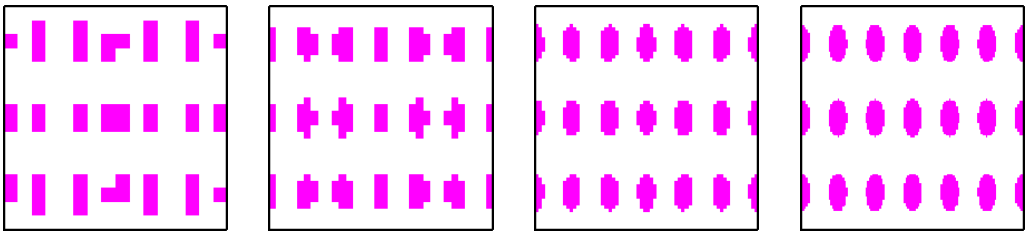
$$\eta_Q = \|\Delta u_h + f\|_{0,Q}, \tag{5.5}$$

and we choose the fixed-percentile criterion (2.12) described in Section 2.2.5.1 for cell-marking. In Figure 5.1, we present the cells marked for refinement by the exact error. The cells marked for refinement by the error indicator given in (5.5) are presented in Figure 5.2.



(a) 16×16 (b) 32×32 (c) 64×64 (d) 128×128

Figure 5.1: Cells marked by exact error with $\psi = 0.8$ in Example 5.1.



(a) 16×16 (b) 32×32 (c) 64×64 (d) 128×128

Figure 5.2: Cells marked by error estimator with $\psi = 0.8$ in Example 5.1, $y_h = \nabla u_h$.

We see that starting from the mesh 32×32, the majorant is able to nicely capture the refinement pattern of exact error. However, from a closer look at the convergence of the exact error and the majorant, see Figure 5.3, we find that though such an estimate is a guaranteed upper bound and very cheap to compute, it over-estimates the exact error, and its convergence is slower than the exact error (due to different operators acting on u_h on both sides).

5.1.2 Cell-wise interpolation

We now discuss a quasi interpolation (cell-wise) approach. Considering the 1D case first, we fix a certain DOF u_i, and set up a local problem that involves only those DOF/basis functions which have an influence on u_i. As discussed in Section 3.1.1.2, the support of a univariate NURBS basis function extends through $p+1$ knot spans (the interval (s_i, s_{i+p+1})). The total number of basis functions whose

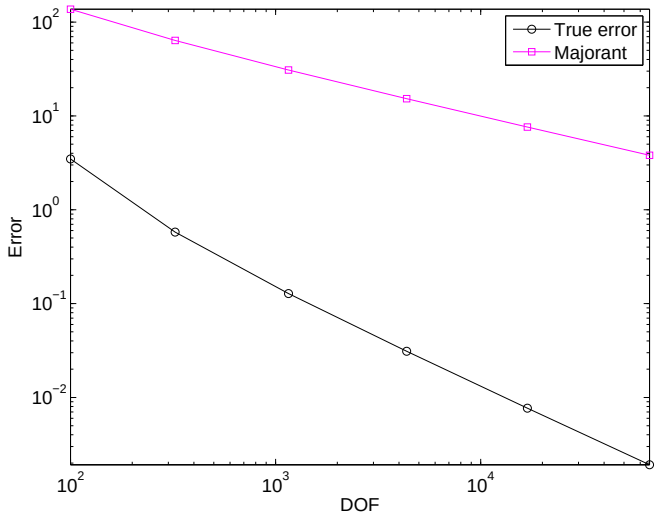


Figure 5.3: Convergence of exact error and the majorant (5.4) for Example 5.1.

support intersect the support of the function $R^s_{i,p}$ is $2p+1$ (the functions with indices $k \in \{i-p, \ldots, i+p\}$). Considering only these basis functions, we obtain a local system matrix which we denote by $\mathbf{L}^i_h$. This matrix is of size $(2p+1) \times (2p+1)$ and has a bandwidth of $2p+1$. For example, if $p = 3$, the matrix $\mathbf{L}^i_h$ is a 7×7-matrix, and its structure is given by

$$\begin{pmatrix} * & * & * & * & 0 & 0 & 0 \\ * & * & * & * & * & 0 & 0 \\ * & * & * & * & * & * & 0 \\ * & * & * & * & * & * & * \\ 0 & * & * & * & * & * & * \\ 0 & 0 & * & * & * & * & * \\ 0 & 0 & 0 & * & * & * & * \end{pmatrix}.$$

For each DOF u_i, such a system (with appropriate right hand side) needs to be solved. The same principle applies for bivariate basis functions. Assume that the polynomial degrees are same in both directions and are denoted by p. Due to the tensor product structure, the support of a bivariate NURBS basis function $R_{(i,j)}$ intersects the support of a total of $(2p+1)^2$ basis functions. Hence, the size of the local system matrix $\mathbf{L}^{(i,j)}_h$ is $(2p+1)^2 \times (2p+1)^2$. Furthermore, if the basis functions are vector-valued with d components, the local system matrix is of size $d(2p+1)^2 \times d(2p+1)^2$. For example, for vector-valued functions of degree 3 with two components, the local system size is 98×98. The solution of the corresponding system will give two coefficients which can be used for both the components associated with the double-index (i,j). Unfortunately, this approach is neither a cheap one (to compute y), nor does it result in desired sharp error bounds, and therefore, we do

not present its results.

5.1.3 Global minimization

In order to obtain a sharp estimate (and not just an error indicator), one has to find a function y which minimizes the right-hand-side of (5.3). For minimizing this estimate numerically, we first modify (5.3). Recall the algebraic Young's inequality

$$2ab \leq \beta a^2 + \tfrac{1}{\beta} b^2$$

which holds for any $a, b \in \mathbb{R}$ and $\beta > 0$. From this, it easily follows that

$$(a+b)^2 \leq (1+\beta)a^2 + (1+\tfrac{1}{\beta})b^2. \tag{5.6}$$

Applying (5.6) to (5.3), we obtain the following corollary from Theorem 5.2 (see also [84]).

Corollary 5.3. *For any $u_h \in V_{gh}$, the upper bound of the error is given by the estimate*

$$\|\nabla u_h - \nabla u\|_A^2 \leq M_\oplus^2(y, \beta), \tag{5.7}$$

where

$$M_\oplus^2(y,\beta) = (1+\beta)\|\nabla u_h - y\|_{\bar{A}}^2 + (1+\tfrac{1}{\beta})C_\Omega^2 \|\operatorname{div} y + f\|_0^2, \tag{5.8}$$

and where y is an arbitrary function in $H(\Omega, \operatorname{div})$, β is an arbitrary positive number, and C_Ω is as in Theorem 5.2.

Hereinafter, for simplicity, we will refer to $M_\oplus^2(y,\beta)$ as *majorant*. Introducing

$$\begin{aligned} b_1 &= 1+\beta, & b_2 &= (1+\tfrac{1}{\beta})C_\Omega^2, \\ U_1 &= \|A\nabla u_h - y\|_{\bar{A}}^2, & U_2 &= \|\operatorname{div} y + f\|_0^2, \end{aligned} \tag{5.9}$$

we can briefly write the majorant as

$$M_\oplus^2(y,\beta) = b_1 U_1 + b_2 U_2.$$

Note that the bound in (5.8) holds for *any* positive β and *any* function $y \in H(\Omega, \operatorname{div})$. The technique for finding such parameters y and β will be discussed in Sections 5.2 and 5.4. Before proceeding, we show Lemma 5.5 below, which ensures that the majorant is indeed a sharp bound. The Lemma and the proof, which we sketch for later reference, can be found in [84].

Definition 5.4. A sequence of finite-dimensional subspaces $\{Y_j\}_{j=1}^\infty$ of a Banach-space Y is called *limit dense in Y*, if for any $\varepsilon > 0$, there exists an index j_ε, such that $\inf_{p_k \in Y_k} \|p_k - v\|_Y < \varepsilon$ for all $k > j_\varepsilon$.

Lemma 5.5. *Let the spaces $\{Y_j\}_{j=1}^\infty$ be limit dense in $H(\Omega, \operatorname{div})$. Then*

$$\lim_{j\to\infty} \inf_{y_j \in Y, \beta > 0} M_\oplus^2(y_j, \beta) = \|\nabla u - \nabla u_h\|_A^2 \tag{5.10}$$

Proof. Recall that the $H(\Omega, \operatorname{div})$-norm $\|\cdot\|_{\operatorname{div}}$ is defined by $\|v\|_{\operatorname{div}}^2 = \|v\|_0^2 + \|\operatorname{div} v\|_0^2$. Let $\varepsilon > 0$ be arbitrarily small, but fixed. Let j_ε be the index such that, for all $k > j_\varepsilon$, there exists a $p_k \in Y_k$ with $\|A\nabla u - p_k\|_{\operatorname{div}} < \varepsilon$. Then,

$$\inf_{y_j \in Y_j, \beta > 0} M_\oplus^2(y_j, \beta) \leq M_\oplus^2(p_k, \varepsilon), \tag{5.11}$$

where

$$M_\oplus^2(p_k, \varepsilon) = (1+\varepsilon)\|A\nabla u_h - p_k\|_{\bar{A}}^2 + (1+\tfrac{1}{\varepsilon})C_\Omega^2 \|f + \operatorname{div} p_k\|_0^2. \tag{5.12}$$

Since $\|Av\|_{\bar{A}} = \|v\|_A$, we can write

$$\begin{aligned} \|A\nabla u_h - p_k\|_{\bar{A}} &\leq \|A\nabla u_h - A\nabla u\|_{\bar{A}} + \|A\nabla u - p_k\|_{\bar{A}} \\ &= \|\nabla u_h - \nabla u\|_A + \|A\nabla u - p_k\|_{\bar{A}}. \end{aligned}$$

The norm $\|\cdot\|_{\bar{A}}$ is equivalent to the L^2-norm, so there exists a constant c_A, such that the second term in the right-hand side can be bounded by

$$\|A\nabla u - p_k\|_{\bar{A}} \leq c_A \|A\nabla u - p_k\|_0 \leq c_A \|A\nabla u - p_k\|_{\operatorname{div}} \leq c_A \varepsilon.$$

Hence, we obtain the following estimate for the first term in (5.12):

$$\|A\nabla u_h - p_k\|_{\bar{A}} \leq \|\nabla u - \nabla u_h\|_A + \mathcal{O}(\varepsilon). \tag{5.13}$$

Since $f = -\operatorname{div} A\nabla u$, we can bound the second term in (5.12) as follows:

$$\|\operatorname{div} p_k + f\|_0 = \|\operatorname{div} p_k - \operatorname{div} A\nabla u\|_0 \leq \|p_k - A\nabla u\|_{\operatorname{div}} \leq \varepsilon. \tag{5.14}$$

With (5.13) and (5.14), we can rewrite (5.12) as

$$\begin{aligned} M_\oplus^2(p_k, \varepsilon) &\leq (1+\varepsilon)(\|\nabla u - \nabla u_h\|_A^2 + \mathcal{O}(\varepsilon)) + (1+\tfrac{1}{\varepsilon})C_\Omega^2 \varepsilon^2 \\ &= \|\nabla u - \nabla u_h\|_A^2 + \mathcal{O}(\varepsilon). \end{aligned} \tag{5.15}$$

Hence, the bound $M_\oplus^2(p_k, \varepsilon) \to \|\nabla u - \nabla u_h\|_A^2$ as $\varepsilon \to 0$. □

Remark 5.6. *Note that $\|\nabla u - \nabla u_h\|_A^2$ also converges to zero as u_h converges to the exact solution u. From* (5.15)*, one can see that it is necessary to choose Y_h such that $\varepsilon \to 0$ faster than $\|\nabla u - \nabla u_h\|_A^2$, i.e., such that Y_h has better approximation properties than V_h, in order to get a sharp bound.*

5.2 Steps involved in minimizing $M_\oplus^2(y, \beta)$

As mentioned above, we need to find parameters y and β which minimize the majorant. To do this, we apply an interleaved iteration process in which we alternately fix one of the variables and minimize with respect to the other. This process, which we summarize in the following, has been described, e.g., in [59, 62].

Step 1 Minimization with respect to y: Assume that $\beta > 0$ is given and fixed, either by an initial guess or as a result of Step 2 below. Thereby, the majorant $M^2_\oplus(y)$ is a quadratic function of y and we calculate its Gateaux-derivative $M^2_\oplus(y)'$ with respect to y in direction $\tilde{y}$. Setting $M^2_\oplus(y)' = 0$, we obtain

$$\begin{aligned} b_1 \int_\Omega A^{-1} y \cdot \tilde{y}\, dx + b_2 \int_\Omega \operatorname{div} y\ \operatorname{div} \tilde{y}\, dx \\ = b_1 \int_\Omega \nabla u_h \cdot \tilde{y}\, dx - b_2 \int_\Omega f\ \operatorname{div} \tilde{y}\, dx, \end{aligned} \tag{5.16}$$

where $b_1 = 1 + \beta$ and $b_2 = (1 + \frac{1}{\beta})C^2_\Omega$, as defined in (5.9). In order to solve (5.16), we choose a finite-dimensional subspace $Y_h \subset H(\Omega, \operatorname{div})$ and search for a solution $y_h \in Y_h$. Testing in all directions $\tilde{y} \in Y_h$ leads to a linear system of equations which we write as

$$\mathbf{L}\mathbf{y} = \mathbf{r}. \tag{5.17}$$

Here, $\mathbf{L}$ and $\mathbf{r}$ are the matrix and the vector induced by the left hand side and the right hand side of equation (5.16), respectively. By solving (5.17), we obtain the coefficient vector $\mathbf{y}$ for the discrete function y_h minimizing $M^2_\oplus(y)$ in $Y_h \subset H(\Omega, \operatorname{div})$. Note that this process requires non-negligible cost as we need to assemble $\mathbf{L}$ and $\mathbf{r}$ and solve the system (5.17).

Step 2 Minimization with respect to β: Assume that y_h is given from Step 1. By direct calculation, we see that $M^2_\oplus(\beta)$ is minimized with respect to β by setting

$$\beta \;=\; C_\Omega \sqrt{\frac{U_2}{U_1}}, \tag{5.18}$$

where U_1 and U_2 are as defined in (5.9). Note that the evaluation of U_1 and U_2 (and thus β) requires only the evaluation of integrals, and thus involves negligible cost.

Steps 1 and 2 are repeated iteratively. We will refer to one loop of applying Step 1 and Step 2 as one *interleaved iteration*. Once we have computed minimizers y_h and β, the computation of the majorant $M^2_\oplus(y_h, \beta)$ is straightforward as it requires only the evaluation of the integrals.

Note that the matrix $\mathbf{L}$ can be written as

$$\mathbf{L} = b_1 \mathbf{L}_1 + b_2 \mathbf{L}_2, \tag{5.19}$$

where $\mathbf{L}_1$ and $\mathbf{L}_2$ correspond to the terms $\int_\Omega A^{-1} y \cdot \tilde{y}\, dx$ and $\int_\Omega \operatorname{div} y\ \operatorname{div} \tilde{y}\, dx$ in (5.16), respectively. Since the matrices $\mathbf{L}_1$ and $\mathbf{L}_2$ in (5.19) do not change in the interleaved iteration process, they need to be assembled only once. Analogously to (5.19), we can write $\mathbf{r}$ as

$$\mathbf{r} = b_1 \mathbf{r}_1 - b_2 \mathbf{r}_2, \tag{5.20}$$

where $\mathbf{r}_1$ and $\mathbf{r}_2$ correspond to the terms $\int_\Omega \nabla u_h \cdot \tilde{y}_h \, dx$ and $\int_\Omega f \ \operatorname{div} \tilde{y} \, dx$ in (5.16), respectively. The terms $\mathbf{r}_1$ and $\mathbf{r}_2$ also need to be assembled only once since they also do not change in the interleaved iteration process. The full matrix $\mathbf{L}$ and vector $\mathbf{r}$, however, do change in each iteration, because of the change in β and y_h. Based on past numerical studies, see, e.g., [59, 62], and the results presented in Sections 5.4 and 5.5, it has been found that for linear problems, one or two such interleaved iterations are enough for obtaining a sufficiently accurate result.

To recapitulate, we summarize the steps for computing the majorant in Algorithm 5.1.

Algorithm 5.1 Computation of the majorant $M_\oplus$

Input: u_h, f, C_Ω, Y_h
Output: $M_\oplus$

$\beta :=$ initial guess
Assemble and store $\mathbf{L}_1$, $\mathbf{L}_2$, $\mathbf{r}_1$, $\mathbf{r}_2$
while convergence is not achieved and maximum number of interleaved iterations is not reached **do**
  $\mathbf{L} := (1+\beta)\mathbf{L}_1 + (1+\frac{1}{\beta})C_\Omega^2 \mathbf{L}_2$
  $\mathbf{r} := (1+\beta)\mathbf{r}_1 - (1+\frac{1}{\beta})C_\Omega^2 \mathbf{r}_2$
  Solve $\mathbf{L}\mathbf{y} = \mathbf{r}$ for $\mathbf{y}$
  $U_1 := \|A\nabla u_h - y_h\|_A^2$
  $U_2 := \|\operatorname{div} y_h + f\|_0^2$
  $\beta := C_\Omega \sqrt{U_2/U_1}$
end while
$M_\oplus(y,\beta) := \sqrt{(1+\beta)U_1 + (1+\frac{1}{\beta})C_\Omega^2 U_2}$

Remark 5.7. *Note that the space $H(\Omega, \operatorname{div})$, where the auxiliary quantity y is sought, is a global space, and for a general complicated problem, it is not immediately clear how to locally compute y without global effect. That being said, a local version of our estimator can be devised for specific problems and data (like equilibration of flux approach), however, that will restrict its generality, which is not very appealing to us. Therefore, we focus on computing the majorant from the global minimization problem.*

5.3 Quality indicator and local error indicator

So far, we have defined the majorant and discussed how we minimize (numerically) the majorant over Y_h. Another important question, especially in the light of adaptive local refinement, is whether a calculated majorant does correctly capture the error distribution. In order to have an indication for how close the computed majorant is

to the exact error, we define the *efficiency index* $I_\oplus$ by

$$I_\oplus = \frac{M_\oplus(y,\beta)}{\|\nabla u - \nabla u_h\|_A}. \tag{5.21}$$

Obviously, the closer $I_\oplus$ is to 1, the better the estimate. From the proof of Lemma 5.5, we recall the following observation:

$$\begin{gathered} b_1U_1 \to \|\nabla u - \nabla u_h\|_A \quad \text{and} \quad b_2U_2 \to 0, \\ \text{as } y_h \in H(\Omega,\mathrm{div}) \to A\nabla u. \end{gathered} \tag{5.22}$$

From this, we deduce the following quality indicator.

Proposition 5.8. *The distribution of the exact error is captured correctly, if*

$$b_1U_1 > C_\oplus\ b_2U_2 \tag{5.23}$$

with some constant $C_\oplus > 1$.

This criterion is easy to check numerically, since the terms appearing in (5.23) are evaluated in the process of minimizing $M^2_\oplus(y,\beta)$. It was found in the numerical examples presented in Sections 5.4 and 5.5 that a correct distribution of the error is obtained, if $C_\oplus \geq 5$, even if this choice may be conservative in some cases.

Remark 5.9. *For the choice of* $C_\oplus \geq 5$, *we have* $b_2U_2 < 1/5\ b_1U_1$, *and therefore,* $\|\nabla u - \nabla u_h\|_A \leq \sqrt{1.2\ b_1U_1}$. *One can see from all the tables in Sections 5.4 and 5.5, that whenever this criterion is satisfied, we have* $I_\oplus \leq 1.2$ *(the ratio of* $\sqrt{b_1U_1}/\|\nabla u - \nabla u_h\|_A$ *is of the same magnitude as* $\sqrt{1+1/C_\oplus}$. *Note that this criterion does not require* b_2U_2 *to be close to zero, but just less than* $1/5$ *of* b_1U_1. *Since these approximations (of the original problem and the auxiliary problem in* $H(\Omega,\mathrm{div})$*) are monotonically convergent, the approximation at any level will only improve at the next refinement level, and this is why the results get better for any further refinement. Clearly, all the terms are fully computable, and thus, usable in an algorithm.*

We define the local error indicator η_Q on a cell Q as the restriction of the first component of the majorant to the cell Q, i.e., by

$$\eta^2_Q(y_h) = \int_Q (\nabla u_h - A^{-1}y_h)(A\nabla u_h - y_h)\,dx. \tag{5.24}$$

The factor $(1+\beta)$ is omitted, since this scalar factor is the same for all cells of the domain. As remarked in the observation (5.22), the first component will converge to the exact error, thus providing a good indicator for the error distribution. A more detailed discussion of this indicator, including a proof of the convergence to the true error distribution, can be found in [84, Sec. 3.6.4].

5.4 Efficient computation/implementation

We now discuss the efficiency and the computational cost of the proposed estimator based on the global minimization steps presented in Section 5.2. We again consider Example 5.1 from Section 5.1. All the computations for this example and the examples presented in Section 5.5 are performed in MATLAB®, and the linear systems (2.10) and (5.17) are solved using the in-built direct solver. One can, however, also use efficient iterative solvers, see, e.g., [42, 43] for (2.10). The right-hand-side f and the boundary conditions g_0 are determined by the prescribed exact solution u.

We study the efficiency of the majorant based on *straight forward* computational procedure, as discussed in Section 5.4.1, and based on *cost-efficient* procedure, as discussed in Section 5.4.2, which coarsens the mesh and increases the polynomial degree simultaneously. This alternative cost-efficient procedure will then be used in Section 5.5 for further numerical examples. In all the numerical results of Example 5.1 in this Section, the initial guess for β is 0.01.

In the tables, we indicate the mesh-size by the number of interior knot spans of the knot vectors s and t, respectively. Recall from Definition 3.1 that by this, we mean the number of knot spans without counting the vanishing knot spans at the beginning and the end of the open knot vectors. For example, if

$$\begin{aligned} s &= (0,0,0,\tfrac{1}{4},\tfrac{1}{2},\tfrac{3}{4},1,1,1) \\ t &= (0,0,0,0,\tfrac{1}{2},\tfrac{1}{2},1,1,1,1), \end{aligned}$$

then the mesh-size is 4×3, since the empty knot span $(\frac{1}{2},\frac{1}{2})$ in t is also counted as an interior knot span.

We compare the timings for assembling and for solving the linear systems (2.10) and (5.17), as well as the total time for assembling and solving. In the presented tables, these timings are shown in the columns labelled "assembling-time", "solving-time", and "sum", respectively. The label "pde" indicates that the column corresponds to solving the partial differential equation (2.10), i.e., to assembling $\mathbf{K}$ and solving (2.10) for $\mathbf{u}$. The label "est." indicates that the timings correspond to the estimator, i.e., assembling $\mathbf{L}$ and solving (5.17) for $\mathbf{y}$. In the column labelled "$\frac{\text{est.}}{\text{pde}}$", we present the ratio of these timings. Note that these timing ratios were computed *before* rounding the numbers, i.e., taking the ratios of the reported numbers may result in slightly different values.

The efficiency indices $I_\oplus$ (see (5.21)) computed in the numerical examples are presented in tables. In order to check the quality criterion discussed in Section 5.3, we present the values of b_1U_1, b_2U_2, and $C_\oplus$ and see whether the inequality (5.23) is fulfilled or not. To indicate the quality of the error distribution captured by the majorant, we plot which cells are marked for refinement based on the exact local error and the fixed-percentile criterion (2.12) (plotted in black), and compare this to the refinement marking based on the criterion (2.12) applied to the computed error estimate (plotted in magenta).

5.4.1 Straightforward procedure

Analogously to V_h in (3.20), we choose a function space $\widehat{Y}_h$ on the parameter domain and we define the function space Y_h by the push-forward

$$Y_h = \widehat{Y}_h \circ G^{-1}.$$

Example 5.1 (Straightforward Procedure) For our first choice for $\widehat{Y}_h$, we use the same mesh as for $\widehat{V}_h$, and we choose

$$\widehat{Y}_h = \mathcal{S}_h^{p+1,p} \otimes \mathcal{S}_h^{p,p+1}, \tag{5.25}$$

where $\mathcal{S}_h^{p,q}$ denotes the space of NURBS functions of degree p and C^{p-1}-continuity in the first coordinate, and degree q and C^{q-1}-continuity in the second coordinate (cf. [21, 23, 24, 38]). The parameter h indicates the characteristic cell-size of the mesh for $\widehat{V}_h$.

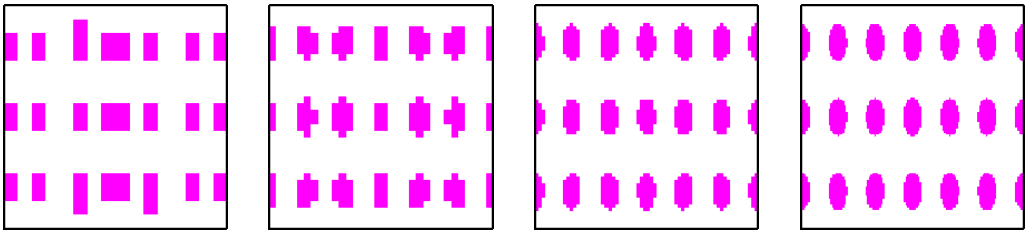
We consider the same setting as presented in Example 5.1 in Section 5.1. In Table 5.1, we present the computed efficiency indices obtained with this choice of Y_h, which show that upper bound approaches 1 (representing exact error) as the mesh is refined. The dashed line in Table 5.1 indicates that the criterion (5.23) is fulfilled with $C_\oplus$ 5 (actually 4.94) starting from the mesh 64×64.

The cells marked by the error estimator are shown in Figure 5.4. When comparing these plots to those presented in Figure 5.1, we see that the error distribution is captured accurately starting from the mesh 16×16.

The timings presented in Table 5.2, however, show that the computation of the error estimate is costlier (about 4.5 times) than assembling and solving the original problem. This is not surprising, since, when N_u denotes the number of degrees of freedom (DOF) of u_h, the number of DOF of y_h, which is vector-valued, is asymptotically $2N_u$. This results in higher assembly time and the solution time for the linear system (where a direct solver is used). Clearly, this straightforward approach is not cost-efficient.

mesh-size	$I_\oplus$	b_1U_1	b_2U_2	$C_\oplus$
8×8	3.43	2.62e+01	1.17e+02	0.2
16×16	1.92	6.07e-01	6.19e-01	1.0
32×32	1.41	2.29e-02	9.71e-03	2.4
64×64	1.20	1.15e-03	2.33e-04	4.9
128×128	1.10	6.51e-05	6.54e-06	10.0
256×256	1.05	3.87e-06	1.95e-07	19.8
512×512	1.03	2.36e-07	5.94e-09	39.7

Table 5.1: Efficiency index and components of the majorant in Example 5.1, $\widehat{Y}_h$ as in (5.25).



(a) 16×16 (b) 32×32 (c) 64×64 (d) 128×128

Figure 5.4: Cells marked by error estimator with $\psi = 0.8$ in Example 5.1, $\widehat{Y}_h$ as in (5.25).

mesh-size	#DOF		assembling-time			solving-time			sum		
	u_h	y_h	pde	est.	"$\frac{\text{est.}}{\text{pde}}$"	pde	est.	"$\frac{\text{est.}}{\text{pde}}$"	pde	est.	"$\frac{\text{est.}}{\text{pde}}$"
8×8	100	220	0.04	0.17	4.39	<0.01	<0.01	5.16	0.04	0.17	4.40
16×16	324	684	0.14	0.59	4.25	<0.01	0.01	5.39	0.14	0.60	4.26
32×32	1156	2380	0.46	2.17	4.70	0.01	0.03	4.71	0.47	2.20	4.70
64×64	4356	8844	1.82	8.51	4.68	0.03	0.20	6.15	1.85	8.70	4.70
128×128	16900	34060	7.38	34.19	4.63	0.15	0.87	5.70	7.54	35.06	4.65
256×256	66564	133644	33.30	149.78	4.50	0.84	5.66	6.78	34.14	155.44	4.55
512×512	264196	529420	191.11	766.10	4.01	3.77	33.92	9.00	194.88	800.03	4.11

Table 5.2: Number of DOF and timings in Example 5.1, $\widehat{Y}_h$ as in (5.25).

5.4.2 Alternative cost-efficient procedure

Recall that the cost of Step 1 of the algorithm presented in Section 5.2 depends on the choice of $Y_h \subset H(\Omega, \operatorname{div})$. As shown in Lemma 5.5, we can make the estimate as sharp as we desire by choosing a suitably large space Y_h. However, the larger Y_h is chosen, the more costly setting up and solving the system (5.17) becomes. Clearly, it is highly desirable to keep the cost for error estimation below the cost for solving the original problem.

As discussed above, choosing $\widehat{Y}_h$ as in (5.25) does not result in a cost-efficient method. Apart from the fact that y_h is vector-valued while u_h is scalar, another aspect contributes to the high cost for the procedure presented in Section 5.4.1. Recall that, by choosing $\widehat{Y}_h$ as in (5.25), we have

$$y_1 \in \mathcal{S}_h^{p+1,p},$$
$$y_2 \in \mathcal{S}_h^{p,p+1},$$

i.e., the components of y_h are in different spline spaces. Hence, we have to compute different basis functions for y_1 and y_2 (note that this can be a costly procedure for higher polynomial degrees). Furthermore, when assembling, for example, the matrix $\mathbf{L}_1$, we need to compute integrals over products of basis functions of the form

$$\int_\Omega R_i R_j \, dx.$$

With $\widehat{Y}_h$ as in (5.25), the product R_iR_j of basis functions of y_1 is different than the product of basis functions of y_2, hence, the integrals have to be evaluated independently for y_1 and y_2.

Example 5.1 (Case 1) In the light of these observations, we study the following alternative choice for $\widehat{Y}_h$.

$$\widehat{Y}_h = \mathcal{S}_h^{p+1,p+1} \otimes \mathcal{S}_h^{p+1,p+1}. \tag{5.26}$$

We refer to this setting as *Case 1* in the remainder of the paper. With this choice, y_1 and y_2 are contained in the same spline spaces. Hence, the basis functions need to be computed only once, and any computed function values can be used for both components of y_h.

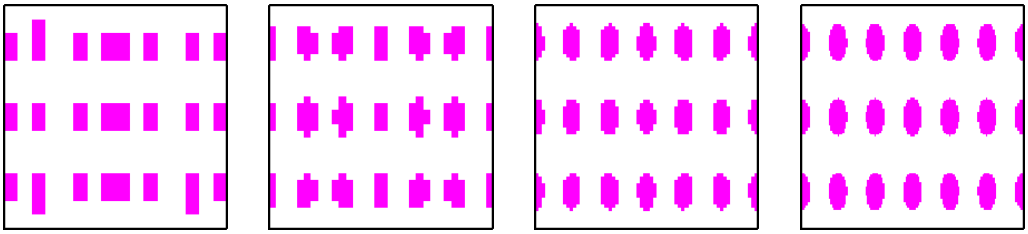
The computed efficiency indices are presented in Table 5.3, which show that we obtain even better (i.e., sharper) upper bounds for the exact error with $\widehat{Y}_h$ as in (5.26) than with the choice (5.25). When we compare the plots of the cells marked by the error estimator in Figure 5.5 to the plots in Figure 5.1, we see that the error distribution is again captured accurately starting from the mesh 16×16. The dashed line in Table 5.3 indicates that the criterion (5.23) is fulfilled with $C_\oplus \geq 5$ starting from the mesh 64×64. Note that we also have $C_\oplus = 6.5 > 5$ on the mesh 8×8. Since this mesh is too coarse to resolve the highly oscillating gradients of the considered exact solution u, this result is considered as an outlier.

The timings obtained with this method are presented in Table 5.4. This approach reduces the total time needed for computing the majorant from a factor of about 4.5 to a factor of approximately 3 compared to the time for assembling and solving the original problem. Nevertheless, a factor of 3 in the timings is still not very appealing, and demands further reduction in cost.

mesh-size	$I_\oplus$	b_1U_1	b_2U_2	$C_\oplus$
8×8	2.77	8.08e+01	1.24e+01	6.5
16×16	1.71	5.75e-01	3.96e-01	1.5
32×32	1.32	2.14e-02	7.05e-03	3.0
64×64	1.16	1.11e-03	1.78e-04	6.2
128×128	1.08	6.39e-05	5.08e-06	12.6
256×256	1.04	3.83e-06	1.53e-07	25.0
512×512	1.02	2.35e-07	4.69e-09	50.1

Table 5.3: Efficiency index and components of the majorant in Example 5.1, Case 1.

Remark 5.10. *Note that the use of equal degree polynomials for both the components of $\widehat{Y}_h$ is only possible because of extra regularity readily available from NURBS basis functions. A counter-part is not possible in FEM case simply because the derivatives of FEM basis functions (with C^0 regularity) is only in L^2, and hence, one can not*



(a) 16×16 (b) 32×32 (c) 64×64 (d) 128×128

Figure 5.5: Cells marked by error estimator with $\psi = 0.8$ in Example 5.1, Case 1.

mesh-size	#DOF		assembling-time			solving-time			sum		
	u_h	y_h	pde	est.	"$\frac{\text{est.}}{\text{pde}}$"	pde	est.	"$\frac{\text{est.}}{\text{pde}}$"	pde	est.	"$\frac{\text{est.}}{\text{pde}}$"
8×8	100	242	0.04	0.11	2.78	<0.01	<0.01	1.51	0.04	0.11	2.76
16×16	324	722	0.12	0.34	2.86	<0.01	0.01	5.33	0.12	0.35	2.90
32×32	1156	2450	0.46	1.35	2.94	0.01	0.05	7.69	0.47	1.40	3.01
64×64	4356	8978	1.77	5.30	2.99	0.03	0.27	8.02	1.80	5.57	3.09
128×128	16900	34322	7.39	21.89	2.96	0.16	1.45	9.26	7.55	23.34	3.09
256×256	66564	134162	33.00	94.69	2.87	0.84	8.83	10.54	33.84	103.52	3.06
512×512	264196	530450	191.59	498.20	2.60	3.83	61.45	16.06	195.42	559.65	2.86

Table 5.4: Number of DOF and timings in Example 5.1, Case 1.

avoid using proper subspaces of $H(\Omega, \mathrm{div})$, e.g., Raviart-Thomas space (with unequal degree polynomials in both the dimensions for both the components). It is further important to note from a close inspection of Tables 5.1 and 5.3 that equal degree components of vector-valued quantity outperformed the unequal degree case.

In order to further reduce the computational cost, we reduce the number of DOF of y_h by coarsening the mesh by a factor K in each dimension. The number of DOF of y_h is thus reduced to $2N_u/K^2$ (asymptotically). The larger K is chosen, the greater the reduction of DOF will be. At the same time, if the coarsening is done too aggressively, sharp features might not be detected properly on coarse meshes. We counter the reduction in accuracy due to mesh-coarsening by increasing the polynomial degree of y_h by some positive integer k, i.e., we choose

$$\widehat{Y}_h = \mathcal{S}_{Kh}^{p+k,p+k} \otimes \mathcal{S}_{Kh}^{p+k,p+k}. \tag{5.27}$$

Note that, if desired, one could also choose different factors K_1 and K_2 and different degree increases k_1 and k_2 for the first and second component, respectively.

Remark 5.11. *With these choices of $\widehat{Y}_h$, we take advantage of the following specific property of univariate NURBS basis functions. For C^{p-1} regularity, increasing the polynomial degree by k only adds a total of k additional basis functions (see k-refinement in Section 3.1.1.4). In other words, the global smoothness can be increased*

at the cost of only a few additional DOF. Coarsening the mesh by a factor K, however, will reduce the number of DOF by the same factor K (asymptotically).

Moreover, as we will see from the three cases of Example 5.1, we get better efficiency indices with higher degree p and coarser meshes as compared to lower degree p and finer meshes. This phenomenon is similar to the p finite element discretization for problems with smooth solutions.

Note that Case 1 discussed above fits into this framework, since Case 1 corresponds to the choice $K = k = 1$.

Example 5.1 (Case 2) For the next setting, we apply moderate mesh-coarsening by choosing

$$K = k = 2 \text{ (i.e., } \widehat{Y}_h = \mathcal{S}_{2h}^{p+2,p+2} \otimes \mathcal{S}_{2h}^{p+2,p+2}\text{)}.$$

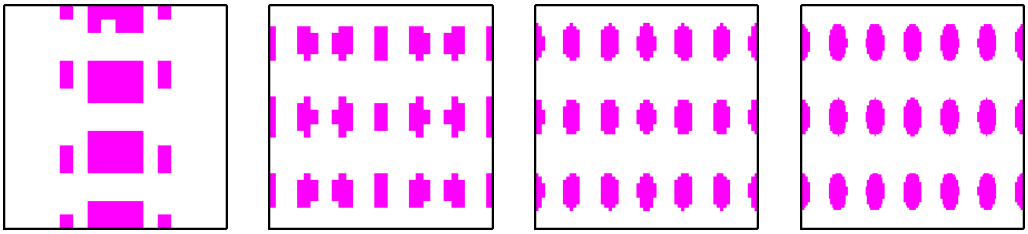
This setting will be referred to as *Case 2* in the remainder of this thesis. The computed efficiency indices along with the magnitudes of the terms b_1U_1 and b_2U_2 and their ratio $C_\oplus = b_1U_1/b_2U_2$ for Case 2 are presented in Table 5.5, and the marked cells are plotted in Figure 5.6. Both indicate that a good upper bound of the error and the correct error distribution are computed on fine meshes. On coarse meshes, however, the efficiency index is larger than in Case 1, which is due to the boundary effects. The timings presented in Table 5.6 show that, even though Case 2 is faster than Case 1, this approach still costs roughly as much as solving the original problem. This is due to the costlier evaluation of the higher degree basis functions, as well as the increased support and overlap of the basis functions, which results in more non-zero entries in $\mathbf{L}$ than in $\mathbf{K}$.

mesh-size	$I_\oplus$	b_1U_1	b_2U_2	$C_\oplus$
8×8	14.19	1.59e+03	8.53e+02	1.9
16×16	8.49	1.97e+01	4.32e+00	4.6
32×32	1.82	3.05e-02	2.41e-02	1.3
64×64	1.16	1.12e-03	1.76e-04	6.4
128×128	1.04	6.14e-05	2.24e-06	27.4
256×256	1.01	3.72e-06	3.32e-08	112.0
512×512	1.00	2.31e-07	5.13e-10	450.3

Table 5.5: Efficiency index and components of the majorant in Example 5.1, Case 2.

Example 5.1 (Case 3) To further improve the timings, we coarsen the mesh more aggressively by a factor of 4 and, at the same time, increase the polynomial degree of y_h by 4, as compared to u_h, i.e.,

$$K = k = 4 \text{ (i.e., } \widehat{Y}_h = \mathcal{S}_{4h}^{p+4,p+4} \otimes \mathcal{S}_{4h}^{p+4,p+4}\text{)}.$$



(a) 16×16 (b) 32×32 (c) 64×64 (d) 128×128

Figure 5.6: Cells marked by error estimator with $\psi = 0.8$ in Example 5.1, Case 2.

mesh-size	#DOF		assembling-time			solving-time			sum		
	u_h	y_h	pde	est.	"$\frac{\text{est.}}{\text{pde}}$"	pde	est.	"$\frac{\text{est.}}{\text{pde}}$"	pde	est.	"$\frac{\text{est.}}{\text{pde}}$"
8×8	100	128	0.03	0.05	1.39	<0.01	<0.01	1.16	0.04	0.05	1.39
16×16	324	288	0.14	0.18	1.29	<0.01	<0.01	0.92	0.14	0.18	1.28
32×32	1156	800	0.54	0.59	1.10	0.01	0.02	2.32	0.55	0.61	1.11
64×64	4356	2592	1.91	2.33	1.22	0.04	0.08	2.09	1.95	2.40	1.23
128×128	16900	9248	7.46	9.54	1.28	0.19	0.51	2.75	7.64	10.05	1.32
256×256	66564	34848	33.93	39.02	1.15	0.90	2.59	2.88	34.82	41.60	1.19
512×512	264196	135200	196.23	177.98	0.91	4.08	15.91	3.90	200.31	193.89	0.97

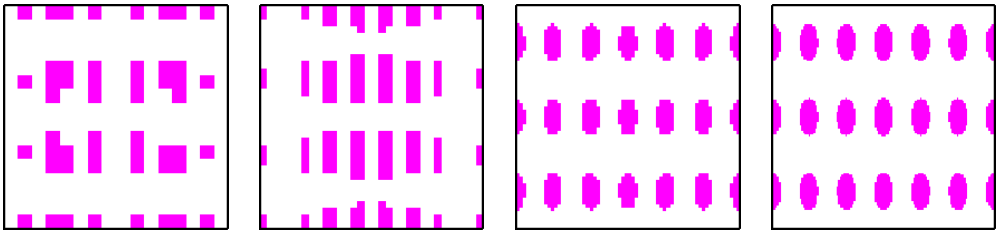
Table 5.6: Number of DOF and timings in Example 5.1, Case 2.

We refer to this setting as *Case 3* in the remainder of this thesis. This aggressive coarsening notably affects the efficiency index on coarse meshes, see Table 5.7. On fine meshes, however, the efficiency indices are close to 1 in all presented cases. The number of DOF of y_h in Case 3 is only $N_u/8$ (asymptotically). The timings presented in Table 5.8 show that this setting results in a method which can be performed significantly faster (at almost half of the cost) than solving the original problem. The more aggressive reduction of DOF outweighs the additional costs mentioned above, even though the polynomial degree is now increased by 4.

mesh-size	$I_\oplus$	b_1U_1	b_2U_2	$C_\oplus$
8×8	11.28	5.38e+02	1.01e+03	0.5
16×16	36.43	2.83e+02	1.60e+02	1.8
32×32	12.63	2.04e+00	5.81e-01	3.5
64×64	1.17	1.13e-03	1.88e-04	6.0
128×128	1.01	5.98e-05	3.79e-07	157.8
256×256	1.00	3.70e-06	1.24e-09	$> 10^3$
512×512	1.00	2.31e-07	5.32e-12	$> 10^4$

Table 5.7: Efficiency index and components of the majorant in Example 5.1, Case 3.

We now comment on the interleaved iterations. The results in the Tables 5.1, 5.3,



(a) 16×16 (b) 32×32 (c) 64×64 (d) 128×128

Figure 5.7: Cells marked by error estimator with $\psi = 0.8$ in Example 5.1, Case 3.

mesh-size	#DOF		assembling-time			solving-time			sum		
	u_h	y_h	pde	est.	"$\frac{\text{est.}}{\text{pde}}$"	pde	est.	"$\frac{\text{est.}}{\text{pde}}$"	pde	est.	"$\frac{\text{est.}}{\text{pde}}$"
8×8	100	128	0.04	0.03	0.76	<0.01	<0.01	1.09	0.04	0.03	0.76
16×16	324	200	0.14	0.10	0.69	<0.01	<0.01	0.61	0.14	0.10	0.69
32×32	1156	392	0.54	0.31	0.57	0.01	<0.01	0.34	0.55	0.31	0.57
64×64	4356	968	1.90	1.19	0.63	0.04	0.01	0.26	1.94	1.20	0.62
128×128	16900	2888	7.49	4.86	0.65	0.16	0.14	0.84	7.66	4.99	0.65
256×256	66564	9800	33.90	20.15	0.59	0.91	0.82	0.91	34.81	20.98	0.60
512×512	264196	35912	194.25	84.70	0.44	4.10	5.45	1.33	198.35	90.15	0.45

Table 5.8: Number of DOF and timings in Example 5.1, Case 3.

5.5, and 5.7 were obtained by applying only two interleaved iterations, as described in Section 5.2. As mentioned there, a sufficiently accurate result can be obtained already after the first such iteration. To illustrate this, we present the efficiency indices for Case 3 in Table 5.9, which were obtained after one, two, and four interleaved iterations, respectively. The efficiency index does vary notably on the coarser meshes, but since all of these values greatly overestimate the exact error, they do not correctly capture the error distribution. On meshes, where the criterion (5.23) is fulfilled, and thus the error distribution is correctly recovered, the differences due to more interleaved iterations are insignificant.

Remark 5.12. *The observations discussed above illustrate that one has to balance the sharpness of the majorant on the one hand, and the required computational effort on the other hand. Note that in typical practical applications, the exact solution (and thus the sharpness of the majorant) is not known. Therefore, to address the balancing between sharpness and required computational effort, we propose the following strategy. If the mesh is coarse and the total computational cost for the error estimate is moderate, we apply no (or only moderate) coarsening. When the original mesh is fine (problem size being large), we coarsen the mesh more aggressively, and thereby, profit from the fast computation of the estimate. While exercising this strategy it is important to enforce the criterion* (5.23) *with* $C_\oplus \geq 5$.

mesh-size	interleaved iterations		
	1	2	4
8×8	11.84	11.28	11.25
16×16	80.31	36.43	33.78
32×32	17.36	12.63	10.11
64×64	1.20	1.17	1.17
128×128	1.01	1.01	1.01
256×256	1.00	1.00	1.00
512×512	1.00	1.00	1.00

Table 5.9: Comparison of $I_{\oplus}$ for different numbers of interleaved iterations, Example 5.1, Case 3.

5.5 Numerical examples for the upper bound

We now present further numerical examples which illustrate the potential of the proposed a posteriori error estimator. We will discuss the following three settings that were also discussed in Section 5.4. *Case 1:* $K = k = 1$, *Case 2:* $K = k = 2$, and *Case 3:* $K = k = 4$. As in Example 5.1, the initial guess for β is 0.01.

As discussed in Section 3.2.1, the parameter domain in all presented examples is the unit square $Q = (0, 1)^2$. The mesh-sizes in the two coordinate directions, which will be presented in the tables, are determined by the respective initial meshes, which in turn, are determined by the geometry mappings. The data presented in the tables is as described in the beginning of Section 5.4.

The figures plotted in black represent the computations based on the exact error, and the figures plotted in magenta represent the computations based on the majorant. In all examples presented in this section, cell-marking is based on the fixed-percentile criterion (2.12) from Section 2.2.5.1.

Example 5.2: Sinus Function on the Unit Square with $p = q = 4$

In this example, we consider a discretization with reduced regularity C^{p-r}, $r > 1$. We consider the same exact solution and the same physical domain as in Example 5.1, i.e.,

$$u = \sin(6\pi x)\sin(3\pi y),$$
$$\Omega = (0, 1)^2.$$

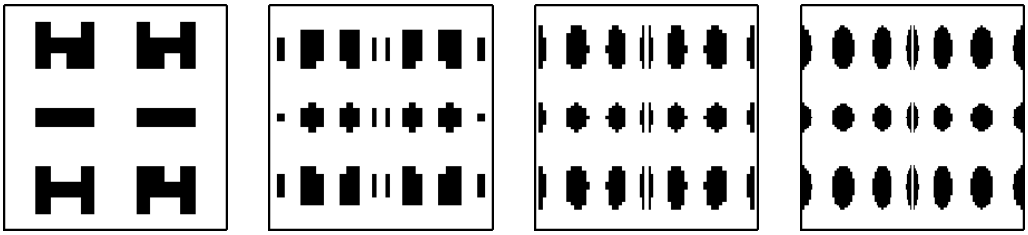
However, we now use B-splines of degree $p = q = 4$ to represent Ω, and we add a triple knot at the coordinates $x = 0.5$ and $y = 0.5$. The initial knot vectors are thus given by

$$s = t = (0, 0, 0, 0, 0, 0.5, 0.5, 0.5, 1, 1, 1, 1, 1),$$

and the geometry mapping is only C^1-continuous at the coordinate 0.5.

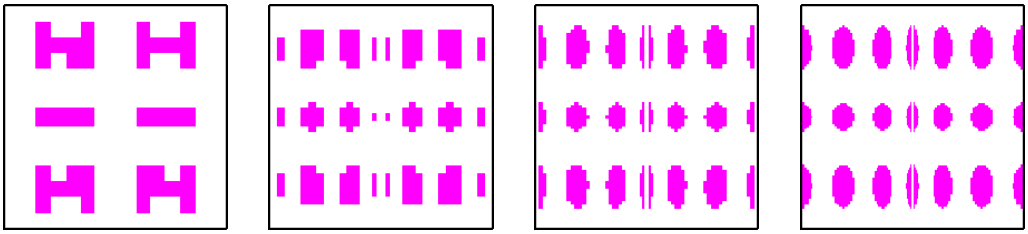
mesh-size	$I_\oplus$	b_1U_1	b_2U_2	$C_\oplus$
Case 1				
18×18	1.84	1.04e-03	9.00e-04	1.6
34×34	1.40	1.78e-06	7.23e-07	2.5
66×66	1.20	5.09e-09	1.00e-09	5.1
130×130	1.10	1.77e-11	1.74e-12	10.2
258×258	1.05	6.61e-14	3.25e-15	20.3
Case 2				
18×18	15.43	7.95e-02	5.75e-02	1.4
34×34	6.04	1.14e-05	3.53e-05	0.3
66×66	1.76	7.52e-09	5.69e-09	1.3
130×130	1.16	1.87e-11	3.01e-12	6.2
258×258	1.04	6.54e-14	2.49e-15	26.3
Case 3				
18×18	132.77	7.38e+00	2.76e+00	2.7
34×34	148.41	1.86e-02	9.53e-03	2.0
66×66	6.42	5.49e-08	1.21e-07	0.5
130×130	1.13	1.83e-11	2.39e-12	7.7
258×258	1.01	6.34e-14	3.78e-16	167.7

Table 5.10: Efficiency index and components of the majorant in Example 5.2.



(a) 18×18. (b) 34×34. (c) 66×66. (d) 130×130.

Figure 5.8: Cells marked by exact error with $\psi = 0.8$ in Example 5.2.



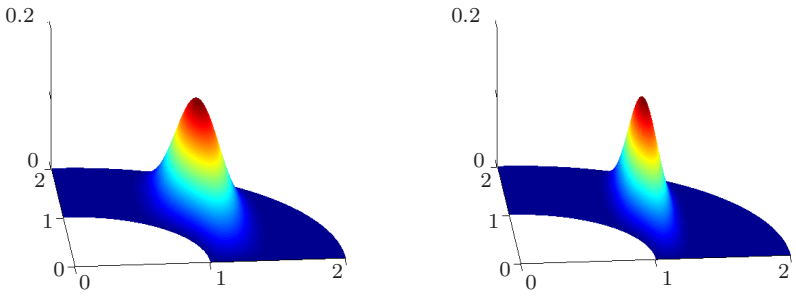
(a) 18×18, Case 1. (b) 34×34, Case 1. (c) 66×66, Case 2. (d) 130×130, Case 3.

Figure 5.9: Cells marked by error estimator with $\psi = 0.8$ in Example 5.2.

The computed efficiency indices are presented in Table 5.10. The dashed lines, which correspond to criterion (5.23) being fulfilled with $C_{\oplus} \geq 5$, again show that more aggressive mesh-coarsening requires a finer initial mesh. By this criterion, we get a good quality of the estimate and the indicated error distribution starting from the mesh 66×66 in Case 1, and from 130×130 in Cases 2 and 3.

We present the cells marked for refinement by the exact error in Figure 5.8, and the cells marked by the error estimator in Figure 5.9. Figure 5.9(a) shows that the error distribution is already captured on the mesh 18×18 in Case 1. In Case 2, we obtain a good indication of the error distribution on the mesh 66×66, i.e., before criterion (5.23) with $C_{\oplus} \geq 5$ is fulfilled. Once the error distribution is captured correctly on a certain mesh, it is also captured on all finer meshes (as in Example 5.1). Hence, we do not show all plots for all meshes and cases, but only the first meshes, on which the error distribution is captured correctly. Also, we omit the presentation of the timings, since the overall behaviour is as in Example 5.1.

Example 5.3: Quarter Annulus



(a) Example 5.3.a ($\alpha = 20$). (b) Example 5.3.b ($\alpha = 50$).

Figure 5.10: Exact solutions u on Ω, Example 5.3.

In this example, we consider a domain with a curved boundary (requiring a NURBS mapping for exact representation) and a problem whose solution has sharp peaks. The domain Ω represents a quarter annulus, which, in polar coordinates, is defined by $(r, \phi) \in (1, 2) \times (0, \frac{\pi}{2})$. Note that the circular parts of the domain boundary are represented exactly by the NURBS geometry mapping of degree 2, i.e., we have $p = q = 2$. We set $A = I$, and we prescribe the exact solution

$$u = (r-1)(r-2)\phi(\phi - \tfrac{\pi}{2})e^{-\alpha(r\cos\phi - 1)^2}.$$

We test our method with two values of α, namely,

$$\text{Example 5.3.a:} \quad \alpha = 20, \qquad \text{Example 5.3.b:} \quad \alpha = 50.$$

In both examples, this function has zero Dirichlet boundary values and a peak at $x = 1$, the sharpness of which is determined by the value of α. The exact solutions

are depicted in Figure 5.10. This example is chosen because of sharp peaks (large gradients) and curved boundary (requiring NURBS mapping for exact representation).

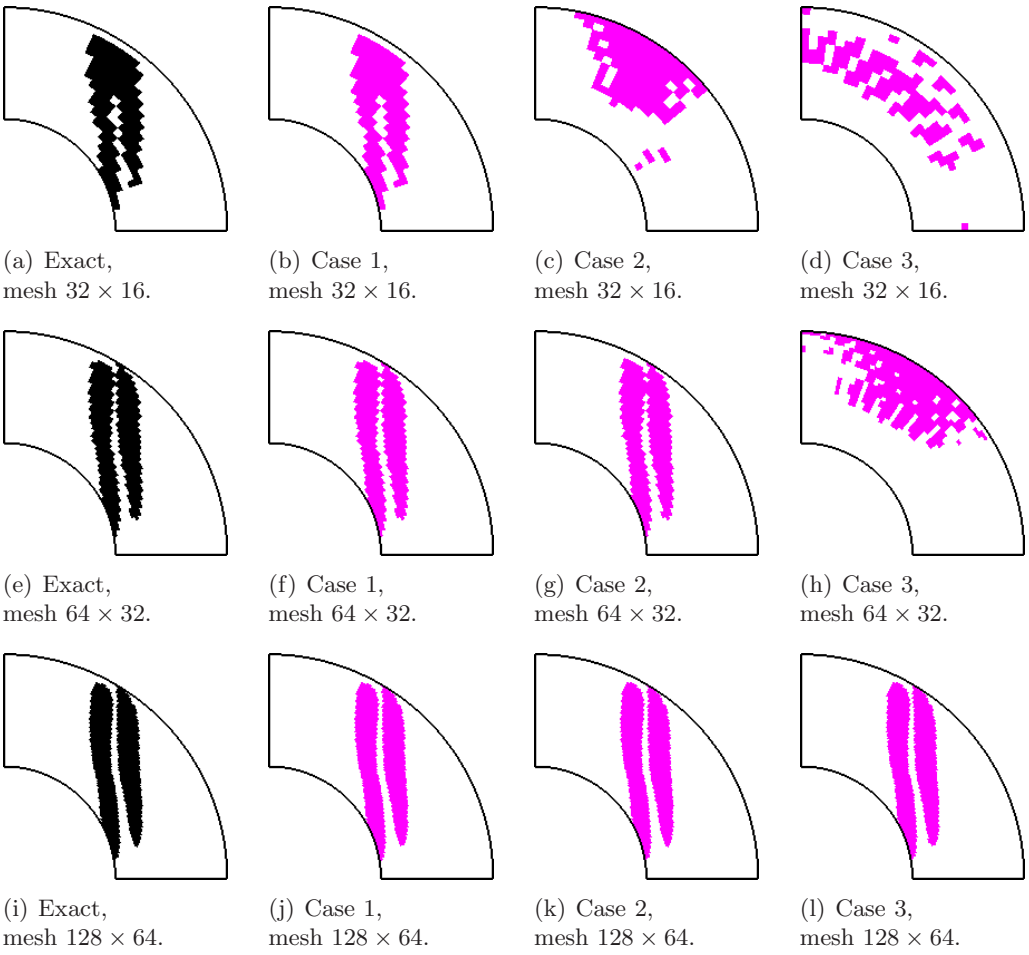
mesh-size	$I_\oplus$	b_1U_1	b_2U_2	$C_\oplus$
Case 1				
16×8	1.83	9.98e-04	3.59e-04	2.8
32×16	1.29	2.08e-05	6.51e-06	3.2
64×32	1.13	1.04e-06	1.44e-07	7.2
128×64	1.07	5.95e-08	4.00e-09	14.9
256×128	1.03	3.58e-09	1.20e-10	29.8
512×256	1.02	2.20e-10	3.67e-12	59.9
Case 2				
16×8	13.99	4.44e-02	3.51e-02	1.3
32×16	4.17	2.00e-04	8.43e-05	2.4
64×32	1.31	1.20e-06	3.66e-07	3.3
128×64	1.06	5.91e-08	3.36e-09	7.4
256×128	1.01	3.51e-09	4.60e-11	76.3
512×256	1.00	2.17e-10	6.96e-13	311.8
Case 3				
16×8	24.87	1.09e-01	1.42e-01	0.8
32×16	56.02	2.92e-02	2.22e-02	1.3
64×32	10.42	7.81e-05	2.16e-05	3.6
128×64	1.11	6.21e-08	6.61e-09	9.4
256×128	1.00	3.49e-09	1.02e-11	342.2
512×256	1.00	2.17e-10	3.27e-14	$> 10^3$

Table 5.11: Efficiency index and components of the majorant in Example 5.3.a ($\alpha = 20$).

In Tables 5.11 and 5.12, the efficiency index $I_\oplus$, the magnitudes of b_1U_1 and b_2U_2, and their ratio $C_\oplus$ are presented for both examples. The dashed lines indicate the mesh-size after which criterion (5.23) with $C_\oplus \geq 5$ is fulfilled. The distribution of the marked cells is depicted in Figures 5.11 and 5.12. As before, we observe that the error distribution is represented correctly if the criterion (5.23) is fulfilled with $C_\oplus \geq 5$.

When comparing Tables 5.11 and 5.12, as well as Figures 5.11 and 5.12, we notice the following. The more aggressive the mesh coarsening, and the sharper the peak, the more refinements are needed before criterion (5.23) is fulfilled and the error distribution is captured correctly.

Since the timings in Example 5.3.a and Example 5.3.b show the same behaviour as in the previous examples, both regarding assembling-time and solving-time, we omit the presentation of these numbers. Clearly, Case 3 outperforms Cases 1 and 2 in terms of cost-efficiency.



(a) Exact, mesh 32×16. (b) Case 1, mesh 32×16. (c) Case 2, mesh 32×16. (d) Case 3, mesh 32×16.

(e) Exact, mesh 64×32. (f) Case 1, mesh 64×32. (g) Case 2, mesh 64×32. (h) Case 3, mesh 64×32.

(i) Exact, mesh 128×64. (j) Case 1, mesh 128×64. (k) Case 2, mesh 128×64. (l) Case 3, mesh 128×64.

Figure 5.11: Marked cells with $\psi = 0.8$ in Example 5.3.a ($\alpha = 20$).

Example 5.4: Adaptive Refinement

We now test a basic adaptive refinement scheme based on the proposed error indicator. The exact solution for this example example is given by

$$u = (x^2 - x)(y^2 - y)e^{-100|(x,y)-(0.8,0.05)|^2 - 100|(x,y)-(0.8,0.95)|^2}.$$

The computational domain is again the unit square $\Omega = (0,1)^2$, and is represented by B-splines of degree $p = q = 2$. The function u, which is illustrated in Figure 5.13, has zero Dirichlet boundary values and has two peaks at the coordinates $(0.8, 0.05)$ and $(0.8, 0.95)$. Since the discussion of isogeometric local refinement schemes is out of the scope of this thesis (see Section 3.3.4 for an overview on local refinement methods), we apply adaptive refinement using tensor-product B-splines.

We apply adaptive refinement based on cell-marking with $\psi = 0.75$, starting on an initial mesh 16×16. On the first four steps, we apply Case 1, then Case 2 on

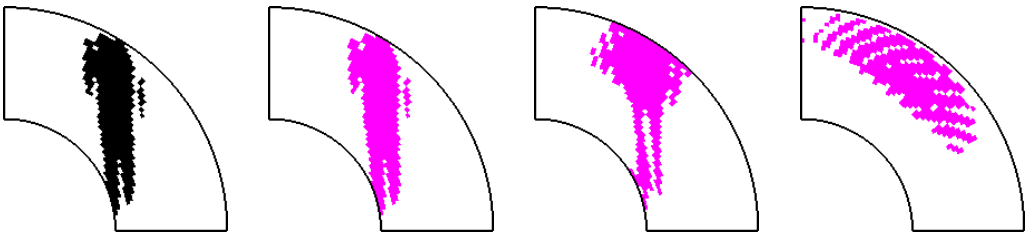
mesh-size	$I_\oplus$	b_1U_1	b_2U_2	$C_\oplus$
Case 1				
16×8	3.02	2.94e-02	1.78e-02	1.7
32×16	1.92	3.57e-04	1.83e-04	2.0
64×32	1.34	9.15e-06	3.22e-06	2.8
128×64	1.16	4.67e-07	7.56e-08	6.2
256×128	1.08	2.67e-08	2.12e-09	12.6
512×256	1.04	1.60e-09	6.32e-11	25.3
Case 2				
16×8	13.84	3.45e-01	6.49e-01	0.5
32×16	16.76	2.58e-02	1.53e-02	1.7
64×32	3.16	4.10e-05	2.80e-05	1.5
128×64	1.25	5.04e-07	1.24e-07	4.1
256×128	1.05	2.61e-08	1.33e-09	19.6
512×256	1.01	1.56e-09	1.89e-11	82.5
Case 3				
16×8	17.20	4.24e-01	1.11e+00	0.4
32×16	76.95	3.24e-01	5.41e-01	0.6
64×32	83.72	3.02e-02	1.83e-02	1.7
128×64	4.19	4.64e-06	2.44e-06	1.9
256×128	1.04	2.59e-08	1.02e-09	25.4
512×256	1.00	1.55e-09	2.22e-12	698.2

Table 5.12: Efficiency index and components of the majorant in Example 5.3.b ($\alpha = 50$).

mesh-size	$I_\oplus$	b_1U_1	b_2U_2	$C_\oplus$	Case
16×16	3.77	9.39e-05	3.49e-05	2.7	1
25×26	2.06	8.62e-07	8.11e-07	1.1	1
38×44	1.69	4.30e-08	2.35e-08	1.8	1
64×74	1.47	2.79e-09	1.19e-09	2.3	1
92×136	2.82	8.19e-10	4.87e-10	1.7	2
184×256	1.30	2.05e-11	4.55e-12	4.5	2
341×492	1.11	1.45e-12	1.47e-13	9.9	2
652×934	1.84	2.55e-13	1.07e-13	2.4	3
1304×1868	1.09	7.40e-15	3.63e-16	20.4	3

Table 5.13: Efficiency index, components of the majorant and applied cases in Example 5.4, adaptive refinement.

the next three steps, and thereafter Case 3. The efficiency indices and the applied cases are shown in Table 5.13. In Figure 5.14, the meshes and the marked cells are shown for steps 4, 7, and 9. Clearly, the correct areas of the domain are identified and

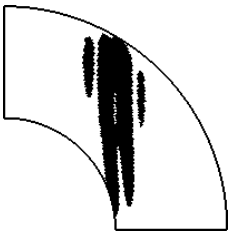


(a) Exact, mesh 64×32.

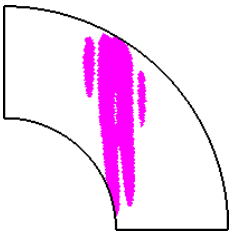
(b) Case 1, mesh 64×32.

(c) Case 2, mesh 64×32.

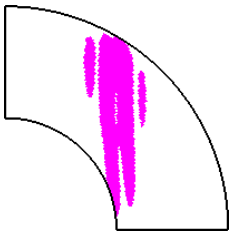
(d) Case 3, mesh 64×32.



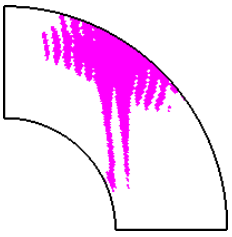
(e) Exact, mesh 128×64.



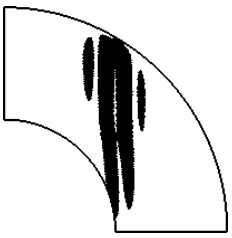
(f) Case 1, mesh 128×64.



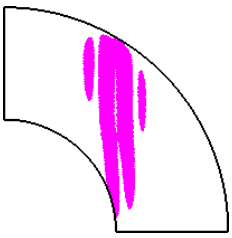
(g) Case 2, mesh 128×64.



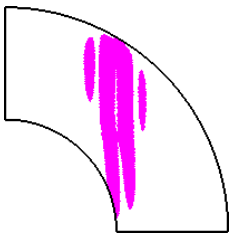
(h) Case 3, mesh 128×64.



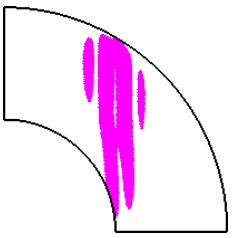
(i) Exact, mesh 256×128.



(j) Case 1, mesh 256×128.



(k) Case 2, mesh 256×128.



(l) Case 3, mesh 256×128.

Figure 5.12: Marked cells with $\psi = 0.8$ in Example 5.3.b ($\alpha = 50$).

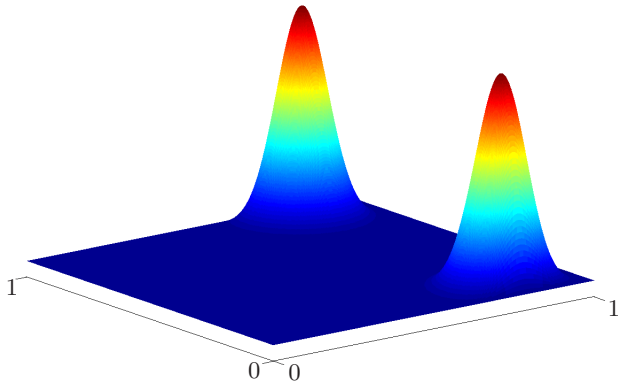
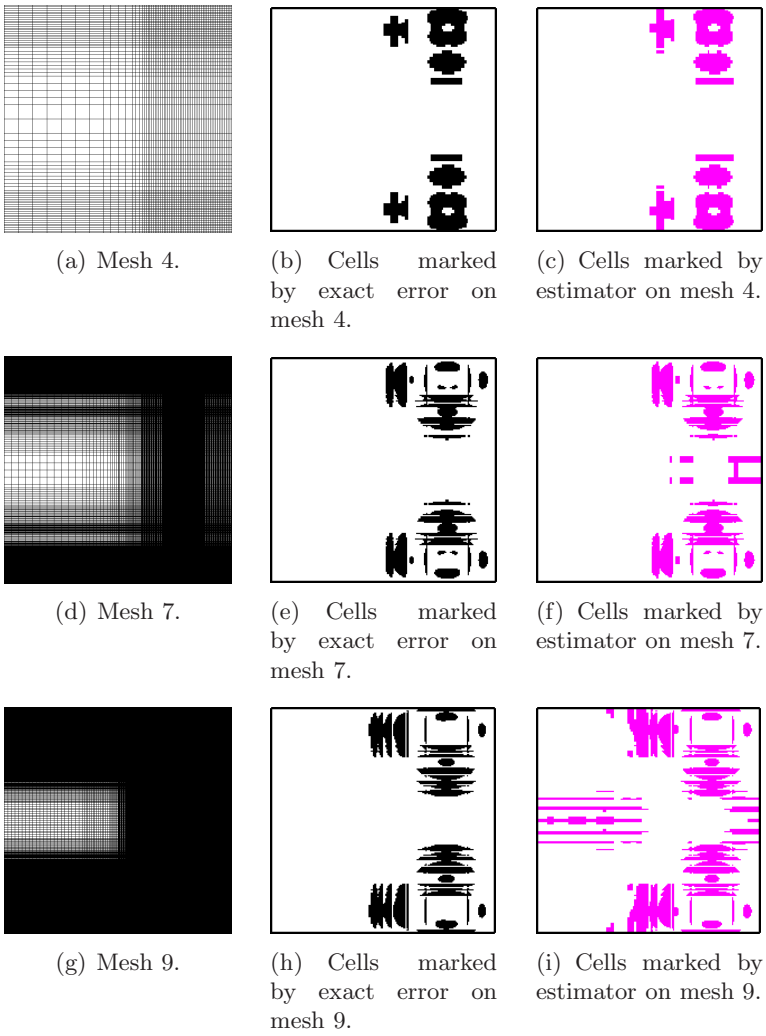


Figure 5.13: Exact solution, Example 5.4.



(a) Mesh 4.

(b) Cells marked by exact error on mesh 4.

(c) Cells marked by estimator on mesh 4.

(d) Mesh 7.

(e) Cells marked by exact error on mesh 7.

(f) Cells marked by estimator on mesh 7.

(g) Mesh 9.

(h) Cells marked by exact error on mesh 9.

(i) Cells marked by estimator on mesh 9.

Figure 5.14: Meshes and marked cells in Example 5.4, $\psi = 0.75$.

marked for refinement, and the error plots in Figure 5.15 show that the adaptive refinement converges faster than uniform refinement. Since the solution of the problem is sufficiently regular, the error plots in Figure 5.15 show that the adaptive refinement converges with the same rate as the uniform refinement, but with a better constant. Note that, due to the tensor-product structure of the mesh, many superfluous DOF are inserted outside of the marked areas, which worsens the rate of convergence for given total DOF.

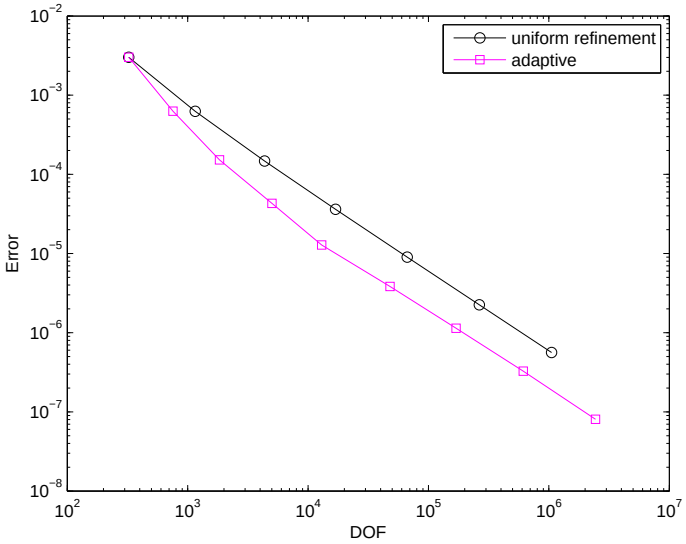


Figure 5.15: Error convergence, Example 5.4.

Example 5.5: L-shaped Domain

This example is a classical example for a posteriori error estimation and adaptive refinement studies (see, e.g., [25, 33, 34, 37, 56]). We consider the Laplace equation

$$\Delta u = 0 \tag{5.28}$$

with Dirichlet boundary conditions on the L-shaped domain $\Omega = (-1,1)^2 \backslash [0,1]^2$. In this example, we use a bilinear geometry mapping, i.e., $p = q = 1$. The function

$$u(r,\phi) = r^{\frac{2}{3}} \sin((2\phi - \pi)/3)$$

solves (5.28) and is used to prescribe Dirichlet boundary conditions. The solution has a singularity at the re-entrant corner at $(0,0)$. We compare uniform refinement and adaptive refinement in the tensor-product setting. In this example, we only apply Case 1, and we use $\psi = 0.9$ for cell-marking.

The magnitudes of the components $b_1 U_1$ and $b_2 U_2$, which are presented in Table 5.14 for uniform refinement, and in Table 5.15 for the adaptive refinement, show that criterion (5.23) with $C_\oplus \geq 5$ is fulfilled on all the considered meshes.

The error plots presented in Figure 5.16 show the expected faster convergence on the adaptively refined mesh, even though we are only using tensor-product splines. In Figure 5.17, meshes and marked cells are shown for steps 2 and 6, again indicating that the error indicator correctly identifies the corner singularity.

Example 5.6: Advection-Dominated Advection-Diffusion-Equation

In our last example, we consider an advection-diffusion equation in a setting which results in sharp layers, and we test the ability of the error estimator to detect these

mesh-size	$I_\oplus$	b_1U_1	b_2U_2	$C_\oplus$
16×8	1.18	5.67e-02	1.71e-03	> 30
32×16	1.14	3.44e-02	8.98e-04	> 30
64×32	1.11	2.09e-02	4.72e-04	> 30
128×64	1.09	1.28e-02	2.49e-04	> 30
256×128	1.07	7.87e-03	1.32e-04	> 30
512×256	1.06	4.86e-03	7.01e-05	> 30
1024×512	1.05	3.01e-03	3.73e-05	> 30

Table 5.14: Efficiency index and components of the majorant in Example 5.5, uniform refinement.

mesh-size	$I_\oplus$	b_1U_1	b_2U_2	$C_\oplus$
16×8	1.18	5.67e-02	1.71e-03	> 30
22×11	1.18	2.68e-02	8.66e-04	> 30
30×16	1.17	1.37e-02	4.32e-04	> 30
39×23	1.16	7.22e-03	2.24e-04	> 30
55×37	1.16	3.52e-03	1.10e-04	> 30
87×60	1.16	1.75e-03	5.41e-05	> 30
133×101	1.15	9.25e-04	2.69e-05	> 30

Table 5.15: Efficiency index and components of the majorant in Example 5.5, adaptive refinement with $\psi = 0.9$.

layers. Note that this type of problem was not specified in Section 2.2.1, because only one example of this type is presented. For details, the reader is referred to the references given at the beginning of Section 2 and, in particular, [78]. The phyiscal domain in this example is the unit square $\Omega = (0,1)^2$, with $p = q = 2$, and the considered problem is formulated as follows.

Find a function $u \in C^2(\Omega) \cap C(\overline{\Omega})$, $u : \overline{\Omega} \to \mathbb{R}$, such that

$$\left.\begin{aligned} -\kappa\Delta u + b \cdot \nabla u &= 0 \quad \text{in } \Omega, \\ u &= g_0 \quad \text{on } \Gamma_0 = \partial\Omega, \end{aligned}\right\} \tag{5.29}$$

where $\kappa = 10^{-6}$ is the diffusion coefficient and $b = (\cos\frac{\pi}{3}, \sin\frac{\pi}{3})^T$ the (vector-valued) advection velocity. The Dirichlet boundary conditions are given as

$$g_0 = \begin{cases} 1, \text{ if } y = 0, \\ 0, \text{ else.} \end{cases}$$

The terms of the variational form (see Section 2.2.2) of (5.29) are given by

$$\begin{aligned} a(u,v) &= \int_\Omega \kappa\, \nabla u \cdot \nabla v + (b \cdot \nabla u)\, v \; dx, \\ \langle f, v\rangle &= \int_\Omega f\, v \; dx. \end{aligned}$$

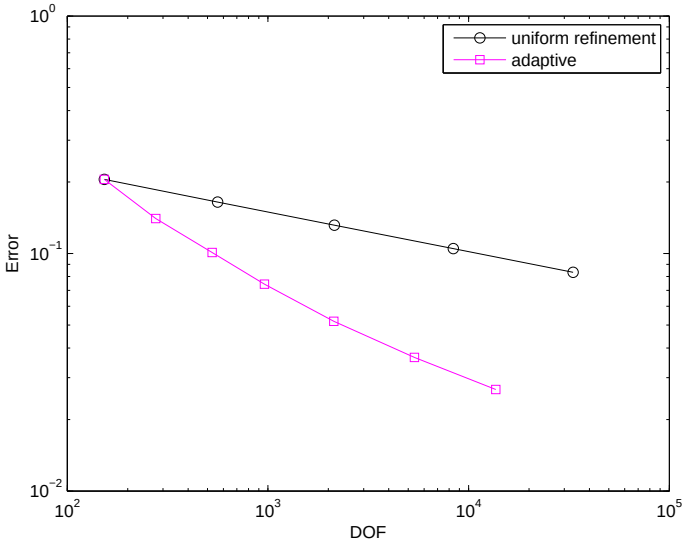
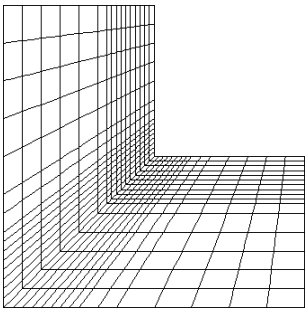
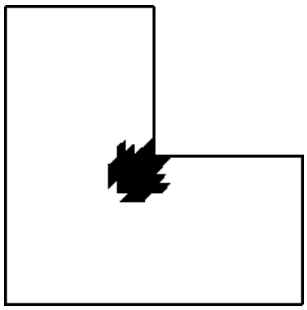


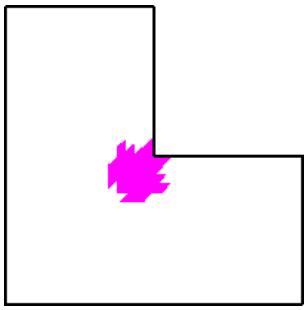
Figure 5.16: Error convergence, Example 5.5.



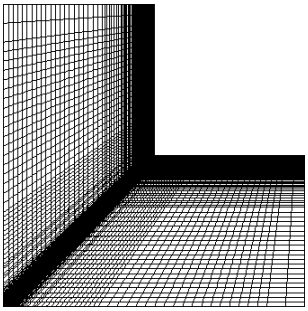
(a) Mesh after 2 refinements.



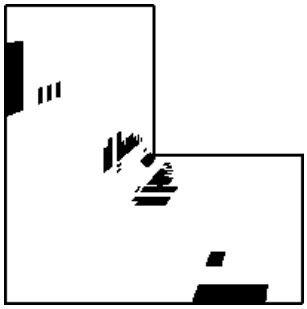
(b) Cells marked by exact error on mesh 2.



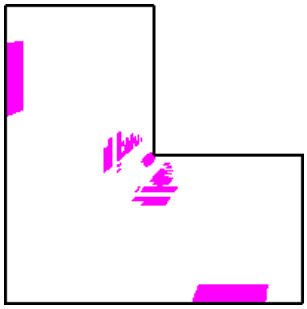
(c) Cells marked by estimator on mesh 2.



(d) Mesh after 6 refinements.



(e) Cells marked by exact error on mesh 6.



(f) Cells marked by estimator on mesh 6.

Figure 5.17: Meshes and marked cells in Example 5.5, $\psi = 0.9$.

The Peclet number Pe is defined by $Pe = L \cdot |b|/\kappa$, where L is the length of the domain. If $Pe > 1$, which is clearly the case here with $Pe \approx 10^6$, the diffusion is dominated by the advection (see, e.g., [78]). The dominating advection and the discontinuous boundary conditions result in the above-mentioned sharp layers. In Figure 5.18(a), the expected positions of the layers are indicated by dashed lines.

We use the standard streamline upwind Petrov-Galerkin (SUPG) scheme for stabilization. The stabilization parameter τ is set to $\tau(Q) = h_b(Q)/2|b|$, where $h_b(Q)$ is the diameter of the cell Q in direction of the flow b, and $|b|$ is the magnitude of the vector b.

For advection-diffusion problems, we have to adapt the majorant. Since the principle method is the same, we refer the reader to [84, Section 4.3.1] for a detailed discussion. In this special case, where the second-order term is given by $-\kappa\Delta u$ with $\kappa \ll |b|$, and with constant velocity vector b, the majorant $M^2_{\oplus,\mathrm{adv}}$ for the advection-diffusion problem is given by

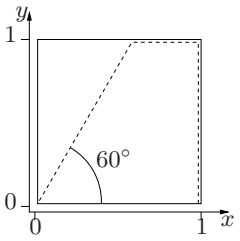
$$M^2_{\oplus,\mathrm{adv}} = (1+\beta)\|A\nabla u_h - y\|^2_A + (1+\tfrac{1}{\beta})C^2_\Omega \|\operatorname{div} y + f - b\cdot\nabla u_h\|^2.$$

mesh-size	b_1U_1	b_2U_2	$C_\oplus$
	Case 1		
16×16	1.98e-07	3.18e-10	$> 10^2$
64×64	6.45e-07	1.15e-09	$> 10^2$
256×256	2.28e-06	4.33e-09	$> 10^2$
	Case 2		
16×16	1.83e-06	9.66e-10	$> 10^3$
64×64	6.50e-06	3.65e-09	$> 10^3$
256×256	1.86e-05	1.24e-08	$> 10^3$
	Case 3		
16×16	3.24e-06	1.29e-09	$> 10^3$
64×64	2.07e-05	6.52e-09	$> 10^3$
256×256	6.86e-05	2.38e-08	$> 10^3$

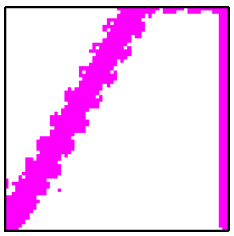
Table 5.16: Comparison of terms b_1U_1 and b_2U_2 in Example 5.6.

The magnitudes of b_1U_1 and b_2U_2 presented in Table 5.16 indicate that the criterion (5.23) with $C_\oplus \geq 5$ is fulfilled on all the considered meshes. The distribution of the marked cells presented in Figures 5.18(b) and 5.18(c) provides the visual indication that the expected layers are detected by the error estimate.

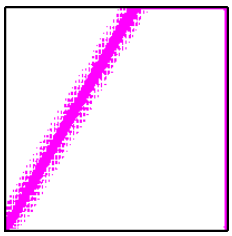
In Table 5.17, the timings are presented. Note that, unlike the previous examples, assembling and solving the system for the estimator is faster than for the original problem not only in Case 3, but also in Case 2. This is due to the SUPG stabilization which is costlier than computing the additional term $b\cdot\nabla u_h$ in the majorant $M^2_{\oplus,\mathrm{adv}}$.



(a) Expected positions of sharp layers.



(b) Marked cells with $\psi = 0.8$, mesh-size 64×64.



(c) Marked cells with $\psi = 0.9$, mesh-size 256×256.

Figure 5.18: Expected layers and marked cells in Example 5.6, Case 3.

mesh-size	#DOF		assembling-time			solving-time			sum		
	u_h	y_h	pde	est.	"$\frac{\text{est.}}{\text{pde}}$"	pde	est.	"$\frac{\text{est.}}{\text{pde}}$"	pde	est.	"$\frac{\text{est.}}{\text{pde}}$"
Case 1											
16×16	324	722	0.25	0.39	1.56	<0.01	0.01	6.38	0.25	0.40	1.59
64×64	4356	8978	3.25	5.32	1.63	0.03	0.26	8.64	3.28	5.58	1.70
256×256	66564	134162	51.22	94.15	1.84	0.85	8.84	10.35	52.07	102.99	1.98
Case 2											
16×16	324	288	0.21	0.14	0.67	<0.01	<0.01	0.50	0.21	0.14	0.67
64×64	4356	2592	3.26	2.10	0.64	0.03	0.06	2.01	3.29	2.16	0.66
256×256	66564	34848	50.83	35.58	0.70	0.85	2.30	2.70	51.68	37.87	0.73
Case 3											
16×16	324	200	0.26	0.10	0.39	<0.01	<0.01	0.58	0.26	0.10	0.40
64×64	4356	968	3.41	1.21	0.35	0.04	0.01	0.26	3.44	1.22	0.35
256×256	66564	9800	52.40	19.83	0.38	1.02	0.91	0.89	53.42	20.74	0.39

Table 5.17: Timings in Example 5.6.

5.6 Guaranteed lower bound of the error

As mentioned in the introduction and at the beginning of this chapter, functional-type a posteriori error estimates can be derived for upper as well as for lower bounds of the unknown true error. In this section, we briefly discuss a guaranteed lower bound which we will denote by $M_{\ominus}^2$ and refer to as *minorant*. Together with the majorant $M_{\oplus}^2$, we have a two-sided estimate of the true error in the form

$$M_{\ominus}^2 \;\leq\; \|\nabla u - \nabla u_h\|_A^2 \;\leq\; M_{\oplus}^2, \tag{5.30}$$

i.e., it is possible to provide a guaranteed and fully computable interval containing the true error.

5.6.1 Definition and computation

The following theorem, which provides a guaranteed lower bound of the unknown error can be found in, e.g., [83, 84].

Theorem 5.13. *Let u be the exact solution of the model problem (I) with homogeneous Dirichlet boundary conditions. Then, the following estimate holds.*

$$M_{\ominus}^2(w) \leq \|u - u_h\|_A^2, \tag{5.31}$$

where

$$M_{\ominus}^2(w) = \int_\Omega 2f\, w - 2A\nabla u_h \cdot \nabla w \, dx - \|\nabla w\|_A^2, \tag{5.32}$$

and where w is an arbitrary funtion in V_0.

The sharpness of the bound in Theorem 5.13 is guaranteed by the following theorem, which can be found in [84].

Theorem 5.14. *Let the sequence of spaces $\{W_j\}_{j=1}^\infty$ be limit dense in V_0, then*

$$\lim_{j\to\infty} \sup_{w_j \in W_j} M_{\ominus}^2 = \|u_h - u\|_A^2. \tag{5.33}$$

Proof. Let ε be arbitrarily small, but fixed. Let j_ε be the index such that, for all $k > j_\varepsilon$, there exists a w_k, such that $\|\nabla(u - u_h - w_k)\|_A < \varepsilon$. Then,

$$\begin{aligned} M_{\ominus}^2(w_k) &= -\|\nabla w_k\|_A^2 - 2\int_\Omega \Big(A\nabla u_h \cdot \nabla w_k - f\, w_k\Big)\, dx \\ &= -\|\nabla w_k\|_A^2 + 2\int_\Omega A(\nabla u - \nabla u_h)\cdot \nabla w_k \, dx \\ &= \|\nabla u - \nabla u_h\|_A^2 - \|\nabla w_k - (\nabla u - \nabla u_h)\|_A^2 \\ &= \|\nabla u - \nabla u_h\|_A^2 - \mathcal{O}(\varepsilon^2). \end{aligned}$$

Hence, $M_{\ominus}^2(w_k) \to \|\nabla u - \nabla u_h\|_A^2$, as $\varepsilon \to 0$. □

In contrast to the upper bound $M_{\oplus}^2(y, \beta)$, we have only one free parameter in the lower bound, namely the scalar-valued function w. In order to minimize $M_{\ominus}^2(w)$ with respect to w, we proceed analogously to Step 1 in Section 5.2. Let $M_{\ominus}^2(w)'$ denote the derivative of $M_{\ominus}^2(w)$ with respect to w. Setting $M_{\ominus}^2(w)' = 0$, we obtain

$$\int_\Omega \nabla w \cdot \nabla \widetilde{w}\, dx = \int_\Omega f\, \widetilde{w} - \nabla u_h \cdot \nabla \widetilde{w}\, dx. \tag{5.34}$$

We choose a finite-dimensional subspace W_h of V_0 and search for a solution w_h for (5.34) in W_h (note that there is no connection between the space $\widetilde{W}_h$ as used in Section 4.2.3 and the space W_h used here). With the same argument as in Remark 5.6, we have to choose W_h such that it has better approximation properties than V_h.

5.6.2 Numerical examples for the lower bound

In this section, we present lower bound computations for the examples discussed in Sections 5.1 and 5.5 (except the advection-diffusion-equation Example 5.6). Due to time-constraints, we shall not discuss a cost-efficient computation of the minorant in detail, and only present first results as a proof-of-concept. We consider a straightforward approach where W_h is obtained from V_0 by applying one step of p-refinement (see Section 3.1.1.4). In this setting, the size of the discretized problem for computing w_h is larger than the original problem for u_h. Since this set-up is costlier, the presented results are not computed on very fine meshes.

To measure the efficiency of the computed lower bounds, we define the efficiency index of the minorant analogously to (5.21) as follows.

$$I_{\ominus} = \frac{M_{\ominus}(w)}{\|\nabla u - \nabla u_h\|_A}. \tag{5.35}$$

The computed efficiency indices $I_{\ominus}$ are presented in Tables 5.18–5.24 for Examples 5.1–5.5, respectively, along with the previously presented efficiency indices of the majorant $I_{\oplus}$. The results clearly show that the obtained bounds are very sharp on the considered meshes.

mesh-size	$I_{\oplus}$			$I_{\ominus}$
	Case 1	Case 2	Case 3	
8×8	2.77	14.19	11.28	0.9924
16×16	1.71	8.49	36.43	0.9918
32×32	1.32	1.82	12.63	0.9966
64×64	1.16	1.16	1.17	0.9990
128×128	1.08	1.04	1.01	0.9997

Table 5.18: Efficiency indices of the majorant and the minorant in Example 5.1 (Sinus-function on unit square with $p = q = 2$, cf. Tables 5.3, 5.5, and 5.7).

mesh-size	$I_{\oplus}$			$I_{\ominus}$
	Case 1	Case 2	Case 3	
18×18	1.84	15.43	132.77	0.9976
34×34	1.40	6.04	148.41	0.9975
66×66	1.20	1.76	6.42	0.9990
130×130	1.10	1.16	1.13	0.9997

Table 5.19: Efficiency indices of the majorant and the minorant in Example 5.2 (Sinus-function on unit square with $p = q = 4$ and only C^1-continuity at $x = 0.5$ and $y = 0.5$, cf. Table 5.10).

mesh-size	$I_\oplus$			$I_\ominus$
	Case 1	Case 2	Case 3	
16×8	1.83	13.99	24.87	0.9922
32×16	1.29	4.17	56.02	0.9953
64×32	1.13	1.31	10.42	0.9985
128×64	1.07	1.06	1.11	0.9996

Table 5.20: Efficiency indices of the majorant and the minorant in Example 5.3.a (peak on quarter annulus, $\alpha = 20$, cf. Table 5.11).

mesh-size	$I_\oplus$			$I_\ominus$
	Case 1	Case 2	Case 3	
16×8	3.02	13.84	17.20	0.9928
32×16	1.92	16.76	76.95	0.9926
64×32	1.34	3.16	83.72	0.9967
128×64	1.16	1.25	4.19	0.9990

Table 5.21: Efficiency indices of the majorant and the minorant in Example 5.3.b (peak on quarter annulus, $\alpha = 50$, cf. Table 5.12).

mesh-size	$I_\oplus$	$I_\ominus$
16×16	3.77	0.9911
25×26	2.06	0.9947
38×44	1.69	0.9971
64×74	1.47	0.9983
92×136	2.82	0.9989
184×256	1.30	0.9985

Table 5.22: Efficiency indices of the majorant and the minorant in Example 5.4, adaptive refinement (two peaks on unit square, cf. Table 5.13).

mesh-size	$I_\oplus$	$I_\ominus$
16×8	1.18	0.9692
32×16	1.14	0.9809
64×32	1.11	0.9880
128×64	1.09	0.9925

Table 5.23: Efficiency indices of the majorant and the minorant in Example 5.5, uniform refinement (L-shaped domain, cf. Table 5.14).

mesh-size	$I_{\oplus}$	$I_{\ominus}$
16×8	1.18	0.9692
22×11	1.18	0.9729
30×16	1.18	0.9784
39×23	1.16	0.9829
55×37	1.16	0.9850
87×60	1.16	0.9896
133×101	1.15	0.9923

Table 5.24: Efficiency indices of the majorant and the minorant in Example 5.5, adaptive refinement (L-shaped domain, cf. Table 5.15).

Chapter 6

Summary and discussion

The main work presented in this thesis is divided into two main parts, the isogeometric tearing and interconnecting method, and functional-type a posteriori error estimation in IGA. These two parts will be discussed separately at first, before discussing possible combinations and extensions.

6.1 Isogeometric tearing and interconnecting method

The IETI-DP method presented and discussed in Chapter 4 (see also [57]) is a first step towards efficient numerical computations on isogeometric multi-patch geometries without the need of merging subdomains into one global mesh. By combining the concepts of IGA and FETI methods, one can profit from the advantages of both fields. On the one hand, the exact geometry representation is preserved, even in the case of complicated multi-patch domains (possibly including holes). The need for data transformation and possible consequent approximation errors in the geometry are thus eliminated. Furthermore, one can benefit from the good approximation properties of NURBS basis functions and isogeometric solvers which are currently being studied. On the other hand, well-developed techniques from FETI methods can be applied in IGA, such as solver and preconditioner design, and the parallelization of the subdomain solvers. Multi-patch geometries can be treated directly without the need for finding a global representation of the given mesh.

While the IETI approach closely follows established FETI methods, there are some principle differences. In the isogeometric setting, we assume that the domain is already given by a multi-patch geometry mapping, i.e., the domain is decomposed from the beginning. As a requirement, for the presented straightforward C^0-coupling (see Section 4.2.1), we have to assume that the given setting has fully matching interfaces (see Definition 4.1 and Assumption 4.2). This requirement is relaxed in the context of local refinement options introduced by the IETI method (see Section 4.3), where coupling of interfaces with hanging knots is discussed. However, it is still required that the initial configuration is fully matching. While fully matching interfaces allow a very simple identification of interfaces and associated DOF, and are thus very

convenient for implementation, this requirement can be restrictive in practical applications. For example, from the design point of view, in general, it may not be natural to require fully matching interfaces. Note that, in FETI methods, the starting point typically is a global mesh of the global domain. When this mesh is decomposed into subdomains, the fully matching property follows naturally.

For the treatment of floating subdomains, the dual-primal approach was followed for two reasons. Firstly, this approach is very general in the sense that this method can be applied to problems with different kernels. The only requirement is to have sufficient primal DOF per subdomain such that the local kernel on floating subdomains is fixed (in the classical FETI and the total FETI approach, additional unknowns are introduced to span the kernel, the formulation thus depends on the specific kernel and has to be adapted to the considered problem). Secondly, in the case of fully matching interfaces, the dual-primal approach is very natural due to the property of NURBS that, in two dimensions, only one basis function has value 1 at a subdomain vertex, while all other basis functions are zero there (see Remark 4.4). Furthermore, in the presented tensor product setting, it is very simple to identify the DOF at subdomain vertices, which makes the dual-primal approach even more attractive and simple to implement. It has to be mentioned, however, that these advantages result from the fully matching setting, and that they cannot necessarily be extended to more general settings in a straightforward manner. While a possible method of handling situations where subdomain vertices lie on edges of neighbouring domain was discussed in Section 4.3.2, the principle problem is not avoided in this straightforward approach.

The performance of the presented method and the discussed preconditioner is very satisfactory in the considered settings. This is illustrated by the numerical examples presented in Section 4.4. The extension to some more general settings, however, may not be straightforward due to the aspects discussed above. While the C^0-coupling across fully matching interfaces can be extended to three-dimensional problems in a straightforward manner, the required fully matching setting may be even more restrictive in three dimensions. Preconditioners from FETI-DP methods can also be applied to a three-dimensional IETI method. The solver design and the choice of primal DOF (including edge averages, which have been proven necessary in three dimensions [54, 95]) are more involved, in particular in the presence of hanging vertices.

6.2 Functional-type a posteriori error estimators in IGA

The results presented in Chapter 5 (see also [58]) show the great potential of functional-type a posteriori error estimators in IGA. The underlying theory is well-studied and the presented application in IGA is very promising. The combination of functional-type a posteriori error estimators and NURBS basis functions is motivated by several aspects, which we recollect here.

The presented method for computing the majorant relies only on the use of NURBS basis functions, for which we assume that an efficient implementation is

available. This is a reasonable assumption, since we are working in the isogeometric framework. The need for constructing a separate, complicated basis functions of $H(\Omega, \mathrm{div})$ is thus eliminated.

Special properties of NURBS basis functions make it possible to increase the polynomial degree and smoothness with adding only few additional DOF. Due to this property, it is possible to formulate the time-efficient approaches referred to as Case 2 and Case 3 in Sections 5.4.2 and 5.5. It is important to note that this cannot be obtained from the classical FEM discretizations based on C^0 basis functions (see Remarks 5.10 and 5.11).

The estimator for the upper bound of the error is fully computable, and provides a guaranteed and sharp bound for the true error, as well as an error indicator which can be used for adaptive refinement. The quality criterion presented in Proposition 5.8 has been derived from the theoretical studies on the sharpness of the majorant. The numerical tests presented in Section 5.5 indicate that this criterion can be used to assess the quality of the computed majorant (see Remark 5.9). This criterion involves only terms that have to be evaluated in the course of computing the majorant. Together with the minorant discussed in Section 5.6, the presented estimates provide not only qualitative, but *sharp two-sided quantitative bounds* for the unknown true error. Such bounds can be very important for the quality assurance in practical applications. This, of course, is a general feature of functional-type a posteriori error estimates and not specific to IGA. Nevertheless, we believe efficient computations of these estimates, as discussed in this thesis, and their applicability for any of the IGA approximations make them a valuable tool within IGA.

6.3 Subjects for further studies

Both the IETI-DP method presented in Chapter 4 (based on [57]) and the functional-type error estimators in IGA presented in Chapter 5 (based on [58]) mark starting points and open several subjects which need to be studied further.

Open issues in the IETI method, which are of particular interest, include the treatment of more general interfaces, including interfaces that are not necessarily geometrically conforming (i.e., that may have small gaps or overlaps), and the incorporation of fast iterative subdomain solvers, such as geometric multigrid solvers and solvers exploiting the tensor product structure, e.g., wavelet solvers.

Combining the isogeometric local refinement methods mentioned in Section 3.3.4 with the IETI method (see Remark 4.7) and with functional-type a posteriori error estimators is, in theory, straightforward. The actual performance and efficiency of the presented methods, however, are the subject of further studies.

Another important open issue is the efficient computation of the lower bound discussed in Section 5.6. While the presented straightforward approach is very costly, exploiting NURBS-specific properties (in a similar fashion as in Section 5.4.2) could lead to more cost- and time-efficient lower bound computations in IGA.

List of Figures

3.1 B-spline basis functions of degrees $p = 1$ to $p = 5$. 17
3.2 B-spline basis functions of degree $p = 3$. 18
3.3 Influence of a control point on the shape of a B-spline curve. 23
3.4 Illustration of the used notation. 24

4.1 Fully matching subdomains and their interface. 33
4.2 Illustration of fully redundant coupling and all floating setting. 35
4.3 Two options for refining a marked area. 46
4.4 Illustration of an interface with hanging knots. 47
4.5 Embedding subdomains into the original parameter domain. 48
4.6 Examples for hanging and not hanging subdomain vertices. 49
4.7 Subdomains refined by substructuring and positions of primal vertices. 49
4.8 Refinement levels of subdomains. 50
4.9 Case (A), bracket with rounded reentrant corner. 52
4.10 Case (B), bracket with sharp reentrant corner. 52
4.11 Condition numbers and (P)CG iterations for cases (A) and (B). 52
4.12 Case (C), bracket with sharp reentrant corner and local h-refinement. 53
4.13 Comparison of the energy norms of u_h in cases (B) and (C). 53
4.14 Bending of a cantilever, problem setting. 55
4.15 Bending of a cantilever, discussed cases. 56
4.16 Yeti's footprint with adaptive refinement. 58

5.1 Marked cells, exact error, $\psi = 0.8$, Example 5.1. 61
5.2 Marked cells, error indicator, $\psi = 0.8$, Example 5.1. 61
5.3 Convergence of exact error and the majorant (5.4) for Example 5.1. . 62
5.4 Cells marked with $\psi = 0.8$ in Example 5.1, $\widehat{Y}_h$ as in (5.25) 70
5.5 Marked cells, error indicator, $\psi = 0.8$, Example 5.1, Case 1. 72
5.6 Marked cells, error indicator, $\psi = 0.8$, Example 5.1, Case 2. 74
5.7 Marked cells, error indicator, $\psi = 0.8$, Example 5.1, Case 3. 75
5.8 Marked cells, exact error, $\psi = 0.8$, Example 5.2. 77
5.9 Marked cells, error indicator, $\psi = 0.8$, Example 5.2. 77
5.10 Exact solutions u on Ω, Example 5.3. 78
5.11 Marked cells, error indicator, $\psi = 0.8$, Example 5.3.a. 80

5.12 Marked cells, error indicator, $\psi = 0.8$, Example 5.3.b. 82
5.13 Exact solution, Example 5.4. 82
5.14 Meshes and marked cells, $\psi = 0.75$, Example 5.4. 83
5.15 Error convergence, Example 5.4. 84
5.16 Error convergence, Example 5.5. 86
5.17 Meshes and marked cells, $\psi = 0.9$, Example 5.5. 86
5.18 Expected layers and marked cells, Example 5.6. 88

List of Tables

5.1 Efficiency index, Example 5.1, $\widehat{Y}_h$ as in (5.25). 69
5.2 Number of DOF and timings in Example 5.1, $\widehat{Y}_h$ as in (5.25). 70
5.3 Efficiency index, Example 5.1, Case 1. 71
5.4 Number of DOF and timings in Example 5.1, Case 1. 72
5.5 Efficiency index, Example 5.1, Case 2. 73
5.6 Number of DOF and timings in Example 5.1, Case 2. 74
5.7 Efficiency index, Example 5.1, Case 3. 74
5.8 Number of DOF and timings in Example 5.1, Case 3. 75
5.9 Comparison of $I_{\oplus}$ for different numbers of interleaved iterations. . . . 76
5.10 Efficiency index, Example 5.2. 77
5.11 Efficiency index, Example 5.3.a. 79
5.12 Efficiency index, Example 5.3.b. 81
5.13 Efficiency index, Example 5.4, adaptive refinement. 81
5.14 Efficiency index, Example 5.5, uniform refinement. 85
5.15 Efficiency index, Example 5.5, adaptive refinement. 85
5.16 Terms b_1U_1 and b_2U_2, Example 5.6. 87
5.17 Timings in Example 5.6. 88
5.18 Efficiency indices of the minorant, Example 5.1. 90
5.19 Efficiency indices of the minorant, Example 5.2. 90
5.20 Efficiency indices of the minorant, Example 5.3.a. 91
5.21 Efficiency indices of the minorant, Example 5.3.b. 91
5.22 Efficiency indices of the minorant, Example 5.4, adaptive refinement. . 91
5.23 Efficiency indices of the minorant, Example 5.5, uniform refinement. . 91
5.24 Efficiency indices of the minorant, Example 5.5, adaptive refinement. . 92

Bibliography

[1] M. Ainsworth and J. Oden. A posteriori error estimation in finite element analysis. *Computer Methods in Applied Mechanics and Engineering*, 142(1-2):1–88, 1997.

[2] M. Ainsworth and J. Oden. *A Posteriori Error Estimation in Finite Element Analysis*. Pure and Applied Math. Wiley, 2000.

[3] F. Auricchio, L. Beirão da Veiga, A. Buffa, C. Lovadina, A. Reali, and G. Sangalli. A fully "locking-free" isogeometric approach for plane linear elasticity problems: A stream function formulation. *Computer Methods in Applied Mechanics and Engineering*, 197(1-4):160–172, 2007.

[4] F. Auricchio, L. Beirão da Veiga, T. Hughes, A. Reali, and G. Sangalli. Isogeometric collocation for elastostatics and explicit dynamics. *Computer Methods in Applied Mechanics and Engineering*, 249-252(0):2–14, 2012.

[5] F. Auricchio, L. Beirão da Veiga, C. Lovadina, and A. Reali. The importance of the exact satisfaction of the incompressibility constraint in nonlinear elasticity: mixed FEMs versus NURBS-based approximations. *Computer Methods in Applied Mechanics and Engineering*, 199(5-8):314–323, 2010.

[6] F. Auricchio, F. Calabrò, T. Hughes, A. Reali, and G. Sangalli. A simple algorithm for obtaining nearly optimal quadrature rules for NURBS-based isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 249-252(0):15–27, 2012.

[7] R. Bank and R. Smith. A posteriori error estimates based on hierarchical bases. *SIAM Journal on Numerical Analysis*, 30(4):921–935, 1993.

[8] Y. Bazilevs, L. Beirão da Veiga, J. Cottrell, T. Hughes, and G. Sangalli. Isogeometric analysis: approximation, stability and error estimates for h-refined meshes. *Mathematical Models and Methods in Applied Sciences*, 16(7):1031–1090, 2006.

[9] Y. Bazilevs, V. Calo, J. Cottrell, J. Evans, T. Hughes, S. Lipton, M. Scott, and T. Sederberg. Isogeometric analysis using T-splines. *Computer Methods in Applied Mechanics and Engineering*, 199(5-8):229–263, 2010.

[10] Y. Bazilevs, V. Calo, J. Cottrell, T. Hughes, A. Reali, and G. Scovazzi. Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows. *Computer Methods in Applied Mechanics and Engineering*, 197(1-4):173–201, 2007.

[11] Y. Bazilevs, V. Calo, T. Hughes, and Y. Zhang. Isogeometric fluid-structure interaction: theory, algorithms, and computations. *Computational Mechanics*, 43:3–37, 2008.

[12] Y. Bazilevs, V. Calo, Y. Zhang, and T. Hughes. Isogeometric fluid-structure interaction analysis with applications to arterial blood flow. *Computational Mechanics*, 38:310–322, 2006.

[13] L. Beirão da Veiga, A. Buffa, D. Cho, and G. Sangalli. IsoGeometric analysis using T-splines on two-patch geometries. *Computer Methods in Applied Mechanics and Engineering*, 200(21-22):1787–1803, 2011.

[14] L. Beirão da Veiga, A. Buffa, J. Rivas, and G. Sangalli. Some estimates for *h*-*p*-*k*-refinement in isogeometric analysis. *Numerische Mathematik*, 118:271–305, 2011.

[15] L. Beirão da Veiga, D. Cho, L. Pavarino, and S. Scacchi. BDDC preconditioners for isogeometric analysis. *Mathematical Models and Methods in Applied Sciences*, 23(06):1099–1142, 2013.

[16] L. Beirão da Veiga, D. Cho, L. Pavarino, and S. Scacchi. Isogeometric schwarz preconditioners for linear elasticity systems. *Computer Methods in Applied Mechanics and Engineering*, 253(0):439–454, 2013.

[17] D. Benson, Y. Bazilevs, M. Hsu, and T. Hughes. Isogeometric shell analysis: The ReissnerMindlin shell. *Computer Methods in Applied Mechanics and Engineering*, 199(5-8):276–289, 2010.

[18] D. Braess. *Finite Elemente*. Springer, 3rd edition, 2003.

[19] S. Brenner and L. Ridgeway Scott. *The Mathematical Theory of Finite Element Methods*. Springer, 3rd edition, 2008.

[20] S. C. Brenner and L.-Y. Sung. BDDC and FETI-DP without matrices or vectors. *Computer Methods in Applied Mechanics and Engineering*, 196(8):1429–1435, 2007.

[21] A. Buffa, C. de Falco, and G. Sangalli. IsoGeometric Analysis: Stable elements for the 2D Stokes equation. *International Journal for Numerical Methods in Fluids*, 65(11-12):1407–1422, 2011.

[22] A. Buffa, H. Harbrecht, A. Kunoth, and G. Sangalli. BPX-preconditioning for isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 265(0):63–70, 2013.

[23] A. Buffa, J. Rivas, G. Sangalli, and R. Vázquez. Isogeometric discrete differential forms in three dimensions. *SIAM Journal on Numerical Analysis*, 49(2):818–844, 2011.

[24] A. Buffa, G. Sangalli, and R. Vázquez. Isogeometric analysis in electromagnetics: B-splines approximation. *Computer Methods in Applied Mechanics and Engineering*, 199(17-20):1143–1152, 2010.

[25] C. Carstensen. Some remarks on the history and future of averaging techniques in a posteriori finite element error analysis. *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift fr Angewandte Mathematik und Mechanik*, 84(1):3–21, 2004.

[26] P. Ciarlet. *The Finite Element Method for Elliptic Problems*. Studies in Mathematics and its Applications. Elsevier Science, 1978.

[27] J. Cottrell, T. Hughes, and Y. Bazilevs. *Isogeometric Analysis: Toward Integration of CAD and FEA*. Wiley, Chichester, 2009.

[28] J. Cottrell, T. Hughes, and A. Reali. Studies of refinement and continuity in isogeometric structural analysis. *Computer Methods in Applied Mechanics and Engineering*, 196:4160–4183, 2007.

[29] J. Cottrell, A. Reali, Y. Bazilevs, and T. Hughes. Isogeometric analysis of structural vibrations. *Computer Methods in Applied Mechanics and Engineering*, 195:5257–5296, 2006.

[30] T. Davis. *Direct Methods for Sparse Linear Systems*. SIAM, Philadelphia, 2006.

[31] J. Deng, F. Chen, X. Li, C. Hu, W. Tong, Z. Yang, and Y. Feng. Polynomial splines over hierarchical T-meshes. *Graphical Models*, 70:76–86, 2008.

[32] T. Dokken, T. Lyche, and K. F. Pettersen. Polynomial splines over locally refined box-partitions. *Computer Aided Geometric Design*, 30(3):331–356, 2013.

[33] M. Dörfel, B. Jüttler, and B. Simeon. Adaptive isogeometric analysis by local h-refinement with T-splines. *Computer Methods in Applied Mechanics and Engineering*, 199(5-8):264–275, 2010.

[34] W. Dörfler. A convergent adaptive algorithm for Poisson's equation. *SIAM Journal on Numerical Analysis*, 33(3):1106–1124, jun 1996.

[35] Z. Dostál, D. Horák, and R. Kučera. Total FETI – An easier implementable variant of the FETI method for numerical solution of elliptic PDE. *Communications in Numerical Methods in Engineering*, 12:1155–1162, 2006.

[36] T. Elguedj, Y. Bazilevs, V. Calo, and T. Hughes. $\bar{B}$ and $\bar{F}$ projection methods for nearly incompressible linear and nonlinear elasticity and plasticity using

higher-order NURBS elements. *Computer Methods in Applied Mechanics and Engineering*, 197(33-40):2732–2762, 2008.

[37] K. Eriksson, D. Estep, P. Hansbo, and C. Johnson. Introduction to Adaptive Methods for Differential Equations. *Acta Numerica*, 4:105–158, 1 1995.

[38] J. Evans and T. Hughes. Isogeometric Divergence-conforming B-splines for the Darcy-Stokes-Brinkman equations. *Mathematical Models and Methods in Applied Sciences*, 23(04):671–741, 2013.

[39] C. Farhat, M. Lesoinne, P. LeTallec, K. Pierson, and D. Rixen. FETI-DP: a dual-primal unified FETI-method – part I: A faster alternative to the two-level FETI method. *International Journal for Numerical Methods in Engineering*, 50:1523–1544, 2001.

[40] C. Farhat, J. Mandel, and F. Roux. Optimal convergence properties of the FETI domain decomposition method. *Computer Methods in Applied Mechanics and Engineering*, 115:265–385, 1994.

[41] C. Farhat and F. Roux. A method of finite element tearing and interconnecting and its parallel solution algorithm. *International Journal for Numerical Methods in Engineering*, 32:1205–1227, 1991.

[42] K. Gahalaut, J. Kraus, and S. Tomar. Multigrid methods for isogeometric discretization. *Computer Methods in Applied Mechanics and Engineering*, 253(1):413–425, 2013.

[43] K. Gahalaut, S. Tomar, and J. Kraus. Algebraic multilevel preconditioning in isogeometric analysis: Construction and numerical studies. *Computer Methods in Applied Mechanics and Engineering*, 266(0):40–56, 2013.

[44] C. Giannelli, B. Jüttler, and H. Speleers. THB-splines: The truncated basis for hierarchical splines. *Computer Aided Geometric Design*, 29(7):485–498, 2012.

[45] T. Hughes, J. Cottrell, and Y. Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Computer Methods in Applied Mechanics and Engineering*, 194(39-41):4135–4195, 2005.

[46] T. Hughes, A. Reali, and G. Sangalli. Duality and unified analysis of discrete approximations in structural dynamics and wave propagation: Comparison of p-method finite elements with k-method NURBS. *Computer Methods in Applied Mechanics and Engineering*, 197(49-50):4104–4124, 2008.

[47] T. Hughes, A. Reali, and G. Sangalli. Efficient quadrature for NURBS-based isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 199(5-8):301–313, 2010.

[48] K. Jankauskas. Time-efficient NURBS curve evaluation algorithms. In *16th International Conference on Information and Software Technologies (IT 2010, Kaunas, Lithuania)*, 2010.

[49] K. A. Johannessen. An adaptive isogeometric finite element analysis. Dissertation, Norwegian University of Science and Technology, June 2009.

[50] H. Kim and C.-O. Lee. A preconditioner for the FETI-DP formulation with mortar methods in two dimensions. *SIAM Journal on Numerical Analysis*, 42(5):2159–2175, 2005.

[51] A. Klawonn, L. Pavarino, and O. Rheinbach. Spectral element FETI-DP and BDDC preconditioners with multi-element subdomains. *Computer Methods in Applied Mechanics and Engineering*, 198:511–523, 2008.

[52] A. Klawonn and O. Widlund. A domain decomposition method with Lagrange multipliers and inexact solvers for linear elasticity. *SIAM Journal on Scientific Computing*, 22(4):1199–1219, 2000.

[53] A. Klawonn and O. Widlund. FETI and Neumann-Neumann iterative substructuring methods: Connections and new results. *Communications on Pure and Applied Mathematics*, 54(1):57–90, 2001.

[54] A. Klawonn, O. Widlund, and M. Dryja. Dual-primal FETI methods for three-dimensional elliptic problems with heterogeneous coefficients. *SIAM Journal on Numerical Analysis*, 40:159–179, 2002.

[55] A. Klawonn and O. B. Widlund. Dual-primal FETI methods for linear elasticity. *Communications on Pure and Applied Mathematics*, 59(11):1523–1572, 2006.

[56] S. Kleiss, B. Jüttler, and W. Zulehner. Enhancing isogeometric analysis by a finite element-based local refinement strategy. *Computer Methods in Applied Mechanics and Engineering*, 213-216(0):168–182, 2012.

[57] S. Kleiss, C. Pechstein, B. Jüttler, and S. Tomar. IETI – Isogeometric tearing and interconnecting. *Computer Methods in Applied Mechanics and Engineering*, 247-248(0):201–215, 2012.

[58] S. Kleiss and S. Tomar. Guaranteed and sharp a posteriori error estimates in isogeometric analysis. Technical Report RICAM Report 2013-06, Radon Institute for Computational and Applied Mathematics (RICAM) Austrian Academy of Sciences (ÖAW). Also available at arXiv:1304.7712.

[59] J. Kraus and S. Tomar. Algebraic multilevel iteration method for lowest order Raviart-Thomas space and applications. *International Journal for Numerical Methods in Engineering*, 86(10):1175–1196, 2011.

[60] U. Langer and C. Pechstein. Coupled finite and boundary element tearing and interconnecting solvers for nonlinear potential problems. *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik*, 86(12):915–931, 2006.

[61] U. Langer and O. Steinbach. Boundary element tearing and interconnecting methods. *Computing*, 71(3):205–228, 2003.

[62] R. Lazarov, S. Repin, and S. Tomar. Functional a posteriori error estimates for discontinuous Galerkin approximations of elliptic problems. *Numerical Methods for Partial Differential Equations*, 25(4):952–971, 2009.

[63] X. Li and M. Scott. On the nesting behavior of T-splines. Technical Report ICES Report 2011-13.

[64] X. Li, J. Zheng, T. Sederberg, T. Hughes, and M. Scott. On linear independence of T-spline blending functions. *Computer Aided Geometric Design*, 29(1):63–76, 2012.

[65] J. Mandel, C. R. Dohrmann, and R. Tezaur. An algebraic theory for primal and dual substructuring methods by constraints. *Applied Numerical Mathematics*, 54(2):167–193, 2005.

[66] J. Mandel and R. Tezaur. Convergence of a substructuring method with Lagrange multipliers. *Numer. Math.*, 73:473–487, 1996.

[67] J. Mandel and R. Tezaur. On the convergence of a dual-primal substructuring method. *Numerische Mathematik*, 88:543–558, 2001.

[68] N. D. Manh, A. Evgrafov, A. R. Gersborg, and J. Gravesen. Isogeometric shape optimization of vibrating membranes. *Computer Methods in Applied Mechanics and Engineering*, 200(1316):1343–1353, 2011.

[69] T. Martin and E. Cohen. Volumetric parameterization of complex objects by respecting multiple materials. *Computers & Graphics*, 34(3):187–197, 2010.

[70] N. Nguyen-Thanh, H. Nguyen-Xuan, S. Bordas, and T. Rabczuk. Isogeometric analysis using polynomial splines over hierarchical T-meshes for two-dimensional elastic solids. *Computer Methods in Applied Mechanics and Engineering*, 200(21-22):1892–1908, 2011.

[71] P. Nielsen, A. Gersborg, J. Gravesen, and N. Pedersen. Discretizations in isogeometric analysis of Navier-Stokes flow. *Computer Methods in Applied Mechanics and Engineering*, 200(45-46):3242–3253, 2011.

[72] G. Of and O. Steinbach. The all-floating boundary element tearing and interconnecting method. *Journal of Numerical Mathematics*, 17(4), 2009.

[73] C. Pechstein. Boundary element tearing and interconnecting methods in unbounded domains. *Applied Numerical Mathematics*, 59(11):2824–2842, 2009.

[74] C. Pechstein. *Finite and Boundary Element Tearing and Interconnecting Solvers for Multiscale Problems*, volume 90 of *Lecture Notes in Computational Science and Engineering*. Springer-Verlag, Berlin Heidelberg, 2013.

[75] C. Pechstein and R. Scheichl. Analysis of FETI methods for multiscale PDEs. *Numerische Mathematik*, 111(2):293–333, 2008.

[76] L. Piegl and W. Tiller. *The NURBS book*. Springer Berlin Heidelberg, 2 edition, 1997.

[77] E. Pilgerstorfer. Construction and analysis of volume parameterizations for isogeometric analysis. Dissertation, Johannes Kepler University Linz, Austria, June 2013.

[78] O. Pironneau. *Finite element methods for fluids*. Wiley, 1989.

[79] S. Repin. A posteriori error estimation for nonlinear variational problems by duality theory. *Zapiski Nauchnych Seminarov POMI*, 243:201–214, 1997.

[80] S. Repin. A posteriori error estimates for approximate solutions to variational problems with strongly convex functionals. *Journal of Mathematical Sciences*, 97:4311–4328, 1999.

[81] S. Repin. A posteriori error estimation for nonlinear variational problems by duality theory. *Journal of Mathematical Sciences*, 99:927–935, 2000. See Also: *Zapiski Nauchnych Seminarov POMI*, 243:201–214, 1997.

[82] S. Repin. A posteriori error estimation for variational problems with uniformly convex functionals. *Mathematics of Computation*, 69(230):481–500, 2000.

[83] S. Repin. Two-sided estimates of deviation from exact solutions of uniformly elliptic equations. In N. Uraltseva, editor, *Proceedings of the St. Petersburg Mathematical Society, Volume IX, American Mathematical Society Translations: Series 2*, volume 209, pages 143–171, 2003.

[84] S. Repin. *A Posteriori Estimates for Partial Differential Equations*. Walter de Gruyter, Berlin, Germany, 2008.

[85] D. Rixen and C. Farhat. Preconditioning the FETI method for problems with intra- and inter-subdomain coefficient jumps. In P. E. Bjørstad, M. Espedal, and D. E. Keyes, editors, *Proceedings of 9th International Conference on Domain Decomposition*, pages 472–479, 1998.

[86] Y. Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, Philadelphia, 2003.

[87] M. Scott, M. Borden, C. Verhoosel, T. Sederberg, and T. Hughes. Isogeometric finite element data structures based on Bézier extraction of T-splines. *International Journal for Numerical Methods in Engineering*, 88:126–156, 2011.

[88] M. Scott, X. Li, T. Sederberg, and T. Hughes. Local refinement of analysis-suitable T-splines. *Computer Methods in Applied Mechanics and Engineering*, 213-216:206–222, 2012.

[89] T. Sederberg, J. Zheng, A. Bakenov, and A. Nasri. T-splines and T-NURCCs. *ACM Transactions on Graphics*, 22(3):161–172, 2003.

[90] T. W. Sederberg, D. L. Cardon, G. T. Finnigan, N. S. North, J. Zheng, and T. Lyche. T-spline simplification and local refinement. *ACM Transactions on Graphics*, 23(3):276–283, aug 2004.

[91] T. Takacs. Regularity and approximation power of isogeometric discretizations for parametrizations with singularities. Dissertation, Johannes Kepler University Linz, Austria, July 2013.

[92] T. Takacs and B. Jüttler. Existence of stiffness matrix integrals for singularly parameterized domains in isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 200:3568–3582, 2011.

[93] T. Takacs and B. Jüttler. Regularity properties of singular parameterizations in isogeometric analysis. *Graphical Models*, 74(6):361–372, 2012.

[94] S. Timoshenko and J. Goodier. *Theory of Elasticity, 3rd edition*. McGraw-Hill Book Company, New York, 1970.

[95] A. Toselli and O. Widlund. *Domain Decomposition Methods - Algorithms and Theory*. Springer-Verlag, Berlin, 2005.

[96] U. Trottenberg, C. Oosterlee, and A. Schüller. *Multigrid*. Academic Press, 2001.

[97] A.-V. Vuong, C. Giannelli, B. Jüttler, and B. Simeon. A hierarchical approach to adaptive local refinement in isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 200(49-52):3554–3567, 2011.

[98] P. Wang, J. Xu, J. Deng, and F. Chen. Adaptive isogeometric analysis using rational PHT-splines. *Computer-Aided Design*, 43(11):1438–1448, 2011.

[99] G. Xu, B. Mourrain, R. Duvigneau, and A. Galligo. Parameterization of computational domain in isogeometric analysis: Methods and comparison. *Computer Methods in Applied Mechanics and Engineering*, 200(23-24):2021–2031, 2011.

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Dissertation selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe.

Die vorliegende Dissertation ist mit dem elektronisch übermittelten Textdokument identisch.

Linz, am 20. Dez. 2013, _____

Stefan K. Kleiss

Curriculum Vitae

Affiliation and address

Dipl.-Ing. Stefan K. Kleiss Bakk. techn.

Radon Institute for Computational and Applied Mathematics (RICAM),
Austrian Academy of Sciences (ÖAW)

c/o Johannes Kepler University Linz (JKU)
Altenbergerstr. 69
4040 Linz, Austria

e-mail: `stefan.kleiss@ricam.oeaw.ac.at`
URL: `http://www.ricam.oeaw.ac.at`

Personal data

Date of birth:	05. Jan. 1982
Place of birth:	Linz, Austria
Nationality:	Austria

Education

since 2010	PhD study "Technische Wissenschaften" at JKU, supported by Austrian Science Fund (FWF), project P21516-N18, and European Union, 7th Framework Programme, project 218536 "EXCITING"
2006 - 2010	Master study "Industriemathematik" at JKU
2001 - 2005	Bachelor study "Technische Mathematik" at JKU
2000	High School Diploma (Matura)

Publications

S. Kleiss and S. Tomar. Guaranteed and sharp a posteriori error estimates in isogeometric analysis. RICAM Report 2013-06, Radon Institute for Computational and Applied Mathematics (RICAM) Austrian Academy of Sciences (ÖAW). Also available at arXiv:1304.7712.

S. Kleiss, C. Pechstein, B. Jüttler, and S. Tomar. IETI - Isogeometric tearing and interconnecting. Computer Methods in Applied Mechanics and Engineering, 247-248(0):201–215, 2012.

S. Kleiss, B. Jüttler, and W. Zulehner. Enhancing isogeometric analysis by a finite element-based local refinement strategy. Computer Methods in Applied Mechanics and Engineering, 213-216(0):168–182, 2012.

S. Kleiss. Enhancing Isogeometric Analysis by a Finite Element-Based Local Refinement Strategy. Master's thesis, Institute of Computational Mathematics, JKU, 2010.

S. Kleiss. Computing Real Inflection Points of Cubic Algebraic Curves. Bachelor's thesis, Institute of Applied Geometry, JKU, 2004.

S. Kleiss. Beschreibung zur Implementierung eines vereinfachten Modells des Kreditrisikomanagementsystems CreditMetrics von J.P. Morgan in *Mathematica*. Bachelor's thesis, Institut für Finanzmathematik, JKU, 2003.

Activities

11. - 13. Oct. 2008	EXCITING Kick-off meeting. Strobl, Austria.
11. - 13. Oct. 2010	EXCITING consortium meeting. Strobl, Austria.
14. - 16. Jun. 2011	EXCITING consortium meeting. Nice, France.
20. - 24. Jun. 2011	Conference on Geometry - Theory and Application. Vorau, Austria.
12. - 17. Feb. 2012	NTAG, New Trends in Applied Geometry 2012. Villa Cagnola, Italy.
12. - 16. Mar. 2012	IGAA, Conference on Isogeometric Analysis and Applications, and EXCITING, final consortium meeting. Linz, Austria.
10. - 11. May 2012	8^{th} Austrian Numerical Analysis Day. Vienna, Austria.

25. - 29. Jun. 2012	DD21, The Twenty First International Conference on Domain Decomposition Methods. Rennes, France.
16. - 18. Jul. 2012	Geometry + Simulation Kick-off meeting. Vorau, Austria.
08. - 12. Jul. 2013	AANMPDE-6-13, 6th Workshop on Analysis and Advanced Numerical Methods for Partial Differential Equations (for Junior Scientists). Strobl, Austria.
26. - 30. Aug. 2013	ENUMATH 2013, European Conference on Numerical Mathematics and Advanced Applications. Lausanne, Switzerland.