JOHANNES KEPLER
UNIVERSITÄT LINZ

Eingereicht von
Dipl.-Ing.
Christoph Hofer,
BSc.

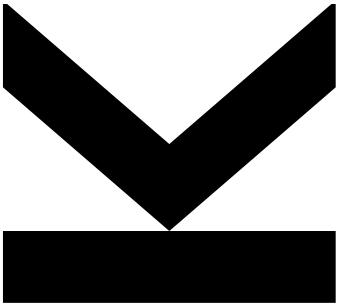
Angefertigt am
Doktoratskolleg
"Computational
Mathematics"

Betreuer und
Erstbeurteiler
O. Univ.-Prof.
Dipl.-Ing. Dr.
Ulrich Langer

Zweitbeurteiler
Prof. Dr.
Giancarlo Sangalli

April 2018

Fast Multipatch Isogeometric Analysis Solvers



Dissertation

zur Erlangung des akademischen Grades

Doktor der technischen Wissenschaften

im Doktoratsstudium

Technische Wissenschaften

JOHANNES KEPLER
UNIVERSITÄT LINZ
Altenbergerstraße 69
4040 Linz, Österreich
www.jku.at
DVR 0093696

Abstract

This thesis is devoted to the generalization of the Dual-Primal Finite Element Tearing and Interconnecting (FETI-DP) method to linear algebraic systems arising from the Isogemetric Analysis (IgA) of linear elliptic boundary value problems, like stationary diffusion or heat conduction problems. This IgA version of the FETI-DP method is called Dual-Primal Isogeometric Tearing and Interconnect (IETI-DP) method. The FETI-DP method is well established as parallel solver for large-scale systems of finite element equations, especially, in the case of heterogeneous coefficients having jumps across subdomain interfaces. These methods belong to the class of non-overlapping domain decomposition methods.

In practise, a complicated domain can often not be represented by a single patch, instead a collection of patches is used to represent the computational domain, called multi-patch domains. Regarding the solver, it is a natural idea to use this already available decomposition into patches directly for the construction of a robust and parallel solver. We investigate the cases where the IgA spaces are continuous or even discontinuous across the patch interfaces, but smooth within the patches. In the latter case, a stable formulation is obtained by means of discontinuous Galerkin (dG) techniques. Such formulations are important for various reasons, e.g, if the IgA spaces are not matching across patch interfaces (different mesh-sizes, different spline degrees) or if the patches are not matching (gap and overlapping regions).

Using ideas from dG-FETI-DP methods, we extend IETI-DP methods in such a way that they can efficiently solve multi-patch dG-IgA schemes. This thesis also provides a theoretical foundation of IETI-DP methods. We prove the quasi-optimal dependence of the convergence behaviour on the mesh-size for both version. Moreover, the numerical experiments indicate robustness of these methods with respect to jumps in the coefficient and a weak dependence on the spline degree. All algorithms are implemented in the C++ library G+Smo.

Finally, this thesis investigates space-time methods for linear parabolic initial-boundary value problems, like instationary diffusion or heat conduction problems. The focus is again on efficient solution techniques. The aim is the development of solvers which are on the one hand robust with respect to certain parameters and on the other hand parallelizeable in space and time. We develop special block smoothers that lead to robust and efficient time-parallel multigrid solvers. The parallelization in space is again achieved by means of IETI-DP methods.

Zusammenfassung

Diese Arbeit ist der Verallgemeinerung der so genannten "Dual-Primal Finite Element Tearing and Interconnecting" (FETI-DP) Methode auf lineare Gleichungssysteme gewidmet, die aus der isogeometrischen Analyse (IgA) von linearen elliptischen Randwertproblemen, wie etwa stationäre Diffusions- oder Wärmeleitproblemen, entstehen. Wir nennen diese Verallgemeinerung "Dual-Primal Isogeometric Tearing and Interconnecting" (IETI-DP) Methode. FETI-DP ist eine weit verbreitete parallele Methode zum Lösen von großen linearen Gleichungssystemen, die aus der Diskretisierung mittels Finiten Elementen entstehen. Insbesondere lässt sich diese Methode parallelisieren und ist besonders geeignet um Probleme mit springenden Koeffizienten zu lösen.

Komplexe Geometrien werden als Vereinigung von "einfachen" Gebieten, genannt "Patches", repräsentiert und als "Multi-patch" Geometrien bezeichnet. Diese bereits bestehende Zerlegung des Rechengebiets in Teilgebiete (Patches) bietet einen natürlichen Zugang zur Konstruktion von effizienten und parallelisierbaren Lösern. Wir betrachten die Verwendung von IgA Räumen, welche im Inneren der Patches glatte Ansatzfunktionen besitzen, aber nur stetig oder sogar unstetig entlang der Patch-Schnittstellen sind. Im Falle von unstetigen IgA Räumen verwenden wir sogenannte unstetige Galerkin (dG) Techniken um eine stabile Formulierung zu erhalten. Diese ermöglichen es uns verschiedene Spline-Grade oder Gitterfeinheiten auf benachbarten Patches, sowie kleine Löcher bzw. Überlappungen zwischen benachbarten Patches, zu betrachten.

Basierend auf der dG-FETI-DP Methode erweitern wir die IETI-DP Methode dahingehend, sodass sie auch für Multi-patch dG-IgA Schemen anwendbar ist. In dieser Arbeit beweisen wir die quasi-optimale Abhängigkeit des Konvergenzverhalten von der Gitterfeinheit, sowohl für die stetige als auch die unstetige Version der IETI-DP Methode. In numerischen Experimenten beobachten wir des Weiteren Robustheit bezüglich Sprünge in den Koeffizienten und eine schwache Abhängigkeit von dem Spline-Grad. Die vorgestellten Algorithmen sind in der C++ Bibliothek G+Smo implementiert.

Zuletzt untersuchen wir sogenannte Raum-Zeit Methoden für lineare parabolische Anfangs-Randwertprobleme, wie etwa instationäre Diffusions- oder Wärmeleitungsprobleme. Wie in den anderen Kapitel liegt der Fokus erneut auf effizienten Lösungsstrategien. Ziel ist es Löser zu entwickeln, die einerseits robust gegenüber verschiedensten Parametern sind und andererseits sich bezüglich Ort und Zeit parallelisieren lassen. Dazu entwickeln wir Glätter, die zu robusten zeit-parallelen Multigrid Methoden führen. Die zusätzliche Parallelisierung im Ort wird mittels IETI-DP erreicht.

Acknowledgments

First of all, I would like to express my deepest gratitude to my supervisor Prof. Ulrich Langer, for his supervision, guidance and support during my Ph.D. studies. I am very grateful for his scientific and non-scientific advices, and for the time he spent reviewing our reports, papers and this thesis. Moreover, I would like to thank him for giving me the latitude to follow also my own interest in the frame of the project, and for encouraging and enabling me to visit many international and national conferences. At the same time I want to thank Prof. Giancarlo Sangalli for co-refereeing this thesis.

The results in this thesis were achieved in the NFN project "Geometry + Simulation", sub-project 3 (NFN S117-03), and in the Doctoral Program "Computational Mathematics" (W1214), project DK4, which were funded by the Austrian Science Fund (FWF). Both programs provided an excellent scientific environment. This support is gratefully acknowledged. I want to thank the former speaker of the DK, Prof. Peter Paule, and the current speaker, Veronika Pillwein, for their valuable support during my studies.

I want to thank the Radon Institute for Computational and Applied Mathematics (RICAM), Austrian Academy of Sciences (OAW), at Linz for the permission to use the distributed memory cluster RADON1. I am very grateful for all the fruitful discussions and helpful hints from my colleges of the Institute of Computational Mathematics, Institute of Applied Geometry, the RICAM groups "Computational Methods for PDEs" and "Geometry in Simulations". Especially, I want to thank Angelos Mantzaflaris for providing assistance for using and contributing to G+Smo and Ioannis Toulopoulos for his extraordinary support and advice during my studies.

Finally, I want to express my sincere thanks to my parents, who supported me during all my studies, and to my girlfriend Katharina, for her faith in me, her understanding during difficult periods and her love.

Christoph Hofer
Linz, April 2018

Contents

Chapter 1

Introduction

The simulation of physical processes, like, heat transfer, diffusion, liquid or gas flow and electro-magnetism, is of great importance in natural sciences, engineering and many other fields. Mathematically, such processes are modelled via partial differential equations (PDEs) or systems of PDEs together with appropriate boundary and/or initial conditions. Usually, such problems cannot be solved in an analytical way, especially not when considering real-world applications. Instead, we are looking for an approximate solution, described by a finite number of parameters, called degrees of freedom (dofs). In many practical applications, the number of dofs is around $10^6 - 10^9$.

In order to construct such an approximation to a PDE, several methods have been developed. Finite difference methods (FDM), finite volume methods (FVM), finite element methods (FEM) or boundary element methods (BEM) are now the most prominent representatives of such discretization methods for PDEs. In real-world applications, the computational domain Ω is modelled by means of computer aided design (CAD), and can be quite complex, e.g, electrical or combustion motors, cars, planes, ships. However, the methods mentioned above require certain approximations of Ω. For instance, when using FEM, we have to perform a triangulation of the domain. Hence, the boundary $\partial\Omega$ of Ω may not be represented exactly. This introduces a systematic error in the calculation. In order to overcome this issue, the concept of *Isogeometric Analysis* (IgA) was introduced.

Usually, the mentioned methods yield a system of linear algebraic equations describing the approximate solution. The challenge is to develop algorithms for computing this solution, which are optimal with respect to the number of dofs, i.e., the time for the computation grows linearly with the number of dofs. Moreover, due to the steadily growing number of available processors in computing centres and also in Desktop PCs, it is very important to have an algorithm which can be parallelized.

The focus of this thesis is on the development of algorithms, which can solve the system of linear algebraic equations arising from Isogeometric Analysis of elliptic and parabolic diffusion problems in almost optimal complexity and in parallel.

State of the Art

Isogeometric Analysis

Isogeometric Analysis is a methodology for solving PDEs numerically. IgA was introduced by Hughes, Cottrell and Bazilevs in [107], and has become a very active field of research. We refer to [14] for the first results on the numerical analysis of IgA, the monograph [40] and the recent survey article [16] for a comprehensive presentation of the mathematical analysis of variational isogeometric methods.

The main idea is to use the basis functions, which are used for the representation of the geometry in CAD models, also for the approximation of the solution of the PDE describing the physical phenomenon which we are going to simulate. The typical choice for such basis functions are B-Splines or Non-Uniform Rational B-Splines (NURBS). One advantage of IgA over the more traditional FEM is the fact that there is no need for decomposing the computational domain into triangles or tetrahedra. Hence, one gets rid of this geometrical error source, at least, in the class of computational domains that can be exactly represented by a CAD system. Typically, a complex geometry cannot be described by a single domain, called patch, but is composed of several patches leading to a so-called multi-patch domain. A further benefit is that it is much easier to build up $C^l, l \geq 1$, conforming basis functions in IgA than in the finite element (FE) framework. This is especially important when dealing with higher-order PDEs, like the biharmonic equation, which appears, e.g., in plate bending theory. For such problems, at least a C^1 basis in a conforming variational setting is required, which is difficult to achieve in the FEM. However, difficulties arise when smooth bases have to be constructed for multi-patch domains. We refer to [114], [33] and [112] for an investigation of higher-order PDEs using smooth isogeometric bases on multi-patch domains.

B-Splines or NURBS provide the same approximation order as the corresponding FE of degree p, see, e.g., [14], [208], [35] and [24]. However, if we consider splines of maximal smoothness, i.e., C^{p-1}, we need much less dofs to get the same order of convergence than for FE. Nevertheless, in general, B-Splines have a much larger support, and are more costly to evaluate than FE basis functions. This introduces additional challenges for the matrix assembly, see, e.g., [36], [163], [101], [164] and references therein for different strategies to overcome this issue, see also [193] for a matrix-free algorithm using weighted quadrature. Similar discrete trace and inverse inequalities can be shown as for FE, see e.g., [14], [59], [148] and [210]. The explicit dependence on the degree p has been shown in [139] by means of symbolic tools. In this thesis, we restrict ourselves to so-called tensor-product B-Splines, where multivariate splines are constructed by univariate splines using the tensor-product. Such splines do not allow local for refinement, and are not suited for adaptive algorithms. There have been invented several splines which allow for local refinement, like truncated hierarchical B-Splines (THB-Splines), see, e.g., [73], [74], [72], T-Splines, see, e.g., [199], [15], [25],

and local refined (LR) Splines, see, e.g., [46], [31].

The computational domain in a CAD system is usually represented via its boundary and material interfaces by using several surface patches. In order to perform IgA, a volumetric parametrization of the domain by several volumetric patches is required. This procedure is called segmentation, and is not a trivial task. It is a common problem that the resulting volumetric patches are not fully matching at the interfaces, and there can occur small gap and overlapping regions, called *segmentation crimes*. We refer to [110], [173], [179], [174], [106], [230] and [231] for a description of segmentation algorithms and [179], [173] and [61] for a discussion of occurring non-matching interfaces. The presence of such gap or overlapping regions introduces additional errors in the discrete solution. Problems on multi-patch domains with gap and overlapping regions have been considered in [97], [99], [98] and [100], where a discontinuous Galerkin (dG) formulation is developed, using Taylor expansions to approximate the unknown numerical fluxes at the non-matching interfaces. We mention that several techniques have been recently investigated for coupling non-matching (or non-conforming) subdomain parametrizations in some weak sense. In [189] and [175], Nitsche's method has been applied to enforce weak coupling conditions along trimmed B-Spline patches. In [4], the most common techniques for weakly imposing the continuity of the solution on the interfaces have been applied and tested on non-linear elasticity problems. The numerical tests have been performed on non-matching grid parametrizations. Furthermore, mortar methods have been applied in IgA utilizing different B-Spline degrees for the Lagrange multiplier in [32].

Beside segmentation crimes, the dG-IgA formulation is used to couple patches where the IgA spaces of adjacent patches are not matching, e.g., different B-Spline degrees and non-matching meshes. It is important to note that the dG method is only used at the patch interfaces, which does not increase the computational complexity as the classical dG method. The dG method has been applied successfully to different PDEs in FE, see, e.g., [188], [150] and [41]. It allows a great flexibility, e.g., having different polynomial degrees on each element and hanging nodes in the triangulation. In contrast to the continuous Galerkin (cG) method, the continuity of the discrete solution is enforced in a weak sense. For a detailed discussion of the dG method for FE, we refer, e.g., to [188], [41], [50] [6], [5], [177]. An analysis of the dG-IgA formulation with extensions to low regularity solutions can be found in [148], [144], [236]. Lastly we want to mention the application of dG-IgA to PDEs on surfaces in [147], [235].

Fast Iterative Solvers

As already pointed out above, after discretization of the PDE, we have to solve a system $\mathbf{Ku} = \mathbf{f}$ of linear algebraic equations with a large number N of dofs. Direct solvers, like Gaussian elimination or Cholesky factorization, do not result in algorithms with optimal complexity, i.e., $O(N)$, but $O(N^\beta)$ with some $\beta > 1$, see, e.g., [71] and

[238]. Alternatively, we can use iterative methods, which compute an approximation $\mathbf{u}_k$ to the solution $\mathbf{u}$ based on previous iterations $\mathbf{u}_{k-1}, \dots, \mathbf{u}_{k-s}$. So-called Krylov subspace methods fit into this framework, see, e.g., [190]. Prominent representatives are the Conjugate Gradient (CG) method [88], the Minimal Residual (MinRes) method [178] and the Generalized Minimal Residual (GMRes) method [191]. The number of iterations required to solve the system up to a given tolerance τ often (not always) depends on the condition number $\kappa(\mathbf{K})$ of the system matrix $\mathbf{K}$. In order to obtain an optimal method, the condition number should be independent of certain "bad" parameters, like mesh-size h, spline degree p or material coefficients. A common strategy is to use a so-called *preconditioner* $\mathbf{C}^{-1}$, i.e., instead of solving $\mathbf{K}\mathbf{u} = \mathbf{f}$, we solve $\mathbf{C}^{-1}\mathbf{K}\mathbf{u} = \mathbf{C}^{-1}\mathbf{f}$. The challenge is to design preconditioners in such a way that the condition number of $\kappa(\mathbf{C}^{-1}\mathbf{K})$ has the desired properties.

One class of such preconditioners are so-called *Domain Decomposition* (DD) methods. The key idea is to decompose the computational domain into smaller subdomains, e.g., via a mesh partitioning, and use local (with respect to the subdomains) solutions in an iterative process in order to construct the global solution. The problems on the subdomains should be small enough such that, e.g., direct solver can be applied efficiently. The class of DD methods is divided into overlapping and non-overlapping methods (also called substructuring methods), where the subdomains are overlapping and non-overlapping, respectively. Typically, these methods result in solvers with optimal $O(N)$ or quasi-optimal $O((1+\log(N))^{\gamma}N)$ complexity. The first DD method was developed by Schwarz in [198] and extended in [202], [166], [7], [154], [155] and [156], nowadays called alternating Schwarz method and is a representative of an overlapping DD method. For applications of various overlapping DD methods in IgA, we refer to [18], [20], [26], [21] and [39]. For a historical overview, we refer to [69], and for a comprehensive discussion of DD for FEM we refer to the monographs [201] [186], [205], [215], [165], [183], [138] as well as the proceedings of the international Domain Decomposition conferences[1].

A prominent method, which is often successfully applied in practice is the Finite Element Tearing and Interconnecting (FETI) method, see [64], [63], [160], [133] and [134], and its dual-primal variant (FETI-DP), see [62], [161], [136], [135], [75] and [183]. First Lagrange multipliers are introduced to couple the independent local subdomain problems and then the original variables are eliminated from the system, giving a problem posed for the Lagrange multipliers. One has to be careful since the local problems may not be uniquely solvable, e.g, in case of the Poisson equation, the local Neumann problem on subdomains that do not touch the Dirichlet boundary are only uniquely solvable up to constants. The dual-primal version circumvents this problem by introducing additional primal variables. The final system for the Lagrange multipliers can efficiently be solved by means of the CG method and an appropriate preconditioner. By simple back-substitution, one obtains the original solution from the Lagrange multipliers. For both FETI and FETI-DP methods, it is shown that the

[1] http://www.ddm.org/Proceedings.php

condition number of the preconditioned system is bounded by $O((1+\log(H/h))^2)$, see, e.g, [134], [136], [183] and [215]. For a variant where the Dirichlet boundary conditions are not eliminated from the local matrices, but incorporated via Lagrange multipliers, called total FETI, we refer to [48]. The FETI-DP method is related to the Balancing Domain Decomposition by Constraints (BDDC) method, that was introduced in [42]. The BDDC method is nothing but a preconditioner for the Schur complement system with respect to the interface dofs. For an analysis, we refer to [158], [42] and [44]. It is shown in [159] and [30] that the FETI-DP and BDDC method have the same spectrum up to a different number of zeros and ones. Finally, FETI-DP and BDDC methods have been considered and discussed for spectral finite elements, e.g., in [180], [181], [132], [125], [182] and [38].

A crucial point in FETI-DP is the choice of the primal variables, which eliminate the local kernels and build up also the coarse problem. Common choices are vertex values, edge or face averages. For example, just using vertex values for the three-dimensional Poisson equation gives a suboptimal condition number bound $O(H/h(1+\log(H/h))^2)$, see [215]. Recently, extensions were developed which choose the primal variables adaptively, based on local eigenvalue problems. For an overview, we refer the reader to the survey articles [184] and [128], see also [117], [126], [127], [118] and [37]. In [44] a more powerful preconditioner using the called deluxe scaling was introduced, which allows for more varying material parameters, see [45], [227], [53] and [55]. Numerical test in [22] and [23] for the case of IgA indicate also a reduced dependence of the condition number on the smoothness of B-Splines across interfaces. For the combination of dG and FETI-DP or BDDC, we refer to [51], [52], [57] and [54]. The extension of the dG-FETI-DP method to IgA and its analysis will be the main topic of Chapter 4.

The FETI-DP and BDDC methods have been successfully adapted to the IgA framework. In this work, we investigate the adaption of FETI-DP for IgA, called dual-primal IsogEometric Tearing and Interconnecting (IETI-DP), which was first introduced in [137]. The extension of BDDC to IgA as well as its analysis can be found in [19]. There, the authors investigate the case of having a decomposition obtained by subdividing a single patch into multiple patches, and their analysis covers the case of NURBS which have $C^l, l \geq 1$, continuity across the patch interfaces. Such a situation leads to more general interfaces, called fat interfaces, which add additional computational cost, especially, for the coarse problem. As already mentioned, a proof of the condition number bound for the BDDC method already implies a bound for the IETI-DP method. The missing proof of having different mappings for each patch, coupled with C^0 continuity, will be given in Chapter 3. Impressive extensions of the method developed in [19] have been presented in [22] and [23], where the authors combine BDDC for IgA with deluxe scaling and an adaptive selection of primal variables to eliminate the bottleneck of having a too large coarse space. As already mentioned, the deluxe scaling additionally reduces the large iteration numbers arising from highly smooth splines at the interfaces.

As already outlined above, it is important that the solver has a good parallel efficiency. Since DD methods are usually build up from independent local problems and a single coarse problem, they provide a quite natural framework for parallelization, where a processor is assigned to one or more subdomains. Therefore, the only communication appears if interface values have to be communicated to the neighbours or if information from the coarse problems has to be distributed to all processors. For numerical experiments, where the method scales up to several ten- or hundred-thousands of cores, we refer, e.g., to [131], [187], [129], [86], [120] and [123]. For a general introduction to parallelization of iterative methods, we refer, e.g., to [49] and [238]. In the FE case, the subdomains are usually defined by a distribution of the mesh elements, e.g., by the mesh partitioning software METIS and PARMETIS, see [115]. However, in IgA, the given patches are chosen to be the subdomains of the DD method. This may cause load imbalances for the parallelization. Furthermore, the size of the local problems might be quite large. To overcome this issue, a splitting of large patches into subpatches has to be performed.

So far, we have considered the solution of local problems using direct solvers. However, they become large if the discretization is refined. In this case, inexact solvers for the local sub-problems, as introduced in [130], could be superior to direct solvers. Indeed, Multigrid methods, see, e.g., [84], [220], are often used as inexact solver. For further research on inexact solver used in FETI-DP, especially, for the inexact solution of the coarse problem, we refer to [122], [120], [130], [131] and [132] and for the related BDDC method to [217], [218], [43], [152], [232] and [233]. Hybrid versions combining FETI and FETI-DP have been investigated, e.g., in [131] and [187]. In Chapter 6, we investigate inexact versions for IETI-DP methods. On the one hand, we investigate the use of p-robust Multigrid methods, see e.g., [103], [102], [204], [203], [209] and [47], and, on the other side, Fast Diagonalization methods, see the original works [157] and [13], and the recent applications to IgA in [192], [168] and [211].

Finally, we want to highlight applications of FETI-DP methods to non-linear problems in [119], [122], [120] and [124]. Non-linear problems are often solved by means of Newton methods, where the inner linearized problem is solved by fast iterative methods. In the mentioned works, the authors develop a FETI-DP method, which can be directly applied to the non-linear equation.

Time-Parallel Multigrid Methods for Space-Time Problems

The philosophy of space-time methods is to treat the time t as just another variable, say x_{d+1}, where $x_1, \dots, x_d$ are the d-dimensional spatial variables. Therefore, the computational domain Q is considered to be a subset of $\mathbb{R}^{d+1}$. In case of moving spatial domains Ω_t, $t \in [0, T]$, the moving boundary or interface is fixed in the space-time domain and can be taken into account, when constructing the mesh, see, e.g., [212] and [224]. However, in this work, we will restrict ourselves to non-moving domains,

which lead to a tensor-product structure of the space-time elements. Additionally, using a space-time mesh allows for local refinement and adaptive strategies to resolve special features of the solution, like corner or edge singularities, or high oscillatory parts. A posteriori error estimates that can be used for space-time adaptivity are derived in [145] and [58]

Space-time finite element methods for parabolic and hyperbolic PDEs have a long history and go back to the 80s and 90s of the last century, see, e.g., [108], [109], [8] [9] and [85]. Due to the availability of massively-parallel computers with more than hundred-thousands of cores, these methods came into the focus of current the research, see [197], [172], [222], [223], [2], [11], [170], [167], [221], [206], [3], [200], [12], [149], [216], [95] and [171] for recent papers on space-time methods for parabolic problems. A nice historical overview on space-time methods can be found in [67].

We will focus on space-time methods, which use space-time slabs and investigate Multigrid methods for solving the corresponding huge linear system of algebraic equations. Such methods were first introduced in [83] and extended, e.g., in [104], [105], [226] and [68]. Such methods allow for an additional parallelization in time direction, leading to methods which are parallel in space and time. Other methods, which allow for a parallelization in time-direction, are the *parareal* method, see, e.g., [153] and [70], and multiple shooting methods, see, e.g., [116] and [176]. Furthermore, for the application of BDDC methods to construct parallel solvers see [10].

On this work

This thesis investigates fast iterative solvers based on DD to solve system of linear algebraic equations arising from IgA of elliptic and parabolic diffusion problems. To be more precise, we consider the IETI-DP method, being the adaption of the FETI-DP method to IgA, applied to continuous and discontinuous Galerkin IgA. In case of parabolic problems, we develop time-parallel Multigrid methods with robust smoothers, which can efficiently be implemented in parallel, e.g., by means of IETI-DP methods.

Main Achievements

Analysis of the IETI-DP method for Multi-Patch cG Formulations and Inexact Variants The condition number analysis of the IETI-DP method is extended to the case of two-dimensional multi-patch domains, where each patch has its own geometrical mapping. We consider the case of having only vertex values as primal variables and globally constant diffusion coefficient. The proof is based on the results established in [19]. Moreover, we propose variants where the patch-local sparse direct

solvers are replaced by inexact solvers, which are robust with respect to the mesh-size h and spline degree p.

Development and Analysis of IETI-DP for dG Formulations We use ideas from [52] and [57] to extend the IETI-DP method to variational formulations posed on two- and three-dimensional domains, where the dG method is used at patch interfaces. Moreover, the condition number analysis of the method is performed for the case a two-dimensional domain, having only vertex values as primal variables and constant diffusion coefficient. The proof is based on the results established in [19]. The algorithm is also successfully applied to domains with segmentation crimes, i.e., small gap and overlapping regions at the interfaces.

Parallelization of the cG-IETI-DP and dG-IETI-DP Methods The cG-IETI-DP and dG-IETI-DP methods are well suited for parallelization. They are parallelized by means of the message passing interface (MPI). We present weak and strong scaling results up to 1024 cores for both methods on two- and three-dimensional domains and different B-Spline degrees.

Robust and Parallel Smoother for a Time-Parallel Multigrid Method The smoother used in the time-parallel Multigrid method, introduced in [68], requires the approximate solution of the space-time slab problems. Utilizing the tensor-product structure, a decomposition into a series of spatial problems is performed. We propose and analyze robust block preconditioners for the obtained spatial sub-problems. The construction of the preconditioners follows the ideas in [237]. They allow for additional parallelization in the spatial dimensions by standard preconditioners for elliptic problems.

Outline

The thesis is organized as follows:

- Chapter 2 introduces basic notations and definitions used in the subsequent chapters. This chapter deals with recalling definitions and some important properties regarding Sobolev spaces and variational equations. We introduce the concept of Isogeometric Analysis, and conclude the chapter with the formulation of our discrete problems in a continuous and discontinuous Galerkin setting.
- Chapter 3 deals with the IETI-DP method in a continuous Galerkin setting, where the patches are coupled by C^0 continuity. The presentation of the method follows the lines in [183]. We proof a quasi-optimal condition number bound with respect to the mesh-size h of the preconditioned IETI-DP operator, i.e.,

$\kappa \leq C(1 + \ln(H/h))^2$, where H is the patch diameter, for both the coefficient and stiffness scaling. We conclude the chapter with numerical experiments for two- and three-dimensional domains.

- Chapter 4 has a similar structure as Chapter 3, hence, after the formal definition of the method, we provide the analysis of the condition number of the preconditioned dG-IETI-DP operator and present numerical examples, validating the theory. In the last section, we investigate segmentations with non-matching interfaces, i.e., having gap and overlapping regions. Our contribution to this section is the construction and implementation of IETI-DP methods on such domains.

- Chapter 5 presents the parallelization of the cG-IETI-DP and dG-IETI-DP methods by means of MPI. We investigate their weak and strong scaling behaviour on two- and three-dimensional domains. We study the influence of different smoothness of the B-Spline basis in the interior of the patches on the scalability of the methods. In order to fit the number of patches to the number of available processors, we base our parallelization strategy on a splitting of patches via increasing the multiplicity of knots at the desired interfaces, i.e., reducing the smoothness there to C^0.

- Chapter 6 investigates the use of inexact solvers for the patch-local problems, namely a p-robust Multigrid method and the Fast Diagonalization method, both being robust in the spline degree p and mesh-size h. Throughout the section, we only consider the continuous Galerkin case. In IgA, these methods are usually applied to discretizations on a single patch. The combination with IETI-DP is one way to extend these methods to multi-patch domains. Finally, we present numerical results for two- and three-dimensional domains.

- Chapter 7 formulates the space-time variational formulation for parabolic initial-boundary value problems in non-moving domains. We decompose the space-time cylinder into time-slabs and introduce dG terms to couple the different time-slabs. We shortly discuss efficient assembling strategies utilizing the tensor-product structure of the patch-local matrices. Furthermore, we introduce the time-parallel Multigrid method developed in [68]. The main part is the construction of robust smoothers, which can be parallelized in space. We perform a decomposition of the slab-local space-time problem into a series of spatial problems, for which we propose and analyze robust block-diagonal preconditioners. We investigate parallelization in space, by applying IETI-DP methods to the non-symmetric space-time problem, posed on the time-slabs.

- Chapter 8 draws a short conclusion, and presents possible extensions and further work to the discussed topics.

Parts of this thesis have already been published by the author and corresponding co-authors in peer-reviewed journals and proceedings as follows:

- Parts of Chapter 3 are published in

  [94] C. Hofer and U. Langer. Dual-primal isogeometric tearing and interconnecting methods. In B. Chetverushkin, W. Fitzgibbon, Y. Kuznetsov, P. Neittaanmäki, O. Pironneau, and J. Periaux, editors, *Contributions to Partial Differential Equations and Applications*, volume 47 of *Computational Methods in Applied Sciences*. Springer International Publishing, Berlin, Heidelberg, New York, 2019.

- Parts of Chapter 4 are published in

  [93] C. Hofer and U. Langer. Dual-primal isogeometric tearing and interconnecting solvers for multipatch dG-IgA equations. *Computer Methods in Applied Mechanics and Engineering*, 316:2 – 21, 2017.

  [91] C. Hofer. Analysis of discontinuous Galerkin dual-primal isogeometric tearing and interconnecting methods. *Mathematical Models & Methods in Applied Sciences*, 28(1):131–158, 2018.

  [92] C. Hofer and U. Langer. Dual-primal isogeometric tearing and interconnecting solvers for multipatch continuous and discontinuous Galerkin IgA equations. *PAMM*, 16(1):747–748, 2016.

  [97] C. Hofer, U. Langer, and I. Toulopoulos. Discontinuous Galerkin Isogeometric Analysis of Elliptic Diffusion Problems on Segmentations with Gaps. *SIAM Journal on Scientific Computing*, 38:A3430 – A3460, 2016.

  [99] C. Hofer and I. Toulopoulos. Discontinuous Galerkin Isogeometric Analysis of elliptic problems on segmentations with non-matching interfaces. *Computers & Mathematics with Applications*, 72(7):1811 – 1827, 2016.

- Parts of Chapter 5 are published in

  [90] C. Hofer. Parallelization of continuous and discontinuous Galerkin dual-primal isogeometric tearing and interconnecting methods. *Computers & Mathematics with Applications*, 74(7):1607 – 1625, 2017.

- Parts of Chapter 6 are published in

  [96] C. Hofer, U. Langer, and S. Takacs. Inexact Dual-Primal Isogeometric Tearing and Interconnecting Methods. In *Domain decomposition methods in science and engineering XXIV*. Springer, Berlin, Heidelberg, 2018. accepted.

Chapter 2

Preliminaries

In this chapter we introduce basic notation and definitions used in the following chapters. We start recalling the definition and some important properties of Sobolev spaces. Then we state the inhomogeneous Poisson equation as our model problem. We proceed with a detailed description of Isogeometric Analysis, and conclude the chapter with the formulation of our discrete problems in a continuous and discontinuous Galerkin setting.

2.1 Model Problem

We start with the most basic definitions for formulating and analyzing variational equations. After the introduction of Sobolev spaces, we shortly comment on trace operators and recall two important inequalities for showing existence and uniqueness, the Friedrichs and Poincaré inequalities. Finally, we derive the variational formulation of our model problem and shortly discuss its well-posedness. For more details on Sobolev spaces as well as related inequalities and properties, we refer the readers to the monographs [1] and [60].

2.1.1 Function Spaces

Let $\Omega \subset \mathbb{R}^d$ be a bounded Lipschitz domain with $d = 1, 2, 3$. We introduce the Lesbesque spaces $L^q(\Omega)$ of all functions $u : \Omega \to \mathbb{R}$, such that $\|u\|_{L^q(\Omega)} < \infty$, where for $q \in [1, \infty)$

$$\|u\|^q_{L^q(\Omega)} := \int_\Omega |u|^q \, dx,$$

and

$$\|u\|_{L^\infty(\Omega)} := \inf\{C \geq 0 : |u(x)| \leq C \, \forall x \in \Omega \, a.e.\}.$$

These spaces are complete and therefore Banach spaces. In this work we only consider the special case $q = 2$, which gives a Hilbert space, equipped with the inner product

$$(u, v)_{L^2(\Omega)} := \int_\Omega uv \, dx.$$

Based on these spaces, we can define spaces including weak derivatives of functions, so-called Sobolev spaces. Let $L^1_{loc}(\Omega)$ be the space of functions which are in $L^1(K)$ for every compact subset K of Ω. If $f \in L^1_{loc}(\Omega)$ and satisfies the identity

$$\int_\Omega \frac{\partial f}{\partial x_i} \phi \, dx = -\int_\Omega f \frac{\partial \phi}{\partial x_i} dx \quad \forall \phi \in C_0^\infty(\Omega),$$

then f possesses the *weak derivative* $\frac{\partial f}{\partial x_i}$. This concept can be generalized to higher-order derivatives. Let $\alpha = (\alpha_1, \ldots, \alpha_d) \in \mathbb{N}_0^d$ be a multi-index and $|\alpha| = \sum_{i=1}^d \alpha_i$. We write

$$D^\alpha f = \frac{\partial_1^\alpha}{\partial x_1^{\alpha_1}} \cdots \frac{\partial_d^\alpha}{\partial x_d^{\alpha_d}} f$$

for the α-th derivative of a function f. If $f \in L^1_{loc}(\Omega)$ satisfies the identity

$$\int_\Omega D^\alpha f \phi \, dx = (-1)^{|\alpha|} \int_\Omega f D^\alpha \phi \, dx \quad \forall \phi \in C_0^\infty(\Omega),$$

then f possesses the higher-order weak derivative $D^\alpha f$. Let $l \in \mathbb{N}_0$, the Sobolev space $W^{l,q}$ is given by

$$W^{l,q}(\Omega) := \{v \in L^q(\Omega) : D^\alpha v \in L^q(\Omega) \quad \text{for all multi-indices } \alpha, |\alpha| \leq l\},$$

i.e., all functions in $L^q(\Omega)$, which possess the weak derivatives up to $|\alpha| \leq l$ in $L^q(\Omega)$. This space is equipped with the norm and semi-norm

$$\|v\|^q_{W^{l,q}(\Omega)} = \sum_{|\alpha| \leq l} \|D^\alpha v\|^q_{L^q(\Omega)} \text{ and } |v|^q_{W^{l,q}(\Omega)} = \sum_{|\alpha| = l} \|D^\alpha v\|^q_{L^q(\Omega)},$$

respectively. The beforehand introduced definitions can be generalized to the case $q = \infty$, see, e.g., [60]. The Sobolev spaces $W^{l,q}$ are again Banach spaces. As for the Lebesque spaces, the special case $q = 2$ leads to a Hilbert space with the inner product

$$(u, v)_{W^{l,2}(\Omega)} := \sum_{|\alpha| \leq l} (D^\alpha u, D^\alpha v)_{L^2(\Omega)},$$

and in the following we will denote this space by $H^l(\Omega)$.

The concept of Sobolev spaces can be generalized to spaces having a real Sobolev index, i.e., $H^s(\Omega)$ with $s \in \mathbb{R}^+, s \geq 0$. We split the index s in its integer part $\lfloor s \rfloor$ and its real part $\sigma \in (0,1)$ such that $s = \lfloor s \rfloor + \sigma$. The norm of $H^s(\Omega)$ can be defined as

$$\|u\|^2_{H^s(\Omega)} := \|u\|^2_{H^{\lfloor s \rfloor}(\Omega)} + \sum_{|\alpha| = \lfloor s \rfloor} \int_{\Omega \times \Omega} \frac{|D^\alpha u(x) - D^\alpha u(y)|^2}{|x-y|^{d+2\sigma}} d(x,y).$$

An alternative characterization of Sobolev spaces in case of Lipschitz domains Ω is as closure of C^∞ functions in the $\|\cdot\|$ norm, i.e.,

$$H^k(\Omega) = \overline{C^\infty(\overline{\Omega})}^{\|\cdot\|_{H^k(\Omega)}} \quad \forall k \geq 0.$$

Moreover, we introduce

$$H^k_0(\Omega) = \overline{C^\infty_0(\overline{\Omega})}^{\|\cdot\|_{H^k(\Omega)}} \subset H^k(\Omega) \quad \forall k \geq 0.$$

We have that $H^0(\Omega) = H^0_0(\Omega) = L^2(\Omega)$, and we will see in the following paragraphs that $H^1_0(\Omega)$ consists of functions which vanish on $\partial\Omega$. Furthermore, we can define Sobolev spaces with negative index $H^{-s}(\Omega)$ as the dual of $H^s_0(\Omega)$, being a subspace of the space of distributions.

In general, functions in the Sobolev spaces $H^s(\Omega)$ do not have well defined point values on $\partial\Omega$, but one can define a trace operator. If Ω is a Lipschitz domain, then the *trace operator*

$$\gamma_0 : C^\infty(\overline{\Omega}) \to C^\infty(\partial\Omega) \quad \gamma_0 u := u|_{\partial\Omega}$$

has a unique extension as linear bounded operator $H^1(\Omega) \to H^{1/2}(\partial\Omega)$, i.e., there exists a $C_T > 0$ such that

$$\|\gamma_0 u\|_{H^{1/2}(\partial\Omega)} \leq C_T \|u\|^2_{H^1(\Omega)} \quad \forall u \in H^1(\partial\Omega),$$

see [183] and references therein. In the following, we will often use $u|_{\partial\Omega}$ for the trace of u, i.e., $\gamma_0 u$. One can further characterize $H^1_0(\Omega)$ as

$$H^1_0(\Omega) = \{u \in H^1(\Omega) : \gamma_0 u = 0\}.$$

This concept can be extended to the spaces $H^k(\Omega)$, $k > 1$, and we can characterize the space $H^k_0(\Omega)$ of functions in $H^k(\Omega)$ where all normal-derivatives up to order $k-1$ vanish on $\partial\Omega$. The normal derivative of a function v is denoted by $\frac{\partial v}{\partial n}$.

We conclude this section with two important inequalities, Friedrichs' inequality and Poincaré's inequality. For the proof we refer, e.g., to [183] and [29].

Lemma 2.1. *Let Γ_D be a subset of $\partial\Omega$ with $|\Gamma_D| > 0$. Then, for any function $u \in H^1(\Omega)$ with $u|_{\Gamma_D} = 0$, we have the Friedrich inequality*

$$\|u\|_{L^2(\Omega)} \leq C_F |u|_{H^1(\Omega)},$$

where the constant C_F depends only on Ω and Γ_D.

Lemma 2.2. *For any function $u \in H^1(\Omega)$ with zero mean, i.e. $\int_\Omega u\,dx = 0$, we have the Poincaré inequality*

$$\|u\|_{L^2(\Omega)} \leq C_P |u|_{H^1(\Omega)},$$

where the constant C_P depends only on Ω.

Alternatively, Lemma 2.2 can be also formulated in the following ways

$$\|u - \overline{u}^\Omega\|_{L^2(\Omega)} \leq C_P |u|_{H^1(\Omega)} \quad \forall u \in H^1(\Omega),$$

where $\overline{u}^\Omega := |\Omega|^{-1} \int_\Omega u\,dx$, and

$$\|u\|_{L^2(\Omega)} \leq \left(C_P^2 |u|^2_{H^1(\Omega)} + |\Omega|^{-1} \left(\int_\Omega u\,dx \right)^2 \right)^{1/2} \quad \forall u \in H^1(\Omega).$$

2.1.2 Variational Formulation

In this thesis, we consider the following second-order elliptic boundary value problem in a bounded Lipschitz domain $\Omega \subset \mathbb{R}, d \in \{2, 3\}$, as a typical model problem: find $u : \overline{\Omega} \to \mathbb{R}$ such that

$$\begin{aligned} -\operatorname{div}(\alpha \nabla u) &= f \text{ in } \Omega, \\ u &= 0 \text{ on } \Gamma_D, \\ \alpha \frac{\partial u}{\partial n} &= g_N \text{ on } \Gamma_N, \end{aligned} \tag{2.1}$$

with given, sufficient smooth data f, g_N and α, where the coefficient α is uniformly bounded from below and above by some positive constants α_{min} and α_{max}, respectively. The boundary $\partial\Omega$ of the computational domain Ω consists of a Dirichlet part Γ_D of positive boundary measure and a Neumann part Γ_N. Without loss of generality, we assume homogeneous Dirichlet conditions. This can always be obtained by homogenization.

We multiply the partial differential equation (2.1) by a test function $v \in C^\infty(\Omega)$ that vanish on the Dirichlet boundary, and integrate over Ω. By applying integration by parts to the term with the second-order derivatives, we obtain the following variational problem: find $u \in V_D := \{u \in H^1(\Omega) : u|_{\Gamma_D} = 0 \text{ on } \Gamma_D\}$ such that

$$a(u, v) = \langle F, v \rangle \quad \forall v \in V_D. \tag{2.2}$$

The bilinear form $a(\cdot, \cdot) : V_D \times V_D \to \mathbb{R}$ and the linear form $\langle F, \cdot \rangle : V_D \to \mathbb{R}$ are given by the expressions

$$a(u, v) := \int_\Omega \alpha \nabla u \nabla v\,dx \quad \text{and} \quad \langle F, v \rangle := \int_\Omega f v\,dx + \int_{\Gamma_N} g_N v\,ds,$$

respectively. The standard tool to show existence and uniqueness of a solution to (2.2) is the Lemma of Lax-Milgram.

Lemma 2.3. *Let X be a Hilbert space with the norm $\|\cdot\|$. Let the bilinear form $a(\cdot,\cdot) : X \times X \to \mathbb{R}$ be* continuous*, i.e., there exists a positive constant $\overline{c}$ such that*

$$a(u,v) \leq \overline{c}\,\|u\|_X \|v\|_X \quad \forall u, v \in X,$$

and coercive *on X, i.e., there exists a positive constant $\underline{c}$ such that*

$$a(u,u) \geq \underline{c}\,\|u\|_X^2 \quad \forall u \in X.$$

Then, for any $F \in X^*$, the variational problem,

$$\text{find } u \in X: \quad a(u,v) = \langle F, v\rangle \quad \forall v \in X, \tag{2.3}$$

has a unique solution u which fulfils the a-priori estimate

$$\|u\|_X \leq \underline{c}^{-1}\,\|F\|_{X^*}$$

Proof. See, e.g., [183] or [29]. □

If the bilinear form is additionally symmetric, i.e., $a(u,v) = a(v,u)$, then its possible to show equivalence to a minimization problem. This is known as Ritz Lemma.

Lemma 2.4. *Let X be a Hilbert space with the norm $\|\cdot\|$, and let the bilinear form $a(\cdot,\cdot) : X \times X \to \mathbb{R}$ fulfil the assumptions of Lemma 2.3, and be additionally symmetric. Furthermore, let $F : X \to \mathbb{R}$ be a bounded linear functional. Then the unique solution of* (2.3) *is the unique minimizer of the* Ritz energy functional

$$u = \underset{v \in X}{\mathit{argmin}}\, \frac{1}{2} a(v,v) - \langle F, v\rangle,$$

and vice versa.

Proof. See, e.g., [29]. □

It is easy to check that equation (2.2) fulfils the assumptions of Lemma 2.3. The coercivity follows from Lemma 2.1.

2.2 Isogeometric Analysis

B-Splines and NURBS play an important role in computer aided design and computer graphics. Here, we will use these splines for building our trial and test spaces for the Galerkin approximations to (2.2), as proposed in [107], see also the monograph [40]. This section provides the definition of B-Splines in one dimension and in higher dimensions via a tensor-product structure. We will give an overview of isogeometric discretization, and summarize the approximation properties of these B-Splines and NURBS. Finally, we state important inequalities, like the discrete trace and inverse inequalities. A more detailed introduction to IgA can be found in [40] and [16], and for more informations on B-Splines and NURBS we refer to [185].

2.2.1 Univariate B-Splines

First we start with the definition of a knot vector, which is one of the important building blocks of a B-Spline.

Definition 2.5. *Let $0 = \xi_1 \leq \ldots \leq \xi_m = 1$ be a finite, real-valued, monotonically increasing sequence of real numbers. The set $\Xi = \{\xi_1, \ldots, \xi_m\}$ is called* knot vector*. An entry $\xi_i, i \in \{1, \ldots, m\}$, is called* knot*, and is called an* interior knot *if $(\xi_1 < \xi_i) \wedge (\xi_i < \xi_m)$. If r knots have the same value, we say that the knot has multiplicity r, i.e., $r = |\{j \in \{1, \ldots, m\} : \xi_j = \xi_i\}|$ is the cardinal number of the set $\{j \in \{1, \ldots, m\} : \xi_j = \xi_i\}$. The interval between two knots is called* knot span*. A knot span is called* empty *if $\xi_i = \xi_{i+1}$ and is called interior if $\xi_1 < \xi_{i+1} \wedge \xi_i < \xi_m$. The knots provide a partitioning of the parameter domain into elements. If the knots are equally spaced, we call it* uniform*, otherwise* non-uniform*.*

Based on a knot vector, we can define the B-Spline functions recursively.

Definition 2.6. *Let $p \in \mathbb{N}$ and Ξ be a knot vector with multiplicity of any interior knot of at most p. Then, by the* Cox-de Boor *recursion formula, we can define the $M = m - p - 1$* univariate B-Spline *basis functions on $[\xi_1, \xi_m]$ as follows:*

$$\hat{N}_{i,0}(\xi) = \begin{cases} 1 & \text{if } \xi_i \leq \xi \leq \xi_{i+1} \\ 0 & \text{otherwise} \end{cases}, \tag{2.4}$$

$$\hat{N}_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} \hat{N}_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} \hat{N}_{i+1,p-1}(\xi), \tag{2.5}$$

where $i = 1, \ldots, M$. If in (2.5) appears an expression $0/0$ we define it as 0. The number p is then called degree *of the B-Spline.*

Definition 2.7. *Let Ξ be a knot vector. We say that the knot vector is* open *if the multiplicity of the first knot and the last knot are $p+1$, whereas the multiplicity of the other knots is at most p.*

An example of B-Splines defined on an open knot vector is presented in Figure 2.1. B-Splines defined on open knot vectors have the property that they are interpolatory at the boundary of the parameter interval $[0, 1]$, while all other basis functions are zero there. Hence it is possible to distinguish between basis functions corresponding to the interior and the boundary. Since later on, we will only be interested in C^0 continuity across the interfaces, we restrict our analysis to open knot vectors.

Assumption 1. We consider all knot vectors used as open knot vectors.

We summarize some important properties of B-Splines:

1. The B-Splines basis functions $\hat{N}_{i,p}$ form a partition of unity, i.e.

$$\sum_{i=1}^{M} \hat{N}_{i,p}(\xi) \equiv 1$$

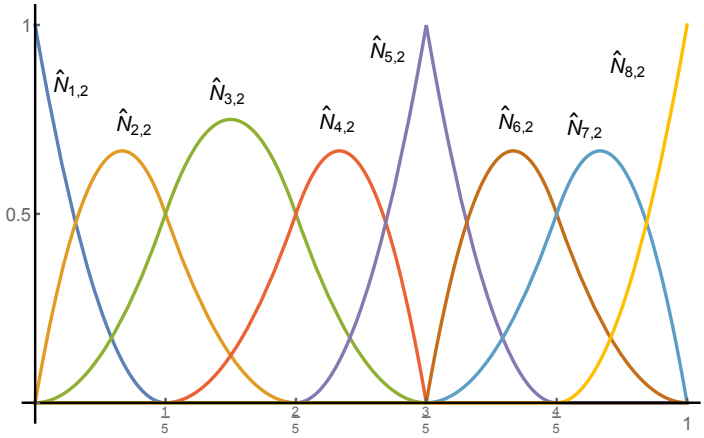


Figure 2.1: Illustration of B-Spline basis of degree 2, corresponding to the knot vector $\Xi = \{0, 0, 0, 1/5, 2/5, 3/5, 3/5, 4/5, 1, 1, 1\}$. The repeated knot $\xi_i = 3/5$ with multiplicity 2 leads to a nodal basis function at ξ_i.

for all $p = 0, 1, \ldots$.

2. The B-Spline basis functions are non-negative, i.e.

$$\forall \xi \in [\xi_1, \xi_m] \, \forall i \in \{1, \ldots, M\} : \hat{N}_{i,p}(\xi) \geq 0.$$

3. The support $\operatorname{supp} \hat{N}_{i,p}$ of $\hat{N}_{i,p}$ is local:

$$\operatorname{supp} \hat{N}_{i,p} \subseteq (\xi_i, \xi_{i+p+1}), \quad \forall i \in \{1, \ldots, M\}.$$

where $i \in \{1, \ldots, m-1\}$.

4. On each knotspan the B-Spline basis functions are piecewise polynomials of degree p and, without multiple knots in the interior, C^{p-1} continuous. At a knot with multiplicity r, it has C^{p-r} continuity. Hence, the continuity is reduced in the presence of multiple knots.

5. If a knot ξ_l has the multiplicity $r = p$, then there is one basis function $\hat{N}_{i,p}$, such $\hat{N}_{i,p}(\xi_l) = 1$ and all other basis functions have zero value there, i.e., the basis is interpolatory at ξ_l.

The proofs are elementary, and we refer to [185] for a more detailed discussion. Comparing with finite elements, the B-Splines have very similar properties. They form a partition of unity, they are piecewise polynomials, and they have local support. However, one significant difference is that the B-Splines are not a *nodal* basis and their support is not restricted to one element, but is within p neighbouring elements. When increasing the multiplicity of all interior knots up to p, one basically obtains again a nodal basis, where the support of each function is only one element. However, this is not the preferable way of working with B-Splines.

We point out that the order of approximation does not depend on the smoothness of the B-Splines. Hence, using B-Splines of maximal smoothness is optimal in the sense of approximation order and number of degrees of freedom. To be more precise, a basis with maximal smoothness has $O(n+p)$ basis functions, whereas, the C^0 version has $O(np)$, where n is the number of interior knot-spans and p the degree.

2.2.2 Tensor-Product B-Splines

Until now, we have considered B-Splines only in one dimension. There are several ways, how to generalize the concept to multi-dimensional case. One straightforward way for the construction is by taking the tensor-product of one-dimensional B-Splines, which is the method of choice for many applications. It is easy to see that a tensor-product does not allow local refinement of a single element. One has to consider other concepts of generalizations in order to handle local refinement appropriately like T-Splines, HB-Splines or THB-Splines, see, e.g., [73] and [15]. Let us now formally define tensor-product B-Splines.

Definition 2.8. *Let $(p^1,\dots,p^d)$ be a vector in $\mathbb{N}^d$, and let, for all $l=1,\dots,d$, Ξ^l be a knot vector. Furthermore, we denote the i^l univariate B-Spline defined on the knot vector Ξ^l by $\hat{N}^l_{i^l,p}(\xi^l)$. Then the d-dimensional tensor-product B-Spline (TB-Spline) is defined by*

$$\hat{N}_{(i^1,\dots,i^d),(p^1,\dots,p^d)}(\xi)=\prod_{l=1}^{d}\hat{N}^l_{i^l,p^l}(\xi^l). \tag{2.6}$$

In order to avoid cumbersome notations we will again denote the tensor-product B-Spline by $\hat{N}_{i,p}$, and interpret i and p as multi-indices. Additionally, we define the set of multi-indices by $\mathcal{I}:=\{(i^1,\dots,i^d):i^l\in\{1,\dots,M_l\}\,\forall l\in\{1,\dots,d\}\}$, where M_l are the number of B-Spline basis function for dimension l. Due to the tensor-product structure, the TB-Splines posses similar properties as the univariate B-Splines.

In case of TB-Splines, we will call a non-empty knotspan $\hat{Q}_i=(\xi_i,\xi_{i+1}), i\in\mathcal{I}_{\hat{\mathcal{Q}}}$ also *cell*, where $\mathcal{I}_{\hat{\mathcal{Q}}}=\{i\in\mathcal{I}|i^l\neq M_l\}$, and (ξ_i,ξ_{i+1}) is defined as

$$(\xi_i,\xi_{i+1}):=(\xi^1_{i^1},\xi^1_{i^1+1})\times\ldots\times(\xi^d_{i^d},\xi^d_{i^d+1}).$$

The mesh created by these cells is denoted by $\hat{\mathcal{Q}}_h$, i.e.

$$\hat{\mathcal{Q}}_h:=\{(\xi_i,\xi_{i+1})|i\in\mathcal{I}_{\hat{\mathcal{Q}}}\}.$$

It is important to note that manipulating one univariate knot vector influences the TB-Splines globally. For example, if we increase the multiplicity of a certain knot ξ^*, we obtain reduced continuity across the plane $\{x\,|\,x\in[0,1]^d,\,x_i=\xi^*\}$. In the following, we only consider TB-Splines and will denote them as B-Splines.

Similar as in Section 2.2.1, a basis with maximal smoothness has $O((n+p)^d)$ basis functions, whereas the basis with only C^0 continuity has $O((np)^d)$ basis functions, where n is the number of interior knot spans, p the B-Spline degree and d the dimension.

2.2.3 B-Spline Geometries and Geometrical Mapping

The B-Splines are used to represent a d-dimensional geometry in $\mathbb{R}^g$, where $d \leq g$. However, in the following, we will restrict ourselves to the case $d = g \in \{2, 3\}$.

Definition 2.9. *Let $\{\hat{N}_{i,p}\}_{i\in\mathcal{I}}$ be a family of B-Spline basis functions. Given* control points *$P_i \in \mathbb{R}^d$, $i \in \mathcal{I}$, the* B-Spline volume *is defined by*

$$G : \hat{\Omega} := (0,1)^d \to \mathbb{R}^d$$
$$G(\xi) := \sum_{i\in\mathcal{I}} P_i \hat{N}_{i,p}(\xi).$$

We call G the geometrical mapping*, the domain $\hat{\Omega}$ of G* parameter domain*, and the image $\Omega := G(\hat{\Omega}) \subset \mathbb{R}^d$* physical domain*. The physical domain is the computational domain. The geometrical mapping is called* regular *if $\det \nabla G(\xi) \neq 0$, $\forall \xi \in [0,1]^d$.*

The knot vector Ξ provides a partition of the parameter domain into cells, and, by means of the geometrical mapping, we receive a corresponding partition of the physical space into cells Q_i as well, where $Q_i = G(\hat{Q}_i)$ for $\hat{Q}_i \in \hat{\mathcal{Q}}_h$. If we collect all these cells, we obtain a mesh

$$\mathcal{Q}_h := \{Q = G(\hat{Q}) | \hat{Q} \in \hat{\mathcal{Q}}_h\}$$

for the physical domain Ω As is the case of finite elements, we need some restrictions imposed on the mesh.

Definition 2.10. *A family of meshes $\{\mathcal{Q}_h\}_{h\in H}$ is called* quasi uniform *if there exists a constant $\theta \geq 1$ for all $\mathcal{Q}_h \in \{\mathcal{Q}_h\}_{h\in H}$, such that $\theta^{-1} \leq diam(Q)/diam(Q') \leq \theta$ for all $Q, Q' \in \mathcal{Q}_h$. The characteristic mesh-size of $\hat{\mathcal{Q}}_h$ and $\mathcal{Q}_h$ is denoted by $\hat{h}$ and h, respectively.*

In this theses, we do not investigate local refinement. Therefore, it is enough to assume quasi uniform meshes.

Assumption 2. All considered meshes are quasi uniform.

Moreover, we make the following assumption on the gradient of G.

Assumption 3. We assume that the geometrical mapping G has the properties

$$\|\nabla G\|_{L^\infty(\hat{\Omega})} \approx H \quad \text{and} \quad \|\det \nabla G\|_{L^\infty(\hat{\Omega})} \approx H^d,$$

where the hidden constants are independent of the mesh-size h and the diameter H of the domain Ω.

2.2.4 Multi-Patch Geometries

Up to this point, we have just considered a single domain being the image of the geometrical mapping G. In many practical applications, it is not possible to describe the physical computational domain Ω just with a single geometrical mapping G. For example, domains with holes have to be represented with several patches. Moreover, for the stability and accuracy of the numerical method, it is advantageous to have patches, which are topologically equivalent to a cube. Therefore, we represent the physical domain Ω by N non-overlapping domains $\Omega^{(k)}$, called *patches*. Each $\Omega^{(k)}$ is the image of an associated geometrical mapping $G^{(k)}$, defined on the parameter domain $\hat{\Omega} := (0,1)^d$, i.e., $\Omega^{(k)} = G^{(k)}(\hat{\Omega})$ for $k = 1, \dots, N$, and $\overline{\Omega} = \bigcup_{k=1}^{N} \overline{\Omega}^{(k)}$. Clearly, each patch has a mesh $\mathcal{Q}_h^{(k)}$ in the physical domain and a mesh $\hat{\mathcal{Q}}_h^{(k)}$ in the parameter domain, consisting of cells $Q^{(k)}$ and $\hat{Q}^{(k)}$.

We denote the interface between the two patches $\Omega^{(k)}$ and $\Omega^{(l)}$ by $F^{(kl)}$, and the collection of all interfaces by Γ, i.e.,

$$F^{(kl)} = \overline{\Omega}^{(k)} \cap \overline{\Omega}^{(l)} \quad \text{and} \quad \Gamma := \bigcup_{l>k} F^{(kl)}.$$

Furthermore, the boundary of the domain is denoted by $\partial\Omega$. The interface Γ is often called *skeleton*. We denote the set of all indices l such that $\Omega^{(k)}$ and $\Omega^{(l)}$ have a common interface $F^{(kl)}$ by $\mathcal{I}_{\mathcal{F}}^{(k)}$. We restrict ourselves to the case that the geometry is C^0 continuous at the interfaces. For certain applications, such as higher-order PDEs, it may be necessary to consider higher smoothness of the geometry.

At the boundary of a patch $\Omega^{(k)}$, the B-Splines based on open knot vectors are interpolatory. This is an useful property, when incorporating Dirichlet boundary conditions and coupling patches in a continuous manner. However, it is quite hard to couple patches in a smooth way, e.g., higher geometric continuity $G^k, k > 1$, see e.g., [113] and references therein.

2.2.5 Isogeometric Discretization

The key point in isogeometric analysis is the use of the same basis functions for representing the geometry as well as for the solution space. This motives the definition of the basis functions in the physical domain via the push-forward of the basis functions in the parameter domain, i.e.,

$$N_{i,p}(x) := \hat{N}_{i,p} \circ G^{-1}(x).$$

Thus, we define our discrete function space V_h by

$$V_h := \operatorname{span}\{N_{i,p}\}_{i \in \mathcal{I}} \subset H^1(\Omega). \tag{2.7}$$

The function u_h from the IgA space V_h can therefore be represented in the form

$$u_h(x) = \sum_{i \in \mathcal{I}} \boldsymbol{u}_i N_{i,p}(x).$$

Hence, each function $u_h(x)$ is associated with the vector $\boldsymbol{u} = (\boldsymbol{u}_i)_{i \in \mathcal{I}}$. This map is known as *Ritz isomorphism*. One usually writes this relation as $u_h \leftrightarrow \boldsymbol{u}$, and we will use it in the following without further comments. For completeness, we also define as S_h the space of spline functions in the parameter domain, i.e.,

$$S_h = \operatorname{span}\{\hat{N}_{i,p}\}_{i \in \mathcal{I}} \subset H^1(\hat{\Omega}).$$

If we consider a single patch $\Omega^{(k)}$ of a multi-patch domain Ω, we will use the notation $V_h^{(k)}, N_{i,p}^{(k)}, \hat{N}_{i,p}^{(k)}, G^{(k)}, \ldots$ with the analogous patch-wise definitions. For example, the finite dimensional function space $V_h^{(k)}$ is defined as

$$V_h^{(k)} = \operatorname{span}\{N_{i,p}^{(k)}\}_{i \in \mathcal{I}^{(k)}} \subset H^1(\Omega^{(k)}). \tag{2.8}$$

To keep notation simple, we will use h_k and $\hat{h}_k$ instead of $h^{(k)}$ and $\hat{h}^{(k)}$, respectively.

2.2.6 Approximation Properties

This section lists some important properties of the approximation power of B-Splines from [14]. First of all, we state a result about the relation of the H^m norms between the function in the physical and the parameter domain, summarized in Proposition 2.11 and Corollary 2.12, which are proved in [14], see Lemma 3.5.

Proposition 2.11. *Let m be a non-negative integer, $\hat{Q} \in \hat{\mathcal{Q}}_h$ and $Q = G(\hat{Q})$. Then the equivalence inequalities*

$$\begin{aligned} |\hat{v}|_{H^m(\hat{Q})} &\leq C_{shape} \left\| \det \nabla G^{-1} \right\|_{L^\infty(Q)}^{1/2} \sum_{j=0}^{m} \|\nabla G\|_{L^\infty(\hat{Q})}^{j} |v|_{H^j(Q)}, \\ |v|_{H^m(Q)} &\leq C_{shape} \|\det \nabla G\|_{L^\infty(\hat{Q})}^{1/2} \|\nabla G\|_{L^\infty(\hat{Q})}^{-m} \sum_{j=0}^{m} |\hat{v}|_{H^j(\hat{Q})}. \end{aligned} \tag{2.9}$$

hold for all $v \in H^m(Q)$ and their counterparts $\hat{v} \in H^m(\hat{Q})$, where C_{shape} are positive generic constants that only depend on the shape of the geometry Ω and its parametrization.

From Proposition 2.11, one obtains

$$C_1 \|\hat{v}\|_{H^m(\hat{Q})} \leq \|v\|_{H^m(Q)} \leq C_2 \|\hat{v}\|_{H^m(\hat{Q})}, \tag{2.10}$$

where the constants C_1, C_2 depend only on ∇G and the shape of the geometry Ω. We note that the 0-order terms in the upper bounds of Proposition 2.11 are not needed for $m > 0$. They are incorporated in order to give a unified presentation for $m \geq 0$. Hence, as a special case of Proposition 2.11, we obtain the following estimates for the L^2 norm and H^1 semi-norm

Corollary 2.12. *Let $\hat{Q} \in \hat{\mathcal{Q}}_h$ and $Q = G(\hat{Q})$. For $v \in L^2(Q)$, we have*

$$\begin{aligned} \|\hat{v}\|_{L^2(\hat{Q})} &\leq C_{shape} \left\|\det \nabla G^{-1}\right\|^{1/2}_{L^\infty(Q)} \|v\|_{L^2(Q)}, \\ \|v\|_{L^2(Q)} &\leq C_{shape} \|\det \nabla G\|^{1/2}_{L^\infty(\hat{Q})} \|\hat{v}\|_{L^2(\hat{Q})}, \end{aligned} \tag{2.11}$$

and, for $v \in H^1(Q)$, we can write

$$\begin{aligned} |\hat{v}|_{H^1(\hat{Q})} &\leq C_{shape} \left\|\det \nabla G^{-1}\right\|^{1/2}_{L^\infty(Q)} \|\nabla G\|_{L^\infty(\hat{Q})} |v|_{H^1(Q)}, \\ |v|_{H^1(Q)} &\leq C_{shape} \|\det \nabla G\|^{1/2}_{L^\infty(\hat{Q})} \|\nabla G\|^{-1}_{L^\infty(\hat{Q})} |\hat{v}|_{H^1(\hat{Q})}, \end{aligned} \tag{2.12}$$

where C_{shape} is as in Proposition 2.11.

The next results describes the quantitative approximation power of B-Splines. It basically states that the B-Splines space has the same approximation power as a FE space of same degree. We do not present any further technical details. We refer the reader to [14], [24], [35] and [208] for a more comprehensive discussion regarding approximation properties of B-Splines and NURBS. In particular, for the proof of the following theorem we refer to Corollary 3.1 in [208], see also Theorem 3.2 in [14].

Theorem 2.13. *Let $v \in H^l(\Omega)$ be a function defined in the physical domain Ω. Given an integer k such that $0 \leq k \leq p+1$, $k \leq l$, and $k \leq s+1$, where s is the smoothness of the considered B-Spline basis. Then there exists a projection operator $\Pi_{V_h} : L^2(\Omega) \to V_h$ such that the approximation error estimates*

$$|v - \Pi_{V_h} v|^2_{H^k(\Omega)} \leq C h^{\delta - k} |v|^2_{H^l(\Omega)},$$

holds, where $\delta := \min(p+1, l)$ and the constant C is independent of h.

2.2.7 Important Discrete Inequalities

In this section, we want to formulate important inequalities for IgA, which are well known for FE. This inequalities have been proven in the more general case of having $W^{l,q}(\Omega)$ spaces, e.g., in [148]. For the purpose of this theses, we reformulate them in the case of Sobolev spaces $H^l(\Omega) = W^{l,2}(\Omega)$. We refer to [59] and [14] for more discussions on discrete trace and inverse inequalities in IgA. We start with the continuous trace inequality, see Lemma 1 in [148].

Lemma 2.14. *Let $u \in H^l(\Omega), l > 1$, there is a constant C, which depends on the problem data and the constants from* (2.10)*, but not on h, such that for any face $F \subset \partial\Omega$*

$$\|v\|_{L^2(F)} \leq C \left(h^{-1} \|v\|_{L^2(\Omega)} + h \| \nabla v\|_{L^2(\Omega)} \right).$$

We follow the list with the discrete trace inequality for $v \in S_h$, see Lemma 3 in [148].

Lemma 2.15. *For all $\hat{v} \in S_h$ and for all $\hat{F} \subset \partial\hat{Q}$, there is a constant C, which depends on the mesh quasi-uniformity constant θ from Definition 2.10, such that*

$$\|\hat{v}\|_{L^2(\hat{F})} \leq C\hat{h}^{-\frac{1}{2}} \|\hat{v}\|_{L^2(\hat{Q})}.$$

Finally, we end this section with a typical inverse inequality, see Lemma 2 in [148].

Lemma 2.16. *For all $\hat{v} \in S_h$, there is a constant C, which depends on the mesh quasi-uniformity constant θ from Definition 2.10, such that*

$$\| \nabla \hat{v}\|_{L^2(\hat{Q})} \leq C\hat{h}^{-1} \|\hat{v}\|_{L^2(\hat{Q})}.$$

To summarize, we obtain the same h dependence as for FE, where the constant C depends on the smoothness k and the degree p. By means of Proposition 2.11 and Corollary 2.12, we obtain the corresponding inequalities for the physical domain, where the constant C additionally depends on the geometrical mapping G.

2.3 Galerkin Discretizatons

A common and well understood framework for obtaining approximate solutions to variational equations is to use the Galerkin principle. We look for approximations in a finite dimensional space $V_{h,D}$, and choose the test functions from the same space. One could choose different spaces for the solution and test space, leading to the so-called *Petrov-Galerkin* methods. For the purpose of this thesis, we restrict ourselves to the classical Galerkin methods. Note, $V_{h,D}$ is not necessarily a subspace of V_D. If $V_{h,D} \subset V_D$, we call the discretization *continuous Galerkin (cG)* method. Otherwise, we investigate the so-called *discontinuous Galerkin (dG)* method.

2.3.1 Continuous Galerkin Discretization

In Section 2.1.2, we derived the weak formulation (2.2) of (2.1) in the following variational setting: find $u \in V_D$

$$a(u, v) = \langle F, v \rangle \quad \forall v \in V_D.$$

We are considering the finite dimensional subspace V_h^{cG} of $H^1(\Omega)$ given by

$$V_h^{cG} := \{v \mid v|_{\Omega^{(k)}} \in V_h^{(k)}\} \cap H^1(\Omega).$$

Since, we restrict ourselves to homogeneous Dirichlet conditions, we look for the Galerkin approximate u_h from $V_{D,h}^{cG} \subset V_h^{cG}$, where $V_{D,h}^{cG}$ contains all functions, which vanish on the Dirichlet boundary. The continuous Galerkin IgA scheme reads as follows: find $u_h \in V_{D,h}^{cG}$ such that

$$a(u_h, v_h) = \langle F, v_h \rangle \quad \forall v_h \in V_{D,h}^{cG}. \tag{2.13}$$

Since $V_{D,h}^{cG} \subset V_D$, the continuous coercivity implies the discrete one. Hence, there exists a unique IgA solution $u_h \in V_{D,h}^{cG}$ of (2.13). Moreover, one can show that this solution converges to the solution $u \in V_D$ of (2.2) for h tends to 0. A basis for this space is given by the B-Spline functions $\{N_{i,p}\}_{i \in \mathcal{I}_0}$, where $\mathcal{I}_0$ contains all indices of $\mathcal{I}$ which do not have a support on Γ_D. Once a basis is chosen, the continuous Galerkin IgA scheme (2.13) is equivalent to the linear system of algebraic equations

$$\boldsymbol{K}\boldsymbol{u} = \boldsymbol{f}, \tag{2.14}$$

where $\boldsymbol{K} = (\boldsymbol{K}_{i,j})_{i,j \in \mathcal{I}_0}$ and $\boldsymbol{f} = (\boldsymbol{f}_i)_{i \in \mathcal{I}_0}$ denote the stiffness matrix and the load vector, respectively, with $\boldsymbol{K}_{i,j} = a(N_{j,p}, N_{i,p})$ and $\boldsymbol{f}_i = \langle F, N_{i,p} \rangle$, and $\boldsymbol{u}$ is the vector representation of u_h given by the IgA isomorphism. Moreover, we have the relation $a(u_h, v_h) = (\boldsymbol{K}\boldsymbol{u}, \boldsymbol{v})_{\ell_2}$ with euclidean inner product $(\cdot, \cdot)_{\ell_2}$. In order to keep the notation simple, we will reuse the symbol $\mathcal{I}$ for the set $\mathcal{I}_0$ in the following. Note, the matrix $\boldsymbol{K}$ is symmetric and positive definite (SPD).

In the conforming case, the error analysis is done by means of Cea's Lemma.

Lemma 2.17. *Let the assumptions of Lemma 2.3 be fulfilled, let $u \in V_D$ and $u_h \in V_{D,h}^{cG}$ be the solution of* (2.2) *and* (2.13)*, respectively. Then we have*

$$\|u - u_h\|_{H^1(\Omega)} \leq \frac{\overline{c}}{\underline{c}} \inf_{v_h \in V_{D,h}^{cG}} \|u - v_h\|_{H^1(\Omega)}.$$

Proof. See, e.g., [29]. □

This means that we can estimate the error by the best approximation. By using a projection operator $\Pi_{V_{D,h}^{cG}} : L^2(\Omega) \to V_{D,h}^{cG}$, as in Section 2.2.6 we can further estimate the best approximation error by the quasi-interpolation error, i.e.,

$$\|u - u_h\|_{H^1(\Omega)} \leq \frac{\overline{c}}{\underline{c}} \|u - \Pi_{V_{D,h}^{cG}} u\|_{H^1(\Omega)}.$$

By means of Theorem 2.13, we obtain the a-priori error bound

$$\|u - u_h\|_{H^1} \leq C h^{\gamma} \|u\|_{H^l(\Omega)},$$

provided that $u \in H^l(\Omega)$, where $\gamma = \min(p+1, l)-1$ and p is the B-Spline degree.

Regarding efficient solution techniques by means of iterative solvers, the (spectral) condition number κ_2, defined as

$$\kappa_2(\boldsymbol{K}) := \|\boldsymbol{K}\|_2 \|\boldsymbol{K}^{-1}\|_2,$$

plays an important role. Certainly, one can define different condition numbers based on different norms. If not specified, we only consider the spectral condition number κ_2 and denote it by κ. In the case of a symmetric matrix, the condition number is given by the ratio of the largest to the smallest eigenvalue (by moduli).

Considering the stiffness matrix $\boldsymbol{K}$ and mass matrix $\boldsymbol{M} := ((\hat{N}_{i,p}, \hat{N}_{j,p})_{L_2})_{i,j\in\mathcal{I}}$ on the parameter domain, one can show the following bounds on the condition number

$$\kappa(\boldsymbol{K}) \leq C h^{-2} p^d 4^{pd}$$
$$\kappa(\boldsymbol{M}) \leq C p^{2(d-1)} 4^{dp},$$

where d is the dimension, see, e.g., [194], [66] and references therein.

2.3.2 Discontinuous Galerkin Discretization

In Section 2.3.1, we used functions which are continuous on the whole multi-patch domain Ω. In this section, we want to relax this restriction and consider spaces, which are conforming on each patch, but are not necessarily continuous across the patch interfaces. For the dG-IgA scheme, we again use the spaces $V_h^{(k)}$ as defined in (2.8), whereas now discontinuities are allowed across the patch interfaces $F^{(kl)}$. The continuity of the function value and its co-normal derivative are then enforced in a weak sense by adding additional terms to the bilinear form. This situation is especially important when we consider non-matching grids on different patches across the interface or gaps and overlaps due to segmentation crimes, see Section 4.4.

We define the dG-IgA space as

$$V_h^{dG} := \{v \,|\, v|_{\Omega^{(k)}} \in V_h^{(k)}\}. \tag{2.15}$$

We now follow the notation used in [50] and [52]. A comprehensive study of dG schemes for FE can be found in [188] and [41]. For the analysis of dG-IgA schemes, we refer to [148] and [144].

Dirichlet boundary conditions can be handled in different ways. We can use the dG technique to incorporate them in a weak sense, see, e.g., [5] and [6]. This method for imposing Dirichlet boundary conditions was already proposed by Nitsche [177]. Another method consists in enforcing them in a strong sense via an L^2 projection and homogenization. In this paper, for simplicity of presentation, we will follow the latter one, where we assume that the given Dirichlet data can exactly be represented

by means of B-Splines. Hence, we define $V_{D,h}^{dG}$ as the space of all functions from V_h^{dG} which vanish on the Dirichlet boundary Γ_D.

Having these definitions at hand, we can define the discrete problem based on the Symmetric Interior Penalty (SIP) dG formulation as follows: find $u_h \in V_{D,h}^{dG}$ such that

$$a_h(u_h, v_h) = \langle F, v_h \rangle \quad \forall v_h \in V_{D,h}^{dG}, \tag{2.16}$$

where

$$a_h(u,v) := \sum_{k=1}^{N} a_e^{(k)}(u,v) \quad \text{and} \quad \langle F, v \rangle := \sum_{k=1}^{N} \int_{\Omega^{(k)}} f v^{(k)}\, dx + \int_{\Gamma_N^{(k)}} g_N v^{(k)}\, ds,$$
$$a_e^{(k)}(u,v) := a^{(k)}(u,v) + s^{(k)}(u,v) + p^{(k)}(u,v),$$

and

$$a^{(k)}(u,v) := \int_{\Omega^{(k)}} \alpha^{(k)} \nabla u^{(k)} \nabla v^{(k)}\, dx,$$
$$s^{(k)}(u,v) := \sum_{l \in \mathcal{I}_{\mathcal{F}}^{(k)}} \int_{F^{(kl)}} \frac{\alpha^{(k)}}{2} \left(\frac{\partial u^{(k)}}{\partial n}(v^{(l)} - v^{(k)}) + \frac{\partial v^{(k)}}{\partial n}(u^{(l)} - u^{(k)}) \right) ds,$$
$$p^{(k)}(u,v) := \sum_{l \in \mathcal{I}_{\mathcal{F}}^{(k)}} \int_{F^{(kl)}} \frac{\delta \alpha^{(k)}}{h_{kl}} (u^{(l)} - u^{(k)})(v^{(l)} - v^{(k)})\, ds.$$

Here, δ is a positive, sufficiently large penalty parameter, and h_{kl} is the harmonic average of the adjacent mesh-sizes, i.e., $h_{kl} = 2h_k h_l/(h_k + h_l)$. We equip $V_{D,h}^{dG}$ with the so-called dG norm

$$\|u\|_{dG}^2 := \sum_{k=1}^{N} \left[\alpha^{(k)} \left\| \nabla u^{(k)} \right\|_{\Omega^{(k)}}^2 + \sum_{l \in \mathcal{I}_{\mathcal{F}}^{(k)}} \frac{\delta \alpha^{(k)}}{h_{kl}} \int_{F^{(kl)}} (u^{(k)} - u^{(l)})^2 ds \right]. \tag{2.17}$$

Furthermore, we define the bilinear forms

$$d_h(u,v) := \sum_{k=1}^{N} d^{(k)}(u,v) \quad \text{and} \quad d^{(k)}(u,v) := a^{(k)}(u,v) + p^{(k)}(u,v)$$

for later use. We note that $\|u_h\|_{dG}^2 = d_h(u_h, u_h)$. We are now able to show existence and uniqueness of a solution to (2.16). The following Lemma is an equivalent statement of Lemma 2.1 in [52] for IgA, and the proof is based on the results in [148].

Lemma 2.18. *Let δ be sufficiently large. Then there exist two positive constants γ_0 and γ_1 which are independent of $h_k, H_k, \delta, \alpha^{(k)}$ and u_h such that the inequalities*

$$\gamma_0 d^{(k)}(u_h, u_h) \leq a_e^{(k)}(u_h, u_h) \leq \gamma_1 d^{(k)}(u_h, u_h), \quad \forall u_h \in V_{D,h}^{dG} \tag{2.18}$$

are valid for all $k = 1, 2, \ldots, N$. *Furthermore, we have the inequalities*

$$\gamma_0 \|u_h\|_{dG}^2 \leq a_h(u_h, u_h) \leq \gamma_1 \|u_h\|_{dG}^2, \quad \forall u_h \in V_{D,h}^{dG}. \tag{2.19}$$

Proof. Rewriting the proofs of Lemma 6 and Lemma 7 in [148] for a single patch gives the desired inequalities (2.18). In order to show the boundedness, we additionally have to apply the inverse inequality $\|\nabla u_h\|_{L^2(F^{(kl)})}^2 \leq C h_k^{-1} \|\nabla u_h\|_{L^2(\Omega^{(k)})}^2$, see Lemma 2.16, to the term $\sum_{l \in \mathcal{I}_{\mathcal{F}}^{(k)}} \alpha^{(k)} h_k \|\nabla u_h\|_{L^2(F^{(kl)})}^2$ appearing in the bound of Lemma 7 in [148]. We then arrive at the estimate

$$\sum_{l \in \mathcal{I}_{\mathcal{F}}^{(k)}} \alpha^{(k)} h_k \|\nabla u_h\|_{L^2(F^{(kl)})}^2 \leq C \sum_{l \in \mathcal{I}_{\mathcal{F}}^{(k)}} \alpha^{(k)} \|\nabla u_h\|_{L^2(\Omega^{(k)})}^2 .$$

Hence, the right-hand side can be bounded by $d^{(k)}(u_h, u_h)$. Formula (2.19) immediately follows from (2.18), which concludes the proof. □

We note that, in [148], the results were obtained for the Incomplete Interior Penalty (IIP) scheme. An extension to SIP-dG and the use of harmonic averages for h and/or α are discussed in Remark 3.1 in [148], see also [144].

A direct implication of (2.19) is the well-posedness of the discrete problem (2.16) that immediately follows from Lax-Milgram's lemma. The consistency of the method together with quasi-interpolation estimates for B-spline quasi-interpolant lead to the following a-priori error estimate, as established in [148].

Theorem 2.19. *Let* $u \in H^1(\Omega) \cap \prod_{k=1}^N W^{l+1,q}(\Omega^{(k)})$ *with some* $q \in (\min(1, 2d/(d + 2l)), 2]$ *and some integer* $l \geq 1$, *solves (2.2), and let* $u_h \in V_{D,h}^{dG}$ *solves the discrete problem (2.16). Then the following a-priori error estimate holds*

$$\|u - u_h\|_{dG}^2 \leq \sum_{k=1}^N C^{(k)} \Big({h_k}^{2r} + \sum_{j \in \mathcal{I}_{\mathcal{F}}^{(k)}} \alpha^{(k)} \frac{h_k}{h_j} {h_k}^{2r} \Big),$$

where $r = \min(l + (\frac{d}{2} - \frac{d}{q}), p)$, *and* $C^{(k)}$ *is a positive constant which depends on* p, $\|u\|_{W^{l+1,q}(\Omega^{(k)})}$, *and* $\max_{l_0 \leq l+1} \|\nabla^{l_0} G^{(k)}\|_{L^\infty(\Omega^{(k)})}$, *but not on* h.

As explained in Section 2.3.1, we choose the B-Spline functions $\{N_{i,p}\}_{i \in \mathcal{I}_0}$ as basis for the space $V_{D,h}^{dG}$, see (2.15), where $\mathcal{I}_0$ contains all indices of $\mathcal{I}$, which do not have a support on the Dirichlet boundary. Hence, the dG-IgA scheme (2.16) is equivalent to the system of linear equations

$$\boldsymbol{K}\boldsymbol{u} = \boldsymbol{f}, \tag{2.20}$$

where $\boldsymbol{K} = (\boldsymbol{K}_{i,j})_{i,j \in \mathcal{I}_0}$ and $\boldsymbol{f} = (\boldsymbol{f}_i)_{i \in \mathcal{I}_0}$ denote the stiffness matrix and the load vector, respectively, with $\boldsymbol{K}_{i,j} = a_h(N_{j,p}, N_{i,p})$ and $\boldsymbol{f}_i = \langle F, N_{i,p} \rangle$, and $\boldsymbol{u}$ is the vector representation of u_h. As in the previous section, we reuse the symbol $\mathcal{I}$ for $\mathcal{I}_0$.

To summarize, for both the cG and dG formulation, we choose the B-Spline function $\{N_{i,p}\}_{i\in\mathcal{I}}$ as basis for the space $V_{D,h}^X, X \in \{cG, dG\}$. In the cG case, the basis functions on the interface are identified accordingly to obtain a conforming subspace of V_D. For the remainder of this thesis, we drop the subscript D for the symbols V_D and $V_{D,h}^X, X \in \{cG, dG\}$. Moreover, we will also drop the superscript $X \in \{cG, dG\}$ and use the symbol V_h for both formulations. Depending on the considered formulation, one needs to use the right space $V_h^X, X \in \{cG, dG\}$. Finally, we are mostly interested in the approximate solution u_h, hence, will also omit the subscript h in u_h to simplify the notation. If a distinction has to be made, we will write the subscript explicitly.

Chapter 3

Continuous Galerkin IETI-DP Methods

This chapter is devoted to the continuous Galerkin dual-primal isogeometric tearing and interconnecting methods (cG-IETI-DP). These methods were first introduced in [137] for the case of having only vertex primal variables in two dimensions. Following the ideas and notation in [183] for the FE version of IETI-DP, we extend the cG-IETI-DP algorithm to three-dimensional domains and more general primal variables such as continuous face or edge averages. The presented approach considers energy minimizing primal subspaces, where the primal space is orthogonal to the dual space with respect to the Schur complement S. The advantage is a block diagonal structure of one of the operators involved in the IETI-DP operator. Moreover, we incorporate the primal variables via constraints.

In Section 3.1, we will present the derivation of the method as well as important implementation details. The analysis of the condition number will be given in Section 3.3 and follows the proof in [19] for the BDDC preconditioner. We consider the case of having a two-dimensional domain, globally constant diffusion coefficient and only vertex values as primal variables. Finally, we present numerical results for two- and three-dimensional domains and jumping diffusion coefficients in Section 3.4.

3.1 Derivation of the Method

The starting point for the derivation is the discrete variational formulation (2.13). In the following, let V_h be the conforming IgA space which fulfils the Dirichlet boundary conditions as defined in Section 2.3.1. Furthermore, we denote the B-Spline basis of this space by $\{N_{i,p}\}_{i\in\mathcal{I}}$. The idea of IETI-DP is to introduce local spaces, which are independent of each other. The coupling and the continuity across interfaces is received via additional constraints. It is common practice to formulate the method in

terms of the Schur complement with respect to dofs on the interface Γ. This requires a splitting of the space V_h into a space corresponding to the interface dofs (B) and the interior dofs (I). It is important to note that in the implementation the calculation of the Schur complement is not required. The presentation of this section follows the lines in [19] and [183].

3.1.1 Schur Complement System

Since we are using open knot vectors, we can distinguish between basis functions on the interface Γ and in the interior of each patch $\Omega^{(k)}$, $k = 1, \dots, N$. We introduce the space of splines associated to the interface by

$$V_{\Gamma,h} := \operatorname{span}\{N_{i,p} | \operatorname{supp}\{N_{i,p}\} \cap \Gamma \neq \emptyset, i \in \mathcal{I}\} \cap H^1(\Omega) \subset V, \tag{3.1}$$

and for each patch $\Omega^{(k)}$ the corresponding space associated to its interior by

$$V_{I,h}^{(k)} := V_h^{(k)} \cap H_0^1(\Omega^{(k)}). \tag{3.2}$$

These spaces allow us to formulate the following decomposition

$$V_h = \prod_{k=1}^{N} V_{I,h}^{(k)} \oplus \mathcal{H}\left(V_{\Gamma,h}\right),$$

where $\mathcal{H}$ is the *discrete IgA harmonic extension* defined by

$$\begin{aligned} &\mathcal{H} : V_{\Gamma,h} \to V_h : \\ &\begin{cases} \text{find } \mathcal{H}v_B \in V_h : \\ \quad a(\mathcal{H}v_B, v^{(k)}) = 0 & \forall v^{(k)} \in V_{I,h}^{(k)},\ 1 \leq k \leq N, \\ \quad \mathcal{H}v_B = v_B & \text{on } \Gamma, \end{cases} \end{aligned} \tag{3.3}$$

see [19] and [196] for a more extensive discussion.

Introducing the bilinear form

$$s : V_{\Gamma,h} \times V_{\Gamma,h} \to \mathbb{R}, \quad s(w, v) = a(\mathcal{H}w, \mathcal{H}v),$$

one can show that the interface component u_B of the solution u to the IgA scheme (2.13) satisfies the variational identity

$$s(u_B, v_B) = \langle g, v_B \rangle \quad \forall v_B \in V_{\Gamma,h}, \tag{3.4}$$

where $g \in V_{\Gamma,h}^*$ is a suitable functional. By choosing the B-Spline basis for $V_{\Gamma,h}$, the variational identity (3.4) is equivalent to the linear system

$$\boldsymbol{S}\boldsymbol{u}_B = \boldsymbol{g}.$$

The matrix $\boldsymbol{S}$ is the Schur complement of $\boldsymbol{K}$ with respect to the interface dofs. If we reorder the entries of the stiffness matrix $\boldsymbol{K}$ and the load vector $\boldsymbol{f}$ such that the dofs corresponding to the interface are arranged at first, i.e.,

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{K}_{BB} & \boldsymbol{K}_{BI} \\ \boldsymbol{K}_{IB} & \boldsymbol{K}_{II} \end{bmatrix} \text{ and } \quad \boldsymbol{f} = \begin{bmatrix} \boldsymbol{f}_B \\ \boldsymbol{f}_I \end{bmatrix},$$

then $\boldsymbol{S}$ and $\boldsymbol{g}$ can be represented in the form

$$\boldsymbol{S} = \boldsymbol{K}_{BB} - \boldsymbol{K}_{BI}(\boldsymbol{K}_{II})^{-1}\boldsymbol{K}_{IB}, \qquad \boldsymbol{g} = \boldsymbol{f}_B - \boldsymbol{K}_{BI}(\boldsymbol{K}_{II})^{-1}\boldsymbol{f}_I.$$

Once $\boldsymbol{u}_B$ is calculated, we obtain $\boldsymbol{u}_I$ as the solution of the system

$$\boldsymbol{K}_{II}\boldsymbol{u}_I = \boldsymbol{f}_I - \boldsymbol{K}_{BI}\boldsymbol{u}_B.$$

Instead of the Schur complement matrix $\boldsymbol{S}$ we will mostly use its operator representation:

$$S : V_{\Gamma,h} \to V_{\Gamma,h}^*, \quad \langle Sv, w\rangle = (\boldsymbol{S}\boldsymbol{v}, \boldsymbol{w})_{\ell_2}.$$

3.1.2 Local Spaces and Jump Operator

We define the local interface space $W^{(k)}$ as the restriction of $V_{\Gamma,h}$ to $\Omega^{(k)}$, i.e.,

$$W^{(k)} := \operatorname{span}\{N_{i,p} | \operatorname{supp}\{N_{i,p}\} \cap \Gamma^{(k)} \neq \emptyset, i \in \mathcal{I}\}.$$

Furthermore, we define the space of functions, which are locally in $W^{(k)}$, by

$$W := \prod_{k=1}^{N} W^{(k)}.$$

We note that functions from W are, in general, not continuous across the interface, i.e. $W \not\subseteq C^0$. A function $w \in W$ has components $w := \left[w^{(k)}\right]_{k=1}^N \leftrightarrow \left[\boldsymbol{w}^{(k)}\right]_{k=1}^N =: \boldsymbol{w}$.

Later on, we will use constraints, which enforce the continuity across the interface. Let $\mathcal{B}(k,l)$ be the set of all coupled B-Spline basis functions between $\Omega^{(k)}$ and $\Omega^{(l)}$, then this constraints are given by

$$\boldsymbol{w}_i^{(k)} - \boldsymbol{w}_j^{(l)} = 0 \qquad \forall (i,j) \in \mathcal{B}(k,l), k > l. \tag{3.5}$$

The operator $B : W \to U^* := \mathbb{R}^\Lambda$, which realizes the constraints (3.5) in the form $Bw = 0$, is called *jump operator*. Note, the definition of $\mathcal{B}(k,l)$ leads to so-called fully redundant constraints, which implies that, in general, B does not have full rank. The space of all functions in W which belong to the kernel of B is denoted by $\widehat{W}$, and can be identified with $V_{\Gamma,h}$, i.e.

$$\widehat{W} = \{w \in W | \, Bw = 0\} \equiv V_{\Gamma,h}.$$

An illustration of the constraints introduced in (3.5) can be found in Figure 3.1.

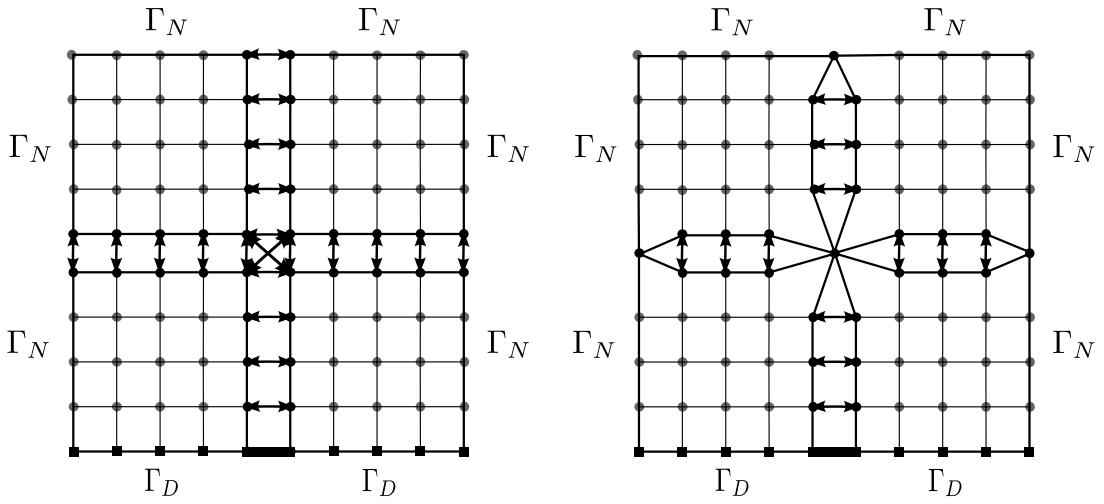


Figure 3.1: Left picture illustrates the fully redundant constraints introduced in (3.5). Right picture presents an illustration of the intermediate space $\widetilde{W}$ in the case of continuity of vertex values (ALG. A).

Remark 3.1. *In [137] a more general form of the jump operator is considered. It enforces constrains of the form*

$$\boldsymbol{w}_i^{(k)} - \sum_{r=1}^{n_i} z_{i,j_r}^{(k,l)} \boldsymbol{w}_j^{(l)} = 0, \qquad \forall (i, j_1, \ldots, j_{n_i}) \in \tilde{\mathcal{B}}(k,l),$$

which also allow non-matching but nested meshes. The set $\tilde{\mathcal{B}}(k,l)$ contains tuples of coupled indices between $\Omega^{(k)}$ and $\Omega^{(l)}$. This situation occurs, if the mesh on one side of the interface is a refinement of the mesh on the other side.

Remark 3.2. *In Chapter 4, we investigate a dG approach to handle non-matching meshes, which are not required to be nested.*

Remark 3.3. *A different approach of incorporating non-matching meshes would be to use mortar methods, see, e.g, [87], [56] and [228]. Instead of enforcing the continuity at the interface in a strong way, it is enforced via a variational equation. Therefore, the jump operator is defined as*

$$\langle Bw, \mu \rangle = \int_\Gamma [\![w]\!] \mu ds,$$

where $[\![w]\!]$ denotes the jump of w across Γ.

3.1.3 Saddle-Point Formulation

Due to the multi-patch structure of our physical domain, we can decompose the bilinear form and the right-hand side functional as follows

$$a(u,v) = \sum_{k=1}^{N} a^{(k)}(u^{(k)}, v^{(k)}), \quad \langle F, v\rangle = \sum_{k=1}^{N} \langle F^{(k)}, v^{(k)}\rangle,$$

where $u, v \in V_h$. By means of the B-Spline basis we can rewrite the variational problem as linear system

$$\left(\sum_{k=1}^{N} \boldsymbol{A}_{\Omega^{(k)}} \boldsymbol{K}^{(k)} \boldsymbol{A}_{\Omega^{(k)}}^{T}\right) \boldsymbol{u} = \sum_{k=1}^{N} \boldsymbol{A}_{\Omega^{(k)}} \boldsymbol{f}^{(k)}, \tag{3.6}$$

where $\boldsymbol{A}_{\Omega^{(k)}}^{T}$ is the Boolean patch-assembling matrix. Analogously to Section 3.1.1, we can reorder the entries of the patch-local stiffness matrix and right-hand side as follows

$$\boldsymbol{K}^{(k)} = \begin{bmatrix} \boldsymbol{K}_{BB}^{(k)} & \boldsymbol{K}_{BI}^{(k)} \\ \boldsymbol{K}_{IB}^{(k)} & \boldsymbol{K}_{II}^{(k)} \end{bmatrix}, \qquad \boldsymbol{f}^{(k)} = \begin{bmatrix} \boldsymbol{f}_{B}^{(k)} \\ \boldsymbol{f}_{I}^{(k)} \end{bmatrix}.$$

Eliminating the local interface dofs, we obtain that (3.6) can be reformulated as

$$\left(\sum_{k=1}^{N} \boldsymbol{A}_{\Gamma^{(k)}} \boldsymbol{S}^{(k)} \boldsymbol{A}_{\Gamma^{(k)}}^{T}\right) \boldsymbol{u}_B = \sum_{k=1}^{N} \boldsymbol{A}_{\Gamma^{(k)}} \boldsymbol{g}^{(k)}, \tag{3.7}$$

where $\boldsymbol{S}^{(k)} = \boldsymbol{K}_{BB}^{(k)} - \boldsymbol{K}_{BI}^{(k)}(\boldsymbol{K}_{II}^{(k)})^{-1}\boldsymbol{K}_{IB}^{(k)}$, $\boldsymbol{g}^{(k)} = \boldsymbol{f}_{B}^{(k)} - \boldsymbol{K}_{BI}^{(k)}(\boldsymbol{K}_{II}^{(k)})^{-1}\boldsymbol{f}_{I}^{(k)}$ and the Boolean matrix $\boldsymbol{A}_{\Gamma^{(k)}}$ is the corresponding assembling matrix for the interfaces. Similarly, we can express (3.7) in operator notation as

$$\sum_{k=1}^{N} \langle S^{(k)} u_B^{(k)}, v_B^{(k)}\rangle = \sum_{k=1}^{N} \langle g^{(k)}, v_B^{(k)}\rangle \quad \forall v_B \in V_{\Gamma,h}, \tag{3.8}$$

where $u_B \in V_{\Gamma,h}$, $g^{(k)} \in W^{(k)^*}$ and $S^{(k)} : W^{(k)} \to W^{(k)^*}$.

In the following, we introduce the Schur complement on the space W and the corresponding right hand side and denote them again by S and g, respectively. Note, in Section 3.1.1 we defined S and g as $S : V_{\Gamma,h} \to V_{\Gamma,h}^*$ and $g \in V_{\Gamma,h}^*$. The more appropriate symbol for them would have been $\widehat{S}$ and $\widehat{g}$, since they can be interpreted as defined on $\widehat{W} \equiv V_{\Gamma,h}$. The same applies for their matrix representations $\boldsymbol{S}$ and $\boldsymbol{g}$. For the purpose of simpler notation in the current introduction of the IETI-DP algorithm, we accept this slight abuse of notation. In Chapter 4, the introduced notation for the dG

version will be more technical and we will be more strict regarding this.

$$S : W \to W^*, \quad \langle Sv, w\rangle := \sum_{k=1}^{N} \langle S^{(k)} v^{(k)}, w^{(k)}\rangle \quad \text{for } v, w \in W,$$

$$g \in W^*, \quad \langle g, w\rangle := \sum_{k=1}^{N} \langle g^{(k)}, w^{(k)}\rangle \quad \text{for } w \in W.$$

Expressed in matrix form, we can write $\boldsymbol{S}$ and $\boldsymbol{g}$ as

$$\boldsymbol{S} := \operatorname{diag}(\boldsymbol{S}^{(k)})_{k=1}^{N}, \quad \boldsymbol{g} := [\boldsymbol{g}^{(k)}]_{k=1}^{N}.$$

The next step is to reformulate (3.8) in terms of S and B in the space W. Due to the symmetry of $a(\cdot,\cdot)$, we can write (3.8) as minimization problem

$$u_B = \operatorname*{argmin}_{v \in V_{\Gamma,h}} \sum_{k=1}^{N} \left(\frac{1}{2} \langle S^{(k)} v^{(k)}, v^{(k)}\rangle - \langle g^{(k)}, v^{(k)}\rangle \right),$$

which can be reformulated as a constraint minimization problem in W

$$u_B = \operatorname*{argmin}_{w \in W, Bw = 0} \frac{1}{2} \langle Sw, w\rangle - \langle g, w\rangle. \tag{3.9}$$

In the following we will only work with the Schur complement system. Hence, to simplify the notation, we will use u instead of u_B. If a distinction between u, u_B and u_I is necessary, we will write the subscript.

Problem (3.9) can be rewritten as the following saddle-point problem: find $(u, \lambda) \in W \times U$ such that

$$\begin{bmatrix} S & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} u \\ \lambda \end{bmatrix} = \begin{bmatrix} g \\ 0 \end{bmatrix}, \tag{3.10}$$

where $U := \mathbb{R}^{|\Lambda|}$ and $|\Lambda|$ are the number of Lagrange multipliers, i.e., the number of rows of B. We immediately observe the following result, cf. Lemma 2.11 in [183].

Lemma 3.4. *If $ker(S) \cap ker(B) = \{0\}$, then the above problem is uniquely solvable up to adding elements from $ker(B^T)$ to λ.*

Remark 3.5. *We note that not all $S^{(k)}$ are regular, since those which do not lie on a Dirichlet boundary correspond to pure Neumann problems. The usual strategy for solving* (3.10) *is to work with its Schur complement, but since some blocks of S are singular, the Schur complement is not well defined. Considering the classical FETI method, one adds the basis of each local kernel to the space and regularizes the matrix. Unfortunately, to do this one needs exact knowledge of the kernels, which is in general not trivial. The dual-primal approach presented below circumvents this issue by restricting the space W to ensure that S is invertible.*

3.1.4 Intermediate Space and Primal Constraints

In order to guarantee the positive definiteness of S, we are looking for an intermediate space $\widetilde{W}$ in the sense $\widehat{W} \subset \widetilde{W} \subset W$ such that S restricted to $\widetilde{W}$ is SPD. Let $\Psi \subset V_{\Gamma,h}^*$ be a set of linearly independent functionals, called *primal variables*. Then we define the spaces

$$\widetilde{W} := \{w \in W : \forall \psi \in \Psi : \psi(w^{(k)}) = \psi(w^{(l)}),\ k, l \in \{1, \dots, N\} \text{ with } k > l\},$$
$$W_\Delta := \prod_{k=1}^{N} W_\Delta^{(k)}, \quad \text{where } W_\Delta^{(k)} := \{w^{(k)} \in W^{(k)} : \forall \psi \in \Psi : \psi(w^{(k)}) = 0\}.$$

Moreover, we introduce the space $W_\Pi \subset \widehat{W}$, such that

$$\widetilde{W} = W_\Pi \oplus W_\Delta.$$

We call W_Π *primal space* and W_Δ *dual space*. If we choose Ψ, such that $\widetilde{W} \cap \ker(S) = \{0\}$, then

$$\widetilde{S} : \widetilde{W} \to \widetilde{W}^*, \quad \langle \widetilde{S}v, w\rangle = \langle Sv, w\rangle \quad \forall v, w \in \widetilde{W}$$

is invertible. In the following, we will always assume that such a Ψ is chosen.

In the literature, there are the following typical choices for the primal variables ψ:

- Vertex evaluation: $\psi^{\mathcal{V}}(v) = v(\mathcal{V})$,
- Edge averages: $\psi^{\mathcal{E}}(v) = \frac{1}{|\mathcal{E}|} \int_{\mathcal{E}} v \, ds$,
- Face averages: $\psi^{\mathcal{F}}(v) = \frac{1}{|\mathcal{F}|} \int_{\mathcal{F}} v \, ds$.

The typical choices for Ψ are usually called Algorithm A – C:

- Algorithm A: $\Psi = \{\psi^{\mathcal{V}}\}$,
- Algorithm B: $\Psi = \{\psi^{\mathcal{V}}\} \cup \{\psi^{\mathcal{E}}\} \cup \{\psi^{\mathcal{F}}\}$,
- Algorithm C: $\Psi = \{\psi^{\mathcal{V}}\} \cup \{\psi^{\mathcal{E}}\}$.

Moreover, in literature one finds references to two further choices for Ψ, commonly referred to as Algorithm D and E, which are aiming for a reduced set of primal variables, see, e.g., Algorithm 6.28 and 6.29 in [215]. These algorithms address the issue of the rapidly increasing number of primal variables. The space $\widetilde{W}$ for the case of having continuity of the vertex values is illustrated in Figure 3.1.

Remark 3.6. *For domains $\Omega \subset \mathbb{R}^2$, Algorithm A will provide a quasi optimal method for the Poisson problem. By choosing additional primal variables, the coarse problem will grow. Hence, it becomes computationally more demanding, but will reduce the condition number. For three-dimensional domains, it can be shown that just choosing vertex evaluation does not lead to a quasi optimal method. One obtains a bound*

$O(H/h(1+\log(H/h))^2)$, see Remark 6.39 in [215], which is also observed in the numerical examples, see Section 3.4. In such cases, additional primal variables have to be chosen.

3.1.5 IETI - DP

Since $\widetilde{W} \subset W$, there is a natural embedding $\widetilde{I} : \widetilde{W} \to W$. We define the jump operator restricted to $\widetilde{W}$ as $\widetilde{B} := B\widetilde{I} : \widetilde{W} \to U^*$. Then we can formulate the saddle-point problem in $\widetilde{W}$ as follows: find $(u, \lambda) \in \widetilde{W} \times U$:

$$\begin{bmatrix} \widetilde{S} & \widetilde{B}^T \\ \widetilde{B} & 0 \end{bmatrix} \begin{bmatrix} u \\ \lambda \end{bmatrix} = \begin{bmatrix} \widetilde{g} \\ 0 \end{bmatrix}, \tag{3.11}$$

where $\widetilde{g} := \widetilde{I}^T g$, and $\widetilde{B}^T = \widetilde{I}^T B^T$. Here, $\widetilde{I}^T : W^* \to \widetilde{W}^*$ denotes the adjoint of $\widetilde{I}$.

By construction, $\widetilde{S}$ is SPD on $\widetilde{W}$. Hence, we can define the Schur complement F and the corresponding right-hand side d of equation (3.11) as follows:

$$F := \widetilde{B}\widetilde{S}^{-1}\widetilde{B}^T, \quad d := \widetilde{B}\widetilde{S}^{-1}\widetilde{g}.$$

Finally, we can formulate the following problem for the Lagrange multipliers

$$\text{find } \lambda \in U : \quad F\lambda = d. \tag{3.12}$$

By means of Brezzi's theorem, we obtain the following result, cf. Lemma 5.9 in [183].

Lemma 3.7. *The above problem is uniquely solvable up to adding elements from $ker(\widetilde{B}^T)$ to λ. The unique solution*

$$u = \widetilde{S}^{-1}(\widetilde{g} - \widetilde{B}^T\lambda)$$

satisfies $u \in \widehat{W} \equiv V_{\Gamma,h}$ and is the solution of (3.4).

We note that F is symmetric and positive semi-definite on U. According to [183], if we set

$$\widetilde{U} := U_{/\ker(\widetilde{B}^T)}, \quad \widetilde{U}^* := \mathcal{R}(\widetilde{B}),$$

where $\widetilde{U}^*$ is in fact the dual of $\widetilde{U}$, then F restricted to $\widetilde{U}$ is SPD, i.e. $F_{|\widetilde{U}} : \widetilde{U} \to \widetilde{U}^*$. Hence, it is possible to solve (3.12) with the PCG method.

3.1.6 Preconditioning

Since the condition number of F is not optimal with respect to h and H, there is need for preconditioning. The classical choice is the so-called *scaled Dirichlet preconditioner*. It is known for FE that one obtains a quasi-optimal condition number bound of the preconditioned system in terms of the ratio of subdomain diameter over mesh-size, i.e., H/h. Moreover, the condition number is robust with respect to jumping diffusion coefficients. The scaled Dirichlet preconditioner has the following form

$$M_{sD}^{-1} := B_D S B_D^T, \tag{3.13}$$

where B_D is a scaled version of B, which enforces the constraints:

$$\begin{aligned} &\delta_j^{\dagger(l)} \boldsymbol{w}_i^{(k)} - \delta_i^{\dagger(k)} \boldsymbol{w}_j^{(l)} = 0 \qquad \forall (i,j) \in \mathcal{B}(k,l), k > l, \\ &\text{with } \delta_i^{\dagger(k)} = \frac{\rho_i^{(k)}}{\sum_l \rho_{j_l}^{(l)}}, \end{aligned}$$

where j_l is the corresponding coefficient index on the neighbouring patch $\Omega^{(l)}$. We note that this preconditioner just uses the block diagonal version of the Schur complement. Therefore, the application can be performed in parallel. Moreover, it can be shown that the scaled Dirichlet preconditioner is SPD on $\widetilde{U}^*$, provided that Ψ are chosen as in Section 3.1.4, see Lemma 5.14 in [183]. This justifies the use as a preconditioner in the PCG algorithm.

Typical choices for $\rho_i^{(k)}$ are

- Multiplicity Scaling: $\rho_i^{(k)} = 1$,
- Coefficient Scaling: If $\alpha(x)_{|\Omega^{(k)}} = \alpha^{(k)}$, choose $\rho_i^{(k)} = \alpha^{(k)}$,
- Stiffness Scaling: $\rho_i^{(k)} = \boldsymbol{K}_{i,i}^{(k)}$.

Remark 3.8. *According to [19], we use a modified version of the stiffness scaling for the analysis. In this modified version, we make the choice $\rho_i^{(k)} = \boldsymbol{K}_{r,r}^{(k)}$, where $\boldsymbol{K}_{r,r}^{(k)}$ is one representative of the values $\{\boldsymbol{K}_{i,i}^{(k)}\}_{i \in \mathcal{I}_B}$. Often these values are very similar on one patch, which is due to the tensor-product structure of B-Splines and the constant material value on a patch. Numerical experiments show only minor differences between the original stiffness scaling as introduced above and the modified one. In the numerical experiments in Section 3.4, the original stiffness scaling is used, whereas for the condition number estimate in Section 3.3, we consider the modified one.*

If the diffusion coefficient α is constant and identical on each patch, then the multiplicity and the coefficient scaling are the identical. If the diffusion coefficient has a jump at the patch interface, the coefficient scaling is preferable.

Remark 3.9. *An alternative and under certain circumstances more efficient version is the so-called* lumped Dirichlet preconditioner*, see Section 2.2.4.3 in [183] and [64] for the classical FETI method. Instead of applying the Schur complement S, one just uses K_{BB}. The preconditioner is then given by $M_{sD}^{-1} = B_D K_{BB} B_D^T$, This circumvents the need for solving a system with K_{II}, but leads to an increased condition number.*

Theorem 3.10. *Let H_k be the diameter, h_k the local mesh-size of $\Omega^{(k)}$, and M_{sD}^{-1} the scaled Dirichlet preconditioner as defined in* (3.13)*. Then, under Assumption 2, we have*

$$\kappa(M_{sD}^{-1} F_{\widetilde{U}}) \leq C \max_k \left(1 + \log\left(\frac{H_k}{h_k}\right)\right)^2,$$

where the positive constant C is independent of h and H.

The proof will be given in Section 3.3. In the case of IgA, a more general proof in the sense that not only C^0 smoothness across patch interfaces is allowed but also C^l, $l \geq 0$, smoothness, can be found in [19]. However, the proof is restricted to the case of a domain decomposition that is obtained by subdividing a single patch, i.e., performing a decomposition of the parameter domain. Hence, always the same geometrical mapping G is used. Furthermore, due to the C^l, $l \geq 0$, smoothness across interfaces, only a condition number bound of $O\left((1 + \log H/h)H/h\right)$ could be proven for stiffness scaling. In Section 3.3, we will extend the proof given in [19] to multi-patch domains, which consists of different geometrical mappings $G^{(k)}$ for each patch. Additionally, for $l = 0$, we also obtain quasi-optimal condition number bounds for the stiffness scaling.

Remark 3.11. *The condition number bound given in Theorem 3.10 does not cover the dependence on the B-Spline degree p. However, the numerical experiments performed in Section 3.4.4 indicate that the condition number of the preconditioned IETI-DP operator depends only logarithmically or at most linearly on p. For spectral finite elements a logarithmic dependence of the condition number on the degree is known, see the discussion in [181], [132] and [125] and references therein.*

3.1.7 BDDC - Preconditioner

The balancing domain decomposition by constraints (BDDC) preconditioner is closely related to the IETI-DP method, see, e.g., [42]. It was shown in [159] that the spectra of the two operators are identical up to zeros and ones. For completeness we give in this section a short overview of the construction of the BDDC preconditioner, where we follow the lines in [183]. We start from the equation

$$\widehat{S}\widehat{u} = \widehat{g}, \tag{3.14}$$

where the notation $\widehat{\cdot}$ indicates that the operator and the functions are restricted to the continuous space $V_{\Gamma,h}$, i.e., (3.14) is the standard Schur complement system. Since $\widehat{S}$ is

SPD, we can solve system (3.14) by the PCG method preconditioned by a symmetric positive definite preconditioner, which will be the BDDC preconditioner M^{-1}_{BDDC}.

As in the previous sections, we use the same set of linearly independent primal variables Ψ and the corresponding spaces. We then define the BDDC preconditioner as follows

$$M^{-1}_{BDDC} := \widetilde{E}_D \widetilde{S}^{-1} \widetilde{E}^T_D,$$

where $\widetilde{E}^T_D$ is defined via the formulas

$$\begin{aligned}
\widetilde{E}_D &= E_D \widetilde{I} : \quad \widetilde{W} \to \widehat{W}, \\
E_D &= I - P_D : \quad W \to \widehat{W}, \\
P_D &= B^T_D B : \quad W \to W.
\end{aligned}$$

For more details we refer the reader to [183] and references therein. One can give an alternative formulation see, e.g., [19]. Here we assume that, after a change of basis, each primal variable corresponds to one degree of freedom, see, e.g., [135]. Let $\boldsymbol{K}^{(k)}$ be the stiffness matrix corresponding to $\Omega^{(k)}$. We split the degrees of freedom into interior (I) and interface (B) ones. Furthermore, we again split the interface degrees of freedoms into primal (Π) and dual (Δ) ones. This provides a partition of $\boldsymbol{K}^{(k)}$ into 2×2 and 3×3 block systems:

$$\boldsymbol{K}^{(k)} = \begin{bmatrix} \boldsymbol{K}^{(k)}_{II} & \boldsymbol{K}^{(k)}_{IB} \\ \boldsymbol{K}^{(k)}_{BI} & \boldsymbol{K}^{(k)}_{BB} \end{bmatrix} = \begin{bmatrix} \boldsymbol{K}^{(k)}_{II} & \boldsymbol{K}^{(k)}_{I\Delta} & \boldsymbol{K}^{(k)}_{I\Pi} \\ \boldsymbol{K}^{(k)}_{\Delta I} & \boldsymbol{K}^{(k)}_{\Delta\Delta} & \boldsymbol{K}^{(k)}_{\Delta\Pi} \\ \boldsymbol{K}^{(k)}_{\Pi I} & \boldsymbol{K}^{(k)}_{\Pi\Delta} & \boldsymbol{K}^{(k)}_{\Pi\Pi} \end{bmatrix}.$$

In order to the define the preconditioner, we need the following restriction and interpolation operators:

$$\begin{aligned}
R_{B\Delta} &: \widetilde{W} \to W_\Delta, \qquad & R^{(k)}_\Delta &: W_\Delta \to W^{(k)}_\Delta, \\
R_{B\Pi} &: \widetilde{W} \to W_\Pi, \qquad & R^{(k)}_\Pi &: W_\Pi \to W^{(k)}_\Pi.
\end{aligned} \tag{3.15}$$

We define a scaled version $R^{(k)}_{D,\Delta}$ of $R^{(k)}_\Delta$ by multiplying its i-th row by $\delta^{\dagger(k)}_i$, and then we define

$$R_{D,B} := R_{B\Pi} \oplus \left(\sum_{k=1}^{N} R^{(k)}_{D,\Delta} \right) R_{B\Delta}. \tag{3.16}$$

The BDDC preconditioner is determined by

$$M^{-1}_{BDDC} := R^T_{D,B} \widetilde{S}^{-1} R_{D,B},$$

where

$$\widetilde{S}^{-1} = R_{B\Delta}^T \left(\sum_{k=1}^{N} \begin{bmatrix} 0 & R_\Delta^{(k)^T} \end{bmatrix} \begin{bmatrix} \boldsymbol{K}_{II}^{(k)} & \boldsymbol{K}_{I\Delta}^{(k)} \\ \boldsymbol{K}_{\Delta I}^{(k)} & \boldsymbol{K}_{\Delta\Delta}^{(k)} \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ R_\Delta^{(k)} \end{bmatrix} \right) R_{B\Delta} + \Phi \boldsymbol{S}_{\Pi\Pi}^{-1} \Phi^T.$$

Here the matrices $\boldsymbol{S}_{\Pi\Pi}$ and Φ are given by

$$\boldsymbol{S}_{\Pi\Pi} = \sum_{k=1}^{N} R_\Pi^{(k)^T} \left(\boldsymbol{K}_{\Pi\Pi}^{(k)} - \begin{bmatrix} \boldsymbol{K}_{\Pi I}^{(k)} & \boldsymbol{K}_{\Pi\Delta}^{(k)} \end{bmatrix} \begin{bmatrix} \boldsymbol{K}_{II}^{(k)} & \boldsymbol{K}_{I\Delta}^{(k)} \\ \boldsymbol{K}_{\Delta I}^{(k)} & \boldsymbol{K}_{\Delta\Delta}^{(k)} \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{K}_{I\Pi}^{(k)} \\ \boldsymbol{K}_{\Delta\Pi}^{(k)} \end{bmatrix} \right) R_\Pi^{(k)}$$

and

$$\Phi = R_{B\Pi}^T - R_{B\Delta}^T \sum_{k=1}^{N} \left(\begin{bmatrix} 0 & R_\Delta^{(k)^T} \end{bmatrix} \begin{bmatrix} \boldsymbol{K}_{II}^{(k)} & \boldsymbol{K}_{I\Delta}^{(k)} \\ \boldsymbol{K}_{\Delta I}^{(k)} & \boldsymbol{K}_{\Delta\Delta}^{(k)} \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{K}_{I\Pi}^{(k)} \\ \boldsymbol{K}_{\Delta\Pi}^{(k)} \end{bmatrix} \right) R_\Pi^{(k)},$$

respectively.

3.2 Implementation of the IETI-DP method

For the presentation of the implementation, we follow the approach presented in Section 5.3 in [183]. Since F is symmetric and at least positive semi-definite and positive definite on $\widetilde{U}$, we can solve the linear system $F\lambda = d$ of the algebraic equations by means of the PCG algorithm, where we use M_{sD}^{-1} as preconditioner. For completeness, the PCG method is summarized in Algorithm 1.

Algorithm 1 PCG method with initial guess λ_0,

λ_0 given
$r_0 = d - F\lambda_0, \quad k = 0, \quad \beta_{-1} = 0$
repeat
  $s_k = M_{sD}^{-1} r_k$
  $\beta_{k-1} = \frac{(r_k, s_k)}{(r_{k-1}, s_{k-1})}$
  $p_k = s_k + \beta_{k-1} p_{k-1}$
  $\alpha_k = \frac{(r_k, s_k)}{(Fp_k, p_k)}$
  $\lambda_{k+1} = r_k + \alpha_k p_k$
  $r_{k+1} = r_k - \alpha_k F p_k$
  $k = k + 1$
until stopping criterion fulfilled

It is very expensive to build up the matrix representation of F and M_{sD}^{-1}. Fortunately, the PCG algorithm only requires the application of the matrix to a vector. Hence, we

want to present a way to efficiently apply F and M_{sD}^{-1} without calculating their matrix form. The challenging part is the application of $\widetilde{S}^{-1}$, which is part of F. The idea is to split $\widetilde{W}$ into $W_\Pi \oplus W_\Delta$, such that $W_\Pi \perp_S W_\Delta$. For a more detailed discussion, we refer the reader to Section 5.3 in [183].

3.2.1 Choosing a Basis for W_Π

The first step is to provide an appropriate space W_Π and its basis $\{\widetilde{\phi}_j\}_{j=1}^{n_\Pi}$, where n_Π is the dimension of W_Π, i.e., the number of primal variables. We require that this basis is nodal with respect to the primal variables, i.e.,

$$\psi_i(\widetilde{\phi}_j) = \delta_{i,j}, \quad \forall i,j \in \{1,\dots,n_\Pi\}.$$

Additionally, we require that

$$\widetilde{\phi}_{j|\Omega^{(k)}} = 0 \quad \text{if } \psi_j \text{ is not associated to } \Omega^{(k)},$$

i.e., the basis has a local support in a certain sense.

There are many choices for the subspace W_Π. Following the approach presented in [183], we will choose that one which is orthogonal to W_Δ with respect to S, i.e.,

$$\langle Sw_\Pi, w_\Delta\rangle = 0, \quad \forall w_\Pi \in W_\Pi, w_\Delta \in W_\Delta.$$

This choice, which will simplify the application of $\widetilde{S}^{-1}$ significantly, is known as *energy minimizing primal subspace* in the literature, cf. [183] and [42]. In order to find such a basis, we define the constraint matrix $C^{(k)} : W^{(k)} \to \mathbb{R}^{n_\Pi^{(k)}}$ for each patch $\Omega^{(k)}$ which realizes the action of the primal variables:

$$[C^{(k)}v]_j = \psi_{i(k,j)}(v) \quad \forall v \in W \, \forall j \in \{1,\dots,n_\Pi^{(k)}\},$$

where $n_\Pi^{(k)}$ are the number of primal variables associated with $\Omega^{(k)}$ and $i(k,j)$ the global index of the j-th primal variable on $\Omega^{(k)}$. Note that a function $w_\Delta^{(k)} \in W_\Delta^{(k)}$ is in the kernel of $C^{(k)}$, i.e., $C^{(k)}w_\Delta^{(k)} = 0$.

For each patch $\Omega^{(k)}$, the basis functions $\{\widetilde{\phi}_j^{(k)}\}_{j=1}^{n_\Pi^{(k)}}$ of $W_\Pi^{(k)}$ are then given by the solution of the system

$$\begin{bmatrix} S^{(k)} & C^{(k)T} \\ C^{(k)} & 0 \end{bmatrix} \begin{bmatrix} \widetilde{\phi}_j^{(k)} \\ \widetilde{\boldsymbol{\mu}}_j^{(k)} \end{bmatrix} = \begin{bmatrix} 0 \\ \boldsymbol{e}_j^{(k)} \end{bmatrix}, \quad \forall j \in \{1,\dots,n_\Pi^{(k)}\}, \tag{3.17}$$

where $\boldsymbol{e}_j^{(k)} \in \mathbb{R}^{n_\Pi^{(k)}}$ is the j-th unit vector. The Lagrange multipliers $\widetilde{\boldsymbol{\mu}}_j^{(k)} \in \mathbb{R}^{n_\Pi^{(k)}}$ will become important later on. By means of $\ker(S) \cap \widetilde{W} = \{0\}$, it is easy to see that this

system has a unique solution. Due to the fact that building the Schur complement $S^{(k)}$ is not efficient, we use an equivalent formulation by means of $K^{(k)}$:

$$\begin{bmatrix} K_{BB}^{(k)} & K_{BI}^{(k)} & {C^{(k)}}^T \\ K_{IB}^{(k)} & K_{II}^{(k)} & 0 \\ C^{(k)} & 0 & 0 \end{bmatrix} \begin{bmatrix} \widetilde{\phi}_j^{(k)} \\ \cdot \\ \widetilde{\boldsymbol{\mu}}_j^{(k)} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \boldsymbol{e}_j^{(k)} \end{bmatrix}. \tag{3.18}$$

For each patch $\Omega^{(k)}$, the sparse LU factorization of this matrix is computed and stored.

Remark 3.12. *A different approach would be to construct a basis for this space by means of a basis transformation, such that to each primal variable there is an associated basis function. This approach has been studied, e.g., in [129] and [135].*

3.2.2 Application of $\widetilde{S}^{-1}$

Assume that $f := \{\boldsymbol{f}_\Pi, \{f_\Delta^{(k)}\}\} \in \widetilde{W}^*$ is already given. We are now looking for $w := \{\boldsymbol{w}_\Pi, \{w_\Delta^{(k)}\}\} \in \widetilde{W}$ such that $w = \widetilde{S}^{-1} f$. Let $S_{\Pi\Pi}, S_{\Delta\Pi}, S_{\Pi\Delta}, S_{\Delta\Delta}$ be the restrictions of $\widetilde{S}$ to the corresponding subspaces, i.e.,

$$\begin{aligned} \langle S_{\Pi\Pi} v_\Pi, w_\Pi \rangle &= \langle \widetilde{S} v_\Pi, w_\Pi \rangle && \text{for } v_\Pi, w_\Pi \in W_\Pi, \\ \langle S_{\Pi\Delta} v_\Pi, w_D \rangle &= \langle \widetilde{S} v_\Pi, w_D \rangle && \text{for } v_\Pi \in W_\Pi, w_D \in W_\Delta, \\ \langle S_{\Delta\Delta} v_D, w_D \rangle &= \langle \widetilde{S} v_D, w_D \rangle && \text{for } v_D, w_D \in W_\Delta, \end{aligned}$$

and $S_{\Delta\Pi} = S_{\Pi\Delta}^T$. We note that $S_{\Delta\Delta}$ can be seen as a block diagonal operator, i.e., $S_{\Delta\Delta} = \operatorname{diag}(S_{\Delta\Delta}^{(k)})$. Due to our special choice $\widetilde{W}_\Pi := \widetilde{W}_\Delta^{\perp_S}$, we have $S_{\Delta\Pi} = S_{\Pi\Delta} = 0$. Based on this splitting, we obtain the block forms

$$\widetilde{S} = \begin{bmatrix} \boldsymbol{S}_{\Pi\Pi} & 0 \\ 0 & S_{\Delta\Delta} \end{bmatrix} \text{ and } \widetilde{S}^{-1} = \begin{bmatrix} \boldsymbol{S}_{\Pi\Pi}^{-1} & 0 \\ 0 & S_{\Delta\Delta}^{-1} \end{bmatrix}.$$

Therefore, the application of $\widetilde{S}^{-1}$ reduces to an application of one global coarse problem involving $\boldsymbol{S}_{\Pi\Pi}^{-1}$ and N local problems involving ${S_{\Delta\Delta}^{(k)}}^{-1}$, i.e.,

$$\boldsymbol{w}_\Pi = \boldsymbol{S}_{\Pi\Pi}^{-1} \boldsymbol{f}_\Pi, \quad \text{and} \quad w_\Delta^{(k)} = {S_{\Delta\Delta}^{(k)}}^{-1} f_\Delta^{(k)} \quad \forall k = 1, \ldots, N.$$

Application of ${S_{\Delta\Delta}^{(k)}}^{-1}$: The application of ${S_{\Delta\Delta}^{(k)}}^{-1}$ corresponds to solving a local Neumann problem in the space W_Δ, i.e., $S^{(k)} w^{(k)} = f_\Delta^{(k)}$, with the constraint $C^{(k)} w^{(k)} = 0$. This problem can be rewritten as the following saddle-point problem

$$\begin{bmatrix} S^{(k)} & {C^{(k)}}^T \\ C^{(k)} & 0 \end{bmatrix} \begin{bmatrix} w^{(k)} \\ \cdot \end{bmatrix} = \begin{bmatrix} f_\Delta^{(k)} \\ 0 \end{bmatrix}.$$

The same method as used in (3.17) for rewriting that equation in terms of K applies here, and the LU factorization of the matrix is already available.

Application of $\boldsymbol{S}_{\Pi\Pi}{}^{-1}$: The matrix $\boldsymbol{S}_{\Pi\Pi}$ can be assembled from the patch-local matrices $\boldsymbol{S}^{(k)}_{\Pi\Pi}$. Let $\{\widetilde{\phi}^{(k)}_j\}_{j=1}^{n^{(k)}_\Pi}$ be the basis of $W^{(k)}_\Pi$. In order to assemble $\boldsymbol{S}^{(k)}_{\Pi\Pi}$, in general, we have to compute its entries via

$$[\boldsymbol{S}^{(k)}_{\Pi\Pi}]_{i,j} = \langle S^{(k)}\widetilde{\phi}^{(k)}_i, \widetilde{\phi}^{(k)}_j\rangle, \quad i,j \in \{1,\dots,n^{(k)}_\Pi\}.$$

The construction of $\{\widetilde{\phi}^{(k)}_j\}_{j=1}^{n^{(k)}_\Pi}$ in (3.18) provides

$$\begin{aligned}[\boldsymbol{S}^{(k)}_{\Pi\Pi}]_{i,j} &= \langle S^{(k)}\widetilde{\phi}^{(k)}_i, \widetilde{\phi}^{(k)}_j\rangle = -\langle {C^{(k)}}^T\widetilde{\boldsymbol{\mu}}^{(k)}_i, \widetilde{\phi}^{(k)}_j\rangle = -(\widetilde{\boldsymbol{\mu}}^{(k)}_i, C^{(k)}\widetilde{\phi}^{(k)}_j)_{\ell_2}\\ &= -(\widetilde{\boldsymbol{\mu}}^{(k)}_i, \boldsymbol{e}^{(k)}_j)_{\ell_2} = -[\widetilde{\boldsymbol{\mu}}^{(k)}_i]_j.\end{aligned}$$

Therefore, we can reuse the Lagrange multipliers $\widetilde{\boldsymbol{\mu}}^{(k)}_i$ obtained in (3.18), and assemble $\boldsymbol{S}^{(k)}_{\Pi\Pi}$ from them. Once $\boldsymbol{S}_{\Pi\Pi}$ is assembled, the sparse LU factorization can be calculated and stored.

3.2.3 Application of $\widetilde{I}$ and $\widetilde{I}^T$

The last building block is the embedding $\widetilde{I} : \widetilde{W} \to W$ and its adjoint $\widetilde{I}^T : W^* \to \widetilde{W}^*$. Recall the direct splitting $W^{(k)} = W^{(k)}_\Delta \oplus W^{(k)}_\Pi$. Let us denote by $\Phi^{(k)} = [\widetilde{\phi}^{(k)}_1, \dots, \widetilde{\phi}^{(k)}_{n^{(k)}_\Pi}]$ the basis of $W^{(k)}_\Pi$. Given the primal part $\boldsymbol{w}_\Pi$, we obtain its restriction to $\Omega^{(k)}$ via an appropriately defined restriction matrix $\boldsymbol{R}^{(k)}$. Following the lines in Section 5.3.3 of [183], we can formulate the operator $\widetilde{I} : \widetilde{W} \to W$ as

$$\begin{bmatrix}\boldsymbol{w}_\Pi\\ w_\Delta\end{bmatrix} \mapsto w := \Phi\boldsymbol{R}\boldsymbol{w}_\Pi + w_\Delta,$$

where Φ and $\boldsymbol{R}$ are block versions of $\Phi^{(k)}$ and $\boldsymbol{R}^{(k)}$, respectively. The second function is its adjoint operation $\widetilde{I}^T : W^* \to \widetilde{W}^*$. It can be realized in the following way

$$f \mapsto \begin{bmatrix}\boldsymbol{f}_\Pi\\ f_\Delta\end{bmatrix} = \begin{bmatrix}\boldsymbol{A}\Phi^T\boldsymbol{f}\\ f - C^T\Phi^T f\end{bmatrix},$$

where $\boldsymbol{A}$ is the corresponding assembling operator to $\boldsymbol{R}$, i.e., $\boldsymbol{A} = \boldsymbol{R}^T$. For a more extensive discussion and derivation we refer the reader to [183].

3.2.4 Application of the Preconditioner

The application of the scaled Dirichlet preconditioner $M^{-1}_{sD} = B_D S B^T_D$ is basically given by the application of S where

$$S = \operatorname{diag}(S^{(k)}), \quad \text{with} \quad S^{(k)} = K^{(k)}_{BB} - K^{(k)}_{BI}(K^{(k)}_{II})^{-1}K^{(k)}_{IB}.$$

The calculation of $v^{(k)} = S^{(k)}w^{(k)}$ consists of 2 steps:

1. Solve: $K_{II}^{(k)} x^{(k)} = -K_{IB}^{(k)} w^{(k)}$ (*Dirichlet problem*)
2. $v^{(k)} = K_{BB}^{(k)} w^{(k)} + K_{BI}^{(k)} x^{(k)}$

Again, the LU factorization of $K_{II}^{(k)}$ can be computed in advance and stored.

3.2.5 Summary for the Application of F and M_{sD}^{-1}

We now summarize the application of F and M_{sD}^{-1} to vectors in Algorithm 2 and Algorithm 3, respectively. In the previous sections, we have not specifically discussed the application of B and B^T. These matrices have entries $-1, 0$ and 1. In the case of more general jump operators, as described in Remark 3.1 and Remark 3.3, they can take different values. Usually, not the matrix is stored, but only a function that performs the application is provided.

Algorithm 2 Algorithm for calculating $\nu = F\lambda$ for given $\lambda \in U$

procedure $F(\lambda)$
  Application of $B^T : \{f^{(k)}\}_{k=1}^N = B^T \lambda$
  Application of $\widetilde{I}^T : \{\boldsymbol{f}_\Pi, \{f_\Delta^{(k)}\}_{k=1}^N\} = \widetilde{I}^T \left(\{f^{(k)}\}_{k=1}^N\right)$
  Application of $\widetilde{S}^{-1}$:
  Begin
    $\boldsymbol{w}_\Pi = \boldsymbol{S}_{\Pi\Pi}^{-1} \boldsymbol{f}_\Pi$
    $w_\Delta^{(k)} = {S_{\Delta\Delta}^{(k)}}^{-1} f_\Delta^{(k)} \quad \forall k = 1, \ldots, N$
  End
  Application of $\widetilde{I} : \{w^{(k)}\}_{k=1}^N = \widetilde{I}\left(\{\boldsymbol{w}_\Pi, \{w_\Delta^{(k)}\}_{k=1}^N\}\right)$
  Application of $B : \nu = B\left(\{w^{(k)}\}_{k=1}^N\right)$
end procedure

Algorithm 3 Algorithm for calculating $\nu = M_{sD}^{-1}\lambda$ for given $\lambda \in U$

procedure $M_{sD}^{-1}(\lambda)$
  Application of $B_D^T : \{w^{(k)}\}_{k=1}^N = B_D^T \lambda$
  Application of S_e :
  Begin
    Solve $K_{II}^{(k)} x^{(k)} = -K_{IB}^{(k)} w^{(k)} \quad \forall k = 1, \ldots, N$
    $v^{(k)} = K_{BB}^{(k)} w^{(k)} + K_{BI}^{(k)} x^{(k)}. \quad \forall k = 1, \ldots, N$
  End
  Application of $B_D : \nu = B_D\left(\{v^{(k)}\}_{k=1}^N\right)$
end procedure

3.3 Analyzing the Condition Number

In this section we rephrase the results and notations established in [19], and extend them to multi-patch domains, consisting of a different geometrical mapping $G^{(k)}$ for each patch. However, we only allow C^0 smoothness across the patch interfaces, and restrict our analysis to the 2d case of Algorithm A and having a globally constant diffusion coefficient.

3.3.1 General Results

Let u be a function from V_h. Its restriction to a patch $\Omega^{(k)}$ belongs to $V_h^{(k)}$, and can be written as

$$u^{(k)} := u|_{\Omega^{(k)}} = \sum_{i \in \mathcal{I}^{(k)}} \boldsymbol{u}_i^{(k)} N_{i,p}^{(k)},$$

where $\mathcal{I}^{(k)}$ contains all indices corresponding to basis functions that have no support on the Dirichlet boundary in the physical space $\Omega^{(k)}$. The corresponding spline function in the parameter space is denoted by $\hat{u}^{(k)} \in \mathrm{S}_h^{(k)}$. It is important to note that the geometrical map $G^{(k)}$ and its inverse ${G^{(k)}}^{-1}$ are independent of h_k, since it is fixed on a coarse discretization. When the basis becomes refined, $G^{(k)}$ stays the same. Clearly, the same applies for the gradients.

We will now define a local discrete semi-norm based on the coefficients $\boldsymbol{u}_i$, where we refer to [19] for a motivation and a more detailed discussion.

Definition 3.13. *Let $u \in V_h^{(k)}$, and $\hat{u}$ be its counterpart in the parameter domain. Then we define the discrete semi-norm*

$$|\hat{u}|_{k,\nabla}^2 := \sum_{i \in \mathcal{I}_1^{(k)}} |\boldsymbol{u}_{(i_1,i_2)}^{(k)} - \boldsymbol{u}_{(i_1-1,i_2)}^{(k)}|^2 + \sum_{i \in \mathcal{I}_2^{(k)}} |\boldsymbol{u}_{(i_1,i_2)}^{(k)} - \boldsymbol{u}_{(i_1,i_2-1)}^{(k)}|^2,$$

where for $i \in \mathcal{I}_1^{(k)}, \mathcal{I}_2^{(k)} \subset \mathcal{I}^{(k)}$ are defined such that $\boldsymbol{u}_{(i_1-1,i_2)}^{(k)}$ and $\boldsymbol{u}_{(i_1,i_2-1)}^{(k)}$ are well defined, respectively.

Proposition 3.14. *Let $u \in V_h^{(k)}$ and $\hat{u}$ its counterpart in the parameter domain. Then*

$$|\hat{u}|_{\nabla}^2 \approx |\hat{u}|_{H^1((0,1)^2)}^2 \approx |u|_{H^1(\Omega^{(k)})}^2$$

holds, where the hidden constants are independent of h_k and H_k.

Proof. The proof follows from the equivalence of the norms in the parameter and physical domain, see Corollary 2.12, Assumption 3 and Proposition 5.2 in [19] □

The second step is to provide properties in the local index spaces. Since we consider only the two-dimensional problem, we can interpret the coefficients $(\boldsymbol{u}_i)_{i\in\mathcal{I}^{(k)}}$ as entries of a matrix $\boldsymbol{C}^{(k)} = (\boldsymbol{u}_i)_{i\in\mathcal{I}^{(k)}} \in \mathcal{R}^{(k)} := \mathbb{R}^{M_1^{(k)}\times M_2^{(k)}}$, where $M_\iota^{(k)}$ are the number of basis functions on $\Omega^{(k)}$ along dimension ι. When applying the semi-norm $|\cdot|_\nabla$ to the coefficient matrix $\boldsymbol{C}^{(k)}$, we mean the application to the corresponding spline function $\hat{u}$.

The entries of the matrix $\boldsymbol{C}^{(k)}$ can be interpreted as values on a uniform grid $\mathcal{T}^{(k)}$. This motivates the definition of an operator $(\cdot)_I : C([0,1]^2) \to \mathcal{R}^{(k)}$, which evaluates a continuous function on the grid points $(x_i) = (x_{i^1 i^2})$, and an operator $\chi^{(k)} : \mathcal{R}^{(k)} \to \mathcal{Q}_1(\mathcal{T}^{(k)}) \subset H^1([0,1]^2)$ that provides a piecewise bilinear interpolation of the given grid values, where $\mathcal{Q}_1(\mathcal{T}^{(k)})$ is the space of piecewise bilinear functions on $\mathcal{T}^{(k)}$.

Furthermore, given values on an edge e on $[0,1]^2$ along dimension ι, we need to define its linear interpolation and a discrete harmonic extension to the interior. In order to do so, let us denote by $\mathcal{I}(e)$ all indices of grid points x_i associated to e. Additionally, let $\mathcal{P}_1(\mathcal{T}^{(k)}|_e)$ be the space of piecewise linear spline functions on $\mathcal{T}^{(k)}|_e$. We define the interpolation of values on $\mathcal{I}(e)$ by the restriction of the operator $\chi^{(k)}$ to e, denoted by $\chi_e^{(k)} : \mathbb{R}^{M_\iota^{(k)}} \to H^1(e)$ with an analogous definition. In a similar way, we define the interpolation operator for the whole boundary $\partial[0,1]^2$, denoted by $\chi_\partial^{(k)} : \mathbb{R}^{|\mathcal{I}(\partial)|} \to H^1(\partial[0,1]^2)$, where $\mathcal{I}(\partial) := \{i : x_i \in \partial[0,1]^2\}$. This leads to a definition of a semi-norm for grid points on an edge e via the interpolation to functions in $\mathcal{P}_1(\mathcal{T}|_e^{(k)})$:

Definition 3.15. *Let e be an edge of $[0,1]^2$ along dimension ι, and let $\boldsymbol{v}$ be a vector in $\mathbb{R}^{M_\iota^{(k)}}$. Then we define the semi-norm $\|\boldsymbol{v}\|_e := |\chi_e^{(k)}(\boldsymbol{v})|_{H^{1/2}(e)}$ for all $\boldsymbol{v} \in \mathbb{R}^{M_\iota^{(k)}}$.*

Remark 3.16. *For the interpolation operator $\chi_e^{(k)}$ defined on an edge e, it is easy to see that $\chi^{(k)}(\boldsymbol{C}^{(k)})|_e = \chi_e^{(k)}(\boldsymbol{C}|_e^{(k)})$ and*

$$\|\boldsymbol{C}|_e^{(k)}\|_e = |\chi_e^{(k)}(\boldsymbol{C}|_e^{(k)})|_{H^{1/2}(e)} = |\chi^{(k)}(\boldsymbol{C}^{(k)})|_e|_{H^{1/2}(e)}$$

hold.

Finally, we are able to define the discrete harmonic extension in $\mathcal{R}^{(k)}$.

Definition 3.17. *Let $\mathcal{H}_{\mathcal{Q}_1}$ be the standard discrete harmonic extension into the piecewise bilinear space $\mathcal{Q}_1$. This defines the lifting operator $\boldsymbol{H} : \mathbb{R}^{|\mathcal{I}(\partial)|} \to \mathcal{R}^{(k)}$ by*

$$\boldsymbol{b} \mapsto \boldsymbol{H}(\boldsymbol{b}) := (\mathcal{H}_{\mathcal{Q}_1}(\chi_\partial(\boldsymbol{b})))_I.$$

Theorem 3.18. *Let e be a particular side on the boundary of $[0,1]^2$ and the constant $\beta \in \mathbb{R}^+$ such that $\beta^{-1}M_2^{(k)} \le M_1^{(k)} \le \beta M_2^{(k)}$. Then the following statements hold:*

- *For all $\boldsymbol{b} \in \mathbb{R}^{2M_1^{(k)}+2M_2^{(k)}-4}$ that vanish on the four components corresponding to the four corners, the estimate*

$$\|\boldsymbol{H}(\boldsymbol{b})\|_\nabla^2 \le c(1+\log^2 M_1^{(k)}) \sum_{e\in\partial[0,1]^2} \|\boldsymbol{b}|_e\|_e^2,$$

holds, where the constant c depends only on β.

- *The estimate $|\boldsymbol{C}|_\nabla \geq c\|\boldsymbol{C}|_e\|_e$ is valid for all $\boldsymbol{C} \in \mathbb{R}^{M_1^{(k)} \times M_2^{(k)}}$, where the constant c depends only on β.*

Proof. See [19]. □

3.3.2 Condition Number Estimate

The goal of this section is to establish a condition number bound for $P = M_{BDDC}^{-1}\hat{S}$. Following [19], we assume that the mesh is quasi-uniform on each subdomain and the diffusion coefficient is globally constant. We focus now on a single patch $\Omega^{(k)}$, $k \in \{1, \dots, N\}$. For notational simplicity, we assume that the considered patch $\Omega^{(k)}$ does not touch the boundary $\partial\Omega$.

We define the four edges of the parameter domain $[0,1]^2$ by $\hat{E}_r$, and their images by $E_r^{(k)} = G^{(k)}(\hat{E}_r)$, $r = 1, 2, 3, 4$. Moreover, we denote by $\mathcal{I}(E_r^{(k)})$ the coefficient indices corresponding to the basis functions on $E_r^{(k)}$ and by $\mathcal{I}(\Gamma^{(k)})$ the indices corresponding to the whole boundary. Let $u^{(k)} \in V_h^{(k)}$, then $u^{(k)}$ is determined by its coefficients $(\boldsymbol{u}_i)_{i \in \mathcal{I}^{(k)}}$, which can be interpreted as a $M_1^{(k)} \times M_2^{(k)}$ matrix $\boldsymbol{C}$. In a similar way, we can identify functions on the trace space $W^{(k)}$.

Finally, let $W_\Delta^{(k)} \subset W^{(k)}$ be the space of spline functions which vanish on the primal variables, i.e., in the corner points. The following theorem provides an abstract estimate of the condition number using the multiplicity scaling:

Theorem 3.19. *Let $\delta^{\dagger(k)}$ be chosen accordingly to the multiplicity scaling strategy. Assume that there exist two positive constants c_*, c^*, and a boundary semi-norm $|\cdot|_{W^{(k)}}$ on $W^{(k)}$, $k = 1, \dots, N$, such that*

$$|w^{(k)}|_{W^{(k)}}^2 \leq c^* s^{(k)}(w^{(k)}, w^{(k)}) \quad \forall w^{(k)} \in W^{(k)}, \tag{3.19}$$

$$|w^{(k)}|_{W^{(k)}}^2 \geq c_* s^{(k)}(w^{(k)}, w^{(k)}) \quad \forall w^{(k)} \in W_\Delta^{(k)}, \tag{3.20}$$

$$|w^{(k)}|_{W^{(k)}}^2 = \sum_{r=1}^{4} |w^{(k)}|_{E_r^{(k)}}|_{W_r^{(k)}} \quad \forall w^{(k)} \in W^{(k)}, \tag{3.21}$$

where $|\cdot|_{W_r^{(k)}}$ is a semi-norm associated to the edge spaces $W^{(k)}|_{E_r^{(k)}}$, with $r = 1, 2, 3, 4$. Then the condition number of the preconditioned BDDC operator P satisfies the bound

$$\kappa(M_{BDDC}^{-1}\hat{S}) \leq C(1 + c_*^{-1}c^*),$$

where the constant C is independent of h and H.

Proof. See [19] or [17]. □

Using this abstract framework, we obtain the following condition number estimate for the BDDC preconditioner.

Theorem 3.20. *Let $\Omega \subset \mathbb{R}^2$, and let $\widetilde{W}$ be given as set of all functions from W, that are continuous at the corners of $\Omega^{(k)}$ for $k = 1, \dots, N$. Moreover, we assume that the diffusion coefficient α is globally constant. Then there exists a boundary semi-norm such that the constants c_* and c^* of Theorem 3.19 are bounded by*

$$c^* \leq C_1 \quad \text{and} \quad c_*^{-1} \leq C_2 \max_{1 \leq k \leq N} \left(1 + \log^2 \left(H_k/h_k\right)\right),$$

where the constants C_1 and C_2 are independent of H_k and h_k. Therefore, the condition number of the isogeometric preconditioned BDDC operator is bounded by

$$\kappa(M_{BDDC}^{-1}\hat{S}) \leq C \max_{1 \leq k \leq N} \left(1 + \log^2 \left(H_k/h_k\right)\right),$$

where the constant C is independent of H_k and h_k.

Proof. The proof essentially follows the lines of the proof given in [19] with a minor modification due to the different geometrical mappings $G^{(k)}$. We note that we only consider C^0 continuity across the patch interfaces, which makes the proof less technical.

The first step is to appropriately define the semi-norm $|\cdot|^2_{W^{(k)}}$ in $W^{(k)}$:

$$|w^{(k)}|^2_{W^{(k)}} := \sum_{r=1}^{4} \left(\| w^{(k)}|_{E_r^{(k)}} \|^2_{E_r^{(k)}} + |w^{(k)}|_{E_r^{(k)}}|^2_{k,\nabla} \right), \tag{3.22}$$

where $\| w^{(k)}|_{E_r^{(k)}} \|_{E_r^{(k)}} := \| \boldsymbol{v} \|_{\hat{E}_r}$, with $\boldsymbol{v}$ being the values $(\boldsymbol{w}_i)_{i \in \mathcal{I}(E_r^{(k)})}$ written as a vector and $|w^{(k)}|_{\hat{E}_r}|^2_{k,\nabla}$ has to be understood as the restriction of the discrete semi-norm to the edge $\hat{E}_r$.

Let $u^{(k)} \in V_h^{(k)}$ be the IgA harmonic extension of $w^{(k)}$, and $\hat{u}^{(k)}$ its representation in the parameter domain. Additionally, let e be any edge of the parameter domain of $\Omega^{(k)}$. Due to the fact that $\boldsymbol{u}_i = \boldsymbol{w}_i$ for $i \in \mathcal{I}(\Gamma^{(k)})$, and denoting $\boldsymbol{C}^{(k)} = (\boldsymbol{u}_i)_{i \in \mathcal{I}^{(k)}}$, we obtain

$$\| w^{(k)}|_e \|^2_e = \| \boldsymbol{C}^{(k)}|_e \|^2_e \leq c \| \boldsymbol{C}^{(k)} \|^2_\nabla$$

by means of Theorem 3.18. From the definition of $\| \boldsymbol{C}^{(k)} \|^2_\nabla$ and the definition of $|w^{(k)}|_{E_r^{(k)}}|^2_{W_r^{(k)}}$, we get

$$|w^{(k)}|_e|^2_{W_r^{(k)}} \leq c \| \boldsymbol{C}^{(k)} \|^2_\nabla.$$

Furthermore, we have

$$|w^{(k)}|_e|^2_{W_r^{(k)}} \leq c \| \boldsymbol{C}^{(k)} \|^2_\nabla \leq c |\hat{u}^{(k)}|^2_\nabla \leq c |\hat{u}^{(k)}|^2_{H^1(\hat{\Omega})} \leq c |u^{(k)}|^2_{H^1(\Omega^{(k)})}.$$

Since

$$|u^{(k)}|^2_{H^1(\Omega^{(k)})} = |\mathcal{H}^{(k)}(w^{(k)})|^2_{H^1(\Omega^{(k)})} = s^{(k)}(w^{(k)}, w^{(k)}),$$

we arrive at the estimate $|w^{(k)}|_e|^2_{W_r^{(k)}} \leq c\, s^{(k)}(w^{(k)}, w^{(k)})$. These estimates hold for all edges of $\Omega^{(k)}$. Hence, it follows that

$$|w^{(k)}|^2_{W^{(k)}} \leq c^* s^{(k)}(w^{(k)}, w^{(k)}) \quad \text{for } w \in W^{(k)},$$

where the constant does not depend on h_k and H_k. This proves the upper bound, i.e., estimate (3.19).

Let be $w^{(k)} \in W_\Delta^{(k)}$, $\hat{w}^{(k)}$ its representation in the parameter domain, and $(\boldsymbol{w}_i)_{i\in\mathcal{I}(\Gamma^{(k)})}$ its coefficient representation. We apply the lifting operator $\boldsymbol{H}^{(k)}$ to $(\boldsymbol{w}_i)_{i\in\mathcal{I}(\Gamma^{(k)})}$, and obtain a matrix $\boldsymbol{H}^{(k)}(\hat{w}^{(k)})$ with entries $(\boldsymbol{w}_i^{H^{(k)}})_{i\in\mathcal{I}^{(k)}}$. These entries define a spline function $\hat{u}^{(k)} := \sum_{i\in\mathcal{I}^{(k)}} \boldsymbol{w}_i^{H^{(k)}} \hat{N}_{i,p}^{(k)}$. The following estimate

$$|\!|\!|\boldsymbol{H}^{(k)}(\hat{w}^{(k)})|\!|\!|^2_\nabla = |\hat{u}^{(k)}|^2_\nabla \geq c|\hat{u}^{(k)}|^2_{H^1(\hat{\Omega})} \geq c|u^{(k)}|^2_{H^1(\Omega^{(k)})} \geq c|\mathcal{H}(w^{(k)})|^2_{H^1(\Omega^{(k)})},$$

holds, where the last inequality is valid due to the fact that the discrete IgA harmonic extension minimizes the energy among functions with given boundary data w. The constant c does not depend on h_k or H_k.

Recalling the definition of $|w^{(k)}|^2_{W^{(k)}}$ and using Theorem 3.18, we arrive at the estimates

$$|\!|\!|\boldsymbol{H}^{(k)}(\hat{w}^{(k)})|\!|\!|^2_\nabla \leq c(1+\log^2 M^{(k)}) \sum_{e\in\partial[0,1]^2} |\!|\!|\hat{w}|_e^{(k)}|\!|\!|^2_e \leq c(1+\log^2 M^{(k)})|w^{(k)}|^2_{W^{(k)}}.$$

Due to the mesh regularity, we have $M^{(k)} \approx H_k/h_k$, and, hence, we obtain

$$s^{(k)}(w^{(k)}, w^{(k)}) = |\mathcal{H}(u^{(k)})|^2_{H^1(\Omega^{(k)})} \leq c(1+\log^2(H_k/h_k))|w^{(k)}|^2_{W^{(k)}}.$$

which provides the desired estimate for c_*^{-1} by taking the maximum over all patches. □

The next theorem provides the corresponding estimates for the modified stiffness scaling, see Remark 3.8.

Theorem 3.21. *Let the counting functions be chosen according to the stiffness scaling strategy. Assume that there exist two positive constants c_*, c^*, and a boundary seminorm $|\cdot|_{W^{(k)}}$ on $W^{(k)}$, $k = 1, \ldots, N$, such that the three conditions of Theorem 3.19 hold. Moreover, we assume that it exits a constant c^*_{STIFF} such that*

$$|w^{(k)}|_{W^{(k)}} \leq c^*_{STIFF} s(\delta w^{(k)}, \delta w^{(k)}) \quad \forall w^{(k)} \in W_\Delta^{(k)}, \tag{3.23}$$

where the coefficients of $\delta w^{(k)}$ are given by $\boldsymbol{w}_i^{(k)}\delta_i^{(k)}$. Then the condition number of the preconditioned BDDC operator $M^{-1}_{BDDC}\hat{S}$ satisfies the bound

$$\kappa(M^{-1}_{BDDC}\hat{S}) \leq c(1 + c_*^{-1}c^* + c_*^{-1}c^*_{STIFF})$$

for some constant c which is independent of h_k and H_k.

Proof. See [19]. □

Lemma 3.22. *The bound (3.23) holds with $c^*_{STIFF} \leq C_1$, where C_1 is the constant appearing in Theorem 3.20. Hence, the condition number of the BDDC preconditioned system in the case of stiffness scaling is bounded by*

$$\kappa(M^{-1}_{BDDC}\hat{S}) \leq C \max_{1\leq k\leq N}\left(1+\log^2\left(H_k/h_k\right)\right),$$

where the constant C is independent of H_k and h_k.

Proof. The inequality $|w^{(k)}|^2_{W^{(k)}} \leq c^*_{STIFF} s(\delta w^{(k)}, \delta w^{(k)})$ is equivalent to

$$|\delta^\dagger w^{(k)}|^2_{W^{(k)}} \leq c^*_{STIFF} s(w^{(k)}, w^{(k)}) \text{ for } w \in W^{(k)}_\Delta.$$

We have already proven that

$$|w^{(k)}|^2_{W^{(k)}} \leq c^* s(w^{(k)}, w^{(k)}) \quad \text{for } w \in W^{(k)}_\Delta \subset W^{(k)}.$$

Hence, it is enough to show the inequality

$$|\delta^\dagger w^{(k)}|^2_{W^{(k)}} \leq c_{h,H}|w^{(k)}|^2_{W^{(k)}} \quad \text{for } w \in W^{(k)}_\Delta,$$

where the constant $c_{h,H}$ may depend on $h^{(k)}$ and $H^{(k)}$. Recalling the definition of $|\delta^\dagger w^{(k)}|^2_{W^{(k)}}$ as the sum of $|\delta^\dagger w^{(k)}|^2_{E^{(k)}_r}|_{W^{(k)}_r}$, $r = 1, 2, 3, 4$, see (3.22), we have to estimate only its parts, e.g., $|w^{(k)}|_{E^{(k)}_1}|_{W^{(k)}_1}$. The other three terms follow analogously. From the fact that $\delta^{\dagger(k)}_i \leq 1$ and $\delta^{\dagger(k)}_i = \delta^{\dagger(k)}_{i+1}$ for all $k \in \{1, \ldots, N\}$ and $i \in \mathcal{I}(E^{(k)}_1)$, it follows that

$$\|\delta^\dagger w^{(k)}|_{E^{(k)}_1}\|_{E^{(k)}_1} = \delta^\dagger \|w^{(k)}|_{E^{(k)}_1}\|_{E^{(k)}_1} \leq \|w^{(k)}|_{E^{(k)}_1}\|_{E^{(k)}_1}$$

and

$$\begin{aligned}\sum_{i^2=1}^{M_2{}^{(k)}-1} |\delta^\dagger_{(1,i^2+1)}\boldsymbol{w}_{(1,i^2+1)} - \delta^\dagger_{(1,i^2)}\boldsymbol{w}_{(1,i^2)}|^2 &= \sum_{i^2=1}^{M_2{}^{(k)}-1} \delta^\dagger_i |\boldsymbol{w}_{(1,i^2+1)} - \boldsymbol{w}_{(1,i^2)}|^2 \\ &\leq \sum_{i^2=1}^{M^{(k)}_2-1} |\boldsymbol{w}_{(1,i^2+1)} - \boldsymbol{w}_{(1,i^2)}|^2.\end{aligned}$$

These estimates provide the inequalities $|\delta^\dagger w^{(k)}|_{E^{(k)}_1}|_{W^{(k)}_1} \leq |w^{(k)}|_{E^{(k)}_1}|_{W^{(k)}_1}$, and, finally,

$$|\delta^\dagger w^{(k)}|_{W^{(k)}} \leq |w^{(k)}|_{W^{(k)}}.$$

This concludes the proof with $c^*_{STIFF} \leq c^*$, and the desired condition number bound. □

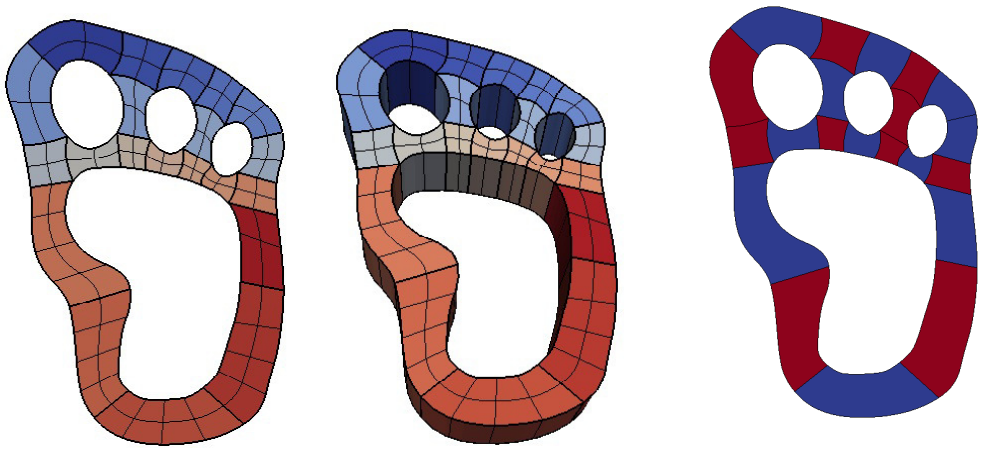


Figure 3.2: The domain Ω in 2d (left) and 3d (middle), and the coefficient pattern (right).

3.4 Numerical Examples

In this section, we test the implemented IETI-DP algorithm for solving large scale systems arising from the IgA discretization of (2.2) on the so-called YETI-footprint domains illustrated in Figure 3.2. The computational domain consists of 21 subdomains in both 2d and 3d. In both cases, one side of a patch boundary has inhomogeneous Dirichlet conditions, whereas all other sides have homogeneous Neumann conditions. Each subdomain has a diameter of H and an associated mesh-size of h. The degree of the B-Splines is chosen as $p = 2$ and $p = 4$ with maximal smoothness. For the three-dimensional tests, we perform one additional refinement in z-direction. The linear system (3.12), is solved by a PCG algorithm with the scaled Dirichlet preconditioner (3.13). We use zero initial guess, and a reduction of the initial residual by a factor of 10^{-8} as stopping criterion. The numerical examples illustrate the dependence of the condition number of the IETI-DP preconditioned system on jumps in the diffusion coefficient α, patch size H, mesh-size h and the degree p. We use the C++ library G+Smo, see [162] and [111], for describing the geometry and performing the numerical tests. For performance studies in terms of computation time we refer to the Section 4.3.6. There, the cG and dG version of the IETI-DP method is compared.

3.4.1 Homogeneous Diffusion Coefficients

We present numerical tests for problem (2.2) with a globally constant diffusion coefficient $\alpha = 1$. The 2d results are summarized in Table 3.1, whereas the 3d results are presented in Table 3.2. The results confirm that the preconditioned systems using

		ALG. A				ALG. C			
$p=2$		coeff. scal.		stiff. scal.		coeff. scal.		stiff. scal.	
#dofs	H/h	κ	It.	κ	It.	κ	It.	κ	It.
800	8	6.6	15	6.5	15	1.6	8	1.6	7
2364	16	8.8	17	8.7	16	1.7	9	1.7	9
7892	32	11.5	19	11.5	20	2.0	10	2.0	10
28548	64	14.6	21	14.6	21	2.3	11	2.3	11
108260	128	18.2	23	18.2	23	2.7	12	2.6	12
$p=4$		coeff. scal.		stiff. scal.		coeff. scal.		stiff. scal.	
#dofs	H/h	κ	It.	κ	It.	κ	It.	κ	It.
1418	8	8.8	17	8.7	18	1.8	9	1.7	10
3350	16	11.4	20	11.2	20	2.0	10	1.9	11
9614	32	15.5	21	14.3	21	2.4	12	2.3	12
31742	64	18.0	23	17.9	22	2.8	13	2.8	13
114398	128	22.1	25	22.0	24	3.3	14	3.3	14

Table 3.1: 2d example with homogeneous diffusion coefficient and $p = 2$ and $p = 4$. Choice of primal variables: vertex evaluation (Alg. A), vertex evaluation and edge averages (Alg. C).

coefficient scaling as well as stiffness scaling provide a quasi optimal condition number bound according to Theorem 3.20 and Theorem 3.21.

3.4.2 Jumping Diffusion Coefficients

We investigate numerical examples with patch-wise constant diffusion coefficient α, the jumping pattern of which is shown in Figure 3.2 (right). The diffusion coefficient has values $\alpha^{(k)} \in \{\text{blue}, \text{red}\} := \{10^{-3}, 10^3\}$. The 2d results are summarized in Table 3.3, whereas the 3d results are shown in Table 3.4. We again observe a quasi optimal condition number bound which is clearly independent of the diffusion coefficient and its jumps across the subdomain interfaces.

3.4.3 Weak Scaling

The aim of this section is to investigate the weak scaling behaviour of the IETI-DP method. Here we fix the ratio H/h, i.e. fixing the number of dofs on each patch, and increasing the number of patches. In the following tests, we perform a uniform splitting of the patches, i.e., by splitting them into 2^d subpatches. This procedure is performed via a knot insertion of p knots at the midpoints of each patch side, giving C^0 interfaces between the newly introduced patches. As in the previous examples,

		ALG. A				ALG. C			
$p=2$		coeff.	scal.	stiff.	scal.	coeff.	scal.	stiff.	scal.
#dofs	H/h	κ	It.	κ	It.	κ	It.	κ	It.
1364	2	18.7	24	18.5	24	2.9	11	2.9	11
4984	4	46.9	27	46.7	27	3.5	13	3.4	13
24008	8	115.1	34	115.1	32	4.4	15	4.3	15
142792	16	273.3	43	277.3	43	5.6	17	5.5	17
$p=4$		coeff.	scal.	stiff.	scal.	coeff.	scal.	stiff.	scal.
#dofs	H/h	κ	It.	κ	It.	κ	It.	κ	It.
4696	2	54.7	27	54.6	27	3.6	14	3.5	14
11620	4	125.3	33	124.8	33	4.4	15	4.3	15
40660	8	302.4	44	300.1	40	5.6	17	5.4	17
193108	16	710.8	57	710.2	51	7.0	20	6.8	19

Table 3.2: 3d example with homogeneous diffusion coefficient and $p = 2$ and $p = 4$. Choice of primal variables: vertex evaluation (Alg. A), vertex evaluation, edge averages and face averages (Alg. C).

		ALG. A				ALG. C			
$p=2$		coeff.	scal.	stiff.	scal.	coeff.	scal.	stiff.	scal.
#dofs	H/h	κ	It.	κ	It.	κ	It.	κ	It.
800	8	3.8	11	3.6	11	1.3	6	1.3	6
2364	16	5.0	12	4.8	12	1.6	6	1.6	6
7892	32	6.5	13	6.1	13	2.0	7	1.9	7
28548	64	8.1	13	7.5	13	2.4	8	2.2	8
108260	128	10.0	14	9.1	14	2.8	9	2.6	8
$p=4$		coeff.	scal.	stiff.	scal.	coeff.	scal.	stiff.	scal.
#dofs	H/h	κ	It.	κ	It.	κ	It.	κ	It.
1418	8	5.6	13	5.3	13	1.8	7	1.7	7
3350	16	7.0	13	6.4	13	2.2	8	2.0	7
9614	32	8.7	13	7.8	13	2.6	8	2.3	8
31742	64	10.6	14	9.3	14	3.0	10	2.7	10
114398	128	12.7	14	11.0	14	3.5	9	3.1	9

Table 3.3: 2d example with jumping diffusion coefficient and $p = 2$ and $p = 4$. Choice of primal variables: vertex evaluation (Alg. A), vertex evaluation and edge averages.

		ALG. A				ALG. C			
$p = 2$		coeff. scal.		stiff. scal.		coeff. scal.		stiff. scal.	
#dofs	H/h	κ	It.	κ	It.	κ	It.	κ	It.
1364	2	12.1	14	12.0	13	2.0	9	2.0	9
4984	4	35.5	16	34.7	16	2.8	10	2.7	10
24008	8	95.3	20	92.2	18	3.7	11	3.6	11
142792	16	239.0	29	230.1	29	4.8	12	4.5	12
$p = 4$		coeff. scal.		stiff. scal.		coeff. scal.		stiff. scal.	
#dofs	H/h	κ	It.	κ	It.	κ	It.	κ	It.
4696	2	41.5	18	40.1	16	3.1	10	3.0	10
11620	4	103.2	20	97.4	20	3.9	11	3.7	11
40660	8	264.4	27	244.0	25	5.1	12	4.7	12
193108	16	640.7	33	584.8	33	6.3	13	5.7	12

Table 3.4: 3d example with jumping diffusion coefficient and $p = 2$ and $p = 4$. Choice of primal variables: vertex evaluation (Alg. A), vertex evaluation and edge averages and face averages (Alg. B).

we choose the two- and three-dimensional YETI footprint as computational domain and globally constant diffusion coefficient. On each patch we perform three initial refinements, and use B-Spline degree 4 in the two-dimensional setting. For the three-dimensional domain, we perform one initial refinement and use B-Spline of degree 3. Due to the fact that the condition number depends only on the factor H/h for a fixed degree, we expect constant condition numbers and number of iterations. The results are summarized in Table 3.5 and show the expected behaviour. We only observe a slight increase of the condition number for the two-dimensional domain using Algorithm A. Moreover, in the three-dimensional example we observe that the method does not scale well using Algorithm A, due to the sub-optimal bound $H/h(1 + \log(H/h))^2$ on the condition number.

3.4.4 Dependence on the Degree p

We want to examine the dependence of the condition number on the B-Spline degree p. When increasing the B-Spline degree, one can either increase the multiplicity of the knots and keep the smoothness, or increase the smoothness, while keeping the multiplicity. The computational domain Ω is chosen as the 2d and 3d YETI-footprint presented in Figure 3.2. For the three-dimensional tests, we do not perform the additional refinement in z-direction. The diffusion coefficient α is chosen to be equal to 1. The results are summarized in Table 3.6 and Table 3.7, where we observe a possibly logarithmic dependence of the condition number on the polynomial degree p in case of the coefficient scaling as well as of the stiffness scaling, see Figure 3.3.

		ALG. A					ALG. C				
2D			coeff. scal		stiff. scal			coeff. scal		stiff. scal	
N	Dofs	n_Π	κ	It.	κ	It.	n_Π	κ	It.	κ	It.
21	9614	34	14.5	25	14.4	25	58	2.3	14	2.3	15
84	38464	114	12.6	31	12.6	31	246	2.7	16	2.8	17
336	133386	400	11.5	34	11.5	34	1000	2.8	17	2.8	17
1344	533558	1476	11.3	35	11.3	35	4020	3.1	18	3.1	18
5376	2134254	5644	12.6	37	12.6	37	16108	3.5	18	3.5	18
3D			coeff. scal		stiff. scal			coeff. scal		stiff. scal	
N	Dofs	n_Π	κ	It.	κ	It.	n_Π	κ	It.	κ	It.
21	7846	150	85.7	37	84.9	36	140	4.0	18	3.9	17
168	62720	1038	102.3	78	102.5	80	1018	4.3	20	4.3	20
1344	451530	7026	101.1	97	101.4	97	6974	4.3	22	4.3	22
10752	3612342	49962	109.8	103	110.5	103	49860	4.6	22	4.6	22

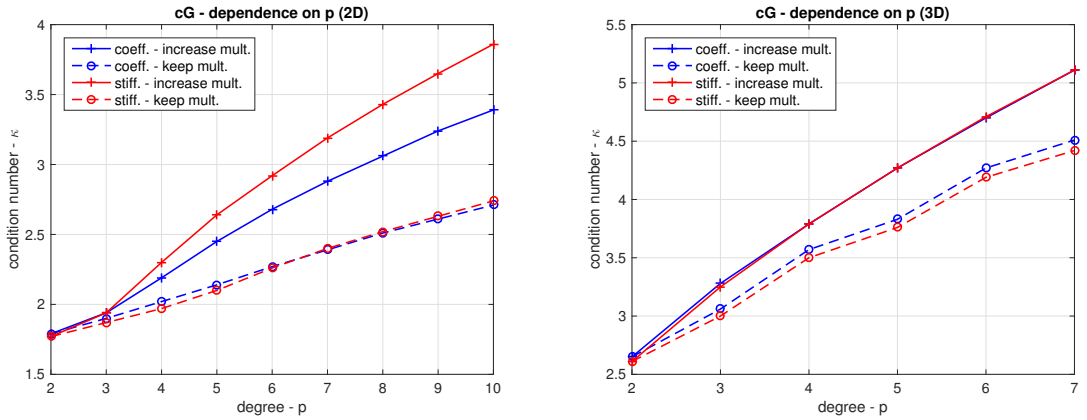
Table 3.5: Weak scaling results for two- and three-dimensional computational domain having a B-Spline degree of $p = 4$ and $p = 3$, respectively.

Increasing the multiplicity						Increasing the smoothness					
ALG. C		coeff. scal		stiff. scal.		ALG. C		coeff. scal		stiff. scal.	
#dofs	degree	κ	It.	κ	It.	#dofs	degree	κ	It.	κ	It.
2364	2	1.7	11	1.7	10	2364	2	1.7	11	1.7	10
7892	3	1.9	13	1.9	12	2836	3	1.9	12	1.7	11
16620	4	2.1	14	2.1	14	3350	4	2.0	13	1.8	12
28548	5	2.4	15	2.3	15	3906	5	2.1	13	1.8	13
43676	6	2.6	16	2.5	16	4504	6	2.2	14	1.9	13
62004	7	2.8	17	2.7	16	5144	7	2.3	15	2.0	14
83532	8	3.0	17	2.8	17	5826	8	2.5	15	2.1	14
108260	9	3.2	18	3.0	18	6550	9	2.6	15	2.2	15
136188	10	3.3	18	3.1	18	7316	10	2.7	16	2.3	15

Table 3.6: 2d example with fixed initial mesh and homogeneous diffusion coefficient. Dependence of the condition number κ and the number It. of iterations on p for the preconditioned system with coefficient and stiffness scaling. Choice of primal variables: vertex evaluation and edge averages.

Increasing the multiplicity						Increasing the smoothness					
ALG. C		coeff. scal		stiff. scal.		ALG. C		coeff. scal		stiff. scal.	
#dofs	degree	κ	It.	κ	It.	#dofs	degree	κ	It.	κ	It.
1000	2	2.6	12	2.6	12	1000	12	2.6	12	2.6	8
4138	3	3.2	15	3.2	16	2148	15	3.0	15	3.0	9
10604	4	3.7	18	3.7	18	3898	17	3.5	17	3.5	10
21598	5	4.2	19	4.2	20	6376	18	3.8	18	3.8	11
38320	6	4.7	20	4.7	21	9708	20	4.2	20	4.2	11
61970	7	5.1	21	5.1	23	14020	20	4.5	20	4.4	12

Table 3.7: 3d example with fixed initial mesh and homogeneous diffusion coefficient. Dependence of the condition number κ and the number It. of iterations on p for the preconditioned system with coefficient and stiffness scaling. Choice of primal variables: vertex evaluation, edge averages and face averages.



(a) 2d setting (b) 3d setting

Figure 3.3: Dependence of the condition number on the B-Spline degree p for the 2d and 3d domain. We compare the influence of the considered scaling strategy and method with increasing the degree. Choice of primal variables: vertex evaluation, edge averages and face averages (Alg. B).

Chapter 4

Discontinuous Galerkin IETI-DP Methods

In this chapter we are going to develop an efficient and robust solver for large-scale systems of algebraic equations arising from IgA discretizations where the different patches are coupled by means of a discontinuous Galerkin (dG) method, called dG-IETI-DP. This setting is of special importance when considering non-matching meshes, see, e.g., [148], and in case of non-matching interface parametrizations, resulting in gap and overlapping regions at the interfaces. The latter one will be investigated in Section 4.4, where the focus of this thesis is on the application of dG-IETI-DP methods to such problems.

The proposed method is based on the corresponding version for FE proposed in [52] and [57]. In these works a rigorous analysis of the dG version of the FETI-DP method shows the same quasi-optimal condition number bounds with respect to the ratio of patch diameter and mesh-size (H/h) of the preconditioned FETI-DP operator as for classical FETI-DP methods for two- and three-dimensional domains. We present an analysis of the dG-IETI-DP method that provides a quasi-optimal condition number bound of the preconditioned system with respect to H/h. However, the presented bound depends polynomially on the ratio of neighbouring mesh-sizes.

We introduce the dG-IETI-DP method and the required notation in Section 4.1. In Section 4.2 we perform the condition number analysis of the dG-IETI-DP operator with respect to H/h considering a two-dimensional domain, globally constant diffusion coefficient and only vertex values as primal variables. Numerical experiments for two- and three-dimensional domains are presented in Section 4.3. Finally, in Section 4.4 we apply the dG-IETI-DP method to domains having gap and overlapping regions at the interfaces.

4.1 Derivation of the Method

Let us consider a multi-patch domain, where the interfaces are geometrically matching, but not the meshes, i.e., the meshes can be different on neighbouring patches. Therefore, the considered solution and test spaces do not provide continuity across the patch interfaces. Hence, we cannot enforce continuity of the solution by means of the same jump operator as for the conforming IETI-DP method introduced in Chapter 3. As proposed in [52], we introduce an additional layer of dofs on the interfaces and enforce continuity between the different layers. This method can then be seen as a conforming IETI-DP method on an extended grid of dofs. We will follow the derivation presented in [52] and [57] with adopted notations. In the following, let V_h be the dG-IgA space which fulfils the Dirichlet boundary conditions as defined in Section 2.3.2, and we denote by $\{N_{i,p}\}_{i\in\mathcal{I}}$ the corresponding B-Spline basis.

4.1.1 Basic Setup and Local Space Description

As already introduced in Section 2.3.2, let $\mathcal{I}_{\mathcal{F}}^{(k)}$ be the set of all indices l such that $\Omega^{(k)}$ and $\Omega^{(l)}$ share a common edge/face. In the following, we will often denote $\mathcal{I}_{\mathcal{F}}^{(k)}$ by $\mathcal{F}^{(k)}$. If we consider three-dimensional domains, we additionally define $\overline{E}^{(klm)}$ as the edge shared by the patches $\Omega^{(k)}$, $\Omega^{(l)}$ and $\Omega^{(m)}$, i.e., $\overline{E}^{(klm)} := \partial F^{(kl)} \cap \partial F^{(km)}$ for $l \in \mathcal{F}^{(k)}$ and $m \in \mathcal{F}^{(k)}$. The set of all indices (l, m) of $\Omega^{(l)}$ and $\Omega^{(m)}$, such that $\overline{E}^{(klm)}$ is an edge of patch $\Omega^{(k)}$ is denoted by $\mathcal{E}^{(k)}$. For two-dimensional domains, the set $\mathcal{E}$ is empty. Note, although $\overline{F}^{(lk)} \subset \partial\Omega^{(l)}$ and $\overline{F}^{(kl)} \subset \partial\Omega^{(k)}$ are geometrically the same, they are treated as different objects. The same applies to the edges $\overline{E}^{(klm)}$, $\overline{E}^{(lkm)}$ and $\overline{E}^{(mkl)}$. In order to keep the presentation of the method simple, we assume that the considered patch $\Omega^{(k)}$ does not touch the Dirichlet boundary. The other case can be handled in an analogous way.

As already introduced in Section 2.2, the computational domain Ω is given by $\overline{\Omega} = \bigcup_{k=1}^{N} \overline{\Omega}^{(k)}$, where $\Omega^{(k)} = G^{(k)}(\hat{\Omega})$ for $k = 1, \dots, N$, and the interface corresponding to $\Omega^{(k)}$ by $\Gamma^{(k)} = \overline{\partial\Omega^{(k)} \backslash \partial\Omega}$. For each patch $\Omega^{(k)}$, we introduce its extended version $\Omega_e^{(k)}$ via the union with all neighbouring interfaces $\overline{F}^{(lk)} \subset \partial\Omega^{(l)}$:

$$\overline{\Omega}_e^{(k)} := \overline{\Omega}^{(k)} \cup \{ \bigcup_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}} \overline{F}^{(lk)} \}.$$

Moreover, the extended interface $\Gamma_e^{(k)}$ is given by the union of $\Gamma^{(k)}$ with all neighbouring interfaces

$$\Gamma_e^{(k)} := \Gamma^{(k)} \cup \{ \bigcup_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}} \overline{F}^{(lk)} \}.$$

Based on the definitions above, we can introduce

$$\overline{\Omega}_e := \bigcup_{k=1}^{N} \overline{\Omega}_e^{(k)}, \quad \Gamma := \bigcup_{k=1}^{N} \Gamma^{(k)} \text{ and } \quad \Gamma_e := \bigcup_{k=1}^{N} \Gamma_e^{(k)}.$$

An illustration of the domain $\Omega_e^{(k)}$ is given in Figure 4.1. Finally, we introduce the set of vertices $\mathcal{V}^{(k)}$ associated to $\Omega_e^{(k)}$ by

$$\mathcal{V}^{(k)} := \{ \bigcup_{l \in \mathcal{I}_{\mathcal{F}}^{(k)}} \partial F^{(kl)} \} \cup \{ \bigcup_{l \in \mathcal{I}_{\mathcal{F}}^{(k)}} \partial F^{(lk)} \},$$

for two-dimensional domains and by

$$\mathcal{V}^{(k)} := \{ \bigcup_{(l,m) \in \mathcal{E}^{(k)}} \partial E^{(klm)} \} \cup \{ \bigcup_{(l,m) \in \mathcal{E}^{(k)}} \partial E^{(lkm)} \cup \partial E^{(mkl)} \},$$

for three-dimensional domains. The set $\mathcal{V}$ is then given by the union of all $\mathcal{V}^{(k)}$. Moreover, we denote by $\mathcal{V}^{(kl)} \subset \mathcal{V}$ all vertices which belong to the interface $F^{(kl)}$.

The next step is to describe appropriate discrete function spaces to reformulate (2.16) in order to treat the new formulation in the spirit of the conforming IETI-DP method. We start with a description of the discrete function spaces for a single patch. As defined in (2.8), let $V_h^{(k)}$ be the discrete function space defined on the patch $\Omega^{(k)}$. Then we define the extended function space by

$$V_{h,e}^{(k)} := V_h^{(k)} \times \prod_{l \in \mathcal{I}_{\mathcal{F}}^{(k)}} V_h^{(lk)},$$

where $V_h^{(lk)} \subset V_h^{(l)}$ is given by

$$V_h^{(lk)} := \operatorname{span}\{ N_{i,p}^{(l)} \,|\, \operatorname{supp}\{N_{i,p}^{(l)}\} \cap \overline{F}^{(kl)} \neq \emptyset \}.$$

Moreover, we denote by $V_h^{(kl)}$ the trace space of $V_h^{(k)}$ on $F^{(kl)}$, i.e., for $u^{(k)} \in V_h^{(k)}$ we have that $u^{(k)}|_{F^{(kl)}} \in V_h^{(kl)}$. The space $V_{h,e}^{(k)}$ can be illustrated by means of the extended domain $\overline{\Omega}_e^{(k)}$. Note, for the analysis $\overline{\Omega}_e^{(k)}$ is not required. An overview of the local spaces is given in Figure 4.2, cf. Figure 3.1. in [52].

According to [52], we will represent a function $u^{(k)} \in V_{h,e}^{(k)}$ as

$$u^{(k)} = (u^{(k,k)}, (u^{(k,l)})_{l \in \mathcal{I}_{\mathcal{F}}^{(k)}}), \tag{4.1}$$

where the functions $u^{(k,k)} \in V_h^{(k)}$ and $u^{(k,l)} \in V_h^{(lk)}$ possess the representations

$$u^{(k,k)} = \sum_{i \in \mathcal{I}^{(k,k)}} \boldsymbol{u}_i^{(k,k)} N_{i,p}^{(k)} \quad \text{and} \quad u^{(k,l)} = \sum_{i \in \mathcal{I}^{(k,l)}} \boldsymbol{u}_i^{(k,l)} N_{i,p}^{(l)}|_{F^{(kl)}}, \tag{4.2}$$

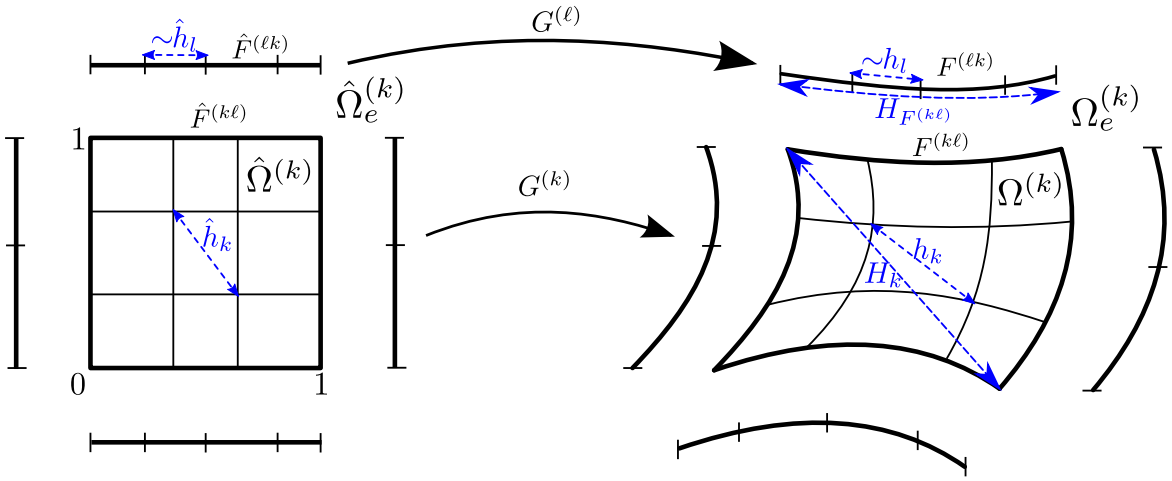


Figure 4.1: Illustration of the mesh in the parameter domain and in the physical domain, presenting the used notation.

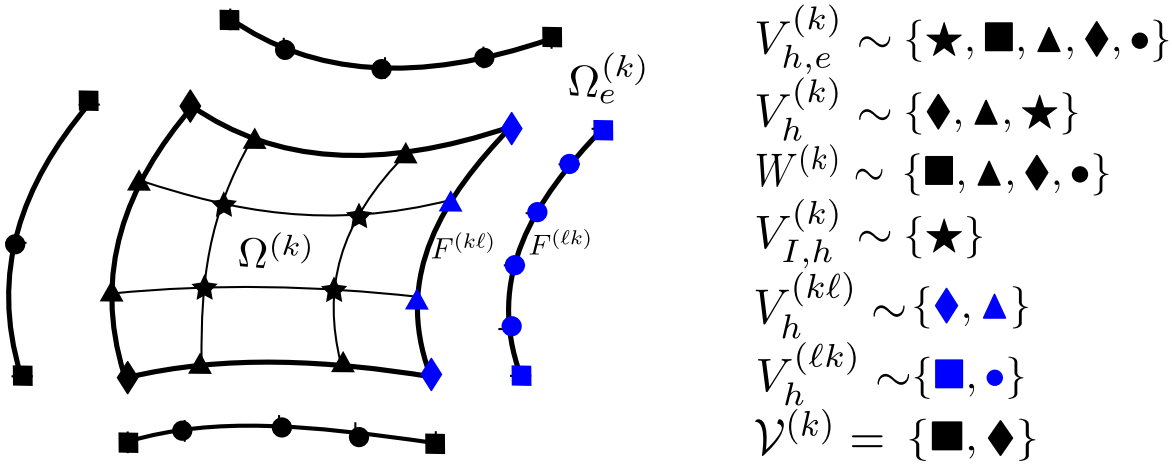


Figure 4.2: Illustration of the local spaces $V_{h,e}^{(k)}, V_h^{(k)}, V_h^{(kl)}, V_h^{(lk)}, V_{I,h}^{(k)}$ and $W^{(k)}$, and the vertices $\mathcal{V}^{(k)}$ of $\Omega_e^{(k)}$. The symbols coloured in blue are still part of sets with the corresponding black symbols. They indicate the sets belonging to a specific neighbouring patch.

respectively. Here, $\mathcal{I}^{(k,k)}$ contains all indices of basis functions associated to the space $V_h^{(k)}$, whereas $\mathcal{I}^{(k,l)}$ contains those indices of basis functions associated to $V_h^{(lk)}$. Using the notation from Figure 4.2, we have that $\mathcal{I}^{(k,k)} := \{\blacklozenge, \blacktriangle, \bigstar\}$ and $\mathcal{I}^{(k,l)} := \{\blacksquare, \bullet\}$. By introducing a suitable ordering, a function $u^{(k)} \in V_{h,e}^{(k)}$ has a vector representation of the form $\boldsymbol{u}^{(k)} = (\boldsymbol{u}_i^{(k)})_{i \in \mathcal{I}_e^{(k)}}$. Moreover, we introduce an additional representation of $V_{h,e}^{(k)}$ as $V_{I,h}^{(k)} \times W^{(k)}$, i.e., for $u^{(k)} \in V_{h,e}^{(k)}$, we have $u^{(k)} = (u_I^{(k)}, u_{B_e}^{(k)})$, where

$$\begin{aligned} V_{I,h}^{(k)} :=& V_{h,e}^{(k)} \cap H_0^1(\Omega^{(k)}) \subset V_{h,e}^{(k)}, \\ W^{(k)} :=& \operatorname{span}\{N_{i,p}^{(k)} | \operatorname{supp} N_{i,p}^{(k)} \cap \Gamma^{(k)} \neq \emptyset\} \\ & \times \operatorname{span}\{N_{i,p}^{(l)} | \operatorname{supp} N_{i,p}^{(l)} \cap \Gamma^{(k)} \neq \emptyset, l \in \mathcal{I}_{\mathcal{F}}^{(k)}\} \subset V_{h,e}^{(k)}. \end{aligned}$$

By repeating the same steps as above for the space $S_h^{(k)}$, one can define the spaces $S_{h,e}^{(k)}$, $S_h^{(lk)}$ and $S_h^{(kl)}$. Let $u^{(k)} \in V_{h,e}^{(k)}$ and $\hat{u}^{(k)} = (\hat{u}^{(k,k)}, (\hat{u}^{(k,l)})_{l \in \mathcal{I}_{\mathcal{F}}^{(k)}})$ be its representation in $S_{h,e}^{(k)}$, then we have the following representation for $\hat{u}^{(k,k)} \in S_h^{(k)}$ and $\hat{u}^{(k,l)} \in S_h^{(lk)}$

$$\hat{u}^{(k,k)} = \sum_{i \in \mathcal{I}^{(k,k)}} \boldsymbol{u}_i^{(k,k)} \hat{N}_{i,p}^{(k)} \quad \text{and} \quad \hat{u}^{(k,l)} = \sum_{i \in \mathcal{I}^{(k,l)}} \boldsymbol{u}_i^{(k,l)} \hat{N}_{i,p}^{(l)}|_{\hat{\Gamma}^{(kl)}}. \tag{4.3}$$

As already mentioned in Section 2.2, the coefficients $\boldsymbol{u}_i^{(k,k)}$ and $\boldsymbol{u}_i^{(k,l)}$ in (4.2) and (4.3) are the same for $u^{(k)} \in V_{h,e}^{(k)}$ and its representation $\hat{u}^{(k)} \in S_{h,e}^{(k)}$.

4.1.2 Schur Complement and Discrete Harmonic Extensions

We note that the patch local bilinear form $a_e^{(k)}(\cdot,\cdot)$, introduced in Section 2.3.2, can be interpreted as it were defined on the space $V_{h,e}^{(k)} \times V_{h,e}^{(k)}$, since it requires function values of the neighbouring patches $\Omega^{(l)}, l \in \mathcal{I}_{\mathcal{F}}^{(k)}$. Similarly, the bilinear form $d^{(k)}(\cdot,\cdot)$ can also be interpreted as it were defined on the space $V_{h,e}^{(k)} \times V_{h,e}^{(k)}$ and it introduces a semi-norm on $V_{h,e}^{(k)}$. Hence, $a_e(\cdot,\cdot)$ induces a matrix representation $\boldsymbol{K}_e$ satisfying the identity

$$a_e^{(k)}(u^{(k)}, v^{(k)}) = (\boldsymbol{K}_e^{(k)} \boldsymbol{u}, \boldsymbol{v})_{\ell_2} \quad \text{for } u^{(k)}, v^{(k)} \in V_{h,e}^{(k)},$$

where $\boldsymbol{u}$ and $\boldsymbol{v}$ denote the vector representation of $u^{(k)}$ and $v^{(k)}$, respectively. By means of the representation $V_{I,h}^{(k)} \times W^{(k)}$ for $V_{h,e}^{(k)}$, we can structure the matrix $\boldsymbol{K}_e^{(k)}$ in the following way

$$\boldsymbol{K}_e^{(k)} = \begin{bmatrix} \boldsymbol{K}_{e,II}^{(k)} & \boldsymbol{K}_{e,IB_e}^{(k)} \\ \boldsymbol{K}_{e,B_eI}^{(k)} & \boldsymbol{K}_{e,B_eB_e}^{(k)} \end{bmatrix}. \tag{4.4}$$

We note that $\boldsymbol{K}_{e,II}^{(k)} = \boldsymbol{K}_{II}^{(k)}$. This enables us to define the Schur complement of $\boldsymbol{K}_e^{(k)}$ with respect to $W^{(k)}$ as

$$\boldsymbol{S}_e^{(k)} := \boldsymbol{K}_{e,B_eB_e}^{(k)} - \boldsymbol{K}_{e,B_eI}^{(k)} \left(\boldsymbol{K}_{e,II}^{(k)}\right)^{-1} \boldsymbol{K}_{e,IB_e}^{(k)}. \tag{4.5}$$

We denote the corresponding bilinear form by $s_e^{(k)}(\cdot,\cdot)$, and the corresponding operator by $S_e^{(k)} : W^{(k)} \to W^{(k)^*}$, i.e.,

$$(\boldsymbol{S}_e^{(k)} \boldsymbol{u}_{B_e}^{(k)}, \boldsymbol{v}_{B_e}^{(k)})_{\ell_2} = \langle S_e^{(k)} u_{B_e}^{(k)}, v_{B_e}^{(k)} \rangle = s_e^{(k)}(u_{B_e}^{(k)}, v_{B_e}^{(k)}), \quad \forall u_{B_e}^{(k)}, v_{B_e}^{(k)} \in W^{(k)}.$$

The Schur complement has the minimization property

$$\langle S_e^{(k)} u_{B_e}^{(k)}, u_{B_e}^{(k)} \rangle = \min_{w^{(k)} = (w_I^{(k)}, w_{B_e}^{(k)}) \in V_{h,e}^{(k)}} a_e^{(k)}(w^{(k)}, w^{(k)}), \tag{4.6}$$

subject to $w_{B_e}^{(k)} = u_{B_e}^{(k)}$ on $\Gamma_e^{(k)}$. We now define the *discrete IgA harmonic extension* $\mathcal{H}_e^{(k)}$ (in the sense of $a_e^{(k)}(\cdot,\cdot)$) for the patch $\Omega_e^{(k)}$ by

$$\begin{aligned} &\mathcal{H}_e^{(k)} : W^{(k)} \to V_{h,e}^{(k)} : \\ &\quad \begin{cases} \text{find } \mathcal{H}_e^{(k)} u_{B_e} \in V_{h,e}^{(k)} : & \\ \quad a_e^{(k)}(\mathcal{H}_e^{(k)} u_{B_e}, u^{(k)}) = 0, & \forall u^{(k)} \in V_{I,h}^{(k)}, \\ \quad \mathcal{H}_e^{(k)} u_{B_e} = u_{B_e} & \text{on } \Gamma_e^{(k)}. \end{cases} \end{aligned} \tag{4.7}$$

One can show that the minimizer in (4.6) is given by $\mathcal{H}_e^{(k)} u_{B_e}$. In addition, we introduce the *standard discrete IgA harmonic extension* $\mathcal{H}^{(k)}$ (in the sense of $a^{(k)}(\cdot,\cdot)$) of $u_{B_e}^{(k)}$ as follows:

$$\begin{aligned} &\mathcal{H}^{(k)} : W^{(k)} \to V_{h,e}^{(k)} : \\ &\quad \begin{cases} \text{find } \mathcal{H}^{(k)} u_{B_e} \in V_{h,e}^{(k)} : & \\ \quad a^{(k)}(\mathcal{H}^{(k)} u_{B_e}, u^{(k)}) = 0, & \forall u^{(k)} \in V_{I,h}^{(k)}, \\ \quad \mathcal{H}^{(k)} u_{B_e} = u_{B_e} & \text{on } \Gamma_e^{(k)}. \end{cases} \end{aligned} \tag{4.8}$$

where $a^{(k)}(\cdot,\cdot)$ is interpreted as bilinear form on the space $V_{h,e}^{(k)} \times V_{h,e}^{(k)}$. The crucial point is to show the equivalence in the energy norm $d_h(u_h, u_h)$, defined in Section 2.3.2, between functions which are discrete harmonic in the sense of $\mathcal{H}_e^{(k)}$ and $\mathcal{H}^{(k)}$. This property is summarized in the following Lemma, cf. also Lemma 3.1 in [52].

Lemma 4.1. *There exists a positive constant C which is independent of $\delta, h_k, H_k, \alpha^{(k)}$ and $u_{B_e}^{(k)}$ such that the inequalities*

$$d^{(k)}(\mathcal{H}^{(k)} u_{B_e}, \mathcal{H}^{(k)} u_{B_e}) \leq d^{(k)}(\mathcal{H}_e^{(k)} u_{B_e}, \mathcal{H}_e^{(k)} u_{B_e}) \leq C d^{(k)}(\mathcal{H}^{(k)} u_{B_e}, \mathcal{H}^{(k)} u_{B_e}), \tag{4.9}$$

hold for all $u_{B_e}^{(k)} \in W^{(k)}$.

Proof. The proof is identical to that one presented for the FE case in Lemma 4.1 in [51]. Note, for the IgA version, we have to use the corresponding discrete trace inequality

$$\|u\|^2_{L^2(\partial\Omega^{(k)})} \leq Ch^{-1} \|u\|^2_{L^2(\Omega^{(k)})}$$

that is valid for all IgA functions u from $V_h^{(k)}$, see Lemma 2.15. □

The subsequent statement immediately follows from Lemma 2.18 and Lemma 4.1, see also [52].

Corollary 4.2. *The spectral equivalence inequalities*

$$C_0 d^{(k)}(\mathcal{H}^{(k)} u_{B_e}, \mathcal{H}^{(k)} u_{B_e}) \leq a_e^{(k)}(\mathcal{H}_e^{(k)} u_{B_e}, \mathcal{H}_e^{(k)} u_{B_e}) \leq C_1 d^{(k)}(\mathcal{H}^{(k)} u_{B_e}, \mathcal{H}^{(k)} u_{B_e}), \tag{4.10}$$

hold for all $u_{B_e}^{(k)} \in W^{(k)}$*, where the constants* C_0 *and* C_1 *are independent of* $\delta, h_k, H_k, \alpha^{(k)}$ *and* $u_{B_e}^{(k)}$*.*

4.1.3 Global Space Description

Based on the definitions of the local spaces in Section 4.1.1, we can now introduce the global spaces

$$V_{h,e} := \prod_{k=1}^{N} V_{h,e}^{(k)} \quad \text{and} \quad W := \prod_{k=1}^{N} W^{(k)}.$$

We note that, according to [52], we will also interpret W as subspace of $V_{h,e}$, where its functions are discrete harmonic in the sense of $\mathcal{H}_e^{(k)}$ on each $\Omega^{(k)}$. For completeness, we define the discrete IgA harmonic extension in the sense of $\sum_{k=1}^{N} a_e^{(k)}(\cdot,\cdot)$ and $\sum_{k=1}^{N} a^{(k)}(\cdot,\cdot)$ for W as $\mathcal{H}_e u = (\mathcal{H}_e^{(k)} u^{(k)})_{k=1}^{N}$ and $\mathcal{H} u = (\mathcal{H}^{(k)} u^{(k)})_{k=1}^{N}$, respectively.

We aim at reformulating (2.16) and (2.20) in terms of the extended space W and introducing Lagrange multipliers in order to couple the independent interface dofs. In the context of tearing and interconnecting methods, we need a "continuous" subspace $\widehat{W}$ of W such that $\widehat{W}$ is equivalent to $V_{\Gamma,h}$, i.e., $\widehat{W} \equiv V_{\Gamma,h}$. Similarly, we need $\widehat{V}_{h,e} \subset V_{h,e}$, such that $\widehat{V}_{h,e} \equiv V_h$. Since the space $V_{\Gamma,h}$ consists of functions which are non-matching across the patch interfaces, the understanding $\widehat{V}_{h,e}$ and $\widehat{W}$ as "continuous" space makes no sense. We follow [51], where an appropriate definition of continuity in the context of the spaces $\widehat{W}, W, V_{\Gamma,h}, V_{h,e}, \widehat{V}_{h,e}$ and V_h is provided. An illustration is given in Figure 4.3.

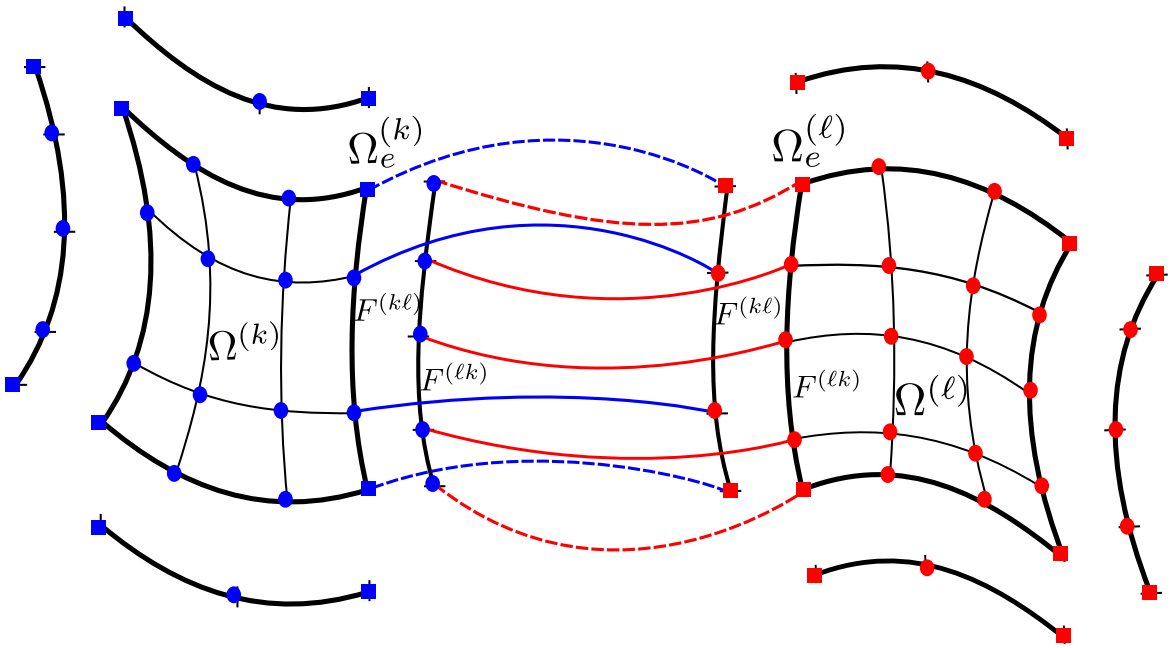


Figure 4.3: Illustration of the Lagrange multipliers used to enforce continuity of the local solutions. The dashed line indicates the continuity of the corner values (■) as incorporated into $\widetilde{W}$.

Definition 4.3. *We say that $u \in V_{h,e}$ is continuous on Γ_e if the relations*

$$\boldsymbol{u}_i^{(k,k)} = \boldsymbol{u}_j^{(l,k)} \quad \forall (i,j) \in \mathcal{B}_e(k,l),\ \forall l \in \mathcal{I}_{\mathcal{F}}^{(k)} \tag{4.11}$$

and

$$\boldsymbol{u}_i^{(k,l)} = \boldsymbol{u}_j^{(l,l)} \quad \forall (i,j) \in \mathcal{B}_e(l,k),\ \forall l \in \mathcal{I}_{\mathcal{F}}^{(k)} \tag{4.12}$$

hold for all $k \in \{1,\dots,N\}$. We denote the set of index pairs (i,j) such that the i-th basis function in $V_h^{(k)}$ can be identified with the j-th basis function in $V_h^{(l)}(\overline{F}^{(kl)})$ by $\mathcal{B}_e(k,l)$. We note that $\mathcal{B}_e(k,l) \neq \mathcal{B}_e(l,k)$. Moreover, $\widehat{V}_{h,e}$ denotes the subspace of continuous functions on Γ_e of $V_{h,e}$. Furthermore, $\widehat{V}_{h,e}$ can be identified with V_h.

The operator $B : W \to U^* := \mathbb{R}^\Lambda$, which realizes constraints (4.11) and (4.12) of the form $Bu = 0$ is called *jump operator*.

The space of all functions in W which belong to the kernel of B is denoted by $\widehat{W}$, and can be identified with $V_{\Gamma,h}$, i.e.,

$$\widehat{W} := \{w \in W \,|\, Bw = 0\} \equiv V_{\Gamma,h}.$$

Furthermore, we define by $\widehat{W}^{(k)}$ the restriction of $\widehat{W}$ to $\Omega_e^{(k)}$.

Now we are in the position to reformulate (2.20) in terms of $\widehat{V}_{h,e}$, leading to the system

$$\widehat{\boldsymbol{K}}_e \boldsymbol{u}_e = \widehat{\boldsymbol{f}}_e, \tag{4.13}$$

where the matrix $\widehat{\boldsymbol{K}}_e$ and the right-hand side $\widehat{\boldsymbol{f}}_e$ are given by the assembly of the patch-wise matrices $\boldsymbol{K}_e^{(k)}$ and right-hand sides $\boldsymbol{f}_e^{(k)}$, i.e.,

$$\widehat{\boldsymbol{K}}_e = \sum_{k=1}^{N} \boldsymbol{A}_{\Omega_e^{(k)}} \boldsymbol{K}_e^{(k)} \boldsymbol{A}_{\Omega_e^{(k)}}^T \text{ and } \quad \widehat{\boldsymbol{f}}_e = \sum_{k=1}^{N} \boldsymbol{A}_{\Omega_e^{(k)}} \boldsymbol{f}_e^{(k)}, \tag{4.14}$$

respectively. Here $\boldsymbol{A}_{\Omega_e^{(k)}}$ denotes the Boolean patch assembling matrix for $\Omega_e^{(k)}$. By means of the local Schur complements $\boldsymbol{S}_e^{(k)}$, see (4.4) and (4.5), we can reformulate equation (4.13) as

$$\widehat{\boldsymbol{S}}_e u_{B_e} = \widehat{\boldsymbol{g}}_e, \tag{4.15}$$

where $\widehat{\boldsymbol{S}}_e$ and $\widehat{\boldsymbol{g}}_e$ are given by

$$\widehat{\boldsymbol{S}}_e = \left(\sum_{k=1}^{N} \boldsymbol{A}_{\Gamma_e^{(k)}} \boldsymbol{S}_e^{(k)} \boldsymbol{A}_{\Gamma_e^{(k)}}^T \right) \text{ and } \quad \widehat{\boldsymbol{g}}_e = \sum_{k=1}^{N} \boldsymbol{A}_{\Gamma_e^{(k)}} \boldsymbol{g}_e^{(k)}. \tag{4.16}$$

The Boolean matrix $\boldsymbol{A}_{\Gamma_e^{(k)}}$ is the corresponding assembling matrix. The vector $\boldsymbol{g}^{(k)}$ is defined by $\boldsymbol{g}^{(k)} = \boldsymbol{f}_{e,B_e}^{(k)} - \boldsymbol{K}_{e,B_eI}^{(k)} \left(\boldsymbol{K}_{e,II}^{(k)} \right)^{-1} \boldsymbol{f}_{e,I}^{(k)}$. Furthermore, we can rewrite (4.16) in variational form as

$$\sum_{k=1}^{N} \langle S_e^{(k)} u_{B_e}^{(k)}, v^{(k)} \rangle = \sum_{k=1}^{N} \langle g_e^{(k)}, v^{(k)} \rangle \quad \forall v \in \widehat{W}, \tag{4.17}$$

where $u_{B_e} \in \widehat{W}$, $g_e^{(k)} \in \widehat{W}^{(k)*}$ and $S_e^{(k)} : \widehat{W}^{(k)} \to \widehat{W}^{(k)*}$.

In order to formulate the IETI-DP algorithm, we also define the Schur complement and the right-hand side functional on the "discontinuous" space W, i.e.,

$$S_e : W \to W^*, \quad \langle S_e v, w \rangle := \sum_{k=1}^{N} \langle S_e^{(k)} v^{(k)}, w^{(k)} \rangle \quad \forall v, w \in W,$$

and

$$g_e \in W^*, \quad \langle g_e, w \rangle := \sum_{k=1}^{N} \langle g_e^{(k)}, w^{(k)} \rangle \quad \forall w \in W.$$

In matrix and vector forms, we can write $\boldsymbol{S}$ and $\boldsymbol{g}$ as

$$\boldsymbol{S}_e := \operatorname{diag}(\boldsymbol{S}_e^{(k)})_{k=1}^{N} \text{ and } \quad \boldsymbol{g}_e := [\boldsymbol{g}_e^{(k)}]_{k=1}^{N}.$$

It is easy to see that problem (4.15) is equivalent to the minimization problem

$$u_{B_e} = \underset{w \in W, Bw=0}{\operatorname{argmin}} \frac{1}{2}\langle S_e w, w\rangle - \langle g_e, w\rangle. \tag{4.18}$$

In the following, we will only work with the Schur complement system. In order to simplify the notation, we will use u instead of u_B, when we consider functions in $V_{\Gamma,h}$. If we have to made a distinction between u, u_B and u_I, we will add the subscripts again.

4.1.4 Intermediate Space and Primal Constraints

The crucial point of the dual-primal approach is the definition of an intermediate space $\widetilde{W}$ in the sense $\widehat{W} \subset \widetilde{W} \subset W$ such that S_e restricted to $\widetilde{W}$ is positive definite. The construction of these spaces is identical as in Section 3.1.4, for completeness, we repeat the definitions again. Let $\Psi \subset V_{\Gamma,h}^*$ be a set of linearly independent *primal variables*. Then we define the spaces

$$\widetilde{W} := \{w \in W : \psi(w^{(k)}) = \psi(w^{(l)}), \forall \psi \in \Psi,\, k, l \in \{1, \dots, N\} \text{ with } k > l\}$$

and

$$W_\Delta := \prod_{k=1}^{N} W_\Delta^{(k)}, \text{ with } \quad W_\Delta^{(k)} := \{w^{(k)} \in W^{(k)} :\, \psi(w^{(k)}) = 0\ \forall \psi \in \Psi\}.$$

Moreover, we introduce the space $W_\Pi \subset \widehat{W}$ such that $\widetilde{W} = W_\Pi \oplus W_\Delta$. We call W_Π *primal space* and W_Δ *dual space*. If we choose Ψ such that $\widetilde{W} \cap \ker(S_e) = \{0\}$, then

$$\widetilde{S}_e : \widetilde{W} \to \widetilde{W}^*, \text{ with } \langle \widetilde{S}_e v, w\rangle = \langle S_e v, w\rangle \quad \forall v, w \in \widetilde{W},$$

is invertible and we assume that such a set is chosen. Since we are considering non-conforming test spaces $\widehat{W}$ and V_h, we cannot literally use the same set of primal variables as presented in Section 3.1. As proposed in [52] and [57], we will use the following interpretation of continuity at corners, and continuity of edge and face averages. We refer to Figure 4.3 for an illustration of continuity at the vertices, cf. Figure 4.1 in [52].

Definition 4.4. *Let $\mathcal{V}^{(k)}$, $\mathcal{E}^{(k)}$ and $\mathcal{F}^{(k)}$ be the set of vertices, edges and faces, respectively, for the patch $\Omega_e^{(k)}$.*

We say that $u \in W$ is continuous at $\mathcal{V}^{(k)}$, $k \in \{1, \dots, N\}$, if the relations

$$\boldsymbol{u}_i^{(k,k)} = \boldsymbol{u}_j^{(l,k)} \quad \forall (i,j) \in \mathcal{B}_\mathcal{V}(k,l) \tag{4.19}$$

are valid for all $l \in \mathcal{I}_{\mathcal{F}}^{(k)}$*, where* $\mathcal{B}_{\mathcal{V}}(k,l) \subset \mathcal{B}(k,l)$ *is given by all index pairs corresponding to the vertices* $\mathcal{V}^{(k)}$*. We define the corresponding primal variable as*

$$\psi^{\nu^{(kl)}}(v) := \begin{cases} \boldsymbol{v}_i^{(k,k)} & \text{if } v \in W^{(k)}, \\ \boldsymbol{v}_j^{(l,k)} & \text{if } v \in W^{(l)}, \\ 0 & \text{else}, \end{cases} \tag{4.20}$$

where $l \in \mathcal{I}_{\mathcal{F}}^{(k)}$*,* $\nu^{(kl)} \in \mathcal{V}^{(kl)}$ *and* $(i,j) \in \mathcal{B}_{\mathcal{V}}(k,l)$ *corresponds to* $\nu^{(kl)}$*.*

We say that $u \in W$ *has continuous (inter-)face averages at* $\mathcal{F}^{(k)}$*,* $k \in \{1,\dots,N\}$*, if the relations*

$$\frac{1}{|F^{(kl)}|}\int_{F^{(kl)}} u^{(k,k)}\,ds = \frac{1}{|F^{(kl)}|}\int_{F^{(kl)}} u^{(l,k)}\,ds \tag{4.21}$$

hold for all $l \in \mathcal{I}_{\mathcal{F}}^{(k)}$*. We define the corresponding primal variable as*

$$\psi^{F^{(kl)}}(v) := \begin{cases} \frac{1}{|F^{(kl)}|}\int_{F^{(kl)}} u^{(k,k)}\,ds & \text{if } v \in W^{(k)}, \\ \frac{1}{|F^{(kl)}|}\int_{F^{(kl)}} u^{(l,k)}\,ds & \text{if } v \in W^{(l)}, \\ 0 & \text{else}, \end{cases} \tag{4.22}$$

where $l \in \mathcal{I}_{\mathcal{F}}^{(k)}$*.*

We say that $u \in W$ *has continuous edge averages at* $\mathcal{E}^{(k)}$*,* $k \in \{1,\dots,N\}$*, if the relations*

$$\frac{1}{|E^{(klm)}|}\int_{E^{(klm)}} u^{(k,k)}\,ds = \frac{1}{|E^{(klm)}|}\int_{E^{(klm)}} u^{(l,k)}\,ds, \tag{4.23}$$

$$\frac{1}{|E^{(klm)}|}\int_{E^{(klm)}} u^{(k,k)}\,ds = \frac{1}{|E^{(klm)}|}\int_{E^{(klm)}} u^{(m,k)}\,ds \tag{4.24}$$

hold for all $(l,m) \in \mathcal{E}^{(k)}$*. We define the corresponding primal variable as*

$$\psi^{E^{(klm)}}(v) := \begin{cases} \frac{1}{|E^{(klm)}|}\int_{E^{(klm)}} u^{(k,k)}\,ds & \text{if } v \in W^{(k)}, \\ \frac{1}{|E^{(klm)}|}\int_{E^{(klm)}} u^{(l,k)}\,ds & \text{if } v \in W^{(l)}, \\ \frac{1}{|E^{(klm)}|}\int_{E^{(klm)}} u^{(m,k)}\,ds & \text{if } v \in W^{(m)}, \\ 0 & \text{else}, \end{cases} \tag{4.25}$$

where $(l,m) \in \mathcal{E}^{(k)}$*.*

By means of Definition 4.4, we can now introduce different sets of primal variables:

- Algorithm A: $\Psi^A := \{\psi^\nu,\ \forall \nu \in \mathcal{V}^{(k)}\}_{k=1}^N$,
- Algorithm B: $\Psi^B := \{\psi^\nu,\ \forall \nu \in \mathcal{V}^{(k)}\}_{k=1}^N \cup \{\psi^E,\ \forall E \in \mathcal{E}^{(k)}\}_{k=1}^N \cup \{\psi^F,\ \forall F \in \mathcal{E}^{(k)}\}_{k=1}^N$,

- Algorithm C: $\Psi^C := \{\psi^\nu,\ \forall \nu \in \mathcal{V}^{(k)}\}_{k=1}^N \cup \{\psi^E,\ \forall E \in \mathcal{E}^{(k)}\}_{k=1}^N$.

Remark 4.5. *As mentioned in Section 3.1 for the cG-IETI-DP method, one can introduce primal sets, which aim at reducing the number of primal variables, often denoted by Algorithm D and E.*

4.1.5 IETI-DP and preconditioning

Since $\widetilde{W} \subset W$, there is a natural embedding $\widetilde{I} : \widetilde{W} \to W$. We define the jump operator restricted to $\widetilde{W}$ as $\widetilde{B} := B\widetilde{I} : \widetilde{W} \to U^*$. Then we can formulate problem (4.18) as saddle-point problem in $\widetilde{W}$ as follows: find $(u, \lambda) \in \widetilde{W} \times U$:

$$\begin{bmatrix} \widetilde{S}_e & \widetilde{B}^T \\ \widetilde{B} & 0 \end{bmatrix} \begin{bmatrix} u \\ \lambda \end{bmatrix} = \begin{bmatrix} \widetilde{g} \\ 0 \end{bmatrix}, \tag{4.26}$$

where $\widetilde{g} := \widetilde{I}^T g$, and $\widetilde{B}^T = \widetilde{I}^T B^T$. By construction, $\widetilde{S}_e$ is SPD on $\widetilde{W}$. Hence, the saddle-point system (4.26) is equivalent to the Schur complement problem:

$$\text{find } \lambda \in U : \quad F\lambda = d, \tag{4.27}$$

with the Schur complement $F := \widetilde{B}\widetilde{S}_e^{-1}\widetilde{B}^T$ and the corresponding right-hand side $d := \widetilde{B}\widetilde{S}_e^{-1}\widetilde{g}$. Equation (4.27) is solved by means of the PCG algorithm, but it requires an appropriate preconditioner in order to obtain an efficient solver. According to [52] and [57], the right choice for FE is the *scaled Dirichlet preconditioner*, adapted to the extended set of dofs. A rigorous proof for two-dimensional domains, having globally constant diffusion coefficient and using Algorithm A will be given in Section 4.2. The numerical tests presented in Section 4.3 indicate that the scaled Dirichlet preconditioner works well also for other sets of primal variables and diffusion coefficients, which have jumps across the patch interfaces.

Recall the definition of $S_e = \operatorname{diag}(S_e^{(k)})_{k=1}^N$, we define the scaled Dirichlet preconditioner M_{sD}^{-1} as

$$M_{sD}^{-1} := B_D S_e B_D^T, \tag{4.28}$$

where B_D is a scaled version of the jump operator B. The scaled jump operator B_D is defined such that the operator enforces the constraints

$$\delta^{\dagger}_j{}^{(l)} \boldsymbol{u}_i^{(k,k)} - \delta^{\dagger}_i{}^{(k)} \boldsymbol{u}_j^{(l,k)} = 0 \quad \forall (i,j) \in \mathcal{B}_e(k,l),\ \forall l \in \mathcal{I}_{\mathcal{F}}^{(k)}, \tag{4.29}$$

and

$$\delta^{\dagger}_j{}^{(l)} \boldsymbol{u}_i^{(k,l)} - \delta^{\dagger}_i{}^{(k)} \boldsymbol{u}_j^{(l,l)} = 0 \quad \forall (i,j) \in \mathcal{B}_e(l,k),\ \forall l \in \mathcal{I}_{\mathcal{F}}^{(k)}, \tag{4.30}$$

where, for $(i,j) \in \mathcal{B}_e(k,l)$,

$$\delta^{\dagger}{}_i^{(k)} := \frac{\rho_i^{(k)}}{\sum_{l \in \mathcal{I}_{\mathcal{F}}^{(k)}} \rho_j^{(l)}}$$

is an appropriate scaling. As in Section 3.1.6, typical choices for $\rho_i^{(k)}$ are

- Multiplicity Scaling: $\rho_i^{(k)} := 1$,
- Coefficient Scaling: If $\alpha(x)_{|\Omega^{(k)}} = \alpha^{(k)}$, choose $\rho_i^{(k)} := \alpha^{(k)}$,
- Stiffness Scaling: $\rho_i^{(k)} := \boldsymbol{K}_{e\,i,i}^{(k)}$.

Since we can consider the dG-IETI-DP method as a conforming Galerkin (cG) method on an extended grid, we can implement the dG-IETI-DP algorithm following the implementation of the corresponding cG-IETI-DP method given in Section 3.2. For completeness, we give an outline of the algorithm. The Schur complement system (4.27) is solved using a CG algorithm with the preconditioner given in (4.28). The application of F and M_{sD}^{-1} is outlined in Algorithm 4 and Algorithm 5, cf. Algorithm 2 and Algorithm 3 in Chapter 3 for the cG-IETI-DP method.

Algorithm 4 Algorithm for the calculation of $\nu = F\lambda$ for given $\lambda \in U$

procedure $F(\lambda)$
  Application of $B^T : \{f^{(k)}\}_{k=1}^N = B^T\lambda$
  Application of $\widetilde{I}^T : \{\boldsymbol{f}_\Pi, \{f_\Delta^{(k)}\}_{k=1}^N\} = \widetilde{I}^T\left(\{f^{(k)}\}_{k=1}^N\right)$
  Application of $\widetilde{S}_e^{-1}$:
  Begin
    $\boldsymbol{w}_\Pi = \boldsymbol{S}_{e,\Pi\Pi}^{-1}\boldsymbol{f}_\Pi$
    $w_\Delta^{(k)} = {S_{e,\Delta\Delta}^{(k)}}^{-1} f_\Delta^{(k)} \quad \forall k = 1,\ldots,N$
  End
  Application of $\widetilde{I} : \{w^{(k)}\}_{k=1}^N = \widetilde{I}\left(\{\boldsymbol{w}_\Pi, \{w_\Delta^{(k)}\}_{k=1}^N\}\right)$
  Application of $B : \nu = B\left(\{w^{(k)}\}_{k=1}^N\right)$
end procedure

4.2 Analysis of the Condition Number

This section deals with the analysis of the condition number for the discontinuous Galerkin variant of the IETI-DP algorithm.

In [52] and [57], it is proven for FE that the condition number of the preconditioned dG-FETI-DP method has the same polylogarithmic bound with respect to H/h as the

Algorithm 5 Algorithm for the calculation of $\nu = M_{sD}^{-1}\lambda$ for given $\lambda \in U$

procedure $M_{sD}^{-1}(\lambda)$
Application of $B_D^T : \{w^{(k)}\}_{k=1}^N = B_D^T\lambda$
Application of S_e :
Begin
Solve: $K_{e,II}^{(k)}x^{(k)} = -K_{e,IB}^{(k)}w^{(k)}$
$v^{(k)} = K_{e,BB}^{(k)}w^{(k)} + K_{e,BI}^{(k)}x^{(k)}$.
End
Application of $B_D : \nu = B_D\left(\{v^{(k)}\}_{k=1}^N\right)$
end procedure

continuous FETI-DP method, see also [51] for dG-BDDC FE preconditioners. From Section 3.3, we know that the condition number of the continuous IETI-DP and BDDC-IgA operators is also quasi-optimal with respect to the ratio of patch and mesh-size. We prove that the condition number for the dG-IETI-DP operator behaves as

$$\kappa(M_{sD}^{-1}F_{|\ker(\widetilde{B}^T)}) \leq C \max_k \left(1 + \log\left(\frac{H_k}{h_k}\right)\right)^2,$$

where H_k and h_k are the patch size and mesh-size, respectively, and the positive constant C is independent of H_k, h_k, but depends on h_k/h_l and δ. However, the bound proven in [52] for the FE version is independent of the two parameters δ and h_k/h_l. We use the fact that the IETI-DP method and the BDDC preconditioner have the same spectrum, up to some zeros and ones, which was proven in [159] based on algebraic arguments. So we will prove the condition number bound for the corresponding BDDC method and the result then also applies to the dG-IETI-DP method. We can use the framework developed in [17] also for the dG variant, since the dG-IETI-DP method can be seen as a IETI-DP method on an extended space W. In the next section we provide some auxiliary results, which will be needed for the proof in Section 4.2.2.

4.2.1 Auxiliary Results

In this section, we define a discrete norm $|\cdot|_{k,dG}$ for the space $V_{h,e}^{(k)}$, based on the coefficient vector $\boldsymbol{u}^{(k)} = (\boldsymbol{u}_i^{(k)})_{i\in\mathcal{I}_e^{(k)}}$, which can be seen as the discrete analogue of the norm induced by $d^{(k)}(\cdot,\cdot)$, which is denoted by $\|\cdot\|_{k,dG}$, see also Section 2.3.2. The difficulty is that the grids on $F^{(kl)}$ and $F^{(lk)}$ do not match and, hence, the coefficients corresponding to that part cannot be directly related. We will resolve that issue using a L^2-projection onto $F^{(lk)}$. In the following, we will always focus on the case $d = 2$ with continuous vertex values only.

We rephrase important definitions and results from [19] and Section 3.3 with small adjustment due to considering the dG-formulation.

Let h_k and $\hat{h}_k$ be the characteristic mesh-sizes in $\Omega^{(k)}$ and $\hat{\Omega}^{(k)}$, respectively. Since the geometry mapping $G^{(k)}$ is fixed on a coarse discretization, it is independent of h_k. Moreover, by basic properties of $G^{(k)}$, we can assume that there exists a constant C, independent of H_k and h_k, such that

$$C^{-1}\hat{h}_k \leq h_k/H_k \leq C\hat{h}_k, \tag{4.31}$$

where H_k is the diameter of $\Omega^{(k)}$. Given a face $F^{(kl)}$ in $\Omega^{(k)}$ with diameter $H_{F^{(lk)}}$, we denote its parameter domain representation as $\hat{F}^{(kl)}$. The mesh-size on $F^{(kl)}$ and $F^{(lk)}$ is given by $h_{F^{(kl)}}$ and $h_{F^{(kl)}}$, respectively. Moreover, we assume for all $l \in \mathcal{I}_{\mathcal{F}}^{(k)}$ that $h_{F^{(kl)}} \approx h_k$, $h_{F^{(lk)}} \approx h_l$ and $H_k \approx H_{F^{(kl)}} \approx H_l$. Together with (4.31), it follows that

$$h_{kl} \approx H_k\hat{h}_{kl} \approx H_l\hat{h}_{kl}. \tag{4.32}$$

An illustration is given in Figure 4.1.

Assumption 4. We assume that there exists a constant $\beta \in \mathbb{R}^+$ such that $\beta^{-1}M_2^{(k)} \leq M_1^{(k)} \leq \beta M_2^{(k)}$, where $M_1^{(k)}$ and $M_2^{(k)}$ are the number of basis functions on $\Omega^{(k)}$ along dimension 1 and 2, respectively, see Section 2.2.

According to [19] and Section 3.3, we define a discrete norm and semi-norm based on the coefficients $(\boldsymbol{u}_i)_{i\in\mathcal{I}}$, which should mimic the L_2 norm and H^1 semi-norm, respectively.

Definition 4.6. *Let $u^{(k)} \in V_{h,e}^{(k)}$, and let $\hat{u}^{(k)}$ be its counterpart in $S_{h,e}^{(k)}$. We define the discrete norm $\|\cdot\|_{k,\square}^2$ and semi-norm $|\cdot|_{k,\nabla}^2$ on $S_{h,e}^{(k)}$ as*

$$\begin{aligned}
\|\hat{u}^{(k,k)}\|_{k,\square}^2 &:= \sum_{i\in\mathcal{I}^{(k,k)}} |\boldsymbol{u}_i^{(k,k)}|^2\hat{h}_k^2, \\
|\hat{u}^{(k,k)}|_{k,\nabla}^2 &:= \sum_{i\in\mathcal{I}_1^{(k,k)}} |\boldsymbol{u}_{(i_1,i_2)}^{(k,k)} - \boldsymbol{u}_{(i_1-1,i_2)}^{(k,k)}|^2 + \sum_{i\in\mathcal{I}_2^{(k,k)}} |\boldsymbol{u}_{(i_1,i_2)}^{(k,k)} - \boldsymbol{u}_{(i_1,i_2-1)}^{(k,k)}|^2,
\end{aligned}$$

where for $i \in \mathcal{I}_1^{(k,k)}, \mathcal{I}_2^{(k,k)} \subset \mathcal{I}^{(k,k)}$ are defined such that $\boldsymbol{u}_{(i_1-1,i_2)}^{(k,k)}$ and $\boldsymbol{u}_{(i_1,i_2-1)}^{(k,k)}$ are well defined, respectively.

Analogously, we define the discrete norm $\|\cdot\|_{lk}^2$ on the space $S_h^{(lk)}$, as

$$\|\hat{u}^{(k,l)}\|_{lk}^2 := \sum_{i\in\mathcal{I}^{(k,l)}} |\boldsymbol{u}_i^{(k,l)}|^2\hat{h}_l.$$

Proposition 4.7. *Let $u^{(k)} \in V_{h,e}^{(k)}$, and let $\hat{u}^{(k)}$ be its counterpart in $S_{h,e}^{(k)}$. We have that*

$$\begin{aligned}
\|\hat{u}^{(k,k)}\|_{k,\square}^2 &\approx \|\hat{u}^{(k,k)}\|_{L^2((0,1)^2)}^2 \approx H_k^{-2}\|u^{(k,k)}\|_{L^2(\Omega^{(k)})}^2, \\
|\hat{u}^{(k,k)}|_{k,\nabla}^2 &\approx |\hat{u}^{(k,k)}|_{H^1((0,1)^2)}^2 \approx |u^{(k,k)}|_{H^1(\Omega^{(k)})}^2, \\
\|\hat{u}^{(k,l)}\|_{lk}^2 &\approx \|\hat{u}^{(k,l)}\|_{L^2((0,1))}^2 \approx H_k^{-1}\|u^{(k,l)}\|_{L^2(F^{(kl)})}^2,
\end{aligned}$$

where the hidden constants do not depend on h_k or H_k.

Proof. Follows directly from Corollary 5.1. and Proposition 5.2. in [19], Assumption 3 and the equivalence between of norms in the parameter and physical space, see Corollary 2.12. □

Next, we define the L^2-projection, in order to provide an approximation of $u^{(k,l)}$ on $F^{(lk)}$.

Definition 4.8. *We denote the L^2-orthogonal projection onto $V_h^{(lk)}$ by $\pi_{F^{(lk)}} : L^2(F^{(kl)}) \to V_h^{(lk)}$ and its coefficients by $\tilde{\boldsymbol{u}}_i^{(k,l)}$, i.e.,*

$$\pi_{F^{(lk)}} v := \sum_{i \in \mathcal{I}^{(k,l)}} \tilde{\boldsymbol{u}}_i^{(k,l)} N_{i,p}^{(l)}|_{F^{(kl)}}. \tag{4.33}$$

Lemma 4.9. *Let $v^{(k)} \in V_h^{(k)}$ and $\pi_{F^{(lk)}}$ be as in Definition 4.8. Then the estimate*

$$\|v^{(k)} - \pi_{F^{(lk)}} v^{(k)}\|^2_{L^2(F^{(kl)})} \leq C h_l \frac{h_l}{h_k} |v^{(k)}|^2_{H^1(\Omega^{(k)})}$$

holds, where the generic constant C is independent of h_k, h_l or H_k.

Proof. Since $\pi_{F^{(lk)}}$ is the L^2-orthogonal projection, we have that

$$\|v^{(k)} - \pi_{F^{(lk)}} v^{(k)}\|^2_{L^2(F^{(kl)})} \leq \|v^{(k)} - \mathcal{I}_{F^{(lk)}} v^{(k)}\|^2_{L^2(F^{(kl)})},$$

where $\mathcal{I}_{F^{(lk)}} : L^2(F^{(kl)}) \to V_h^{(lk)}$ is the B-Spline quasi-interpolant. By means of the interpolation estimate

$$\|v^{(k)} - \mathcal{I}_{F^{(lk)}} v^{(k)}\|^2_{L^2(F^{(kl)})} \leq C h_l^2 |v^{(k)}|^2_{H^1(F^{(kl)})}$$

and the discrete trace inequality, see, e.g., Lemma 4.3. in [59],

$$|v^{(k)}|^2_{H^1(F^{(kl)})} \leq C h_k^{-1} |v^{(k)}|^2_{H^1(\Omega^{(k)})},$$

we have

$$\|v^{(k)} - \pi_{F^{(lk)}} v^{(k)}\|^2_{L^2(F^{(kl)})} \leq C h_l^2 |v^{(k)}|^2_{H^1(F^{(kl)})} \leq C h_l \frac{h_l}{h_k} |v^{(k)}|^2_{H^1(\Omega^{(k)})},$$

which proves the estimate. □

Now, we are in the position to define the discrete dG-norm, and prove bounds in terms of $\|\cdot\|_{k,dG}$, as introduced in the beginning of Section 4.2.1.

Definition 4.10. *Let $u^{(k)} \in V_{h,e}^{(k)}$, and let $\hat{u}^{(k)}$ be its counterpart in $S_{h,e}^{(k)}$. Moreover, for $l \in \mathcal{I}_{\mathcal{F}}^{(k)}$, let $\pi_{F^{(lk)}} u_{F^{(kl)}}^{(k,k)}$ be the L^2-projection onto $V_h^{(lk)}$ according to Definition 4.8*

with coefficients $(\tilde{\boldsymbol{u}}_i^{(k,l)})_{i\in\mathcal{I}^{(k,l)}}$ *as in (4.33). We define the discrete dG-norm* $|\cdot|_{k,dG}$ *on* $V_{h,e}^{(k)}$ *as*

$$|\hat{u}^{(k)}|_{k,dG}^2 := |\hat{u}^{(k,k)}|_{k,\nabla}^2 + \sum_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}} \frac{\delta}{\hat{h}_{kl}} \| \ u^{(k,l)} - \pi_{F^{(lk)}} \ u^{(k,k)}_{|F^{(kl)}} \|_{lk}^2 \tag{4.34}$$

$$= |\hat{u}^{(k,k)}|_{k,\nabla}^2 + \sum_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}} \frac{\delta}{\hat{h}_{kl}} \sum_{i\in\mathcal{I}^{(k,l)}} |\boldsymbol{u}_i^{(k,l)} - \tilde{\boldsymbol{u}}_i^{(k,l)}|^2 \hat{h}_l, \tag{4.35}$$

where δ *is as in (2.17).*

Proposition 4.11. *Let* $u^{(k)} \in V_{h,e}^{(k)}$*, and let* $\hat{u}^{(k)}$ *be its counterpart in* $S_{h,e}^{(k)}$*. Then we have*

$$|\hat{u}^{(k)}|_{k,dG}^2 \leq C \|u^{(k)}\|_{k,dG}^2, \tag{4.36}$$

and

$$\|u^{(k)}\|_{k,dG}^2 \leq C\delta q_h^{(k)} |\hat{u}^{(k)}|_{k,dG}^2, \tag{4.37}$$

where $q_h^{(k)} := \max_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}} \left\{1, \left(\frac{h_l}{h_k} + \frac{h_l^2}{h_k^2}\right)\right\}$ *and the generic constant* C *is independent of* h_k, H_k *and* δ*.*

Proof. We first prove (4.36). The discrete dG-norm is defined as

$$|\hat{u}^{(k)}|_{k,dG}^2 = |\hat{u}^{(k,k)}|_{k,\nabla}^2 + \sum_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}} \frac{\delta}{\hat{h}_{kl}} \|\hat{u}^{(k,l)} - \pi_{F^{(lk)}} \hat{u}^{(k,k)}_{|F^{(kl)}}\|_{lk}^2,$$

where we can immediately bound the first term according to Proposition 4.7 by

$$|\hat{u}^{(k,k)}|_{k,\nabla}^2 \leq C |u^{(k,k)}|_{H^1(\Omega^{(k)})}^2. \tag{4.38}$$

For the second term, it holds

$$\begin{aligned}
\sum_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}} \frac{\delta}{\hat{h}_{kl}} \|\hat{u}^{(k,l)} - \pi_{F^{(lk)}} \hat{u}^{(k,k)}_{|F^{(kl)}}\|_{lk}^2 \leq & C \sum_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}} \frac{\delta H_k}{h_{kl}} \|\hat{u}^{(k,l)} - \pi_{F^{(lk)}} \hat{u}^{(k,k)}_{|F^{(kl)}}\|_{lk}^2 \\
\leq & C \sum_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}} \frac{\delta}{h_{kl}} \|u^{(k,l)} - \pi_{F^{(lk)}} u^{(k,k)}_{|F^{(kl)}}\|_{L^2(F^{(kl)})}^2 \\
= & C \sum_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}} \frac{\delta}{h_{kl}} \|\pi_{F^{(lk)}} \left(u^{(k,l)} - u^{(k,k)}_{|F^{(kl)}}\right)\|_{L^2(F^{(kl)})}^2 \\
\leq & C \sum_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}} \frac{\delta}{h_{kl}} \|u^{(k,l)} - u^{(k,k)}_{|F^{(kl)}}\|_{L^2(F^{(kl)})}^2,
\end{aligned} \tag{4.39}$$

where we used (4.32), Proposition 4.7, the fact that $\pi_{F^{(lk)}} u^{(k,l)} = u^{(k,l)}$ and the stability of the L^2-projection in the L^2 norm. Combining (4.38) and (4.39) gives

$$|\hat{u}^{(k)}|^2_{k,dG} \leq C\Big(|u^{(k,k)}|^2_{H^1(\Omega^{(k)})} + \sum_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}} \frac{\delta}{h_{kl}} \|u^{(k,l)} - u^{(k,k)}_{|F^{(kl)}}\|^2_{L^2(F^{(kl)})}\Big) = C\|u^{(k)}\|^2_{k,dG},$$

where C is a generic constant independent of h and H. We now proof the second estimate (4.37). The dG-norm reads

$$\|u^{(k)}\|^2_{k,dG} = |u^{(k)}|^2_{H^1(\Omega^{(k)})} + \sum_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}} \frac{\delta}{h_{kl}} \|u^{(k,l)} - u^{(k,k)}_{|F^{(kl)}}\|^2_{L^2(F^{(kl)})}.$$

Similar as before, we bound the first term by means of Proposition 4.7 via

$$|u^{(k)}|^2_{H^1(\Omega^{(k)})} \leq C|\hat{u}^{(k,k)}|^2_{k,\nabla}. \tag{4.40}$$

For the second term, we have for $l \in \mathcal{I}_{\mathcal{F}}^{(k)}$ by means of the triangle inequality

$$\|u^{(k,l)} - u^{(k,k)}_{|F^{(kl)}}\|^2_{L^2(F^{(kl)})} \leq \|u^{(k,l)} - \pi_{F^{(lk)}} u^{(k,k)}_{|F^{(kl)}}\|^2_{L^2(F^{(kl)})} + \|\pi_{F^{(lk)}} u^{(k,k)}_{|F^{(kl)}} - u^{(k,k)}_{|F^{(kl)}}\|^2_{L^2(F^{(kl)})}. \tag{4.41}$$

The first term in the right-hand side of (4.41) can be estimated by means of Proposition 4.7 by

$$\|u^{(k,l)} - \pi_{F^{(lk)}} u^{(k,k)}_{|F^{(kl)}}\|^2_{L^2(F^{(kl)})} \leq CH_k \|\hat{u}^{(k,l)} - \pi_{F^{(lk)}} \hat{u}^{(k,k)}_{|F^{(kl)}}\|^2_{lk}. \tag{4.42}$$

For the second term in (4.41), Lemma 4.9 yields

$$\|\pi_{F^{(lk)}} u^{(k,k)}_{|F^{(kl)}} - u^{(k,k)}_{|F^{(kl)}}\|^2_{L^2(F^{(kl)})} \leq Ch_l \frac{h_l}{h_k} |u^{(k,k)}|^2_{H^1(\Omega^{(k)})}.$$

Since $u^{(k,k)} \in V_h^{(k)}$ and according to Proposition 4.7, we obtain

$$\|\pi_{F^{(lk)}} u^{(k,k)}_{|F^{(kl)}} - u^{(k,k)}_{|F^{(kl)}}\|^2_{L^2(F^{(kl)})} \leq Ch_l \frac{h_l}{h_k} |\hat{u}^{(k,k)}|^2_{k,\nabla}. \tag{4.43}$$

Combining (4.41) with (4.42) and (4.43) and using $\frac{h_l}{h_{kl}} \frac{h_l}{h_k} = \frac{1}{2}\left(\frac{h_l}{h_k} + \frac{h_l^2}{h_k^2}\right)$, $|\mathcal{I}_{\mathcal{F}}^{(k)}| \leq 4$ and (4.32), we obtain the estimate

$$\begin{aligned}\sum_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}} \frac{\delta}{h_{kl}} \|u^{(k,l)} - u^{(k,k)}_{|F^{(kl)}}\|^2_{L^2(F^{(kl)})} \leq C\Bigg(&2\delta \max_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}} \left(\frac{h_l}{h_k} + \frac{h_l^2}{h_k^2}\right) |\hat{u}^{(k,k)}|^2_{k,\nabla} \\ &+ \sum_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}} \frac{\delta}{\hat{h}_{kl}} \|\hat{u}^{(k,l)} - \pi_{F^{(lk)}} \hat{u}^{(k,k)}_{|F^{(kl)}}\|^2_{lk}\Bigg).\end{aligned} \tag{4.44}$$

Summing up (4.40) and (4.44) concludes the proof. □

We now provide properties of the local index spaces. Since we consider only the two-dimensional problem, we can interpret the coefficients $(\boldsymbol{u}_i^{(k)})_{i\in\mathcal{I}_e^{(k)}}$ of $u^{(k)} \in V_{h,e}^{(k)}$ as a matrix plus four additional vectors for the extra boundary, i.e.,

$$\boldsymbol{C}_e^{(k)} := (\boldsymbol{C}^{(k,k)}, (\boldsymbol{c}^{(k,l)})_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}}) \in \mathcal{R}_e^{(k)} := \mathbb{R}^{M_1^{(k)}\times M_2^{(k)}} \times \prod_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}} \mathbb{R}^{M^{(lk)}},$$

where $\boldsymbol{C}^{(k,k)} := (\boldsymbol{u}_i^{(k)})_{i\in\mathcal{I}^{(k,k)}}$ and $\boldsymbol{c}^{(k,l)} := (\boldsymbol{u}_i^{(k,l)})_{i\in\mathcal{I}^{(k,l)}}$ for $l \in \mathcal{I}_{\mathcal{F}}^{(k)}$. The number $M^{(lk)}$ denotes the number of coefficients associated to $\hat{F}^{(lk)}$, i.e., $M^{(lk)} = |\mathcal{I}^{(k,l)}|$. We note that there exists a $\iota^* \in \{1,2\}$ such that $M^{(lk)} = M_{\iota^*}^{(l)}$.

The entries of the matrix $\boldsymbol{C}_e^{(k)}$ can be interpreted as values on a uniform grid $\mathcal{T}_e^{(k)} := \mathcal{T}^{(k)} \cup \bigcup_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}} \mathcal{T}^{(kl)}$ on $\overline{\hat{\Omega}}_e$, where $\mathcal{T}^{(k)}$ and $\mathcal{T}^{(kl)}$ are the grids corresponding to $\boldsymbol{C}^{(k,k)}$ and $\boldsymbol{c}^{(k,l)}$, respectively. The meshes $\mathcal{T}^{(k)}$ and $\mathcal{T}^{(kl)}$ have a characteristic mesh-size

$$\tilde{h}_k := \left(\frac{1}{(M_1^{(k)}-1)^2} + \frac{1}{(M_2^{(k)}-1)^2}\right)^{1/2} \text{ and } \tilde{h}^{(lk)} := \frac{1}{M^{(lk)}-1},$$

respectively. Hence, we have

$$C_{\beta,k}^{-1}\frac{1}{M_1^{(k)}} \le \tilde{h}_k \le C_{\beta,k}\frac{1}{M_1^{(k)}}, \quad \text{and} \quad C_{\beta,l}^{-1}\tilde{h}_l \le \tilde{h}^{(lk)} \le C_{\beta,l}\tilde{h}_l,$$

where the constants $C_{\beta,k}$ and $C_{\beta,l}$ depend only on β. By the basic properties of the geometrical mapping G and the B-Splines $\hat{N}_{i,p}$, it is easy to see that

$$C_{G,\beta}^{-1}\tilde{h}_k \le h_k/H_k \le \tilde{h}_k C_{G,\beta} \text{ and } C_\beta^{-1}\tilde{h}_k \le \hat{h}_k \le C_\beta \tilde{h}_k, \tag{4.45}$$

where the the constant C_β depends only on β, and the constant $C_{G,\beta}$ additionally also on G. Finally, we define the harmonic average $\tilde{h}_{kl} := 2\tilde{h}_k\tilde{h}_l/(\tilde{h}_k + \tilde{h}_l)$.

We are now able to introduce a dG-norm on the discrete coefficient-space $\mathcal{R}_e^{(k)}$ as follows

$$|\!|\!|\boldsymbol{C}_e^{(k)}|\!|\!|_{k,dG}^2 := |\boldsymbol{C}^{(k,k)}|_{k,\nabla}^2 + \sum_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}} \frac{\delta}{\tilde{h}_{kl}} \sum_{i=1}^{M^{(lk)}} |\boldsymbol{u}_i^{(k,l)} - \tilde{\boldsymbol{u}}_i^{(k,l)}|^2 \tilde{h}_l, \tag{4.46}$$

where $|\boldsymbol{C}^{(k,k)}|_{k,\nabla}^2$ has to be understood as applying the norm $|\cdot|_{k,\nabla}^2$ to the corresponding function in $V_h^{(k)}$ and $\tilde{\boldsymbol{u}}_i^{(k,l)}$, is defined analogously as in Definition 4.10. We note that, for given function $u \in V_{h,e}^{(k)}$ with coefficient representation $\boldsymbol{C}_e \in \mathcal{R}_e^{(k)}$, we have

$$C^{-1}|\hat{u}|_{k,dG}^2 \le |\!|\!|\boldsymbol{C}_e|\!|\!|_{k,dG}^2 \le C|\hat{u}|_{k,dG}^2, \tag{4.47}$$

where the constant C depends only on the constants $C_{G,\beta}$ and $C_{\beta,k}^{-1}$ from patch k and all its neighbouring patches.

This motivates the definition of an operator $(\cdot)_I : C(\overline{\hat{\Omega}}_e^{(k)}) \to \mathcal{R}_e^{(k)}$, where $C(\overline{\hat{\Omega}}_e^{(k)}) := C(\overline{\hat{\Omega}}^{(k)}) \times \prod_{i=1}^{4} C(\overline{\hat{F}}^{(kl)})$, which evaluates a continuous function on $\overline{\hat{\Omega}}_e^{(k)}$ in the grid points x_i of $\mathcal{T}_e$. Moreover, we introduce an operator $\chi^{(k)} : \mathcal{R}_e^{(k)} \to H^1(\hat{\Omega}_e) := H^1(\hat{\Omega}) \times \prod_{l \in \mathcal{I}_{\mathcal{F}}^{(k)}} H^1(\hat{F}^{(lk)})$, that provides a piecewise bilinear interpolation of the given grid values, i.e., $\chi^{(k)}(\boldsymbol{v}) \in \mathcal{Q}_1(\mathcal{T}_e^{(k)}) := \mathcal{Q}_1(\mathcal{T}^{(k)}) \times \prod_{l \in \mathcal{I}_{\mathcal{F}}^{(k)}} \mathcal{P}_1(\mathcal{T}^{(kl)})$. Here $\mathcal{Q}_1(\mathcal{T}^{(k)})$ is the space of piecewise bilinear functions on $\mathcal{T}^{(k)}$ and $\mathcal{P}_1(\mathcal{T}^{(kl)})$ the space of piecewise linear functions on $\mathcal{T}^{(kl)}$.

Given values on an edge $\hat{F}_e^{(kl)} := \hat{F}^{(kl)} \cup \hat{F}^{(lk)}$ and its associated grid $\mathcal{T}_e^{(kl)} := \mathcal{T}^{(k)}_{|\hat{F}^{(kl)}} \cup \mathcal{T}^{(l,k)}$, we need to define its linear interpolation and a discrete harmonic extension to the interior. In order to do so, let us denote all indices of grid points x_i associated to $\hat{F}_e^{(kl)}$ by $\mathcal{I}(\hat{F}_e^{(kl)})$. Additionally, let $\mathcal{P}_1(\mathcal{T}_e^{(kl)}) := \mathcal{P}_1(\mathcal{T}^{(kl)}) \times \mathcal{P}_1(\mathcal{T}^{(lk)})$ be the space of piecewise linear spline functions on $\mathcal{T}_e^{(kl)}$. We define the interpolation of values on $\hat{F}_e^{(kl)}$ via the restriction of the operator $\chi^{(k)}$ to $\hat{F}_e^{(kl)}$, denoted by $\chi^{(k)}_{\hat{F}_e^{(kl)}} : \mathbb{R}^{M_\iota^{(k)} + M^{(lk)}} \to H^1(\hat{F}^{(kl)}) \times H^1(\hat{F}^{(lk)})$ with an analogous definition. In a similar way, we define the interpolation operator for the whole boundary $\Gamma^{(k)}$, denoted by $\chi^{(k)}_{\Gamma,e} : \mathbb{R}^{M_\Gamma^{(k)}} \to H^1(\partial\hat{\Omega}^{(k)}) \times \prod_{l \in \mathcal{I}_{\mathcal{F}}^{(k)}} H^1(\hat{F}^{(lk)})$, where $M_\Gamma^{(k)} := 2M_1^{(k)} + 2M_2^{(k)} - 4 + \sum_{l \in \mathcal{I}_{\mathcal{F}}^{(k)}} M^{(lk)}$. Similar as in Section 3.3, we define a semi-norm for values on grid points on an edge $\hat{F}^{(kl)}$ via the interpolation to functions in $\mathcal{P}_1(\mathcal{T}_e^{(kl)})$:

Definition 4.12. *Let $\hat{F}^{(kl)}$ be an edge of $\hat{\Omega}^{(k)}$. Then we define the semi-norm $\|\boldsymbol{v}\|_{\hat{F}^{(kl)}} := |\chi^{(k)}_{\hat{F}^{(kl)}}(\boldsymbol{v})|_{H^{1/2}(\hat{F}^{(kl)})}$ for all $\boldsymbol{v} \in \mathbb{R}^{|\mathcal{I}(\hat{F}^{(kl)})|}$.*

Definition 4.13. *Let $\mathcal{H}^{(k)}_{\mathcal{Q}_{1,e}}$ be the standard discrete harmonic extension in the sense of $a_e^{(k)}(\cdot,\cdot)$ into the piecewise bilinear space $\mathcal{Q}_{1,e}$, see [52] for a formal definition. This defines the lifting operator $\boldsymbol{H}_e^{(k)} : \mathbb{R}^{M_\Gamma^{(k)}} \to \mathcal{R}_e^{(k)}$ by*

$$\boldsymbol{b} \mapsto \boldsymbol{H}_e^{(k)}(\boldsymbol{b}) := (\mathcal{H}^{(k)}_{\mathcal{Q}_{1,e}}(\chi^{(k)}_{\Gamma,e}(\boldsymbol{b})))_I.$$

Theorem 4.14. *Let $\hat{F}^{(kl)}$ be a particular side of $\partial\hat{\Omega}^{(k)}$ and Assumption 4 be fulfilled. Then the following statements hold:*

1. *For all $\boldsymbol{b} \in \mathbb{R}^{M_\Gamma^{(k)}}$ that vanish on the twelve components corresponding to the twelve corners $\mathcal{V}^{(k)}$, see Figure 4.2 and Figure 4.3, the estimate*

$$\begin{aligned}\|\boldsymbol{H}_e^{(k)}(\boldsymbol{b})\|^2_{k,dG} \leq & C\delta\tilde{q}_h^{(k)}(1+\log^2 \tilde{h}_k^{-1}) \\ & \sum_{l \in \mathcal{I}_{\mathcal{F}}^{(k)}} \Big(\|\boldsymbol{b}|_{\hat{F}^{(kl)}}\|^2_{\hat{F}^{(kl)}} + \frac{\delta}{\tilde{h}_{kl}} \sum_{i=1}^{M^{(lk)}} |b_i^{(k,l)} - \tilde{b}_i^{(k,l)}|^2 \tilde{h}_l \Big),\end{aligned}$$

holds, where $\tilde{q}_h^{(k)} := \max_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}} \left\{1, \left(\frac{\tilde{h}_l}{\tilde{h}_k} + \frac{\tilde{h}_l^2}{\tilde{h}_k^2}\right)\right\}$ *and the constant* C *does not depend on* h_k, H_k *and* δ.

2. *The estimate*

$$\|\!|\boldsymbol{C}|\!\|_{k,dG}^2 \geq C\Big(\|\!|\boldsymbol{C}|_{\hat{F}^{(kl)}}\|\!|_{\hat{F}^{(kl)}}^2 + \sum_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}} \frac{\delta}{\tilde{h}_{kl}} \sum_{i=1}^{M^{(lk)}} |\boldsymbol{u}_i^{(k,l)} - \tilde{\boldsymbol{u}}_i^{(k,l)}|^2 \tilde{h}_l\Big)$$

is valid for all $\boldsymbol{C} \in \mathcal{R}_e^{(k)}$, *and the constant* C *does not depend on* h_k, H_k *and* δ.

Proof. For a better readability, we will omit the superscript (k).

The discrete dG-norm for matrices is defined as

$$\|\!|\boldsymbol{H}_e(\boldsymbol{b})|\!\|_{k,dG}^2 = |\boldsymbol{H}_e(\boldsymbol{b})|_{k,\nabla}^2 + \sum_{l\in\mathcal{I}_{\mathcal{F}}} \frac{\delta}{\tilde{h}_{kl}} \sum_{i=1}^{M^{(lk)}} |b_i^{(k,l)} - \tilde{b}_i^{(k,l)}|^2 \tilde{h}_l.$$

By means of a similar estimate for piecewise bilinear functions as in Proposition 4.7, we obtain the estimates

$$\begin{aligned}
|\boldsymbol{H}_e(\boldsymbol{b})|_{k,\nabla}^2 = |\mathcal{H}_{\mathcal{Q}_{1,e}}(\chi_{\Gamma,e}(\boldsymbol{b}))|_{k,\nabla}^2 &\leq C|\mathcal{H}_{\mathcal{Q}_{1,e}}(\chi_{\Gamma,e}(\boldsymbol{b}))|_{H^1(\hat{\Omega})}^2,\\
\sum_{i=1}^{M^{(lk)}} |b_i^{(k,l)} - \tilde{b}_i^{(k,l)}|^2 \tilde{h}_l^2 &= \|\chi_{\hat{F}^{(lk)}}(\boldsymbol{b}) - \pi_{\hat{F}^{(lk)}}\chi_{\hat{F}^{(kl)}}(\boldsymbol{b})\|_{lk}^2\\
&\leq C\|\chi_{\hat{F}^{(lk)}}(\boldsymbol{b}) - \pi_{\hat{F}^{(lk)}}\chi_{\hat{F}^{(kl)}}(\boldsymbol{b})\|_{L^2(\hat{F}^{(kl)})}^2\\
&\leq C\|\chi_{\hat{F}^{(lk)}}(\boldsymbol{b}) - \chi_{\hat{F}^{(kl)}}(\boldsymbol{b})\|_{L^2(\hat{F}^{(kl)})}^2,
\end{aligned}$$

and, therefore,

$$\begin{aligned}
\|\!|\boldsymbol{H}_e(\boldsymbol{b})|\!\|_{k,dG}^2 \leq& C\Big(|\mathcal{H}_{\mathcal{Q}_{1,e}}(\chi_{\Gamma,e}(\boldsymbol{b}))|_{H^1(\hat{\Omega})}^2\\
&+ \sum_{l\in\mathcal{I}_{\mathcal{F}}} \frac{\delta}{\tilde{h}_{kl}} \|\chi_{\hat{F}^{(lk)}}(\boldsymbol{b}) - \chi_{\hat{F}^{(kl)}}(\boldsymbol{b})\|_{L^2(\hat{F}^{(kl)})}^2\Big)\\
=& C\|\mathcal{H}_{\mathcal{Q}_{1,e}}(\chi_{\Gamma,e}(\boldsymbol{b}))\|_{k,dG}^2.
\end{aligned}$$

We use the FE equivalent of Lemma 4.1, see Lemma 3.1 in [51], in order to estimate $\|\mathcal{H}_{\mathcal{Q}_{1,e}}(\chi_{\Gamma,e}(\boldsymbol{b}))\|_{k,dG}^2 \leq C\|\mathcal{H}_{\mathcal{Q}_1}(\chi_{\Gamma,e}(\boldsymbol{b}))\|_{k,dG}^2$, where the constant C is independent of h_k, H_k and δ, and $\mathcal{H}_{\mathcal{Q}_1}$ is the standard discrete harmonic extension in the sense of $a^{(k)}(\cdot,\cdot)$. Hence, we obtain

$$\begin{aligned}
\|\mathcal{H}_{\mathcal{Q}_{1,e}}(\chi_{\Gamma,e}(\boldsymbol{b}))\|_{k,dG}^2 \leq& C\Big(|\mathcal{H}_{\mathcal{Q}_1}\Big(\chi_{\Gamma,e}(\boldsymbol{b})\Big)|_{H^1(\hat{\Omega})}^2\\
&+ \sum_{l\in\mathcal{I}_{\mathcal{F}}} \frac{\delta}{\tilde{h}_{kl}} \|\chi_{\hat{F}^{(lk)}}(\boldsymbol{b}) - \chi_{\hat{F}^{(kl)}}(\boldsymbol{b})\|_{L^2(\hat{F}^{(kl)})}^2\Big).
\end{aligned} \tag{4.48}$$

For the second term on the right-hand side of inequality (4.48), we use the estimate

$$\begin{aligned}\|\chi_{\hat{F}^{(lk)}}(\boldsymbol{b}) - \chi_{\hat{F}^{(kl)}}(\boldsymbol{b})\|^2_{L^2(\hat{F}^{(kl)})} \leq & \|\chi_{\hat{F}^{(lk)}}(\boldsymbol{b}) - \pi_{\hat{F}^{(lk)}}\chi_{\hat{F}^{(kl)}}(\boldsymbol{b})\|^2_{L^2(\hat{F}^{(kl)})} \\ & + \|\pi_{\hat{F}^{(lk)}}\chi_{\hat{F}^{(kl)}}(\boldsymbol{b}) - \chi_{\hat{F}^{(kl)}}(\boldsymbol{b})\|^2_{L^2(\hat{F}^{(kl)})}, \end{aligned} \tag{4.49}$$

where the first term can be estimated by the FE equivalent of Proposition 4.7 with

$$\begin{aligned}\|\chi_{\hat{F}^{(lk)}}(\boldsymbol{b}) - \pi_{\hat{F}^{(lk)}}\chi_{\hat{F}^{(kl)}}(\boldsymbol{b})\|^2_{L^2(\hat{F}^{(kl)})} &\leq C\|\chi_{\hat{F}^{(lk)}}(\boldsymbol{b}) - \pi_{\hat{F}^{(lk)}}\chi_{\hat{F}^{(kl)}}(\boldsymbol{b})\|^2_{lk} \\ &= C\sum_{i=1}^{M^{(lk)}} |b_i^{(k,l)} - \tilde{b}_i^{(k,l)}|^2 \tilde{h}_l. \end{aligned} \tag{4.50}$$

Since $\chi_{\hat{F}^{(kl)}}(\boldsymbol{b})$ is a piecewise linear function, we can use already existent estimates for the L^2-projection. We use the following estimate to bound the second term of (4.49)

$$\|\pi_{\hat{F}^{(lk)}}\chi_{\hat{F}^{(kl)}}(\boldsymbol{b}) - \chi_{\hat{F}^{(kl)}}(\boldsymbol{b})\|^2_{L^2(\hat{F}^{(kl)})} \leq C\tilde{h}_l \frac{\tilde{h}_l}{\tilde{h}_k} |\mathcal{H}_{\mathcal{Q}_1}\left(\chi_{\Gamma,e}(\boldsymbol{b})\right)|^2_{H^1(\hat{\Omega})}, \tag{4.51}$$

which follows by repeating the same arguments as in the proof of Lemma 4.9 for bilinear functions. Combining inequalities (4.50) and (4.51) with (4.49) and using it in (4.48) gives

$$\begin{aligned}\|\mathcal{H}_{\mathcal{Q}_1}(\chi_{\Gamma,e}(\boldsymbol{b}))\|^2_{k,dG} \leq C\Bigg(& \delta\tilde{q}_h |\mathcal{H}_{\mathcal{Q}_1}\left(\chi_{\Gamma,e}(\boldsymbol{b})\right)|^2_{H^1(\hat{\Omega})} \\ & + \sum_{l\in\mathcal{I}_\mathcal{F}} \frac{\delta}{\tilde{h}_{kl}} \sum_{i=1}^{M^{(lk)}} |b_i^{(k,l)} - \tilde{b}_i^{(k,l)}|^2 \tilde{h}_l \Bigg). \end{aligned} \tag{4.52}$$

We are now in the position to use the available theory for the standard discrete harmonic extension $\mathcal{H}_{\mathcal{Q}_1}$ to estimate the first term of (4.52). Recalling the estimate

$$|\mathcal{H}_{\mathcal{Q}_1}\left(\chi_{\Gamma,e}(\boldsymbol{b})\right)|^2_{H^1(\hat{\Omega})} \leq C(1+\log^2 \tilde{h}_k^{-1}) \sum_{l\in\mathcal{I}_\mathcal{F}} |\chi_{\hat{F}^{(kl)}}(\boldsymbol{b})|^2_{H^{1/2}(\hat{F}^{(kl)})},$$

see Theorem. 5 in [158] or the proof of Theorem 5.1. in [19], and since $|\chi_{\hat{F}^{(kl)}}(\boldsymbol{b})|^2_{H^{1/2}(\hat{F}^{(kl)})} = \|\boldsymbol{b}|_{\hat{F}^{(kl)}}\|^2_{\hat{F}^{(kl)}}$, we obtain

$$\begin{aligned}\|\boldsymbol{H}_e(\boldsymbol{b})\|^2_{k,dG} \leq & C\delta\tilde{q}_h^{(k)}(1+\log^2 \tilde{h}_k^{-1}) \\ & \sum_{l\in\mathcal{I}_\mathcal{F}} \left(\|\boldsymbol{b}|_{\hat{F}^{(kl)}}\|^2_{\hat{F}^{(kl)}} + \frac{\delta}{\tilde{h}_{kl}} \sum_{i=1}^{M^{(lk)}} |b_i^{(k,l)} - \tilde{b}_i^{(k,l)}|^2 \tilde{h}_l \right). \end{aligned}$$

This proves the first inequality. Again, by means of a similar estimate for piecewise bilinear functions as in Proposition 4.7 and according to Theorem 5.1(b) in [19], we have

$$|\boldsymbol{C}^{(k,k)}|^2_{k,\nabla} \geq C|\chi(\boldsymbol{C}^{(k,k)})|^2_{H^1(\hat{\Omega})} \geq C\|\boldsymbol{C}|_{\hat{F}^{(kl)}}\|^2_{\hat{F}^{(kl)}}.$$

Therefore, we finally arrive at the estimate

$$\begin{aligned}\|\boldsymbol{C}\|^2_{k,dG} &= \|\boldsymbol{C}^{(k,k)}\|^2_{k,\nabla} + \sum_{l\in\mathcal{I}_{\mathcal{F}}} \frac{\delta}{\tilde{h}_{kl}} \sum_{i=1}^{M^{(lk)}} |\boldsymbol{u}_i^{(k,l)} - \tilde{\boldsymbol{u}}_i^{(k,l)}|^2 \tilde{h}_l \\ &\geq C\bigg(\|\boldsymbol{C}|_{\hat{F}^{(kl)}}\|^2_{\hat{F}^{(kl)}} + \sum_{l\in\mathcal{I}_{\mathcal{F}}} \frac{\delta}{\tilde{h}_{kl}} \sum_{i=1}^{M^{(lk)}} |\boldsymbol{u}_i^{(k,l)} - \tilde{\boldsymbol{u}}_i^{(k,l)}|^2 \tilde{h}_l \bigg),\end{aligned}$$

which concludes the proof. □

4.2.2 Condition Number Bound

The goal of this section is to establish the condition number bound for $M^{-1}_{BDDC}\widehat{S}$. Following [19], we assume that the mesh is quasi-uniform on each subdomain and the diffusion coefficient is globally constant. Moreover, in [19], one can also find a formal definition of the BDDC preconditioner M^{-1}_{BDDC}. It was already pointed out that the spectrum of $M^{-1}_{BDDC}\widehat{S}$ is equal to $M^{-1}_{sD}F$ up to zeros and ones. For simplicity, we focus on a patch $\Omega^{(k)}$, with $k \in \{1,\dots,N\}$, which does not touch the boundary $\partial\Omega$.

Let $u^{(k)} \in V^{(k)}_{h,e}$, then $u^{(k)}$ is determined by its coefficients $(\boldsymbol{u}_i^{(k)}), i \in \mathcal{I}$, which can be interpreted as a matrix $\boldsymbol{C}_e^{(k)} = (\boldsymbol{C}^{(k,k)}, (\boldsymbol{c}^{(k,l)})_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}}) \in \mathcal{R}_e^{(k)}$. In a similar way, we can identify functions on the trace space $W^{(k)}$. For completeness we rephrase Theorem 3.19 according to the current notation, cf. Theorem 6.1 in [18], which provides an abstract estimate of the condition number using the multiplicity scaling.

Theorem 4.15. *Let $\delta^{\dagger(k)}$ be chosen accordingly to the multiplicity scaling strategy. Assume that there exist two positive constants c_*, c^* and a boundary semi-norm $|\cdot|_{W^{(k)}}$ on $W^{(k)}$, $k = 1,\dots,N$, such that*

$$|w^{(k)}|^2_{W^{(k)}} \leq c^* s_e^{(k)}(w^{(k)}, w^{(k)}) \quad \forall w^{(k)} \in W^{(k)}, \tag{4.53}$$

$$|w^{(k)}|^2_{W^{(k)}} \geq c_* s_e^{(k)}(w^{(k)}, w^{(k)}) \quad \forall w^{(k)} \in W^{(k)}_{\Delta}, \tag{4.54}$$

$$|w^{(k)}|^2_{W^{(k)}} = \sum_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}} |w^{(k)}|_{F_e^{(kl)}}|_{W^{(kl)}} \quad \forall w^{(k)} \in W^{(k)}, \tag{4.55}$$

where $|\cdot|_{W^{(kl)}}$ is a semi-norm associated to the edge spaces $W^{(k)}|_{F_e^{(kl)}}$ with $l \in \mathcal{I}_{\mathcal{F}}^{(k)}$. Then the condition number of the preconditioned BDDC operator $M^{-1}_{BDDC}\widehat{S}$ satisfies the bound

$$\kappa(M^{-1}_{BDDC}\widehat{S}) \leq C(1 + c_*^{-1}c^*),$$

where the constant C is independent of h and H.

Using this abstract framework, we obtain the following condition number estimate for the BDDC preconditioner.

Theorem 4.16. *Let $\Omega \subset \mathbb{R}^2$, and let $\widetilde{W}$ be given as set of all functions from W, that are continuous at the corners $\nu \in \mathcal{V}^{(k)}$ of $\Omega^{(k)}, k = 1, \dots N$, cf., Figure 4.2 and Figure 4.3. Moreover, we assume that the diffusion coefficient α is globally constant. Then there exists a boundary semi-norm such that the constants c_* and c^* of Theorem 4.15 are bounded by*

$$c^* \leq C_1 \quad \text{and} \quad c_*^{-1} \leq \delta^2 C_2 \max_{\substack{1\leq k\leq N \\ l\in\mathcal{I}_{\mathcal{F}}^{(k)}}} \left(\frac{h_l}{h_k} + \frac{h_l^2}{h_k^2}\right)^2 \max_{1\leq k\leq N} \left(1 + \log^2\left(\frac{H_k}{h_k}\right)\right),$$

where the C_1 and C_2 positive constants that are independent of H and h . Therefore, the spectral condition number of the isogeometric preconditioned BDDC operator is bounded by

$$\kappa(M_{BDDC}^{-1}\widehat{S}) \leq C\delta^2 \max_{\substack{1\leq k\leq N \\ l\in\mathcal{I}_{\mathcal{F}}^{(k)}}} \left(\frac{h_l}{h_k} + \frac{h_l^2}{h_k^2}\right)^2 \max_{1\leq k\leq N} \left(1 + \log^2\left(\frac{H_k}{h_k}\right)\right),$$

where the constant C is independent of H, h and δ.

Proof. The first step is to appropriately define the semi-norm $|\cdot|^2_{W^{(k)}}$ in $W^{(k)}$:

$$|w^{(k)}|^2_{W^{(k)}} := \sum_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}} |w^{(k)}|_{F_e^{(kl)}}|^2_{W^{(kl)}},$$

$$|w^{(k)}|_{F_e^{(kl)}}|^2_{W^{(kl)}} := \|\!|w^{(k)}|_{F^{(kl)}}|\!\|^2_{F^{(kl)}} + |w^{(k)}|_{F^{(kl)}}|^2_{k,\nabla} + \frac{\delta}{\tilde{h}_{kl}}\|w^{(k,l)} - \pi_{F^{(lk)}} w^{(k)}_{|F^{(kl)}}\|^2_{lk},$$

where $|w^{(k)}|_{F^{(kl)}}|^2_{k,\nabla}$ has to be understood as the restriction of the discrete semi-norm to $F^{(kl)}$, cf., Definition 4.6. This essentially gives the differences along $F^{(kl)}$. Furthermore, we define $\|\!|w^{(k)}|_{F^{(kl)}}|\!\|_{F^{(kl)}} := \|\!|\boldsymbol{w}|\!\|_{F^{(kl)}}$, where $\boldsymbol{w}$ are the values $(\boldsymbol{w}_i)_{i\in\mathcal{I}(F^{(kl)})}$ written as a vector.

Given $w^{(k)} \in W^{(k)}$, we define its IgA harmonic extension by $u^{(k)} = \mathcal{H}_e^{(k)}(w^{(k)})$ with coefficients $\boldsymbol{C}_e^{(k)} := (\boldsymbol{C}^{(k,k)}, (\boldsymbol{c}^{(k,l)})_{l\in\mathcal{I}_{\mathcal{F}}^{(k)}})$. Now we consider a single edge $F^{(kl)}$, since $\|\!|w^{(k)}|_{F^{(kl)}}|\!\|^2_{F^{(kl)}} = \|\!|\boldsymbol{C}|^{(k,k)}_{\hat{F}^{(kl)}}|\!\|^2_{\hat{F}^{(kl)}}$ and $\|w^{(k,l)} - \pi_{F^{(lk)}} w^{(k)}_{|F^{(kl)}}\|^2_{lk} = \sum_{i\in\mathcal{I}^{(k,l)}} |\boldsymbol{w}_i^{(k,l)} - \tilde{\boldsymbol{w}}_i^{(k,l)}|^2 \hat{h}_l$, we can estimate

$$\|\!|w^{(k)}|_{F^{(kl)}}|\!\|^2_{F^{(kl)}} + \frac{\delta}{\tilde{h}_{kl}}\|w^{(k,l)} - \pi_{F^{(lk)}} w^{(k)}_{|F^{(kl)}}\|^2_{lk} \leq C\|\!|\boldsymbol{C}_e^{(k)}|\!\|^2_{k,dG},$$

by means of Theorem 4.14(b) and (4.45). Moreover, the second term of $|\cdot|^2_{W^{(kl)}}$ is a part of $\|\!|\cdot|\!\|^2_{k,dG}$. Hence, we obtain the inequality

$$|w^{(k)}|_{F^{(kl)}}|^2_{W^{(kl)}} \leq C\|\!|\boldsymbol{C}_e^{(k)}|\!\|^2_{k,dG}.$$

By means of Proposition 4.11, it follows that

$$|w^{(k)}|_{F^{(kl)}}|^2_{W^{(kl)}} \leq C\|\boldsymbol{C}_e^{(k)}\|^2_{k,dG} \leq C|\hat{u}^{(k)}|^2_{k,dG} \leq C\|u^{(k)}\|^2_{k,dG}.$$

Using Lemma 4.1 and Corollary 4.2, we can estimate

$$\begin{aligned}\|u^{(k)}\|^2_{k,dG} =&\|\mathcal{H}_e^{(k)}(w^{(k)})\|^2_{k,dG} \leq C\|\mathcal{H}^{(k)}(w^{(k)})\|^2_{k,dG}\\ &\leq Ca_e^{(k)}(\mathcal{H}_e^{(k)}(w^{(k)}),\mathcal{H}_e^{(k)}(w^{(k)})) = Cs_e^{(k)}(w^{(k)},w^{(k)}),\end{aligned}$$

and we arrive at $|w^{(k)}|_{F^{(kl)}}|^2_{W^{(kl)}} \leq Cs_e^{(k)}(w^{(k)},w^{(k)})$. Since this estimate holds for the four edges of the patch, we obtain

$$|w^{(k)}|^2_{W^{(k)}} \leq Cs_e^{(k)}(w^{(k)},w^{(k)}) \quad \forall w \in W^{(k)},$$

where the constant C is independent of h_k and H_k, which proves the upper bound.

For the lower bound, let $w^{(k)} \in W_\Delta^{(k)}$. We apply the lifting operator $\boldsymbol{H}_e^{(k)}$ to its coefficient representation $\boldsymbol{w}^{(k)} \in \mathbb{R}^{M_\Gamma^{(k)}}$, and obtain a matrix $\boldsymbol{H}_e^{(k)}(w^{(k)})$ with entries $(\boldsymbol{u}_i{}^{(k)})_{i\in\mathcal{I}^{(k)}}$. According to (4.2), these entries define a spline function $u^{(k)} := (u^{(k,k)},(u^{(k,l)})_{l\in\mathcal{I}_\mathcal{F}^{(k)}})$. We observe the estimate

$$\begin{aligned}\delta q_h^{(k)}\|\boldsymbol{H}_e^{(k)}(w^{(k)})\|^2_{k,dG} &\geq C\delta q_h^{(k)}|\hat{u}^{(k)}|^2_{k,dG} \geq C\|u^{(k)}\|^2_{k,dG} \geq Ca_e^{(k)}(u^{(k)},u^{(k)})\\ &\geq Ca_e^{(k)}(\mathcal{H}_e^{(k)}(w^{(k)}),\mathcal{H}_e^{(k)}(w^{(k)})) = Cs_e^{(k)}(w^{(k)},w^{(k)}),\end{aligned} \tag{4.56}$$

where we used inequality (4.37), (4.47), Lemma 2.18 and the fact that $\mathcal{H}_e^{(k)}(w^{(k)})$ minimizes the energy among given boundary data $w^{(k)}$. By means of Theorem 4.14(a), we can estimate

$$\begin{aligned}\|\boldsymbol{H}_e^{(k)}(w^{(k)})\|^2_{k,dG} \leq& C\delta\tilde{q}_h^{(k)}(1+\log^2\tilde{h}_k^{-1})\\ &\sum_{l\in\mathcal{I}_\mathcal{F}^{(k)}}\left(\|w^{(k)}|_{F^{(kl)}}\|^2_{F^{(kl)}} + \frac{\delta}{\tilde{h}_{kl}}\sum_{i=1}^{M^{(lk)}}|\boldsymbol{u}_i^{(k,l)}-\tilde{\boldsymbol{u}}_i^{(k,l)}|^2\tilde{h}_l\right)\\ \leq& C\tilde{q}_h^{(k)}(1+\log^2\tilde{h}_k^{-1})|w^{(k)}|^2_{W^{(k)}}.\end{aligned} \tag{4.57}$$

Combining (4.56) and (4.57) gives

$$s_e^{(k)}(w^{(k)},w^{(k)}) \leq C\delta^2 q_h^{(k)}\tilde{q}_h^{(k)}(1+\log^2\tilde{h}_k^{-1})|w^{(k)}|^2_{W^{(k)}}.$$

Due to (4.45), we have $\tilde{h}_k \approx h_k/H_k$, and since $H_k \approx H_l$ we obtain $\tilde{q}_h \approx q_h$. Taking the maximum over all patches proves the upper bound. By applying Theorem 4.15, the condition number bound follows.

□

Remark 4.17. *The dependence of the condition number on h_l/h_k and δ originates from* (4.37)*, when estimating the L^2-norm of the jumps on $F^{(kl)}$ by the discrete norm $\|\cdot\|_{lk}$ using the L^2-orthogonal projection. To be more precise, when using Lemma 4.9 in order to estimate the $\|\pi_{F^{(lk)}} u^{(k,k)}_{|F^{(kl)}} - u^{(k,k)}_{|F^{(kl)}}\|^2_{L^2(F^{(kl)})}$ by the H^1-semi-norm the ratio h_k/h_l and the parameter δ are added to the semi-norm. Moreover, in the proof of Theorem 4.14, we obtain again the same dependence by repeating the proof of Proposition 4.11 for piecewise linear functions. By combining the estimates in the proof of Theorem 4.16, the two parameters enter the condition number bound.*

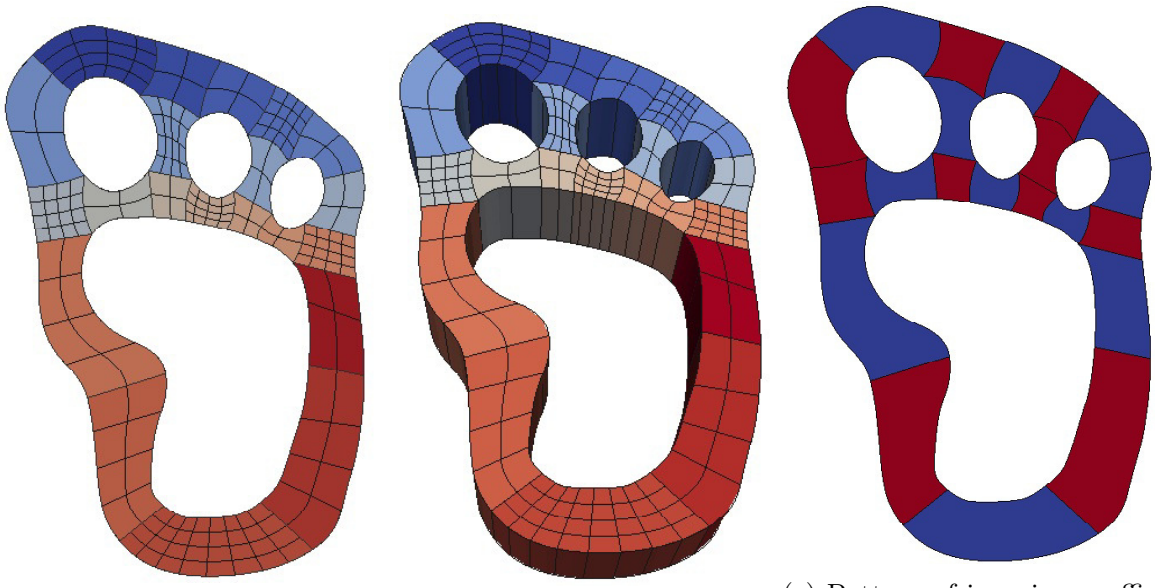
Remark 4.18. *The extension to the three-dimensional case and other primal variables is certainly possible, but even more technical. The definitions and notations introduced in Section 4.2.1 can be extended to three dimensions in a straightforward way. Theorem 4.14 is specific to the two-dimensional case with primal vertex dofs and has to be extended in order to handle continuous averages and/or higher dimension.*

Theorem 4.16 provides the theoretical basis for the numerical results obtained in Section 4.3 for the two-dimensional case with only vertex primal variables. The numerical results indicate that this bound also holds for continuous edge averages as primal variables and for three-dimensional problems with additional interface and/or edge averages. Although the presented proof does not cover the case of jumping diffusion coefficients, we also observed robustness of the condition number in such cases in Section 4.3. Note that the condition number bound obtained in Theorem 4.16 depends on the ratio h_l/h_k. However, numerical results do not reproduce this behaviour, cf., Section 4.3.3. In [22], the more complex deluxe scaling is considered, for which it may be possible to prove a bound, which is independent of h_l/h_k. We point out that the presented analysis does not answer the dependence on the B-Spline degree p. The numerical experiments in Section 4.3 indicate that the condition number depends on the degree, but very moderately in a logarithmic way for two-dimensional domains. For three-dimensional domains we observe a linear dependence. Similar results have been obtained in [18] for the cG BDDC-IgA preconditioner, see also Section 3.4.4 for the cG-IETI-DP method. Moreover, the condition number bound proven here is not independent of the parameter δ in the dG penalty term in contrast to the bound given in [52] for the FE equivalent. In the numerical experiments δ is chosen to be $(p+1)(p+d)$ according to the inverse inequalities used, where d is the dimension. Hence, δ is independent of the mesh-size h and the influence of δ on the algorithm is implicitly contained in the experiments about the p-dependence.

4.3 Numerical Examples

In this section, we present numerical results documenting the numerical behaviour of the implemented dG-IETI-DP algorithm for solving large-scale linear systems arising from higher-order IgA discretizations of (2.1) in the domains illustrated in Figure 4.4(a)

and Figure 4.4(b). The computational domain consists of 21 subdomains in both 2d and 3d. In both cases, one side of a patch boundary has inhomogeneous Dirichlet conditions, whereas all other sides have homogeneous Neumann conditions. We consider the case of non-matching meshes, i.e., two neighbouring patches may have different mesh-sizes h_k and h_l. Due to our implementation of the dG formulation, we only consider nested meshes on the interface, i.e., the B-Spline spaces on the interfaces are nested. However, we note that the presented algorithm does not rely on this assumption. Each subdomain has a diameter of H_k and an associated mesh-size of h_k. In the following, we use the abbreviation $H/h = \max_k H_k/h_k$. We consider B-Splines, where its degree is chosen as $p = 2$ and $p = 4$. In all numerical examples, when increasing the degree from 2 to 4, we keep the smoothness of the space, i.e., we increase the multiplicity of the knots on the coarsest mesh. In order to solve the linear system (4.27), a PCG algorithm with the scaled Dirichlet preconditioner (4.28) is performed. We use a zero initial guess, and a reduction of the initial residual by a factor of 10^{-6} as stopping criterion. The numerical examples illustrate the dependence of the condition number of the IETI-DP preconditioned system on jumps in the diffusion coefficient α, patch size H, mesh-size h and the degree p. In Section 4.3.3, we investigate the special case of increasing h_k/h_l and its influence on the condition number. In all other tests we consider a fixed ratio h_k/h_l. We use the C++ library G+Smo for describing the geometry and performing the numerical tests, see also [111] and [162].



(a) 2d YETI-footprint

(b) 3d YETI-footprint

(c) Pattern of jumping coefficients

Figure 4.4: Figure (a) and (b) show the computational domain, the decomposition into patches and its initial mesh in 2d and 3d, respectively. Figure (c) presents the pattern of the jumping diffusion coefficient.

		ALG. A				ALG. C			
$p = 2$		coeff. scal.		stiff. scal.		coeff. scal.		stiff. scal.	
#dofs	H/h	κ	It.	κ	It.	κ	It.	κ	It.
1610	8	3.0	17	3.1	17	1.3	9	1.3	9
4706	16	4.0	19	4.2	19	1.6	11	1.6	11
15602	32	5.2	21	5.4	21	1.9	12	1.9	12
56210	64	6.5	23	6.8	23	2.4	14	2.4	14
212690	128	8.0	24	8.4	24	2.8	16	2.8	16
$p = 4$		coeff. scal.		stiff. scal.		coeff. scal.		stiff. scal.	
#dofs	H/h	κ	It.	κ	It.	κ	It.	κ	It.
4706	8	1.3	9	1.3	9	1.7	11	1.7	12
9370	16	1.6	11	1.6	11	2.1	13	2.1	13
23402	32	1.9	12	1.9	12	2.5	15	2.5	15
70282	64	2.4	14	2.4	14	3.0	16	3.0	16
239306	128	2.8	16	2.8	16	3.5	17	3.5	18

Table 4.1: 2d example with $p = 2$ and $p = 4$, and homogeneous diffusion coefficient. Dependence of the condition number κ and the number It. of iterations on H/h for the preconditioned system with coefficient and stiffness scaling. Choice of primal variables: vertex evaluation, vertex evaluation and edge averages.

4.3.1 Homogeneous Diffusion Coefficient

We first consider the case of a homogeneous diffusion coefficient, i.e., $\alpha = 1$ on Ω. The 2d results are summarized in Table 4.1, whereas the 3d results are presented in Table 4.2. We observe that the condition number of the preconditioned system grows logarithmically with respect to H/h. Moreover, the numerical results indicate a dependence of the condition number on the degree p, which will be investigated in more detail in Section 4.3.5.

4.3.2 Inhomogeneous Diffusion Coefficient

In this subsection, we investigate the case of patch-wise constant diffusion coefficient, but with jumps across the patch interfaces. The diffusion coefficient takes values $\alpha^{(k)} \in \{\text{blue}, \text{red}\} := \{10^{-4}, 10^{4}\}$, with a jumping pattern according to Figure 4.4 (c). The 2d results are summarized in Table 4.3, and the 3d results are presented in Table 4.4. Comparing Table 4.1 and Table 4.2 with Table 4.3 and Table 4.4, one clearly sees the robustness with respect to jumping coefficients of the considered method and the quasi optimal dependence of the condition number on H/h. The dependence of the degree will again be studied in Section 4.3.5.

		ALG. A				ALG. C			
$p=2$		coeff. scal.		stiff. scal.		coeff. scal.		stiff. scal.	
#dofs	H/h	κ	It.	κ	It.	κ	It.	κ	It.
2800	3	49.9	44	49.9	44	1.4	9	1.4	9
9478	6	72.3	48	70.3	49	15.8	17	14.7	17
42922	12	169.1	70	165.6	69	19.8	30	18.4	29
244594	25	376.4	91	368.7	92	24.0	37	22.4	36
$p=4$		coeff. scal.		stiff. scal.		coeff. scal.		stiff. scal.	
#dofs	H/h	κ	It.	κ	It.	κ	It.	κ	It.
22204	8	203.1	77	199.0	79	23.3	33	21.5	34
42922	16	248.9	82	240.7	83	26.7	34	24.7	34
116110	32	506.3	104	488.1	104	31.3	42	29.1	41
443926	128	1090.1	130	1060.7	129	36.4	44	34.1	45

Table 4.2: 3d example with $p = 2$ and $p = 4$, and homogeneous diffusion coefficient. Dependence of the condition number κ and the number It. of iterations on H/h for the preconditioned system with coefficient and stiffness scaling. Choice of primal variables: vertex evaluation, vertex evaluation and edge averages.

		ALG. A				ALG. C			
$p=2$		coeff. scal.		stiff. scal.		coeff. scal.		stiff. scal.	
#dofs	H/h	κ	It.	κ	It.	κ	It.	κ	It.
1610	8	3.8	12	4.0	12	1.4	7	1.4	7
4706	16	5.1	13	5.4	13	1.7	7	1.7	7
15602	32	6.5	15	7.1	15	2.0	8	2.1	8
56210	64	8.2	15	9.0	16	2.4	8	2.6	8
212690	128	10.1	17	11.1	18	2.9	9	3.1	9
$p=4$		coeff. scal.		stiff. scal.		coeff. scal.		stiff. scal.	
#dofs	H/h	κ	It.	κ	It.	κ	It.	κ	It.
4706	8	5.7	14	6.1	14	1.8	8	1.9	8
9370	16	7.0	14	7.7	15	2.1	8	2.3	8
23402	32	8.7	15	9.6	17	2.5	9	2.8	9
70282	64	10.7	18	11.8	18	3.0	9	3.3	9
239306	128	12.8	18	14.2	18	3.5	10	3.9	10

Table 4.3: 2d example with $p = 2$ and $p = 4$, and jumping diffusion coefficient. Dependence of the condition number κ and the number It. of iterations on H/h for the preconditioned system with coefficient and stiffness scaling. Choice of primal variables: vertex evaluation, vertex evaluation and edge averages.

		ALG. A				ALG. C			
$p = 2$		coeff. scal.		stiff. scal.		coeff. scal.		stiff. scal.	
#dofs	H/h	κ	It.	κ	It.	κ	It.	κ	It.
2800	8	50.3	28	57.9	25	2.1	11	2.1	11
9478	16	72.2	29	83.4	29	12.6	17	14.6	18
42922	32	176.1	43	203.7	42	15.7	22	18.2	24
244594	128	400.6	52	463.0	58	18.9	28	22.0	30
$p = 4$		coeff. scal.		stiff. scal.		coeff. scal.		stiff. scal.	
#dofs	H/h	κ	It.	κ	It.	κ	It.	κ	It.
22204	8	203.3	44	236.1	45	17.7	15	20.7	15
42922	16	250.2	43	290.4	42	20.5	17	23.9	17
116110	32	520.9	58	605.7	57	24.0	19	28.0	21
443926	128	1143.1	72	1331.9	80	28.0	22	32.6	22

Table 4.4: 3d example with $p = 2$ and $p = 4$, and jumping diffusion coefficient. Dependence of the condition number κ and the number It. of iterations on H/h for the preconditioned system with coefficient and stiffness scaling. Choice of primal variables: vertex evaluation, vertex evaluation and edge averages and face averages.

4.3.3 Dependence on h_k/h_l

In this subsection, we deal with dependence of the condition number on the ratio $q := h_k/h_l$ of mesh-sizes corresponding to neighbouring patches. The initial domain is the same as given in Figure 4.4, but without the additional refinements in certain patches, i.e., $h_k/h_l = 1$. Then we consequently perform uniform refinement in the considered patches and obtain $h_k/h_l = 2^r$, where r is the number of refinements. In the numerical tests, we only consider the cases $\alpha \in \{10^{-4}, 10^4\}$ and $p = 4$. The results for 2d and 3d are summarized in Table 4.5 and indicate that the condition number is independent of the ratio h_k/h_l for 2d and 3d, as also predicted for FE in [52]. We note that the increasing condition number and number of iterations come along with the increased ratio H/h and comparing the numbers of this test with the corresponding ones from Table 4.4 we observe an agreement. Thus, it is noteworthy that, although in 2d the ratio H/h is increasing, the condition number stays constant.

4.3.4 Weak Scaling

Similar to Section 3.4.3 we investigate the weak scaling behaviour of the dG-IETI-DP method. Here we fix the ratio H/h and increase the number of patches. In the following tests, we perform a uniform splitting of the patches, i.e., by splitting them into 2^d subpatches. We choose the two- and three-dimensional YETI footprint as computational domain, see Figure 4.4. One each patch we perform one initial refinement, fix

			ALG. A				ALG. C			
dim = 2			coeff. scal.		stiff. scal.		coeff. scal.		stiff. scal.	
#dofs	q	H/h	κ	It.	κ	It.	κ	It.	κ	It.
1816	1	2	4.9	13	5.2	14	1.6	7	1.7	7
2134	2	4	4.9	13	5.3	14	1.6	7	1.7	7
2962	4	8	4.9	13	5.5	14	1.6	7	1.8	7
5386	8	16	4.9	13	5.6	14	1.6	7	1.8	7
13306	16	32	4.9	13	5.7	14	1.6	7	1.8	7
41434	32	64	4.9	13	5.6	14	1.6	7	1.8	7
			ALG. C				ALG. B			
dim = 3			coeff. scal.		stiff. scal.		coeff. scal.		stiff. scal.	
#dofs	q	H/h	κ	It.	κ	It.	κ	It.	κ	It.
9362	1	2	14.8	21	14.9	21	14.8	15	16.6	15
11902	2	4	17.7	23	22.1	24	19.5	15	28.7	17
20426	4	8	29.2	24	37.5	27	29.2	16	37.5	18
56626	8	16	52.2	26	67.2	27	52.2	17	67.2	18
345268	16	32	98.4	27	124.4	28	98.4	17	124.8	19
1758004	32	64	191.1	28	240.6	28	308.1	20	240.6	20

Table 4.5: 2d and 3d example with $p = 4$, and jumping diffusion coefficient. Dependence of the condition number κ and the number It. of iterations on the ratio $q = h_k/h_l$ for the preconditioned system with coefficient and stiffness scaling. Choice of primal variables: in 2d vertex evaluation, vertex evaluation and edge averages, in 3d vertex evaluation and edge averages, vertex evaluation, edge averages and face averages.

2D		ALG. A	coeff. scal		ALG. C	coeff. scal	
N	Dofs	n_Π	κ	It.	n_Π	κ	It.
21	10524	82	6.7	23	106	2.3	14
84	43688	332	7.7	30	464	2.8	16
336	155920	1336	10.0	33	1936	2.8	17
1344	629024	5360	11.3	35	7904	3.1	18
5376	2526784	21472	12.6	37	31936	3.5	19
3D		ALG. A	coeff. scal		ALG. C*	coeff. scal	
N	Dofs	n_Π	κ	It.	n_Π	κ	It.
21	3990	160	75.3	51	176	26.1	48
168	40480	1316	395.4	175	1856	65.3	80
1344	337720	10632	-	$\geq$200	15776	74.1	95
10752	2829200	85640	-	$\geq$200	128006	85.4	102

Table 4.6: Weak scaling results for two- and three-dimensional computational domain having B-Spline of degree $p = 4$ and jumping diffusion coefficient $\alpha \in \{10^{-3}, 10^3\}$.

the polynomial degree to 4 and used a homogeneous diffusion coefficient. For the initial three-dimensional geometry, we perform one additional refinement in z-direction. Moreover, we only report the condition number and CG iterations for the coefficient scaling, the stiffness scaling gives very similar results. We expect constant condition numbers and number of iterations. The results are summarized in Table 4.6. For the two-dimensional domain, we only observe a slight increase of the condition number, in particular in case of Algorithm C. The results also indicate a similar behaviour for the three-dimensional domain, where Algorithm A does not provide a scalable method (as expected). For the second test, we use a variant of the Algorithm C, denoted by C^*, having only edge averages. This reduces the number of primal variables, circumventing memory limitations. Also in this case, we observe an increasing condition number, which seems to slowly saturate.

4.3.5 Dependence on the Degree p

In this subsection, we study the dependence of the condition number on the degree p of the B-Spline space. There are two ways to dealing with degree elevation. One method is to keep the smoothness of the space, i.e., the multiplicity of the knots is increased in each step. The other way keeps the multiplicity of the knots, while increasing the smoothness of the B-Spline. The first method retains the support of the B-Spline basis small, with the drawback of a larger number of dofs, while the second method does it vice versa, i.e., increasing the support of the B-Spline, while having a smaller number of dofs. The aim of this section is to investigate the effect of the two different elevation techniques on the condition number.

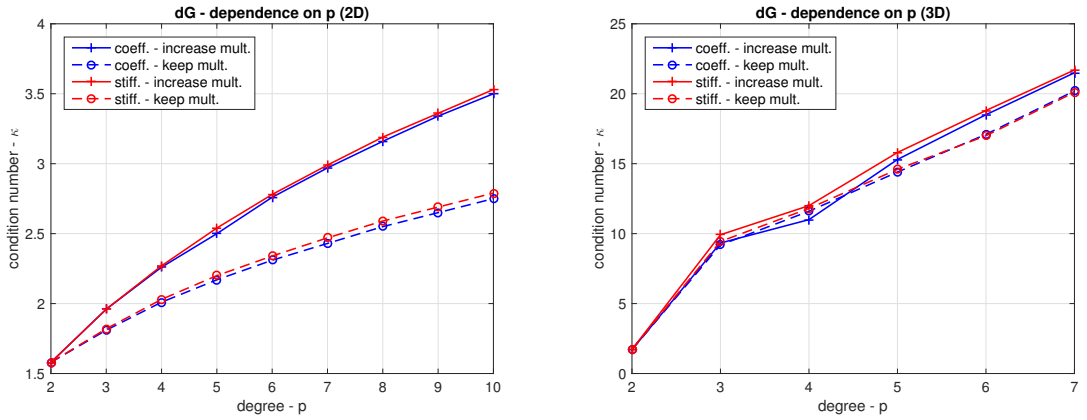
We choose the computational domain as the 2d and 3d YETI-footprint presented in Figure 4.4 and the diffusion coefficient is chosen to be globally constant. The results are summarized in Table 4.7 and in Table 4.8 for the 2d and 3d domain, respectively. The numerical results indicate a at most linear dependence of the condition number of the preconditioned system on the B-Spline degree p. When considering the 2d domain, the dependence on the degree seems to be even logarithmic, see Figure 4.5. One observes a significant increase of the condition number, when increasing the degree from 2 to 3 in 3d as illustrated in Figure 4.5 (b).

Increasing the multiplicity						Increasing the smoothness					
ALG. C		coeff. scal		stiff. scal.		ALG. C		coeff. scal		stiff. scal.	
#dofs	degree	κ	It.	κ	It.	#dofs	degree	κ	It.	κ	It.
2794	2	1.5	11	1.5	10	2794	2	1.5	11	1.5	10
8706	3	1.9	13	1.9	13	3314	3	1.8	12	1.8	12
17818	4	2.2	14	2.2	14	3876	4	2.0	13	2.0	13
30130	5	2.5	15	2.5	15	4480	5	2.1	13	2.2	14
45642	6	2.7	16	2.7	16	5126	6	2.3	14	2.3	14
64354	7	2.9	17	2.9	17	5814	7	2.4	15	2.4	15
86266	8	3.1	17	3.1	18	6544	8	2.5	15	2.5	15
111378	9	3.3	18	3.3	18	7316	9	2.6	16	2.6	16
139690	10	3.5	18	3.5	18	8130	10	2.7	16	2.7	16

Table 4.7: 2d example with fixed initial mesh and homogeneous diffusion coefficient. Dependence of the condition number κ and the number It. of iterations on H/h for the preconditioned system with coefficient and stiffness scaling. Choice of primal variables: vertex evaluation and edge averages.

Increasing the multiplicity						Increasing the smoothness					
ALG. B		coeff. scal		stiff. scal.		ALG. B		coeff. scal		stiff. scal.	
#dofs	degree	κ	It.	κ	It.	#dofs	degree	κ	It.	κ	It.
1474	2	1.7	12	1.7	12	1474	2	1.7	12	1.7	12
5472	3	9.3	29	9.9	26	3004	3	9.1	19	9.4	19
13182	4	11.0	35	12.0	34	5232	4	11.6	32	11.8	28
25804	5	15.3	42	15.8	39	8284	5	14.4	35	14.6	31
44538	6	18.5	46	18.8	45	12286	6	17.1	44	17.0	38
70584	7	21.5	51	21.7	49	17364	7	20.2	47	20.1	40

Table 4.8: 3d example with fixed initial mesh and homogeneous diffusion coefficient. Dependence of the condition number κ and the number It. of iterations on H/h for the preconditioned system with coefficient and stiffness scaling. Choice of primal variables: vertex evaluation, edge averages and face averages.



(a) 2d setting (b) 3d setting

Figure 4.5: Dependence of the condition number on the B-Spline degree p for the 2d and 3d domain. We compare the influence of the considered scaling strategy and method for increasing the degree.

4.3.6 Performance of cG-IETI-DP and dG-IETI-DP Methods

The problems were calculated on a Desktop PC with an Intel(R) Xeon(R) CPU E5-1650 v2 @ 3.50GHz and 16 GB main memory. The LU-factorizations for the local solvers are performed by means of the PARDISO 5.0.0 Solver Project, see [140]. In Table 4.9, we investigate the runtime of a serial implementation and compare it with the timings of the cG-IETI-DP algorithm. In order to compare the timings for the continuous IETI-DP and discontinuous IETI-DP method, we use the setting in Section 3.4. More precisely, the computational domain is the 2d example from Figure 4.4, but with fully matching patches, i.e., no additional refinements on selected patches. For this test we have 114398 total degrees of freedom, 1644 Lagrange multipliers, and on each patch approximate 4600 local degrees of freedom on each patch. We select a run with coefficient scaling and obtain a condition number of $\kappa = 3.53$ and 9 iterations. We observe from Table 4.9 that the dG-IETI-DP method shows a very similar performance. The increased number of primal variables and the extended version of the stiffness matrix $\boldsymbol{K}_e$ leads to a slightly larger runtime.

4.4 Segmentation Crimes

In all other sections, we consider multi-patch geometries, which are geometrically matching, i.e., the interfaces of adjacent patches are identical. In this section, we consider domains having small gaps and overlaps at the patch interfaces. To solve PDEs on such domains, we use a dG approach and construct appropriate numerical fluxes to provide the information transfer across the gaps and overlaps. The case of

	Wall-clock time		relative time in %	
	cG	dG	cG	dG
Preparing the bookkeeping	0.012 s	0.02 s	0.06	0.11
Assembling all patch-local $\boldsymbol{K}^{(k)}$	6.4 s	6.8 s	35.56	35.79
Partitioning w.r.t. B and I	0.085 s	0.12 s	0.48	0.63
Assembling $\boldsymbol{C}$	0.017 s	0.034 s	0.09	0.18
LU-fact. of $\boldsymbol{K}_{II}^{(k)}$	2.5 s	2.5 s	13.89	13.16
LU-fact. of $\begin{bmatrix} \boldsymbol{K}^{(k)} & \boldsymbol{C}^{(k)^T} \\ \boldsymbol{C}^{(k)} & 0 \end{bmatrix}$	3.9 s	4 s	21.67	21.05
Assembling and LU-fact. of $\boldsymbol{S}_{\Pi\Pi}$	0.78 s	1.1 s	4.33	5.79
Assemble rhs.	0.13 s	0.13 s	0.72	0.68
Total assembling	14 s	15 s	77.78	78.95
One PCG iteration	0.34 s	0.36 s	-	-
Solving the system	3.4 s	3.6 s	18.89	18.95
Calculating the solution $\boldsymbol{u}$	0.33 s	0.33 s	1.83	1.74
Total spent time	18 s	19 s	100.00	100.00

Table 4.9: Serial computation times of the 2d example with coefficient scaling and Algorithm C. Column 2 and 3 present the absolute spent time, whereas column 4 and 5 present the relative one for the cG-IETI-DP and dG-IETI-DP method.

overlaps is theoretically more demanding, due to the presence of possibly two different diffusion coefficients at the overlapping region, see [100] and [98].

This sections gives an overview of the results established in [97], [99], [98] and [100]. We mention, that the topic of this thesis is the development of solvers for systems arising from IgA of diffusion problems. Therefore, in this section, we focus on the application of IETI-DP methods to such kind of problems. Nevertheless, we give a short introduction to the construction of appropriate dG-schemes. For the simplicity of the presentation, we only give a very brief overview of the construction of dG-schemes for overlapping regions. In order to prevent lengthy formulas and to keep the presentation of this section simple, we will use the notation u_i for the restriction of u to patch $\Omega^{(i)}$ instead of $u^{(i)}$. Similarly, we use ρ_i for the diffusion coefficient ρ restricted to patch $\Omega^{(i)}$. For simplicity, we only consider homogeneous Dirichlet conditions.

4.4.1 Motivation and Setup

As already mentioned in the introduction, a CAD system provides the description of the computational domain only via its boundary representation. In order to perform IgA, it is necessary to get a volume representation of the geometry. This procedure is called volume *segmentation*, and is quite complicated and involved, see, e.g., [110],

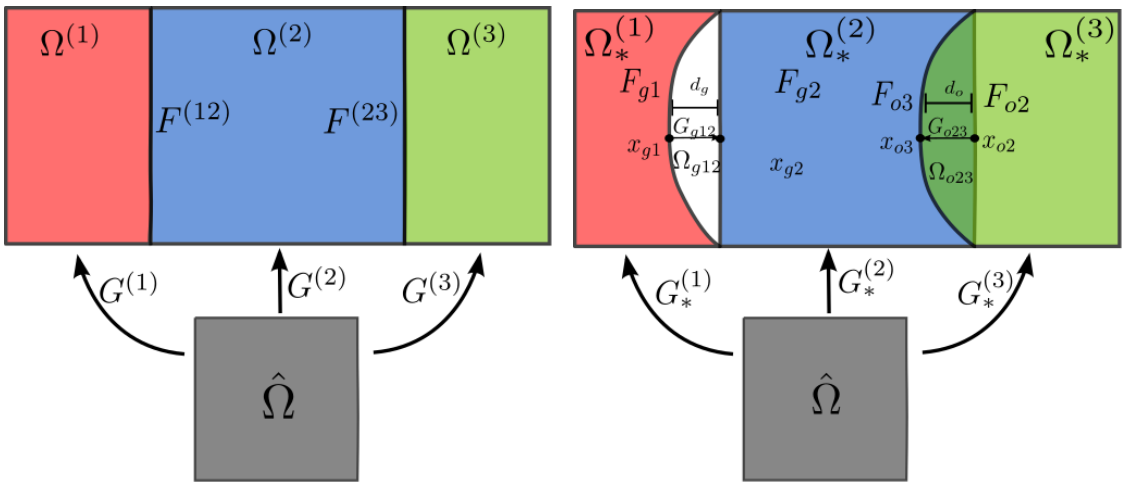


Figure 4.6: (a) Illustration of a decomposition with matching interfaces, (b) occurrence of gaps and overlaps.

[173], [179], [174] and [106]. During this procedure, the domain is cut in such a way that the created pieces (subdomains) are topologically equivalent to cubes. After the calculation of the corresponding control points, each of those subdomains is then parametrized via splines giving a volume representation. The collection of all this volume patches then forms the resulting multi-patch geometry.

However, in certain cases, it is possible that non-conforming parametrizations of the patch interfaces occur, this means that the patch interfaces are not identical. More precisely, during the construction of the parametrization of a patch, lets say $\Omega_*^{(i)}$, the control points which are related to an interface may have not appropriately been determined with the corresponding control points of the adjacent patch $\Omega_*^{(j)}$ for $i \neq j$. The result is a decomposition of Ω in patches $\Omega_*^{(i)}$, where small gap and/or overlapping regions can occur at the interfaces, see Figure 4.6(b). We refer to such decompositions as *non-matching interface parametrizations* or *segmentation crimes*. For the analysis of the method, we assume that there exits so-called *physical* patches $\Omega^{(i)}, i = 1, \ldots, N$, which form a non-overlapping decomposition of Ω. The corresponding interfaces $F^{(ij)} := \overline{\Omega}^{(i)} \cap \overline{\Omega}^{(j)}$ are called *physical* interfaces, see Figure 4.6(a).

Gap Regions

In this section, we describe a gap region which is located between two patches. Let $\Omega_*^{(1)}$ and $\Omega_*^{(2)}$ be two adjacent patches with the corresponding non-matching interface parametrizations $G_*^{(1)} : \hat{\Omega} \to \Omega_*^{(1)}$ and $G_*^{(2)} : \hat{\Omega} \to \Omega_*^{(2)}$. Further, let Ω_{g12} be the gap region such that $\overline{\Omega}_{g12} \subset \overline{\Omega}$ and $|\partial\Omega_{g12} \cap \partial\Omega| = 0$. Based on our assumption, we have

$$\overline{\Omega}_*^{(1)} \cup \overline{\Omega}_*^{(2)} \cup \overline{\Omega}_{g12} = \overline{\Omega}^{(1)} \cup \overline{\Omega}^{(2)}.$$

For simplicity of the presentation, we assume that the face F_{g2} coincides with the physical interface $F^{(12)}$. This implies that $\Omega_*^{(2)} = \Omega^{(2)}$. An illustration is given in Figure 4.6(b). Without loss of generality, we consider the boundary of the gap region as $\partial\Omega_{g12} = F_{g1} \cup F_{g2}$, with $F_{gi} \subset \Omega_*^{(i)}$, $i = 1, 2$. Moreover, we assume that the face F_{g1} can be described as the set of points $x = (x_1, x_2, x_3)$ satisfying

$$0 \leq x_1 \leq M_{x_1},\ 0 \leq x_2 \leq M_{x_2},\ x_3 = \zeta_g(x_1, x_2), \tag{4.58}$$

where M_{x_1} and M_{x_2} are real numbers, and $\zeta_g(x_1, x_2)$ is a B-Spline function of the same degree as the mapping $G_*^{(1)}$. More precisely, the face F_{g1} is the image of a face of $\partial\hat{\Omega}$ under the mapping $G_*^{(1)}$. Next, we construct a parametrization of F_{g1} as $G_{g21} : F_{g2} \to F_{g1}$ defined as

$$x_{g2} \in F_{g2} \to G_{g21}(x_{g2}) := x_{g1} \in F_{g1}, \quad \text{with} \quad G_{g21}(x_{g2}) = x_{g2} + \zeta_g(x_{g2}) n_{F_{g2}}, \tag{4.59}$$

where $n_{F_{g2}}$ is the unit normal vector on F_{g2}, and ζ_g is as in (4.58).

We will couple the resulting discrete problems in $\Omega_*^{(1)}$ and $\Omega_*^{(2)}$ using dG techniques, i.e., by introducing appropriate numerical fluxes on F_{g1} and F_{g2}. For their construction, we need to assign the points located on F_{g2} to the diametrically opposite points located on F_{g1}, described by the mapping G_{g21}. We are only interested in small gap regions, cf. (4.61), and, thereby, if $n_{F_{g1}}$ is the unit normal vector on F_{g1}, we can suppose that $n_{F_{g2}} \approx -n_{F_{g1}}$. Consequently, we can define the mapping $G_{g12} : F_{g1} \to F_{g2}$ to be

$$G_{g12}(x_{g1}) = x_{g2}, \ \text{with} \quad G_{g21}(x_{g2}) = x_{g1}. \tag{4.60}$$

We note that both parametrizations G_{g21} in (4.59) and G_{g12} in (4.60) have been constructed under the consideration that one side is planar and $n_{F_{g2}} \approx -n_{F_{g1}}$. In Section 4.4.3, we present numerical tests where all the faces of the gap and overlapping regions are curved surfaces. In order to quantify the size of the gap, we introduce the gap width as follows

$$d_g = \max_{x_{g2} \in F_{g2}} |x_{g2} - G_{g21}(x_{g2})|.$$

We focus on gap regions whose width decreases polynomially in h, that is,

$$d_g \leq C h^{\lambda}, \quad \text{with some} \quad \lambda \geq 1. \tag{4.61}$$

Overlapping Regions

We now describe the case of having an overlapping region at the interface. For simplicity of the presentation, we assume that the overlapping region is formed by involving two patches. Due to an incorrect segmentation procedure, we obtain two mappings, lets say $G_*^{(2)} : \hat{\Omega} \to \Omega_*^{(2)}$ and $G_*^{(3)} : \hat{\Omega} \to \Omega_*^{(3)}$, which cannot exactly parametrize the

two physical patches $\Omega^{(2)}$ and $\Omega^{(3)}$. This leads to the corresponding "incorrect" patches $\Omega_*^{(2)}$ and $\Omega_*^{(3)}$. The overlapping region is then given by $\Omega_{o23} = \Omega_*^{(2)} \cap \Omega_*^{(3)}$. We denote the interior boundary faces by $F_{oi} = \partial\Omega_*^{(i)} \cap \Omega_*^{(j)}$, with $i, j = 2, 3$ and $i \neq j$, and their unit normal vector by $n_{F_{oi}}$. An illustration is given in Figure 4.6. Having analogous assumptions as in the case of gap regions and following a similar argumentation, we can introduce the corresponding mappings G_{o23} and G_{o32}, and define the width of the overlapping region as

$$d_o = \max_{x_{o2} \in F_{o2}} |x_{o2} - G_{o23}(x_{o2})|.$$

We focus on overlapping regions whose distance decreases polynomially in h, i.e.,

$$d_o \leq Ch^\lambda, \quad \text{with some} \quad \lambda \geq 1. \tag{4.62}$$

Remark 4.19. *We note that the mappings G_{g12}, G_{g21}, G_{o23} and G_{o32} are introduced and used only for deriving the discretization error analysis. They are not used in the computation of the entries of the system matrix of the discrete dG-IgA scheme*

4.4.2 Discrete dG-Scheme

In this section, we formulate the discrete dG-scheme for domains with small gap and overlapping regions located the patch interfaces. As already mentioned in the beginning of Section 4.4, we will only briefly comment on the construction of such schemes in case of overlaps.

Gap Regions

First, we need some assumptions on the smoothness of the exact solution. Similar as in Theorem 2.19, we make the following assumption.

Assumption 5. We assume that the solution u of (2.2) belongs to the space $V := H^1(\Omega) \cap H^l(\mathcal{T}_H(\Omega))$ with some $l \geq 2$, where $H^l(\mathcal{T}_H(\Omega)) := \{u \in L^2(\Omega) : u_i := u|_{\Omega^{(i)}} \in H^l(\Omega^{(i)}), \text{for } i = 1, 2\}$.

For a $u \in V$, which solves (2.2), we can deduce the following interface conditions

$$\begin{aligned} [\![u]\!] &:= u_1 - u_2 = 0 \text{ on } F^{(12)}, \\ [\![\rho\nabla u]\!] \cdot n_{F^{(12)}} &:= (\rho_1 \nabla u_1 - \rho_2 \nabla u_2) \cdot n_{F^{(12)}} = 0 \text{ on } F^{(12)}. \end{aligned} \tag{4.63}$$

By Assumption 5, we have that $u_i \in H^l(\Omega_*^{(i)})$, where $l \geq 2$, $i = 1, 2$, and the following interface conditions for the solution u on the two faces of $\partial\Omega_{g12}$,

$$[\![u]\!]\big|_{F_{gi}} = 0 \quad \text{and} \quad [\![\rho\nabla u]\!] \cdot n_{F_{gi}}\big|_{F_{gi}} = 0 \text{ with } i = 1, 2. \tag{4.64}$$

The idea for deriving dG-IgA schemes on multi-patch domains with gap regions is to approximate the normal fluxes on $\partial\Omega_{g12}$ by means of known interior values. More precisely, the normal fluxes are replaced by Taylor expansions. They help us to construct appropriate numerical fluxes across opposite faces at $\partial\Omega_{g12}$. Finally, these fluxes play the role of a bridge for coupling the discrete patch-wise problems. We note that for matching interfaces, these numerical fluxes reduce to the classical dG numerical fluxes. For completeness, we present the Taylor expansions with integral remainder term

$$u(y) = u(x) + \nabla u(x) \cdot (y - x) + R^2 u_y, \tag{4.65}$$

where $R^2 u_y$ is the second-order remainder term defined by

$$R^2 u_y = \sum_{|\alpha|=2} (y - x)^\alpha \frac{2}{\alpha!} \int_0^1 s D^\alpha u(y + s(x - y))\, ds.$$

By simple computations, one obtains the following identities from (4.65):

$$\begin{aligned} \nabla u(y)(y - x) &= \nabla u(x) \cdot (y - x) + R^2 u_x + R^2 u_y, \\ -\big(u(x) - u(y)\big) &= \nabla u(x) \cdot (y - x) + R^2 u_y. \end{aligned} \tag{4.66}$$

Let $x_{g2} \in F_{g2}$ and let $x_{g1} \in F_{g1}$ be two opposite points such that $x_{g1} = G_{g21}(x_{g2})$ and $x_{g2} = G_{g12}(x_{g1})$. Denoting $r_{g1} = x_{g1} - x_{g2}$ and having $r_{g2} = -r_{g1}$, we obtain that the normals are given by $n_{F_{g1}} = r_{g1}/|r_{g1}|$ and $n_{F_{g2}} = r_{g2}/|r_{g2}|$. As usual, we denote the average of the diffusion coefficient across by $\{\rho\} := (\rho_1 + \rho_2)/2$. By considering x_{g2} and x_{g1} to play the role of x and y in (4.66) and using (4.64), we can derive the following expressions for the normal fluxes and jump terms of u on $\partial\Omega_{g12}$.

$$\begin{aligned} \nabla u_{g12}(x_{g2}) \cdot n_{F_{g2}} &= \nabla u_1(x_{g1}) \cdot n_{F_{g2}} - \frac{1}{|r_{g1}|}\big(R^2 u_{x_{g2}} + R^2 u_{x_{g1}}\big), \\ -\frac{1}{h}\big(u_2(x_{g2}) - u_1(x_{g1})\big) &= \frac{|r_{g1}|}{h} \nabla u_{g12}(x_{g2}) \cdot n_{F_{g2}} + \frac{1}{h} R^2 u_{x_{g1}}. \end{aligned} \tag{4.67}$$

Now, following the dG framework, we multiply (2.1) by a $\phi_h \in V_h$, apply integration by parts and replace the normal fluxes on $\partial\Omega_{g12}$ by those, given in (4.67). We arrive

at the variational formulation

$$\begin{aligned}
\sum_{i=1}^{2}\int_{\Omega_*^{(i)}}\rho_i\nabla u\cdot\nabla\phi_h\,dx-\int_{\partial\Omega_*^{(i)}\cap\partial\Omega}\rho_i\nabla u\cdot n_{\partial\Omega_*^{(i)}}\phi_h\,d\sigma\\
-\int_{F_{g2}}\Big(\frac{\rho_2}{2}\nabla u_2+\frac{\rho_1}{2}\nabla u_1\Big)\cdot n_{F_{g2}}\phi_h-\frac{\{\rho\}}{h}\big(u_2-u_1\big)\phi_h\,d\sigma\\
-\int_{F_{g1}}\Big(\frac{\rho_1}{2}\nabla u_1+\frac{\rho_2}{2}\nabla u_2\Big)\cdot n_{F_{g1}}\phi_h-\frac{\{\rho\}}{h}\big(u_1-u_2\big)\phi_h\,d\sigma\\
+\int_{F_{g2}}\Big(\frac{\rho_{g2}}{2|r_{g2}|}\Big(R^2u_{x_{g1}}+R^2u_{x_{g2}}\Big)-\frac{\{\rho\}}{h}\Big(|r_{g2}|\nabla u_{g2}\cdot n_{F_{g2}}+R^2u_{x_{g2}}\Big)\Big)\phi_h\,d\sigma\\
+\int_{F_{g1}}\Big(\frac{\rho_{g1}}{2|r_{g1}|}\Big(R^2u_{x_{g1}}+R^2u_{x_{g2}}\Big)-\frac{\{\rho\}}{h}\Big(|r_{g1}|\nabla u_{g1}\cdot n_{F_{g1}}+R^2u_{x_{g1}}\Big)\Big)\phi_h\,d\sigma\\
=\int_{\Omega\setminus\overline{\Omega}_{g12}}f\phi_h\,dx,\ \text{for all }\phi_h\in V_h.\quad(4.68)
\end{aligned}$$

Remark 4.20. *The spline space V_h is defined via the geometrical mappings $G_*^{(i)}$.*

Clearly, (4.68) cannot be directly implemented due to the presence of the Taylor residuals. For the final dG-IgA formulation (4.70), see below, the Taylor residuals we will be omitted.

For all $v\in V_h^*:=V+V_h$ we introduce the dG-norm $\|.\|_{dG}$

$$\|v\|_{dG}^2=\sum_{i=1}^{2}\Big(\rho_i\|\nabla v_i\|^2_{L^2(\Omega_*^{(i)})}+\frac{\rho_i}{h}\|v_i\|^2_{L^2(\partial\Omega_*^{(i)}\cap\partial\Omega)}+\sum_{F_{gi}\subset\partial\Omega_*^{(i)}}\frac{\{\rho\}}{h}\|v_i\|^2_{L^2(F_{gi})}\Big).\quad(4.69)$$

Recalling the dG-scheme (4.68), we define the bilinear form $B_{\Omega_*}(\cdot,\cdot):V_h^*\times V_h\to\mathbb{R}$, and the linear functional $F_h:V_h\to\mathbb{R}$ by the relation

$$\begin{aligned}
B_{\Omega_*}(u,\phi_h)=&\sum_{i=1}^{2}\Big(\int_{\Omega_*^{(i)}}\rho_i\nabla u_i\cdot\nabla\phi_h\,dx-\int_{\partial\Omega_*^{(i)}\cap\partial\Omega}\rho_i\nabla u_i\cdot n_{\partial\Omega_*^{(i)}}\phi_h\,d\sigma\\
&-\sum_{F_{gj}\subset\partial\Omega_*^{(i)}}\int_{F_{gj}}\Big(\frac{\rho_i}{2}\nabla u_i+\frac{\rho_j}{2}\nabla u_j\Big)\cdot n_{F_{gj}}\phi_h-\frac{\eta\{\rho\}}{h}\big(u_i-u_j\big)\phi_h\,d\sigma\Big)\qquad(4.70)\\
F_h(\phi_h)=&\sum_{i=1}^{2}\int_{\Omega_*^{(i)}}f\phi_h\,dx,
\end{aligned}$$

where $\eta>0$ is a parameter that will be defined in order to achieve the coercivity of the resulting dG bilinear form on the IgA space V_h. Hence, by incorporating homogeneous Dirichlet conditions in a weak sense, we introduce the bilinear form $B_h(\cdot,\cdot):V_h\times V_h\to$

$\mathbb{R}$ as follows

$$B_h(u_h, \phi_h) = B_{\Omega_*}(u_h, \phi_h) + \sum_{i=1}^{2} \frac{\eta \rho_i}{h} \int_{\partial\Omega_*^{(i)} \cap \partial\Omega} u_h \phi_h \, d\sigma \tag{4.71}$$

Finally, the dG-IgA scheme, which we are going to solve, reads as follows: find $u_h \in V_h$ such that

$$B_h(u_h, \phi_h) = F_h(\phi_h), \quad \text{for all } \phi_h \in V_h. \tag{4.72}$$

Lemma 4.21. *The bilinear form $B_h(\cdot,\cdot)$ in (4.71) is elliptic on V_h, i.e., there is a positive constant C_m such that the estimate*

$$B_h(v_h, v_h) \geq C_m \|v_h\|_{dG}^2, \tag{4.73}$$

hold for all $v_h \in V_h$ provided that η is sufficiently large.

Proof. See Lemma 3.3 in [97]. □

The ellipticity of the bilinear form $B_h(\cdot,\cdot)$ already yields that the discrete problem (4.72) has a unique solution.

The error analysis uses a variation of Strang's Lemma, and for sufficiently small d_g, i.e., $d_g \leq Ch^\lambda$ with $\lambda \geq p - 1/2$, it is proven that the method has optimal approximation properties. For a detailed discussion about constructing dG-IgA schemes for decompositions having gap regions and their corresponding error analysis, we refer to [97] and [99]. We conclude by presenting the main discretization error bound.

Theorem 4.22. *Let Assumption 5 hold, let u be the solution of problem (2.2), let u_h be the corresponding dG-IgA solution of problem (4.72), and let $d_g \leq Ch^\lambda$ with $\lambda \geq 1$. Then the a-priori error estimate*

$$\|u - u_h\|_{dG} \lesssim h^r \big(\sum_{i=1}^{2} \|u\|_{H^l(\Omega_*^{(i)})} + \mathcal{K}_g \big), \tag{4.74}$$

holds, where $r = \min(s, \beta)$ with $s = \min(p+1, l) - 1$ and $\beta = \lambda - \frac{1}{2}$, and $\mathcal{K}_g$ depends on the Taylor residuals appearing in (4.68).

Proof. See Theorem 4.3 in [97]. □

Remark 4.23. *The estimate (4.74) is referred to the case where the maximum width d_g is of order $\mathcal{O}(h^\lambda)$. If the width d_g is fixed, i.e., is not decreased as we refine the meshes, we are in the case where $\lambda = 0$. Following the lines of the proof, we can infer that the estimate (4.74) will take the form*

$$\|u - u_h\|_{dG} \lesssim h^s \sum_{i=1}^{2} \|u\|_{H^l(\Omega_*^{(i)})} + d_g \, h^{-\frac{1}{2}} \, \mathcal{K}_g, \tag{4.75}$$

where $s = \min(p+1, l) - 1$ and $\mathcal{K}_g$ is as in Theorem 4.22.

Overlapping Regions

The case of overlapping regions is more involved and very technical due to the presence of possibly different diffusion coefficients on the overlapping region. This leads to the co-existence of two different numerical solutions on the overlap. Therefore, we will only give a brief overview of this topic. For a more detailed discussion, we refer to [100].

The idea is to introduce local (patch-wise) auxiliary variational problems, which are compatible with the overlapping nature of the multi-patch representation of Ω. The solutions are denoted by u_2^* and u_3^*, and we identify their pair (u_2^*, u_3^*) by u^*, which is equal to u_i^* on $\Omega_*^{(i)}$ for $i = 2, 3$. On each $\Omega_*^{(i)}$, $i = 1, 2$, we consider the following auxiliary problem

$$(P) \begin{cases} -\mathrm{div}(\rho_i \nabla u_i^*) & = f, \text{ in } \Omega_*^{(i)}, \\ u_i^* = 0 & \text{on } \partial\Omega_*^{(i)} \cap \partial\Omega, \\ u_i^* = u & \text{on } F_{oi}. \end{cases}$$

Note that the solution u of (2.1) does not satisfy in general (P) on $\Omega_*^{(3)}$. This implies that an additional consistency error between the solution u^* and u has to be taken into account.

In order to derive our dG-IgA scheme, we consider the problem (P) on both domains $\Omega_*^{(2)}$ and $\Omega_*^{(3)}$, multiply them by a testfunction $\phi_h \in V_h$ and apply integration by parts. We follow the same ideas as in the case of gap regions by using Taylor expansions to provide approximations of the numerical fluxes on F_{o2} and F_{o3}. This allows us to formulate a variational formulation for u^* including Taylor residual terms, similar as in the case of gaps, cf. (4.68). Analogously, we can derive the following variational problem for u_h^*: find $u_h^* \in V_h$ such that

$$B_h(u_h^*, \phi_h) = F_h(\phi_h), \quad \text{for all } \phi_h \in V_h, \tag{4.76}$$

where the bilinear form $B_h(\cdot,\cdot) : V_h \times V_h \to \mathbb{R}$ and the linear functional $F_h : V_h \to \mathbb{R}$ are defined by

$$B_h(u_h^*, \phi_h) = B_{\Omega_*}(u_h^*, \phi_h) + \sum_{i=2}^{3} \frac{\eta \rho_i}{h} \int_{\partial\Omega_*^{(i)} \cap \partial\Omega} u_h^* \phi_h \, d\sigma$$

$$F_h(\phi_h) = \sum_{i=2}^{3} \int_{\Omega_*^{(i)}} f \phi_h \, dx.$$

with

$$B_{\Omega_*}(u^*,\phi_h) = \sum_{i=2}^{3}\Big(\int_{\Omega_*^{(i)}} \rho_i \nabla u_i^* \cdot \nabla \phi_h \, dx - \int_{\partial\Omega_*^{(i)}\cap\partial\Omega} \rho_i \nabla u_i^* \cdot n_{\partial\Omega_*^{(i)}} \phi_h \, d\sigma$$
$$- \sum_{F_{oj}\subset\partial\Omega_*^{(i)}} \int_{F_{oj}} \Big(\frac{\rho_i}{2}\nabla u_i^* + \frac{\rho_j}{2}\nabla u_j^*\Big)\cdot n_{F_{oj}}\phi_h - \frac{\eta\{\rho\}}{h}\big(u_i^* - u_j^*\big)\phi_h \, d\sigma\Big).$$

It is possible to show coercivity of $B_h(u_h^*, \phi_h)$, see Lemma 2 in [100]. However, we can not directly infer that u_h^* can approximate the solution u of the original problem in an optimal way. By means of the triangle inequality, the discretization error $\|u - u_h^*\|_{dG}$ is split as follows

$$\|u - u_h^*\|_{dG} \leq \|u^* - u_h^*\|_{dG} + \|u - u^*\|_{dG}, \tag{4.77}$$

where each term has to be analyzed separately. Finally, in [100] it is shown, that both parts of the right hand side of (4.77) can be bounded in terms of the mesh size h, which leads to the following discretization error estimate.

Theorem 4.24. *Let u be the solution of problem (2.2), let u_h^* be the dG-IgA solution of (4.76), and let $d_o \leq Ch^\lambda$ with $\lambda \geq 1$. Then, under additional regularity assumptions on u and u^*, the error estimate*

$$\|u - u_h^*\|_{dG} \lesssim h^r\Big(\sum_{i=2}^{3}\Big(\|u\|_{H^l(\Omega_*^{(i)})} + \|u_i^*\|_{H^l(\Omega_*^{(i)})}\Big) + \mathcal{K}_o\Big), \tag{4.78}$$

holds, where $r = \min(s, \beta)$ with $s = \min(p+1, l) - 1$ and $\beta = \lambda - \frac{1}{2}$, and $\mathcal{K}_o$ depends on the Taylor residuals of u and u^*, and on f.

Proof. See Theorem 2 in [100]. □

Remark 4.25. *According to Remark 4.23, we obtain for a fixed size of the overlapping region $d_o = O(1)$ the following bound*

$$\|u - u_h^*\|_{dG} \lesssim h^s \sum_{i=2}^{3}\Big(\|u\|_{H^l(\Omega_*^{(i)})} + \|u_i^*\|_{H^l(\Omega_*^{(i)})}\Big) + d_o\, h^{-\frac{1}{2}}\,\mathcal{K}_o,$$

where $s = \min(p+1, l) - 1$ and $\mathcal{K}_o$ as in Theorem 4.24.

4.4.3 Numerical Examples

In this section, we perform several numerical tests with different shapes of gap and overlapping regions as well as combinations with inhomogeneous diffusion coefficients

	B-Spline degree $p \geq 2$			
	Smooth solutions, $u \in H^{l \geq p+1}(\Omega)$			
$d_M = h^\lambda$	$\lambda = 1$	$\lambda = 2$	$\lambda = 2.5$	$\lambda = 3$
$\beta :=$	0.5	1.5	2	2.5
$s :=$	p	p	p	p
$r :=$	0.5	1.5	$\min(p, 2)$	$\min(p, 2.5)$

Table 4.10: The values of the predicted order r of the estimate (4.74) and (4.78).

for two- and three-dimensional domains. We investigate the order of accuracy of the dG-IgA scheme proposed in (4.72) and (4.76). We compare the error convergence rates versus the grid size h for several gap/overlapping distances $d_M := \max(d_o, d_g) = h^\lambda$, with $\lambda \in \{1, 2, 2.5, 3\}$. Every example has been solved applying several mesh refinement steps with $h_i, h_{i+1}, \ldots$, satisfying Assumption 2. Moreover, the dG techniques allow us to consider non-matching meshes. If we keep a constant linear relation between the sizes of the different meshes, the approximation properties of the method are not affected, see [148]. The numerical convergence rates r have been computed by the ratio $r = \ln(e_i/e_{i+1})/\ln(h_i/h_{i+1})$, $i = 1, 2, \ldots$, where $e_i := \|u - u_h\|_{dG}$ is the error. In order to have a unified presentation, we will also use u_h instead of u_h^* for the dG-IgA solution in case of overlaps. We use highly smooth solutions on each patch, i.e., $p + 1 \leq l$, and therefore the order s in (4.74) and (4.78) becomes $s = p$. The predicted values of power β, the order s and the order r for several values of λ are displayed in Table 4.10. All tests have been performed in G+SMo.

The gap and overlapping regions are artificially created by moving the control points, which are related to the interfaces $F^{(ij)}$, in the direction of $n_{F^{(ij)}}$ or of $-n_{F^{(ij)}}$. This allows us to control the width of the gap or overlapping region by just moving the corresponding control points.

In order to solve the resulting linear system, we use the dG-IETI-DP method, that is described in Section 4.1, as a fast and robust solver. The derived dG-formulation fits perfectly in the framework of the dG-IETI-DP method. We use the conjugate gradient method for solving (4.27) preconditioned with the scaled Dirichlet preconditioner (4.28). We start with zero initial guess and iterate until we reach a reduction of the initial residual in the ℓ_2-norm by a factor 10^{-10}. We use such a high accuracy only for studying the convergence behaviour of the dG-IETI-DP method. Of course, in practice, the stopping criterion is adapted to the discretization error.

Two-Dimensional Numerical Examples

Example 1: Uniform Diffusion Coefficient $\rho_i = 1, i = 1, \ldots, N$. The first numerical example is a simple test case demonstrating the applicability of the proposed technique for constructing dG-IgA schemes on decompositions including gaps

and overlaps with general shape. The domain Ω with the $N = 21$ subdomains $\Omega_*^{(i)}$ and the initial mesh are shown in Figure 4.7(a). We note that we consider non-matching meshes across the interfaces. The Dirichlet boundary condition and the right-hand side f are determined by the smooth exact solution $u(x, y) = \sin(\pi(x + 0.4)/6) \sin(\pi(y + 0.3)/3) + x + y$. In this example, we consider a homogeneous diffusion coefficient, i.e., $\rho_i = 1$ for all $\Omega_*^{(i)}$, $i = 1, \ldots, N$.

We compute convergence rates for the following four different values of $\lambda \in \{1, 2, 2.5, 3\}$. The width of the gap and overlapping region is derived according to h^λ. Since we are using B-Splines of degree two we expect optimal convergence rates for $\lambda = 2.5$ and $\lambda = 3$, see Table 4.10. The results are in very good agreement with the theoretically predicted estimates given in Table 4.10. We observe optimal rates for the cases where $\lambda \geq 2.5$. The rates are visualized in Figure 4.7(b).

As a second test, we solve the problem considering gap and overlapping regions with fixed width $d_M = 0.004$, cf., Remark 4.23 and Remark 4.25. We start by solving the problem on coarse meshes with $h^2 > d_M h^{-\frac{1}{2}}$, then we continue to use finer meshes and finally we solve the problem on meshes with grid size $h^2 < d_M h^{-\frac{1}{2}}$. The convergence rates are shown in Figure 4.7(c). For the first mesh levels, we obtain the expected optimal convergence rates, since the error coming from the approximation properties of the B-Spline space dominates the approximation error coming from the approximation of the normal fluxes at the interfaces. On the finer levels, where $h^2 \sim d_M h^{-\frac{1}{2}}$, the rates are gradually reduced and get closer to zero, because the whole discretization error is not further decreasing as we refine the mesh. As we move into the most refined meshes, the error related to the approximation of B-Splines is negligible compared to the error related to the approximation of the fluxes on the interface. This should yield the rate -0.5. In the numerical computations, this fact is depicted in the negative rate $r = -0.48$ that we have found on the last level of refinement in Figure 4.8(c).

The dG-IETI-DP method performs very well. The condition number κ and CG iterations (It.) are summarized in Table 4.11 for the case $d_M = h$ with coefficient and stiffness scaling. We observe the quasi-optimal behaviour of the condition number with respect to h. Moreover, the existence of gaps and overlaps does not affect the condition number, cf. Theorem 4.16. Furthermore, both scalings give nearly identical results.

Example 2: Different Diffusion Coefficient. In the second example, we study the case of having smooth solutions on each $\Omega^{(i)}$ but discontinuous coefficients across the interfaces. We choose $\rho_i \in \{1, \frac{3\pi}{4} \approx 2.35, 10^4\}$ according to the pattern in Figure 4.8(b). We consider a square build by a 3×3 grid of patches, giving in total $N = 9$ patches. The domain Ω with its patches $\Omega_*^{(i)}$ and the solution after one refinement is

		ALG. A				ALG. C			
		coefficient scal.		stiffness scal.		coefficient scal.		stiffness scal.	
# dofs	H/h	κ	It.	κ	It.	κ	It.	κ	It.
654	4	1.6	10	1.6	10	1.2	7	1.2	7
1616	8	2.0	12	2.0	11	1.3	8	1.3	8
4716	16	2.4	13	2.5	13	1.5	10	1.5	10
15620	32	3.0	14	3.0	15	1.8	11	1.8	11
56244	64	3.5	15	3.7	16	2.2	12	2.2	12
212756	128	4.1	17	4.6	18	2.6	14	2.7	14
826836	256	4.8	18	5.9	21	3.1	15	3.2	17

Table 4.11: Example 1: Dependence of the condition number κ and the number It. of iterations on H/h for the preconditioned system with coefficient and stiffness scaling. Choice of primal variables: vertex evaluation (left), vertex evaluation and edge averages (right).

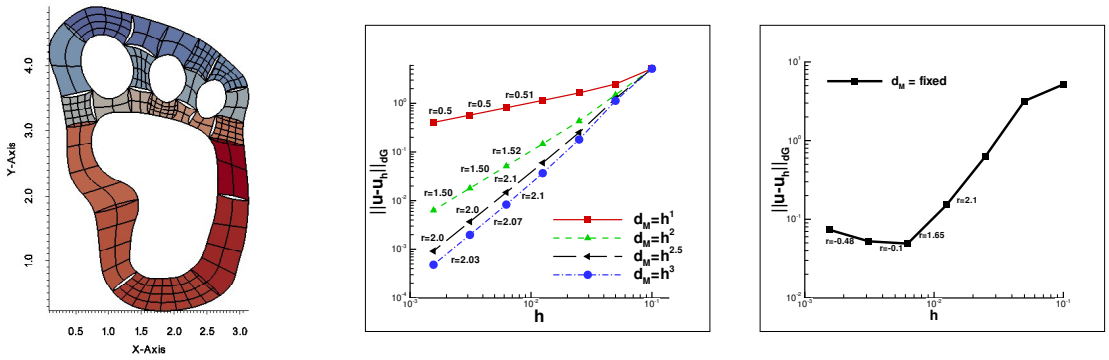


Figure 4.7: Example 1: (a) The patches $\Omega_*^{(i)}$ with the initial mesh. (b) The convergence rates for the different values of λ. (c) The convergence rates for fixed gap and overlapping widths, $d_M = 0.004$.

presented in Figure 4.8(a). The exact solution is given by

$$u(x,y) = \begin{cases} \exp(-\sin(3\pi x))\sin(\frac{7\pi y}{2}) & \text{if } \ x \in [0,1] \\ ((2x^2-1)+(x-1)^2(x-2)\gamma)\sin(\frac{7\pi y}{2}) & \text{if } x \in (1,2] \\ 7\exp(-\cos(\pi(x+1)/2)\sin(\frac{7\pi y}{2}) & \text{if } x \in (2,3], \end{cases} \tag{4.79}$$

where $\gamma = 14 \cdot 10^4 - 8$. The boundary conditions and the source function f are determined by (4.79). Note that in this test case, we have $[\![u]\!]|_F = 0$ as well as $[\![\rho \nabla u]\!]|_F \cdot n_F = 0$ for the normal flux on the interfaces. The problem has been solved on several meshes following a sequential refinement process, where we set $d_M = h^\lambda$, with $\lambda \in \{1, 2, 3, 3.5, 4\}$. For the numerical tests, we use B-Splines of the degree $p = 3$. Hence, we expect optimal rates for $\lambda = 3.5$ and $\lambda = 4$. The computed rates are presented in Figure 4.8(c). By this example, we numerically observe the same convergence rates for

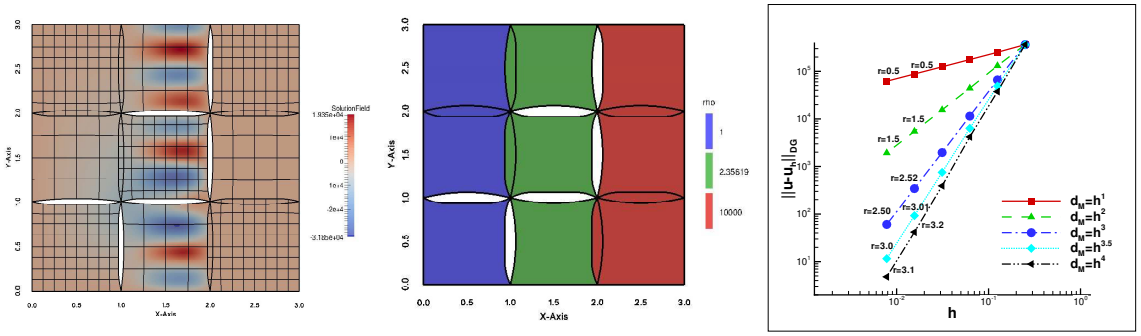


Figure 4.8: Example 2: (a) The contours of u_h on the subdomains $\Omega^{(i)}$ with $d_g = 0.06$. (b) Pattern of diffusion coefficients ρ_i. (c) The convergence rates for the 5 choices of λ.

diffusion problems with discontinuous coefficient on decompositions having gaps and overlapping regions.

Moreover, the dG-IETI-DP method seems to be robust with respect to jumping diffusion coefficients in the presence of complicated gaps and overlaps. The condition number κ and CG iterations (It.) are summarized in Table 4.12 for the case $d_M = h$ using coefficient and stiffness scaling. As in the previous example, we observe the expected quasi-optimal dependence of the condition number on H/h. For this domain, the usage of more primal variables brings a significant advantage, see column ALG. C in Table 4.12.

		ALG. A				ALG. C			
		coefficient scal.		stiffness scal.		coefficient scal.		stiffness scal.	
# dofs	H/h	κ	It.	κ	It.	κ	It.	κ	It.
788	4	3.9	11	3.9	11	1.6	8	1.6	8
1452	8	3.8	11	3.8	11	1.6	8	1.6	9
3356	16	4.3	12	4.4	12	1.7	9	1.7	10
9468	32	5.1	13	5.3	15	2.0	10	2.0	11
30908	64	6.0	15	6.5	16	2.3	11	2.3	12
110652	128	7.1	16	8.3	17	2.6	12	2.7	13

Table 4.12: Example 2: Dependence of the condition number (κ) and the number CG iterations (It.) on H/h for the preconditioned system with coefficient and stiffness scaling. Choice of primal variables: vertex evaluation (left), vertex evaluation and edge averages (right).

Example 3: Gap-Overlap on the same Interface. In this test, we apply the proposed method to decompositions having a more complex gap and overlapping region. The domain is given by 4 patches with non-matching meshes, where we artificially created both gap and overlapping region on one of the interfaces as depicted

in Figure 4.9(a). This region is located at the interface between $\Omega^{(2)}$ and $\Omega^{(3)}$ at $x = 0$. We note that the gap and the overlap are not separated by patches as in the previous examples. In addition, we consider inhomogeneous diffusion coefficients, i.e., $\rho_1 = \rho_2 = 3\pi$ and $\rho_3 = \rho_4 = 3$. The exact solution of the problem is given by

$$u(x,y) = \begin{cases} \sin(\pi(3x+y)) & \text{if } (x,y) \in \Omega^{(1)}, \Omega^{(2)}, \\ \sin(\pi(3\pi x+y)) & \text{if } (x,y) \in \Omega^{(3)}, \Omega^{(4)}. \end{cases} \tag{4.80}$$

The source function f and u_D are manufactured by the exact solution. We mention that the interface conditions $[\![u]\!]|_F = 0$ and $[\![\rho\nabla u]\!]|_F \cdot n_F = 0$ hold.

We have computed the convergence rates for varying size $d_M = h^\lambda$ for $\lambda = 1, 2, 2.5$ and $\lambda = 3$. In Figure 4.9(b), we plot the contours of u on the domains $\Omega_*^{(1)}, \ldots, \Omega_*^{(4)}$. In Figure 4.9(c), we present the obtained convergence rates. We observe that the convergent rates for all different cases of λ confirm the theoretically predicted values, see Table 4.10. The computational rates attain the optimal value $r = 2$ for $\lambda = 2.5$ and $\lambda = 3$, which is in agreement with the other examples.

The dG-IETI-DP method also performs very well in this case. Since we have only a small number of interface dofs, we get a quite small condition number and iteration count, see Table 4.13. However, due to having only few interface dofs, the usage of additional face averages as primal variables does not offer any significant advantage.

		ALG. A				ALG. C			
		coefficient scal.		stiffness scal.		coefficient scal.		stiffness scal.	
# dofs	H/h	κ	It.	κ	It.	κ	It.	κ	It.
184	2	1.1	4	1.1	5	1.1	4	1.1	4
460	4	1.1	4	1.1	5	1.1	5	1.1	5
1372	8	1.1	5	1.2	6	1.1	5	1.2	6
4636	16	1.2	5	1.3	6	1.2	5	1.3	6
16924	32	1.2	5	1.4	7	1.2	6	1.4	7
64540	64	1.2	5	1.9	9	1.2	6	1.9	9
251932	128	1.3	5	2.8	11	1.3	6	2.8	11

Table 4.13: Example 3, 2d example with inhomogeneous diffusion coefficient on a domain with a complicated interface. Dependence of the condition number (κ) and the number CG iterations (It.) on H/h for the preconditioned system with coefficient and stiffness scaling. Choice of primal variables: vertex evaluation (left), vertex evaluation and edge averages (right).

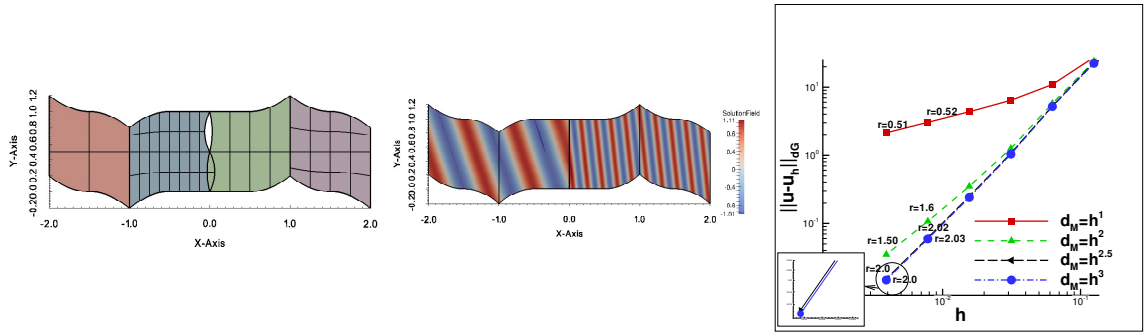


Figure 4.9: Example 3: (a) The 4 patches of the domain with its initial mesh and the gap and overlap region with $d_g = h$. (b) The contours of u on Ω on the domain without gap. (c) Convergence rates r for the 4 values of λ.

Three-Dimensional Numerical Examples

In the three-dimensional tests, the domain Ω has been constructed by a straight prolongation to the z-direction of the two-dimensional domains of Figure 4.7(a) and Figure 4.9(a). However, in contrast to the two-dimensional case, we start with matching meshes, as depicted in Figure 4.10(a) and Figure 4.11(a). The knot vector in z-direction is simply $\Xi_i^3 = \{0, 0, 0, 0.5, 1, 1, 1\}$ with $i = 1, \ldots, N$. The B-Spline parametrizations of these domains are constructed by adding a third component to the control points with the following values $\{0, 0.5, 1\}$. Again, the gap and overlapping regions are artificially constructed by moving only the interior control points located at the interface into the normal direction n_F of the related interfaces F. Due to the fact that the gap and overlap has to be inside of the domain, we have to provide cuts though the domain in order to visualize them, cf. Figure 4.10(b) and Figure 4.11(b).

Example 4: 3d Test with $\rho_i = 1$ for $i = 1, \ldots, N$. The computational domain is chosen as the domain from Example 1, extended to z-dimension as described above. The exact solution is given by $u(x, y, z) = \sin(\pi(x + 0.4)) \sin(2\pi(y + 0.3)) \sin(0.2\pi(z + 0.6))$ with homogeneous diffusion coefficient $\rho_i = 1$ for $i = 1, \ldots N$. The set up of the problem is illustrated in Figure 4.10. In Figure 4.10(a), we present the domain Ω with its $N = 21$ patches and the initial mesh. We use matching grids across the interface. In Figure 4.10(b), we plot the contours of the solution u_h resulting from the solution of the problem in case of having a gap and overlapping widths such that $d_M = 0.5$. Also, in Figure 4.10(b), we see the shape of the gaps as it appears on an oblique cut of the domain Ω. We note that at the interfaces in Figure 4.10(b) where no gap is visible, are overlapping regions, cf. Figure 4.7(a).

We have computed the convergence rates for four different values $\lambda \in \{1, 2, 2.5, 3\}$. The results of the computed rates are plotted in Figure 4.10(c). We observe that the obtained rates are in agreement with the convergent rates predicted by the theory, see Table 4.10. As already observed in Section 3.4 and Section 4.3, the condition number

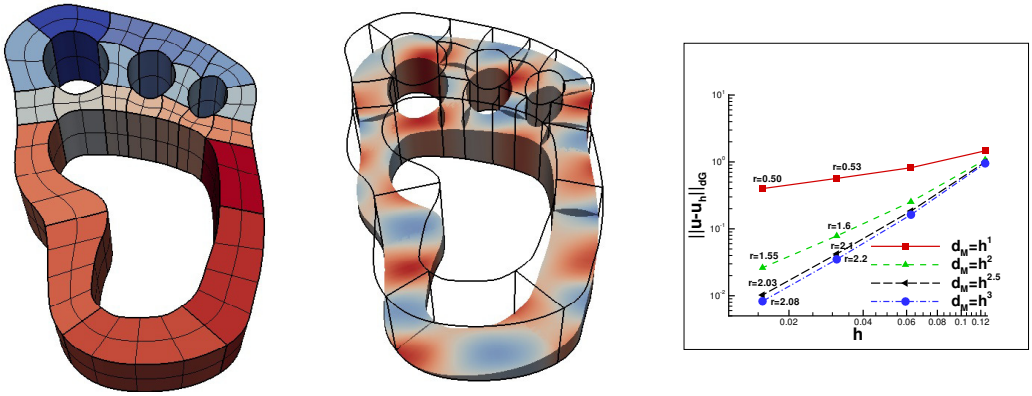


Figure 4.10: Example 4, $\Omega \subset \mathbb{R}^3$: (a) The multi-patch system with initial mesh. (b) The contours of u_h computed on $\Omega \setminus \overline{\Omega}_g$ with $d_g = 0.05$. (c) Convergence rates r for the three values of λ.

and CG iterations increases when having three-dimensional domains. The same behaviour can be observed here, although the condition number stays quite small, see Table 4.14. Moreover, we observe that the stiffness scaling provides better results than the coefficient scaling. For this example, it is not beneficial to use additional face averages as primal variable. Both algorithms give identical results.

		ALG. C				ALG. B			
		coefficient scal.		stiffness scal.		coefficient scal.		stiffness scal.	
# dofs	H/h	κ	It.	κ	It.	κ	It.	κ	It.
1488	2	1.2	10	1.1	10	1.2	10	1.1	10
5508	3	7.8	26	4.4	22	7.8	26	4.4	22
25908	7	10.9	32	6.3	26	10.9	32	6.3	26
149748	15	13.5	36	8.1	30	13.5	36	8.1	30
998388	31	16.3	40	10.2	34	16.3	40	10.2	34

Table 4.14: Example 4, 3d example. Dependence of the condition number (κ) and the number CG iterations (It.) on H/h for the preconditioned system with coefficient and stiffness scaling. Choice of primal variables: vertex evaluation and edge averages (left), vertex evaluation, edge and face averages (right).

Example 5: 3d Gap-Overlap Region on one Interface. For the last example, we choose the domain from Example 3 and extend it towards the z-direction with matching grids. As in Example 3, we have a gap and an overlap simultaneously on an interface. This fact is visualized in Figure 4.9(b). We consider a manufactured

problem, where the solution is given by

$$u(x,y,z) = \begin{cases} u_1 := \sin(\pi(2x+y+z)) & \text{if } (x,y,z) \in \Omega^{(1)}, \Omega^{(2)} \\ u_2 := \sin(\pi(3\pi x+y+z)) & \text{if } (x,y,z) \in \Omega^{(3)}, \Omega^{(4)} \end{cases} \tag{4.81}$$

and the diffusion coefficient is defined to be $\rho_1 = \rho_2 = 2\pi$ and $\rho_3 = \rho_4 = 3\pi$, as in Example 3. It is easy to see that we have the interface continuity conditions $[\![u]\!]|_F = [\![\rho\nabla u]\!]|_F \cdot n_F = 0$. As above, we solve the problem for $\lambda \in \{1, 2, 2.5, 3\}$. The contours of the solution u on Ω without gaps and overlapping regions are shown in Figure 4.11(a). In Figure 4.11(b), we plot the contours of the solution u_h on an oblique cut through the domain Ω, where the gap/overlap width is $d_M = 0.025$. We have computed the convergence rates r for the four different sizes d_M, obtained by the different values of λ. The results are plotted in Figure 4.11(c). We observe that the rates are approaching the values that have been mentioned in Table 4.10.

In this last example, the dG-IETI-DP method again provides satisfactory results, summarized in Table 4.15. Similar to Example 3, we only have three interfaces, therefore, only a small number of dofs corresponding to those. Hence, we again observe quite small condition numbers and iteration counts. Both choices of primal variables have again identical performance, but in contrast to the previous example, the coefficient scaling clearly provide better results than the stiffness scaling.

		ALG. C				ALG. B			
		coefficient scal.		stiffness scal.		coefficient scal.		stiffness scal.	
# dofs	H/h	κ	It.	κ	It.	κ	It.	κ	It.
424	2	1.2	7	2.0	9	1.2	7	2.0	9
1416	3	1.2	7	3.1	12	1.2	7	3.1	12
6376	7	1.3	7	4.1	16	1.3	7	4.1	16
36264	15	1.3	7	3.6	16	1.3	7	3.6	16
240424	31	1.3	7	3.0	14	1.3	7	3.0	14

Table 4.15: Example 5, 3d example with $p = 2$ and inhomogeneous diffusion coefficient. Dependence of the condition number (κ) and the number CG iterations (It.) on H/h for the preconditioned system with coefficient and stiffness scaling. Choice of primal variables: vertex evaluation and edge averages (left), vertex evaluation, edge and face averages (right).

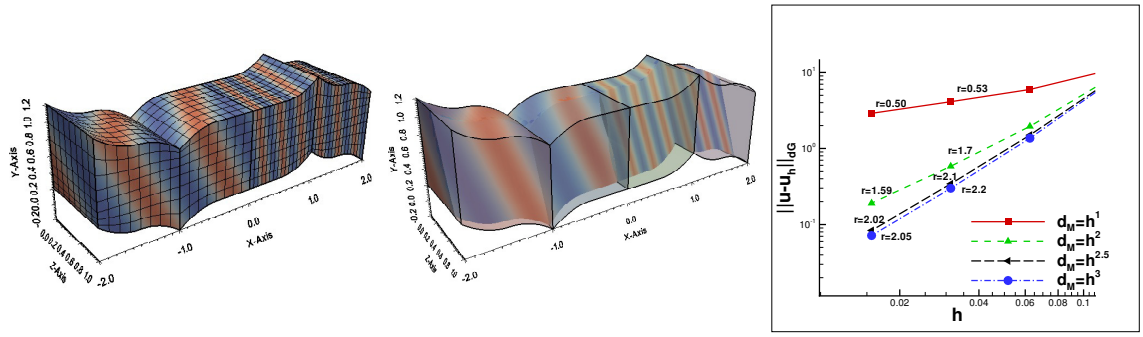


Figure 4.11: Example 5: (a) The contours of u computed on Ω. (b) The contours of u_h computed on Ω with $d_g = 0.025$. (c) Convergence rates r for the different d_g sizes.

Chapter 5

Parallelization of IETI-DP Methods

In this chapter, we present our approach to the parallelization of cG-IETI-DP and dG-IETI-DP methods. The parallelization of non-overlapping domain decomposition methods, like the IETI-DP methods, is usually realized with respect to the subdomains. This setting perfectly fits to the parallelization in a distributed memory framework. As investigated in [123], one can further perform additional parallelization in a shared memory setting on the individual subdomains with, e.g., OpenMP. We investigate the parallel scalability of the cG and dG IETI-DP methods as well as weak and strong scaling in two- and three-dimensional domains for different B-Spline degrees. The implemented algorithms are based on energy minimizing primal subspaces, which simplifies the parallelization of the solver part, but having more effort in the assembling phase. Its implementation is already discussed in Section 3.2. We focus on how to realize the communication by means of Message Passing Interface (MPI) and present numerical examples. Impressing scalability tests have already been presented for FETI-DP version, e.g. in [120], [121], [131], for the BDDC method in [232], [234] and for its IgA version, e.g., in [23].

Our parallelization strategy is based on a splitting of patches via increasing the multiplicity of knots at the desired interfaces, i.e., reducing the continuity there to C^0. We propose a hybrid cG-dG version, where we first use the dG version (C^{-1} continuity) to handle material interfaces (jumping coefficients), non-matching meshes and different polynomial degrees, and as a second step using the cG version with C^0 interfaces for a further splitting of the single patches. The goal is to have at least as many patches as processors and an almost equal number of dofs on each patch. As investigated in Section 5.2.4, enforcing C^0 continuity on edges or faces via knot multiplicity p yields only a small increase in the number of dofs. This approach leads to a great flexibility in parallelization and to high efficiency in terms of the total CPU time. However, one has to keep in mind that for higher-order PDEs, the outlined procedure does not work, since higher smoothness than C^0 is required, e.g., C^1 in the case of the biharmonic equation. Nevertheless, one can circumvent C^1 continuity by dG-techniques as well, see, e.g., [169].

In Section 5.1 we give a short outline how the parallelization is realized, while numerical examples are presented in Section 5.2. We investigate weak and strong scaling for two- and three-dimensional domains and different B-Spline degrees.

5.1 Parallelization of the Building Blocks

Here we investigate how the single operations can be executed in parallel in a distributed memory setting. The parallelization of the method is performed with respect to the patches, i.e., one or several patches are assigned to a processor. The required communication has to be understood as communication between patches, which are assigned to different processors. The majority of the used MPI methods are performed in its non-blocking version. We aim at overlapping computations with communications wherever its possible.

5.1.1 Parallel Version of PCG

We solve $F\lambda = d$ with the preconditioned CG method. This requires a parallel implementation of CG, where we follow the approach presented in Section 2.2.5.5 in [183], see also [49]. This approach is based on the concept of accumulated and distributed vectors. We say a vector $\lambda_{acc} = [\lambda_{acc}^{(q)}]$ is an *accumulated* representation of λ, if $\lambda_{acc}^{(q)}(k_q(i)) = \lambda(i)$, where i is the global index corresponding to the local index $k_q(i)$ on processor q. On the contrary, $\lambda_{dist} = [\lambda_{dist}^{(q)}]$ is a *distributed* representation of λ, if the sum of all processor local contributions give the global vector, i.e., $\lambda_{dist}(i) = \sum_q \lambda_{dist}^{(q)}(k_q(i))$. Hence, each processor only holds the part of λ, which belongs to its patches, either in a distributed or accumulated description. The Lagrange multipliers and the search directions of the CG are represented in the accumulated setting, whereas the residuals are given in the distributed representation. In order to achieve the accumulated representation, information exchange between the neighbours of a patch is required. This is done after applying the matrix and the preconditioner, and implemented via `MPI_Send` and `MPI_Recv` operations.

The last aspect in the parallel CG implementation is the realization of scalar products. Given a distributed representation u_{dist} of u and an accumulated representation of v_{acc} of v, the scalar product $(u, v)_{\ell^2}$ is then given by $(u, v)_{\ell^2} = \sum_q (u_{dist}^{(q)}, v_{acc}^{(q)})_{\ell^2}$, i.e., first the local scalar products are formed, globally added, and distributed with `MPI_Allreduce`.

5.1.2 Assembling

The assembling routine of the IETI-DP algorithm consists of the following steps:

1. Assemble the patch-local stiffness matrices and right-hand side,
2. assemble the system matrix in (3.18) and calculating its LU-factorization,
3. assemble $\boldsymbol{S}_{\Pi\Pi}$ and calculating its LU-factorization,
4. calculate the LU-factorization of $K_{II}^{(k)}$,
5. calculate the right-hand side $\{g_\Pi, g_\Delta\} = \widetilde{g} \in \widetilde{W}^*$, with $g^{(k)} = f_B^{(k)} - K_{BI}^{(k)}(K_{II}^{(k)})^{-1} f_I^{(k)}$.

Most of the tasks are completely independent of each other. Hence, they can be performed in parallel. Only the calculation of $\boldsymbol{S}_{\Pi\Pi}$ and $\widetilde{g} = \widetilde{I}^T g$ require communication, which will be explained in Section 5.1.3.

The LU-factorization of $\boldsymbol{S}_{\Pi\Pi}$ is only required at one processor, since it has to be solved only once per CG iteration. It may be advantageous to distribute this matrix to all other processors in order to reduce communication in the solver part. We refer to [131] and references therein for improving scalability based on a different approach. In the current paper, we investigate cases, where one, several and all processors hold the LU-factorization of $\boldsymbol{S}_{\Pi\Pi}$. Therefore, each processor is assigned to exactly one holder of $\boldsymbol{S}_{\Pi\Pi}$. This relation is implemented by means of an additional MPI communicator.

We note that, for extremely large scale problems with $\geq 10^5$ subdomains, one has to consider different strategies dealing with $\boldsymbol{S}_{\Pi\Pi}^{-1}$. Most commonly one uses AMG and solves $\boldsymbol{S}_{\Pi\Pi}\boldsymbol{u}_\Pi = \boldsymbol{f}_\Pi$ in an inexact way, see, e.g., [120] and [130]. When considering a moderate number of patches, i.e., $10^3 - 10^4$, the approach using the LU-factorization of $\boldsymbol{S}_{\Pi\Pi}$ is the most efficient one. In this paper, we restrict ourselves to this case.

The patch-local matrix $\boldsymbol{S}_{\Pi\Pi}^{(k)}$ is obtained as a part of the solutions of (3.18) and the assembling of the global matrix $\boldsymbol{S}_{\Pi\Pi}$ is basically a `MPI_gatherv` operation. In the case where all processors hold $\boldsymbol{S}_{\Pi\Pi}$ we use `MPI_allgatherv`. If several processors hold the LU factorization, we just call `MPI_gatherv` on each of these processors. A different possibility would be to first assemble $\boldsymbol{S}_{\Pi\Pi}$ on one patch, distribute it to the other holders and then calculate the LU-factorization on each of the processors.

5.1.3 Solver and Preconditioner

More communication is involved in the solver part. According to Algorithm 2 and Algorithm 3 and, see also Algorithm 4 and Algorithm 5 for the corresponding dG-variant, we have to perform the following operations:

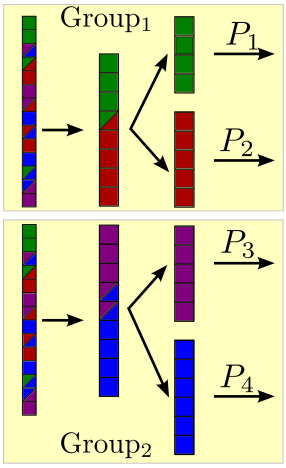
1. application of B and B^T and its scaled versions
2. application of $\widetilde{I}$ and $\widetilde{I}^T$
3. application of $\widetilde{S}^{-1}$

4. application of S^{-1}

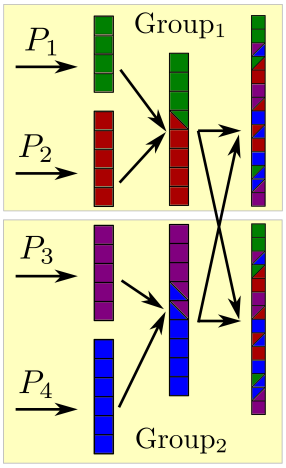
The only operations which require communication are $\widetilde{I}$ and $\widetilde{I}^T$. To be more precise, the communication is hidden in the operators $\boldsymbol{A}$ and $\boldsymbol{R}$, see Section 3.2, all other operations are block operations, where the corresponding matrices are stored locally on each processor. In principle, their implementation is given by accumulating and distributing values. The actual implementation depends on how many processors hold the coarse grid problem.

In order to implement $\widetilde{I}$, we need the distribution operation $\boldsymbol{R}$. If all processors hold $\boldsymbol{S}_{\Pi\Pi}$, this operation reduces to just extracting the right entries. Hence it is local and no communication is required. Otherwise, each holder of $\boldsymbol{S}_{\Pi\Pi}$ reorders and duplicates the entries of $\boldsymbol{w}_\Pi$ in such a way, that all entries corresponding to the patches of a single slave are in a contiguous block of memory. Then we utilize the `MPI_scatter` method to distribute only the necessary data to all slave processors. See Figure 5.1 for an illustration.

We now arrive at the implementation of $\widetilde{I}^T$. Each processor stores the values of $\boldsymbol{w}_\Pi^{(k)}$ in a vector $\widetilde{\boldsymbol{w}}_\Pi^{(k)}$ of length n_Π already in such a way, that $\sum_{k=1}^{N} \widetilde{\boldsymbol{w}}_\Pi^{(k)} = \boldsymbol{w}_\Pi$. Storing the entries in this way enables us to use the MPI reduction operations for an efficient assembly of the local contributions. If only one processor holds the coarse problem, we use the `MPI_Reduce` method to perform this operation. Similarly, if all processors hold $\boldsymbol{S}_{\Pi\Pi}$, we utilize the `MPI_Allreduce` method. If several processors have the coarse grid problem, then we use a two level approach. First, each master processor collects the local contributions from its slaves using the `MPI_Reduce` operation. In the second step, all the master processors perform an `MPI_Allreduce` operation to accumulate the contributions from each group and simultaneously distribute the result under them. This procedure is visualized in Figure 5.1.

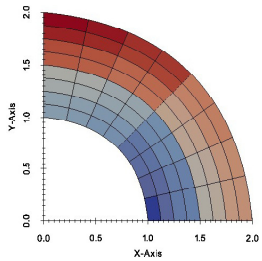


(a) Distribution operation

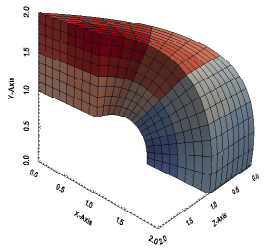


(b) Assembling operation

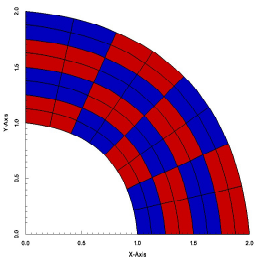
Figure 5.1: Distribution and assembling operation, illustrated for four processors, partitioned into two groups corresponding to two $\boldsymbol{S}_{\Pi\Pi}^{-1}$ holder.



(a) Representation of 2d quarter annulus with $64 = 8 \times 8$ patches.



(b) Representation of 3d twisted quarter annulus with $1024 = 8 \times 8 \times 16$ patches.



(c) Jumping diffusion coefficient pattern α, where $\{\text{blue}, \text{red}\} := \{10^{-3}, 10^{3}\}$.

Figure 5.2: Illustration of the two- and three-dimensional computational domain and jumping coefficient pattern.

5.2 Numerical Examples

We consider the model problem (2.1) on two- and three-dimensional computational domains decomposed into 1024 patches. In the two-dimensional case, we use the quarter annulus divided into 32×32 patches, illustrated as a decomposition in 8×8 patches in Figure 5.2(a). The three-dimensional domain is the twisted quarter annulus decomposed into $8 \times 8 \times 16$ patches as illustrated in Figure 5.2(b). For numerical tests on the unit-square, we refer to [89]. We note that, in the IgA framework, we cannot choose the number of patches as freely as in the finite element case since they are fixed by the geometry. Therefore, the number of 1024 patches stays constant throughout the tests. We investigate the case of having jumping diffusion coefficients across the patch interfaces and constant value inside the patch. We arrange the values $\{10^{-3}, 10^{3}\}$ of α in a checkerboard like manner, as illustrated for the two-dimensional domain in Figure 5.2(c), where, in two dimensions, always 4 patches have the same diffusion coefficient, and 8 patches in three dimensions. In all tests for the two-dimensional domain we consider the smooth right-hand side $f(x, y) = 20\pi^2 \sin(4\pi(x + 0.4)) \sin(2\pi(y + 0.3))$, whereas, in the three-dimensional case, we use $f(x, y, z) = 29\pi^2 \sin(4\pi(x + 0.4)) \sin(2\pi(y + 0.3)) \sin(3\pi(z + 0.2))$. The boundary conditions are chosen in such a way that, in case of homogeneous diffusion coefficient, the solution would be $u(x, y) = \sin(4\pi(x + 0.4)) \sin(2\pi(y + 0.3))$ and $u(x, y) = \sin(4\pi(x + 0.4)) \sin(2\pi(y + 0.3)) \sin(3\pi(z + 0.2))$, respectively. For the discretization, we use tensor B-Spline spaces V_h of different degrees p. We increase the B-Spline degree in such a way that the number of knots stay the same, i.e., the smoothness k of V_h increases. We restrict ourselves to the case of B-Splines of maximal smoothness $k = p - 1$, since they provide the smallest number of dofs for the same order of convergence. In Section 5.2.4, we analyze the effect of different continuity in the interior of the patches on the proposed method and its scalability.

The aim of this section is to investigate the scaling behaviour of the cG-IETI-DP and dG-IETI-DP method. Although, we also consider the dG variant, we restrict ourselves

to matching meshes on the patch interfaces. Otherwise, it would not be possible to compare the two methods. Moreover, having different meshes on each patch may lead to larger number of dofs on some patches, which results in load imbalances and affects the scaling in a negative way. The domain is refined in a uniform way by inserting a single knot for each dimension on each knot span. We denote by H_k the patch diameter and by h_k the characteristic mesh-size on $\Omega^{(k)}$. The set of primal variables is chosen by continuous patch vertices and interface averages for the two-dimensional setting. For the three-dimensional examples, we choose only continuous edge averages in order to keep the number of primal variables small. In Section 5.2.3, we investigate the influence of different sets of primal variables on the scalability of the method.

In [22], the authors investigate the related BDDC method while having higher continuity across the interfaces, i.e., the need of having fat interfaces, leading to an increased condition number unless the so-called deluxe scaling is used. The powerful but expensive deluxe scaling definitely reduces the condition number (and also the CG iteration) compared to the here used coefficient scaling. Also, with respect to p-dependence, the deluxe scaling provides promising results. Moreover, in the recent paper [23], a very promising combination of an adaptive selection of primal constraints with the mentioned technology is presented, eliminating the drawback of having too large coarse spaces. However, in both papers, the focus is on the scalability of the condition number and CG-iterations, considering different polynomial degrees, mesh-size and jumping diffusion coefficient. In contrast to this, here we focus on the parallel scalability, investigating the weak and strong scaling with respect to the number of used cores. For comparison, we also report on the number of CG iterations in the tables. For further numerical results regarding the scalability of the considered method with respect to H/h and p, and robustness with respect to jumps in the diffusion coefficient we refer to Section 3.4 and Section 4.3.

The PCG method is used to solve (4.27) with the scaled Dirichlet preconditioner M_{sD}^{-1}. We choose zero initial guess and a relative reduction of the residual of 10^{-8}. For solving the local systems and the coarse grid problem, a direct solver is used.

The algorithm is realized in the isogeometric open source C++ library G+SMO [162], which is based on the Eigen library [76]. We utilize the PARDISO 5.0.0 Solver Project [140] for performing the LU factorizations. The code is compiled with the `gcc 4.8.3` compiler with optimization flag `-O3`. For the communication between the processors, we use the `MPI 2` standard with the `OpenMPI 1.10.2` implementation. The results are obtained on the RADON1[1] cluster at Linz. We use 64 out of 68 available nodes, each equipped with 2x Xeon E5-2630v3 "Haswell" CPU (8 Cores, 2.4Ghz, 20MB Cache) and 128 GB RAM. This gives the total number of 1024 available cores.

We investigate two quantities, the assembling phase and the solving phase. In the assembling phase, we account for the time used for

[1] https://www.ricam.oeaw.ac.at/hpc/

- assembling the local matrices and right-hand sides,
- LU-factorization of K_{II},
- LU-factorization of $\begin{bmatrix} K & C^T \\ C & 0 \end{bmatrix}$,
- calculation of $\widetilde{\Phi}$ and $\widetilde{\mu}$,
- assembling the coarse grid matrix $\boldsymbol{S}_{\Pi\Pi}$ and calculation of its LU factorization.

As already indicated in Section 5.1, $\boldsymbol{S}_{\Pi\Pi}$ is only assembled on certain processors. The solving phase consists of the CG algorithm for (4.27) and the back-substitution to obtain the solution from the Lagrange multipliers. The main ingredients are

- application of F,
- application of M_{sD}^{-1}.

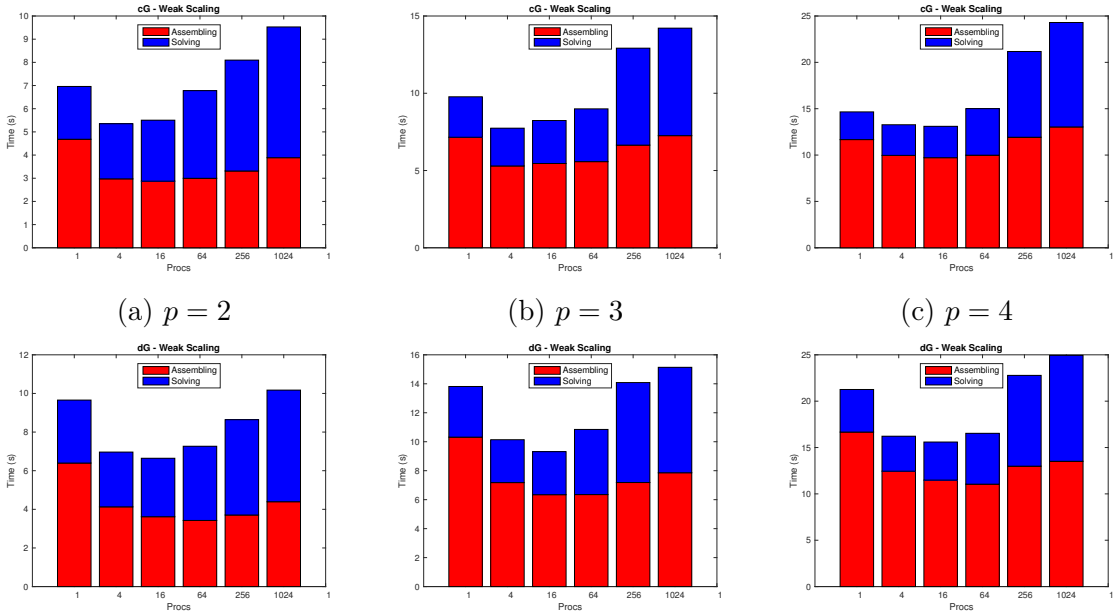
In Section 5.2.1 and Section 5.2.2, we study the weak and strong scaling behaviour for the cG-IETI-DP and the dG-IETI-DP method. In this two sections, we assume that only one processor holds the coarse grid matrix $\boldsymbol{S}_{\Pi\Pi}$. The comparison of having a different number of $\boldsymbol{S}_{\Pi\Pi}$ holders is done in Section 5.2.5.

5.2.1 Weak Scaling

In this section, we investigate the weak scaling behaviour, i.e., the relation of problem size and number of processors is constant. In each refinement step we multiply the number of used cores by $2^d, d \in \{2, 3\}$. The ideal behaviour would be a constant time for each refinement.

First, we consider the two-dimensional case. We apply three initial refinements and start with a single processor and perform up to additional 5 refinements with maximum 1024 processors. We choose as primal variables continuous vertex values and edge averages. The results for degree $p \in \{2, 3, 4\}$ are illustrated in Figure 5.3. The first row of figures corresponds to the cG-IETI-DP method, while the second one corresponds to the dG-IETI-DP method. The left column of Table 5.1 summarizes timings and the speedup for the cG-IETI-DP method, whereas the right column presents the results for the dG-IETI-DP method. For each method, we investigate the weak scaling for the assembling (red) and solution (blue) phase. As in Figure 5.3, we present the scaling and timings for $p \in \{2, 3, 4\}$.

We observe that the time used for the assembling stays almost constant, hence shows quite optimal behaviour. However, the time for solving the system increases when refining and increasing the number of used processors. In the cG case, there is already a significant increase of solving time when increasing the number of processors from 16 to 64 of a factor around 1.5. Up to 1024 cores, one observes a steady increase



(a) $p = 2$ (b) $p = 3$ (c) $p = 4$

(d) $p = 2$ (e) $p = 3$ (f) $p = 4$

Figure 5.3: Weak scaling of the cG-IETI-DP (first row) and dG-IETI-DP (second row) method for B-Spline degrees $p \in \{2, 3, 4\}$ in two dimensions. Each degree corresponds to one column.

by 1-3 seconds each refinement. In the dG case, one can consider the increase in the solving time as acceptable up to 64 cores, then one also observes an increase by a constant factor at each refinement. Overall, one can say, that the solving time doubles or triples, when going from 1 to 1024 processors. This is due to the quasi optimal condition number bound with respect to H/h of the IETI-DP type methods, i.e., the number of CG iterations increases logarithmically with decreasing h and fixed H, cf. Theorem 3.20. Secondly, as already pointed out in Section 5.1, the solving phase consists of more communication between processors, which cannot be completely overlapped with computations. Moreover, one also has to take in account global synchronization points in the conjugate gradient method.

Next, we consider the weak scaling for the three-dimensional case. As already indicated in the introduction of this section, we choose only continuous edge averages as primal variables. We perform the tests in the same way as for the two-dimensional case. However, we already start with two processors and perform two initial refinements. Multiplying the number of used processors by 8 with each refinement, we end up again with 1024 processors on the finest grid. The two algorithms behave similar to the two-dimensional case, but show an even stronger increase in the solving time. Especially, the step from 128 to 1024 cores shows a strong increase for both the assembling and solving time. However for both the two- and three-dimensional case, the assembling and solving time is either well balanced or the assembling part is the dominant factor,

	cG-IETI-DP		$p = 2$			dG-IETI-DP		$p = 2$		
# procs	#dofs	Iter.	Ass. Time	Solv. Time	Total Time	#dofs	Iter.	Ass. Time	Solv. Time	Total Time
1	98241	10	4.6	2.2	6.9	132868	11	6.4	3.2	9.6
4	326593	11	2.9	2.3	5.3	392964	11	4.1	2.8	6.9
16	1176513	13	2.8	2.6	5.5	1306372	13	3.6	3.0	6.6
64	4449217	14	3.0	3.7	6.7	4706052	14	3.4	3.8	7.2
256	17286081	15	3.3	4.7	8.1	17796868	15	3.7	4.9	8.6
1024	68125633	16	3.8	5.6	9.5	69144324	16	4.3	5.7	10.17
	cG-IETI-DP		$p = 3$			dG-IETI-DP		$p = 3$		
# procs	#dofs	Iter.	Ass. Time	Solv. Time	Total Time	#dofs	Iter.	Ass. Time	Solv. Time	Total Time
1	119617	11	6.8	2.9	9.8	158212	11	9.6	3.8	13.5
4	364353	12	5.1	3.2	8.4	434692	12	7.0	3.8	10.8
16	1247041	13	5.3	3.4	8.8	1380868	13	6.3	3.6	10.0
64	4585281	15	5.4	5.9	11.3	4846084	15	6.1	6.3	12.5
256	17553217	16	6.1	7.5	13.7	18067972	16	6.7	7.7	14.4
1024	68654913	17	7.3	9.5	16.9	69677572	17	7.8	9.7	17.6
	cG-IETI-DP		$p = 4$			dG-IETI-DP		$p = 4$		
# procs	#dofs	Iter.	Ass. Time	Solv. Time	Total Time	#dofs	Iter.	Ass. Time	Solv. Time	Total Time
1	143041	12	10.8	3.6	14.5	185604	12	15.4	4.9	20.3
4	404161	13	9.3	4.7	14.0	478468	13	11.8	4.7	16.5
16	1319617	14	9.2	4.4	13.7	1457412	14	11.0	5.1	16.1
64	4723393	15	9.2	8.1	17.4	4988164	15	10.4	8.3	18.8
256	17822401	16	10.7	11.0	21.8	18341124	17	11.1	10.4	21.5
1024	69186241	17	12.6	14.4	27.0	70212868	17	13.1	14.6	27.7

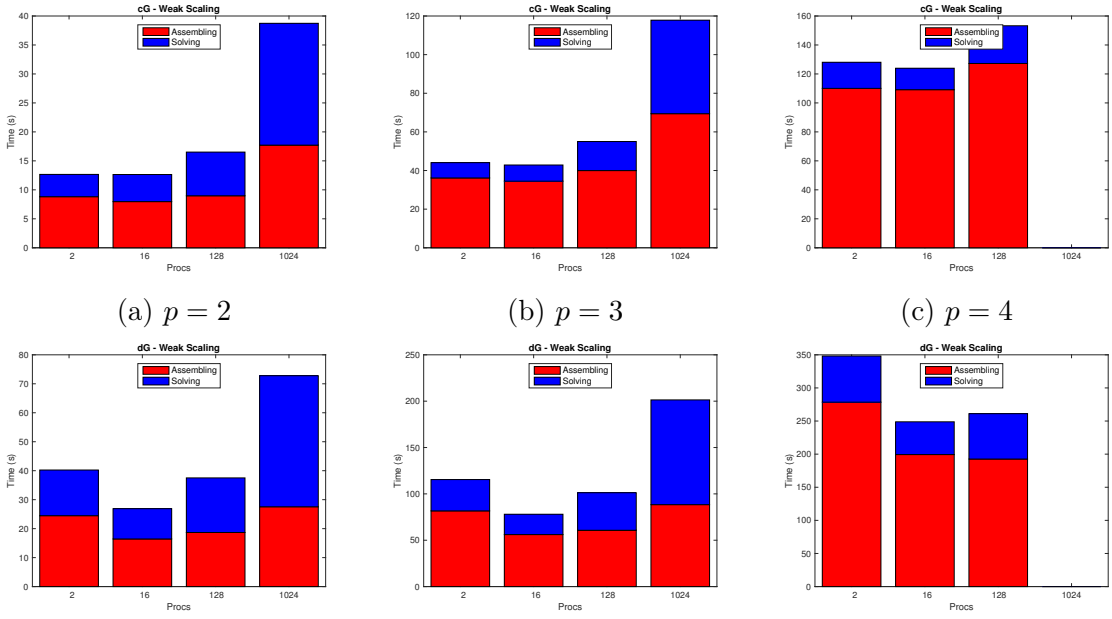
Table 5.1: Weak scaling results for the two-dimensional testcase for the cG and dG IETI-DP method. Left column contains results for the cG variant and the right column for the dG version. Each row corresponds to a fixed B-Spline degree $p \in \{2, 3, 4\}$

which gives an overall acceptable weak scaling. The results are visualized in Figure 5.4 and summarized in Table 5.2. Note, for both methods, using $p = 4$, we exceeded the memory capacity of the cluster.

5.2.2 Strong Scaling

Secondly, we are investigating the strong scaling behaviour. Now we fix the problem size and increase the number of processors. In the optimal case, the time used for a certain operation reduces in the same way as the number of used processors increases. We use the same primal variables for the strong scaling studies as in the weak scaling studies in Section 5.2.1. Note, we use 1024 patches in all tests. Therefore, the number of iterations remains constant when increasing the number of processors.

Again as in Section 5.2.1, we begin with the two-dimensional example. We perform 7



(a) $p = 2$ (b) $p = 3$ (c) $p = 4$

(d) $p = 2$ (e) $p = 3$ (f) $p = 4$

Figure 5.4: Weak scaling of the cG-IETI-DP (first row) and dG-IETI-DP (second row) method for B-Spline degrees $p \in \{2, 3, 4\}$ in three dimensions. Each degree corresponds to one column. No timings are obtained in the case of 1024 cores in (c) and (f) due to memory limitations.

initials refinements and end up with 17 Mio. dofs on 1024 patches. We start already with 4 processors in the initial case and do 8 refinements until we reach 1024 cores. Similar to Section 5.2.1, the results for $p \in \{2, 3, 4\}$ are illustrated in Figure 5.5 and summarized in Table 5.3.

We observe that the assembling phase has a quite good scaling performance, as already observed for the weak scaling results in Section 5.2.1. Moreover, the higher the B-Spline degree, the better the parallel performance behaves. This holds due to increased computational costs for the parallel part. Similar to the weak scaling results, the solving phase does not provide such an excellent scaling as the assembling phase. Still, we obtain a scaling of around 700 when using 1024 processors. We note that the degree of the B-Splines does not seem to have such a significant effect on the scaling for both solving phase and the assembling phase.

In the three-dimensional example we perform four initial refinements and obtain around 5 Mio. dofs. The presentation of the results is done in the same way as in the previous examples, see Figure 5.6 and Table 5.4. For higher B-Spline degrees, the methods also provide an excellent scaling behaviour for the assembling phase. For the case $p = 2$ in both cG and dG case, the parallel part of the algorithm seems to be too small for better scalability. In comparison with the 2d version, the solving phase does not scale so well any more. We only obtain a speedup of around 500 using 1024

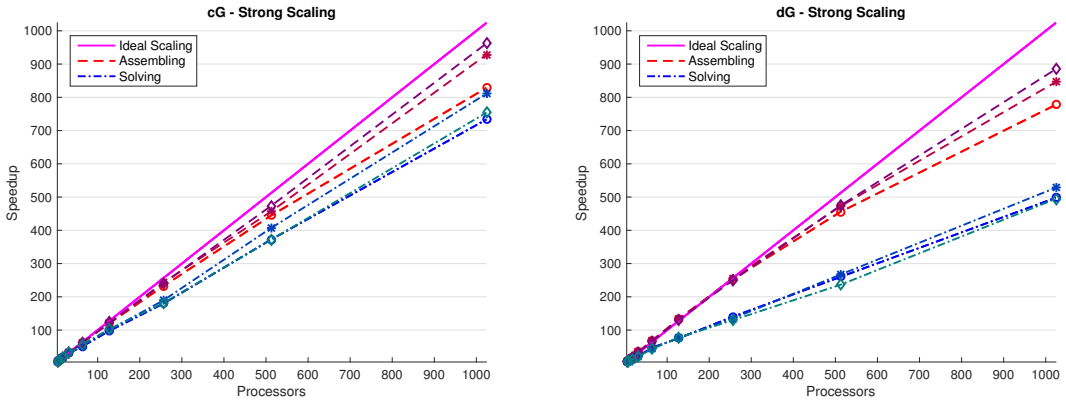
	cG-IETI-DP		$p = 2$			dG-IETI-DP		$p = 2$		
# procs	#dofs	Iter.	Ass. Time	Solv. Time	Total Time	#dofs	Iter.	Ass. Time	Solv. Time	Total Time
2	198904	15	8.9	3.8	12.7	383456	26	24.4	15.7	40.2
16	961272	16	8.1	3.4	11.5	1488864	28	16.4	10.4	26.8
128	5766904	18	9.9	6.8	16.7	7508960	30	18.6	18.8	37.5
1024	39511800	21	18.8	21.4	40.2	45796320	32	27.5	45.2	72.7
	cG-IETI-DP		$p = 3$			dG-IETI-DP		$p = 3$		
# procs	#dofs	Iter.	Ass. Time	Solv. Time	Total Time	#dofs	Iter.	Ass. Time	Solv. Time	Total Time
2	320760	16	31.2	8.4	39.6	574560	30	81.6	33.7	115.4
16	1286904	18	30.1	7.2	37.3	1927776	32	56.1	21.9	78.0
128	6795000	20	36.8	14.5	51.3	8738400	34	60.7	40.5	101.2
1024	43124472	22	63.0	47.8	110.8	49786464	35	88.4	112.9	201.3
	cG-IETI-DP		$p = 4$			dG-IETI-DP		$p = 4$		
# procs	#dofs	Iter.	Ass. Time	Solv. Time	Total Time	#dofs	Iter.	Ass. Time	Solv. Time	Total Time
2	484344	18	110.0	18.0	128.0	818400	33	278.4	69.7	348.2
16	1678840	19	109.1	14.8	123.9	2444000	35	199.3	49.4	248.8
128	7938552	21	127.2	26.0	153.2	10094304	37	192.6	68.5	261.1
1024	∼48000000	x	x	x	x	∼53000000	x	x	x	x

Table 5.2: Weak scaling results for the three-dimensional testcase for the cG and dG IETI-DP method. Left column contains results for the cG variant and the right column for the dG version. Each row corresponds to a fixed B-Spline degree $p \in \{2, 3, 4\}$. No timings are available for both methods with $p = 4$ on 1024 cores due to memory limitations.

cores. Comparing the cG with the dG version, the latter one has a slightly worse performance. Having a closer look at the timings, we observe that the different scaling of the two methods originates from small load imbalances in the interior domains due to the additional layer of dofs and the larger number of primal variables. The latter one leads to an increased time in solving (3.18) due to a larger number of right-hand sides on the interior patches. One can further optimize the three-dimensional case by considering different strategies for the primal variables, where one aims for smaller and more equally distributed numbers of primal variables.

5.2.3 Influence of the Primal Variables

In this section, we investigate the influence of different sets of primal variables on the strong scalability. This is especially important for three-dimensional problems, because the number of primal variables may grow rapidly with the number of patches. A common choice for primal variables in three dimensions are combinations of vertex values $\mathcal{V}$, edge averages $\mathcal{E}$ and face averages $\mathcal{F}$. We perform tests for the cG-IETI-DP and dG-IETI-DP method on the twisted quarter annulus as in Fig. 5.2(b) having a



(a) cG-IETI-DP (b) dG-IETI-DP

Figure 5.5: Strong scaling of the cG-IETI-DP (left column) and dG-IETI-DP (right column) method for B-Spline degrees $p \in \{2, 3, 4\}$ in two dimensions. The markers $\{\circ, *, \diamond\}$ as well as different shades of red (assembling phase) and blue (solving phase) correspond to the degrees $\{2, 3, 4\}$.

decomposition in 1024 patches , B-Spline degree $p = 3$ and a homogeneous diffusion coefficient. In addition to the timings for the assembling and solution phase, we also report on the number of primal variables, the number of CG-iterations and the obtained speedup for 1024 processors. The results are listed in Table 5.6. For illustration we summarized the number of primal variables for different settings and with increasing number of patches in case of the twisted quarter annulus, see Table 5.5.

The results validate the well known fact that for three-dimensional problems using only vertex values as primal variables leads to a large number of iterations, see also Section 3.4 and Section 4.3 for numerical experiments on the scalability of H/h. The choice $\mathcal{V}+\mathcal{E}+\mathcal{F}$ of primal variables results in a large set of primal variables, especially in the dG setting, but gives the smallest number of CG-iterations. Hence, one has to find a balance between a rich set of primal variables and efficiency of the method. Overall, for our purposes, the choice of having only edge averages $\mathcal{E}$ gives the most appropriate and efficient setting, i.e. having the smallest total computation time. Concerning parallel scalability on a fixed number of domains, all four methods give satisfactory results, although only vertex values as primal variables are practically useless for three-dimensional domains, since the number of CG iterations does not depend in a quasi-optimal way on H/h.

5.2.4 Effect of Continuity in the Interior of the Patches

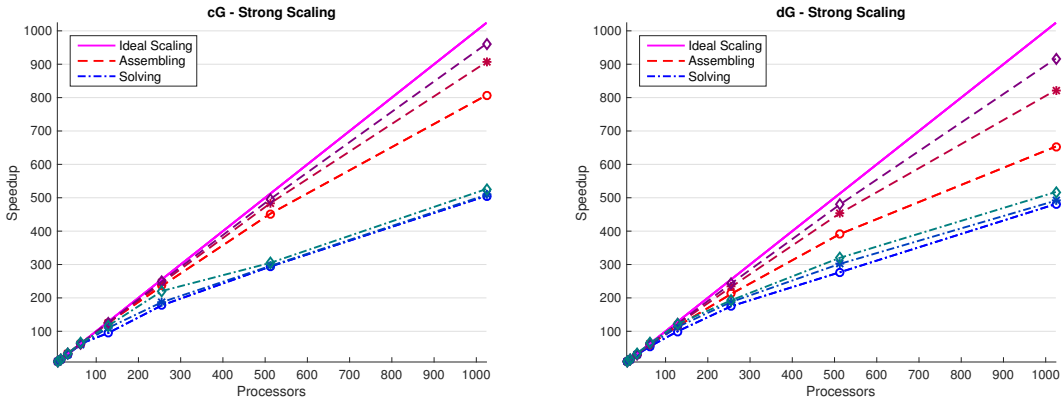
This section deals with the influence of different degrees of smoothness of the B-Splines in the interior of the patches on the parallel scalability. We only investigate the two-dimensional case on the quarter annulus illustrated in Fig.5.2(a) on a decomposition

2d	$p=2$				$p=3$				$p=4$			
cG-IETI-DP	assembling phase		solving phase		assembling phase		solving phase		assembling phase		solving phase	
# procs	Time	Sp.	Time	Sp.	Time	Sp.	Time	Sp.	Time	Sp.	Time	Sp.
4	215.2	4	237.8	4	408.8	4	401.2	4	719.5	4	529.4	4
8	107.2	8	118.8	8	204.3	8	200.5	8	359.7	8	264.6	8
16	53.5	16	59.2	16	101.8	16	100.7	15	179.3	16	132.6	15
32	26.7	32	30.6	31	50.8	32	51.2	31	89.8	32	67.2	31
64	13.6	63	18.5	51	26.2	62	29.7	53	46.0	62	36.9	57
128	7.1	121	9.7	97	13.4	121	15.9	100	23.3	123	20.4	103
256	3.7	232	5.3	179	6.7	242	8.4	189	11.9	240	11.7	180
512	1.9	444	2.5	372	3.5	456	3.9	406	6.0	473	5.7	371
1024	1.0	828	1.3	733	1.7	928	1.9	811	2.9	962	2.8	753
Iter.	15				16				16			
dG-IETI-DP	assembling phase		solving phase		assembling phase		solving phase		assembling phase		solving phase	
# procs	Time	Sp.	Time	Sp.	Time	Sp.	Time	Sp.	Time	Sp.	Time	Sp.
4	244.7	4	244.5	4	452.1	4	407.3	4	783.8	4	576.7	4
8	122.3	8	122.2	8	226.0	8	203.6	8	391.8	8	288.3	8
16	60.5	16	61.6	15	112.7	16	102.8	15	196.8	15	146.9	15
32	30.1	32	31.0	31	56.0	32	52.3	31	98.3	31	74.3	31
64	15.3	63	19.4	50	28.9	62	29.6	55	50.0	62	41.7	55
128	7.9	123	10.3	94	14.6	123	15.9	102	25.0	125	22.5	102
256	4.0	239	5.5	176	7.4	242	9.0	179	13.0	240	12.4	186
512	2.1	454	2.6	364	3.8	471	4.1	397	6.5	481	6.1	376
1024	1.1	847	1.4	671	1.8	955	2.1	744	3.2	962	3.0	748
Iter.	15				16				17			

Table 5.3: Strong scaling results: Time (s) and Speedup for $p \in \{2, 3, 4\}$ in two dimensions having approximately 17 Mio. dofs. First row shows results for the cG variant of the IETI-DP method, whereas the second row contains results for the dG version. Each column corresponds to a degree p.

in 1024 patches. We use a homogeneous diffusion coefficient, edge averages and vertex values as primal variables, and we fix the B-Spline degree $p = 4$. In contrast to the other sections, where we used B-Spline with maximal smoothness $k = p - 1$, we vary the smoothness of the B-Splines in the interior of the patches. We want to note that, in case of the cG-IETI-DP method, we still have C^0 smoothness across the interfaces and discontinuous B-Spline spaces in the dG version. The results are presented in Table 5.7.

We observe that the number of IETI-DP iterations is robust with respect to the smoothness. When increasing the multiplicity, the time spent in the assembling phase is significantly increasing. This origins from the fact that the local LU factorizations require more time due to the larger system size, whereas the assembling of the local matrices stays approximately constant. However, the total number of dofs increases drastically, when reducing the smoothness of the B-Splines in the interior of



(a) cG-IETI-DP (b) dG-IETI-DP

Figure 5.6: Strong scaling of the cG-IETI-DP (left column) and dG-IETI-DP (right column) method for B-Spline degrees $p \in \{2, 3, 4\}$ in three dimensions. The markers $\{\circ, *, \diamond\}$ as well as different shades of red (assembling phase) and blue (solving phase) correspond to the degrees $\{2, 3, 4\}$.

the patches. Moreover, no additional order of approximation in terms of powers of h is obtained by reducing the continuity. On the other hand, the increased computational load per patch has a positive influence on the parallel scalability.

In contrast to the experiments above, performing parallelization by subdividing patches via reducing the continuity at certain knots to C^0 can still be recommended. In Table 5.8, for comparison, we included the total number of dofs on a domain with smooth splines (here $p = 3$ and $k = 2$), and consequently enforcing C^0 continuity at certain knots to subdivide the patch into subpatches with C^0 continuity across the sub-patch interfaces. Clearly, this increases the total number of dofs, but if the interior of a single patch is still sufficiently large, the total number of dofs does not explode. Moreover, comparing the effort to be taken to handle smooth spline spaces over interfaces via the use of fat interfaces and the required expensive preconditioner, using C^0 will pay off in many cases.

5.2.5 Study on the Number of $\boldsymbol{S}_{\Pi\Pi}^{-1}$ Holders

In this last section of the numerical experiments, we want to investigate the influence of the number of holders of $\boldsymbol{S}_{\Pi\Pi}^{-1}$ on the scaling behaviour. As already indicated in Section 5.1.3, if more processors hold the LU-factorization of the coarse grid matrix, it is possible to decrease the communication effort after applying $\boldsymbol{S}_{\Pi\Pi}^{-1}$, while having more communication before the application. The advantage of this strategy is to be able to have a better overlap of communication with computations. However, one has to take into account that this also increases the communication in the assembling phase, since the local contribution $\boldsymbol{S}_{\Pi\Pi}^{(k)}$ has to be sent to all the master processors.

3d	$p=2$				$p=3$				$p=4$			
cG-IETI-DP	assembling phase		solving phase		assembling phase		solving phase		assembling phase		solving phase	
# procs	Time	Sp.	Time	Sp.	Time	Sp.	Time	Sp.	Time	Sp.	Time	Sp.
8	156.5	8	92.4	8	592.3	8	205.7	8	2028.0	8	372.8	8
16	77.2	16	46.5	15	296.3	15	102.4	16	1014.7	15	186.8	15
32	38.4	32	23.7	31	146.6	32	52.3	31	502.4	32	95.2	31
64	19.4	64	13.9	53	73.4	64	35.6	46	249.7	64	59.2	50
128	10.5	119	7.2	102	37.6	125	15.8	104	127.0	127	28.5	104
256	5.6	221	4.4	167	19.2	245	9.3	177	64.2	252	18.1	164
512	3.1	396	2.2	332	10.1	468	5.0	324	32.8	494	9.6	308
1024	1.9	638	1.3	537	5.5	861	3.0	535	17.1	947	5.7	518
Iter.	18				20				21			
dG-IETI-DP	assembling phase		solving phase		assembling phase		solving phase		assembling phase		solving phase	
# procs	Time	Sp.	Time	Sp.	Time	Sp.	Time	Sp.	Time	Sp.	Time	Sp.
8	250.4	8	209.8	8	888.1	8	451.4	8	2906.8	8	855.4	8
16	125.6	15	104.5	16	443.5	16	226.3	15	1447.9	16	423.1	16
32	64.9	30	56.0	29	223.4	31	121.3	29	733.0	31	215.8	31
64	34.3	58	33.7	49	115.1	61	65.6	55	378.3	61	114.2	59
128	19.1	104	20.6	81	62.1	114	40.9	88	195.7	118	78.3	87
256	10.6	188	11.5	145	32.3	219	26.3	137	99.8	232	53.0	129
512	5.7	347	5.8	285	17.0	416	13.8	261	52.5	442	26.3	260
1024	3.5	572	3.5	476	9.1	773	7.3	488	26.8	867	13.9	491
Iter.	30				34				37			

Table 5.4: Strong scaling results: Time (s) and Speedup for $p \in \{2,3,4\}$ in three dimensions having approximately 5 Mio. dofs. First row shows results for the cG variant of the IETI-DP method, whereas the second row contains results for the dG version. Each column corresponds to a degree p.

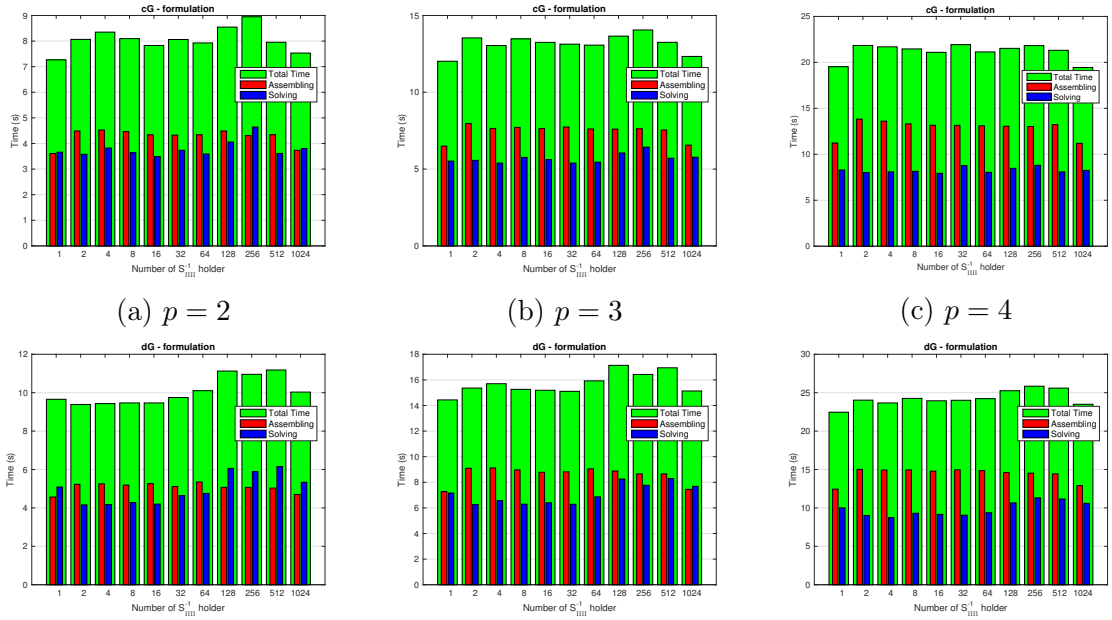
We only consider the two-dimensional domain, where we perform 7 initials refinements, but on a decomposition with 4096 subdomains and end up with around 70 Mio. dofs. This gives a comparable setting as in Section 5.2.1 having the most refined domain. In order to better observe the influence of the number of $\boldsymbol{S}_{\Pi\Pi}^{-1}$ holders, we increase the number of subdomains, leading to a larger coarse grid problem. We only investigate the case of using 1024 processors and the number of $\boldsymbol{S}_{\Pi\Pi}$ holders given by $2^n, n \in \{0,1,\ldots,10\}$. Hence, we obtain the number of master processors ranging from 1 to 1024, such that each master has the same number of slaves. The results are summarized in Figure 5.7 and Table 5.9.

We observe that choosing several holders of the coarse grid problem in the cG version does not really have a significant effect. However, in the dG version, due to an increased number of primal variables, the use of several holders actually increases the performance of the solver by around 10%. Nevertheless, what is gained in the solving part does not pay off with the additional effort in the assembling phase. Considering the total computation time in Table 5.9, the best options is still either using only a

	cG-IETI-DP				dG-IETI-DP			
#patches	$\mathcal{V}$	$\mathcal{V}+\mathcal{E}+\mathcal{F}$	$\mathcal{V}+\mathcal{E}$	$\mathcal{E}$	$\mathcal{V}$	$\mathcal{V}+\mathcal{E}+\mathcal{F}$	$\mathcal{V}+\mathcal{E}$	$\mathcal{E}$
$2 = 1 \times 1 \times 2$	4	9	8	4	8	17	16	8
$16 = 2 \times 2 \times 4$	37	129	101	64	120	308	280	160
$128 = 4 \times 4 \times 8$	217	1017	713	496	1016	2792	2488	1472
$1024 = 8 \times 8 \times 16$	1369	7737	4985	3616	8184	23096	20344	12160

Table 5.5: Number of primal variables for different configurations and increasing number of patches in absence of eliminated Dirichlet boundaries.

single coarse grid problem on one processor or making a redundant factorization on each processor.



(a) $p = 2$ (b) $p = 3$ (c) $p = 4$

(d) $p = 2$ (e) $p = 3$ (f) $p = 4$

Figure 5.7: Influence of the number of $\boldsymbol{S}_{\Pi\Pi}^{-1}$ holders on the scaling. First row corresponds to cG-IETI-DP, second row to dG-IETI-DP. Each column has a fixed degree $p \in \{2, 3, 4\}$. Figures (a-c) summarizes the cG version and Figures (d-f) the dG version, respectively.

3d	$\mathcal{V}$		$\mathcal{V}+\mathcal{E}+\mathcal{F}$		$\mathcal{V}+\mathcal{E}$		$\mathcal{E}$	
cG-IETI-DP #procs	Ass.	Solv.	Ass.	Solv.	Ass.	Solv.	Ass.	Solv.
8	584.8	1310.7	658.3	188.2	602.0	203.8	592.6	204.9
16	292.5	654.6	329.0	93.2	302.0	102.3	296.3	102.0
32	145.0	335.3	161.8	47.9	148.6	52.8	146.7	52.5
64	72.3	219.4	80.8	30.3	73.9	32.2	73.1	31.5
128	37.0	101.2	41.9	15.0	38.0	17.2	37.3	15.5
256	18.8	59.8	21.9	8.9	19.9	9.8	19.2	9.8
512	9.8	32.0	11.3	4.6	10.1	5.0	10.0	4.9
1024	5.3	19.4	6.2	2.8	5.6	3.1	5.4	3.0
#primal dofs	735		5951		3199		2464	
Iterations	133		18		20		20	
Largest speedup Ass. / Solv.	881/538		849/523		859/525		868/531	
dG-IETI-DP #procs	Ass.	Solv.	Ass.	Solv.	Ass.	Solv.	Ass.	Solv.
8	843.8	1599.3	1058.4	367.2	952.4	346.3	884.0	376.9
16	422.7	806.9	530.2	179.6	473.1	171.7	441.2	190.9
32	213.0	420.2	274.9	95.1	245.3	90.8	221.8	98.3
64	108.6	221.5	142.5	53.3	126.2	49.0	114.8	55.2
128	57.8	144.8	76.4	35.1	68.5	32.5	60.9	34.6
256	29.9	84.9	40.7	22.2	36.5	19.8	32.0	21.3
512	15.4	40.2	21.8	10.6	19.6	10.1	16.6	10.3
1024	8.3	24.8	12.4	7.1	11.0	6.4	9.0	6.1
#primal dofs	5880		18488		15736		9856	
Iterations	124		26		25		28	
Largest speedup Ass. / Solv.	809/515		679/413		691/430		784/490	

Table 5.6: Strong scaling results for different set of primal variables: Time (s) for fixed degree $p = 3$ in three dimensions having approximately 7 Mio. dofs. First row shows results for the cG variant of the IETI-DP method, whereas the second row contains results for the dG version. Each column corresponds a different set of primal variables. Number of primal variables, number of iterations and speedup of the assembling and solution phase is summarized for each set of primal variables.

2d	C^3		C^2		C^1		C^0	
cG-IETI-DP #procs	Ass.	Solv.	Ass.	Solv.	Ass.	Solv.	Ass.	Solv.
4	160.8	84.1	375.4	340.9	494.8	652.1	507.3	616.1
8	79.6	41.6	187.9	171.1	246.4	332.4	252.4	307.4
16	40.8	21.3	94.2	86.1	123.7	162.4	126.3	153.5
32	19.7	10.7	46.8	44.3	61.7	83.1	63.4	77.5
64	10.3	7.3	24.2	25.2	31.8	44.9	32.2	43.8
128	5.2	3.9	11.8	14.1	16.2	25.1	16.4	24.2
256	2.6	2.1	6.3	7.4	8.3	13.4	8.5	12.3
512	1.3	1.2	3.3	3.6	4.4	6.1	4.7	5.7
1024	0.6	0.7	1.6	1.8	2.0	3.1	2.3	3.0
#dofs	4723393		17553217		38511553		67598401	
Iterations	15		15		16		16	
Largest speedup Ass. / Solv.	959/487		923/745		950/823		872/822	
dG-IETI-DP #procs	Ass.	Solv.	Ass.	Solv.	Ass.	Solv.	Ass.	Solv.
4	181.3	91.1	416.3	354.4	561.7	668.8	586.2	632.4
8	90.6	45.5	208.1	177.1	280.8	334.4	293.1	316.2
16	45.1	22.9	102.3	88.2	139.5	165.9	146.5	156.8
32	22.5	11.6	50.9	45.0	69.4	85.1	72.8	79.0
64	11.4	7.2	26.3	25.8	35.0	46.3	36.5	46.4
128	5.8	4.4	13.3	14.2	18.1	25.4	18.7	29.4
256	2.9	2.4	6.9	7.8	9.1	12.9	9.8	12.8
512	1.5	1.2	3.6	3.8	4.8	6.2	5.3	5.8
1024	0.8	0.7	1.8	1.9	2.4	3.2	2.7	3.1
# dofs	4988164		18067972		39276292		68613124	
Iterations	15		15		15		16	
Largest speedup Ass. / Solv.	862/463		917/728		925/820		842/816	

Table 5.7: Strong scaling results for different smoothness k in the interior of the patches: Time (s) for fixed degree $p = 4$ in two dimensions. On the interface we either have C^0 continuity or discontinuous B-Splines. First row shows results for the cG variant of the IETI-DP method, whereas the second row contains results for the dG version. Each column corresponds a to different smoothness of the splines. The total number of dofs, number of iterations and speedup of the assembling and solution phase is summarized for each considered spline smoothness.

	2d			3d	
#patches	cG-IETI-DP	dG-IETI-DP	#patches	cG-IETI-DP	dG-IETI-DP
$1 = 1 \times 1$	1054729	1054729	$2 = 1 \times 1 \times 2$	4496178	4530496
$4 = 2 \times 2$	1060893	1065004	$16 = 2 \times 2 \times 4$	4812125	5063368
$16 = 4 \times 4$	1073257	1085632	$128 = 4 \times 4 \times 8$	5487201	6230368
$64 = 8 \times 8$	1098129	1127200	$1024 = 8 \times 8 \times 16$	7016801	8988544
$256 = 16 \times 16$	1148449	1211584			
$1024 = 64 \times 64$	1251393	1385344			

Table 5.8: Total number of dofs for fixed mesh-size h, but different decomposition of the geometry, i.e., h-refinement and patch-refinement sums up to 10 (2d) and 6 (3d) refinement steps. The polynomial degree is fixed with $p = 3$ and no elimination of Dirichlet dofs is considered. At the interface we consider either C^0 continuous (cG-IETI-DP) or discontinuous B-Spline spaces (dG-IETI-DP).

cG	$p = 2$			$p = 3$			$p = 4$		
# $\boldsymbol{S}_{\Pi\Pi}^{-1}$ Holder	Assemble Time	Solving Time	Total Time	Assemble Time	Solving Time	Total Time	Assemble Time	Solving Time	Total Time
1	3.61	3.66	7.27	6.50	5.52	12.02	11.23	8.30	19.53
2	4.49	3.58	8.07	7.97	5.57	13.54	13.83	8.02	21.85
4	4.53	3.82	8.35	7.65	5.40	13.05	13.60	8.09	21.69
8	4.46	3.63	8.09	7.72	5.76	13.48	13.32	8.15	21.47
16	4.34	3.49	7.83	7.64	5.61	13.25	13.16	7.93	21.09
32	4.33	3.73	8.06	7.74	5.39	13.13	13.15	8.78	21.93
64	4.34	3.59	7.93	7.62	5.45	13.07	13.10	8.04	21.14
128	4.49	4.06	8.55	7.60	6.05	13.65	13.06	8.47	21.53
256	4.31	4.64	8.95	7.63	6.43	14.06	13.02	8.81	21.83
512	4.34	3.61	7.95	7.55	5.71	13.26	13.23	8.09	21.32
1024	3.73	3.80	7.53	6.56	5.77	12.33	11.19	8.26	19.45
dG	$p = 2$			$p = 3$			$p = 4$		
# $\boldsymbol{S}_{\Pi\Pi}^{-1}$ Holder	Assemble Time	Solving Time	Total Time	Assemble Time	Solving Time	Total Time	Assemble Time	Solving Time	Total Time
1	4.57	5.09	9.66	7.28	7.16	14.44	12.44	10.01	22.45
2	5.23	4.16	9.39	9.10	6.26	15.36	15.02	9.01	24.03
4	5.25	4.18	9.43	9.12	6.58	15.70	14.93	8.73	23.66
8	5.19	4.28	9.47	8.97	6.29	15.26	14.95	9.30	24.25
16	5.26	4.20	9.46	8.78	6.41	15.19	14.79	9.16	23.95
32	5.11	4.64	9.75	8.82	6.29	15.11	14.96	9.05	24.01
64	5.35	4.75	10.1	9.06	6.87	15.93	14.85	9.37	24.22
128	5.07	6.06	11.13	8.88	8.25	17.13	14.61	10.65	25.26
256	5.07	5.89	10.96	8.66	7.77	16.43	14.52	11.32	25.84
512	5.03	6.15	11.18	8.66	8.29	16.95	14.43	11.16	25.59
1024	4.70	5.33	10.03	7.45	7.68	15.13	12.89	10.60	23.49

Table 5.9: Influence of the number of processors having an LU-factorization of $\boldsymbol{S}_{\Pi\Pi}$. Timings in seconds for 1024 Processors on a domain with around 70 Mio. dofs and 2048 subdomains.

Chapter 6

Inexact Variants

This chapter deals with the incorporation of inexact solvers in the IETI-DP algorithm. Due to the fact that direct solvers require a large amount of memory, especially in the case of high B-Spline degree p when the bases functions have a large support, we aim at replacing them by inexact solvers, e.g., Multigrid (MG) and Fast Diagonalization (FD) methods. Throughout the section, we only consider the continuous Galerkin case. In IgA, these methods are usually applied to discretizations on a single patch. The combination with IETI-DP is one way to extend these methods to multi-patch domains. Recently, an extension of p-robust MG to multi-patch domains using additive Schwarz smoothers is presented in [204]. For better readability, we do not use boldface letters to highlight matrices and vectors throughout this chapter.

After a short description of the MG and FD methods in Section 6.1, we explain the use of the preconditioners in the IETI-DP algorithm in Section 6.2. Finally, we present numerical experiments in Section 6.3.

6.1 Recalling of IgA Multigrid and Fast Diagonalization Methods

In this section, for completeness, we give a short overview of MG for IgA and the FD method. For a more comprehensive description of MG we refer, e.g., to [84], [220], and for applications of Multigrid and Multilevel algorithms in IgA, we refer to [34], [47] or [65]. Recent results on p-robust MG can be found in [102]. A more detailed description of FD methods and recent results can be found in [192], [168] and [211]. The origin of this methods dates back to the works in [157] and [13].

6.1.1 Multigrid

The ingredients to setup a MG algorithm are the restrictions and prolongations operators, the coarse gird solver and the smoother. Let $\{V_{h,l}\}_{l=1}^{L}$, with $V_{h,l} \subset V_{h,l+1}$, $l = 1, \ldots, L-1$ be a sequence of nested spaces , where $V_{h,1}$ is the coarsest and $V_h = V_{h,L}$ the finest space. Since the spaces are nested, there exists a natural prolongation operator $P : V_l \to V_{l+1}$. Based on it, we can define the canonical restriction operators $P^T : V_{l+1} \to V_l$ as its transposed. As coarse grid solver, we consider a sparse direct solver on the coarsest level $l = 1$. The last ingredient is the smoother S^ν, where the Gauss Seidel or damped Jacobi smoother provides a h-robust method. The same holds true for IgA. However, the condition number of the problem preconditioned by MG still depends on the B-Spline degree p, unfortunately, in an exponential way. In Algorithm 6, we summarize the MG-algorithm using ν smoothing steps. For the settings $\gamma = 1$ and $\gamma = 2$, we obtain the V- and W-cycle, respectively.

Algorithm 6 Multigrid algorithm for solving $Ku = f$

procedure MGM(K^l, u^l, f^l, l)
  if $l == 1$ **then**
    Solve $K^1 u^1 = f^1$
  else
    $\hat{u}^l = S^\nu(K^l, u^l, f^l)$
    $d^l = f^l - K^l \hat{u}^l$
    $d^{l-1} = P^T d^l$
    $w_0^{l-1} = 0$
    for $j = 0, \ldots, \gamma$ **do**
      $w_j^{l-1} = MGM^\gamma(K^{l-1}, w_j^{l-1}, d^{l-1}, l-1)$
    end for
    $w^l = P w_\Gamma^{l-1}$
    $\tilde{u}^l = \hat{u}^l + w^l$
    $u^l = S^\nu(K^l, \tilde{u}^l, f^l)$
  end if
  return u^l
end procedure

One way to obtain a p-robust method is by using a special smoother, e.g., as defined in [102]. This smoother is based on a stable splitting of the spline space into a large subspace where a robust inverse inequality holds, and smaller subspaces. For the large subspace the mass smoother is chosen and for the smaller subspaces direct solvers. The individual smoothers are then combined by means of the subspace correction approach. Note, this smoother does not lead to a MG method, which is robust with respect to the geometrical mapping. By a suitable combination of a forward and backward Gauss-Seidel smoother and the p-robust smoother, one can reduce the influence of the geometrical mapping on the number of iterations.

6.1.2 Fast Diagonalization Method

The FD method is a direct solver for matrices which have a certain tensor-product structure. However, the mass and stiffness matrix in IgA do not have tensor-product structure on the physical domain due to the presence of the geometrical mapping. Provided that the diffusion coefficient α is constant on each patch, these matrices have tensor-product structure on the parameter domain. Therefore, the FD method provides a solver on the parameter domain. Hence, it can be used as a preconditioner in an iterative method for the problem posed on the physical domain. Since, the matrices from the parameter and physical domain are spectrally equivalent with constants only depending on the geometrical mapping, the preconditioner is robust with respect to the degree p and mesh-size h.

For the presentation of the idea, we restrict ourselves to the two-dimensional case of a symmetric stiffness matrix, extensions to three dimensions and non-symmetric matrices can be found in [192] and [211]. The stiffness matrix corresponding to an IgA discretization of the Poisson equation for a two-dimensional domain has the following structure

$$K = K_1 \otimes M_2 + M_1 \otimes K_2, \tag{6.1}$$

where M_i and K_i are the mass and stiffness matrix along dimension i, respectively. The main idea is to perform a generalized eigendecomposition of the pairs (K_i, M_i), $i = 1$ and 2, i.e.,

$$K_i U_i = M_i U_i D_i,$$

with the property that $U_i^T M_i U_i = I$ and D_i is diagonal matrix containing the eigenvalues of $M_i^{-1} K_i$. This decomposition allow us to rewrite

$$K_i = U_i^{-T} D_i U_i^{-1} \quad \text{and} \quad M_i = U_i^{-T} U_i^{-1} \quad \text{for } i = 1 \text{ and } 2. \tag{6.2}$$

Using (6.2) in (6.1), we can write K as

$$\begin{aligned} K &= U_1^{-T} D_1 U_1^{-1} \otimes U_2^{-T} U_2^{-1} + U_1^{-T} U_1^{-1} \otimes U_2^{-T} D_2 U_2^{-1} \\ &= (U_1^{-T} \otimes U_2^{-T})(D_1 \otimes I + I \otimes D_2)(U_1^{-1} \otimes U_2^{-1}). \end{aligned}$$

Hence, we can represent the inverse K^{-1} in the following way

$$K^{-1} = (U_1 \otimes U_2)(D_1 \otimes I + I \otimes D_2)^{-1}(U_1^T \otimes U_2^T).$$

Therefore, after the pre-computation of U_i and D_i for $i = 1$ and 2, the algorithm consists of applying two matrices with tensor-product structure, and a diagonal scaling with $(D_1 \otimes I + I \otimes D_2)^{-1}$. The algorithm is summarized in Algorithm 7. For a detailed discussion about the complexity in terms of floating point operations of the FD method, we refer to [192].

Algorithm 7 Fast Diagonalization method for solving $Ku = f$ on the parameter domain

procedure FD(u, f)
  if generalized eigendecomposition (6.2) has not been computed **then**
    Compute (6.2)
  end if
  $d = (U_1^T \otimes U_2^T) f$
  $\tilde{d} = (D_1 \otimes I + I \otimes D_2)^{-1} d$
  $u = (U_1 \otimes U_2)\tilde{d}$
  return u
end procedure

6.2 Incorporating Inexact Solvers in IETI-DP

In this section, we want to investigate the different possibilities to incorporate inexact solvers into the IETI-DP algorithms. Recalling the IETI-DP algorithm from Section 3.2, we have to deal with local Dirichlet problems and local Neumann problems. Note that linear systems appearing in the application of the matrix F or for constructing the right-hand side require higher accuracy than those in the preconditioner.

6.2.1 Local Dirichlet Problems

We have to solve linear systems with the system matrix $K_{II}^{(k)}$ in the application of S in the preconditioner and when calculating the right-hand side $g^{(k)} = f_B^{(k)} - K_{IB}^{(k)}(K_{II}^{(k)})^{-1} f_I^{(k)}$ for $k = 1, \ldots, N$. These linear systems are Dirichlet problems. We mention that they would have Neumann boundary conditions only if the patch boundary contribute to the Neumann boundary of the whole domain. The right-hand side g has to be computed very accurately, i.e., at least up to discretization error.

The application of the scaled Dirichlet preconditioner M_{sD}^{-1} is basically given by the application of the local Schur complements $S^{(k)} := K_{BB}^{(k)} - K_{BI}^{(k)}(K_{II}^{(k)})^{-1} K_{IB}$. Hence, in each application we have to solve a linear system with the matrix $K_{II}^{(k)}$.

One has to be careful when replacing the action of $(K_{II}^{(k)})^{-1}$ by the action of an preconditioner $(\hat{K}_{II}^{(k)})^{-1}$. The resulting approximation $\hat{S}^{(k)}$ is in general not spectrally equivalent to $S^{(k)}$ with constants independent of h_k, even if $(\hat{K}_{II}^{(k)})^{-1}$ has such a property, like in the Multigrid case. To be more precise, one obtains a factor $O(1 + \ln(H_k/h_k))$, see [79] and [80]. In order to guarantee that the approximation $\hat{S}^{(k)}$ has spectral constants independent of h, we have to apply $O(1 + \ln(H_k/h_k))$ Multigrid cycles or, in general, preconditioned Richardson iterations as in the case of the FD method. Alternatively, one can use a bounded extension preconditioner, see [82], [81], [78] and [77]. Since

the number $\ln(H_k/h_k)$ is usually small in the range of practical applications, a few Multigrid cycles or preconditioned Richardson iterations are often enough to ensure that the corresponding inexact scaled Dirichlet preconditioner works well, cf. [121] and references therein. Finally, we note that the matrix $K_{II}^{(k)}$ is always symmetric and positive definite.

6.2.2 Local Neumann Problems

The second class of local problems are Neumann problems. They appear in the construction of the S-orthogonal basis for W_Π and in the application of $S_{\Delta\Delta}$. Let us first investigate the construction of the basis $\{\phi_j^{(k)}\}_j$ for $W_\Pi^{(k)}$. Since we look for a nodal basis, which is S-orthogonal, we have to solve the following linear system

$$\begin{bmatrix} S^{(k)} & C^{(k)^T} \\ C^{(k)} & 0 \end{bmatrix} \begin{bmatrix} \phi_j^{(k)} \\ \mu_j^{(k)} \end{bmatrix} = \begin{bmatrix} 0 \\ e_j^{(k)} \end{bmatrix}, \quad \forall j \in \{1, \dots, n_\Pi^{(k)}\}, \tag{6.3}$$

where $e_j^{(k)} \in \mathbb{R}^{n_\Pi^{(k)}}$ is the j-th unit vector and the matrix $C^{(k)}$ realizes the $n_\Pi^{(k)}$ primal variables associated to the patch $\Omega^{(k)}$. This system has to be solved for $n_\Pi^{(k)}$ right-hand sides, which is an advantage for direct solvers over iterative solvers because the expensive factorization must be computed only once. Instead of solving (6.3) directly, we use same approach as in Section 3.2, solve

$$\begin{bmatrix} K^{(k)} & C^{(k)^T} \\ C^{(k)} & 0 \end{bmatrix} \begin{bmatrix} \overline{\phi}_j^{(k)} \\ \mu_j^{(k)} \end{bmatrix} = \begin{bmatrix} 0 \\ e_j^{(k)} \end{bmatrix}, \quad \forall j \in \{1, \dots, n_\Pi^{(k)}\}, \tag{6.4}$$

and obtain the desired basis functions by $\phi_j = \overline{\phi}_j|_{\Gamma^{(k)}}$. Note that $\{\overline{\phi}_j^{(k)}\}_j$ is a K-orthogonal basis. If the patch $\Omega^{(k)}$ does not touch the boundary $\partial\Omega$, the upper left block becomes semi-definite due to the presence of a kernel. We are looking for a way to use the CG algorithm. As long as there is a trivial kernel, i.e., where $\partial\Omega^{(k)} \cap \partial\Omega \neq \emptyset$, one straightforward way would be to use the Bramble-Pasciak conjugate gradient (BPCG) algorithm or one of its variations, see [28] and [207]. However, these iterative methods require that the upper left block is positive definite. The remedy is a special preconditioner and a non-standard inner product for the CG algorithm, leading to the *Schöberl-Zulehner* (SZ) preconditioner, see [195]. An alternative approach would be to use the MinRes method, see [178], with a block diagonal preconditioner or GMRes, see [191] with a non-symmetric preconditioner. However, numerical experiments indicated that the use of MinRes leads to an increased number of iterations and overall computation time. We also observed an increased overall computation time for GMRes.

First we give a short outline of the Schöberl-Zulehner preconditioner, see [195] for a more detailed discussion. Let us consider a general saddle-point matrix of the

form

$$\mathcal{K} := \begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix},$$

where A is a symmetric positive semi-definite matrix and B has full rank. Moreover, we assume that A is positive definite on $\ker(B)$, this guarantees the invertibility of $\mathcal{K}$. Next, we define the preconditioner

$$\hat{\mathcal{K}} := \begin{bmatrix} \hat{A} & B^T \\ B & B\hat{A}^{-1}B^T - \hat{H} \end{bmatrix},$$

where $\hat{A}$ and $\hat{H}$ are symmetric positive definite preconditioners for A and $H := B\hat{A}^{-1}B^T$, respectively. The matrix H is often called *inexact Schur complement*. One can proof if $\hat{A} > A$ and $\hat{H} < H$, then $\hat{\mathcal{K}}^{-1}\mathcal{K}$ is symmetric and positive definite in the scalar produced induced by

$$D := \begin{bmatrix} \hat{A} - A & 0 \\ 0 & H - \hat{H} \end{bmatrix}.$$

This already enables the use of the CG algorithm to solve $\hat{\mathcal{K}}^{-1}\mathcal{K}x = \hat{\mathcal{K}}^{-1}y$, in the non-standard scalar product induced by D. Moreover, if one can find constants $\alpha > 0$ and $\beta > 0$ such that $A > \alpha\hat{A}$ and $\hat{H} < H \leq \beta\hat{H}$, then one can proof explicit bounds on the maximal an minimal eigenvalue of $\hat{\mathcal{K}}^{-1}\mathcal{K}$, see Theorem 2.2 in [195]. One obtains then the following bound on the condition number

$$\kappa(\hat{\mathcal{K}}^{-1}\mathcal{K}) \leq \frac{\beta}{\alpha}(1 + \sqrt{1 - 1/\beta}) \left(\frac{\sqrt{1 - 1/\beta} + \sqrt{5 - 1/\beta}}{2} \right)^2.$$

If the constants α and β are independent of parameters like h or p, then also the condition number κ.

The SZ preconditioner for (6.4) requires preconditioners $\hat{K}^{(k)}$ and $\hat{H}^{(k)}$ for the upper left block $K^{(k)}$ and its inexact Schur complement $H^{(k)} := C^{(k)}(\hat{K}^{(k)})^{-1}C^{(k)^T}$, respectively. It is required that $\hat{K}^{(k)} > K^{(k)}$, which implies that $\hat{K}^{(k)}$ has to be positive definite. In order to handle also the case where $K^{(k)}$ is singular, we need to set up the preconditioner based on a regularized matrix $K_M^{(k)} := K^{(k)} + \alpha H_k^d \widehat{M}^{(k)}$, where α is chosen to appropriately and $\widehat{M}^{(k)}$ is the mass matrix on the parameter domain. Note, we can exploit the tensor-product structure to efficiently assemble the mass matrix $\widehat{M}^{(k)}$. The spectral inequality $\hat{K}^{(k)} > K^{(k)}$ can then be guaranteed by a suitable scaling of $\hat{K}^{(k)}$. Finally, this provides us with an appropriate preconditioner $\hat{K}^{(k)}$ for $K^{(k)}$. Secondly, the SZ preconditioner requires that $\hat{H}^{(k)} < H^{(k)}$. Since in our case the number of rows of $C^{(k)}$ is given by $n_\Pi^{(k)}$, which is a small number that does not change during refinement, we calculate the inexact Schur complement exactly. This can be performed by applying $(\hat{K}^{(k)})^{-1}$ to $n_\Pi^{(k)}$ vectors. Finally, by a suitable scaling, e.g.,

$\hat{H}^{(k)} := 0.99H^{(k)}$, we obtain the desired matrix inequality. Having the preconditioners $\hat{K}^{(k)}$ and $\hat{H}^{(k)}$, we apply CG with the SZ preconditioner to construct the basis for $W_{\Pi}^{(k)}$.

The second type of Neumann problem appears in the application of F. We look for a solution of the system $S_{\Delta\Delta}^{(k)} w_{\Delta}^{(k)} = f_{\Delta}^{(k)}$, which can be written as

$$\begin{bmatrix} S^{(k)} & C^{(k)^T} \\ C^{(k)} & 0 \end{bmatrix} \begin{bmatrix} w_{\Delta}^{(k)} \\ \mu^{(k)} \end{bmatrix} = \begin{bmatrix} f^{(k)} \\ 0 \end{bmatrix}. \tag{6.5}$$

Certainly, one can use the same method as above. However, we can utilize the fact that we search for a minimizer of $\frac{1}{2}(S^{(k)}w^{(k)}, w^{(k)}) - (w^{(k)}, f^{(k)})$ in the subspace given by $C^{(k)}w^{(k)} = 0$. This solution can be computed by first solving the unconstrained problem and projecting the minimizer into the subspace using a energy-minimizing projection. The projection is trivial because the decomposition of $\widetilde{W}$ into W_{Π} and W_{Δ} is S-orthogonal.

Note that the CG algorithm, when applied to a positive semi-definite matrix, stays in the factor space with respect to the kernel, and computes one of the minimizers. The solution of the constrained minimization problem is, as outlined above, obtained by applying the projection. As long as the number of CG iterations is not too large, numerical instabilities are not observed when applying CG to a positive semi-definite problem.

The S-orthogonal basis has to be computed very accurate in order to maintain the orthogonality. Because the equation $S_{\Delta\Delta}^{(k)} w_{\Delta}^{(k)} = f_{\Delta}^{(k)}$ appears in the system matrix F, its solution also requires an accuracy of at least the discretization error.

Remark 6.1. *Since we realize the primal variables by means of constraints, the parameter domain matrix $\widehat{K}^{(k)}$ corresponding to $K^{(k)}$ in* (6.4) *has tensor-product structure. This is an important requirement for the p-robust MG and FD method. Using a basis transformation to incorporate the primal variables would destroy the tensor-product structure.*

Remark 6.2. *The solutions to the problems* (6.4) *and* (6.5) *have to be computed up to a high accuracy. However, we observe numerical instabilities if the used tolerance is too close at machine precision. Therefore, one has to find a balance between computing the solution sufficiently accurate and avoiding numerical instabilities. In the numerical experiments in Section 6.3 the choice $5 \cdot 10^{-12}$ for* (6.4) *and $5 \cdot 10^{-10}$ for* (6.5) *works well. Nevertheless, this is an issue, which has to be taken into account, when using inexact variants of IETI-DP.*

6.2.3 Different Inexact Formulations

From the discussion above, we deduce four (reasonable) combinations of the IETI-DP method with direct solvers (D) and inexact solver/preconditioner (I).

(D-D) This is the classical IETI-DP method, where we use direct solvers everywhere.

(D-I) We use an inexact solver in the scaled preconditioner for the solution of the local Dirichlet problems and the transformation of the right-hand side, see Section 3.2. As already mentioned, the required accuracy for computing $\widetilde{g}$ has to be of the order of discretization error.

(I-I) We use the inexact solver for all patch-local problems, i.e., the local Dirichlet and Neumann problems. This implies that also the calculation of the basis for W_Δ is performed by means of an inexact method, which turns out to be very costly. Moreover, for each application of F, we have to solve a local Neumann problem in W_Δ with the accuracy in the order of the discretization error.

(I-I-S) To overcome the efficiency problem of applying the inexact solver at each iteration up to a small precision, we use the saddle-point formulation (3.11) instead of F. On the one hand, at each iteration step, we only have to apply a given matrix instead of solving a linear system. On the other hand, we now have to deal with a saddle-point problem. Moreover, the iteration is not only applied to the interface dofs, but also to the dofs in the whole domain.

We will always assume that the considered multi-patch domain has only a moderate number of patches such that the coarse problem can still be handled by a spare direct solver. For extensions to inexact version for the coarse problem, we refer to, e.g., [130].

For the first three methods, we use the CG method to solve $F\lambda = d$ as outer iteration. For the last setting (I-I-S), we have to deal with the saddle-point problem (3.11), which we solve using the BPCG method. The building blocks for this method are a preconditioner $\widehat{\widetilde{K}}$ for $\widetilde{K}$ and $\hat{F}$ for the Schur complement F. The construction of $\widehat{\widetilde{K}}$ follows the same steps as in the previous section, but we only apply either a few MG cycles or a few iterations of the preconditioned Richardson iteration. Concerning $\hat{F}$, a good choice is the scaled Dirichlet preconditioner M_{sD}^{-1}, cf. [130].

6.3 Numerical Experiments

We solve the model problem (2.2) on a two- and three-dimensional computational domain. In the two-dimensional case, we use the quarter annulus divided into $32 = 8 \times 4$ patches, as illustrated in Figure 6.1(a). The three-dimensional domain is the twisted quarter annulus, decomposed into $128 = 4 \times 4 \times 8$ patches as presented in Figure 6.1(b). We consider as primal variables vertex values and edge averages for the two-dimensional example and only edge averages for the three-dimensional example.

As inexact solver we investigate a p-robust Multigrid method and the Fast Diagonalization method. Regarding the tests using Multigrid, we use a standard MG method

(a) Quarter annulus

(b) Twisted quarter annulus

Figure 6.1: Illustration of the two- and three-dimensional computational domain.

based on a hierarchy of nested grids keeping p fixed and use a standard Gauss Seidel (GS) smoother for the examples with polynomial degree $p = 2$. For the examples with higher polynomial degree ($p = 4$ or 7), we have used $p = 1$ on all grid levels but the finest grid. This does not yield nested spaces. Thus, we cannot use the canonical embedding and restriction. Instead, we use L^2-projections to realize them. On the finest grid, we use a MG smoother suitable for high-order IgA, namely a variant of the subspace-corrected mass smoother proposed and analyzed in [102]. For this smoother, it was shown that the resulting MG method is robust with respect to both the grid size and the polynomial degree. However, for $p = 1$ or 2, standard approaches are more efficient. Thus, we again use this smoother only for the finest level, while for all other grid levels we use standard GS smoothers. To archive better results, we have modified the subspace-corrected mass smoother by incorporating a rank-one approximation of the geometry transformation. Similarly, we construct the FD method based on the one-dimensional stiffness and mass matrix in the physical space K_i and M_i for $i = 1, \dots, d$, respectively, in order to incorporate some information of the geometrical mapping in the FD method.

For the outer CG or BPCG iteration, we use a zero initial guess, and the reduction of the initial residual by the factor 10^{-6} as stopping criterion. The local problems related to the calculation of the S-orthogonal basis are solved up to a tolerance of $5 \cdot 10^{-12}$. In case of the (I-I) version, the local Neumann problems (6.5) in W_Δ are solved up to a relative error of $5 \cdot 10^{-10}$. The number of steps of the inexact method in the preconditioner is fixed.

The algorithm is realized with the open source C++ library G+Smo [162], which uses the linear algebra facilities of the Eigen library [76]. We utilize the PARDISO 5.0.0 Solver [140] for performing the LU factorizations.[1] In the following the presented timings are given in seconds. We note that they include in addition to the setup and solving time also the time spent for assembling the local matrices. For each row of

[1]Our code is compiled with the `gcc 4.8.3` compiler with optimization flag `-O3`. The results are obtain on the RADON1 cluster at Linz, see `https://www.ricam.oeaw.ac.at/hpc/`. We use a single core of a node, equipped with 2x Xeon E5-2630v3 "Haswell" CPU (8 Cores, 2.4Ghz, 20MB Cache) and 128 GB RAM.

Tables 6.2, 6.6		Tables 6.3, 6.7		Tables 6.4, 6.8		Tables 6.5, 6.9	
$p=2$	$p=7$	$p=2$	$p=4$	$p=2$	$p=7$	$p=2$	$p=4$
2.2	16.8	1.1	20.4	0.5	16.8	0.1	2.4
8.8	67.7	8.4	171.8	2.2	67.0	1.1	20.5
34.8	271.8	65.8	1404.3	9.1	270.5	9.5	167.4
138.7	1081.0	521.9	11513.2	37.8	1085.4	95.8	1375.5
565.4	4328.6			166.8	4381.4		

Table 6.1: The time in seconds used for assembling the local matrices. These timings are included in Table 6.2 to Table 6.9.

$p=2$	**D-D**		**MG-D**		**MG-MG**		**MG-MG-S**	
Dofs	It.	Time	It.	Time	It.	Time	It.	Time
134421	9	9.5	9	7.8	9	12.5	14	14.4
530965	10	45.4	10	37.0	10	54.4	15	90.1
2110485	11	224.1	11	172.4	11	272.5	16	568.6
8415253	11	1005.6	11	762.5	11	1181.4	15	3394.1
33607701	OoM		OoM		13	5070	OoM	
$p=7$	**D-D**		**MG-D**		**MG-MG**		**MG-MG-S**	
Dofs	It.	Time	It.	Time	It.	Time	It.	Time
45753	10	25.7	10	26.7	10	56.7	14	53.5
155961	11	108.4	11	110.7	11	225.1	15	211.8
572985	12	498.8	12	495.5	12	1048.5	17	1013.8
2193465	13	2384.5	13	2265.4	14	4427.2	18	4344.5
8580153	OoM		OoM		15	18484.1	20	19958.6

Table 6.2: Number of outer iterations and timings for the four different formulations using the quarter annulus, see Figure 6.1(a). GS smoother is used for $p = 2$ and p-robust subspace corrected mass smoother for $p = 7$.

Table 6.2 to Table 6.9, the included assembling times are listed separately in Table 6.1. Note, the assembling of the local matrices does not depend on the chosen variant and on the used inexact solver.

6.3.1 Multigrid as Inexact Solver

For the local Dirichlet problems in the scaled Dirichlet preconditioner, we use 2 V-cycles. The local Neumann problems, which appear in the preconditioner of the (MG-MG-S) version, are approximately solved by 3 V-cycles. Moreover, the regularization parameter α is chosen to be 1. In the following, we report on the number of CG iterations to solve (3.12) and BP-CG iterations for (3.11) and the total time, which includes the assembling, the IETI-DP setup and solving phase.

$p=2$	**D-D**		**MG-D**		**MG-MG**		**MG-MG-S**	
Dofs	It.	Time	It.	Time	It.	Time	It.	Time
14079	11	2.6	11	2.5	11	7.6	25	7.2
86975	12	19.3	12	19.1	12	59.1	26	59.1
606015	14	213.2	14	197.8	14	484.0	30	616.5
4513343	OoM		16	2764.4	16	5244.5	35	11657.5
$p=4$	**D-D**		**MG-D**		**MG-MG**		**MG-MG-S**	
Dofs	It.	Time	It.	Time	It.	Time	It.	Time
40095	13	29.5	13	32.8	13	112.5	23	104.8
160863	15	234.5	15	254.2	15	660.0	28	631.4
849375	16	2237.3	17	2356.8	17	5400.0	32	5312.4
5390559	OoM		OoM		19	45243.1	37	53378.3

Table 6.3: Number of outer iterations and timings for the four different formulations using the twisted quarter annulus, see Figure 6.1(b). GS smoother is used for $p = 2$ and p-robust subspace corrected mass smoother for $p = 4$.

In Table 6.2, we summarize the results for the two-dimensional domain for $p = 2$ and 7. We observe that replacing the direct solver in the preconditioner with two MG V-cycles does not change the number of outer iterations. Moreover, going from the Schur complement to the saddle-point formulation and using BPCG there, leads only to a minor increase in the number of outer iterations. In all cases, the logarithmic dependence of the condition number on h is preserved. The advantage of the formulation using only MG, especially (MG-MG), is its smaller memory footprint, therefore, the possibility of solving larger systems. However, the setting with the best performance is (MG-D). Concluding, for small polynomial degrees and using the GS smoother, (MG-MG) gives reasonable trade off between performance and memory usage and for larger polynomial degrees, this setting can be still recommended if memory consumption is an issue.

In the case $p = 2$, for the inner iterations, we have observed that the CG needed on average 8 iterations to compute $\widetilde{g}$, the calculation of the S-orthogonal basis needed on average 14 iterations and the solution of (6.5) required on average 10 iterations. For the second case, $p = 7$, we needed 9 iterations to compute $\widetilde{g}$, 13 iterations for the calculation of the S-orthogonal basis and 10 iterations for the solutions of (6.5). Here and in what follows, we have taken the average over the patches, the individual levels and the individual steps of the outer iteration. We mention that the number of inner iterations was only varying slightly.

In Table 6.3, we summarize the results for the three-dimensional domain and for $p = 2$ and 4. We observe that replacing the direct solver in the preconditioner with two MG V-cycles does not change the number of outer iterations. We further observe that the results behave similar to the one of the two-dimensional case. However, the number of iterations almost doubled when using BPCG for (MG-MG-S). In all cases,

$p=2$			**D-D**		**MG-D**		**MG-MG**		**MG-MG-S**	
n_Π	N	Dofs	It.	Time	It.	Time	It.	Time	It.	Time
73	32	34453	8	2.4	8	1.8	8	3.8	19	3.7
337	128	138601	9	10.0	9	7.6	9	19.1	13	14.7
1441	512	555985	9	40.4	9	30.7	9	83.1	12	60.5
5953	2048	2227105	8	160.9	8	120.7	8	330.8	11	242.3
24193	8192	8914753	8	722.3	8	489.0	8	1357.8	12	973.2
$p=7$			**D-D**		**MG-D**		**MG-MG**		**MG-MG-S**	
n_Π	N	Dofs	It.	Time	It.	Time	It.	Time	It.	Time
73	32	45753	10	27.2	10	26.7	10	61.7	20	60.1
337	128	183921	11	110.0	11	108.9	11	268.5	15	234.8
1441	512	737505	11	446.8	11	438.8	11	1111.2	13	943.0
5953	2048	2953665	10	1777.3	10	1729.3	10	4468.4	12	3821.0
24193	8192	11821953	OoM		OoM		10	19691.5	11	15392.3

Table 6.4: Number of outer iterations and timings for the four different formulations having a fixed ratio H/h, using the quarter annulus, see Figure 6.1(a). GS smoother is used for $p = 2$ and p-robust subspace corrected mass smoother for $p = 7$.

the logarithmic dependence of the condition number on h is preserved. The advantage of the formulation using only MG, especially (MG-MG), is its smaller memory footprint, therefore the possibility of solving larger systems. The best performance is obtained sometimes by (D-D) and sometimes by (MG-D), where both approaches are comparable.

Concerning the inner iterations, for $p = 2$, we need on average 15 CG iterations to compute $\widetilde{g}$, 22 CG iterations to build up each S-orthogonal basis function, and 18 CG iterations to solve (6.5). In the case of $p = 4$, we needed on average only 10 iterations to compute $\widetilde{g}$, 14 iterations for the construction of the S-orthogonal basis functions, and 11 iterations for solving (6.5).

Next, we investigate the weak scalability of the method, i.e., fixing the ratio H/h and increasing the number of patches N. Note, we introduce a C^0 coupling between the newly introduced patches. We expect constant number of iterations and a linear increase of the computation time. We first consider the two-dimensional problem. In Table 6.4, we report on the size of the coarse space n_Π, the number of patches N and for each method we summarize the number of iterations and computation time. We observe that the number of iterations and computation time behave as expected for both considered degrees. The number of inner MG iterations are identical to those obtained for the test summarized in Table 6.2. Similarly, for the three-dimensional example, see Table 6.5, we observe the expected behaviour of all the methods with identical inner iterations to Table 6.3.

$p=2$			D-D		MG-D		MG-MG		MG-MG-S	
n_Π	N	Dofs	It.	Time	It.	Time	It.	Time	It.	Time
16	16	1539	8	0.2	7	0.24	8	0.5	19	0.4
240	128	14079	11	2.7	11	2.5	11	7.7	25	7.1
2464	1024	120159	12	25.2	12	23.2	12	84.0	27	76.4
22080	8192	992319	12	283.7	12	215.4	12	1030.9	29	709.5
$p=4$			D-D		MG-D		MG-MG		MG-MG-S	
n_Π	N	Dofs	It.	Time	It.	Time	It.	Time	It.	Time
16	16	4563	9	3.3	9	3.5	9	8.0	16	7.5
240	128	40095	13	30.5	13	31.9	13	113.7	25	106.5
2464	1024	335775	14	260.8	14	271.6	14	1168.5	27	1088.8
22080	8192	2747583	14	2259.8	14	2277.8	14	11221.9	26	9941.4

Table 6.5: Number of outer iterations and timings for the four different formulations having a fixed ratio H/h, using the twisted quarter annulus, see Figure 6.1(b). GS smoother is used for $p = 2$ and p-robust subspace corrected mass smoother for $p = 4$.

6.3.2 Fast Diagonalization Method as Inexact Solver

In this section, we use the FD method as local solver. As already mentioned in Section 6.1.2, this method is robust in p and in h, and the condition number of the preconditioned system depends only on the geometrical mapping. For the local Dirichlet problems in the scaled Dirichlet preconditioner, we use 5 preconditioned Richardson steps. The local Neumann problems, which appear in the preconditioner of the (MG-MG-S) version, are approximately solved by 3 preconditioned Richardson steps. Moreover, the regularization parameter α is chosen to be 10. In the following, we report on the number of CG iterations to solve (3.12) and BP-CG iterations for (3.11) and the total time, which includes the assembling, the IETI-DP setup and solving phase.

In Table 6.6, we summarize the results for the two-dimensional domain for $p = 2$ and 7. We observe that results using the FD method behave similar to the one using the MG method. However, for the case of $p = 7$, the (FD-FD) and (FD-FD-S) version can even beat the classical IETI-DP method, where (FD-FD) is the most efficient one. For the tests with $p = 2$, the best performance is obtained sometimes by (FD-D). Concerning the inner iterations, for $p = 2$, we need on average 9 CG iterations to compute $\widetilde{g}$, 16 CG iterations to build up each S-orthogonal basis function, and 9 CG iterations to solve (6.5). Similar for the case of $p = 7$, we need on average only 9 iterations to compute $\widetilde{g}$, 16 iterations for the construction of the S-orthogonal basis functions, and 9 iterations for solving (6.5).

The results for the three-dimensional domain for $p = 2$ and 4 are summarized in Table 6.7. In contrast to the two-dimensional case, the best performance is obtained by either (D-D) or (FD-D), where the results for (FD-D) were only slightly better.

$p = 2$	**D-D**		**FD-D**		**FD-FD**		**FD-FD-S**	
Dofs	It.	Time	It.	Time	It.	Time	It.	Time
134421	9	9.5	9	7.1	9	6.8	12	8.4
530965	10	45.4	10	35.3	10	39.3	14	74.2
2110485	11	224.1	11	167.5	11	213.1	15	497.5
8415253	11	1005.6	13	825.7	13	1218.2	17	3508.9
33607701	OoM		OoM		15	7290.5	OoM	
$p = 7$	**D-D**		**FD-D**		**FD-FD**		**FD-FD- S**	
Dofs	It.	Time	It.	Time	It.	Time	It.	Time
45753	10	25.7	10	23.5	10	22.3	13	22.9
155961	11	108.4	11	97.6	11	89.5	14	92.9
572985	12	498.8	12	430.2	12	395.6	15	431.6
2193465	13	2384.5	14	1971.2	14	1659.4	18	1981.8
8580153	OoM		OoM		17	7187.5	20	10648.1

Table 6.6: Number of outer iterations and timings for the four different formulations using the quarter annulus, see Figure 6.1(a). FD method is used as inexact solver.

$p = 2$	**D-D**		**FD-D**		**FD-FD**		**FD-FD-S**	
Dofs	It.	Time	It.	Time	It.	Time	It.	Time
14079	11	2.6	11	2.5	11	3.6	25	4.0
86975	12	19.3	12	17.8	12	22.1	27	28.0
606015	14	213.2	14	180.7	14	180.8	33	398.3
4513343	OoM		17	2586.5	19	2265.7	40	9780.6
$p = 4$	**D-D**		**FD-D**		**FD-FD**		**FD-FD-S**	
Dofs	It.	Time	It.	Time	It.	Time	It.	Time
40095	13	29.5	13	28.8	13	36.4	33	39.8
160863	15	234.5	15	226.2	15	262.6	37	280.4
849375	16	2237.3	17	2069.2	19	2481.8	41	2624.8
5390559	OoM		OoM		34	21342.2	58	37322.1

Table 6.7: Number of outer iterations and timings for the four different formulations using the twisted quarter annulus, see Figure 6.1(b). FD method is used as inexact solver.

$p=2$			**D-D**		**FD-D**		**FD-FD**		**FD-FD-S**	
n_Π	N	Dofs	It.	Time	It.	Time	It.	Time	It.	Time
73	32	34453	8	2.4	8	1.7	8	1.6	11	1.8
337	128	138601	9	10.0	9	7.2	9	6.7	11	8.2
1441	512	555985	9	40.4	9	28.9	9	26.8	11	33.3
5953	2048	2227105	8	160.9	9	114.4	8	109.1	11	123.9
24193	8192	8914753	8	722.3	8	497.9	*	*	10	497.8
$p=7$			**D-D**		**FD-D**		**FD-FD**		**FD-FD-S**	
n_Π	N	Dofs	It.	Time	It.	Time	It.	Time	It.	Time
73	32	45753	10	27.2	10	23.7	10	22.2	13	22.7
337	128	183921	11	110.0	11	95.2	11	90.0	13	92.5
1441	512	737505	11	446.8	11	382.9	11	359.6	15	377.2
5953	2048	2953665	10	1777.3	10	1522.7	10	1443.2	13	1486.1
24193	8192	11821953	OoM		OoM		*	*	10	5873.1

Table 6.8: Number of outer iterations and timings for the four different formulations having a fixed ratio H/h, using the quarter annulus, see Figure 6.1(a). FD method is used as inexact solver. The * indicate, that no results could be obtained due to numerical instabilities.

One reason is the more distorted geometry, which leads to a larger number of inner iterations. However, the pure inexact variants (FD-FD) and (FD-FD-S) still have a significant memory advantage. Concerning the inner iterations, for $p = 2$, we need on average 16 CG iterations to compute $\widetilde{g}$, 24 CG iterations to build up each S-orthogonal basis function, and 17 CG iterations to solve (6.5). In the case of $p = 4$, we need on average only 16 iterations to compute $\widetilde{g}$, 25 iterations for the construction of the S-orthogonal basis functions, and 18 iterations for solving (6.5).

As in the previous section, we also investigate the weak scalability of the method for different polynomial degrees and for the two- and three-dimensional domain. As mentioned in the previous section, we introduce a C^0 coupling between the newly introduced patches. In Table 6.8, we report on the size of the coarse space n_Π, the number of patches N and for each method we summarize the number of iterations and computation time. We observe that the number of iterations and computation time behave as expected for both considered degrees. The number of inner iterations are identical to those obtained for the test summarized in Table 6.6. Similarly, for the three-dimensional example, see Table 6.9, we observe the expected behaviour of all the methods with identical inner iterations to Table 6.7. However using the FD-FD version for the last refinement, we observe numerical instabilities and no convergence of the IETI-DP method, cf. Remark 6.2. These could not be resolved, even by increasing the tolerance of the iterative methods.

$p=2$			**D-D**		**FD-D**		**FD-FD**		**FD-FD-S**	
n_Π	N	Dofs	It.	Time	It.	Time	It.	Time	It.	Time
16	16	1539	8	0.3	8	0.2	8	0.3	19	0.3
240	128	14079	11	2.7	11	2.5	11	3.5	25	4.1
2464	1024	120159	12	25.2	11	22.5	11	35.0	29	43.0
22080	8192	992319	12	283.7	11	209.2	11	370.8	32	395.0
$p=4$			**D-D**		**FD-D**		**FD-FD**		**FD-FD-S**	
n_Π	N	Dofs	It.	Time	It.	Time	It.	Time	It.	Time
16	16	4563	10	3.4	10	3.1	9	3.6	25	3.6
240	128	40095	13	30.3	13	28.7	13	35.7	32	38.8
2464	1024	335775	14	260.8	14	246.5	14	322.8	35	354.8
22080	8192	2747583	14	2259.8	14	2059.3	14	2983.4	38	3037.1

Table 6.9: Number of outer iterations and timings for the four different formulations having a fixed ratio H/h, using the twisted quarter annulus, see Figure 6.1(b). FD method is used as inexact solver.

Chapter 7

IETI-DP for Space-Time Formulations

Time-dependent parabolic PDEs play an important role in the simulation of various physical processes, like heat conduction and diffusion problems or 2d eddy current problems in electro-magnetics, and they are often given as initial-boundary value problem (IBVP). In this chapter, we consider the linear parabolic IBVP: find $u : \overline{Q} \to \mathbb{R}$ such that

$$\partial_t u - \Delta u = f \text{ in } Q, \ u = 0 \text{ on } \Sigma, \text{ and } u = u_0 \text{ on } \overline{\Sigma}_0. \tag{7.1}$$

This IBVP is posed in the space-time cylinder $\overline{Q} = \overline{\Omega} \times [0, T] = Q \cup \Sigma \cup \overline{\Sigma}_0 \cup \overline{\Sigma}_T$, where $\Sigma := \partial\Omega \times (0, T)$, $\Sigma_0 := \Omega \times \{0\}$, $\Sigma_T := \Omega \times \{T\}$, and Ω is a bounded Lipschitz domain.

The discretization of such problem is usually either performed by first discretizing in time by a time-stepping method and then in space by, e.g., finite elements or vice versa. The former approach is often denoted as Rothe's method [143] and the latter one vertical method of lines [213]. Both of the two approaches are sequential in time. In order to treat such problems on massively parallel computers, new techniques are required to overcome the sequential structure. There exist different approaches for parallelization in time. Here, we focus on space-time methods. We refer to [67] for an overview of different time-parallel methods.

In [146], a time-upwind test functions were used to construct a stable single-patch discretization scheme in the IgA framework. Here, we extend this approach to multiple patches in time, where each space-time patch Q_n is given as space-time slabs, i.e., $Q_n := Q^{(n)} := \Omega \times (t_{n-1}, t_n)$ with an appropriate decomposition $0 = t_0 < t_1 < \ldots < t_N = T$ of the time interval $[0, T]$. A discontinuous Galerkin (dG) approach is used for the coupling of the space-time-slabs. The final huge linear system $\boldsymbol{L}_h \boldsymbol{u}_h = \boldsymbol{f}_h$ is solved by the space-time MG method introduced in [68], where we develop robust and parallelizeable smoothers. Their construction is based generalized eigendecomposition to decouple the space-time problem into a series of spatial problems, where we use

ideas from [211] and [192] as used for the FD method. We propose and analyze preconditioners based on the work in [237] for the resulting symmetric indefinite problems corresponding to complex eigenvalues.

In Section 7.1 we introduce the dG space-time formulation of (7.1) on time-slabs, state results on exists, uniqueness and discretization error estimates, and discuss the issue of fast assembly. The main part of this chapter is Section 7.2, where we develop efficient solvers for the discrete problem. First we give a short introduction to space-time MG in Section 7.2.1 and continue with the development of efficient solvers, which are used inside the MG algorithm in Section 7.2.2. After a short discussion of the use of IETI-DP as additional space-parallel solver in Section 7.2.6, we conclude this chapter with numerical experiments in Section 7.3.

7.1 Continuous and Discrete Space-Time Formulation

Let $J = (0, T)$ be the time interval with some final time $T > 0$. For later use, we define the space-time cylinder $Q = \Omega \times J$ and its boundary parts $\Sigma = \partial\Omega \times J$, $\Sigma_T = \Omega \times \{T\}$ and $\Sigma_0 = \Omega \times \{0\}$ such that $\partial Q = \Sigma \cup \overline{\Sigma}_0 \cup \overline{\Sigma}_T$. According to the definition of $\partial_x^{\boldsymbol{\alpha}}$, we now define the spatial gradient $\nabla_x v = (\partial_{x_1} v, \dots, \partial_{x_d} v)$. Let ℓ and m be non-negative integers. For functions defined in the space-time cylinder Q, we define the Sobolev spaces

$$H^{\ell,m}(Q) = \{v \in L_2(Q) : \partial_x^{\boldsymbol{\alpha}} v \in L_2(Q) \text{ for } 0 \le |\boldsymbol{\alpha}| \le \ell, \text{ and } \partial_t^i v \in L_2(Q),\ i = 0, \dots, m\},$$

where $\partial_t = \partial/\partial t$, and, in particular, the subspaces

$$\begin{aligned} H_0^{1,0}(Q) =& \{v \in L_2(Q) : \nabla_x v \in [L_2(Q)]^d,\ v = 0 \text{ on } \Sigma\} \text{ and} \\ H_{0,\bar{0}}^{1,1}(Q) =& \{v \in L_2(Q) : \nabla_x v \in [L_2(Q)]^d,\ \partial_t v \in L_2(Q), v = 0 \text{ on } \Sigma,\ v = 0 \text{ on } \Sigma_T\}. \end{aligned}$$

We equip the above spaces with the norms and semi-norms

$$\|v\|_{H^{\ell,m}(Q)} = \Big(\sum_{|\boldsymbol{\alpha}| \le \ell} \|\partial_x^{(\alpha_1,\dots,\alpha_d)} v\|_{L_2(Q)}^2 + \sum_{m_0=0}^{m} \|\partial_t^{m_0} v\|_{L_2(Q)}^2 \Big)^{\frac{1}{2}}$$

and

$$|v|_{H^{\ell,m}(Q)} = \Big(\sum_{|\boldsymbol{\alpha}| = \ell} \|\partial_x^{(\alpha_1,\dots,\alpha_d)} v\|_{L_2(Q)}^2 + \|\partial_t^{m} v\|_{L_2(Q)}^2 \Big)^{\frac{1}{2}},$$

respectively.

Using the standard procedure and integration by parts with respect to both x and t, we can easily derive the following space-time variational formulation of (7.1): find $u \in H_0^{1,0}(Q)$ such that

$$a(u, v) = l(v) \quad \text{for all } v \in H_{0,\bar{0}}^{1,1}(Q), \tag{7.2}$$

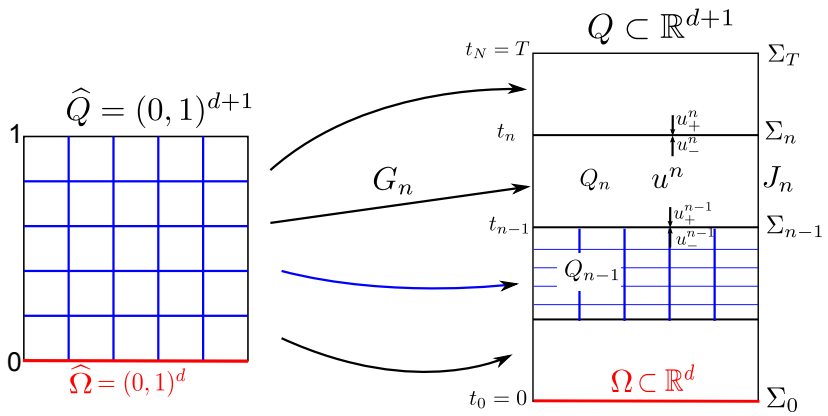


Figure 7.1: Decomposition of the space-time domain Q into time-slabs Q_n.

with the bilinear form

$$a(u,v) = -\int_Q u(x,t)\partial_t v(x,t)\,dx\,dt + \int_Q \nabla_x u(x,t)\,\cdot\,\nabla_x v(x,t)\,dx\,dt$$

and the linear form

$$l(v) = \int_Q f(x,t)v(x,t)\,dx\,dt + \int_\Omega u_0(x)v(x,0)\,dx,$$

where the source $f \in L_2(Q)$ and the initial conditions $u_0 \in L_2(\Omega)$ are given.

For simplicity, we only consider homogeneous Dirichlet boundary conditions on Σ. However, the method presented here can easily be generalized to other classes of boundary conditions. The space-time variational formulation (7.2) has a unique solution, see, e.g, [141] and [142], where one can also find a priori estimates and regularity results.

Assumption 6. We assume that the solution u of (7.2) belongs to $V = H_0^{1,0}(Q) \cap H^{\ell,m}(Q)$ with some $\ell \geq 2$ and $m \geq 1$.

We describe the space-time cylinder Q as a union of non-overlapping time patches Q_1, Q_2,...,Q_N. We consider a partition $0 = t_0 < t_1 < \ldots < t_N = T$ of the time interval $\overline{J} = [0,T]$, and denote the sub-intervals by $J_n = (t_{n-1}, t_n)$. We now define the time patches $Q_n = \Omega \times J_n$ and the faces $\Sigma_n = \overline{Q}_{n+1} \cap \overline{Q}_n = \Omega \times \{t_n\}$ between the time patches, where we identify Σ_T and Σ_N. In that way, we have the decomposition $\overline{Q} = \cup_{n=1}^N \overline{Q}_n$, where each space-time cylinder Q_n has a geometrical mapping $G_n : \overline{\widehat{Q}} := [0,1]^{d+1} \to \overline{Q}_n := \overline{\Omega} \times \overline{J}_n$. To keep the notation simple, in what follows, we will use the sup-index n to denote the restrictions to Q_n, e.g., $u^n := u|_{Q_n}$. An illustration is given in Figure 7.1.

Remark 7.1. *We note that the spatial domain Ω can also be a multi-patch domain. This leads to a representation of Q_n as union of non-overlapping space-time patches*

$Q_{n,k}, k = 1, \ldots, K$, *i.e.,* $\overline{Q}_n = \cup_{k=1}^{K} \overline{Q}_{n,k}$. *The corresponding bases are then coupled in a conforming way.*

We denote the global discontinuous B-Spline space and the local continuous patch-wise B-Spline spaces by

$$V_{0h} = \{v_h \in L_2(Q) : v_h|_{Q_n} \in V_h^n, \text{ for } n = 1, \ldots, N, \text{ and } v_h|_\Sigma = 0\} \tag{7.3}$$

and

$$V_{0h}^n = \{v_h \in V_h^n, \text{ for } n = 1, \ldots, N, \text{ and } v_h|_\Sigma = 0\}, \tag{7.4}$$

respectively. Notice that $v_h \in V_{0h}$ is in general discontinuous across Σ_n. We introduce the notations

$$v_{h,+}^n = \lim_{\varepsilon \to 0^+} v_h(t_n + \varepsilon),\ v_{h,-}^n = \lim_{\varepsilon \to 0^-} v_h(t_n + \varepsilon),\ [\![v_h]\!]^n = v_{h,+}^n - v_{h,-}^n,\ [\![v_h]\!]^0 = v_{h,+}^0,$$

where $[\![v_h]\!]^n$ denotes the jump of v_h across Σ_n for $n \geq 1$, and $[\![v_h]\!]^0 = v_{h,+}^0$ denotes the trace of v_h on Σ_0. For a smooth function u, we obviously have $[\![u]\!]^n = u_+^n - u_-^n = 0$ for $n \geq 1$, and $[\![u]\!]^0 = u|_{\Sigma_0}$.

7.1.1 Stable Multi-Patch Space-Time dG-IgA Discretization

Let us now consider the space-time slab Q_n, and let us denote the outer normal to ∂Q_n by $\boldsymbol{n} = (n_1, \ldots, n_d, n_{d+1}) = (\boldsymbol{n}_x, n_t)$. For the time being, we assume that u^{n-1} is known. Let $v_h^n \in V_{0h}^n$ and $w_h^n = v_h^n + \theta_n\, h_n \partial_t v_h^n$ with some positive parameter θ_n, which will be defined later. We note that $w_h^n\big|_\Sigma = 0$. Multiplying $\partial_t u - \Delta u = f$ by w_h^n, integrating over Q_n, and applying integration by parts, we arrive at the variational identity

$$\begin{aligned}
&\int_{Q_n} (\partial_t u\, (v_h^n + \theta_n\, h_n \partial_t v_h^n) + \nabla_x u \cdot \nabla_x v_h^n + \theta_n\, h_n \nabla_x u \cdot \nabla_x \partial_t v_h^n)\, dx\, dt \\
&- \int_{\partial Q_n} n_x \cdot \nabla_x u (v_h^n + \theta_n\, h_n \partial_t v_h^n)\, dx + \int_{\Sigma_{n-1}} u_+^{n-1}\, v_{h,+}^{n-1}\, dx \\
&= \int_{Q_n} f\, (v_h^n + \theta_n\, h_n \partial_t v_h^n)\, dx\, dt + \int_{\Sigma_{n-1}} u_-^{n-1}\, v_{h,+}^{n-1}\, dx
\end{aligned}$$

for $n = 1, \ldots, N$, where we used that $u_-^{n-1} = u_+^{n-1} = u^{n-1}$ on every Σ_{n-1}. Furthermore, using $n_x|_{\Sigma_n} = 0$ and $w_h = 0$ on Σ, we have

$$\begin{aligned}
a_{Q_n}(u, v_h) := &\int_{Q_n} (\partial_t u\, (v_h^n + \theta_n\, h_n \partial_t v_h^n) + \nabla_x u \cdot \nabla_x v_h^n + \theta_n\, h_n \nabla_x u \cdot \nabla_x \partial_t v_h^n)\, dx\, dt \\
&+ \int_{\Sigma_{n-1}} [\![u]\!]^{n-1}\, v_{h,+}^{n-1}\, dx = \int_{Q_n} f\, (v_h^n + \theta_n\, h_n \partial_t v_h^n)\, dx\, dt,
\end{aligned}$$

for all $n = 2, \ldots, N$, and

$$a_{Q_1}(u, v_h) := \int_{Q_1} (\partial_t u\,(v_h^1 + \theta_1\, h_1 \partial_t v_h^1) + \nabla_x u \cdot \nabla_x v_h^1 + \theta_1\, h_1 \nabla_x u \cdot \nabla_x \partial_t v_h^1)\, dx\, dt$$
$$+ \int_{\Sigma_0} [\![u]\!]^0\, v_{h,+}^0\, dx = \int_{Q_1} f\,(v_h^1 + \theta_1\, h_1 \partial_t v_h^1)\, dx\, dt + \int_{\Sigma_0} u_0\, v_{h,+}^0\, dx.$$

Summing over all Q_n, we conclude that

$$a_h(u, v_h) = l_h(v_h), \quad \forall v_h \in V_{0h}, \tag{7.5}$$

where

$$a_h(u, v_h) = \sum_{n=1}^{N} a_{Q_n}(u, v_h)$$

and

$$l_h(v_h) = \sum_{n=1}^{N} \int_{Q_n} f\,(v_h^n + \theta_n\, h_n \partial_t v_h^n)\, dx\, dt + \int_{\Sigma_0} u_0\, v_{h,+}^0\, dx.$$

Now, the space-time dG IgA variational scheme for (7.1) reads as follows: find $u_h \in V_{0h}$ such that

$$a_h(u_h, v_h) = l_h(v_h), \quad \forall v_h \in V_{0h}. \tag{7.6}$$

Motivated by the definition of the bilinear form $a_h(\cdot, \cdot)$ in (7.6), we introduce the mesh-dependent dG norm

$$\|v\|_{dG} := \Big(\sum_{n=1}^{N} \Big(\|\nabla_x v\|_{L_2(Q_n)}^2 + \theta_n\, h_n\, \|\partial_t v\|_{L_2(Q_n)}^2 + \frac{1}{2} \|[\![v]\!]^{n-1}\|_{L_2(\Sigma_{n-1})}^2 \Big) + \frac{1}{2} \|v\|_{L_2(\Sigma_N)}^2 \Big)^{\frac{1}{2}}.$$

In the following, we recall some important properties of the IgA scheme (7.6) respectively the bilinear form $a_h(\cdot, \cdot)$. For the proofs, we refer to [95].

Lemma 7.2. *The bilinear form $a_h(\cdot, \cdot)$, defined in (7.6), is V_{0h}-elliptic, i.e.,*

$$a_h(v_h, v_h) \geq C_e \|v_h\|_{dG}^2, \quad \text{for } v_h \in V_{0h}, \tag{7.7}$$

where $C_e = 0.5$ for $\theta_n \leq C_{inv,0}^{-2}$, with the positive, h_n-independent constant $C_{inv,0}$ from the inverse inequality

$$\|v_h\|_{L_2(\Sigma_{n-1})}^2 \leq C_{inv,0} h_n^{-1} \|v_h\|_{L_2(Q_n)}^2$$

that holds for all $v_h \in V_h^n$, $n = 1, \ldots, N$.

The V_{0h}-ellipticity of the bilinear form $a_h(\cdot, \cdot)$ implies that there exists a unique solution to (7.5). In order to obtain a priori error estimates, we introduce the space $V_{0h,*} = V + V_{0h}$ endowed with the norm

$$\|v\|_{dG,*} := \Big(\|v\|_{dG}^2 + \sum_{n=1}^{N} (\theta_n h_n)^{-1} \|v\|_{L_2(Q_n)}^2 + \sum_{n=2}^{N} \|v_-^{n-1}\|_{L_2(\Sigma_{n-1})}^2 \Big)^{\frac{1}{2}}. \tag{7.8}$$

Lemma 7.3. *The boundedness inequality*

$$|a_h(u, v_h)| \leq C_b \|u\|_{dG,*} \|v_h\|_{dG}$$

holds for all $u \in V_{0h,*}$ and $v_h \in V_{0h}$, where the constant $C_b = \max(C_{inv,1}\, \theta_{max}, 2)$, with $\theta_{max} = \max_n\{\theta_n\} \leq C_{inv,0}^{-2}$ and the positive, h_n-independent constant $C_{inv,1}$ from the inverse inequalities

$$\|\partial_t \partial_{x_i} v_h\|^2_{L_2(Q_n)} \leq C_{inv,1} h_n^{-2} \|\partial_{x_i} v_h\|^2_{L_2(Q_n)}$$

that holds for all $v_h \in V_h^n$, $n = 1, \ldots, N$, $i = 1, \ldots, d$.

Theorem 7.4. *Let u and u_h solve (7.2) and (7.6), respectively. Under the regularity Assumption 6, there exists a positive generic constant C, which is independent of $h = \max(h_n)$, such that*

$$\|u - u_h\|_{dG} \leq C(h^{\ell-1} + h^{m-\frac{1}{2}})\, \|u\|_{H^{\ell,m}(Q)}. \tag{7.9}$$

provided that $p + 1 \geq \max(l, m)$.

Remark 7.5. *We remark that, for the case of highly smooth solutions, i.e., $p + 1 \leq \min(\ell, m)$, estimate (7.9) takes the form*

$$\|u - u_h\|_{dG} \leq C\, h^p\, \|u\|_{H^{p+1,p+1}(Q)}.$$

7.1.2 Efficient Matrix Assembly

We recall the discrete variational problem given in (7.6), where we want to find $u_h \in V_{0h}$ such that

$$a_h(u_h, v_h) = \langle F_h, v_h \rangle, \quad \forall v_h \in V_{0h},$$

with $V_{0h} := V_{0h}^1 \times \ldots \times V_{0h}^N$ and

$$a_h(u_h, v_h) = \sum_{n=1}^{N} a_{Q_n}(u_h, v_h).$$

The local bilinear form for each space-time slab Q_n is given by

$$\begin{aligned}
a_{Q_n}(u_h, v_h) &= \int_{Q_n} \partial_t\, u_h^n\, (v_h^n + \theta_n\, h_n \partial_t v_h^n) + \nabla_x\, u_h^n \cdot \nabla_x (v_h^n + \theta_n\, h_n \partial_t v_h^n)\, dx\, dt \\
&\quad + \int_{\Sigma_{n-1}} [\![u_h]\!]^{n-1}\, v_{h,+}^{n-1}\, ds \\
&= \int_{Q_n} \partial_t\, u_h^n\, (v_h^n + \theta_n\, h_n \partial_t v_h^n) + \nabla_x\, u_h^n \cdot \nabla_x (v_h^n + \theta_n\, h_n \partial_t v_h^n)\, dx\, dt \\
&\quad + \int_{\Sigma_{n-1}} u_{h,+}^{n-1}\, v_{h,+}^{n-1}\, ds - \int_{\Sigma_{n-1}} u_{h,-}^{n-1}\, v_{h,+}^{n-1}\, ds \\
&=: b_{Q_n}(u_h^n, v_h^n) - \int_{\Sigma_{n-1}} u_{h,-}^{n-1}\, v_{h,+}^{n-1}\, ds,
\end{aligned}$$

where $n = 1, \dots, N$. For the local spaces V_{0h}^n defined by (7.4), we now introduce the simpler notation φ_j^n for the B-Spline basis functions such that

$$V_{0h}^n = \operatorname{span}\{\varphi_j^n\}_{j=1}^{N_n}$$

for $n = 1, \dots, N$. Once the basis is chosen, from the IgA variational scheme (7.6), we immediately obtain the linear system

$$\boldsymbol{L}_h \boldsymbol{u}_h := \begin{pmatrix} \boldsymbol{A}_1 & & & & \\ -\boldsymbol{B}_2 & \boldsymbol{A}_2 & & & \\ & \ddots & \ddots & & \\ & & & -\boldsymbol{B}_N & \boldsymbol{A}_N \end{pmatrix} \begin{pmatrix} \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \\ \vdots \\ \boldsymbol{u}_N \end{pmatrix} = \begin{pmatrix} \boldsymbol{f}_1 \\ \boldsymbol{f}_2 \\ \vdots \\ \boldsymbol{f}_N \end{pmatrix} =: \boldsymbol{f}_h, \tag{7.10}$$

with the matrices

$$\boldsymbol{A}_n[i,j] := b_{Q_n}(\varphi_j^n, \varphi_i^n) \quad \text{for } i,j = 1, \dots, N_n$$

on the diagonal for $n = 1, \dots, N$, and the matrices

$$\boldsymbol{B}_n[i,k] := \int_{\Sigma_{n-1}} \varphi_{k,-}^{n-1}\, \varphi_{i,+}^{n-1}\, ds \quad \text{for } k = 1, \dots, N_{n-1} \text{ and } i = 1, \dots, N_n$$

on the lower off diagonal for $n = 2, \dots, N$. Moreover, for $n = 1, \dots, N$ the right hand sides are given by

$$\boldsymbol{f}_n[i] := l_h(\varphi_i^n), \quad i = 1, \dots, N_n.$$

If the geometrical mappings $G_n : \overline{\widehat{Q}} \to \overline{Q}_n, n = 1, \dots, N$, preserve the tensor product structure of the IgA basis functions φ_i^n, we can use this information to save assembling time and storage costs for the linear system (7.10). In this case, we can write the basis functions φ_i^n in the form

$$\varphi_i^n(x,t) = \phi_{i_x}^n(x)\psi_{i_t}^n(t) \quad \text{with } i_x \in \{1, \dots, N_{n,x}\} \text{ and } i_t \in \{1, \dots, N_{n,t}\},$$

where $N_n = N_{n,x}N_{n,t}$. Using this representation, we can write the matrices $\boldsymbol{A}_n$ as

$$\boldsymbol{A}_n = \boldsymbol{K}_{n,t} \otimes \boldsymbol{M}_{n,x} + \boldsymbol{M}_{n,t} \otimes \boldsymbol{K}_{n,x}, \quad n = 1, \dots, N \tag{7.11}$$

with the standard mass and stiffness matrices with respect to space

$$\boldsymbol{M}_{n,x}[i_x, j_x] := \int_\Omega \phi_{j_x}^n \phi_{i_x}^n\, dx, \qquad \boldsymbol{K}_{n,x}[i_x, j_x] := \int_\Omega \nabla_x \phi_{j_x}^n \cdot \nabla_x \phi_{i_x}^n\, dx,$$

where $i_x, j_x = 1, \dots, N_{n,x}$, and the corresponding matrices with respect to time

$$\begin{aligned} \boldsymbol{K}_{n,t}[i_t, j_t] &:= \int_{t_{n-1}}^{t_n} \partial_t \psi_{j_t}^n (\psi_{i_t}^n + \theta_n h_n \partial_t \psi_{i_t}^n)\, dt + \psi_{j_t}^n(t_{n-1})\psi_{i_t}^n(t_{n-1}), \\ \boldsymbol{M}_{n,t}[i_t, j_t] &:= \int_{t_{n-1}}^{t_n} \psi_{j_t}^n (\psi_{i_t}^n + \theta_n h_n \partial_t \psi_{i_t}^n)\, dt, \end{aligned} \tag{7.12}$$

with $i_t, j_t = 1, \dots, N_{n,t}$. The matrices on the off diagonal $\boldsymbol{B}_n, n = 2, \dots, N$, can be written in the form

$$\boldsymbol{B}_n := \boldsymbol{N}_{n,t} \otimes \widetilde{\boldsymbol{M}}_{n,x},$$

with the matrices

$$\widetilde{\boldsymbol{M}}_{n,x}[i_x, k_x] := \int_\Omega \phi_{k_x}^{n-1} \phi_{i_x}^n \, dx \quad \text{and} \quad \boldsymbol{N}_{n,t}[i_t, k_t] := \psi_{k_t}^{n-1}(t_{n-1}) \psi_{i_t}^n(t_{n-1}),$$

where $i_x = 1, \dots, N_{n,x}$, $k_x = 1, \dots, N_{n-1,x}$, $i_t = 1, \dots, N_{n,t}$ and $k_t = 1, \dots, N_{n-1,t}$.

7.2 Solvers for Space-Time Problems

This section aims at the development of efficient solvers for the huge space-time system (7.10). Our new solver is based on the time-parallel Multigrid method proposed in [68], see also the PhD thesis [170]. The key point in realizing the method efficiently is the application of the smoother, which is the most costly part of the algorithm. The goal is to utilize the structure of the involved matrix $\boldsymbol{A}_n^{-1}$, which then allows for a faster application.

Remark 7.6. *The approach presented in this section is not restricted to a continuous Galerkin discretization for the spatial domains. The approach immediately extends to a spatial coupling of the interface dofs via dG terms as in Section 2.3.2.*

7.2.1 Space-time Multigrid

In this section, we give an review of the time parallel Multigrid from [170]. As already mentioned in Section 6.1.1, Multigrid consists of three main ingredients: the coarse grid solver, the smoother, and the prolongation/restriction operators. The coarse grid solver will be either a direct solver or an iterative solver based on the IETI-DP method.

Concerning the restriction and prolongation operator it is advantageous to consider coarsening in space and in time separately. The prolongation is then just defined as the transposed operator. The coarsening in space is performed as described in Section 6.1.1, whereas the coarsening in time realized by combining two successive time-slabs to a single time-slab, see the illustration in Figure 7.2. For simplicity, we assume that the considered basis is identical on each patch, which greatly simplifies the calculation of the restriction operator. Certainly, it is possible to extend the method the case, where the bases are not identical. In the easiest version, we only perform coarsening in time, which is proven to converge without restrictions on the mesh-size h, the parameter θ or polynomial degree p. The drawback of this strategy is that

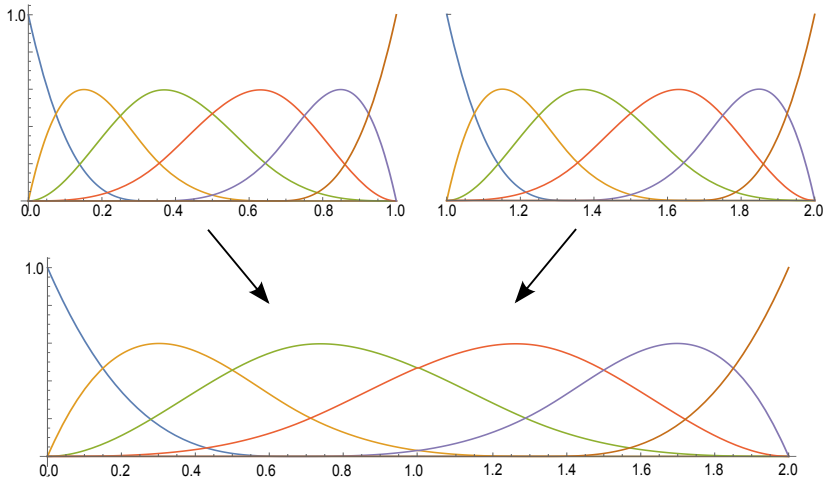


Figure 7.2: Illustration of time coarsening: two consecutive time-slabs are merged into a single time-slab.

on the coarsest level, the number of dofs may be still be quite large, especially if we consider relatively fine meshes for three-dimensional spatial domains. The more advanced strategy is to combine coarsening in space and time, but this has to be performed in a suitable way. In [170], for a one dimensional spatial domain, it is shown that the factor $\tau_L h_L^{-2}$ must be large enough in order to guarantee convergence, where τ_L is the mesh-size on the time-slab on level L in time direction and h_L the corresponding spatial mesh size on level L. Hence, we have to fulfil an inequality of the form $\tau_L h_L^{-2} > c_{critical}$, where the threshold $c_{critical}$ is independent of h, τ and L. Based on this inequality, we have to decide at each time-level, whether we perform a full space-time coarsening or, only a coarsening in time.

In this work, we are mostly interested in the smoother, that is an (inexact) damped block Jacobi smoother of the form, i.e.,

$$\boldsymbol{u}_h^{k+1} = \boldsymbol{u}_h^k + \omega \boldsymbol{D}_h^{-1} \left[\boldsymbol{f}_h - \boldsymbol{L}_h \boldsymbol{u}_h^k \right] \quad \text{for } k = 1, 2, \ldots .$$

We use the block diagonal matrix $\boldsymbol{D}_h := \operatorname{diag}\{\boldsymbol{A}_n\}_{n=1}^N$ and the damping parameter $\omega = 0.5$, see also [68]. The application of the smoother can be accelerated by replacing the inverse of $\boldsymbol{D}_h$ by some approximation, i.e., an approximation $\hat{\boldsymbol{A}}_n^{-1}$ to $\boldsymbol{A}_n^{-1}$. The aim of this work is to find a procedure, that allows us an efficient application of $\hat{\boldsymbol{A}}_n^{-1}$ to a vector. In order to achieve this, we will heavily exploiting the special tensor structure of $\boldsymbol{A}_n$.

7.2.2 General Construction of an Approximation for $\boldsymbol{A}_n^{-1}$

In this section, for notational simplicity, we drop the subscript n when considering matrices and vectors defined on the space-time-slab Q_n. We recall the structure of the

matrix $\boldsymbol{A}$,

$$\boldsymbol{A} = \boldsymbol{K}_t \otimes \boldsymbol{M}_x + \boldsymbol{M}_t \otimes \boldsymbol{K}_x,$$

where the matrices $\boldsymbol{M}_x$ and $\boldsymbol{K}_x$ are symmetric and positive definite, while the matrices $\boldsymbol{K}_t$ and $\boldsymbol{M}_t$ are non-symmetric, cf. (7.11). The matrices $\boldsymbol{M}_x$ and $\boldsymbol{K}_x$ correspond to a d-dimensional spatial problem, whereas $\boldsymbol{K}_t$ and $\boldsymbol{M}_t$ are only related to a one dimensional problem at one time-slab. Hence, the size of the latter two matrices is much smaller than the first two. The idea is to use already available preconditioners for symmetric and positive definite problems of the form $\boldsymbol{K}_x + \gamma \boldsymbol{M}_x$ with $\gamma > 0$ to construct efficient and robust preconditioners for $\boldsymbol{A}^{-1}$. The ideas of this section are based on the results developed in [211] and [192].

We will achieve this by performing a decomposition of $\boldsymbol{M}_t^{-1}\boldsymbol{K}_t$ using one of the following three methods: *Diagonalization*, *Complex-Schur decomposition*, *Real-Schur decomposition*. Using these techniques, we obtain a decomposition of the form $\boldsymbol{M}_t^{-1}\boldsymbol{K}_t = \boldsymbol{X}^{-1}\boldsymbol{Z}\boldsymbol{X}$, where the entries of the matrices $\boldsymbol{X}$ and $\boldsymbol{Z}$ are complex or real numbers, and $\boldsymbol{Z}$ has some sort of "simple" structure. A detailed specification will be presented in Section 7.2.3, Section 7.2.4 and Section 7.2.5.

By defining $\boldsymbol{Y} := (\boldsymbol{M}_t\boldsymbol{X})^{-1}$, we obtain the following representations

$$\boldsymbol{M}_t = \boldsymbol{Y}^{-1}\boldsymbol{X}^{-1} \quad \text{and} \quad \boldsymbol{K}_t = \boldsymbol{Y}^{-1}\boldsymbol{Z}\boldsymbol{X}^{-1}.$$

Now we can rewrite $\boldsymbol{A}$ in the form

$$\begin{aligned}\boldsymbol{A} &= \boldsymbol{K}_t \otimes \boldsymbol{M}_x + \boldsymbol{M}_t \otimes \boldsymbol{K}_x \\ &= (\boldsymbol{Y}^{-1}\boldsymbol{Z}\boldsymbol{X}^{-1}) \otimes \boldsymbol{M}_x + (\boldsymbol{Y}^{-1}\boldsymbol{X}^{-1}) \otimes \boldsymbol{K}_x \\ &= (\boldsymbol{Y}^{-1} \otimes \boldsymbol{I}) \cdot (\boldsymbol{Z} \otimes \boldsymbol{M}_x + \boldsymbol{I} \otimes \boldsymbol{K}_x) \cdot (\boldsymbol{X}^{-1} \otimes \boldsymbol{I}).\end{aligned}$$

Using the well-known fact that $(\boldsymbol{Y}^{-1} \otimes \boldsymbol{I})^{-1} = \boldsymbol{Y} \otimes \boldsymbol{I}$ and $(\boldsymbol{X}^{-1} \otimes \boldsymbol{I})^{-1} = \boldsymbol{X} \otimes \boldsymbol{I}$, we obtain

$$\boldsymbol{A}^{-1} = (\boldsymbol{X} \otimes \boldsymbol{I}) \cdot (\boldsymbol{Z} \otimes \boldsymbol{M}_x + \boldsymbol{I} \otimes \boldsymbol{K}_x)^{-1} \cdot (\boldsymbol{Y} \otimes \boldsymbol{I}). \tag{7.13}$$

In the subsequent subsections, we will investigate the structure of the matrix $(\boldsymbol{Z} \otimes \boldsymbol{M}_x + \boldsymbol{I} \otimes \boldsymbol{K}_x)$ for each of the three decomposition methods, and we will look for efficient ways of (approximate) inversion.

In the following, the generalized eigenvalues $\lambda_i := \alpha_i + \imath\beta_i \in \mathbb{C}$ of $(\boldsymbol{K}_t, \boldsymbol{M}_t)$, i.e.,

$$\boldsymbol{K}_t \boldsymbol{z}_i = \lambda_i \boldsymbol{M}_t \boldsymbol{z}_i, \tag{7.14}$$

with the eigenvector $\boldsymbol{z} := \boldsymbol{x} + \imath\boldsymbol{y}$, will play an important role for constructing an efficient application of (7.13). First of all, for $0 < \theta_n \leq C_{inv}^{-2}$, where C_{inv} denotes the constant from the inverse inequality

$$|v(t_{n-1})|^2 \leq C_{inv}^2 h_n^{-1} \|v\|^2_{L_2(t_{n-1},t_n)} \; \forall v \leftrightarrow \boldsymbol{v} \in \mathbb{R}^{N_t}, \tag{7.15}$$

we have the positiveness of the matrices $\boldsymbol{K}_t$ and $\boldsymbol{M}_t$, see also [225] for an explicit formula of $C_{inv} = C_{inv}(p)$ in the case of polynomials of the degree p.

Lemma 7.7. *Let $\boldsymbol{K}_t$ and $\boldsymbol{M}_t$ be given by (7.12), and let the constant $C_{inv} > 0$ be defined according to (7.15). If $\theta_n > 0$, then the matrix $\boldsymbol{K}_t$ is positive definite, i.e., $\boldsymbol{v}^T \boldsymbol{K}_t \boldsymbol{v} > 0$ for all $\boldsymbol{v} \in \mathbb{R}^{N_t} \setminus \{\boldsymbol{0}\}$, and if $\theta_n < 2C_{inv}^{-2}$, then the matrix $\boldsymbol{M}_t$ is positive definite.*

Proof. We first consider the matrix $\boldsymbol{K}_t$. We can write $\boldsymbol{v}^T \boldsymbol{K}_t \boldsymbol{v}$ in the following way:

$$\begin{aligned}
\boldsymbol{v}^T \boldsymbol{K}_t \boldsymbol{v} = (\boldsymbol{K}_t \boldsymbol{v}, \boldsymbol{v}) &= \int_{t_{n-1}}^{t_n} (v'(t)v(t) + \theta_n h_n (v'(t))^2)\, dt + |v(t_{n-1})|^2 \\
&= \theta_n h_n \|v'\|^2_{L_2(t_{n-1},t_n)} + \frac{1}{2}\int_{t_{n-1}}^{t_n} (v^2)'(t)\, dt + |v(t_{n-1})|^2 \\
&= \theta_n h_n \|v'\|^2_{L_2(t_{n-1},t_n)} + \frac{1}{2}|v(t_n)|^2 - \frac{1}{2}|v(t_{n-1})|^2 + |v(t_{n-1})|^2 \\
&= \theta_n h_n \|v'\|^2_{L_2(t_{n-1},t_n)} + \frac{1}{2}(|v(t_n)|^2 + |v(t_{n-1})|^2) > 0.
\end{aligned}$$

for all $v \leftrightarrow \boldsymbol{v} \in \mathbb{R}^{N_t} \setminus \{\boldsymbol{0}\}$. Using (7.15), we similarly obtain

$$\begin{aligned}
\boldsymbol{v}^T \boldsymbol{M}_t \boldsymbol{v} = (\boldsymbol{M}_t \boldsymbol{v}, \boldsymbol{v}) &= \int_{t_{n-1}}^{t_n} (v(t)^2 + \theta_n h_n v'(t)v(t))\, dt \\
&= \|v\|^2_{L_2(t_{n-1},t_n)} + \frac{1}{2}\theta_n h_n (|v(t_n)|^2 - |v(t_{n-1})|^2) \\
&\geq \left(1 - \frac{C_{inv}^2 \theta_n}{2}\right) \|v\|^2_{L_2(t_{n-1},t_n)} + \frac{1}{2}\theta_n h_n |v(t_n)|^2 > 0.
\end{aligned}$$

for all $v \leftrightarrow \boldsymbol{v} \in \mathbb{R}^{N_t} \setminus \{\boldsymbol{0}\}$. □

Next we are going to investigate the generalized eigenvalues in (7.14). More precisely, we want to find conditions under which the real part α is positive. However, for a generalized eigenvalue problem $\boldsymbol{A}\boldsymbol{z} = \lambda \boldsymbol{B}\boldsymbol{z}$, this does not follow from the positivity of $\boldsymbol{A}$ and $\boldsymbol{B}$ as following example shows.

Example 1. Let the matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ be given by

$$\boldsymbol{A} = \begin{bmatrix} 5 & -2 \\ 13 & 18 \end{bmatrix} \quad \text{and} \quad \boldsymbol{B} = \begin{bmatrix} 4 & 10 \\ -10 & 9 \end{bmatrix}.$$

For the spectra, we have $\sigma(\boldsymbol{A}) = \{9 \pm 2\sqrt{5}\imath\}$ and $\sigma(\boldsymbol{B}) = \{\frac{13}{2} \pm 5\sqrt{15}\imath\}$. However, the generalized eigenvalues are $\sigma(\boldsymbol{B}^{-1}\boldsymbol{A}) = \{-\frac{103}{272} \pm \sqrt{4435}\imath\}$.

Let $\boldsymbol{z}$ be the eigenvector to the eigenvalue $\lambda = \alpha + \imath\beta$, i.e., $(\boldsymbol{A} - \lambda \boldsymbol{B})\boldsymbol{z} = 0$. Multiplying from the left with $(\boldsymbol{x} - \imath \boldsymbol{y})^T$ yields

$$(\boldsymbol{x} - \imath \boldsymbol{y})^T (\boldsymbol{A} - (\alpha + \imath\beta)\boldsymbol{B})(\boldsymbol{x} + \imath \boldsymbol{y}) = 0.$$

Separating the real and imaginary part, we obtain

$$\begin{aligned} \alpha(\boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} + \boldsymbol{y}^T\boldsymbol{B}\boldsymbol{y}) - \beta(\boldsymbol{x}^T(\boldsymbol{B} - \boldsymbol{B}^T)\boldsymbol{y}) &= \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} + \boldsymbol{y}^T\boldsymbol{A}\boldsymbol{y} \\ \alpha(\boldsymbol{x}^T(\boldsymbol{B} - \boldsymbol{B}^T)\boldsymbol{y}) + \beta(\boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} + \boldsymbol{y}^T\boldsymbol{B}\boldsymbol{y}) &= \boldsymbol{x}^T(\boldsymbol{A} - \boldsymbol{A}^T)\boldsymbol{y}. \end{aligned} \tag{7.16}$$

Introducing the abbreviations $a := \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} + \boldsymbol{y}^T\boldsymbol{A}\boldsymbol{y}$, $b := \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} + \boldsymbol{y}^T\boldsymbol{B}\boldsymbol{y}$, $c := \boldsymbol{x}^T(\boldsymbol{B} - \boldsymbol{B}^T)\boldsymbol{y}$ and $d := \boldsymbol{x}^T(\boldsymbol{A} - \boldsymbol{A}^T)\boldsymbol{y}$, we can rewrite this system in the compact form

$$\begin{bmatrix} b & -c \\ c & b \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} a \\ d \end{bmatrix},$$

and α is then given by the formula

$$\alpha = \frac{1}{b^2 + c^2}(ab + cd). \tag{7.17}$$

We can easily observe the statements of the following lemma.

Lemma 7.8. *Let $\boldsymbol{A}$ and $\boldsymbol{B}$ be positive definite matrices, then the following statements hold:*

1. *$a > 0$ and $b > 0$*
2. *If $\beta = 0$, i.e., the eigenvalue $\lambda \in \mathbb{R}$, then $\lambda = \alpha > 0$.*
3. *If either $\boldsymbol{A}$ or $\boldsymbol{B}$ are symmetric, then $\alpha > 0$.*

If $\boldsymbol{A}$ is only non-negative, then these inequalities hold with $\geq$ instead of $>$.

Proof. The positivity of a and b immediately follows from the definition. If the eigenvalue λ is real, i.e., $\beta = 0$, we obtain from the first equation of (7.16) that $\alpha = a/b > 0$. If either $\boldsymbol{A}$ or $\boldsymbol{B}$ is symmetric, then either d or c is zero. Hence, by (7.17), α is positive. □

Let us now consider the special case of $\boldsymbol{A} = \boldsymbol{K}_t$ and $\boldsymbol{B} = \boldsymbol{M}_t$. For notational simplicity we consider the interval $[0, T]$. First we observe that

$$\begin{aligned} c = \boldsymbol{x}^T(\boldsymbol{B} - \boldsymbol{B}^T)\boldsymbol{y} &= \theta h \int_0^T y'(t)x(t) - x'(t)y(t)\,dt \\ d = \boldsymbol{x}^T(\boldsymbol{A} - \boldsymbol{A}^T)\boldsymbol{y} &= \int_0^T x'(t)y(t) - y'(t)x(t)\,dt. \end{aligned}$$

Hence, it follows that $c = -\theta h d$. This relation leads to the following formula for α:

$$\alpha = \frac{1}{b^2 + c^2}(ab - \theta h d^2). \tag{7.18}$$

The problem then reduces to check the relation $ab - \theta h d^2 > 0$, which then reads as

$$(\boldsymbol{x}^T \boldsymbol{A}\boldsymbol{x} + \boldsymbol{y}^T \boldsymbol{A}\boldsymbol{y})(\boldsymbol{x}^T \boldsymbol{B}\boldsymbol{x} + \boldsymbol{y}^T \boldsymbol{B}\boldsymbol{y}) - \theta h (x^T(\boldsymbol{A} - \boldsymbol{A}^T)\boldsymbol{y})^2 > 0, \tag{7.19}$$

for the eigenvector $\boldsymbol{z} = \boldsymbol{x} + \imath \boldsymbol{y}$ corresponding to $\lambda = \alpha + \imath\beta$. Rewriting (7.19) in terms of functions, we get the relation

$$\begin{aligned}
&\left(\theta_n h_n \|x'\|^2 + \frac{1}{2}(|x(T)|^2 + |x(0)|^2) + \theta_n h_n \|y'\|^2 + \frac{1}{2}(|y(T)|^2 + |y(0)|^2)\right) \\
&\cdot \left(\|x\|^2 + \frac{1}{2}\theta_n h_n (|x(T)|^2 - |x(0)|^2) + \|y\|^2 + \frac{1}{2}\theta_n h_n (|y(T)|^2 - |y(0)|^2)\right) \\
&- \theta h \left(\int_0^T x'(t)y(t) - y'(t)x(t)\, dt\right)^2 > 0
\end{aligned}$$

Unfortunately, in this work, we cannot give a complete characterization of the conditions under which the last inequality holds.

Let us consider the special case $\theta = 0$. First of all, we note that $\boldsymbol{v}^T \boldsymbol{K}_t \boldsymbol{v} = \frac{1}{2}(|v(t_{n-1})|^2 + |v(t_n)|^2)$, which then only defines a semi-norm. Hence, discrete coercivity is not valid. Therefore, this case is not covered by the analysis presented in [95]. For its analysis, we refer to [206], where an inf-sup condition and error estimates are proven. The matrix $\boldsymbol{M}_t$ is symmetric and $\boldsymbol{v}^T \boldsymbol{M}_t \boldsymbol{v} = \|v\|_{L^2}^2$. From this fact, we can deduce the following statement by means of Lemma 7.8:

Proposition 7.9. *Let $\boldsymbol{K}_t$ and $\boldsymbol{M}_t$ be as defined above with $\theta = 0$. Then $\alpha \geq 0$.*

Remark 7.10. *In the condition number analysis of the following subsections, we consider matrices of the form $\boldsymbol{K}_x + \alpha \boldsymbol{M}_x$, which are required to be positive definite. Therefore, the positivity of α can be relaxed in the case that $|\Gamma_D| > 0$.*

Remark 7.11. *A more detailed investigation of* (7.16) *shows that*

$$\alpha = 0 \Longleftrightarrow x(0) = x(T) = y(0) = y(T) = 0, \tag{7.20}$$

for the eigenvector $\boldsymbol{z} = \boldsymbol{x} + \imath \boldsymbol{y}$ corresponding to $\alpha + \imath\beta$.

For the case $p = 1$, we an even show that, for an eigenvector corresponding to an purely imaginary eigenvalue, the property $x(0) = x(T) = y(0) = y(T) = 0$ cannot hold. Considering a uniform knot vector in $[0, 1]$ with B-Splines of degree $p = 1$ and $N_t \geq 3$, it holds

$$\boldsymbol{K}_t = \frac{1}{2}\begin{bmatrix} 1 & 1 & & & & \\ -1 & 0 & 1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & -1 & 0 & 1 & \\ & & & -1 & 0 & 1 \\ & & & & -1 & 1 \end{bmatrix} \quad \textit{and} \quad \boldsymbol{M}_t = C_n \begin{bmatrix} 2 & 1 & & & & \\ 1 & 4 & 1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & 1 & 4 & 1 & \\ & & & 1 & 4 & 1 \\ & & & & 1 & 2 \end{bmatrix},$$

where $C_n > 0$ depends on N_t. Rewriting $\boldsymbol{K}_t\boldsymbol{z} = \imath\beta\boldsymbol{M}_t\boldsymbol{z}$ as recurrence relation for $\boldsymbol{z} = [z_1, z_2, \ldots, z_{N_t-1}, z_{N_t}]$, we obtain

$$\begin{aligned} z_1 + z_2 &= \imath\beta(2z_1 + z_2), \\ -z_{i-1} + z_{i+1} &= \imath\beta(z_{i-1} + 4z_i + z_{i+1}) \quad i = 2, \ldots, N_t - 1, \\ -z_{N_t-1} + z_{N_t} &= \imath\beta(z_{N_t-1} + 2z_{N_t}), \end{aligned} \tag{7.21}$$

where we put the real number C_n and the $1/2$ in front of $\boldsymbol{K}_t$ into the eigenvalue $\imath\beta$. In order to ensure that $\boldsymbol{z} = [0, z_2, \ldots, z_{N_t-1}, 0]$ is an eigenvector, it must fulfil the relation

$$z_2 = \imath\beta z_2 \Leftrightarrow (1 - \imath\beta)z_2 = 0,$$

which results from the first line of (7.21). Since $(1-\imath\beta)$ cannot be zero, it follows that $z_2 = 0$. Considering now the second line of (7.21) and assuming $z_1 = \ldots = z_j = 0$, we observe that

$$z_{j+1} = \imath\beta z_{j+1} \Leftrightarrow (1 - \imath\beta)z_{j+1} = 0,$$

for $i = j$. Therefore, $z_{j+1} = 0$. By induction, it follows that $z = 0$. Hence, it cannot be an eigenvector.

In the case of $p > 1$, the matrices $\boldsymbol{K}_t$ and $\boldsymbol{M}_t$ have more than one off diagonal and such a relation would not follow so easily. Numerical experiments in Section 7.3.2 indicate that the real part of λ is positive for the case $p > 1$ too.

Remark 7.12. *Let us consider the case $|\Gamma_D| > 0$. From Remark 7.10, Proposition 7.9 and the continuous dependence of α on θ, we obtain that $\boldsymbol{K}_x + \alpha\boldsymbol{M}_x$ must be positive definite for sufficiently small θ.*

Remark 7.13. *Numerical experiments performed for various values of θ, p and h_n in Section 7.3.2 indicate that the generalized eigenvalues λ_i have a positive real part α provided that the real part of the eigenvalues of $\boldsymbol{M}_t$ is positive. Moreover, in the practical implementation, one has to compute the eigenvalues λ_i anyway. Therefore, we always have an a-posteriori control on the positivity of α. If it happens that $\alpha \leq 0$, than we have to use a smaller θ.*

7.2.3 Diagonalization

If the matrix $\boldsymbol{M}_t^{-1}\boldsymbol{K}_t$ is diagonalizable, the eigenvalue decomposition allows us to write

$$\boldsymbol{M}_t^{-1}\boldsymbol{K}_t = \boldsymbol{X}^{-1}\boldsymbol{D}\boldsymbol{X}, \tag{7.22}$$

where $\boldsymbol{D} = \mathrm{diag}(\lambda_i)$, $\lambda_i \in \mathbb{C}$, is a diagonal matrix with possibly complex eigenvalues on the diagonal, and $\boldsymbol{X} \in \mathbb{C}^{N_t \times N_t}$ denotes the matrix of the possibly complex eigenvectors.

Due to the fact that the matrix $\boldsymbol{M}_t^{-1}\boldsymbol{K}_t$ is non-symmetric, the eigenvectors do not form an orthogonal basis, i.e. $\boldsymbol{X}^{-1} \neq \boldsymbol{X}^*$. An efficient calculation can be performed by means of solving the generalized eigenvalue problem $\boldsymbol{K}_t x = \lambda \boldsymbol{M}_t x$.

Thanks to (7.22), the matrix $(\boldsymbol{Z} \otimes \boldsymbol{M}_x + \boldsymbol{I} \otimes \boldsymbol{K}_x)^{-1}$ from (7.13) then takes the form

$$(\boldsymbol{Z} \otimes \boldsymbol{M}_x + \boldsymbol{I} \otimes \boldsymbol{K}_x)^{-1} = (\boldsymbol{D} \otimes \boldsymbol{M}_x + \boldsymbol{I} \otimes \boldsymbol{K}_x)^{-1} = \mathrm{diag}_{i=1,\dots,N_t}((\boldsymbol{K}_x + \lambda_i \boldsymbol{M}_x)^{-1}).$$

Therefore, only N_t problems of the form $(\boldsymbol{K}_x + \lambda_i \boldsymbol{M}_x)$ have to be solved, independently of each other. We have to distinguish two cases: the first case where the eigenvalue λ_i is a positive real number, and the second one where λ_i is a complex number.

In the first case, we consider $\lambda_i = \alpha_i \in \mathbb{R}^+$. In this case the matrix $\boldsymbol{K}_x + \lambda_i \boldsymbol{M}_x$ is symmetric and positive definite. Linear algebraic systems with a SPD system matrix $\boldsymbol{K}_x + \lambda_i \boldsymbol{M}_x$ can efficiently be solved by different robust solution strategies like Multigrid [103], [204], [102], [47], Domain Decomposition type methods [90], [23], [18] or Fast Diagonalization type methods [192], [211].

The second case, where $\lambda_i = \alpha + \imath\beta \in \mathbb{C}$ with $\alpha, \beta \in \mathbb{R}, \alpha > 0$, is more difficult to handle. We note that $(\boldsymbol{K}_x + \lambda_i \boldsymbol{M}_x)^* \neq \boldsymbol{K}_x + \lambda_i \boldsymbol{M}_x$. Separating the real and imaginary parts, we can rewrite the complex system $(\boldsymbol{K}_x + \lambda_i \boldsymbol{M}_x)\boldsymbol{z} = \boldsymbol{h}$ as a real system with a real block system matrix of twice size.

$$\begin{aligned}
&(\boldsymbol{K}_x + \lambda_i \boldsymbol{M}_x)\boldsymbol{z} = \boldsymbol{h} \\
\iff &\begin{bmatrix} \boldsymbol{K}_x + \alpha_i \boldsymbol{M}_x & -\beta \boldsymbol{M}_x \\ \beta \boldsymbol{M}_x & \boldsymbol{K}_x + \alpha_i \boldsymbol{M}_x \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{g} \end{bmatrix} \\
\iff &\underbrace{\begin{bmatrix} \boldsymbol{K}_x + \alpha_i \boldsymbol{M}_x & \beta_i \boldsymbol{M}_x \\ \beta_i \boldsymbol{M}_x & -(\boldsymbol{K}_x + \alpha_i \boldsymbol{M}_x) \end{bmatrix}}_{=:\overline{\boldsymbol{A}}_i} \begin{bmatrix} \boldsymbol{x} \\ -\boldsymbol{y} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{g} \end{bmatrix},
\end{aligned}$$

where $\boldsymbol{z} = \boldsymbol{x} + \imath\boldsymbol{y}$ and $\boldsymbol{h} = \boldsymbol{f} + \imath\boldsymbol{g}$. The matrix $\overline{\boldsymbol{A}}_i \in \mathbb{R}^{2N_x \times 2N_x}$ is symmetric, but indefinite. We are now looking for an robust preconditioner for $\overline{\boldsymbol{A}}_i$. In order to construct such a preconditioner, we use operator interpolation technique, see, e.g., [237], [27] and [1]. We follow the presentation in [229]. First, we need the definition of the geometric mean of two operators and the general operator interpolation theorem, see Definition. 2.28 and Theorem. 2.29 in [229] and references therein.

Definition 7.14. *Let $\boldsymbol{A}$ and $\boldsymbol{B}$ be real, symmetric and positive definite matrices. We define the geometric mean of $\boldsymbol{A}$ and $\boldsymbol{B}$ by the relation*

$$[\boldsymbol{A}, \boldsymbol{B}]_{1/2} = \boldsymbol{A}^{1/2}\big(\boldsymbol{A}^{-1/2}\boldsymbol{B}\boldsymbol{A}^{-1/2}\big)^{1/2}\boldsymbol{A}^{1/2}.$$

Moreover, for any $\vartheta \in [0,1]$, we define the symmetric and positive definite matrix by

$$[\boldsymbol{A}, \boldsymbol{B}]_{\vartheta} = \boldsymbol{A}^{1/2}\big(\boldsymbol{A}^{-1/2}\boldsymbol{B}\boldsymbol{A}^{-1/2}\big)^{\vartheta}\boldsymbol{A}^{1/2}.$$

Theorem 7.15. *Let $\mathcal{A} : \mathbb{R}^n \to \mathbb{R}^n$ such that the inequalities*

$$\underline{c}_0\|\boldsymbol{u}\|_{X_0} \leq \|\mathcal{A}\boldsymbol{u}\|_{Y_0} \leq \overline{c}_0\|\boldsymbol{u}\|_{X_0} \quad \text{and} \quad \underline{c}_1\|\boldsymbol{u}\|_{X_1} \leq \|\mathcal{A}u\|_{Y_1} \leq \overline{c}_1\|\boldsymbol{u}\|_{X_1} \quad \forall \boldsymbol{u} \in \mathbb{R}^n$$

hold, where the linear vector spaces $X_j = \mathbb{R}^n$ and $Y_j = \mathbb{R}^n$ with $j \in \{0, 1\}$ are equipped with the norms $\|\cdot\|_{X_j}$ and $\|\cdot\|_{Y_j}$, which are associated to the inner products

$$(\boldsymbol{u}, \boldsymbol{v})_{X_j} = (\boldsymbol{M}_j\boldsymbol{u}, \boldsymbol{v})_{\ell_2} \quad \text{and} \quad (\boldsymbol{u}, \boldsymbol{v})_{Y_j} = (\boldsymbol{N}_j\boldsymbol{u}, \boldsymbol{v})_{\ell_2},$$

given by the symmetric and positive definite matrices $\boldsymbol{M}_0, \boldsymbol{M}_1, \boldsymbol{N}_0$ and $\boldsymbol{N}_1$. Then, for $X_\vartheta = [X_0, X_1]_\vartheta$ and $Y_\vartheta = [Y_0, Y_1]_\vartheta$, with $\vartheta \in [0, 1]$, the inequalities

$$\underline{c}_0^{1-\vartheta}\underline{c}_1^{\vartheta}\|\boldsymbol{u}\|_{X_\vartheta} \leq \|\mathcal{A}\boldsymbol{u}\|_{Y_\vartheta} \leq \overline{c}_0^{1-\vartheta}\overline{c}_1^{\vartheta}\|\boldsymbol{u}\|_{X_\vartheta} \quad \forall \boldsymbol{u} \in \mathbb{R}^n. \tag{7.23}$$

hold, where the norms $\|\cdot\|_{X_\vartheta}$ and $\|\cdot\|_{Y_\vartheta}$ are the norms associated to the inner products

$$\begin{aligned}(\boldsymbol{u}, \boldsymbol{v})_{X_\vartheta} &= (\boldsymbol{M}_\vartheta\boldsymbol{u}, \boldsymbol{v})_{\ell_2}, \quad \text{with} \quad \boldsymbol{M}_\vartheta = [\boldsymbol{M}_0, \boldsymbol{M}_1]_\vartheta, \quad \text{and} \\ (\boldsymbol{u}, \boldsymbol{v})_{Y_\vartheta} &= (\boldsymbol{N}_\vartheta\boldsymbol{u}, \boldsymbol{v})_{\ell_2}, \quad \text{with} \quad \boldsymbol{N}_\vartheta = [\boldsymbol{N}_0, \boldsymbol{N}_1]_\vartheta,\end{aligned}$$

respectively.

Proof. For the proof, we refer to the proof of Theorem 2.29 in [229] and references therein, see also [1]. □

Remark 7.16. *Using the notation from Theorem 7.15, one can show the alternative representation*

$$\|\boldsymbol{u}\|_{X_\vartheta}^2 = \frac{2\sin(\vartheta\pi)}{\pi}\int_0^\pi t^{-(2\vartheta+1)}K(t;\boldsymbol{u})^2\,dt$$

of $\|\boldsymbol{u}\|_{X_\vartheta}$, where $K(t;\boldsymbol{x}) = \inf_{\boldsymbol{x}=\boldsymbol{x}_0+\boldsymbol{x}_1}(\|\boldsymbol{x}_0\|_{X_0}^2 + t^2\|\boldsymbol{x}_1\|_{X_1}^2)^{1/2}$. From this representation, we observe that

$$[X_0, X_1]_\vartheta = [X_1, X_0]_{1-\vartheta}. \tag{7.24}$$

Let us consider a general saddle-point matrix

$$\mathcal{A} = \begin{bmatrix}\boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{B}^T & -\boldsymbol{C}\end{bmatrix},$$

where $\boldsymbol{A}$ and $\boldsymbol{C}$ are symmetric and positive definite matrices. We can define two possible negative Schur complements

$$\boldsymbol{S} := \boldsymbol{C} + \boldsymbol{B}\boldsymbol{A}^{-1}\boldsymbol{B}^T \quad \text{and} \quad \boldsymbol{R} := \boldsymbol{A} + \boldsymbol{B}\boldsymbol{C}^{-1}\boldsymbol{B}, \tag{7.25}$$

and the associated block diagonal preconditioners

$$\boldsymbol{P}_0 := \begin{bmatrix}\boldsymbol{A} & 0 \\ 0 & \boldsymbol{S}\end{bmatrix} \quad \text{and} \quad \boldsymbol{P}_1 := \begin{bmatrix}\boldsymbol{R} & 0 \\ 0 & \boldsymbol{C}\end{bmatrix}.$$

For $\boldsymbol{P}_0$ and $\boldsymbol{P}_1$, the following spectral inequalities are known

$$(\sqrt{5}-1)/2\|\boldsymbol{u}\|_{P_j} \leq \|\mathcal{A}\boldsymbol{u}\|_{P_j^{-1}} \leq (\sqrt{5}+1)/2\|\boldsymbol{u}\|_{P_j} \quad j \in \{0,1\},$$

see Theorem 2.26 and Corollary 2.27 in [229] and references therein. Based on these two preconditioners, we construct a preconditioner $\boldsymbol{P}_\vartheta$ with $\vartheta = 1/2$ by an interpolation of the preconditioners $\boldsymbol{P}_0$ and $\boldsymbol{P}_1$:

$$\boldsymbol{P}_{1/2} = [\boldsymbol{P}_0, \boldsymbol{P}_1]_{1/2} = \begin{bmatrix} [\boldsymbol{A}, \boldsymbol{R}]_{1/2} & 0 \\ 0 & [\boldsymbol{S}, \boldsymbol{C}]_{1/2} \end{bmatrix}.$$

By means of Theorem 7.15 and the setting $\boldsymbol{M}_0 = \boldsymbol{P}_0, \boldsymbol{M}_1 = \boldsymbol{P}_1, \boldsymbol{N}_0 = \boldsymbol{P}_0^{-1}$ and $\boldsymbol{N}_1 = \boldsymbol{P}_1^{-1}$, it follows that

$$(\sqrt{5}-1)/2\|\boldsymbol{u}\|_{P_{1/2}} \leq \|\overline{\boldsymbol{A}}\boldsymbol{u}\|_{\boldsymbol{P}_{1/2}^{-1}} \leq (\sqrt{5}+1)/2\|\boldsymbol{u}\|_{P_{1/2}}.$$

Hence, $\kappa_{P_{1/2}}(\boldsymbol{P}_{1/2}^{-1}\mathcal{A}) \leq (\sqrt{5}+1)/(\sqrt{5}-1)$. Note, this condition number estimate would hold for all $\vartheta \in [0,1]$. In the following, we are looking for an approximation of $\boldsymbol{P}_{1/2}$, which can easily be realized in an implementation.

Theorem 7.17. *Let $\boldsymbol{K}_x$ and $\boldsymbol{M}_x$ be symmetric and positive definite matrices, and let α and β be real numbers with $\alpha > 0$. Furthermore, we define the block matrices*

$$\overline{\boldsymbol{A}} := \begin{bmatrix} \boldsymbol{K}_x + \alpha\boldsymbol{M}_x & \beta\boldsymbol{M}_x \\ \beta\boldsymbol{M}_x & -(\boldsymbol{K}_x + \alpha\boldsymbol{M}_x) \end{bmatrix}, \tag{7.26}$$

$$\boldsymbol{P} := \begin{bmatrix} \boldsymbol{K}_x + (\alpha + |\beta|)\boldsymbol{M}_x & 0 \\ 0 & \boldsymbol{K}_x + (\alpha + |\beta|)\boldsymbol{M}_x \end{bmatrix}. \tag{7.27}$$

Then the condition number estimate

$$\kappa_P(\boldsymbol{P}^{-1}\overline{\boldsymbol{A}}) \leq \sqrt{2}\frac{\sqrt{5}+1}{\sqrt{5}-1} \tag{7.28}$$

holds.

Proof. The proof follows the lines in [229], Section 3.3. For simplicity, we introduce the notations $\mathcal{K} := \boldsymbol{K}_x + \alpha\boldsymbol{M}_x$ and $\mathcal{M} := \boldsymbol{M}_x$. Recall the system matrix

$$\overline{\boldsymbol{A}} = \begin{bmatrix} \boldsymbol{K}_x + \alpha\boldsymbol{M}_x & \beta\boldsymbol{M}_x \\ \beta\boldsymbol{M}_x & -(\boldsymbol{K}_x + \alpha\boldsymbol{M}_x) \end{bmatrix} = \begin{bmatrix} \mathcal{K} & \beta\mathcal{M} \\ \beta\mathcal{M} & -\mathcal{K} \end{bmatrix}.$$

Since $\mathcal{K}$ is symmetric and, due to $\alpha > 0$, also positive definite, we can reformulate the two Schur complements from (7.25) for the matrix $\overline{A}$ as follows:

$$\boldsymbol{S} = \boldsymbol{R} = \mathcal{K} + \beta^2\mathcal{M}\mathcal{K}^{-1}\mathcal{M}.$$

We are looking for an spectral equivalent approximation $\boldsymbol{P}$ of $\boldsymbol{P}_{1/2}$, which is easy to realize and fulfils the spectral inequalities

$$\underline{c}\boldsymbol{P} \leq \boldsymbol{P}_{1/2} \leq \overline{c}\boldsymbol{P}, \tag{7.29}$$

where the constants $\underline{c}$ and $\overline{c}$ are independent of α and β. Next we estimate $[\mathcal{K}, \boldsymbol{R}]_{1/2}$ and $[\boldsymbol{S}, \mathcal{K}]_{1/2}$. Here we make use of the following matrix inequalities

$$\frac{1}{\sqrt{2}}(\sqrt{a}\boldsymbol{I} + \sqrt{b}\boldsymbol{X}^{1/2}) \leq (a\boldsymbol{I} + b\boldsymbol{X})^{1/2} \leq \sqrt{a}\boldsymbol{I} + \sqrt{b}\boldsymbol{X}^{1/2}, \tag{7.30}$$

where $\boldsymbol{X}$ is a symmetric and positive definite matrix, and $\boldsymbol{I}$ denotes the identity matrix. First we derive an upper bound for $[\mathcal{K}, R]_{1/2}$:

$$\begin{aligned}
[\mathcal{K}, \boldsymbol{R}]_{1/2} &= \mathcal{K}^{1/2}\big(\mathcal{K}^{-1/2}\boldsymbol{R}\mathcal{K}^{-1/2}\big)^{1/2}\mathcal{K}^{1/2} \\
&= \mathcal{K}^{1/2}\big(\mathcal{K}^{-1/2}(\mathcal{K} + \beta^2\mathcal{M}\mathcal{K}^{-1}\mathcal{M})\mathcal{K}^{-1/2}\big)^{1/2}\mathcal{K}^{1/2} \\
&= \mathcal{K}^{1/2}\big(I + \beta^2\mathcal{K}^{-1/2}\mathcal{M}\mathcal{K}^{-1}\mathcal{M}\mathcal{K}^{-1/2}\big)^{1/2}\mathcal{K}^{1/2} \\
&\leq \mathcal{K}^{1/2}\big(I + (\beta^2\mathcal{K}^{-1/2}\mathcal{M}\mathcal{K}^{-1}\mathcal{M}\mathcal{K}^{-1/2})^{1/2}\big)\mathcal{K}^{1/2} \\
&= \mathcal{K} + |\beta|\mathcal{K}^{1/2}(\mathcal{K}^{-1/2}\mathcal{M}\mathcal{K}^{-1}\mathcal{M}\mathcal{K}^{-1/2})^{1/2}\mathcal{K}^{1/2} \\
&= \mathcal{K} + |\beta|\mathcal{K}^{1/2}(\mathcal{K}^{-1/2}\mathcal{M}\mathcal{K}^{-1/2})^{1/2}(\mathcal{K}^{-1/2}\mathcal{M}\mathcal{K}^{-1/2})^{1/2}\mathcal{K}^{1/2} \\
&= \mathcal{K} + |\beta|\mathcal{K}^{1/2}(\mathcal{K}^{-1/2}\mathcal{M}\mathcal{K}^{-1/2})\mathcal{K}^{1/2} \\
&= \mathcal{K} + |\beta|\mathcal{M}.
\end{aligned}$$

Similarly, for the lower bound, we obtain

$$\begin{aligned}
[\mathcal{K}, \boldsymbol{R}]_{1/2} &= \mathcal{K}^{1/2}\big(\mathcal{K}^{-1/2}\boldsymbol{R}\mathcal{K}^{-1/2}\big)^{1/2}\mathcal{K}^{1/2} \\
&= \mathcal{K}^{1/2}\big(I + \beta^2\mathcal{K}^{-1/2}\mathcal{M}\mathcal{K}^{-1}\mathcal{M}\mathcal{K}^{-1/2}\big)^{1/2}\mathcal{K}^{1/2} \\
&\geq \mathcal{K}^{1/2}\big(\frac{1}{\sqrt{2}}(I + (\beta^2\mathcal{K}^{-1/2}\mathcal{M}\mathcal{K}^{-1}\mathcal{M}\mathcal{K}^{-1/2})^{1/2})\big)\mathcal{K}^{1/2} \\
&= \frac{1}{\sqrt{2}}(\mathcal{K} + |\beta|\mathcal{K}^{1/2}(\mathcal{K}^{-1/2}\mathcal{M}\mathcal{K}^{-1}\mathcal{M}\mathcal{K}^{-1/2})^{1/2}\mathcal{K}^{1/2}) \\
&= \frac{1}{\sqrt{2}}(\mathcal{K} + |\beta|\mathcal{M}).
\end{aligned}$$

The missing estimate from above and below for $[\boldsymbol{S}, \mathcal{K}]_{1/2}$ follow from the fact that $[\boldsymbol{S}, \mathcal{K}]_{1/2} = [\mathcal{K}, \boldsymbol{S}]_{1/2} = [\mathcal{K}, \boldsymbol{R}]_{1/2}$, see (7.24). Hence, for the preconditioner

$$\boldsymbol{P} = \begin{bmatrix} \mathcal{K} + |\beta|\mathcal{M} & 0 \\ 0 & \mathcal{K} + |\beta|\mathcal{M} \end{bmatrix} = \begin{bmatrix} \boldsymbol{K}_x + (\alpha + |\beta|)\boldsymbol{M}_x & 0 \\ 0 & \boldsymbol{K}_x + (\alpha + |\beta|)\boldsymbol{M}_x \end{bmatrix},$$

we obtain the spectral constants $\underline{c} = \frac{1}{\sqrt{2}}$ and $\overline{c} = 1$ in (7.29). Finally, we arrive at the estimate

$$\kappa_P(\boldsymbol{P}^{-1}\overline{\boldsymbol{A}}) = \|\boldsymbol{P}^{-1}\overline{\boldsymbol{A}}\|_P \|\overline{\boldsymbol{A}}^{-1}P\|_P \leq \sqrt{2}\|\boldsymbol{P}_{1/2}^{-1}\overline{\boldsymbol{A}}\|_{P_{1/2}} \|\overline{\boldsymbol{A}}^{-1}\boldsymbol{P}_{1/2}\|_{P_{1/2}} \leq \sqrt{2}\frac{\sqrt{5}+1}{\sqrt{5}-1}. \tag{7.31}$$

□

Remark 7.18. *The estimate* (7.31) *of the condition number* $\kappa_P(\boldsymbol{P}^{-1}\overline{\boldsymbol{A}})$ *can be improved by solving the generalized eigenvalue problem*

$$\overline{\boldsymbol{A}} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} = \lambda \boldsymbol{P} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix}.$$

directly. Following the procedure outlined in Remark 9 in [237], see also the proof of Theorem 7.19, we find that the generalized eigenvalues satisfy the estimates

$$|\lambda_{min}| \geq \frac{1}{\sqrt{2}} \quad \textit{and} \quad |\lambda_{max}| \leq 1,$$

which leads to the improved condition number estimate $\kappa_P(\boldsymbol{P}^{-1}\overline{\boldsymbol{A}}) \leq \sqrt{2}$.

We note that both block-diagonal entries of $\boldsymbol{P}$ are identical, and the matrix $\boldsymbol{K}_x + (\alpha + |\beta|)\boldsymbol{M}_x$ is symmetric and positive definite. This opens various possibilities for preconditioning based on standard techniques for symmetric and positive definite matrices. We can solve the linear system with system matrix $\overline{\boldsymbol{A}}$, e.g., by means of MinRes preconditioned by $\boldsymbol{P}^{-1}$. We can even use an spectral equivalent approximation $\hat{\boldsymbol{P}}^{-1}$, i.e., $c\hat{\boldsymbol{P}}^{-1} \leq \boldsymbol{P}^{-1} \leq C\hat{\boldsymbol{P}}^{-1}$, with constants c and C, independent of α and β. Moreover, this approach allows for a further parallelization by applying $\boldsymbol{A}_n$ in parallel for $n = 1, \ldots, N_t$.

Unfortunately, this approach has a severe drawback. Due to the fact that the matrix $\boldsymbol{M}_t^{-1}\boldsymbol{K}_t$ is non-symmetric, the matrix $\boldsymbol{X}$ of eigenvectors is not unitary and, therefore, $\kappa(\boldsymbol{X}) \neq 1$. Actually, numerical tests in Section 7.3.1 show that, for large B-Spline degree or small h_t, we observe that the condition number $\kappa(\boldsymbol{X}) \approx 10^{12}$. In that case we cannot correctly apply (7.13) and the algorithm fails. This problem can be circumvented by using the Complex or Real-Schur decomposition, as presented in the subsequent two subsections.

7.2.4 Complex-Schur Decomposition

In this section, we investigate an alternative possibility for decomposing $\boldsymbol{M}_t^{-1}\boldsymbol{K}_t$. The Complex-Schur decomposition provides a decomposition of the form

$$\boldsymbol{M}_t^{-1}\boldsymbol{K}_t = \boldsymbol{Q}^*\boldsymbol{T}\boldsymbol{Q}, \tag{7.32}$$

where $\boldsymbol{Q} \in \mathbb{C}^{N_t \times N_t}$, and $\boldsymbol{T} \in \mathbb{C}^{N_t \times N_t}$ is a upper triangular matrix with $T_{ii} = \lambda_i$. The advantage of the (Complex) Schur decomposition is the fact that we obtain a unitary matrix $\boldsymbol{Q}$. Hence, $\kappa(\boldsymbol{Q}) = 1$, but the diagonal matrix $\boldsymbol{D}$ in the decomposition (7.22) is now replaced by the upper triangular matrix $\boldsymbol{T}$ in the decomposition (7.32), By means of (7.32), the matrix $\boldsymbol{Z} \otimes \boldsymbol{M}_x + \boldsymbol{I} \otimes \boldsymbol{K}_x$ from (7.13) takes the form

$$
\begin{aligned}
(\boldsymbol{Z} \otimes \boldsymbol{M}_x + \boldsymbol{I} \otimes \boldsymbol{K}_x)^{-1} &= (\boldsymbol{T} \otimes \boldsymbol{M}_x + \boldsymbol{I} \otimes \boldsymbol{K}_x)^{-1} \\
&= \begin{bmatrix} \boldsymbol{K}_x + T_{11}\boldsymbol{M}_x & T_{12}\boldsymbol{M}_x & \dots & \\ 0 & \boldsymbol{K}_x + T_{22}\boldsymbol{M}_x & T_{23}\boldsymbol{M}_x & \\ \vdots & 0 & \ddots & T_{N_tN_t-1}\boldsymbol{M}_x \\ 0 & \dots & 0 & \boldsymbol{K}_x + T_{N_tN_t}\boldsymbol{M}_x \end{bmatrix}^{-1} \\
&= \begin{bmatrix} \boldsymbol{K}_x + \lambda_1\boldsymbol{M}_x & T_{12}\boldsymbol{M}_x & \dots & \\ 0 & \boldsymbol{K}_x + \lambda_2\boldsymbol{M}_x & T_{23}\boldsymbol{M}_x & \\ \vdots & 0 & \ddots & T_{N_tN_t-1}\boldsymbol{M}_x \\ 0 & \dots & 0 & \boldsymbol{K}_x + \lambda_{N_t}\boldsymbol{M}_x \end{bmatrix}^{-1}
\end{aligned}
$$

The application of $(\boldsymbol{T} \otimes \boldsymbol{M}_x + \boldsymbol{I} \otimes \boldsymbol{K}_x)^{-1}$ to some vector f can be performed staggered way as presented in Algorithm 8.

Algorithm 8 Calculation of $\boldsymbol{y} = (\boldsymbol{T} \otimes \boldsymbol{M}_x + \boldsymbol{I} \otimes \boldsymbol{K}_x)^{-1}\boldsymbol{f}$

```
for i = N_t, N_t − 1 ..., 1 do
    g = f_i
    for j = i + 1, i + 2 ..., N_t do
        g = g − T_ij y_j
    end for
    Solve (K_x + λ_i M_x) y_i = g, where λ_i = T_ii.
end for
return y
```

In order to solve the linear systems $(\boldsymbol{K}_x + \lambda_i \boldsymbol{M}_x)\boldsymbol{y}_i = \boldsymbol{g}_i, i = 1, \dots, N_t$ in Algorithm 8, we can use the techniques developed in the previous subsection. This decomposition method allows us to have a well conditioned transformation matrix Q, however at the cost that the linear system cannot be solved independently of each other. We note that this method and the eigenvalue decomposition require complex arithmetic, which is more expensive than the real one. In the following subsection, we investigate the Real-Schur decomposition, which eliminates the need for having complex arithmetic.

7.2.5 Real-Schur Decomposition

In this subsection, we look at the decomposition of $\boldsymbol{M}_t^{-1}\boldsymbol{K}_t$ by means of the Real-Schur decomposition. It provides a decomposition of the form

$$\boldsymbol{M}_t^{-1}\boldsymbol{K}_t = \boldsymbol{Q}^*\boldsymbol{T}\boldsymbol{Q}, \tag{7.33}$$

where $\boldsymbol{Q} \in \mathbb{R}^{N_t \times N_t}$. The matrix $\boldsymbol{T} \in \mathbb{R}^{N_t \times N_t}$ is a upper quasi-triangular matrix, i.e., the diagonal consists of 1×1 and 2×2 blocks. The values of the 1×1 blocks correspond to the real eigenvalues, while the 2×2 blocks correspond to the complex eigenvalues of $\boldsymbol{M}_t^{-1}\boldsymbol{K}_t$.

By additionally performing a Givens rotation, the 2×2 block can be transformed to the structure

$$\begin{bmatrix} \alpha & \beta_1 \\ \beta_2 & \alpha \end{bmatrix},$$

where $\alpha, \beta_1, \beta_2 \in \mathbb{R}$ and $\beta_1 \neq \beta_2 \neq 0$. The eigenvalues of this matrix are given by $\alpha \pm \sqrt{\beta_1\beta_2}$. Due to the fact that the eigenvalues have to be complex and the real part has to be positive, we obtain that $\alpha > 0$ and β_1 and β_2 have different signs. Therefore, we can write the eigenvalues as $\alpha \pm \imath\sqrt{|\beta_1\beta_2|}$.

Using this decomposition, the matrix $\boldsymbol{Z} \otimes \boldsymbol{M}_x + \boldsymbol{I} \otimes \boldsymbol{K}_x$ appearing in (7.13) has a structure, which is similar to that one of the Complex-Schur decomposition. The corresponding system of linear algebraic equations can also again be solved in a staggered way as presented in Algorithm 8. One has to adapt the algorithm in such a way that, if the diagonal block is a 2×2 block, one has to work with two-block vectors and a 2×2 block matrix. It remains to investigate the solution strategy for the 2×2 block matrix. As already mentioned, the 2×2 block of T is non-symmetric. Hence, the 2×2 block matrix is also non-symmetric and is given in the following way

$$\begin{bmatrix} \boldsymbol{K}_x + \alpha\boldsymbol{M}_x & \beta_1\boldsymbol{M}_x \\ \beta_2\boldsymbol{M}_x & \boldsymbol{K}_x + \alpha\boldsymbol{M}_x \end{bmatrix}.$$

The structure of the matrix is very similar to $\overline{\boldsymbol{A}}$ in Theorem 7.17 up to the non-symmetry, which origins just from the different scalings β_1 and β_2 and their different sign. By a proper rescaling, we can transform this linear system into an equivalent system with a symmetric, but indefinite system matrix:

$$\begin{aligned} \begin{bmatrix} \boldsymbol{K}_x + \alpha\boldsymbol{M}_x & \beta_1\boldsymbol{M}_x \\ \beta_2\boldsymbol{M}_x & \boldsymbol{K}_x + \alpha\boldsymbol{M}_x \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} &= \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{g} \end{bmatrix} \\ \iff \begin{bmatrix} \boldsymbol{K}_x + \alpha\boldsymbol{M}_x & -\beta_1\boldsymbol{M}_x \\ \beta_2\boldsymbol{M}_x & -(\boldsymbol{K}_x + \alpha\boldsymbol{M}_x) \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ -\boldsymbol{y} \end{bmatrix} &= \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{g} \end{bmatrix} \\ \iff \underbrace{\begin{bmatrix} |\beta_2|(\boldsymbol{K}_x + \alpha\boldsymbol{M}_x) & -\beta_1|\beta_2|\boldsymbol{M}_x \\ |\beta_1|\beta_2\boldsymbol{M}_x & -|\beta_1|(\boldsymbol{K}_x + \alpha\boldsymbol{M}_x) \end{bmatrix}}_{=:\overline{\boldsymbol{A}}} \begin{bmatrix} \boldsymbol{x} \\ -\boldsymbol{y} \end{bmatrix} &= \begin{bmatrix} |\beta_2|\boldsymbol{f} \\ |\beta_1|\boldsymbol{g} \end{bmatrix}, \end{aligned}$$

We note that β_1 and β_2 have different signs. Hence, $-\beta_1|\beta_2| = -\beta_2|\beta_1|$. Motivated by the construction of the preconditioner in the case of the eigenvalue decomposition, we can come up with an optimal preconditioner. The following theorem presents this optimal preconditioner for the matrix $\overline{\boldsymbol{A}}$.

Theorem 7.19. *Let $\boldsymbol{K}_x$ and $\boldsymbol{M}_x$ be symmetric and positive matrices, and let α, β_1, β_2 be real numbers with $\alpha > 0$. Furthermore, we define the block matrices*

$$\overline{\boldsymbol{A}} := \begin{bmatrix} |\beta_2|(\boldsymbol{K}_x + \alpha\boldsymbol{M}_x) & -\beta_1|\beta_2|\boldsymbol{M}_x \\ |\beta_1|\beta_2\boldsymbol{M}_x & -|\beta_1|(\boldsymbol{K}_x + \alpha\boldsymbol{M}_x) \end{bmatrix},$$
$$\boldsymbol{P} := \begin{bmatrix} |\beta_2|(\boldsymbol{K}_x + (\alpha + \sqrt{|\beta_1\beta_2|})\boldsymbol{M}_x) & 0 \\ 0 & |\beta_1|(\boldsymbol{K}_x + (\alpha + \sqrt{|\beta_1\beta_2|})\boldsymbol{M}_x) \end{bmatrix}.$$

Then the condition number estimate

$$\kappa_P(\boldsymbol{P}^{-1}\overline{\boldsymbol{A}}) \leq \sqrt{2}.$$

holds.

Proof. The proof follows the lines from Remark 9 in [237], which gives a sharper bound than using interpolation theory as in [229]. For notational simplicity, we introduce the abbreviations $\mathcal{K} := \boldsymbol{K}_x + \alpha\boldsymbol{M}_x$ and $\mathcal{M} := \boldsymbol{M}_x$. We now investigate the generalized eigenvalue problem $\overline{\boldsymbol{A}}\boldsymbol{u} = \lambda\boldsymbol{P}\boldsymbol{u}$, which reads

$$\begin{bmatrix} |\beta_2|\mathcal{K} & -\beta_1|\beta_2|\mathcal{M} \\ |\beta_1|\beta_2\mathcal{M} & -|\beta_1|\mathcal{K} \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} = \lambda \begin{bmatrix} |\beta_2|(\mathcal{K} + \sqrt{|\beta_1\beta_2|}\mathcal{M}) & 0 \\ 0 & |\beta_1|(\mathcal{K} + \sqrt{|\beta_1\beta_2|}\mathcal{M}) \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix}. \tag{7.34}$$

At first we consider the following generalized eigenvalue problem

$$\mathcal{K}\boldsymbol{z} = \mu(\mathcal{K} + \sqrt{|\beta_1\beta_2|}\mathcal{M})\boldsymbol{z}.$$

Due to the fact that $\mathcal{K}$ and $\mathcal{M}$ are symmetric, there exists a basis $\{\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_{N_x}\}$ of eigenvectors, which are orthonormal with respect to the inner product generated by $\mathcal{K} + \sqrt{|\beta_1\beta_2|}\mathcal{M}$, and corresponding eigenvalues μ_j. Since $\mathcal{K}$ is dominated by $\mathcal{K} + \sqrt{|\beta_1\beta_2|}\mathcal{M}$ and due to their positivity, we have that $\mu_j \in [0, 1]$. Therefore, we can express $\boldsymbol{x}$ and $\boldsymbol{y}$ as a linear combination of $\boldsymbol{e}_j$ with coefficients $\hat{x}_j$ and $\hat{y}_j$, respectively. Moreover, $\mathcal{M}\boldsymbol{z}$ fulfils the following identity

$$\begin{aligned} \mathcal{M}\boldsymbol{z} &= (|\beta_1\beta_2|)^{-1/2}(\sqrt{|\beta_1\beta_2|}\mathcal{M} + \mathcal{K})\boldsymbol{z} - (|\beta_1\beta_2|)^{-1/2}\mathcal{K}\boldsymbol{z} \\ &= (|\beta_1\beta_2|)^{-1/2}(\sqrt{|\beta_1\beta_2|}\mathcal{M} + \mathcal{K})\boldsymbol{z} - (|\beta_1\beta_2|)^{-1/2}\mu(\mathcal{K} + \sqrt{|\beta_1\beta_2|}\mathcal{M})\boldsymbol{z} \\ &= (|\beta_1\beta_2|)^{-1/2}(1-\mu)(\sqrt{|\beta_1\beta_2|}\mathcal{M} + \mathcal{K})\boldsymbol{z}. \end{aligned}$$

Using the expansion of $\boldsymbol{x}$ and $\boldsymbol{y}$ into the eigenvectors $\{\boldsymbol{e}_j\}$, system (7.34) decomposes into the 2×2 systems

$$\begin{bmatrix} |\beta_2|\mu_j & -\beta_1|\beta_2||\beta_1\beta_2|^{-1/2}(1-\mu_j) \\ |\beta_1|\beta_2|\beta_1\beta_2|^{-1/2}(1-\mu_j) & -|\beta_1|\mu_j \end{bmatrix} \begin{bmatrix} \hat{x}_j \\ \hat{y}_j \end{bmatrix} = \lambda \begin{bmatrix} |\beta_2| & 0 \\ 0 & |\beta_1| \end{bmatrix} \begin{bmatrix} \hat{x}_j \\ \hat{y}_j \end{bmatrix}.$$

Since there exists at least one pair $(\hat{x}_j, \hat{y}_j)$ which is non-zero, the determinant of the system matrix must be zero, i.e.,

$$\det\left(\begin{bmatrix} |\beta_2|\mu_j & -\beta_1|\beta_2||\beta_1\beta_2|^{-1/2}(1-\mu_j) \\ |\beta_1|\beta_2|\beta_1\beta_2|^{-1/2}(1-\mu_j) & -|\beta_1|\mu_j \end{bmatrix} - \lambda \begin{bmatrix} |\beta_2| & 0 \\ 0 & |\beta_1| \end{bmatrix}\right) = 0,$$

which reduces to

$$|\beta_1\beta_2|(\lambda^2 - \mu_j^2) - (\beta_1\beta_2)^2|\beta_1\beta_2|^{-1}(1-\mu_j)^2 = 0,$$

where we used that $-\beta_1|\beta_2| = |\beta_1|\beta_2 \neq 0$. We immediately obtain that $|\lambda| = \sqrt{\mu_j^2 + (1-\mu_j)^2}$ for $\mu_i \in [0,1]$, and it follows that $\frac{1}{\sqrt{2}} \leq |\lambda| \leq 1$, which gives the desired bound on the condition number of $\boldsymbol{P}^{-1}\overline{\boldsymbol{A}}$. □

Now we can again use the MinRes preconditioned by $\boldsymbol{P}$ as iterative solver for systems with the system matrix $\overline{\boldsymbol{A}}$, and we obtain a robust method. Moreover, due to the use of real arithmetic, this approach is usually more efficient than that one using the Complex-Schur decomposition.

7.2.6 IETI-DP for Time Slices – Parallelization in Space

In this section, we discussion two options for using IETI-DP as a preconditioner for the problems related to a time-slab Q_n. The first options uses IETI-DP as preconditioner for the spatial problems $\boldsymbol{K}_x + \gamma\boldsymbol{M}_x$, appearing in the previous sections. For such problems, we have the theoretical foundation that the condition number of the preconditioned system is of order $O((1+\log(H/h))^2)$. The second approach uses IETI-DP directly as preconditioner for the non-symmetric space-time slab problem $\boldsymbol{A}_n$. In order to realize additional parallelization in space, parallel matrix application is required. Moreover, in order to enable coarsening in space and time, parallel space-transfer operators are needed.

IETI-DP for the Spatial Problems $\boldsymbol{K} + \gamma\boldsymbol{M}_x$

As already mentioned in the beginning of this section, the matrices $\boldsymbol{K} + \gamma\boldsymbol{M}_x$ appear as a result of decomposing the matrix $\boldsymbol{A}_n$. Either they are directly appearing as the system matrices, then γ equals an real eigenvalue of $\boldsymbol{M}_t^{-1}\boldsymbol{K}_t$, or they form the diagonal of a block diagonal preconditioner, where γ is a combination of the real and imaginary part, see Section 7.2.2. In either case, we can assume that $\gamma > 0$, see Remark 7.10 to Remark 7.13 for a discussion about the positivity. Therefore, the matrix $\boldsymbol{K} + \gamma\boldsymbol{M}_x$ is symmetric and positive definite which perfectly fits into the framework of the IETI-DP method. In Chapter 3 and Chapter 4, we proved that the condition number of the preconditioned system is bounded by $O((1+\log(H/h))^2)$ for case without mass matrix

$\boldsymbol{M}_x$. However, adding a positive multiple of the mass matrix does not affect on its condition number.

Concerning the realization, the IETI-DP method calculates a LL^T decomposition, of $\boldsymbol{K}^{(k)} + \gamma \boldsymbol{M}_x^{(k)}$ on each spatial patch $\Omega^{(k)}$. However, if γ varies, we have to compute the factorization for each different γ. From this perspective, Multigrid methods would have an advantage over the IETI-DP method. Moreover, the IETI-DP has some noticeable overhead, when applied only for a few iterations due to the transformation to the Schur complement formulation and vice-versa.

Regarding parallelization, using a space parallel method for these problems might not be very efficient in terms of communication. The reason is that their application appear in the innermost loop of the time-parallel MG algorithm. Hence, they have to be called very frequently on a relatively small number of dofs. This is especially problematic, when IETI-DP is used for preconditioning the matrix $\boldsymbol{P}$ as defined in Theorem 7.17 and Theorem 7.19.Therefore, we do not investigate the parallel scalability of this method in Section 7.3.

IETI-DP for the Space-Time Problem $\boldsymbol{A}_n$

Alternatively, one can apply IETI-DP directly to the non-symmetric problem $\boldsymbol{A}_n$ at the price of loosing much of the theoretical foundation. We refer to [219] for an analyis of BDDC for FE and to [214] for an analysis of the FETI method. Although, there are not much theoretical results, numerical tests show a good performance of the method, see, e.g., [120], [151] and references therein.

When adapting the algorithm to the non-symmetric case, we have to be careful when dealing with $S^{(k)}$-orthogonal decomposition of $\widetilde{W}$ into W_Π and W_Δ, see Section 3.2. We constructed the basis of W_Π in such a way that $W_\Pi \perp^S W_\Delta$. Since S is non-symmetric, this does not make sense, because $(S\cdot, \cdot)$ defines not a scalar product. However, we still can construct the basis Φ in such a way, that

$$\begin{bmatrix} S^{(k)} & C^{(k)T} \\ C^{(k)} & 0 \end{bmatrix} \begin{bmatrix} \widetilde{\phi}_j^{(k)} \\ \widetilde{\boldsymbol{\mu}}_j^{(k)} \end{bmatrix} = \begin{bmatrix} 0 \\ \boldsymbol{e}_j^{(k)} \end{bmatrix}, \quad \forall j \in \{1, \dots, n_\Pi^{(k)}\}.$$

The result is that $\widetilde{S}$ does not have block diagonal structure, but a upper triangular block structure, i.e.,

$$\widetilde{S} = \begin{bmatrix} \boldsymbol{S}_{\Pi\Pi} & S_{\Pi\Delta} \\ 0 & S_{\Delta\Delta} \end{bmatrix}.$$

The application of $S_{\Pi\Delta} v_\Delta$ can be realized by first applying S, and then extracting the primal component, i.e.,

$$\boldsymbol{f}_\Pi = \boldsymbol{A}\boldsymbol{\Phi}^T S v_\Delta,$$

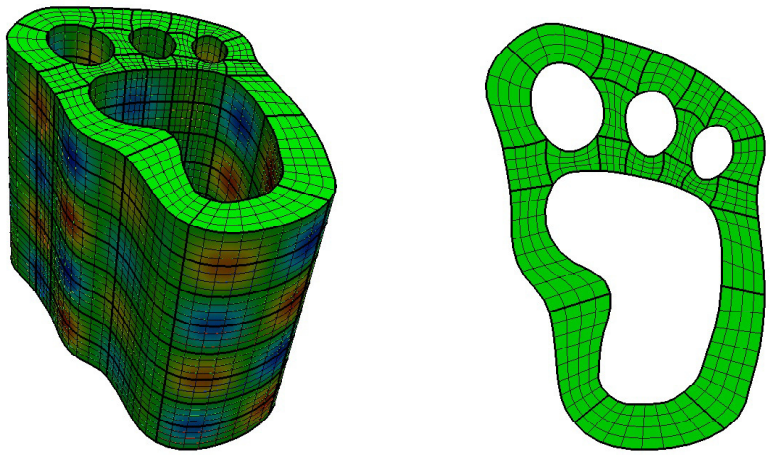


Figure 7.3: Left picture shows the space-time domain $Q = \Omega \times (0, T)$ with 8 time-slabs; right picture presents the spatial domain Ω, consisting of 21 patches.

where $\boldsymbol{A}$ is the assembling operator for the primal components and $\boldsymbol{\Phi}$ denotes the coefficient representation of the basis functions of W_Π, cf. Section 5.3.4.3. in [183]. The application of S requires the solution of a Dirichlet problem, see Section 3.2.4. Moreover, the final linear system (3.12) is non-symmetric and has to be solved by means of GMRes, see [191].

From a parallelization point of view, the most efficient strategy would probably be to use the techniques developed in Section 7.2.2 to efficiently solve the local problems in the IETI-DP method. No communication would be required to solve a local system. This approach would follow the idea of a coarse-grained parallelization, whereas the former corresponds to a fine-grained parallelization.

7.3 Numerical Examples

In this section, we test the proposed preconditioners on the three (2+1) dimensional space-time cylinder Q illustrated in Figure 7.3. The two-dimensional spatial domain Ω consists of 21 spatial subdomains (volumetric patches). For each time slab, we use conforming B-Splines of degree p. The problems were calculated on a Desktop PC with an Intel(R) Xeon(R) CPU E5-1650 v2 @ 3.50GHz and 16 GB main memory. We use the C++ library G+Smo for describing the geometry and performing the numerical tests, see also [111] and [162].

$N_t - p \setminus p$	2	3	4	5	6	7	8
2	64	309	362	766	1706	3907	9501
4	481	1036	3037	9419	41959	39323	73946
8	2869	16118	39693	74370	180054	472758	1e+06
16	34332	188263	463148	1e+06	6e+06	3e+07	1e+08
32	701306	2e+06	1e+07	6e+07	4e+08	7e+09	1e+10
64	5e+07	4e+07	3e+08	3e+09	6e+10	3e+11	1e+12
128	2e+08	1e+09	1e+10	3e+11	2e+13	5e+13	4e+14

Table 7.1: Condition number of $\boldsymbol{X}$: $\theta = 0.01$ and $|t_{n+1} - t_n| = 0.1$.

7.3.1 Condition Number of Eigenvectors

Here, we study the condition number κ of the generalized eigenvectors of $(\boldsymbol{K}_t, \boldsymbol{M}_t)$. Due to the non-symmetry of $\boldsymbol{K}_t$ and $\boldsymbol{M}_t$, we do not obtain an orthogonal basis of eigenvectors. Hence, the condition number is not 1. Actually, it can be quite large. We report on the condition number for different p and N_t in Table 7.1. We observe that the condition number grows exponentially with p and N_t. We conclude, that for small p or small number of dofs in time direction, the approach presented in Section 7.2.3 may be still feasible.

7.3.2 Smallest Real Part of the Eigenvalue of $\boldsymbol{M}_t^{-1}\boldsymbol{K}_t$

In Section 7.2.2, we observed the necessity that the smallest real part of the eigenvalues of $\boldsymbol{M}_t^{-1}\boldsymbol{K}_t$ is positive. In this section, we present numerical studies for different p, h and θ, where we fix the time interval to $[0, 1]$. The results are summarized in Table 7.2, where the entries with $*$ indicate that the matrix $\boldsymbol{M}_t$ had at least one eigenvalue with negative real part. Consequently, the smallest real part of the generalized eigenvalues was also negative. We observe that, if $\boldsymbol{M}_t > 0$, then also the real part of $\boldsymbol{M}_t^{-1}\boldsymbol{K}_t$ is positive. The positive real part of the eigenvalues for the $p = 1$ and $\theta = 0$ is in agreement with Remark 7.11. Moreover, for $\theta = 0$ and increasing p we observe even an increase of the smallest real part of the eigenvalues, cf. Proposition 7.9 and Remark 7.11. The numerical tests indicate that, for sufficiently small θ, the smallest real part of the generalized eigenvalues stays positive.

7.3.3 Condition Number of Preconditioned $\boldsymbol{K}_x + \lambda\boldsymbol{M}_x$

The aim of this section is to verify the optimal condition number bound presented in Theorem 7.17 and Theorem 7.19. To do so, we report on the maximum number of MinRes-iterations in order to solve $\boldsymbol{K}_x + \lambda_i\boldsymbol{M}_x$, where $\lambda_i \in \mathbb{C}$ are the generalized eigenvalues of $(\boldsymbol{K}_t, \boldsymbol{M}_t)$. We use zero initial guess, and a reduction of the initial

	2 uniform refinements							4 uniform refinements						
$\theta \backslash p$	1	2	3	4	5	6	7	1	2	3	4	5	6	7
0	1.5	2.4	3.2	3.8	4.3	4.7	5.0	0.2	0.5	0.9	1.5	2.1	2.7	3.4
0.01	1.6	2.5	3.2	3.6	4.0	4.4	4.9	0.7	0.7	1.1	1.6	2.2	2.8	3.3
0.1	2.5	2.9	3.2	3.6	4.0	4.5	5.2	4.8	2.9	2.7	3.0	3.4	3.6	4.1
1	4.1	4.5	4.7	*	*	*	*	12.4	12.0	9.2	*	*	*	*
10	4.6	5.2	5.2	*	*	*	*	6.7	11.8	*	*	*	*	*
	6 uniform refinements							8 uniform refinements						
$\theta \backslash p$	1	2	3	4	5	6	7	1	2	3	4	5	6	7
0	0.01	0.03	0.06	0.1	0.1	0.2	0.2	0.0008	0.002	0.004	0.006	0.009	0.01	0.02
0.01	1.9	1.0	0.8	0.7	0.6	0.6	0.6	7.7	4.0	3.0	2.5	2.0	1.8	1.6
0.1	18.6	9.9	7.4	6.0	5.1	4.5	4.0	34.8	33.8	29.5	23.8	20.0	17.2	15.1
1	34.2	35.1	33.8	*	*	*	*	34.8	34.4	34.5	*	*	*	*
10	11.4	17.4	*	*	*	*	*	29.0	32.2	*	*	*	*	*

Table 7.2: Smallest real part of generalized eigenvalues $\boldsymbol{K}_t x = \lambda \boldsymbol{M}_t x$ for different B-Spline degrees p, θ and number of dofs. The $*$ indicates that, the matrix $\boldsymbol{M}_t$ has at least one eigenvalue with negative real part.

	C-Schur decomp.					R-Schur decomp.				
ref. x and $t \setminus p$	2	3	4	5	6	2	3	4	5	6
0	23	22	26	26	26	18	18	20	21	22
1	25	24	24	27	26	20	20	22	22	22
2	25	25	25	27	27	22	22	22	22	22
3	24	26	26	27	27	22	22	22	22	21
4	25	25	26	27	26	22	22	22	22	20

Table 7.3: Maximum number of MinRes iterations to solve $\boldsymbol{K}_x + \lambda_i \boldsymbol{M}_x$, $i = 1, \dots, n_t$ resulting from the Complex- and Real-Schur decomposition. Refinement is performed uniformly in x and t.

residual by 10^{-10}. We choose $\theta = 0.1$. In Table 7.3, we investigate the robustness of the preconditioners from Theorem 7.17 and Theorem 7.19. We observe that the number of iterations stays bounded for various p and h.

7.3.4 Application to Space-Time Multigrid

This section deals with the use of the iterative methods developed in Section 7.2.2 as smoothers in the space-time Multigrid. The realization of the preconditioner P, see Theorem 7.17 and Theorem 7.19 is performed via a sparse direct solver. We use the PARDISO 5.0.0 Solver Project [140] for performing the LU factorizations. We compare the three different approaches, presented in Section 7.2.2, with the exact realization of $\boldsymbol{A}_n^{-1}$ via the sparse direct solver PARDISO. For approximating $\boldsymbol{A}_n^{-1}$ via MinRes, we use zero initial guess and a reduction of the initial residuum by 10^{-4}. In Table 7.4, we report on the single-core computation time of the MG algorithm to setup the data-structures and solve the system via the MG iteration. The setup time

#dofs	ref		#slabs	MG-It	Direct		Diag	
	x	t			Setup	Solving	Setup	Solving
15950	2	3	2	7	1.9	0.7	0.04	2.3
97020	3	3	4	7	38.6	8.5	0.3	19.4
665720	4	3	8	7	1008	94.6	3.7	183.8
#dofs	ref		#slabs	MG-It	C-Schur		R-Schur	
15950	2	3	2	7	0.05	2.4	0.04	1.3
97020	3	3	4	7	0.5	19.9	0.3	11.1
665720	4	3	8	7	5.4	187.3	3.7	108.0

Table 7.4: Comparison of the Eigenvalue, Complex-Schur and Real-Schur decomposition with a sparse direct solver used for approximating $\boldsymbol{A}_n^{-1}$. The timings are given in seconds.

includes the required LU factorizations, but not the assembling of the matrices, which is implemented as in Section 7.1.2 and does not give a significant contribution to the overall time. For the MG iteration, we use zero initial guess and a reduction of the initial residuum by 10^{-8}. We choose $\theta = 0.01$, $|t_n - t_{n+1}| = 0.1$ and the polynomial degree by $p = 3$ for both space and time directions. Moreover, we fix the number of dofs in time direction of a time slab, but increase the number of time slabs. The MG method uses coarsening in space as well as in time.

We observe that the LU factorization of $\boldsymbol{A}_n$ needs a quite large amount of time, whereas the setup time is almost negligible for the three preconditioners proposed. The little increase in the solution time definitely pays off by the small setup time. In addition, the Real-Schur decomposition almost provides the same solution time as the direct solver. Because of the complex arithmetic of the Diagonalization or the Complex-Schur decomposition, their computational effort doubles, which we observe also in the numerical test. Finally, due to the quite accurate approximation of $\boldsymbol{A}_n^{-1}$ (up to 10^{-4}), we do not observe a deterioration of the MG iteration numbers. It took around 12 MinRes-iterations to reach the desired tolerance of 10^{-4}.

7.3.5 Parallelization

We conclude the numerical examples with studies regarding the parallelization in space and time. The results are obtain on the RADON1[1] cluster at Linz. Each node is equipped with 2x Xeon E5-2630v3 "Haswell" CPU (8 Cores, 2.4Ghz, 20MB Cache) and 128 GB RAM. We consider a similar setup as in Section 7.3.4, i.e., we use B-Spline degree 3, perform two refinements in spatial direction, 3 in time direction, θ is given by 0.01 and $|t_n - t_{n-1}| = 0.1$. We perform coarsening in space and time. The systems with the system matrix $(\boldsymbol{K}_x + \gamma \boldsymbol{M}_x)$, $\gamma \in \mathbb{R}^+$ are solved by means of PARDISO, i.e.,

[1] https://www.ricam.oeaw.ac.at/hpc/

#dofs	#procs= #slabs	It.	Setup	Solving
48510	2	7	0.088	2.8
97020	4	7	0.092	3.1
194040	8	7	0.093	3.2
388080	16	7	0.093	3.3
776160	32	7	0.094	3.4
1552320	64	7	0.096	3.5
3104640	128	7	0.100	3.5
6209280	256	7	0.104	3.9

Table 7.5: Weak scaling results for parallelization in time, timings are given in seconds. Real-Schur decomposition is used to realize the application of $\boldsymbol{A}_n^{-1}$.

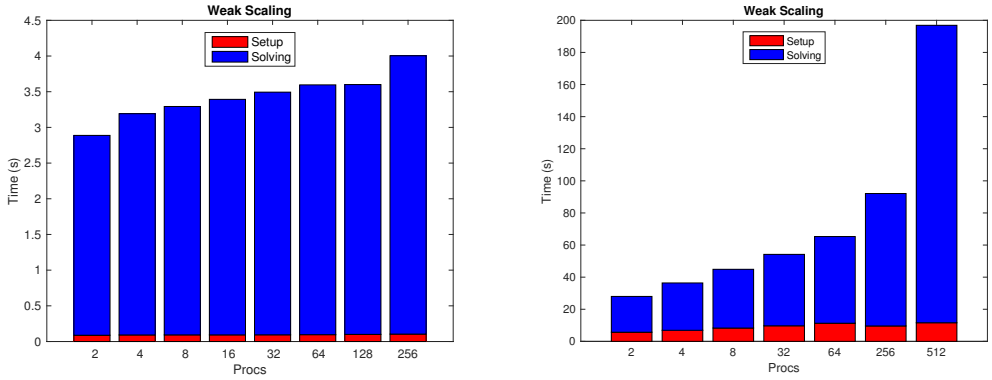


Figure 7.4: Visualization of the weak scaling results from Table 7.5 (left) and Table 7.6 (right).

a sparse direct solver. For the time-parallel MG method, we use a reduction of the initial residuum by 10^{-8} as stopping criterion, while a relative tolerance of 10^{-4} is used for the linear system in (7.13). First, we consider only parallelization in time and we use the same number of processors and time-slabs, i.e., each time-slab is associated with one processor. In Table 7.5, we report on the weak scaling results using the Real-Schur decomposition to realize the application of $\boldsymbol{A}_n^{-1}$. The results are visualized in Figure 7.4.

Next, we investigate the parallel scaling behaviour in space and time. We restrict ourselves to the second option discussed in Section 7.2.6, where we apply IETI-DP methods to the non-symmetric matrix $\boldsymbol{A}_n$ arising from the space-time sub-problem posed on the time-slabs. As already mentioned in Section 7.2.6, we do not perform experiments for other possibilities of realizing additional parallelization in space.

In order to eliminate the influence of load imbalances regarding different number of dofs on the patches, we use a simple square domain, decomposed into 4×4 patches as spatial domain. In order to approximate $\boldsymbol{A}_n^{-1}$, we perform 3 GMRes-IETI-DP iterations,

#dofs	ref x	#sl.	#proc.	c_x	c_t	It.	Setup	Solving
23276	2	4	2	1	2	8	5.7	22.3
46552	2	8	4	1	4	8	6.9	29.5
93104	2	16	8	1	8	8	8.3	36.6
267696	3	16	32	4	8	8	9.7	44.5
535392	3	32	64	4	16	8	11.3	54.0
1774432	4	32	256	16	16	7	9.6	82.5
3548864	4	64	512	16	32	10	11.6	185.3

Table 7.6: Weak scaling results for parallelization in space and time, timings are given in seconds. The IETI-DP method is used to realize the application of the space-time slab matrix $\boldsymbol{A}_n^{-1}$.

where we use a direct solver for the patch-local space-time problems. However, we use coarsening only in time direction. In Table 7.6, we report on the weak scaling results. The number of processors grow in the same way as the number of dofs, e.g., when performing uniform refinement in space, we increase the number of processors used for space parallelization by a factor of four. The number of processors used for parallelization in space and time are denoted by c_x and c_t, respectively.

We observe a increase of computation time, especially, when going from 32 to 64 time slabs. The reason is the missing coarsening in space, because we have to solve at each time-level still a large space-time problem. Moreover, the number of MG iterations increases, because the IETI-DP method is not completely robust with respect to the mesh-size h.

Chapter 8

Conclusion and Outlook

Conclusion

In this thesis, we investigated fast solvers for systems of linear algebraic equations arising from IgA of elliptic diffusion problems. In more detail, we considered the adaption of the FETI-DP method to IgA, called IETI-DP. We investigated the cases where we have a cG formulation with C^0 coupling across the patch interfaces, and a coupling using the dG method, which allows for discontinuities at the interfaces. In the interior of the domain, we assumed a smooth B-Spline basis of degree p, preferably with maximum smoothness C^{p-1}. This setting allowed an extension of the FETI-DP method for cG and dG formulations to IgA.

For two-dimensional domains we could prove quasi-optimal condition number bounds of the preconditioned cG- and dG-IETI-DP operator with respect to the ratio H/h of patch-diameter H and patch-local mesh-size h. Our analysis did not cover the dependence on the B-Spline degree p and on jumps of the diffusion coefficient α across patch interfaces. Nevertheless, numerical experiments for both methods showed at most a linear dependence (or even logarithmic) on p, and robustness with respect to jumps in α. In order to obtain a scalable algorithm also for three-dimensional domains, we implemented in addition edge and face averages as primal variables. All primal variables were incorporated by means of constraints. We applied the dG-IETI-DP also to domains, where gap and overlapping regions are located at the patch interfaces. The numerical examples indicated, that the performance of the solver is not affected by such segmentation crimes.

The FETI-DP method is a well established and widely used FE solver, which shows excellent parallel scalability. Therefore, we investigated the parallel performance of the IETI-DP methods. For FETI-DP, a given mesh is distributed to the different processors by a mesh-partitioning software to obtain equally distributed dofs, avoiding load imbalances. However, in IgA, the partition of the domain into patches is already

prescribed by the geometry, which may not fit the number of available processors and the number of dofs associated to a patch may not be equally distributed. To overcome this problem, large patches can be split into smaller ones to obtain a more equally distributed number of dofs. In this work, we increased the multiplicity of the corresponding knots such that the newly introduced patches are again coupled with C^0 continuity. However, this approach leads to a larger number of unknown. If the number of dofs associated to the interface does not dominate the interior dofs, we still obtain a scalable method. In contrast to this, we could stick to the basis that remains smooth at the introduced interfaces, but in turn, we have to deal with so-called fat interfaces, as introduced and analyzed in [19], [22] and [23].

When refining the mesh and increasing the B-Spline degree, the local problem can become quite large. In such cases, sparse direct solvers, which are usually used, cannot efficiently handle these problems, neither in terms of computation time nor memory consumption. We investigated the use of inexact solvers, like MG or FD methods as replacement for the sparse direct solvers. We proposed several variants, each of them having a different combination of direct and inexact solver. We observed, that for small degree p, we almost cannot take any advantage using inexact variants in terms of computation time, but for large degree p especially the FD method provided promising results.

Finally, we investigated solvers for large-scale systems of algebraic equations arising from space-time IgA discretization of parabolic diffusion problems. The goal was to construct robust and parallelizeable smoothers for a time-parallel MG method posed on time-slabs. The construction used a generalized eigendecomposition to decouple the space-time problem into a series of spatial problems, following the ideas of the FD method. Instabilities arising from the ill-conditioned matrix of generalized eigenvectors were overcome by means of a real or complex Schur decomposition instead. The resulting spatial problems corresponding to complex eigenvalues were rewritten as symmetric indefinite problems, for which we constructed optimal and robust block-diagonal preconditioners based on [237]. The issue of additional parallelization in space was handled only by the application of IETI-DP methods for the non-symmetric space-time sub-problems posed on the time-slabs. Moreover, we only used coarsening in time, which does not lead to a scalable method. The use of coarsening in space requires (space-)parallel inter-grid transfers, which are not available in the used library G+Smo. Further investigation and the implementation of the corresponding methods is the topic of current research.

Outlook

The work presented in this thesis can be extend in several directions

- We proved the quasi-optimal condition number bound with respect to the mesh-

size h of the preconditioned cG- and dG-IETI-DP operator only for the two-dimensional case. Moreover, we have not theoretically investigated the dependence on jumps of the diffusion coefficients and on the B-Spline degree p. A rigorous proof for the dependence on α and p, as well as the extension to three-dimensions is still open.

- The number of primal variables used in the dG-IETI-DP method increases compared to the cG case, especially for three-dimensional problems, which has a negative impact on the parallel scalability if the number patch becomes larger. It would be interesting to combine the adaptive selection of primal variables with the dG-IETI-DP to reduce their total number.

- In Chapter 5, we discussed the introduction of patches by a uniform splitting of the patches in the original domain. The goal is to fit the number of available processors and have an almost equal load distribution. For the numerical test, we only considered domains, where the original domain has already an optimal distribution and no such procedure is necessary. Certainly, for realistic applications, an algorithm is required which performs a patch refinement, such that each processor is assign to an almost equal number of dofs and the resulting geometry is still fully matching, i.e., no hanging vertices. Moreover, the assignment of processors to patches should be in such a way, that the required communication between processors is minimal.

- In Chapter 6, we considered inexact solvers for the local problems only for the cG formulation. Both the MG and FD method rely on the tensor product structure of the corresponding parameter domain matrices. However, this structure is lost for the dG version of IETI-DP. Such an extension would be important, but requires new ideas.

- The geometry transformation has quite a significant influence on the performance of the MG and the FD method. Although we incorporated information of the geometrical mapping via a rank-one approximation, we still observed an increased number of iterations for heavily distorted patches. This makes the fine-tuning of parameters, which are used in the inexact versions, very hard. One way to deal with this problem would be to develop a hybrid direct-inexact version, where inexact solvers are used on the very regular patches and direct solvers on the more distorted patches. Nevertheless, one has to take into account the different complexity of sparse direct solvers on the one side and inexact method on the other side. Hence, the patches, where the direct solver is used should be smaller and have less dofs.

- In Chapter 6, we mentioned that solving the problems (6.3) and (6.5) up to a very small relative tolerance is not always possible due to numerical instabilities. However, this is not necessary if the overall approximation error is much larger. Hence, the stopping criteria of the involved iterative methods should be chosen dependent on the (estimated) approximation error. To conclude, a careful and

smart selection of stopping criteria is important for the performance and reliable application of these methods, and has to be further investigated.

- The work presented in Chapter 7 has to been seen as a first step towards the construction of robust and parallel smoothers for the time-parallel MG method. As already discussed in Section 7.2.6, there are various options to use parallel solvers, like the IETI-DP method, to enable additional parallelization in space. Besides IETI-DP, other fast multi-patch solvers can be used, like the multi-patch MG method, as introduced in [204].

Bibliography

[1] R. A. Adams and J. J. F. Fournier. *Sobolev spaces*, volume 140 of *Pure and Applied Mathematics (Amsterdam)*. Elsevier/Academic Press, Amsterdam, second edition, 2003.

[2] R. Andreev. Stability of sparse space-time finite element discretizations of linear parabolic evolution equations. *IMA Journal of Numerical Analysis*, 33(1):242–260, 2013.

[3] R. Andreev, O. Scherzer, and W. Zulehner. Simultaneous optical flow and source estimation: Spacetime discretization and preconditioning. *Applied Numerical Mathematics*, 96:72–81, 2015.

[4] A. Apostolatos, R. Schmidt, R. Wüchner, and K. U. Bletzinger. A Nitsche-type formulation and comparison of the most common domain decomposition methods in isogeometric analysis. *International Journal for Numerical Methods in Engineering*, 97:473–504, 2014.

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

[6] D. N. Arnold, F. Brezzi, B. Cockburn, and L. D. Marini. Unified analysis of discontinuous Galerkin methods for elliptic problems. *SIAM Journal on Numerical Analysis*, 39(5):1749–1779, 2002.

[7] I. Babuška. On the Schwarz algorithm in the theory of differential equations of mathematical physics. *Tchecosl. Math. J.*, 8(83):328–342, 1958. (in Russian).

[8] I. Babuška and T. Janik. The h-p version of the finite element method for parabolic equations. Part I. The p-version in time. *Numerical Methods for Partial Differential Equations*, 5(4):363–399, 1989.

[9] I. Babuška and T. Janik. The h-p version of the finite element method for parabolic equations. II. The h-p version in time. *Numerical Methods for Partial Differential Equations*, 6(4):343–369, 1990.

[10] S. Badia and M. Olm. Space-time balancing domain decomposition. *SIAM Journal on Scientific Computing*, 39(2):C194–C213, 2017.

[11] R. E. Bank and M. S. Metti. An error analysis of some higher order space-time moving finite elements. *Computing and Visualization in Science*, 16(5):219–229, 2013.

[12] R. E. Bank, P. S. Vassilevski, and L. T. Zikatanov. Arbitrary dimension convection-diffusion schemes for space-time discretizations. *Journal of Computational and Applied Mathematics*, 310:19–31, 2017.

[13] R. H. Bartels and G. W. Stewart. Solution of the Matrix Equation AX + XB = C. *Communications of the ACM*, 15(9):820–826, 1972.

[14] Y. Bazilevs, L. Beirão da Veiga, J. A. Cottrell, T. J. R. Hughes, and G. Sangalli. Isogeometric analysis: Approximation, stability and error estimates for h-refined meshes. *Mathematical Models & Methods in Applied Sciences*, 16(7):1031–1090, 2006.

[15] 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.

[16] L. Beirão da Veiga, A. Buffa, G. Sangalli, and R. Vázquez. Mathematical analysis of variational isogeometric methods. *Acta Numerica*, 23:157–287, 2014.

[17] L. Beirão da Veiga, C. Chinosi, C. Lovadina, and L. F. Pavarino. Robust BDDC preconditioners for Reissner-Mindlin plate bending problems and MITC elements. *SIAM Journal on Numerical Analysis*, 47(6):4214–4238, 2010.

[18] L. Beirão da Veiga, D. Cho, L. F. Pavarino, and S. Scacchi. Overlapping Schwarz methods for isogeometric analysis. *SIAM Journal on Numerical Analysis*, 50(3):1394–1416, 2012.

[19] L. Beirão da Veiga, D. Cho, L. F. Pavarino, and S. Scacchi. BDDC preconditioners for isogeometric analysis. *Mathematical Models & Methods in Applied Sciences*, 23(6):1099–1142, 2013.

[20] L. Beirão da Veiga, D. Cho, L. F. Pavarino, and S. Scacchi. Isogeometric Schwarz preconditioners for linear elasticity systems. *Computer Methods in Applied Mechanics and Engineering*, 253:439–454, 2013.

[21] L. Beirão da Veiga, D. Cho, L. F. Pavarino, and S. Scacchi. Overlapping Schwarz preconditioners for isogeometric collocation methods. *Computer Methods in Applied Mechanics and Engineering*, 278:239 – 253, 2014.

[22] L. Beirão da Veiga, L. F. Pavarino, S. Scacchi, O. B. Widlund, and S. Zampini. Isogeometric BDDC preconditioners with deluxe scaling. *SIAM Journal on Scientific Computing*, 36(3):a1118–a1139, 2014.

[23] L. Beirão da Veiga, L. F. Pavarino, S. Scacchi, O. B. Widlund, and S. Zampini. Adaptive selection of primal constraints for isogeometric BDDC deluxe preconditioners. *SIAM Journal on Scientific Computing*, 39(1):A281–A302, 2017.

[24] L. Beirão da Veiga, A. Buffa, J. Rivas, and G. Sangalli. Some estimates for h–p–k-refinement in Isogeometric Analysis. *Numerische Mathematik*, 118(2):271–305, 2011.

[25] L. Beirão da Veiga, A. Buffa, G. Sangalli, and R. Vázquez. Analysis-suitable T-splines of arbitrary degree: definition, linear independence and approximation properties. *Mathematical Models and Methods in Applied Sciences*, 23(11):1979–2003, 2013.

[26] M. Bercovier and I. Soloveichik. Overlapping non Matching Meshes Domain Decomposition Method in Isogeometric Analysis. *ArXiv e-prints: 1502.03756*, 2015.

[27] J. Bergh and J. Löfström. *Interpolation spaces. An introduction.* Springer, Berlin, New York, 1976.

[28] J. H. Bramble and J. E. Pasciak. A preconditioning technique for indefinite systems resulting from mixed approximations of elliptic problems. *Mathematics of Computation*, 50(181):1–17, 1988.

[29] S. C. Brenner and L. R. Scott. *The mathematical theory of finite element methods*, volume 15 of *Texts in Applied Mathematics*. Springer, New York, third edition, 2008.

[30] 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.

[31] A. Bressan and B. Jüttler. A hierarchical construction of LR meshes in 2D. *Computer Aided Geometric Design*, 37:9 – 24, 2015.

[32] E. Brivadis, A. Buffa, B. Wohlmuth, and L. Wunderlich. Isogeometric mortar methods. *Computer Methods in Applied Mechanics and Engineering*, 284(0):292 – 319, 2015.

[33] F. Buchegger, B. Jüttler, and A. Mantzaflaris. Adaptively refined multi-patch B-splines with enhanced smoothness. *Applied Mathematics and Computation*, 272(Part 1):159 – 172, 2016.

[34] A. Buffa, H. Harbrecht, A. Kunoth, and G. Sangalli. BPX-preconditioning for isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 265:63 – 70, 2013.

[35] A. Buffa, R. H. Vázquez, G. Sangalli, and L. B. da Veiga. Approximation estimates for isogeometric spaces in multipatch geometries. *Numerical Methods for Partial Differential Equations*, 31(2):422–438.

[36] F. Calabrò, G. Sangalli, and M. Tani. Fast formation of isogeometric Galerkin matrices by weighted quadrature. *Computer Methods in Applied Mechanics and Engineering*, 316:606–622, 2017.

[37] J. G. Calvo and O. B. Widlund. An adaptive choice of primal constraints for BDDC domain decomposition algorithms. *ETNA. Electronic Transactions on Numerical Analysis*, 45:524–544, 2016.

[38] C. Canuto, L. F. Pavarino, and A. B. Pieri. BDDC preconditioners for continuous and discontinuous Galerkin methods using spectral/hp elements with variable local polynomial degree. *IMA Journal of Numerical Analysis*, 34(3):879–903, 2014.

[39] L. A. Charawi. Isogeometric overlapping additive Schwarz solvers for the bidomain system. In T. Dickopf, M. J. Gander, L. Halpern, R. Krause, and L. F. Pavarino, editors, *Domain decomposition methods in science and engineering XXII*, volume 104 of *Lecture Notes in Computational Science and Engineering*, pages 127–135. Springer, Cham, 2016.

[40] J. A. Cottrell, T. J. R. Hughes, and Y. Bazilevs. *Isogeometric analysis. Toward integration of CAD and FEA.* John Wiley & Sons, Chichester, 2009.

[41] D. A. Di Pietro and A. Ern. *Mathematical aspects of discontinuous Galerkin methods*, volume 69 of *Mathematics & Applications*. Springer, Berlin, Heidelberg, 2012.

[42] C. R. Dohrmann. A preconditioner for substructuring based on constrained energy minimization. *SIAM Journal on Scientific Computing*, 25(1):246–258, 2003.

[43] C. R. Dohrmann. An approximate BDDC preconditioner. *Numerical Linear Algebra with Applications*, 14(2):149–168, 2007.

[44] C. R. Dohrmann and O. B. Widlund. Some recent tools and a BDDC algorithm for 3D problems in $H(\mathrm{curl})$. In R. Bank, M. Holst, O. Widlund, and J. Xu, editors, *Domain decomposition methods in science and engineering XX*, volume 91 of *Lecture Notes in Computational Science and Engineering*, pages 15–25. Springer, Heidelberg, 2013.

[45] C. R. Dohrmann and O. B. Widlund. A BDDC algorithm with deluxe scaling for three-dimensional $H(\mathbf{curl})$ problems. *Communications on Pure and Applied Mathematics*, 69(4):745–770, 2016.

[46] T. Dokken, T. Lyche, and K. F. Pettersen. Polynomial splines over locally refined box-partitions. *Computer Aided Geometric Design*, 30(3):331 – 356, 2013.

[47] M. Donatelli, C. Garoni, C. Manni, S. Serra-Capizzano, and H. Speleers. Symbol-based multigrid methods for galerkin b-spline isogeometric analysis. *SIAM Journal on Numerical Analysis*, 55(1):31–62, 2017.

[48] 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*, 22(12):1155–1162, 2006.

[49] C. C. Douglas, G. Haase, and U. Langer. *A tutorial on elliptic PDE solvers and their parallelization*, volume 16 of *Software, Environments, and Tools*. Society for Industrial and Applied Mathematics, Philadelphia, 2003.

[50] M. Dryja. On discontinuous Galerkin methods for elliptic problems with discontinuous coefficients. *Computational Methods in Applied Mathematics*, 3(1):76–85, 2003.

[51] M. Dryja, J. Galvis, and M. Sarkis. BDDC methods for discontinuous Galerkin discretization of elliptic problems. *Journal of Complexity*, 23(4-6):715–739, 2007.

[52] M. Dryja, J. Galvis, and M. Sarkis. A FETI-DP preconditioner for a composite finite element and discontinuous Galerkin method. *SIAM Journal on Numerical Analysis*, 51(1):400–422, 2013.

[53] M. Dryja, J. Galvis, and M. Sarkis. A deluxe FETI-DP preconditioner for a composite finite element and DG method. *Computational Methods in Applied Mathematics*, 15(4):465–482, 2015.

[54] M. Dryja, J. Galvis, and M. Sarkis. The analysis of a FETI-DP preconditioner for a full DG discretization of elliptic problems in two dimensions. *Numerische Mathematik*, 131(4):737–770, 2015.

[55] M. Dryja, J. Galvis, and M. Sarkis. A deluxe FETI-DP method for full DG discretization of elliptic problems. In T. Dickopf, M. J. Gander, L. Halpern, R. Krause, and L. F. Pavarino, editors, *Domain decomposition methods in science and engineering XXII*, volume 104 of *Lecture Notes in Computational Science and Engineering*, pages 157–165. Springer, Cham, 2016.

[56] M. Dryja and W. Proskurowski. A FETI-DP method for the mortar discretization of elliptic problems with discontinuous coefficients. In T. J. Barth, M. Griebel, D. E. Keyes, R. M. Nieminen, D. Roose, T. Schlick, R. Kornhuber, R. Hoppe, J. Périaux, O. Pironneau, O. Widlund, and J. Xu, editors, *Domain decomposition methods in science and engineering*, volume 40 of *Lecture Notes in Computational Science and Engineering*, pages 345–352. Springer, Berlin, 2005.

[57] M. Dryja and M. Sarkis. 3-D FETI-DP preconditioners for composite finite element-discontinuous Galerkin methods. In J. Erhel, M. J. Gander, L. Halpern, G. Pichot, T. Sassi, and O. Widlund, editors, *Domain decomposition methods*

in science and engineering XXI, volume 98 of *Lecture Notes in Computational Science and Engineering*, pages 127–140. Springer, Cham, 2014.

[58] A. Ern, I. Smears, and M. Vohralik. Equilibrated flux a posteriori error estimates in $L^2(H^1)$-norms for high-order discretizations of parabolic problems. *ArXiv e-prints: 1703.04987*, 2017.

[59] J. A. Evans and T. J. R. Hughes. Explicit trace inequalities for isogeometric analysis and parametric hexahedral finite elements. *Numerische Mathematik*, 123(2):259–290, 2013.

[60] L. C. Evans. *Partial differential equations*, volume 19 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, second edition, 2010.

[61] A. Falini, J. Špeh, and B. Jüttler. Planar domain parameterization with THB-splines. *Computer Aided Geometric Design*, 35/36:95–108, 2015.

[62] 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(7):1523–1544, 2001.

[63] C. Farhat, J. Mandel, and F. X. Roux. Optimal convergence properties of the FETI domain decomposition method. *Computer Methods in Applied Mechanics and Engineering*, 115(3):365 – 385, 1994.

[64] C. Farhat and F.-X. Roux. A method of finite element tearing and interconnecting and its parallel solution algorithm. *International Journal for Numerical Methods in Engineering*, 32(6):1205–1227, 1991.

[65] K. Gahalaut, J. Kraus, and S. Tomar. Multigrid methods for isogeometric discretization. *Computer Methods in Applied Mechanics and Engineering*, 253:413 – 425, 2013.

[66] M. Gahalaut. *Isogeometric Analysis: Condition NumberEstimates and Fast Solvers*. PhD thesis, Johannes Kepler University, May 2013.

[67] M. Gander. 50 Years of Time Parallel Time Integration. In T. Carraro, M. Geiger, S. Körkel, and R. Rannacher, editors, *Multiple Shooting and Time Domain Decomposition*, pages 69–114. Springer, 2015.

[68] M. Gander and M. Neumüller. Analysis of a new space-time parallel multigrid algorithm for parabolic problems. *SIAM Journal on Scientific Computing*, 38(4):A2173–A2208, 2016.

[69] M. J. Gander. Schwarz methods over the course of time. *ETNA. Electronic Transactions on Numerical Analysis*, 31:228–255, 2008.

[70] M. J. Gander and S. Vandewalle. Analysis of the parareal time-parallel time-integration method. *SIAM Journal on Scientific Computing*, 29(2):556–578, 2007.

[71] D. Garcia, D. Pardo, L. Dalcin, M. Paszyński, N. Collier, and V. M. Calo. The value of continuity: Refined isogeometric analysis and fast direct solvers. *Computer Methods in Applied Mechanics and Engineering*, 316:586 – 605, 2017.

[72] C. Giannelli, B. Jüttler, S. K. Kleiss, A. Mantzaflaris, B. Simeon, and J. Speh. THB-splines: An effective mathematical technology for adaptive refinement in geometric design and isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 299:337 – 365, 2016.

[73] C. Giannelli, B. Jüttler, and H. Speleers. THB-splines: the truncated basis for hierarchical splines. *Computer Aided Geometric Design*, 29, 2012.

[74] C. Giannelli, B. Jüttler, and H. Speleers. Strongly stable bases for adaptively refined multilevel spline spaces. *Advances in Computational Mathematics*, 40:459–490, 2014.

[75] S. Gippert, A. Klawonn, and O. Rheinbach. Analysis of FETI-DP and BDDC for linear elasticity in 3D with almost incompressible components and varying coefficients inside subdomains. *SIAM Journal on Numerical Analysis*, 50(5):2208–2236, 2012.

[76] G. Guennebaud, B. Jacob, et al. Eigen v3. http://eigen.tuxfamily.org, 2010.

[77] G. Haase. Hierarchical extension operators plus smoothing in domain decomposition preconditioners. *Applied Numerical Mathematics*, 23(3):327 – 346, 1997.

[78] G. Haase and U. Langer. The non-overlapping domain decomposition multiplicative schwarz method. *International Journal of Computer Mathematics*, 44(1-4):223–242, 1992.

[79] G. Haase, U. Langer, and A. Meyer. The approximate Dirichlet Domain Decomposition method. Part I: An algebraic approach. *Computing*, 47(2):137–151, 1991.

[80] G. Haase, U. Langer, and A. Meyer. The approximate Dirichlet Domain Decomposition method. Part II: Applications to 2nd-order Elliptic B.V.P.s. *Computing*, 47(2):153–167, 1991.

[81] G. Haase, U. Langer, A. Meyer, and S. Nepomnyaschikh. Hierarchical extension operators and local multigrid methods in domain decomposition preconditioners. *East-West Journal of Numerical Mathematics*, 2(3):173–193, 1994.

[82] G. Haase and S. Nepomnyaschikh. Explicit extension operators on hierarchical grids. *East-West Journal of Numerical Mathematics*, 5(4):231–248, 1998.

[83] W. Hackbusch. Parabolic multigrid methods. In *Computing methods in applied sciences and engineering, VI (Versailles, 1983)*, pages 189–197. North-Holland, Amsterdam, 1984.

[84] W. Hackbusch. *Multi-Grid Methods and Applications*, volume 4. Springer, Berlin, Heidelberg, 1985.

[85] P. Hansbo. Space-time oriented streamline diffusion methods for nonlinear conservation laws in one dimension. *Communications in Numerical Methods in Engineering*, 10(3):203–215, 1994.

[86] A. Heinlein, U. Hetmaniuk, A. Klawonn, and O. Rheinbach. The approximate component mode synthesis special finite element method in two dimensions: parallel implementation and numerical results. *Journal of Computational and Applied Mathematics*, 289:116–133, 2015.

[87] C. Hesch and P. Betsch. Isogeometric analysis and domain decomposition methods. *Computer Methods in Applied Mechanics and Engineering*, 213-216:104–112, 2012.

[88] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49:409–436, 1952.

[89] C. Hofer. Parallelization of continuous and discontinuous Galerkin dual-primal Isogeometric tearing and interconnecting methods. *ArXiv e-prints: 1611.08122*, 2016.

[90] C. Hofer. Parallelization of continuous and discontinuous Galerkin dual-primal isogeometric tearing and interconnecting methods. *Computers & Mathematics with Applications*, 74(7):1607 – 1625, 2017.

[91] C. Hofer. Analysis of discontinuous Galerkin dual-primal isogeometric tearing and interconnecting methods. *Mathematical Models & Methods in Applied Sciences*, 28(1):131–158, 2018.

[92] C. Hofer and U. Langer. Dual-primal isogeometric tearing and interconnecting solvers for multipatch continuous and discontinuous Galerkin IgA equations. *PAMM*, 16(1):747–748, 2016.

[93] C. Hofer and U. Langer. Dual-primal isogeometric tearing and interconnecting solvers for multipatch dG-IgA equations. *Computer Methods in Applied Mechanics and Engineering*, 316:2 – 21, 2017.

[94] C. Hofer and U. Langer. Dual-primal isogeometric tearing and interconnecting methods. In B. Chetverushkin, W. Fitzgibbon, Y. Kuznetsov, P. Neittaanmäki, O. Pironneau, and J. Periaux, editors, *Contributions to Partial Differential Equations and Applications*, volume 47 of *Computational Methods in Applied Sciences*. Springer International Publishing, Berlin, Heidelberg, New York, 2019.

[95] C. Hofer, U. Langer, M. Neumüller, and I. Toulopoulos. Time-Multipatch Discontinuous Galerkin Space-Time Isogeometric Analysis of Parabolic Evolution Problems. Technical Report No. 2017-05, DK Computational Mathematics Linz Report Series, 2017. available at https://www.dk-compmath.jku.at/publications/dk-reports/2017-08-01.

[96] C. Hofer, U. Langer, and S. Takacs. Inexact Dual-Primal Isogeometric Tearing and Interconnecting Methods. In *Domain decomposition methods in science and engineering XXIV*. Springer, Berlin, Heidelberg, 2018. accepted.

[97] C. Hofer, U. Langer, and I. Toulopoulos. Discontinuous Galerkin Isogeometric Analysis of Elliptic Diffusion Problems on Segmentations with Gaps. *SIAM Journal on Scientific Computing*, 38:A3430 – A3460, 2016.

[98] C. Hofer, U. Langer, and I. Toulopoulos. Discontinuous Galerkin Isogeometric Analysis on non-matching segmentation: error estimates and efficient solvers. RICAM-Report 23, Johann Radon Institute for Computational and Applied Mathematics, Austrian Academy of Sciences, 2016. available at https://www.ricam.oeaw.ac.at/publications/ricam-reports/Report No. 2016-23.

[99] C. Hofer and I. Toulopoulos. Discontinuous Galerkin Isogeometric Analysis of elliptic problems on segmentations with non-matching interfaces. *Computers & Mathematics with Applications*, 72(7):1811 – 1827, 2016.

[100] C. Hofer and I. Toulopoulos. Discontinuous Galerkin Isogeometric Analysis for segmentations generating overlapping regions. *ArXiv e-prints: 1803.09616*, 2018.

[101] C. Hofreither. A black-box low-rank approximation algorithm for fast matrix assembly in Isogeometric Analysis. *Computer Methods in Applied Mechanics and Engineering*, 333:311–330, 2018.

[102] C. Hofreither and S. Takacs. Robust multigrid for isogeometric analysis based on stable splittings of spline spaces. *SIAM Journal on Numerical Analysis*, 55(4):2004–2024, 2017.

[103] C. Hofreither, S. Takacs, and W. Zulehner. A robust multigrid method for Isogeometric Analysis in two dimensions using boundary correction. *Computer Methods in Applied Mechanics and Engineering*, 316:22–42, 2017.

[104] G. Horton. The time-parallel multigrid method. *Communications in Applied Numerical Methods*, 8(9):585–595, 1992.

[105] G. Horton and S. Vandewalle. A space-time multigrid method for parabolic partial differential equations. *SIAM Journal on Scientific Computing*, 16(4):848–864, 1995.

[106] J. Hoschek and D. Lasser. *Fundamentals of computer aided geometric design*. A K Peters, Wellesley, 1993. Translated from the 1992 German edition by Larry L. Schumaker.

[107] T. J. R. Hughes, J. A. Cottrell, and Y. Bazilevs. Isogeometric Analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Computer Methods in Applied Mechanics and Engineering*, 194:4135–4195, 2005.

[108] C. Johnson, U. Nävert, and J. Pitkäranta. Finite element methods for linear hyperbolic problems. *Computer Methods in Applied Mechanics and Engineering*, 45(1-3):285–312, 1984.

[109] C. Johnson and J. Saranen. Streamline diffusion methods for the incompressible Euler and Navier-Stokes equations. *Mathematics of Computation*, 47(175):1–18, 1986.

[110] B. Jüttler, M. Kapl, D.-M. Nguyen, Q. Pan, and M. Pauley. Isogeometric segmentation: The case of contractible solids without non-convex edges. *Computer-Aided Design*, 57:74–90, 2014.

[111] B. Jüttler, U. Langer, A. Mantzaflaris, S. E. Moore, and W. Zulehner. Geometry + Simulation Modules: Implementing Isogeometric Analysis. *PAMM*, 14(1):961–962, 2014.

[112] M. Kapl, F. Buchegger, M. Bercovier, and B. Jüttler. Isogeometric analysis with geometrically continuous functions on planar multi-patch geometries. *Computer Methods in Applied Mechanics and Engineering*, 316:209–234, 2017.

[113] M. Kapl, G. Sangalli, and T. Takacs. Construction of analysis-suitable G1 planar multi-patch parameterizations. *Computer-Aided Design*, 97:41 – 55, 2018.

[114] M. Kapl, V. Vitrih, B. Jüttler, and K. Birner. Isogeometric analysis with geometrically continuous functions on two-patch geometries. *Computers & Mathematics with Applications*, 70(7):1518 – 1538, 2015.

[115] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20(1):359–392, 1998.

[116] H. Keller. *Numerical Solution of Two Point Boundary Value Problems*. Society for Industrial and Applied Mathematics, Philadelphia, 1976.

[117] A. Klawonn, M. Kühn, and O. Rheinbach. Adaptive FETI-DP and BDDC methods with a generalized transformation of basis for heterogeneous problems. *ETNA. Electronic Transactions on Numerical Analysis*, 49:1–27, 2018.

[118] A. Klawonn, M. Lanser, P. Radtke, and O. Rheinbach. On an adaptive coarse space and on nonlinear domain decomposition. In J. Erhel, M. J. Gander, L. Halpern, G. Pichot, T. Sassi, and O. Widlund, editors, *Domain decomposition methods in science and engineering XXI*, volume 98 of *Lecture Notes in Computational Science and Engineering*, pages 71–83. Springer, Cham, 2014.

[119] A. Klawonn, M. Lanser, and O. Rheinbach. Nonlinear FETI-DP and BDDC methods. *SIAM Journal on Scientific Computing*, 36(2):737–765, 2014.

[120] A. Klawonn, M. Lanser, and O. Rheinbach. Toward extremely scalable nonlinear domain decomposition methods for elliptic partial differential equations. *SIAM Journal on Scientific Computing*, 37(6):c667–c696, 2015.

[121] A. Klawonn, M. Lanser, and O. Rheinbach. A highly scalable implementation of inexact nonlinear FETI-DP without sparse direct solvers. In B. Karasözen, M. Manguoğlu, M. Tezer-Sezgin, S. Göktepe, and Ö. Uğur, editors, *Numerical mathematics and advanced applications—ENUMATH 2015*, volume 112 of *Lecture Notes in Computational Science and Engineering*, pages 255–264. Springer, Cham, 2016.

[122] A. Klawonn, M. Lanser, and O. Rheinbach. A nonlinear FETI-DP method with an inexact coarse problem. In T. Dickopf, M. J. Gander, L. Halpern, R. Krause, and L. F. Pavarino, editors, *Domain decomposition methods in science and engineering XXII*, volume 104 of *Lecture Notes in Computational Science and Engineering*, pages 41–52. Springer, Cham, 2016.

[123] A. Klawonn, M. Lanser, O. Rheinbach, H. Stengel, and G. Wellein. *Hybrid MPI/OpenMP Parallelization in FETI-DP Methods*, pages 67–84. Springer International Publishing, Cham, 2015.

[124] A. Klawonn, M. Lanser, O. Rheinbach, and M. Uran. Nonlinear FETI-DP and BDDC methods: a unified framework and parallel results. *SIAM Journal on Scientific Computing*, 39(6):c417–c451, 2017.

[125] A. Klawonn, L. F. Pavarino, and O. Rheinbach. Spectral element FETI-DP and BDDC preconditioners with multi-element subdomains. *Computer Methods in Applied Mechanics and Engineering*, 198(3–4):511 – 523, 2008.

[126] A. Klawonn, P. Radtke, and O. Rheinbach. FETI-DP with different scalings for adaptive coarse spaces. *PAMM - Proceedings in Applied Mathematics and Mechanics*, 14(1):835–836, 2014.

[127] A. Klawonn, P. Radtke, and O. Rheinbach. FETI-DP methods with an adaptive coarse space. *SIAM Journal on Numerical Analysis*, 53(1):297–320, 2015.

[128] A. Klawonn, P. Radtke, and O. Rheinbach. A comparison of adaptive coarse spaces for iterative substructuring in two dimensions. *ETNA. Electronic Transactions on Numerical Analysis*, 45:75–106, 2016.

[129] A. Klawonn and O. Rheinbach. A parallel implementation of Dual-Primal FETI methods for three-dimensional linear elasticity using a transformation of basis. *SIAM Journal on Scientific Computing*, 28(5):1886–1906, 2006.

[130] A. Klawonn and O. Rheinbach. Inexact FETI-DP methods. *International journal for numerical methods in engineering*, 69(2):284–307, 2007.

[131] A. Klawonn and O. Rheinbach. Highly scalable parallel domain decomposition methods with an application to biomechanics. *ZAMM. Zeitschrift für Angewandte Mathematik und Mechanik. Journal of Applied Mathematics and Mechanics*, 90(1):5–32, 2010.

[132] A. Klawonn, O. Rheinbach, and L. F. Pavarino. Exact and inexact FETI-DP methods for spectral elements in two dimensions. In U. Langer, M. Discacciati, D. E. Keyes, O. B. Widlund, and W. Zulehner, editors, *Domain decomposition methods in science and engineering XVII*, volume 60 of *Lecture Notes in Computational Science and Engineering*, pages 279–286. Springer, Berlin, 2008.

[133] A. Klawonn and O. B. Widlund. A domain decomposition method with Lagrange multipliers and inexact solvers for linear elasticity. *SIAM Journal on Scientific Computing*, 22(4):1199–1219, 2000.

[134] A. Klawonn and O. B. Widlund. FETI and Neumann-Neumann iterative substructuring methods: Connections and new results. *Communications on Pure and Applied Mathematics*, 54(1):57–90, 2001.

[135] A. Klawonn and O. B. Widlund. Dual-primal FETI methods for linear elasticity. *Communications on Pure and Applied Mathematics*, 59(11):1523–1572, 2006.

[136] A. Klawonn, O. B. Widlund, and M. Dryja. Dual-primal FETI methods for three-dimensional elliptic problems with heterogeneous coefficients. *SIAM Journal on Numerical Analysis*, 40(1):159–179, 2002.

[137] S. Kleiss, C. Pechstein, B. Jüttler, and S. Tomar. IETI–isogeometric tearing and interconnecting. *Computer Methods in Applied Mechanics and Engineering*, 247:201–215, 2012.

[138] V. G. Korneev and U. Langer. *Dirichlet-Dirichlet domain decomposition methods for elliptic problems: h and hp finite element discretizations.* World Scientific Publishing, Hackensack, 2015.

[139] C. Koutschan, M. Neumüller, and C.-S. Radu. Inverse inequality estimates with symbolic computation. *Advances in Applied Mathematics*, 80:1 – 23, 2016.

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

[141] O. A. Ladyzhenskaya. *The Boundary Value Problems of Mathematical Physics.* Nauka, Moscow, 1973. In Russian. Translated in *Applied Mathematical Sciences* 49, Springer, 1985.

[142] O. A. Ladyzhenskaya, V. A. Solonnikov, and N. N. Uraltseva. *Linear and Quasilinear Equations of Parabolic Type*. Nauka, Moscow, 1967. In Russian. Translated in *Applied Mathematical Sciences*, Providence, 1968.

[143] J. Lang. *Adaptive Multilevel Solution of Nonlinear Parabolic PDE Systems. Theory, Algorithm, and Applications*, volume 16 of *Lecture Notes in Computational Sciences and Engineering*. Springer, Berlin, Heidelberg, 2000.

[144] U. Langer, A. Mantzaflaris, S. E. Moore, and I. Toulopoulos. Multipatch discontinuous Galerkin Isogeometric Analysis. In B. Jüttler and B. Simeon, editors, *Isogeometric Analysis and Applications IGAA 2014*, volume 107 of *Lecture Notes in Computer Science*, pages 1–32, Heidelberg, 2015. Springer.

[145] U. Langer, S. Matculevich, and S. Repin. Functional type error control for stabilised space-time IgA approximations to parabolic problems. In I. Lirkov and S. Margenov, editors, *Large-Scale Scientific Computing (LSSC 2017)*, volume 10665 of *Lecture Notes in Computer Science*, pages 55–65, Cham, 2017. Springer.

[146] U. Langer, S. Moore, and M. Neumüller. Space-time isogeometric analysis of parabolic evolution equations. *Computer methods in applied mechanics and engineering*, 306:342–363, 2016.

[147] U. Langer and S. E. Moore. Discontinuous Galerkin Isogeometric Analysis of elliptic PDEs on surfaces. In T. Dickopf, M. J. Gander, L. Halpern, R. Krause, and L. F. Pavarino, editors, *Domain decomposition methods in science and engineering XXII*, volume 104 of *Lecture Notes in Computational Science and Engineering*, pages 319–326. Springer, Cham, 2016.

[148] U. Langer and I. Toulopoulos. Analysis of multipatch discontinuous Galerkin IgA approximations to elliptic boundary value problems. *Computing and Visualization in Science*, 17(5):217–233, 2015.

[149] S. Larsson and M. Molten. Numerical solution of parabolic problems based on a weak space-time formulation. *Computational Methods in Applied Mathematics*, 17(1):65–84, 2017.

[150] B. Q. Li. *Discontinuous finite elements in fluid dynamics and heat transfer*. Computational Fluid and Solid Mechanics. Springer, London, 2006.

[151] J. Li. *Dual-primal FETI methods for stationary Stokes and Navier-Stokes equations*. PhD thesis, Department of Mathematics, Courant Institute, New York University, June 2002.

[152] J. Li and O. B. Widlund. On the use of inexact subdomain solvers for BDDC algorithms. *Computer Methods in Applied Mechanics and Engineering*, 196(8):1415–1428, 2007.

[153] J.-L. Lions, Y. Maday, and G. Turinici. A "parareal" in time discretization of PDE's. *Comptes Rendus de l'Académie des Sciences. Série I. Mathématique*, 332:661–668, 2001.

[154] P.-L. Lions. On the Schwarz alternating method. I. In R. Glowinski, G. H. Golub, G. A. Meurant, and J. Périaux, editors, *First international symposium on domain decomposition methods for partial differential equations*, pages 1–42. Paris, France, 1988.

[155] P.-L. Lions. On the Schwarz alternating method. II. Stochastic interpretation and order properties. In T. Chan, R. Glowinski, J. Périaux, and O. Widlund, editors, *Domain decomposition methods (Los Angeles, CA, 1988)*, pages 47–70, Philadelphia, 1989. Society for Industrial and Applied Mathematics.

[156] P.-L. Lions. On the Schwarz alternating method. III: a variant for nonoverlapping subdomains. In T. Chan, R. Glowinski, J. Périaux, and O. Widlund, editors, *Third international symposium on domain decomposition methods for partial differential equations*, volume 6, pages 202–223, Philadelphia, 1990. Society for Industrial and Applied Mathematics.

[157] R. E. Lynch, J. R. Rice, and D. H. Thomas. Direct solution of partial difference equations by tensor product methods. *Numerische Mathematik*, 6(1):185–199, 1964.

[158] J. Mandel and C. R. Dohrmann. Convergence of a balancing domain decomposition by constraints and energy minimization. *Numerical Linear Algebra with Applications*, 10(7):639–659, 2003.

[159] 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.

[160] J. Mandel and R. Tezaur. Convergence of a substructuring method with Lagrange multipliers. *Numerische Mathematik*, 73(4):473–487, 1996.

[161] J. Mandel and R. Tezaur. On the convergence of a dual-primal substructuring method. *Numerische Mathematik*, 88(3):543–558, 2001.

[162] A. Mantzaflaris, C. Hofer, et al. G+Smo (Geometry plus Simulation modules) v0.8.1. http://gs.jku.at/gismo, 2015.

[163] A. Mantzaflaris and B. Jüttler. Integration by interpolation and look-up for Galerkin-based isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 284:373–400, 2015.

[164] A. Mantzaflaris, B. Jüttler, B. Khoromskij, and U. Langer. Low rank tensor methods in Galerkin-based Isogeometric Analysis. *Computer Methods in Applied Mechanics and Engineering*, 316:1062–1085, 2017.

[165] T. P. A. Mathew. *Domain decomposition methods for the numerical solution of partial differential equations*, volume 61 of *Lecture Notes in Computational Science and Engineering*. Springer, Berlin, 2008.

[166] S. G. Mikhlin. On the Schwarz algorithm. *Dokl.Akad. Nauk S.S.S.R.*, 77(4):569–571, 1951. (in Russian).

[167] C. Mollet. Stability of Petrov-Galerkin discretizations: Application to the space-time weak formulation for parabolic evolution problems. *Computational Methods in Applied Mathematics*, 14(2):231–255, 2014.

[168] M. Montardini, G. Sangalli, and M. Tani. Robust isogeometric preconditioners for the Stokes system based on the Fast Diagonalization method. *ArXiv e-prints: 1712.00403*, 2017.

[169] S. Moore. *Nonstandard Discretization Strategies In Isogeometric Analysis for Partial Differential Equations*. PhD thesis, Johannes Kepler University Linz, Linz, 2017.

[170] M. Neumüller. *Space-Time Methods: Fast Solvers and Applications*, volume 20 of *Monographic Series TU Graz: Computation in Engineering and Science*. TU Graz, 2013.

[171] M. Neumüller and I. Smears. Time-parallel iterative solvers for parabolic evolution equations. *ArXiv e-prints: 1802.08126*, 2018.

[172] M. Neumüller and O. Steinbach. Refinement of flexible space-time finite element meshes and discontinuous Galerkin methods. *Computing and Visualization in Science*, 14(5):189–205, 2011.

[173] D.-M. Nguyen, M. Pauley, and B. Jüttler. Isogeometric Segmentation. Part II: On the segmentability of contractible solids with non-convex edges. *Graphical Models*, 76:426–439, 2014.

[174] D.-M. Nguyen, M. Pauley, and B. Jüttler. Isogeometric segmentation: Construction of auxiliary curves. *Computer-Aided Design*, 70:89–99, 2016.

[175] V. P. Nguyen, P. Kerfriden, M. Brino, S. P. A. Bordas, and E. Bonisoli. Nitsche's method for two and three dimensional NURBS patch coupling. *Computational Mechanics*, 53(6):1163–1182, 2014.

[176] J. Nievergelt. Parallel methods for integrating ordinary differential equations. *Communications of the ACM*, 7:731–733, 1964.

[177] J. Nitsche. Über ein Variationsprinzip zur Lösung von Dirichlet-Problemen bei Verwendung von Teilräumen, die keinen Randbedingungen unterworfen sind. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 36:9–15, 1971. (in German).

[178] C. C. Paige and M. A. Saunders. Solution of Sparse Indefinite Systems of Linear Equations. *SIAM Journal on Numerical Analysis*, 12(4):617–629, 1975.

[179] M. Pauley, D.-M. Nguyen, D. Mayer, J. Speh, O. Weeger, and B. Jüttler. The isogeometric segmentation pipeline. In B. Jüttler and B. Simeon, editors, *Isogeometric Analysis and Applications IGAA 2014*, Lecture Notes in Computer Science, Heidelberg, 2015. Springer.

[180] L. F. Pavarino. Neumann-Neumann algorithms for spectral elements in three dimensions. *ESAIM: Mathematical Modelling and Numerical Analysis*, 31(4):471–493, 1997.

[181] L. F. Pavarino. BDDC and FETI-DP preconditioners for spectral element discretizations. *Computer Methods in Applied Mechanics and Engineering*, 196(8):1380 – 1388, 2007.

[182] L. F. Pavarino, O. B. Widlund, and S. Zampini. BDDC preconditioners for spectral element discretizations of almost incompressible elasticity in three dimensions. *SIAM Journal on Scientific Computing*, 32(6):3604–3626, 2010.

[183] C. Pechstein. *Finite and boundary element tearing and interconnecting solvers for multiscale problems.*, volume 90 of *Lecture Notes in Computational Science and Engineering*. Springer, Berlin, Heidelberg, 2013.

[184] C. Pechstein and C. R. Dohrmann. A unified framework for adaptive BDDC. *ETNA. Electronic Transactions on Numerical Analysis*, 46:273–336, 2017.

[185] L. Piegl and W. Tiller. *The NURBS Book (2Nd Ed.)*. Springer, New York, 1997.

[186] A. Quarteroni and A. Valli. *Domain decomposition methods for partial differential equations*. Numerical Mathematics and Scientific Computation. The Clarendon Press, Oxford University Press, New York, 1999.

[187] O. Rheinbach. Parallel iterative substructuring in structural mechanics. *Archives of Computational Methods in Engineering*, 16(4):425–463, 2009.

[188] B. Rivière. *Discontinuous Galerkin methods for solving elliptic and parabolic equations*, volume 35 of *Frontiers in Applied Mathematics*. Society for Industrial and Applied Mathematics, Philadelphia, 2008. Theory and implementation.

[189] M. Ruess, D. Schillinger, A. I. Özcan, and E. Rank. Weak coupling for isogeometric analysis of non-matching and trimmed multi-patch geometries. *Computer Methods in Applied Mechanics and Engineering*, 269(0):46 – 71, 2014.

[190] Y. Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, Philadelphia, second edition, 2003.

[191] Y. Saad and M. H. Schultz. GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems. *SIAM Journal on Scientific and Statistical Computing*, 7(3):856–869, 1986.

[192] G. Sangalli and M. Tani. Isogeometric preconditioners based on fast solvers for the sylvester equation. *SIAM Journal on Scientific Computing*, 38(6):A3644–A3671, 2016.

[193] G. Sangalli and M. Tani. Matrix-free isogeometric analysis: the computationally efficient k-method. *ArXiv e-prints: 1712.08565*, 2017.

[194] K. Scherer and A. Y. Shadrin. New upper bound for the B-spline basis condition number. II. A proof of de Boor's 2^k-conjecture. *Journal of Approximation Theory*, 99(2):217–229, 1999.

[195] J. Schöberl and W. Zulehner. Symmetric indefinite preconditioners for saddle point problems with applications to PDE-constrained optimization problems. *SIAM Journal on Matrix Analysis and Applications*, 29(3):752–773, 2007.

[196] L. L. Schumaker. *Spline functions: basic theory*. Cambridge Mathematical Library. Cambridge University Press, Cambridge, third edition, 2007.

[197] C. Schwab and R. Stevenson. Space-time adaptive wavelet methods for parabolic evolution problems. *Mathematics of Computation*, 78(267):1293–1318, 2009.

[198] H. A. Schwarz. II. Über einen Grenzübergang durch alternirendes Verfahren. Wolf J. XV. 272-286. 1870.

[199] T. W. Sederberg, J. Zheng, A. Bakenov, and A. Nasri. T-splines and T-NURCCs. *ACM Transactions on Graphics*, 22(3):477–484, 2003.

[200] I. Smears. Robust and efficient preconditioners for the discontinuous galerkin time-stepping method. *IMA Journal of Numerical Analysis*, 37(4):1961–1985, 2017.

[201] B. F. Smith, P. E. Bjørstad, and W. Gropp. *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*. Cambridge University Press, Cambridge, 1996.

[202] S. L. Sobolev. L'Algorithme de Schwarz dans la Théorie de l'Elasticité. *Comptes Rendus (Doklady) de l'Académie des Sciences de l'URSS*, IV((XIII) 6):243–246, 1936.

[203] J. Sogn and S. Takacs. Robust multigrid solvers for the biharmonic problem in isogeometric analysis. *ArXiv e-prints: 1802.00220*, 2018.

[204] S.Takacs. Robust approximation error estimates and multigrid solvers for isogeometric multi-patch discretizations. *ArXiv e-prints: 1709.05375*, 2017.

[205] O. Steinbach. *Stability estimates for hybrid coupled domain decomposition methods*, volume 1809 of *Lecture Notes in Mathematics*. Springer, Heidelberg, 2003.

[206] O. Steinbach. Space-time finite element methods for parabolic problems. *Computational Methods in Applied Mathematics*, 15(4):551–566, 2015.

[207] M. Stoll and A. Wathen. Combination preconditioning and the Bramble-Pasciak $^+$ preconditioner. *SIAM Journal on Matrix Analysis and Applications*, 30(2):582–608, 2008.

[208] A. Tagliabue, L. Dedé, and A. Quarteroni. Isogeometric Analysis and error estimates for high order partial differential equations in fluid dynamics. *Computers & Fluids*, 102:277–303, 2014.

[209] S. Takacs. Robust multigrid methods for isogeometric discretizations of the Stokes equations. *ArXiv e-prints: 1705.04481*, 2017.

[210] S. Takacs and T. Takacs. Approximation error estimates and inverse inequalities for B-splines of maximum smoothness. *Mathematical Models and Methods in Applied Sciences*, pages 1–35, 2016.

[211] M. Tani. A preconditioning strategy for linear systems arising from nonsymmetric schemes in isogeometric analysis. *Computers & Mathematics with Applications*, 74(7):1690 – 1702, 2017.

[212] T. E. Tezduyar. Interface-tracking and interface-capturing techniques for finite element computation of moving boundaries and interfaces. *Computer Methods in Applied Mechanics and Engineering*, 195(23-24):2983–3000, 2006.

[213] V. Thomée. *Galerkin finite element methods for parabolic problems*, volume 25 of *Springer Series in Computational Mathematics*. Springer, Berlin, second edition, 2006.

[214] A. Toselli. FETI domain decomposition methods for scalar advection-diffusion problems. *Computer Methods in Applied Mechanics and Engineering*, 190(43):5759 – 5776, 2001.

[215] A. Toselli and O. B. Widlund. *Domain decomposition methods – algorithms and theory.*, volume 34 of *Springer Series in Computational Mathematics*. Springer, Berlin, 2005.

[216] I. Toulopoulos. Stabilized space-time finite element methods of parabolic evolution problems. RICAM Report 2017-19, Johann Radon Institute for Computational and Applied Mathematics, Austrian Academy of Sciences, Linz, 2017.

[217] X. Tu. Three-level BDDC in three dimensions. *SIAM Journal on Scientific Computing*, 29(4):1759–1780, 2007.

[218] X. Tu. Three-level BDDC in two dimensions. *International journal for numerical methods in engineering*, 69(1):33–59, 2007.

[219] X. Tu and J. Li. BDDC for nonsymmetric positive definite and symmetric indefinite problems. In M. Bercovier, M. J. Gander, R. Kornhuber, and O. Widlund, editors, *Domain decomposition methods in science and engineering XVIII*, vol-

ume 70 of *Lecture Notes in Computational Science and Engineering*, pages 75–86. Springer, Berlin, 2009.

[220] C. O. U. Trottenberg and A. Schuller. *Multigrid*. Academic Press, San Diego, 2001.

[221] K. Urban and A. T. Patera. An improved error bound for reduced basis approximation of linear parabolic problems. *Math. Comp.*, 83(288):1599–1615, 2014.

[222] J. J. W. van der Vegt and S. Rhebergen. *hp*-multigrid as smoother algorithm for higher order discontinuous Galerkin discretizations of advection dominated flows: Part I. Multilevel analysis. *Journal of Computational Physics*, 231(22):7537–7563, 2012.

[223] J. J. W. van der Vegt and S. Rhebergen. *hp*-multigrid as smoother algorithm for higher order discontinuous Galerkin discretizations of advection dominated flows. Part II: Optimization of the Runge-Kutta smoother. *Journal of Computational Physics*, 231(22):7564–7583, 2012.

[224] J. J. W. van der Vegt and H. van der Ven. Space-time discontinuous Galerkin finite element method with dynamic grid motion for inviscid compressible flows. I: General formulation. *Journal of Computational Physics*, 182(2):546–585, 2002.

[225] T. Warburton and J. Hesthaven. On the constants in hp-finite element trace inverse inequalities. *Computer Methods in Applied Mechanics and Engineering*, 192(25):2765 – 2773, 2003.

[226] T. Weinzierl and T. Köppl. A geometric space-time multigrid algorithm for the heat equation. *Numerical Mathematics: Theory, Methods and Applications*, 5(1):110–130, 2012.

[227] O. B. Widlund and C. R. Dohrmann. BDDC deluxe domain decomposition. In T. Dickopf, M. J. Gander, L. Halpern, R. Krause, and L. F. Pavarino, editors, *Domain decomposition methods in science and engineering XXII*, volume 104 of *Lecture Notes in Computational Science and Engineering*, pages 93–103. Springer, Cham, 2016.

[228] B. Wohlmuth. A mortar finite element method using dual spaces for the Lagrange multiplier. *SIAM Journal on Numerical Analysis*, 38(3):989–1012, 2000.

[229] M. Wolfmayr. *Multiharmonic Finite Element Analysis of Parabolic Time-Periodic Simulation and Optimal Control Problems*. PhD thesis, Johannes Kepler University, Feb. 2014.

[230] G. Xu, B. Mourrain, R. Duvigneau, and A. Galligo. Analysis-suitable volume parameterization of multi-block computational domain in isogeometric applications. *Computer-Aided Design*, 45(2):395–404, 2013.

[231] G. Xu, B. Mourrain, R. Duvigneau, and A. Galligo. Constructing analysis-suitable parameterization of computational domain from cad boundary by variational harmonic method. *Journal of Computational Physics*, 252(Supplement C):275 – 289, 2013.

[232] S. Zampini. Dual-Primal methods for the cardiac bidomain model. *Mathematical Models and Methods in Applied Sciences*, 24(04):667–696, 2014.

[233] S. Zampini. Inexact BDDC methods for the cardiac bidomain model. In J. Erhel, M. J. Gander, L. Halpern, G. Pichot, T. Sassi, and O. Widlund, editors, *Domain decomposition methods in science and engineering XXI*, volume 98 of *Lecture Notes in Computational Science and Engineering*, pages 247–255. Springer, Cham, 2014.

[234] S. Zampini. PCBDDC: A Class of Robust Dual-Primal Methods in PETSc. *SIAM Journal on Scientific Computing*, 38(5):S282–S306, 2016.

[235] F. Zhang, Y. Xu, and F. Chen. Discontinuous Galerkin methods for Isogeometric Analysis for elliptic equations on surfaces. *Communications in Mathematics and Statistics*, 2(3):431–461, 2014.

[236] F. Zhang, Y. Xu, and F. Chen. Discontinuous Galerkin based Isogeometric Analysis for geometric flows and applications in geometric modeling. *Journal of Scientific Computing*, pages 1–22, 2016.

[237] W. Zulehner. Nonstandard norms and robust estimates for saddle point problems. *SIAM Journal on Matrix Analysis and Applications*, 32(2):536–560, 2011.

[238] G. Zumbusch. *Parallel multilevel methods. Adaptive mesh refinement and load-balancing.* Teubner, Wiesbaden, 2003.

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 sinngemäßentnommenen Stellen als solche kenntlich gemacht habe. Die vorliegende Dissertation ist mit dem elektronisch übermittelten Textdokument identisch.

Linz, April 2018

Christoph Hofer

Curriculum Vitae

Name: Christoph Hofer

Nationality: Austria

Date of Birth: 15. March, 1990

Place of Birth: Linz, Austria

Education:

1996–2000	Volksschule (elementary school) VS 35, Linz
2000–2008	Bundesrealgymnasium (secondary comprehensive school) BRG Fadingerstraße, Linz
2009–2012	Bachelorstudies Technical Mathematics, Johannes Kepler University, Linz
2012–2014	Masterstudies Industrial Mathematics, Johannes Kepler University, Linz
2014–2016	Doctoral Program in Engineering Sciences, NFN-Project "Geometry+Simulation", Subproject 03, Radon Institute of Computational and Applied Mathematics, Linz
since 2016	Doctoral Program in Engineering Sciences, Doctoral Program "Computational Mathematics", Project 04, Johannes Kepler University, Linz

Selected Presentations:

February 2017	Talk in Minisymposium "Parallel Solvers for Isogeometric Analysis" at the 24^{th} Domain Decomposition conference,

	Svalbard, Norway.
September 2017	Talk in Minisymposium "Efficient implementation of IGA" at the IGA 2017, Pavia, Italy.

Teaching

February 2016	Project instructor at Project Week "Applied Mathematics" for High School Students
Spring 2017	Tutorial "Numerical Methods for Elliptic PDEs"
Spring 2018	Tutorial "Numerical Methods for Elliptic PDEs"

Awards

2009–2013	Merit-based Scholarship, JKU Linz
2014	Würdigungspreis, Austrian Ministry for Science and Research (awarded to the best 50 graduates of Austrian universities per year)

Special Interest: Piano, Taekwondo, volunteering work for the ambulance (ASB)