Multilevel Solution Strategies for Unfitted Finite Element Methods

Doctoral Dissertation submitted to the
Faculty of Informatics of the Università della Svizzera Italiana
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

presented by
Hardik Kothari

under the supervision of
Prof. Rolf Krause

June 2020

Dissertation Committee

Prof. Christian Hesch	Universität Siegen, Germany
Prof. André Massing	Norwegian University of Science and Technology, Norway
Prof. Arnold Reusken	RWTH Aachen University, Germany
Prof. Igor Pivkin	Università della Svizzera italiana, Switzerland
Prof. Olaf Schenk	Università della Svizzera italiana, Switzerland

Dissertation accepted on 22 June 2020

Research Advisor	PhD Program Director
Prof. Rolf Krause	**Prof. Silvia Santini**

I certify that except where due acknowledgement has been given, the work presented in this thesis is that of the author alone; the work has not been submitted previously, in whole or in part, to qualify for any other academic award; and the content of the thesis is the result of work which has been carried out since the official commencement date of the approved research program.

Hardik Kothari
Lugano, 22 June 2020

To My Beloved Parents

The purpose of computing is
insight, not numbers.

Richard Hamming

Abstract

In the unfitted finite element methods, traditionally we can use Nitsche's method or the method of Lagrange multipliers to enforce the boundary/interface conditions. In this work, we present tailored multilevel methods for solving the problems stemming from either of these discretizations. Generally, multigrid methods require a hierarchy of finite element (FE) spaces which can be created geometrically using a hierarchy of nested meshes. However, in the unfitted FE framework, standard multigrid methods might demonstrate poor convergence properties if the hierarchy of FE spaces employed is not nested. We design a prolongation operator for the multigrid methods in such a way that it can accommodate the arbitrary shape of the boundaries/interfaces and recursively induces a nested FE space hierarchy. The prolongation operator is constructed using so-called pseudo-L^2-projections; as common, the adjoint of the prolongation operator is employed as the restriction operator. We employ this transfer operator in our multigrid method and solve the linear system of equations that arise from using Nitsche's method. In the numerical experiments, we show that our multigrid method is robust with respect to highly varying coefficients and the number of interfaces in a domain. It shows level independent convergence rates when applied to different variants of Nitsche's method.

Additionally, we present a generalized multigrid method for solving the problems stemming from the discretization of the interface conditions using Lagrange multipliers. This method can be used to solve the quadratic minimization problems with linear equality/inequality constraints, efficiently. The essential component of this multigrid method is the technique that decouples the linear constraints by projecting them into a new basis. The decoupled constraints are then handled by a modified version of the projected Gauss-Seidel method. By means of several numerical experiments, we exhibit the robustness of our multigrid method for the boundary or interface conditions with respect to varying coefficients. In addition, we demonstrate that this multigrid method can also handle the inequality constraints arising from the contact problems in the unfitted framework.

Acknowledgements

Firstly, I would like to express my gratitude to my advisor, Prof. Rolf Krause, for initiating me in this topic, for his patient supervision and support over the last years. Also, I am grateful to Prof. Krause for giving me the opportunity to present my work at various conferences and for the opportunity to attend several summer schools.

I am thankful to Prof. Arnold Reusken for inviting me to RWTH, Aachen for a scientific visit; and Thomas Ludescher for the fruitful discussions during the visit that helped me to refine my understanding of the unfitted finite element methods. Also, I am thankful to Prof. André Massing for his brief visit to Lugano, which helped me to improve my knowledge of the CutFEM method. I am also very grateful to Prof. Christian Hesch, Prof. André Massing, Prof. Igor Pivkin, Prof. Arnold Reusken, and Prof. Olaf Schenk for participating in the thesis committee.

I am thankful to my colleagues at the institute for a friendly working atmosphere, countless scientific discussions, and help over the last years. I would like to thank Dr. Simone Pezzuto for proofreading parts of this thesis and for many discussions that we had over the years.

I would like to thank Arne Hansen for his friendship, moral support, insightful discussions, and accompanying me in several sports activities to compensate for the academic workload.

Especially, I would like to thank Alena Kopaničáková for many scientific discussions, for proofreading this thesis, for help with the implementation of the generalized multigrid method in the Utopia library. I would also like to thank her for encouragement, entertainment, love, and for her support during my Ph.D.

Last but not least I am extremely grateful to my parents and my brother for their love, encouragement, support, and sacrifices they had to make for my education.

Contents

Figures

Tables

List of Algorithms

Chapter 1

Introduction

In the last two decades, unfitted finite element methods have become quite popular. An unfitted method can be defined as any method where the computational domain does not match the mesh exactly. The rise in the popularity of the unfitted method is due to the fact that the finite element method (FEM) [BS07, Bra07, LB13] poses certain challenges for modeling problems on the complex domains or for problems with static or dynamic discontinuities. For modeling the problems on complex domains, it is essential to generate a mesh that could explicitly represent the computational domain. While for the problems with a static discontinuity, it becomes essential to create a mesh, such that a discontinuity is resolved by the mesh. Whereas, for the problems with the dynamic discontinuities, it would become necessary to adapt the mesh over time as the discontinuity evolves. In many of these cases, it can be a cumbersome, time-consuming, and computationally demanding task to create high-quality meshes, and failing to do so can result in usually sub-optimal approximation properties of FEM. Geometrically unfitted methods overcome these problems, as they just require a background mesh and a finite element space defined on the background mesh. Clearly, the latter has to be modified to enforce the boundary conditions or the interface conditions. Here, an interface can be described as codimension one entity embedded in the domain, across which a function may exhibit non-smooth properties.

There is a huge variety of unfitted methods. The fictitious domain method can be listed as one of the oldest variants of an unfitted method [GPP94]. Later, Babuška and Melenk introduced the partition of unity finite element method which falls in the category of the meshless methods, as the meshes in the classical sense were not required [MB96, BM97]. Based on that work, Belytschko and Black proposed the idea of modeling crack propagation problems with minimal remeshing of the finite element mesh, where the nodes near the crack surfaces and the crack tip were enriched to describe the crack [BB99]. This work was extended by Moës et al. by introducing the Heaviside function and crack-tip function, and this new method is termed as eXtended Finite Element Method (XFEM) [MDB99]. Later, it was applied to the problems with the voids and inclusions by employing different types of enrichment functions [SCMB01, DMD+00]. Simultaneously, a similar approach to solve the interface problems without the need of remeshing was considered by Hansbo and Hansbo [HH02, HH04]. The methodology to enrich the background mesh used by Hansbo and Hansbo is very similar to the XFEM method introduced by Moës et al. [MCCR03], where the Heaviside functions are used with absolute shifted enrichments. In the work of Hansbo and Hansbo [HH02], in addition to the enrichments, Nitsche's method was utilized to enforce the interface conditions in the unfitted finite element framework. A similar XFEM approach is also considered for the two-phase flow problems in fluid dynamics to enrich the pressure variable [GRR06, Reu08, GR11]. Even though the XFEM has been introduced for problems in fracture mechanics with crack propagation, it has been later applied to many different problems [FB10, MDS17]. In addition to the interface problems, the unfitted methods have been widely applied in the context of fictitious domain methods [BH10, BH12, BH14]. The Finite Cell Method (FCM) can be considered as an extension of the fictitious domain method to higher-order function space [PDR07, SR15]. The FCM has been recently extended by employing the spline-based finite elements for harmonic and bi-harmonic problems [EDH10]. Another examples of unfitted methods can be given as the CutFEM method [BCH+15, CBM15], which normally emoploys a form of ghost penalty term to imporve the stability [Bur10]. In addition to these methods, we can also list other unfitted methods as the immersed boundary methods [Pes02, SDS+12], immersogeometric analysis [KHS+15], the trace finite element method [OR17], etc.

In the unfitted methods, a background mesh captures the computational domain of arbitrary shape, thus the elements are allowed to be cut arbitrarily by the boundaries or interfaces. This could give rise to a highly ill-conditioned system of linear equations. Due to this reason, it becomes essential to develop efficient solution strategies or the preconditioning strategies for solving the linear systems arising from the unfitted discretization methods. Tailored preconditioning methods for solving the interface problems with Nitsche based XFEM discretization were proposed and studied in [LMDM14, LR17, GLOR16].

In this work, we focus on developing the robust multilevel solution and preconditioning strategies in the unfitted finite element framework. Multilevel methods are ideal iterative solvers for many large-scale linear/nonlinear problems, as they are of optimal complexity [Hac86, TOS00, BvEHM00]. The optimal complexity implies that the convergence rate of the multilevel methods is bounded independently from the size of the problem, and the amount of numerical operations done in the algorithm is proportional to the size of the problem. The robustness of multilevel iteration results from a sophisticated combination of smoothing iterations and coarse level corrections. Ideally, these components are complementary to each other as they reduce errors in different parts of the spectrum. Traditionally, the mesh hierarchy for multilevel methods is created by either coarsening or refinement strategies, and a simple interpolation operator and its adjoint are used to transfer the information between different levels.

There have been some efforts to develop multilevel solution strategies for the XFEM discretization. Initial approaches propose to modify the algebraic multigrid method (AMG). A domain decomposition-based AMG preconditioner is proposed for the fracture problems [BVWH$^+$12], where the domain is decomposed into 'cracked' and 'intact' domain and AMG is applied to the 'intact' domain, and the 'cracked' domain is solved with a direct solver. Another approach using AMG for the XFEM discretization for the fracture problems is exploited in [GT13]. In an alternative approach, known as a quasi-algebraic multigrid method, the sparsity pattern of the interpolation operator is modified to prevent the interpolation across the interfaces [HTW$^+$12]. Recently, a new multigrid method is also proposed for the elliptic interface problems, with an interface smoother in [lud20].

1.1 Overview

The main objective of this thesis is to develop efficient multilevel methods for solving the optimization problems arising from the unfitted finite element discretizations.

We start our discussion with a presentation of the discretization framework for the unfitted FEM. We introduce the XFEM with first-order finite element spaces for discretizing the fictitious domain and the interface problems. Here, we focus on the strategies for enforcing the boundary conditions and the interface conditions in the unfitted FEM framework. To enforce the boundary/interface conditions in a weak sense, we utilize the penalty method, Nitsche's method, and the method of Lagrange multipliers. Next, we discuss in detail the numerical challenges posed by each method and evaluate the robustness of these methods by comparing their discretization errors and the condition numbers of arising linear systems.

Then, we propose tailored multigrid methods for solving the system of equations arising from Nitsche's method and the method of Lagrange multipliers. The multigrid method requires transfer operators to transfer the information between the mesh hierarchy. In the unfitted framework, the mesh hierarchy is generally created by the either coarsening or refinement of the background meshes. Thus, even though the background meshes are nested, the meshes associated with the computational domain may not be nested. Hence, we propose a new transfer operator for the unfitted meshes computed using L^2-projection and pseudo-L^2-projection [DK11, DK14, KK19]. Additionally, the system of equations arising from the method of Lagrange multipliers has a saddle point structure, which can also be formulated as a quadratic minimization problem with linear equality constraints. Here, we introduce a new generalized multigrid method for solving such problems. This generalized multigrid method is an extension of the monotone multigrid method [Kor94, Kor96, KK01], which was introduced for solving quadratic minimization problems with pointwise constraints.

As a culmination of this thesis, we employ the unfitted finite element methods for solving contact problems. We use the method of Lagrange multipliers to discretize the non-penetration condition for contact problems. In addition, we use our novel generalized multigrid method for solving the quadratic minimization problems with linear inequality constraints.

Here, we provide a brief list of the contributions made in this thesis:

- We compare the penalty method, Nitsche's method, and the method of Lagrange multipliers used for enforcing the boundary/interface conditions. We evaluate the performance of these methods by comparing their discretization errors and condition numbers of the linear system by means of several numerical examples.

- We introduce a new transfer operator in the unfitted framework. We evaluate the robustness of this transfer operator by employing it in the semi-geometric multigrid method for solving the linear systems, which arise from fictitious domain method and interface problems when using the penalty method or Nitsche's method.

- We evaluate the performance of the standard iterative methods for solving the saddle point problems that arise due to the method of Lagrange multipliers. We also assess the performance of the semi-geometric multigrid method as a solution strategy for solving the primal problem.

- We introduce a generalized multigrid method for solving the saddle point problem, or an equivalent quadratic minimization problem with linear constraints, arising from the method of Lagrange multipliers. In order to handle the linear constraints, we propose a technique to decouple the constraints and introduce a variant of the projected Gauss-Seidel method.

- Finally, we employ all these ingredients for solving the contact problem in the unfitted framework. We use the method of Lagrange multipliers to enforce the non-penetration conditions for Signorini's problem and two-body contact problems. In the end, we extend the generalized multigrid method for solving contact problems.

1.2 Function Spaces

In this work, we discuss the finite element method for solving the second-order partial differential equations. The Sobolev spaces are a natural choice for the variational problems of this kind. Here, we give an introduction to the notations used in this work and later give a short description of the Sobolev spaces. For the detailed review of the Sobolev spaces, we refer to the monographs [BS07, Bra07, Sal08].

Notations: In this work, the scalar quantities, such as functions, constants, operators are denoted by lower case and upper case characters e.g., u, v, f, C. The vector quantities are denoted by bold symbols in lower case characters e.g., $\boldsymbol{a}, \boldsymbol{b}$, and the matrix quantities are denoted by bold symbols in upper case characters e.g., $\boldsymbol{A}, \boldsymbol{B}$. We denote a matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, where the symbols $\mathbb{R}^{m\times n}$ denote the set of $m \times n$ matrices with real entries. For a given matrix $\boldsymbol{A}$, its transpose is denoted by $\boldsymbol{A}^{\mathsf{T}}$. The components of these vector and matrix quantities are given as a_i, b_j, A_{ij}, B_{kl} for some indices i, j, k, l. By symbols $\mathcal{V}$, $\mathcal{W}$, we denote real normed function spaces. Given a function space $\mathcal{V}$, we denote its dual space that contains all bounded linear functionals by $\mathcal{V}^*$. We denote the vector-valued function spaces by $\boldsymbol{\mathcal{V}} = (\mathcal{V})^d$, where $d \in \{2, 3\}$.

The Euclidean inner product is defined as $\boldsymbol{u} \cdot \boldsymbol{v} := \sum_i u_i v_i$ for $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^n$. We define scalar energy product $(\cdot, \cdot)_A$ as $(\boldsymbol{u}, \boldsymbol{v})_A := \boldsymbol{u} \cdot \boldsymbol{A}\boldsymbol{v}$, for all $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^n$ and the induced energy norm is defined as $\|\cdot\|_A^2 := (\cdot, \cdot)_A$. Additionally, we define the Kronecker delta δ_{ij} for some indices i, j by

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{if } i \neq j. \end{cases} \tag{1.1}$$

Sobolev Spaces: We consider a simple and connected domain $\Omega \subset \mathbb{R}^d$ for $d \in \{2,3\}$. The domain Ω is an open and bounded in the Euclidean space with Lipschitz boundary $\partial\Omega$ and the outer normal vector from the domain is defined as $\boldsymbol{n}$. We assume, the boundary $\partial\Omega$ can be decomposed into two subsets; the closed Dirichlet boundary $\partial\Omega_D$ and the open Neumann boundary $\partial\Omega_N$, such that

$$\overline{\partial\Omega_D} \cup \overline{\partial\Omega_N} = \overline{\partial\Omega} \quad \text{and} \quad \partial\Omega_D \cap \partial\Omega_N = \emptyset.$$

We denote the elements of Ω by $\boldsymbol{x} = (x_1, x_2, \dots, x_d)$. On the domain Ω, a scalar valued function is defined as $u : \Omega \to \mathbb{R}$ and a vector valued function is defined as $\boldsymbol{v} : \Omega \to \mathbb{R}^d$.

We define a multi-index as a d-tuple $a = (a_1, a_2, \dots, a_d), \forall a_i \in \mathbb{N}$. The length of a multi-index is given by $|a| := \sum_{1 \leqslant i \leqslant d} a_i$. The a-th derivative of order $|a|$ is denoted by

$$\partial^a = \frac{\partial^{a_1}}{\partial x_1^{a_1}} \frac{\partial^{a_2}}{\partial x_2^{a_2}} \cdots \frac{\partial^{a_d}}{\partial x_d^{a_d}}.$$

Let $L^2(\Omega)$ be a Lebesgue space of square-integrable function on the domain Ω. The inner product on $L^2(\Omega)$ is denoted as $(u, v)_{L^2(\Omega)} := \int_\Omega u(\boldsymbol{x}) v(\boldsymbol{x}) \, d\Omega$, and the induced L^2-norm is defined as $\| \cdot \|^2_{L^2(\Omega)} := (\cdot, \cdot)_{L^2(\Omega)}$. The symbol $L^\infty(\Omega)$ denotes the space of essentially bounded function with norm $\|v\|_{L^\infty(\Omega)} := \text{ess sup}_{\boldsymbol{x} \in \Omega} |v(\boldsymbol{x})|$.

Additionally, by $H^k(\Omega)$, we denote the Sobolev space of function with $k \geqslant 0$ square-integrable weak derivatives on the domain Ω. We note, $L^2(\Omega) = H^0(\Omega)$. Then ∂^a denotes the weak differentiation and the corresponding norm in $H^k(\Omega)$ are given as

$$\|v\|^2_{H^k(\Omega)} := \|v\|^2_{L^2(\Omega)} + \sum_{1 \leqslant |a| \leqslant k} \|\partial^a v\|^2_{L^2(\Omega)}.$$

The space $H^k(\Omega)$ is a Hilbert space with respect to the scalar product

$$(u, v)_{H^k(\Omega)} = (u, v)_{L^2(\Omega)} + \sum_{1 \leqslant |a| \leqslant k} (\partial^a u, \partial^a v)_{L^2(\Omega)}.$$

In this work, we use several subspaces of $H^1(\Omega)$ and $L^2(\Omega)$. We denote the subspace of $H^1(\Omega)$ with all the function that vanish on the Dirichlet boundary $\partial\Omega_D$ is denoted as $H^1_D(\Omega)$, we have

$$H^1_D(\Omega) = \{v \in H^1(\Omega) : v|_{\partial\Omega_D} = 0\},$$

and if the whole boundary is Dirichlet boundary $\partial\Omega_D = \partial\Omega$, we have

$$H^1_0(\Omega) = \{v \in H^1(\Omega) : v|_{\partial\Omega} = 0\}.$$

When the boundary $\partial\Omega$ is Lipschitz continuous, we introduce a linear and continuous operator

$$\gamma_0(\cdot) : H^1(\Omega) \to H^{\frac{1}{2}}(\partial\Omega).$$

The operator γ_0 is surjective and it is known as a trace operator. For simplicity, we denote the restriction of a function u on the boundary $\partial\Omega$ by $u|_{\partial\Omega}$. Given $u \in H^1(\Omega)$, the image of the trace operator $\gamma_0(u)$ coincides with the restriction of u to the boundary $\partial\Omega$, given as $\gamma_0(u) = u|_{\partial\Omega}$, which we refer to as trace of u on the boundary $\partial\Omega$. In addition, the dual of the space of $H^{\frac{1}{2}}(\partial\Omega)$ is denoted as $H^{-\frac{1}{2}}(\partial\Omega)$, and the duality paring between these two spaces is given as

$$\langle \cdot, \cdot \rangle_{\partial\Omega} : H^{-\frac{1}{2}}(\partial\Omega) \times H^{\frac{1}{2}}(\partial\Omega) \to \mathbb{R}.$$

1.3 Outline

This thesis is organized as follows.

- In Chapter 2, we introduce the fictitious domain method in the unfitted finite element framework. We introduce the penalty method, Nitsche's method, and the method of Lagrange multipliers for enforcing the Dirichlet boundary conditions. Also, we review various approaches for estimating the value of the stabilization parameter used in Nitsche's method. In addition, we also discuss the necessity of satisfying the inf-sup condition and introduce the vital vertex algorithm to construct a stable multiplier space.

- In Chapter 3, we introduce the interface problem as overlapping fictitious domains in the XFEM framework. Our presentation is given as an extension of Nitsche's method and the method of Lagrange multipliers from enforcing boundary conditions to enforcing interface conditions.

- In Chapter 4, we review the multigrid method as subspace correction method. Later, we introduce the multilevel framework for creating a nested hierarchy of the FE space from a hierarchy of the background meshes. We present the variational transfer approach to compute the transfer operators by means of L^2-projections and pseudo-L^2-projections.

- In Chapter 5, we discuss standard iterative methods for solving saddle point problems and review some preconditioners for the dual systems. Then, we introduce the generalized multigrid method for solving the quadratic minimization problem with linear equality constraints.

- In Chapter 6, we introduce contact problems in the unfitted framework. We use the method of Lagrange multipliers for imposing the non-penetration condition on the unfitted interface and we use our generalized multigrid method for solving the contact problems.

Chapter 2

Fictitious Domain Method

The fictitious domain method is among one of the earliest variants of unfitted methods [GPP94, GG95]. This method was introduced to simplify the process of numerically solving partial differential equations on complex domains using regular structured meshes. In this approach, the solution of the problem is defined only up to the boundary of the computational domain. Thus, the weak formulation of a problem is defined only on the elements that are located in the interior of the domain. This method is a subset of a Galerkin method, and it inherits the approximation properties of the standard finite element methods.

The unfitted method simplifies the task of creating high-quality meshes that fit the domains, but it gives rise to some other challenges. Firstly, we have to store various details about the actual computational domain and identify the part of the background mesh that is associated with the domain. Secondly, we have to pay attention to the numerical integration of the elements that are intersected by the boundary and are only partially inside of the domain. The last challenge is concerning the imposition of the Dirichlet boundary conditions. As the boundary of the computational domain does not fit the background mesh exactly, we have to enforce the Dirichlet boundary conditions in a weak sense.

In this chapter, we provide an introduction to the fictitious domain method in the extended finite element framework and discuss various methods to enforce the boundary conditions. The concepts outlined in this chapter are extended in the next chapter for tackling the interface problems. Additionally, in order to highlight elements of the discretization method, we limit our presentation in this chapter to a model diffusion problem.

In Section 2.1, we define a diffusion problem on a domain Ω. The assumptions on the background mesh and the finite element function spaces are discussed in Section 2.2. In Section 2.3, we give a weak formulation of the diffusion problem and pose the problem in an optimization framework with Dirichlet boundary conditions as constraints. Further, we investigate different strategies to enforce the Dirichlet boundary conditions in the XFEM framework, e.g., the penalty method, the method of Lagrange multipliers, and Nitsche's method in Section 2.4. Later, we discuss the stabilization parameter in Nitsche's method which plays an important role in establishing the coercivity of the bilinear form. In Section 2.5, we discuss strategies to implicitly estimate the value of the stabilization parameter. Then, we discuss a method to create a stable space for the Lagrange multipliers in Section 2.6. In the last section, we perform some numerical experiments to show the convergence properties of the penalty method, Nitsche's method, and the method of Lagrange multipliers for different test examples.

2.1 Model Problem

We assume a bounded domain $\Omega \subset \mathbb{R}^d$, $d \in \{2, 3\}$ with the boundary $\partial\Omega$. In this section, we denote the boundary as $\Gamma := \partial\Omega$. The boundary Γ is assumed to be Lipschitz continuous. We consider the following diffusion problem as a model problem.

Given the data $f \in L^2(\Omega)$ and a function $g_D \in H^{\frac{1}{2}}(\Gamma)$, find a function u such that

$$\begin{aligned} -\nabla \cdot \alpha(\boldsymbol{x})\nabla u &= f \quad \text{in } \Omega, \\ u &= g_D \quad \text{on } \Gamma, \end{aligned} \tag{2.1}$$

where the coefficient $\alpha : \Omega \to \mathbb{R}^+$ is piecewise constant satisfying

$$\alpha(\boldsymbol{x}) \geqslant \alpha_0 > 0.$$

With a slight abuse of notation, we write $\alpha = \alpha(\boldsymbol{x})$. The model problem (2.1) has a unique solution $u \in H^1(\Omega)$ that satisfies

$$\|u\|_{H^1(\Omega)} \leqslant C(\|f\|_{L^2(\Omega)} + \|g_D\|_{H^{\frac{1}{2}}(\Gamma)}).$$

The weak formulation of the problem (2.1), which has inhomogeneous Dirichlet boundary conditions, is given as, find $u \in H^1_0(\Omega)$ such that

$$(\alpha\nabla u, \nabla v)_{L^2(\Omega)} = (f, v)_{L^2(\Omega)} - (\alpha\nabla u_g, \nabla v)_{L^2(\Omega)} \quad \forall v \in H^1_0(\Omega), \tag{2.2}$$

where $u_g \in H^1(\Omega)$ with $u_g|_\Gamma = g_D$ in the sense of trace theorem.

We remark that the formulation (2.2) is equivalent to the following minimization problem. Find $u \in H_0^1(\Omega)$ such that

$$\min_{u \in H_0^1(\Omega)} \frac{1}{2}(\alpha \nabla(u + u_g), \nabla u)_{L^2(\Omega)} - (f, u)_{L^2(\Omega)}. \tag{2.3}$$

In both of the above formulations (2.2) and (2.3), the solution is sought in the $H_0^1(\Omega)$ space.

2.2 Finite Element Discretization

In the previous section, the weak formulation is still given in a continuous sense. In this section, we introduce the necessary ingredients for the discretization of the diffusion problem in the extended finite element framework.

2.2.1 Background Mesh

We assume a domain $\mathscr{D}$, that encapsulates the computational domain Ω, i.e., $\Omega \subset \mathscr{D}$. We define shape regular, quasi-uniform, conforming triangulation on the background domain. The partition of the background domain $\mathscr{D}$ is given by simplicies $K_i \in \mathbb{R}^d$ such that

$$\mathscr{D}_h = \{K_1, K_2, \dots\}.$$

Now, we can define the closure of the background domain as union of closure of all simplicies, given as

$$\overline{\mathscr{D}}_h = \{\overline{K}_1, \overline{K}_2, \dots\} = \cup_{K \in \mathscr{D}_h} \overline{K}.$$

The background triangulation which is associated with our domain is defined as follows:

$$\widetilde{\mathcal{T}}_h = \overline{\mathscr{D}}_h \backslash \partial \overline{\mathscr{D}}_h.$$

The domain Ω is encapsulated by the mesh, $\Omega \subset \widetilde{\mathcal{T}}_h$, but the mesh is not fitted to the boundary of the domain. The boundary of the domain is resolved sufficiently well by the mesh $\widetilde{\mathcal{T}}_h$ and the curvature of the boundary is bounded.

We use $\widetilde{\mathcal{T}}_h$ as a background triangulation which captures the domain. Let h_K be the diameter of the element K, and mesh size is defined as $h = \max_{K \in \widetilde{\mathcal{T}}_h} h_K$. We define so-called active mesh, which is strictly intersected by the domain Ω as

$$\mathcal{T}_h = \{K \in \widetilde{\mathcal{T}}_h : K \cap \Omega \neq \emptyset\}.$$

The active mesh $\mathcal{T}_h$ excludes all the elements that are neither intersected by the boundary Γ nor are in the interior of the domain. In Figures (2.1a, 2.1b), we can see an example of a background mesh $\widetilde{\mathcal{T}}_h$ and an active mesh $\mathcal{T}_h$ that encapsulates a circular domain.

We define a set of elements that are intersected by the boundary Γ as

$$\mathcal{T}_{h,\Gamma} = \{K \in \widetilde{\mathcal{T}}_h : K \cap \Gamma \neq \emptyset\}.$$

For all elements $K \in \mathcal{T}_{h,\Gamma}$, let $K_\Omega := K \cap \Omega$ be part of K in domain Ω. While for all elements $K \in \mathcal{T}_h \setminus \mathcal{T}_{h,\Gamma}$ are strictly in the interior of domain Ω. For all $K \in \mathcal{T}_{h,\Gamma}$, let $\Gamma_K := \Gamma \cap K$ be part of Γ in K. Additionally, we define the set of faces $\mathcal{G}_{h,\Gamma}$

$$\mathcal{G}_{h,\Gamma} = \{G \subset \partial K : \partial K \cap \partial \mathcal{T}_h = \emptyset, \forall K \in \mathcal{T}_{h,\Gamma}\}.$$

This set includes all the faces that are associated with the cut elements, except the ones that are on the boundary of the mesh $\partial \mathcal{T}_h$. In Figure 2.1c, we can see the distinction between the elements of mesh $\mathcal{T}_{h,\Gamma}$ and the elements of mesh $\mathcal{T}_h \backslash \mathcal{T}_{h,\Gamma}$. While in Figure 2.1d we can see the set of faces that belong to the set $\mathcal{G}_{h,\Gamma}$.

2.2.2 Extended Finite Element Space

We define a continuous first order finite element (FE) space over the triangulation $\widetilde{\mathcal{T}}_h$ as

$$\widetilde{\mathcal{V}}_h = \{v \in H^1(\widetilde{\mathcal{T}}_h) : v|_K \in \mathcal{P}_1(K),\ \forall K \in \widetilde{\mathcal{T}}_h\},$$

where $\mathcal{P}_1$ denotes the space of piecewise linear functions. Following the original XFEM literature [MDB99], we define a characteristic function of the computational domain Ω, given as

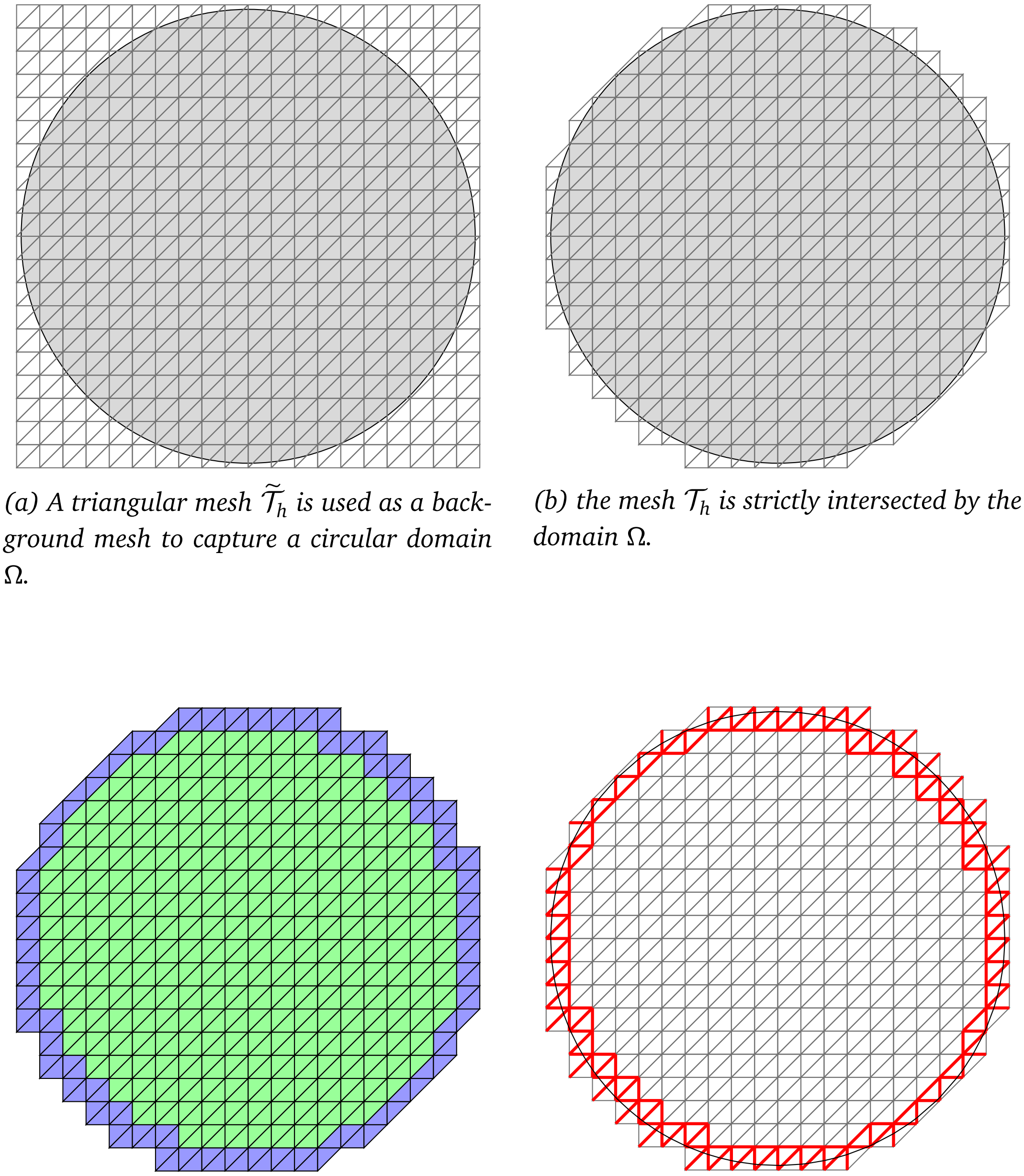
$$\chi_\Omega : \mathbb{R}^d \to \mathbb{R}, \qquad \chi_\Omega(\boldsymbol{x}) = \begin{cases} 1 & \forall \boldsymbol{x} \in \overline{\Omega}, \\ 0 & \text{otherwise.} \end{cases} \tag{2.4}$$

The characteristic function is used to restrict the support of the finite element space $\widetilde{\mathcal{V}}_h$ to the domain Ω. Thus, the space of finite elements in the domain Ω is defined as

$$\mathcal{V}_h = \chi_\Omega(\boldsymbol{x}) \widetilde{\mathcal{V}}_h. \tag{2.5}$$

We obtain the “cut” basis associated with a node p as,

$$\phi_h^p = \chi_\Omega(\boldsymbol{x}) \widetilde{\phi}_h^p \qquad \forall p \in \mathcal{N}_h.$$



(a) A triangular mesh $\widetilde{\mathcal{T}}_h$ is used as a background mesh to capture a circular domain Ω.

(b) the mesh $\mathcal{T}_h$ is strictly intersected by the domain Ω.

(c) The mesh $\mathcal{T}_{h,\Gamma}$ is shaded in blue, while the interior mesh $\mathcal{T}_h \backslash \mathcal{T}_{h,\Gamma}$ is shaded in green.

(d) The set of faces $\mathcal{G}_{h,\Gamma}$ is shown in the red.

Figure 2.1. An example of a domain Ω with a background mesh $\widetilde{\mathcal{T}}_h$.

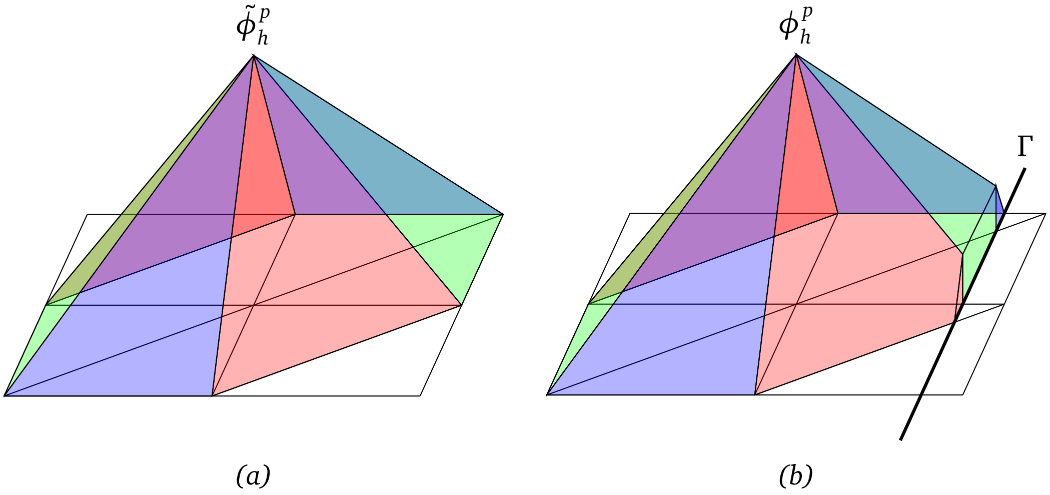


Figure 2.2. (a) The basis function $\widetilde{\phi}_h^p$ associated with a node p, (b) The "cut" basis function $\phi_h^p = \chi_{\Omega_i}(\boldsymbol{x})\widetilde{\phi}_h^p$ associated with the node p

Here, $\widetilde{\mathcal{N}_h}$ denotes the set of nodes of the active mesh $\widetilde{\mathcal{T}}_h$. Now, the set of nodes associated with the domain Ω can be given as

$$\mathcal{N}_h = \{p \in \widetilde{\mathcal{N}_h} : \operatorname{supp}(\phi_h^p) \cap \Omega \neq \emptyset\}.$$

The FE space $\mathcal{V}_h$ is spanned by the nodal basis functions given as, $\Phi_h = (\phi_h^p)_{p\in\mathcal{N}_h}$. In Figure 2.2, we can see the basis function $\widetilde{\phi}_h^p$ associated with the original function space $\widetilde{\mathcal{V}}_h$ and corresponding cut basis function ϕ_h^p associated with the FE space $\mathcal{V}_h$.

2.3 Variational Formulation

Now, we discretize the problem (2.1) using the FE space $\mathcal{V}_h$ defined on the active mesh $\mathcal{T}_h$. We write the discretized variational formulation of the diffusion problem (2.1) as a constrained minimization problem. Find $u_h \in \mathcal{V}_h$ such that

$$\begin{aligned} \min_{u_h\in\mathcal{V}_h} \qquad & J(u_h) = \frac{1}{2}a(u_h,u_h) - F(u_h) \\ \text{subject to} \qquad & u_h = g_D \qquad \text{on } \Gamma, \end{aligned} \tag{2.6}$$

where $J(\cdot) : \mathcal{V}_h \to \mathbb{R}$ denotes the energy functional, $a(\cdot,\cdot) : \mathcal{V}_h \times \mathcal{V}_h \to \mathbb{R}$ is a continuous, symmetric and coercive bilinear form and $F(\cdot) : \mathcal{V}_h \to \mathbb{R}$ denotes a continuous linear form. The bilinear and the linear forms are defined as

$$\begin{aligned} a(u_h, v_h) &:= (\alpha \nabla u_h, \nabla v_h)_{L^2(\Omega)}, \\ F(v_h) &:= (f, v_h)_{L^2(\Omega)}. \end{aligned}$$

We remark that the formulation (2.3) is posed as an unconstrained minimization problem, while (2.6) is formulated as a constrained minimization problem. In the unfitted finite element framework, it is not possible to impose the Dirichlet boundary condition in a pointwise manner as the mesh is not fitted to the boundary of the domain. Thus, we have to rely on other methods to impose the Dirichlet boundary conditions in a weak manner. We also mention that, we seek the solution of (2.3) in $H_0^1(\Omega)$ space while for (2.6) the solution is sought in a larger space $\mathcal{V}_h \subseteq H^1(\Omega)$.

Before discussing the weak formulation in the XFEM framework we need to define the appropriate norms. We define mesh-dependent inner products on the boundary as

$$\begin{aligned} (u, v)_{H^{\frac{1}{2}}(\Gamma),h} &:= \sum_{K \in \mathcal{T}_{h,\Gamma}} (h_K^{-1} u, v)_{L^2(\Gamma_K)}, \\ (u, v)_{H^{-\frac{1}{2}}(\Gamma),h} &:= \sum_{K \in \mathcal{T}_{h,\Gamma}} (h_K u, v)_{L^2(\Gamma_K)}. \end{aligned}$$

The induced mesh-dependent norms at the boundary are given as

$$\begin{aligned} \|v\|^2_{H^{\frac{1}{2}}(\Gamma),h} &= \sum_{K \in \mathcal{T}_{h,\Gamma}} \frac{1}{h_K} \|v\|^2_{L^2(\Gamma_K)}, \\ \|v\|^2_{H^{-\frac{1}{2}}(\Gamma),h} &= \sum_{K \in \mathcal{T}_{h,\Gamma}} h_K \|v\|^2_{L^2(\Gamma_K)}. \end{aligned} \tag{2.7}$$

We recall, $\langle \cdot, \cdot \rangle_\Gamma : H^{-\frac{1}{2}}(\Gamma) \times H^{\frac{1}{2}}(\Gamma) \to \mathbb{R}$, denotes a duality paring on the boundary. On the duality pairing, we have the following inequality due to Cauchy-Schwarz

$$\langle u, v \rangle_\Gamma \leqslant \|u\|_{H^{-\frac{1}{2}}(\Gamma),h} \|v\|_{H^{\frac{1}{2}}(\Gamma),h}. \tag{2.8}$$

For the FE space $\mathcal{V}_h$ an important trace inequality on each element $K \in \mathcal{T}_{h,\Gamma}$ is given as

$$\|\alpha \nabla_{\boldsymbol{n}} u_h\|^2_{H^{-\frac{1}{2}}(\Gamma_K),h} \leqslant C_\gamma \|\alpha^{\frac{1}{2}} \nabla u_h\|^2_{L^2(K_\Omega)}, \tag{2.9}$$

where, $\nabla_{\boldsymbol{n}} u = \boldsymbol{n} \cdot \nabla u$. Here, we note that the constant C_γ depends on the shape regularity of element K_Ω not the background element K.

2.3.1 Causes of Ill-conditioning

In the unfitted methods, a background mesh captures the computational domain of arbitrary shape, hence the elements are allowed to be cut arbitrarily by the boundary. In general, this flexibility can result in disproportionally cut elements that might not be shape regular anymore. For an element $K \in \mathcal{T}_h$, the bound on the gradient of a function is established by the following inverse inequality relation

$$\|\nabla u_h\|_{L^2(K)} \leqslant C h_K^{-1} \|u_h\|_{L^2(K)}, \tag{2.10}$$

where, the constant C depends on the shape regularity of element K. Now, for the cut elements, $\forall\ K \in \mathcal{T}_{h,\Gamma}$, the constant C depends on the shape regularity of element K_Ω and on the mesh size of the element h_K, which can become arbitrarily small. Therefore, the bounds on the gradients of the function can become arbitrarily weak depending on the location of the boundary with respect to the background mesh. The inverse inequality relation (2.10) is necessary to provide the bounds for the condition number estimates of the system matrix, hence a large value of the constant C in the above inequality gives rise to linear systems of equation with large condition numbers. Indirectly, the condition number of the system matrix depends on the cut position and the conditioning of the linear system may become arbitrarily poor when an interface passes very close to element faces or nodes. In order to tackle the issue of poor conditioning of the system matrix, the ghost penalty method was introduced.

2.3.2 Ghost Penalty Stabilization

In this section, we introduce a new stabilization term, called the ghost penalty, in the variational formulation. The ghost penalty term was introduced to recover the control over the gradients of the function on the cut elements [Bur10]. The ghost penalty method was introduced in the context of Nitsche's method to improve the robustness of the method irrespective of the location of the boundary [BH12, Bur10]. The idea of such a stabilization term was used for the problems with dominant transport to penalize the jumps in the normal derivative across the interior faces of elements [DD76]. This type of penalty term was recently also applied in the context of convection-diffusion-reaction problem [BH04], Stoke's problem [BH06] and in the XFEM context for incompressible elasticity problems to penalize the jump in pressure [BBH09].

The ghost penalty term consists of a least-square penalization of the flux jumps across the element boundaries, which weakly enforces the continuity of normal flux at the element interfaces in the neighborhood of the boundary. This penalty term has to be chosen in such a way that it provides sufficient stability and it stays weakly consistent with the original formulation for smooth solutions. The ghost penalty term is enforced on the set of faces $\mathcal{G}_{h,\Gamma}$, and it is defined as

$$g(u_h, v_h) = \sum_{G \in \mathcal{G}_{h,\Gamma}} \epsilon_G h_G \alpha([\![\nabla_{\boldsymbol{n}_G} \mathcal{E}_h u_h]\!], [\![\nabla_{\boldsymbol{n}_G} \mathcal{E}_h v_h]\!])_{L^2(G)}, \tag{2.11}$$

where h_G is the diameter of face G, $\boldsymbol{n}_G$ denotes unit normal to face G, and ϵ_G is a positive constant. Here, $\mathcal{E}_h$ denotes the canonical extension of the function from the domain to the background mesh, which is defined as $\mathcal{E}_h : \mathcal{V}_h|_{K_\Omega} \to \widetilde{\mathcal{V}}_h|_K$.

By adding the ghost penalty term, we regain the control over the gradients of the function on the cut elements with very small support and by extension we overcome the issue of ill-conditioning. The coercivity of the bilinear form $a(\cdot,\cdot)$ in the discrete sense is defined only up to the boundary of the computational domain. Adding the stabilization term to the bilinear form $a(\cdot,\cdot)$ extends the coercivity from the computational domain to the active mesh,

$$a(v_h, v_h) + g(v_h, v_h) \geqslant C_s \sum_{K \in \mathcal{T}_h} \|\alpha^{\frac{1}{2}} \nabla v_h\|^2_{L^2(K)}.$$

In [Bur10], it is shown that the extended coercivity is enough to ensure a uniform upper bound on the condition number of the system matrix. The condition number of the system matrix associated with the updated bilinear form does not depend on the location of the boundary with respect to the background mesh.

Even though, the ghost penalty stabilization term was introduced in the context of Nitsche's method, we add this term to the bilinear form regardless of the method chosen to enforce the Dirichlet boundary condition. We modify the energy formulation in (2.6), by introducing an additional ghost penalty stabilization term (2.11). The modified energy functional is given as

$$J(u_h) := \frac{1}{2}\Big(a(u_h, u_h) + g(u_h, u_h)\Big) - F(u_h). \tag{2.12}$$

Since its introduction, the ghost penalty term has become quite popular in the unfitted finite element framework. The ghost penalty terms have been used for different variants of the unfitted finite element framework e.g., in so-called CutFEM methods[BCH$^+$15], cut discontinuous Galerkin methods [GM18, GM19].

2.4 Enforcing Boundary Conditions

In this section, we discuss different strategies for enforcing the Dirichlet boundary condition. We reformulate the problem (2.6) with the modified energy functional (2.12) that includes the ghost penalty term. Thus, the updated minimization problem is given below. Find $u_h \in \mathcal{V}_h$ such that

$$\begin{aligned} \min_{u_h \in \mathcal{V}_h} \qquad & J(u_h) = \frac{1}{2}\Big(a(u_h, u_h) + g(u_h, u_h) \Big) - F(u_h) \\ \text{subject to} \qquad & u_h = g_D \qquad \text{on } \Gamma. \end{aligned} \tag{2.13}$$

In this section, we consider a constrained optimization framework to derive a weak formulation of the above problem.

2.4.1 The Penalty Method

The idea of enforcing the Dirichlet boundary condition in the finite element framework in a weak sense dates back to the work of Babuška [Bab73b]. The penalty method is one of the simplest ways to recast an equality constrained minimization problem to an unconstrained minimization problem. This is done by adding an extra term to the energy functional which penalizes the violation of the constraints.

Here, we consider a quadratic penalty method that penalizes the constraints in a least squares sense. The modified energy functional is given as

$$\min_{u_h \in \mathcal{V}_h} J_P(u_h) := J(u_h) + \frac{\gamma_p}{2} \|u_h - g_D\|^2_{H^{\frac{1}{2}}(\Gamma),h},$$

where $\gamma_p \in \mathbb{R}^+$ is the penalty parameter. If the penalty parameter is chosen to be large enough, then the solution of the above minimization problem leads to a solution that satisfies the Dirichlet boundary condition approximately.

The Euler-Lagrange condition corresponding to the above minimization problem yields:

$$\text{find } u_h \in \mathcal{V}_h \text{ such that} \qquad A_P(u_h, v_h) = F_P(v_h) \qquad \forall v_h \in \mathcal{V}_h. \tag{2.14}$$

Here, the bilinear functional $A_P(\cdot,\cdot)$ and the linear functional $F_P(\cdot)$ are defined as

$$\begin{aligned} A_P(u_h, v_h) &= a(u_h, v_h) + g(u_h, v_h) + (\gamma_p u_h, v_h)_{H^{\frac{1}{2}}(\Gamma),h}, \\ F_P(v_h) &= F(v_h) + (\gamma_p g_D, v_h)_{H^{\frac{1}{2}}(\Gamma),h}. \end{aligned}$$

This formulation can be also achieved by reformulating the Dirichlet boundary condition to a Neumann or a Robin type condition and then using the Neumann boundary condition in the Green's formula for the problem (2.1) [BE86].

Although the penalty method is trivial to implement, it is not widely used since it is not consistent with the strong formulation (2.1). The system matrix associated with the bilinear functional $A_P(\cdot,\cdot)$ can produce a highly ill-conditioned system if a large penalty parameter is chosen. The convergence of the discretization error depends on the value of the penalty parameter. If a large enough penalty parameter is not chosen, the method may produce sub-optimal converge rates. Analysis of the penalty method for the fitted method is carried out in [Bab73b, BE86], but in our knowledge, the analysis for the penalty method in the unfitted finite element framework is not available.

2.4.2 The Method of Lagrange Multipliers

The method of Lagrange multipliers was also introduced by Babuška in the finite element framework to impose the Dirichlet boundary conditions [Bab73a]. It is also extensively used in the field of constrained optimization problems to enforce equality constraints. This method results in the mixed formulation which requires the solution of the primal variable and an additional multiplier.

We define a Lagrangian function $\mathcal{L}(\cdot,\cdot):\mathcal{V}_h\times\mathcal{M}_h\to\mathbb{R}$, where $\mathcal{M}_h$ is a multiplier space, $\mathcal{M}_h\subseteq H^{-\frac{1}{2}}(\Gamma)$. The Lagrangian function for enforcing the Dirichlet boundary condition is defined as

$$\mathcal{L}(u_h,\lambda_h)=J(u_h)+\langle\lambda_h,u_h-g_D\rangle_\Gamma. \tag{2.15}$$

The saddle-point $(u_h,\lambda_h)\in\mathcal{V}_h\times\mathcal{M}_h$ is the solution of the above problem given as

$$\mathcal{L}(u_h,\mu_h)\leqslant\mathcal{L}(u_h,\lambda_h)\leqslant\mathcal{L}(v_h,\lambda_h)\quad\forall(v_h,\mu_h)\in\mathcal{V}_h\times\mathcal{M}_h.$$

The first order optimality conditions of the Lagrangian formulation (2.15) can be reformulated into the following equivalent formulation.
Find $(u_h,\lambda_h)\in\mathcal{V}_h\times\mathcal{M}_h$ such that

$$\begin{aligned} a(u_h,v_h)+g(u_h,v_h)+b(\lambda_h,v_h) &= F(v_h) && \forall v_h\in\mathcal{V}_h,\\ b(\mu_h,u_h) &= G_D(\mu_h) && \forall\mu_h\in\mathcal{M}_h. \end{aligned} \tag{2.16}$$

Here, the bilinear form $b(\cdot,\cdot):\mathcal{M}_h\times\mathcal{V}_h\to\mathbb{R}$ and the linear form $G_D:\mathcal{M}_h\to\mathbb{R}$ are defined as

$$b(\lambda_h,u_h):=\sum_{K\in\mathcal{T}_{h,\Gamma}}\langle\lambda_h,u_h\rangle_{\Gamma_K}\quad\text{and}\quad G_D(\lambda_h):=\sum_{K\in\mathcal{T}_{h,\Gamma}}\langle\lambda_h,g_D\rangle_{\Gamma_K}.$$

The method of Lagrange multipliers is an attractive option for enforcing the Dirichlet boundary conditions. However, it is stable only if the following discrete inf-sup condition is satisfied

$$\inf_{\lambda_h \in \mathcal{M}_h} \sup_{u_h \in \mathcal{V}_h} \frac{b(\lambda_h, u_h)}{\|\lambda_h\|_{H^{-\frac{1}{2}}(\Gamma),h} \|u_h\|_{H^1(\Omega)}} \geqslant \beta > 0, \tag{2.17}$$

where the constant β does not depend on the mesh size h. The choice of a finite element space for the Lagrange multiplier is very essential to have a stable discretization method. In the XFEM framework, most naive options for the primal and the multiplier spaces do not satisfy the inf-sup sup condition. In the original work of the fictitious domain methods, the Lagrange multiplier method was used for imposing the Dirichlet boundary conditions [GPP94, GG95]. In that case, the primal variable and the Lagrange multiplier are discretized on two different meshes. The Lagrange multiplier is discretized on a coarser mesh such that the mesh size of the coarse mesh is at least three times larger than the mesh chosen to discretize the primal variable [GPP94].

The inf-sup condition (2.17) can be circumvented by introducing additional stabilization terms. In the work of Barbosa and Hughes [BH91, BH92a], the inf-sup condition is avoided by adding a least-square penalty term that minimizes the difference between the multiplier and its physical interpretation. Their method is also introduced to the XFEM framework and analyzed in detail for the fictitious domain problem in [HR09].

We discuss stable discretization spaces for the Lagrange multiplier and ways to circumvent the inf-sup condition in more detail in Section 2.4.2.

2.4.3 Nitsche's Method

Nitsche's method was introduced as an alternative to the penalty method and the method of Lagrange multipliers to enforce the Dirichlet boundary conditions [Nit71]. This method is utilized in many different discretization methods as it is simple to use. For example, in the discontinuous Galerkin (DG) method, Nitsche's method is used to enforce the continuity between each element faces [Arn82]. It is also used in a domain decomposition methods to mortar the interfaces between non-matching meshes [BFMR97, BHS03]. Nitsche's method is also a popular choice for enforcing the Dirichlet boundary conditions for mesh-free methods and particle methods [GS03].

This method can be regarded as a variationally consistent penalty method. An alternative interpretation of the method is as a stabilized Lagrange multiplier method, where the Lagrange multiplier is explicitly expressed by its physical interpretation as an outward flux in the primal variable [Ste95].

We start with an augmented Lagrangian functional $\mathcal{L}_A(\cdot,\cdot) : \mathcal{V}_h \times \mathcal{M}_h \to \mathbb{R}$, where the formulation is given as

$$\mathcal{L}_A(u_h,\lambda_h) = J(u_h) + \langle \lambda_h, u_h - g_D\rangle_\Gamma + \frac{\gamma_p}{2}\|u_h - g_D\|^2_{H^{\frac{1}{2}}(\Gamma),h}. \tag{2.18}$$

If we solve this problem, the saddle-point point $(u_h,\lambda_h) \in \mathcal{V}_h \times \mathcal{M}_h$ is the solution of the above problem given as

$$\mathcal{L}_A(u_h,\mu_h) \leqslant \mathcal{L}_A(u_h,\lambda_h) \leqslant \mathcal{L}_A(v_h,\lambda_h) \qquad \forall (v_h,\mu_h) \in \mathcal{V}_h \times \mathcal{M}_h.$$

But rather than solving for the saddle-point problem, we replace the Lagrange multiplier by its physical interpretation. In the context of problem (2.1) the Lagrange multiplier can be interpreted as outward flux at the boundary, given as

$$\lambda_h = -\alpha\nabla_{\boldsymbol{n}} u_h.$$

Using this information, we reformulate the augmented Lagrangian formulation (2.18) from mixed formulation to primal formulation. We define energy functional, $J_N(\cdot) : \mathcal{V}_h \to \mathbb{R}$ as $J_N(u_h) := \mathcal{L}_A(u_h, -\alpha\nabla_{\boldsymbol{n}} u_h)$. The modified energy functional is given as

$$\min_{u_h\in\mathcal{V}_h} J_N(u_h) = J(u_h) - \langle \alpha\nabla_{\boldsymbol{n}} u_h, u_h - g_D\rangle_\Gamma + \frac{\gamma_p}{2}\|u_h - g_D\|^2_{H^{\frac{1}{2}}(\Gamma),h}. \tag{2.19}$$

This formulation can be referred to as the energy formulation of Nitsche's method.

The first order optimality condition corresponding to the minimization problem (2.19) yields the abstract variational problem.

$$\text{Find } u_h \in \mathcal{V}_h \text{ such that} \qquad A_N(u_h,v_h) = F_N(v_h) \qquad \forall v_h \in \mathcal{V}_h, \tag{2.20}$$

where the bilinear from $A_N(\cdot,\cdot)$ and the linear form $F_N(\cdot)$ are defined as

$$\begin{aligned} A_N(u_h,v_h) = {} & a(u_h,v_h) + g(u_h,v_h) - \langle \alpha\nabla_{\boldsymbol{n}} u_h, v_h\rangle_\Gamma - \langle \alpha\nabla_{\boldsymbol{n}} v_h, u_h\rangle_\Gamma \\ & + (\gamma_p u_h, v_h)_{H^{\frac{1}{2}}(\Gamma),h}, \\ F_N(v_h) = {} & F(v_h) - \langle \alpha\nabla_{\boldsymbol{n}} v_h, g_D\rangle_\Gamma + (\gamma_p g_D, v_h)_{H^{\frac{1}{2}}(\Gamma),h}. \end{aligned} \tag{2.21}$$

The connection between the Lagrange multiplier method of Barbosa-Hughes and Nitsche's method is established in [Ste95]. Nitsche's method can be regarded as a penalty method with extra consistency terms with the normal derivatives across the boundary.

Nitsche's method was introduced in the unfitted finite element method in context of the interface problem to enforce the interface conditions [HH02]. Later, this work was extended to more generic problems and applied to the fictitious domain problems [BH12]. Under reasonable mesh assumptions on the background mesh, a priori error estimates are given by [BH12],

$$\begin{aligned} |||u-u_h|||_h &\leqslant C\ h\|u\|_{H^2(\Omega)} \quad \forall u \in \mathcal{V}_h, \\ \|u-u_h\|_{L^2(\Omega)} &\leqslant C\ h^2\|u\|_{H^2(\Omega)} \quad \forall u \in \mathcal{V}_h, \end{aligned}$$

where the constant C is completely independent of the location of the interface in the mesh. The mesh-dependent energy norms $|||\cdot|||_h$ in the above estimates are defined as

$$|||v|||_h^2 := \|\nabla v\|_{L^2(\Omega)}^2 + \gamma_p\|v\|^2_{H^{\frac{1}{2}}(\Gamma),h} + \|\nabla_{\boldsymbol{n}} v\|^2_{H^{-\frac{1}{2}}(\Gamma),h}.$$

2.5 Stabilization Parameter in Nitsche's Method

As mentioned in the previous section, Nitsche's formulation is stable only if the bilinear form $A_N(\cdot,\cdot)$ is coercive, for example by choosing a sufficiently large stabilization parameter γ_p. Estimation of such a stabilization parameter is a very delicate task. If the value of the stabilization parameter is too large, it gives rise to an ill-conditioned system matrix. It becomes increasingly difficult to estimate the stabilization parameter for irregular meshes and higher-order finite elements.

In this section, we explore different methods for estimating the stabilization parameter. Following the coercivity of the bilinear form (2.21), we have

$$\begin{aligned} A_N(v_h,v_h) \geqslant & \sum_{K\in\mathcal{T}_h\setminus\mathcal{T}_{h,\Gamma}} \|\alpha^{\frac{1}{2}}\nabla v_h\|^2_{L^2(K)} + \frac{1}{\epsilon}\|\alpha^{\frac{1}{2}}\nabla_{\boldsymbol{n}} v_h\|^2_{H^{-\frac{1}{2}}(\Gamma),h} + g(v_h,v_h) \\ & + \sum_{K\in\mathcal{T}_{h,\Gamma}} \Big(1-\frac{2C_\gamma}{\epsilon}\Big)\|\alpha^{\frac{1}{2}}\nabla v_h\|^2_{L^2(K_\Omega)} + (\gamma_p-\epsilon)\|v_h\|^2_{H^{\frac{1}{2}}(\Gamma),h}. \end{aligned}$$

This inequality utilizes Young's inequality for some $\epsilon > 0$ and follows trace inequality (2.9) (see A.1). The bilinear form is coercive if the positivity of two terms in the last line is ensured, given by $\epsilon \geqslant 2C_\gamma$, and $\gamma_p \geqslant \epsilon$. Thus, the stabilization parameter can be given with the bound $\gamma_p \geqslant 2C_\gamma$.

The stabilization parameter thus influences two different aspects. Firstly, if the stabilization parameter has to be chosen sufficiently large such that the stability of the method is insured. Secondly, we know that the large stabilization parameter enforces the Dirichlet boundary conditions more accurately. Hence, a large stabilization parameter reduces the discretization error at the boundary, but simultaneously it increases the condition number of the linear system. Thus, it is necessary to strike a balance between the discretization error and the conditioning of the system. Therefore, we aim to choose the value of the stabilization parameter larger than the value required to ensure stability but not too large that the system matrix becomes poorly conditioned.

In the next sections, we discuss two approaches: the first one for estimating the stabilization parameter and the second one for circumventing the need to compute the stabilization parameter.

2.5.1 Eigenvalue Problem

Here, we discuss the idea of estimating the stabilization parameter by solving a generalized eigenvalue problem. This method was used for a particle-partition of unity method and later explored more for spline-based finite elements applied to harmonic and bi-harmonic problems [EDH10, GS03]. This approach is also widely used in the finite cell methods [SR15, RSB$^+$13, RSOR14, JADH15], where Nitsche's method is used to enforce Dirichlet boundary condition.

The coercivity of the bilinear form relies on trace inequality (2.9). A good estimate of C_γ can be achieved by solving a generalized eigenvalue problem, as C_γ is bounded from below by the largest eigenvalue of the auxiliary problem (2.22).

We pose eigenvalue problems for each $K \in \mathcal{T}_{h,\Gamma}$, and solve series of locally given element-wise problems, find $\max(\lambda_K) \in \mathbb{R}$ such that

$$(\alpha \nabla_{\boldsymbol{n}} v_h, \alpha \nabla_{\boldsymbol{n}} v_h)_{H^{-\frac{1}{2}}(\Gamma_K),h} = \lambda_K (\alpha \nabla v_h, \nabla v_h)_{L^2(K_\Omega)} \quad \forall v_h \in \mathcal{V}_h|_{K_\Omega}, \tag{2.22}$$

where $\mathcal{V}_h|_{K_\Omega}$ is restriction of $\mathcal{V}_h$ on a given element K and λ_K denotes the set of eigenvalues. In order to solve the generalized eigenvalue problem (2.22), it is necessary that the $(\alpha \nabla v_h, \nabla v_h)_{L^2(K_\Omega)}$ has only a trivial kernel. This can be achieved if the function space $\mathcal{V}_h|_K$ is defined in the space of polynomials which are orthogonal to constants. From the construction, it is clear that $(\alpha \nabla v_h, \nabla v_h)_{L^2(K_\Omega)}$ is a representation of a local stiffness matrix. The kernel of the local stiffness matrix is known to be a constant vector. Algebraically, we can use a deflation method to eliminate the influence of the trivial kernel from matrix representation of both terms in (2.22) and still retain the spectral properties of matrices. Thus, solving the generalized eigenvalue problem of the deflated system is equivalent to solving the original eigenvalue problem (2.22). This allows us to use the largest eigenvalue to estimate the stabilization parameter. To ensure the boundedness of (2.9), we take the value of the element-wise stabilization parameter 4 times larger than the largest eigenvalue. Hence, the stabilization parameter is computed element-wise as, $\gamma_p = 4\max(\lambda_K)$ to satisfy the condition $\gamma_p \geqslant 2C_\gamma$.

Above, we have defined local eigenvalue problems for each element $K \in \mathcal{T}_{h,\Gamma}$. An alternate option is to create a global system for all cut elements and solve a global eigenvalue problem [GS03]. In this case, the stabilization parameter is estimated by the largest eigenvalue of the global system. For irregular meshes and complex domains, an interface can intersect the mesh arbitrarily and a very small cut in one element can influence the largest eigenvalue in the global setting. Hence, we choose to solve a series of local eigenvalue problems and estimate the local stabilization parameter for each cut element [DH09]. This approach is more beneficial, as we do not have to solve a global generalized eigenvalue problem and the effect of small cut elements is localized.

2.5.2 Lifting Operator

In order to avoid the computation of the stabilization parameter in Nitsche's method, an alternative method is proposed in [Leh16, Leh15]. In this method, the stabilization parameter is chosen locally in an implicit manner, similarly to the previous section. This strategy is common in the DG method [BR97]. The stability and error analysis of the DG discretization equipped with the lifting operators is carried out in [BMM$^+$00].

First, we introduce an element-wise lifting operator $\mathcal{R}_K(\cdot) : \mathcal{V}_h|_K \to \mathcal{W}_h$ that lifts the functions defined on the cut elements into the space of polynomials which are orthogonal to constants. The space $\mathcal{W}_h$ is given as

$$\mathcal{W}_h := \{u_h \in L^2(K) : u_h|_K \in \mathcal{P}_1 \cap (\mathcal{P}_0)^\perp \ \forall K \in \mathcal{T}_{h,\Gamma}\}.$$

On the uncut elements, i.e., $K \in \widetilde{\mathcal{T}}_h \setminus \mathcal{T}_{h,\Gamma}$ the lifting operator is defined as $\mathcal{R}_K(u_h) = 0$. While on the cut elements, the lifting operator is defined as, find $w_h := \mathcal{R}_K(u_h) \in \mathcal{W}_h$ such that

$$(\alpha \nabla w_h, \nabla v_h)_{L^2(K_\Omega)} = -\langle \alpha \nabla_{\boldsymbol{n}} v_h, u_h \rangle_\Gamma \quad \forall u_h, v_h \in \mathcal{V}_h|_K.$$

The coercivity of the bilinear form of the original formulation (2.21) can be ensured if an additional term stemming from the lifting operators is added

$$a(u_h, u_h) - 2\langle \alpha \nabla_{\boldsymbol{n}} u_h, u_h \rangle_\Gamma + 2 \sum_{K \in \mathcal{T}_{h,\Gamma}} a(\mathcal{R}_K(u_h), \mathcal{R}_K(u_h)) \geqslant \frac{1}{2} a(u_h, v_h).$$

Addition of such a term in the bilinear form ensures the coercivity for any positive stabilization parameter. Thus for simplicity, we choose the value of the stabilization parameter as $\gamma_p = 1$. Now, the updated bilinear formulation is given as

$$\begin{aligned} A_N^{\mathcal{R}}(u_h, v_h) = \quad & a(u_h, v_h) + g(u_h, v_h) - \langle \alpha \nabla_{\boldsymbol{n}} u_h, v_h \rangle_\Gamma - \langle \alpha \nabla_{\boldsymbol{n}} v_h, u_h \rangle_\Gamma \\ & + (u_h, v_h)_{H^{\frac{1}{2}}(\Gamma),h} + 2 \sum_{K \in \mathcal{T}_{h,\Gamma}} a(\mathcal{R}_K(u_h), \mathcal{R}_K(v_h)), \end{aligned} \tag{2.23}$$

which can be replaced with the original bilinear form $A_N(\cdot,\cdot)$ in (2.20).

2.6 Discretization of Lagrange Multiplier Space

In the finite element framework, the stability of the mixed formulation is ensured if the discrete inf-sup condition (2.17) is satisfied [Bre74]. The detailed analysis of the method of Lagrange multipliers for enforcing the Dirichlet boundary condition is carried out by Pitkäranta in the context of fitted FE framework [Pit79, Pit80]. In order to achieve optimal convergence rates of the discretization method, the choice of the FE spaces for primal variable $u_h \in \mathcal{V}_h$ and the dual variable $\lambda_h \in \mathcal{M}_h$ is crucial. Also, the most convenient options for $\mathcal{V}_h$ and $\mathcal{M}_h$ are very rarely stable. In the unfitted finite element framework, it is shown that the most convenient approach to construct the multiplier spaces gives rise to instabilities [DMB01, BPM$^+$03, JD04].

To circumvent the strict requirement of satisfying the inf-sup condition, a different approach was introduced by Barbosa and Hughes [BH91]. In this approach, the restriction over the choices for FE spaces is dropped and the stability of the formulation is ensured using a stabilization term [BH92b]. In the Barbosa-Hughes approach, the stabilization term penalizes the jump between the multiplier and its physical interpretation. The Barbosa-Hughes approach is extended by Haslinger and Renard to fictitious domain method in the XFEM framework [HR09]. In the unfitted FEM framework, a different type of a stabilization method was introduced by Burman and Hansbo [BH10]. In that work, the multiplier is chosen as piecewise constant function and the stability of the saddle-point formulation is achieved by penalizing the jump of the multiplier over the element faces [BH10].

In general, the inf-sup condition is satisfied when the FE spaces for the primal variable and the multiplier are compatible. The multiplier is used to enforce the constraints on the primal variable. The compatibility between FE spaces suggests that the primal variable is neither over-constrained nor under-constrained. If the multiplier space is chosen to be too large, it causes the primal variable to be over-constrained and it gives rise to the phenomenon called boundary locking. In this case, either a richer primal space has to be chosen or the multiplier space has to be coarsened. In the work of Babuška [Bab73a, BOL78], the error estimates were established under the condition that the ratio between the boundary mesh size and the mesh size in the domain is greater than some constant depending on the domain. When the method of Lagrange multipliers was employed in the original fictitious domain method to enforce the boundary condition, the stability of the method was achieved by coarsening the multiplier space [GPP94, GG95]. Another option to satisfy the inf-sup condition is to choose a richer primal space while taking the most-convenient multiplier space. In the context of unfitted FEM, bubble-stabilized primal spaces were considered such that they satisfy inf-sup condition [MDH07, DF08]. In the earlier work, it is shown that the bubble-stabilized method and Nitsche's method bear a strong resemblance [BFMR97, MDH07].

Another approach where the primal space was kept the same and a coarser multiplier space was considered, and it can be found in several works [MBT06, KDL06, BMW09]. Béchet et al. developed a stable Lagrange multiplier space based on a vital vertex algorithm [BMW09], which was later extended by Hautefeuille et al. [HAD12]. This method does not require any stabilization terms and also the primal space $\mathcal{V}_h$ is not modified. Only the multiplier space $\mathcal{M}_h$ is designed carefully such that it satisfies the inf-sup condition and ensures optimal convergence of discretization error. In the next section, we discuss the algorithm for creating a stable Lagrange multiplier space.

2.6.1 Vital Vertex Algorithm

In order to create a finite element space, we have to create a mesh. There are multiple ways to create a mesh on the embedded boundary on the background mesh. The simplest way to create a mesh would be to extract the submesh of all elements which are intersected by the boundary. But the Lagrange multiplier discretized on this mesh does not satisfy the inf-sup condition.

In Figure 2.3a, all edges that are intersected by the boundary are marked with a thick line, and the intersection points that lie on those edges are marked with green circles. From this point onwards, we refer to the intersection points as vertices. One of the most straightforward ways for creating a mesh is to create a trace mesh with all vertices. But a trace mesh that can be used to discretize the embedded boundary has too many vertices. If this mesh is used to discretize the multiplier space, it clearly violates the inf-sup condition. Another, less common, approach is to create a new trace mesh on the boundary and using that mesh for the discretization of the multiplier space [KDL06]. We follow the strategy of Béchet et al. and Hautefeuille et al. to create a stable multiplier space [BMW09, HAD12]. This approach is robust for the curved boundaries and it can easily be extended to higher dimensions. In this approach, the trace mesh, created by the vertices, is methodically coarsened. The coarse mesh is obtained by algorithmically selecting a subset of vital vertices. In this section, we describe the vital vertex algorithm.

The vital vertex algorithm utilizes elements of graph theory to get a subset of vital vertices. As a first step, a vertex graph is created, where the vertices are set as the nodes of this graph. In the vertex graph, two vertices are connected if the edges on which the vertices lie have a common endpoint. Once the vertex graph is created, we declare a vertex as vital or non-vital based on the following two rules:

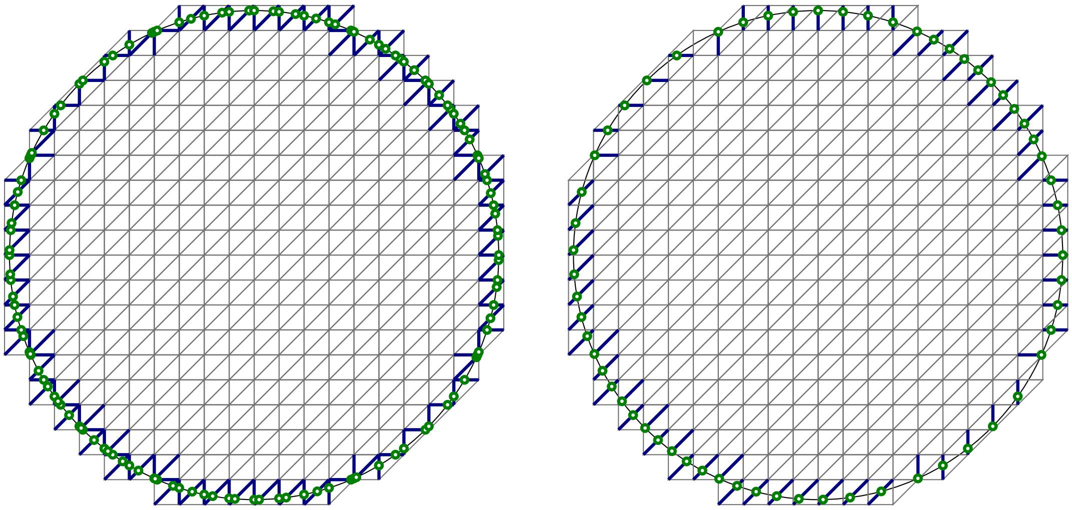
(a) A vital vertex is never connected to another vital vertex.

(b) A non-vital vertex is always connected to a vital vertex.

The first rule provides an upper bound on the number of vital vertices, and thus it allows for coarsening. The second rule prevents aggressive coarsening and can be interpreted as a lower bound on the number of non-vital vertices. These two rules are complementary and restrict the number of multipliers from becoming too few or too many. The algorithm for selecting vital vertices described below satisfies both of these rules.

The algorithm to identify the vital vertices is given as follows:

1. Get a list of all elements that are intersected by the boundary and generate a submesh $\mathcal{T}_{h,\Gamma}$.
2. Traverse the mesh $\mathcal{T}_{h,\Gamma}$ and create a list of all vertices indexed by the corresponding cut edges.
3. Generate a map from the vertices to the endpoints of the corresponding cut edges and an inverse map from the endpoints of the edges to the vertices.
4. Create a graph of the vertices, where the vertices represent the nodes of the graph and the edges of the graph are represented by a common endpoint.
5. Sort the nodes in this graph by the number of edges.
6. Lexicographically sort all nodes with an equal number of edges.
7. Create two empty sets *vital* and *non-vital*.
8. Go to the first vertex in the sorted list and add it to the *vital* set.



(a) All vertices are marked with a green circle, all cut edges which are intersected by the boundary are marked with blue line.

(b) Vital vertices are marked with a green circle, the cut edges on which the vital vertices lie are marked with blue line.

Figure 2.3. Nodes associated with a naive Lagrange multiplier space and the multiplier space due to the vital vertex method.

9. Move to the next vertex and if the current vertex is connected to any vertex from the *vital* set, add the current vertex to the *non-vital* set; otherwise add the current vertex to the *vital* set.

10. Go to 9 and repeat until the sorted vertex graph is traversed.

The algorithm is designed in such a way that the vertices which have the least connections are added to the *vital* list first. Hence, the vertices with fewer connections have a higher likelihood to be in the *vital* list. The step 5 and 6 in the algorithm ensure that the set of the vital vertices has the highest possible cardinality. In general, there is a deterministic list of vital vertices, but there are multiple acceptable sets of vital vertices. Due to the sorting procedure, under a small perturbation of the boundary, the list of vital vertices remains unchanged. In Figure 2.3a, a list of all vertices that are intersected by the boundary and the corresponding cut edges are shown. While in Figure 2.3b, we can see a list of vital vertices and the corresponding cut edges.

We denote the list of vital vertices on the boundary by $\mathscr{V}_{h,\Gamma}$, and the dimension of the multiplier space $\mathcal{M}_h$ is given as $|\mathscr{V}_{h,\Gamma}|$. As, we have now defined a mesh on the embedded boundary, in the next section we discuss a definition of the basis functions on the vital vertices.

2.6.2 A Stable Lagrange Multiplier Space

In this section, we describe the last remaining detail about the construction of a stable Lagrange multiplier space. We define a multiplier on each vital vertex, where we have to define a new basis function for the corresponding multiplier. In two-dimensions, a simple choice of the basis function is to use standard hat function, defined on the segments connecting two vital vertices. While in three-dimension, one can construct a mesh of boundary using the vital vertices under some regularity assumptions (e.g., using a Delaunay approach). If the basis functions for the multiplier are computed using this strategy, the inf-sup condition per element can not be satisfied in pathological cases [BMW09]. Also, these strategies are not very simple from the implementation perspective. We can simplify this aspect if the definition of the basis functions for the Lagrange multiplier is defined on the background mesh.

Hence, rather than creating a new set of basis functions for the multiplier space, the basis functions are defined as a trace of the basis function of the background mesh. The basis functions on the vital vertices are computed as a linear combination of the basis functions defined on the nodes of the background mesh. This basis functions on the vital vertices that should also satisfy the partition of unity condition and should have local support on the boundary.

For each vital vertex $p \in \mathscr{V}_{h,\Gamma}$, we define the associated basis function μ_p as a linear combination of the basis functions of nodal basis ϕ_q, $\forall q \in \mathscr{N}_h$, restricted to the boundary Γ. We introduce a set of nodes $\mathscr{N}_{h,\Gamma}$ given as

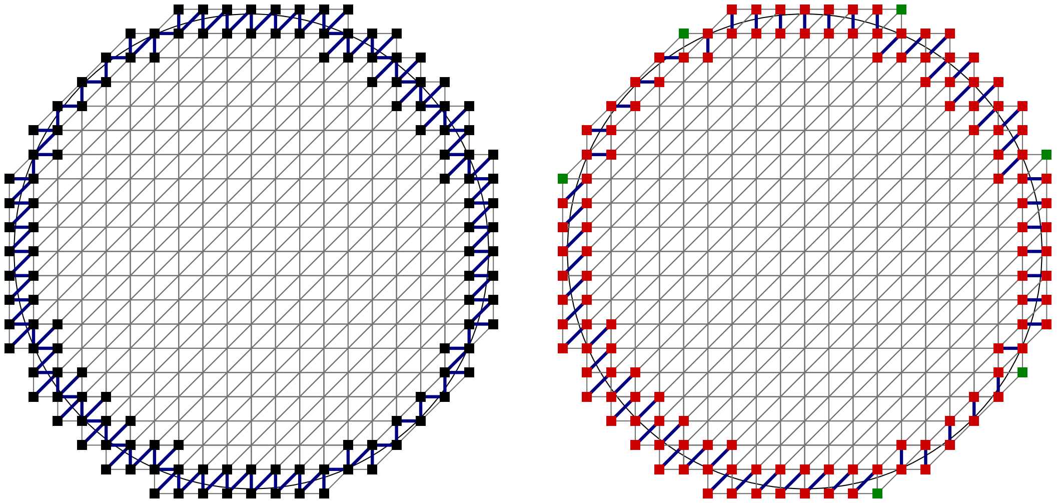
$$\mathscr{N}_{h,\Gamma} := \{q \in \mathscr{N}_h : \phi_q|_\Gamma \neq 0\},$$

where $\mathscr{N}_{h,\Gamma}$ includes all nodes that are endpoints of the cut-edges. In Figure 2.4a these nodes are marked by black squares. Now, we define the nodal basis function for each vital vertex $p \in \mathscr{V}_{h,\Gamma}$ as

$$\mu_p := \sum_{q \in \mathscr{N}_{h,\Gamma}} w_{pq} \phi_q|_\Gamma \qquad \forall q \in \mathscr{N}_{h,\Gamma},$$

where w_{pq} are coefficients of the linear combination.

For each vital vertex $p \in \mathscr{V}_{h,\Gamma}$, we define a set $\mathscr{P}_p \subset \mathscr{N}_{h,\Gamma}$. The set $\mathscr{P}_p$ contains all nodes that are endpoints of the edge on which the vital vertex p is located. We define a set of active nodes $\mathscr{N}^A_{h,\Gamma} := \cup_{p \in \mathscr{V}_{h,\Gamma}} \mathscr{P}_p$ and a set of inactive nodes $\mathscr{N}^I_{h,\Gamma} := \mathscr{N}_{h,\Gamma} \setminus \mathscr{N}^A_{h,\Gamma}$. The inactive nodes are set of nodes that are not connected to any vital vertex, and it is possible that $\mathscr{N}^I_{h,\Gamma}$ is an emptyset. Also, all inactive nodes, $q \in \mathscr{N}^I_{h,\Gamma}$, are always connected to a non-vital vertex. Due to the rule (b), we know each non-vital vertex is connected to a vital vertex, hence for $q \in \mathscr{N}^I_{h,\Gamma}$



(a) The nodes $\mathcal{N}_{h,\Gamma}$ that belong to the sub-mesh $\mathcal{T}_{h,\Gamma}$ are marked with black squares

(b) All active nodes $\mathcal{N}^{A}_{h,\Gamma}$ are marked with red squares, and the inactive nodes $\mathcal{N}^{I}_{h,\Gamma}$ are marked with green squares

Figure 2.4. Different type of nodes characterized by the vital-vertex algorithm.

there exists another endpoint of the cut edge which is an active node. All $\mathcal{P}_p$ are pairwise disjoint, thus there exists a unique $p \in \mathcal{V}_{h,\Gamma}$ such that one endpoint of this edge is in $\mathcal{P}_p$ and the other endpoint is placed in $\mathcal{Q}_p$, where $\mathcal{Q}_p$ denotes set of inactive nodes associated with vertex p. We denote the number of nodes in $\mathcal{Q}_p$ as n_q. Also, the set of global inactive nodes can be given as $\mathcal{N}^{I}_{h,\Gamma} = \cup_{p \in \mathcal{V}_{h,\Gamma}} \mathcal{Q}_p$.

Now, we can define the values of the coefficients in terms of these subsets of nodes. The values of the coefficients w_{pq}, $p \in \mathcal{V}_{h,\Gamma}$ and $q \in \mathcal{N}_{h,\Gamma}$ are given as

$$w_{pq} := \begin{cases} 1 & q \in \mathcal{P}_p, \\ \dfrac{1}{n_q} & q \in \mathcal{Q}_p, \\ 0 & \text{otherwise.} \end{cases}$$

Now, the basis function μ_p can be written as

$$\mu_p = \sum_{q \in \mathcal{P}_p} \phi_q + \sum_{q \in \mathcal{Q}_p} \frac{1}{n_q} \phi_q .$$

The basis function defined using this method has local support on the boundary and it also has a positive partition of unity property.

The multiplier space $\mathcal{M}_h$ discretized using the vital vertices has the basis functions which are carefully constructed as a weighted sum of traces of the basis functions of background mesh. This multiplier space $\mathcal{M}_h$ satisfies a uniform inf-sup condition (2.17). In our knowledge the detailed error analysis of the method is not available, but based on the abstract saddle point theory [BBF13] the discretization error could be given as

$$\begin{aligned} |||u-u_h|||_h &\leqslant Ch\|u\|_{H^2(\Omega)} \quad \forall u_h \in \mathcal{V}_h, \\ \|\lambda-\lambda_h\|_{H^{-\frac{1}{2}}(\Gamma)} &\leqslant Ch\|\lambda\|_{H^{\frac{1}{2}}(\Gamma)} \quad \forall \lambda_h \in \mathcal{M}_h. \end{aligned}$$

The mesh-dependent energy norm $|||\cdot|||_h$ in the above estimates is defined as

$$|||v|||_h^2 := \|\nabla v\|_{L^2(\Omega)}^2 + \|v\|_{H^{\frac{1}{2}}(\Gamma),h}^2.$$

2.7 Numerical Results

In this section, we carry out some numerical experiments to study the convergence behavior of all methods discussed until now. We consider two different examples for numerical experiments. We compare the condition numbers of the arising linear system of equations and the discretization errors in different norms.

2.7.1 Problem Description

All experiments in this chapter are carried out on a structured grid with the quadrilateral elements, defined on $[0,1]^2$. We start with a background mesh that has 100 elements in each direction, denoted as mesh on level $L1$, and uniformly refine the mesh to obtain finer meshes as shown in Table 2.1. We also use the same mesh hierarchy in the next chapters for the multilevel solution strategies. We note that in Table 2.1, we have only shown the active nodes or the active degrees of freedom (DOFs) and the number of active elements associated with the domain.

Example 1-FD For this example, we consider a diffusion problem on a domain of superellipse shape, where the superellipse is defined by a level set function. The boundary of the domain is defined as a zero level set of a function

$$\Lambda_e(\boldsymbol{x}) := r_1^8 - \left|\frac{x-c_x}{a}\right|^8 - \left|\frac{y-c_y}{b}\right|^8.$$

Here, r_1 denotes the radius of the superellipse, given as $r_1 = 0.4901$. The symbols a and b denote the major and minor axis of the superellipse, chosen as $a = 1, b = 0.8$. The components of the position are given as $\boldsymbol{x} = (x, y)$. Lastly,

levels	$h_{\max}$	Superellipse Domain		Circular Domain	
		# DOFs	# Elems	# DOFs	# Elems
$L1$	1.41421E-02	7,691	7,514	5,729	5,560
$L2$	7.07107E-03	30,036	29,685	22,209	21,876
$L3$	3.53553E-03	118,904	118,204	87,585	86,920
$L4$	1.76777E-03	473,123	471,725	347,629	346,300
$L5$	8.83883E-04	1,887,036	1,884,243	1,385,209	1,382,556

Table 2.1. The multilevel hierarchy of meshes with quadrilateral elements, with total number of active DOFs and number of active elements on a given level.

by c_x and c_y, we denote the center of the superellipse, given as $(0.5, 0.5)$. The domain Ω is defined as the region where the level set of the function Λ_e is positive, $\Lambda_e(\boldsymbol{x}) > 0$. For simplicity, we choose the right-hand side f_1 and the Dirichlet boundary condition in such a way that the exact solution is given as defined by the level set function, $u_1 := \Lambda_e$.

Example 2-FD For this example, we consider again a diffusion problem on a circular domain. The boundary of the domain Γ_c is defined as a zero level set of a function $\Lambda_c(\boldsymbol{x}) := r_2^2 - \|\boldsymbol{x} - \boldsymbol{c}\|_2^2$ with radius, $r_2^2 = 3 - 2^{1/2}$, and $\boldsymbol{c}$ is the center of the circle $(0.5, 0.5)$. The domain Ω is defined by the region where the value of level set function is positive, $\Lambda_c(\boldsymbol{x}) > 0$. The right-hand side f_2 and the Dirichlet boundary condition are chosen in such a way that the exact solution, $u_2 = (\exp(-500s)-1)(\exp(-500t)-1)(\exp(-500yy)-1)(1-3rr)^2$ is satisfied. Here, $s := (x - 1/3)^2$, $t := (x - 2/3)^2$, $xx := (x - 1/2)^2$, $yy := (y - 1/2)^2$ and $rr := xx + yy$ [Woh00a].

In both examples, we have deliberately chosen the radius of the circle and the superellipse in such a way that the interface would not coincide with the element edges or the nodes. We compare these two examples against the different methods, namely the penalty method, Nitsche's method, and the method of Lagrange multipliers. For the penalty method, the value of the stabilization parameter or the penalty parameter is chosen as $\gamma_p = 100$. We have discussed several methods to compute the stabilization parameter in Nitsche's formulation, but for the numerical experiments, we used the generalized eigenvalue problem to compute the stabilization term, as described in Section 2.5.1. The Lagrange multiplier approach uses the vital vertex algorithm to create a stable multiplier space. All these methods include the ghost penalty term, with the parameter $\epsilon_G = 0.1$.

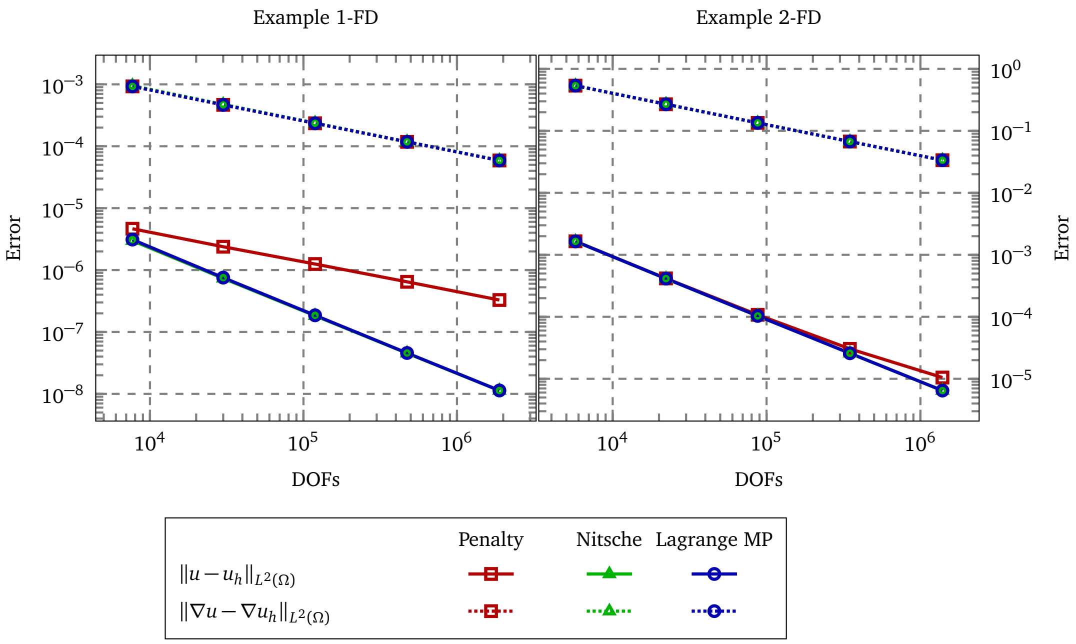


Figure 2.5. Discretization error in L^2-norm and H^1-seminorm for the different methods applied to Example 1-FD and Example 2-FD.

2.7.2 Study of Discretization Errors

In this section, we compare the convergence rate of the discretization error in L^2-norm, H^1-seminorm on the domain. As we are focusing on the methods to enforce the boundary conditions using the different strategies, we also compare the discretization error on the boundary. Thus, we also compare the discretization error of the function on the boundary in $H^{\frac{1}{2}}(\Gamma), h$-norm and the outward flux on the boundary in $H^{-\frac{1}{2}}(\Gamma), h$-norm.

Domain Error: Figure 2.5 depicts the convergence of the discretization error in the domain. It is clear from Figure 2.5 that the error in the H^1-seminorm for both examples reduces with the number of DOFs. As the error estimates suggest, the rate of convergence of the discretization error in H^1-seminorm is of order h for all methods. While for the discretization error in L^2-norm, the convergence rate of Nitsche's method and the method of Lagrange multipliers is of order h^2. Although, this does not hold for the penalty method. The penalty method is known to have suboptimal convergence rates if sufficiently large penalty parameter is not chosen. For the Example 1-FD, the convergence rate of the discretization error in L^2-norm is of order h, while for the Example 2-FD the convergence rate is

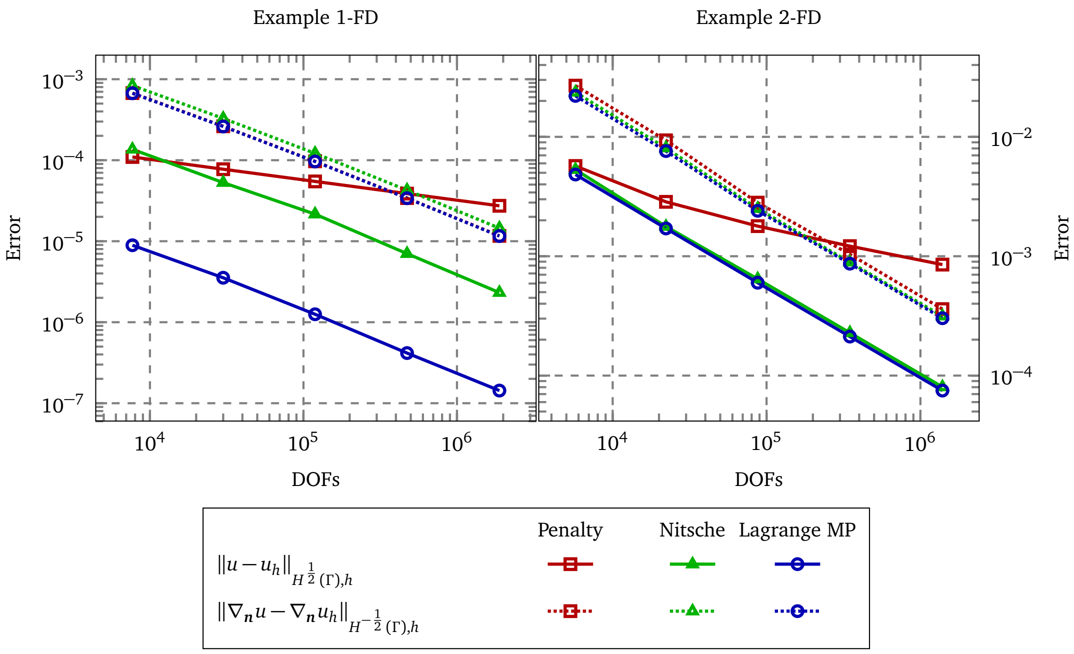


Figure 2.6. Discretization error on the boundary, the error of the function in $H^{\frac{1}{2}}(\Gamma), h$-norm and flux in $H^{-\frac{1}{2}}(\Gamma), h$-norm for different methods applied to Example 1-FD and Example 2-FD.

optimal for larger mesh sizes but we observe the impediment in the convergence rate with decreasing mesh size.

Boundary Error: Figure 2.6 demonstrated that the convergence of the discretization error on the boundary for different mesh sizes. From the error estimates, it is known that the convergence rate of the discretization error of the function in $H^{\frac{1}{2}}(\Gamma), h$-norm and the outward flux in $H^{-\frac{1}{2}}(\Gamma), h$-norm is of order $h^{\frac{3}{2}}$. From Figure 2.6, we observe that the convergence rate of the discretization errors of the outward flux in the corresponding norm at the boundary is of optimal order for the penalty method, Nitsche's method and the method of Lagrange multipliers for both examples. By closely observing the error rates, we see that even with the same convergence rates the method of Lagrange multipliers has the smallest error in the outward flux. The convergence rate of the discretization error of the function at the boundary has the optimal rates for Nitsche's method and the method of Lagrange multipliers. Although, the convergence rates are same for Nitsche's method and the method of Lagrange multipliers, the discretization error due to the method of Lagrange multipliers is order of magnitude smaller than Nitsche's method. While for the penalty method, we again observe suboptimal convergence rates of the discretization error of the function at the boundary. Specifically, for the Example 1-FD the converge rate of the error of the function is of order $h^{\frac{1}{2}}$, while the convergence rate of the error for the Example 2-FD deteriorates slowly with decreasing mesh size.

2.7.3 Comparison of the Condition Numbers

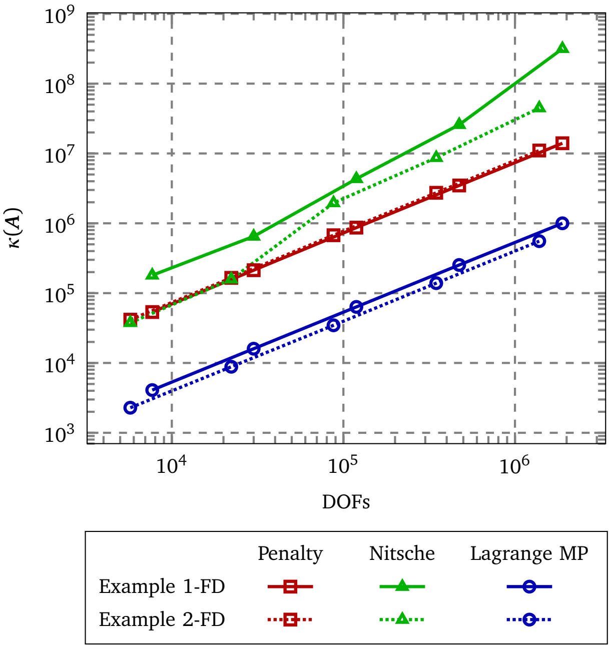


Figure 2.7. The condition numbers of the stiffness matrices **A** *arising from different methods against the number of DOFs, for Example 1-FD and Example 2-FD.*

In this section, we compare the condition numbers of the stiffness matrices arising from all three methods used to enforce the Dirichlet boundary conditions. The condition number of the stiffness matrices for the standard FEM discretization increases with decreasing mesh size and is of order h^{-2}. This does not hold for the unfitted discretization methods. The condition number of the system matrices arising from the unfitted discretization methods depends on the size of the cut elements, as the condition number of the stiffness matrices is dominated by the size of the smallest elements. In this chapter, we discussed the ghost penalty stabilization approach to eliminate the effect of the small cut elements on the condition numbers. The other source of the ill-conditioning of the stiffness matrices in the unfitted discretization methods is due to the stabilization parameter used by the penalty method and Nitsche's method. The condition number of a matrix $\boldsymbol{A}$ is given as

$$\kappa(\boldsymbol{A}) = \frac{|\lambda_{\max}(\boldsymbol{A})|}{|\lambda_{\min}(\boldsymbol{A})|},$$

where $\lambda_{\max}(\boldsymbol{A})$ and $\lambda_{\min}(\boldsymbol{A})$ denote the largest and the smallest eigenvalue of the matrix $\boldsymbol{A}$, respectively. As we know, the stiffness matrices arising from the method of Lagrange multipliers are symmetric positive semidefinite. The smallest eigenvalue of these matrices is always zero, and hence the condition number of these matrices is theoretically infinite. Due to this reason, in this work, we have to modify the method to compute the condition number of the positive semidefinite matrices. In this case, we rather use the smallest nonzero eigenvalue of a matrix to compute the condition number of the positive semidefinite matrices.

From Figure 2.7, we can see that the condition number of the stiffness matrices stemming from the method of Lagrange multipliers is the smallest for both examples. While for the stiffness matrices arising from the penalty method, the condition number is an order of magnitude higher. The system matrices stemming from the method of Lagrange multipliers do not consist of a stabilization parameter, while the stabilization parameter in the penalty method is chosen as $\gamma_p = 100$. Here, we note that the condition number of the system matrices still increases with decreasing mesh size with order h^{-2}. We do not observe any influence of the disproportionally cut elements on the condition numbers. This behavior is attributed to the inclusion of the ghost penalty stabilization term in the weak formulation. The largest condition number for all system matrices can be observed for Nitsche's method and unlike for the other methods, the condition numbers do not increase with order h^{-2}. This can be attributed to the value of the stabilization parameter in Nitsche's method. As the value of the stabilization parameters is computed using the generalized eigenvalue problem, its value depends on the size of the cut elements. Thus, based on the size of the cut elements on a given mesh, we can see oscillations in the condition numbers.

Chapter 3

Interface Problems

In many real-world applications, the field quantities change rapidly in the domain. These types of problems can be categorized as interface problems. In such cases, two or more distinct materials have different properties and due to this reason a function on the domain may exhibit non-smooth behavior. When the interface is smooth enough, the function is also smooth in each subdomain individually, but on the whole domain, the regularity or the smoothness of the function is generally very low. The interface problem is encountered often in solid mechanics, fluid dynamics, material science, geophysics, etc. In the solid mechanics, this type of problem arises in presence of cracks, dislocations or inclusions [BB99, SCMB01, HH04]. While in the field of fluid mechanics, the interface problem arises in multi-phase flows and in the presence of boundary layers [Reu08].

In this chapter, we discuss the discretization methods for interface problems. In the XFEM framework, the interface problems can be regarded as an extension of the fictitious domain method, where the interface problems are treated as overlapping fictitious domains. The interface problem in the traditional FEM framework requires a mesh such that the nodes on the interface and edges of the mesh align with the interface. This strict condition is waived in the XFEM framework, as the FE space is defined in the background mesh. In this chapter, we describe the XFEM discretization for the interface problem and the different methods to enforce the interface conditions. As we have seen in the last chapter the penalty method does not always produce optimal convergence rates, therefore we limit our discussion to Nitsche's method and the method of Lagrange multipliers.

In Section 3.1, we define a diffusion problem on a domain Ω with an interface. The diffusion coefficients across the interface are different, which leads to a non-smooth solution on the domain. In Section 3.2, we discuss the extended finite element method, the background mesh, and the extended finite element function spaces. Further, we provide a weak formulation of the interface problem and pose the problem as a constrained minimization problem with the interface condition as constraints. In Section 3.3, we review Nitsche's method and its extension from enforcing the Dirichlet boundary condition to enforcing the interface conditions. Later, we consider the method of Lagrange multipliers in Section 3.5, which employs the vital vertex algorithm to create a stable multiplier space. In the last section, we carry out some numerical experiments to show the convergence properties of the discussed methods and also to study the condition number of the stiffness matrices arising from all methods.

3.1 Model Problem

We consider a bounded Lipschitz domain $\Omega \subset \mathbb{R}^d$, $d \in \{2,3\}$ with interface Γ which decomposes the domain Ω into two non-overlapping subdomains Ω_1 and Ω_2, such that $\overline{\Omega} = \overline{\Omega_1 \cup \Omega_2 \cup \Gamma}$. The interface is defined as $\overline{\Gamma} = \partial\Omega_1 \cap \partial\Omega_2$ and it is assumed to be sufficiently smooth. For simplicity, the interface Γ is defined as polygonal. We define a sufficiently regular function $u_i : \Omega_i \cup \Gamma \to \mathbb{R}$ as a pair $(u_1, u_2) =: u$ in Ω. The jump in function u over the interface is defined as

$$[\![u]\!] := u_1|_\Gamma - u_2|_\Gamma,$$

where $u_i|_\Gamma$ is the restriction of u_i to Γ.

We consider a stationary diffusion problem with discontinuous coefficient α as

$$\begin{aligned} -\nabla \cdot \alpha \nabla u &= f && \text{in } \Omega_1 \cup \Omega_2, \\ u &= 0 && \text{on } \partial\Omega, \\ [\![u]\!] &= g_I && \text{on } \Gamma, \\ [\![\alpha \nabla_{\boldsymbol{n}} u]\!] &= 0 && \text{on } \Gamma, \end{aligned} \tag{3.1}$$

where $f \in L^2(\Omega)$ and $g_I \in H^{\frac{1}{2}}(\Gamma)$. The coefficient $\alpha \in \mathbb{R}^+$ is piecewise constant defined as

$$\alpha(\boldsymbol{x}) = \alpha_i \geqslant \alpha_0 > 0 \quad \forall \boldsymbol{x} \in \Omega_i.$$

The function g_I denotes a jump of the function u over the interface.

In problem (3.1), continuity of the function u and the continuity of the flux across the interface is enforced. Recall, the outward flux from the interface is defined as $\nabla_{\boldsymbol{n}} u = \boldsymbol{n} \cdot \nabla u$. For definiteness, we take the unit normal $\boldsymbol{n}$ as the outward pointing normal from Ω_1 on Γ, $\boldsymbol{n} = \boldsymbol{n}_1 = -\boldsymbol{n}_2$.

Problem (3.1) is consistent with the Poisson problem, if the coefficients are chosen as, $\alpha_1 = \alpha_2 = 1$. For the standard FEM approach, the interface conditions in (3.1) can only be imposed if nodes are placed on the interface explicitly. For the fitted finite element method, the detailed analysis of the interface problem with discontinuous coefficients is carried out under the assumption that the function is non-smooth only at the interface, the problem has a unique solution in H^2 on each convex subdomain Ω_i [CZ98].

3.2 Finite Element Discretization

In standard FEM discretization, the interface across which the function u is discontinuous has to be aligned with the element faces. In contrast, for the XFEM approach, this requirement is relaxed, and the interface is allowed to be anywhere in the domain. The XFEM discretization captures the interfaces by enriching the FEM solution space and then duplicating the elements that are intersected by the interface. The new degrees of freedom (DOFs) are then associated with the duplicated elements.

We assume a shape regular, quasi-uniform, conforming triangulation $\widetilde{\mathcal{T}}_h$ on the domain Ω. Unlike the last chapter, we assume that the triangulation $\widetilde{\mathcal{T}}_h$ is fitted on the domain Ω. Even though, this is not necessary as we can employ the methods from the last chapter to enforce the boundary conditions on the unfitted boundary. The triangulation $\widetilde{\mathcal{T}}_h$ thus captures both subdomains $(\Omega_1 \cup \Omega_2) \subseteq \widetilde{\mathcal{T}}_h$. We define the active mesh associated with each subdomain as

$$\mathcal{T}_{h,i} = \{K \in \widetilde{\mathcal{T}}_h : K \cap \Omega_i \neq \emptyset\} \quad i \in \{1,2\}.$$

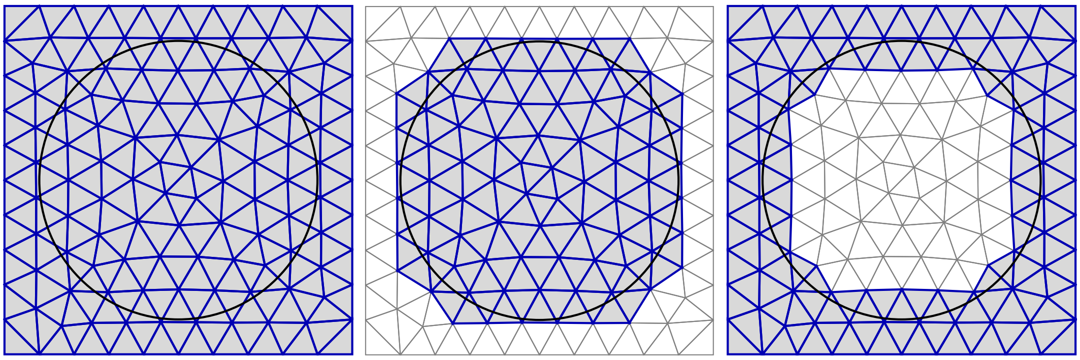
As shown in Figure (3.1), each subdomain Ω_i is captured by respective active mesh, $\Omega_i \subset \mathcal{T}_{h,i}$. Now, we define a submesh as union of all elements that are intersected with the interface Γ

$$\mathcal{T}_{h,\Gamma} = \{K \in \widetilde{\mathcal{T}}_h : K \cap \Gamma \neq \emptyset\}.$$

The interface triangulation $\mathcal{T}_{h,\Gamma}$ is doubled, $\mathcal{T}^i_{h,\Gamma} := \mathcal{T}_{h,\Gamma}$. This interface triangulation is part of both active meshes, denoted as $\mathcal{T}^i_{h,\Gamma} \subset \mathcal{T}_{h,i}, i \in \{1,2\}$. Additionally, for any element K, let $K_i = K \cap \Omega_i$ be part of K in domain Ω_i and for $K \in \mathcal{T}^i_{h,\Gamma}$, let $\Gamma_K := \Gamma \cap K$ be part of Γ in K.

We define continuous low order finite element space over the triangulation $\widetilde{\mathcal{T}}_h$, which vanishes on the boundary,

$$\widetilde{\mathcal{V}}_h = \{v \in H^1(\widetilde{\mathcal{T}}_h) : v|_K \in \mathcal{P}_1(K),\ v|_{\partial \widetilde{\mathcal{T}}_h} = 0,\ \forall K \in \widetilde{\mathcal{T}}_h\}.$$



(a) Triangular mesh $\widetilde{\mathcal{T}}_h$

(b) Active mesh $\mathcal{T}_{h,1}$ encapsulates Ω_1

(c) Active mesh $\mathcal{T}_{h,2}$ encapsulates Ω_2

Figure 3.1. 2D Triangular mesh is divided into two parts $\mathcal{T}_{h,1}$ and $\mathcal{T}_{h,2}$ as depicted in (b) and (c) due to the interface Γ, elements which belong to the respective domain are shaded.

Following the same strategy as in last chapter, we define a characteristic function of each subdomain $\Omega_i, i = 1, 2$, given as

$$\chi_{\Omega_i} : \mathbb{R}^d \to \mathbb{R}, \qquad \chi_{\Omega_i}(\boldsymbol{x}) = \begin{cases} 1 & \forall \boldsymbol{x} \in \overline{\Omega}_i, \\ 0 & \text{otherwise.} \end{cases} \tag{3.2}$$

The characteristic function χ_{Ω_i} is used to restrict the support of the finite element space to domain Ω_i, thus the space of finite elements in the domain Ω_i is defined as

$$\mathcal{V}_{h,i} = \chi_{\Omega_i}(\boldsymbol{x})\widetilde{\mathcal{V}}_h.$$

We seek the approximation $u_h = (u_{h,1} \oplus u_{h,2})$ in space $\mathcal{V}_h = \mathcal{V}_{h,1} \oplus \mathcal{V}_{h,2}$. From the definition of the FE space, it is clear that the function is allowed to be discontinuous across the interface.

The function space $\widetilde{\mathcal{V}}_h$ is spanned by the nodal basis functions $\widetilde{\Phi}_h = (\widetilde{\phi}_h^p)_{p \in \widetilde{\mathcal{N}_h}}$, where $\widetilde{\mathcal{N}_h}$ denotes the set of nodes of the background triangulation $\widetilde{\mathcal{T}}_h$. We define the set of nodes on the active meshes $\mathcal{T}_{h,i}, i = 1, 2$ associated with domain Ω_i as

$$\mathcal{N}_{h,i} := \{p \in \widetilde{\mathcal{N}_h} : \operatorname{supp}(\widetilde{\phi}_h^p) \cap \Omega_i \neq \emptyset\} \quad i = 1, 2.$$

We now define the "cut" basis function associated with a node p as

$$\phi_h^p = \chi_{\Omega_i}(\boldsymbol{x})\widetilde{\phi}_h^p \qquad \forall p \in \mathcal{N}_{h,i}, i = 1, 2.$$

The function space $\mathcal{V}_{h,i}$ is spanned by the nodal basis functions $\Phi_{h,i} = (\phi_h^p)_{p \in \mathcal{N}_i}$. We define the span of nodal basis function on $\mathcal{V}_h$ as $\Phi_h = \Phi_{h,1} \oplus \Phi_{h,2}$, and the set of nodes associated with the mesh $\mathcal{T}_h$ is given by $\mathcal{N}_h = \mathcal{N}_{h,1} \oplus \mathcal{N}_{h,2}$.

We have defined the FE space $\mathcal{V}_h$ on the mesh $\mathcal{T}_h$, now we can discretize the problem (3.1). Following the last chapter, we write the discretized variational problem as a constrained minimization problem. Find $u_h \in \mathcal{V}_h$ such that

$$\begin{aligned} \min_{u_h \in \mathcal{V}_h} \qquad & J(u_h) = \frac{1}{2}a(u_h, u_h) - F(u_h) \\ \text{subject to} \qquad & [\![u_h]\!] = g_I \qquad \text{on } \Gamma, \end{aligned} \tag{3.3}$$

where $J(\cdot)$ denotes the energy functional, $a(\cdot,\cdot) : \mathcal{V}_h \times \mathcal{V}_h \to \mathbb{R}$ is a continuous, symmetric bilinear form and $F(\cdot) : \mathcal{V}_h \to \mathbb{R}$ denotes a continuous linear form. The bilinear form and the linear form for the interface problem are defined as

$$\begin{aligned} a(u_h, v_h) &:= \sum_{i=1,2} (\alpha_i \nabla u_h, \nabla v_h)_{L^2(\Omega_i)}, \\ F(v_h) &:= \sum_{i=1,2} (f, v_h)_{L^2(\Omega_i)}. \end{aligned} \tag{3.4}$$

In the next sections, we discuss different strategies to impose interface conditions.

3.3 Nitsche's Method

In this section, we discuss Nitsche's method for imposing the interface conditions for the problem (3.1). Nitsche's method was introduced and analyzed for elliptic interface problems with the discontinuous coefficients in the unfitted finite element framework by Hansbo and Hansbo [HH02].

Before moving to the weak formulation, we define the weighted average function as

$$\{\!\{u\}\!\} = \left(\beta_1 u_1 + \beta_2 u_2\right) \quad \text{and} \quad \{\!\{\alpha \nabla_n u\}\!\} = \left(\beta_1 \alpha_1 \nabla_n u_1 + \beta_2 \alpha_2 \nabla_n u_2\right), \tag{3.5}$$

where $\beta_i \in \mathbb{R}^+$ are the weighting parameters. Here a choice of the weighting parameters play an important role, but for now we define the weighting parameters as the measure fraction defined on element $K \in \mathcal{T}_{h,\Gamma}$ [HH02], i.e.,

$$\beta_i = \frac{\text{meas}_d(K_i)}{\text{meas}_d(K)}. \tag{3.6}$$

As in Section 2.4, we have already introduced Nitsche's method for imposing the Dirichlet boundary condition. Here, we give the abstract variational formulation of Nitsche's method for the interface problem (3.1) in the context of unfitted FEM.

$$\text{Find } u_h \in \mathcal{V}_h \text{ such that} \qquad A_N(u_h, v_h) = F_N(v_h) \qquad \forall v_h \in \mathcal{V}_h. \tag{3.7}$$

Here, the bilinear from $A_N(\cdot,\cdot)$ and the linear form $F_N(\cdot)$ are defined as

$$\begin{aligned} A_N(u_h,v_h) = & \ a(u_h,v_h) - \langle \{\!\{\alpha\nabla_{\boldsymbol{n}} u_h\}\!\}, [\![v_h]\!]\rangle_\Gamma - \langle \{\!\{\alpha\nabla_{\boldsymbol{n}} v_h\}\!\}, [\![u_h]\!]\rangle_\Gamma \\ & + (\gamma_p [\![u_h]\!], [\![v_h]\!])_{H^{\frac{1}{2}}(\Gamma),h}, \\ F_N(v_h) = & \ F(v_h) - \langle \{\!\{\alpha\nabla_{\boldsymbol{n}} v_h\}\!\}, g_I\rangle_\Gamma + (\gamma_p g_I, [\![v_h]\!])_{H^{\frac{1}{2}}(\Gamma),h}. \end{aligned} \tag{3.8}$$

Following [HH02], the formulation (3.7) can be shown to be consistent with the strong formulation (3.1) and stable for a sufficiently large stabilization parameter. In the detailed analysis of this method, a priori error estimates are given under reasonable mesh assumptions [HH02]

$$\begin{aligned} |||u-u_h|||_h &\leqslant C\, h \sum_{i=1,2} \|u\|_{H^2(\Omega_i)} \qquad \forall u \in H_0^1(\Omega)\cap H^2(\Omega_1\cup\Omega_2), \\ \|u-u_h\|_{L^2(\Omega)} &\leqslant C\, h^2 \sum_{i=1,2} \|u\|_{H^2(\Omega_i)} \qquad \forall u \in H_0^1(\Omega)\cap H^2(\Omega_1\cup\Omega_2), \end{aligned} \tag{3.9}$$

where the constant C is completely independent of the location of the interface in the mesh. The mesh-dependent energy norms $|||\cdot|||_h$ in the above estimates are defined as

$$|||v|||_h^2 := \|\nabla v\|^2_{L^2(\Omega_1\cup\Omega_2)} + \|[\![v]\!]\|^2_{H^{\frac{1}{2}}(\Gamma),h} + \|\{\!\{\nabla_{\boldsymbol{n}} v\}\!\}\|^2_{H^{-\frac{1}{2}}(\Gamma),h}.$$

We note that unlike Nitsche's formulation for the fictitious domain method, we have not added the ghost penalty term to the weak formulation. The ghost penalty term can also be added to Nitsche's formulation (3.7) and of course, it improves the conditioning of the system of linear equation for the same reasons discussed in Section 2.3.2. For the interface problem, Nitsche's method includes the weighted averaging function $\{\!\{\cdot\}\!\}$, which enforces the flux at the interfaces to take the average value of the fluxes from both subdomains. This averaging function introduces extra stability to Nitsche's formulation for the interface problems. In the fictitious domain method, such a term does not exist, thus we added the ghost penalty term by default to the bilinear form in order to impose the Dirichlet boundary conditions in a stable manner.

Coercivity: As we mentioned earlier, the stability and convergence of Nitsche's method rely on the choice of the stabilization parameter. Following the coercivity of the bilinear form $A_N(\cdot,\cdot)$, we have

$$\begin{aligned} A_N(v_h,v_h) \geqslant & \sum_{K\in\widetilde{\mathcal{T}}_h\setminus\mathcal{T}_{h,\Gamma}} \|\alpha^{\frac{1}{2}}\nabla v_h\|^2_{L^2(K)} + \frac{1}{\epsilon}\|\{\!\{\alpha\nabla_{\boldsymbol{n}} v_h\}\!\}\|^2_{H^{-\frac{1}{2}}(\Gamma),h} \\ & + \sum_{K\in\mathcal{T}_{h,\Gamma}} \Big(1-\frac{2C_\gamma}{\epsilon}\Big)\|\alpha^{\frac{1}{2}}\nabla v_h\|^2_{L^2(K)} + (\gamma_p-\epsilon)\|[\![v_h]\!]\|^2_{H^{\frac{1}{2}}(\Gamma),h}. \end{aligned}$$

This inequality uses Young's inequality for some $\epsilon > 0$, the following trace inequality on the interface (see A.2)

$$\|\{\!\{\alpha \nabla_n u_h\}\!\}\|^2_{H^{-\frac{1}{2}}(\Gamma_K),h} \leqslant C_\gamma \sum_{i=1}^{2} \|\alpha_i^{\frac{1}{2}} \nabla u_h\|^2_{L^2(K_i)} \quad \forall K \in \mathcal{T}_{h,\Gamma}. \tag{3.10}$$

Following the same methodology from the last chapter, the bilinear form is coercive if $\gamma_p \geqslant \epsilon$ and $\epsilon \geqslant 2C_\gamma$. Thus, the stabilization parameter can be given with the bound $\gamma_p \geqslant 2C_\gamma$. The constant C_γ in the trace inequality depends on the shape regularity of the cut elements K_i, when weighting parameters in the definition of the averaging function are given by (3.6). In the next part, we consider a robust option for the weighting parameters.

Robust Weighting parameters: In addition to the stabilization parameter, the weighting parameters also play an important role in the stability of Nitsche's formulation. If the stabilization parameter is chosen by estimating the constant C_γ in the trace inequality (3.10), it is beneficial to make the constant C_γ a function of shape regularity of cut elements and the coefficients. A robust option for the weighting parameters is given in [BBDL12, AHD12], where it is suggested to use the weighting parameters that include diffusion coefficient α and the measure of the cut element. The updated weighting parameters are given as

$$\beta_i = \frac{\text{meas}_d(K_i)/\alpha_i}{\text{meas}_d(K_1)/\alpha_1 + \text{meas}_d(K_2)/\alpha_2} \quad \text{for } i = 1, 2. \tag{3.11}$$

These weighting parameters provide better averaging for the discontinuous coefficients, specially for highly varying coefficients. In [BBDL12], the jump of discrete flux and the L^2-error in the flux were compared for a different definition of the weighting parameters. For highly varying coefficients, it was demonstrated that as an interface approaches a node or an edge of the mesh, Nitsche's method equipped the original weighting parameter (3.6) leads to large error compared to the updated weighting parameters [BBDL12].

Unless otherwise specified, we utilize the updated weighting parameters in our work. In the next section, we discuss modifications to (3.7), and approaches to estimate the stabilization parameters by estimating the constant C_γ.

3.4 Variants of Nitsche's Method

In this section, we describe three different variants of Nitsche's method. In the first variant, we estimate the stabilization parameter by solving a generalized eigenvalue problem. For the second variant, we use the lifting operator approach to circumvent the process of carefully choosing the stabilization parameter. While in the last approach, we use the ghost penalty stabilization in Nitsche's method and also change the definition of the weighting parameters to make the formulation robust with respect to highly varying coefficients [BZ12].

3.4.1 Eigenvalue Problem

We restate the weak formulation of Nitsche's method as,

$$\text{find } u_h \in \mathcal{V}_h \text{ such that} \qquad A_N^1(u_h, v_h) = F_N^1(v_h) \qquad \forall v_h \in \mathcal{V}_h, \tag{IN-EV}$$

where

$$\begin{aligned} A_N^1(u_h, v_h) &= a(u_h, v_h) - \langle \{\!\{\alpha \nabla_{\boldsymbol{n}} u_h\}\!\}, [\![v_h]\!] \rangle_\Gamma - \langle [\![u_h]\!], \{\!\{\alpha \nabla_{\boldsymbol{n}} v_h\}\!\} \rangle_\Gamma \\ &\quad + (\gamma_p^1 [\![u_h]\!], [\![v_h]\!])_{H^{\frac{1}{2}}(\Gamma),h}, \\ F_N^1(v_h) &= F(v_h) - \langle g_I, \{\!\{\alpha \nabla_{\boldsymbol{n}} v_h\}\!\} \rangle_\Gamma + (\gamma_p^1 g_I, [\![v_h]\!])_{H^{\frac{1}{2}}(\Gamma),h}, \end{aligned}$$

where the weighting parameters β_i are defined as in (3.11). As we know, one can estimate the constant C_γ in the trace inequality at the interface by solving a generalized eigenvalue problem.

The eigenvalue problem for each $K \in \mathcal{T}_{h,\Gamma}$ is given as, find $\max(\lambda_K) \in \mathbb{R}$ such that

$$(\{\!\{\alpha \nabla_{\boldsymbol{n}} v_h\}\!\}, \{\!\{\alpha \nabla_{\boldsymbol{n}} v_h\}\!\})_{H^{-\frac{1}{2}}(\Gamma_K),h} = \lambda_K \sum_{i=1}^{2} (\alpha_i \nabla v_h, \nabla v_h)_{L^2(K_i)} \qquad \forall v_h \in \mathcal{V}_h|_K, \tag{3.12}$$

where $\mathcal{V}_h|_K$ is restriction of $\mathcal{V}_h$ on a given element K, and λ_K denotes the set of eigenvalues. In order to solve the eigenvalue problem (3.12), we still have to remove the influence of non-trivial kernel from the right hand side term. For the interface problem, we note that there exist two non-trivial kernels associated with cut elements on each subdomain. We follow the same strategy as in Section 2.5.1 and use a deflation method.

Once we have the largest eigenvalue, the stabilization parameter is computed element-wise to satisfy the condition $\gamma_p^1 \geqslant 2C_\gamma$, thus it is given as

$$\gamma_p^1 = 4 \max(\lambda_K).$$

3.4.2 Lifting Operator

In this section, we borrow the same definition of the lifting operator $\mathcal{R}_K(\cdot)$ and the space of polynomials which are orthogonal to constants $\mathcal{W}_h$ from Section 2.5.2. On the uncut elements, i.e., $K \in \widetilde{\mathcal{T}_h} \setminus \mathcal{T}_{h,\Gamma}$, we set the lifting operator as $\mathcal{R}_K(u_h) = 0$. While on the cut elements, the definition of the lifting operator is updated for the interface problem. Find $w_h := \mathcal{R}_K(u_h) \in \mathcal{W}_h$ such that

$$\sum_{i=1}^{2} (\alpha \nabla w_h, \nabla v_h)_{L^2(K_i)} = -\langle \{\!\{\alpha \nabla_{\boldsymbol{n}} v_h\}\!\}, [\![u_h]\!]\rangle_{\Gamma_K} \quad \forall u_h, v_h \in \mathcal{V}_h|_K.$$

Now, by adding the new term stemming from the lifting operator of the original formulation (3.7) we can ensure the coercivity of the bilinear form,

$$a(u_h,u_h) - 2\langle \{\!\{\alpha \nabla_{\boldsymbol{n}} u_h\}\!\}, [\![u_h]\!]\rangle_\Gamma + 2a(\mathcal{R}_K(u_h), \mathcal{R}_K(u_h)) \geqslant \frac{1}{2} a(u_h,u_h).$$

As the coercivity of the bilinear form is ensured regardless of the value of the stabilization parameter, we can have any positive value for the stabilization parameter. The updated weak formulation is given as,

$$\text{find } u_h \in \mathcal{V}_h \text{ such that} \quad A_N^2(u_h,v_h) = F_N^2(v_h) \qquad \forall v_h \in \mathcal{V}_h, \tag{IN-LO}$$

where

$$\begin{aligned} a_N^2(u_h,v_h) &= a(u_h,v_h) - \langle \{\!\{\alpha \nabla_{\boldsymbol{n}} u_h\}\!\}, [\![v_h]\!]\rangle_\Gamma - \langle \{\!\{\alpha \nabla_{\boldsymbol{n}} v_h\}\!\}, [\![u_h]\!]\rangle_\Gamma \\ &\quad + 2a(\mathcal{R}_K(u_h), \mathcal{R}_K(v_h)) + (\gamma_p^2 [\![u_h]\!], [\![v_h]\!])_{H^{\frac{1}{2}}(\Gamma),h}, \\ F_N^2(v_h) &= F(v_h) - \langle g_I, \{\!\{\alpha \nabla_{\boldsymbol{n}} v_h\}\!\}\rangle_\Gamma + (\gamma_p^2 g_I, [\![v_h]\!])_{H^{\frac{1}{2}}(\Gamma),h}. \end{aligned}$$

In the original work [Leh16], the weighting parameters are chosen as volume fraction of the cut element and the original element (3.6), and the stabilization parameter is chosen as $\gamma_p = 1$. In case of highly varying coefficients, we have noticed that the updated weighting parameters given in (3.11) produce better results. Hence, we use the robust weighting parameters and for the stabilization parameter we employ harmonic averaging of the coefficients

$$\gamma_p^2 = \frac{2\alpha_1\alpha_2}{\alpha_1 + \alpha_2}.$$

Here, we prefer the harmonic averaging of the coefficients as a stabilization parameter, as it bounds the value of the stabilization parameter from above with the smaller value of coefficients, e.g., $\gamma_p^2 \leqslant 2\inf(\alpha_1, \alpha_2)$. This definition of the stabilization parameter avoids the cases where the value of the stabilization parameter becomes dominant in the presence of very small coefficients, e.g., $\alpha_i \ll 1$.

3.4.3 Ghost Penalty Stabilization

For the standard Nitsche's formulation, as we have mentioned earlier, the condition number of the system matrix depends on the cut position and it can become arbitrarily bad when an interface passes very close to element faces or nodes. The ghost penalty method overcomes this issue by extending the coercivity of the bilinear form, from the computational domain to the fictitious part. If the ghost penalty stabilization term is added to the formulation, Nitsche's method becomes stable even for pathological cases [dPLM18].

We define the set of faces $\mathcal{G}^i_{h,\Gamma}$ for each subdomain Ω_i

$$\mathcal{G}^i_{h,\Gamma} = \{G \subset \partial K \mid K \in \mathcal{T}^i_{h,\Gamma},\ \partial K \cap \partial \mathcal{T}_i = \emptyset\} \quad i = 1,2.$$

The ghost penalty term is defined as

$$g(u_h, v_h) = \sum_{i=1}^{2} \sum_{G \in \mathcal{G}^i_{h,\Gamma}} \epsilon_G h_G \alpha_i (\llbracket \nabla_{\boldsymbol{n}_G} \mathcal{E}_{h,i} u_h \rrbracket, \llbracket \nabla_{\boldsymbol{n}_G} \mathcal{E}_{h,i} v_h \rrbracket)_{L^2(G)}, \tag{3.13}$$

where h_G denotes the diameter of face G, $\boldsymbol{n}_G$ is unit normal to face G and ϵ_G is a positive constant. Here, $\mathcal{E}_{h,i}$ denotes the canonical extension of the function from the domain to the background mesh, which is defined as $\mathcal{E}_{h,i} : \mathcal{V}_h|_{K_i} \to \widetilde{\mathcal{V}}_h|_K$. In the previous section, we observed that the choice of the averaging function is also quite important. For the problems with highly varying coefficients α, this stabilization term is an attractive option, when the averaging function does not provide sufficient stability. Following [BZ12] the definition of the β_i is changed for the highly varying coefficients to

$$\beta_1 = \frac{\alpha_2}{\alpha_1 + \alpha_2} \quad \text{and} \quad \beta_2 = \frac{\alpha_1}{\alpha_1 + \alpha_2}.$$

The stabilization parameter can also be chosen to be coefficient dependent, thus as

$$\gamma^3_p = \gamma_{gp} \frac{2\alpha_1\alpha_2}{\alpha_1 + \alpha_2}, \quad \text{where } \gamma_{gp} \in \mathbb{R}^+.$$

Now, the weak formulation equipped with the ghost penalty term is given as,

$$\text{find } u_h \in \mathcal{V}_h \text{ such that} \quad A^3_N(u_h, v_h) = F^3_N(v_h) \qquad \forall v_h \in \mathcal{V}_h, \tag{IN-GP}$$

where

$$\begin{aligned} a^3_N(u_h, v_h) &= a(u_h, v_h) - \langle \{\!\{\alpha \nabla_{\boldsymbol{n}} u_h\}\!\}, \llbracket v_h \rrbracket \rangle_\Gamma - \langle \{\!\{\alpha \nabla_{\boldsymbol{n}} v_h\}\!\}, \llbracket u_h \rrbracket \rangle_\Gamma \\ &\quad + (\gamma^3_p \llbracket u_h \rrbracket, \llbracket v_h \rrbracket)_{H^{\frac{1}{2}}(\Gamma),h} + g(u_h, v_h), \\ F^3_N(v_h) &= F(v_h) - \langle g_I, \{\!\{\alpha \nabla_{\boldsymbol{n}} v_h\}\!\} \rangle_\Gamma + (\gamma^3_p g_I, \llbracket v_h \rrbracket)_{H^{\frac{1}{2}}(\Gamma),h}. \end{aligned}$$

The error analysis of this method was carried out in [BZ12] and it was shown to have optimal convergence rates in the L^2-norm and the mesh-dependent energy norm for highly varying coefficients.

3.5 The Lagrange Multiplier Formulation

In this section, we describe the method of Lagrange multiplier for imposing the interface conditions for the problem (3.1).

We have already introduced the method of Lagrange multiplier for imposing the Dirichlet boundary conditions in the previous chapter. The abstract variational formulation of the Lagrange multiplier method of the interface problem in the XFEM framework can be given as, find $(u_h, \lambda_h) \in \mathcal{V}_h \times \mathcal{M}_h$ such that

$$\begin{aligned} a(u_h, v_h) + g(u_h, v_h) + b(\lambda_h, v_h) &= F(v_h) \qquad \forall v_h \in \mathcal{V}_h, \\ b(\mu_h, u_h) &= G_I(\mu_h) \qquad \forall \mu_h \in \mathcal{M}_h. \end{aligned} \tag{3.14}$$

Here, the bilinear form $b(\cdot,\cdot) : \mathcal{M}_h \times \mathcal{V}_h \to \mathbb{R}$ and the linear form $G_I : \mathcal{M}_h \to \mathbb{R}$ are defined as

$$b(\lambda_h, u_h) := \sum_{K \in \mathcal{T}_{h,\Gamma}} \langle \lambda_h [\![u_h]\!] \rangle_{\Gamma_K} \quad \text{and} \quad G_I(\lambda_h) := \sum_{K \in \mathcal{T}_{h,\Gamma}} \langle \lambda_h, g_I \rangle_{\Gamma_K},$$

while the ghost penalty term is defined as (3.13). In Nitsche's method there exist a weighted averaging function that stabilizes the flux at the interface. As we lack a similar term in the Lagrange multiplier formulation, we introduce the ghost penalty stabilization term in the bilinear form.

As discussed in Section 2.6 at length, for the method of Lagrange multipliers it is essential to create the multiplier space in such a way that discrete inf-sup condition (2.17) is satisfied. Following the discussion in the previous chapter, we use the vital vertex algorithm to construct the multiplier space. We note that even though we employed the vital vertex algorithm in the fictitious domain method for enforcing boundary conditions. The original work by Béchet et al. [BMW09] was developed for enforcing the stiff interface conditions. In our knowledge the detailed error analysis of the method is not available, but based on the abstract saddle point theory the discretization error could be given as

$$\begin{aligned} |||u - u_h|||_h &\leqslant Ch \|u\|_{H^2(\Omega)} \quad \forall u_h \in \mathcal{V}_h, \\ \|\lambda - \lambda_h\|_{H^{-\frac{1}{2}}(\Gamma)} &\leqslant Ch \|\lambda\|_{H^{\frac{1}{2}}(\Gamma)} \quad \forall \lambda_h \in \mathcal{M}_h. \end{aligned} \tag{3.15}$$

Here, the mesh-dependent energy norm $|||\cdot|||_h$ in the above estimate is defined as

$$|||v|||_h^2 := \|\nabla v\|^2_{L^2(\Omega)} + \|[\![v]\!]\|^2_{H^{\frac{1}{2}}(\Gamma),h}.$$

3.6 Numerical Results

In this section, we introduce three different numerical examples. Through these numerical examples, we compare the different variants of Nitsche's methods and the method of Lagrange multipliers in terms of the discretization errors and the condition number of the system matrices.

3.6.1 Problem Description

We consider a domain $\Omega = [0,1]^2$ with two different types of interfaces, a linear interface, and a circular interface. All experiments are carried out on a triangular structured mesh, except one experiment that was carried out on a quadrilateral mesh. We start with a mesh that has 100 elements in each direction, denoted as $L1$, and uniformly refine the mesh on level $L1$ to obtain different meshes as shown in Table 3.1. The same mesh hierarchy is later used as the multilevel hierarchy in the multigrid method. We remark that the DOFs and number of elements reported in Table 3.1 include the enriched nodes and the enriched elements. We consider problems with continuous and discontinuous coefficients to analyze the effect of different variants of Nitsche's method on the condition number of the linear systems and the numerical accuracy of the discretization methods.

Example 1-IF We consider a diffusion problem where $\alpha_1 = \alpha_2 = 1$, and a linear interface Γ_l. For this example, the right-hand side f_1 and the Dirichlet boundary conditions are chosen in such a way that the exact solution, $u_1 = (\exp(-500s)-1)(\exp(-500t)-1)(\exp(-500yy)-1)(1-3rr)^2$ is satisfied. Here, $s := (x-1/3)^2$, $t := (x-2/3)^2$, $xx := (x-1/2)^2$, $yy := (y-1/2)^2$ and $rr := xx+yy$. The linear interface Γ_l is defined as a zero level set of the function $\Lambda_l(x,y) := x - 1/\sqrt{2}$.

levels	$h_{\max}$	Linear Interface Γ_l		Circular Interface Γ_c	
		# DOFs	# Elems	# DOFs	# Elems
$L1$	1.41421E-02	10,403	20,200	10,767	20,566
$L2$	7.07107E-03	40,803	80,400	41,531	81,130
$L3$	3.53553E-03	161,603	320,800	163,063	322,262
$L4$	1.76777E-03	643,203	1,281,600	646,127	1,284,526
$L5$	8.83883E-04	2,566,403	5,123,200	2,572,251	5,129,050

Table 3.1. The multilevel hierarchy of meshes with triangular elements, with total number of DOFs and number of elements on a given level.

We have deliberately chosen the location of interface Γ_l in such a way that the interface would stay close to edges of the elements for all levels of the refinement process and would not coincide with the element edges. Thus, the enriched elements are divided into disproportional fractions.

Example 2-IF For this example, we consider a problem with discontinuous coefficients and a circular interface Γ_c. The circular interface Γ_c is defined as a zero level set of a function $\Lambda_c(\boldsymbol{x}) := r_0^2 - \|\boldsymbol{x}-\boldsymbol{c}\|_2^2$ with radius, $r_0^2 = 3-\sqrt{2}$, and $\boldsymbol{c}$ is the center of the circle $(0.5, 0.5)$. The circular interface decomposes the domain Ω into Ω_1, where $\Lambda_c(\boldsymbol{x}) > 0$ and Ω_2 where $\Lambda_c(\boldsymbol{x}) < 0$.

We consider a diffusion problem, where we choose coefficients as $\alpha_2 = 1$ and $\alpha_1 = \{10^{-1}, 10^{-5}, 10^{-9}\}$. For this example, the right-hand side is chosen as $f_2 = -4\alpha_1\alpha_2$, and the Dirichlet boundary conditions satisfy the exact solution

$$u_2(\boldsymbol{x}) = \begin{cases} \alpha_2(\|\boldsymbol{x}-\boldsymbol{c}\|_2^2 - r_0^2), & \text{if } \boldsymbol{x} \in \Omega_1, \\ \alpha_1(\|\boldsymbol{x}-\boldsymbol{c}\|_2^2 - r_0^2), & \text{if } \boldsymbol{x} \in \Omega_2. \end{cases}$$

Example 3-IF Here, we consider the same circular interface as in Example 2-IF. In this example, we consider the coefficients as $\alpha_1 = 1$ and $\alpha_2 = \{10, 10^5, 10^9\}$. The right-hand side is chosen as, $f_3 = -4$, and the Dirichlet boundary conditions satisfy the exact solution

$$u_3(\boldsymbol{x}) = \begin{cases} \dfrac{\|\boldsymbol{x}-\boldsymbol{c}\|_2^2}{\alpha_1}, & \text{if } \boldsymbol{x} \in \Omega_1, \\ \dfrac{\|\boldsymbol{x}-\boldsymbol{c}\|_2^2 - r_0^2}{\alpha_2} + \dfrac{r_0^2}{\alpha_1}, & \text{if } \boldsymbol{x} \in \Omega_2. \end{cases}$$

Using these examples we compare different variants of Nitsche's methods, with the eigenvalue problem, with the lifting operator method, and with the ghost penalty stabilization. Also, we compare the Example 2-IF and Example 3-IF for the method of Lagrange multipliers. The parameters in the ghost penalty term are defined as $\epsilon_G = 0.1$ and the stabilization parameter is defined as $\gamma_{gp} = 10$. For the method of Lagrange multipliers, we use both the triangular and the quadrilateral elements with the same mesh sizes in the hierarchy, and we denote these discretizations as LM-T and LM-Q, respectively.

3.6.2 Study of Discretization Errors

We compare the convergence rate of error in L^2-norm and in H^1-seminorm against the condition number of the stiffness matrix, denoted as $\kappa(\boldsymbol{A})$. The motivation behind this comparison is to investigate the influence of previously discussed variants of Nitsche's methods and the method of Lagrange multipliers on condition numbers and the discretization errors. As the background mesh and the location of the interface is fixed for a given example, the condition number of the system matrix is only affected by the choice of stabilization parameter γ_p and the weighting parameters β_i.

Domain Error: From Figure 3.2 and Figure 3.3, it is clear that all variants of Nitsche's methods have almost identical convergence rates for the discretization error in both L^2-norm and the H^1-seminorm. As the error estimates in (3.9) and (3.15) suggest, the convergence rate of discretization error in L^2-norm is of order h^2 and in the H^1-seminorm is of order h. Figure 3.2 demonstrates that FEM and XFEM methods have the same approximation properties since both methods produce the same discretization error for the same mesh size. The discretization error for Example 2-IF and Example 3-IF are also almost identical in both norms for all variants of Nitsche's method and both discretizations of the Lagrange multiplier method. From Figure 3.3, we see the convergence rates for both discretizations of the Lagrange multiplier method are optimal, but the error in the L^2-norm for the quadrilateral meshes is little larger than for its counterpart which employs triangular meshes.

Thus, we can conclude that all variants of Nitsche's formulations and the Lagrange multiplier formulations are robust with respect to highly varying coefficients.

Interface Error: In Figure 3.4, we see the discretization error at the interfaces for different mesh sizes used for enforcing the interface conditions. From the error estimates of the interface errors, it is known that the convergence rate of the discretization error of the function at the interface in $H^{\frac{1}{2}}(\Gamma), h$-norm and the outward flux at the interface in $H^{-\frac{1}{2}}(\Gamma), h$-norm is of order $h^{\frac{3}{2}}$. We can observe from Figure 3.4 that the convergence rates for all the variants of Nitsche's method in both norms are of optimal order. Even with the highly varying coefficients, we do not observe any deterioration in the interface errors.

For the method of Lagrange multipliers, the errors at the interfaces are not of optimal order for the triangular meshes. This is due to the fact that we have used the vital vertex algorithm to generate a stable multiplier space. As we know,

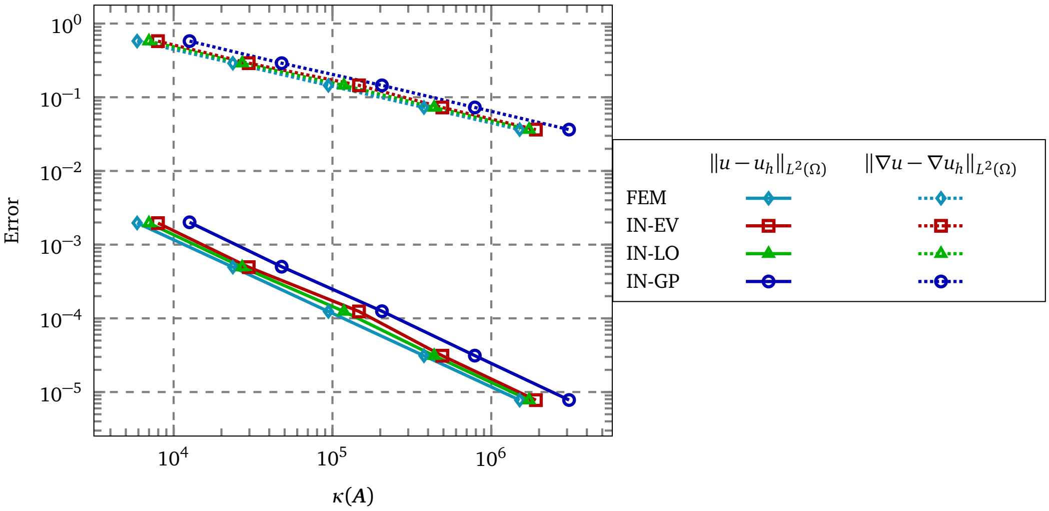
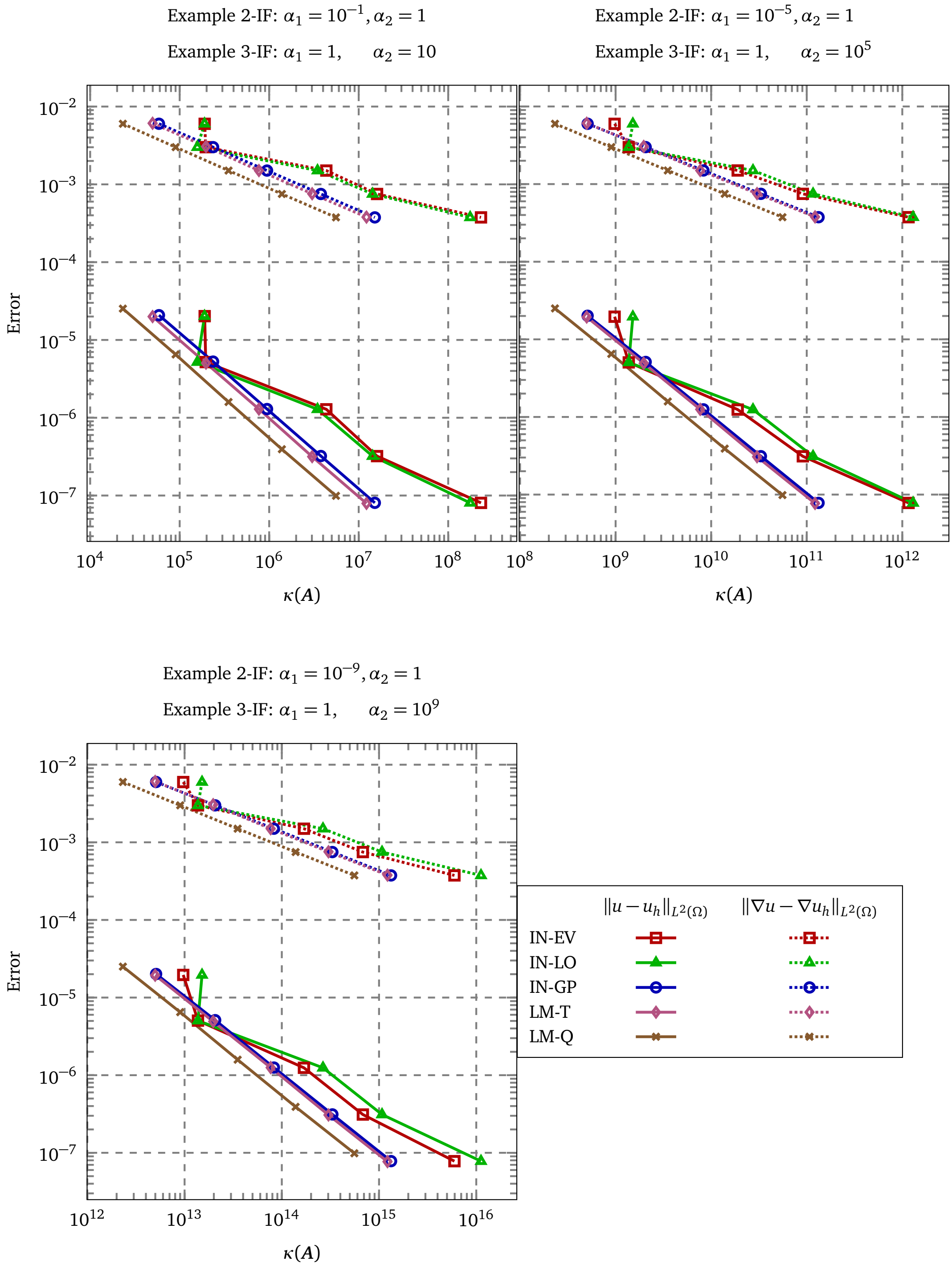


Figure 3.2. Discretization error in L^2-norm and mesh-dependent energy norm (except for FEM formulation, where we use H^1-seminorm) for different variants of Nitsche's method applied to Example 1-IF.

the coarsening for the multiplier space influences the approximation error of the primal variable (solution). For the triangular meshes, the number of edges that are connected with the endpoints of the edges is comparatively larger, and thus the coarsening of the multiplier space reduces the approximation of the function at the interfaces. We can observe that for the LM-T discretization, the error of the function and the outward flux at the interfaces are suboptimal compared to Nitsche's method. If we change the discretization from the triangular mesh to the quadrilateral mesh, we recover the optimal convergence rates for the error at the interfaces. The discretization error of the outward flux in $H^{-\frac{1}{2}}(\Gamma), h$-norm produced by the LM-Q discretization is identical to Nitsche's methods, while for the errors of the function in $H^{\frac{1}{2}}(\Gamma), h$-norm produced by the same discretization is larger than Nitsche's methods. Thus, for the method of Lagrange multipliers, we use only the quadrilateral meshes from now onwards.

3.6.3 Comparison of the Condition Numbers

As the Example 1-IF has continuous coefficients, we can compare the FEM discretization with the XFEM discretization on the same background mesh. We observe that FEM discretization has the smallest $\kappa(\boldsymbol{A})$ for all different mesh sizes in comparison with its XFEM counterparts. The XFEM discretizations produce the system of linear equations with a larger condition number for all cases. The



Example 2-IF: $\alpha_1 = 10^{-1}, \alpha_2 = 1$

Example 3-IF: $\alpha_1 = 1, \quad \alpha_2 = 10$

Example 2-IF: $\alpha_1 = 10^{-5}, \alpha_2 = 1$

Example 3-IF: $\alpha_1 = 1, \quad \alpha_2 = 10^5$

Example 2-IF: $\alpha_1 = 10^{-9}, \alpha_2 = 1$

Example 3-IF: $\alpha_1 = 1, \quad \alpha_2 = 10^9$

Figure 3.3. Discretization error in L^2-norm and H^1-seminorm in the domain Ω for different methods applied to Example 2-IF and Example 3-IF.

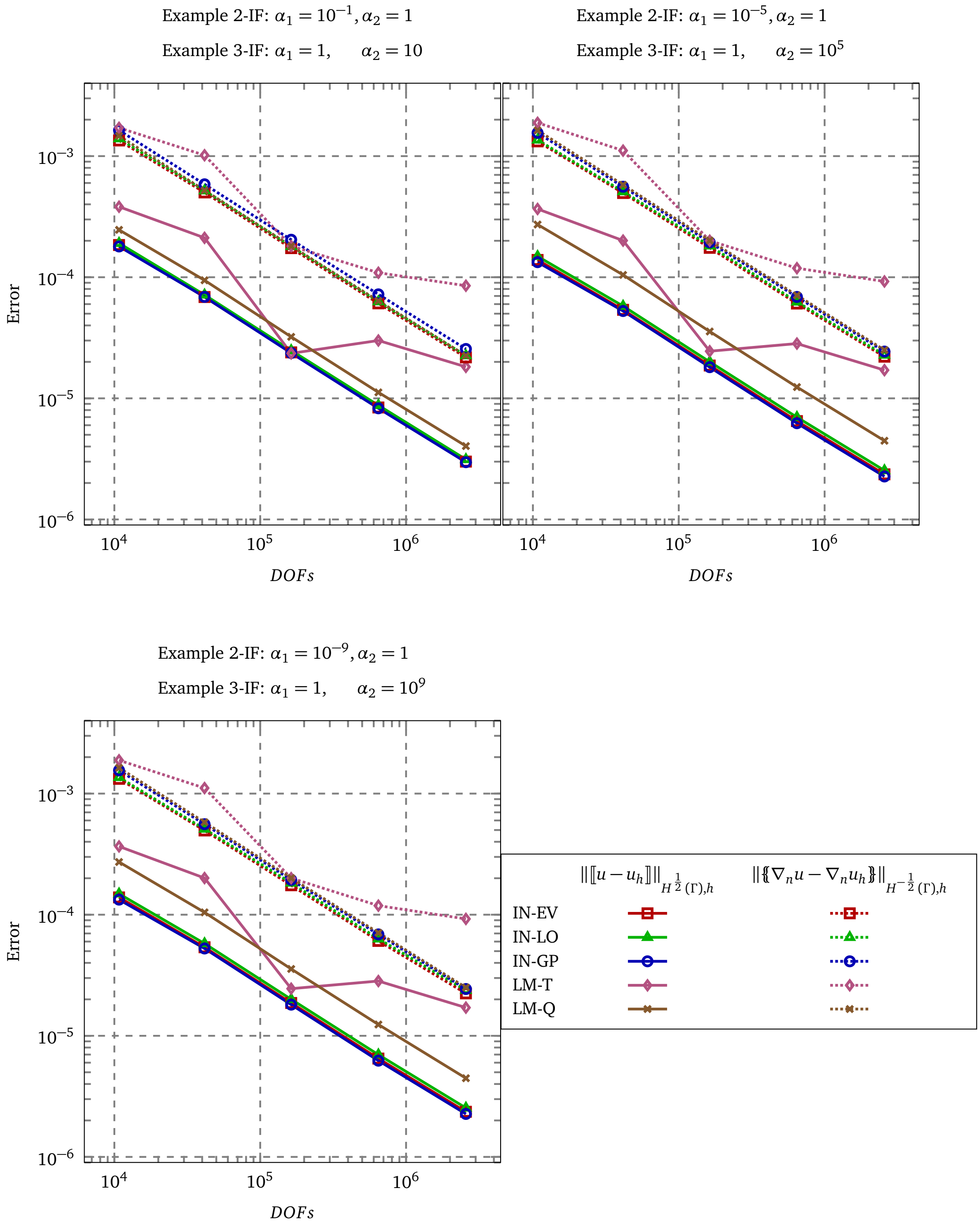


Figure 3.4. Discretization error of the function in $H^{\frac{1}{2}}(\Gamma), h$-norm and flux in $H^{-\frac{1}{2}}(\Gamma), h$-norm for different methods applied to Example 2-IF and Example 3-IF.

condition number for Nitsche's formulation equipped with the lifting operator is closest to the FEM discretization, while (IN-EV) formulation is the close second. The condition numbers of the (IN-GP) formulation is the largest for all mesh sizes, but this can be attributed to the value of the stabilization parameter.

Example 2-IF and Example 3-IF are different in terms of the coefficients. The ratio between the smallest and the largest coefficient is kept the same for both examples. For a given method to enforce the interface condition and a given ratio between coefficients, we witness the identical results in terms of error and condition number of the system for both examples. Application of the Dirichlet boundary condition to the stiffness matrix in both examples causes the distribution of the eigenvalues in the spectrum to vary, but the ratio between the largest and the smallest eigenvalues stays the same. Figure 3.3 shows a comparison of the condition numbers against the discretization errors for both examples for various coefficients. It is evident from Figure 3.3, that the condition numbers and the discretization errors in L^2-norm and H^1-seminorm for both examples are identical. The condition number is the smallest for the method of Lagrange multipliers discretized on the quadrilateral mesh. Among the variants of Nitsche's method, the condition number is the smallest for the ghost penalty discretization, regardless of the scale of coefficients, and $\kappa(\boldsymbol{A})$ grows with decreasing mesh size, $\mathcal{O}(h^{-2})$. The theoretical estimates of Nitsche's method with the ghost penalty term, suggest that $\kappa(\boldsymbol{A})$ is completely independent of the location of the interface on mesh and only depends on the coefficients. This is also evident for the condition numbers of the system matrices arising from the method of Lagrange multipliers discretizations LM-T and LM-Q, as we have also used the ghost penalty term in the Lagrangian formulations. The theoretical estimates of Nitsche's method with the ghost penalty term, suggest that $\kappa(\boldsymbol{A})$ is completely independent of the location of the interface on mesh and only depends on the coefficients. Experimental results also support the theoretical estimates of (IN-GP) discretization for all examples. There are no theoretical bounds established on $\kappa(\boldsymbol{A})$ for (IN-LO) and (IN-EV) discretizations. For different mesh sizes, $\kappa(\boldsymbol{A})$ is larger than the ghost penalty formulation in almost all cases but it grows as the ratio between the coefficients increases. We also observe the effect of irregular intersection between the interface and the meshes at different levels.

From this discussion, it is clear that Nitsche's method with the ghost penalty term is more robust amongst variants of Nitsche's methods. While for the method of Lagrange multipliers, the discretization on the quadrilateral meshes is stable. The other two variants of Nitsche's methods produce the system matrices with larger condition numbers, but still, these variants are stable as their condition numbers do not grow sporadically.

Chapter 4

A Multigrid Method for Nitsche-XFEM

As mentioned in the introduction, a multigrid method is a combination of smoothing and coarse level corrections. The efficiency of the coarse level corrections is heavily dependent on well-chosen transfer operators that can be used to restrict residual from a fine level to a coarse level and prolongate correction from a coarse level to a fine level. In this chapter, we introduce our multigrid method for the XFEM discretization and discuss a new transfer operator based on the L^2-projection and pseudo-L^2-projection in the XFEM framework. We note, this multilevel method relies only on the unfitted meshes and the enriched FE spaces and it is agnostic of any method chosen to enforce the boundary or the interface conditions.

This chapter is organized as follows. In the next section, we describe the linear system, arising from either the penalty method or Nitsche's method. In Section 4.2, we present the framework of the subspace correction method for solving the linear system of equations and motivate the two-grid method. The abstract multigrid method is introduced in Section 4.3, where we demonstrate the subspace decomposition for two-grid and multigrid methods. In Section 4.4, we introduce a method to create a hierarchy of multilevel meshes and then the process to create a hierarchy of nested FE spaces, and present the semi-geometric multigrid method. In Section 4.5, we discuss the variational transfer approach to compute the transfer operator in the XFEM framework. Lastly, in Section 4.6 we provide results of the numerical experiments compare the SMG method with other solution strategies and demonstrate the robustness of our multigrid method.

4.1 Linear Problem

In this section, we discuss the solution methods for the problems stemming from the unfitted finite element framework. The problems that arise from the XFEM discretization depend on the method that we used for enforcing the boundary conditions or interface conditions. When the penalty method or Nitsche's methods is employed to enforce the boundary/interface conditions, we have a problem of the following type.

$$\text{Find } u_h \in \mathcal{V}_h \text{ such that} \qquad A(u_h, v_h) = F(v_h) \qquad \forall v_h \in \mathcal{V}_h. \tag{4.1}$$

Here, the bilinear form $A(\cdot,\cdot)$ and linear form $F(\cdot)$ are abstract formulation, which may arise from either the penalty method or Nitsche's method. With a slight abuse of notation, we have dropped here the discretization parameter h as a subscript from u_h and v_h.

We define a linear operators $\mathcal{A} : \mathcal{V}_h \to \mathcal{V}_h^*$ associated with the bilinear form $A(\cdot,\cdot)$, and its adjoint $\mathcal{A}^\mathsf{T} : \mathcal{V}_h^* \to \mathcal{V}_h$ such that

$$A(u,v) = \langle \mathcal{A}u, v\rangle_{\mathcal{V}_h^* \times \mathcal{V}_h} = \langle u, \mathcal{A}^\mathsf{T} v\rangle_{\mathcal{V}_h \times \mathcal{V}_h^*} \qquad \forall u, v \in \mathcal{V}_h.$$

The linear operator $\mathcal{A}$ is continuous symmetric positive definite. We also denote the linear functional $F(\cdot)$ by choosing a function $f \in \mathcal{V}_h^*$ such that

$$F(v) = \langle f, v\rangle_{\mathcal{V}_h^* \times \mathcal{V}_h} \qquad \forall v \in \mathcal{V}_h.$$

We can rewrite the problem (4.1) using the operators as,

$$\text{find } u \in \mathcal{V}_h \text{ such that} \qquad \mathcal{A}u = f \qquad \text{in } \mathcal{V}_h^*. \tag{4.2}$$

4.2 Basic Iterative Methods

In this section, we describe basic iterative methods, for solving the linear problem (4.2), where the operator $\mathcal{A}$ is symmetric positive definite (SPD).

For a given initial guess $u^{(0)} \in \mathcal{V}_h$, a general iterative method for solving a linear system can be written as

$$u^{(k)} = u^{(k-1)} + \mathcal{P}(f - \mathcal{A}u^{(k-1)}) \qquad k = 1, 2, \ldots, \tag{4.3}$$

where $\mathcal{P} : \mathcal{V}_h^* \to \mathcal{V}_h$ is a linear operator. Here, we denote the k-th iterate as $u^{(k)}$. We assume the operator $\mathcal{P}$ to be an approximate inverse of operator $\mathcal{A}$. The iteration can be rewritten as

$$u^{(k)} = (\mathcal{I} - \mathcal{P}\mathcal{A})u^{(k-1)} + \mathcal{P}f \qquad k = 1, 2, \ldots,$$

where $\mathcal{I} : \mathcal{V}_h \rightarrow \mathcal{V}_h$ is the identity operator. It is easy to see that

$$u - u^{(k)} = (\mathcal{I} - \mathcal{P}\mathcal{A})(u - u^{(k-1)}) = (\mathcal{I} - \mathcal{P}\mathcal{A})^k (u - u^{(0)}).$$

This iterative method is convergent for an arbitrary initial value of $u^{(0)}$ if and only if $\rho(\mathcal{I} - \mathcal{P}\mathcal{A}) < 1$, where ρ denotes the spectral radius of the iterator.

Based on the above simple iteration scheme, we can define multiple iterative methods. The linear problem (4.2) can be written in the matrix-vector formulation as

$$\boldsymbol{A}\boldsymbol{u} = \boldsymbol{f},$$

where $\boldsymbol{A} \in \mathbb{R}^{n \times n}$, $\boldsymbol{f} \in \mathbb{R}^n$ and solution $\boldsymbol{u} \in \mathbb{R}^n$. We start with the decomposition of the matrix $\boldsymbol{A}$ as

$$\boldsymbol{A} = \boldsymbol{D} + \boldsymbol{L} + \boldsymbol{U},$$

where $\boldsymbol{D}$ is the diagonal part of $\boldsymbol{A}$, and $\boldsymbol{L}$ and $\boldsymbol{U}$ are the strict lower and upper triangular part of $\boldsymbol{A}$, respectively. Now, we can write different iterative methods:

- Richardson method:

$$\boldsymbol{u}^{(k+1)} = \boldsymbol{u}^{(k)} + (\boldsymbol{f} - \boldsymbol{A}\boldsymbol{u}^{(k)}) \quad k = 1, 2, \ldots$$

- Jacobi method:

$$\boldsymbol{u}^{(k+1)} = \boldsymbol{u}^{(k)} + \boldsymbol{D}^{-1}(\boldsymbol{f} - \boldsymbol{A}\boldsymbol{u}^{(k)}) \quad k = 1, 2, \ldots$$

- Gauss-Seidel method:

$$\boldsymbol{u}^{(k+1)} = \boldsymbol{u}^{(k)} + (\boldsymbol{D} + \boldsymbol{L})^{-1}(\boldsymbol{f} - \boldsymbol{A}\boldsymbol{u}^{(k)}), \quad k = 1, 2, \ldots$$

- symmetric Gauss-Seidel method:

$$\boldsymbol{u}^{(k+1)} = \boldsymbol{u}^{(k)} + (\boldsymbol{D} + \boldsymbol{L})^{-1}\boldsymbol{D}(\boldsymbol{D} + \boldsymbol{U})^{-1}(\boldsymbol{f} - \boldsymbol{A}\boldsymbol{u}^{(k)}), \quad k = 1, 2, \ldots$$

4.2.1 Subspace Correction Methods

The subspace correction method is generalization of many linear iterative methods for solving the linear system of equations arising from the partial differential equations [Xu92, Xu01]. Due to its abstract nature, many iterative methods can be viewed as a subspace correction method, for example, stationary iterative methods, Krylov subspace methods, multigrid methods, domain decomposition methods, etc. The idea of the subspace correction method is to decompose the global problem into a number of local subproblems, and then to update the iterate by adding the solution of each local subproblems.

We consider a sequence of spaces $\mathcal{V}_1, \ldots, \mathcal{V}_n$. This sequence of spaces is defined by subspaces of the $\mathcal{V}_h$, such that each, $\mathcal{V}_i \subset \mathcal{V}_h$ for $i = 1, \ldots, n$. These subspaces are known as auxiliary spaces, and they are related to the original FE space $\mathcal{V}_h$ by a linear operator

$$\Pi_i^h : \mathcal{V}_i \to \mathcal{V}_h.$$

We assume that the original FE space $\mathcal{V}_h$ can be decomposed into the sequence of auxiliary subspaces

$$\mathcal{V}_h = \sum_{i=1}^n \Pi_i^h \mathcal{V}_i.$$

For each function $v \in \mathcal{V}_h$ there exists $v_i \in \mathcal{V}_i$, for $i = 1, \ldots, n$, such that

$$v = \sum_{i=1}^n \Pi_i^h v_i.$$

We remark that this summation does not have to be a direct sum. As the representation of v_i is not necessarily unique and the subspaces can also be decomposed with redundancy. In a similar way, we define adjoint of Π_i^h as

$$(\Pi_i^h)^\mathsf{T} : \mathcal{V}_h^* \to \mathcal{V}_i^*,$$

such that

$$\langle (\Pi_i^h)^\mathsf{T} f, v_i \rangle_{\mathcal{V}_i^* \times \mathcal{V}_i} = \langle f, \Pi_i^h v_i \rangle_{\mathcal{V}_h^* \times \mathcal{V}_h} \qquad f \in \mathcal{V}_h^*,\ v_i \in \mathcal{V}_i.$$

We assume that on each subspace $\mathcal{V}_i$ the bilinear form is given as $A_i(\cdot, \cdot)$. An equivalent linear operator on each subspace $\mathcal{V}_i$ is defined as $\mathcal{A}_i :\ \mathcal{V}_i \to \mathcal{V}_i^*$, such that

$$A_i(u_i, v_i) = \langle \mathcal{A}_i u_i, v_i \rangle_{\mathcal{V}_i^* \times \mathcal{V}_i} = \langle f_i, v_i \rangle_{\mathcal{V}_i^* \times \mathcal{V}_i} \qquad u_i, v_i \in \mathcal{V}_i.$$

For each such linear operator $\mathcal{A}_i$, there exists approximate inverse $\mathcal{P}_i$ given as

$$\mathcal{P}_i : \mathcal{V}_i^* \to \mathcal{V}_i.$$

Thus, by employing the approximate inverses in each subspace, we can rewrite (4.3) as a subspace corrections iteration,

$$u^{(k)} = u^{(k-1)} + \Pi_i^h \mathcal{P}_i (\Pi_i^h)^\mathsf{T} (f - \mathcal{A} u^{(k-1)}) \qquad k = 1, 2, \ldots \tag{4.4}$$

Algorithm 4.1: Parallel Subspace Correction Iteration

```
Input  : $u^{(0)} \in \mathcal{V}_h$
Output: $u^{(k)} \in \mathcal{V}_h$
for $k = 1, 2, \ldots$ do
    $u^{(k)} \mapsfrom u^{(k-1)} + \sum_{i=1}^{n} \Pi_i^h \mathcal{P}_i (\Pi_i^h)^{\mathsf{T}} (f - \mathcal{A} u^{(k-1)})$
```

Here, rather than computing the correction or update by employing an approximate inverse on the original function space $\mathcal{V}_h$, the correction is computed by employing a sequence of approximate inverses defined on the corresponding subspaces. The subspace correction method is a very abstract formulation as it accommodates many different decompositions of the subspaces.

The subspace correction method can be classified into two major categories, namely parallel subspace correction (PSC) method, and successive subspace correction (SSC) method. The idea of the parallel subspace correction method is to compute the corrections on each subspace independently and then to update the current iterate by adding the sum of local corrections from each subspace. This approach can be viewed as an additive Schwarz method in the domain decomposition framework.

We recall that, the FE space $\mathcal{V}_h$ is spanned by the nodal basis functions $\varphi_h^p \in \varphi_h$, for nodes $p \in \mathcal{N}_h$. The cardinality of the nodal set $\mathcal{N}_h$ is given as $n = |\mathcal{N}_h|$. One of the simplest form of the subspace decomposition can be achieved by direct splitting of the FE space $\mathcal{V}_h$ into one-dimensional subspaces spanned by the nodal basis functions, where $\mathcal{V}_i = \mathrm{span}\{\phi_h^i\}$, for $i = 1, \ldots, n$. For this specific decomposition, the PSC method is equivalent to the Jacobi method if the exact inverses are used on each subspace. If the subspace decomposition is given by the subspaces of larger sizes, and if the exact inverses are employed on each subspace then the PSC method leads to a block Jacobi method. An abstract iteration of the PSC method is given in Algorithm 4.1.

In general, the PSC method is very slow to converge, especially if the dimension of the subspaces is small. In order to improve the convergence behavior of this method, one can choose to update the corrections successively. This means, each local subspace problem is solved sequentially, and the correction in each subspace utilizes the most updated iterate. The abstract iteration of the SSC methods is given in Algorithm 4.2. When the subspace decomposition is given by the subspaces spanned by one-dimensional nodal basis functions, the SSC method is equivalent to the Gauss-Seidel method. In the context of the domain decomposition methods, the SSC method is a variant of the multiplica-

Algorithm 4.2: Successive Subspace Correction Iteration

```
Input  : $u^{(0)} \in \mathcal{V}_h$
Output: $u^{(k)} \in \mathcal{V}_h$
for $k = 1, 2, \dots$ do
    $v^{(0)} \mapsfrom u^{(k-1)}$
    for $i = 1 \dots n$ do
        $v^{(i)} \mapsfrom v^{(i-1)} + \Pi_i^h \mathcal{P}_i (\Pi_i^h)^\top (f - \mathcal{A} v^{(i-1)})$
    $u^{(k)} \mapsfrom v^{(n)}$
```

tive Schwarz method. The SSC method, as given in the Algorithm 4.2, is not symmetric. The symmetric version of the SSC method involves one step of SSC iteration in the regular ordering of subspaces and one more iteration in the reverse ordering of the subspaces. Thus, the same subspace decomposition gives rise to a symmetric Gauss-Seidel method. The symmetric variant of the Gauss-Seidel method is preferred in many cases since this variant can be employed as a preconditioner in the conjugate gradient method.

4.2.2 Extended Subspace Correction Method

As discussed in the previous section, the subspace correction methods such as the Jacobi method and the Gauss-Seidel method utilize the subspaces spanned by the nodal basis functions. The nodal basis functions are very high-frequency functions and can not represent the low-frequency contribution of the error accurately. In order to improve the convergence rate of the subspace correction methods described earlier, it is essential to extend the decomposition of subspaces in such a way that low-frequency contributions of the error are also captured. This objective is possible to achieve by introducing an additional subspace, which is spanned by the nodal basis functions with larger support. The subspace spanned by the basis functions with the larger support can be given by some FE space $\mathcal{V}_H$ with the mesh size $h < H$, where $\mathcal{V}_H$ is not necessarily a subspace of $\mathcal{V}_h$. We assume the subspace $\mathcal{V}_H$ is related to the FE space $\mathcal{V}_h$ by a linear operator $\Pi_H : \mathcal{V}_H \to \mathcal{V}_h$. Hence, an extended subspace decomposition can be given by

$$\mathcal{V}_h = \sum_{i=1}^{n} \Pi_i^h \mathcal{V}_i + \Pi_H^h \mathcal{V}_H.$$

The subspace $\mathcal{V}_H$ can also be decomposed into sequence of subspaces given as $\mathcal{V}_{n+1}, \dots, \mathcal{V}_{n+m}$. Following the same strategy as before, we define a linear operator that connects the FE space $\mathcal{V}_H$ to $\mathcal{V}_{n+1}, \ldots, \mathcal{V}_{n+m}$ by

$$\Pi_j^H : \mathcal{V}_j \to \mathcal{V}_H.$$

Thus, the extended subspace decomposition can be given as

$$\mathcal{V}_h = \sum_{i=1}^{n} \Pi_i^h \mathcal{V}_i + \Pi_H^h \sum_{j=1+n}^{m+n} \Pi_j^H \mathcal{V}_j.$$

This extended decomposition can be regarded as a variant of a two-level method. Each of these subspaces $\mathcal{V}_{n+j}$ for $j = 1, \ldots, m$ can be assumed to be spanned by the nodal basis functions defined on the space $\mathcal{V}_H$. This decomposition in the subspace correction framework can give rise to the Jacobi method or Gauss-Seidel method on the space $\mathcal{V}_H$. For the Jacobi or Gauss-Seidel method, the corrections are obtained in the local direction of the basis functions of the space $\mathcal{V}_h$, while the nodal basis functions of the space $\mathcal{V}_H$ provide additional search directions. The corrections in the local direction of the basis functions of the space $\mathcal{V}_h$ and $\mathcal{V}_H$ can be referred to as the fine level and the coarse level corrections, respectively. This framework accelerates the subspace correction method by exploiting the additional coarse level corrections. This idea can be considered as a first step towards the multilevel methods. In the multilevel framework, the FE space $\mathcal{V}_h$ is decomposed into multiple subspaces which are spanned by the basis functions with increasingly larger supports. This allows for tackling different components of the error simultaneously on each subspace.

In the next section, we discuss the multigrid method as a subspace correction method.

4.3 Abstract Multigrid Method

In the last section, we discussed the basic iterative methods for solving linear problems. It is well-known that the convergence rate of these methods deteriorates with increasing problem sizes. In order to improve the convergence rates of the basic iterative methods, we can employ the extended subspace correction method, i.e., multigrid methods.

The multigrid methods are one of the most efficient techniques for solving the linear systems of equations originating from the discretization of the partial differential equations [Hac86]. The efficiency of the multigrid methods depends heavily on the underlying multilevel hierarchy of meshes and the hierarchy of multilevel spaces. In the multigrid methods, the multilevel decomposition of the

FE space is done in such a way that the FE spaces associated with a coarse level are subspaces of a fine level FE space. This is not necessarily true for the FE spaces in the XFEM framework. In order to avoid the confusion, we limit our discussion in this section to abstract FE spaces that do not necessarily arise from the XFEM discretization.

4.3.1 Two-grid Method

The basic iterative methods such as the Richardson method, the Jacobi method, and the Gauss-Seidel method are not very robust. These methods converge rapidly in a few initial iterations, but then their convergence rates deteriorate. The Jacobi and the Gauss-Seidel method operate on the nodal basis functions, which span the FE space $\mathcal{V}_h$, locally. These methods can eliminate the high-frequency components of the error easily but they are relatively slow to eliminate the low-frequency components of the error [Hac16]. As these methods can remove the oscillatory components of the error quickly, after a few iterations only the smooth components of the error remain. This property is called smoothing property, and we refer to these iterative methods as smoothers. The idea of the two-grid method is to decompose the problem into a fine and a coarse problem. In this way, the high-frequency and the low-frequency components of the error are treated separately on different levels. The error components which are too smooth on the fine level can be tackled on the coarse level.

Let $\mathcal{Y}_h$ be an abstract FE space defined on a fine grid, and $\mathcal{Y}_H$ be a FE space defined on a coarse grid, where $\mathcal{Y}_H \subset \mathcal{Y}_h$. As another component of the two-grid method, let $\mathcal{S}_h$ be a smoother defined on the fine level, as $\mathcal{S}_h : \mathcal{Y}_h^* \to \mathcal{Y}_h$. Let Π_H^h be a linear operator that connects the coarse space with the fine space. Following the multigrid terminology, we call Π_H^h a prolongation operator and its adjoint a restriction operator. In addition, we define an exact or approximate inverse of the coarse level operator $\mathcal{A}_H$ as $\mathcal{P}_H : \mathcal{Y}_H^* \to \mathcal{Y}_H$.

An abstract iteration of the two-grid method is given in Algorithm 4.3. In general, a single iteration of the smoother may not be sufficient, hence in many cases, it becomes essential to perform multiple smoothing iterations. We denote the number of smoothing steps performed before and after the coarse level correction, by parameters $\nu_1, \nu_2 > 0$ respectively.

4.3.2 Multigrid Method

In many practical applications, the coarse level discretization in the two-grid method might be too large to be solved quickly. In such cases, the multigrid

```
Algorithm 4.3: Two-grid Method
    Input  : $u^{(0)} \in \mathcal{Y}_h$
    Output: $u^{(k)} \in \mathcal{Y}_h$
    for $k = 1, 2, \dots$ do
        $v \mapsfrom u^{(k-1)} + \mathcal{S}_h^{\nu_1}(f - \mathcal{A}u^{(k-1)})$ ;        ▷ $\nu_1$ pre-smoothing steps
        $v \mapsfrom v + \Pi_H^h \mathcal{P}_H (\Pi_H^h)^\mathsf{T}(f - \mathcal{A}v)$ ;      ▷ coarse level correction
        $u^{(k)} \mapsfrom v + \mathcal{S}_h^{\nu_2}(f - \mathcal{A}v)$ ;            ▷ $\nu_2$ post-smoothing steps
```

methods can be viewed as a natural extension of the two-grid methods, where the two-grid algorithm is called recursively until the problem on the coarsest level is computationally inexpensive to solve.

To obtain the multigrid method, we assume an abstract FE space $\mathcal{Y}_h$ is decomposed into L levels, such that

$$\mathcal{Y}_h = \mathcal{Y}_L + \sum_{\ell=1}^{L-1} \Pi_\ell^{\ell+1} \mathcal{Y}_\ell + \Pi_0^1 \mathcal{Y}_0,$$

where $\mathcal{Y}_{\ell-1} \subset \mathcal{Y}_\ell$ and $\Pi_{\ell-1}^\ell : \mathcal{Y}_{\ell-1} \to \mathcal{Y}_\ell$ are associated linear operators for all $\ell = 1, \dots, L$. This subspace decomposition consists of a lot of redundancies, but the redundant subspaces are quite essential for achieving the optimal convergence of the multigrid methods. The hierarchy of nested FE spaces $(\mathcal{Y}_\ell)_{\ell=0,\dots,L}$ is constructed by the discretization on the hierarchy of nested meshes. Also, let $\mathcal{S}_\ell : \mathcal{Y}_\ell^* \to \mathcal{Y}_\ell$, for $\ell = 1, \dots, L$ be smoothers, that damp oscillatory components of the error on a given level. At each level, $\ell > 0$, the smoothing operator, associated with space $\mathcal{Y}_\ell$, reduces the high-frequency contributions of the error with respect to the mesh size. We assume that the remaining error may be approximated well enough in the subspace $\mathcal{Y}_{\ell-1}$, as it can be resolved on a coarser mesh. On the coarsest level, we employ an exact inverse of the coarse level operator which can get rid of the low-frequency components of the error that remain after the smoothing iterations.

The subspace decomposition in the multigrid framework can be given as

$$\mathcal{Y}_h = \sum_{i=1}^{n_L} \Pi_i^L \mathcal{Y}_i + \sum_{\ell=1}^{L-1} \Pi_\ell^{\ell+1} \sum_{j=1}^{n_\ell} \Pi_j^\ell \mathcal{Y}_j + \Pi_0^1 \mathcal{Y}_0.$$

Here, except on the coarsest level, the subspaces on all other levels are again decomposed in one dimensional subspaces, as we use the Jacobi, the Gauss-Seidel or the symmetric Gauss-Seidel methods as smoothers. In many practical applications, specifically, if the multigrid method is chosen as a preconditioner for a

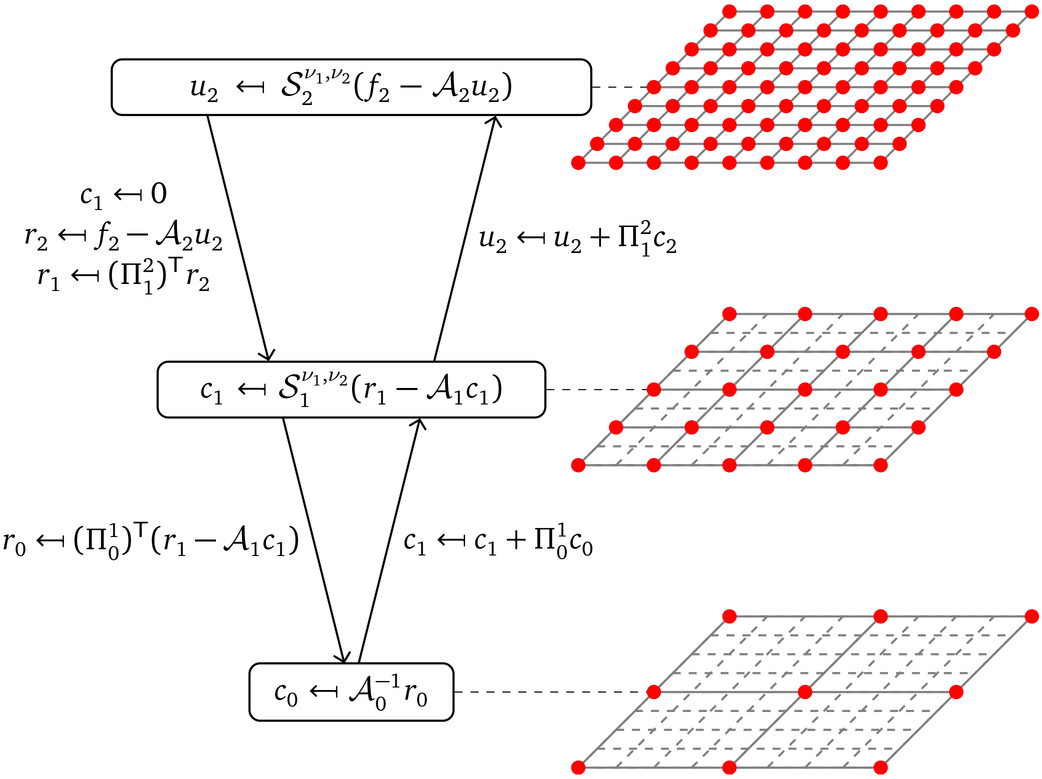

$u_2 \leftarrow \mathcal{S}_2^{\nu_1,\nu_2}(f_2 - \mathcal{A}_2 u_2)$
$c_1 \leftarrow 0$
$r_2 \leftarrow f_2 - \mathcal{A}_2 u_2$
$r_1 \leftarrow (\Pi_1^2)^{\mathsf{T}} r_2$
$u_2 \leftarrow u_2 + \Pi_1^2 c_2$
$c_1 \leftarrow \mathcal{S}_1^{\nu_1,\nu_2}(r_1 - \mathcal{A}_1 c_1)$
$r_0 \leftarrow (\Pi_0^1)^{\mathsf{T}}(r_1 - \mathcal{A}_1 c_1)$
$c_1 \leftarrow c_1 + \Pi_0^1 c_0$
$c_0 \leftarrow \mathcal{A}_0^{-1} r_0$

Figure 4.1. A multigrid $V(\nu_1, \nu_2)$-cycle, where ν_1 and ν_2 denote number of pre-smoothing and post-smoothing steps. On the right, we can see a hierarchy of meshes associated with each level.

Krylov subspace method, it is necessary to choose symmetric smoothing operators. In Figure 4.1, we can see an example of an abstract $V(\nu_1, \nu_2)$-cycle. A multigrid method is a combination of the smoothing iterations and the coarse level corrections. The coarse level corrections are heavily dependent on well-chosen transfer operators which can be used to restrict residuals from a fine level to a coarse level and prolongate corrections from a coarse level to a fine level. In order to improve the convergence of the multigrid methods, we can design more suitable smoothers and/or the transfer operators for a given problem. In the geometric multigrid methods, the standard interpolation operator is usually employed as the prolongation operator, while the transpose of the interpolation operator is chosen as the restriction operator. For the smoother, we could choose any of the basic iterative methods. We note that the Richardson method and the Jacobi methods are not necessarily convergent, hence weighted Richardson or weighted Jacobi methods should be used as smoother. While the Gauss-Seidel and the symmetric Gauss-Seidel method are always convergent if

$\boldsymbol{A}$ is diagonally dominant and symmetric positive definite, and traditionally the weighted counterparts of these methods are rarely used as smoothers.

In the traditional FE framework, the subspaces used in the multigrid hierarchy are chosen in such a way that the hierarchy of the spaces is nested. The FE spaces associated with the meshes would also be nested if the hierarchy of the meshes is obtained by uniform refinement or coarsening strategy. This is a convenient way to create a hierarchy of nested FE spaces, but unfortunately in the XFEM framework even if we have a nested mesh hierarchy of the background mesh, we have a non-nested mesh hierarchy for each subdomain due to the arbitrary location of interfaces. In the next section, we discuss a framework for creating nested FE spaces in the XFEM framework and the way to utilize this framework in a multigrid method.

4.4 Multilevel method for XFEM Discretization

In this section, we discuss a method for creating a multilevel hierarchy of the meshes and the corresponding hierarchy of FE spaces. The approach for creating a hierarchy of the nested FE spaces from the non-nested meshes was first proposed as an auxiliary subspace correction method [Xu96]. Later, we describe the semi- geometric multigrid (SMG) method for the XFEM discretization.

4.4.1 Multilevel Space Hierarchy

In this section, we provide a framework for creating a hierarchy of nested XFEM spaces from the hierarchy of background meshes. We carry out the discussion for a generic case, for both the fictitious domain method and the interface problems. For the interface problems, we include cases with multiple interfaces.

Let us define a mesh hierarchy of background meshes for levels, ℓ, where $\ell \in \{0, \dots, L\}$. The coarsest level is denoted by $\ell = 0$ and the finest level is denoted by $\ell = L$. The original background mesh is considered to be the mesh on the finest level, i.e., $\widetilde{\mathcal{T}}_L := \widetilde{\mathcal{T}}_h$. We define the coarse level mesh hierarchy as, $\widetilde{\mathcal{T}}_\ell$, for all $\ell \in \{0, \dots, L-1\}$. It is necessary that, the domain Ω is encapsulated by the background mesh on each level ℓ, $\Omega \subset (\widetilde{\mathcal{T}}_\ell)_{\ell=0,\dots,L-1}$. Now, on this mesh hierarchy, we define finite element spaces associated with each of these background meshes as

$$\widetilde{\mathcal{V}}_\ell = \{v \in H^1(\widetilde{\mathcal{T}}_\ell) : v|_K \in \mathcal{P}_1,\ v|_{\partial \widetilde{\mathcal{T}}_\ell} = 0,\ \forall K \in \widetilde{\mathcal{T}}_\ell\} \quad \forall \ell \in \{0, \dots, L-1\}.$$

On the finest level, the definition of the finite element space is taken directly from the original problem, i.e., $\widetilde{\mathcal{V}}_L := \widetilde{\mathcal{V}}_h$. When the hierarchy of the meshes is nested, the hierarchy of the FE spaces associated with these meshes is also nested, thus

$$\widetilde{\mathcal{V}}_{\ell-1} \subset \widetilde{\mathcal{V}}_\ell \qquad \forall \ell \in \{1, \dots, L\}.$$

In the XFEM discretization, the background mesh is enriched, decomposed and then it is associated with each subdomain $i = 1, .., r$, where r is the number of total subdomains. In the multigrid framework this procedure is carried out on each level thus

$$\mathcal{T}_{\ell,i} = \{K \in \widetilde{\mathcal{T}}_\ell : K \cap \Omega_i \neq \emptyset\} \qquad \forall \ell \in \{0, \dots, L-1\},\ \forall i \in \{1, \dots, r\}.$$

If possible, the mesh hierarchy can be obtained by uniform coarsening of the original background mesh used for the discretization on the finest level.

Now, we define the characteristic function of each subdomain Ω_i, for $i = 1, \dots, r$

$$\chi_{\Omega_i} : \mathbb{R}^d \to \mathbb{R}, \qquad \chi_{\Omega_i}(\boldsymbol{x}) = \begin{cases} 1 & \forall \boldsymbol{x} \in \overline{\Omega}_i, \\ 0 & \text{otherwise.} \end{cases}$$

We exploit the definition of the characteristic function to restrict the finite element space $(\widetilde{\mathcal{V}}_\ell)_{\ell=0,\dots,L-1}$ to each subdomain Ω_i, as

$$\mathcal{V}_{\ell,i} = \chi_{\Omega_i}(\boldsymbol{x}) \widetilde{\mathcal{V}}_\ell \qquad \forall \ell \in \{0, \dots, L-1\},\ \forall i \in \{1, \dots, r\}.$$

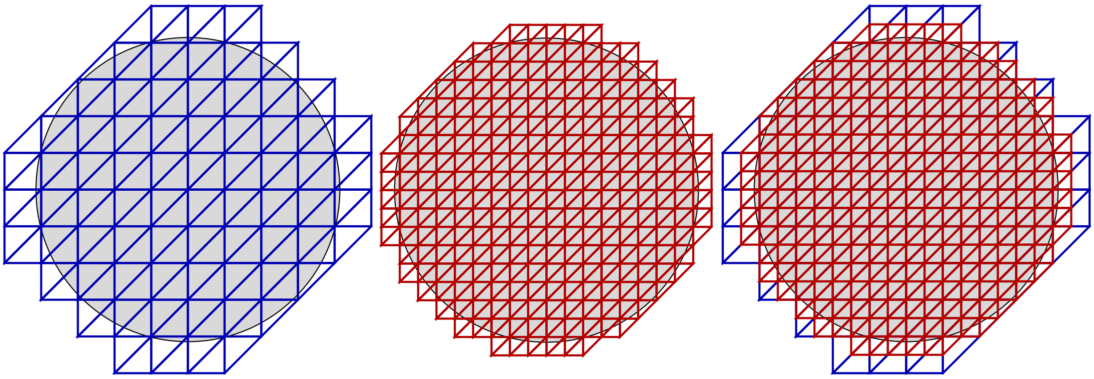
Similar to the finest level mesh, we borrow the definition of FE space from the finest level on each subdomain, i.e., $\mathcal{V}_{L,i} := \mathcal{V}_{h,i}$. In the XFEM framework, even if we have a nested mesh hierarchy of the background mesh, due to the arbitrary location of interfaces, we could have a non-nested mesh hierarchy for each subdomain, see an example in Figure 4.2. There, the meshes associated with the domain Ω_i at different levels are not nested. As the meshes are not nested, the finite element spaces associated with the meshes are also not nested, $\mathcal{V}_{\ell-1,i} \not\subset \mathcal{V}_{\ell,i}$.

In this work, we want to create the hierarchy of nested FE spaces associated with each subdomain. We define the prolongation operators which connects the spaces, $(\mathcal{V}_{\ell,i})_{\ell=0,\dots,L}$ as

$$\Pi^\ell_{\ell-1,i} : \mathcal{V}_{\ell-1,i} \to \mathcal{V}_{\ell,i} \qquad \forall \ell \in \{1, \dots, L\},\ \forall i \in \{1, \dots, r\}, \tag{4.5}$$

such that $\Pi^\ell_{\ell-1,i} \mathcal{V}_{\ell-1,i} \subset \mathcal{V}_{\ell,i}$. Now using this prolongation operator, we can construct a FE space associated with mesh $\mathcal{T}_{\ell,i}$ by composition of the prolongation operators,

$$\mathcal{X}_{\ell,i} := \Pi^L_{L-1,i} \cdots \Pi^{\ell+1}_{\ell,i} \mathcal{V}_{\ell,i} \qquad \forall \ell \in \{1, \dots, L-1\},\ \forall i \in \{1, \dots, r\}.$$



(a) Triangular mesh $\mathcal{T}_{\ell-1,i}$ (b) Triangular mesh $\mathcal{T}_{\ell,i}$ (c) Superimposition of $\mathcal{T}_{\ell,i}$ on $\mathcal{T}_{\ell-1,i}$

Figure 4.2. 2D Triangular meshes on different levels encapsulating the domain Ω_i, (domain Ω_i is shaded in gray).

We define $\mathcal{X}_{L,i} := \mathcal{V}_{L,i}$ on the finest level, as the FE space on the finest level is not modified. We can create the hierarchy of the nested spaces for $i = 1, \ldots, r$ by recursive application of the prolongation operators as

$$\underbrace{(\Pi^L_{L-1,i} \cdots \Pi^{\ell+1}_{\ell,i})(\Pi^{\ell}_{\ell-1,i}\mathcal{V}_{\ell-1,i})}_{=:\mathcal{X}_{\ell-1,i}} \subset \underbrace{(\Pi^L_{L-1,i} \cdots \Pi^{\ell+1}_{\ell,i})(\mathcal{V}_{\ell,i})}_{=:\mathcal{X}_{\ell,i}} \quad \forall \ell \in \{1, \ldots, L-1\}.$$

Thus, the composition of the prolongation operators applied on the finest level generates a nested hierarchy of the FE spaces, i.e.,

$$\mathcal{X}_{0,i} \subset \mathcal{X}_{1,i} \subset \cdots \subset \mathcal{X}_{\ell-1,i} \subset \mathcal{X}_{\ell,i} \subset \cdots \subset \mathcal{X}_{L-1,i} \subset \mathcal{X}_{L,i} \qquad \forall i \in \{1, \ldots, r\}.$$

For simplicity and compactness, we define the prolongation operators for the whole domain as a direct sum of the prolongation operator defined for each subdomain, i.e.,

$$\Pi^{\ell}_{\ell-1} := \bigoplus_{i=1}^{r} \Pi^{\ell}_{\ell-1,i} \qquad \forall i \in \{1, \ldots, r\}.$$

This prolongation operator $\Pi^{\ell}_{\ell-1}$ inherits the same properties from its counterparts defined on each subdomain. A hierarchy of nested spaces for the whole domain can be generated with the same procedure on the enriched FE space, i.e.,

$$\mathcal{X}_{\ell} = \Pi^{\ell}_{\ell-1} \underbrace{\bigoplus_{i=1}^{r} \mathcal{V}_{\ell,i}}_{=:\mathcal{V}_{\ell}} = \underbrace{\bigoplus_{i=1}^{r} (\Pi^L_{L-1,i} \cdots \Pi^{\ell+1}_{\ell,i} \mathcal{V}_{\ell,i})}_{=\oplus_{i=1}^{r} \mathcal{X}_{\ell,i}} \quad \forall \ell \in \{0, \ldots, L-1\}.$$

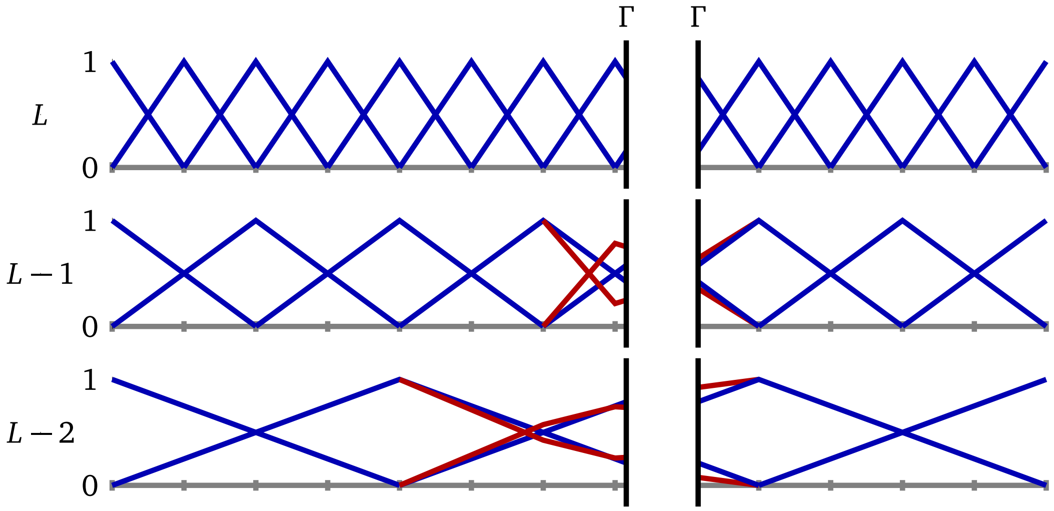


Figure 4.3. Basis of the cut domain in 1D, in blue we see the original basis $\mathcal{V}_{L,i} \not\supset \mathcal{V}_{L-1,i} \not\supset \mathcal{V}_{L-2,i}$, in red we see the basis computed with L^2-projections which are nested, $\mathcal{X}_{L,i} \supset \mathcal{X}_{L-1,i} \supset \mathcal{X}_{L-2,i}$.

Thus, we can create the sequence of nested spaces $(\mathcal{X}_\ell)_{\ell=0,\dots,L}$. In Figure 4.3, we see how the basis functions created by the nested FE spaces differ from the non-nested FE spaces.

By construction, it is clear that the transfer operator for the XFEM discretization treats each subdomain separately. This can be regarded as an additive subspace splitting strategy to compute the transfer operator. Intuitively, it can also be seen as all different transfer operators assembled on each subdomain, where the support of the FE space is restricted to the respective subdomain. The global transfer operator is computed as a direct sum of all such transfer operators on each subdomain. This ensures that the information transfer between levels is restricted to each subdomain, and there is no cross-information transfer across the interfaces.

Let $\{\zeta_\ell^p\}_{p\in\mathcal{N}_\ell}$ be the basis of a FE space $\mathcal{X}_\ell$, where $\mathcal{N}_\ell$ denotes the set of nodes associated with the mesh $\mathcal{T}_\ell$. Also, we define the cardinality of the nodal set $\mathcal{N}_\ell$ as, $n_\ell = |\mathcal{N}_\ell|$. Using the nodal basis of the finite element spaces $\mathcal{X}_\ell$ and $\mathcal{X}_{\ell-1}$, we can compute the matrix representation of the transfer operator $\Pi_{\ell-1}^{\ell}$. We denote the matrix formulation of the transfer operator $\Pi_{\ell-1}^{\ell}$ as $\boldsymbol{T}_{\ell-1}^{\ell} \in \mathbb{R}^{n_\ell \times n_{\ell-1}}$. In the next section, we discuss some approaches to compute the transfer operator algebraically.

This transfer operator can be used to compute the basis functions associated with all coarse spaces $(\mathcal{X}_\ell)_{\ell=0,\ldots,\ell-1}$ as

$$\zeta^q_{\ell-1} := \sum_{p\in\mathcal{N}_\ell} (\boldsymbol{T}^\ell_{\ell-1})_{pq}\zeta^p_\ell \qquad \forall q \in \mathcal{N}_{\ell-1}, \forall \ell \in \{1,\ldots,L\}. \tag{4.6}$$

Thus, the basis functions of a coarse FE space can be computed recursively as a linear combination of the basis functions of a fine level FE space. For the finest level, as we have mentioned earlier the definition of the FE space is kept untouched, thus the basis functions on the finest levels are defined as $\zeta^p_L := \phi^p_h$, for all $p \in \mathcal{N}_h$. In Figure 4.2, we can observe how the basis function associated with the space $\mathcal{V}_\ell$ and $\mathcal{X}_\ell$ for all $\ell = 0,\ldots,L$ differ. From Figure 4.2, we can see that the basis functions of the coarse levels are piecewise linear with respect to the mesh associated with the finest level, provided the basis functions are linear also on the finest level.

4.4.2 Semi-geometric Multigrid Method

Now, we have created a hierarchy of nested FE spaces $\mathcal{X}_\ell$ from the hierarchy of non-nested FE spaces $\mathcal{V}_\ell$. This is quite essential for ensuring the robustness of the multigrid method.

Considering the abstract weak formulation (4.1) given for either Nitsche's method or the penalty method, we recall the abstract problem,

$$\text{find } u \in \mathcal{V}_L = \mathcal{X}_L \text{ such that } \quad A(u,v) = F(v) \qquad \forall v \in \mathcal{V}_L = \mathcal{X}_L.$$

We denote the algebraic representation of the bilinear form as $\boldsymbol{A}_L$, also known as a stiffness matrix, with the entries $(A_L)_{pq} := A(\zeta^q_L, \zeta^p_L)$ for all $p,q \in \mathcal{N}_L$. The right hand side is represented by $\boldsymbol{f}_L$, with the local entries $(f_L)_p := F(\zeta^p_L)$, for all $p \in \mathcal{N}_L$. We can write the algebraic variant of our problem as

$$\boldsymbol{A}_L \boldsymbol{u}_L = \boldsymbol{f}_L,$$

where unknown $\boldsymbol{u}_L$ represents a vector of the coefficients $\boldsymbol{u}_L = (u_L)_p$ for all $p \in \mathcal{N}_L$, which are associated with the finite element approximation space $\mathcal{X}_L$.

From a practical point of view, we need a preliminary step before the multigrid algorithm can be invoked. In the preparation step, we compute a sequence of prolongation operators $(\boldsymbol{T}^\ell_{\ell-1})_{\ell=1,\ldots,L}$ between successive levels of the FE space hierarchy. These prolongation operators are used to transfer a coarse level correction to a fine level and the adjoint of these operators is used as restriction operators to transfer the residual from a fine level to a coarse level. The next stage of the setup consists of computation of the coarse level stiffness matrices,

Algorithm 4.4: Setup Semi-geometric Multigrid algorithm

```
Input  : $\boldsymbol{A}_L$, $(\mathcal{T}_\ell)_{\ell=0,\dots,L}$
Output: $(\boldsymbol{T}^\ell_{\ell-1})_{\ell=1,\dots,L}$, $(\boldsymbol{A}_\ell)_{\ell=0,\dots,L-1}$
Function: Setup SMG
for $\ell \mapsfrom L, \dots, 1$ do
    $\boldsymbol{T}^\ell_{\ell-1} \mapsfrom (\mathcal{T}_\ell, \mathcal{T}_{\ell-1})$ ;          ▷ assemble prolongation operator
    $\boldsymbol{A}_{\ell-1} \mapsfrom (\boldsymbol{T}^\ell_{\ell-1})^\top \boldsymbol{A}_\ell \boldsymbol{T}^\ell_{\ell-1}$ ;          ▷ coarse level assembly
```

$(\boldsymbol{A}_\ell)_{\ell=0,\dots,L-1}$. We use the Galerkin assembly approach to compute the coarse level stiffness matrices, defined as

$$\boldsymbol{A}_{\ell-1} := (\boldsymbol{T}^\ell_{\ell-1})^\top \boldsymbol{A}_\ell \boldsymbol{T}^\ell_{\ell-1} \qquad \forall \ell \in \{1, \dots, L\}. \tag{4.7}$$

Here, as the finest level stiffness matrix $\boldsymbol{A}_L$ is symmetric positive definite, and the transfer operator $\boldsymbol{T}^\ell_{\ell-1}$ has full rank then for all $\ell = 0, \dots, L-1$ the coarse level matrices are also symmetric positive definite. The Galerkin assembly is quite essential in the semi-geometric multigrid algorithm. If the assembly of the stiffness matrix is done on each level, the coarse level operator would be constructed using the FE spaces $(\mathcal{V}_\ell)_{\ell=0,\dots,L-1}$. In contrast, using the Galerkin assembly assures the stiffness matrix is recursively constructed in the nested FE spaces $(\mathcal{X}_\ell)_{\ell=0,\dots,L-1}$.

After the computation of the coarse level stiffness matrices and the prolongation operators, we invoke the multigrid algorithm. The multigrid iterations can be used for preconditioning or solving a linear system. Therefore, the SMG can be written in an abstract way, such that it returns the correction $\boldsymbol{c}_L$ rather than the iterate explicitly. The residual on the finest level is given as

$$\boldsymbol{r}_L = \boldsymbol{f}_L - \boldsymbol{A}_L \boldsymbol{u}_L.$$

The semi-geometric multigrid algorithm is given in Algorithm 4.5, where ν_1, ν_2 are the number of pre-smoothing and post-smoothing steps, respectively. For $\gamma = 1$ and $\gamma = 2$, the multigrid method in Algorithm 4.5 can be transformed to the $V(\nu_1, \nu_2)$-cycle and the $W(\nu_1, \nu_2)$-cycle, respectively.

Until now, we have only discussed the transfer operators in the abstract sense. In the next section, we provide the framework for computing the transfer operators algebraically through variational transfer.

Algorithm 4.5: Semi-geometric Multigrid algorithm

```
Input  : $(\boldsymbol{A}_\ell)_{\ell=0,\dots,L}$, $\boldsymbol{r}_L$, $L$, $\nu_1$, $\nu_2$, $(\boldsymbol{T}^\ell_{\ell-1})_{\ell=1,\dots,L}$, $\gamma$
Output: $\boldsymbol{c}_L$
Function: SMG($\boldsymbol{A}_\ell$, $\boldsymbol{r}_\ell$, $\ell$, $\nu_1$, $\nu_2$, $\boldsymbol{T}^\ell_{\ell-1}$, $\gamma$):
if $\ell \neq 0$ then
    $\boldsymbol{c}_\ell \mapsfrom \boldsymbol{0}$ ;                                  ▷ initialize correction
    $\boldsymbol{c}_\ell \mapsfrom$ Smoother($\boldsymbol{A}_\ell, \boldsymbol{c}_\ell, \boldsymbol{r}_\ell, \nu_1$) ;          ▷ $\nu_1$ pre-smoothing steps
    $\boldsymbol{r}_{\ell-1} \mapsfrom (\boldsymbol{T}^\ell_{\ell-1})^\top(\boldsymbol{r}_\ell - \boldsymbol{A}_\ell \boldsymbol{c}_\ell)$ ;            ▷ restriction
    $\boldsymbol{c}_{\ell-1} \mapsfrom \boldsymbol{0}$ ;                       ▷ initialize coarse level correction
    for $i = 1, \dots, \gamma$ do
        $\boldsymbol{c}_{\ell-1} \mapsfrom \boldsymbol{c}_{\ell-1}+$ SMG($\boldsymbol{A}_{\ell-1}$, $\boldsymbol{r}_{\ell-1}$, $\ell-1$, $\nu_1$, $\nu_2$, $\boldsymbol{T}^{\ell-1}_{\ell-2}$,$\gamma$) ;   ▷ coarse cycle
    $\boldsymbol{c}_\ell \mapsfrom \boldsymbol{c}_\ell + \boldsymbol{T}^\ell_{\ell-1}\ \boldsymbol{c}_{\ell-1}$ ;                  ▷ prolongation
    $\boldsymbol{c}_\ell \mapsfrom$ Smoother($\boldsymbol{A}_\ell, \boldsymbol{c}_\ell, \boldsymbol{r}_\ell, \nu_2$) ;         ▷ $\nu_2$ post-smoothing steps
else
    $\boldsymbol{c}_0 \mapsfrom \boldsymbol{A}_0^{-1} \boldsymbol{r}_0$ ;                           ▷ direct solver
```

4.5 Variational Transfer

In this section, we discuss the computation of the transfer operator for the XFEM framework. Following the multigrid terminology, we refer to the transfer operator as a prolongation operator while to the adjoint of the transfer operator as a restriction operator.

In the context of non-conforming domain decomposition methods and contact problems, the information transfer between non-conforming meshes is realized through global L^2-projections [Bel99, Woh00a]. The mortar methods were introduced to couple different discretizations on the interfaces of subdomains, where the meshes of the subdomains do not necessarily match at the interface. In the mortar method, the L^2-projections are performed to couple these discretizations on the trace spaces defined on the boundaries. We exploit the same strategy to couple different FE spaces defined on a fine and a coarse level, i.e., we create the transfer operators between the successive meshes in the multilevel hierarchy.

In the theory of the mortar method, a mortar side and a non-mortar side are chosen such that the transfer operator maps a function from a mortar side to a non-mortar side. Thus, a mortar side and a non-mortar side can be regarded as a domain and an image of a transfer operator. In the multigrid framework, we associate a fine space with the non-mortar side, while a coarse space is associated with the mortar side, as we aim to compute the prolongation operator from a coarse space to a fine space. The transfer operators between connected spaces $(\mathcal{V}_\ell)_{\ell=0,\ldots,L}$ in our mesh hierarchy are defined as in (4.5). For example, the operator $\Pi^\ell_{\ell-1}$ projects the element $v \in \mathcal{V}_{\ell-1}$ into $\mathcal{V}_\ell$, given as $\Pi^\ell_{\ell-1} v := w$, where $w \in \mathcal{V}_\ell$.

We define the prolongation operator, $\Pi^\ell_{\ell-1} : \mathcal{V}_{\ell-1} \to \mathcal{V}_\ell$ for all $\ell = 1, \ldots, L$, by enforcing the following condition $\Pi^\ell_{\ell-1} v = w$, in weak sense using the method of Lagrange multipliers. Thus, the weak formulation of this problem can be given as,

$$\text{find } \Pi^\ell_{\ell-1} v \in \mathcal{V}_\ell \text{ such that } \quad (\Pi^\ell_{\ell-1} v, \mu)_{L^2(\Omega)} = (w, \mu)_{L^2(\Omega)} \quad \forall \mu \in \mathcal{M}_\ell, \tag{4.8}$$

where $\mathcal{M}_\ell$ is a space of Lagrange multipliers defined on a finer level. Thus, the dimension of spaces $\mathcal{M}_\ell$ and $\mathcal{V}_\ell$ are equal, as they both are defined on the same mesh $\mathcal{T}_\ell$. Reformulating (4.8), we get weak equality condition on the domain Ω,

$$(\Pi^\ell_{\ell-1} v - w, \mu)_{L^2(\Omega)} = 0 \qquad \forall \mu \in \mathcal{M}_\ell.$$

Let $\{\phi^j_{\ell-1}\}_{j \in \mathcal{N}_{\ell-1}}$ be a basis of $\mathcal{V}_{\ell-1}$, $\{\phi^k_\ell\}_{k \in \mathcal{N}_\ell}$ be a basis of $\mathcal{V}_\ell$ and $\{\theta^i_\ell\}_{i \in \mathcal{N}_\mu}$ be basis of the multiplier space $\mathcal{M}_\ell$, where $\mathcal{N}_{\ell-1}$, $\mathcal{N}_\ell$ and $\mathcal{N}_\mu$ denote the set of nodes associated with respective FE space.

Writing the functions $v \in \mathcal{V}_{\ell-1}$ and $w \in \mathcal{V}_\ell$ as a linear combination of the basis functions, we get $v = \sum_{j \in \mathcal{N}_{\ell-1}} v_j \phi^j_{\ell-1}$ and $w = \sum_{k \in \mathcal{N}_\ell} w_k \phi^k_\ell$, with coefficients $\{v_j\}_{j \in \mathcal{N}_{\ell-1}}$ and $\{w_k\}_{k \in \mathcal{N}_\ell}$. Now, inserting the respective basis function in (4.8), we get

$$\sum_{j \in \mathcal{N}_{\ell-1}} v_j (\phi^j_{\ell-1}, \theta^i_\ell)_{L^2(\Omega)} = \sum_{k \in \mathcal{N}_\ell} w_k (\phi^k_\ell, \theta^i_\ell)_{L^2(\Omega)} \qquad \forall i \in \mathcal{N}_\mu. \tag{4.9}$$

The formulation (4.9) in matrix-vector form is given as

$$\boldsymbol{N}\boldsymbol{v} = \boldsymbol{M}\boldsymbol{w}.$$

Here, the matrix $\boldsymbol{N}$ is defined between a fine and a coarse space, with entries $N_{ij} = (\phi^j_{\ell-1}, \theta^i_\ell)_{L^2(\Omega)}$ and $\boldsymbol{M}$ matrix is defined on a fine level, with entries $M_{ik} = (\phi^k_\ell, \theta^i_\ell)_{L^2(\Omega)}$. As Lagrange multiplier space and the finite element space on fine level ℓ are of the same dimension, $\boldsymbol{M} \in \mathbb{R}^{n_\ell \times n_\mu}$ is a square matrix, as $n_\ell = n_\mu$, where $n_\ell = |\mathcal{N}_\ell|$ and $n_\mu = |\mathcal{N}_\mu|$. Whereas, $\boldsymbol{N} \in \mathbb{R}^{n_\ell \times n_{\ell-1}}$ is a rectangular matrix, here $n_{\ell-1} = |\mathcal{N}_{\ell-1}|$ denotes the dimension of FE space $\mathcal{V}_{\ell-1}$. The vectors $\boldsymbol{v}$ and $\boldsymbol{w}$ are representation of function v, w on level $\ell-1$ and ℓ, respectively. The formula for computing the transfer operator can be expressed algebraically as

$$\boldsymbol{w} = \boldsymbol{M}^{-1}\boldsymbol{N}\boldsymbol{v} = \boldsymbol{T}\boldsymbol{v}. \tag{4.10}$$

The matrix $\boldsymbol{T} \in \mathbb{R}^{n_\ell \times n_{\ell-1}}$ is the discrete representation of the transfer operator $\Pi^\ell_{\ell-1}$, which we use as a prolongation operator in the multigrid method.

4.5.1 L^2-projections

The choice of different Lagrange multiplier spaces in the formulation (4.8) can lead to different transfer operators. The Lagrange multiplier space can be taken as the finite element space $\mathcal{M}_\ell := \mathcal{V}_\ell$. Thus, we take the same basis functions $\{\phi_\ell^i\}_{i\in\mathcal{N}_\mu}$ for $\mathcal{M}_\ell$ and $\mathcal{V}_\ell$. In this particular case, the scaled mass matrix $\boldsymbol{N}$, between a coarse and a fine space, has the entries

$$N_{ij} = (\phi_{\ell-1}^j, \phi_\ell^i)_{L^2(\Omega)} \qquad \forall i \in \mathcal{N}_\ell, \forall j \in \mathcal{N}_{\ell-1}.$$

The matrix $\boldsymbol{M}$ is the mass matrix on a fine level with the entries

$$M_{ik} = (\phi_\ell^k, \phi_\ell^i)_{L^2(\Omega)} \qquad \forall i, k \ \in \mathcal{N}_\ell.$$

The usage of the transfer operator computed with the L^2-projection does not guarantee a computationally efficient multigrid algorithm. In the multigrid algorithm, we employ the Galerkin assembly approach (4.7) to create coarse level operators. Hence, it is necessary to compute the transfer operator $\boldsymbol{T}$, which requires an inverse of a sparse block diagonal mass matrix $\boldsymbol{M}$. As the inverse of matrix $\boldsymbol{M}$ is dense, the resulting transfer operator computed using (4.10) produces a dense transfer operator. If the dense transfer operator is used to construct the coarse level basis functions using (4.6), the coarse level basis functions would have large supports. Due to this reason, the coarse level stiffness matrices $(\boldsymbol{A}_\ell)_{\ell=0,\dots,L-1}$ computed using Galerkin projection also become dense. If this transfer operator is employed within the multigrid method, it impedes the performance of the overall algorithm, as all the matrix-vector multiplications have to be performed on the dense systems.

4.5.2 Pseudo-L^2-projections

In order to reduce the computational cost of the application of the transfer operator, we require the transfer operator $\boldsymbol{T}$ to be sparse. This can be achieved by choosing a different definition of Lagrange multiplier space [Woh00a, DK11, DK14]. The basis functions of this multiplier space are chosen in such a way that they are biorthogonal to the standard Lagrange FE basis with respect to L^2-inner product.

We define the dual space, $\mathcal{M}_\ell := \text{span}\{\psi_\ell^i\}_{i\in\mathcal{N}_\mu}$, where ψ_ℓ^i are defined as the dual functions which satisfy the following biorthogonality condition

$$(\phi_\ell^q, \psi_\ell^p)_{L^2(\Omega)} = \delta_{pq}(\phi_\ell^p, \mathbb{1})_{L^2(\Omega)} \qquad \forall p, q \in \mathcal{N}_\ell,$$

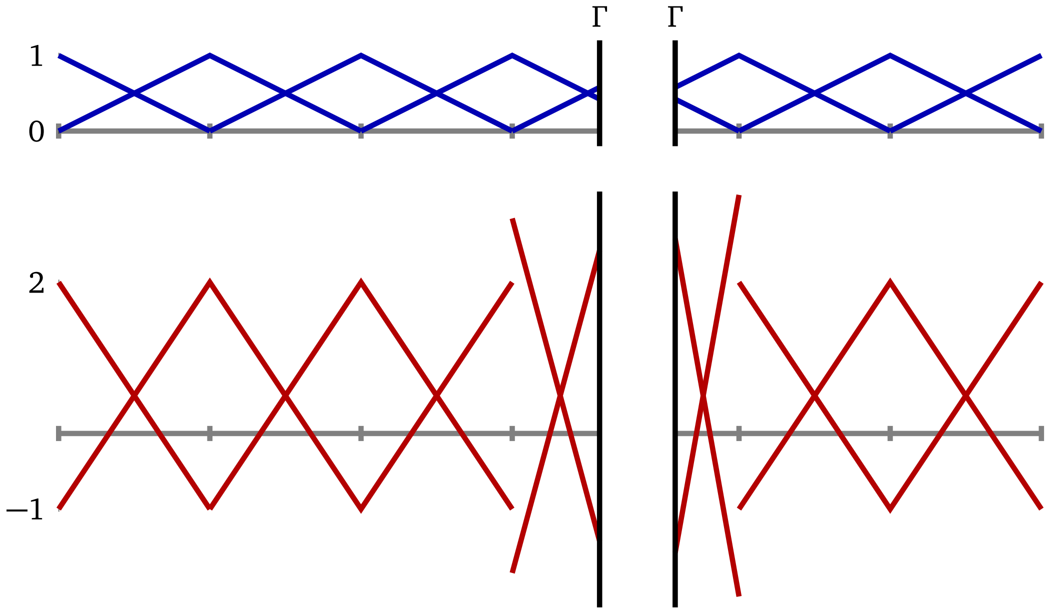


Figure 4.4. Lagrange basis functions (in blue) for the cut domain and corresponding biorthogonal basis functions (in red), the biorthogonal basis for the cut-element have been modified to satisfy biorthogonality condition for truncated support.

where $\mathbb{1}$ denotes the constant function with value 1 and δ_{pq} denotes the Kronecker delta, defined as in (1.1).

For linear and bilinear elements, it is possible to compute the biorthogonal basis $(\psi_\ell^p)_{p\in\widetilde{\mathcal{N}_\ell}}$ as a linear combination of the Lagrange basis $(\phi_\ell^k)_{k\in\mathcal{N}_\ell}$ for each element $K \in \widetilde{\mathcal{T}}_\ell$,

$$\psi_\ell^p = C_{pq}\phi_\ell^q \qquad \forall p,q \in \mathcal{N}_K, \tag{4.11}$$

where C_{pq} denotes the entries of the coefficient matrix $\boldsymbol{C}_K$, which defines the coefficient of the linear combination and $\mathcal{N}_K$ denotes set of nodes of a given triangulation K. Using the formula (4.11), we can compute the coefficients matrix $\boldsymbol{C}_K$, for each element $K \in \mathcal{T}_{\ell,i}$, for all $i \in \{1,\dots,r\}$ as

$$C_{pq}(\phi_\ell^q,\phi_\ell^p)_{L^2(K_i)} = \delta_{pq}(\phi_\ell^p,\mathbb{1})_{L^2(K_i)} \qquad \forall p,q \in \mathcal{N}_K.$$

Here, we note for each subdomain the L^2-inner product is restricted to the element $K_i = K \cap \Omega_i$, for all, $i = 1\dots,r$. The matrix representation of the multiplication of element-wise basis function is defined as $(M_K)_{pq} = (\phi_\ell^p,\phi_\ell^q)_{L^2(K_i)}$ and $(N_K)_{pq} = \delta_{pq}(\phi_\ell^p,\mathbb{1})_{L^2(K_i)}$. Here, the elemental matrix $\boldsymbol{N}_K$ is diagonal and $\boldsymbol{M}_K$ is an elemental mass matrix with the entries $(N_K)_{pq}$ and $(M_K)_{pq}$, respectively. In the algebraic form, it can be written as

$$\boldsymbol{C}_K \boldsymbol{M}_K = \boldsymbol{N}_K.$$

The element-wise coefficients of the linear combination could be computed as

$$\boldsymbol{C}_K = \boldsymbol{N}_K (\boldsymbol{M}_K)^{-1}.$$

Once, the coefficients matrix $\boldsymbol{C}_K$ is computed, we can construct the biorthogonal basis function using (4.11).

In general, it is not necessary to compute the biorthogonal basis functions for each element. For the uncut elements, we can compute the coefficients for an element and construct the biorthogonal basis from the pre-computed coefficients. While for the cut elements, it is essential to compute the coefficients explicitly for each cut element. For the cut elements, the support of the Lagrange basis function depends on the location of the interface with respect to the background mesh. Hence, also the biorthogonal basis for the corresponding cut elements depends on the support of the Lagrange basis functions. The biorthogonal basis functions computed using this strategy are not necessarily continuous for the cut elements. In Figure 4.4, we can see the Lagrange basis and the corresponding biorthogonal basis functions for a cut mesh.

Now, we define the pseudo-L^2-projection operator, where the Lagrange multiplier space is spanned by the biorthogonal basis functions. In the discrete setting, the entries of matrix $\boldsymbol{M}$ are given by $M_{ik} = (\phi_\ell^k, \psi_\ell^i)_{L^2(\Omega)} = (\phi_\ell^i, \mathbb{1})_{L^2(\Omega)}$. Due to this definition, the matrix $\boldsymbol{M}$ computed using the biorthogonal basis is diagonal. This matrix is computationally trivial to invert, and the inverse of the matrix is also diagonal. The matrix $\boldsymbol{N}$, defined between a coarse and a fine level can be given with the entries $N_{ij} = (\phi_{\ell-1}^j, \psi_\ell^i)_{L^2(\Omega)}$. The transfer operator computed using this method has a sparse structure, and the support of the basis functions on the coarse level is also smaller compared to the standard L^2-projection operator.

Hence in the semi-geometric multigrid method, we use the transfer operator computed using the pseudo-L^2-projections.

4.6 Numerical Results

In this section, we evaluate the performance of our semi-geometric multigrid method for different variants of Nitsche's method using the examples discussed in Section 2.7 for the fictitious domain method and Section 3.6 for the interface problem. We employ the multigrid method as a solution method and as a preconditioner and compare its performance against other preconditioners.

Our examples have highly varying coefficients, hence, to compare all solution methods on the same scale we choose relative residual in energy norm as a termination criterion, as

$$\frac{\|\boldsymbol{f}-\boldsymbol{A}\boldsymbol{u}^{(k)}\|_A}{\|\boldsymbol{f}-\boldsymbol{A}\boldsymbol{u}^{(0)}\|_A} < 10^{-12}. \tag{4.12}$$

Additionally, we define the asymptotic convergence rate of an iterative solver as

$$\rho^* := \frac{\|\boldsymbol{u}^{(k+1)}-\boldsymbol{u}^{(k)}\|_A}{\|\boldsymbol{u}^{(k)}-\boldsymbol{u}^{(k-1)}\|_A},$$

where the iterate $\boldsymbol{u}^{(k+1)}$ satisfies the termination criterion (4.12).

4.6.1 Comparison with Other Preconditioners

The system of linear equations arising from Nitsche's method and the penalty method are symmetric positive definite (SPD). The most natural choice of an iterative solver for such problems is the conjugate gradient (CG) method. Although, the CG method has the best convergence rate amongst all Krylov subspace solvers for SPD systems, in practice preconditioned CG method is used to ensure the fast convergence and, in some cases, to ensure the convergence of the solver up to a certain tolerance. We use the preconditioned CG method as a solver in our numerical experiments with Jacobi, symmetric Gauss-Seidel (SGS), and semi-geometric multigrid methods as preconditioners, which will be denotes as CG-Jacobi, CG-SGS, CG-SMG, respectively.

The experiments are carried out on the system of linear equations with around 2.5×10^6 DOFs ($L5$). Our semi-geometric multigrid method is set up with 5-levels, and symmetric Gauss-Seidel is chosen as smoother with 3 pre-smoothing and 3 post-smoothing steps at each level, and we perform a single V-cycle as a preconditioner.

Preconditioners

Table 4.1 shows the number of iterations required by different methods to reach the termination criterion (4.12). We observe that the CG method preconditioned with the Jacobi method has the slowest convergence amongst all solvers. CG method with SGS as a preconditioner is significantly better than the Jacobi preconditioner, the number of iterations is reduced in more than half for most of the problems. The best performance from all the preconditioners is clearly shown by the SMG preconditioner.

		CG-Jacobi	CG-SGS	CG-SMG	SMG	(ρ^*)
Penalty	Example 1-FD	3424	1245	14	28	(0.493)
	Example 2-FD	3417	1243	19	41	(0.793)
Nitsche	Example 1-FD	3256	1121	8	10	(0.116)
	Example 2-FD	3101	1235	10	13	(0.117)

(a) Penalty and Nitsche's method for fictitious domain problem

	CG-Jacobi	CG-SGS	CG-SMG	SMG	(ρ^*)
IN-EV	6363	2048	8	9	(0.113)
IN-LO	5365	2001	7	8	(0.087)
IN-GP	5396	2051	9	12	(0.181)

(b) Nitsche's method for solving Example 1-IF

		CG-Jacobi	CG-SGS	CG-SMG	SMG	(ρ^*)
$\alpha_1 = 10^{-1}$ $\alpha_2 = 1$	IN-EV	5276	1779	8	10	(0.157)
	IN-LO	4710	1724	7	9	(0.112)
	IN-GP	4510	1787	7	8	(0.101)
$\alpha_1 = 10^{-5}$ $\alpha_2 = 1$	IN-EV	4036	1626	8	10	(0.172)
	IN-LO	4027	1626	8	10	(0.195)
	IN-GP	4494	1761	7	7	(0.092)
$\alpha_1 = 10^{-9}$ $\alpha_2 = 1$	IN-EV	4139	1661	8	10	(0.171)
	IN-LO	4132	1655	8	10	(0.195)
	IN-GP	4969	1675	7	7	(0.092)

(c) Nitsche's method for solving Example 2-IF

		CG-Jacobi	CG-SGS	CG-SMG	SMG	(ρ^*)
$\alpha_1 = 1$ $\alpha_2 = 10$	IN-EV	5192	1794	7	9	(0.135)
	IN-LO	4655	1784	7	7	(0.031)
	IN-EV	4428	1803	7	7	(0.033)
$\alpha_1 = 1$ $\alpha_2 = 10^5$	IN-EV	3761	1635	6	7	(0.029)
	IN-LO	3752	1635	6	7	(0.029)
	IN-GP	4417	1684	6	7	(0.029)
$\alpha_1 = 1$ $\alpha_2 = 10^9$	IN-EV	3535	1509	6	7	(0.029)
	IN-LO	3520	1509	6	7	(0.029)
	IN-GP	3941	1550	6	7	(0.029)

(d) Nitsche's method for solving Example 3-IF

Table 4.1. The number of iterations required by different solution methods to reach the predefined tolerance, the last column shows the asymptotic convergence rates of the SMG method.

The convergence rate of the conjugate gradient method depends on the distribution of the spectrum of $\boldsymbol{A}$, and the method performs very well if the eigenvalues are clustered in a certain region of the spectrum, rather than being uniformly distributed. We observe that the CG-SMG method is stable for all discussed discretization methods and coefficients, as the number of iterations required to reach the predefined tolerance stays stable.

Performance as a Solution Method

For the fictitious domain method, the SMG method converges significantly better for Nitsche's method than for the penalty method. From Figure 2.7, it is clear that the condition number of the system matrix arising from Nitsche's method is larger than the system matrices arising from the penalty method. But still, the large value of the stabilization parameter in the penalty method affects the asymptotic convergence rate significantly more than the Nitsche's method. This could be attributed to the different distribution of the eigenvalues in the penalty method and Nitsche's method.

While for the interface problem, we observe Nitsche's method with the ghost penalty stabilization term converges fastest for highly varying coefficients, while it is slowest for the continuous coefficients. Even though the difference is not significantly high, a few more iterations can be attributed to the large value of the stabilization parameter. For the (IN-EV) and (IN-LO) the number of iterations to reach the tolerance stays more or less stable. The multigrid method can be considered quite robust in terms of the asymptotic convergence rates, as for all the experiments we observe $\rho^* < 0.2$. A multigrid method can be interpreted as a Richardson method with SMG as a preconditioner, and the CG method is known to be far superior to the Richardson method. Hence, we observe that the number of iterations required is smaller in all cases when the semi-geometric multigrid is used as a preconditioner rather than a solution method.

Level Independence

In this part, we evaluate the performance of the CG-SMG method with respect to the number of levels in the multigrid hierarchy. The finest level is kept the same as in the previous experiments, and the number of levels used in the multilevel hierarchy is changed. As we use a direct solver on the coarsest level, the coarse level corrections become increasingly more accurate as the number of levels is reduced. The higher number of levels is computationally cheaper since a smaller linear system of equations is solved by the direct solver on the coarsest level.

# levels		2	3	4	5
$\alpha_1 = 1$ $\alpha_2 = 10$	IN-EV	7	7	7	7
	IN-LO	6	6	6	7
	IN-GP	7	7	7	7
$\alpha_1 = 1$ $\alpha_2 = 10^5$	IN-EV	6	6	6	6
	IN-LO	6	6	6	6
	IN-GP	6	6	6	6
$\alpha_1 = 1$ $\alpha_2 = 10^9$	IN-EV	6	6	6	6
	IN-LO	6	6	6	6
	IN-GP	6	6	6	6

Table 4.2. The number of iterations required by the PCG-SMG method to reach a predefined tolerance with a different number of levels in the multigrid hierarchy for solving Example 3-IF.

# interfaces	1	2	4	6	8	10
IN-EV	9	9	9	9	9	9
IN-LO	8	8	8	8	8	8
IN-GP	9	9	9	9	9	9

Table 4.3. The number of iterations required by the PCG-SMG method to reach a predefined tolerance for solving the problem with multiple interfaces in the domain.

Table 4.2 demonstrates that the number of iterations stays constant regardless of the number of levels. We observe that the change in the ratio between the coefficients α_1, α_2 does not affect the performance of the CG-SMG method. This result shows the level independence property of our multigrid method.

4.6.2 Effect of Multiple Interfaces on the Multigrid Method

The last set of experiments demonstrates the robustness of the SMG method with respect to the number of interfaces within a domain. We consider Example 1-IF with continuous coefficients. The finest level is kept the same as in the previous cases, and the multigrid hierarchy consists of 5-levels. This test is performed for all the discussed variants of Nitsche's methods with multiple interfaces.

The interfaces are represented by zero level set of the following functions

$$\Lambda_i(x) := \begin{cases} x - 0.1\left(\frac{1}{\sqrt{2}} + i - 1\right) & \text{for all } i \in \{1, \ldots, 5\}, \\ x + 0.1\left(\frac{1}{\sqrt{2}} - i\right) & \text{for all } i \in \{6, \ldots, 10\}. \end{cases}$$

All the interfaces are linear and parallel to the original interface Γ_l. In the performed experiment, we start with a single interface and increase up to 10 interfaces in the domain. From Table 4.3, we can observe that the proposed multigrid method is stable, as the number of iterations do not change with increasing number of interfaces in the domain.

Hence, we can conclude that our semi-geometric method is a robust solution method and even more robust when employed as a preconditioner. In particular, the method is stable for all variants of Nitsche's method with respect to highly varying coefficients, the number of levels in the multilevel hierarchy, and also with respect to the number of interfaces in the domain.

Chapter 5

A Multigrid Method for Linear Constraints

In this chapter, we discuss the saddle point problem in the abstract sense, and we introduce a new multigrid method for solving the saddle point problems arising from the Lagrange multiplier based XFEM discretization.

The traditional multigrid methods for solving the saddle point systems require a different class of smoothers and the transfer operators [BDW99, Bac14]. As we have discussed, the efficiency of multigrid methods is heavily dependent on both of these ingredients. Depending on the type of problems, there are many options for the smoothers for the saddle point systems [Zul00, Woh00b, BLS14, Krz04], which can be explored for our problem. The transfer operators used in the saddle point system require to prolongate and restrict both primal variable (solution) and the dual variable (Lagrange multiplier). In the context of our saddle point problem, it is not an easy task to create such a transfer operator. For transferring the primal variables, we can employ the pseudo-L^2-projections as discussed in Section 4.5. But the design of the transfer operator for the Lagrange multiplier is not straightforward. The Lagrange multipliers defined on the vital vertices are not defined uniquely, thus creating a relationship between them in a multilevel framework is challenging. The multigrid method proposed in this section requires the transfer operator only for the primal variable. The novel multigrid method is based on the semi-geometric multigrid method from the last chapter, with some modifications on the finest level.

This chapter is organized as follows. In the next section, we provide a description of the saddle point problem. In Section 5.2, we discuss the standard iterative methods for solving the saddle point problems. Further, we introduce our new multigrid method in Section 5.3. In the last section, we carry out numerical experiments to compare the performance of the standard iterative solvers and the new multigrid method.

5.1 Saddle Point Problem

In this section, we discuss the abstract variational formulation arising from the Lagrange multiplier formulation. This formulation can be stemming from imposing either the Dirichlet boundary condition or the interface conditions. The saddle point formulation is given as, find $(u, \lambda) \in \mathcal{V}_h \times \mathcal{M}_h$ such that

$$\begin{aligned} A(u, v) + b(\lambda, v) &= F(v) \qquad && \forall v \in \mathcal{V}_h, \\ b(\mu, u) &= G(\mu) && \forall \mu \in \mathcal{M}_h. \end{aligned} \tag{5.1}$$

Recall, the linear operators $\mathcal{A} : \mathcal{V}_h \to \mathcal{V}_h^*$ and its adjoint $\mathcal{A}^\mathsf{T} : \mathcal{V}_h^* \to \mathcal{V}_h$ are associated with the bilinear form $A(\cdot, \cdot)$. The linear operator $\mathcal{A}$ is continuous symmetric, and it is assumed to be positive semidefinite. In the fictitious domain method, if the operator is not modified to incorporate the boundary conditions explicitly then a non-trivial kernel of the operator $\mathcal{A}$ exists. For the interface problems, the floating domains may exist in the presence of close interfaces, especially, if the close interfaces do not intersect with a Dirichlet boundary.

We define a linear operator $\mathcal{B} : \mathcal{V}_h \to \mathcal{M}_h^*$ associated with bilinear form $b(\cdot, \cdot)$ such that

$$b(u, \mu) = \langle \mathcal{B}u, \mu \rangle_{\mathcal{M}_h^* \times \mathcal{M}_h} \qquad \forall u \in \mathcal{V}_h, \ \forall \mu \in \mathcal{M}_h.$$

An adjoint of the operator $\mathcal{B}$ is defined as $\mathcal{B}^\mathsf{T} : \mathcal{M}_h \to \mathcal{V}_h^*$ such that

$$\langle u, \mathcal{B}^\mathsf{T} \mu \rangle_{\mathcal{V}_h^* \times \mathcal{V}_h} := \langle \mathcal{B}u, \mu \rangle_{\mathcal{M}_h^* \times \mathcal{M}_h} \qquad \forall u \in \mathcal{V}_h, \ \forall \mu \in \mathcal{M}_h.$$

Here, $G(\cdot)$ denotes a linear functional that enforces either Dirichlet boundary conditions or interface conditions. We define the functional $G(\cdot)$ in (5.1) by choosing a function $g_\lambda \in \mathcal{M}_h^*$ such that

$$G(\mu) = \langle g_\lambda, \mu \rangle_{\mathcal{M}_h^* \times \mathcal{M}_h} \quad \forall \mu \in \mathcal{M}_h.$$

We write the saddle point problem (5.1) in operator form as, find $(u, \lambda) \in \mathcal{V}_h \times \mathcal{M}_h$ such that

$$\begin{aligned} \mathcal{A}u + \mathcal{B}^\mathsf{T} \lambda &= f \qquad && \text{in } \mathcal{V}_h^*, \\ \mathcal{B}u &= g_\lambda && \text{in } \mathcal{M}_h^*. \end{aligned} \tag{5.2}$$

The operator $\mathcal{A}$ is a symmetric positive semidefinite, and it does not have an exact inverse. The problem (5.2) is solvable if and only if, $(f - B^\mathsf{T} \lambda) \perp \mathrm{Ker}(\mathcal{A})$, where $\mathrm{Ker}(\mathcal{A})$ denotes the kernel of operator $\mathcal{A}$. The saddle point problem has a unique solution if $\mathrm{Ker}(\mathcal{A}) \cap \mathrm{Ker}(\mathcal{B}) = \{0\}$.

We rewrite the saddle point problem in the matrix-vector formulation as

$$\begin{aligned} \boldsymbol{A}\boldsymbol{u} + \boldsymbol{B}^{\mathsf{T}}\boldsymbol{\lambda} &= \boldsymbol{f} \\ \boldsymbol{B}\boldsymbol{u} &= \boldsymbol{g}_{\lambda}, \end{aligned} \tag{5.3}$$

where $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, $\boldsymbol{f} \in \mathbb{R}^{n}$, $\boldsymbol{B} \in \mathbb{R}^{m\times n}$, $\mathrm{rank}(\boldsymbol{B}) = m$, $\boldsymbol{g}_{\lambda} \in \mathbb{R}^{m}$, $m \ll n$, with the unknowns $\boldsymbol{u} \in \mathbb{R}^{n}$ and the Lagrange multiplier $\boldsymbol{\lambda} \in \mathbb{R}^{m}$. The matrix $\boldsymbol{A}$ is semidefinite, and its condition number increases with order h^{-2}, while the matrix $\boldsymbol{B}$ is a scaled mass matrix on the interface and its condition number increases with order h^{-1}. In practice, special care is required in order to solve the system with a non-trivial kernel. In particular, if the null space of the matrix $\boldsymbol{A}$ is known, the iterative process can create a sequence of iterates that are orthogonal to $\mathrm{Ker}(\boldsymbol{A})$ [BL05]. As this work is not focused on the iterative solvers for positive semidefinite systems, we take a different approach and transform the block matrix $\boldsymbol{A}$ into an equivalent symmetric positive definite matrix.

The minimization problem associated with the saddle point problem (5.2) can be posed as,

$$\begin{aligned} &\min_{\boldsymbol{u}} J(\boldsymbol{u}) = \frac{1}{2}\boldsymbol{u}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{u} - \boldsymbol{u}^{\mathsf{T}}\boldsymbol{f} \\ &\text{subject to} \quad \boldsymbol{B}\boldsymbol{u} = \boldsymbol{g}_{\lambda}. \end{aligned} \tag{5.4}$$

Employing the augmented Lagrangian approach we reformulate the problem (5.4) as,

$$\begin{aligned} &\min_{\boldsymbol{u}} J(\boldsymbol{u}) = \frac{1}{2}\boldsymbol{u}^{\mathsf{T}}(\boldsymbol{A} + \gamma_s\boldsymbol{B}^{\mathsf{T}}\boldsymbol{B})\boldsymbol{u} - \boldsymbol{u}^{\mathsf{T}}(\boldsymbol{f} + \gamma_s\boldsymbol{B}^{\mathsf{T}}\boldsymbol{g}_{\lambda}) \\ &\text{subject to} \quad \boldsymbol{B}\boldsymbol{u} = \boldsymbol{g}_{\lambda}, \end{aligned} \tag{5.5}$$

where $\gamma_s > 0$ is a penalty parameter. Using (5.5), we can replace the saddle point formulation (5.3) with the equivalent problem

$$\begin{aligned} \boldsymbol{A}_{\gamma}\boldsymbol{u} + \boldsymbol{B}^{\mathsf{T}}\boldsymbol{\lambda} &= \boldsymbol{f}_{\gamma} \\ \boldsymbol{B}\boldsymbol{u} &= \boldsymbol{g}_{\lambda}, \end{aligned} \tag{5.6}$$

where $\boldsymbol{A}_{\gamma} := \boldsymbol{A} + \gamma_s\boldsymbol{B}^{\mathsf{T}}\boldsymbol{B}$ and $\boldsymbol{f}_{\gamma} := \boldsymbol{f} + \gamma_s\boldsymbol{B}^{\mathsf{T}}\boldsymbol{g}_{\lambda}$ are the augmented matrices and vectors. Now, the matrix $\boldsymbol{A}_{\gamma}$ in the saddle point problem (5.6) is a positive definite matrix. In this work, we choose the value of the penalty parameter as

$$\gamma_s := \frac{\lambda_{\max}(\boldsymbol{A})}{\lambda_{\max}(\boldsymbol{B}^{\mathsf{T}}\boldsymbol{B})},$$

where $\lambda_{\max}$ denotes the largest eigenvalue of the corresponding matrix. This penalty parameter is quite an attractive option as it minimizes the condition number of the whole saddle point system and by extension enhances the convergence of the standard iterative methods [GGV04].

From now onwards, all the methods discussed in this chapter are concerned with solving the modified saddle point problem (5.6).

5.2 Standard Iterative Solvers

The saddle point problem arises in a wide variety of applications, for instance, the mixed finite element discretization, especially in fluid and solid mechanics [BBF13], mortar methods [Woh01], etc. In the optimization framework, this type of problem arises from interior point methods for both linear and nonlinear optimization problems [NW00]. Even though the saddle point systems arising from the different problems have very different properties, the standard iterative methods used to solve them have the same characteristics [Zul01, BGL05]. In this section, we discuss some of the basic iterative schemes for solving the problems arising from the Lagrange multiplier based XFEM discretization.

5.2.1 Schur Complement Reduction

The Schur-complement method is one of the most common approaches for solving the saddle point problem. The idea of this method is to first solve the smaller problem by computing the Lagrange multiplier, and once the multiplier is known, we can compute the primal variable.

As the matrix $\boldsymbol{A}_\gamma$ is invertible, we can rewrite the first part of the saddle point problem as

$$\boldsymbol{u} = \boldsymbol{A}_\gamma^{-1}(\boldsymbol{f}_\gamma - \boldsymbol{B}^\top \boldsymbol{\lambda}). \tag{5.7}$$

By substituting the above equation in the second part of (5.6) we get the dual formulation,

$$\boldsymbol{B}\boldsymbol{A}_\gamma^{-1}\boldsymbol{B}^\top \boldsymbol{\lambda} = \boldsymbol{B}\boldsymbol{A}_\gamma^{-1}\boldsymbol{f}_\gamma - \boldsymbol{g}_\lambda. \tag{5.8}$$

This dual system is of the reduced dimension m, the dimension of the multiplier space. The new operator $\boldsymbol{B}\boldsymbol{A}_\gamma^{-1}\boldsymbol{B}^\top$ is referred to as Schur-complement matrix, which is also a symmetric positive definite matrix. Once the Lagrange multiplier $\boldsymbol{\lambda}$ is computed from (5.8), we obtain $\boldsymbol{u}$ by solving (5.7). This is one of the simplest, but quite expensive approach.

The explicit computation of the Schur-complement matrix and the right-hand side of the dual system is quite expensive, as we need to compute the inverse of the matrix $\boldsymbol{A}_\gamma$. Here, we can use the direct methods for computation of the Schur-complement system, but we rather employ iterative methods such as Krylov subspace methods. In the iterative methods, we can compute the action of the inverse on a vector rather than computing the explicit inverse. If we can carry out this step economically, it directly influences the computational complexity of the overall algorithm. We can use any of the linear iterative solution methods to compute the action of the inverse. As we have a robust solution method for solving the linear system of this kind, we can use the semi-geometric multigrid method (Algorithm 4.5), or conjugate gradient method preconditioned with the semi-geometric multigrid method. The dual system, in general, is not easy to solve as the Schur-complement matrices are normally poorly conditioned (especially when the matrix $\boldsymbol{A}_\gamma$ is ill-conditioned). In such cases, the solution schemes for solving the dual problem may not reach prescribed tolerance or may even diverge, without suitable preconditioning strategies. Also, it can be a tricky task to design the preconditioners for the Schur-complement system when we don't have an explicit representation of the Schur-complement matrix.

In Section 5.2.3, we discuss some of the common preconditioning strategies for solving the dual system.

5.2.2 Uzawa Methods

The first iterative schemes for solving the saddle point problems were developed by Arrow, Hurwicz, and Uzawa. The Arrow-Hurwicz method and the Uzawa methods are the simplest kind of iterative schemes for solving the coupled saddle point problems [Bac06]. The Uzawa method is widely used for solving saddle point problems arising from the Stoke's problem, incompressible solid and fluid mechanics problems. It consists of the following coupled iteration

$$\begin{aligned} \boldsymbol{A}_\gamma \boldsymbol{u}^{(k+1)} &= \boldsymbol{f}_\gamma - \boldsymbol{B}^\top \boldsymbol{\lambda}^{(k)}, \\ \boldsymbol{\lambda}^{(k+1)} &= \boldsymbol{\lambda}^{(k)} + \omega(\boldsymbol{B}\boldsymbol{u}^{(k+1)} - \boldsymbol{g}_\lambda), \end{aligned} \tag{5.9}$$

where $\omega > 0$ is a relaxation parameter. If we use the first equation to eliminate $\boldsymbol{u}^{(k+1)}$ from the second equation, we have a stationary iterative method for the Schur-complement system

$$\boldsymbol{\lambda}^{(k+1)} = \boldsymbol{\lambda}^{(k)} + \omega(\boldsymbol{B}\boldsymbol{A}_\gamma^{-1}(\boldsymbol{f}_\gamma - \boldsymbol{B}^\top \boldsymbol{\lambda}_k) - \boldsymbol{g}_\lambda).$$

This method can be viewed as a classical Richardson method for the Schur-complement system.

In the last section, the primal and dual linear systems are decoupled and solved separately, while in the case of the Uzawa method the coupled saddle point system is solved. This iteration scheme also requires us to solve the primal problem exactly in each iteration. This is often not necessary, instead one can use the inexact solution methods [IXZ14]. The Uzawa methods are generally slow to converge, as their convergence rate depends on the choice of the relaxation parameter. Therefore, it is desirable to replace the Richardson method in the Uzawa scheme by more robust steepest descent or conjugate gradient methods [Bra07, BS13]. Following the same strategy from the last section, we employ the semi-geometric multigrid method for solving the primal problem. Simultaneously, we also employ some preconditioning strategies to accelerate the convergence of the Uzawa iteration (5.9) and to reduce the overall computational cost of the solution scheme. The preconditioned Uzawa method can be written as

$$\begin{aligned} \boldsymbol{A}_\gamma \boldsymbol{u}^{(k+1)} &= \boldsymbol{f}_\gamma - \boldsymbol{B}^\top \boldsymbol{\lambda}^{(k)} \\ \boldsymbol{\lambda}^{(k+1)} &= \boldsymbol{\lambda}^{(k)} + \boldsymbol{P}^{-1}(\boldsymbol{B}\boldsymbol{u}^{(k+1)} - \boldsymbol{g}_\lambda), \end{aligned} \tag{5.10}$$

where $\boldsymbol{P} \in \mathbb{R}^{m \times m}$ denotes the preconditioner matrix for the dual system.

In the next section, we describe some of the preconditioning techniques for solving the dual system.

5.2.3 Preconditioners for Dual System

Since the saddle point problem appears in many practical applications, there have been many efforts for developing optimal preconditioning strategies. In this section, we focus on the preconditioners developed in two different applications, namely FETI methods and SIMPLE methods.

FETI Preconditioners

The Finite Element Tearing and Interconnecting (FETI) methods were introduced by Farhat and Raux as a non-overlapping domain decomposition methods [FR91]. FETI is a group of iterative sub-structuring methods for solving large systems of linear equations arising from FEM discretization. By design, the FETI methods are parallel solution methods, where the computational domain is decomposed into multiple subdomains. These subdomains are distributed among multiple processors. On each processor, a local Neumann problem is solved along with a coarse problem, used for global information transfer. The continuity between all subdomains is imposed by means of the method of Lagrange multipliers.

We can leverage some tricks from the FETI methods for solving our problem as the algebraic formulation of the FETI method also gives rise to the saddle point system. We write the algebraic problem arising from the FETI method is given as

$$\begin{aligned} \boldsymbol{A}\boldsymbol{u} + \boldsymbol{B}^\top \boldsymbol{\lambda} &= \boldsymbol{f} \\ \boldsymbol{B}\boldsymbol{u} &= \boldsymbol{0}. \end{aligned} \tag{5.11}$$

Here, the matrix $\boldsymbol{B}$ is a constraint matrix with values 1, −1 and 0. The matrix $\boldsymbol{A}$ has a block diagonal structure, representing different subdomains which are separated by the interfaces. If a subdomain does not contain any Dirichlet nodes, that subdomain is called a floating domain. The submatrices associated with the floating domains have a non-trivial kernel. If any submatrix of $\boldsymbol{A}$ has a non-trivial kernel, the pseudoinverse of each submatrix is computed instead of the explicit inverse. For a general case, we denote by $\boldsymbol{A}^\dagger$ a pseudoinverse of matrix $\boldsymbol{A}$. The matrix $\boldsymbol{A}^\dagger$ is a block diagonal matrix composed of local Moore-Penrose pseudoinverses associated with each subdomain. This parallel solution method reduces the saddle point problem to a Schur-complement system after eliminating the interior nodes and the problem is solved only on the interface. The primal variable $\boldsymbol{u}$ is eliminated from (5.11), which gives rise to a dual problem, given as

$$\boldsymbol{B}\boldsymbol{A}^\dagger\boldsymbol{B}^\top \boldsymbol{\lambda} = \boldsymbol{B}\boldsymbol{A}^\dagger\boldsymbol{f}.$$

For simplicity, the Schur-complement matrix is defined as $\boldsymbol{S}^\dagger := \boldsymbol{B}\boldsymbol{A}^\dagger\boldsymbol{B}^\top$ and the right hand side is defined as $\boldsymbol{d}^\dagger := \boldsymbol{B}\boldsymbol{A}^\dagger\boldsymbol{f}$. This version of the FETI method is scalable with the number of subdomains, but the condition number of the Schur-complement system grows polynomially with the number of unknowns in the subdomains. Hence, it becomes essential to devise preconditioners for the dual systems. The preconditioned dual system is defined as

$$\boldsymbol{P}^{-1}\boldsymbol{S}^\dagger\boldsymbol{\lambda} = \boldsymbol{P}^{-1}\boldsymbol{d}^\dagger,$$

where $\boldsymbol{P} \in \mathbb{R}^{m\times m}$ denotes a preconditioner matrix. This dual system is closely related to the dual system (5.8). We remark that for brevity we have omitted the necessary details about the properties of the null space of the matrix $\boldsymbol{A}$. As, this section aims to establish the connection between the FETI methods and the saddle point problems, rather than providing an introduction to the FETI methods.

In earlier FETI literature [FMR94], Farhat et al. proposed the Dirichlet preconditioner, given as

$$\boldsymbol{P}_D^{-1} := \boldsymbol{B}\boldsymbol{A}\boldsymbol{B}^\top. \tag{5.12}$$

This preconditioner, even though being useful, is quite elementary. More complex and better alternatives for the preconditioners were proposed by Lacour [Lac96], given as

$$\boldsymbol{P}_L^{-1} := (\mathrm{diag}(\boldsymbol{B}\boldsymbol{B}^\top))^{-1}\boldsymbol{B}\boldsymbol{A}\boldsymbol{B}^\top(\mathrm{diag}(\boldsymbol{B}\boldsymbol{B}^\top))^{-1} \tag{5.13}$$

and by Klawonn and Widlund [KW01], given as

$$\boldsymbol{P}_{FETI}^{-1} := (\boldsymbol{B}\boldsymbol{D}^{-1}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}\boldsymbol{D}^{-1}\boldsymbol{A}\boldsymbol{D}^{-1}\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{D}^{-1}\boldsymbol{B}^\top)^{-1} \tag{5.14}$$

where, $\boldsymbol{D} = \mathrm{diag}(\boldsymbol{A})$. These preconditioners have been shown to perform better than the Dirichlet preconditioner (5.12). For comparison and more possibilities of FETI preconditioners, we refer to the work of Stefanica [Ste01].

SIMPLE Preconditioner

Another option for a preconditioner is used in the solution techniques for the problems arising from Navier-Stokes equations [Pat80]. This preconditioner is known as SIMPLE which is an acronym for Semi-Implicit Method for Pressure Linked Equation. We consider this preconditioned as the saddle point system for incompressible Navier-Stokes equation has the same algebraic formulation as (5.11). In this framework, the unknowns $\boldsymbol{u}$ represents the velocity vector and $\boldsymbol{\lambda}$ represents a pressure vector, while the matrix $\boldsymbol{B}$ represents the constraint on the pressure vector. The SIMPLE type preconditioner in the original formulation can be viewed as a semi-implicit Uzawa method and the preconditioner is normally used for a coupled iteration. In this work, we aim to only employ the preconditioner for solving the dual problem. The SIMPLE preconditioner is given as

$$\boldsymbol{P}_{SIMPLE}^{-1} := (\boldsymbol{B}\boldsymbol{D}^{-1}\boldsymbol{B}^\top)^{-1}.$$

The SIMPLE and FETI preconditioners are very attractive possibilities for solving the dual systems arising from the saddle point system (5.3). We aim to use these preconditioners for solving the dual problems arising in the Uzawa method (5.10) and in the Schur-complement method (5.8). While for solving the primal system, we rely on the robust semi-geometric multigrid method.

5.2.4 Conjugate Projected Gradient Method

In this section, we discuss a null-space based method for solving the saddle point problem (5.3). The conjugate projected gradient (CPG) method is used to solve a quadratic optimization problem, with the equality constraints (5.4) [CV01]. This quadratic problem can be solved by computing a basis $\boldsymbol{Z}_B \in \mathbb{R}^{n\times(n-m)}$ that spans the null space of matrix $\boldsymbol{B}$. These basis are used to eliminate the constraints and allow us to apply the conjugate gradient method on the reduced problem. We follow the strategy of Gould et al. [GHN01], where the need for computing a null space of $\boldsymbol{B}$ is circumvented. In this way, we only need to solve one linear system of equations, which reduces the computational cost of the solution method.

Here, we give an example of the CPG algorithm for an abstract projection operator $\boldsymbol{P}^{\perp} \in \mathbb{R}^{n\times n}$. From (5.4), it is clear that the solution of the optimization problem has to be in the null space of matrix $\boldsymbol{B}$. We choose the projection operator as an orthogonal projector to the row space of matrix $\boldsymbol{B}$. The projection operator without any preconditioner is given as

$$\boldsymbol{P}_I^{\perp} = \boldsymbol{I} - \boldsymbol{B}^{\mathsf{T}}(\boldsymbol{B}\boldsymbol{B}^{\mathsf{T}})^{-1}\boldsymbol{B}, \tag{5.15}$$

where $\boldsymbol{I} \in \mathbb{R}^{n\times n}$ is an identity matrix. In this process, we need to compute the inverse of $\boldsymbol{B}\boldsymbol{B}^{T}$, which could be performed using the Cholesky decomposition. It is also possible to use the projection operator in combination with some other preconditioners. For example, the projection operator with the Jacobi preconditioner is expressed by

$$\boldsymbol{P}_J^{\perp} = \boldsymbol{D}^{-1}(\boldsymbol{I} - \boldsymbol{B}^{T}(\boldsymbol{B}\boldsymbol{D}^{-1}\boldsymbol{B}^{T})^{-1}\boldsymbol{B}\boldsymbol{D}^{-1}), \tag{5.16}$$

where $\boldsymbol{D} = \text{diag}(\boldsymbol{A}_\gamma)$.

The detailed description of the CPG method is given in Algorithm 5.1. From the implementation perspective, the computation of the projected gradient can give rise to significant round-off errors that may prevent the iterates from remaining in the null space of the matrix $\boldsymbol{B}$, as the algorithm approaches the solution. As the iterations proceeds, the projected vector $\boldsymbol{z}^{(*)}$, becomes increasingly small with respect to $\boldsymbol{r}^{(*)}$. This can be remedied by employing the residual update strategy, where the residual is redefined in such a way that the norm of $\boldsymbol{r}^{(*)}$ stays close to the norm of $\boldsymbol{g}^{(*)}$. Thus, we redefine $\boldsymbol{r}^{(*)}$ immediately after its computation, as

$$\boldsymbol{r}^{(*)} \hookleftarrow \boldsymbol{P}^{\perp}\boldsymbol{r}^{(*)}.$$

Instead of the Jacobi preconditioner in the projection operators, we can also use a multigrid as preconditioner. We replace the matrix $\boldsymbol{D}$ in (5.16) by the matrix $\boldsymbol{A}_\gamma$

$$\boldsymbol{P}_{MG}^{\perp} = \boldsymbol{A}_\gamma^{-1}(\boldsymbol{I} - \boldsymbol{B}^{\mathsf{T}}(\boldsymbol{B}\boldsymbol{A}_\gamma^{-1}\boldsymbol{B}^{\mathsf{T}})^{-1}\boldsymbol{B}\boldsymbol{A}_\gamma^{-1}). \tag{5.17}$$

In order to utilize this projection operator (5.17), we do not have to compute the inverse of the matrix $\boldsymbol{A}_\gamma$, explicitly. We can perform a single $V(\nu_1,\nu_2)$-cycle of the semi-geometric multigrid iteration to compute the action of the inverse instead.

Algorithm 5.1: Conjugate projected gradient method

```
Input  : $A$, $f$, $u$ (s.t. $Bu = g_\lambda$), $P^\perp$
Output: $u$
Set: $r \hookleftarrow f - Au^{(0)}$, $z = P^\perp r$, $p = z$
while not converged do
    $\alpha \hookleftarrow \frac{r^\top r}{p^T A p}$ ;                 ▹ compute step size
    $u \hookleftarrow u + \alpha p$ ;                              ▹ update the iterate
    $r^{(*)} \hookleftarrow r - \alpha A p$ ;                        ▹ compute new residual
    $z^{(*)} \hookleftarrow P^\perp r^{(*)}$ ;                         ▹ projected gradient
    $\beta \hookleftarrow \frac{(r^{(*)})^\top z^{(*)}}{r^\top z}$;      ▹ orthogonalization
    $p \hookleftarrow z^{(*)} + \beta p$ ;                          ▹ new direction
    $r \hookleftarrow r^{(*)}$, $z \hookleftarrow z^{(*)}$ ;               ▹ next iteration
```

5.3 Generalized Multigrid Method for Linear Constraints

In this section, we introduce a new generalized multigrid method for solving a quadratic minimization problem with linear constraints (5.4) or an equivalent saddle point problem (5.6). This multigrid method is motivated by the monotone multigrid method [Kor94, Kor96, Kor97], which was originally developed to solve a quadratic minimization problem with pointwise equality and/or inequality constraints. Here, we present an extension of this method for solving the quadratic minimization problem with linear equality and/or inequality constraints.

The monotone multigrid method is an iterative method, where in each iteration the energy functional is minimized successively such that the current iterate satisfies the constraints. This task is carried out by the projected Gauss-Seidel method, which simultaneously minimizes the energy functional and projects the current iterate onto a feasible set. For the linear constraints, represented by a linear combination of several variables, the traditional projected Gauss-Seidel method is unusable. To overcome this difficulty, we introduce the orthogonal transformation and a variant of the projected Gauss-Seidel method that can handle the linear constraints. In addition, in this multigrid method we employ the transfer operators constructed using the pseudo-L^2-projections.

In this section, by slight abuse of notation, we drop the subscript γ from the matrix $\boldsymbol{A}_\gamma$ and the vector $\boldsymbol{f}_\gamma$. However, we still compute the matrix $\boldsymbol{A}$ and the vector $\boldsymbol{g}$ by means of augmented Lagrangian formulation, which ensures that the matrix $\boldsymbol{A}$ is symmetric positive definite.

5.3.1 Orthogonal Transformation

In this section, we introduce the orthogonal transformation for the original problem. This orthogonal transformation is necessary to decouple the linear constraints, which in turn allows us to use the modified projected Gauss-Seidel method in the next section.

We rewrite the saddle point problem as constrained minimization problem in the algebraic formulation as

$$\begin{aligned} &\min_{\boldsymbol{u}} J(\boldsymbol{u}) = \frac{1}{2}\boldsymbol{u}^\top \boldsymbol{A}\boldsymbol{u} - \boldsymbol{u}^\top \boldsymbol{f} \\ &\text{subject to} \quad \boldsymbol{B}\boldsymbol{u} = \boldsymbol{g}_\lambda, \end{aligned} \tag{5.18}$$

where $\boldsymbol{u}, \boldsymbol{f} \in \mathbb{R}^n$, $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, $\boldsymbol{B} \in \mathbb{R}^{m\times n}, \boldsymbol{g}_\lambda \in \mathbb{R}^m, m \ll n$ and $\operatorname{rank}(\boldsymbol{B}) = m$.

In order to decouple the constraints, we perform a QR decomposition of the constrained matrix $\boldsymbol{B}^\top$ to obtain

$$\boldsymbol{B}^\top = \boldsymbol{Q}\boldsymbol{R} \ \text{ and } \ \boldsymbol{B} = \boldsymbol{R}^\top \boldsymbol{Q}^\top,$$

where $\boldsymbol{Q} \in \mathbb{R}^{n\times n}$ is an orthonormal matrix, which means the adjoint of this matrix is its own inverse. Thus, we have $\boldsymbol{Q}\boldsymbol{Q}^\top = \boldsymbol{Q}^\top\boldsymbol{Q} = \boldsymbol{I}$, where $\boldsymbol{I} \in \mathbb{R}^{n\times n}$, represents the identity matrix. The decomposition of the matrix $\boldsymbol{R} \in \mathbb{R}^{n\times m}$ is given by $\boldsymbol{R} = [\boldsymbol{R}_1 \ \boldsymbol{O}_1]^\top$, where $\boldsymbol{R}_1 \in \mathbb{R}^{m\times m}$ is an upper triangular matrix and $\boldsymbol{O}_1 \in \mathbb{R}^{(n-m)\times n}$ is a matrix with all zero entries. The orthonormal matrix $\boldsymbol{Q}$ simply provides a change of basis, and on this new basis, the representation of the constraint is modified. It is clear from the structure of the new constraint matrix $\boldsymbol{R}_1$ that in the new basis the constraints are sequentially dependent on the previous linear constraint.

We use this orthonormal matrix and project the problem on a different basis. The orthonormal matrix $\boldsymbol{Q}$ is used to define the variables in the new basis system, given as $\widetilde{\boldsymbol{u}} = \boldsymbol{Q}^\top \boldsymbol{u}$ and $\boldsymbol{u} = \boldsymbol{Q}\widetilde{\boldsymbol{u}}$. Moreover, we can observe that $\boldsymbol{Q}^\top \boldsymbol{B}^\top = \boldsymbol{R}$ and $\boldsymbol{B}\boldsymbol{Q} = \boldsymbol{R}^\top$. By incorporating, the transformed matrices and the vectors, we can reformulate the constrained minimization problem (5.18) as

$$\begin{aligned} &\min_{\widetilde{\boldsymbol{u}}} J(\widetilde{\boldsymbol{u}}) = \frac{1}{2}\widetilde{\boldsymbol{u}}^\top \widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{u}} - \widetilde{\boldsymbol{u}}^\top \widetilde{\boldsymbol{f}} \\ &\text{subject to} \quad \boldsymbol{R}^\top \widetilde{\boldsymbol{u}} = \boldsymbol{g}_\lambda, \end{aligned} \tag{5.19}$$

where $\widetilde{\boldsymbol{A}} = \boldsymbol{Q}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{Q}$ and $\widetilde{\boldsymbol{f}} = \boldsymbol{Q}^{\mathsf{T}}\boldsymbol{f}$. As $\boldsymbol{Q}$ is an orthonormal matrix, the spectral properties of the $\widetilde{\boldsymbol{A}}$ and $\boldsymbol{A}$ are equivalent. But, the sparsity pattern of the original matrix $\boldsymbol{A}$ and its rotated variant $\widetilde{\boldsymbol{A}}$ are quite different. In practice, the matrix $\widetilde{\boldsymbol{A}}$ is denser than the original matrix, which in turn increases the computational cost of the matrix-vector products in the algorithm. The new constraint matrix $\boldsymbol{R}^{\mathsf{T}}$ has a lower triangular structure, which can be handled easily by forward substitution. Here, it is important to note that, this type of constraint can be handled easily by the Gauss-Seidel method, due to its inherent sequential nature.

Now, we define a constrained subspace or a feasible set as

$$\tilde{\mathcal{K}} = \{\widetilde{\boldsymbol{u}} \in \mathbb{R}^n : \boldsymbol{R}^{\mathsf{T}}\widetilde{\boldsymbol{u}} = \boldsymbol{g}_{\lambda}\}.$$

We pose our problem as an energy minimization problem in the algebraic formulation:

$$\text{find } \widetilde{\boldsymbol{u}} \in \tilde{\mathcal{K}} \text{ such that} \qquad J(\widetilde{\boldsymbol{u}}) \leqslant J(\widetilde{\boldsymbol{v}}) \qquad \forall \widetilde{\boldsymbol{v}} \in \tilde{\mathcal{K}}. \tag{5.20}$$

The transformed saddle point system related to the above minimization problem is given as

$$\begin{pmatrix} \widetilde{\boldsymbol{A}} & \boldsymbol{R} \\ \boldsymbol{R}^{\mathsf{T}} & \boldsymbol{O} \end{pmatrix} \begin{pmatrix} \widetilde{\boldsymbol{u}} \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \widetilde{\boldsymbol{f}} \\ \boldsymbol{g}_{\lambda} \end{pmatrix},$$

where $\boldsymbol{O} \in \mathbb{R}^{m\times m}$ is a zero matrix.

5.3.2 Modified Projected Gauss-Seidel Method

Here, we introduce a modified projected Gauss-Seidel method for solving the problem (5.20).

The Gauss-Seidel method is known to minimize the energy functional $J(\cdot)$ in each local iteration step. The energy minimization takes place in the direction of the nodal basis functions that span the FE space. As we have discussed earlier in Section 4.2.1, the Gauss-Seidel method can be written as a subspace correction method, where the subspace decomposition is achieved by a direct splitting of the underlying FE space into one-dimensional subspaces spanned by the nodal basis function. The projected Gauss-Seidel method is used widely to solve various forms of obstacle problems, and it is known to be globally convergent [Kor94, KK01]. We remark that decoupling of the constraints with respect to the nodal basis function is essential for the global convergence of the Gauss-Seidel method [Kor97, Glo84]. The original linear constraints $\boldsymbol{B}\boldsymbol{u} = \boldsymbol{g}_{\lambda}$ do not satisfy this property, as the constraints are represented by the linear combination of basis functions. The QR decomposition allows us to decouple the constraints by expressing them in new basis as $\boldsymbol{R}^{\mathsf{T}}\widetilde{\boldsymbol{u}} = \boldsymbol{g}_{\lambda}$. As we are solving the minimization problem with equality constraints all constraints are binding. We define the set of all active nodes by

$$\mathscr{A} := \{p : (\boldsymbol{R}\widetilde{\boldsymbol{u}})_p = (\boldsymbol{g}_{\lambda})_p\}.$$

Otherwise, for regular upper and lower bounds, the active set is defined by the list of nodes where the constraints are binding. The matrix $\boldsymbol{R}^{\mathsf{T}}$ being a lower triangular matrix allows us to write the constraints as a linear combination of the current nodal basis function and previously constrained basis. This key idea allows us to use the projected Gauss-Seidel method to solve the problem (5.20).

The iterative process is given as follows. For a given k-th iterate $\widetilde{\boldsymbol{u}}^{(k)} \in \tilde{\mathcal{K}}$, we compute a sequence of local intermediate iterates, $\boldsymbol{w}^{(0)}, \boldsymbol{w}^{(1)}, \ldots, \boldsymbol{w}^{(n)}$ on the given subspace $\mathcal{X}_i \subset \mathcal{X}_h$. We begin with the first local iterate $\boldsymbol{w}^{(0)} = \widetilde{\boldsymbol{u}}^{(k)}$, and the next local iterates are given by $\boldsymbol{w}^{(i)} = \boldsymbol{w}^{(i-1)} + \boldsymbol{c}^{(i)}$, for $i = 1, \ldots, n$. Once all local intermediate iterates are computed, the new iterate is given by $\widetilde{\boldsymbol{u}}^{(k)} = \boldsymbol{w}^{(n)}$. The corrections $\boldsymbol{c}^{(i)}$ are the unique solution of the local subproblems, given as,

$$\text{find } \boldsymbol{c}^{(i)} \in \mathcal{D}^{(i)} \text{ such that} \quad J(\boldsymbol{w}^{(i-1)} + \boldsymbol{c}^{(i)}) \leqslant J(\boldsymbol{w}^{(i-1)} + \boldsymbol{c}) \qquad \forall \boldsymbol{c} \in \mathcal{D}^{(i)},$$

with closed, convex set $\mathcal{D}^{(i)}$ for abstract upper bound $\boldsymbol{ub} \in \mathbb{R}^m$ and lower bound $\boldsymbol{lb} \in \mathbb{R}^m$ is defined as

$$\mathcal{D}^{(i)} = \{\boldsymbol{c}^{(i)} \in \mathbb{R}^n : \boldsymbol{lb} - \boldsymbol{R}\boldsymbol{w}^{(i-1)} \leqslant \boldsymbol{R}\boldsymbol{c}^{(i)} \leqslant \boldsymbol{ub} - \boldsymbol{R}\boldsymbol{w}^{(i-1)}\}. \tag{5.21}$$

As, our problem has equality constraints, the upper bound and the lower bounds are defined as $\boldsymbol{lb} = \boldsymbol{ub} = \boldsymbol{g}_{\lambda}$. Each intermediate step ensures that the iterate does not violate the constraints. If the current iterate violates the constraints, then it is projected back into the admissible space, accordingly. Finally, the next iterate $\widetilde{\boldsymbol{u}}^{(k+1)}$ is given by

$$\widetilde{\boldsymbol{u}}^{(k+1)} = \widetilde{\boldsymbol{u}}^{(k)} + \sum_{i=1}^{n} \boldsymbol{c}^{(i)}.$$

The projected Gauss-Seidel method for a generic linear inequality constrained minimization problem is summarized in the Algorithm 5.2. Here, it is necessary to pay attention to the values of diagonal entries of the matrix $\boldsymbol{R}$, as the positive and negative values affect the inequality bounds differently. In the Algorithm 5.2, we have assumed that the values of diagonal entries of the matrix $\boldsymbol{R}$ are positive.

Thus, we have a globally convergent projected Gauss-Seidel method that can be used to solve the problem (5.20). But the convergence rate of the Gauss-Seidel method is known to deteriorate as the size of the problem increases. Hence, in order to accelerate the convergence, we employ the semi-geometric multigrid method and the modified projected Gauss-Seidel method is used as a smoother.

Algorithm 5.2: Projected Gauss-Seidel method

```
Input  : $\widetilde{\boldsymbol{A}}, \widetilde{\boldsymbol{f}}, \boldsymbol{R}, \widetilde{\boldsymbol{u}}^{(0)}, \boldsymbol{lb}, \boldsymbol{ub}, \nu_*$
Output: $\widetilde{\boldsymbol{u}}^{(\nu_*)}, \mathscr{A}$
Function: Projected GS($\widetilde{\boldsymbol{A}}, \widetilde{\boldsymbol{f}}, \boldsymbol{R}, \widetilde{\boldsymbol{u}}^{(0)}, \boldsymbol{lb}, \boldsymbol{ub}, \nu_*$):
for  $k = 1, 2, \dots, \nu_*$ do
    $\mathscr{A} \mapsfrom \emptyset$ ;                                      ▷ initialize empty active set
    for $i = 1, 2, \dots, n$ do
        $\widetilde{\boldsymbol{u}}_i^{(k)} = \frac{1}{\tilde{A}_{ii}}(\widetilde{\boldsymbol{f}}_i - \sum_{j<i} \tilde{A}_{ij} \widetilde{\boldsymbol{u}}_j^{(k)} - \sum_{j>i} \tilde{A}_{ij} \boldsymbol{u}_j^{(k-1)})$ ;      ▷ update
        if $i \leqslant m$ then
            $l_t = \frac{1}{R_{ii}}\left(\boldsymbol{lb}_i - \sum_{j=1}^{i-1} R_{ji} \widetilde{\boldsymbol{u}}_j^{(k)}\right)$ ;           ▷ updated lower bound
            $u_t = \frac{1}{R_{ii}}\left(\boldsymbol{ub}_i - \sum_{j=1}^{i-1} R_{ji} \widetilde{\boldsymbol{u}}_j^{(k)}\right)$ ;           ▷ updated upper bound
            if $l_t < \widetilde{\boldsymbol{u}}_i^{(k)} < u_t$ then
                $\widetilde{\boldsymbol{u}}_i^{(k)} = \max(l_t, \min(\widetilde{\boldsymbol{u}}_i^{(k)}, u_t))$ ;   ▷ projection onto feasible set
                $\mathscr{A} \mapsfrom \mathscr{A} \cup i$ ;              ▷ add current index to the active set
```

5.3.3 The Multigrid Method

In this section, we present the generalized multigrid method, by modifying the semi-geometric multigrid method from Section 4.4.2, for solving the saddle point problem (5.20).

Following the discussion in the previous chapter, we have a sequence of non-nested finite element spaces $(\mathcal{V}_\ell)_{\ell=0,\dots,L}$ associated with the hierarchy of meshes $(\mathcal{T}_\ell)_{\ell=1,\dots,L}$. Recall, the nodal basis functions are given by ϕ_h^p for all nodes $p \in \mathscr{N}_h$ associated with the mesh $\mathcal{T}_h$. Following Section 4.5, we have the transfer operators $(\Pi_{\ell-1}^{\ell})_{\ell=1,\dots,L}$ which are computed using the pseudo-L^2-projections. By means of these transfer operators, we create a hierarchy of nested finite element spaces $(\mathcal{X}_\ell)_{\ell=0,\dots,L}$ from the hierarchy of background meshes.

The prolongation matrices associated with the transfer operators are given as $(\boldsymbol{T}_{\ell-1}^{\ell})_{\ell=1,\dots,L}$. The orthogonal transformation of the matrix $\boldsymbol{B}^\top$ still plays a vital role in this multigrid method. Recall, the basis functions $\zeta_L^p = \phi_L^p$ for all $p \in \mathscr{N}_L$ are associated with the finest level $\mathcal{X}_L = \mathcal{V}_L$. These basis functions are also

Algorithm 5.3: Generalized Multigrid algorithm

```
Input  : $\boldsymbol{A}_L, \boldsymbol{f}_L, L, \nu_1, \nu_2, (\boldsymbol{T}^{\ell}_{\ell-1})_{\ell=1,\dots,L}, \boldsymbol{B}, \boldsymbol{lb}, \boldsymbol{ub}, \gamma$
Output: $\boldsymbol{u}_L \leftarrow \boldsymbol{Q}\widetilde{\boldsymbol{u}}_L$
Function: GMG$(\boldsymbol{A}_L, \boldsymbol{f}_L, L, \nu_1, \nu_2, (\boldsymbol{T}^{\ell}_{\ell-1})_{\ell=1,\dots,L}, \boldsymbol{B}, \boldsymbol{lb}, \boldsymbol{ub}, \gamma)$:
$\widetilde{\boldsymbol{u}}_L \leftarrow \boldsymbol{0}$;                                      ▷ initialize solution
$\boldsymbol{Q}, \boldsymbol{R} \leftarrow$ QR Transformation$(\boldsymbol{B}^{\top})$;          ▷ QR decomposition
$\widetilde{\boldsymbol{T}}^{L}_{L-1} \leftarrow \boldsymbol{Q}^{\top}\boldsymbol{T}^{L}_{L-1}$; $\widetilde{\boldsymbol{A}}_L \leftarrow \boldsymbol{Q}^{\top}\boldsymbol{A}_L\boldsymbol{Q}$; $\widetilde{\boldsymbol{f}}_L \leftarrow \boldsymbol{Q}^{\top}\boldsymbol{f}_L$;    ▷ orthogonal rotation
while not converged do
    $\widetilde{\boldsymbol{u}}_L, \mathscr{A}_L \leftarrow \widetilde{\boldsymbol{u}}_L +$ Projected GS$(\widetilde{\boldsymbol{A}}_L, \widetilde{\boldsymbol{f}}_L, \boldsymbol{R}^{\top}, \widetilde{\boldsymbol{u}}_L, \boldsymbol{lb}, \boldsymbol{ub}, \nu_1)$;    ▷ $\nu_1$ pre-smoothing steps
    $\widetilde{\boldsymbol{r}}_L \leftarrow \widetilde{\boldsymbol{f}}_L - \widetilde{\boldsymbol{A}}_L\widetilde{\boldsymbol{u}}_L$;      ▷ residual
    $\widetilde{\boldsymbol{r}}_{trc} \leftarrow$ trc$(\widetilde{\boldsymbol{r}}_L, \mathscr{A}_L)$; $\widetilde{\boldsymbol{A}}_{trc} \leftarrow$ trc$(\widetilde{\boldsymbol{A}}_L, \mathscr{A}_L)$;    ▷ truncation
    $\boldsymbol{r}_{L-1} \leftarrow (\widetilde{\boldsymbol{T}}^{L}_{L-1})^{\top}\widetilde{\boldsymbol{r}}_{trc}$ ;     ▷ restriction
    $\boldsymbol{A}_{L-1} \leftarrow (\widetilde{\boldsymbol{T}}^{L}_{L-1})^{\top}\widetilde{\boldsymbol{A}}_{trc}\widetilde{\boldsymbol{T}}^{L}_{L-1}$;     ▷ Galerkin projection
    $\boldsymbol{c}_{L-1} \leftarrow \boldsymbol{0}$ ;     ▷ initialize coarse level correction
    for $i = 1, \dots, \gamma$ do
        $\boldsymbol{c}_{L-1} \leftarrow \boldsymbol{c}_{L-1} +$ coarseMG$(\boldsymbol{A}_{L-1}, \boldsymbol{r}_{L-1}, L-1, \nu_1, \nu_2, \boldsymbol{T}^{L-1}_{L-2}, \gamma)$;    ▷ coarse level cycle
    $\widetilde{\boldsymbol{c}}_L \leftarrow \widetilde{\boldsymbol{T}}^{L}_{L-1}\boldsymbol{c}_{L-1}$;     ▷ prolongation
    $\widetilde{\boldsymbol{c}}_{\text{trc}} \leftarrow$ trc$(\boldsymbol{c}_L, \mathscr{A}_L)$;     ▷ truncation
    $\widetilde{\boldsymbol{u}}_L \leftarrow \widetilde{\boldsymbol{u}}_L + \widetilde{\boldsymbol{c}}_{\text{trc}}$;     ▷ update iterate
    $\widetilde{\boldsymbol{u}}_L, \mathscr{A}_L \leftarrow \widetilde{\boldsymbol{u}}_L +$ Projected GS$(\widetilde{\boldsymbol{A}}_L, \widetilde{\boldsymbol{f}}_L, \boldsymbol{R}^{\top}, \widetilde{\boldsymbol{u}}_L, \boldsymbol{lb}, \boldsymbol{ub}, \nu_2)$;    ▷ $\nu_2$ post-smoothing steps
```

modified or rotated after the orthogonal transformation, which can be written as

$$\tilde{\zeta}^{q}_{L} := \sum_{p \in \mathcal{N}_L} \boldsymbol{Q}_{pq}\zeta^{p}_{L} \qquad \forall q \in \mathcal{N}_L.$$

In Section 4.5, the transfer operators are computed using the basis functions that span the FE space on a coarse level and a fine level. With the modified basis functions on the finest level, it becomes essential to compute the transfer operator associated with the finest level such that the vector and the matrix quantities are projected on the FE space spanned by the modified basis. The modified prolongation operator $\boldsymbol{T}^{L}_{L-1}$ is defined as $\widetilde{\boldsymbol{T}}^{L}_{L-1} := \boldsymbol{Q}^{\top}\boldsymbol{T}^{L}_{L-1}$. Now, the basis function

Algorithm 5.4: Coarse level cycle

```
Input  : $(\boldsymbol{A}_\ell)_{\ell=0,\dots,L-1}, \boldsymbol{r}_{L-1}, L-1, \nu_1, \nu_2, (\boldsymbol{T}^\ell_{\ell-1})_{\ell=1,\dots,L-1}, \gamma$
Output: $\boldsymbol{c}_{L-1}$
Function: coarseMG($\boldsymbol{A}_\ell, \boldsymbol{r}_\ell, \ell, \nu_1, \nu_2, \boldsymbol{T}^\ell_{\ell-1}, \gamma$):
if $\ell \neq 0$ then
    $\boldsymbol{c}_\ell \leftarrow \boldsymbol{0}$;                                        ▷ initialize correction
    $\boldsymbol{c}_\ell \leftarrow$ Smoother($\boldsymbol{A}_\ell, \boldsymbol{c}_\ell, \boldsymbol{r}_\ell, \nu_1$);                    ▷ $\nu_1$ pre-smoothing steps
    $\boldsymbol{r}_{\ell-1} \leftarrow (\boldsymbol{T}^\ell_{\ell-1})^\top(\boldsymbol{r}_\ell - \boldsymbol{A}_\ell \boldsymbol{c}_\ell)$;                    ▷ restriction
    $\boldsymbol{A}_{\ell-1} \leftarrow (\boldsymbol{T}^\ell_{\ell-1})^\top \boldsymbol{A}_\ell \boldsymbol{T}^\ell_{\ell-1}$;                         ▷ Galerkin projection
    $\boldsymbol{c}_{\ell-1} \leftarrow \boldsymbol{0}$ ;                                   ▷ initialize coarse level correction
    for $i = 1, \dots, \gamma$ do
        $\boldsymbol{c}_{\ell-1} \leftarrow \boldsymbol{c}_{\ell-1}+$ coarseMG($\boldsymbol{A}_{\ell-1}, \boldsymbol{r}_{\ell-1}, \ell-1, \nu_1, \nu_2, \boldsymbol{T}^{\ell-1}_{\ell-2}, \gamma$);   ▷ coarse level cycle
    $\boldsymbol{c}_\ell \leftarrow \boldsymbol{c}_\ell + \boldsymbol{T}^\ell_{\ell-1}\boldsymbol{c}_{\ell-1}$;                              ▷ prolongation
    $\boldsymbol{c}_\ell \leftarrow$ Smoother($\boldsymbol{A}_\ell, \boldsymbol{c}_\ell, \boldsymbol{r}_\ell, \nu_2$);                   ▷ $\nu_2$ post-smoothing steps
else
    $\boldsymbol{c}_0 \leftarrow \boldsymbol{A}_0^{-1}\boldsymbol{r}_0$;                                   ▷ direct solver
```

associated with the FE space $\mathcal{X}_{L-1}$ can be given as

$$\tilde{\zeta}^q_{L-1} := \sum_{p\in\mathcal{N}_L} (\boldsymbol{Q}^\top \boldsymbol{T}^L_{L-1})_{pq}\,\zeta^p_L = \sum_{p\in\mathcal{N}_L} (\tilde{\boldsymbol{T}}^L_{L-1})_{pq}\,\zeta^p_L \qquad \forall q \in \mathcal{N}_{L-1}.$$

This modification of the transfer operator is only required on the finest level, while all other transfer operators $(\boldsymbol{T}^\ell_{\ell-1})_{\ell=0,\dots,L-1}$ remain the same.

The modified projected Gauss-Seidel method is employed as a smoother in the generalized multigrid method only on the finest level. It minimizes the energy functional in each local iteration in each smoothing step. At the end of the smoothing iterations, we obtain a list of active nodes where the constraints are binding. The most crucial feature of this multigrid method is that the coarse level corrections do not violate the fine level constraints. As a consequence, we solve the constrained optimization problem only on the finest level, while on the coarse levels $\ell = 0, \dots, L-1$, we solve the unconstrained linear problem. This is very convenient for solving the problem (5.18), as we do not require a coarse level representation of the constraint matrices.

In order to ensure that the coarse level corrections do not violate the constraint on the finest level, we modify the restriction of the residual and the stiffness matrix, and the prolongation of the coarse level correction. Following the discussion of the semi-geometric multigrid method, we know the basis functions associated with the coarse level FE space are computed as a linear combination of the basis function defined on the FE space on the finest level. If the value of a basis function on the finest level is set to zero, the basis function on the coarse levels would be represented by truncated basis functions. For all nodes which are in the active set, we set the corresponding entries of the residual or the prolongated correction to zero. While for the stiffness matrix, we set the rows and columns associated with the active set to be zero. Setting the entries of the stiffness matrix and residual vector to zero is equivalent to removing the nodal basis function associated with all nodes in the active set.

As we are using transfer operators constructed by the pseudo-L^2-projection, this multigrid method including the truncation process can be carried out algebraically. In comparison with the semi-geometric multigrid method for the linear system, this algorithm is computationally more expensive. This can be attributed to the cost of computing the orthogonal transformation of the matrix $\boldsymbol{B}^\top$ and then projecting the problem on a new basis. Even though the multigrid method is more expensive, the algorithm has optimal convergence properties. If we are solving an optimization problem with inequality constraints, the active set changes in a few initial multigrid iterations but once the active set of the solution is found, the algorithm converges linearly.

In Algorithm 5.3, we can see the detailed generalized multigrid algorithm for the finest level with the modified projected Gauss-Seidel smoother. The Algorithm 5.4 shows the coarse level cycle with any regular smoother. Here, we note that the algorithm is given in a more abstract setting for linear equality and/or inequality constraints, assuming the active set may change in each multigrid iteration. This is not the case for the problems with equality constraints, as we know the active set in advance and thus the truncated stiffness matrix does not change. The computational cost of the multigrid method, therefore, can be reduced for problems with the equality constraint. This can be achieved by invoking the setup phase of the semi-geometric algorithm (Algorithm 4.4) and precomputing the coarse level stiffness matrices.

5.4 Numerical Results

In this section, we evaluate the performance of the various solution schemes, discussed in this chapter, for solving the saddle point problem arising from the Lagrange multiplier discretization in the XFEM framework.

5.4.1 Comparison of Standard Iterative Methods

Here, we consider the fictitious domain problem, specifically Example 1-FD, where the method of Lagrange multiplier is used to enforce the Dirichlet boundary condition. The experiments in this section are carried out on the mesh hierarchy described in Table 2.1 on a superellipse domain. We compare the performance of three main strategies discussed in Section 5.2 i.e., Schur-complement method, Uzawa methods, and conjugate projected gradient method. In all of these methods, we employ the semi-geometric multigrid method for solving the primal problem. The semi-geometric multigrid method is set up as a $V(3,3)$-cycle with a symmetric Gauss-Seidel method as a smoother. The mesh hierarchy at different levels has different multigrid hierarchy associated with them. For example, the coarsest mesh in the multigrid hierarchy is kept the same for all the discretization levels $(L1,\dots,L5)$. Thus, when we are solving the problem on level $L1$, we have a two-grid method where the coarse level is created by uniform coarsening of the background mesh. While, for solving the problem on the level $L2$, we have 3-levels in the mesh hierarchy. This procedure is carried out for setting up the multigrid hierarchy on all levels. The termination criterion for solving the primal problem is given by the relative residual in the energy norm, thus

$$\frac{\|\boldsymbol{f}-\boldsymbol{A}\boldsymbol{u}^{(k)}\|_A}{\|\boldsymbol{f}-\boldsymbol{A}\boldsymbol{u}^{(0)}\|_A} < 10^{-10}.$$

Schur Complement Reduction

As we have mentioned, the Schur-complement system is of a smaller dimension. Since the Schur-complement matrix is symmetric positive definite, we can employ the Krylov subspace methods. Here, we use the conjugate gradient (CG) method and the preconditioned conjugate gradient (PCG) method with SIMPLE and FETI preconditioners. We assume that if the residual of the dual problem in the Euclidean norm is smaller than 10^{-13}, it satisfies the termination criterion. We prefer this measure, as the computation of the residual in the energy norm would require an explicit representation of the Schur-complement matrix. The termination criterion for the dual problem is chosen to be an order of magnitude smaller than for the primal problem because the approximation error of the dual variable clearly affects the approximation of the primal variable.

Table 5.1 illustrates the number of iterations required to reach the predefined tolerance by the iterative methods for solving the Schur-complement systems. It is evident from the table that the PCG method with the FETI preconditioner outperforms the SIMPLE preconditioner almost by a factor of 2 and the CG method

levels	CG		PCG-SIMPLE		PCG-FETI	
	# iter	(# MG iter)	# iter	(# MG iter)	# iter	(# MG iter)
$L1$	23	(102)	11	(54)	6	(34)
$L2$	23	(104)	10	(52)	5	(33)
$L3$	22	(100)	10	(52)	5	(36)
$L4$	19	(88)	9	(48)	5	(36)
$L5$	18	(89)	9	(53)	5	(38)

Table 5.1. The number of iterations required by CG and PCG methods to reach a predefined tolerance for solving the Schur Complement system arising from Example 1-FD. In bracket, we have the total number of SMG iterations for solving the primal problem.

levels	Uzawa CG		Uzawa PCG-SIMPLE		Uzawa PCG-FETI	
	# iter	(# MG iter)	# iter	(# MG iter)	# iter	(# MG iter)
$L1$	23	(97)	11	(49)	6	(29)
$L2$	23	(99)	10	(47)	5	(28)
$L3$	22	(95)	10	(47)	5	(31)
$L4$	19	(83)	9	(43)	5	(31)
$L5$	18	(83)	9	(47)	5	(32)

Table 5.2. The number of iterations required by Uzawa-CG and Uzawa-PCG methods to reach a predefined tolerance for solving the saddle point problem arising from Example 1-FD. In bracket, we have the total number of SMG iterations for solving the primal problem.

by a factor of 4. Even with the increasing problem size, the number of iterations required to solve the dual problem does not increase for all the methods. This can be attributed to the choice of the penalty parameter γ_s in the augmented system, as it stabilizes the condition number of the Schur-complement matrix regardless of the discretization levels. Similar behavior is also evident for the multigrid iterations, as the total number of the iterations used to solve the primal problem also does not increase with decreasing mesh size, thus we get the desirable level independent convergence.

Uzawa Methods

In this section, we evaluate the performance of the Uzawa methods. As the Uzawa methods are slow to converge in its original formulation, we use the

	$L1$	$L2$	$L3$	$L4$	$L5$
$\boldsymbol{P}_I^{\perp}$	282	527	1021	1909	3706
$\boldsymbol{P}_J^{\perp}$	268	512	1009	1994	3759
$\boldsymbol{P}_{MG}^{\perp}$	8	8	8	9	9

Table 5.3. The number of iterations required by the projected CG method (with various projection operators) to reach a predefined tolerance for solving the saddle point problem arising from Example 1-FD.

modified Uzawa conjugate gradient method (see Algorithm C.1) and Uzawa preconditioned conjugate gradient method (see Algorithm C.1) instead. Here, we assume that if the residual of the dual problem and primal problem in the Euclidean norm is smaller than 10^{-13}, it satisfies the termination criterion.

From Table 5.2, we can see that the number of iterations required by all variants of the Uzawa methods is identical to their Schur-complement counterparts. The only difference is in the number of total iterations of the semi-geometric multigrid method, which is due to the coupled nature of the Uzawa algorithm. The Uzawa method updates both the primal and the dual variables simultaneously. In contrast, for the Schur-complement method once we have a solution for the dual variable, we have to solve one more linear system of questions to achieve the solution of the primal variable.

Conjugate Projected Gradient Method

Here, we compare the performance of the CPG method with different projection operators. In the Algorithm 5.1, the termination criterion is set to be $\sqrt{\boldsymbol{r}^{\top}\boldsymbol{g}} < 10^{-16}$. We compare the different projection operators: $\boldsymbol{P}_I^{\perp}$ defined as (5.15), $\boldsymbol{P}_J^{\perp}$ defined as (5.16) and $\boldsymbol{P}_{MG}^{\perp}$ defined as (5.17). From Table 5.3, it is clear that the number of iterations required to solve the problem increases proportionally to the problem size. We also notice that employing the projection operator with the Jacobi preconditioner does not offer any significant improvement in the number of iterations and in some cases, the performance may even worsen. In this case, we have only performed one multigrid iteration in the computation of the projection operator. While using the projection operator $\boldsymbol{P}_{MG}^{\perp}$ is very effective as the number of iterations does not increase with the increasing problem size, even though, we have only performed one multigrid iteration. But, we note that using the multigrid method here is quite expensive, as we need to carry out 2 multigrid iterations per CPG iteration in addition to computing the action of inverse $\boldsymbol{A}_{\gamma}^{-1}\boldsymbol{B}^{\top}$.

5.4.2 Convergence of the Multigrid Method

Here, we consider the fictitious domain problem and interface problems for evaluating the performance of the generalized multigrid method. For these experiments, we consider Example 1-FD and Example 2-FD to test the fictitious domain problems. While for the interface problems, we consider Example 2-IF and Example 3-IF. In this set of experiments, we choose the values of the coefficient $\alpha_2 = 1$ and $\alpha_1 = \{0.1, 0.9\}$ for Example 2-IF. While for Example 3-IF, the coefficients are chosen as $\alpha_1 = 1$ and $\alpha_2 = \{1.1, 10\}$. For both of these examples, the hierarchy of the meshes is depicted in Table 2.1 and Table 3.1, where we have added an extra level $L0$ to the mesh hierarchy by uniformly coarsening the background mesh associated with level $L1$. For all the problems, the finest level mesh is defined by discretization on $L5$. Here, the termination criterion for the method is defined in the norm of corrections as

$$||\boldsymbol{u}^{(k+1)} - \boldsymbol{u}^{(k)}|| < 10^{-10}.$$

While, the asymptotic convergence rate of the iterative method is given as

$$\rho^* := \frac{||\boldsymbol{u}^{(k+1)} - \boldsymbol{u}^{(k)}||}{||\boldsymbol{u}^{(k)} - \boldsymbol{u}^{(k-1)}||},$$

where $\boldsymbol{u}^{(k+1)}$ satisfies the termination condition.

In the generalized multigrid method, it is required to compute the orthogonal transformation of the matrix $\boldsymbol{B}^{\top}$. We utilize the Givens rotation algorithm for performing the QR decomposition [GvL12]. The examples in this section are carried out with the implementation in the Utopia library [ZKN$^+$16]. On the finest level, we perform 3 pre-smoothing and 3 post-smoothing steps of the projected Gauss-Seidel method (Algorithm 5.2). While, we employ the symmetric Gauss-Seidel method as a smoother with 3 pre-smoothing and 3 post-smoothing steps on all other levels, except on the coarsest level. On the coarsest level, we use the direct solver from the MUMPS library [ADKL01].

Table 5.4 and Table 5.5 illustrate the number of iterations required by the generalized multigrid method to reach the predefined termination criterion for the fictitious domain method and the interface problem. From the tables, it is clear that the number of iterations does not change with the increasing number of levels. Thus, the convergence of the generalized multigrid methods does not depend on the number of levels in the multigrid hierarchy. In Table 5.4 and Table 5.5, we can also observe the asymptotic convergence rate of the multigrid method for different examples. Here, we observe that for the fictitious domain problems the asymptotic convergence rate is quite small ($\rho^* < 0.06$), which

# levels	Example 1-FD		Example 2-FD	
	# iter	(ρ^*)	# iter	(ρ^*)
2	5	(0.016)	6	(0.053)
3	5	(0.015)	6	(0.018)
4	5	(0.013)	6	(0.034)
5	5	(0.010)	6	(0.038)
6	5	(0.008)	6	(0.024)

Table 5.4. The number of iterations required by the generalized multigrid method to reach a predefined tolerance for solving the saddle point problem arising from the fictitious domain method.

# levels	Example 2-IF				Example 3-IF			
	$\alpha_1 = 0.1$ $\alpha_2 = 1$		$\alpha_1 = 0.9$ $\alpha_2 = 1$		$\alpha_1 = 1$ $\alpha_2 = 1.1$		$\alpha_1 = 1$ $\alpha_2 = 10$	
	# iter	(ρ^*)	# iter	(ρ^*)	# iter	(ρ^*)	# iter	(ρ^*)
2	8	(0.155)	8	(0.022)	8	(0.025)	8	(0.142)
3	8	(0.155)	8	(0.027)	8	(0.036)	8	(0.143)
4	8	(0.156)	8	(0.023)	8	(0.033)	8	(0.145)
5	8	(0.155)	8	(0.034)	8	(0.031)	8	(0.143)
6	8	(0.155)	8	(0.024)	8	(0.034)	8	(0.136)

Table 5.5. The number of iterations required by the generalized multigrid method to reach a predefined tolerance for solving the saddle point problem arising from the interface problem.

shows the robustness of our multigrid method. We remark that for Example 2-FD even though the number of iterations is exactly the same for all levels, the asymptotic convergence rate varies with the number of levels. While for the interface problems, we observe that the convergence rate is quite small ($\rho^* < 0.04$) when the coefficients on the subdomains have a smaller jump. The asymptotic convergence rate is larger ($\rho^* < 0.16$) when the coefficients are different by an order of magnitude.

From these experiments, we can conclude that the generalized multigrid method is a robust iterative method for solving the equality constraints problems arising from the Lagrange multiplier discretization of the constraints in the XFEM framework.

Chapter 6

Contact Problems

Contact phenomena are virtually ubiquitous in nature. From the modeling point of view, the contact problems are challenging to solve as the contact boundary is not known a priori. In this work, we present the contact problem in the unfitted finite element framework and propose to solve the arising linear system of equations with the novel generalized multigrid method. Here, we consider the frictionless contact problems, where we can neglect the tangential forces on the contact interfaces. The literature on the topic of contact mechanics is quite vast, here we provide a brief introduction to the frictionless contact problems, we refer to the monographs [Wri06, KO87, Lau13] for a detailed overview.

In Section 6.1, we provide a brief introduction to the linearized elasticity and discuss its weak formulation. Signorini's problem is presented in Section 6.2, where we describe the variational inequality formulation and its discretization within the XFEM framework. Section 6.3 discusses the two-body contact problem and its XFEM discretization. In both contact problems, we employ the vital vertex algorithm to discretize the non-penetration condition on the embedded contact boundary/interface. In the later sections, we discuss the generalized multigrid method from the previous chapter and propose to employ basis transformation before the algorithm can be invoked for solving the contact problems. In Section 6.5, we perform several experiments to demonstrate the robustness of the proposed multigrid method.

6.1 Linear Elasticity

In this section, we review the deformation of an elastic body under the influence of the external forces. We give a very compact introduction to the elasticity theory for the linear elastic material model. A detailed description of the concepts of continuum mechanics can be found in [Cia97, Hol00, BW08].

We assume a body $\Omega \in \mathbb{R}^d$, $d \in \{2,3\}$ with the Lipschitz continuous boundary $\partial\Omega$. The boundary $\partial\Omega$ is decomposed into two different parts: the Dirichlet boundary $\partial\Omega_D$ and the Neumann boundary $\partial\Omega_N$. The body is assumed to be subjected to volume forces $\boldsymbol{f} : \Omega \to \mathbb{R}^d$ and the traction/surface forces $\boldsymbol{t}_N : \partial\Omega_N \to \mathbb{R}^d$. As a result, under the influence of the external forces, the body undergoes deformation. A material point $\boldsymbol{x} \in \Omega$ in the undeformed state moves to the location $\boldsymbol{x} + \boldsymbol{u}$ after the deformation. Here, the vector-valued quantity $\boldsymbol{u} : \Omega \to \mathbb{R}^d$ describes displacement of the material point $\boldsymbol{x}$, thus $\boldsymbol{u} = \boldsymbol{u}(\boldsymbol{x})$.

In elastostatics, the equation of the equilibrium can be written as

$$\begin{aligned} \nabla \cdot \boldsymbol{\sigma} + \boldsymbol{f} &= 0 \qquad \text{in } \Omega, \\ \boldsymbol{u} &= 0 \quad \text{on } \partial\Omega_D, \\ \boldsymbol{\sigma}\boldsymbol{n} &= \boldsymbol{t}_N \quad \text{on } \partial\Omega_N, \end{aligned} \tag{6.1}$$

where, $\boldsymbol{\sigma} = \boldsymbol{\sigma}(\boldsymbol{u})$ is the Cauchy stress tensor $\boldsymbol{\sigma} = \boldsymbol{\sigma}^{\mathsf{T}}$, $\boldsymbol{f}$ is the body force per unit volume, $\boldsymbol{t}_N$ is the traction on the Neumann boundary, $\boldsymbol{n}$ denotes the outward normal, and the Dirichlet boundary has fixed displacements. We assume that the body Ω is linear elastic, where the constitutive law is provided by Hooke's law

$$\boldsymbol{\sigma} = \lambda \mathrm{tr}(\boldsymbol{\varepsilon})\boldsymbol{I} + 2\mu\boldsymbol{\varepsilon}, \tag{6.2}$$

where λ and μ are Lamé parameters, and $\mathrm{tr}(\cdot)$ denotes the trace operator. Next, we consider the case of small strain theory and neglect the non-linear terms from the finite strain tensor. Thus, the linearized strain tensor $\boldsymbol{\varepsilon} = \boldsymbol{\varepsilon}(\boldsymbol{u})$ is defined as

$$\boldsymbol{\varepsilon}(\boldsymbol{u}) := \frac{1}{2}\big(\nabla\boldsymbol{u} + (\nabla\boldsymbol{u})^{\mathsf{T}}\big). \tag{6.3}$$

The variational formulation of the problem (6.1) using the principle of virtual work is defined as

$$\text{find } \boldsymbol{u} \in \boldsymbol{H}^1_D(\Omega) \text{ such that} \qquad a(\boldsymbol{u},\boldsymbol{v}) = F(\boldsymbol{v}) \qquad \forall \boldsymbol{v} \in \boldsymbol{H}^1_D(\Omega), \tag{6.4}$$

where $a(\cdot,\cdot) : \boldsymbol{H}^1_D \times \boldsymbol{H}^1_D \to \mathbb{R}$ is as symmetric continuous coercive bilinear form and $F(\cdot) : \boldsymbol{H}^1_D \to \mathbb{R}$ denotes continuous linear form. The test functions are chosen in such a way that they vanish on the Dirichlet boundary, which leads to

$$\boldsymbol{H}^1_D(\Omega) := \{\boldsymbol{v} \in \boldsymbol{H}^1(\Omega) : \boldsymbol{v} = 0 \text{ on } \partial\Omega_D\}.$$

The bilinear and the linear forms are defined as

$$\begin{aligned} a(\boldsymbol{u},\boldsymbol{v}) &= (\boldsymbol{\sigma}(\boldsymbol{u}), \boldsymbol{\varepsilon}(\boldsymbol{v}))_{L^2(\Omega)} = \int_\Omega \boldsymbol{\sigma}(\boldsymbol{u}) : \boldsymbol{\varepsilon}(\boldsymbol{v}) d\boldsymbol{x}, \\ F(\boldsymbol{v}) &= (\boldsymbol{f},\boldsymbol{v})_{L^2(\Omega)} + \langle \boldsymbol{t}_N, \boldsymbol{v}\rangle_{\partial\Omega_N} = \int_\Omega \boldsymbol{f} \cdot \boldsymbol{v} d\boldsymbol{x} + \int_{\partial\Omega_N} \boldsymbol{t}_N \cdot \boldsymbol{v} d\boldsymbol{s}. \end{aligned} \tag{6.5}$$

In the above formulation, the test function $\boldsymbol{v}$ can be interpreted as virtual displacements. We consider this formulation as the foundation for the contact problems discussed in the next sections.

6.2 Signorini's Problem

In this section, we consider contact problems for the linear elastic body in terms of variational inequalities.

6.2.1 Problem Formulation

The Signorini's problem models the contact of a linear elastic body to a frictionless rigid foundation.

We assume the boundary of the domain $\partial\Omega$ is decomposed into three different parts, given as

$$\partial\Omega = \partial\Omega_D \cup \partial\Omega_N \cup \partial\Omega_C. \tag{6.6}$$

All these parts of the boundary are assumed to be disjoint and $\text{meas}(\partial\Omega_D) > 0$. The body is assumed to be clamped at the Dirichlet boundary, thus

$$\boldsymbol{u} = 0 \quad \text{on} \;\; \partial\Omega_D. \tag{6.7}$$

The Neumann or the traction boundary condition is defined as

$$\boldsymbol{\sigma n} = \boldsymbol{t}_N \quad \text{on} \;\; \partial\Omega_N, \tag{6.8}$$

where $\boldsymbol{n}$ is the outward normal to the surface and $\boldsymbol{t}_N$ denotes the surface forces or the pressure on the body.

The last remaining part of the boundary condition concerns the contact boundary. This part of the boundary may come in contact with the rigid foundation $\Sigma \in \mathbb{R}^d$. For brevity, the possible contact boundary is denoted as $\Gamma_C = \partial\Omega_C$. We assume that the initial gap function $g_c : \Gamma_C \to \mathbb{R}^+$ is given between the body and the rigid foundation in the outward normal direction $\boldsymbol{n}$. For a vector $\boldsymbol{v}$, we distinguish between its normal component $v_n = \boldsymbol{v} \cdot \boldsymbol{n}$ and the tangential component $\boldsymbol{v}_t = \boldsymbol{v} - v_n\boldsymbol{n}$ at the boundary. The non-penetration condition on the possible contact boundary Γ_C is given as

$$u_n \leqslant g_c \quad \text{on } \Gamma_C, \tag{6.9}$$

where $\boldsymbol{u} \cdot \boldsymbol{n} = u_n$ denotes the displacement in the normal direction. Similar to the displacement vector, we can also decompose the stress vector or traction at the contact boundary into the normal and tangential components, given as $\boldsymbol{\sigma n} = \sigma_n \cdot \boldsymbol{n} + \boldsymbol{\sigma}_t$. Here, $\sigma_n = \sigma_n(\boldsymbol{u})$ denotes the contact pressure developed in the normal direction on Γ_C. The contact stress is either compressive or it vanishes if the body is not in contact with the rigid foundation, which can be represented as

$$\sigma_n \leqslant 0 \quad \text{on } \Gamma_C, \tag{6.10}$$

where $\sigma_n = \boldsymbol{n} \cdot \boldsymbol{\sigma n} = \sigma_{ij} n_i n_j$. We are considering the frictionless contact problem, so the body is allowed to move freely in the tangential direction and the induced tangential stresses are given as

$$\boldsymbol{\sigma}_t = 0 \quad \text{on } \Gamma_C, \tag{6.11}$$

where $(\boldsymbol{\sigma}_t(\boldsymbol{u}))_j = \sigma_{ij} n_i - \sigma_n n_j$. Finally, the last contact condition is the complementarity condition, which forces the gap between the body and the rigid obstacle to be zero when non-zero contact pressure occurs and the contact pressure is zero when there is no contact, thus

$$(u_n - g_c)\sigma_n = 0 \quad \text{on } \Gamma_C.$$

Now, combining (6.9),(6.10),(6.11), we can summarize the frictionless contact conditions as

$$\left.\begin{aligned} u_n - g_c &\leqslant 0 \\ \sigma_n &\leqslant 0 \\ (u_n - g_c)\sigma_n &= 0 \\ \boldsymbol{\sigma}_t &= 0 \end{aligned}\right\} \text{on } \Gamma_C. \tag{6.12}$$

These conditions are known in contact mechanics as Hertz–Signorini–Moreau conditions for frictionless contact or Karush–Kuhn–Tucker (KKT) conditions of the constraints in optimization literature. The boundary value problem (6.1) with the above contact conditions is called the Signorini's problem. In Figure 6.1, we can see an example of a Signorini's contact problem.

Following the discussion of the linear elastic formulation, for brevity, we define space

$$\boldsymbol{\mathcal{V}} := \boldsymbol{H}^1_D = \{\boldsymbol{v} \in \boldsymbol{H}^1(\Omega) : \boldsymbol{v} = 0 \text{ on } \partial\Omega_D\}.$$

The virtual displacements $\boldsymbol{v}$ vanish on the Dirichlet boundary. But in order to provide the weak formulation of the Signorini's problem, we define admissible space such that the virtual displacements also satisfy the contact conditions. We define, $\boldsymbol{\mathcal{K}}$ as a set of admissible displacements with respect to the contact conditions (6.9), as a convex subset of $\boldsymbol{\mathcal{V}}$

$$\boldsymbol{\mathcal{K}} := \{\boldsymbol{v} \in \boldsymbol{\mathcal{V}} : v_n - g_c \leqslant 0 \text{ on } \Gamma_C\}. \tag{6.13}$$

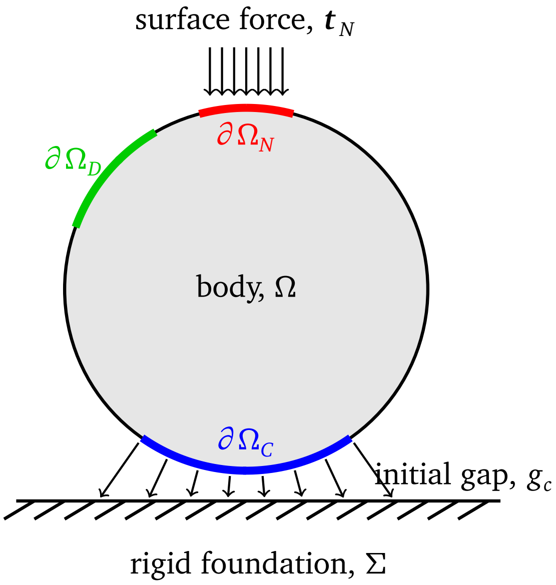


Figure 6.1. Signorini's problem.

Let $\boldsymbol{u} \in \mathcal{K}$ be the solution of the Signorini's problem, we can multiply equilibrium equation in (6.1) with the virtual displacements $\boldsymbol{v}-\boldsymbol{u} \in \mathcal{K}$. Then by performing the integration by parts we obtain

$$a(\boldsymbol{u},\boldsymbol{v}-\boldsymbol{u}) = F(\boldsymbol{v}-\boldsymbol{u}) + \langle \sigma_n, \boldsymbol{v}\cdot\boldsymbol{n} - \boldsymbol{u}\cdot\boldsymbol{n}\rangle_{\Gamma_C}, \tag{6.14}$$

where the bilinear form $a(\cdot,\cdot)$ and the linear form $F(\cdot)$ are defined as in (6.5). From the contact condition, we can deduce $\langle \sigma_n, \boldsymbol{v}\cdot\boldsymbol{n} - \boldsymbol{u}\cdot\boldsymbol{n}\rangle_{\Gamma_C} \geqslant 0$. By introducing this condition in the weak formulation, we obtain the variational inequality formulation of the Signorini's problem, given as

$$\text{find } \boldsymbol{u} \in \mathcal{K} \text{ such that} \quad a(\boldsymbol{u},\boldsymbol{v}-\boldsymbol{u}) \geqslant F(\boldsymbol{v}-\boldsymbol{u}) \qquad \forall \boldsymbol{v} \in \mathcal{K}. \tag{6.15}$$

Due to this inequality condition the contact problem is inherently non-linear even in case of linear elasticity.

The variational formulation (6.15) can also be written as an equivalent energy minimization problem

$$\text{find } \boldsymbol{u} \in \mathcal{K} \text{ such that} \quad J(\boldsymbol{u}) \leqslant J(\boldsymbol{v}) \qquad \forall \boldsymbol{v} \in \mathcal{K}, \tag{6.16}$$

for the quadratic energy functional $J(\boldsymbol{u}) := \frac{1}{2}a(\boldsymbol{u},\boldsymbol{u}) - F(\boldsymbol{u})$.

6.2.2 XFEM Discretization

The variational formulation of Signorini's problem (6.16) is still not discretized. In this section we discuss the discretization of the contact problem in the unfitted finite element framework. For simplicity we assume that only the contact boundary is unfitted, while Dirichlet and Neumann boundaries are fitted in the mesh.

We consider the quadrilateral background mesh $\widetilde{\mathcal{T}}_h$ that encapsulates the domain Ω. We define the low order FE space over the background mesh as

$$\widetilde{\boldsymbol{\mathcal{V}}}_h = \{\boldsymbol{v} \in \boldsymbol{H}^1(\widetilde{\mathcal{T}}_h) : \boldsymbol{v}|_K \in \mathcal{Q}_1(K), v|_{(\partial \widetilde{\mathcal{T}}_h)_D} = 0, \forall K \in \widetilde{\mathcal{T}}_h\},$$

where $\mathcal{Q}_1$ denotes the space of piecewise bilinear functions. Following the discussion from Section 2.2, we create an active mesh $\mathcal{T}_h$ and then define the extended FE space $\boldsymbol{\mathcal{V}}_h$ using the characteristic function on the active mesh. Here, the basis function on this FE space $\boldsymbol{\mathcal{V}}_h$ has support only up to the boundary $\partial\Omega$ of the domain.

In the XFEM framework, the discretized Signorini's problem can be written as a constrained minimization problem with inequality constraints given as, find $\boldsymbol{u}_h \in \boldsymbol{\mathcal{V}}_h$ such that

$$\begin{aligned} \min_{\boldsymbol{u}_h \in \boldsymbol{\mathcal{V}}_h} J(\boldsymbol{u}_h) &= \frac{1}{2} A_s(\boldsymbol{u}_h, \boldsymbol{u}_h) - F(\boldsymbol{u}_h) \\ \text{subject to} \quad \boldsymbol{u}_h \cdot \boldsymbol{n} &\leqslant g_c \qquad \text{on } \Gamma_C, \end{aligned} \tag{6.17}$$

where the bilinear form $A_s(\cdot,\cdot)$ is defined as

$$A_s(\boldsymbol{u}_h, \boldsymbol{v}_h) = a(\boldsymbol{u}_h, \boldsymbol{v}_h) + g(\boldsymbol{u}_h, \boldsymbol{v}_h),$$

where $a(\cdot,\cdot)$ and $F(\cdot)$ are defined as in (6.5). We have included the ghost penalty term in the bilinear form $A_s(\cdot,\cdot)$ to improve the stability of the XFEM discretization [HLL17]. The ghost penalty term is enforced on the set of edges $\mathcal{G}_{h,\Gamma_C}$, and it is defined as

$$g(\boldsymbol{u}_h, \boldsymbol{v}_h) = \sum_{G \in \mathcal{G}_{h,\Gamma_C}} \epsilon_G h_G ([\![\nabla_{\boldsymbol{n}_G} \mathcal{E}_h \boldsymbol{u}_h]\!], [\![\nabla_{\boldsymbol{n}_G} \mathcal{E}_h \boldsymbol{v}_h]\!])_{L^2(G)}.$$

This ghost penalty term is enforced in the normal derivatives of the displacement field, while we could also impose the ghost penalty term in the normal derivatives of the stress field [CK18]. Here, we have discretized the bilinear form and the linear form, but we still have to discretize the contact condition.

Traditionally, in the fitted finite element framework, we can use the method of Lagrange multipliers, the penalty method, Nitsche's method, the regularization methods, the augmented Lagrangian methods, etc. to impose the contact conditions [Woh11]. Here, we employ the method of Lagrange multipliers to enforce the contact conditions, rather than Nitsche's method. This is due to the fact that the Lagrange multiplier formulation does not require modification of the primal formulation, and the contact condition can be handled by the Lagrange multipliers implicitly. Whereas, Nitsche's formulation for the contact problem is more complicated as we have to handle the inequality conditions in the primal formulation. Nitsche's method for the contact problems in the fitted finite element method is implemented and analysed in [WZ07, GSV20, CHLR19, CHR14]. In the context of the CutFEM method, the contact problems are considered in [CK18].

We impose the contact condition by using the method of Lagrange multipliers, where the multiplier space is constructed by employing the vital vertex algorithm (Section 2.6). We introduce the multiplier space $\mathcal{M}_h \subseteq H^{-\frac{1}{2}}(\Gamma_C)$ and define the bilinear form $b(\cdot,\cdot) : \mathcal{M}_h \times \boldsymbol{\mathcal{V}}_h \to \mathbb{R}$ by

$$b(\mu_h, \boldsymbol{u}_h) := \sum_{K \in \mathcal{T}_{h,\Gamma_c}} \langle \mu_h, \boldsymbol{u}_h \cdot \boldsymbol{n} \rangle_{\Gamma_K} \quad \forall \mu_h \in \mathcal{M}_h, \forall \boldsymbol{u}_h \in \boldsymbol{\mathcal{V}}_h, \tag{6.18}$$

and the linear form $G_C(\cdot) : \mathcal{M}_h \to \mathbb{R}$ by

$$G_C(\mu_h) := \sum_{K \in \mathcal{T}_{h,\Gamma_c}} \langle \mu, g_c \rangle_{\Gamma_K} \quad \forall \mu_h \in \mathcal{M}_h.$$

Thus, after the discretization of the non-penetration condition, we obtain reformulated Signorini's contact problem as, find $\boldsymbol{u}_h \in \boldsymbol{\mathcal{V}}_h$ such that

$$\begin{aligned} &\min_{\boldsymbol{u}_h \in \boldsymbol{\mathcal{V}}_h} J(\boldsymbol{u}_h) = \frac{1}{2} A_s(\boldsymbol{u}_h, \boldsymbol{u}_h) - F(\boldsymbol{u}_h) \\ &\text{subject to} \quad b(\mu_h, \boldsymbol{u}_h) \leqslant G_C(\mu_h) \quad \forall \mu_h \in \mathcal{M}_h. \end{aligned} \tag{6.19}$$

6.3 Two-body Contact Problem

In this section, we discuss more general the frictionless contact problem between two-bodies.

6.3.1 Problem Formulation

Generally, in two body problems master-slave concept is used, where one body is denoted as a slave body, and the other body is denoted as a master body. The contact condition is imposed as a non-penetration condition between the slave surface and the master surface. In the two-body problem, we seek the displacement field $\boldsymbol{u}^i$ of bodies, Ω^i where $i = m, s$, where symbols m, s denote the master and the slave side, respectively. As discussed in the previous section, the boundary of the domain is represented as the union of disjoint boundaries, where we assume both bodies consist of Dirichlet boundary condition with a nonzero measure. In this case, we consider the linear elasticity formulation for the two-body problem. The equilibrium conditions for the displacement $\boldsymbol{u} := (\boldsymbol{u}^m, \boldsymbol{u}^s)$ are given as

$$\begin{aligned} \nabla\boldsymbol{\sigma} + \boldsymbol{f} = 0 &\qquad \text{in } \Omega := \Omega^m \cup \Omega^s, \\ \boldsymbol{u} = \boldsymbol{u}_D &\qquad \text{on } \partial\Omega_D := \partial\Omega^m_D \cup \partial\Omega^s_D, \\ \boldsymbol{\sigma}\cdot\boldsymbol{n} = \boldsymbol{t}_N &\qquad \text{on } \partial\Omega_N := \partial\Omega^m_N \cup \partial\Omega^s_D. \end{aligned} \tag{6.20}$$

The contact boundary condition is shared by the boundary on both sides which may come in contact with each other. As both bodies share the same contact boundary, we have

$$\Gamma_C = \partial\Omega_C = \partial\Omega^m_C = \partial\Omega^s_C.$$

We assume that on the contact boundary the outward normal is smooth such that, $\boldsymbol{n} = \boldsymbol{n}^m = -\boldsymbol{n}^s$. We define the point-wise gap in the displacement fields of both domains as

$$[\![\boldsymbol{u}\cdot\boldsymbol{n}]\!] := \boldsymbol{u}^m\cdot\boldsymbol{n}^m + \boldsymbol{u}^s\cdot\boldsymbol{n}^s = (\boldsymbol{u}^m - \boldsymbol{u}^s)\cdot\boldsymbol{n}^m,$$

where the normal $\boldsymbol{n}$ is defined as the outward normal to Ω^m. A simple sketch of the two-body contact problem can be seen in Figure 6.2.

The contact pressure or stress developed in the normal direction on Γ_C is compressive and equal on the contact boundary on both surfaces. The gap function g_c is defined as the distance from the possible contact boundary of the master body to the slave body in the normal direction. Thus, the contact conditions are given as

$$\left.\begin{aligned} [\![\boldsymbol{u}\cdot\boldsymbol{n}]\!] - g_c &\leqslant 0 \\ \sigma_n &\leqslant 0 \\ ([\![\boldsymbol{u}\cdot\boldsymbol{n}]\!] - g_c)\sigma_n &= 0 \\ \boldsymbol{\sigma}_t &= 0 \end{aligned}\right\} \text{on } \Gamma_C. \tag{6.21}$$

The weak formulation of the above two body problem is analogous to Signorini's problem. For the two body contact problem, the function space is defined as

$$\boldsymbol{\mathcal{V}} := \{(\boldsymbol{v}^m, \boldsymbol{v}^s) = \boldsymbol{v} \in \boldsymbol{H}^1(\Omega) : \boldsymbol{v} = 0 \text{ on } \partial\Omega^m_D \cup \partial\Omega^s_D\}.$$

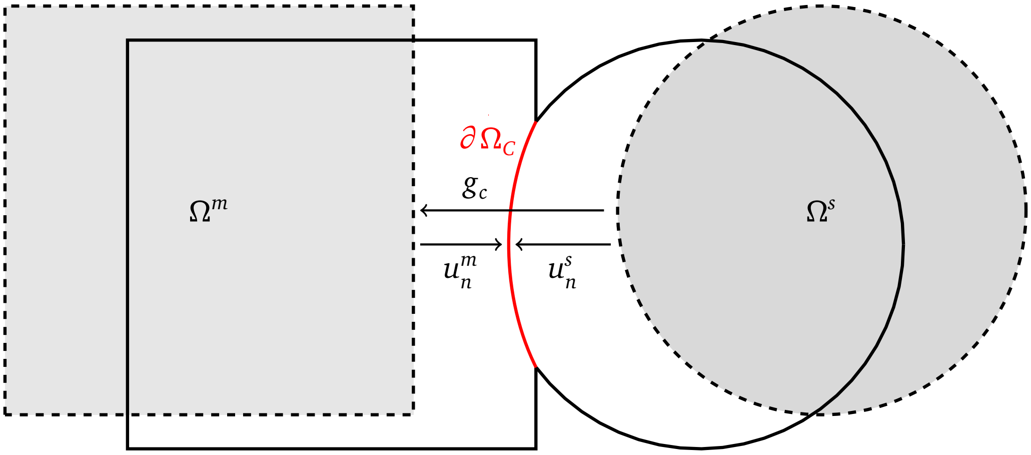


Figure 6.2. Two body contact problem.

We seek the solution in the admissible space with respect to the contact condition (6.21). The admissible space $\mathcal{K}$ is a convex subset of the space $\mathcal{V}$ defined as

$$\mathcal{K} := \{\boldsymbol{v} \in \mathcal{V} : [\![\boldsymbol{v} \cdot \boldsymbol{n}]\!] \leqslant g_c \text{ on } \Gamma_C\}. \tag{6.22}$$

The weak formulation of the two-body contact problem can also be formulated as a constrained minimization problem, given as (6.16). The bilinear and the linear forms for the two-body contact problem are given as

$$\begin{aligned} a_t(\boldsymbol{u},\boldsymbol{v}) &= \sum_{i=m,s} (\boldsymbol{\sigma}(\boldsymbol{u}^i), \boldsymbol{\varepsilon}(\boldsymbol{v}^i))_{L^2(\Omega)} = \sum_{i=m,s} \int_\Omega \boldsymbol{\sigma}(\boldsymbol{u}^i) : \boldsymbol{\varepsilon}(\boldsymbol{v}^i) d\boldsymbol{x}, \\ F_t(\boldsymbol{v}) &= \sum_{i=m,s} \Big((\boldsymbol{f}, \boldsymbol{v}^i)_{L^2(\Omega)} + \langle \boldsymbol{t}_N, \boldsymbol{v}^i \rangle_{\partial\Omega_N} \Big) \\ &= \sum_{i=m,s} \Big(\int_\Omega \boldsymbol{f} \cdot \boldsymbol{v}^i d\boldsymbol{x} + \int_{\partial\Omega_N} \boldsymbol{t}_N \cdot \boldsymbol{v}^i d\boldsymbol{s} \Big). \end{aligned} \tag{6.23}$$

Here, the bilinear and the linear form are denoted with a subscript t to distinguish between their counterparts used for the Signorini's problem.

6.3.2 XFEM Discretization

As we have a variational formulation of the two-body contact problem, in this section we discretize the problem in the XFEM framework. Here, we assume the only contact boundary Γ_C is unfitted, while the Dirichlet and Neumann boundaries are fitted.

We assume the quadrilateral background mesh $\widetilde{\mathcal{T}}_h$ that encapsulates both domains Ω^m and Ω^s. For simplicity, we assume that these two bodies are already in contact, and the interface between these two bodies is considered as contact boundary Γ_C. On the background mesh $\widetilde{\mathcal{T}}_h$, we define a low order FE space with the piecewise quadrilateral functions that vanish on the Dirichlet boundaries $\widetilde{\boldsymbol{\mathcal{V}}}_h \subset \boldsymbol{H}^1_D(\Omega)$. Now, following the discussion from Section 3.2, we create an active mesh associated with each domain by decomposing and then enriching the background mesh. Afterward, we exploit the definition of the Heaviside function (3.2) and obtain the FE space defined on each domain as $\boldsymbol{\mathcal{V}}_h^m$ and $\boldsymbol{\mathcal{V}}_h^s$. The basis functions of these FE spaces are defined only up to the boundary of the respective domain. We define the FE space $\boldsymbol{\mathcal{V}}_h$ on both domains Ω^m and Ω^s as $\boldsymbol{\mathcal{V}}_h = \boldsymbol{\mathcal{V}}_h^m \oplus \boldsymbol{\mathcal{V}}_h^s$.

The discretized two-body contact problem in the XFEM framework can be written as, find $\boldsymbol{u}_h := (\boldsymbol{u}_h^m, \boldsymbol{u}_h^s) \in \boldsymbol{\mathcal{V}}_h$ such that

$$\begin{aligned} &\min_{\boldsymbol{u}_h \in \boldsymbol{\mathcal{V}}_h} J(\boldsymbol{u}_h) = \frac{1}{2} A_t(\boldsymbol{u}_h, \boldsymbol{u}_h) - F_t(\boldsymbol{u}_h) \\ &\text{subject to} \quad [\![\boldsymbol{u}_h \cdot \boldsymbol{n}]\!] \leqslant g_c \quad \text{on } \Gamma_C. \end{aligned} \tag{6.24}$$

The bilinear form $A_t(\cdot,\cdot)$ is defined as

$$A_t(\boldsymbol{u}_h, \boldsymbol{v}_h) = a_t(\boldsymbol{u}_h, \boldsymbol{v}_h) + g_t(\boldsymbol{u}_h, \boldsymbol{v}_h) \tag{6.25}$$

where, the bilinear form $a_t(\cdot,\cdot)$ is defined as in (6.23) and the bilinear form $g_t(\cdot,\cdot)$ denotes the ghost penalty term. The ghost penalty term is enforced in the neighborhood of all the elements that are intersected by the contact boundary on both master and slave domains, also see Section 3.4.3. The ghost penalty term is defined as

$$g_t(\boldsymbol{u}_h, \boldsymbol{u}_h) = \sum_{i=m,s} \sum_{G \in \mathcal{G}^i_{h,\Gamma_C}} \epsilon_G h_G ([\![\nabla_{\boldsymbol{n}_G} \mathcal{E}_h \boldsymbol{u}_h^i]\!], [\![\nabla_{\boldsymbol{n}_G} \mathcal{E}_h \boldsymbol{v}_h^i]\!])_{L^2(G)},$$

where, ϵ_G is a penalty parameter and h_G is the diameter of face G. Similarly to the last section, we also discretize the non-penetration condition by means of the Lagrange multipliers. Here, we can define the multiplier space on the contact boundary given either on the master domain or the slave domain, as the discretized representation of the contact boundary is identical on both sides. We utilize again the definition of the multiplier space as $\mathcal{M}_h \subseteq H^{-\frac{1}{2}}(\Gamma_C)$. The multiplier space is discretized using the vital vertex algorithm, discussed in Section 2.6. Thus, the discretized two-body contact problem is given as, find $\boldsymbol{u}_h \in \boldsymbol{\mathcal{V}}_h$ such that

$$\begin{aligned} &\min_{\boldsymbol{u}_h \in \boldsymbol{\mathcal{V}}_h} J(\boldsymbol{u}_h) = \frac{1}{2} A_t(\boldsymbol{u}_h, \boldsymbol{u}_h) - F_t(\boldsymbol{u}_h) \\ &\text{subject to} \quad b_t(\mu_h, \boldsymbol{u}_h) \leqslant G_C(\mu_h) \quad \forall \mu_h \in \mathcal{M}_h. \end{aligned} \tag{6.26}$$

Here, the bilinear form $b_t(\cdot,\cdot)$ is defined as

$$b_t(\mu_h, \boldsymbol{u}_h) := \sum_{K \in \mathcal{T}_{h,\Gamma_c}} \langle \mu_h, [\![\boldsymbol{u} \cdot \boldsymbol{n}]\!] \rangle_{\Gamma_C} \quad \forall \mu_h \in \mathcal{M}_h, \forall \boldsymbol{u}_h \in \boldsymbol{\mathcal{V}}_h. \tag{6.27}$$

6.4 Generalized Multigrid Method for Contact Problems

The solution of the contact problem requires us to solve a quadratic optimization problem with linear inequality constraints. As these types of problems arise in many applications in practice, there have been some efforts to develop the solution strategies to tackle such problems.

One of the simple approaches is to first convert the problem in its dual formulation and then solve the constrained minimization problem with the point-wise constraints. Here, the Lagrange multipliers are interpreted as contact pressure, and from the contact conditions (6.12) and (6.21), we know the constraints on the Lagrange multiplier is always point-wise. This approach circumvents the need for tackling the linear inequality constraints, by imposing a simpler set of constraints. Once the dual problem is solved, we can compute the displacement field [Dos09]. As an alternative, we can employ the semi-smooth Newton or active set methods to solve such problems, see [HKK04, HW10, PWGW12]. In addition to these methods, the interior-point methods are also an attractive option, see [Wri97, NW00, KMNv13]. These methods are quite expensive, as in one form or another they require inner solutions schemes to solve linear problems, while the outer iteration schemes are used for tackling the constraints.

In this section, we propose to employ the generalized multigrid method, introduced in Section 5.3, for solving the contact problems arising from the XFEM discretization, where the non-penetration conditions are discretized using the method of Lagrange multipliers. Ideally, the only essential modification required in the generalized multigrid method is updating the projected Gauss-Seidel method to handle the inequality constraints. But, to improve the numerical stability of the generalized multigrid method, we perform an additional local basis transformation.

6.4.1 Basis Transformation

We note, the non-penetration constraints in the contact problem are given by the relative displacement of the bodies in the normal direction. Thus, the constraint at any node is given by the coupling of the DOFs on the node. In order to create the constraint matrix such that the non-penetration condition is enforced only on one DOF per node, we transform the system into a new basis.

Let $\{\boldsymbol{e}_i\}_{i=1,\dots,d}$ be the Euclidean basis of $\mathbb{R}^d$ and $\boldsymbol{n}_p$ denotes the outward normal on the node p. Recall the definition of the nodal sets on the mesh, we define the set of nodes as $\mathcal{N}_{h,\Gamma_C} = \{p \in \mathcal{N}_h : \phi_q|_{\Gamma_C} \neq 0\}$. On each node $p \in \mathcal{N}_{h,\Gamma_C}$, we redefine the Euclidean basis $\boldsymbol{e}_1(p) = \boldsymbol{n}_p$ and also change $\{\boldsymbol{e}_i\}_{i=2,\dots,d}$ such that these redefined basis are orthonormal, while for all $q \in \mathcal{N}_h \setminus \mathcal{N}_{h,\Gamma_C}$, the definition of the Euclidean basis remain same. This approach was introduced for Signorini's problems in [Kra01] and later applied to multi-body contact problems in [DK09, Kra09, WK03]. The transformed basis can be constructed by using local Householder transformation on $\mathbb{R}^d$. We can compute the local Householder transformation as

$$\boldsymbol{H}_{pp} = \boldsymbol{I} - 2(\boldsymbol{v}_p \otimes \boldsymbol{v}_p) \qquad \forall p \in \mathcal{N}_{h,\Gamma_C},$$

where the vector $\boldsymbol{v}_p$ is computed by $\boldsymbol{v}_p = (\boldsymbol{n}_p - \boldsymbol{e}_1)/\|\boldsymbol{n}_p - \boldsymbol{e}_1\|_2$. While for all $p \in \mathcal{N}_h \setminus \mathcal{N}_{h,\Gamma_C}$, we define $\boldsymbol{H}_{pp} = \boldsymbol{I}$. Thus, by using these local transformation matrices, we can construct global matrix $\boldsymbol{H} \in \mathbb{R}^{nd \times nd}$ where $\boldsymbol{H} = \oplus_{p \in \mathcal{N}_h} H_{pp}$, which is an orthonormal matrix with the properties, $\boldsymbol{H} = \boldsymbol{H}^\top = \boldsymbol{H}^{-1}$.

This transformation decouples and locally modifies the constraints such that the constraint matrix $\boldsymbol{B}$ would be sparser in the new basis. In the generalized multigrid method, we employ the QR transformation to decouple the constraints further. Hence the sparse matrix $\boldsymbol{B}$ will create a sparser orthogonal matrix $\boldsymbol{Q}$ and upper triangular matrix $\boldsymbol{R}$. The bilinear form (6.18) for the Signorini's problem can be reformulated as

$$b(\mu_h, \boldsymbol{u}_h) := \sum_{K \in \mathcal{T}_{h,\Gamma_C}} \langle \mu_h, \boldsymbol{u}_h \cdot \boldsymbol{e}_1 \rangle_{\Gamma_K} \quad \forall \mu_h \in \mathcal{M}_h, \forall \boldsymbol{u}_h \in \boldsymbol{\mathcal{V}}_h,$$

and the corresponding bilinear form (6.27) for the two-body contact problem can be given as

$$b_t(\mu_h, \boldsymbol{u}_h) := \sum_{K \in \mathcal{T}_{h,\Gamma_C}} \langle \mu_h, [\![\boldsymbol{u} \cdot \boldsymbol{e}_1]\!] \rangle_{\Gamma_K} \quad \forall \mu_h \in \mathcal{M}_h, \forall \boldsymbol{u}_h \in \boldsymbol{\mathcal{V}}_h.$$

Thus, the entries of the constraint matrix $\boldsymbol{B}$ for Signorini's problem are given as

$$b_{pq} = b(\mu_p, \phi_q \cdot \boldsymbol{e}_1) \qquad \forall p \in \mathcal{V}_{h,\Gamma_C}, \forall q \in \mathcal{N}_h,$$

where $\mathcal{V}_{h,\Gamma_C}$ denotes the set of vital vertices, and by μ_p and ϕ_q we denote the basis functions associated with nodes p,q in the FE space $\mathcal{M}_h$ and $\boldsymbol{\mathcal{V}}_h$, respectively. While for the two body problem the entries of the constraint matrix are given as

$$\begin{aligned}(b_{pq})_t &= b(\mu_p, \phi_q \cdot \boldsymbol{e}_1) \qquad && \forall p \in \mathcal{V}_{h,\Gamma_C}, \forall q \in \mathcal{N}_h^m,\\ (b_{pr})_t &= -b(\mu_p, \phi_r \cdot \boldsymbol{e}_1) \qquad && \forall p \in \mathcal{V}_{h,\Gamma_C}, \forall r \in \mathcal{N}_h^s,\end{aligned}$$

where $\mathcal{N}_h^m$ and $\mathcal{N}_h^s$ denote the set of node of the active meshes associated with master and slave bodies, respectively.

As we have changed the definition of the Euclidean basis, we have to also modify the primal problem. The matrix $\boldsymbol{H}$ can be used to transform the variables into the new basis as $\overline{\boldsymbol{u}} = \boldsymbol{H}\boldsymbol{u}$ and $\boldsymbol{u} = \boldsymbol{H}\overline{\boldsymbol{u}}$, where $\boldsymbol{u}$ denotes the unknown displacements. Similarly, the stiffness matrix $\boldsymbol{A}$ and the right hand side $\boldsymbol{f}$ in the new basis can be given as $\overline{\boldsymbol{A}} = \boldsymbol{HAH}$ and $\overline{\boldsymbol{f}} = \boldsymbol{Hf}$. The algebraic formulation of the contact problem in the new basis system is given as find $\overline{\boldsymbol{u}} \in \mathbb{R}^{d\times n}$ such that

$$\begin{aligned}&\min_{\overline{\boldsymbol{u}}\in\mathbb{R}^{d\times n}} J(\overline{\boldsymbol{u}}) = \frac{1}{2}\overline{\boldsymbol{u}}^\top\overline{\boldsymbol{A}}\overline{\boldsymbol{u}} - \overline{\boldsymbol{u}}^\top\overline{\boldsymbol{f}}\\ &\text{subject to} \quad \boldsymbol{B}\overline{\boldsymbol{u}} \leqslant \boldsymbol{g}_c,\end{aligned} \tag{6.28}$$

where, $\boldsymbol{g}_c$ denotes the gap between the body and the rigid foundation on the contact boundary or the gap between the master and the slave body on the contact boundary. As we are going to use the multigrid method, we have to also modify the prolongation operator associated with the finest level, given as

$$\overline{\boldsymbol{T}}_{L-1}^{L} = \boldsymbol{H}\boldsymbol{T}_{L-1}^{L}.$$

It is well-known that round-off errors play a vital role in the field of numerical computing and affect the accuracy and stability of the algorithms. We recall, before employing the projected Gauss-Seidel method, we have to perform QR decomposition to decouple the linear constraints. Performing the QR decomposition of the constraint matrix $\boldsymbol{B}$ can give rise to the round-off errors in the computation of the orthogonal matrix $\boldsymbol{Q}$ and upper triangular matrix $\boldsymbol{R}$. The round-off errors also accumulate when repeatedly multiplying matrices and vectors to the matrix $\boldsymbol{Q}$. Thus, by performing local basis transformation, we reduce the sparsity of the matrix $\boldsymbol{B}$. If the matrix $\boldsymbol{B}$ is sparser, it gives rise to a sparser matrix $\boldsymbol{Q}$ and the matrix $\boldsymbol{R}$, and in this way we can reduce the accumulation of the round-off errors.

6.4.2 The Multigrid Method

As mentioned earlier, once we have the updated problem (6.28) after the local basis transformation we still perform the QR transformation described in Section 5.3.1. Thus, after computation of the matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$ from the constraint matrix $\boldsymbol{B}$, we have to project the problem onto new basis. The orthonormal matrix is used to define the variables in the new basis as $\widetilde{\boldsymbol{u}} = \boldsymbol{Q}^\mathsf{T}\overline{\boldsymbol{u}} = \boldsymbol{Q}^\mathsf{T}\boldsymbol{H}\boldsymbol{u}$ and $\boldsymbol{u} = \boldsymbol{H}\overline{\boldsymbol{u}} = \boldsymbol{H}\boldsymbol{Q}\widetilde{\boldsymbol{u}}$. By incorporating the transformed matrices and the vectors, we can reformulate the constrained minimization problem (6.28) as

$$\begin{aligned} &\min_{\widetilde{u}} J(\widetilde{\boldsymbol{u}}) = \frac{1}{2}\widetilde{\boldsymbol{u}}^\mathsf{T}\widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{u}} - \widetilde{\boldsymbol{u}}^\mathsf{T}\widetilde{\boldsymbol{f}} \\ &\text{subject to} \quad \boldsymbol{R}^\mathsf{T}\widetilde{\boldsymbol{u}} \leqslant \boldsymbol{g}_c, \end{aligned} \tag{6.29}$$

where $\widetilde{\boldsymbol{A}} = \boldsymbol{Q}^\mathsf{T}\overline{\boldsymbol{A}}\boldsymbol{Q} = \boldsymbol{Q}^\mathsf{T}\boldsymbol{H}\boldsymbol{A}\boldsymbol{H}\boldsymbol{Q}$ and $\widetilde{\boldsymbol{f}} = \boldsymbol{Q}^\mathsf{T}\overline{\boldsymbol{f}} = \boldsymbol{Q}^\mathsf{T}\boldsymbol{H}\boldsymbol{f}$. In practice, the matrices $\widetilde{\boldsymbol{A}}$, $\overline{\boldsymbol{A}}$, and $\boldsymbol{A}$ have the same spectral properties, but they have different sparsity pattern. The new constraint matrix $\boldsymbol{R}$ has been decoupled such that we can employ the projected Gauss-Seidel method (Algorithm 5.2) as the finest level smoother in the multigrid method.

Now, we define a constrained subspace or a feasible set as

$$\widetilde{\mathcal{K}} = \{\widetilde{\boldsymbol{u}} \in \mathbb{R}^{d\times n} : \boldsymbol{R}^\mathsf{T}\widetilde{\boldsymbol{u}} \leqslant \boldsymbol{g}_C\}.$$

We pose our problem as an energy minimization problem (6.29) in the algebraic formulation:

$$\text{find } \widetilde{\boldsymbol{u}} \in \widetilde{\mathcal{K}} \text{ such that} \qquad J(\widetilde{\boldsymbol{u}}) \leqslant J(\widetilde{\boldsymbol{v}}) \qquad \forall \widetilde{\boldsymbol{v}} \in \widetilde{\mathcal{K}}.$$

As discussed in Section 5.3.2, the projected Gauss-Seidel method minimizes energy functional such that the intermediate iterates also remain in the feasible set $\widetilde{\mathcal{K}}$. In order to utilize the projected Gauss-Seidel method, we have to modify the upper bound and the lower bounds in (5.21) given as, $\boldsymbol{lb} = -\infty$ and $\boldsymbol{ub} = \boldsymbol{g}_C$. This also influences the active nodes, the nodes where the constraints are binding, which are given as

$$\mathscr{A} := \{p : (\boldsymbol{R}\widetilde{\boldsymbol{u}})_p = (\boldsymbol{g}_C)_p\}.$$

Based on this active set, we perform the truncation of the stiffness matrix, residual and correction vectors on the finest level. The last modification is required in the prolongation operator associated with the finest level. The updated transfer operator is given as

$$\widetilde{\boldsymbol{T}}_{L-1}^{L} = \boldsymbol{Q}^\mathsf{T}\overline{\boldsymbol{T}}_{L-1}^{L} = \boldsymbol{Q}^\mathsf{T}\boldsymbol{H}\boldsymbol{T}_{L-1}^{L}.$$

Thus, after these modifications, we can call the generalized multigrid method from Algorithm 5.3 for solving the contact problem (6.29).

6.5 Numerical Results

In this section, we evaluate the performance of the proposed generalized multigrid method for Signorini's problem and two-body contact problem. The non-penetration condition at the contact interface is discretized using the method of Lagrange multipliers in the unfitted finite element framework.

6.5.1 Signorini's Problem

In this section, we describe the problem setup for Signorini's problem for two different types of rigid obstacles.

Problem Description

All experiments in this section are carried out on a structured background mesh with the quadrilateral elements. The background mesh is defined on a coarsest level, on a rectangle of dimension $[-1.09, 1.09] \times [0, 1.09]$, with the elements 100 in X-direction and 50 in Y-direction, denoted as mesh on level $L1$. By uniformly refining this mesh, we obtain a hierarchy of meshes up to the level $L5$. In this experiment, we consider a semicircular domain, where the contact boundary of the domain is defined by a zero level set of a function $\Lambda_s(\boldsymbol{x}) := r^2 - \|\boldsymbol{x} - \boldsymbol{c}\|_2^2$ with radius $r = 0.9$, and $\boldsymbol{c}$ is the center of the circle $(0, 1)$. The domain Ω is defined by the region where the value of the level set is positive, $\Lambda_s > 0$. The Dirichlet boundary for this problem is fitted on the background mesh, and the Dirichlet boundary condition is defined as $\boldsymbol{u} = 0$ on $x = [-1.09, 1.09]$ and $y = 0$. The body force for this example is considered to be zero. Also, the method of the Lagrange multiplier utilizes the vital vertex algorithm to discretize the contact boundary. In addition, we are employing the ghost-penalty stabilization term in the bilinear form, with the parameter $\epsilon_G = 0.1$. In these experiments, the material parameters are chosen as Young's modulus $E = 10$ and Poisson's ratio $\nu = 0.3$. We can compute Lamé parameters λ and μ using the following relation:

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

Example 1-SC For this example, we consider a rigid foundation Σ defined by a line, given as $y = 0.12$. Thus the body Ω would be pressed against the rigid foundation, and the maximum magnitude of the displacement on the body would be given as $u = 0.02$. The setup of this example is depicted in Figure 6.3a, where we observe the magnitude of the displacement field due to the contact with a rigid foundation.

Example 2-SC This example considers a more complex rigid foundation, which does not satisfy the small strain theory, as we are enforcing a large displacement field. But this experiment is carried out to test the robustness of the multigrid method with respect to the shape of a rigid foundation. Here, we define a circular rigid foundation, defined by a circular level set function $\Lambda_o(\boldsymbol{x}) := r_o^2 - \|\boldsymbol{x} - \boldsymbol{c}_o\|_2^2$, with radius $r_o = 0.9$, and the center $\boldsymbol{c}_o = (0, -0.35)$. In Figure 6.3b, we can observe the setup and the magnitude of the displacement field due to contact with the circular rigid foundation.

Performance of the Multigrid method

Following the generalized multigrid method in the previous chapter, we are using the Givens rotation algorithm to compute the orthogonal transformation of the matrix $\boldsymbol{B}^\top$. The examples in this section are carried out in the Utopia library. For these experiments choose correction in energy norm as a termination criterion, given as

$$\|\boldsymbol{u}^{(k+1)} - \boldsymbol{u}^{(k)}\|_A < 10^{-10}. \tag{6.30}$$

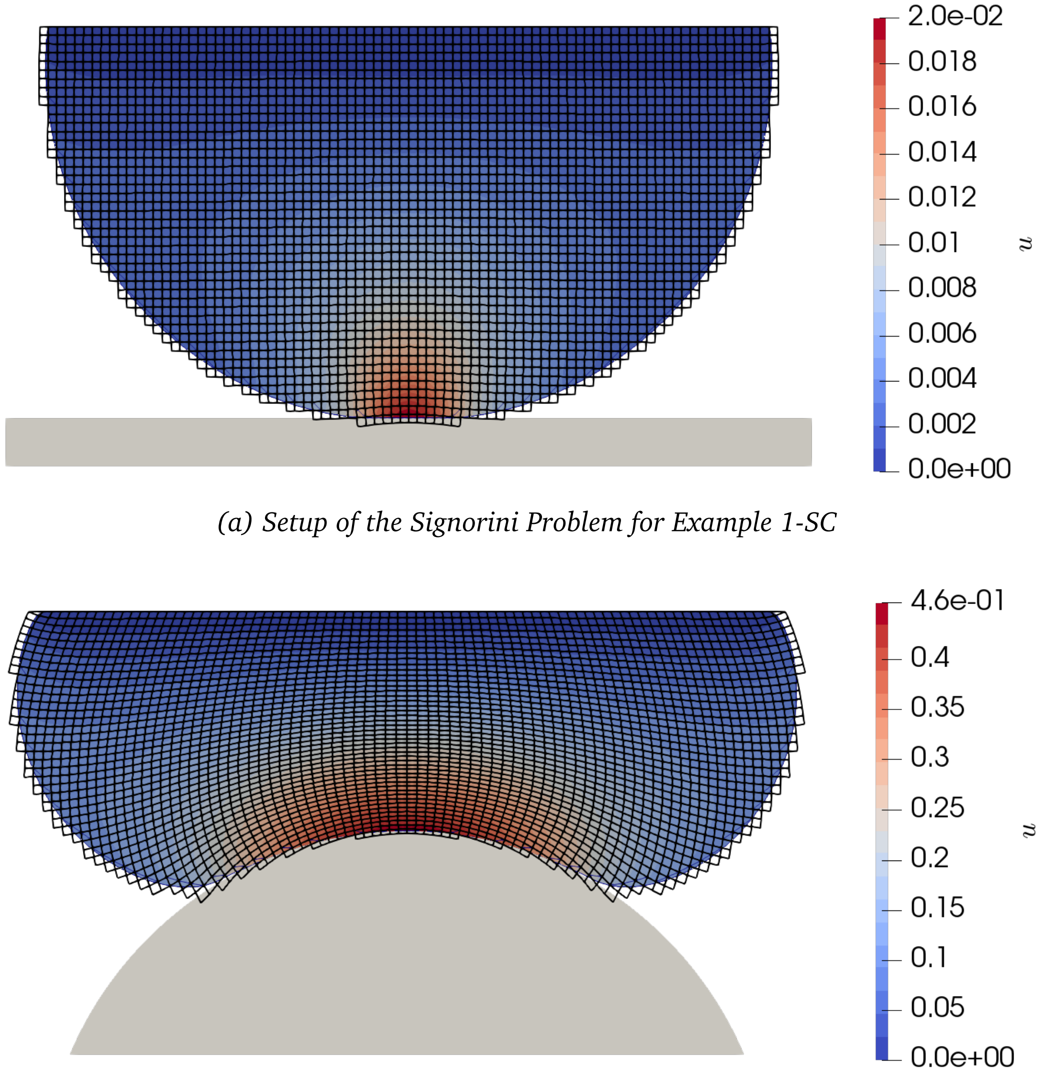
Also, we define the asymptotic convergence rate of an iterative solver as

$$\rho^* := \frac{\|\boldsymbol{u}^{(k+1)} - \boldsymbol{u}^{(k)}\|_A}{\|\boldsymbol{u}^{(k)} - \boldsymbol{u}^{(k-1)}\|_A},$$

where the iterate $\boldsymbol{u}^{(k+1)}$ satisfies the termination criterion (6.30). Here, all the experiments are carried out on the discretization $L5$, with around 2.5×10^6 DOFs, and in Figure 6.4 we can observe the resultant displacement field and the stress

# levels	Example 1-SC		Example 2-SC	
	# iter	(ρ^*)	# iter	(ρ^*)
2	12	(0.070)	13	(0.077)
3	12	(0.070)	13	(0.076)
4	12	(0.070)	13	(0.074)
5	12	(0.070)	13	(0.073)

Table 6.1. The number of iterations required by the generalized multigrid method to reach a predefined tolerance for solving Signorini's problems (with two different kind of obstacles).



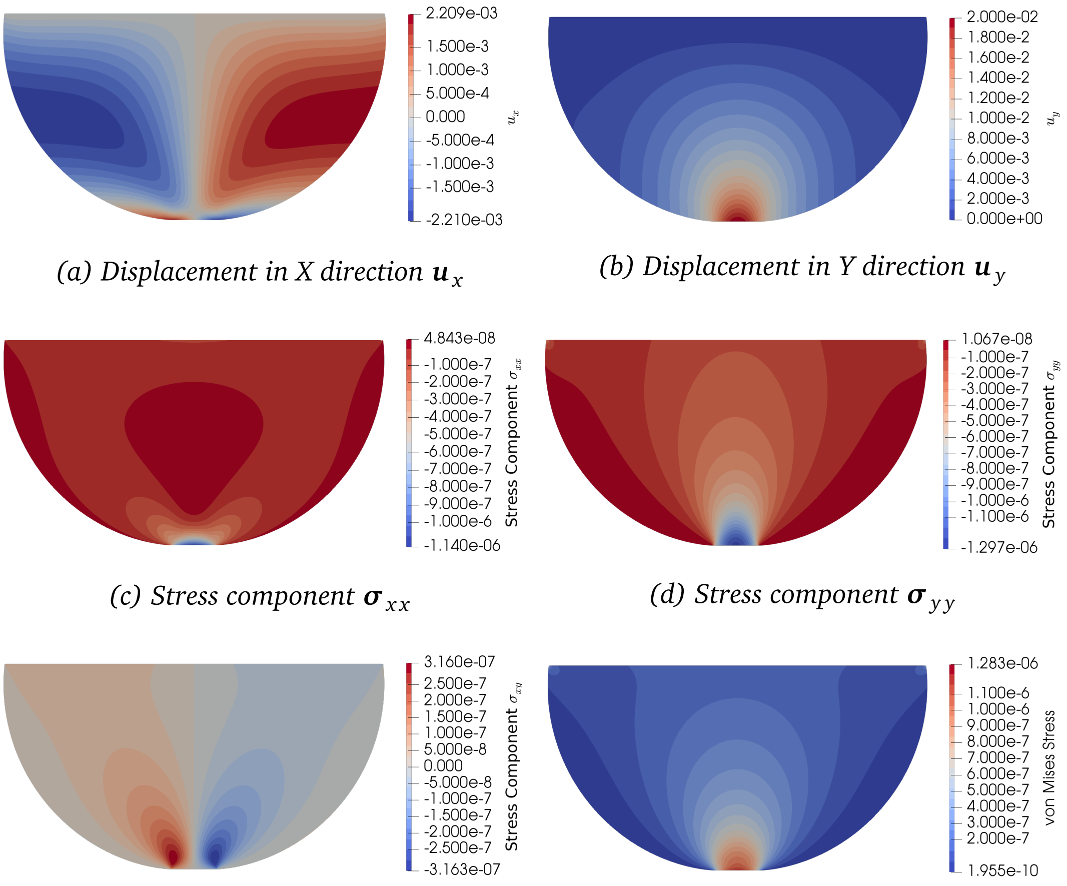
(a) Setup of the Signorini Problem for Example 1-SC

(b) Setup of the Signorini Problem for Example 2-SC

Figure 6.3. Setup of the Signorini's problem for experiments, the object in the gray scale is the rigid obstacle. We can see the active background mesh and the displacement field.

field. On the finest level, we perform 3 pre-smoothing and 3 post-smoothing steps of the modified projected Gauss-Seidel method. While, we employ the symmetric Gauss-Seidel method as a smoother with 3 pre-smoothing and 3 post-smoothing steps on all other levels, except on the coarsest level.

Table 6.1 illustrates the number of iterations required by the generalized



(a) Displacement in X direction $\boldsymbol{u}_x$

(b) Displacement in Y direction $\boldsymbol{u}_y$

(c) Stress component $\boldsymbol{\sigma}_{xx}$

(d) Stress component $\boldsymbol{\sigma}_{yy}$

(e) Stress component $\boldsymbol{\sigma}_{xy}$

(f) von Mises Stress

Figure 6.4. Resultant displacement field and stress field of Signorini's problem Example 1-SC.

multigrid method to reach the termination criterion (6.30). From the table, it is evident that the number of iterations does not change with the increasing number of levels in the multigrid hierarchy. For Example 2-SC we have a considerably complex rigid obstacle than for Example 1-SC, but the multigrid method converges with almost equal number of iterations for both examples. Also in Table 6.1, we can observe the asymptotic convergence rate of the multigrid method. Here, we can observe that the asymptotic convergence rate for both of these examples is quite small ($\rho^* < 0.1$).

6.5.2 Two-body Contact Problem

For the two-body contact problem, we consider two different types of embedded interfaces: a circular interface and an elliptical interface.

Problem Description

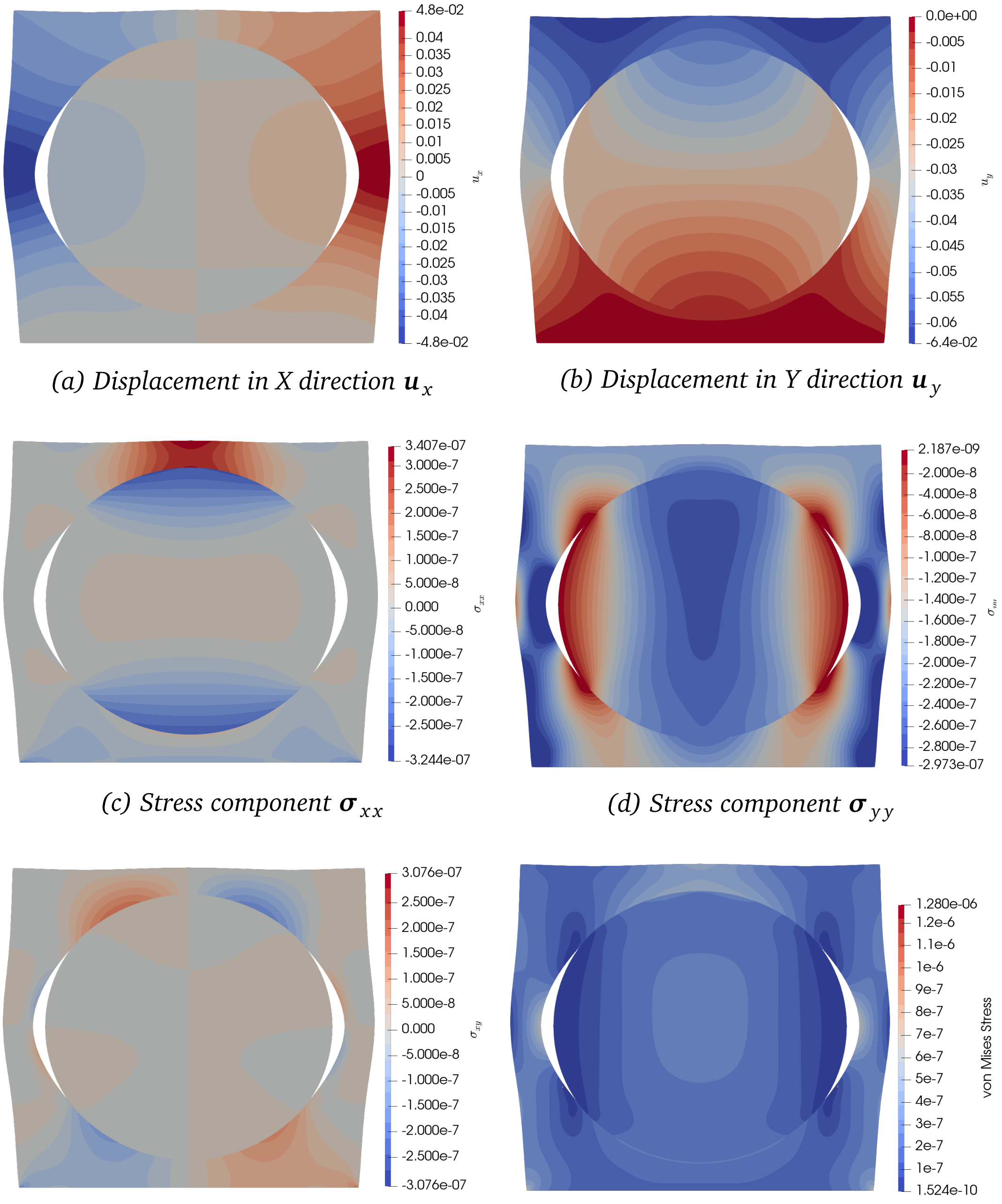
The experiments in this section are carried out on a structured background mesh with quadrilateral elements. Both bodies are considered to be in contact with each other in the absence of external body force or the Neumann boundary conditions. The background mesh is given in domain $\Omega = \Omega_1 \cup \Omega_2$ in $[0,1]^2$. We start with 100 elements in each direction, this mesh is denoted by $L1$. We create a hierarchy of the meshes by uniformly refining this mesh until we have 1600 elements in each direction. The sequence of the meshes is given by, $L1, \ldots, L5$. The Dirichlet boundary conditions is defined as $\boldsymbol{u} = 0$ on $x = [0,1]$ and $y = 0$, while the Neumann boundary condition is defined as $\boldsymbol{\sigma n} = (0,5)$ on $x = [0,1]$ and $y = 1$. The body force for this example is considered to be zero.

Example 1-TC For this example, we consider a circular contact interface denoted as Γ_c. The circular interface is defined as a zero level set of a function $\Lambda_c(\boldsymbol{x}) := r_0^2 - \|\boldsymbol{x} - \boldsymbol{c}\|_2^2$, with radius $r_0^2 = 3 - 2^{1/2}$, and $\boldsymbol{c}$ is the center of the circle, chosen as $(0.5, 0.5)$. The circular interface decomposes the domain Ω into Ω_1 where $\Lambda_c(\boldsymbol{x}) > 0$ and Ω_2 where $\Lambda_c(\boldsymbol{x}) < 0$. For this example, we consider two different sets of material parameters. We choose Young's modulus as $E_1 = 10$ and $E_2 = \{10, 50\}$ and the Poisson's ratio is chosen as $\nu = \nu_1 = \nu_2 = 0.3$.

Example 2-TC This example considers an elliptical contact interface denoted as Γ_e. The interface is defined as a zero level set of a function

$$\Lambda_e(\boldsymbol{x}) := r_1^2 - \left|\frac{x - c_x}{a}\right|^2 - \left|\frac{y - c_y}{b}\right|^2.$$

Here, r_1 denotes the radius of the ellipse, chosen as $r_1^2 = 2(3 - 2^{1/2})$. The symbols a and b denote the major and minor axis of the ellipse, chosen as $a = 1, b = 0.8$. Here the center of the circle is chosen as $(0.5, 0.5)$. The circular interface decomposes the domain Ω into Ω_1, where $\Lambda_e(\boldsymbol{x}) > 0$ and Ω_2 where $\Lambda_e(\boldsymbol{x}) < 0$. For this example, we consider the same set of material parameters, as used in the previous example. Young's modulus is chosen as $E_1 = 10$ and $E_2 = \{10, 50\}$ and the Poisson's ratio is chosen as $\nu = \nu_1 = \nu_2 = 0.3$.



(a) Displacement in X direction $\boldsymbol{u}_x$

(b) Displacement in Y direction $\boldsymbol{u}_y$

(c) Stress component $\boldsymbol{\sigma}_{xx}$

(d) Stress component $\boldsymbol{\sigma}_{yy}$

(e) Stress component $\boldsymbol{\sigma}_{xy}$

(f) von Mises Stress

Figure 6.5. Resultant displacement field and stress field, as a solution of the two-body contact problem, Example 1-TC, with Young's modulus $E_1 = E_2 = 10$*, where the domain* Ω_2 *is the circle.*

# levels	Example 1-TC				Example 2-TC			
	$E_1 = 10, \nu_1 = 0.3$ $E_2 = 10, \nu_2 = 0.3$		$E_1 = 10, \nu_1 = 0.3$ $E_2 = 50, \nu_2 = 0.3$		$E_1 = 10, \nu_1 = 0.3$ $E_2 = 10, \nu_2 = 0.3$		$E_1 = 10, \nu_1 = 0.3$ $E_2 = 50, \nu_2 = 0.3$	
	# iter	(ρ^*)	# iter	(ρ^*)	# iter	(ρ^*)	# iter	(ρ^*)
2	14	(0.071)	16	(0.159)	14	(0.075)	14	(0.153)
3	14	(0.073)	16	(0.224)	14	(0.077)	14	(0.151)
4	14	(0.075)	16	(0.271)	14	(0.078)	15	(0.138)
5	14	(0.073)	16	(0.171)	14	(0.079)	15	(0.139)

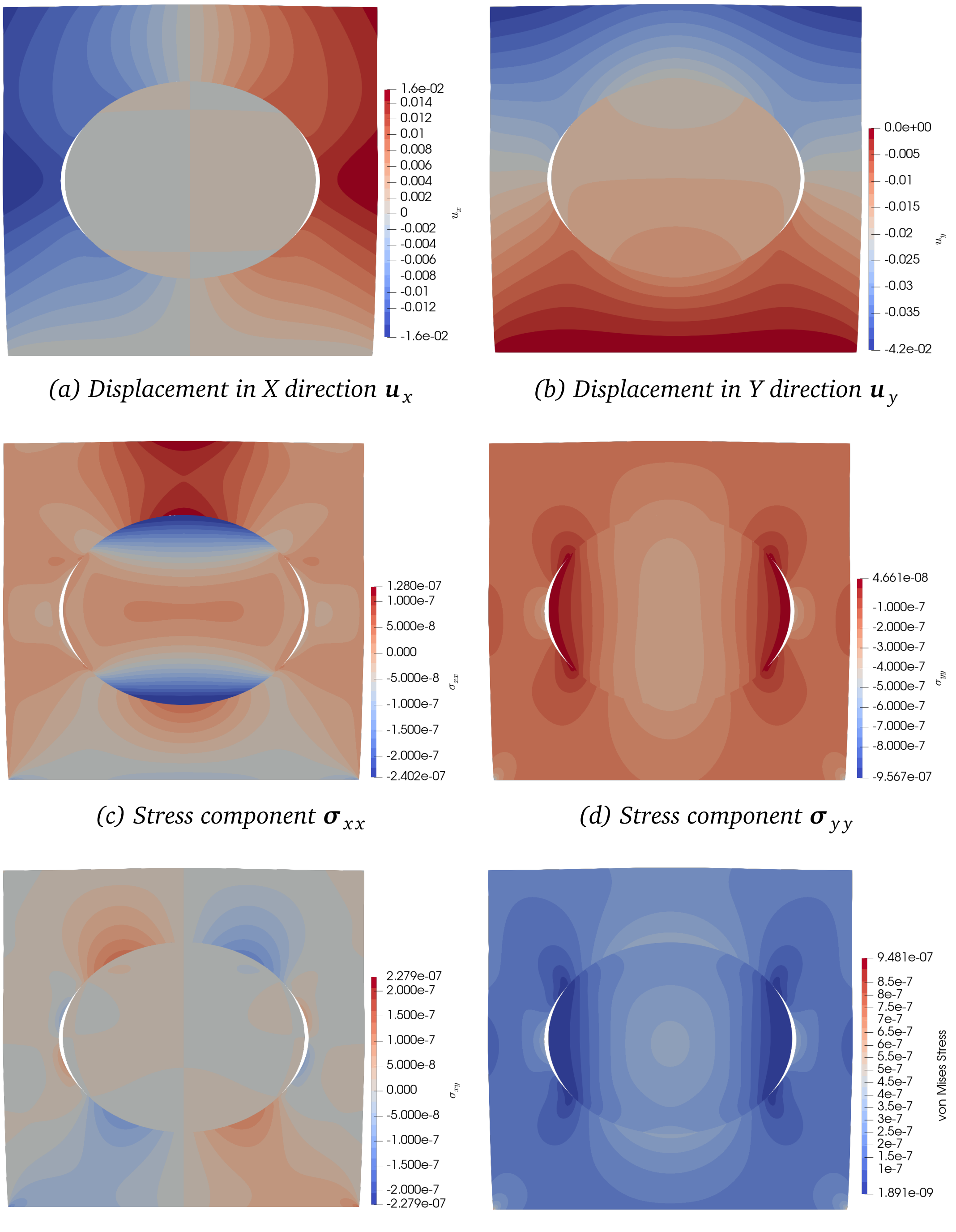
Table 6.2. The number of iterations required by the generalized multigrid method to reach a predefined tolerance for solving two-body contact problems.

Performance of the Multigrid method

Here, all the experiments are carried out on the discretization $L5$, with around 5.1×10^6 DOFs. In Figure 6.5 we can observe the resultant displacement field and the stress field for two-body problem with circular interface Example 1-TC with $E_1 = E_2 = 10$. While in Figure 6.6, we observe the result of the two-body contact problem with an elliptic interface Example 2-TC with $E_1 = 50$ and $E_2 = 10$. As a solution method, we employ the multigrid method with V-cycle and 3 pre-smoothing and 3 post-smoothing steps, with the projected Gauss-Seidel on the finest level and SSOR method on the coarser levels.

Table 6.2 shows the number of iterations required by the generalized multigrid method to reach the termination criterion (6.30). We can conclude from the table, that the number of iterations does not change with an increasing number of levels in the multigrid hierarchy. Also, in Table 6.1, we can observe the asymptotic convergence rate of the multigrid method. Here, we can see that even though the number of iterations required to reach the termination criterion is almost the same, we can see the difference in the asymptotic converge rates. For the case with an equal value of Young's modulus, the asymptotic convergence rate is quite low ($\rho^* < 0.01$). While, for the case with different values of Young's modulus, the asymptotic convergence rate is much larger ($\rho^* < 0.3$).

Thus, we can conclude the proposed generalized multigrid method is robust with respect to the number of levels, the material parameters, type of obstacle or the shape of the interface.



(a) Displacement in X direction $\boldsymbol{u}_x$

(b) Displacement in Y direction $\boldsymbol{u}_y$

(c) Stress component $\boldsymbol{\sigma}_{xx}$

(d) Stress component $\boldsymbol{\sigma}_{yy}$

(e) Stress component $\boldsymbol{\sigma}_{xy}$

(f) von Mises Stress

Figure 6.6. Resultant displacement field and stress field, as a solution of the two-body contact problem, Example 2-TC, with Young's modulus $E_1 = 50$ *and* $E_2 = 10$*, where the domain* Ω_2 *is the ellipse.*

Chapter 7

Conclusion

7.1 Summary

In this thesis, we reviewed several strategies for enforcing the boundary conditions and the interface conditions in the context of the unfitted finite element framework. We applied the ghost penalty method in order to overcome the ill-conditioning of the linear systems arising from the XFEM discretizations. We also reviewed two different strategies to implicitly estimate the value of the stabilization parameter in Nitsche's formulation. Later, we discussed the vital vertex algorithm for constructing a stable multiplier space that satisfies the discrete inf-sup condition for the method of Lagrange multipliers. We numerically compared the stability of these methods for continuous and highly varying coefficients in terms of discretization error and condition numbers of the arising linear systems.

We introduced a semi-geometric multigrid (SMG) method for solving linear problems arising from Nitsche's method in the unfitted finite element framework. We presented the pseudo-L^2-projection approach to construct the transfer operator for the XFEM discretization. This novel transfer operator was designed in such a way that it induces a hierarchy of the nested finite element spaces from the hierarchy of the background meshes. In the series of numerical experiments, we demonstrated the robustness of our SMG method equipped with the tailored transfer operator. We showed that the proposed multigrid method has level independent convergence rates, and it is robust with respect to highly varying coefficients and the number of interfaces in a domain. Further, we also employed the SMG method for solving the primal system in the saddle point problem, where the method also proved to be robust.

Later, we introduced the generalized multigrid method for solving the quadratic minimization problems with linear equality/inequality constraints. In the unfitted framework, these types of problems arise when the method of Lagrange multipliers is used for enforcing the boundary/interface conditions. Our generalized multigrid method also employed the pseudo-L^2-projection approach for computing the transfer operator. In order to handle the linear constraints, we proposed a decoupling technique that projects the constraints in new basis. This task was carried out by QR decomposition of the constraint matrix. Further, we introduced a variant of the projected Gauss-Seidel method to handle such decoupled constraints. In the numerical experiments, we demonstrated the robustness of the novel generalized multigrid method. The proposed multigrid exhibited optimal convergence properties when applied to the problems with equality constraints, stemming from imposing the boundary/interface conditions. This multigrid method was later also applied to the contact problems in the XFEM framework, where we also observed optimal convergence rates.

7.2 Outlook

The multigrid methods proposed in this work can be used for many other unfitted finite element discretization. In the future, we aim to extend the generalized multigrid method for contact problems with friction. Besides, we also aim to use the multigrid method for more complex coupled problems, such as fluid-structure interaction problems, thermoelastic contact problems, crack propagation problems in fracture mechanics, etc.

All the implementation and numerical experiments in this thesis were carried out in Matlab. As future work, we would like to extend this implementation to parallel architectures in order to handle large-scale problems. As a first step, we aim to port the implementation of the unfitted discretizations using the DMDA features of the PETSc library [BAA+20, BGMS97], as they already provide a scalable-parallel data structures for handling Cartesian structured meshes. As a second step, we aim to implement the L^2-projections based transfer operator for the XFEM discretization in the ParMOONoLith library [Zul16, KZ16], which can already compute the L^2-projection on the complex geometries on distributed computing architecture. Lastly, we plan to port the multigrid methods, along with the smoothers proposed in this thesis, to the Utopia library [ZKN+16].

Appendix A

Coercivity in Nitsche's Formulation

A.1 Fictitious Domain Problem

The coercivity of the bilinear form (2.21) is given as

$$
\begin{aligned}
A_N(v_h, v_h) &= \|\alpha^{\frac{1}{2}} \nabla v_h\|^2_{L^2(\Omega)} - 2\langle \alpha \nabla_{\boldsymbol{n}} v_h, v_h \rangle_\Gamma + \|\gamma_p^{\frac{1}{2}} v_h\|^2_{H^{\frac{1}{2}}(\Gamma),h} + g(v_h, v_h) \\
&\geqslant \|\alpha^{\frac{1}{2}} \nabla v_h\|^2_{L^2(\Omega)} - 2\|\alpha \nabla_{\boldsymbol{n}} v_h\|_{H^{\frac{1}{2}}(\Gamma),h} \|v_h\|_{H^{-\frac{1}{2}}(\Gamma),h} + \|\gamma_p^{\frac{1}{2}} v_h\|^2_{H^{\frac{1}{2}}(\Gamma),h} + g(v_h, v_h) \\
&\geqslant \|\alpha^{\frac{1}{2}} \nabla v_h\|^2_{L^2(\Omega)} - \frac{1}{\epsilon}\|\alpha \nabla_{\boldsymbol{n}} v_h\|^2_{H^{-\frac{1}{2}}(\Gamma),h} - \epsilon\|v_h\|^2_{H^{\frac{1}{2}}(\Gamma),h} + \gamma_p\|v_h\|^2_{H^{\frac{1}{2}}(\Gamma),h} + g(v_h, v_h) \\
&= \|\alpha^{\frac{1}{2}} \nabla v_h\|^2_{L^2(\Omega)} + \left(\frac{1}{\epsilon} - \frac{2}{\epsilon}\right)\|\alpha \nabla_{\boldsymbol{n}} v_h\|^2_{H^{-\frac{1}{2}}(\Gamma),h} + (\gamma_p - \epsilon)\|v_h\|^2_{H^{\frac{1}{2}}(\Gamma),h} + g(v_h, v_h) \\
&\geqslant \|\alpha^{\frac{1}{2}} \nabla v_h\|^2_{L^2(\Omega)} + \frac{1}{\epsilon}\|\alpha \nabla_{\boldsymbol{n}} v_h\|^2_{H^{-\frac{1}{2}}(\Gamma),h} - \sum_{K \in \mathcal{T}_{h,\Gamma}} \left(\frac{2C_\gamma}{\epsilon}\right)\|\alpha^{\frac{1}{2}} \nabla v_h\|^2_{L^2(K_\Omega)} \\
&\qquad + (\gamma_p - \epsilon)\|v_h\|^2_{H^{\frac{1}{2}}(\Gamma),h} + g(v_h, v_h) \\
&= \sum_{K \in \mathcal{T}_h \setminus \mathcal{T}_{h,\Gamma}} \|\alpha^{\frac{1}{2}} \nabla v_h\|^2_{L^2(K)} + \sum_{K \in \mathcal{T}_{h,\Gamma}} \|\alpha^{\frac{1}{2}} \nabla v_h\|^2_{L^2(K_\Omega)} + \frac{1}{\epsilon}\|\alpha \nabla_{\boldsymbol{n}} v_h\|^2_{H^{-\frac{1}{2}}(\Gamma),h} \\
&\qquad - \sum_{K \in \mathcal{T}_{h,\Gamma}} \left(\frac{2C_\gamma}{\epsilon}\right)\|\alpha^{\frac{1}{2}} \nabla v_h\|^2_{L^2(K_\Omega)} + (\gamma_p - \epsilon)\|v_h\|^2_{H^{\frac{1}{2}}(\Gamma),h} + g(v_h, v_h) \\
&= \sum_{K \in \mathcal{T}_h \setminus \mathcal{T}_{h,\Gamma}} \|\alpha^{\frac{1}{2}} \nabla v_h\|^2_{L^2(K)} + \frac{1}{\epsilon}\|\alpha \nabla_{\boldsymbol{n}} v_h\|^2_{H^{-\frac{1}{2}}(\Gamma),h} + (\gamma_p - \epsilon)\|v_h\|^2_{H^{\frac{1}{2}}(\Gamma),h} \\
&\qquad + \sum_{K \in \mathcal{T}_{h,\Gamma}} \left(1 - \frac{2C_\gamma}{\epsilon}\right)\|\alpha^{\frac{1}{2}} \nabla v_h\|^2_{L^2(K_\Omega)} + g(v_h, v_h).
\end{aligned}
$$

Here, the second line uses the Cauchy-Schwarz inequality (2.8) on the duality paring on the boundary. The third line utilizes Young's inequality for some $\epsilon > 0$ and the fifth line follows from the trace inequality (2.9).

A.2 Interface Problem

The coercivity of the bilinear form (3.8) for the interface problem is given as

$$
\begin{aligned}
A_N(v_h, v_h) &= \sum_{i=1}^{2} \|\alpha^{\frac{1}{2}}\nabla v_h\|^2_{L^2(\Omega_i)} - 2\langle \{\!\{\alpha\nabla_{\boldsymbol{n}} v_h\}\!\}, [\![v_h]\!]\rangle_\Gamma + \|\gamma_p^{\frac{1}{2}}[\![v_h]\!]\|^2_{H^{\frac{1}{2}}(\Gamma),h} \\
&\geqslant \sum_{i=1}^{2} \|\alpha^{\frac{1}{2}}\nabla v_h\|^2_{L^2(\Omega_i)} - 2\|\{\!\{\alpha\nabla_{\boldsymbol{n}} v_h\}\!\}\|_{H^{\frac{1}{2}}(\Gamma),h}\|[\![v_h]\!]\|_{H^{-\frac{1}{2}}(\Gamma),h} + \|\gamma_p^{\frac{1}{2}}[\![v_h]\!]\|^2_{H^{\frac{1}{2}}(\Gamma),h} \\
&\geqslant \sum_{i=1}^{2} \|\alpha^{\frac{1}{2}}\nabla v_h\|^2_{L^2(\Omega_i)} - \frac{1}{\epsilon}\|\{\!\{\alpha\nabla_{\boldsymbol{n}} v_h\}\!\}\|^2_{H^{-\frac{1}{2}}(\Gamma),h} - \epsilon\|[\![v_h]\!]\|^2_{H^{\frac{1}{2}}(\Gamma),h} + \gamma_p\|[\![v_h]\!]\|^2_{H^{\frac{1}{2}}(\Gamma),h} \\
&= \sum_{i=1}^{2} \|\alpha^{\frac{1}{2}}\nabla v_h\|^2_{L^2(\Omega_i)} + \Big(\frac{1}{\epsilon} - \frac{2}{\epsilon}\Big)\|\{\!\{\alpha\nabla_{\boldsymbol{n}} v_h\}\!\}\|^2_{H^{-\frac{1}{2}}(\Gamma),h} + (\gamma_p - \epsilon)\|[\![v_h]\!]\|^2_{H^{\frac{1}{2}}(\Gamma),h} \\
&\geqslant \sum_{i=1}^{2} \|\alpha^{\frac{1}{2}}\nabla v_h\|^2_{L^2(\Omega_i)} + \frac{1}{\epsilon}\|\{\!\{\alpha\nabla_{\boldsymbol{n}} v_h\}\!\}\|^2_{H^{-\frac{1}{2}}(\Gamma),h} - \sum_{K\in\mathcal{T}_{h,\Gamma}} \Big(\frac{2C_\gamma}{\epsilon}\Big)\|\alpha^{\frac{1}{2}}\nabla v_h\|^2_{L^2(K)} \\
&\qquad + (\gamma_p - \epsilon)\|[\![v_h]\!]\|^2_{H^{\frac{1}{2}}(\Gamma),h} \\
&= \sum_{K\in\widetilde{\mathcal{T}}_h\setminus\mathcal{T}_{h,\Gamma}} \|\alpha^{\frac{1}{2}}\nabla v_h\|^2_{L^2(K)} + \sum_{K\in\mathcal{T}_{h,\Gamma}} \|\alpha^{\frac{1}{2}}\nabla v_h\|^2_{L^2(K)} + \frac{1}{\epsilon}\|\{\!\{\alpha\nabla_{\boldsymbol{n}} v_h\}\!\}\|^2_{H^{-\frac{1}{2}}(\Gamma),h} \\
&\qquad - \sum_{K\in\mathcal{T}_{h,\Gamma}} \Big(\frac{2C_\gamma}{\epsilon}\Big)\|\alpha^{\frac{1}{2}}\nabla v_h\|^2_{L^2(K)} + (\gamma_p - \epsilon)\|[\![v_h]\!]\|^2_{H^{\frac{1}{2}}(\Gamma),h} \\
&= \sum_{K\in\widetilde{\mathcal{T}}_h\setminus\mathcal{T}_{h,\Gamma}} \|\alpha^{\frac{1}{2}}\nabla v_h\|^2_{L^2(K)} + \frac{1}{\epsilon}\|\{\!\{\alpha\nabla_{\boldsymbol{n}} v_h\}\!\}\|^2_{H^{-\frac{1}{2}}(\Gamma),h} \\
&\qquad + \sum_{K\in\mathcal{T}_{h,\Gamma}} \Big(1 - \frac{2C_\gamma}{\epsilon}\Big)\|\alpha^{\frac{1}{2}}\nabla v_h\|^2_{L^2(K)} + (\gamma_p - \epsilon)\|[\![v_h]\!]\|^2_{H^{\frac{1}{2}}(\Gamma),h}.
\end{aligned}
$$

Similar to the last section, the second line uses the Cauchy-Schwarz inequality (2.8) on the duality paring on the boundary. The third line utilizes Young's inequality for some $\epsilon > 0$ and then the trace inequality (3.10) is used in the fifth line.

Appendix B

Numerical Integration

B.1 Numerical Integration in XFEM

Here, we describe the numerical integration on the cut elements for the interface problems. These new basis functions in the XFEM framework have compact support only in a restriction on its subdomain Ω_i. Here, we describe modified quadrature rules for numerical integration of the cut elements and the interfaces.

For any given element K, let $K_i = K \cap \Omega_i$ be the restriction of the element K in the domain Ω_i. The set of all the elements intersected by the interface Γ is denoted by $\mathcal{T}_{h,\Gamma} := \{K \in \widetilde{\mathcal{T}_h} : K \cap \Gamma \neq \emptyset\}$. Also for an element $K \in \mathcal{T}_{h,\Gamma}$, let $\Gamma_K := \Gamma \cap K$ be the part of Γ in K.

Figure (B.2) and Figure (B.3) provide a graphical representation of the modified quadrature rule. An element K is intersected by the interface, hence it is doubled by adding extra degrees of freedom at the same location of the nodes. Thus, the element K can be seen as two overlapping elements K_1 and K_2. The element K_1 in this particular case is split into a triangle and its counterpart element K_2 is cut into a quadrilateral. The numerical integration of the enriched elements and the discrete interface is then done in the following steps:

- We determine the set of elements intersected by the interface, $\mathcal{T}_{h,\Gamma}$.
- Then, we need to determine on which side of the interface the element K_i belong.
- We map the quadrature points defined on the reference element to the enriched current element using the affine map $\varphi : \hat{K} \to K_i$.
- In the next step, the inverse of the affine map $\varphi^{-1} : K_i \to \hat{K}$ is used to map the quadrature points back to the reference element.

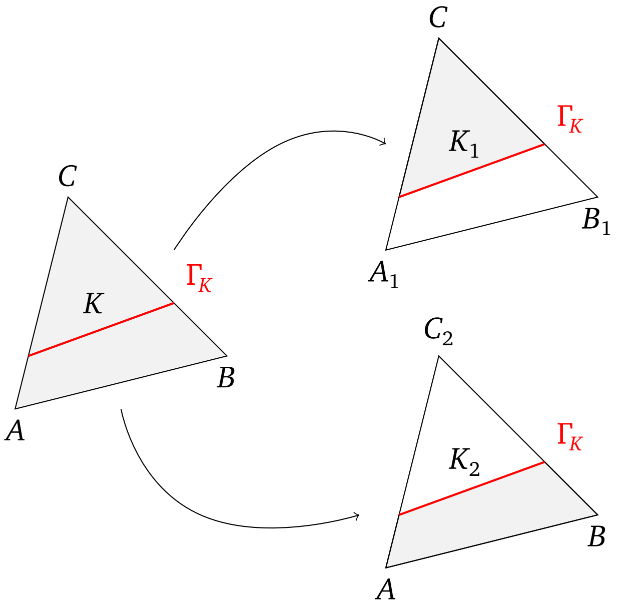


Figure B.1. Splitting of a triangular element with the XFEM enrichments.

The numerical integration of the discrete interface Γ_K is then done in the following steps:

- First a set of all Γ_K is determined.
- We define a map $\vartheta : \hat{E} \rightarrow \Gamma_K$, where E is a reference element of the dimension of the interface.
- In the last step, the inverse of the affine map $\varphi^{-1} : \Gamma_K \rightarrow \hat{K}$ is used to map the quadrature points in the reference element.

Once the proper quadrature points are computed, we can compute the element-wise contribution of the bilinear and linear forms.

B.2 Numerical Integration for L^2-projection

As we discussed, to use the multilevel method we need to compute a transfer operator based on L^2 projection. In order to compute the transfer operator $\boldsymbol{T}$, we need to compute $\boldsymbol{T} = \boldsymbol{D}^{-1}\boldsymbol{B}$ in (4.10), where $\boldsymbol{D}$ is the mass matrix of the fine level mesh and $\boldsymbol{B}$ is the coupling operator between the fine and coarse level meshes. These computations are done in the following way:

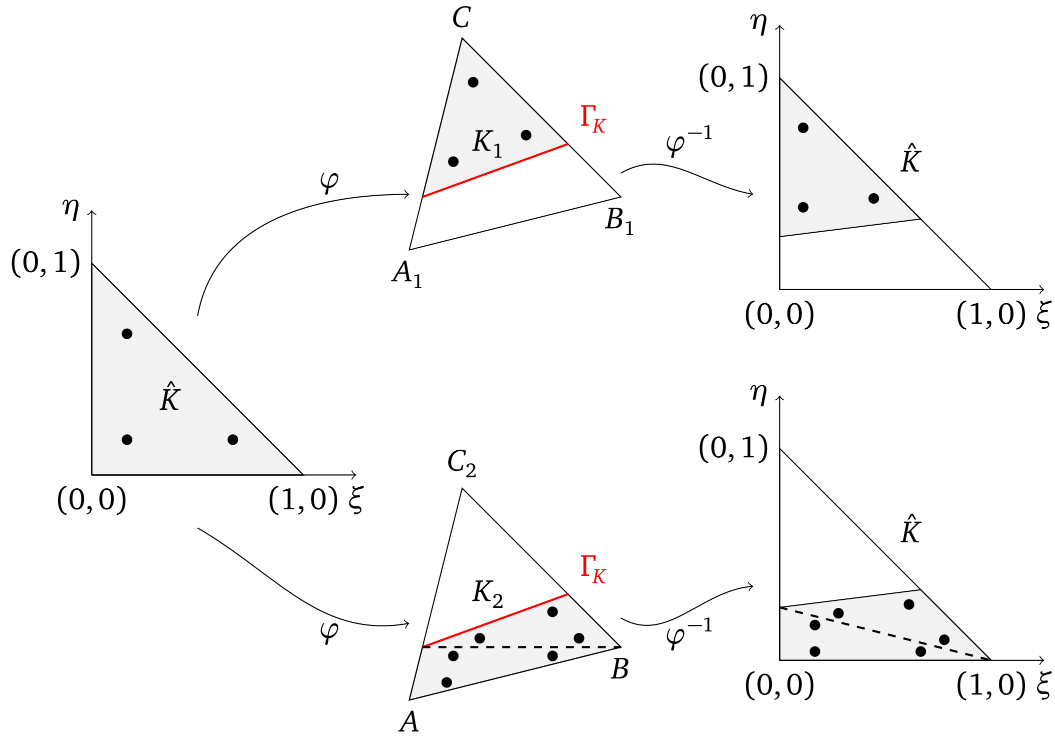


Figure B.2. Mapping the quadrature points on the enriched elements, the gray region represents the region where support of the basis function is nonzero.

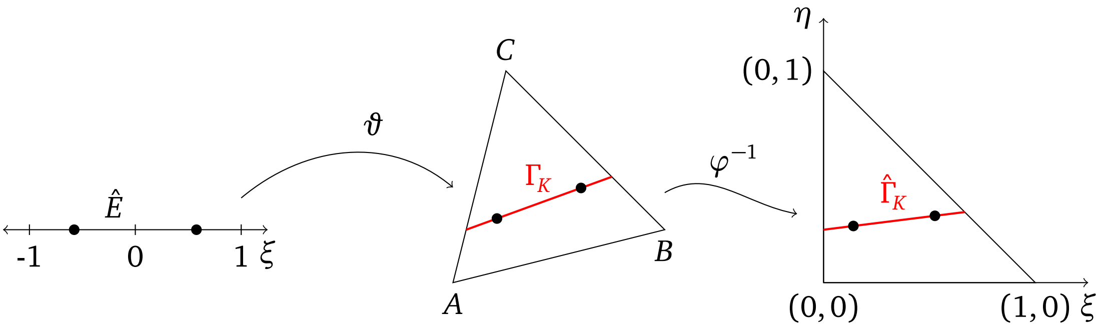


Figure B.3. Mapping the quadrature points on the interface segment Γ_K.

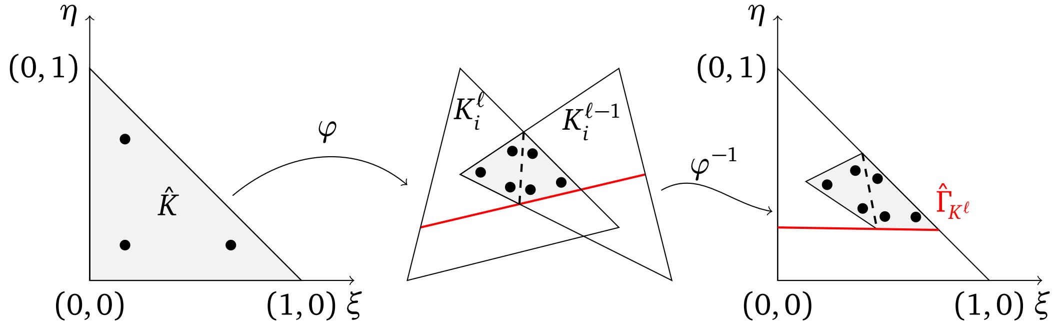


Figure B.4. Mapping the quadrature points on the intersection between the enriched elements from different levels.

- First compute set of all elements on each level which are intersected by the interface. Then we determine on which side of the interface the elements belong.

- We determine the pair of intersecting elements $\langle K_i^\ell, K_i^{\ell-1}\rangle$, where $K^\ell \in \widetilde{\mathcal{T}}_\ell$ and $K^{\ell-1} \in \widetilde{\mathcal{T}}_{\ell-1}$.

- For each pair $\langle K_i^\ell, K_i^{\ell-1}\rangle$, we need to compute the intersection polytope $I_s = K_i^\ell \cap K_i^{\ell-1}$.

- If the intersection polytope is not a triangle, we further need to subdivide the intersection into triangles. The triangulation is not explicitly computed but it is only used to map the reference triangulation to the intersection.

- In the next step, we map the quadrature points from the intersection to back to the reference element $\hat{K}$.

- We compute the local element-wise contribution by means of the numerical quadrature and assemble $\boldsymbol{B}$ and $\boldsymbol{D}$.

Figure (B.4) gives a graphical representation of how the numerical integration is carried out.

Appendix C

Uzawa Algorithms

Uzawa algorithm with conjugate directions

Here, in C.1 we have given the algorithm for conjugate gradient Uzawa method.

Algorithm C.1: Uzawa-Conjugate Gradient Method

```
Input  : $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{f}, \boldsymbol{u}, \boldsymbol{g}_{\lambda}, \boldsymbol{\lambda}$
Output: $\boldsymbol{u}, \boldsymbol{\lambda}$
$\boldsymbol{\lambda} \in \mathbb{R}^m$ ;                                          ▷ initial guess
$\boldsymbol{u} \mapsfrom \boldsymbol{A}^{-1}(\boldsymbol{f} - \boldsymbol{B}^{\top}\boldsymbol{\lambda})$ ;                          ▷ solve primal problem
$\boldsymbol{q} \mapsfrom \boldsymbol{g}_{\lambda} - \boldsymbol{B}\boldsymbol{u}$ ;                                  ▷ compute residual
$\boldsymbol{d} \mapsfrom -\boldsymbol{q}$ ;                                         ▷ set direction
while $\boldsymbol{q} \neq 0$ do
    $\boldsymbol{p} \mapsfrom \boldsymbol{B}^{\top}\boldsymbol{d}$ ;                                   ▷ intermediate step
    $\boldsymbol{h} \mapsfrom \boldsymbol{A}^{-1}\boldsymbol{p}$ ;                                  ▷ solve primal problem
    $\alpha \mapsfrom \frac{\boldsymbol{q}^{\top}\boldsymbol{q}}{\boldsymbol{p}^{T}\boldsymbol{h}}$ ;                                 ▷ compute step size
    $\boldsymbol{\lambda} \mapsfrom \boldsymbol{\lambda} + \alpha\boldsymbol{d}$ ;                            ▷ update the dual iterate
    $\boldsymbol{u} \mapsfrom \boldsymbol{u} - \alpha\boldsymbol{h}$ ;                          ▷ update the primal iterate
    $\boldsymbol{q}^{(*)} \mapsfrom \boldsymbol{g}_{\lambda} - \boldsymbol{B}\boldsymbol{u}$ ;                           ▷ compute new residual
    $\beta \mapsfrom \frac{(\boldsymbol{q}^{(*)})^{\top}\boldsymbol{q}^{(*)}}{\boldsymbol{q}^{\top}\boldsymbol{q}}$ ;                          ▷ orthogonalization
    $\boldsymbol{d} \mapsfrom -\boldsymbol{q}^{(*)} + \beta\boldsymbol{d}$ ;                           ▷ new direction
    $\boldsymbol{q} \mapsfrom \boldsymbol{q}^{(*)}$ ;                          ▷ updated for new iteration
```

In C.2 we have given the algorithm for preconditioned conjugate gradient

Uzawa method.

Algorithm C.2: Uzawa-Preconditioned Conjugate Gradient Method

```
Input  : $A, B, f, u, g_\lambda, \lambda, P$
Output: $u, \lambda$
$\lambda \in \mathbb{R}^m$ ;                                    ▷ initial guess
$u \leftarrow A^{-1}(f - B^\top \lambda)$ ;                     ▷ solve primal problem
$q \leftarrow g_\lambda - Bu$ ;                                 ▷ compute residual
$z \leftarrow P^{-1}q$ ;                                        ▷ preconditioning
$d \leftarrow -z$ ;                                             ▷ set direction
while $q \neq 0$ do
    $p \leftarrow B^\top d$ ;                                   ▷ intermediate step
    $h \leftarrow A^{-1}p$ ;                                    ▷ solve primal problem
    $\alpha \leftarrow \frac{q^\top z}{p^T h}$ ;                ▷ compute step size
    $\lambda \leftarrow \lambda + \alpha d$ ;                   ▷ update the dual iterate
    $u \leftarrow u - \alpha h$ ;                               ▷ update the primal iterate
    $q^{(*)} \leftarrow g_\lambda - Bu$ ;                       ▷ compute new residual
    $z^{(*)} \leftarrow P^{-1}q^{(*)}$ ;                        ▷ preconditioning
    $\beta \leftarrow \frac{(q^{(*)})^\top z^{(*)}}{q^\top z}$; ▷ orthogonalization
    $d \leftarrow -z^{(*)} + \beta d$ ;                         ▷ new direction
    $q \leftarrow q^{(*)}$ ;                                    ▷ updated for new iteration
```

Bibliography

[ADKL01] P. Amestoy, I. S. Duff, J. Koster, and J.-Y. L'Excellent. A fully asynchronous multifrontal solver using distributed dynamic scheduling. *SIAM Journal on Matrix Analysis and Applications*, 23(1):15–41, 2001.

[AHD12] C. Annavarapu, M. Hautefeuille, and J. E. Dolbow. A robust Nitsche's formulation for interface problems. *Computer Methods in Applied Mechanics and Engineering*, 225-228:44–54, June 2012.

[Arn82] D. N. Arnold. An Interior Penalty Finite Element Method with Discontinuous Elements. *SIAM Journal on Numerical Analysis*, 19(4):742–760, August 1982.

[BAA$^+$20] S. Balay, S. Abhyankar, M. F. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, A. Dener, V. Eijkhout, W. D. Gropp, D. Karpeyev, D. Kaushik, M. G. Knepley, D. A. May, L. C. McInnes, R. T. Mills, T. Munson, K. Rupp, P. Sanan, B. F. Smith, S. Zampini, H. Zhang, and H. Zhang. PETSc users manual. Technical Report ANL-95/11 - Revision 3.13, Argonne National Laboratory, 2020.

[Bab73a] I. Babuška. The finite element method with Lagrangian multipliers. *Numerische Mathematik*, 20(3):179–192, June 1973.

[Bab73b] I. Babuška. The finite element method with Penalty. *Mathematics of Computation*, 27(122):221–228, 1973.

[Bac06] C. Bacuta. A Unified Approach for Uzawa Algorithms. *SIAM Journal on Numerical Analysis*, 44(6):2633–2649, January 2006.

[Bac14] C. Bacuta. Cascadic multilevel algorithms for symmetric saddle point systems. *Computers & Mathematics with Applications*, 67(10):1905–1913, June 2014.

[BB99] T. Belytschko and T. Black. Elastic crack growth in finite elements with minimal remeshing. *International Journal for Numerical Methods in Engineering*, 45(5):601–620, June 1999.

[BBDL12] N. Barrau, R. Becker, E. Dubach, and R. Luce. A robust variant of NXFEM for the interface problem. *Comptes Rendus Mathematique*, 350(15-16):789–792, August 2012.

[BBF13] D. Boffi, F. Brezzi, and M. Fortin. *Mixed Finite Element Methods and Applications*. Springer Science & Business Media, Jul 2013.

[BBH09] R. Becker, E. Burman, and P. Hansbo. A Nitsche extended finite element method for incompressible elasticity with discontinuous modulus of elasticity. *Computer Methods in Applied Mechanics and Engineering*, 198(41):3352–3360, September 2009.

[BCH$^+$15] E. Burman, S. Claus, P. Hansbo, M. G. Larson, and A. Massing. CutFEM: Discretizing geometry and partial differential equations. *International Journal for Numerical Methods in Engineering*, 104(7):472–501, November 2015.

[BDW99] D. Braess, W. Dahmen, and C. Wieners. A Multigrid Algorithm for the Mortar Finite Element Method. *SIAM Journal on Numerical Analysis*, 37(1):48–69, January 1999.

[BE86] J. W. Barrett and C. M. Elliott. Finite element approximation of the Dirichlet problem using the boundary penalty method. *Numerische Mathematik*, 49(4):343–366, July 1986.

[Bel99] F. B. Belgacem. The Mortar finite element method with Lagrange multipliers. *Numerische Mathematik*, 84(2):173–197, December 1999.

[BFMR97] F. Brezzi, L. P. Franca, D. L. Marini, and A. Russo. Stabilization Techniques for Domain Decomposition Methods with Non-matching Grids. Technical report, Centre National de la Recherche Scientifique, Paris, France, France, 1997.

[BGL05] M. Benzi, G. H. Golub, and J. Liesen. Numerical solution of saddle point problems. *Acta Numerica*, 14:1–137, May 2005.

[BGMS97] S. Balay, W. D. Gropp, L. C. McInnes, and B. F. Smith. Efficient management of parallelism in object oriented numerical software libraries. In E. Arge, A. M. Bruaset, and H. P. Langtangen, editors, *Modern Software Tools in Scientific Computing*, pages 163–202. Birkhäuser Press, 1997.

[BH91] H. J. C. Barbosa and T. J. R. Hughes. The finite element method with Lagrange multipliers on the boundary: circumventing the Babuška-Brezzi condition. *Computer Methods in Applied Mechanics and Engineering*, 85(1):109–128, January 1991.

[BH92a] H. J. C. Barbosa and T. J. R. Hughes. Boundary Lagrange multipliers in finite element methods: Error analysis in natural norms. *Numerische Mathematik*, 62(1):1–15, December 1992.

[BH92b] H. J. C. Barbosa and T. J. R. Hughes. Circumventing the Babuška-Brezzi condition in mixed finite element approximations of elliptic variational inequalities. *Computer Methods in Applied Mechanics and Engineering*, 97(2):193–210, June 1992.

[BH04] E. Burman and P. Hansbo. Edge stabilization for Galerkin approximations of convection–diffusion–reaction problems. *Computer Methods in Applied Mechanics and Engineering*, 193(15-16):1437–1453, April 2004.

[BH06] E. Burman and P. Hansbo. Edge stabilization for the generalized Stokes problem: A continuous interior penalty method. *Computer Methods in Applied Mechanics and Engineering*, 195(19-22):2393–2410, April 2006.

[BH10] E. Burman and P. Hansbo. Fictitious domain finite element methods using cut elements: I. A stabilized Lagrange multiplier method. *Computer Methods in Applied Mechanics and Engineering*, 199(41-44):2680–2686, October 2010.

[BH12] E. Burman and P. Hansbo. Fictitious domain finite element methods using cut elements: II. A stabilized Nitsche method. *Applied Numerical Mathematics*, 62(4):328–341, April 2012.

[BH14] E. Burman and P. Hansbo. Fictitious domain methods using cut elements: III. A stabilized Nitsche method for Stokes' problem. *ESAIM:*

Mathematical Modelling and Numerical Analysis, 48(3):859–874, May 2014.

[BHS03] R. Becker, P. Hansbo, and R. Stenberg. A finite element method for domain decomposition with non-matching grids. *ESAIM: Mathematical Modelling and Numerical Analysis*, 37(2):209–225, March 2003.

[BL05] P. Bochev and R. B. Lehoucq. On the Finite Element Solution of the Pure Neumann Problem. *SIAM Review*, 47(1):50–66, January 2005.

[BLS14] S. C. Brenner, H. Li, and L.-Y. Sung. Multigrid methods for saddle point problems: Stokes and Lamé systems. *Numerische Mathematik*, 128(2):193–216, October 2014.

[BM97] I. Babuška and J. M. Melenk. The partition of unity method. *International Journal for Numerical Methods in Engineering*, 40(4):727–758, February 1997.

[BMM$^+$00] F. Brezzi, G. Manzini, D. Marini, P. Pietra, and A. Russo. Discontinuous Galerkin approximations for elliptic problems. *Numerical Methods for Partial Differential Equations*, 16(4):365–378, 2000.

[BMW09] E. Béchet, N. Moës, and B. Wohlmuth. A stable Lagrange multiplier space for stiff interface conditions within the extended finite element method. *International Journal for Numerical Methods in Engineering*, 78(8):931–954, May 2009.

[BOL78] I. Babuška, J. T. Oden, and J. K. Lee. Mixed-hybrid finite element approximations of second-order elliptic boundary value problems: Part 2 — weak-hybrid methods. *Computer Methods in Applied Mechanics and Engineering*, 14(1):1–22, April 1978.

[BPM$^+$03] T. Belytschko, C. Parimi, N. Moës, N. Sukumar, and S. Usui. Structured extended finite element methods for solids defined by implicit surfaces. *International Journal for Numerical Methods in Engineering*, 56(4):609–635, 2003.

[BR97] F. Bassi and S. Rebay. A High-Order Accurate Discontinuous Finite Element Method for the Numerical Solution of the Compressible Navier–Stokes Equations. *Journal of Computational Physics*, 131(2):267–279, March 1997.

[Bra07] D. Braess. *Finite Elements: Theory, Fast Solvers, and Applications in Solid Mechanics*. Cambridge University Press, Apr 2007.

[Bre74] F. Brezzi. On the existence, uniqueness and approximation of saddle-point problems arising from lagrangian multipliers. *Revue française d'automatique, informatique, recherche opérationnelle. Analyse numérique*, 8(R2):129–151, 1974.

[BS07] S. Brenner and R. Scott. *The Mathematical Theory of Finite Element Methods*. Springer, Dec 2007.

[BS13] C. Bacuta and L. Shu. Multilevel Gradient Uzawa Algorithms for Symmetric Saddle Point Problems. *Journal of Scientific Computing*, 57(1):105–123, October 2013.

[Bur10] E. Burman. Ghost penalty. *Comptes Rendus Mathematique*, 348(21-22):1217–1220, November 2010.

[BvEHM00] W. L. Briggs, van Emden Henson, and S. F. McCormick. *A Multigrid Tutorial: Second Edition*. SIAM, Jul 2000.

[BVWH$^+$12] L. Berger-Vergiat, H. Waisman, B. Hiriyur, R. Tuminaro, and D. Keyes. Inexact Schwarz-algebraic multigrid preconditioners for crack problems modeled by extended finite element methods: XFEM_dd. *International Journal for Numerical Methods in Engineering*, 90(3):311–328, April 2012.

[BW08] J. Bonet and R. D. Wood. *Nonlinear Continuum Mechanics for Finite Element Analysis*. Cambridge University Press, Mar 2008.

[BZ12] E. Burman and P. Zunino. Numerical Approximation of Large Contrast Problems with the Unfitted Nitsche Method. In J. Blowey and M. Jensen, editors, *Frontiers in Numerical Analysis - Durham 2010*, Lecture Notes in Computational Science and Engineering, pages 227–282. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[CBM15] S. Claus, E. Burman, and A. Massing. CutFEM : a stabilised Nitsche XFEM method for multi-physics problems. In *DIVA*, pages 171–174. Swansea University, 2015.

[CHLR19] F. Chouly, P. Hild, V. Lleras, and Y. Renard. Nitsche-Based Finite Element Method for Contact with Coulomb Friction. In F. A. Radu,

K. Kumar, I. Berre, J. M. Nordbotten, and I. S. Pop, editors, *Numerical Mathematics and Advanced Applications ENUMATH 2017*, Lecture Notes in Computational Science and Engineering, pages 839–847. Springer International Publishing, 2019.

[CHR14] F. Chouly, P. Hild, and Y. Renard. Symmetric and non-symmetric variants of Nitsche's method for contact problems in elasticity: theory and numerical experiments. *Mathematics of Computation*, 84(293):1089–1112, October 2014.

[Cia97] P. G. Ciarlet. *Mathematical Elasticity: Volume 1*. North-Holland, Aug 1997.

[CK18] S. Claus and P. Kerfriden. A stable and optimally convergent LaTIn-CutFEM algorithm for multiple unilateral contact problems: A stable and optimally convergent LaTIn-CutFEM algorithm for multiple unilateral contact problems. *International Journal for Numerical Methods in Engineering*, 113(6):938–966, February 2018.

[CV01] T. F. Coleman and A. Verma. A Preconditioned Conjugate Gradient Approach to Linear Equality Constrained Minimization. *Computational Optimization and Applications*, 20(1):61–72, October 2001.

[CZ98] Z. Chen and J. Zou. Finite element methods and their convergence for elliptic and parabolic interface problems. *Numerische Mathematik*, 79(2):175–202, April 1998.

[DD76] J. Douglas and T. Dupont. Interior Penalty Procedures for Elliptic and Parabolic Galerkin Methods. In R. Glowinski and J. L. Lions, editors, *Computing Methods in Applied Sciences*, Lecture Notes in Physics, pages 207–216. Springer Berlin Heidelberg, 1976.

[DF08] J. E. Dolbow and L. P. Franca. Residual-free bubbles for embedded Dirichlet problems. *Computer Methods in Applied Mechanics and Engineering*, 197(45-48):3751–3759, August 2008.

[DH09] J. Dolbow and I. Harari. An efficient finite element method for embedded interface problems. *International Journal for Numerical Methods in Engineering*, 78(2):229–252, April 2009.

[DK09] T. Dickopf and R. Krause. Efficient simulation of multi-body contact problems on complex geometries: A flexible decomposition approach using constrained minimization. *International Journal for Numerical Methods in Engineering*, 77(13):1834–1862, March 2009.

[DK11] T. Dickopf and R. Krause. A Study of Prolongation Operators Between Non-nested Meshes. In Y. Huang, R. Kornhuber, O. Widlund, and J. Xu, editors, *Domain Decomposition Methods in Science and Engineering XIX*, volume 78, pages 343–350. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[DK14] T. Dickopf and R. Krause. Evaluating local approximations of the L^2-orthogonal projection between non-nested finite element spaces. *Numerical Mathematics: Theory, Methods and Applications*, 7(3):288–316, August 2014.

[DMB01] J. Dolbow, N. Moës, and T. Belytschko. An extended finite element method for modeling crack growth with frictional contact. *Computer Methods in Applied Mechanics and Engineering*, 190(51):6825–6846, October 2001.

[DMD$^+$00] C. Daux, N. Moës, J. Dolbow, N. Sukumar, and T. Belytschko. Arbitrary branched and intersecting cracks with the extended finite element method. *International Journal for Numerical Methods in Engineering*, 48(12):1741–1760, 2000.

[Dos09] Z. Dostál. *Optimal Quadratic Programming Algorithms: With Applications to Variational Inequalities (Springer Optimization and Its Applications)*. Springer, Mar 2009.

[dPLM18] F. [de Prenter], C. Lehrenfeld, and A. Massing. A note on the stability parameter in nitsche's method for unfitted boundary value problems. *Computers & Mathematics with Applications*, 75(12):4322 – 4336, 2018.

[EDH10] A. Embar, J. Dolbow, and I. Harari. Imposing Dirichlet boundary conditions with Nitsche's method and spline-based finite elements. *International Journal for Numerical Methods in Engineering*, 83(7):877–898, August 2010.

[FB10] T.-P. Fries and T. Belytschko. The extended/generalized finite element method: An overview of the method and its applications. *International Journal for Numerical Methods in Engineering*, 84(3):253–304, October 2010.

[FMR94] 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, May 1994.

[FR91] 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, October 1991.

[GG95] V. Girault and R. Glowinski. Error analysis of a fictitious domain method applied to a Dirichlet problem. *Japan Journal of Industrial and Applied Mathematics*, 12(3):487, October 1995.

[GGV04] C. Greif, G. H. Golub, and J. M. Varah. Augmented Lagrangian Techniques for Solving Saddle Point Linear Systems. Technical report, SIAM J. Matrix Anal. Appl, 2004.

[GHN01] N. I. M. Gould, M. E. Hribar, and J. Nocedal. On the Solution of Equality Constrained Quadratic Programming Problems Arising in Optimization. *SIAM Journal on Scientific Computing*, 23(4):1376–1395, January 2001.

[Glo84] R. Glowinski. *Numerical Methods for Nonlinear Variational Problems*. Springer-Verlag, Feb 1984.

[GLOR16] S. Groß, T. Ludescher, M. Olshanskii, and A. Reusken. Robust Preconditioning for XFEM Applied to Time-Dependent Stokes Problems. *SIAM Journal on Scientific Computing*, 38(6):A3492–A3514, January 2016.

[GM18] C. Gürkan and A. Massing. A stabilized cut discontinuous Galerkin framework: II. Hyperbolic problems. *arXiv:1807.05634 [cs, math]*, July 2018. arXiv: 1807.05634.

[GM19] C. Gürkan and A. Massing. A stabilized cut discontinuous Galerkin framework for elliptic boundary value and interface problems. *Computer Methods in Applied Mechanics and Engineering*, 348:466–499, May 2019.

[GPP94] R. Glowinski, T.-W. Pan, and J. Periaux. A fictitious domain method for Dirichlet problem and applications. *Computer Methods in Applied Mechanics and Engineering*, 111(3-4):283–303, January 1994.

[GR11] S. Gross and A. Reusken. *Numerical Methods for Two-Phase Incompressible Flows*. Springer Science & Business Media, Apr 2011.

[GRR06] S. Groß, V. Reichelt, and A. Reusken. A finite element based level set method for two-phase incompressible flows. *Computing and Visualization in Science*, 9(4):239–257, December 2006.

[GS03] M. Griebel and M. A. Schweitzer. A Particle-Partition of Unity Method Part V: Boundary Conditions. In S. Hildebrandt and H. Karcher, editors, *Geometric Analysis and Nonlinear Partial Differential Equations*, pages 519–542. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[GSV20] T. Gustafsson, R. Stenberg, and J. Videman. On nitsche's method for elastic contact problems. *SIAM Journal on Scientific Computing*, 42(2):B425–B446, 2020.

[GT13] A. Gerstenberger and R. S. Tuminaro. An algebraic multigrid approach to solve extended finite element method based fracture problems. *International Journal for Numerical Methods in Engineering*, 94(3):248–272, April 2013.

[GvL12] G. H. Golub and C. F. van Loan. *Matrix Computations*. JHU Press, Dec 2012.

[Hac86] W. Hackbusch. *Multi-Grid Methods and Applications*. Springer-Verlag, Jan 1986.

[Hac16] W. Hackbusch. *Iterative Solution of Large Sparse Systems of Equations*. Springer, Jun 2016.

[HAD12] M. Hautefeuille, C. Annavarapu, and J. E. Dolbow. Robust imposition of Dirichlet boundary conditions on embedded surfaces. *International Journal for Numerical Methods in Engineering*, 90(1):40–64, 2012.

[HH02] A. Hansbo and P. Hansbo. An unfitted finite element method, based on Nitsche's method, for elliptic interface problems. *Computer*

Methods in Applied Mechanics and Engineering, 191(47):5537–5552, November 2002.

[HH04] A. Hansbo and P. Hansbo. A finite element method for the simulation of strong and weak discontinuities in solid mechanics. *Computer Methods in Applied Mechanics and Engineering*, 193(33):3523–3540, August 2004.

[HKK04] M. Hintermüller, V. Kovtunenko, and K. Kunisch. Semismooth newton methods for a class of unilaterally constrained variational problems. *Mathematical Programming*, 14, 01 2004.

[HLL17] P. Hansbo, M. G. Larson, and K. Larsson. Cut Finite Element Methods for Linear Elasticity Problems. In S. P. A. Bordas, E. Burman, M. G. Larson, and M. A. Olshanskii, editors, *Geometrically Unfitted Finite Element Methods and Applications*, Lecture Notes in Computational Science and Engineering, pages 25–63. Springer International Publishing, 2017.

[Hol00] G. A. Holzapfel. *Nonlinear Solid Mechanics: A Continuum Approach for Engineering*. Wiley, Mar 2000.

[HR09] J. Haslinger and Y. Renard. A New Fictitious Domain Approach Inspired by the Extended Finite Element Method. *SIAM Journal on Numerical Analysis*, 47, January 2009.

[HTW$^+$12] B. Hiriyur, R. S. Tuminaro, H. Waisman, E. G. Boman, and D. E. Keyes. A Quasi-algebraic Multigrid Approach to Fracture Problems Based on Extended Finite Elements. *SIAM Journal on Scientific Computing*, 34(2):A603–A626, January 2012.

[HW10] C. Hager and B. I. Wohlmuth. Semismooth Newton methods for variational problems with inequality constraints. *GAMM-Mitteilungen*, 33(1):8–24, 2010.

[IXZ14] K. Ito, H. Xiang, and J. Zou. An Inexact Uzawa Algorithm for Generalized Saddle-Point Problems and Its Convergence. *arXiv:1408.5547 [math]*, August 2014. arXiv: 1408.5547.

[JADH15] W. Jiang, C. Annavarapu, J. E. Dolbow, and I. Harari. A robust Nitsche's formulation for interface problems with spline-based finite elements: A robust Nitsche's formulation for interface problems with spline-based finite elements. *International Journal for Numerical Methods in Engineering*, 104(7):676–696, November 2015.

[JD04] H. Ji and J. E. Dolbow. On strategies for enforcing interfacial constraints and evaluating jump conditions with the extended finite element method. *International Journal for Numerical Methods in Engineering*, 61(14):2508–2535, 2004.

[KDL06] T. Y. Kim, J. Dolbow, and T. Laursen. A mortared finite element method for frictional contact on arbitrary interfaces. *Computational Mechanics*, 39(3):223–235, November 2006.

[KHS$^+$15] D. Kamensky, M.-C. Hsu, D. Schillinger, J. A. Evans, A. Aggarwal, Y. Bazilevs, M. S. Sacks, and T. J. R. Hughes. An immersogeometric variational framework for fluid–structure interaction: Application to bioprosthetic heart valves. *Computer Methods in Applied Mechanics and Engineering*, 284:1005–1053, February 2015.

[KK01] R. Kornhuber and R. Krause. Adaptive multigrid methods for Signorini's problem in linear elasticity. *Computing and Visualization in Science*, 4(1):9–20, November 2001.

[KK19] H. Kothari and R. Krause. A multigrid method for a nitsche-based extended finite element method. *arXiv preprint arXiv:1912.00496*, 2019.

[KMNv13] R. Kučera, J. Machalová, H. Netuka, and P. Ženčák. An interior-point algorithm for the minimization arising from 3D contact problems with friction. *Optimization Methods and Software*, 28(6):1195–1217, December 2013.

[KO87] N. Kikuchi and J. T. Oden. *Contact Problems in Elasticity: A Study of Variational Inequalities and Finite Element Methods (Studies in Applied and Numerical Mathematics)*. Society for Industrial and Applied Mathematics, Jan 1987.

[Kor94] R. Kornhuber. Monotone multigrid methods for elliptic variational inequalities I. *Numerische Mathematik*, 69(2):167–184, December 1994.

[Kor96] R. Kornhuber. Monotone multigrid methods for elliptic variational inequalities II. *Numerische Mathematik*, 72(4):481–499, February 1996.

[Kor97] R. Kornhuber. *Adaptive Monotone Multigrid Methods for Nonlinear Variational Problems*. Vieweg+Teubner Verlag, 1997.

[Kra01] R. Krause. *Monotone multigrid methods for Signorini's problem with friction*. PhD thesis, Freie Universität, Berlin, 2001.

[Kra09] R. Krause. A Nonsmooth Multiscale Method for Solving Frictional Two-Body Contact Problems in 2d and 3d with Multigrid Efficiency. *SIAM Journal on Scientific Computing*, 31(2):1399–1423, January 2009.

[Krz04] P. Krzyżanowski. A Class of Block Smoothers for Multigrid Solution of Saddle Point Problems with Application to Fluid Flow. In *Parallel Processing and Applied Mathematics*, volume 3019, pages 1006–1013. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[KW01] A. Klawonn and O. Widlund. FETI and Neumann-Neumann iterative substructuring methods: Connections and new results. *Communications on Pure and Applied Mathematics*, 54(1):57–90, 2001.

[KZ16] R. Krause and P. Zulian. A Parallel Approach to the Variational Transfer of Discrete Fields between Arbitrarily Distributed Unstructured Finite Element Meshes. *SIAM Journal on Scientific Computing*, 38(3):C307–C333, January 2016.

[Lac96] C. Lacour. Iterative Substructuring Preconditioners for the Mortar Finite Element Method. pages 406–412. Domain Decompostion Press, June 1996.

[Lau13] T. A. Laursen. *Computational Contact and Impact Mechanics: Fundamentals of Modeling Interfacial Phenomena in Nonlinear Finite Element Analysis*. Springer Science & Business Media, Mar 2013.

[LB13] M. G. Larson and F. Bengzon. *The Finite Element Method: Theory, Implementation, and Applications*. Springer, Jan 2013.

[Leh15] C. Lehrenfeld. *On a Space-Time Extended Finite Element Method for the Solution of a Class of Two-Phase Mass Transport Problems*. PhD thesis, RWTH, Aachen, feb 2015.

[Leh16] C. Lehrenfeld. Removing the stabilization parameter in fitted and unfitted symmetric Nitsche formulations. In *Proceedings of the VII*

European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS Congress 2016), pages 373–383, Crete Island, Greece, 2016. Institute of Structural Analysis and Antiseismic Research School of Civil Engineering National Technical University of Athens (NTUA) Greece.

[LMDM14] C. Lang, D. Makhija, A. Doostan, and K. Maute. A simple and efficient preconditioning scheme for heaviside enriched XFEM. *Computational Mechanics*, 54(5):1357–1374, November 2014.

[LR17] C. Lehrenfeld and A. Reusken. Optimal preconditioners for Nitsche-XFEM discretizations of interface problems. *Numerische Mathematik*, 135(2):313–332, February 2017.

[lud20] A multigrid method for unfitted finite element discretizations of elliptic interface problems. *SIAM Journal on Scientific Computing*, 42(1):A318–A342, 2020.

[MB96] J. M. Melenk and I. Babuška. The partition of unity finite element method: Basic theory and applications. *Computer Methods in Applied Mechanics and Engineering*, 139(1):289–314, December 1996.

[MBT06] N. Moës, E. Béchet, and M. Tourbier. Imposing Dirichlet boundary conditions in the extended finite element method. *International Journal for Numerical Methods in Engineering*, 67(12):1641–1669, September 2006.

[MCCR03] N. Moës, M. Cloirec, P. Cartraud, and J. F. Remacle. A computational approach to handle complex microstructure geometries. *Computer Methods in Applied Mechanics and Engineering*, 192(28):3163–3177, July 2003.

[MDB99] N. Moës, J. Dolbow, and T. Belytschko. A finite element method for crack growth without remeshing. *International Journal for Numerical Methods in Engineering*, 46(1):131–150, September 1999.

[MDH07] H. M. Mourad, J. Dolbow, and I. Harari. A bubble-stabilized finite element method for Dirichlet constraints on embedded interfaces. *International Journal for Numerical Methods in Engineering*, 69(4):772–793, 2007.

[MDS17] N. Moës, J. E. Dolbow, and N. Sukumar. Extended Finite Element Methods. In E. Stein, R. de Borst, and T. J. R. Hughes, editors, *Encyclopedia of Computational Mechanics Second Edition*, pages 1–21. John Wiley & Sons, Ltd, Chichester, UK, December 2017.

[Nit71] 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(1):9–15, July 1971.

[NW00] J. Nocedal and S. Wright. *Numerical Optimization*. Springer, Apr 2000.

[OR17] M. A. Olshanskii and A. Reusken. Trace finite element methods for pdes on surfaces. In S. P. A. Bordas, E. Burman, M. G. Larson, and M. A. Olshanskii, editors, *Geometrically Unfitted Finite Element Methods and Applications*, pages 211–258, Cham, 2017. Springer International Publishing.

[Pat80] S. V. Patankar. *Numerical Heat Transfer and Fluid Flow*. CRC Press, Jan 1980.

[PDR07] J. Parvizian, A. Düster, and E. Rank. Finite cell method. *Computational Mechanics*, 41(1):121–133, December 2007.

[Pes02] C. S. Peskin. The immersed boundary method. *Acta Numerica*, 11, January 2002.

[Pit79] J. Pitkäranta. Boundary subspaces for the finite element method with Lagrange multipliers. *Numerische Mathematik*, 33(3):273–289, September 1979.

[Pit80] J. Pitkäranta. Local stability conditions for the Babuška method of Lagrange multipliers. *Mathematics of Computation*, 35(152):1113–1129, 1980.

[PWGW12] A. Popp, B. I. Wohlmuth, M. W. Gee, and W. A. Wall. Dual Quadratic Mortar Finite Element Methods for 3d Finite Deformation Contact. *SIAM Journal on Scientific Computing*, 34(4):B421–B446, January 2012.

[Reu08] A. Reusken. Analysis of an extended pressure finite element space for two-phase incompressible flows. *Computing and Visualization in Science*, 11(4-6):293–305, September 2008.

[RSB$^+$13] M. Ruess, D. Schillinger, Y. Bazilevs, V. Varduhn, and E. Rank. Weakly enforced essential boundary conditions for NURBS-embedded and trimmed NURBS geometries on the basis of the finite cell method: WEAK BOUNDARY CONDITIONS FOR THE FCM. *International Journal for Numerical Methods in Engineering*, 95(10):811–846, September 2013.

[RSOR14] 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*, 269:46–71, February 2014.

[Sal08] S. Salsa. *Partial Differential Equations in Action*. Springer, Feb 2008.

[SCMB01] N. Sukumar, D. L. Chopp, N. Moës, and T. Belytschko. Modeling holes and inclusions by level sets in the extended finite-element method. *Computer Methods in Applied Mechanics and Engineering*, 190(46):6183–6200, September 2001.

[SDS$^+$12] D. Schillinger, L. Dedè, M. A. Scott, J. A. Evans, M. J. Borden, E. Rank, and T. J. R. Hughes. An isogeometric design-through-analysis methodology based on adaptive hierarchical refinement of NURBS, immersed boundary methods, and T-spline CAD surfaces. *Computer Methods in Applied Mechanics and Engineering*, 249-252:116–150, December 2012.

[SR15] D. Schillinger and M. Ruess. The Finite Cell Method: A Review in the Context of Higher-Order Structural Analysis of CAD and Image-Based Geometric Models. *Archives of Computational Methods in Engineering*, 22(3):391–455, July 2015.

[Ste95] R. Stenberg. On some techniques for approximating boundary conditions in the finite element method. *Journal of Computational and Applied Mathematics*, 63(1-3):139–148, November 1995.

[Ste01] D. Stefanica. A Numerical Study of FETI Algorithms for Mortar Finite Element Methods. *SIAM Journal on Scientific Computing*, 23(4):1135–1160, January 2001.

[TOS00] U. Trottenberg, C. W. Oosterlee, and A. Schuller. *Multigrid*. Academic Press, Nov 2000.

[WK03] B. I. Wohlmuth and R. H. Krause. Monotone Multigrid Methods on Nonmatching Grids for Nonlinear Multibody Contact Problems. *SIAM Journal on Scientific Computing*, 25(1):324–347, January 2003.

[Woh00a] B. I. Wohlmuth. A Mortar Finite Element Method Using Dual Spaces for the Lagrange Multiplier. *SIAM Journal on Numerical Analysis*, 38(3):989–1012, January 2000.

[Woh00b] B. I. Wohlmuth. A multigrid method for saddle point problems arising from mortar finite element discretizations. *Electron Trans. Numer. Anal*, 11:43–54, 2000.

[Woh01] B. I. Wohlmuth. *Discretization Methods and Iterative Solvers Based on Domain Decomposition*. Springer, Feb 2001.

[Woh11] B. Wohlmuth. Variationally consistent discretization schemes and numerical algorithms for contact problems. *Acta Numerica*, 20:569–734, May 2011.

[Wri97] S. J. Wright. *Primal-Dual Interior-Point Methods*. Society for Industrial and Applied Mathematics, USA, 1997.

[Wri06] P. Wriggers. *Computational Contact Mechanics*. Springer, Jun 2006.

[WZ07] P. Wriggers and G. Zavarise. A formulation for frictionless contact problems using a weak form introduced by Nitsche. *Computational Mechanics*, 41(3):407–420, December 2007.

[Xu92] J. Xu. Iterative Methods by Space Decomposition and Subspace Correction. *SIAM Review*, 34(4):581–613, December 1992.

[Xu96] J. Xu. The auxiliary space method and optimal multigrid preconditioning techniques for unstructured grids. *Computing*, 56(3):215–235, September 1996.

[Xu01] J. Xu. The method of subspace corrections. *Journal of Computational and Applied Mathematics*, 128(1):335–362, March 2001.

[ZKN$^+$16] P. Zulian, A. Kopaničáková, M. C. G. Nestola, A. Fink, N. Fadel, A. Rigazzi, V. Magri, T. Schneider, E. Botter, J. Mankau, and R. Krause. Utopia: A C++ embedded domain specific language for scientific computing. Git repository. https://bitbucket.org/zulianp/utopia, 2016.

[Zul00] W. Zulehner. A Class of Smoothers for Saddle Point Problems. *Computing*, 65(3):227–246, December 2000.

[Zul01] W. Zulehner. Analysis of iterative methods for saddle point problems: a unified approach. *Mathematics of Computation*, 71(238):479–506, May 2001.

[Zul16] P. Zulian. ParMOONoLith: parallel intersection detection and automatic load-balancing library. Git repository. https://bitbucket.org/zulianp/par_moonolith, 2016.