



UPPSALA
UNIVERSITET

15072

Examensarbete 45 hp
August 2015

A Cut-Cell Implementation of the Finite Element Method in deal.II

Afonso Alborghetti Lonero

*[...] the awesome splendor of
the universe is much easier to
deal with if you think of it as a
series of small chunks.*

TERRY PRATCHETT, *Mort*

Abstract

A Cut-Cell Implementation of the Finite Element Method in deal.II

Afonso Alborghetti Londero

**Teknisk- naturvetenskaplig fakultet
UTH-enheten**

Besöksadress:
Ångströmlaboratoriet
Lägerhyddsvägen 1
Hus 4, Plan 0

Postadress:
Box 536
751 21 Uppsala

Teléfono:
018 - 471 30 03

Telefax:
018 - 471 30 00

Hemsida:
<http://www.teknat.uu.se/student>

The modeling of problems where the boundary changes significantly over time may become challenging as the mesh needs to be adapted constantly. In this context, computational methods where the mesh does not conform to the boundary are of great interest. This paper proposes a stabilized cut-cell approach to solve partial differential equations using unfitted meshes using the Finite Element Method. The open-source library deal.II was used for implementation. In order to evaluate the method, three problems in two-dimensions were tested: the Poisson problem, a pure diffusion Laplace-Beltrami problem and a reaction diffusion case. Stabilization effects on the stiffness matrix were studied for the first two test cases, and the theoretical dependence of the condition number with mesh size was confirmed. In addition, an optimal stabilization parameter was defined. Optimal convergence rates were obtained for the first two test cases.

Acknowledgments

During the development of this project in Uppsala University I have ventured into a new area for me which has been most interesting and instigating. For this I am deeply grateful to my advisor, Prof. Gunilla Kreiss, who introduced me to the world of unfitted finite element methods. With her guidance and constructive criticism throughout this year I have learned more than I ever hoped to.

I cannot thank enough Simon Sticko, without whom this project would not have gone this far. His insights in mathematical analysis, numerical methods and programming techniques paved the way for my project since my first day of work. His willingness and ability to help were essential for the development of this thesis. Gunilla and Simon's genuine interest in my progress motivated me to work as hard as I could.

I wish to express my sincere thanks to Prof. Andreas Hellander for his suggestions and support on reaction diffusion problems and methods to solve them. I am very grateful to Prof. Per Lötstedt for accepting the role of reviewer of this thesis.

I would like to extend my thanks to Uppsala University for supporting this work and for being a place where I could find help whenever I asked for it.

From the Federal University of Santa Catarina, I thank my advisor, Prof. Luismar Porto, for his advice and support over this exchange period.

I would like to thank my parents, Evandro and Sandra, and my brother Alexandre, for supporting me unconditionally during this period, enduring the distance of one more year of studies abroad.

I have immense thanks to my girlfriend Amanda Floriani, my eternal partner and friend, for bearing with me all the stresses of distance, for motivating me to go further in my work whenever I needed. I thank her for our endless conversations which made my days much more enjoyable, for visiting me and making my experience in Europe truly unforgettable.

Finally, I thank the Erasmus programme for providing me with the opportunity of this exchange program in Sweden, along with the financial support which was essential for my studies.

Contents

Acknowledgments	v
1 Introduction	11
1.1 Thesis Outline	12
1.2 Nitsche's Method	12
1.3 Laplace-Beltrami Problem	13
1.4 Reaction-Diffusion Problems	14
2 Theory	16
2.1 Nitsche's Method for Dirichlet and Neumann Boundary Value Problems	16
2.1.1 Poisson Model Problem	16
2.1.2 Finite Element Discretization	18
2.2 Mesh Characteristics	19
2.3 Finite Element Formulation	21
2.4 Derivation of Linear System of Equations	21
2.5 Parametric Mapping and Numerical Integration	22
2.5.1 Numerical Evaluation of integrals	23
2.6 Implementation Details	25
2.6.1 deal.II Finite Element Library	25
2.6.2 Implicit representation of the surface	25
2.6.3 Integral Evaluation on Cut-Cells	30
3 Test Cases	34
3.1 Poisson Problem	34
3.1.1 Problem setup	34
3.1.2 Convergence Analysis	34
3.2 Laplace-Beltrami Problem	35
3.2.1 The Laplace-Beltrami Model Problem	35
3.2.2 Approximation of the Domain	36
3.2.3 The Laplace-Beltrami Operator	37
3.2.4 Finite Element Formulation	38
3.2.5 Problem Setup	39
3.2.6 Convergence Analysis	39
3.3 Reaction Diffusion Problem	40
3.3.1 The Reaction Diffusion Problem	40
3.3.2 Finite Element Formulation	41

3.3.3	Problem Setup	43
3.3.4	Mass Conservation	44
4	Results	45
4.1	Poisson's Problem	45
4.1.1	Solution	45
4.1.2	Convergence Analysis	46
4.1.3	Matrix conditioning analysis	46
4.2	Laplace-Beltrami Problem	48
4.2.1	Solution	48
4.2.2	Convergence Analysis	48
4.2.3	Matrix conditioning analysis	49
4.3	Reaction-Diffusion Problem	49
4.3.1	Solution	49
4.3.2	Mass conservation	50
5	Discussion	52
5.1	Poisson's Problem	52
5.2	Laplace-Beltrami Problem	52
5.3	Reaction-Diffusion Problem	53
6	Conclusions	54
6.1	Conclusive Remarks	54
6.2	Suggestions and Future Work	54
7	References	56

List of Figures

2.1	Triangulation with cells $K \in G_h$ highlighted (■)	19
2.2	Faces $F \in \mathcal{F}_G$ are shown in thick red lines. Faces of $\mathcal{F}_S \subset \mathcal{F}_G$ have three small crossed lines.	20
2.3	Example of a cell cut by the interface and the resulting level-set values on the nodes.	26
2.4	The level-set function ((2.38)) projected onto a mesh, with the zero values represented by the isocontour in white.	27
2.5	Diagram of a domain T_h cut by an interface Γ given by the level-set function $\phi = x^2 + y^2 - 1$ and the resulting classification of cells and domain characterization.	28
2.6	Example of a cell intersected by the boundary and the resulting cut-cell, with nodes ordered counterclockwise.	29
3.1	Representation of the described domains. $U(\Gamma)$ is the green region where a unique closest point $p(\mathbf{x}) \in \Gamma$ is defined for each $\mathbf{x} \in U(\Gamma)$	37
3.2	Diagram of the initial setup of the reaction-diffusion problem.	43
4.1	Solution for the Poisson problem with B.C. imposed weakly with Nitsche's method.	45
4.2	Convergence analysis in H^1 (orange) and L_2 (blue) norms. The dashed lines are proportional h (orange) and h^2 (blue).	46
4.3	Condition number as a function of the inverse of the cell diameter. The dashed line is proportional to h^{-2}	46
4.4	Condition number as a function of γ_1	47
4.5	Plot of the L_2 error by the stabilization parameter.	47
4.6	Solution on a boundary defined by the level set function.	48
4.7	Convergence analysis in H^1 (orange) and L_2 (blue) norms. The dashed lines are proportional h (orange) and h^2 (blue).	49
4.8	Dependence of condition number with the element size. The dashed line is proportional to h^{-2}	49
4.9	Time snaps of concentration of component A. The boundary is represented by the level set function in white.	50

4.10	Concentration profile of component B on the boundary Γ_h	50
4.11	Mass conservation over the time integration interval. .	51

1. Introduction

There is a growing interest in studying problems involving phenomena that take place on interfaces and in bulk domains and how to accurately mathematically represent them. Some examples include the modeling of surfactants in oil recovery processes and other applications, the simulation of the behaviour of cells including the diffusion of metabolites on the surface and their generation/consumption through reactions. The solution in the bulk domain may couple with the interface through diffusion or adsorption from the bulk to the surface and in the reverse way, transport from the surface to the bulk.

The evolution of the boundary may also depend on the distribution of the concentration over the surface due to the modification of the interfacial forces.

As these processes are intrinsically dependent on the surface shape and behaviour, the interface must be accurately represented. The Finite Element Method has been successfully used to represent such phenomena and benefit from efficient computations over complex geometries and meshes.

Solving a coupled bulk-surface system of equations on an evolving domain can be quite challenging from a numerical point of view. The domain may change drastically, for example stretching, breaking or coalescing with other domains.

In the standard FEM, the mesh is generated as to fit the domains accordingly, in which the boundary is represented by the mesh's facets. In order to account for the constant geometrical change, a continuous remeshing of the interface is necessary. This is an expensive process and can account for large part of the computational effort and time.

The use of meshes not conforming to the surface has been studied as an alternative to account for the boundary representation on interface problems. In this case, one takes advantage of having a fixed background mesh with an interface that is allowed to be arbitrarily located.

Among these methods are the Immersed FEM [32, 33], where special basis functions are constructed for the elements that are intersected by the boundary; the extended FEM (XFEM) [12, 19], in which the finite element space is enriched with particular discontinuous basis functions; and the unfitted FEM [21], where a discrete solution is obtained from separate solutions from each subdomain, and the interface condition or essential boundary conditions are weakly enforced using an extended

method derived from Nitsche [39]. The present work focuses on developing a cut-cell technique based on the aforementioned method by Hansbo and Hansbo [21] to model coupled bulk-surface problems involving reaction and diffusion. The Finite Element library deal.II [3] will be used for the computational implementation.

1.1 Thesis Outline

The remainder of this chapter is dedicated to introduce the problems proposed to test the cut-cell method.

Chapter 2 presents the basic framework for solving PDE's using the FEM. The classical Poisson equation using Nitsche's method to impose boundary conditions with the cut-cell approach is used to demonstrate the method.

In chapter 3, we introduce the mathematical formulation of each test case, the problem setup and which analysis were performed to test the cut-cell method.

Chapter 4 reports the results of each test case and in chapter 5 the results are discussed and evaluated.

Chapter 6 concludes the report and presents suggestions for improvement and future work.

1.2 Nitsche's Method

In his classical paper, Nitsche [39] introduced a novel method to incorporate Dirichlet boundary conditions weakly, i.e., without specifying nodal values on the boundary. The idea gained strength in the Finite Element community as an alternative to the Lagrange multiplier and penalty methods. The method presents the advantage of generality in the treatment of interface problems, where arbitrary degree polynomial approximations, different geometrical grids and physical models can be used on arbitrary sides of a given interface [23].

Nitsche's method can be interpreted as an improvement of the penalty method, in the sense that it also imposes boundary conditions via penalization, but introduces new terms that maintain consistency and coercivity of the bilinear form. The resulting stiffness matrix avoids the ill-conditioning and lack of consistency that the penalty method exhibits. In contrast to the mentioned methods, Nitsche's method lacks a straightforward generalization for its implementation. The weak form and choice of penalization parameters depend heavily on the set of partial differential equations and associated boundary conditions. For a thorough

comparison of the Lagrange multiplier, penalty and Nitsche's method, see [18].

Nitsche's method has been extended to cover various types of boundary, such as interface problems [1, 20, 22], Robin and Neumann boundary conditions [28]. Moreover, the method is a good alternative to impose boundary conditions in unfitted finite element methods, where the imposition of Dirichlet values on prescribed nodes may be inconvenient. In this context, the method has been successfully used to solve problems on composite meshes [5, 24, 34], in the extended FEM method [13, 46] and in the cut-cell method [8, 9, 11, 21].

1.3 Laplace-Beltrami Problem

Mathematical models involving transport phenomena between an interface and a bulk domain appear naturally in several situations, such as fluid dynamics, biological applications, colloidal and surface phenomena. Examples include the modeling of multiphase flow where surfactants play an important role, such as enhanced oil recovery, industrial emulsification, liquid-liquid extraction and several other applications [27, 41]. Surfactants may induce tangential surface tension stresses causing the Marangoni effect [36]. In such situations, soluble surfactants are dissolved in the bulk fluid but also get absorbed on the interface. Mathematical models therefore couple surface equations with equations in a domain which includes the surface. In general, if the problem includes a domain which can be considered thin enough, one can simplify the model by writing a formulation with PDE's on a lower dimensional geometry, for example on a curve or in a surface.

A well known approach for solving elliptic equations on surfaces is given in [15], where it is proposed a piecewise polygonal approximation for the surface using a finite element space on a discretization of this surface. For a comprehensive review about finite element methods for solving partial differential equations on surfaces, see [16] and the references therein.

In transient problems the interface may experience constant geometrical changes. Using a standard FEM requires the computation of a new triangulation at each time step, which becomes computationally cumbersome. In this context, it is advantageous to use a computational method that allows the interface to cut arbitrarily through a background mesh. An unfitted approach for the problem involving the Laplace-Beltrami operator was addressed in [42] and more recently by Burman, et al., [8], where they discussed a general framework to solve PDE's on surfaces using the FEM on cut meshes under the approach of the so called Cut-FEM method. Unfitted methods may result in ill-conditioning of the

stiffness matrix due to the arbitrary nature of the boundary cut over the background mesh. This issue has been studied for problems with the Laplace-Beltrami operator as well, cf. [10, 41]. We propose a test case of a pure diffusion problem on a surface, which can be generalized by the equation

$$-\Delta_\Gamma u = f$$

evaluated on a closed one dimensional domain embedded in the two-dimensional space. The test case proposed presents a cut-cell method for solving PDE's on surfaces based on the formulation reported in [8].

1.4 Reaction-Diffusion Problems

Problems involving reaction-diffusion equations appear frequently in many situations in various scientific fields, especially in biochemical and engineering applications. Typically, a chemical reaction arises from the random encounter of molecules and results in the formation of new chemical species and/or energy. In most cases, the local generation or consumption of a chemical element creates a gradient of concentration which is the driving force of the process of diffusion. The diffusion phenomenon is found in several scientific applications and is paramount to understand underlying metabolic processes occurring in living organisms.

In this work, we focus on reaction-diffusion process occurring in biological cells, where several reactions take place both in the cytoplasm and in the membrane and diffuse throughout the cell body. One typical system occurring in the *Escherichia coli* microorganism is the regulation of the division site, which can be determined by the complex dynamic of Min proteins. The Min oscillation mechanism has been extensively studied and modeled via deterministic [25, 26] or stochastic [43] approaches.

First, we introduce a time-dependent model of reaction-diffusion in a cell in two-dimensions, with a coupled bulk-surface system. In such systems, there are components that live in the cell cytoplasm, represented as a bulk domain (Ω), and species that are membrane-bound, i.e., are restricted to the membrane sub-domain ($\Gamma = \partial\Omega$). The model for reaction-diffusion systems occurring on a coupled bulk and membrane system can be generalized as [40]:

$$\frac{\partial u_i}{\partial t} = -\nabla \cdot D_i \nabla u_i + r_i, \quad u_i \in H^1(\Omega)$$

For species i living on the bulk domain (Ω) and

$$\frac{\partial u_j}{\partial t} = -\nabla_\Gamma \cdot D_j \nabla_\Gamma u_j + r_j, \quad u_j \in H^1(\Gamma)$$

for components j bound to the membrane domain (Γ). D_k are the diffusion coefficients, u_k the concentrations, r_k the rates of reaction and ∇_Γ is the Laplace-Beltrami operator.

2. Theory

The Finite Element Method (FEM), also known as Finite Element Analysis (FEA) is a numerical method that has been successfully used to solve many types of boundary value problems. Initially used mainly to solve elasticity and structural analysis problems in civil and aeronautical engineering [51], the FEM has quickly expanded its range of applications and is now generalized for modeling a plethora of engineering problems, e.g., fluid dynamics, electromagnetism, heat and mass transfer.

In this section, we introduce the procedure for solving FEM problems using Nitsche's method to impose boundary conditions. The classical Poisson problem is used to introduce the method, and also serves as a first test case for the implementation of the cut-cell technique. The interested reader is referred to [7, 31, 51] for a comprehensive introduction to the Finite Element theory.

2.1 Nitsche's Method for Dirichlet and Neumann Boundary Value Problems

2.1.1 Poisson Model Problem

The classical Poisson problem will be used as a test case for the implementation of the cut-cell method using Nitsche's approach. Also, we use this problem to introduce the general method for solving PDEs using the finite element method.

Let Ω be a bounded domain in \mathbb{R}^2 with a smooth interface Γ denoting the boundary of the domain. The interface may contain Dirichlet (Γ_D) and Neumann (Γ_N) parts, such that $\Gamma = \Gamma_D \cup \Gamma_N$, with outward pointing unit normal \mathbf{n}_Γ . The model problem is defined as follows:

$$-\Delta u = f \text{ in } \Omega \tag{2.1}$$

$$u = g_D \text{ on } \Gamma_D$$

$$\mathbf{n}_\Gamma \cdot \nabla u = g_N \text{ on } \Gamma_N.$$

Let X be the computational mesh or a subset thereof, and Y a subset of the boundary such that $Y \subset \Gamma$. The L^2 -inner product on $X \subset \mathbb{R}^{dim}$ with associated norm $\|u\|_X = (u, u)_X^{\frac{1}{2}}$ is

$$(u, v)_X = \int_X uv \, d\mathbf{x}, \quad (2.2)$$

and the L^2 -inner product over $Y \subset \mathbb{R}^{dim-1}$ with associated norm $\|u\|_Y = \langle u, v \rangle_Y^{\frac{1}{2}}$ is

$$\langle u, v \rangle_Y = \int_Y uv \, ds. \quad (2.3)$$

The derivation of the Finite Element Method starts by rewriting the set of PDE's that describe the problem in a computable form, the so called weak form.

We multiply (2.1) by a test function $v \in V$:

$$V = H^1(\Omega)$$

and integrate using Green's formula. Nitsche's method adds new terms to ensure that the stiffness matrix is symmetric and positive definite, in addition to penalty terms containing parameters γ_D and γ_N that are used to impose boundary conditions. The bilinear form becomes:

$$a(u, v) = \int_{\Omega} \nabla u \cdot \nabla v \, d\Omega - \underbrace{\int_{\Gamma_D} v \mathbf{n}_{\Gamma} \cdot \nabla u \, d\Gamma}_{\text{consistency}} - \underbrace{\int_{\Gamma_D} u \mathbf{n}_{\Gamma} \cdot \nabla v \, d\Gamma}_{\text{symmetrization}} + \underbrace{\int_{\Gamma_D} \gamma_D h^{-1} u v \, d\Gamma + \int_{\Gamma_N} \gamma_N h \mathbf{n}_{\Gamma} \cdot \nabla u \, \mathbf{n}_{\Gamma} \cdot \nabla v \, d\Gamma}_{\text{penalization}}, \quad (2.4)$$

where h is the element size. The second term arises naturally from the integration by parts and ensures the consistency of the method. The third term lets the problem be symmetric, and the last terms come from the penalization necessary to guarantee stability [2]. The right hand side becomes:

$$L(v) = \int_{\Omega} f v \, d\Omega + \int_{\Gamma_N} g_N v \, d\Gamma + \underbrace{\int_{\Gamma_D} g_D \mathbf{n}_{\Gamma} \cdot \nabla v \, d\Gamma}_{\text{symmetrization}} + \underbrace{\int_{\Gamma_D} g_D \gamma_D h^{-1} v \, d\Gamma + \int_{\Gamma_N} g_N \gamma_N h \mathbf{n}_{\Gamma} \cdot \nabla v \, d\Gamma}_{\text{penalization}}, \quad (2.5)$$

with similar properties¹. For a thorough analysis of the deduction of weak formulation and the properties of each term, see [28].

The weak formulation then reads: find $u \in V$ such that:

$$a(u, v) = L(v), \forall v \in V. \quad (2.6)$$

2.1.2 Finite Element Discretization

The finite element process involves stepping from a continuous to a discrete form of a problem. This can be achieved by constructing finite dimensional subspaces V_h of spaces $V = H^1(\Omega)$ which can approximate the solution u and make the problem computable. The discretization of the weak formulation (2.6) consists in finding approximations $u_h \in V_h$.

The next step is to choose a discretization for the unknown function $u_h \in U_h \subset V$ and for the test function $v_h \in V_h \subset V$. Let T_h be a triangulation (or equivalently, mesh) of Ω and let V_h be the space of bilinear elements on T_h . In this work we use quadrilateral Lagrange finite elements, which are represented by K . The triangulation or mesh is given by $T_h = \{K\}$. The space of bilinear polynomials can be represented by $Q_1(K)$ and is defined by:

$$Q_1(K) = \{\phi : \phi = c_0 + c_1 x + c_2 y + c_3 x y, (x, y) \in K, c_i \in \mathbb{R}\}, \quad (2.7)$$

where $c_i, i = 0, 1, 2, 3$ are uniquely defined by the nodal values of the degrees of freedom. By requiring v to belong to Q_1 and to be continuous along elements, we obtain the space of continuous bilinear polynomials V_h :

$$V_h = \{v : v \in C^0(\Omega), v|_K \in Q_1(K), \forall K \in T_h\}. \quad (2.8)$$

We follow the Galerkin approach, where the same discrete space is chosen for the unknown and test spaces, $U_h = V_h$. Finally, u_h can be written as the linear combination

$$u_h = \sum_{j=0}^{N-1} U_j \phi_j, \quad (2.9)$$

where N is the number of nodes of the mesh and $\mathbf{U} = [U_0, \dots, U_{N-1}]$ represents the vector of unknowns to be determined.

¹Note that by eliminating consistency, symmetrization and penalization terms , forcing $v|_{\Gamma_D} = 0$ and imposing Dirichlet conditions strongly, the problem becomes the standard FEM formulation.

2.2 Mesh Characteristics

In this section, we introduce the specific characteristics of the mesh and notation that will be used throughout the work.

Consider a computational triangulation T_h such that $T_h = \{\bar{K}\}$, where \bar{K} denotes a regular quadrilateral element. The intersection of two elements is always the empty set, a vertex or a face (in 2D, corresponding to an edge). For a representation of the described notation, see Figure (2.1).

It is assumed that the domain Ω lies entirely inside the triangulation T_h ($\Omega \subset T_h$), but not that the facets of the mesh T_h are fitted to the boundary of Ω . We define by K all elements \bar{K} contained in Ω , such that $\bar{K} \cap \Omega = \emptyset$ are later excluded from the computation. For all elements $K \in T_h$, we have $K \cap \Omega \neq \emptyset$, meaning that K is either completely inside Ω or partially contained in Ω and crossed by Γ . The domain covered by the mesh T_h is represented by Ω_T .

The notation for mesh parameters are as follows: The set of elements intersected by the interface is denoted by $G_h = \{K \in T_h : K \cap \Gamma \neq \emptyset\}$. Let $\Gamma_K = \Gamma \cap K$ be the part of Γ in an element $K \in G_h$. The diameter of K is given by h_K , and $h = \max_{K \in T_h} h_K$.

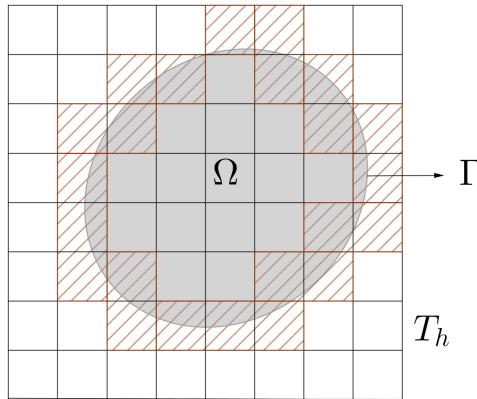


Figure 2.1. Triangulation with cells $K \in G_h$ highlighted (■)

The following assumptions were made regarding the mesh and the interface:

1. The grid is formed by uniformly sized squares, so that $h = h_K$, $\forall K \in T_h$ and consequently the triangulation is non-degenerate.
2. The mesh is fine enough so that Γ intersects each element boundary ∂K exactly twice, and each facet at most once.

The assumptions are not very restrictive, in the sense that they ensure that the curvature of the boundary Γ is well resolved by the mesh.

Stabilization

As the boundary cuts arbitrarily through elements, it may approach element boundaries and it is not unusual to observe the fraction of the cut element $K \in G_h$ inside the domain Ω to be very small. For situations like this, the stiffness matrix becomes ill-conditioned and may cause failure to direct or iterative linear solvers [49]. Alternatives to solve this problem include the use of a scaled matrix, as described in [41], or the use preconditioning techniques as in [52]. In this work we will use the method introduced by Burman and Hansbo [9], where they stabilized the classical method by Nitsche for the imposition of inhomogeneous Dirichlet boundary conditions with penalty terms for normal-derivative jumps between pairs of elements, where at least one is intersected by the interface. The stabilization terms are represented by $j(u, v)$ and are added to the stiffness matrix as shown in section (3.1).

Prior to outlining the stabilization terms, it is convenient to define the jump terms and the following relevant sets. The set \mathcal{F}_G of element faces contains all faces which belong to an element $K \in G_h$ and an immediate neighbor K' , such that K and K' have a face F in common: $F = K \cap K'$. In other terms, the set \mathcal{F}_G contains all faces of elements $K \in G_h$ except those having both nodes in Ω .

The set \mathcal{F}_S is a subset of \mathcal{F}_G containing all faces of an element $K \in G_h$ which are shared with a neighbor $K'' \in G_h$, such that $F = K \cap K''$. In other terms, the set \mathcal{F}_S contains all faces crossed by the boundary. Figure (2.2) illustrates the described sets.

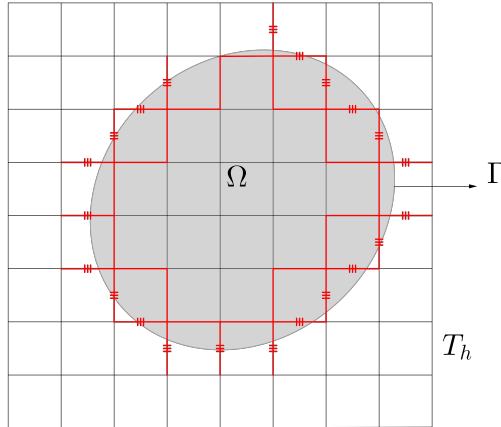


Figure 2.2. Faces $F \in \mathcal{F}_G$ are shown in thick red lines. Faces of $\mathcal{F}_S \subset \mathcal{F}_G$ have three small crossed lines.

The jump of the gradient of $v_h \in V_h$ is defined by

$$[\nabla v_h] = \mathbf{n}_F \cdot \nabla v_h|_K - \mathbf{n}_F \cdot \nabla v_h|_{K'}, \quad (2.10)$$

where \mathbf{n}_F denotes a unit normal to the face F with fixed and arbitrary orientation.

2.3 Finite Element Formulation

The stabilized finite element discretization of the Poisson equation using Nitsche's method becomes, as described in [9]:

Find $u_h \in V_h$ such that

$$A_h(u_h, v_h) = L(v_h), \forall v_h \in V_h, \quad (2.11)$$

where

$$\begin{aligned} L(v_h) &= (f, v_h)_\Omega + \langle g_D, \gamma_D h^{-1} v_h - \mathbf{n}_\Gamma \cdot \nabla v_h \rangle_{\Gamma_D} \\ &\quad + \langle \Gamma_N, v_h + \gamma_N h \mathbf{n}_\Gamma \cdot \nabla v_h \rangle_{\Gamma_N}, \end{aligned} \quad (2.12)$$

and

$$A_h(u_h, v_h) = a_h(u_h, v_h) + \gamma_1 h j(u_h, v_h), \quad (2.13)$$

with

$$\begin{aligned} a_h(u_h, v_h) &= a(u_h, v_h) - \langle \mathbf{n}_\Gamma \cdot \nabla u_h, v_h \rangle_{\Gamma_D} - \langle \mathbf{n}_\Gamma \cdot \nabla v_h, u_h \rangle_{\Gamma_D} + \\ &\quad \langle \gamma_D h^{-1} u_h, v_h \rangle_{\Gamma_D} + \langle \gamma_N h \mathbf{n}_\Gamma \cdot \nabla u_h, \mathbf{n}_\Gamma \cdot \nabla v_h \rangle_{\Gamma_N}, \end{aligned} \quad (2.14)$$

and the stabilization term:

$$j(u_h, v_h) = \sum_{F \in \mathcal{F}_G} \langle [\nabla u_h], [\nabla v_h] \rangle_F. \quad (2.15)$$

2.4 Derivation of Linear System of Equations

Applying (2.9) to (2.11)-(2.15), we obtain a system of N linear algebraic equations for the unknowns U_j 's:

$$\begin{aligned} A_{ij} &= (\nabla \phi_j, \nabla \phi_i)_\Omega - \langle \mathbf{n}_\Gamma \cdot \nabla \phi_j, \phi_i \rangle_{\Gamma_D} - \langle \mathbf{n}_\Gamma \cdot \nabla \phi_i, \phi_j \rangle_{\Gamma_D} + \\ &\quad \langle \gamma_D h^{-1} \phi_j, \phi_i \rangle_{\Gamma_D} + \langle \gamma_N h \mathbf{n}_\Gamma \cdot \nabla \phi_j, \mathbf{n}_\Gamma \cdot \nabla \phi_i \rangle_{\Gamma_N} + \gamma_1 h j(\phi_i, \phi_j) \end{aligned} \quad (2.16)$$

and

$$L_i = (f, \phi_i)_\Omega + \langle g_D, \gamma_D h^{-1} \phi_i - \mathbf{n}_\Gamma \cdot \nabla \phi_i \rangle_{\Gamma_D} + \langle g_N, \phi_i + \gamma_N h \mathbf{n}_\Gamma \cdot \nabla \phi_i \rangle_{\Gamma_N}. \quad (2.17)$$

The linear system becomes:

$$A U = L, \quad (2.18)$$

where A is known as the **stiffness matrix**, L is the **load vector** and U is the **solution vector**.

2.5 Parametric Mapping and Numerical Integration

In the finite element method, the basis functions are usually defined locally, on a reference element \hat{K} . The reference element is linked to the actual grid cell K by a mapping Φ . This procedure greatly facilitates the code implementation of numerical integration over cell and boundary elements. The mapping Φ can be defined as a transformation $\mathbf{x} = \Phi(\hat{\mathbf{x}})$ that maps the reference element to the physical space. The coordinates of the reference element are represented by $\hat{\mathbf{x}} = (\hat{x}_0, \dots, \hat{x}_{dim}) \in \mathbb{R}^{dim}$, whilst the natural space is represented by the Cartesian coordinates $\mathbf{x} = (x_0, \dots, x_{dim}) \in \mathbb{R}^{dim}$. The transformation is then defined as:

$$\Phi : \hat{K} \rightarrow K \quad (2.19)$$

$$\mathbf{x} = \Phi(\hat{\mathbf{x}}) \quad (2.20)$$

It is convenient to introduce the notation for the Jacobian $J_K(\hat{\mathbf{x}}) = \nabla \Phi(\hat{\mathbf{x}})$, in order to describe the use of the mapping in different situations. For the two-dimensional case ($dim = 2$), $\mathbf{x} = (x, y)$, and

$$J_K(\hat{\mathbf{x}}) = \begin{pmatrix} \frac{\partial x}{\partial \hat{x}} & \frac{\partial x}{\partial \hat{y}} \\ \frac{\partial y}{\partial \hat{x}} & \frac{\partial y}{\partial \hat{y}} \end{pmatrix} \quad (2.21)$$

To guarantee that the Jacobian be invertible, the mapping must be smooth and invertible. In terms of shape functions, the mapping is defined as:

$$\mathbf{x} = \Phi(\hat{\mathbf{x}}) = \sum_{i=0}^{n-1} \hat{\phi}_i(\hat{\mathbf{x}}) \mathbf{x}_i \quad (2.22)$$

where $\mathbf{x}_i = (x_i, y_i)$ are the coordinates of the i th node in the physical space, $\hat{\phi}_i$ are the finite element hat functions in the reference element

and n is the number of nodes in the cell. In this work, the reference element is given by $[0, 1]^2$.

In addition to the described mapping from physical to reference element, it is convenient to define a mapping to integrate boundary elements, in order to integrate boundary terms such as those appearing in eqs. (2.11)-(2.15). In two-dimensions, surface elements are lines, such as the edges or faces of a cell. Let $\Gamma \subset \mathbb{R}^{dim}$ be a hyperspace embedded in the physical space representing the faces of an element K . A mapping Σ transforms a point in Γ to the reference line space $\hat{\Gamma} \subset \mathbb{R}^{dim-1}$. The mapping can be written as:

$$\Sigma : \hat{\Gamma} \rightarrow \Gamma \quad (2.23)$$

$$\mathbf{t} \in \hat{\Gamma} \rightarrow \mathbf{x} = \Sigma(\mathbf{t}) \in \Gamma. \quad (2.24)$$

The parameter space is defined by independent parameters

$$\mathbf{t} = (t_0, \dots, t_{dim-1}) \in \mathbb{R}^{dim-1} \quad (2.25)$$

and for the natural space, Cartesian coordinates are used. For the two-dimensional case, one can simplify $\mathbf{t} = t$, and the parameters are chosen to lie on the unit interval. We choose a linear mapping, that can be obtained by:

$$\mathbf{x} = \Sigma(t) = \mathbf{x}_0 + (\mathbf{x}_1 - \mathbf{x}_0)t, \quad (2.26)$$

where $\mathbf{x}_0 = (x_0, y_0)$ and $\mathbf{x}_1 = (x_1, y_1)$ represent the coordinates of the first and last nodes of the face in physical space, determined counterclockwise. The Jacobian of the transformation is represented by $J_\Gamma(t) = \nabla \Sigma(t)$ and in two-dimensions is given by:

$$J_\Gamma(t) = (x_1 - x_0, y_1 - y_0) \quad (2.27)$$

2.5.1 Numerical Evaluation of integrals

In order to assemble the finite element matrices one must perform the numerical integration of terms appearing in eqs. (2.16)-(2.17). It is convenient to perform a mapping as previously described and evaluate these integrals on the reference element (K) are of the type:

$$A'_{ij} = \int_K \rho(\mathbf{x}) \, dK, \quad (2.28)$$

where $\rho(\mathbf{x})$ is any function to be integrated over the domain, e.g., $\rho(\mathbf{x}) = \nabla \phi_j \cdot \nabla \phi_i$. The mapped formulation is:

$$A'_{ij} = \int_{\hat{K}} \hat{\rho}(\hat{\mathbf{x}}) \cdot |J_K(\hat{\mathbf{x}})| d\hat{K}, \quad (2.29)$$

where $|J_\Omega(\hat{\mathbf{x}})|$ is the determinant of the Jacobian. For example, for the bilinear term,

$$a_{ij} = \int_K \nabla \phi_j \cdot \nabla \phi_i dK = \int_{\hat{K}} J_K^{-1}(\hat{\mathbf{x}}) \hat{\nabla} \hat{\phi}_i(\hat{\mathbf{x}}) \cdot J_K^{-1}(\hat{\mathbf{x}}) \hat{\nabla} \hat{\phi}_j(\hat{\mathbf{x}}) |J_K(\hat{\mathbf{x}})| d\hat{K}. \quad (2.30)$$

Integrals over boundary elements are generalized as:

$$b_i = \int_{\Gamma} \sigma(\mathbf{x}) d\Gamma, \quad (2.31)$$

where $\sigma(\mathbf{x})$ can be any function to be integrated over the boundary, e.g. $\sigma(\mathbf{x}) = g_N \phi_i$. The mapped equation is:

$$b_i = \int_0^1 \hat{\sigma}(t) |J_\Gamma(t)| d\hat{\Gamma}, \quad (2.32)$$

where $|J_\Gamma(t)|$ represents the length of the face. The integration is then numerically performed using Gauss quadrature rule, with $n_{K,q} = 4$ points for integration over Ω and $n_{\Gamma,q} = 2$ points for integration over faces. The integration of cut-elements becomes:

$$A'_{ij} = \sum_{q=0}^{n_{K,q}-1} \hat{\rho}(\hat{\mathbf{x}}_q) |J_K(\hat{\mathbf{x}}_q)| w_q. \quad (2.33)$$

For example, the integration of the bilinear term is evaluated as:

$$a_{ij} = \sum_{q=0}^{n_{K,q}-1} J_K^{-1}(\hat{\mathbf{x}}_q) \hat{\nabla} \hat{\phi}_i(\hat{\mathbf{x}}_q) \cdot J_K^{-1}(\hat{\mathbf{x}}_q) \hat{\nabla} \hat{\phi}_j(\hat{\mathbf{x}}_q) |J_K(\hat{\mathbf{x}}_q)| w_q \quad (2.34)$$

For boundary terms,

$$b_i = \sum_{q=0}^{n_{\Gamma,q}-1} \hat{\sigma}(t_q) |J_\Gamma(t_q)| w_q \quad (2.35)$$

The subscript q indicates the quadrature point where the function is evaluated. Weights w_q and points of integration $\hat{\mathbf{x}}_q$ and t_q were computed as described in [44].

2.6 Implementation Details

2.6.1 deal.II Finite Element Library

deal.II² (Differential Equations Analysis Library) is an open-source library written in C++ to solve partial differential equations using the finite element method. It was introduced in 1999 in [4] and its current version, 8.2, was released in January, 2015 [3].

In order to solve the FEM problem, deal.II requires a low level description of the problem to be solved, i.e., it requires user specification to define a PDE problem in the weak form and the corresponding assembly of linear systems. The lower level description facilitates direct manipulation of the process of setting up and solving the finite element problem. This is specially advantageous in the cut-cell framework, since it permits direct access to mesh entities and enables an easy manipulation of system matrices entries.

A simplified algorithm for the solution of a problem using deal.II is given in (2.1):

Algorithm 2.1 Setting up a problem in deal.II

```
Grid generation
    Generate triangulation
    Setup and associate DoF's to the triangulation
Matrices assembly
    Define type of finite element
    Define quadrature rule
    Loop over cells
        Loop over DoF's
            Compute  $A_{ij}, b_i$ 
    Assemble  $A, b$ 
Solve
    Solve  $U = A^{-1}b$ 
Output solution
```

2.6.2 Implicit representation of the surface

One of the crucial steps of the problem discretization in the cut-cell FEM framework is to accurately represent the geometry of the boundary in a background mesh. The implicit representation of curves and surfaces has shown to be efficient in several computational applications, and its characterization through a geometrical description is a versatile and simple way to construct the discretization. Moreover, it is advantageous to

²<http://www.dealii.org/>

choose a method that can be extended for more complex cases, for example, problems with moving or evolving boundaries.

In this work, the surface or boundary of the problem is represented by the standard level-set method [48]. The location of the boundary is described by the zero level set of a signed distance function, and the implicit representation of a dim -dimensional surface is:

$$\Gamma = \left\{ \mathbf{x} \in \mathbb{R}^{dim+1} \mid \phi(\mathbf{x}) = 0 \right\}, \quad (2.36)$$

where dim is the dimension of the hypersurface and \mathbf{x} is a point on the surface defined by the function $\phi : \mathbb{R}^{dim+1} \rightarrow \mathbb{R}$.

The representation can decompose the given domain T_h into an inner subdomain Ω_0 , the common interface Γ and an outer part Ω_2 , such that $\forall \mathbf{x} \in T_h$:

$$\begin{cases} \phi(\mathbf{x}) > 0 & \iff \mathbf{x} \in \Omega_2 \\ \phi(\mathbf{x}) = 0 & \iff \mathbf{x} \in \Gamma \\ \phi(\mathbf{x}) < 0 & \iff \mathbf{x} \in \Omega_0. \end{cases} \quad (2.37)$$

For example, the unit circle geometry in 2D, used in this work, can be represented by the signed distance function:

$$\phi(\mathbf{x}) = \sqrt{x^2 + y^2} - 1. \quad (2.38)$$

The function is projected onto the computational mesh so that each node is marked with a level-set value. An example of implicit representation of the surface on a cell is shown in Figure (2.3).

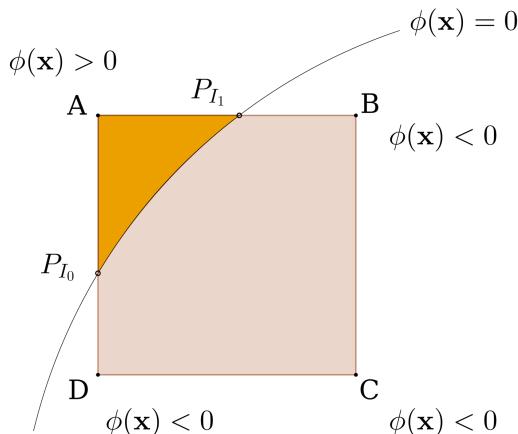


Figure 2.3. Example of a cell cut by the interface and the resulting level-set values on the nodes.

An example of the projection of the level-set function on a mesh is represented in Figure (2.4). The isocontour of zero values, representing $\phi(\mathbf{x}) = 0$, is obtained by interpolation and is shown as a white line.

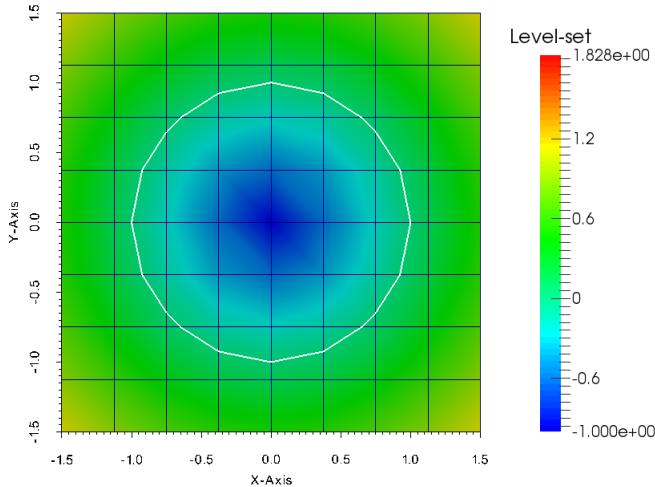


Figure 2.4. The level-set function (2.38) projected onto a mesh, with the zero values represented by the isocontour in white.

Boundary representation

The first step for the interface approximation is to project the level-set function onto the triangulation and identify elements that are fully contained in Ω_0 , fully contained in Ω_2 or those which are cut by the interface Γ , $K \in G_h$. To identify these elements, one has to analyze the projected values of the level-set function on the nodes of the element. In order to facilitate the identification of elements, the following classification is proposed:

Nodes:

- Type 0 nodes: node $P_0 = (\mathbf{x})$, $\phi(\mathbf{x}) < 0$
- Type 1 nodes: node $P_1 = (\mathbf{x})$, $\phi(\mathbf{x}) = 0$
- Type 2 nodes: node $P_2 = (\mathbf{x})$, $\phi(\mathbf{x}) > 0$.

Cells:

It is convenient to evaluate the number of nodes of each type inside a cell as such: N_{P_0} , N_{P_1} and N_{P_2} for type 0, 1 and 2 nodes respectively. A diagram of element characterization is shown in Figure (2.5).

- Type 0 cell: a cell is considered to be "inside" if it is completely inside the subdomain Ω_0 if it possesses all nodes of the type 0 or type 1.
 - ▷ $K \in \Omega_0$ if $N_{P_2} = 0$ and $N_{P_0} + N_{P_1} = 4$.
- Type 1 cell: a cell is classified as an "interface" cell if it has at least one node of type 0 and one node of type 2.
 - ▷ $K \in G_h$ if $N_{P_1} \geq 0$ and $4 > N_{P_0} > 0$ and $4 > N_{P_2} > 0$.
- Type 2 cell: a cell is classified as an "outside" cell if all its nodes are of the type 1 or 2.
 - ▷ $K \in \Omega_2$ if $N_{P_0} = 0$.

Faces:

It is relevant to characterize only faces belonging to elements of Type 1 cell, i.e., $K \in G_h$. Faces are characterized as:

- Type 0 faces: faces $F \in \mathcal{F}_G \setminus \mathcal{F}_S$, i.e., $F = K \cap K'$, where $K \in G_h$ and $K' \in \Omega_0$.
 - ▷ if at least one node is of type P_0 and the other is of type P_0 or P_1 .
- Type 1 faces: faces intersected by Γ , i.e., $F \in \mathcal{F}_S$.
 - ▷ if one node is of type P_0 and the other is of type P_2 .
- Type 2 faces: faces $F \notin \mathcal{F}_G$.
 - ▷ if one node is a node of type P_2 and the other is of type P_1 or P_2 .

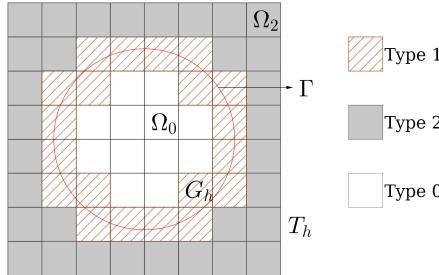


Figure 2.5. Diagram of a domain T_h cut by an interface Γ given by the level-set function $\phi = x^2 + y^2 - 1$ and the resulting classification of cells and domain characterization.

Intersection Detection

The next step of the boundary characterization is to find the points of intersection of Γ with the faces of cells $K \in G_h$. An intersected face

is characterized by having the level-set value on two consecutive nodes with different signs (face of Type 1). One can compute the coordinates of the intersection points via a linear interpolation of the two values of the level-set on these nodes. In boundary problems where the domain of interest is only Ω_0 , a "new" cut-cell structure is created for elements $K \in G_h$. For elements $K \in \Omega_0$, cell information is retrieved from the native deal.II framework for cell data. In Figure (2.6) an example of a cell intersected by the boundary is shown. Note that in the original cell there are 4 faces: $\overline{AB}, \overline{AD}, \overline{DC}$ and \overline{CB} . In the second case, the cut-cell has 3 new faces: $\overline{P_{I_0}B}, \overline{P_{I_0}P_{I_1}}$ and $\overline{P_{I_1}D}$ plus 2 original faces: \overline{DC} and \overline{CB} . The new cut face representing the local boundary of the cell, $\overline{P_{I_0}P_{I_1}}$, is marked as a *boundary face*, whereas other faces are *internal faces*. The described algorithm and structure of cell data does not lose generality for the case where the new cut cell has 4 or 3 faces.

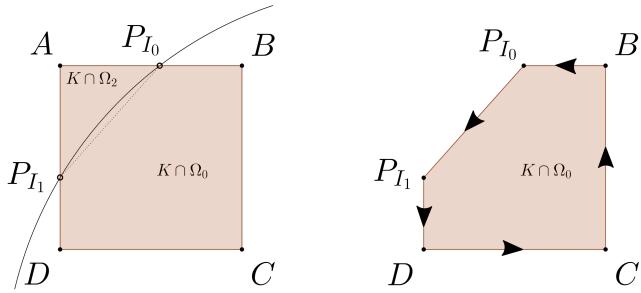


Figure 2.6. Example of a cell intersected by the boundary and the resulting cut-cell, with nodes ordered counterclockwise.

In Algorithm (2.2) it is shown a simple sequence of steps used to detect intersection points and compute new relevant information about the cut cell. The process starts by looping over all cells from set G_h , then identifying which faces are cut by the interface and which are completely inside Ω_0 or Ω_1 . New faces are created as described earlier and nodes are reordered in an anti-clockwise manner, to facilitate further parametric representation and numerical integration.

This information is stored into a class type, in order to facilitate the retrieval of data in future computations. This class possesses the following relevant information about the cut-cell:

- Cell index
- Centroid coordinates
- Number of faces
- Face information
 - ▷ Nodes' coordinates
 - ▷ Normal vector
 - ▷ Face length

- ▷ Face index
- ▷ Face identification (boundary face/internal face).

Some properties of the new cut cell may be inherited from the native non-cut-cell from which it derives. For instance, normal vectors of faces that are not intersected by the boundary remain the same from the original cell. In these cases, the information is merely copied from the native cell container provided by deal.II. When new data need to be computed, functions inside the class calculate and set these new parameters. This approach is also advantageous because it provides an easy and efficient way to retrieve cells and faces through indices and to verify the type of faces, facilitating the integration of terms over the boundary.

Algorithm 2.2 Intersection Detection

```

for each  $K \in G_h$ 
  for each  $F \in K$ 
    if  $\phi(X_0) * \phi(X_1) \leq 0$ 
      Compute intersection point  $P_{I_i}$ 
       $i \leftarrow i + 1$ 
    if  $\phi(X_0) < 0$ 
      Create new face  $\overline{X_0 P_{I_i}}$ 
    else if  $\phi(X_1) < 0$ 
      Create new face  $\overline{X_1 P_{I_i}}$ 
    Reorder nodes in anti-clockwise order
    Compute normal vector
  Create new face  $\overline{P_{I_0} P_{I_1}}$ 
  Reorder nodes in anti-clockwise order
  Compute normal vector
  Store data about cut-cell

```

2.6.3 Integral Evaluation on Cut-Cells

The next step is to evaluate integrals defined on the domain Ω_0 and in the boundary Γ . These integrals can be, for example, of the type

$$\int_K \nabla \phi_i \cdot \nabla \phi_j \, dx \, dy \text{ and } \int_K f \phi_j \, dx \, dy \quad (2.39)$$

from (2.16) on the area of cut-cells $K \cap \Omega$ and

$$\int_{\Gamma_N} g_N \phi_j \, d\Gamma \quad (2.40)$$

(or similar) from eq. (2.17) on the boundary Γ , as explained in section (2.5.1).

The numerical integration procedure provided by deal.II evaluates integrals over standard quadrilateral elements and faces only. By cutting arbitrarily an element, one can have as a result elements with 5, 4 or 3 faces, as shown in section (2.6). The integrals must be evaluated only on the part of the cut-cell inside the domain Ω . One possible alternative would be to split this element and create new quadrilaterals conforming to the boundary, a technique frequently used in the extended FEM [6, 50]. These new quadrilaterals would be part of a new "sub-triangulation" and one could take advantage of the native integration over quadrilaterals provided by deal.II. However, this process would require the generation of a new triangulation structure which may need additional vertices, becoming expensive and defeating the purpose of avoiding grid reinitialization. An alternative approach is to use Schwarz-Christoffel conformal mappings, as described by Natarajan, et al. [37, 38] in the X-FEM framework. This method is efficient for 2D problems but appears to be difficult to extend to three dimensional space.

The approach used in this work is based on the reduction of face integrals to boundary integrals through Green's Divergence theorem. This method was described by Mirtich [35] and was efficiently applied to calculate solid mechanics properties such as moments and product of inertia in polyhedral bodies. The method was extended to the finite element framework by Massing, et al., [34] and successfully used to evaluate integrals such as (2.39) in complex polygons and polyhedra.

First, the concept of multi-index and its notation are introduced, which are used throughout the text.

A multi-index $\alpha = (\alpha_0, \alpha_1, \dots, \alpha_{dim-1}) \in \mathbb{N}_0^{dim-1}$ is defined as a dim -tuple of non-negative integers α_i . Its order $|\alpha|$ is

$$|\alpha| = \sum_{i=0}^{dim-1} \alpha_i. \quad (2.41)$$

Based on this, the classical partial derivative can be written as

$$D^\alpha u = \prod_0^{dim-1} \left(\frac{\partial}{\partial x_i} \right)^{\alpha_i} u. \quad (2.42)$$

The method for evaluating integrals over cut-cells is outlined by the following steps:

1. *Rewrite integrands on a monomial basis.*

The terms to be integrated on cut-cells such as (2.39) can be interpolated onto a monomial basis. Let $\mathbf{x} = [x_0, \dots, x_{dim-1}] \in \mathbb{R}^2$ be the natural space for Cartesian coordinates. A polynomial $f(x_0, x_1, \dots)$ can be represented as

$$f(\mathbf{x}) = c_0(\mathbf{x})^{\alpha_0} + \cdots + c_n(\mathbf{x})^{\alpha_n} = \sum_{i=0}^n c_i(\mathbf{x})^{\alpha_i}, \quad (2.43)$$

where α is a multi-index *dim*-tuple and

$$\mathbf{x}^\alpha = x_0^{\alpha_0} \cdot x_1^{\alpha_1} \cdots x_{dim-1}^{\alpha_{dim-1}} = \prod_{j=0}^{dim-1} x_j^{\alpha_j}. \quad (2.44)$$

2. Reduce integrals over surfaces ($dim = 2$) to line integrals.

The two-dimensional Divergence Theorem of Gauss reads [47]: Let R be a domain in the xy -plane with boundary curve C . If \vec{F} is a smooth vector field,

$$\iint_R \nabla \cdot \vec{F} dA = \oint_C \vec{F} \cdot \hat{\mathbf{n}} dS \quad (2.45)$$

To employ this theorem, one must rewrite the polynomial integrand $f(x, y)$ as the divergence of the vector field $\nabla \cdot \vec{F}$. This can be done by taking the anti-derivative of the monomials in the following fashion:

$$\mathbf{x}^\alpha = \nabla \cdot \sum_{j=0}^{dim-1} \frac{\mathbf{x}^{\alpha+\mathbf{e}_j}}{dim(\alpha_j + 1)}, \quad (2.46)$$

where \mathbf{e}_j is the j -th unit vector.

It is easy to see that for the two-dimensional case the relation becomes:

$$(x, y)^\alpha = \nabla \cdot \left(\frac{x^{\alpha_0+1} \cdot y^{\alpha_1}}{2(\alpha_0 + 1)}, \frac{x^{\alpha_0} \cdot y^{\alpha_1+1}}{2(\alpha_1 + 1)} \right). \quad (2.47)$$

Now the polynomial $f(x, y)$ can be written as

$$f(\mathbf{x}) = \nabla \cdot \vec{F} \quad (2.48)$$

where

$$\vec{F} = \sum_{i=0}^n c_i \sum_{j=0}^{dim-1} \frac{\mathbf{x}^{\alpha_i+\mathbf{e}_j}}{dim(\alpha_{ij} + 1)}. \quad (2.49)$$

Equation (2.45) applied to an integral such as (2.39) over a 2D cell becomes:

$$\int_K f(x, y) dx dy = \sum_{i=0}^n c_i \sum_{j=0}^{dim-1} \oint_C \frac{\mathbf{x}^{\alpha_i+\mathbf{e}_j}}{dim(\alpha_{ij} + 1)} \cdot \hat{\mathbf{n}} dS \quad (2.50)$$

The closed curve integral can be approximated by the sum of line integrals along the faces of the element, that together form a closed convex polygon:

$$\begin{aligned} & \int_K f(x, y) dx dy \\ &= \sum_{i=0}^n c_i \sum_{j=0}^{dim-1} \sum_{k=0}^{n_{faces}-1} \int_{F_k \in K} \frac{\mathbf{x}^{\alpha_i + \mathbf{e}_j}}{dim(\alpha_{ij} + 1)} \cdot \hat{\mathbf{n}}_{F_k} dF, \end{aligned} \quad (2.51)$$

where F_k refers to the k th- face of the cut-cell K and n_{faces} to its number of faces.

3. Map face and evaluate line integrals.

The next step is to map and evaluate line integrals appearing in (2.51). To simplify the equations, the face integral will be represented as:

$$\int_{F \in K} \frac{(\mathbf{x})^{\alpha_j + \mathbf{e}_i}}{dim(\alpha_{j_i} + 1)} \cdot \hat{\mathbf{n}}_F dF = \int_F \vec{G}(x, y) \cdot \hat{\mathbf{n}} dF. \quad (2.52)$$

To evaluate the integral of eq. (2.52) it is advantageous to parameterize the face F onto a 1D line. The coordinates of a point on the face F are represented as functions of independent parameters through geometrical mappings. The parameters usually lie in the unit interval.

F can be parameterized by the vector function $\vec{r}(t)$, with $t \in [0, 1]$, such that

$$\vec{r}(t) = (x(t), y(t)), \quad (2.53)$$

where $\vec{r}(t)$ is an affine transformation from Cartesian natural coordinates $x, y \in \mathbb{R}^2$ to parameter space $t \in \mathbb{R}$. The differential becomes:

$$dF = |\vec{r}'(t)| dt,$$

where $|\vec{r}'(t)|$ is equal to length of the face, L_e .

The integral of (2.52) can finally be written as

$$\int_F \vec{G}(x, y) \cdot \hat{\mathbf{n}} dF = \int_0^1 \vec{G}(x(t), y(t)) \cdot \hat{\mathbf{n}} L_e dt, \quad (2.54)$$

which can be readily integrated by the Gauss quadrature method as described in (2.5.1). As an alternative, Mirtich (1996) evaluated these integrals with the use of Bernstein's polynomials, which do not require the evaluation of quadrature points beforehand.

3. Test Cases

The present chapter introduces numerical examples solved using the cut-cell finite element method. First, we solve the classical Poisson equation as described in section (2.1). Next, the Laplace-Beltrami problem is proposed, and finally, a reaction-diffusion case is introduced.

3.1 Poisson Problem

3.1.1 Problem setup

The model described by Poisson's equation with pure Dirichlet boundary conditions imposed weakly by Nitsche's approach was solved using the cut-cell method in a two-dimensional disk with radius $r = 1$. The boundary is represented by the level set function

$$\phi(x, y) = \sqrt{(x - x_0)^2 + (y - y_0)^2} - r.$$

Recalling (2.1) and setting the forcing function, $f = 1$, and Dirichlet value, $g_D = 0$, the equation to be solved is:

$$-\Delta u = 1 \text{ in } \Omega$$

$$u = 0 \text{ on } \Gamma = \partial\Omega,$$

which has an analytic solution:

$$u(\mathbf{x}) = \frac{1 - |\mathbf{x}|^2}{4}.$$

The values of γ_D and γ_1 from equations (2.12) and (2.15) were set as 5.0 and 0.1, respectively. The domain Ω was embedded in a rectangular domain $[-1.5, 1.5] \times [-1.5, 1.5]$ partitioned into squares of equal size. For details about the finite element formulation, see section (2.3).

3.1.2 Convergence Analysis

Condition number of the stiffness matrix

The described finite element formulation yields a condition number of the stiffness matrix with similar upper bound as the one resulting from

the standard finite element method. More details about the analysis can be found in [17].

Let U be any vector such that $U \in \mathbb{R}^N$ where $N = \dim V_h$, and denote its corresponding finite element function in V_h by u_h . The standard Euclidean norm is defined by $|U|_N$. Denote the mass matrix by \mathcal{M} , which is defined by the bilinear form (u_h, v_h) , and the system matrix by A , which is defined by the bilinear form $A_h(u_h, v_h)$. The triangulation T_h is conforming to the domain Ω_T . Therefore the following estimate hold:

$$\mu_{\min}^{1/2} \|U\|_N \leq \|u_h\|_{0, \Omega_T} \leq \mu_{\max}^{1/2} \|U\|_N,$$

where μ_{\min} and μ_{\max} refer to the smallest and largest eigenvalues of the mass matrix \mathcal{M} . The condition number of the stiffness matrix A is represented by $\kappa(A)$ and by definition is given by:

$$\kappa(A) = \|A\| \|A^{-1}\|,$$

where

$$\|A\| = \sup_{U \in \mathbb{R}^N} \frac{\|AU\|_N}{\|U\|_N}.$$

The condition number of the system matrix arising from the finite element formulation using Nitsche's method (2.13) satisfies the upper bound:

$$\kappa(A) \leq C h^{-2} \tag{3.1}$$

where C is a constant independent of how the boundary transverses the mesh. The proof and definition of C are given in [9].

Optimal error bounds

The finite element solution u_h satisfies the following error estimates in H^1 and L_2 , respectively [9]:

$$\|u - u_h\|_{H^1(\Omega)} \leq C h \|u\|_{H^2(\Omega)}$$

$$\|u - u_h\|_{L_2(\Omega)} \leq C h^2 \|u\|_{H^2(\Omega)}.$$

3.2 Laplace-Beltrami Problem

3.2.1 The Laplace-Beltrami Model Problem

The pure diffusion equation on a closed surface Γ of co-dimension one is proposed as a model problem:

$$-\Delta_\Gamma u = f \text{ on } \Gamma, \quad (3.2)$$

where Δ_Γ denotes the Laplace-Beltrami operator, which will be defined in the following sections. Since any constant function is a solution to this problem, one has to impose an additional constraint to obtain a unique solution. In this problem the mean value of u is chosen to vanish on the boundary, so that:

$$\int_{\Gamma} u \, ds = 0. \quad (3.3)$$

The first step to obtain a computable form of (3.2) is to define a suitable function space. The space of continuous Lagrange bilinear polynomials is used. The space V_S is then introduced, with the enforcement of constraint using a Lagrange multiplier method:

$$V_S = \left\{ v \in H^1(\Gamma) \mid \int_{\Gamma} v \, ds = 0 \right\}. \quad (3.4)$$

To obtain the variational form, the same procedure outlined in section (2.1.1) is employed, as well as the notation regarding mesh characteristics. First, multiply the strong form (3.2) by a test function $v \in V_S$ and integrate using Green's formula. The weak formulation then reads: find $u \in V_S$ such that

$$a(u, v) = f(v), \quad \forall v \in V_S \quad (3.5)$$

where $a(u, v)$ denotes the symmetric bilinear form on Γ and $f(v)$, the linear functional form:

$$a(u, v) = (\nabla_\Gamma u, \nabla_\Gamma v)_\Gamma \quad (3.6)$$

$$f(v) = (f, v)_\Gamma \quad (3.7)$$

3.2.2 Approximation of the Domain

Let $p(\mathbf{x})$ be the closest point mapping on Γ to a point $\mathbf{x} \in \mathbb{R}^{dim}$ and $d(\mathbf{x})$ a signed distance function given by

$$d(\mathbf{x}) = |p(\mathbf{x}) - \mathbf{x}| \text{ in } \mathbb{R}^{dim} \setminus \Omega \quad (3.8)$$

$$d(\mathbf{x}) = -|p(\mathbf{x}) - \mathbf{x}| \text{ in } \Omega. \quad (3.9)$$

The open tubular neighborhood of the boundary Γ is defined as:

$$U(\Gamma) = \left\{ \mathbf{x} \in \mathbb{R}^{dim} : |d(\mathbf{x})| < \delta \right\}, \quad (3.10)$$

where δ is small enough so that for each $\mathbf{x} \in U(\Gamma)$ there is a uniquely determined $p(\mathbf{x}) \in \Gamma$. See fig. (3.1) for a representation of the domains of interest. Let $\phi : \Gamma \rightarrow \mathbb{R}$ be any function defined on the hypersurface Γ . The smooth extension of ϕ to the neighborhood U is defined as:

$$\tilde{\phi}(\mathbf{x}) = \phi \circ p(\mathbf{x}). \quad (3.11)$$

Note that $\tilde{\phi}|_{\Gamma} = \phi$.

Let Ω_1 be a domain in \mathbb{R}^{dim} containing $\Omega \cup U(\Gamma)$ and T_h an uniform triangulation of Ω in regular quadrilaterals K with mesh diameter h . A continuous piecewise linear approximation Γ_h of Γ is considered such that $\Gamma_h \subset U(\Gamma)$ and $\Gamma_h \cap K$ is a subset of a hypersurface in \mathbb{R}^{dim} .

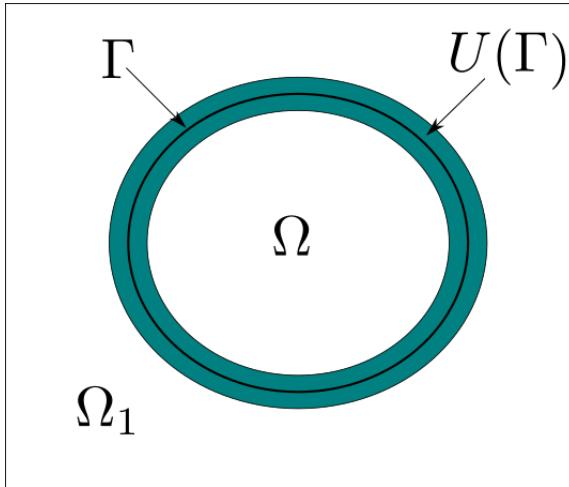


Figure 3.1. Representation of the described domains. $U(\Gamma)$ is the green region where a unique closest point $p(\mathbf{x}) \in \Gamma$ is defined for each $\mathbf{x} \in U(\Gamma)$.

3.2.3 The Laplace-Beltrami Operator

The Laplace-Beltrami operator is a generalization of the Laplace operator that operates on functions defined on surfaces in Euclidean space. Similarly to the Laplacian, the Laplace-Beltrami operator can be defined as the divergence of a gradient. In this section only a brief review of the operator is outlined, for more details refer to any book on differential geometry [45, 30].

For each $\mathbf{x} \in U(\Gamma)$, the projection of \mathbb{R}^{dim} onto the tangent plane of Γ is defined by

$$P_\Gamma = I - \mathbf{n} \otimes \mathbf{n}, \quad (3.12)$$

where I is the identity tensor and \mathbf{n} , the normal vector. The tangential gradient of a function $\phi \in C^1(U(\Gamma))$ at $\mathbf{x} \in \Gamma$ is given by:

$$\nabla_\Gamma \phi(\mathbf{x}) = P_\Gamma \nabla \phi(\mathbf{x}), \quad (3.13)$$

where ∇ is the usual gradient operator in \mathbb{R}^{dim} . Finally, the Laplace-Beltrami operator for a $C^2(U(\Gamma))$ function ϕ is defined as:

$$\Delta_\Gamma \phi(\mathbf{x}) = \nabla_\Gamma \cdot \nabla_\Gamma \phi(\mathbf{x}). \quad (3.14)$$

3.2.4 Finite Element Formulation

In order to discretize the formulation, the set of elements K associated with the boundary Γ is defined as follows, in the same fashion as in section (2.2):

$$G_h = \{K \in T_h : K \cap \Gamma_h \neq \emptyset\}, \quad \Omega_h = \bigcup_{K \in G_h} K. \quad (3.15)$$

Let $V^{h,1}$ be the space of continuous piecewise linear functions defined on the triangulation T_h . The function space V^h is defined as a continuous piecewise linear function on G_h with average zero along the boundary:

$$V^h = \left\{ v_h \in H^1 : V^{h,1}|_{\Omega_h} : \int_\Gamma v_h \, d\Gamma = 0 \right\}. \quad (3.16)$$

The standard Galerkin discretization of (3.5) reads: find $u_h \in V^h$ such that

$$A_h(u_h, v_h) = L(v_h), \quad \forall v_h \in V^h, \quad (3.17)$$

where the bilinear form is given by

$$A_h(u_h, v_h) = (u_h, v_h)_{\Gamma_h} + \gamma_S h j_h(u_h, v_h) \quad (3.18)$$

and

$$L(v_h) = (f, v_h)_{\Gamma_h}. \quad (3.19)$$

The stabilization term $j_h(u_h, v_h)$ is defined by

$$j_h(u_h, v_h) = \sum_{F \in \mathcal{F}_S} ([\nabla u_h], [\nabla v_h]_F)_\Gamma. \quad (3.20)$$

where $[\nabla u]$ denotes the jump of u over the face F . For definitions of jump term and set \mathcal{F}_S , see (2.2).

3.2.5 Problem Setup

The pure diffusion Laplace-Beltrami problem was solved using the cut-cell method in a circular domain with radius $r = 1$. The right hand side function f was chosen to be:

$$f(x, y) = 9 \sin(3 \arctan(x/y)) \quad (3.21)$$

as in [29] so that the analytical solution is

$$u(x, y) = \sin(3 \arctan(x/y)). \quad (3.22)$$

The boundary domain Γ is defined by a circle of radius $r = 1$, and is obtained by the level-set function

$$\phi(x, y) = \sqrt{(x - x_0)^2 + (y - y_0)^2} - r.$$

The domain is embedded in a rectangular triangulation

$$T_h = [-1.5, 1.5] \times [-1.5, 1.5]$$

comprised of squares of equal size. The triangulation is divided into sub-domains Ω_0 , G_h and Ω_2 as described in section (2.6.2). Cells from domain Ω_0 and Ω_2 were excluded from the computation. The stabilization parameter γ_S was set to 0.01.

3.2.6 Convergence Analysis

Condition Number of the Stiffness Matrix

Similarly to the Poisson model problem, the condition number of the stiffness matrix $\kappa(A)$ is proportional to h^{-2} and the following estimate holds independently of the boundary intersection:

$$\kappa(A) \leq C h^{-2}. \quad (3.23)$$

Optimal error bounds

The error estimates in H^1 and L_2 norms are, respectively:

$$\|u - u_h\|_{H^1(\Gamma)} \leq C h \|u\|_{H^1(\Gamma)} \quad (3.24)$$

$$\|u - u_h\|_{L_2(\Gamma)} \leq C h^2 \|u\|_{L_2(\Gamma)}. \quad (3.25)$$

For an outline of the proofs, see [11, 42].

3.3 Reaction Diffusion Problem

3.3.1 The Reaction Diffusion Problem

The proposed system is composed of a circular domain where a species A diffuses on the bulk domain and reacts in a patch of the membrane, identified by $\Gamma_1 \subset \Gamma$, yielding a component B . The reaction mechanism is given by a simple first-order reaction with reaction rate constant k :



The component B is membrane-bound and it is allowed to diffuse freely on it. The component A can be present both in bulk and membrane domains. The rates of reaction are expressed as:

$$r_A = -r_B = -k u_A. \quad (3.27)$$

The reaction occurs only on part of the boundary, hence the reaction term is defined by a Kronecker delta function $\delta(\mathbf{x})$. We choose Neumann no-flux boundary conditions on all the boundary, meaning that it is an enclosed domain with no exchange with the exterior. The final coupled system reads:

$$\frac{\partial u_A}{\partial t} - \nabla \cdot D_A \nabla u_A + \delta(\mathbf{x}) k u_A = 0, \quad \text{in } \Omega \quad (3.28)$$

$$\mathbf{n} \cdot \nabla u_A = 0, \quad \text{on } \Gamma \quad (3.29)$$

$$\frac{\partial u_B}{\partial t} - \nabla_\Gamma \cdot D_B \nabla_\Gamma u_B - \delta(\mathbf{x}) k u_A = 0, \quad \text{on } \Gamma. \quad (3.30)$$

The weak form is obtained by the same procedure previously outlined. For the bulk equation: multiply (3.28) by a test function $v_A \in H^1(\Omega)$, integrate using Green's theorem:

$$\left(\frac{\partial u_A}{\partial t}, v_A \right) + a(u_A, v_A) + \langle \delta(\mathbf{x}) k u_A, v_A \rangle - L(v_A) = 0, \quad (3.31)$$

where $L(v_A) = \langle v_A, g_N \rangle$. For the surface equation, multiply (3.29) by a test function $v_B \in H^1(\Gamma)$, integrate:

$$\left\langle \frac{\partial u_B}{\partial t}, v_B \right\rangle + a(u_B, v_B) - \langle \delta(\mathbf{x}) k u_A, v_B \rangle = 0. \quad (3.32)$$

3.3.2 Finite Element Formulation

The space discretization of the PDE system is based on the finite element formulation from the previous test cases. We use the space of bilinear Lagrange elements defined in eq. (2.8):

$$V_A^h = \{v : v \in C^0(\Omega), v|_K \in Q_1(K), \forall K \in T_h\}$$

and in eq. (3.4) (except we do not enforce the constraint on the boundary):

$$V_B^h \subset V = H^1(\Gamma).$$

The discretization then becomes:

$$\left(\frac{\partial u_A^h}{\partial t}, v_A^h \right) + A_h(u_A^h, v_A^h) + (\delta(\mathbf{x}) k u_A^h, v_A^h) - L(v_A^h) = 0 \quad (3.33)$$

$$\left\langle \frac{\partial u_B^h}{\partial t}, v_B^h \right\rangle + A_h(u_B^h, v_B^h) - \langle \delta(\Gamma) k u_A^h, v_B^h \rangle = 0. \quad (3.34)$$

Time discretization was performed using the Crank-Nicolson method [14], which is stable for the PDE system. The solution u , where u can be u_A or u_B , is replaced by the following approximation:

$$u(\mathbf{x}, t) = (1 - \theta) u^{n-1} + \theta u^n \quad (3.35)$$

where n is the time step and θ is a parameter that tune the approximation. For example, if $\theta = 0$, the formulation reduces to the explicit Euler method. If $\theta = 1$, it reduces to the pure implicit backward Euler method. In the Crank-Nicolson method, we choose $\theta = 0.5$ and the time stepping is implicit and unconditionally stable. The derivative with time is approximated by:

$$\frac{\partial u(\mathbf{x}, t)}{\partial t} = \frac{u^n - u^{n-1}}{\Delta t} \quad (3.36)$$

To obtain the final linear system, the eqs. (3.35) - (3.36) are replaced into (3.33). After, the solution u^h is replaced by $u^h = \sum_j U_j \phi_j$ and the test function by $v^h = \phi_i$ as usual.

The stabilized Nitsche's method is used to enforce Neumann boundary condition in (3.33), using the formulation (2.13) for the bilinear term $A_h(v_A^h, u_A^h)$ and the formulation (2.12) for the term $L(v_A^h)$.

The final equation to be solved is:

$$\begin{aligned} & [M_\Omega + \theta \Delta t (A_\Omega + k M_\Gamma^R)] U_A^n \\ &= [M_\Omega - (1 - \theta) \Delta t (k M_\Gamma^R + A_\Omega)] U_A^{n-1}, \end{aligned} \quad (3.37)$$

where the stiffness matrix A_Ω is

$$\begin{aligned} (A_\Omega)_{ij} = & (\nabla \phi_i, \nabla \phi_j)_\Omega + \langle \gamma_N h \mathbf{n}_\Gamma \cdot \nabla \phi_i, \mathbf{n}_\Gamma \cdot \nabla \phi_j \rangle_\Gamma \\ & + \gamma_1 h j_h (\phi_i, \phi_j)_{\mathcal{F}_G}. \end{aligned} \quad (3.38)$$

The mass matrix M_Ω is also stabilized by adding stabilization terms:

$$(M_\Omega)_{ij} = (\phi_i, \phi_j)_\Omega + \gamma_M h^2 j_h (\phi_i, \phi_j)_{\mathcal{F}_G}. \quad (3.39)$$

The mass matrix M_Γ^R is the mass matrix arising from the reaction term, which occurs only on part of the domain, defined by the function $\delta(\mathbf{x})$. The matrix is also stabilized:

$$(M_\Gamma^R)_{ij} = (\delta(\mathbf{x}) \phi_i, \phi_j)_\Omega + \gamma_M h^2 j_h (\phi_i, \phi_j)_{\mathcal{F}_S}. \quad (3.40)$$

The linear system used to solve eq. (3.34) is obtained by taking the same steps. The final equation is:

$$\begin{aligned} & [M_\Gamma + \theta \Delta t A_\Gamma] U_B^n \\ &= [M_\Gamma - (1 - \theta) \Delta t A_\Gamma] U_B^{n-1} + k C_\Gamma \Delta t. \end{aligned} \quad (3.41)$$

Here, the stabilized form of the bilinear term (3.18) was used to define $A_h \langle u_B^h, v_B^h \rangle$, so that the stiffness matrix A_Γ is

$$(A_\Gamma)_{ij} = \langle \nabla_\Gamma \phi_i, \nabla_\Gamma \phi_j \rangle + \gamma_S h j_h (\phi_i, \phi_j)_{\mathcal{F}_S}. \quad (3.42)$$

The mass matrix M_Γ is also stabilized and becomes:

$$(M_\Gamma)_{ij} = \langle \phi_i, \phi_j \rangle_\Gamma + \gamma_M h^2 j_h (\phi_i, \phi_j)_{\mathcal{F}_S}. \quad (3.43)$$

The vector C_Γ^n is obtained from the integration of the reaction term and depends on the concentration of u_A^n . It is given by:

$$(C_\Gamma^n)_{ij} = \langle \delta(\Gamma) u_A, \phi_i \rangle_\Gamma, \quad (3.44)$$

where u_A is obtained by eq. (3.35) from the already computed u_A^n and u_A^{n-1} . The resulting linear systems to be solved at each time step are eqs. (3.37) and (3.41).

3.3.3 Problem Setup

The set of PDE's was solved using the cut-cell method in a circular domain of radius $r = 1$, so that the boundary Γ is represented by the level-set function

$$\phi(x, y) = \sqrt{(x - x_0)^2 + (y - y_0)^2} - r.$$

The boundary Γ was split into two subdomains: Γ_1 , represented by the third quadrant of the circle, given by

$$\Gamma_1 = \{(x, y) \in \Gamma : x < 0, y < 0\} \quad (3.45)$$

and Γ_2 , represented by the remaining quadrants:

$$\Gamma_2 = \{(x, y) \in \Gamma \setminus \Gamma_1\}. \quad (3.46)$$

The delta function $\delta(\mathbf{x})$ is then defined as:

$$\delta(\mathbf{x}) = \begin{cases} 1 & \text{if } \Gamma = \Gamma_1 \\ 0 & \text{if } \Gamma = \Gamma_2 \end{cases}$$

The reaction rate constant was set to $k = 100$. The diffusion coefficients are homogeneous over all the domain and were both set to $D_A = D_B = 0.1$. As initial condition, a concentration of $u_A = 400$ was set on a square $[-0.25, 0.25] \times [-0.25, 0.25]$ and equal to zero elsewhere. A diagram of the initial conditions and geometry is shown in Figure (3.2). Concentration of B was set to zero everywhere.

The solution was solved for a total time of $t_f = 3.0$ s, with a time step equal to $\Delta t = 0.004$ s. The parameters were selected as follows: $\gamma_1 = 0.1; \gamma_S = 0.01; \gamma_M = 0.1; \gamma_D = 5$ and $\gamma_N = 1$.

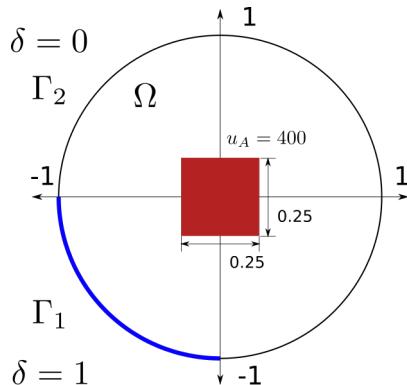


Figure 3.2. Diagram of the initial setup of the reaction-diffusion problem.

3.3.4 Mass Conservation

Since there are no generation terms nor flux across the boundary, the total variation of mass in the domain should approximate zero. In order to ensure that the mass is conserved throughout the time interval, we compute the total mass M_{t_i} in the domain at each time step t_i using the following equation:

$$M_{t_i} = \sum_{K_i \in \Omega_{K_i}}^{n_K} \int u_A^{t_i} d\Omega + \sum_{\Gamma_{K_i} \in G_h}^{n_{G_h}} \int u_B^{t_i} d\Gamma \quad (3.47)$$

and evaluate the variation of M_{t_i} relative to the initial mass present in the domain with

$$\Delta M_{t_i} = \left| \frac{M_{t_i} - M_{t_0}}{M_{t_0}} \right| \times 100\%. \quad (3.48)$$

4. Results

4.1 Poisson's Problem

4.1.1 Solution

The solution for the problem is shown in Figure (4.1) below. The boundary is represented by the thick black line. After removing the elements completely outside Ω , the mesh contains 13104 elements, each of diameter 0.0220971.

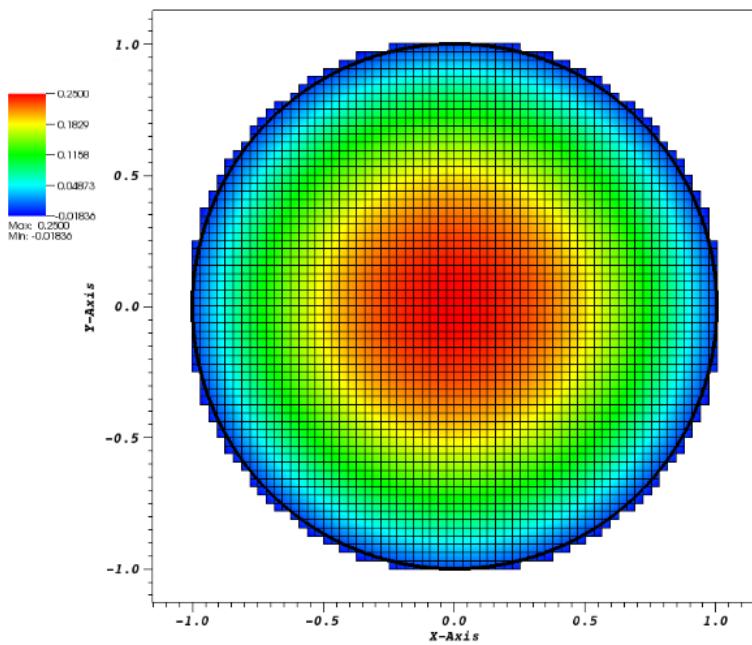


Figure 4.1. Solution for the Poisson problem with B.C. imposed weakly with Nitsche's method.

4.1.2 Convergence Analysis

The numerical solution was compared with the exact solution and the $L_2(\Omega)$ and $H^1(\Omega)$ norms were evaluated, as shown in Figure (4.2).

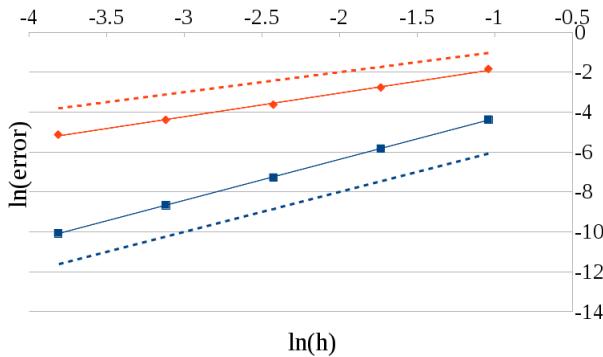


Figure 4.2. Convergence analysis in H^1 (orange) and L_2 (blue) norms. The dashed lines are proportional h (orange) and h^2 (blue).

4.1.3 Matrix conditioning analysis

This section shows the effects on the condition number of the stiffness matrix with respect to the mesh size and the stabilization parameter.

First the relation $\kappa(A) \leq C h^{-2}$, detailed in (3.1.2) was analyzed. In Fig. (4.3) the condition number of the stiffness matrix is plotted versus the inverse of the cell diameter.

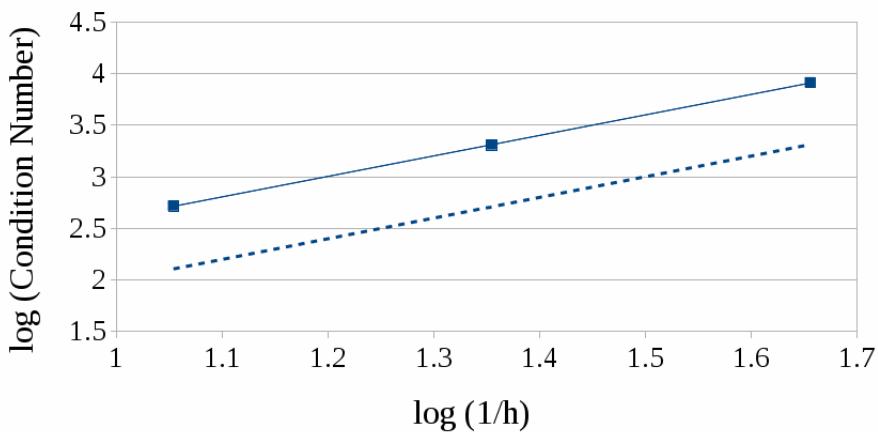


Figure 4.3. Condition number as a function of the inverse of the cell diameter. The dashed line is proportional to h^{-2} .

Next, the effect of stabilization parameter γ_1 is evaluated, for the most refined case as shown in section (4.1.1).

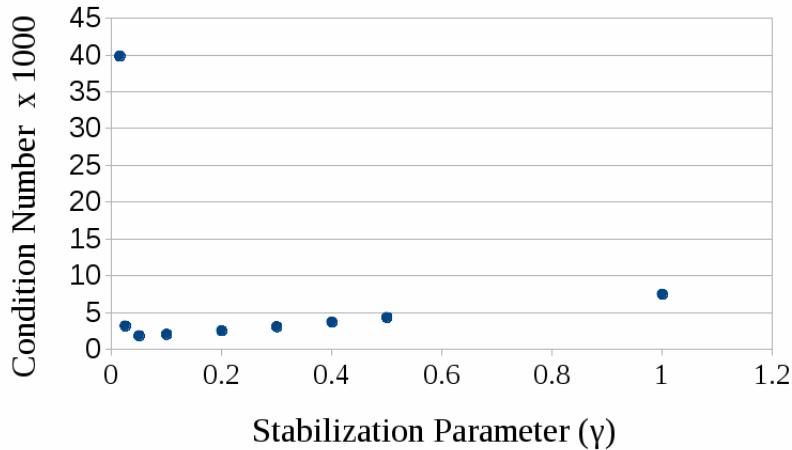


Figure 4.4. Condition number as a function of γ_1 .

In order to evaluate the dependence of the solution with the stabilization parameter, the L_2 norm was plot versus γ_1 and the results can be seen in Figure (4.5).

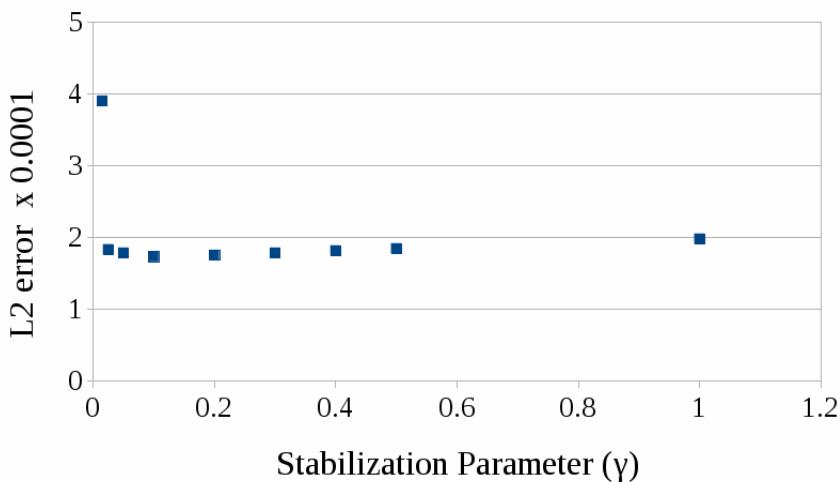


Figure 4.5. Plot of the L_2 error by the stabilization parameter.

4.2 Laplace-Beltrami Problem

4.2.1 Solution

The equation was solved on cells $K \in G_h$ and the solution is shown on Γ_h , as can be seen in (4.6). The cell diameter is $h = 0.022$ and the mesh contains 3332 cells, with 504 degrees of freedom.

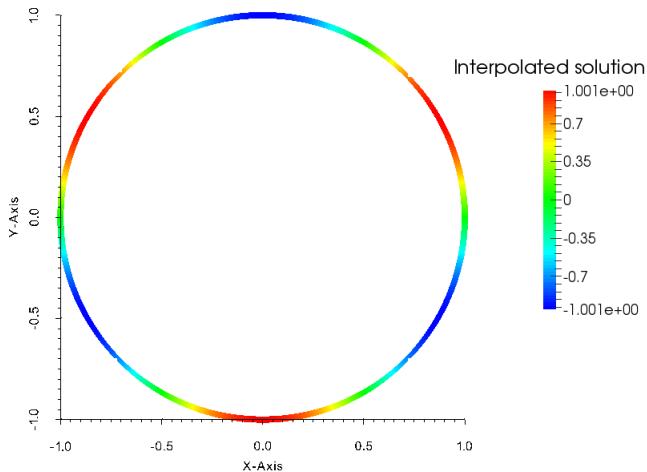


Figure 4.6. Solution on a boundary defined by the level set function.

4.2.2 Convergence Analysis

The H^1 and L_2 errors were evaluated based on the solution obtained on Γ_h for several mesh sizes. The results are shown in Figure (4.7).

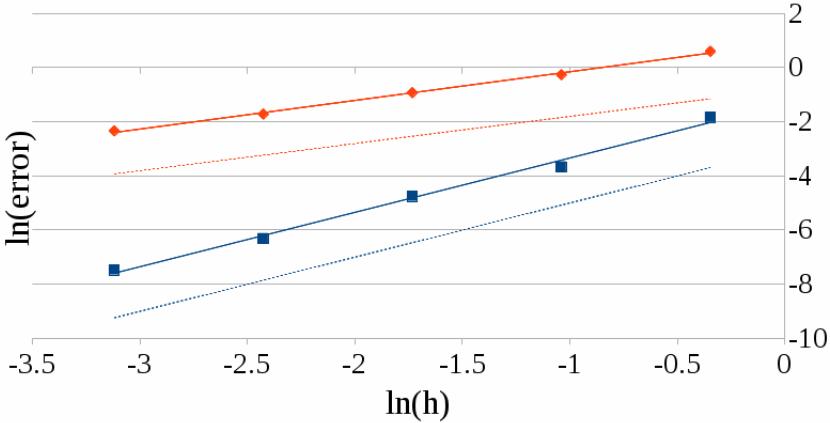


Figure 4.7. Convergence analysis in H^1 (orange) and L_2 (blue) norms. The dashed lines are proportional h (orange) and h^2 (blue).

4.2.3 Matrix conditioning analysis

The mesh dependence of the condition number was evaluated and the results can be seen in Figure (4.8).

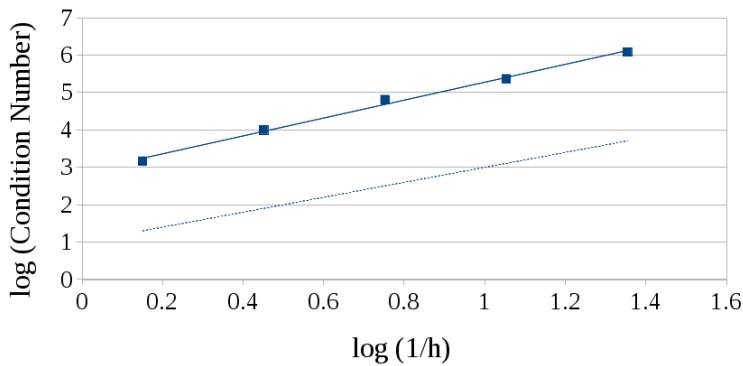


Figure 4.8. Dependence of condition number with the element size. The dashed line is proportional to h^{-2} .

4.3 Reaction-Diffusion Problem

4.3.1 Solution

The concentration profile of component A is shown in Figure (4.9), with the correspondent time associated. The mesh contains 13104 elements, each of size 0.0220971.

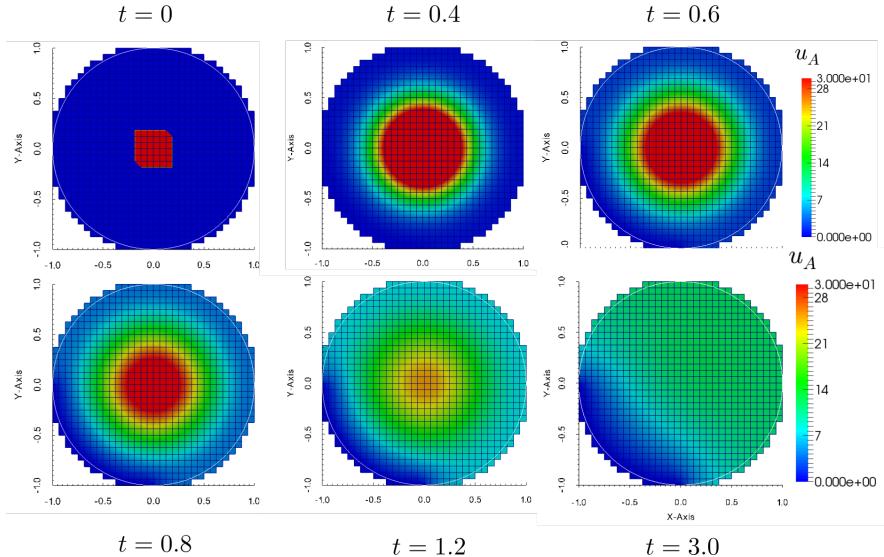


Figure 4.9. Time snaps of concentration of component A. The boundary is represented by the level set function in white.

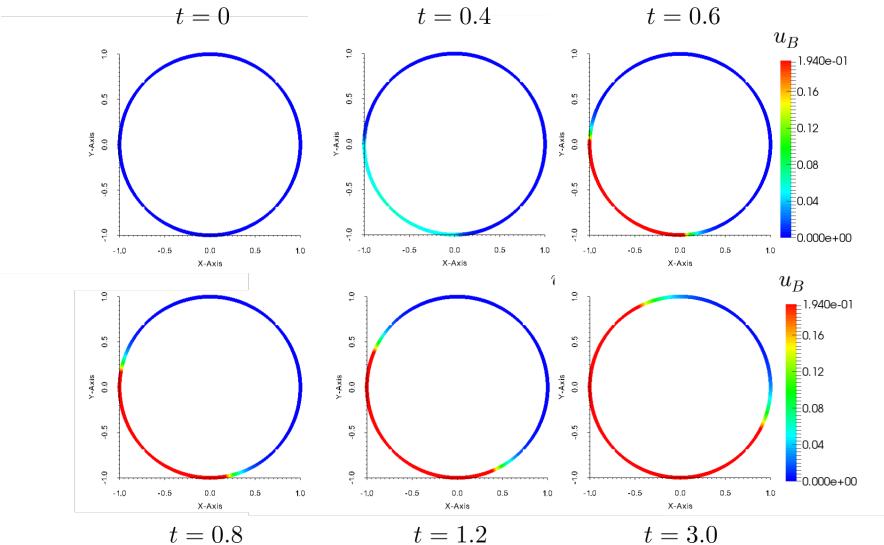


Figure 4.10. Concentration profile of component B on the boundary Γ_h .

4.3.2 Mass conservation

Mass conservation was calculated using (3.47) and the results of total mass variation are depicted in Figure (4.11).

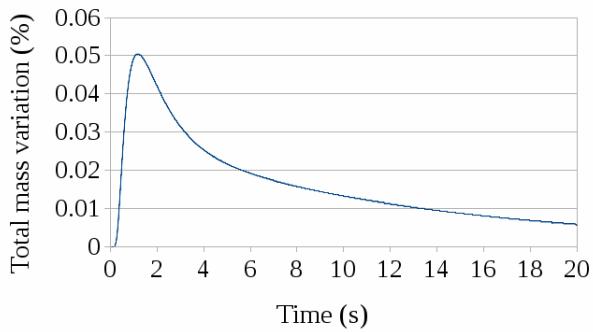


Figure 4.11. Mass conservation over the time integration interval.

5. Discussion

5.1 Poisson's Problem

The solution for the Poisson's problem is shown in section (4.1). It is important to note how the final solution contains nodes that are in fact outside the domain of the problem. These nodes are of "type 2" as represented by node A on section (2.3). These nodes were kept in the solution representation to illustrate how they play a role in the solution process, but do not represent the final solution of the problem. Moreover, the solution clearly follows the well known expected pattern of the homogeneous Poisson's equation on a disk.

The convergence analysis shows optimal convergence of second order in L_2 and first order in H^1 as expected from the finite element formulation.

The integration over small parts of elements adversely affects the condition of the stiffness matrix. This is overcome by adding stabilization terms (2.15) in the finite element formulation, which can be tuned by the parameter γ_1 . In Fig. (4.4), the plot of the condition number as a function of the stabilization parameter γ_1 is shown. It was observed that very small values of γ_1 lead to bad conditioning ($\gamma_1 = 0.015$), whereas the simulation with $\gamma_1 = 0.05$ resulted in the smallest condition number observed. After this value, the condition number increases steadily with the increase of γ_1 . Burman and Hansbo [9] report a very resembling pattern for a similar problem, where they found the condition number to be optimal for $\gamma_1 = 0.01$.

In addition, the L_2 error was evaluated for different values of the stabilization parameter. The error has a peak for $\gamma_1 = 0.015$ and the lowest value when $\gamma_1 = 0.1$, and grows linearly afterwards from $\gamma_1 = 0.1$ to 1. This relationship shows a close resemblance with the dependence of the condition number on γ_1 (Fig. (4.4)). Based on these results, it is suggested that in order to achieve a low condition number and error, one should set the stabilization parameter to $\gamma_1 = 0.05$ or 0.1.

The dependence of the condition number on the mesh size was evaluated, and confirmed to be of second order as estimated.

5.2 Laplace-Beltrami Problem

The solution for the Laplace Beltrami Problem shown in Figure (4.1) follows closely the same pattern of the exact solution. Similar to the

previous problem, the stiffness matrix becomes severely ill conditioned if not stabilized. The stabilization solves this problem, without increasing the relative error. Kamilis [29] reports similar results, with drastic decrease of the condition number, however he obtains higher L_2 error for the stabilized case. The conditioning analysis reports the condition number to be proportional to the square of the inverse of the cell size and the theoretical estimate (3.23) was confirmed.

5.3 Reaction-Diffusion Problem

Molecules of A start with a high concentration in the middle as set by the initial condition, and diffuse uniformly towards the boundary up to time $t = 0.6$, where it starts to interact with the boundary on Γ_1 and react, generating B . When component A reaches the non-reactive part of the boundary, Γ_2 , it accumulates and the gradient of concentration smooths. By $t = 3$, the concentration is nearly homogeneous in the region far from the reactive boundary.

The variation of concentration of B with time is shown in Figure (4.10). For the plot of these results, the solution was defined on the boundary given by the level-set function. As the constant of reaction is very fast for this case, the concentration of B quickly raises as A reaches the boundary, around $t = 0.3$. The diffusion process follows and the molecules migrate along the boundary.

The total mass inside the domain was analyzed and compared to the initial mass as reference. The model does not include a generation term and the boundary is defined as no-flux through the imposition of Neumann boundary condition. Results in Fig. (4.11) show that the variation of mass grows up to 0.05% around $t = 1.1 \sim 1.3$ then declines steadily with time. This is an acceptable range of variation for the total mass inside the domain, considering that there are sharp gradients in the beginning of the simulation and a fast reaction occurring on the boundary.

6. Conclusions

6.1 Conclusive Remarks

The aim of this thesis has been to evaluate the application of a cut-cell technique using the finite element method. The motivation for developing methods where the boundary or interface does not conform to the mesh was outlined, as well as a review of current methods using this approach. We have reviewed the FEM using Nitsche's method to impose boundary conditions weakly and how to implement it computationally. In this context, the cut-cell method was introduced using the Poisson equation as a model problem. Steps that are important in the cut-cell method were highlighted, such as domain and mesh characterization and stabilization techniques. The implicit representation of the surface and the use of level set functions to define it were outlined, in order to explain the intersection detection mechanism and the domain compartmentalization necessary to implement the method.

Three test cases were proposed to evaluate the cut-cell method: the Poisson problem, the pure diffusion Laplace-Beltrami problem and a reaction diffusion problem. The solutions of the first and second test cases were evaluated and conformed to the exact solutions. Convergence analysis in H^1 and L_2 norms were performed to evaluate the dependence of the error on the mesh size and the theoretical estimates were confirmed. The condition number of the stiffness matrices of both test cases was analyzed in order to evaluate the possible ill-conditioning effect arising from the unfitted method.

The reaction diffusion problem was proposed as a time-dependent, coupled bulk-surface test case. The mass variation over the domain was evaluated in order to verify the method and was shown to be conserved under a low bound.

The cut-cell method proved to be a suitable alternative to the standard FEM to solve elliptic and parabolic partial differential equations. The implementation of the method was efficient and flexible under deal.II framework, which showed to be a suitable computational platform to apply the method.

6.2 Suggestions and Future Work

The successful implementation of the cut-cell method for the proposed problems paves the way for the modeling of several interesting problems

where the boundary and its relationship with the bulk domain play an important role. Suggestions for improvement include:

- Evaluate refinement of elements along the boundary to increase accuracy in computationally heavy transient problems;
- Analyze the effect of adaptive methods, eg., refine cut-cells where the elements become too small;
- Study cases where the boundary changes over time.

The work is intended to continue by modeling more realistic reaction diffusion problems in a three dimensional geometry.

7. References

- [1] Douglas N. Arnold. An interior penalty finite element method with discontinuous elements. *SIAM Journal on Numerical Analysis*, 19(4):742–760, 1982.
- [2] Douglas N. Arnold, Franco Brezzi, Bernardo Cockburn, and L. Donatella Marini. Unified Analysis of Discontinuous Galerkin Methods for Elliptic Problems. *SIAM J. Numer. Anal.*, 39(5):1749–1779, May 2001.
- [3] W. Bangerth, T. Heister, L. Heltai, G. Kanschat, M. Kronbichler, M. Maier, B. Turcksin, and T. D. Young. The `deal.II` library, version 8.2. *Archive of Numerical Software*, 3, 2015.
- [4] W. Bangerth and G. Kanschat. Concepts for object-oriented finite element software – the `deal.II` library. Preprint 99-43 (SFB 359), IWR Heidelberg, October 1999.
- [5] Roland Becker, Peter Hansbo, and Rolf Stenberg. A finite element method for domain decomposition with non-matching grids. *ESAIM: Mathematical Modelling and Numerical Analysis*, 37:209–225, 3 2003.
- [6] T. Belytschko, N. Moës, S. Usui, and C. Parimi. Arbitrary discontinuities in finite elements. *International Journal for Numerical Methods in Engineering*, 50(4):993–1013, 2001.
- [7] S.C. Brenner and R. Scott. *The Mathematical Theory of Finite Element Methods*. Texts in Applied Mathematics. Springer, 2008.
- [8] Erik Burman, Susanne Claus, Peter Hansbo, Mats G. Larson, and André Massing. CutFEM: Discretizing geometry and partial differential equations. *International Journal for Numerical Methods in Engineering*, pages n/a–n/a, 2014.
- [9] Erik Burman and Peter Hansbo. Fictitious domain finite element methods using cut elements: II. A stabilized Nitsche method. *Applied Numerical Mathematics*, 62(4):328 – 341, 2012. Third Chilean Workshop on Numerical Analysis of Partial Differential Equations (WONAPDE 2010).
- [10] Erik Burman, Peter Hansbo, and Mats G. Larson. A stabilized cut finite element method for partial differential equations on surfaces: The Laplace-Beltrami operator. *Computer Methods in Applied Mechanics and Engineering*, 285(0):188 – 207, 2015.
- [11] Erik Burman, Peter Hansbo, Mats G Larson, and Sara Zahedi. Cut Finite Element Methods for Coupled Bulk-Surface Problems. *arXiv preprint arXiv:1403.6580*, 2014.
- [12] J. Chessa and T. Belytschko. An extended finite element method for two-phase fluids. *Journal of Applied Mechanics*, 70(1):10–17, January 2003.

- [13] E. T. Coon, B. E. Shaw, and M. Spiegelman. A Nitsche- extended finite element method for earthquake rupture on complex fault systems. *Computer Methods in Applied Mechanics and Engineering*, May 2011.
- [14] J. Crank and P. Nicolson. A practical method for numerical evaluation of solutions of partial differential equations of the heat-conduction type. *Advances in Computational Mathematics*, 6(1):207–226, 1996.
- [15] Gerhard Dziuk. Finite Elements for the Beltrami operator on arbitrary surfaces. In Stefan Hildebrandt and Rolf Leis, editors, *Partial Differential Equations and Calculus of Variations*, volume 1357 of *Lecture Notes in Mathematics*, pages 142–155. Springer Berlin Heidelberg, 1988.
- [16] Gerhard Dziuk and Charles M. Elliott. Finite element methods for surface PDEs. *Acta Numerica*, 22:289–396, 5 2013.
- [17] Alexandre Ern and Jean-Luc Guermond. Evaluation of the condition number in linear systems arising in finite element approximations. *ESAIM: Mathematical Modelling and Numerical Analysis*, 40:29–48, 1 2006.
- [18] Sonia Fernández-Méndez and Antonio Huerta. Imposing essential boundary conditions in mesh free methods. *Computer Methods in Applied Mechanics and Engineering*, 193(12 - 14):1257 – 1275, 2004. Meshfree Methods: Recent Advances and New Applications.
- [19] Thomas-Peter Fries and Ted Belytschko. The intrinsic XFEM: a method for arbitrary discontinuities without additional unknowns. *International Journal for Numerical Methods in Engineering*, 68(13):1358–1385, 2006.
- [20] A. Fritz, S. Hüeber, and B.I. Wohlmuth. A comparison of mortar and Nitsche techniques for linear elasticity. *CALCOLO*, 41(3):115–137, 2004.
- [21] Anita Hansbo and Peter Hansbo. An unfitted finite element method, based on Nitsche’s method, for elliptic interface problems. *Computer Methods in Applied Mechanics and Engineering*, 191(47 - 48):5537 – 5552, 2002.
- [22] Anita Hansbo and Peter Hansbo. A finite element method for the simulation of strong and weak discontinuities in solid mechanics. *Computer Methods in Applied Mechanics and Engineering*, 193(33 - 35):3523 – 3540, 2004.
- [23] Peter Hansbo. Nitsche’s method for interface problems in computational mechanics. *GAMM-Mitteilungen*, 28(2):183–206, 2005.
- [24] Hansbo, Anita, Hansbo, Peter, and Larson, Mats G. A finite element method on composite grids based on nitsche’s method. *ESAIM: M2AN*, 37(3):495–514, 2003.
- [25] Martin Howard, Andrew D. Rutenberg, and Simon de Vet. Dynamic compartmentalization of bacteria: Accurate division in *E. Coli*. *Phys. Rev. Lett.*, 87:278102, Dec 2001.
- [26] Kerwyn Casey Huang, Yigal Meir, and Ned S. Wingreen. Dynamic structures in *Escherichia coli*: Spontaneous formation of min rings and mid polar zones. *Proceedings of the National Academy of Sciences*, 100(22):12724–12728, 2003.
- [27] Ashley J. James and John Lowengrub. A surfactant-conserving volume-of-fluid method for interfacial flows with insoluble surfactant. *J.*

- Comput. Phys.*, 201(2):685–722, December 2004.
- [28] Mika Juntunen and Rolf Stenberg. Nitsche’s method for general boundary conditions. *Math. Comput.*, 78(267):1353–1374, 2009.
- [29] Dimitrios Kamilis. Numerical Methods for the PDES on Curves and Surfaces. Master’s thesis, Umeå University, 2013.
- [30] E. Kreyszig. *Differential Geometry*. Dover Books on Mathematics. Dover Publications, 2013.
- [31] Mats Larson and Fredrik Bengzon. *The Finite Element Method: Theory, Implementation, and Applications*, volume 10. Springer, 2013.
- [32] Z. Li, T. Lin, Y. Lin, and R. C. Rogers. An immersed finite element space and its approximation capability. *Numerical Methods for Partial Differential Equations*, 20(3):338–367, 2004.
- [33] Zhilin Li, Tao Lin, and Xiaohui Wu. New cartesian grid methods for interface problems using the finite element formulation. *Numerische Mathematik*, 96(1):61–98, 2003.
- [34] A. Massing, M. Larson, and A. Logg. Efficient implementation of finite element methods on nonmatching and overlapping meshes in three dimensions. *SIAM Journal on Scientific Computing*, 35(1):C23–C47, 2013.
- [35] Brian Mirtich. Fast and accurate computation of polyhedral mass properties. *J. Graph. Tools*, 1(2):31–50, February 1996.
- [36] Metin Muradoglu and Gretar Tryggvason. A front-tracking method for computation of interfacial flows with soluble surfactants. *Journal of Computational Physics*, 227(4):2238 – 2262, 2008.
- [37] Sundararajan Natarajan, Stéphane Bordas, and D. Roy Mahapatra. Numerical integration over arbitrary polygonal domains based on schwarz - christoffel conformal mapping. *International Journal for Numerical Methods in Engineering*, 80(1):103–134, 2009.
- [38] Sundararajan Natarajan, D. Roy Mahapatra, and Stéphane P. A. Bordas. Integrating strong and weak discontinuities without integration subcells and example applications in an xfem-gfem framework. *International Journal for Numerical Methods in Engineering*, 83(3):269–294, 2010.
- [39] 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, 1971.
- [40] Igor L. Novak, Fei Gao, Yung-Sze Choi, Diana Resasco, James C. Schaff, and Boris M. Slepchenko. Diffusion on a curved surface coupled to diffusion in the volume: Application to cell biology. *Journal of Computational Physics*, 226(2):1271 – 1290, 2007.
- [41] Maxim Olshanskii and Arnold Reusken. A finite element method for surface pdes: matrix properties. *Numerische Mathematik*, 114(3):491–520, 2010.
- [42] Maxim A. Olshanskii, Arnold Reusken, and Jörg Grande. A Finite Element Method for Elliptic Equations on Surfaces. *SIAM Journal on Numerical Analysis*, 47(5):3339–3358, 2009.

- [43] Nenad Pavin, Hana Čipčíć Paljetak, and Vladimir Krstić. Min-protein oscillations in *Escherichia coli* with spontaneous formation of two-stranded filaments in a three-dimensional stochastic reaction-diffusion model. *Phys. Rev. E*, 73:021904, Feb 2006.
- [44] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge University Press, New York, NY, USA, 3 edition, 2007.
- [45] A.N. Pressley. *Elementary Differential Geometry*. Springer Undergraduate Mathematics Series. Springer London, 2013.
- [46] Arnold Reusken and Trungchieu Nguyen. Nitsche's method for a transport problem in two-phase incompressible flows. *Journal of Fourier Analysis and Applications*, 15(5):663–683, 2009.
- [47] Christopher Essex Robert Adams. *Calculus: A Complete Course, Seventh Edition*. Pearson Education Canada, 2009.
- [48] J.A. Sethian. *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science*. Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, 1999.
- [49] Eddie Wadbro, Sara Zahedi, Gunilla Kreiss, and Martin Berggren. A uniformly well-conditioned, unfitted Nitsche method for interface problems. *BIT Numerical Mathematics*, 53(3):791–820, 2013.
- [50] Abdelaziz Yazid, Nabbou Abdelkader, and Hamouine Abdelmajjid. A state-of-the-art review of the X-FEM for computational fracture mechanics. *Applied Mathematical Modelling*, 33(12):4269 – 4282, 2009.
- [51] O.C. Zienkiewicz, R.L. Taylor, and J.Z. Zhu. *The Finite Element Method: Its Basis and Fundamentals: Its Basis and Fundamentals*. Elsevier Science, 2013.
- [52] Paolo Zunino, Laura Cattaneo, and Claudia Maria Colciago. An unfitted interface penalty method for the numerical approximation of contrast problems. *Applied Numerical Mathematics*, 61(10):1059 – 1076, 2011.