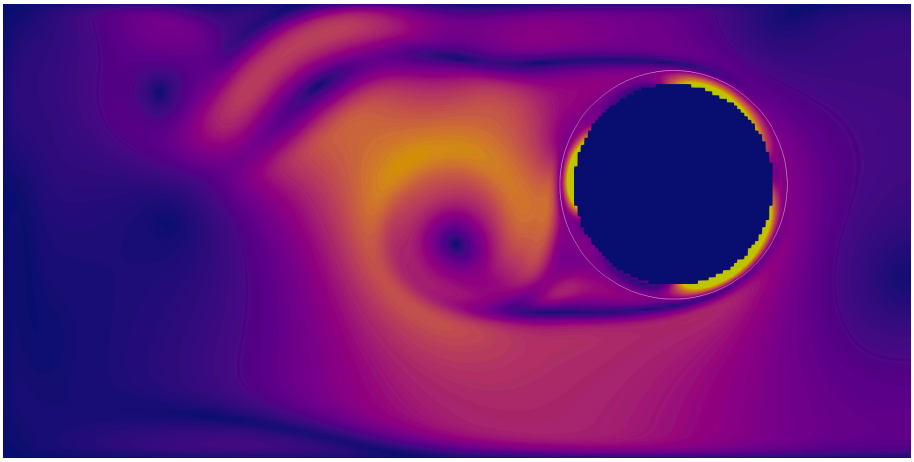
Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Mathematical Sciences

Sigmund Eggen Holm

Cut finite element methods for partial differential equations on moving domains

Master's thesis in Applied Physics and Mathematics
Supervisor: André Massing
March 2022



NTNU
Norwegian University of
Science and Technology

Sigmund Eggen Holm

Cut finite element methods for partial differential equations on moving domains

Master's thesis in Applied Physics and Mathematics
Supervisor: André Massing
March 2022

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Mathematical Sciences

NTNU
Norwegian University of
Science and Technology

Abstract

When modelling physical phenomena in science and engineering, Partial Differential Equations (PDEs) are ubiquitous. A wide range of problems involves describing how fluids interact with the surroundings. Examples include fluid-structure interaction problems and multi-phase flows. Complicated multiphysics problems often involve moving computational domains undergoing large geometrical deformations. A classical finite element method (FEM) with moving meshes might lead to a highly deformed or invalid mesh. Then costly re-meshing might be required. The Cut Finite Element Method (CutFEM) is a promising discretisation technique that allows the geometry of the domain to be represented independently of the computational grid. This method can significantly simplify the mesh generation for problems posed on complex domains.

As an unfitted method, CutFEM allows complex and moving boundaries to be represented independently of the finite element mesh. The geometry of the domain is described as a level set function, and Dirichlet boundary conditions are enforced using Nitsche's method. By extending the weak formulation with a stabilisation term, CutFEM remains a stable and optimally convergent approximation method for arbitrary cut configurations. Since generating high-quality meshes can be costly, CutFEM can offer a vast improvement on computing time, especially when PDEs are solved on moving domains. For stationary domains, the theory of CutFEM is well-developed, and several numerical studies corroborate the theoretical results. However, there are not many detailed studies investigating the convergence properties of CutFEM on moving domains.

In this thesis, we perform a detailed investigation of CutFEM's suitability for solving complex PDEs on *moving* domains. The main focus is to assess the stability and convergence properties of CutFEM. This is done by performing thorough numerical experiments of parabolic problems and flow problems on domains with prescribed motion. We investigate the existing theory for parabolic problems on moving domains. Then, the presented technique is applied for solving fluid dynamical problems on moving domains. The experiments require the practical implementation of CutFEM to cope with challenges not present in the stationary domain case. The detailed experiments show optimal convergence for the heat, Stokes and Navier-Stokes equations on both stationary and moving domains.

Sammendrag

Når kompliserte fysiske fenomener innenfor forskning og ingeniørvitenskap undersøkes, er modellering med partielle differensialligninger (PDE-er) nærmest uunngåelig. Et vidt utvalg av fysiske problemer omhandler hvordan fluider beveger seg og interagerer med omgivelsene. Eksempler på dette kan være problemer som beskriver fluid/struktur-interaksjoner eller flerfasestrømninger. Kompliserte problemer innen multifysikk involverer ofte at domenet gjennomgår store geometriske deformasjoner. Hvis slike problemer løses med den klassiske elementmetoden ved å transformere nettet, kan dette føre til store deformasjoner eller et ugyldig nett. CutFEM er en variant av elementmetoden, og er en lovende diskretiseringsteknikk som tillater geometrien til domenet å representeres uavhengig av elementoppdelingen. Med CutFEM trenger man altså ikke tilpasse nettet til domenet. Denne metoden kan derfor forenkle nettgenereringen betraktelig for problemer på kompliserte domener.

Siden CutFEM er en ikke-tilpasset metode, kan komplekse og bevegelige domenegrenser representeres uavhengig av nettet. Geometrien til domenet beskrives av en nivåmengdefunksjon, og Dirichlet grensebetingelser settes ved hjelp at Nitsches metode. Ved å utvide den svake formuleringen med et stabiliseringsledd, blir CutFEM en stabil og optimalt konvergent diskretiseringsmetode for vilkårlige kuttkonfigurasjoner. Siden generering av et høykvalitetsnett kan være kostbart, kan bruk av CutFEM føre til en stor forbedring i beregningstid. Dette gjelder spesielt for PDE-er løst på bevegelige domener. Når det gjelder stasjonære domener, er teorien bak CutFEM godt utviklet, og et utvalg numeriske studier bekrefter de teoretiske resultatene. Imidlertid finnes det få detaljerte studier som undersøker konvergensegenskapene til CutFEM på bevegelige domener.

I denne oppgaven foretar vi en detaljert undersøkelse av CutFEMs egnethet for løsing av komplekse PDE-er på *bevegelige* domener. Hovedfokuset vil være å evaluere stabilitets- og konvergensegenskapene til metoden. Dette gjøres ved å gjennomføre grundige numeriske eksperimenter av parabolske og fluiddynamiske problemer på domener med kjent bevegelse. Vi undersøker eksisterende teori og teknikker for løsing av parabolske problemer på bevegelige domener. Videre bruker vi disse teknikkene for å løse fluiddynamiske problemer på bevegelige domener. Eksperimentene krever at den praktiske implementeringen av CutFEM

håndterer utfordringer som ikke forekommer for problemer på stasjonære domener. De grundige konvergensstudiene viser at CutFEM oppnår optimal konvergens for varmeligningen, Stokes-problemet og Navier-Stokes-ligningene på både stasjonære og bevegelige domener.

Preface

This master thesis marks the end of my years at the Norwegian University of Science and Technology and concludes my degree in Applied Physics and Mathematics.

When I started my studies at NTNU, I initially enrolled in a Computer Science degree. I was aiming for artificial intelligence as my field of study, as it seemed to be the 'most' mathematical of the available specialisations. However, at the start of each semester, I always found myself browsing for various subjects on the Department of Mathematical Science's web pages. In particular, I remember finding the course 'Partial Differential Equations', which piqued my interest. After the third year, I decided to take the plunge and start a degree in Applied Physics and Mathematics. I am very satisfied with my change of course, and with the colourful animations I have produced during the work with this thesis.

First of all, I want to thank Associate Professor André Massing. He has been an invaluable supervisor, and his enthusiasm has been greatly motivating and inspiring. I immensely appreciate his helpful debugging advice and his patience when answering all of my theoretical questions.

Secondly, I want to thank Simon Sticko. Without his help and deep knowledge of numerical mathematics, I would not have had the rapid progress when learning `C++` and `deal.II`. I also want to thank Bjørn Atle Angelsen for inspiring the idea behind this thesis. He has provided me with fascinating insight into how mathematical modelling is used in medical research.

Lastly, I want to thank (and apologise to) my friends and family for enduring my rants about bugs, whether new or old. Also, they should be thanked for providing me with support and motivation throughout the work with this thesis.

Sigmund Eggen Holm, Trondheim, March 2021

Contents

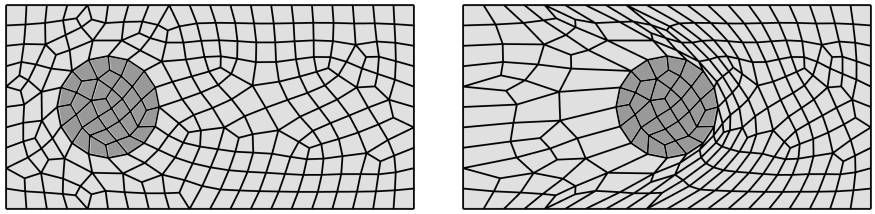
Chapter 1

Introduction

In mathematical modelling, partial differential equations (PDEs) have proved to be an especially powerful and versatile tool. PDEs have successfully been employed to model phenomena from nature, economy and engineering and has been an essential part of numerous technological and scientific advances. However, solving PDEs analytically is only possible for problems in their most simple form. Therefore, one must mostly resort to numerical approximations when simulating realistic problems with advanced geometries or based on multiphysics.

Among many other discretisation methods, the finite element method (FEM) is a widespread technique for solving PDEs numerically. As both computer processors and memory has kept the pace of capacity increase according to Moore's law, the size and resolution of PDE problems solved have been able to follow the same steep growth. However, computing power and computing time are still limited resources, now as ever. Therefore, developing methods that lead to more efficient algorithms is the core research motivation in numerical mathematics.

When applying FEM, creating a mesh for the physical domain can be a markedly part of the computing effort. This is especially the case for time-dependent problems on moving domains. When dealing with moving domains in classical fitted FEM, the Arbitrary Lagrangian-Eulerian (ALE) formulation is often preferred [Donea et al., 2004, Fernández et al., 2009]. When solving, e.g. fluid-structure interaction problems [Richter, 2017] one is interested in the fluid velocity field in the fluid domain and the material displacement in the structural domain. Since the fluid boundary is deformed, the computational mesh needs to follow this deformation. The PDEs governing the fluid is typically given in Eulerian coordinates, but the moving domains are often represented using moving meshes with an artificial motion. Therefore, the fluid flow equations need to be rewritten using ALE coordinates. However, for large deformations or topological changes, the underlying mesh motion might lead to highly distorted or invalid finite element meshes, see Figure 1.1. This might require costly re-meshing.



(a) Initial mesh. **(b)** Transformed mesh.

Figure 1.1: On the left we have the initial configuration of a matched mesh. The transformed mesh on the right, leads to a mesh with lower quality and remeshing might be needed.

Therefore, it can be advantageous to consider an unfitted method since the geometry of the physical domain can then be represented independently of a fixed background mesh, see Figure 1.2. This eliminates the need for complicated mesh generation or algorithms for moving the mesh along with the domain.

A particular instance of unfitted FEM is the eXtended Finite Element Method (XFEM), which was introduced in [Moës et al., 1999]. In this method, the polynomial space is enriched to allow for the representation of cracks or interfaces independently of the mesh [Fries and Belytschko, 2010]. However, the method is not stabilised, so the error and the condition number of the system matrix are therefore not necessarily robust in the presence of so-called small cut elements. These are elements cut by the domain boundary in such a way that the intersection of the element and the domain is significantly smaller than the element itself.

The Cut Finite Element Method (CutFEM) is an alternative unfitted method. This method was developed in [Burman and Hansbo, 2012, Burman and Hansbo, 2014, Burman et al., 2015], and allows full decoupling of the physical domain from the computational mesh. In CutFEM, the domain is represented as the zero contours of a level set function, allowed to cut arbitrary through a fixed background mesh. Dirichlet boundary conditions are then enforced weakly using Nitsche's method. In addition, the weak formulation is extended with stabilisation terms. This simplifies the translation from classical fitted discretisation methods to the unfitted scenario. In addition to the advantages of easy mesh generation, CutFEM offers the same convergence rates of the error and scaling of the condition number of the stiffness matrix as classical FEM. Combined, this makes CutFEM a promising alternative when solving time-dependent PDEs on complex geometries. In problems with moving interfaces, CutFEM is even more advantageous. Since a pre-generated general background mesh can be used, moving the domain's interface amounts to moving a level set function defined on top of the mesh. In [Burman et al., 2015] CutFEM was used to discretise an inter-

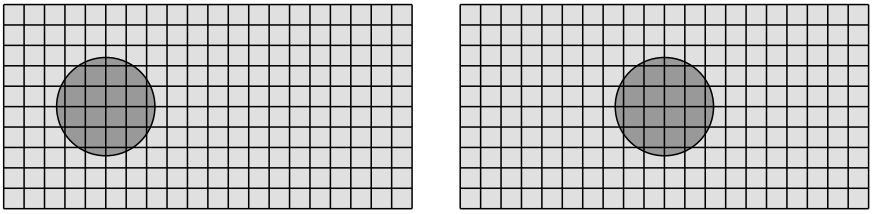


Figure 1.2: When using an unfitted method, the geometry of the domain can be moved arbitary over the background mesh.

face problem of a variant of the Poisson problem, and in [Massing et al., 2014] the Stokes problem was solved. A moving domain method for CutFEM was introduced in [Lehrenfeld and Olshanskii, 2019] for the Poisson problem, and for the Stokes problem in [Burman et al., 2022]. Since then, CutFEM has been used in a multitude of model problems. In [Massing et al., 2015, Schott et al., 2019] fluid-structure interaction problems were solved, and in [Claus and Kerfriden, 2019, Frachon and Zahedi, 2019] two-phase flow problems were analysed. In [Burman et al., 2019] a method for flow in fractured porous media was proposed. The Trace Finite Element Method (TraceFEM) introduced in [Olshanskii et al., 2009] is a variant of CutFEM, developed for solving surface and surface-bulk problems. Instead of extending the surface functions, one takes the trace of the background volumetric finite element functions on the embedded surface. The method was further extended for evolving surfaces in [Olshanskii and Xu, 2017, Lehrenfeld et al., 2018].

For stationary domain problems, the CutFEM theory is well developed, and the promising properties of the method have been assessed through numerous numerical studies. However, there are few theoretical results for moving domains, and numerical investigations of the stability of CutFEM is largely lacking.

Contributions and outline. In this thesis, we evaluate a practical realisation and the suitability of CutFEM for numerically solving complex PDEs on moving domains. The robustness and correctness of the method will be analysed through detailed numerical experiments of flow problems.

We start by assessing the CutFEM theory for parabolic problems and flow problems on stationary domains. Further, we examine the theory and techniques for solving moving domain problems. We then apply the CutFEM techniques for parabolic problems on moving domains to flow problems. Extensive numerical convergence tests are performed with the aim of contributing to filling in the gap of numerical studies of problems on moving domains. The work culminates in the implementation of a one-way coupled fluid rigid body problem, where the fluid is modelled with the Navier-Stokes equations, and the submerged body moves along a known path.

In Chapter 2 an introduction to CutFEM is given. The chapter uses the heat equation as the model problem and derives a CutFEM formulation of the problem. First, we introduce Nitsche's method for weakly enforcing the Dirichlet boundary conditions. The time derivative is discretised using the BDF-method before the stabilisation needed for CutFEM is finally added. The chapter also features a short review of the theory behind how the CutFEM stabilisations assess control of the norms on small cut elements and hence results in a stable approximation. Numerical experiments corroborate the convergence results for the heat equation. The following Chapter 3 explains how CutFEM can be used for solving fluid dynamical problems. By first considering the stationary Stokes problem, the inf-sup condition is briefly explained to lead up to stable function spaces for flow problems. Again, the weak formulation is first extended with the needed Nitsche terms for the Stokes problem before the fully discrete formulation of the time-dependent problem is given. Moreover, the CutFEM stabilisations for the Stokes problem are given, and finally, a CutFEM formulation for the Navier-Stokes problem is derived. The implementations of the time-dependent Stokes problem and Navier-Stokes equations attains optimal order of convergence. In Chapter 4 the CutFEM formulation for the heat equation in Chapter 2 is extended for a moving domain. By extending the active mesh, the stabilised weak formulation implicitly defines an extension operator for the approximated solution. This enables the terms of the BDF-method to be well defined, even as the domain moves. Finally, numerical implementations of the heat equation on a moving domain are presented. The CutFEM formulation for a moving domain is then further extended for flow problems in Chapter 5. CutFEM formulations for the time-dependent Stokes and Navier-Stokes equations are presented before the results of the convergence tests are presented.

Chapter 2

CutFEM for parabolic problems on stationary domains

In this chapter, we start by giving an introduction to Nitsche's method. This method is used for enforcing Dirichlet boundary conditions in a weak manner. When using an unfitted method, the boundary of the domain can cut through the elements of the mesh arbitrarily. Nitsche's method is therefore central for imposing Dirichlet boundary conditions when using CutFEM. As the heat equation is used as the model problem of the section, a multistage time-stepping method is presented to give a fully discrete Nitsche formulation on a fitted mesh. In Section 2.2, an introduction to CutFEM is given, based on the Poisson problem. The Nitsche formulation of the previous section is then further extended by a stabilisation term to ensure the unfitted mesh method is stable and optimally convergent. The section is concluded with a CutFEM formulation for the heat equation. Lastly, numerical results for the solution of the heat equation using CutFEM is presented in Section 2.3. The results corroborate the theoretical results presented in Section 2.2.2, which state that CutFEM achieves the same optimal convergence in error as classical FEM.

2.1 Nitsche's method for the heat equation

The heat equation is a well-known parabolic partial differential equation (PDE). This PDE can describe physical phenomena as the evolution of the concentration of some physical quantity or how the density of heat evolves in some domain over time.

Let $\Omega \subset \mathbb{R}^d$ be an open and bounded domain, with a piecewise smooth boundary $\Gamma = \partial\Omega$. The heat equation is an initial boundary-value problem defined on

Ω, given by

$$u_t - \nu\Delta u = f \quad \text{in } (0,T)\times\Omega, \tag{2.1a}$$
$$u = g \quad \text{on } [0,T]\times\Gamma, \tag{2.1b}$$
$$u(0,\cdot) = u_0 \quad \text{in } \Omega. \tag{2.1c}$$

The function u is a scalar valued function in two variables, such that $u = u(t,x) : [0,T]\times\Omega \to \mathbb{R}$, where $T > 0$ is the end time. The diffusivity coeffisient $\nu > 0$ is often referred to as the thermal diffusivity of the medium. Here, $u_t = \partial_t u = \partial u/\partial t$ is the first order derivative of u with respect to time, while $\Delta = \sum_{i=1}^{d} \partial_{x_i}^2 u$ denotes the Laplacian. The right hand side $f : [0,T]\times\Omega \to \mathbb{R}$ is scalar valued, and represents a heat source. Written on the form above, Dirichlet boundary conditions are applied along the whole boundary with the function $g : [0,T]\times\Gamma \to \mathbb{R}$. The function $u_0 : \Omega \to \mathbb{R}$ is the given initial state of the problem. For a vector $x = (x_1,\dots,x_d) \in \mathbb{R}^d$, we denote the 2-norm by $\|x\|_2 = \sqrt{x\cdot x}$.

For a classical, strong solution to the problem (2.1), we would require the function u to be one time continuously differentiable in time and twice continuously differentiable in space. Finding a function u that satisfies the equation in the strong form might be impossible. Therefore, we will instead search for *weak* solutions to the problem. To find such solutions, we will use the method of Galerkin to introduce a weak formulation of the problem. Discretising this weak formulation is the key idea of the Finite Element Method (FEM).

To derive a weak formulation of the problem, multiply with a suitable test function v on both sides of (2.1a), and integrate by parts over the domain using Green's identity. This gives,

$$\int_\Omega u_t v\,dx + \nu\int_\Omega \nabla u\nabla v\,dx - \nu\int_\Gamma v\partial_n u\,dS = \int_\Omega fv\,dx, \tag{2.2}$$

where $\partial_n v = \nabla v\cdot n$ denotes the directional derivative of v in direction n, the outward pointing normal of the domain. The L^2 inner product over some subset $K \subset \mathbb{R}^d$ will be denoted by $(v,w)_K = \int_K vw\,dx$, and the corresponding L^2-norm is defined by $\|v\|_K \coloneqq (v,v)_K^{1/2}$. For a surface $\partial K \subset \mathbb{R}^{d-1}$ the L^2 inner product is denoted by $(v,w)_{\partial K} = \int vw\,dS$. The space of all square integrable functions on K is denoted by $L^2(K)$, that is

$$L^2(K) = \{v : K \to \mathbb{R} : \int_K v^2\,dx < \infty\}. \tag{2.3}$$

When solving the problem in the weak sense, we search for a solution u satisfying (2.2) for all test functions v in some test function space V. Since u is time-dependent, this must hold for all times $t \in [0,T]$. The choice of function space depends on the required regularity of u and v. In this case, only one weak

derivative is needed. We will therefore let the test function space be the Sobolev space $H^1(\Omega)$, where we define

$$H^m(\Omega) = \{v \in L^2(\Omega) : D^\alpha v \in L^2(\Omega) \text{ for } |\alpha| \leqslant m\}. \tag{2.4}$$

Above, $D^\alpha v$ denotes the weak derivative of order α, where α is a multi index (see e.g. [Borthwick, 2018, Ch. 10]). The Sobolev norm will be denoted by

$$\|v\|_{m,K}^2 = \|v\|_{H^m(K)}^2 \coloneqq (v,v)_K = \sum_{|\alpha|\leqslant m} (D^\alpha v, D^\alpha v)_K, \tag{2.5}$$

and $|v|_{m,K} = \sum_{|\alpha|=m}(D^\alpha v, D^\alpha v)_K$ is the corresponding seminorm.

So far, we have not described what function space u belongs to. Since u is time dependent, we can consider it as a mapping $u : [0,T] \to V$, for some Banach space V. For any fixed $t \in [0,T]$, u takes values in the Banach space V, such that $u(t) = u(t,\cdot) \in V$. The function u is referred to as a vector-valued function in [Tröltzsch, 2010, Ch. 3.4]. Now let the space $L^2(a,b;V)$ denote the space of vector-valued functions $u : [a,b] \to V$ s.t.

$$\|u\|_{L^2(a,b;V)}^2 = \int_a^b \|u\|_V^2 \, dt < \infty. \tag{2.6}$$

When denoting the dual space of V by V', we say that a function

$$u \in L^2(0,T;V), \quad \text{with} \quad u_t \in L^2(0,T;V'), \tag{2.7}$$

is a weak solution to the parabolic initial boundary-value problem (2.1) provided (2.2) holds for all $v \in V$ for all $t \in [0,T]$. See [Evans, 2010, Ch. 7.1] or [Tröltzsch, 2010, Ch. 3.4] for a more thorough description.

The usual FEM approach for solving PDEs with Dirichlet conditions applied along the boundary is to define a lifting function to reduce the problem to the homogeneous Dirichlet condition case. One can then let the space V of test functions be $H_0^1(\Omega)$, the subspace of $H^1(\Omega)$, with functions satisfying $v|_\Gamma = 0$. This leads to a vanishing boundary integral in (2.2). In this way, the boundary conditions are built into the chosen function space. This method will thus only work when the discretisation of the domain approximates the boundary Γ, that is, for *fitted* methods. Since we aim to use an *unfitted* method, where the boundary Γ is allowed to pass straight through the elements T of the triangulation, this method for enforcing the boundary conditions will not suffice.

In the following, we will present a method that allows for imposing Dirichlet boundary conditions in a weak sense. Instead of building the boundary conditions into the functions space, the weak formulation of the problem is extended with certain penalty terms. This boundary penalisation method was introduced in [Nitsche, 1971] and is also presented in [Ern and Guermond, 2021, Ch. 37] and [Hansbo, 2005]. The method is now widely referred to as 'Nitsche's method'. But first, we will need to discretise the weak formulation above.

2.1.1 Semidiscrete formulation

The variational formulation of the heat equation given in the previous section is infinite-dimensional. Since we want to use FEM to solve the problem numerically, we need to discretise the problem. In this section, a spatial discretisation of the formulation will be presented.

To perform a spatial discretisation, we first partition the domain into a mesh $\mathcal{T}_h$ of quadrilateral/hexahedral closed elements T. This is done in such a way that the intersection of two elements are either empty, or exactly equal to their common face, side or vertex. For fitted methods, these elements cover the domain, such that $\overline{\Omega} = \cup_{T_k \in \mathcal{T}_h} T_k$. The size of an element T_k is measured by its diameter $h_k = \operatorname{diam} T_k$, where the diameter of a bounded set $K \subset \mathbb{R}^d$ is defined as

$$\operatorname{diam} K = \sup\{\|x - y\|_2 : x, y \in K\}, \tag{2.8}$$

and $\|\cdot\|_2$ denotes the usual 2-norm in $\mathbb{R}^d$. The mesh size h is given by the largest element diameter in $\mathcal{T}_h$, i.e. $h = \max_{T_k \in \mathcal{T}_h} h_k$. If we let ρ_T be the largest inscribed circle in element T, we say that a family of meshes $\{\mathcal{T}_h\}_{h>0}$ is shape regular if there exists some constant $\delta > 0$ such that

$$\frac{h_T}{\rho_T} \leqslant \delta, \quad \forall T \in \mathcal{T}_h. \tag{2.9}$$

As a finite-dimensional subspace of the function space $H^1(\Omega)$, we will use the space of continuous, piecewise polynomials of degree p on $\mathcal{T}_h$,

$$Q_p^c(\mathcal{T}_h) = \{v : v|_T \in Q_p(T), T \in \mathcal{T}_h\} \cap C(\Omega). \tag{2.10}$$

Above, $Q_p(T)$ denotes a finite element of polynomial degree p on a quadrilateral or hexahedral T (see [Brenner and Scott, 2008, Ch. 3]), while $C(\Omega)$ denotes the space of all continuous functions on Ω.

As a discrete test function space we will now use $V_h = Q_p^c(\mathcal{T}_h)$. Now, restrict the search to finding a spatially discrete solution $u_h \in L^2(0, T; V_h)$ with $u_{t,h} \in L^2(0, T, V_h')$ such that

$$(u_{t,h}, v)_\Omega + \nu(\nabla u_h, \nabla v)_\Omega - \nu(\partial_n u_h, v)_\Gamma = (f, v)_\Omega \quad \forall v \in V_h, t \in [0, T], \tag{2.11}$$

such that $u_h(0, \cdot) = u_0(x)$.

The above formulation is often referred to as a semidiscrete formulation [Thomée, 2007, Ch. 1], [Quarteroni, 2017, Ch. 5]. In a later section, a time-stepping method will be introduced to present a fully discrete problem formulation of the problem (2.1). First, however, we will give a brief introduction to Nitsche's method for weakly applying the Dirichlet boundary conditions.

2.1.2 Nitsche's method

To derive the weak Nitsche formulaton for the problem (2.1), we will continue from the variational formulation in (2.11). The discrete test function space will be $V_h = Q_p^c(\mathcal{T}_h)$. In the current variational formulation, the boundary conditions are not implemented. Therefore, the next step is to extend the weak formulation above with some additional terms to apply the boundary conditions in a weak sense.

First, the term $\mu(u_h - g, v)_\Gamma$ is added to (2.11). This term is the Nitsche penalty term, that in a way enforces the Dirichlet boundary condition $u = g$ on Γ. The positive constant μ is the Nitsche penalty parameter. The semidiscrete weak formulation is then updated to finding a $u_h \in L^2(0, T; V_h)$ with $u_{t,h} \in L^2(0, T; V_h')$ such that

$$\begin{aligned}(u_{t,h}, v)_\Omega + \nu(\nabla u_h, \nabla v)_\Omega - \nu(\partial_n u_h, v)_\Gamma + \mu(u_h, v)_\Gamma \\ = (f, v)_\Omega + \mu(g, v)_\Gamma \qquad \forall v \in V_h, \forall t \in [0, T].\end{aligned} \tag{2.12}$$

Ignoring the term $(u_{t,h}, v)_\Omega$ in the expression above, we note that the bilinear form defined by the terms on the left-hand side is not symmetric. Therefore, a symmetrisation term

$$-\nu(u_h - g, \partial_n v)_\Gamma, \tag{2.13}$$

is added to the weak formulation. Note that the added terms vanish when the solution satisfies $u = g$ on the boundary Γ. The weak formulation is thus still consistent with the original problem. The Nitsche formulation of the problem (2.1), is then formulated as follows. Find a $u_h \in L^2(0, T; V_h)$ with $u_{t,h} \in L^2(0, T, V_h')$ such that

$$(u_{t,h}, v)_\Omega + a_h(u_h, v) = l_h(v) \quad \forall v \in V_h, t \in [0, T], \tag{2.14}$$

if $u_h(0, \cdot) = u_0(x)$. The bilinear form $a_h : V_h \times V_h \to \mathbb{R}$ and the linear form $l_h : V_h \to \mathbb{R}$ are defined as

$$\begin{aligned}a_h(u_h, v) &= \nu(\nabla u_h, \nabla v)_\Omega - \nu(\partial_n u_h, v)_\Gamma - \nu(u_h, \partial_n v)_\Gamma + \mu(u_h, v)_\Gamma, \\ l_h(v) &= (f, v)_\Omega - \nu(g, \partial_n v)_\Gamma + \mu(g, v)_\Gamma.\end{aligned} \tag{2.15}$$

In [Hansbo, 2005] it was shown for the Poisson problem that if the Nitsche penalty parameter is set to

$$\mu = \frac{\nu\gamma}{h}, \qquad \gamma > 0, \tag{2.16}$$

the bilinear form a_h is discretely coercive in a mesh dependent energy norm, implying stability of the discretised method. Also, it was shown that the Nitsche formulation resulted in optimal convergence in the energy norm and the L^2-norm. When using Q_p finite elements, this means that the error converges as h^{p+1} when measured in the L^2-norm, and as h^p when measured in the energy norm.

Table 2.1: Coefficients for the BDF-methods up to order 33.

s	α_0	α_1	α_2	α_3
1	1	-1		
2	$3/2$	-2	$1/2$	
3	$11/6$	-3	$3/2$	$-1/3$

The Nitsche formulation above is still continuous in time, so to be able to solve the equation (2.1) for a discrete number of time steps, a time-stepping method has to be introduced. This will be done by replacing the first term in (2.14) by a finite difference approximation. In the next section, we will see that the time-stepping method for the heat equation involves solving the Poisson problem with an added reaction term in every time step. The results in [Hansbo, 2005] can easily be shown to hold for this problem with the same Nitsche penalty parameter if we include the reaction term in the mesh dependent norm.

2.1.3 A fully discrete Nitsche formulation

To rewrite the semidiscrete Nitsche formulation (2.14) of the problem (2.1) into a fully discrete formulation, the time derivative has to be approximated. This can be done by applying the Backward Difference Formula (BDF) method, a finite difference approximation to the first-order derivative. The BDF-s method is a multistage approximation of order s. First, we discretise the time into uniform time steps of length τ, and write $t_n = n\tau$. Then, a finite element problem is solved in each of these time steps. The total number of time steps is $M = T/\tau$. An upper index denotes the solution at a given time step, such that $u(t_n) = u^n$, and correspondingly $u_h(t_n) = u_h^n$.

The time derivative of the solution u is approximated by the solutions in the previous time steps. The first term in the semidiscrete formulation (2.14), is approximated as

$$(u_t(t_n), v)_\Omega \approx \frac{1}{\tau} \sum_{k=0}^{s} \alpha_k (u^{n-k}, v)_\Omega, \tag{2.17}$$

for an s-stage BDF-method, referred to as BDF-s. A s-stage BDF-method is an approximation of order s, and the coefficients up to order three is listed in Table 2.1. Note that a stage-1 BDF-method is just the implicit Euler method.

When the above approximation for the derivative in (2.17) is inserted into the spatially discrete formulation in (2.14), we arrive at the following fully discrete Nitsche formulation of the heat equation. Given $u_h^j \in V_h$ for $j = 0, \ldots, s-1$, find

u_h^n such that

$$\sum_{k=0}^{s} \alpha_k (u_h^{n-k}, v)_\Omega + a_h(u_h^n, v) = l_h(v) \quad \forall v \in V_h, \tag{2.18}$$

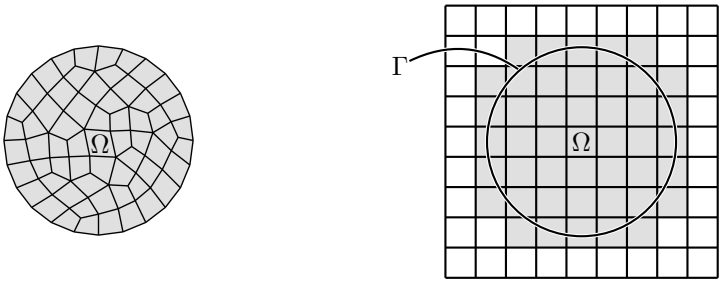
for $n = s, s+1, \ldots, M$. The bilinear form a_h and the linear form l_h are defined as in (2.15). Note that u_h^0 is the interpolated initial value of the problem. Also, note that only the first term u_h^n in the sum is unknown. The other terms in the sum are solutions of previous time steps and can be moved to the right-hand side and assembled into the right-hand side vector of the linear system. When BDF-1 is run, only the interpolated initial values u_h^0 are needed to run the method. If a BDF-method of higher order is run, more steps are needed to start the method. In practice, one would therefore need to run one step each of BDF-1, BDF-2 and so on, up to BDF-$(s-1)$ before the rest of the time-steps can be run with BDF-s. In numerical experiments, all the initial steps needed for the start-up can be interpolated when solving a problem with a known analytical solution.

So far, the discretisation has been for a fitted mesh, where the elements T in the triangulation $\mathcal{T}_h$ approximate the domain boundary. In the following section, the Cut Finite Element Method (CutFEM) will be introduced. This method allows the boundary Γ to cut straight through the mesh elements. This enables the geometry of the problem to be represented as a level set function, and we only need a background mesh covering the domain. This representation will be a significant simplification when solving time-dependent problems on moving domains, which is the topic of Chapter 4 and Chapter 5. However, the decoupling of the mesh and the physical domain introduces new problems with regard to stability. To retain a stable method, the weak formulation needs to be extended with a stabilising *ghost penalty* term. The following section will explain why this stabilisation is needed, and we will give exact realisations of this term.

2.2 An introduction to CutFEM

In the previous section, Nitsche's method enabled us to enforce Dirichlet boundary conditions weakly by extending the weak formulation by specific penalty terms. When using this method, there was no longer a need to only search for solutions in the space of functions already satisfying the Dirichlet boundary conditions. However, the method was still a *fitted* finite element method since the edges of the elements in the triangulation were required to approximate the domain boundary.

In this section, the Cut Finite Element Method (CutFEM) will be presented. This method allows us to use a mesh that does not fit the domain, i.e. we use an *unfitted* mesh, see Figure 2.1. This means that the domain boundary may cut through the elements arbitrarily. This can be a vast simplification of geometry representation and mesh generation. With CutFEM, one only needs a uniform



(a) Fitted mesh. **(b)** Unfitted mesh.

Figure 2.1: On the left, the mesh approximates the domain boundary, while on the right, it cuts through the elements of the background mesh. The gray cells in the figure on the right make up the active mesh $\mathcal{T}_h$, consisting of all elements in the background mesh $\widetilde{\mathcal{T}}_h$ with a non-empty intersection with the physical domain Ω.

background mesh covering the domain instead of generating a mesh that fits the boundary. In other words, CutFEM enables complete decoupling of the physical domain and the mesh triangulation.

CutFEM was first introduced in [Burman and Hansbo, 2012], where the treatment was given based on the Poisson problem, discretised using continuous Galerkin $\mathbb{P}_1$-elements. The Nitsche method for weakly imposed boundary conditions was extended to a fictitious domain. The domain was represented by a level set function defined on the background mesh. The first challenge when dealing with an unfitted mesh is computing the stiffness matrix contributions of cut elements. Since we only want to compute contributions over $T \cap \Omega$, new quadrature rules are needed. The second challenge is reestablishing control of the L^2-norm of the gradient on the cut elements. This instability is especially challenging on small cut elements, where $|T \cap \Omega| \ll |T|$. In [Burman and Hansbo, 2012], it was proved that when the weak formulation was extended with a *ghost penalty* [Burman, 2010,Burman et al., 2015], coercivity could be ensured over the whole computational domain. In the paper, optimal a priori estimates for the H^1- and L^2-norm was proved.

In [Gürkan and Massing, 2019], an abstract framework for a flexible design of CutFEM stabilisations was developed for cut discontinuous Galerkin methods. General assumptions were put on an abstract ghost penalty to prove optimal convergence of the error and scaling of the condition number. This section will give a short review of this paper in the context of continuous CutFEM for the Poisson problem. A more thorough review was done in my project report [Holm, 2021].

2.2.1 Challenges with a cut mesh

When the domain now is allowed to cut through the elements of the mesh, some new notation is needed. Let $\widetilde{\Omega} \subset \mathbb{R}^d$ be a background domain covering the physical domain Ω, such that $\Omega \subset \widetilde{\Omega}$. The background domain is partitioned into a shape regular, quasi-uniform background mesh $\widetilde{\mathcal{T}}_h$ of quadrilateral/hexahedral closed elements T. The physical domain is now represented by a level set function $\phi : \widetilde{\Omega} \to \mathbb{R}$, defined on the background mesh as

$$\Omega = \{x \in \mathbb{R}^d : \phi(x) < 0\}. \tag{2.19}$$

The computational mesh, or the active mesh $\mathcal{T}_h$ is defined as the smallest set of elements in the background mesh covering the domain. This is given by

$$\mathcal{T}_h = \{T \in \widetilde{\mathcal{T}}_h : T \cap \Omega \neq \emptyset\}, \tag{2.20}$$

while the fictitious domain is the union of these elements,

$$\Omega^* = \bigcup_{T \in \mathcal{T}_h} T. \tag{2.21}$$

This will also be referred to as the computational domain. The set of all cut elements is given by

$$\mathcal{T}_\Gamma = \{T \in \mathcal{T}_h : T \cap \Gamma \neq \emptyset\}. \tag{2.22}$$

All interior faces of the active mesh is given by

$$\mathcal{F}_h = \{F = T^+ \cap T^- : T^+, T^- \in \mathcal{T}_h\}. \tag{2.23}$$

For certain cut configurations, Nitsche's method will converge optimally for problems solved on an unfitted mesh. The problem arises on meshes with arbitrary cut elements, particularly when some elements have a so-called 'bad' cut. This is a cut where the cell has an arbitrary small intersection with the domain, i.e. $|T \cap \Omega| \ll |T|$. See Figure 2.2 for a depiction of this situation. The condition number of the stiffness matrix may then grow arbitrary large [Ludvigsson et al., 2018, Gürkan and Massing, 2019].

CutFEM adds a stabilisation term to the weak form to aid with this instability. This term is referred to as the *ghost penalty* g_h and stabilises the faces of cut cells. These faces are called ghost penalty faces, and are denoted by

$$\mathcal{F}_h^g = \{F \in \mathcal{F}_h : T^+ \cap \Gamma \neq \emptyset \vee T^- \cap \Gamma \neq \emptyset\} \tag{2.24}$$

When the weak formulation of the problem is extended with this ghost penalty, we arrive at a stable and optimally converging method [Burman and Hansbo, 2012]. The stiffness matrix also has optimal scaling of the condition number [Gürkan and Massing, 2019], and all these properties are fully independent of the cut configuration.

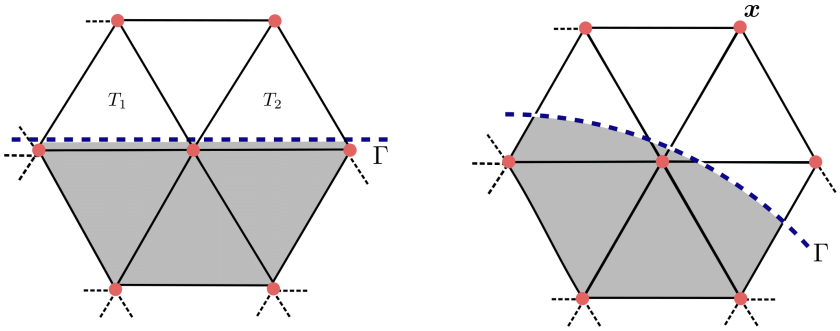


Figure 2.2: Two examples where the boundary Γ cuts the elements in a 'bad' manner. On the left, the constant in the inverse estimate (2.30) becomes unbounded, ruining the coercivity of the weak form. On the right, the bad cut leads to an almost singular stiffness matrix, making it ill-conditioned. The figures were taken from [Gürkan and Massing, 2019].

As in [Gürkan and Massing, 2019], the Poisson problem will be used as our model problem. This is given by

$$-\nu\Delta u = f \quad \text{in } \Omega \tag{2.25a}$$
$$u = g \quad \text{on } \Gamma. \tag{2.25b}$$

Since the the domain now cuts the background mesh, the discretised function space will need to be defined on the whole active mesh. We therefore let

$$V_h = Q_p^c(\mathcal{T}_h) = \{v : v|_T \in Q_p(T), T \in \mathcal{T}_h\} \cap C(\Omega^*). \tag{2.26}$$

Note that the test function space above is defined on the whole fictitious domain. A weak formulation of (2.25) is derived using Nitsche's method, as presented in Section 2.1. This leads to the following variational problem. Find $u_h \in V_h$ such that

$$a_h(u_h, v) = l_h(v) \qquad \forall v \in V_h. \tag{2.27}$$

The bilinear and linear forms above are still defined as in (2.15), but are restated below,

$$\begin{aligned} a_h(u,v) &= \nu(\nabla u, \nabla v)_\Omega - \nu(\partial_n u, v)_\Gamma - \nu(u, \partial_n v)_\Gamma + \mu(u,v)_\Gamma, \\ l_h(v) &= (f,v)_\Omega - \nu(g, \partial_n v)_\Gamma + \mu(g,v)_\Gamma. \end{aligned} \tag{2.28}$$

As before, the Nitsche penalty parameter is set to

$$\mu = \frac{\nu\gamma}{h}, \tag{2.29}$$

for some positive constant γ. The weak formulation in (2.27) is a Nitsche formulation defined on the whole fictitious domain. However, without the stabilising

ghost penalty, it is not stable on arbitrary cut configurations. The stabilised CutFEM formulation is presented in the following section.

2.2.2 Stability and convergence of CutFEM

When the stability of the Nitsche formulation for the Poisson problem was proved in [Hansbo, 2005], an essential ingredient was the following inverse estimate.

Proposition 2.1 (Inverse estimate)
There exists some positive constant C_I such that for all $v \in V_h$

$$\|h^{1/2}\partial_n v\|_\Gamma^2 \leqslant C_I \|\nabla v\|_\Omega^2 \quad \text{for all } v \in V_h. \tag{2.30}$$

where C_I depends on the polynomial degree p of V_h, the spatial dimension d and the chunkiness parameter c_T.

This estimate was used to control the boundary flux terms of the bilinear form by the L^2-norm of the gradient over the domain. This allowed the author to prove discrete coercivity for Nitsche's method in a mesh dependent energy norm. Optimal convergence was shown for the fitted mesh method.

When the active mesh now is extended past the domain boundary, this poses a new challenge for the proof of stability through showing coercivity of the bilinear form. Now we only have the following inverse estimate.

Proposition 2.2 (Cut mesh inverse estimate)
Let $v \in V_h$, then there exists some positive constant C_Γ such that

$$\|h^{1/2}\partial_n v\|_\Gamma^2 \leqslant C_\Gamma \|\nabla v\|_{\mathcal{T}_h}^2. \tag{2.31}$$

Note that this estimate only enables controlling the boundary flux terms by the L^2-norm of the gradient over the *whole* active mesh. Therefore, we will now need a way of controlling the L^2-norm of the gradient over the whole active mesh. One possible remedy for this was introduced in [Burman, 2010,Burman and Hansbo, 2012]. The main idea is to augment the bilinear form with a stabilisation term referred to as the *ghost penalty* g_h. Our revised stabilised cut finite element formulation is now as follows. Find $u_h \in V_h$ such that

$$A_h(u_h, v) \coloneqq a_h(u_h, v) + g_h(u_h, v) = l_h(v) \qquad \forall v \in V_h. \tag{2.32}$$

Note that for A_h to be bilinear, the same must be required from g_h. We will also assume that g_h is symmetric and positive semi-definite. However, we do not require that g_h is consistent with the problem.

For the analysis, the following norms and semi-norms are defined for $v \in V_h$.

$$\begin{aligned} |v|_{g_h}^2 &\coloneqq g_h(v,v) \\ \|v\|_{a_h}^2 &\coloneqq \nu\|\nabla v\|_\Omega^2 + \nu\|h^{-1/2}v\|_\Gamma^2 \\ \|v\|_{A_h}^2 &\coloneqq \|v\|_{a_h}^2 + |v|_{g_h}^2 = \nu\|\nabla v\|_\Omega^2 + \nu\|h^{-1/2}v\|_\Gamma^2 + |v|_{g_h}^2. \end{aligned} \tag{2.33}$$

For $v \in H^2(\mathcal{T}_h) + V_h$ we also define

$$\|v\|_{a_h,*}^2 \coloneqq \|v\|_{a_h}^2 + \|h^{1/2}\partial_n v\|_\Gamma^2. \tag{2.34}$$

The first abstract assumption g_h is required to fulfil is the following.

Assumption 1 (Ghost Penalty)
The ghost penalty *is a symmetric, positive semi-definite bilinear form, and satisfies*

$$\|\nabla v\|_{\mathcal{T}_h}^2 \leqslant C_g \left(\|\nabla v\|_\Omega^2 + |v|_{g_h}^2\right) \tag{2.35}$$

for some positive constant C_g for all $v \in V_h$.

This requirement enables controlling the problematic term $\|\nabla v\|_{\mathcal{T}_h}$, as discussed above. When the bilinear form is extended with the ghost penalty, we are able to control the gradient on the whole active mesh. One can therefor show discrete coercivity of A_h over $v \in V_h$ in the A_h-norm (2.33). Boundedness of A_h over V_h also follows from Proposition 2.2 and Assumption 1. By Lax-Milgram's Theorem [Brenner and Scott, 2008, Ch. 2.7], the discrete coercivity and the boundedness of A_h imply that the formulation yields a unique solution, which is continuous in the data.

Usually, when proving a priori estimates for a finite element method, we combine Cea's Lemma or a Strang type Lemma with a projection or interpolation estimate. Since we are now dealing with a mesh cut by the domain, defining a projection operator is more technical.

Following the approach in [Burman and Hansbo, 2012], a projection operator defined on the whole active mesh can be constructed by first making use of the fact that there exists a bounded extension operator

$$(\cdot)^e : H^m(\Omega) \to H^m(\mathbb{R}^d), \tag{2.36}$$

such that

$$\|v^e\|_{m,\mathbb{R}^d} \lesssim \|v\|_{m,\Omega}, \quad \text{for } v \in H^m(\Omega), \tag{2.37}$$

see [Stein, 1971]. This extension operator can now be used to define a new approximation operator $\pi_h^e : L^2(\Omega) \to V_h$ by

$$\pi_h^e v = \pi_h v^e. \tag{2.38}$$

Here the approximation operator π_h is some quasi interpolator, e.g. the Clément operator (see e.g. [Larson and Bengzon, 2013, Ch. 7.2.1]).

From e.g. [Burman and Hansbo, 2014] we have the following error estimates for the projection interpolant π_h^e. For $v \in H^{m+1}(\Omega)$ it holds that

$$\begin{aligned} \|v^e - \pi_h^e v\|_{r,\mathcal{T}_h} &\lesssim h^{m+1-r}\|v\|_{m+1,\Omega} \quad \text{for} \quad 0 \leqslant r \leqslant m+1, \\ \|v^e - \pi_h^e v\|_{r,\mathcal{F}_h} &\lesssim h^{m-r+1/2}\|v\|_{m+1,\Omega} \quad \text{for} \quad 0 \leqslant r \leqslant m+1/2, \\ \|v^e - \pi_h^e v\|_{r,\Gamma} &\lesssim h^{m-r+1/2}\|v\|_{m+1,\Omega} \quad \text{for} \quad 0 \leqslant r \leqslant s-1/2 \end{aligned} \tag{2.39}$$

These estimates are used to derive an estimate for the projection error in the $\|\cdot\|_{a_h,*}$-norm.

As mentioned earlier in this section, the ghost penalty g_h is not consistent with the problem. This leads to the following weak Galerkin Orthogonality.

Lemma 2.3 (Weak Galerkin Orthogonality)
Let $u \in H^2(\Omega)$ be the solution of (2.25)*, and let $u_h \in V_h$ be a solution to the weak formulation* (2.32)*, then*

$$a_h(u - u_h, v) = g_h(u_h, v) \quad \forall v \in V_h. \tag{2.40}$$

When this weak Galerkin orthogonality is used in proving an a priori estimate, we end up with some additional term $|\pi_h^e u|_{g_h}$. Therefore, the ghost penalty needs to be restricted by another assumption.

Assumption 2 (Weak consistency estimate)
For $v \in H^{m+1}(\Omega)$ and $s = \min\{m, k\}$, the semi-norm $|\cdot|_{g_h}$ satisfies the estimate

$$|\pi_h^e v|_{g_h} \lesssim h^s \|v\|_{s+1,\Omega}. \tag{2.41}$$

Above, the notation $a \lesssim b$ denotes that $a \leqslant Cb$, where C is some positive constant independent of h and the position of Γ relative to the background mesh $\widetilde{\mathcal{T}}_h$. As shown in [Gürkan and Massing, 2019], this lead to the following a priori error estimate for the CutFEM approximation.

Theorem 2.4 (A priori estimate in the a_h*-norm)
Let $u \in H^{m+1}(\Omega)$ for $m \geqslant 1$ be the solution of the Poisson problem (2.25)*, and let $u_h \in V_h = \mathbb{P}_p^c(\mathcal{T}_h)$ be the solution to the weak formulation* (2.32)*. Then for $s = \min\{m, p\}$ we have the following estimate for the approximation error*

$$\|u - u_h\|_{a_h*} \lesssim h^s \|u\|_{s+1,\Omega}. \tag{2.42}$$

Remark. Note that the proofs in [Gürkan and Massing, 2019] was performed for triangular polynomial elements defined by

$$\mathbb{P}_p^c(\mathcal{T}_h) = \{v : v \in \mathbb{P}_p(T), \forall T \in \mathcal{T}_h\} \cap C(\Omega^*), \tag{2.43}$$

where $\mathbb{P}_p(T)$ denotes the space polynomials of degree p on a triangular element T. However, the results still hold for quadrilateral elements $Q_p^c(\mathcal{T}_h)$.

As shown in [Gürkan and Massing, 2019], by applying the Aubin-Nitsche trick (see, e.g. [Quarteroni, 2017, Ch. 4.5]), the above estimate can be used to derive an a priori error estimate in the L^2-norm.

Theorem 2.5 (A priori estimate in the L^2-norm)
Let $u \in H^{m+1}(\Omega)$ for $m \geqslant 1$ be the solution of the Poisson problem (2.25), and let $u_h \in V_h = \mathbb{P}_p^c(\mathcal{T}_h)$ be the solution to the weak formulation (2.32). Also assume that the boundary Γ is C^2. Then for $s = \min\{m, p\}$ we have the following estimate for the approximation error

$$\|u - u_h\|_\Omega \lesssim h^{s+1}\|u\|_{s+1,\Omega}. \tag{2.44}$$

The paper [Gürkan and Massing, 2019] proceeds to prove that the scaling of the condition number of the stiffness matrix with respect to the mesh size h when using CutFEM is identical to that of classical FEM [Quarteroni, 2017, Ch. 4.5]. The condition number estimate requires two additional assumptions on the ghost penalty.

Assumption 3
The ghost penalty g_h satisfies

$$\|v\|_{\mathcal{T}_h}^2 \lesssim \|v\|_\Omega^2 + |v|_{g_h}^2 \quad \forall v \in V_h. \tag{2.45}$$

This assumption implies that adding ghost penalty g_h to the bilinear form enables controlling the L^2-norm of a discrete function over the active mesh by the L^2-norm over the physical domain. The assumption is used to prove a discrete Poincaré inequality, stating that for discrete functions $v \in V_h$, it holds that

$$\|v\|_{\mathcal{T}_h} \lesssim \|v\|_{A_h}. \tag{2.46}$$

Further, a final assumption is needed to prove an inverse estimate in the A_h-norm.

Assumption 4
For $v \in V_h$ the ghost penalty satisfies

$$|v|_{g_h} \lesssim h^{-1}\|v\|_{\mathcal{T}_h}. \tag{2.47}$$

Finally, the condition number of the stiffness matrix resulting from the discrete weak CutFEM formulation (2.32) is proved to satisfy

$$\kappa(A) \lesssim h^{-2}, \tag{2.48}$$

where the hidden constants are independent of the cut configuration. See [Gürkan and Massing, 2019] for the proof.

2.2.3 Ghost penalty realisations

The paper [Gürkan and Massing, 2019] proceeds to present possible realisations of the abstract ghost penalty. The face-based ghost penalty is the stabilisation used in the numerical experiments of this thesis.

Face-based ghost penalties

Recall from the earlier discussion that the problematic element cuts were those leading to a tiny intersection $T \cap \Omega$ with the domain. The proof for the face-based stabilisation in [Gürkan and Massing, 2019] is based on showing how the L^2-norm of a discrete function $v \in V_h$ on an element T_1 can be controlled by the norm on a neighbouring element T_2 when a certain sum is added to the weak formulation.

First, we need to introduce some notation for the normal derivative. Let the multi index $\alpha = (\alpha_1, \dots, \alpha_d)$ be defined such that $|\alpha| = \sum_i \alpha_i$, $n^\alpha = n_2^{\alpha_2} n_2^{\alpha_2} \dots n_d^{\alpha_d}$ and $\alpha! = \alpha_1! \alpha_2! \cdots \alpha_d!$. Then define

$$\partial_n^j v = \sum_{|\alpha|=j} \frac{D^\alpha v(x) n^\alpha}{\alpha!}. \tag{2.49}$$

The jump across an interior face $F \in \mathcal{F}_h$, is denoted by

$$[w]|_F = w_F^+ - w_F^-, \tag{2.50}$$

where $w(x)^\pm = \lim_{t \to 0} w(x \pm tn)$ for some chosen unit normal on the face F.

In [Gürkan and Massing, 2019] the following lemma was proved.

Lemma 2.6
Let $T_1, T_2 \in \mathcal{T}_h$ be two elements sharing a common face F. Then

$$\|v\|_{T_1}^2 \lesssim \|v\|_{T_2}^2 + \sum_{0 \leqslant j \leqslant p} h^{2j+1} ([\partial_n^j v], [\partial_n^j v])_F \quad \forall v \in V_h. \tag{2.51}$$

The hidden constants only depends on the shape regularity of $\mathcal{T}_h$, the polynomial order p and the dimension.

In Section 2.2.1, the ghost penalty faces were defined as

$$\mathcal{F}_h^g = \{F \in \mathcal{F}_h : T^+ \cap \Gamma \neq \emptyset \vee T^- \cap \Gamma \neq \emptyset\}. \tag{2.52}$$

These are the faces between adjacent elements cut by the boundary Γ, or between two elements $T^+ \in T_h$ and $T^- \in \mathcal{T}_\Gamma$. We consider a cut through an element T being 'bad' if $|T \cap \Omega \ll |T|$. These cuts might lead to an almost singular and ill-conditioned stiffness matrix if no ghost penalty is added (see e.g. [Gürkan and Massing, 2019] and a depiction in Figure 2.2). When the mesh is shape-regular, and the boundary is C^2, the following fat intersection property is fulfilled.

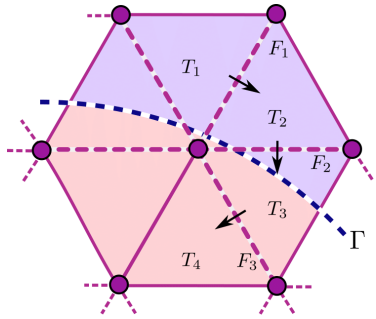


Figure 2.3: Depiction of how the L^2-norm of a discrete function on a barely intersected element T_1 can be controlled by adding a jump-penalty for each face until an element satisfying the fat-intersection property is reached. The figure was taken from [Gürkan and Massing, 2019].

Let a patch P around element T be defined as the union of all elements sharing at least one vertex with T.

Definition 2.1 (Fat intersection property)
For $T \in \mathcal{T}_\Gamma$ there is a patch P of diam $P \lesssim h$ *which contains T and an element T' with a* fat intersection *satisfying*

$$|T' \cap \Omega|_d \geqslant c_s |T'|_d \tag{2.53}$$

for some mesh independent $c_s > 0$.

The fat intersection property above can be combined with Lemma 2.6 to show that the contribution of an element with a 'bad' cut can be controlled by 'walking' across the faces until an element without a cut is found. This is depicted in Figure 2.3. The fat intersection property implies that the number of faces needed to cross is bounded. When Lemma 2.6 is applied to all the elements in the active mesh, one can control the L^2-norm of a function $v \in V_h$ defined on the active mesh, by its norm on the physical domain when the ghost penalty is added. The results enabling controlling the norms on any 'bad cut' implies that the method is stable for any cut configuration.

Finally, [Gürkan and Massing, 2019] provides a proof for that the following ghost penalty satisfies Assumption 1-4,

$$g_h(v, w) \coloneqq \sum_{j=0}^{p} \sum_{F \in \mathcal{F}_h^g} \gamma_j h_F^{2j-1} ([\partial_n^j v], [\partial_n^j v])_F, \tag{2.54}$$

for $v, w \in V_h$, and positive parameters $\{\gamma_j\}_{j=0}^p$.

Projection-based ghost penalties

Another form of ghost penalty introduced in [Burman, 2010] is the local projection-based stabilisation. Let P be a patch of $\text{diam}(P) \lesssim h$, and let $\pi_P : L^2(P) \to \mathbb{P}_p(P)$ be the L^2-projection onto the space of polynomials of order p on the patch P. The global ghost penalty is then defined as

$$g_h(v,w) = \sum_{P\in\mathcal{P}} g_P(v,w), \qquad g_P(v,w) = h^{-2}(v-\pi_P v, w-\pi_P w)_P. \tag{2.55}$$

Above, $v, w \in V_h$ and $\mathcal{P} = \{P\}$ is a collection of patches.

2.2.4 A CutFEM formulation for the heat equation

In this section, we will present a CutFEM formulation of the heat equation (2.1) on an unfitted mesh. In Section 2.1.3 the equation was discretised in both space and time using the BDF-method, and the boundary conditions were applied weakly by the use of Nitsche's method. The section ended with a fully discrete Nitsche formulation of the problem in (2.18). The CutFEM formulation extends this Nitsche formulation with the ghost penalty. When implementing the heat equation, we want to stabilise both the mass matrix and the stiffness matrix. Therefore, we use the following implementation of the ghost penalty in the experiments,

$$g_h(u,v) = \sum_{F\in\mathcal{F}_h^g}\sum_{k=0}^{p} \frac{1}{2k+1}\frac{h^{2k+1}}{(k!)^2}([\partial_n^k u],[\partial_n^k v])_F. \tag{2.56}$$

This stabilisation was implemented and used in [Ludvigsson et al., 2018].

Let V_h be a discrete function space defined on the whole fictitious domain as in (2.26). Given $u_h^j \in V_h$ for $j = 0, 1, \ldots, s-1$, find $u_h^n \in V_h$ such that

$$\sum_{k=0}^{s} \alpha_k (u^{n-k}, v)_\Omega + a_h(u_h^n, v) + \beta \cdot g_h(u_h^n, v) = l_h(v) \quad \forall v \in V_h, \tag{2.57}$$

for $n = s, s+1, \ldots, M$. The constant $\beta > 0$ is a scaling parameter for the ghost penalty, given by

$$\beta = \beta_0(\alpha_0 + \nu\tau/h^2), \tag{2.58}$$

where β_0 is a positive scaling constant. The ghost penalty scales as an L^2-norm. The first term stabilises the mass matrix, while the second term stabilises the stiffness matrix. Since the discrete formulation is multiplied with τ, the time-step is also a factor of the second term.

The bilinear form a_h and the linear form l_h is defined as in (2.15). The formulation is based on a BDF-method of stage s. The earlier steps can either be provided by interpolating the initial steps or solved by first running one step each of BDF-1, BDF-2 and so on up to BDF-$(s-1)$.

2.3 Numerical experiments

In this section, we conduct numerical experiments to corroborate the theoretical results presented in Section 2.2.2. A CutFEM solver for the heat equation (2.1) was implemented, assessing the convergence properties for the BDF-1 and BDF-2 time stepping schemes.

The examples were implemented[1] using the open-source finite element library `deal.II` [Arndt et al., 2020]. When the work on this thesis started, `deal.II` did not have full support for CutFEM. Therefore, the codebase was written using a development version of `deal.II`, developed by Simon Sticko, documented at [Simon Sticko, 2021]. This version of the library included support for a cut mesh. First and foremost, this includes a class for classifying the cut cells as either 'inside', 'intersected' or 'outside' with respect to the domain boundary. Secondly, the development version of the library implemented the quadrature rules in [Saye, 2015], for integrating over cut cells.

2.3.1 Convergence tests

The convergence tests were performed using the method of manufactured solution. The problem is solved for a known solution multiple times on an increasingly globally refined mesh. In each increased refinement level, the mesh size and time step were both halfed. The error for a time-dependent numerical approximation is computed in the L^2L^2- and the L^2H^1-norm, defined by

$$\begin{aligned} \|u - u_h\|_{L^2L^2} &= \int_0^T \|u - u_h\|_\Omega^2 \, dt \approx \tau \sum_{n=0}^{M} \|u - u_h^n\|_\Omega^2, \\ \|u - u_h\|_{L^2H^1} &= \int_0^T \|u - u_h\|_{1,\Omega}^2 \, dt \approx \tau \sum_{n=0}^{M} \|u - u_h^n\|_{1,\Omega}^2. \end{aligned} \tag{2.59}$$

Let the numerical solution computed on a grid with refinement level k be denoted by $u_{h,k}$, and denote the error of this solution by

$$E_k = \|u - u_{h,k}\|_{L^2\omega}, \tag{2.60}$$

where ω is either L^2 or H^1. Let the size of a grid of refinement level k be denoted by h_k. The experimental order of convergence (EOC) is then computed by

$$\mathrm{EOC}_k = \frac{\log(E_k/E_{k-1})}{\log(h_k/h_{k-1})}. \tag{2.61}$$

The discrete function spaces used was the quadrilateral elements $Q_p^c(\mathcal{T}_h)$ for $p = 1, 2$. Recall that the finite element function space (2.26) is defined on the whole active mesh.

[1] The code is openly available at `https://github.com/sigmundholm/microbubble`.

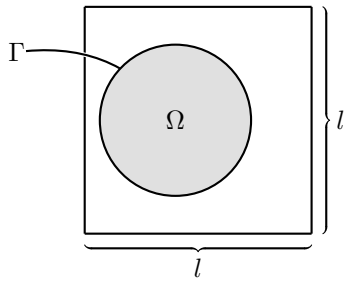


Figure 2.4: Domain used for the convergence tests for the heat equation.

The a priori estimates stated in Theorem 2.4 and Theorem 2.5 showed that the error of CutFEM approximation of a stationary problem is expected to converge as h^p in the H^1-norm, and as h^{p+1} in the L^2-norm. This is the same optimal convergence estimates as in classical FEM (see e.g. [Quarteroni, 2017, Ch. 4]). Furthermore, the BDF-method of stage s converges as τ^s when solving an ODE. Therefore, measured in the norms defined above in (2.59), the best convergence rates we can hope for is

$$\begin{aligned} \|u - u_h\|_{L^2L^2} &\leqslant \mathcal{O}(\tau^s) + \mathcal{O}(h^{p+1}), \\ \|u - u_h\|_{L^2H^1} &\leqslant \mathcal{O}(\tau^s) + \mathcal{O}(h^p). \end{aligned} \tag{2.62}$$

The experiments were conducted in two dimensions with a square background domain with side lengths $l = 2$. The physical domain Ω was set to be a sphere of radius $r = 3l/8$, described by the level set function,

$$\phi(x_1, x_2) = \sqrt{(x_1 - a)^2 + (x_2 - b)^2} - r. \tag{2.63}$$

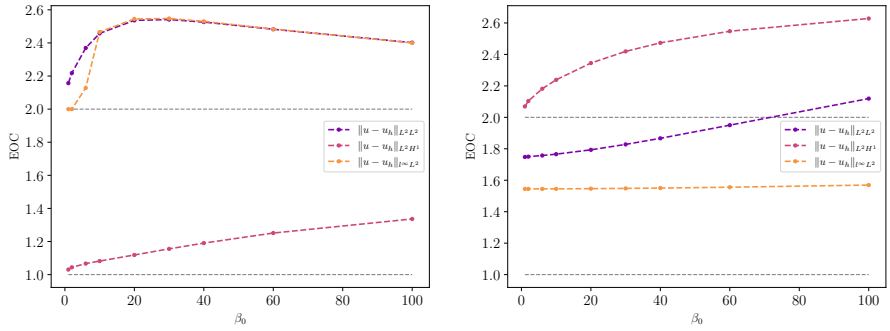
The center of a sphere was set at the point $(a, b) = (-0.9(l/2-r), 0)$. The domain is depicted in Figure 2.4. At the start of the program, the level set function was interpolated onto the discrete function space V_h, to get the discrete level set $\phi_h \in V_h$.

The analytical solution used was

$$u(x, t) = \sin(\pi x_1) \sin(\pi x_2) e^{-t}. \tag{2.64}$$

This is enforced by computing the corresponding right-hand side function f for the heat equation (2.1), and using the known solution for setting Dirichlet boundary conditions along the whole boundary Γ.

The equation parameter ν was set to a value of 2. The end time was set to $T = 1$ to ensure $h = \tau$ for all convergence tests. The mesh consisted of uniform quadrilaterals, and refinement level k produced a background mesh with $N = 2^k$ elements along each side of the background domain. The tests where run from refinement level $k = 3$ through $k = 8$.



(a) Q_1-elements. **(b)** Q_2-elements.

Figure 2.5: The result of the parameter search for the ghost penalty scaling constant β_0. For each data point, a convergence test for the heat equation using BDF-2 (with BDF-1 as the initial step) was run.

The Nitsche penalty parameter was set to

$$\mu = 20p(p+1)/h. \tag{2.65}$$

As the convergence tests were run, it was observed that the ghost penalty scaling constant β_0 had a large impact on the EOC when running BDF-2 with BDF-1 for the initial step. Therefore, a parameter search was performed. The results are depicted in Figure 2.5. In the figure, we see that for $p = 2$, the parameter needs to be set larger than 80 to achieve a second-order convergence in the L^2L^2-norm[2]. With these results in mind, the parameter was set to

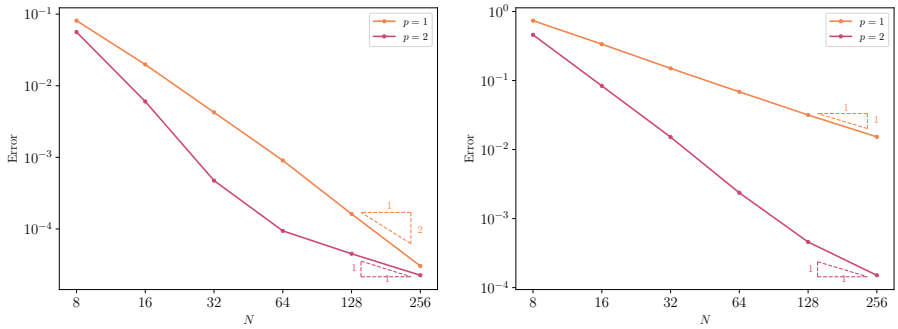
$$\beta_0 = 8 \cdot 10^{p-1}, \tag{2.66}$$

for further experiments. The reason for the factor 10^{p-1}, is that this was set as the p-dependency on this parameter in [Sticko et al., 2021], although that paper used a constant of $\beta_0 = 1 \cdot 10^{p-1}$.

In Figure 2.6 the convergence plot for the heat equation using BDF-1 is shown. Due to the low order of the time-stepping method, one can not expect more than first-order convergence in any norm. We thus see that optimal convergence is attained.

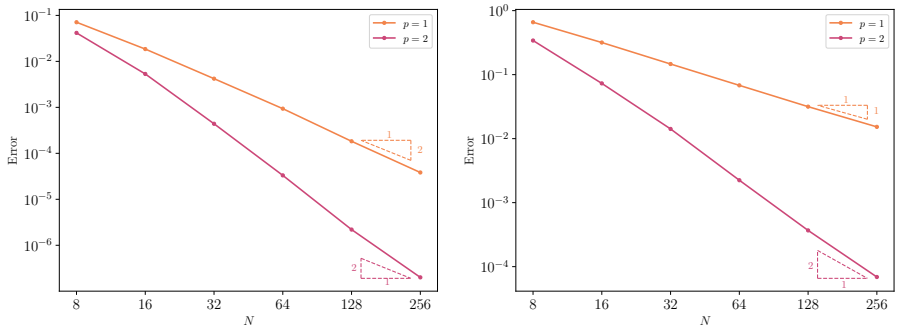
In the case of BDF-2 with interpolated initial steps, we see from Figure 2.7 that full convergence is achieved. This is also the case for BDF-2 with BDF-1 run for the initial step u_h^1, depicted in Figure 2.8.

[2]Note that in the implementation of the ghost penalty (2.56), every face F of each cell in $\mathcal{T}_\Gamma$ is iterated and stabilised given that $F \in \mathcal{F}_h^g$. In effect, every face is stabilised twice. Therefore, in the implementation, the CutFEM stabilisation parameter β_0 is set to half the value stated here.



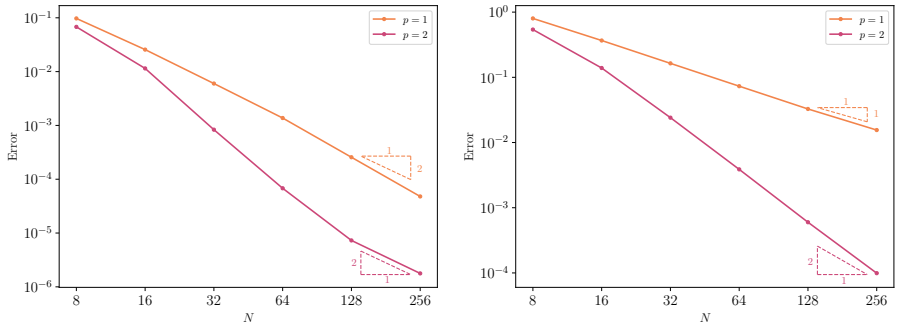
(a) $\|u - u_h\|_{L^2L^2}$ **(b)** $\|u - u_h\|_{L^2H^1}$

Figure 2.6: Convergence plot for the heat equation, solved using BDF-1. The error is plotted against the number of elements N along the sides of the square domain.



(a) $\|u - u_h\|_{L^2L^2}$ **(b)** $\|u - u_h\|_{L^2H^1}$

Figure 2.7: Convergence plot for the heat equation, solved using BDF-2 with the first two steps interpolated. The error is plotted against the number of elements N along the sides of the square domain.



(a) $\|u - u_h\|_{L^2L^2}$ **(b)** $\|u - u_h\|_{L^2H^1}$

Figure 2.8: Convergence plot for the heat equation, solved using BDF-2 with the initial step solved with BDF-1. The error is plotted against the number of elements N along the sides of the square domain.

Chapter 3

CutFEM for flow problems on stationary domains

When modelling physical phenomena using PDEs, problems involving fluids adds multiple complexities. The first challenge involves handling the coupling between pressure and velocity. The theoretical results for the well-posedness of the weak linear Stokes problem are markedly more involved than for elliptical equations. As the Navier-Stokes equations introduce a non-linear convection term to the Stokes problem, well-posedness is a famous open problem.

In the previous chapter, an introduction to CutFEM was given. Since CutFEM allows the physical domain to be represented independently of the finite element mesh, CutFEM can vastly improve computing time. These gains are even more advantageous when solving time-dependent PDEs. Therefore, the features of CutFEM are especially interesting for the simulation of fluid problems.

In this chapter, we review and present theoretical results for CutFEM applied to fluid problems. Due to the new challenges introduced by the coupled problem, Section 3.1 starts by giving a brief introduction to the stationary Stokes problem and how a finite element formulation requires the function spaces to fulfil an inf-sup condition to be stable. In Section 3.2, challenges with regards to the finite element discretisation is discussed. The discrete inf-sup condition leading to stable discrete function space pairs is presented. The variational problem is then extended to a fully discrete Nitsche formulation for the time-dependent Stokes problem. The section concludes with an introduction to CutFEM for flow problems, and finally, we present a CutFEM formulation for the Stokes problem. A discretisation of the Navier-Stokes problem is given in Section 3.4, with the ghost penalty stabilisation required for an unfitted CutFEM formulation. Finally, in Section 3.5 the implemented numerical experiments are presented. The convergence tests corroborate the optimal convergence of CutFEM for flow problems. Classic benchmarks problems for the stationary and time-dependent

Navier-Stokes equations are also presented.

3.1 The Stokes equations

The time-dependent Stokes equations model a viscous, incompressible fluid. The equations model laminar flow where the Reynolds number is low. That is, the equations express the conservation of linear momentum, when the viscous forces are assumed to dominate the inertial forces. Along with the conservation of mass for an incompressible fluid, the vector equations governing viscous flow in an open and bounded domain $\Omega \subset \mathbb{R}^d$ with piecewise smooth boundary $\Gamma = \partial\Omega$, is stated as

$$\rho u_t - \nabla \cdot \sigma = f \quad \text{in } (0,T) \times \Omega \tag{3.1a}$$
$$\nabla \cdot u = 0 \quad \text{in } (0,T) \times \Omega \tag{3.1b}$$
$$u = g \quad \text{on } [0,T] \times \Gamma \tag{3.1c}$$
$$u(0,\cdot) = u_0 \quad \text{on } \Omega \tag{3.1d}$$

Here, $\rho > 0$ denotes the constant density of the fluid, having velocity $u : [0,T] \times \Omega \to \mathbb{R}^d$. The first term of (3.1a), $u_t = \partial_t u = \partial u / \partial t$ denotes the partial derivative of u with respect to time t. The right hand side function $f : [0,T] \times \Omega \to \mathbb{R}^d$ is some body force density, and $\sigma : [0,T] \times \mathbb{R}^d \to \mathbb{R}^{d \times d}$ denotes the Cauchy stress tensor (see e.g. [Eck et al., 2017, Ch. 5]). For a Newtonian, incompressible fluids, the stress tensor is given by

$$\sigma = 2\mu D(u) - pI, \tag{3.2}$$

where $\mu > 0$ is the shear viscosity (dynamic viscosity), $p : [0,T] \times \Omega \to \mathbb{R}$ is the fluid pressure and I denotes the identity matrix. The term $\nabla \cdot \sigma$ express the friction forces between fluid layers. When friction between fluid layers are present, a fast moving fluid particle can accelerate a neighbouring, slower moving particle. This effect will therefore depend upon velocity differences between the layers [Eck et al., 2017, Ch 5.6]. These differences are expressed by the term $D(u)$ which denotes the symmetric part of the velocity gradient ∇u.[1] This symmetric gradient is given by

$$D(u) = \frac{1}{2}\left(\nabla u + (\nabla u)^T\right). \tag{3.3}$$

Now take the divergence of the stress tensor in (3.2), and insert for the symmetric gradient in (3.3). Then use that the velocity field is incompressible (3.1b), and continuously differentiable. We then obtain

$$\nabla \cdot \sigma = \mu \Delta u - \nabla p. \tag{3.4}$$

[1] In this text, ∇u will denote the Jacobian of the vector valued function u, such that $\nabla u = (\nabla u_1 \nabla u_2 \nabla u_3)^T$, for $d = 3$.

This can then be inserted into (3.1a) before dividing by ρ to get the time dependent Stokes equations

$$u_t - \nu\Delta u + \nabla p = f \quad \text{in } (0,T)\times\Omega \tag{3.5a}$$
$$\nabla \cdot u = 0 \quad \text{in } (0,T)\times\Omega \tag{3.5b}$$
$$u = g \quad \text{on } [0,T]\times\Gamma \tag{3.5c}$$
$$u(\cdot,0) = u_0 \quad \text{on } \Omega. \tag{3.5d}$$

Here, the pressure $p : \Omega \times [0,T] \to \mathbb{R}$ and the body force f was redefined by absorbing the reciprocal of the density, and the kinematic viscosity is defined as $\nu = \mu/\rho$. Due to the incompressibility in the equations above, the Dirichlet conditions are required to satisfy the following

$$0 = \int_\Omega \nabla \cdot u \, dx = \int_\Gamma un \, dS = \int_\Gamma gn \, dS. \tag{3.6}$$

Above, n denotes the unit normal vector pointing out of the domain.

Since only the gradient of the pressure is included in the equations (3.5), the pressure variable is only unique up to a constant. The pressure variable can be seen as the variable enforcing the velocity to be incompressible. This is explained in the following section by deriving the Stokes equations to be the first-order optimality condition of a constrained optimisation problem.

Note that in the system of equations (3.5), there are two unknowns, the velocity u and the pressure p, depending on both space and time. Therefore, the problem is said to be coupled, so we need to solve for (u,p) simultaneously. When discretising the problem using FEM, this challenge requires the function spaces to satisfy an inf-sup condition for the variational formulation to be stable. This theory will be briefly presented in the sections to follow.

3.1.1 A saddle point problem

The minimisation of functionals over Banach spaces play an important role in mathematical physics. One of the most famous ones is arguably,

$$\min_{u\in V} J(u) = \frac{1}{2}\|\nabla u\|_\Omega^2 - (f,u)_\Omega. \tag{3.7}$$

This optimisation problem is often referred to as Dirichlet's principle [Borthwick, 2018, Ch. 11], and has it origins in calculus of variations [Evans, 2010, Ch. 8]. Since the functional $J : V \to \mathbb{R}$ is Fréchet differentiable (see [Tröltzsch, 2010, Ch. 2.6]), one can show that a first order necessary optimality condition of (3.7) is

$$\langle J'(u), v\rangle_{V',V} = 0 \quad \forall v \in V. \tag{3.8}$$

Above, the notation for a dual paring is used. Let V' denote the dual space of V, the space of bounded, linear functionals on V. The application of a functional

$F \in V'$ on an element $v \in V$ is then denoted $F(v) = \langle f, v \rangle_{V',V}$. The Fréchet derivative of J in direction u is denoted $J'(u)$, such that, $J' : V \to V'$. By computing this derivative, the optimality condition (3.8) is computed to be

$$(\nabla u, \nabla v)_\Omega = (f, v)_\Omega \quad \forall v \in V. \tag{3.9}$$

We recognise this as the weak formulation of the Poisson problem with Dirichlet boundary conditions when searching for solutions $u \in V = H_0^1(\Omega)$. This example is by no means unique. For many PDEs, we can find a functional J for which the PDE serves as a necessary condition for the minimiser of J [Boffi et al., 2013, Ch. 1].

Optimal control is the study of minimisation problems as the one above but subject to constraints in the form of PDEs. Now let $u : \Omega \to \mathbb{R}^d$ be a vector function, and let $V = [H^1(\Omega)]^d$ be a Sobolev space with vector-valued functions. Consider the constrained problem

$$\min_{u \in V} J(u) = \frac{1}{2} \|\nabla u\|_\Omega^2 - (f, u)_\Omega \tag{3.10}$$

subject to

$$\nabla \cdot u = 0. \tag{3.11}$$

This constraint requires the minimiser of J to be divergence free, that is, incompressible. The optimality conditions for this constrained problem can be found by employing the formal Lagrange method [Tröltzsch, 2010, Ch. 4.7] (also see [Eck et al., 2017, Ch. 6.1.5], [Larson and Bengzon, 2013, Ch. 12.2.3] or [Boffi et al., 2013, 1.3.2] for the current problem). Let V and Q be Hilbert spaces, and $p \in Q$ a Lagrange multiplier. Define the Lagrange functional $\mathcal{L} : V \times Q \to \mathbb{R}$ as

$$\mathcal{L}(u, p) = J(u) - (\nabla \cdot u, p)_\Omega. \tag{3.12}$$

Now we formulate the first order necessary conditions by finding the directional derivatives of the Lagrange functional above,

$$\begin{aligned} \langle \mathcal{L}_u(u, p), v \rangle_{V',V} &= 0 \qquad \forall v \in V, \\ \langle \mathcal{L}_p(u, p), q \rangle_{Q',Q} &= 0 \qquad \forall q \in Q. \end{aligned} \tag{3.13}$$

These optimality conditions amounts to finding the saddle point (u, p) of the unconstrained optimisation problem

$$\min_{v \in V} \max_{p \in Q} \mathcal{L}(u, p). \tag{3.14}$$

A pair $(u, p) \in V \times Q$ is a saddle point of $\mathcal{L}$ if

$$\forall q \in Q, \qquad \mathcal{L}(u, q) \leqslant \mathcal{L}(u, p) \leqslant \mathcal{L}(v, p), \qquad \forall v \in V. \tag{3.15}$$

By Fréchet differentiating the Lagrange functional with respect to u and p respectively, we can write the optimality conditions in (3.13) as

$$\begin{aligned} (\nabla u, \nabla v)_\Omega - (\nabla \cdot v, p)_\Omega &= (f, v)_\Omega \quad && \forall v \in V \\ -(\nabla \cdot u, q)_\Omega &= 0 && \forall q \in Q. \end{aligned} \tag{3.16}$$

As we will see in the next section, this set of optimality conditions is precisely the weak form of the stationary Stokes problem, for the velocity u and pressure p. The pressure arose as a Lagrange multiplier, enforcing the constraint that the velocity should be divergence-free. In the next section we will also see that when the function spaces V and Q, and the inner products in (3.16) satisfies certain conditions, the necessary optimality conditions in (3.16), are indeed sufficient for u to be a solution of the optimisation problem in (3.10)-(3.11) [Ern and Guermond, 2021, Proposition 49.11] [John et al., 2016, Theorem 3.21].

3.1.2 The continuous inf-sup conditions

The coupling of velocity and pressure in the saddle point problem poses a new challenge. Because of this coupling, the finite element function spaces must satisfy certain conditions for the formulation to be stable. The Stokes problem shares this challenge with the Oseen problem, which can be seen as a linearisation of the time-discretised Navier-Stokes equations. In this section, we will therefore give a short introduction to the stability analysis for the stationary Stokes equations. This analysis leads to the discrete inf-sup condition that the approximating function spaces must satisfy.

The presentation below is mainly based on [Brenner and Scott, 2008, Ch. 12] and [Larson and Bengzon, 2013, Ch. 12.2]. Further details, and more general treatments can also be found in e.g. [John et al., 2016, Ch. 3], [Boffi et al., 2013, Ch. 4] or [Ern and Guermond, 2021, Ch. 49].

Consider the stationary, incompressible Stokes equations, given by

$$-\nu \Delta u + \nabla p = f \quad \text{in } \Omega, \tag{3.17a}$$

$$\nabla \cdot u = 0 \quad \text{in } \Omega, \tag{3.17b}$$

$$u = 0 \quad \text{on } \Gamma. \tag{3.17c}$$

Note that when only Dirichlet boundary conditions are applied, and only the pressure gradient is involved, the pressure is unique only up to a constant. Therefore, we let the function space for the pressure have zero-mean to allow for a unique pressure solution.

Let V and Q be Hilbert spaces. In the search for a solution (u, p) to the coupled problem above, we first reformulate the strong form of the system into a weak formulation. This is done by multiplying (3.17a) by a test function $v \in V$,

and multiplying (3.17b) by some $q \in Q$. After integrating by parts we get

$$\begin{aligned} \nu(\nabla u, \nabla v)_\Omega - (\nabla \cdot v, p)_\Omega &= (f, v)_\Omega, \\ (\nabla \cdot u, q)_\Omega &= 0. \end{aligned} \tag{3.18}$$

Let $V = [H_0^1(\Omega)]^d$, and $Q = L_0^2(\Omega) = \{q \in L^2(\Omega) : \int_\Omega p\, dx = 0\}$ [2]. Further, we introduce the two bilinear forms $a : V \times V \to \mathbb{R}$ and $b : V \times Q \to \mathbb{R}$ defined by

$$a(u, v) \coloneqq \nu(\nabla u, \nabla v)_\Omega, \qquad b(u, q) \coloneqq (\nabla \cdot u, q)_\Omega. \tag{3.19}$$

If we now take $F \in V'$ defined by $F(v) = \langle f, v \rangle_{V',V}$, we can write the problem on the form of a general saddle point problem. Find $(u, p) \in V \times Q$ such that

$$\begin{aligned} a(u, v) + b(v, p) &= F(v) \quad \forall v \in V, \\ b(u, q) &= 0 \qquad \forall q \in Q. \end{aligned} \tag{3.20}$$

The bilinear form b above can be defined by a linear operator $B : V \to Q'$, such that

$$b(u, q) = \langle Bu, q \rangle_{Q',Q}. \tag{3.21}$$

If we now for a moment instead search for a solution $u \in \ker B = \{v \in V : b(v, q) = 0 \quad \forall q \in Q\}$, the problem (3.20) reduces to

$$a(u, v) = F(v) \qquad \forall v \in \ker B. \tag{3.22}$$

By Lax-Milgrams Lemma [Brenner and Scott, 2008, Ch. 2.7], this problem is well-posed given that a is coercive on $\ker B$, that is, there exists some $\alpha > 0$ such that

$$\alpha \|v\|_V \leqslant a(v, v) \qquad \forall v \in \ker B. \tag{3.23}$$

Since the pressure vanishes from the equations, this is coherent with what we saw in the previous section. That is, the pressure enforces incompressibility constraint on the velocity u.

With u well defined by (3.22), p can be found such that

$$b(v, p) = F(v) - a(u, v) \qquad \forall v \in V. \tag{3.24}$$

For this problem to be well posed, a new coercivity condition is required of b. This condition requires the existence of a constant $\beta > 0$ such that

$$\beta \|p\|_Q \leqslant \sup_{w \in V} \frac{b(w, p)}{\|w\|_V} \qquad \forall v \in V. \tag{3.25}$$

It is proved in [Brenner and Scott, 2008, Lemma 12.2.12], that if a and b are continuous and b satisfies (3.25), then (3.20) has a unique solution.

[2] The function space V holds vector valued functions, so for dimension $d = 2$, this would mean $[H_0^1(\Omega)]^2 = H_0^1(\Omega) \times H_0^1(\Omega)$.

The condition (3.25) can also be written as

$$\beta \leqslant \inf_{p \in Q} \sup_{w \in V} \frac{|b(w,p)|}{\|w\|_V \|p\|_Q} \qquad \forall v \in V, \tag{3.26}$$

and is hence known as the continuous inf-sup condition.

3.2 Discretisation

In the previous section, we saw that the inf-sup condition was needed for the saddle point problem (3.20) to have a unique solution. In this section, the weak formulation will be discretised. This leads to the need for a discrete version of the inf-sup condition. The discrete function spaces will need to satisfy this condition for the formulation to be stable.

3.2.1 Classical method

The discrete inf-sup condition

When discretising the problem (3.20) above, we first partition the domain Ω into a finite element mesh $\mathcal{T}_h$ of quadrilateral/hexahedral closed elements T. The mesh is shape regular and fitted to the domain, such that $\overline{\Omega} = \cup_{T \in \mathcal{T}_h} T$. On this mesh, define two finite element function spaces V_h and Q_h of piecewise polynomials approximating V and Q. Now, we want to find a discrete solution to the problem $(u_h, p_h) \in V_h \times Q_h$ such that

$$\begin{aligned} a(u_h, v) + b(v, p_h) &= F(v) \quad \forall v \in V_h, \\ b(u_h, q) &= 0 \qquad \forall q \in Q_h. \end{aligned} \tag{3.27}$$

In this coupled finite element problem, the two different variables are approximated using two function spaces. This is referred to as a mixed method.

Recall that the well-posedness of the problem formulation (3.20) was dependent on a being coercive (3.23) on $\ker B$. Now denote the discrete counterpart by

$$\ker B_h = \{v \in V_h : b(v,q) = 0 \quad \forall q \in Q_h\}. \tag{3.28}$$

However, if $u_h \in \ker B_h$ it only means that it is discretely divergence free, that is,

$$b(u_h, q) = \int_\Omega \nabla \cdot u_h q \, dx = 0 \quad \forall q \in Q_h. \tag{3.29}$$

This does not necessary imply that $\nabla \cdot u_h(x) = 0$ for all $x \in \Omega$. In general, $\ker B_h$ is hence not a subspace of $\ker B$. This is also the case for the inf-sup condition (3.25), as it need not hold for V_h and Q_h, even when $V_h \subset V$ and $Q_h \subset Q$ [Larson

and Bengzon, 2013, Ch. 12.2.6], [John et al., 2016, Ch. 3.3]. In the literature, this is known as a variational crime [Brenner and Scott, 2008, Ch. 11].

For the discrete weak formulation (3.27) to be well-posed, the function spaces are required to satisfy a discrete version of the inf-sup condition. That is, there must exists a constant $\beta_h > 0$ such that

$$\beta_h \|p\|_Q \leqslant \sup_{w \in V_h} \frac{b(w,p)}{\|w\|_V} \qquad \forall v \in V_h. \tag{3.30}$$

This condition is also known as the Ladyzhenskaya-Babuska-Brezzi (LBB) condition. The inequality asserts that the velocity function space V_h does not have too many degrees of freedom in comparison to the pressure space Q_h (also see [John et al., 2016, Remark 3.52]). When the discrete inf-sup condition hold, the discrete pressure p_h exists and is unique [Brenner and Scott, 2008, Ch. 12.3].

Stable function spaces

The Taylor-Hood finite elements are widely used approximation spaces for solving incompressible flow problems. For polynomial degree $p \geqslant 2$ and quadrilateral elements T, these are denoted $[Q_p]^d \times Q_{p-1}$, and given by

$$\begin{aligned} V_h &= \{v \in [C(\Omega)]^d : v|_T \in [Q_p(T)]^d, T \in \mathcal{T}_h\} \cap [H^1(\Omega)]^d \\ Q_h &= \{q \in C(\Omega) : v|_T \in Q_{p-1}(T), T \in \mathcal{T}_h\} \cap L^2_0(\Omega) \end{aligned} \tag{3.31}$$

Here, $Q_p(T)$ denotes a finite element of polynomial degree p on a quadrilateral or hexahedral T (see [Brenner and Scott, 2008, Ch. 3]). The exponent d in $[Q_p(T)]^d$ signifies the dimension of the vector valued function element. See [Brenner and Scott, 2008, 12.6] for a proof that this pair of function spaces satisfy the discrete inf-sup condition (3.30) for $p = 2$. A proof for the case $p = 3$, was given in [Brezzi and Falk, 1991], along with the claim that their method could also be used to show the stability for Taylor-Hood elements of arbitrary polynomial degree p for velocities and $p - 1$ for the pressure.

Other stable function space pairs exist, e.g. the MINI element [John et al., 2016, Ch. 3.6]. However, for all experiments of this thesis regarding flow problems, the Taylor-Hood elements will be used.

Another alternative is using a stabilised weak formulation. Then an 'unstable' function space pair can be stabilised by extending the weak form. The continuous interior penalty (CIP) method [Douglas and Dupont, 1976] is a method where equal order interpolation spaces are used for the velocity and pressure function spaces. A face penalty stabilisation term is added to the bilinear form to control the instabilities.

3.2.2 Nitsche's method for the Stokes equations

As in Section 2.1, we want to enforce the Dirichlet boundary conditions weakly by using Nitsche's method. First, we derive a weak formulation for the Stokes problem (3.17). We multiply (3.17a) with a test function $v \in V = [H^1(\Omega)]^d$, and multiply (3.17b) by a test function $q \in Q = L_0^2(\Omega)$. After integrating by parts over the domain we get

$$\nu(\nabla u, \nabla v)_\Omega - \nu(\partial_n u, v)_\Gamma - (\nabla \cdot v, p)_\Omega + (v \cdot n, p)_\Gamma = (f, v)_\Omega, \tag{3.32a}$$

$$(\nabla \cdot u, q)_\Omega = 0. \tag{3.32b}$$

In the expression above, $\partial_n u = \nabla u \cdot n$ denotes the directional derivative of u in direction n. If we now subtract the latter equation above from the former, we obtain

$$\nu(\nabla u, \nabla v)_\Omega - \nu(\partial_n u, v)_\Gamma - (\nabla \cdot v, p)_\Omega + (v \cdot n, p)_\Gamma - (\nabla \cdot u, q)_\Omega = (f, v)_\Omega. \tag{3.33}$$

For the system above, the boundary conditions are not enforced. As was done for the heat equation in Section 2.1, the following boundary penalty term $\mu(u-g, v)_\Gamma$ is added to the equation, where μ is a mesh dependent penalty parameter. When u satisfies the boundary values g on the boundary Γ, this term vanishes, and is thus consistent with the problem. This results in the following weak form,

$$\begin{aligned} \nu(\nabla u, \nabla v)_\Omega - \nu(\partial_n u, v)_\Gamma - (\nabla \cdot v, p)_\Omega + (v \cdot n, p)_\Gamma \\ - (\nabla \cdot u, q)_\Omega + \mu(u, v)_\Gamma = (f, v)_\Omega + \mu(g, v)_\Gamma. \end{aligned} \tag{3.34}$$

The bilinear form defined by the left-hand side in the equation above is not symmetric. To symmetrize the bilinear form, the following terms are added

$$-\nu(u - g, \partial_n v)_\Gamma + ((u - g) \cdot n, q)_\Gamma. \tag{3.35}$$

These terms symmetrizes the second and fourth term in (3.34) respectively. Using the Taylor-Hood finite elements $[Q_p]^d \times Q_{p-1}$ as defined in (3.31), the discrete weak Nitsche formulation of the stationary Stokes problem (3.17) is then: find $(u_h, p_h) \in V_h \times Q_h$ such that

$$a_h(u, v) + b_h(v, p) + b_h(u, q) = l_h(v, q), \qquad \forall (v, q) \in V_h \times Q_h. \tag{3.36}$$

Here we have defined the bilinear forms $a_h : V \times V \to \mathbb{R}$ and $b_h : V \times Q \to \mathbb{R}$ by

$$\begin{aligned} a_h(u, v) &\coloneqq \nu(\nabla u, \nabla v)_\Omega - \nu(\partial_n u, v)_\Gamma - \nu(u, \partial_n v)_\Gamma + \mu(u, v)_\Gamma, \\ b_h(v, p) &\coloneqq -(\nabla \cdot v, p)_\Omega + (v \cdot n, p)_\Gamma, \end{aligned} \tag{3.37}$$

while the linear form $l_h : V_h \times Q_h \to \mathbb{R}$, is given by

$$l_h(v, q) = (f, v)_\Omega + \mu(g, v)_\Gamma - \nu(g, \partial_n v)_\Gamma + (g \cdot n, q)_\Gamma. \tag{3.38}$$

The Nitsche penalty parameter is still

$$\mu = \frac{\nu \gamma}{h}, \tag{3.39}$$

for some scaling constant $\gamma > 0$.

3.2.3 Time discretisation

Now we return to the time dependent Stokes equations from (3.5). This problem only differs from the stationary problem discussed above by the added time derivative u_t. When this term is included in the Nitsche formulation derived above in (3.36), we get the following semidiscrete weak formulation of (3.5). Find $(u_h, p_h) \in V_h \times Q_h$ such that

$$\begin{aligned}(u_{t,h}, v)_\Omega + a_h(u_h, v) + b_h(v, p_h) + b_h(u_h, q) \\ = l_h(v, q) \qquad \forall (v, q) \in V_h \times Q_h, \quad t \in [0, T].\end{aligned} \tag{3.40}$$

This is a semi-discrete formulation of the problem, (see e.g. [Thomée, 2007, Ch. 1]). Similar to the approach for the heat equation in Chapter 2, we look for a weak solution to (3.40) such that

$$\begin{aligned}&u_h \in L^2(0, T; V_h) \quad \text{with} \quad u_{t,h} \in L^2(0, T, V_h'), \\ &p_h \in L^2(0, T; Q_h)\end{aligned} \tag{3.41}$$

However, in this case, we let V_h and Q_h be subspaces of

$$V = [H^1(\Omega)]^d \quad \text{and} \quad Q = L_0^2(\Omega). \tag{3.42}$$

As was done for the time discretisation for the heat equation, the term $(u_t, v)_\Omega$ will be discretised using the BDF-method. This is done by discretising the time t into discrete time steps $t_n = \tau n$, for some constant step length τ. The time derivative of the velocity is then approximated by the solutions in the previous time steps. We thus have the following approximation

$$(u_t(t_n), v)_\Omega \approx \frac{1}{\tau} \sum_{k=0}^{s} \alpha_k (u^{n-k}, v)_\Omega, \tag{3.43}$$

for an s-stage BDF method, referred to as BDF-s. Here, we let $u^n = u$ be the next solution step we want to solve the system for. The coefficients for BDF-s, are denoted by α_k for $k = 0, 1, \ldots, s$. When (3.43) is inserted into (3.40), we get a fully discrete weak formulation of the time dependent Stokes problem. Given $u_h^j \in V_h$ for $j = 0, 1, \ldots, s-1$, find $(u_h^n, p_h^n) \in V_h \times Q_h$ such that

$$\begin{aligned}\frac{1}{\tau} \sum_{k=0}^{s} \alpha_k (u_h^{n-k}, v)_\Omega + a_h(u_h^n, v) + b_h(v, p_h^n) + b_h(u_h^n, q) \\ = l_h(v, q) \quad \forall (v, q) \in V_h \times Q_h,\end{aligned} \tag{3.44}$$

for $n = s, s+1, \ldots, M$, where $M = T/\tau$ is the final time step. To acquire the initial start-up steps for BDF-s, $s \geqslant 2$, we run one step each of BDF-1, BDF-2 up to BDF-$(s-1)$, before BDF-s can be run. If the analytical solution is known, these steps can be interpolated.

3.3 CutFEM for the Stokes problem

In the previous sections, the Stokes problem was discretised on a fitted mesh. As in the previous chapter, we now want to extend this formulation to an unfitted mesh. Let the background domain $\widetilde{\Omega}$ cover the domain Ω, and let the background mesh $\widetilde{\mathcal{T}}_h$ be a quasi-uniform tesselation of $\widetilde{\Omega}$. Now, we allow the boundary Γ of the physical domain Ω to cut the background mesh arbitrarily. First and foremost, this requires the definition of the active mesh $\mathcal{T}_h$ as the set of all elements intersecting the physical domain, that is

$$\mathcal{T}_h = \{T \in \widetilde{\mathcal{T}}_h : T \cap \Omega \neq \emptyset\}. \tag{3.45}$$

The union of the elements in this set make up the computational domain, that is, the fictitious domain, $\Omega^* = \cup_{T\in\mathcal{T}_h} T$. Due to the arbitrary cut configuration of the background mesh, one might find that some of the elements in the active mesh are cut small (also referred to as 'bad' cut elements), which means, $|T \cap \Omega| \ll |T|$. These small cut elements lead to unstable and non-optimal a priori error and condition number estimates. Just as in Section 2.2, the bilinear form is augmented with ghost penalty stabilisations. These ghost stabilisations have the effect of extending the stability properties to the whole fictitious domain Ω^*. However, for this coupled problem, we need to add ghost penalties to stabilise both the velocity and the pressure.

As in Section 2.2, the face-wise ghost penalty stabilisation is used, and the set of ghost penalty faces is defined as

$$\mathcal{F}_h^g = \{F = T^+ \cap T^- : T^+ \in \mathcal{T}_h, T^- \in \mathcal{T}_\Gamma, T^+ \neq T^-\}. \tag{3.46}$$

Recall that $\mathcal{T}_\Gamma$ denotes the set of elements that does not lie completely in Ω:

$$\mathcal{T}_\Gamma = \{T \in \mathcal{T}_h : T \not\subset \Omega\}. \tag{3.47}$$

For the discretisation of the function spaces, we still use the inf-sup stable Taylor-Hood finite elements, but the function spaces will need to be redefined for this unfitted method. Now we let

$$\begin{aligned} V_h &= \{v \in [C(\Omega^*)]^d : v|_T \in [Q_p(T)]^d, T \in \mathcal{T}_h\} \cap [H^1(\Omega^*)]^d \\ Q_h &= \{q \in C(\Omega^*) : v|_T \in Q_{p-1}(T), T \in \mathcal{T}_h\} \cap L_0^2(\Omega^*). \end{aligned} \tag{3.48}$$

Note that the function spaces now include functions defined on the whole fictitious domain Ω^*. The ghost penalties are defined and implemented as

$$\begin{aligned} g_u(u,v) &= \sum_{F\in\mathcal{F}_h^g} \sum_{k=0}^{p} \frac{1}{2k+1} \frac{h^{2k+1}}{(k!)^2} ([\partial_n^k u], [\partial_n^k v])_F, \\ g_p(p,q) &= \sum_{F\in\mathcal{F}_h^g} \sum_{k=0}^{p-1} \frac{1}{2k+1} \frac{h^{2k+1}}{(k!)^2} ([\partial_n^k p], [\partial_n^k q])_F. \end{aligned} \tag{3.49}$$

The Nitsche weak formulation developed in Section 3.2.2 is extended with the ghost penalties above. The resulting formulation will then be stable and optimally convergent for flow problems posed on unfitted meshes.

In [Burman and Hansbo, 2014] the Stokes problem was solved on an unfitted domain using CutFEM. Both inf-sup stable and CIP stabilised velocity and pressure function space pairs were considered. A ghost penalty was used to ensure stability on the cut cells. The paper proved that the inf-sup condition holds for the proposed method over the whole fictitious domain. Optimal a priori error estimates were also proved for an energy norm and the L^2-norm.

The Stokes problem was also studied on an unfitted domain in [Massing et al., 2014]. This was the first contribution where the problem was implemented in three spatial dimensions using CutFEM. However, since low order finite elements were used, the resulting method needed stabilisations to be inf-sub stable. A face-based ghost penalty was used for stability on the whole fictitious domain. Inf-sup stability and an a priori estimate for the fictitious method was proved.

In [Guzmán and Olshanskii, 2018] inf-sup stability for Taylor-Hood elements $[Q_p]^d \times Q_{p-1}$, $p \geqslant 1$, was proved for an unfitted finite element discretisation of the Stokes problem. The analysis can also be applied to a Stokes interface problem, i.e. a two-phase flow problem. Nitsche's method is applied to enforce the Dirichlet boundary conditions, and a face-based ghost penalty is used to attain stability of the pressure over the cut triangles. The paper proves the inf-sup stability over the fictitious domain and optimal convergence for several known stable function space pairs.

The Oseen problem adds a source term and a convection term to the Stokes problem. This is precisely the problem that needs to be solved in each time step of a linearised Navier-Stokes solver, based on a finite difference approximation for the time derivative. In [Massing et al., 2018], CutFEM was applied to this equation. The paper included proofs for in-sup stability and a priori error estimates for the stabilised unfitted finite element method. Since their method used equal order interpolation spaces for the velocity and the pressure, it depended on the continuous interior penalty (CIP) method for stabilising the method. In addition, the method used convection stabilisations as a remedy to deal with convection dominated problems. The paper proved that the facet jump based ghost penalties and the CIP stabilisations could be combined in a single stabilisation for a method more convenient for implementation. However, as mentioned in earlier sections, the implementations for the flow problems in the current thesis will utilise Taylor-Hood finite elements only. Therefore, the technique based on CIP will not be relevant here. Also, since we aim to simulate viscous flow, stabilising high convection is neither relevant.

The CutFEM formulation for the *stationary* Stokes naturally uses Nitsche's method, introduced in Section 3.2.2. The discrete weak Nitsche formulation was given in (3.36). This formulation will now be extended with the ghost penalties defined in (3.49). The CutFEM formulation is now as follows: find $(u_h, p_h) \in$

$V_h \times Q_h$ such that

$$\begin{aligned} a_h(u_h, v) + b_h(p_h, v) + b_h(q, u_h) + \beta_u \cdot g_u(u_h, v) \\ + \beta_p \cdot g_p(p_h, q) = l_h(v, q), \qquad \forall (v, q) \in V_h \times Q_h. \end{aligned} \tag{3.50}$$

Above, the bilinar forms a_h and b_h are defined as in (3.37), while the linear form l_h is defined in (3.38). For the stationary Stokes problem, the ghost penalty stabilisations are scaled with the following constants,

$$\begin{aligned} \beta_u &= \beta_{u,0} \frac{\nu}{h^2} \\ \beta_p &= \frac{\beta_{p,0}}{h}. \end{aligned} \tag{3.51}$$

We now want to develop this formulation of the unfitted stationary Stokes problem into a method for solving the unfitted *time-dependent* Stokes problem. As seen in the previous Section 3.2.3, when the BDF-method is used to discretise the time derivative, a generalised Stokes problem needs to be solved in every time step. The generalized Stokes problem adds a source term scaled by a constant to the left hand side of the stationary Stokes problem, and is thus given by

$$\begin{aligned} \sigma u - \Delta u + \nabla \cdot p &= f \quad \text{in } \Omega \\ \nabla \cdot u &= 0 \quad \text{in } \Omega \\ u &= 0 \quad \text{on } \Gamma. \end{aligned} \tag{3.52}$$

When solved in each time step of a time dependent Stokes problem, the constant $\sigma > 0$ will equal α_0/τ, while the right hand side source function $f : \Omega \to \mathbb{R}^d$ contains the BDF-terms dependent on the previous time steps. Due to the added reaction term, the scaling constants β_u will have to be altered as a remedy of dealing with the emerging mass matrix in this finite element setting. Other than that, the CutFEM formulation for the time dependent formulation is as given in (3.44) extended with the two ghost penalties as presented above in (3.49). Therefore, with the bilinear and linear forms a_h, b_h and l_h unchanged, the discrete weak CutFEM formulation for the time-dependent Stokes problems is the following. Given $u_h^j \in V_h$ for $j = 0, \ldots, s-1$, find $(u_h^n, p_h^n) \in V_h \times Q_h$ such that

$$\begin{aligned} \frac{1}{\tau} \sum_{k=0}^{s} \alpha_k (u_h^{n-k}, v)_\Omega + a_h(u_h^n, v) + b_h(v, p_h^n) + b_h(u_h^n, q) \\ + \beta_u \cdot g_u(u_h^n, v) + \beta_p \cdot g_p(p_h^n, q) = l_h(v, q), \quad \forall (v, q) \in V_h \times Q_h, \end{aligned} \tag{3.53}$$

for $n = s, s+1, \ldots, M$. When implemented for the later numerical experiments, the formulation above is multiplied with the time step τ. Additionally, the inner products including the previous time steps, are naturally included in the right

hand side linear form l_h. The ghost penalty scaling constants will now be set to the values proposed in [Massing et al., 2018, Sec. 3.2], where the Oseen problem was studied. We therefore set,

$$\begin{aligned} \beta_u &= \beta_{u,0}\left(1 + \frac{\tau\nu}{h^2}\right), \\ \beta_p &= \beta_{p,0}\frac{\tau}{\nu + h^2/\tau}. \end{aligned} \tag{3.54}$$

Note that due to the different implementations of the ghost penalty, the scaling constants above differ with a factor of $1/h^2$ when compared to the scalings proposed in [Massing et al., 2018, Sec. 3.2]. Additionally, since we are currently solving the Stokes problem and not the Oseen problem, we omit the convection field weight used for the Oseen problem in the aforementioned paper.

3.4 Navier-Stokes equations

The Navier-Stokes equations express the conservation of linear momentum and the conservation of mass for Newtonian fluids. When compared to the time-dependent Stokes equations, the Navier-Stokes equations adds a non-linear convection term to the system of PDEs. This term accounts for the inertial forces in the fluid. The system of equations are given in the strong form by

$$\begin{aligned} \partial_t u - \nu\Delta u + (u\cdot\nabla)u + \nabla p &= f && \text{in } (0,T)\times\Omega \\ \nabla\cdot u &= 0 && \text{in } (0,T)\times\Omega \\ u &= g && \text{on } [0,T]\times\Gamma \\ u(0,x) &= u_0 && \text{in } \Omega, \end{aligned} \tag{3.55}$$

where $\Omega \subset \mathbb{R}^d$, for $d \in \{2,3\}$ is an open and bounded domain, with piecewise smooth boundary $\Gamma = \partial\Omega$. The non-linearity of the equations adds a significant challenge to the problem. There are no proofs that a strong solution to the Navier-Stokes equations exists in three dimensions [Fefferman, 2000]. However, the existence of a *weak* solution to the Navier-Stokes equations has been proved. The uniqueness of weak solutions to the problem is not known.

Above, the system of equations is stated with Dirichlet boundary conditions along the whole boundary. This is used when enforcing no-slip boundary conditions or when deciding the exact inflow at some part of the boundary. In the numerical experiments, we will also apply the so-called do-nothing boundary condition at parts of the boundary Γ_N. These are given by

$$(\nu\nabla u - pI)n = 0 \quad \text{on } (0,T)\times\Gamma_N. \tag{3.56}$$

This condition is also referred to as *natural* boundary conditions and is usually used for the outflow part of the domain boundary. When considering the weak

formulation of the problem, this condition cancels the terms on the part of the boundary Γ_N. Therefore, no boundary terms need to be assembled along that part of the boundary. Moreover, when the problem is posed using natural boundary conditions, the pressure will be uniquely defined when using the finite element method.

3.4.1 Linearisation of the convection term

For the numerical solution of the Navier-Stokes equations, the challenge of the non-linear convection term will in this presentation be restricted to the choice of discretisation technique. When not using some projection method [Guermond et al., 2006], we mainly have three options for linearising the system of equations.

The first method is using a semi-implicit discretisation of the convection term. In this case the term $(u^n \cdot \nabla)u^n$ is replaced by $(\tilde{u}^n \cdot \nabla)u^n$, where $\tilde{u}^n$ is a k-th order extrapolation of u^n in time, such that

$$u^n - \tilde{u}^n = o(\tau^k) \tag{3.57}$$

The extrapolation formulas up to order $k = 3$ are listed in the table below, and can be found by Taylor-expanding around $t = 0$.

k	$\tilde{u}^n$
1	u^{n-1}
2	$2u^{n-1} - u^{n-2}$
3	$3u^{n-1} - 3u^{n-2} + u^{n-3}$

To attain the correct order of convergence, the extrapolation order used must be combined with a BDF-method of the same order. To ease the implementation, note that the convective term can be rewritten using that

$$(u \cdot \nabla)v = (\nabla v)u = \nabla v u, \tag{3.58}$$

where ∇v denotes the Jacobian of the vector-valued function v. The semi-implicit convection term will be the linearisation method of choice for the numerical implementations in this thesis.

The second option is to handle the term as a full explicit term and include it on the right-hand side. In this case we use the extrapolation method from above, but now we replace the term $(u^n \cdot \nabla)u^n$ with $(\tilde{u}^n \cdot \nabla)\tilde{u}^n$. Therefore, this whole term can be moved over to the right-hand side of the equation and assembled in the right-hand side vector when using FEM.

A fully implicit method is a third alternative for handling the convection term. In this case, an iteration scheme is applied in each time step. First-order convergence is achieved by employing fixed-point iteration [Quarteroni et al., 2010, Ch. 6.3], while Newton-iteration [Tröltzsch, 2010, Ch. 4.11] results in a second-order method. Fixed-point iteration will be applied for solving the stationary Navier-Stokes equations and will be described in the next section.

3.4.2 CutFEM for Navier-Stokes

CutFEM for Stationary Navier-Stokes

The stationary Navier-Stokes equations are given by

$$\begin{aligned} (u \cdot \nabla)u - \nu\Delta u + \nabla p &= f \quad \text{in } \Omega \\ \nabla \cdot u &= 0 \quad \text{in } \Omega \\ u &= g \quad \text{on } \Gamma. \end{aligned} \tag{3.59}$$

Note that do-nothing boundary conditions can optionally be applied for this problem as well. As usual, the weak formulation is found by multiplying the first equation by $v \in V_h$, and the second with $q \in Q_h$. Then subtracting the latter from the former and integrating by parts.

This stationary problem will be solved using fixed-point iteration. In this approach, the solution of the previous iteration u_h^{k-1} is inserted as the convection field when solving for the next approximation (u_h^k, p_h^k). The iteration is stopped when the difference between two subsequent approximations measured in the L^2-norm is less than some desired tolerance level. The iteration is started with some initial approximation u_h^0. When setting $u_h^0 = 0$, the first iteration approximation (u_h^k, p_h^k) will be the solution to the stationary Stokes problem. The iteration goes as follows. As long as

$$\|u_h^k - u_h^{k-1}\|_\Omega < \delta, \tag{3.60}$$

solve the coupled linear variational problem

$$F(u_h, p_h; u_h^{k-1}) = f, \tag{3.61}$$

for $(u_h^k, p_h^k) \coloneqq (u_h, p_h)$, and increase the iteration index k by one.

For CutFEM, we again use the Taylor-Hood discrete function spaces V_h and Q_h defined on the whole fictitious domain, given in (3.48). The variational formulation is the following. Given the previous iteration step $u_h^{k-1} \in V_h$, find $(u_h, p_h) \in V_h \times Q_h$ such that

$$\begin{aligned} &(\nabla u_h \cdot u_h^{k-1}, v)_\Omega + a_h(u_h, v) + b_h(p_h, v) + b_h(q, u_h) \\ &\qquad + g_u(u_h, v) + g_p(p_h, q) = l_h(v, q), \qquad \forall (v, q) \in V_h \times Q_h. \end{aligned} \tag{3.62}$$

This formulation uses the bilinear- and linear forms as defined in (3.37) and (3.38). The ghost penalties are as defined in (3.49), with the ghost penalty scaling constants as in (3.51), identical to the values set for the stationary Stokes problem.

CutFEM for time-dependent Navier-Stokes

The CutFEM formulation of the time-dependent Navier-Stokes equations will naturally only differ from the time-dependent Stokes problem formulation by the

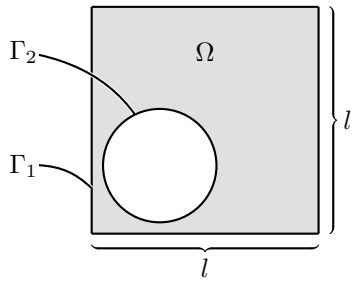


Figure 3.1: The domain used for the convergence tests for the Stokes equation and the Navier-Stokes equation.

convection term. As explained in Section 3.4.1, this term will be linearised using the semi-implicit extrapolation approximation.

Given $u_h^j \in V_h$ for $j = 0, \ldots, s-1$, find $(u_h^n, p_h^n) \in V_h \times Q_h$ such that

$$\begin{aligned} \frac{1}{\tau}\sum_{k=0}^{s} \alpha_k (u_h^{n-k}, v)_\Omega + (\nabla u_h^n \cdot \tilde{u}_h^n, v)_\Omega + a_h(u_h^n, v) + b_h(v, p_h^n) + b_h(u_h^n, q) \\ + g_u(u_h^n, v) + g_p(p_h^n, q) = l_h(v, q), \quad \forall (v, q) \in V_h \times Q_h, \end{aligned} \tag{3.63}$$

for each time step $n = s, s+1, \ldots, M$. The bilinear and linear forms above are still defined as in (3.37) and (3.38). The ghost penalties is unchanged from (3.49), and the ghost penalty scaling constants are now identical to the time dependent Stokes problem, given in (3.54).

3.5 Numerical experiments

In this section, we will investigate the convergence properties of the numerical methods presented in this chapter. This is done by using the method of manufactured solution, with a known non-polynomial analytical solution to the equations. The implementation is also tested by running two classic benchmark examples. The benchmark problems test physical convective flow around a cylinder. The first example is stationary, while the second one is time-dependent. These benchmark tests were first performed by [Schäfer et al., 1996] and have since become a standard way of assessing the numerical implementation of fluid solvers.

As elaborated in Section 2.3, all implementation was done using a CutFEM development branch [Simon Sticko, 2021] of the FEM library `deal.II` [Arndt et al., 2020].

3.5.1 Convergence tests

For the convergence tests the following analytical solution will be used,

$$\begin{aligned} u(x,t) &= \Big(-\cos(\pi x_1)\sin(\pi x_2)e^{-2\pi^2\nu t}, \sin(\pi x_1)\cos(\pi x_2)e^{-2\pi^2\nu t}\Big) \\ p(x,t) &= -\frac{1}{4}(\cos(2\pi x_1)+\cos(2\pi x_2))e^{-4\pi^2\nu t}. \end{aligned} \tag{3.64}$$

The above expression solves the Navier-Stokes equations in two dimensions for $f = 0$ [Ethier and Steinman, 1994]. The solution can thus be used as an analytical solution for the Stokes problem if we let $f = -(u \cdot \nabla)u$. Similarly, when used as an analytical solution to the stationary Navier-Stokes equations, we calculate the right-hand side as $f = -\partial_t u$. Note that the exponential term in the expressions causes a rapid decrease in the solution value as time increases. Therefore, to avoid the solution values vanishing during the convergence tests, the end time will be set to $T = 0.05$. The side of the two-dimensional domain is set, so we get $h = \tau$ for all tests.

The approximation error will be measured in the L^2L^2-norm and the L^2H^1-norms. These norms are defined as in (2.59) of Section 2.3.

The Taylor-Hood finite elements $[Q_p]^d \times Q_{p-1}$ for $p = 2$ are used throughout. Therefore, for a stationary problem one would expect the error converging as h^{p+1} for the L^2-norm of the velocity, and as h^p for the L^2-norm of the pressure and the H^1-norm of the velocity. This is the optimal order of convergence for the spatial discretisation. Moreover, a time dependent problem discretised with BDF-s, converges as τ^s. Thus, measured in the L^2L^2-norm and the L^2H^1-norm, we expect the error of the method to converge as

$$\begin{aligned} \|u - u_h\|_{L^2L^2} &\leqslant \mathcal{O}(\tau^s) + \mathcal{O}(h^{p+1}), \\ \|u - u_h\|_{L^2H^1} &\leqslant \mathcal{O}(\tau^s) + \mathcal{O}(h^{p}), \\ \|p - p_h\|_{L^2L^2} &\leqslant \mathcal{O}(\tau^s) + \mathcal{O}(h^{p}). \end{aligned} \tag{3.65}$$

As explained in Section 2.3, the convergence tests are performed by solving the problem of interest several times on a globally refined mesh. When using the analytical solution, the error is computed using the norms above, which again is used to compute the EOC.

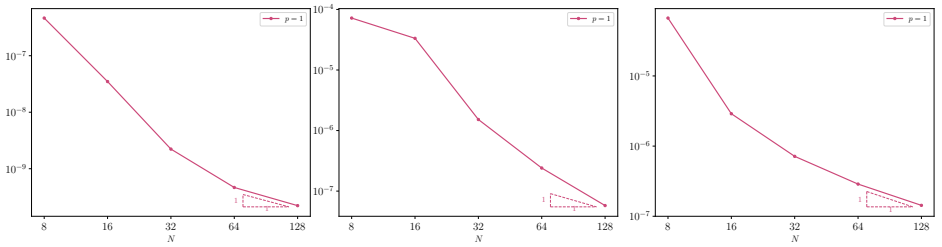
The fluid domain for the convergence tests is depicted in Figure 3.1. The background domain is a square with sides of length l, and the tesselation of this domain into a background mesh $\widetilde{\mathcal{T}}_h$ will consist of uniform rectangles. A sphere defined by a level set function floats on top of the background domain with center in

$$r(t) = (-0.9(l/2 - r_s), -0.9(l/2 - r_s)), \tag{3.66}$$

near the lower left corner. The sphere radius was set to $r_s = 3l/8$. The analytical solution (3.64) will be used for applying Dirichlet boundary conditions along the whole boundary $\partial\Omega = \Gamma = \Gamma_1 \cup \Gamma_2$.

Table 3.1: EOC for the BDF-2 method (with BDF-1 for the initial step), when the equation was solved for the given values of β_0. Taylor-Hood elements $[Q_2]^2 \times Q_1$.

Norm \ β_p	1.0	2.0	6.0	10.0	20.0	30.0	60.0	100.0
$\|u\|_{L^2L^2}$	2.92	3.1	3.41	3.52	3.62	3.69	3.79	3.85
$\|u\|_{L^2H^1}$	2.72	2.72	2.76	2.78	2.79	2.78	2.72	2.65
$\|p\|_{L^2L^2}$	1.9	2.07	2.17	2.23	2.73	3.08	3.4	3.34



(a) $\|u - u_h\|_{L^2L^2}$ **(b)** $\|u - u_h\|_{L^2H^1}$ **(c)** $\|p - p_h\|_{L^2L^2}$

Figure 3.2: Convergence plot for the Stokes equation, solved using BDF-1. The $[Q_2]^2 \times Q_1$ Taylor-Hood finite elements were used. The triangles has a slope corresponding to a convergence order of 1.

Time-dependent Stokes

All convergence tests were run with a relative high viscosity of $\nu = 0.1$. Also, the Nitsche penalty parameter was set to

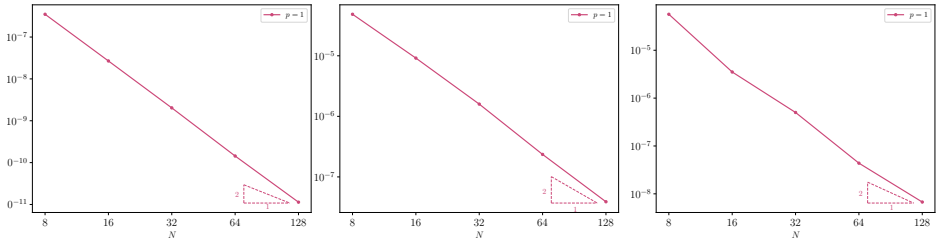
$$\mu = 5\nu p(p+1)/h. \tag{3.67}$$

As was observed in the numerical experiments for the heat equation in Section 2.3, the EOC for BDF-2 with BDF-1 for the initial step was highly influenced by the ghost penalty scaling constants. This was especially the case for the error of the pressure measured in the L^2L^2-norm. The first scaling constant was therefore set to the value used for the parabolic problems,

$$\beta_{u,0} = 8 \cdot 10^{p-1}. \tag{3.68}$$

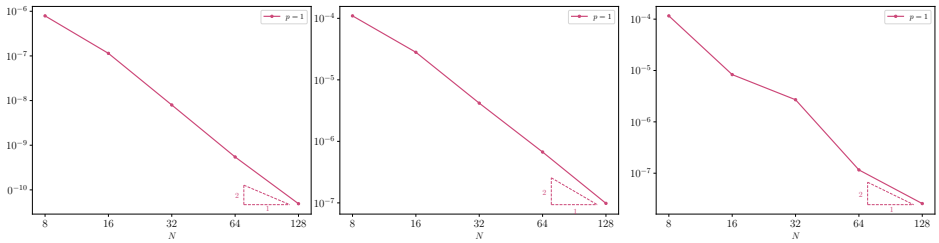
Then, a parameter search was performed for $\beta_{p,0}$, for $[Q_2]^2 \times Q_1$ elements. The results are listed in Table 3.1. The expected second order convergence was achieved in $\|p - p_h\|_{L^2L^2}$ for values of $\beta_{p,0} \geqslant 2$.

The time-dependent Stokes problem was run with the discrete weak formulation given in (3.53). BDF-1 and BDF-2 with interpolated initial steps are shown in Figure 3.2 and Figure 3.3 respectively. Both methods achieve optimal convergence. BDF-2 was also run with BDF-1 for the initial start-up step depicted in



(a) $\|u - u_h\|_{L^2L^2}$ **(b)** $\|u - u_h\|_{L^2H^1}$ **(c)** $\|p - p_h\|_{L^2L^2}$

Figure 3.3: Convergence plot for the Stokes equation, solved using BDF-2 with interpolated initial steps. The $[Q_2]^2 \times Q_1$ Taylor-Hood finite elements were used. The triangles has a slope corresponding to a convergence order of 2.



(a) $\|u - u_h\|_{L^2L^2}$ **(b)** $\|u - u_h\|_{L^2H^1}$ **(c)** $\|p - p_h\|_{L^2L^2}$

Figure 3.4: Convergence plot for the Stokes equation, solved using BDF-2 with BDF-1 for the initial step. The $[Q_2]^2 \times Q_1$ Taylor-Hood finite elements were used. The triangles has a slope corresponding to a convergence order of 2.

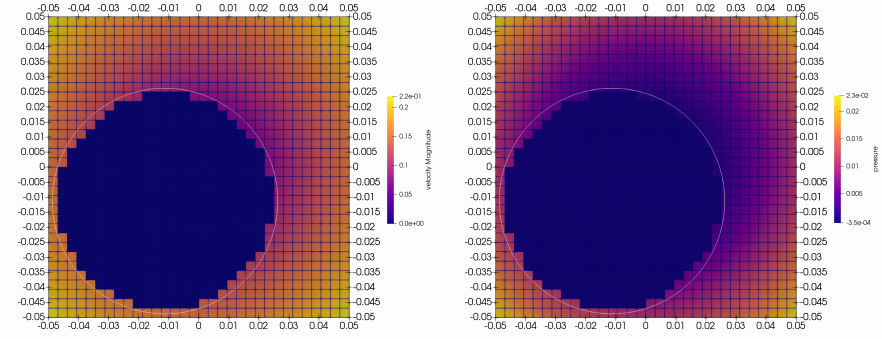
Figure 3.4. We see that the optimal second order convergence is achieved in this example also.

Stationary Navier-Stokes

As described in Section 3.4.2, the stationary Navier-Stokes equations was solved using fixed-point iteration. For these experiments, the iteration error tolerance was set to $\delta = 10^{-11}$. The problem was solved using $[Q_2]^2 \times Q_1$ Taylor-Hood finite elements. The solution of the problem on a grid with $N = 32$ cells along each side, is depicted in Figure 3.5. The error is plotted in Figure 3.6, and we see that optimal convergence is obtained in all norms.

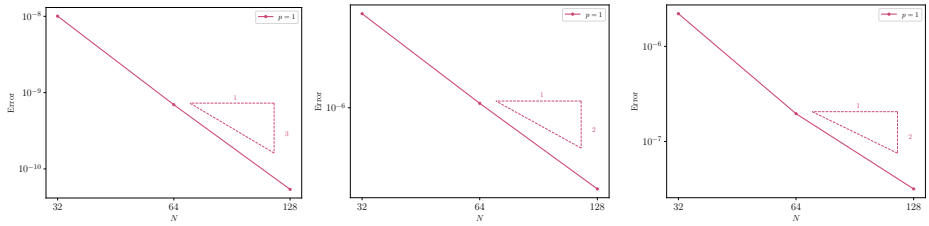
Time-dependent Naiver-Stokes

The time-dependent Navier-Stokes examples were run by using the discrete weak formulation in (3.63). The values for the Nitsche penalty constant μ and the ghost



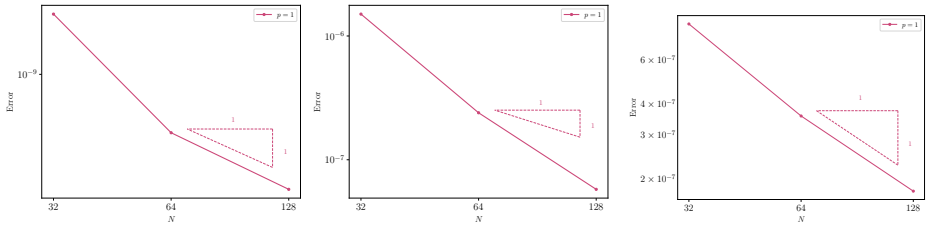
(a) Velocity magnitude. **(b)** Pressure.

Figure 3.5: Solution of the stationary Navier-Stokes equations.



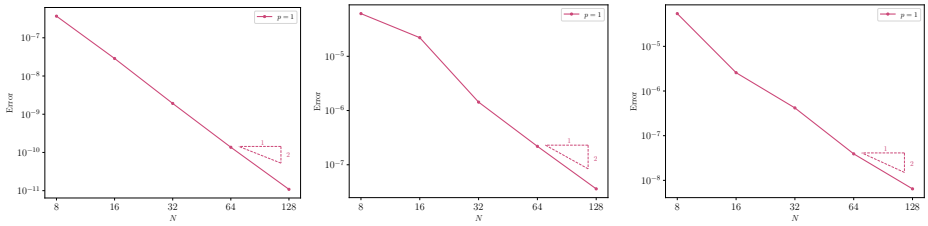
(a) $\|u - u_h\|_\Omega$ **(b)** $\|u - u_h\|_{1,\Omega}$ **(c)** $\|p - p_h\|_\Omega$

Figure 3.6: Convergence plot for the stationary Navier-Stokes equation, solved using fixed point iteration. Taylor-Hood finite elements $[Q_2]^2 \times Q_1$ was used.



(a) $\|u - u_h\|_{L^2L^2}$ **(b)** $\|u - u_h\|_{L^2H^1}$ **(c)** $\|p - p_h\|_{L^2L^2}$

Figure 3.7: Convergence plot for the Navier-Stokes equation, solved using BDF-1. The $[Q_2]^2 \times Q_1$ Taylor-Hood finite elements were used. The triangles has a slope corresponding to a convergence order of 1.



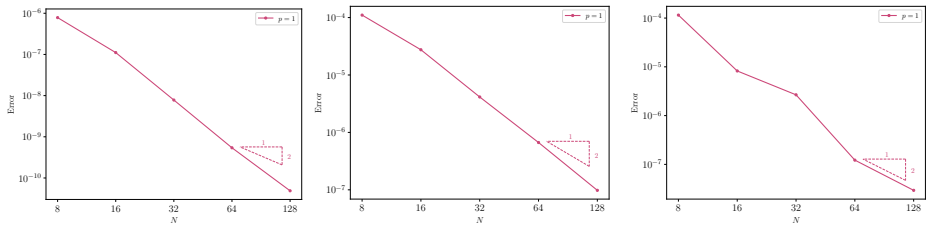
(a) $\|u - u_h\|_{L^2L^2}$ **(b)** $\|u - u_h\|_{L^2H^1}$ **(c)** $\|p - p_h\|_{L^2L^2}$

Figure 3.8: Convergence plot for the Navier-Stokes equation, solved using BDF-2 with interpolated initial steps. The $[Q_2]^2 \times Q_1$ Taylor-Hood finite elements were used. The triangles has a slope corresponding to a convergence order of 2.

penalty scaling parameters β_u and β_p was set to the same as for Stokes. BDF-1 and BDF-2 with interpolated initial conditions achieved optimal convergence and is shown in Figure 3.7 and Figure 3.8, respectively. Full second order convergence was also attained for BDF-2 with BDF-1 for the initial step, see Figure 3.9.

3.5.2 Fluid-rigid body benchmarks

Often it has been seen that numerical implementations of the same physical fluid problems do not produce the exact same results. Therefore, [Schäfer et al., 1996] presented some benchmark fluid problems that could be used for assessing the correctness of an implementation. Since these are problems without analytical solutions, values for the pressure drop and viscous forces are computed at certain points to compare different implementations. In this section, the benchmark tests '2D-1' and '2D-3' of [Schäfer et al., 1996] will be run. Both examples model 'fluid flow around a cylinder' in a rectangular channel. The first example uses the stationary Navier-Stokes equation as the model, while the second example is



(a) $\|u - u_h\|_{L^2L^2}$ **(b)** $\|u - u_h\|_{L^2H^1}$ **(c)** $\|p - p_h\|_{L^2L^2}$

Figure 3.9: Convergence plot for the Navier-Stokes equation, solved using BDF-2 with BDF-1 for the initial step. The $[Q_2]^2 \times Q_1$ Taylor-Hood finite elements were used. The triangles has a slope corresponding to a convergence order of 2.

time-dependent.

Benchmark test 2D-1

In this example, we will again solve the stationary Navier-Stokes equations, but this time for a more physical simulation case. Now, the domain will be a 2D rectangular channel with parabolic inflow and natural outflow. Near the channel entrance, a sphere with radius r_B is placed as an obstacle to the flow. The domain is depicted in Figure 3.10. In this section, the flow problem will be solved on the given domain, and then the pressure drop, lift and drag coefficients over the sphere is computed. This computed data is compared to the benchmark data for this problem available at [Turek et al., 1996a].

The strong form of the equation is the following

$$\begin{aligned} -\nu\Delta u + (u \cdot \nabla)u + \nabla p &= 0 \quad && \text{in } \Omega \setminus B \\ \nabla \cdot u &= 0 \quad && \text{in } \Omega \setminus B \\ u &= 0 \quad && \text{on } (\Gamma_0 \cup \partial B) \\ u &= g \quad && \text{on } \Gamma_I \\ (\nu\nabla u - pI)n &= 0 \quad && \text{on } \Gamma_N. \end{aligned} \tag{3.69}$$

The kinematic viscosity used was $\nu = 0.001$, and the inflow function was defined as

$$g(y) = \left(\hat{u}\left(1 - (y/r)^2\right), 0\right), \qquad \hat{u} = 0.3, \tag{3.70}$$

where $\hat{u}$ is the maximum speed of the inflow. For the implementation, the origin was placed in the center of the rectangular channel. The average inflow speed is

$$\bar{u} = \frac{2}{3}\hat{u} = 0.2. \tag{3.71}$$

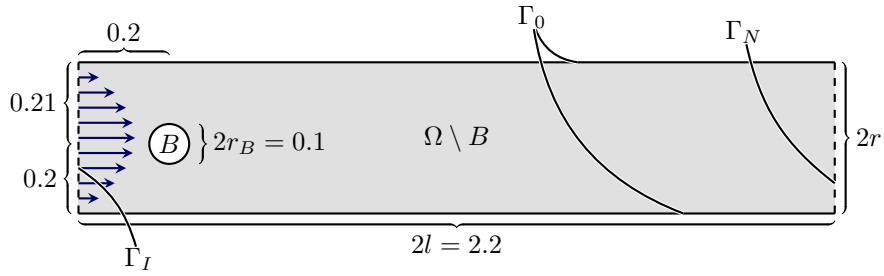


Figure 3.10: Domain for the two benchmarks tests for flow around a cylinder. Note, that the sphere is placed slightly off center along the cross section of the channel.

The Reynolds number for this problem is then

$$\mathrm{Re} = \frac{\bar{u}L}{\nu} = \frac{\bar{u} \cdot 2r_s}{\nu} = \frac{0.2 \cdot 2 \cdot 0.05}{0.001} = 20, \tag{3.72}$$

where L is the characteristic length of the flow configuration. The Reynolds number for this example is quite low, and hence the flow will be laminar.

The viscous forces over the sphere are decomposed into the drag force F_D, acting in the fluid flow direction, and lift F_L, acting perpendicular to the flow direction. These forces are computed by the following surface integral,

$$(F_D, F_L) = \int_{\partial B} \sigma_h n \, dS = \int_{\partial B} (\nu \nabla u_h - p_h I) n \, dS, \tag{3.73}$$

where n is the unit normal vector on the circle B pointing into the fluid domain. The drag and lift coefficients are then computed by

$$C_D = \frac{2}{\bar{u}^2 L} F_D, \qquad C_D = \frac{2}{\bar{u}^2 L} F_L. \tag{3.74}$$

The third flow parameter we compute, is the pressure drop over the sphere

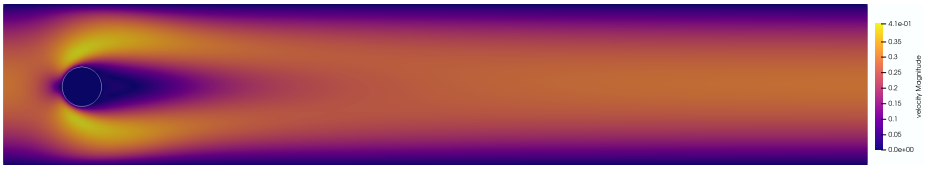
$$\Delta p = p_h(-l + 0.2 - r_B, -0.005) - p_h(-l + 0.2 + r_B, -0.005). \tag{3.75}$$

That is, the numerical pressure field solution is evaluated at the point exactly in front of the sphere, and the point exactly behind it.

As described in Section 3.4.2, the non-linear equations was solved using fixed-point iteration. The stopping criteria used was that the difference in the L^2-errors of two subsequent solutions in the iteration was less than $\delta = 10^{-12}$. For the simulations, a globally refined, uniform mesh was used. The number of elements along the short side of the rectangle is denoted by N, leading to an element size of $h = 2r/N = 0.41/N$. In the horizontal length direction of the channel, we have $4N$ cells. This results in the triangulation having $4N^2$ elements in total.

Table 3.2: The data for benchmark example 2D-1.

N	C_D	C_L	Δp
2^4	4.87689	0.180442	0.108485
2^5	5.38414	-0.0422371	0.120003
2^6	5.43033	-0.00173627	0.115903
2^7	5.55229	0.00990802	0.116985
2^8	5.57303	0.0106727	0.11737
Benchmark	5.57953523384	0.010618948146	0.11752016697



(a) Velocity magnitude.

(b) Pressure field.

Figure 3.11: Solution of the stationary Navier-Stokes equations for benchmark test 2D-1, using $[Q_2]^2 \times Q_1$ Taylor-Hood elements.

The problem was solved at five refinement levels, where for the finest, a grid with $N = 2^8$ elements on the short side of the domain was used. Taylor-Hood elements $[Q_2]^2 \times Q_1$ were used. The Nitsche penalty parameter was set to $\mu = 5p(p+1)/h$ and the ghost penalty parameter scaling constants used was $\beta_{u,0} = \beta_{p,0} = 1$.

The solution of the problem computed on a grid with $N = 2^8$, is depicted in Figure 3.11. The computed pressure drop and the lift and drag coefficients are listed in Table 3.2. From the data in the table, we observe that the computed values approach the benchmark data. Note that the computations have the wrong sign for the lift coefficients at refinement levels 5 and 6. Therefore, if we were to implement a two-way coupled fluid problem dependent on the lift coefficients, one would need a highly refined grid.

Compared to other finite element implementations listed in [Schäfer et al., 1996], this implementation needed a lot more elements for the similar approximations of the benchmark values in the table. One drawback with the current

implementation is the use of a uniform grid, which causes the same refinement level around the sphere as at the end of the channel. Most of the other examples utilise grids better fit for the example, where most of the refinement is focused around the sphere and near the channel's entrance. However, since the current implementation was done using a development version of `deal.II` [Arndt et al., 2020], supporting neither general Q1-mappings nor hanging nodes for the CutFEM stabilisations, a grid with more focused refinement could not be employed.

Benchmark tests 2D-3

This benchmark example is very similar to the previous one, but this example is time-dependent. Here, we implement the Navier-Stokes equations for simulating the flow around a cylinder in a rectangular channel. This is the benchmark tests '2D-3' from [Schäfer et al., 1996]. This benchmark example was also performed for a multimesh based projection method in [Dokken et al., 2020].

The domain is unchanged from the previous tests, depicted in Figure 3.10. Now the velocity $u : [0,T] \times \Omega \setminus B \to \mathbb{R}^2$ and the pressure $p : [0,T] \times \Omega \setminus B \to \mathbb{R}$ are time-dependent, and the strong form of the problems is

$$\begin{aligned} \partial_t u - \nu \Delta u + (u \cdot \nabla) u + \nabla p &= 0 && \text{in } (0,T) \times \Omega \setminus B \\ \nabla \cdot u &= 0 && \text{in } (0,T) \times \Omega \setminus B \\ u &= 0 && \text{on } [0,T] \times (\Gamma_0 \cup \partial B) \\ u &= g && \text{on } [0,T] \times \Gamma_I \\ (\nu \nabla u - pI) n &= 0 && \text{on } (0,T) \times \Gamma_N \\ u(0,x) &= u_0 && \text{in } \Omega, \end{aligned} \tag{3.76}$$

with $\nu = 0.001$.

The inflow profile is now defined as

$$g(t,x,y) = \left(\hat{u} \left(1 - (y/r)^2 \right), 0 \right), \qquad \hat{u} = 1.5 \sin(\pi t/8). \tag{3.77}$$

The inflow will build up to its maximum at $t = 4$, before it fades out at $T = 8$. At $t = 0$, we use $u_0 = (0,0)$ as an initial condition. The increased maximum speed leads to an average speed of

$$\bar{u} = \frac{2}{3} \hat{u} = 1. \tag{3.78}$$

In turn, this increases the Reynolds number for this problem to

$$\mathrm{Re} = \frac{\bar{u} L}{\nu} = \frac{\bar{u} \cdot 2 r_s}{\nu} = \frac{1 \cdot 2 \cdot 0.05}{0.001} = 100. \tag{3.79}$$

The CutFEM simulation was run on a uniform grid with $N = 128$ elements along the vertical side of the domain and with a time step of $\tau = 1/200$. The

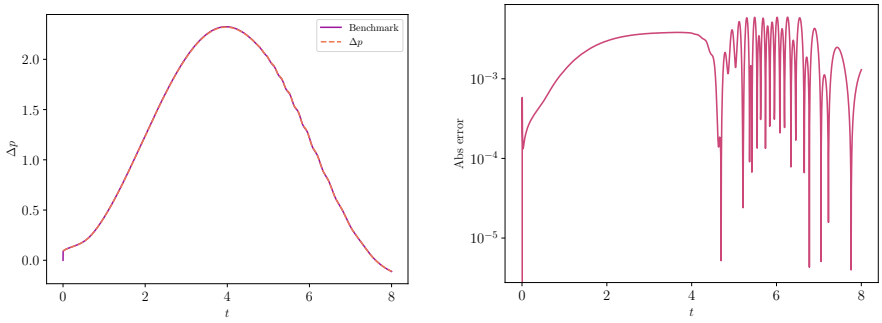
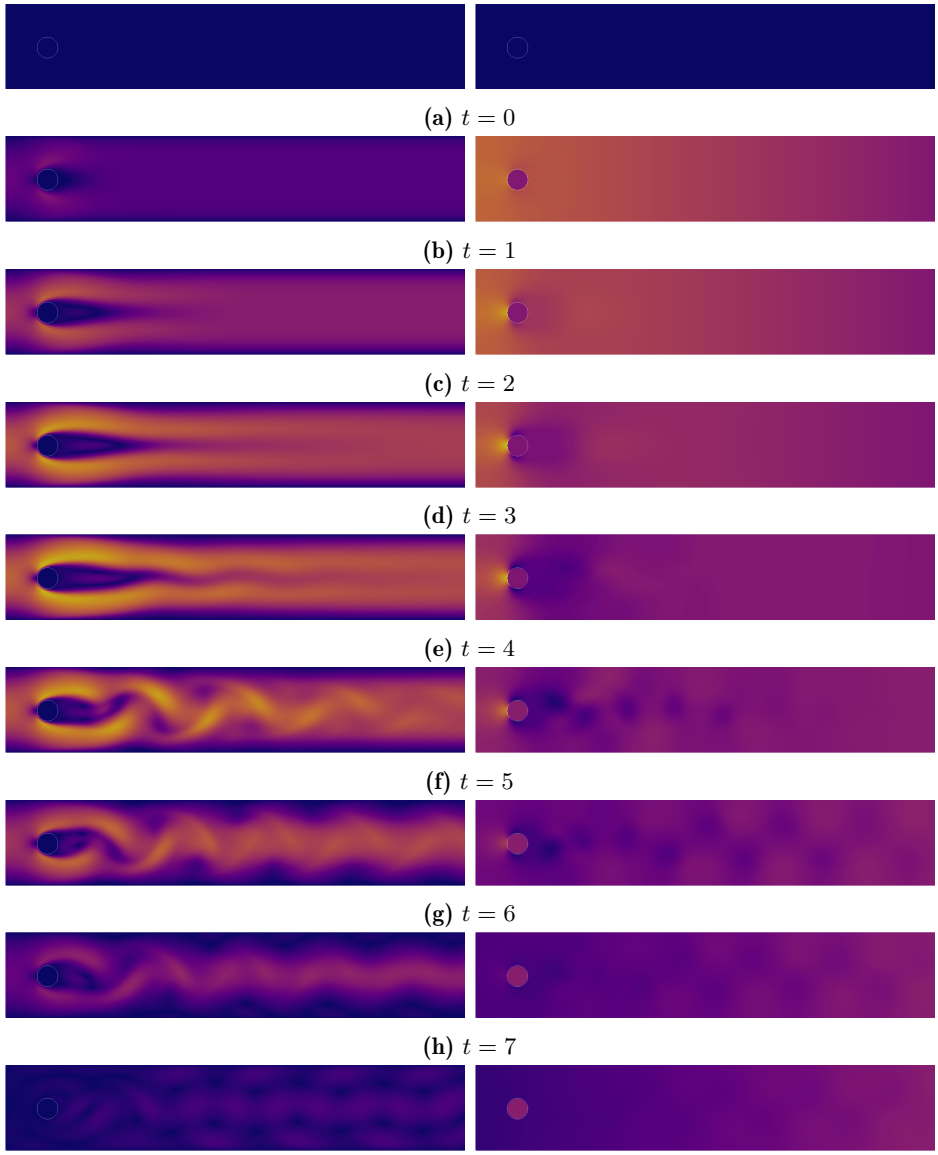


Figure 3.12: Computed values of the pressure difference Δp (dotted line) compared to the benchmark data. The computed values fit the benchmark data well according to the error plot on the right.

BDF-2 method and Taylor-Hood $[Q_2]^2 \times Q_1$ finite elements were used. The Nitsche penalty parameter μ and the ghost penalty scaling parameters $\beta_{u,0}$ and $\beta_{p,0}$ was set to the same values as in the example '2D-1' above.

The lift- and drag-coefficients were computed as in (3.74), and the pressure difference was computed as in (3.75). The benchmark data was downloaded from [Turek et al., 1996b], where a time-step of $\tau = 1/1600$ was used. Some snapshots from the numerical solution are shown in Figure 3.13.

The values for the pressure difference is shown in Figure 3.12. There we see that the computed values fit the benchmark data well. The drag- and lift coefficients could not be reproduced with respect to the benchmark data. This is most likely due to a bug in the drag- and lift computations since all convergence results gave correct convergence rates. The benchmark example was run successfully using CutFEM in both [Schott, 2016] and [von Wahl, 2021].



(a) $t = 0$

(b) $t = 1$

(c) $t = 2$

(d) $t = 3$

(e) $t = 4$

(f) $t = 5$

(g) $t = 6$

(h) $t = 7$

(i) $t = 8$

Figure 3.13: Results from the '2D-3' benchmark example. The velocity magnitude is plotted in the column to the left, and pressure field in right column.

Chapter 4

CutFEM for parabolic problems on moving domains

Even though there is a vast number of parabolic problems on stationary domains of interest, the possibility to solve time-dependent PDEs on moving domains unlocks a whole new branch of possible problems.

One application with moving interfaces is in two-phase flow problems [Barrett et al., 2020, Sec. 8], [Bänsch and Schmidt, 2020, Sec. 3]. In these problems, two fluids are separated by an interface, which evolves in time with the same velocity as the fluids. To avoid the complexities introduced if we were to move the mesh itself around with the same fluid velocity, an unfitted mesh method as CutFEM is an attractive alternative. A related application is found in biomechanics when modelling biomembranes [Barrett et al., 2020, Sec. 10]. In these problems, the evolving membrane interacts with surrounding fluids with possibly different properties. In many such models, the evolution of the interface is curvature driven, which mean curvature flow [Barrett et al., 2020, Sec. 4] is an example of. These coupled problems occur in a multitude of biological systems and are of great interest in the life sciences.

The phase-field method [Du and Feng, 2020] initially occurred as a method for describing an evolving interface between materials during some phase transition. Since an auxiliary phase function is used to describe the phases, and hence their separating interface, the method is closely related to other level set methods [Saye and Sethian, 2020]. However, in the phase method, a diffusive layer is introduced to smoothen the sharp interface of a pure level set method. The diffusive nature of the interface is modelled by either minimising interface energies, surface tensions or the curvature of the interface. The method can be applied to various applications, one of which is naturally multiphase fluid problems. The phase-field method has also been the subject of optimal control [Hintermüller and Keil, 2021], broadening the possible application problems even more.

Shape optimisation amounts to finding the optimal shape of some domain subject to PDE state equations as constraints and an objective function [Allaire, 2001]. The level-set methods have proved especially useful when tracking and controlling the motion of an evolving boundary [Allaire et al., 2021, Sec. 7]. The domain boundary at time t is represented by the zero contours of a level set function, that is, $\phi(t, x) = 0$. When this equation is differentiated with respect to time, a transport equation is obtained,

$$\phi_t + b \cdot \nabla \phi = 0, \tag{4.1}$$

where b is a convection field controlling the movement of the level set. This convection field is the control of the optimisation problem and is updated in every iteration through differentiation of the objective function with respect to the domain boundary. This is precisely the approach used when shape optimisation was studied using CutFEM in [Burman et al., 2018].

When using an unfitted method, a fully Eulerian description will be used for both the PDE and the domain. When the domain is moving from one step to the next, the function spaces will depend on the time-step and will therefore not be equal. This leads to a new challenge when solving parabolic problems. When using Rothe's method [Rothe, 1930] of discretising the equation first in time, then in space, subsequent solution steps u_h^n are used for the finite differences approximations of the time derivative. When the function spaces of two subsequent solution steps differ, this approximation is not well defined. See Figure 4.1a for a depiction of this challenge.

Several papers have presented approaches to deal with this issue. In [Hansbo et al., 2016], as space-time formulation using CutFEM was used in the discretisation of coupled bulk-surface problems on a time-dependent domain. To solve fluid problems on moving domain, [Schott, 2016] introduced a projection operator to extend a solution from the function space of time step t_{n-1} to the function space of the next step t_n, before approximating the time derivative. In the contribution of [Lehrenfeld and Olshanskii, 2019], this extension operator was effectively built into the weak formulation. There, the active mesh was extended sufficiently far past the physical boundary to make sure the finite difference approximations of the time derivative was well defined. This method will be presented in the following and was implemented for the numerical examples of this chapter.

In this chapter, we will start by defining what it means for a domain to be moving. As in Chapter 2, the heat equation will be used as the model problem. In Section 4.2 the challenges related to moving domains are discussed. The key takeaways of the CutFEM approach implemented by [Schott, 2016] is briefly explained before the method of [Lehrenfeld and Olshanskii, 2019] is presented in Section 4.3. This section also presents the discrete weak formulation of the heat equation on a moving domain. In the final Section 4.5, we present convergence tests for the implemented model problem, employing a BDF-method of stages $s = 1, 2$. The tests achieve the optimal order of convergence. Finally, a test

example shows the heat equation's solution on a moving domain, illustrating how the active mesh is extended past the physical boundary.

4.1 Moving domain

Let $\Omega(t) \subset \mathbb{R}^d$ be an open bounded domain, with smooth boundary $\Gamma(t)$. The time variable t indicates that the domain evolves with time. Let a material point in the domain be described by its position X in the reference domain $\Omega_0 = \Omega(0) \subset \mathbb{R}^d$ at the initial time (see e.g. [Eck et al., 2017, Ch. 5.4]). The movement of a material point is then described by the sufficiently smooth mapping

$$\Phi(t, \cdot) : \Omega_0 \to \Omega(t) \quad \text{such that } \Phi(t, X) = x(t, X). \tag{4.2}$$

Let the vector field $b : \Omega(t) \to \mathbb{R}^d$ define the material velocity of the particles in $\Omega(t)$. The mapping then satisfies the following ODE,

$$\begin{aligned} \frac{\partial}{\partial t}\Phi(t, X) &= b(t, \Phi(t, X)) \\ \Phi(0, X) &= X, \end{aligned} \tag{4.3}$$

for $t \in (0, T)$. We assume that b is sufficiently smooth such that the mapping Φ is a diffeomorphism. The deformation of the domain at time t is then given as

$$\Omega(t) = \{\Phi(t, X) : X \in \Omega_0\}. \tag{4.4}$$

The geometry of the domain is now defined by a level set function $\phi : [0, T] \times \mathbb{R}^d \to \mathbb{R}$ depending on time, such that

$$\begin{aligned} \Omega(t) &= \{x \in \mathbb{R}^d : \phi(t, x) < 0\}, \\ \Gamma(t) &= \{x \in \mathbb{R}^d : \phi(t, x) = 0\}. \end{aligned} \tag{4.5}$$

The heat equation defined on the moving domain $\Omega(t)$, has the strong form

$$\begin{aligned} \partial_t u - \nu \Delta u &= f \quad \text{in } \Omega(t), \\ u &= g \quad \text{on } \Gamma(t), \\ u(0, x) &= u_0 \quad \text{on } \Omega(0), \end{aligned} \tag{4.6}$$

for $t \in (0, T)$. The function u will be scalar valued, such that $u : \Omega \to \mathbb{R}$, and the diffusion coefficient $\nu > 0$ is a constant.

4.2 Computational domains

As in the previous chapters, we denote the background domain by $\widetilde{\Omega} \subset \mathbb{R}^d$, but now it has to be sufficiently large to cover the physical domain Ω over the whole

time interval, i.e. $\Omega(t) \subset \widetilde{\Omega}$ for all $t \in [0,T]$. On this background domain, a quasi-uniform background mesh $\widetilde{\mathcal{T}}_h$ is defined. This background mesh consists of shape-regular and closed quadrilaterals/hexahedral covering $\widetilde{\Omega}$. The time interval $[0,T]$ is divided in discrete steps of length τ, such that we have $[t_0, t_1, \ldots, t_M]$, where $t_n = n\tau$, and the end time $T = M\tau$. Since the domain evolves with time, denote the domain at time t_n as $\Omega^n = \Omega(t_n)$, and also let $\Gamma^n = \Gamma(t_n)$.

In the previous chapters describing CutFEM for stationary domains, the active mesh $\mathcal{T}_h$ was defined as all elements T in the background mesh intersecting the interior of the physical domain Ω. Since the domain is now moving, so will the active mesh. Therefore, we denote the active mesh at time t_n by $\mathcal{T}_h^n$. In the case of stationary domains, the active mesh was defined as the set of all elements intersecting the interior of the domain. If we now continue as before, we thus get

$$\mathcal{T}_h^n = \{T \in \mathcal{T}_h : T \cap \Omega^n \neq \emptyset\}. \tag{4.7}$$

As the union of these elements, the computational domain at time t_n is defined as

$$\Omega^{n,*} = \bigcup_{T \in \mathcal{T}_h^n} \overline{T}. \tag{4.8}$$

The set of elements in the active mesh cut by the boundary Γ is denoted by

$$\mathcal{T}_\Gamma^n = \{T \in \mathcal{T}_h^n : T \cap \Gamma \neq \emptyset\}, \tag{4.9}$$

We define the set of elements that does not lie completely in Ω^n as

$$\mathcal{T}_\Gamma^n = \{T \in \mathcal{T}_h^n : T \not\subset \Omega^n\}, \tag{4.10}$$

while the set of uncut elements is

$$\mathcal{T}_u^n = \mathcal{T}_h^n \setminus \mathcal{T}_\Gamma^n. \tag{4.11}$$

We define the set of ghost penalty faces as,

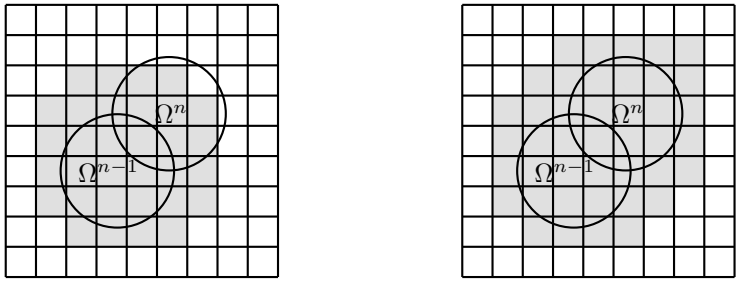
$$\mathcal{F}_h^{g,n} = \{F = T^+ \cap T^- : T^+ \in \mathcal{T}_h^n, T^- \in \mathcal{T}_\Gamma^n, T^+ \neq T^-\}. \tag{4.12}$$

The finite element function space is then defined on the whole fictitious domain $\Omega_h^{n,*}$,

$$V_h^n = \{v \in C(\Omega^{n,*}) : v|_T \in Q_p(T), T \in \mathcal{T}_h^n\} \cap H^1(\Omega^{n,*}), \tag{4.13}$$

where $Q_p(T)$ denotes a finite element of polynomial degree p on a quadrilateral (see [Brenner and Scott, 2008, Ch. 3]). Note, that because the domain is time dependent, expressed with the discrete index n, so is also the active mesh and fictitious domain, and thus also the finite element function space.

The above definition of the active mesh will lead to subsequent solutions $u_h^n \in V_h^n$ and $u_h^{n-1} \in V_h^{n-1}$ being defined on different meshes, and thus live in different



(a) $\Omega^n \not\subset \Omega^{*,n-1}$ **(b)** $\Omega^n \subset \Omega^{*,n-1}$

Figure 4.1: The gray area marks the active mesh $\mathcal{T}_h^{n-1}$, i.e. the fictitious domain $\Omega^{*,n-1} = \cup_{T \in \mathcal{T}_h^{n-1}} T$. On the left this active mesh is insufficient, since it does not cover the physical domain Ω^n in the next time step. This is not the case on the right, where the extended active mesh is sufficiently large for the BDF-terms in the discrete weak formulation to be well defined.

functions spaces V_h^n and V_h^{n-1} when the domain moves. Now, recall that when constructing a fully discretised weak formulation using the BDF-method, we will need to compute the difference $u_h^n - u_h^{n-1}$ for the first-order time discretisation. In general, this difference will neither lie in V_h^n nor V_h^{n-1}. In particular, since the domain is moving, the point x might be inside the fluid domain at time t_{n-1}, but on the outside at the next time step t_n. In such cases, the difference at this point $u_h^n(x) - u_h^{n-1}(x)$ might not be defined at all (see [Richter, 2017, Ch. 2.5]).

When sticking to the fully Eulerian description, it is clear from the above challenges that another definition of the active mesh is needed. A remedy to these challenges was presented in [Schott, 2016, Ch. 3.6.3]. To be able to compute the time derivative approximations, a projection was defined as

$$P^n : V_h^{n-1} \to V_h^n \quad \text{such that} \quad \tilde{u}_h^{n-1} = P^n u_h^{n-1}. \tag{4.14}$$

This projection was used to define an extension. After solving for solution u_h^{n-1} at a time-step, the solution was extended into the function space of the next time-step. The projection P^n was defined as

$$(P^n u_h^{n-1}, v)_{\mathcal{T}_\Gamma^{n-1}} + s_{\Gamma,h}^{n-1}(P^n u_h^{n-1}, v) = (u_h^{n-1}, v)_{\mathcal{T}_\Gamma^{n-1}} \quad \forall v \in V_h^{n-1}, \tag{4.15}$$

where $s_{\Gamma,h}^n$ here is a face jump stabilisation acting on the elements in $\mathcal{T}_\Gamma^n$. The projection then defines the function extension as

$$\mathcal{E}^n u_h^{n-1}|_T = \begin{cases} u_h^{n-1} & T \in \mathcal{T}_u^{n-1}, \\ P^n u_h^{n-1} & T \in \mathcal{T}_h^{n-1} \setminus \mathcal{T}_u^{n-1}. \end{cases} \tag{4.16}$$

To make sure the above projection problem is well defined, [Schott, 2016] imposes a CFL-condition (Courant-Friedrichs-Lewy condition) such that $b\tau \leqslant h$. This

condition assures the two sets of successive boundary elements are connected, such that the extension $\mathcal{E}u_h^{n-1}$ ends up being defined in all needed points. The method was successfully implemented for fluid problems but not analysed theoretically.

The method we will consider in this presentation was introduced in the paper by [Lehrenfeld and Olshanskii, 2019]. There, the time-dependent convection-diffusion equation was solved on a moving domain, using a completely Eulerian description. In that paper, the authors proved stability and a priori error estimates for the method based on the existence of continuous extension operators in Sobolev spaces. By extending the solution at a time step outside the domain boundary, finite difference approximations to the time derivative could be defined properly, see Figure 4.1. The extension to the solution could be approximated numerically by simply stabilising the unfitted weak formulation of the problem on an enlarged active mesh. This stabilisation was based on the ghost penalty [Burman, 2010], presented in Chapter 2.

4.3 Discretisation

Following the approach by [Lehrenfeld and Olshanskii, 2019] outlined above, we now extend the active mesh further outside the domain boundary. On this extended active mesh defining the computational domain, the numerical approximation of the solution extension will be defined when solving the fully discrete problem. Using the level set function ϕ, and the same approach as in [Sticko et al., 2021], we now define the active mesh at time t_n by

$$\mathcal{T}_h^n = \{T \in \mathcal{T}_h : \phi_h(t_n, x_T^c) < \delta\}. \tag{4.17}$$

With this extended active mesh (4.17), the corresponding fictitious domain $\Omega^{n,*}$ will also be enlarged, still defined as the union of all elements in the active mesh. The other sets introduced in the previous section in equation (4.8) to (4.12), still apply, but with the new definition of the active mesh defined above in (4.17).

Here, x_T^c denotes the centre of triangle T, and $\delta > 0$ is a constant defining how far outside the domain boundary the level set is extended. See a depiction of the extended active mesh and the ghost penalty faces in Figure 4.2. We assume the level set is close to a signed distance function, i.e. $\|\nabla\phi\|_2 \approx 1$. To make sure the active mesh includes at least all elements intersecting the physical domain $\Omega(t_n)$, we require that $\delta \geqslant c_\delta h = \sqrt{d}h/2$.

The need to extend the active mesh originated from the challenges of computing the BDF-terms for the time discretisation. Therefore, the active mesh should be extended so that these BDF-expressions are well defined when assembling the system matrix over the cells of the domain. Thus, the constant δ should also depend upon the stage s of the BDF-method, in addition to the maximum speed

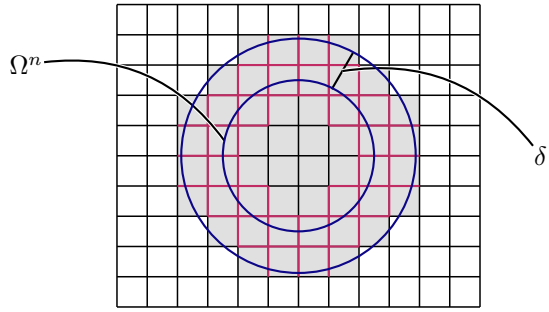


Figure 4.2: The extended active mesh, i.e. the fictitious domain $\Omega^{n,*}$ is the gray cells. The ghost penalty faces $\mathcal{F}_h^{g,n}$ are depicted in purple.

$\|b\|_\infty$ of the domain. Therefore, let

$$\delta = c_\delta h + s\|b\|_\infty \tau. \tag{4.18}$$

Remark. Note that this definition will extend the active mesh by the same amount along the whole boundary Γ. As an alternative, the constant δ could also depend on the speed of the domain at the point x_T^c. Also, if the domain boundary is moving with some speed to the right, there is no need to extend the mesh to the left, as long as the physical domain is sufficiently large concerning its speed and time step, i.e. that $s\|b\|_\infty \tau < \operatorname{diam}\Omega$.

The discrete function space will now include elements fully outside the physical domain, where the solution extension will be defined. The enlarged active mesh must satisfy the requirement that $\Omega^n \subset \Omega^{*,n-j}$ for $j = 0, \ldots, s$, to guarantee that the BDF-terms can be computed. This is depicted in Figure 4.1. The finite element function space is then defined on the whole fictious domain $\Omega_h^{n,*}$, as

$$V_h^n = \{v \in C(\Omega^{n,*}) : v|_T \in Q_p(T), T \in \mathcal{T}_h^n\} \cap H^1(\Omega^{n,*}). \tag{4.19}$$

4.3.1 Discrete weak formulation

The heat equation (4.6) is discretised in the same way as in Chapter 2. The Dirichlet boundary conditions are enforced using Nitsche's method, and the time derivative is discretised using a BDF-method of stage s. The unfitted fully discrete formulation of the problem is the following. Given $u_h^j \in V_h^j$ for $j = 0, \ldots, s-1$. Find $u_h^n \in V_h^n$ such that

$$\frac{1}{\tau}\sum_{j=0}^{s} \alpha_j (u_h^{n-j}, v)_{\Omega^n} + a_h(u_h^n, v) + \beta \cdot g_h^n(u_h^n, v) = l_h(v) \quad \forall v \in V_h^n, \tag{4.20}$$

for each time step $n = s, s+1, \dots, M$. In the formulation above, the bilinear form $a_h : V_h \times V_h \to \mathbb{R}$ and the linear form $l_h : V_h^n \to \mathbb{R}$ are defined as

$$\begin{aligned} a_h(u_h^n, v) &\coloneqq \nu(\nabla u_h^n, \nabla v)_{\Omega^n} - \nu(\partial_n u_h^n, v)_{\Gamma^n} - \nu(u_h^n, \partial_n v)_{\Gamma^n} + \mu(u_h^n, v)_{\Gamma^n} \\ l_h(v) &\coloneqq (f, v)_{\Omega^n} - \nu(g, \partial_n v)_{\Gamma^n} + \mu(g, v)_{\Gamma^n} \end{aligned} \tag{4.21}$$

The Nitsche penalty parameter is still $\mu = \gamma_0 \nu / h$, and β is the ghost penalty scaling parameter. Note that the inner products above are still evaluated over the physical domain Ω^n only, i.e. $\Omega^n \cap \mathcal{T}_h^n$. The stabilisation g_h^n (scaled with the parameter $\beta > 0$) however, is evaluated on bands around Ω^n, that is, on the faces in $\mathcal{F}_h^{g,n}$. Because the active mesh is enlarged, the solution at a time step u_h^{n-j} is extended past its physical domain Ω^{n-j}, and hence the inner products in the first term of (4.20) are well defined.

As in Chapter 2, the stabilisation bilinear form is here referred to as the ghost penalty. And just as in the previous chapters, the stabilisation g_h^n assures that the spatial discretisation of the variational problem (4.20) solved at each time step t_n is stable and optimally convergent when compared to classical FEM. Also, the condition number of the stiffness matrix scales with h^{-2}, again the same as classical FEM. In the setting of time-dependent domains, this stabilisation also enables the discrete weak formulation (4.20) to define a numerical realisation of the extension to the solution at the given time step [Lehrenfeld and Olshanskii, 2019].

As in Chapter 2, we will use the form of the ghost penalty penalising the jumps in the derivatives over facets, as used in [Burman et al., 2015, Burman and Hansbo, 2014]. We will here use the form of the ghost penalty presented and implemented in [Sticko et al., 2021], defined as

$$g_h^n(u, v) = \sum_{F \in \mathcal{F}_h^{g,n}} \sum_{k=1}^{p} \frac{1}{2k+1} \frac{h^{2k+1}}{k!^2} ([\partial_n^k u], [\partial_n^k v])_F. \tag{4.22}$$

In the case of the heat equation, the ghost penalty scaling constant β is set to

$$\beta = \beta_0 \left(\alpha_0 + \nu \tau / h^2 \right). \tag{4.23}$$

This corresponds to the parameter value used in [Lehrenfeld and Olshanskii, 2019], where the model problem also included a transport term.

The ghost penalty g_h^n scales as an L^2-norm. The first term in (4.23) stabilises the mass matrix, while the second term stabilises the stiffness matrix. The dependence on τ aries from that the discrete weak formulation (4.20) is multiplied with τ.

4.4 Theoretical results

As described in the above sections, the paper [Lehrenfeld and Olshanskii, 2019] introduced the ghost penalty stabilisation both for controlling the norms over the cut elements and for defining an implicit extension operator for the computed numerical solution. The analysis was based on the backward Euler (BDF-1) for time-stepping, but the changes needed to adapt the proofs for BDF-2 was outlined. The proofs also accounted for the geometrical error when interpolating the level set function onto a finite element function space. Stability for both the semidiscrete and the fully discrete method was proved. Also, in the a priori estimates in the energy norm, the optimal order of convergence was obtained. In the numerical experiments, a higher convergence rate was observed.

4.5 Numerical experiments

In this section, some numerical examples where the heat equation (4.6) is solved will be presented. For optimal order of convergence, we expect $\mathcal{O}(h^{p+1} + \tau^s)$ in the L^2L^2-norm, and $\mathcal{O}(h^p + \tau^s)$ in the L^2H^1-norm [Lehrenfeld and Olshanskii, 2019].

4.5.1 Convergence tests

To assess the convergence properties of the model problem (4.6), we perform convergence tests for BDF-1 and BDF-2 methods. We set $\nu = 2$ and employ the method of manufactured solutions, with

$$u(t, x_1, x_2) = \sin(\pi x_1) \sin(\pi x_2) e^{-t}. \tag{4.24}$$

The background domain is a square with sides $l = 2$, and the physical domain $\Omega(t)$ is a sphere with radius $r_s = 0.75$ (see Figure 4.3). The sphere moves across the domain, with the sphere centre following the linear path,

$$r_c(t) = (0.9(l/2 - r_s)(2t - 1), 0). \tag{4.25}$$

The level set function describing the domain is given by

$$\phi(t, x) = \|x - r_c(t)\|_2 - r_s. \tag{4.26}$$

Dirichlet boundary conditions are applied along the whole domain boundary $\Gamma(t)$. The end time was set to $T = 1$, and the regular grid was globally refined so that in each solution step, we have $h = \tau$.

The discrete weak problem in (4.20) was solved multiple times on an increasingly refined grid. For each convergence test, refinement levels from $r = 3$ to $r = 8$ was used, where the number of elements along one side was $N = 2^{r-1}$. The

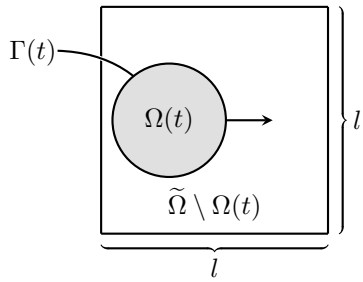


Figure 4.3: The domain used for the heat equation convergence tests.

error was computed in the L^2L^2- and L^2H^1-norms as defined in (2.59), while EOC was computed as in (2.61).

When performing initial convergence tests of the problem, it was observed that the ghost penalty scaling parameter β_0 in (4.23) had a significant effect on the EOC. This was especially the case for BDF-2 with BDF-1 for the initial time step. Therefore, to set this parameter to a reasonable value, the convergence test was performed for a range of values of β_0. The results of this study is depicted in Figure 4.4, with the corresponding data points listed in Table 4.1.

By investigating the plotted results, we observe that optimal convergence is approximately achieved for a value of β_0 set at e.g. 1.0 for Q_1-elements, and a value of 15 for Q_2-elements. These values are reasonable when compared to [Sticko et al., 2021], where a value of $\beta_0 = 10^{p-1}$ was used for the experiments. In the further experiments the ghost penalty scaling parameter will therefore be set to

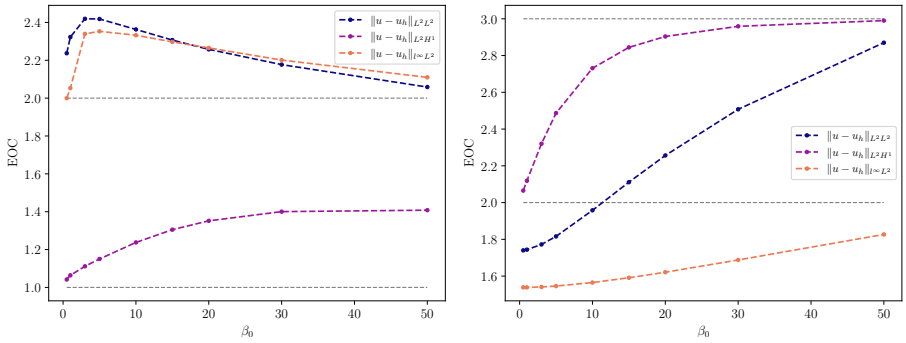
$$\beta_0 = 1.5 \cdot 10^{p-1}. \tag{4.27}$$

The Nitsche penalty parameter used was

$$\gamma = 20p(p+1). \tag{4.28}$$

The convergence plot for the BDF-1 method is shown in Figure 4.5. We observe that the method converges as expected for Q_1- and Q_2-elements. Recall that since BDF-1 is a first-order method, one can not expect that the EOC is larger than 1 in any of the norms.

In Figure 4.6 the convergence plot for BDF-2 with interpolated initial steps is shown. This method achieves optimal convergence in all norms. Because the first step is interpolated, and the interpolation operator has the same convergence rate as the CutFEM approximation, second-order convergence is also achieved in the $l^\infty L^2$-norm. This is not the case for BDF-2 when the initial step is solved using BDF-1 (see Figure 4.7). Because of the initial first order BDF-1 step, only first-order convergence is expected in the l^∞-norms. Optimal convergence is also attained for this method.



(a) Q_1-elements. **(b)** Q_2-elements.

Figure 4.4: EOC plotted as a function of the ghost penalty scaling constants β_0. BDF-2 with BDF-1 for the initial step was used in this case.

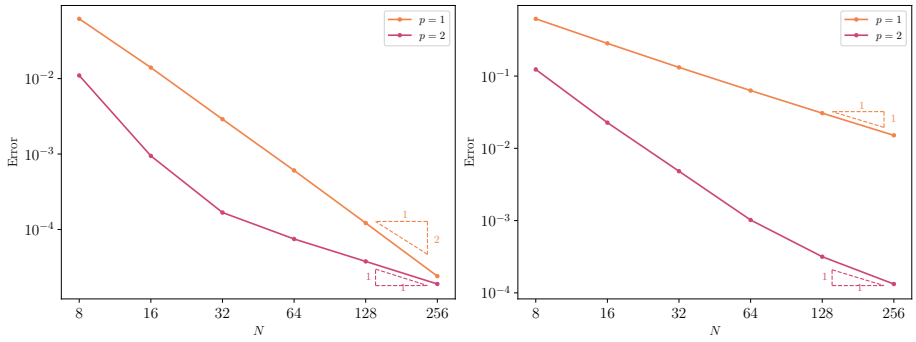
Table 4.1: EOC for the BDF-2 method (with BDF-1 for the initial step), when the equation was solved for the given values of β_0.

(a) Q_1-elements.

Norm \ β_0	0.5	1.0	3.0	5.0	10.0	15.0	20.0	30.0	50.0
$\|u-u_h\|_{L^2L^2}$	2.24	2.32	2.42	2.42	2.36	2.31	2.26	2.18	2.06
$\|u-u_h\|_{L^2H^1}$	1.04	1.06	1.11	1.15	1.24	1.31	1.35	1.4	1.41
$\|u-u_h\|_{l^\infty L^2}$	2.0	2.05	2.34	2.35	2.33	2.3	2.27	2.2	2.11

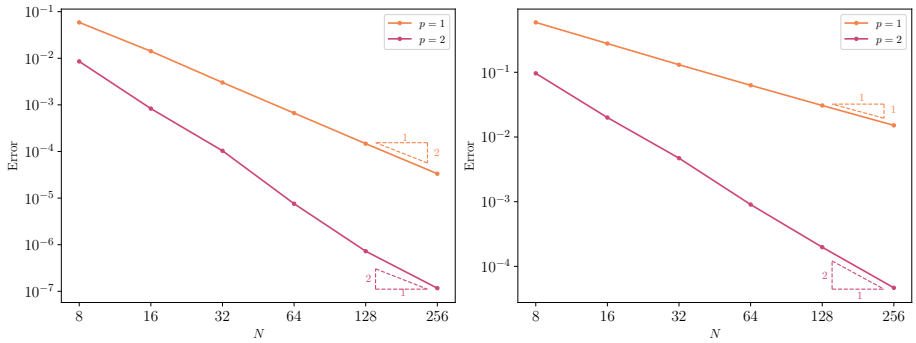
(b) Q_2-elements.

Norm \ β_0	0.5	1.0	3.0	5.0	10.0	15.0	20.0	30.0	50.0
$\|u-u_h\|_{L^2L^2}$	1.74	1.74	1.77	1.82	1.96	2.11	2.26	2.51	2.87
$\|u-u_h\|_{L^2H^1}$	2.07	2.12	2.32	2.49	2.73	2.84	2.9	2.96	2.99
$\|u-u_h\|_{l^\infty L^2}$	1.54	1.54	1.54	1.55	1.56	1.59	1.62	1.69	1.83



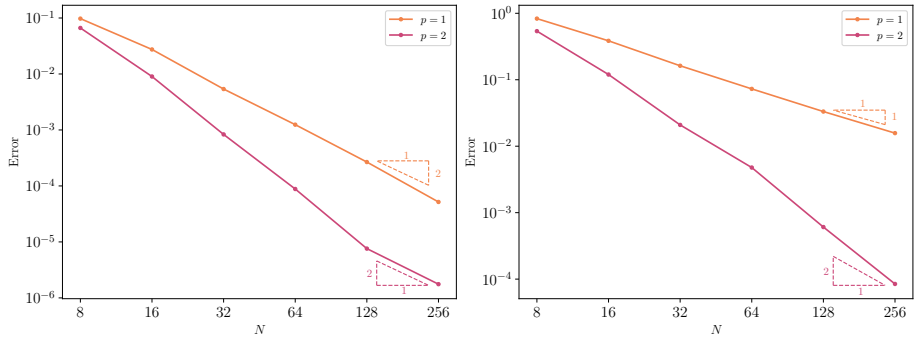
(a) $\|u - u_h\|_{L^2L^2}$ **(b)** $\|u - u_h\|_{L^2H^1}$

Figure 4.5: Heat equation on a moving domain solved using BDF-1.



(a) $\|u - u_h\|_{L^2L^2}$ **(b)** $\|u - u_h\|_{L^2H^1}$

Figure 4.6: Heat equation on a moving domain solved using BDF-2 with interpolated initial steps.



(a) $\|u - u_h\|_{L^2L^2}$ **(b)** $\|u - u_h\|_{L^2H^1}$

Figure 4.7: Heat equation on a moving domain solved using BDF-2 with BDF-1 for the initial step.

4.5.2 Moving domain example

In this example, the heat equation (4.6) is solved on a moving, flower-shaped domain.

The background domain now has length 2 and height 1, and the physical domain is in the shape of a flower, described by the level set function

$$\phi(t, x_1, x_2) = \|x - r_c(t)\|_2 - r + \frac{r}{r_0}\cos(5\,\mathrm{atan2}(x_2 - r_{c,2}(t), x_1 - r_{c,1}(t))). \quad (4.29)$$

We let the center of the flower follow the path

$$r_c(t) = (r_{c,1}, r_{c,2}) = (-\cos(\pi t/2), 0), \quad (4.30)$$

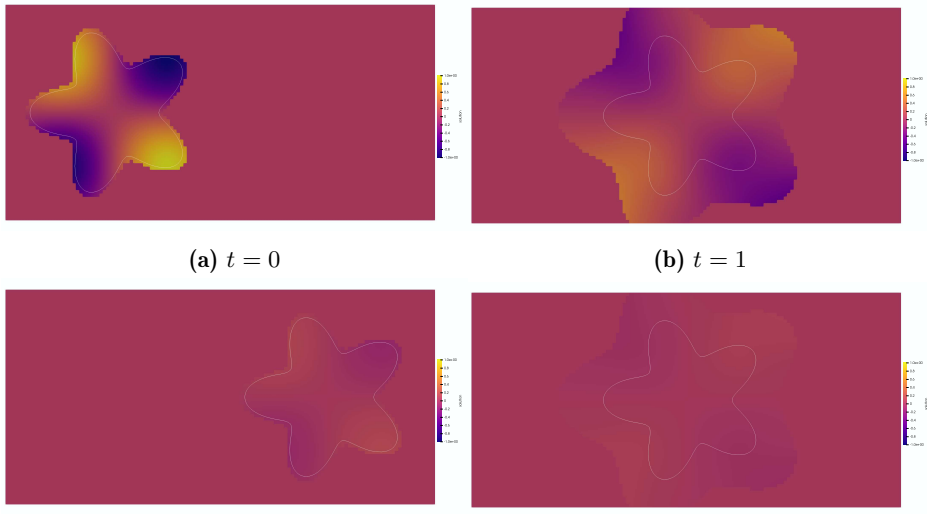
where the parameters are set to $r = 0.6$ and $r_0 = 3.5$. The former parameter describes the size of the flower, while the latter decides the curvature.

Here, the same fictional solution as in the convergence tests is used,

$$u(t, x_1, x_2) = \sin(\pi x_1)\sin(\pi x_2)e^{-t}. \quad (4.31)$$

Dirichlet boundary conditions are set along the whole domain boundary.

The problem was solved with BDF-2 (BDF-1 for the initial step) at refinement level 6, with $T = 6$, $\tau = 1/50$ and for $\nu = 2$. Some snapshots of the solution is shown in Figure 4.8. Note that since the speed of the domain centre now depends on the sin-function, the active mesh will vary in size, depending on the speed at each point. This is because the extended active mesh is defined based on the definition of δ given in (4.18). Also, recall that outside the physical domain, only the face stabilisations are added when assembling the stiffness matrix. This is because the problem (4.6) is only defined inside the domain $\Omega(t)$, and hence, we do not know the solution outside the level set zero contours. Therefore, the solution values outside this white line in the solution snapshots are only an extension.



(a) $t = 0$ **(b)** $t = 1$

(c) $t = 2$ **(d)** $t = 3$

Figure 4.8: Solution of the heat equation on a moving domain. The physical domain is marked by the zero contour level set line in white.

Chapter 5

CutFEM for flow problems on moving domains

Flow problems with moving interfaces have near unlimited applicability in science and engineering. Modelling such problems is essential for technological advances and for expanding our general understanding of the world. Fluid-structure interaction (FSI) [Richter, 2017, Bungartz et al., 2010] is the modelling of multiphysics problems studying how fluids interact with different materials or other fluids. Modelling the interplay of two fluids (two phase-flow) is the core of geometric free boundary problems [Bänsch and Schmidt, 2020]. When modelling the interface between fluids, surface tension effects can be a dominant force. An effective and flexible technique for representing geometry is therefore critical for such applications. FSI problems arise in many applications. One such application is the usage of mathematical modelling in the research of different cardiovascular diseases [Formaggia et al., 2009]. Blood interacts mechanically with the vessel walls and tissue and gives rise to many complex FSI multiphysics problems. Using CutFEM for solving such problems is a vast simplification because of the technique of representing the interface between fluid and structure using a level set function, floating freely atop the mesh.

In this chapter, we extend the approach presented for parabolic problems on moving domains in Chapter 4 to flow problems. This will enable us to solve flow problems on moving domains. As presented in the previous chapter, the main idea is to extend the active mesh further outside the physical domain. This is to enable assembling the time discretisation contributions in the stiffness matrix, even though solutions in subsequent time steps might live in different functions spaces. This method was first studied for the unfitted time-dependent Stokes problem on a moving domain in [Burman et al., 2022] and later in [von Wahl et al., 2021,von Wahl, 2021], where the geometrical error was also included in the analysis. These works are based on [Lehrenfeld and Olshanskii, 2019], where this

method was introduced for the scalar convection-diffusion problem on a moving domain.

In Section 5.1 we present the model problems used for the convergence test and the example problem. In the example, a purely Eulerian approach is used to model the fluid-rigid body movement by applying the kinematic coupling condition (see [Richter, 2017, Ch. 3.1]) along the boundary of a body with some prescribed motion. The fully discrete formulations of the Stokes problem and the Navier-Stokes equations on moving domains are presented in Section 5.2. Solvers for these problems were implemented, for which the convergence will be assessed for inf-sup stable $[Q_2]^d \times Q_1$ Taylor-Hood finite elements in Section 5.3. In the final example, the Navier-Stokes solver was used for simulating the flow around a moving sphere with known motion.

5.1 Model problems

In these examples, the flow around rigid sphere $B(t)$ with prescribed motion is modelled. The centre of the sphere follows some path $r_B(t)$ as it moves across a rectangle background domain $\widetilde{\Omega}$.

5.1.1 Convergence tests

The first model problem will be the time dependent Stokes problem, with the strong form

$$\begin{aligned} \partial_t u - \nu\Delta u + \nabla p &= f \quad \text{in } (0,T) \times (\widetilde{\Omega} \setminus B(t)), \\ \nabla \cdot u &= 0 \quad \text{in } (0,T) \times (\widetilde{\Omega} \setminus B(t)), \\ u(t,x) &= g \quad \text{on } [0,T] \times \Gamma(t), \\ u(0,x) &= u_0 \quad \text{in } \Omega(0). \end{aligned} \tag{5.1}$$

The fluid domain boundary is denoted by $\Gamma(t) = \partial B(t) \cup \partial\widetilde{\Omega}$.

The second model problem only differs from the first by the addition of a non-linear convection term, resulting in the incompressible Navier-Stokes equations

$$\begin{aligned} \partial_t u + (u \cdot \nabla)u - \nu\Delta u + \nabla p &= f \quad \text{in } (0,T) \times (\widetilde{\Omega} \setminus B(t)), \\ \nabla \cdot u &= 0 \quad \text{in } (0,T) \times (\widetilde{\Omega} \setminus B(t)), \\ u(t,x) &= g \quad \text{on } [0,T] \times \Gamma(t), \\ u(0,x) &= u_0 \quad \text{in } \Omega(0). \end{aligned} \tag{5.2}$$

The kinematic viscosity ν is a constant, while the velocity $u : [0,T] \times \Omega(t) \to \mathbb{R}^d$ and the pressure $p : [0,T] \times \Omega(t) \to \mathbb{R}$ are both time-dependent. For the convergence tests, an artificial flow analytical solution will be used, with Dirichlet boundary conditions applied on every part of the fluid boundary $\Gamma(t)$.

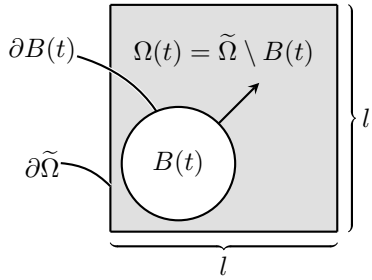


Figure 5.1: Domain used for the convergence tests for both the Stokes equation and the Navier-Stokes equation on a moving domain. The rigid circle $B(t)$ starts near the lower left corner at $t = 0$, and moves diagonally across the square with constant speed.

The problems are solved on identical domains, shown in Figure 5.1. The background mesh $\widetilde{\Omega}$ is a square with side lengths l. The sphere $B(t)$ has radius r_s, and moves diagonally across the background domain with a constant velocity. The fluid domain is then denoted by $\Omega(t) = \widetilde{\Omega} \setminus B(t)$. The sphere follows the linear path

$$r_B(t) = (0.9(l/2 - r_s)(2t/T - 1), 0.9(l/2 - r_s)(2t/T - 1)). \tag{5.3}$$

5.1.2 Moving rigid sphere

In this final example, we want to run a one-way coupled fluid problem using the implemented Navier-Stokes solver. Here, we will simulate the 2D fluid flow surrounding a moving rigid sphere in a channel (see Figure 5.2). The sphere moves along a known path. The problem has the strong form given by

$$\begin{aligned}
\partial_t u + (u \cdot \nabla)u - \nu\Delta u + \nabla p &= 0 && \text{in } (0,T) \times (\widetilde{\Omega} \setminus B(t)),\\
\nabla \cdot u &= 0 && \text{in } (0,T) \times (\widetilde{\Omega} \setminus B(t)),\\
u(t,x) &= 0 && \text{on } [0,T] \times \Gamma_0,\\
u(t,x) &= v_B && \text{on } [0,T] \times \partial B(t),\\
(\nu\nabla u - pI)n &= 0 && \text{on } (0,T) \times \Gamma_N,\\
u(0,x) &= 0 && \text{in } \widetilde{\Omega} \setminus B(0).
\end{aligned} \tag{5.4}$$

The sphere is initially placed on the left side of the channel, with no-slip boundary conditions everywhere for the fluid. Therefore, the boundary conditions is set to 0 along the walls of the domain, and equal to the speed of the sphere $v_B(t)$ along the boundary of the sphere $B(t)$. On the right side of the channel, natural

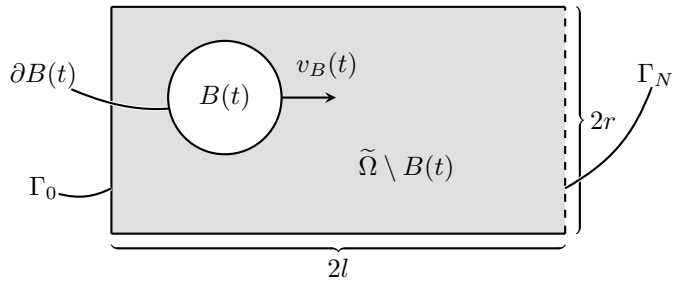


Figure 5.2: Domain used for the example where a rigid sphere moves through a channel.

boundary conditions is applied. The center of the sphere is set to follow the path

$$\begin{aligned} r_B(t) &= \left(-0.9(l - r_s) \cos\left(\frac{\pi t}{8}\right), \frac{r}{5} \right), \\ v_B(t) &= \frac{d}{dt} r_B(t) = \left(\frac{0.9\pi}{8}(l - r_s) \sin(\pi t/8), 0 \right). \end{aligned} \tag{5.5}$$

5.2 Methology

As described in the previous Chapter 4, the moving domain poses some new challenges with regards to how the discretisation of the domain. We define a background domain $\widetilde{\Omega}$ in such a way that $\Omega(t) \subset \widetilde{\Omega}$ for every $t \in [0, T]$. This background domain is discretised into a background mesh $\widetilde{\mathcal{T}}_h$. Here, we use quadrilateral elements for the triangulation. The time interval is discretised in steps of constant length τ, such that $t_n = n\tau$, and the final time step is $M = T/\tau$. The domain at time t_n is denoted by $\Omega(t_n) = \Omega^n$.

Our computational domain, the extended active mesh $\mathcal{T}_h^n$ at time t_n, is defined sufficiently large to assure the BDF-terms can be computed as the domains moves. The union of all elements T in the active mesh make up the fictitious domain $\Omega^{*,n}$ at the corresponding time step, defined as

$$\Omega^{*,n} = \bigcup_{T \in \mathcal{T}_h^n} \bar{T}. \tag{5.6}$$

The active mesh is defined by the level set function $\phi : [0, T] \times \mathbb{R}^d$ as

$$\mathcal{T}_h^n = \{T \in \widetilde{\mathcal{T}}_h : \phi(t_n, x_T^c) < \delta\}, \tag{5.7}$$

where x_T^c is the center of element T. The constant $\delta > 0$ describes how far the solution is extended outside the physical domain. Here, δ is chosen as

$$\delta = c_\delta(s v_\phi \tau + h). \tag{5.8}$$

As in [Sticko et al., 2021], we set the constant $c_\delta = 1.5$. This is to ensure we include all intersected elements when the domain is stationary. In all examples, the domain moves with constant speed v_ϕ by letting the centre of the level set follow some path r_ϕ. Also, recall that s denotes the stage of the BDF-method, and τ is the time step length.

As we saw in Section 3.1.2 when solving fluid problems, the function space pairing needs to fulfil the inf-sup condition to be stable. Therefore, the inf-sup stable function pair of Taylor-Hood elements was introduced (see [John et al., 2016, Ch. 3.6]). In the case of stationary domains, the same active mesh was used through all time steps. Therefore, at time step t_n, we searched for a numerical solution $(u_h^n, p_h^n) \in V_h \times Q_h$, where the discrete function spaces used, was the Taylor-Hood elements $[Q_p]^d \times Q_{p-1}$,

$$\begin{aligned} V_h &= \{v \in [C(\Omega^*)]^d : v|_T \in [Q_p(T)]^d, T \in \mathcal{T}_h\} \cap [H^1(\Omega^*)]^d \\ Q_h &= \{q \in C(\Omega^*) : q|_T \in Q_{p-1}(T), T \in \mathcal{T}_h\} \cap L_0^2(\Omega^*). \end{aligned} \tag{5.9}$$

As before, $Q_p(T)$ denotes a finite element of polynomial degree p on a quadrilateral T (see [Brenner and Scott, 2008, Ch. 3]), and the exponent d in $[Q_p(T)]^d$ denotes the same, but for a vector function on $\mathbb{R}^d$. Note that the function spaces are the same for every time step.

For moving domain problems, the active mesh $\mathcal{T}_h^n$ depends on the time step t_n, which implies that we need separate function spaces for every time step. The function spaces used at time step t_n are now defined on the fictitious domain $\Omega^{*,n}$ at that time step. Therefore, define

$$\begin{aligned} V_h^n &= \{v \in [C(\Omega^{*,n})]^d : v|_T \in [Q_p(T)]^d, T \in \mathcal{T}_h^n\} \cap [H^1(\Omega^{*,n})]^d \\ Q_h^n &= \{q \in C(\Omega^{*,n}) : q|_T \in Q_{p-1}(T), T \in \mathcal{T}_h^n\} \cap L_0^2(\Omega^{*,n}). \end{aligned} \tag{5.10}$$

The difference from (5.9) is that the active mesh and fictitious domain the function spaces are defined on, now depends on the time step t_n.

As described in Section 2.1.3, the BDF-method is used for the time-discretisation. Also, the Nitsche-terms and the ghost penalties added to the weak forms in Chapter 3 on time-dependent flow problems on stationary domains will be identical in this chapter. We can now define the discrete weak CutFEM formulation of the Stokes equations (5.1) and the Navier-Stokes equations (5.2).

5.2.1 The Stokes problem

To solve the Stokes equations (5.1) on a moving domain, the CutFEM discretisation with a BDF method of order s is stated as follows. Given previous solutions $(u_h^{n-j}, p_h^{n-j}) \in V_h^{n-j} \times Q_h^{n-j}$ for $j = 1, \dots, s-1$, find $(u_h^n, p_h^n) \in V_h^n \times Q_h^n$ such

that

$$\begin{aligned}\frac{1}{\tau}\sum_{j=0}^{s}\alpha_j(u_h^{n-j},v)_{\Omega^n}+a_h(u_h^n,v)+b_h(v,p_h)+b_h(u_h^n,q)\\ +\beta_u\cdot g_u^n(u_h^n,v)+\beta_p\cdot g_p^n(p_h^n,q)=l_h(v,q)\qquad\forall(v,q)\in V_h^n\times Q_h^n.\end{aligned}\tag{5.11}$$

for $n=s,\dots,M$. The last time step is $M=T/\tau$. The bilinear forms $a_h : V\times Q\to\mathbb{R}$ and $b_h : V\times V\to\mathbb{R}$ are defined as

$$\begin{aligned}a_h(u_h^n,v)&\coloneqq\nu(\nabla u,\nabla v)_{\Omega^n}-\nu(\partial_n u,v)_{\Gamma^n}-\nu(u,\partial_n v)_{\Gamma^n}+\mu(u,v)_{\Gamma^n}\\ b_h(v,p_h^n)&\coloneqq-(\nabla\cdot v,p)_{\Omega^n}+(v\cdot n,p)_{\Gamma^n}.\end{aligned}\tag{5.12}$$

The corresponding linear form $l_h : V\times Q\to\mathbb{R}$ is

$$l_h(v,q)=(f,v)_{\Omega^n}+\mu(g,v)_{\Gamma^n}-\nu(g,\partial_n v)_{\Gamma^n}+(g\cdot n,q)_{\Gamma^n}.\tag{5.13}$$

As in earlier chapters, the Nitsche penalty term depends on the mesh size h as

$$\mu=\frac{\nu\gamma_0}{h}.\tag{5.14}$$

The ghost penalties used here is the face wise jumpy penalty as defined in Chapter 3 in (3.49), and the same scaling parameters as in (3.54) for time-dependent Stokes on a stationary domain,

$$\begin{aligned}\beta_u&=\beta_{u,0}\left(1+\frac{\nu\tau}{h^2}\right),\\ \beta_p&=-\beta_{p,0}\frac{\tau}{\nu+h^2/\tau}.\end{aligned}\tag{5.15}$$

5.2.2 The Navier-Stokes equations

For solving the Navier-Stokes equations (5.2) on a moving domain, using a BDF-method of stage s, the cut finite element discretisation is given as: Given previous solutions $(u_h^{n-j},p_h^{n-j})\in V_h^{n-j}\times Q_h^{n-j}$ for $j=1,\dots,s-1$, find $(u_h^n,p_h^n)\in V_h^n\times Q_h^n$ such that

$$\begin{aligned}\frac{1}{\tau}\sum_{j=0}^{s}\alpha_j(u_h^{n-j},v)_{\Omega^n}+a_h(u_h^n,v)+c_h(u_h,v;\tilde{u}_h)+b_h(v,p)+b_h(u_h^n,q)\\ +\beta_u\cdot g_u^n(u_h^n,v)+\beta_p\cdot g_u^n(p_h^n,q)=l_h(v,q)\qquad\forall(v,q)\in V_h^n\times Q_h^n.\end{aligned}\tag{5.16}$$

for $n=s,\dots,M$. In the above expression, the bilinear form c_h is the semi-implicit linearisation of the convective term, and is defined as

$$c_h(u_h^n,v;\tilde{u}_h^n)=(\nabla u_h^n\tilde{u}_h^n,v)_{\Omega^n}.\tag{5.17}$$

Recall from Section 3.4.1, that $\tilde{u}_h^n$ is the extrapolation of order s based on the previous solutions. The bilinear forms a_h and b_h, and the linear form l_h is the same as for the Stokes problem above, defined in (5.12) and (5.13). However, the ghost penalty scaling parameters differs from the Stokes problem, and are the same as for Navier-Stokes on a stationary domain in Section 3.4.2:

$$\begin{aligned} \beta_u &= \beta_{u,0} \left(1 + \tau/h + \frac{\nu\tau}{h^2}\right), \\ \beta_p &= -\beta_{p,0} \frac{\tau}{\nu + h + h^2/\tau}. \end{aligned} \tag{5.18}$$

5.3 Numerical experiments

In the first examples, we will numerically assess the convergence properties of CutFEM for time-dependent flow problems on moving domains by solving the Stokes problem and the Navier-Stokes equations. This is done by solving the equations for some known artificial solution, and measuring the error in the L^2L^2- and L^2H^1-norms (2.59) and then computing the EOC (2.61).

For the final example, the Navier-Stokes solver will be used to simulate the fluid flow around a moving sphere in a channel in two dimensions. In this example, no-slip boundary conditions will be applied along the sphere boundary and solid walls to achieve a physical fluid simulation.

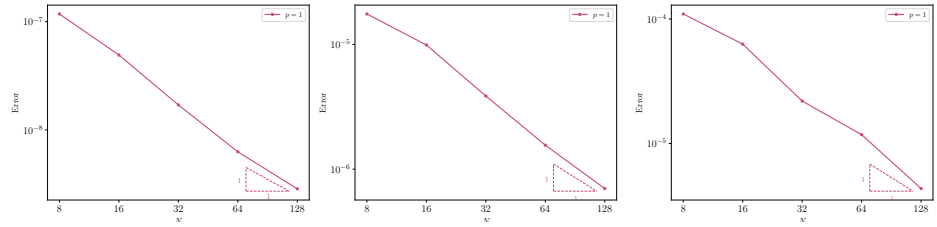
5.3.1 Convergence tests

The convergence test for the Stokes problem (5.1) and for Navier-Stokes equation (5.2), was run with the domain as depicted in Figure 5.1. This is a sphere moving across a square background domain in two dimensions, with the path as given in (5.3). The kinematic viscosity used was $\nu = 0.1$, and the end time is set to $T = 0.05$. This is to avoid that the known solution decreases too much, due to the negative exponential time dependence. The side lengths of the square domain is set to 0.1, and for each iteration of the convergence step, the time-step and grid size is refined to be equal, such that $h = \tau$. The level set describing the physical domain is set to

$$\phi(t, x) = -\|x - r_c(t)\|_2 + r_s. \tag{5.19}$$

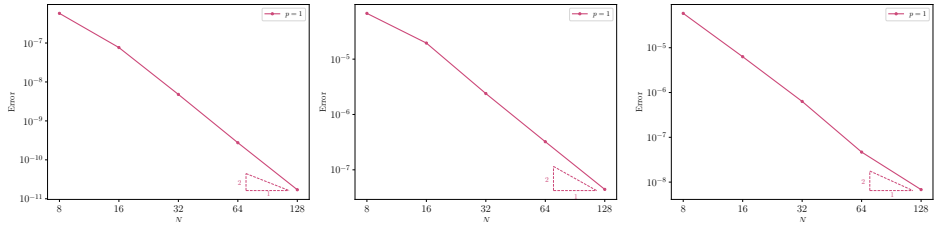
The discrete weak formulation of the time-dependent Stokes problem and the Navier-Stokes equations, were presented in Section 5.2.1 and Section 5.2.2, respectively. In both solvers, the CutFEM stabilisation parameters were set to $\beta_{u,0} = \beta_{p,0} = 1$ were set, while the Nitsche penalty parameter was set to $\mu = 5\nu p(p+1)/h$.

For each convergence test, the method of manufactured solutions is applied with the same analytical solution as given in Section 3.5 in equation (3.64). This artificial solution is used to apply Dirichlet boundary conditions along the whole



(a) $\|u - u_h\|_{L^2L^2}$ **(b)** $\|u - u_h\|_{L^2H^1}$ **(c)** $\|p - p_h\|_{L^2L^2}$

Figure 5.3: Convergence plot for the Stokes equation, solved using BDF-1 for the time discretisation. The Taylor-Hood finite elements $[Q_2]^2 \times Q_1$ were used.



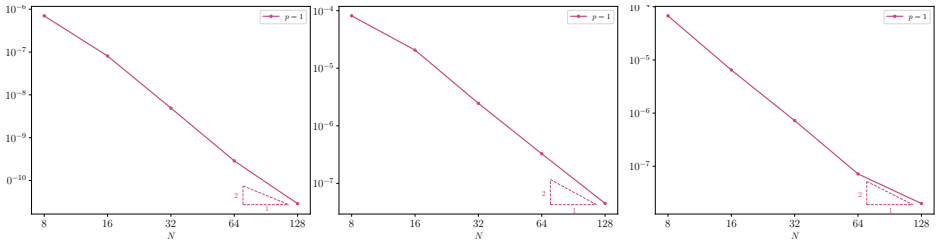
(a) $\|u - u_h\|_{L^2L^2}$ **(b)** $\|u - u_h\|_{L^2H^1}$ **(c)** $\|p - p_h\|_{L^2L^2}$

Figure 5.4: Convergence plot for the Stokes equation, solved using BDF-2 interpolated initial steps. The $[Q_2]^2 \times Q_1$ Taylor-Hood finite elements were used.

boundary $\Gamma(t)$. The problem is then solved several times on an increasingly globally refined regular grid. Here, we use refinement levels $r = 3$ through $r = 7$, while the number of elements along one side of the domain is $N = 2^{r-1}$. Due to some unresolved convergence problems when the code was run with $[Q_3]^2 \times Q_2$ Taylor-Hood finite elements, only the results with $[Q_2]^2 \times Q_1$ elements are included.

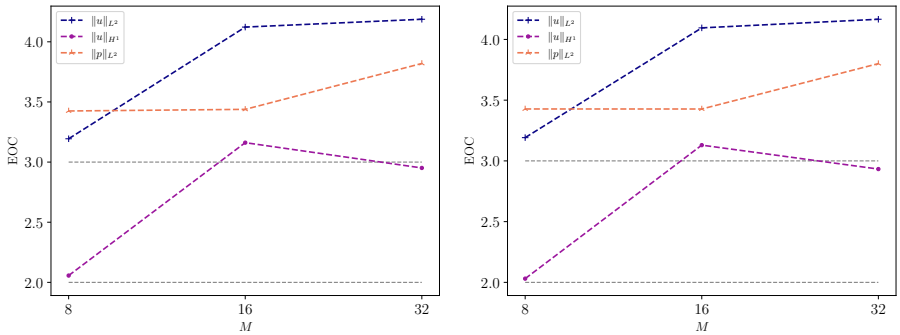
For the Stokes problem, the convergence plot for the BDF-1 method is shown in Figure 5.3, where we see optimal convergence is attained. Optimal convergence is also achieved when using BDF-2 with interpolated initial steps. However, for BDF-2 with BDF-1 for the first step, a small drop in the EOC is observed in the L^2L^2-norm of the pressure. Investigations showed that the BDF-1 step resulted in a spike in error. When these initial steps were dropped from the error computations, we see from Figure 5.6a that optimal convergence is achieved in all norms. The aim of the implemented solvers is to solve problems of a certain time horizon T. Therefore, a lower convergence order in the initial steps can be accepted.

For the solutions to the Navier-Stokes equations, the observations are similar to the Stokes problem above. Optimal convergence of the BDF-1 method can be



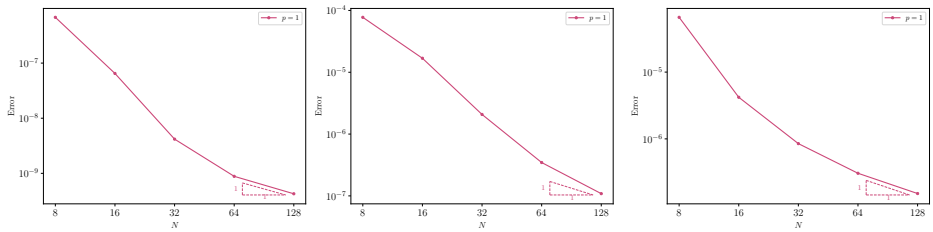
(a) $\|u - u_h\|_{L^2L^2}$ **(b)** $\|u - u_h\|_{L^2H^1}$ **(c)** $\|p - p_h\|_{L^2L^2}$

Figure 5.5: Convergence plot for the Stokes equation, solved using BDF-2 with BDF-1 for the initial step. The $[Q_2]^2 \times Q_1$ Taylor-Hood finite elements were used.



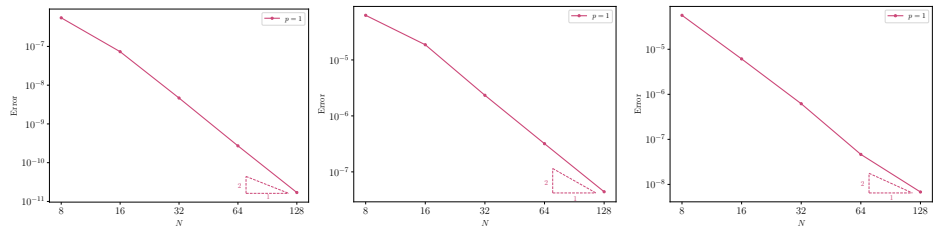
(a) Stokes problem. **(b)** Navier-Stokes.

Figure 5.6: The figures show the computed EOC for the BDF-2 method with BDF-1 for the initial step with $[Q_2]^2 \times Q_1$ elements. The error is summed over the time steps $t \geqslant T/4$ only. When the intial error spike is cut away, optimal convergence is regained.



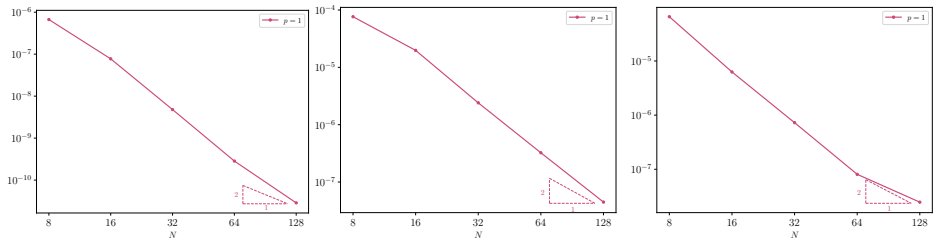
(a) $\|u - u_h\|_{L^2L^2}$ **(b)** $\|u - u_h\|_{L^2H^1}$ **(c)** $\|p - p_h\|_{L^2L^2}$

Figure 5.7: Convergence plot for the Navier-Stokes equation, solved using BDF-1. The $[Q_2]^2 \times Q_1$ Taylor-Hood finite elements were used.



(a) $\|u - u_h\|_{L^2L^2}$ **(b)** $\|u - u_h\|_{L^2H^1}$ **(c)** $\|p - p_h\|_{L^2L^2}$

Figure 5.8: Convergence plot for the Navier-Stokes equation, solved using BDF-2, with interpolated initial steps. The $[Q_2]^2 \times Q_1$ Taylor-Hood finite elements were used.



(a) $\|u - u_h\|_{L^2L^2}$ **(b)** $\|p - p_h\|_{L^2H^1}$ **(c)** $\|p - p_h\|_{L^2L^2}$

Figure 5.9: Convergence plot for the Navier-Stokes equation, solved using BDF-2, with BDF-1 as the first step. The $[Q_2]^2 \times Q_1$ Taylor-Hood finite elements were used.

seen in Figure 5.7 and in Figure 5.8 for BDF-2 with interpolated initial steps. Similar to the BDF-2 solver for the Stokes problem, we observe from Figure 5.9 a small drop in EOC of the L^2L^2-norm of the pressure. When the initial steps are disregarded from the error computations, Figure 5.6b show optimal convergence in all norms.

5.3.2 Moving sphere example

In this example, the goal was to simulate a realistic flow around a moving sphere in a channel. The model was presented earlier in Section 5.1.2, depicted in Figure 5.2.

The background domain is a rectangular channel with half-length $l = 2$ and radius $r = 1$. The end time was set at $T = 16$, and $\tau = 1/30$. Refinement level 7 was used for the simulation, resulting in a regular grid with $N = 2^{7-1}$ cells along the short side of the domain. Taylor-Hood $[Q_2]^2 \times Q_1$-elements was used.

The rigid sphere $B(t)$ has radius $r_s = 0.5$, and the sphere center follows the path $r_B(t)$, with velocity $v_B(t)$ both given in (5.5). The level set function defining the domain was thus given by

$$\phi(t, x) = -\|x - r_B(t)\|_2 + r_s \tag{5.20}$$

The maxium speed of the sphere, and hence the fluid was

$$\hat{v}_B = \frac{0.9\pi}{8}(l - r_s) = 0.53. \tag{5.21}$$

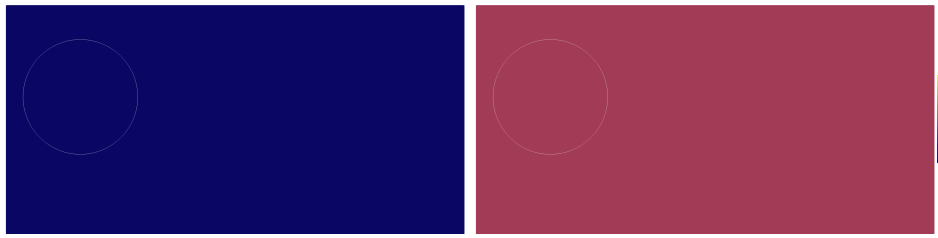
The kinematic viscosity of the fluid was set to $\nu = 0.001$, leading to a Reynolds number of

$$\mathrm{Re} = \frac{\hat{v}_B L}{\nu} = \frac{0.53 \cdot 2r_s}{\nu} = 530. \tag{5.22}$$

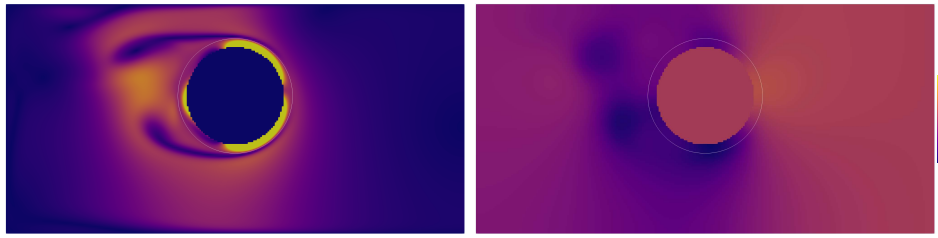
This high Reynolds number will lead to turbulent flows behind the sphere as it stirs the fluid.

For the time discretisation, the BDF-2 method was used, and hence, the procedure first ran one step of BDF-1 to start the multistage BDF-2 method. The discrete weak formulation of the problem was presented in Section 5.2.2. The CutFEM stability parameters were set to $\beta_{u,0} = \beta_{p,0} = 0.5$, while the Nitsche penalty parameter was kept at $\mu = 5\nu p(p + 1)/h$.

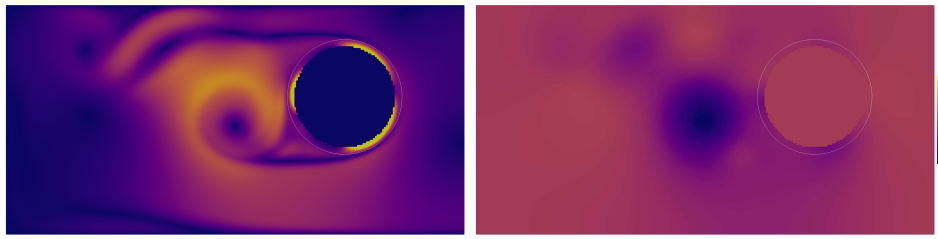
Some snapshots of the velocity and pressure solution at specific time steps are shown in Figure 5.10 and Figure 5.11. In the figure, we observe that the low kinematic viscosity of the fluid causes vortex streaming around the sphere. The sphere and the fluid has a velocity of zero initially, but the no-slip boundary conditions along the sphere boundary cause the fluid to start flowing in the channel as the sphere accelerates. Also, note how far the active mesh extends past the sphere boundary varies as the sphere moves. This is due to the varying horizontal velocity of the sphere.



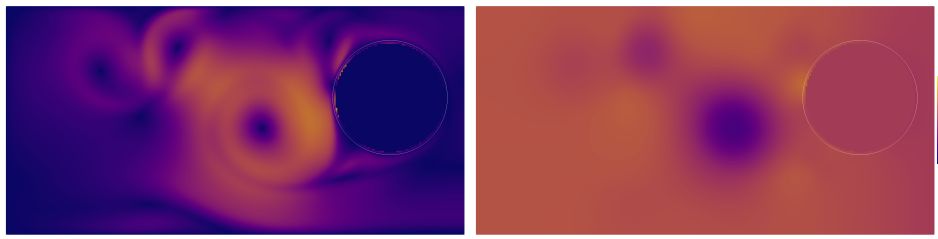
(a) $t = 0$



(b) $t = 4$

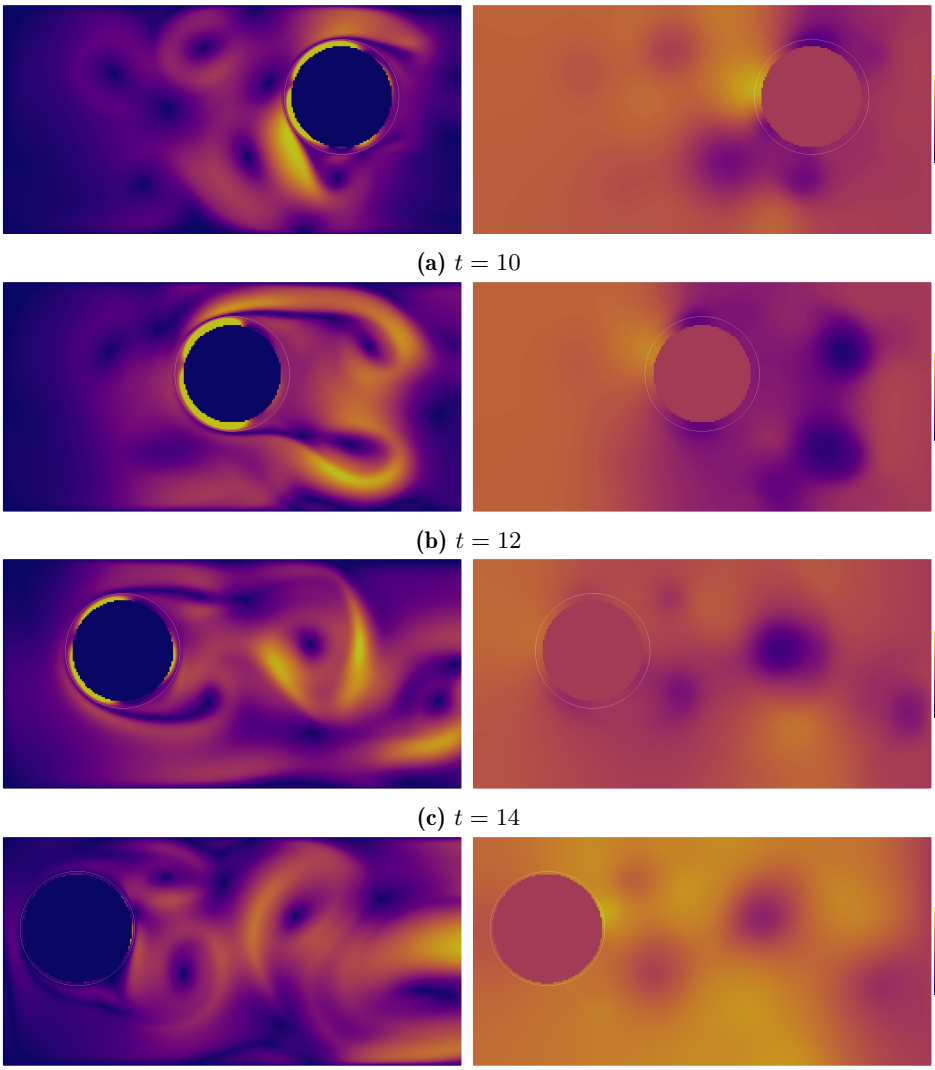


(c) $t = 6$



(d) $t = 8$

Figure 5.10: Navier-Stokes flow around a moving sphere in a channel. The velocity magnitude is plotted in the left column, while the pressure is plotted to the right. No-slip boundary conditions for the fluid along the sphere boundary, causes vortex streams in the channel as the sphere stirs the fluid.



(a) $t = 10$

(b) $t = 12$

(c) $t = 14$

(d) $t = 16$

Figure 5.11: Navier-Stokes flow around a moving sphere in a channel. The velocity magnitude is plotted in the left column, while the pressure is plotted to the right. No-slip boundary conditions for the fluid along the sphere boundary, causes vortex streams in the channel as the sphere stirs the fluid.

Chapter 6

Conclusion and outlook

Conclusion. In this thesis, we show through numerical experiments that CutFEM is a robust and accurate method for both parabolic problems and flow problems undergoing large deformations. The method achieves optimal order of convergence in both space and time for problems with an analytical solution. In classic benchmark examples for Navier-Stokes on a stationary domain, the pressure matches the benchmark data. However, the lift and drag computations did not match the data. Since the convergence tests showed optimal convergence, and the drag and lift values were successfully computed when using CutFEM in [Schott, 2016], this deviation is likely due to an unresolved bug in the surface force computations. These calculations need to be corrected before fully two-way coupled problems can be implemented.

Outlook. This thesis resulted in an implementation of the Navier-Stokes equations on a domain moving with prescribed motion. This can already be used to solve a whole range of interesting problems in science and engineering. A biological application is the simulation of flow in the heart when the movement is known from imaging measurements [Chnafa et al., 2015]. However, for simulating larger problems, full parallelisation of the solvers would be needed. As a further improvement, the solvers could be extended to solve fully-coupled fluid-rigid body problems. This naturally requires correct computations of the viscous forces over the surface of the submerged body to calculate the body's movement accurately. This would widen the range of application problems even more. One such application is to simulate how a microbubble moves in a capillary blood vein when being influenced by an external ultrasound source. This treatment method is an active field of medical research [Lin et al., 2012, van Wamel et al., 2016, Dimcevski et al., 2016, Arango-Restrepo et al., 2021]. At these small scales, medical imaging is not possible. Therefore, developing software to simulate this situation with the help of mathematical modelling could help make or break hy-

potheses on how a vibrating microbubble impacts the vessel wall and surrounding tissue. To achieve this, a possible approach is to use the Rayleigh-Plesset equation [Prosperetti, 1982] for modelling an elastic vibrating bubble in a channel, using a two-way coupling to the implemented Navier-Stokes solver. As blood is a fluid consisting of blood cells and plasma, a Newtonian fluid model is not a good approximation in smaller vessels [Formaggia et al., 2009]. Therefore, using a non-Newtonian fluid model [Irgens, 2014, Pacheco et al., 2021] could also be an interesting extension for such a model problem.

If we want to look for applications beyond solving PDEs on moving domains, shape optimisation [Allaire et al., 2021] is a branch where CutFEM has proved to be a useful tool. In shape optimisation, the aim is to optimise some objective functional with respect to the shape of the domain and PDE constraints. Level set methods [Saye and Sethian, 2020] was studied for shape optimisation problems in [Allaire et al., 2002, Allaire et al., 2014]. In [Burman et al., 2018] CutFEM was applied to a shape optimisation problem for linear elasticity and for Navier-Stokes equations in [Villanueva and Maute, 2017]. In CutFEM, the level set is represented as a finite element function on the background domain. Therefore, the level set can conveniently be constrained by the transport equation, where the convection field constitutes the control of the optimisation problem. This control is then updated through the computation of the shape derivative, minimising the objective functional with respect to the shape of the physical domain. With a Navier-Stokes solver already implemented using CutFEM, delving into shape optimisation could make solving an entirely new form of application problems possible.

Bibliography

[Allaire, 2001] Allaire, G. (2001). *Shape Optimization by the Homogenization Method*, volume 146 of *Applied Mathematical Sciences*. Springer Science & Business Media.

[Allaire et al., 2014] Allaire, G., Dapogny, C., and Frey, P. (2014). Shape optimization with a level set based mesh evolution method. *Computer Methods in Applied Mechanics and Engineering*, 282:22–53.

[Allaire et al., 2021] Allaire, G., Dapogny, C., and Jouve, F. (2021). Shape and topology optimization. In Bonito, A. and Nochetto, R. H., editors, *Geometric Partial Differential Equations - Part II*, volume 22 of *Handbook of Numerical Analysis*, chapter 1, pages 1–132. Elsevier.

[Allaire et al., 2002] Allaire, G., Jouve, F., and Toader, A.-M. (2002). A level-set method for shape optimization. *Comptes Rendus Mathematique*, 334(12):1125–1130.

[Arango-Restrepo et al., 2021] Arango-Restrepo, A., Rubi, J. M., Kjelstrup, S., Angelsen, B. A. J., and de Lange Davies, C. (2021). Enhancing carrier flux for efficient drug delivery in cancer tissues. *Biophysical Journal*, 120(23):5255–5266.

[Arndt et al., 2020] Arndt, D., Bangerth, W., Blais, B., Clevenger, T. C., Fehling, M., Grayver, A. V., Heister, T., Heltai, L., Kronbichler, M., Maier, M., Munch, P., Pelteret, J.-P., Rastak, R., Thomas, I., Turcksin, B., Wang, Z., and Wells, D. (2020). The `deal.II` Library, Version 9.2. *Journal of Numerical Mathematics*, 28(3):131–146.

[Barrett et al., 2020] Barrett, J. W., Garcke, H., and Nürnberg, R. (2020). Parametric finite element approximations of curvature-driven interface evolutions. In Bonito, A. and Nochetto, R. H., editors, *Geometric Partial Differential Equations - Part I*, volume 21 of *Handbook of Numerical Analysis*, chapter 4, pages 275–423. Elsevier.

[Boffi et al., 2013] Boffi, D., Brezzi, F., Fortin, M., et al. (2013). *Mixed Finite Element Methods and Applications*, volume 44 of *Springer Series in Computational Mathematics*. Springer.

[Borthwick, 2018] Borthwick, D. (2018). *Introduction to Partial Differential Equations*. Univesitext. Springer International Publishing AG.

[Brenner and Scott, 2008] Brenner, S. C. and Scott, L. R. (2008). *The Mathematical Theory of Finite Element Methods*, volume 15 of *Texts in Applied Mathematics*. Springer.

[Brezzi and Falk, 1991] Brezzi, F. and Falk, R. S. (1991). Stability of higher-order Hood-Taylor methods. *SIAM Journal on Numerical Analysis*, 28:581–590.

[Bungartz et al., 2010] Bungartz, H.-J., Mehl, M., and Schäfer, M. (2010). *Fluid Structure Interaction II: Modelling, Simulation, Optimization*, volume 73 of *Lecture Notes in Computational Science and Engineering*. Springer Science & Business Media.

[Burman, 2010] Burman, E. (2010). Ghost penalty. *Comptes Rendus Mathematique*, 348(21):1217–1220.

[Burman et al., 2015] Burman, E., Claus, S., Hansbo, P., Larson, M. G., and Massing, A. (2015). CutFEM: Discretizing geometry and partial differential equations. *International Journal for Numerical Methods in Engineering*, 104(7):472–501.

[Burman et al., 2018] Burman, E., Elfverson, D., Hansbo, P., Larson, M. G., and Larsson, K. (2018). Shape Optimization Using the Cut Finite Element Method. *Computer Methods in Applied Mechanics and Engineering*, 328:242–261.

[Burman et al., 2022] Burman, E., Frei, S., and Massing, A. (2022). Eulerian time-stepping schemes for the non-stationary Stokes equations on time-dependent domains. *Numerische Mathematik*, pages 1–56.

[Burman and Hansbo, 2012] Burman, E. and Hansbo, P. (2012). Fictitious domain finite element methods using cut elements: II. A stabilized Nitsche method. *Applied Numerical Mathematics*, 62(4):328–341. Third Chilean Workshop on Numerical Analysis of Partial Differential Equations (WONAPDE 2010).

[Burman and Hansbo, 2014] Burman, E. and Hansbo, P. (2014). Fictitious domain methods using cut elements: III. A stabilized Nitsche method for Stokes' problem. *ESAIM: Mathematical Modelling and Numerical Analysis*, 48(3):859–874.

[Burman et al., 2019] Burman, E., Hansbo, P., Larson, M. G., and Samvin, D. (2019). A cut finite element method for elliptic bulk problems with embedded surfaces. *GEM - International Journal on Geomathematics*, 10(1):10.

[Bänsch and Schmidt, 2020] Bänsch, E. and Schmidt, A. (2020). Free boundary problems in fluids and materials. In Bonito, A. and Nochetto, R. H., editors, *Geometric Partial Differential Equations - Part I*, volume 21 of *Handbook of Numerical Analysis*, chapter 7, pages 555–619. Elsevier.

[Chnafa et al., 2015] Chnafa, C., Mendez, S., Moreno, R., and Nicoud, F. (2015). Using Image-based CFD to Investigate the Intracardiac Turbulence. In Quarteroni, A., editor, *Modeling the Heart and the Circulatory System*, volume 14 of *Modeling, Simulation & Applications*, chapter 4, pages 97–117. Springer International Publishing.

[Claus and Kerfriden, 2019] Claus, S. and Kerfriden, P. (2019). A CutFEM method for two-phase flow problems. *Computer Methods in Applied Mechanics and Engineering*, 348:185–206.

[Dimcevski et al., 2016] Dimcevski, G., Kotopoulis, S., Bjånes, T., Hoem, D., Schjøtt, J., Gjertsen, B. T., Biermann, M., Molven, A., Sorbye, H., McCormack, E., Postema, M., and Gilja, O. H. (2016). A human clinical trial using ultrasound and microbubbles to enhance gemcitabine treatment of inoperable pancreatic cancer. *Journal of Controlled Release*, 243:172–181.

[Dokken et al., 2020] Dokken, J. S., Johansson, A., Massing, A., and Funke, S. W. (2020). A multimesh finite element method for the Navier-Stokes equations based on projection methods. *Computer Methods in Applied Mechanics and Engineering*, 368:113129.

[Donea et al., 2004] Donea, J., Huerta, A., Ponthot, J.-P., and Rodriguez-Ferran, A. (2004). Arbitrary Lagrangia-Eulerian Methods. In Stein, E., de Borst, R., and Hughes, T. J. R., editors, *Encyclopedia of Computational Mechanics*, chapter 14. John Wiley & Sons, Ltd.

[Douglas and Dupont, 1976] Douglas, J. and Dupont, T. (1976). Interior Penalty Procedures for Elliptic and Parabolic Galerkin Methods. In Glowinski, R. and Lions, J. L., editors, *Computing Methods in Applied Sciences*, pages 207–216, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Du and Feng, 2020] Du, Q. and Feng, X. (2020). The phase field method for geometric moving interfaces and their numerical approximations. In Bonito, A. and Nochetto, R. H., editors, *Geometric Partial Differential Equations - Part I*, volume 21 of *Handbook of Numerical Analysis*, chapter 5, pages 425–508. Elsevier.

[Eck et al., 2017] Eck, C., Garcke, H., and Knabner, P. (2017). *Mathematical Modeling*. Springer Undergraduate Mathematical Series. Springer.

[Ern and Guermond, 2021] Ern, A. and Guermond, J.-L. (2021). *Finite Elements II: Galerkin Approximation, Elliptic and Mixed PDEs*, volume 73 of *Text in Applied Mathematics*. Springer Nature.

[Ethier and Steinman, 1994] Ethier, C. R. and Steinman, D. A. (1994). Exact Fully 3D Navier-Stokes Solutions for Benchmarking. *International journal for Numerical Methods in Fluids*, 19(369-375).

[Evans, 2010] Evans, L. C. (2010). *Partial Differential Equations*, volume 19 of *Gratudate Studies in Mathematics*. American Mathematical Society, second edition.

[Fefferman, 2000] Fefferman, C. L. (2000). Problem Description: Existence and smoothness of the Navier-Stokes equations. *Clay Mathematics Institute*.

[Fernández et al., 2009] Fernández, M. A., Formaggia, L., Gerbeau, J., and Quarteroni, A. (2009). The derivation of the equations for fluids and structure. In Formaggia, L., Quarteroni, A., and Veneziani, A., editors, *Cardiovascular Mathematics: Modeling and simulation of the circulatory system*, volume 1 of *Modeling, Simulation & Applications*, chapter 3, pages 77–122. Springer.

[Formaggia et al., 2009] Formaggia, L., Quarteroni, A., and Veneziani, A. (2009). *Cardiovascular Mathematics: Modeling and simulation of the circulatory system*, volume 1 of *Modeling, Simulation & Applications*. Springer Science & Business Media.

[Frachon and Zahedi, 2019] Frachon, T. and Zahedi, S. (2019). A cut finite element method for incompressible two-phase Navier-Stokes flows. *Journal of Computational Physics*, 384:77–98.

[Fries and Belytschko, 2010] Fries, T.-P. and Belytschko, T. (2010). The extended/generalized finite element method: An overview of the method and its applications. *International Journal for Numerical Methods in Engineering*, 84(3):253–304.

[Guermond et al., 2006] Guermond, J., Minev, P., and Shen, J. (2006). An overview of projection methods for incompressible flows. *Computer Methods in Applied Mechanics and Engineering*, 195(44):6011–6045.

[Guzmán and Olshanskii, 2018] Guzmán, J. and Olshanskii, M. (2018). Inf-sup stability of geometrically unfitted Stokes finite elements. *Mathematics of Computation*, 87(313):2091–2112.

[Gürkan and Massing, 2019] Gürkan, C. and Massing, A. (2019). A stabilized cut discontinuous Galerkin framework for elliptic boundary value and interface problems. *Computer Methods in Applied Mechanics and Engineering*, 348:466–499.

[Hansbo, 2005] Hansbo, P. (2005). Nitsche's method for interface problems in computational mechanics. *GAMM-Mitt.*, 2(183-206).

[Hansbo et al., 2016] Hansbo, P., Larson, M. G., and Zahedi, S. (2016). A cut finite element method for coupled bulk-surface problems on time-dependent domains. *Computer Methods in Applied Mechanics and Engineering*, 307:96–116.

[Hintermüller and Keil, 2021] Hintermüller, M. and Keil, T. (2021). Optimal control of geometric partial differential equations. In Bonito, A. and Nochetto, R. H., editors, *Geometric Partial Differential Equations - Part II*, volume 22 of *Handbook of Numerical Analysis*, chapter 3, pages 213–270. Elsevier.

[Holm, 2021] Holm, S. E. (2021). Introduction to the theory of the Cut Finite Element Method. *Project Report, Course TMA4500, NTNU.*

[Irgens, 2014] Irgens, F. (2014). *Rheology and Non-Newtonian Fluids.* Springer International Publishing Switzerland.

[John et al., 2016] John, V. et al. (2016). *Finite Element Methods for Incompressible Flow Problems*, volume 51 of *Springer Series in Computational Mathematics.* Springer.

[Larson and Bengzon, 2013] Larson, M. G. and Bengzon, F. (2013). *The Finite Element Method: Theory, Implementation, and Applications*, volume 10. Springer Science & Business Media.

[Lehrenfeld and Olshanskii, 2019] Lehrenfeld, C. and Olshanskii, M. (2019). An Eulerian finite element method for PDEs in time-dependent domains. *ESAIM: M2AN*, 53(2):585–614.

[Lehrenfeld et al., 2018] Lehrenfeld, C., Olshanskii, M. A., and Xu, X. (2018). A Stabilized Trace Finite Element Method for Partial Differential Equations on Evolving Surfaces. *SIAM Journal on Numerical Analysis*, 56(3):1643–1672.

[Lin et al., 2012] Lin, C.-Y., Li, J.-R., Tseng, H.-C., Wu, M.-F., and Lin, W.-L. (2012). Enhancement of focused ultrasound with microbubbles on the treatments of anticancer nanodrug in mouse tumors. *Nanomedicine: Nanotechnology, Biology and Medicine*, 8(6):900–907.

[Ludvigsson et al., 2018] Ludvigsson, G., Steffen, K. R., Sticko, S., Wang, S., Xia, Q., Epshteyn, Y., and Kreiss, G. (2018). A High-Order Numerical Methods for 2D Parabolic Problems in Single and Composite Domains. *Journal of Scientific Computing*, 76(2):812–847.

[Massing et al., 2015] Massing, A., Larson, M., Logg, A., and Rognes, M. (2015). A Nitsche-based cut finite element method for a fluid-structure interaction problem. *Communications in Applied Mathematics and Computational Science*, 10(2):97–120.

[Massing et al., 2014] Massing, A., Larson, M. G., Logg, A., and Rognes, M. E. (2014). A Stabilized Nitsche Fictitious Domain Method for the Stokes Problem. *Journal of Scientific Computing*, 61(604–628).

[Massing et al., 2018] Massing, A., Schott, B., and Wall, W. (2018). A stabilized Nitsche cut finite element method for the Oseen problem. *Computer Methods in Applied Mechanics and Engineering*, 328:262–300.

[Moës et al., 1999] Moës, N., Dolbow, J., and Belytschko, T. (1999). A finite element method for crack growth without remeshing. *International Journal for Numerical Methods in Engineering*, 46(1):131–150.

[Nitsche, 1971] Nitsche, J. (1971). 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(3):9–15.

[Olshanskii et al., 2009] Olshanskii, M. A., Reusken, A., and Grande, J. (2009). A Finite Element Method for Elliptic Equations on Surfaces. *SIAM Journal on Numerical Analysis*, 47(5):3339–3358.

[Olshanskii and Xu, 2017] Olshanskii, M. A. and Xu, X. (2017). A Trace Finite Element Method for PDEs on Evolving Surfaces. *SIAM Journal on Scientific Computing*, 39(4):A1301–A1319.

[Pacheco et al., 2021] Pacheco, D. R., Schussnig, R., and Fries, T.-P. (2021). An efficient split-step framework for non-Newtonian incompressible flow problems with consistent pressure boundary conditions. *Computer Methods in Applied Mechanics and Engineering*, 382:113888.

[Prosperetti, 1982] Prosperetti, A. (1982). A generalization of the Rayleigh-Plesset equation of bubble dynamics. *The Physics of Fluids*, 25(3):409–410.

[Quarteroni, 2017] Quarteroni, A. (2017). *Numerical Models for Differential Problems*, volume 16 of *Modeling, Simulation & Applications*. Springer International Publishing AG, third edition.

[Quarteroni et al., 2010] Quarteroni, A., Sacco, R., and Saleri, F. (2010). *Numerical Mathematics*, volume 37 of *Text in Applied Mathematics*. Springer Science & Business Media, second edition.

[Richter, 2017] Richter, T. (2017). *Fluid-structure Interactions: Models, Analysis and Finite Elements*, volume 118 of *Lecture Notes in Computational Science and Engineering*. Springer.

[Rothe, 1930] Rothe, E. (1930). Zweidimensionale parabolische Randwertaufgaben als Grenzfall eindimensionaler Randwertaufgaben. *Mathematische Annalen*, 102:650–670.

[Saye, 2015] Saye, R. (2015). High-order quadrature methods for implicitly defined surfaces and volumes in hyperrectangles. *SIAM Journal on Scientific Computing*, 37(2):A993–A1019.

[Saye and Sethian, 2020] Saye, R. I. and Sethian, J. A. (2020). A review of level set methods to model interfaces moving under complex physics: Recent challenges and advances. In Bonito, A. and Nochetto, R. H., editors, *Geometric Partial Differential Equations - Part I*, volume 21 of *Handbook of Numerical Analysis*, chapter 6, pages 509–554. Elsevier.

[Schäfer et al., 1996] Schäfer, M., Turek, S., Durst, F., Krause, E., and Rannacher, R. (1996). Benchmark Computations of Laminar Flow Around a Cylinder. In Hirschel, E. H., editor, *Flow Simulation with High-Performance Computers II*, volume 48 of *Notes on Numerical Fluid Mechanics*, pages 547–566. Vieweg+Teubner Verlag, Wiesbaden.

[Schott, 2016] Schott, B. (2016). *Stabilized Cut Finit Element Metods for Complex Interface Coupled Flow Problems*. PhD thesis, Ph. D. thesis, University College London.

[Schott et al., 2019] Schott, B., Ager, C., and Wall, W. A. (2019). Monolithic cut finite element-based approaches for fluid-structure interaction. *International Journal for Numerical Methods in Engineering*, 119(8):757–796.

[Simon Sticko, 2021] Simon Sticko (2021). The deal.II Library: The step-85 tutorial program. `https://www.dealii.org/developer/doxygen/deal.II/step_85.html`.

[Stein, 1971] Stein, E. M. (1971). *Singular integrals and differentiability properties of functions*, volume 30 of *Princeton Mathematical Series*. Princeton university press.

[Sticko et al., 2021] Sticko, S., Kovacs, B., and Massing, A. (2021). High-Order Backward Difference - Cut Finite Element Full Disretization of Evolving Domain Problems. *Preprint submitted to arXiv*.

[Thomée, 2007] Thomée, V. (2007). *Galerkin Finite Element Methods for Parabolic Problems*, volume 25. Springer Science & Business Media.

[Tröltzsch, 2010] Tröltzsch, F. (2010). *Optimal Control of Parial Differential Equations. Theory, Methods and Applications*, volume 112 of *Graduate Studies in Mathematics*. Americal Mathematical Society.

[Turek et al., 1996a] Turek, S., Becker, C., Kilian, S., Möller, M., Buijssen, S., Göddeke, D., and Köster, M. (1996a). The CFD Benchmarking project: 2D-1. `http://www.mathematik.tu-dortmund.de/~featflow/en/benchmarks/cfdbenchmarking/flow/dfg_benchmark1_re20.html`. Accessed: 21.11.2021, *Technische Universität Dortmund*.

[Turek et al., 1996b] Turek, S., Becker, C., Kilian, S., Möller, M., Buijssen, S., Göddeke, D., and Köster, M. (1996b). The CFD Benchmarking project: 2D-3. `http://www.mathematik.tu-dortmund.de/~featflow/en/benchmarks/cfdbenchmarking/flow/dfg_benchmark3_re100.html`. Accessed: 21.11.2021, *Technische Universität Dortmund*.

[van Wamel et al., 2016] van Wamel, A., Sontum, P. C., Healey, A., Kvåle, S., Bush, N., Bamber, J., and de Lange Davies, C. (2016). Acoustic Cluster Therapy (ACT) enhances the therapeutic efficacy of paclitaxel and Abraxane® for treatment of human prostate adenocarcinoma in mice. *Journal of Controlled Release*, 236:15–21.

[Villanueva and Maute, 2017] Villanueva, C. H. and Maute, K. (2017). CutFEM topology optimization of 3D laminar incompressible flow problems. *Computer Methods in Applied Mechanics and Engineering*, 320:444–473.

[von Wahl et al., 2021] von Wahl, H., Richter, T., and Lehrenfeld, C. (2021). An unfitted Eulerian finite element method for the time-dependent Stokes problem on moving domains. *IMA Journal of Numerical Analysis*.

[von Wahl, 2021] von Wahl, H. M. (2021). *Unfitted Finite Elements for Fluid-Rigid Body Interaction Problems*. PhD thesis, Ph. D. thesis, Otto-von-Guericke-Universität Magdeburg.

Sigmund Eggen Holm

Cut finite element methods for PDEs on moving domains

NTNU
Norwegian University of
Science and Technology