UPPSALA
UNIVERSITET

*Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology* 1656

# High Order Cut Finite Element Methods for Wave Equations

SIMON STICKO

ACTA
UNIVERSITATIS
UPSALIENSIS
UPPSALA
2018

ISSN 1651-6214
ISBN 978-91-513-0300-0
urn:nbn:se:uu:diva-347439

Dissertation presented at Uppsala University to be publicly examined in ITC 2446, Lägerhyddsvägen 2, Uppsala, Friday, 25 May 2018 at 10:15 for the degree of Doctor of Philosophy. The examination will be conducted in English. Faculty examiner: Professor Mats G. Larson (Department of Mathematics and Mathematical Statistics, Umeå University).

**Abstract**

Sticko, S. 2018. High Order Cut Finite Element Methods for Wave Equations. *Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology* 1656. 37 pp. Uppsala: Acta Universitatis Upsaliensis. ISBN 978-91-513-0300-0.

This thesis considers wave propagation problems solved using finite element methods where a boundary or interface of the domain is not aligned with the computational mesh. Such methods are usually referred to as cut or immersed methods. The motivation for using immersed methods for wave propagation comes largely from scattering problems when the geometry of the domain is not known a priori. For wave propagation problems, the amount of computational work per dispersion error is generally lower when using a high order method. For this reason, this thesis aims at studying high order immersed methods.

Nitsche's method is a common way to assign boundary or interface conditions in immersed finite element methods. Here, penalty terms that are consistent with the boundary/interface conditions are added to the weak form. This requires that special quadrature rules are constructed on the intersected elements, which take the location of the immersed boundary/interface into account. A common problem for all immersed methods is small cuts occurring between the elements in the mesh and the computational domain. A suggested way to remedy this is to add terms penalizing jumps in normal derivatives over the faces of the intersected elements.

Paper I and Paper II consider the acoustic wave equation, using first order elements in Paper I, and using higher order elements in Paper II. High order elements are then used for the elastic wave equation in Paper III. Papers I to III all use continuous Galerkin, Nitsche's method, and jump-stabilization. Paper IV compares the errors of this type of cut finite element method with two other numerical methods. One result from Paper II is that the added jump-stabilization results in a mass matrix with a high condition number. This motivates the investigation of alternatives. Paper V considers a hybridizable discontinuous Galerkin method. This paper investigates to what extent local time stepping in combination with cell-merging can be used to overcome the problem of small cuts.

*Keywords:* Cut finite element, Wave equation, Immersed, Fictitious domain

*Simon Sticko, Department of Information Technology, Division of Scientific Computing, Box 337, Uppsala University, SE-751 05 Uppsala, Sweden.*



ISSN 1651-6214
ISBN 978-91-513-0300-0
urn:nbn:se:uu:diva-347439 (http://urn.kb.se/resolve?urn=urn:nbn:se:uu:diva-347439)

# List of papers

This thesis is based on the following papers, which are referred to in the text by their Roman numerals.

I S. Sticko and G. Kreiss. A Stabilized Nitsche Cut Element Method for the Wave Equation. *Computer Methods in Applied Mechanics and Engineering*, 309:364–387, September 2016. ISSN: 00457825
*Contributions:* The author of this thesis performed the implementation of the method and the numerical experiments. The manuscript was written by the author of this thesis in collaboration with the second author.

II S. Sticko and G. Kreiss. Higher Order Cut Finite Elements for the Wave Equation. *ArXiv e-prints: 1608.03107v2*, 2018. (Submitted)
*Contributions:* The author of this thesis performed the implementation of the method and the numerical experiments. The manuscript was written by the author of this thesis in collaboration with the second author.

III S. Sticko, G. Ludvigsson, and G. Kreiss. High Order Cut Finite Elements for the Elastic Wave Equation. *ArXiv e-prints: 1804.00332*, 2018. (Submitted)
*Contributions:* The implementation of the method and the experiments were performed by the first two authors. The manuscript was written by the first two authors in collaboration with the third author.

IV G. Ludvigsson, K. R. Steffen, S. Sticko, S. Wang, Q. Xia, Y. Epshteyn, and G. Kreiss. High-Order Numerical Methods for 2D Parabolic Problems in Single and Composite Domains. *Journal of Scientific Computing*:1–36, January 2018. ISSN: 0885-7474, 1573-7691
*Contributions:* The author of the present thesis and the first author implemented and performed the experiments related to the cut finite element method. The paper was written in collaboration between all authors.

V S. Schoeder, S. Sticko, G. Kreiss, and M. Kronbichler. High Order Cut Discontinuous Galerkin Methods with Local Time Stepping for Acoustics, 2017. (In revision)
*Contributions:* The method was implemented in collaboration between the authors. The manuscript was written by the first author in collaboration with the remaining authors.

# Contents

# 1. Introduction and Motivation

Waves occur in a variety of forms in nature. For example, sound is pressure waves that propagate through the air, an earthquake consists of waves that propagate through the ground, and we send information all around the world as electromagnetic waves. Mathematically, these waves can be described by partial differential equations (PDEs), and in a variety of applications it is necessary to solve these in order to predict how waves propagate. In general, PDEs are too difficult to be solved exactly using only pen and paper. The strategy is then to resort to solving them approximately with computers, also known as solving them numerically. This is the topic of the present thesis. In particular, we are interested in solving these equations using so-called immersed methods.

The setting here is that we have a partial differential equation, posed on a domain, $\Omega_1$, with boundary, $\Gamma = \partial\Omega_1$, such as the one in Figure 1.1. Most numerical methods for solving partial differential equations require a mesh in order to solve the problem. The mesh often consists of connected triangles, as in Figure 1.2, but can also consist of quadrilaterals. The classical approach is to construct a mesh which conforms to the geometry. In an immersed method one instead lets the domain float on top of a background mesh, as illustrated in Figure 1.3. This makes solving the problem more complicated but is sometimes advantageous. In several applications, the domain under consideration moves during the simulation, for example, if the position and shape of the domain depend on time. Moving the nodes of the mesh can make the triangles very distorted, such as illustrated in Figure 1.4. The distortion will degrade the performance of the method. This problem can be solved by creating a new mesh, having the same shape as the deformed geometry, but having triangles with better shapes. This is known as remeshing and is a very time-consuming operation. One reason for using an immersed method is to avoid this. If the domain is floating on top of a background mesh, the mesh can never become distorted. Thus, applications where remeshing is best avoided is the main motivation for immersed methods. A second reason is that in many applications it can be very costly to construct a mesh which conforms to the domain. Using an immersed method could in some cases be less time-consuming.

It might seem odd to consider an immersed method in the context of wave propagation. After all, applications involving wave propagation and time-dependent geometries are rather sparse. The main motivation instead comes from problems where some geometry of interest is not known a priori. One example is so-called photoacoustic imaging, which is illustrated in Figure 1.5.

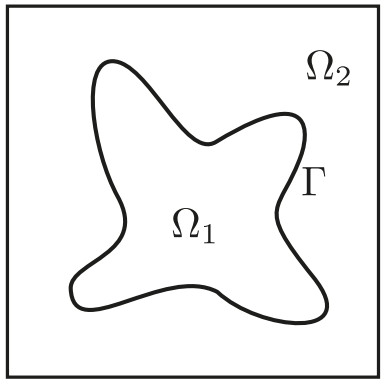


*Figure 1.1.* A given domain to solve a PDE on.

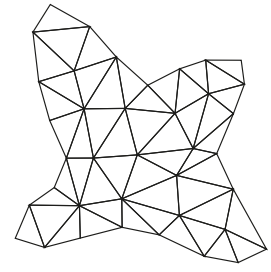

*Figure 1.2.* A computational mesh conforming to the geometry.

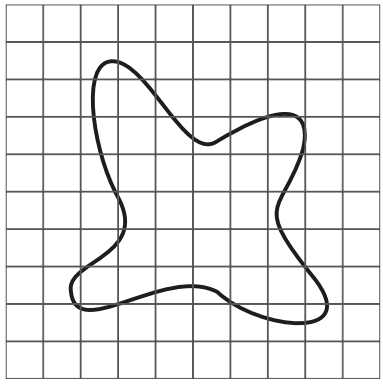

*Figure 1.3.* A domain immersed in a background mesh.

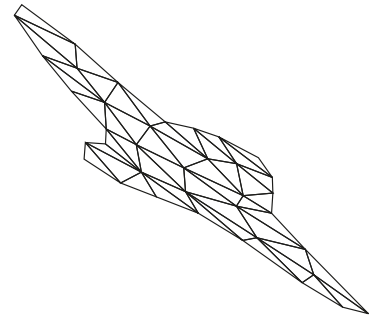

*Figure 1.4.* A distorted mesh with triangles of poor quality.

Here, we have some object (typically tissue) with an unknown geometry that we would like to determine. A laser pulse is delivered to the object, which makes it heat up and expands slightly. This expansion creates acoustic waves which propagate away from the object. These waves can be measured, and given this data, an optimization problem can be solved to reconstruct the geometry of the object. When solving this optimization problem we would need to iterate and solve the problem repeatedly for a number of different geometries. When iterating over different geometries, remeshing is going to be very expensive. Hence, an immersed method might be more efficient. Similar applications exist also in geophysics. Here, waves are typically sent towards an object with the purpose of determining its geometry from the scattered waves.

In order to solve this optimization problem, it is necessary to be able to first efficiently solve the problem for a single geometry, which is the research

direction of the present thesis. In general, for hyperbolic problems work per dispersion error increases slower for higher order methods [6]. This means that higher order methods typically are more efficient for wave propagation problems. This serves as motivation for studying high order immersed methods.

Historically, the first immersed method is usually attributed to Peskin [7, 8]. Nowadays, immersed methods have been developed with several different classical numerical methods as base. Since the work in Paper I-V is based on the finite element method, we shall in the following chapters focus on discussing this case.

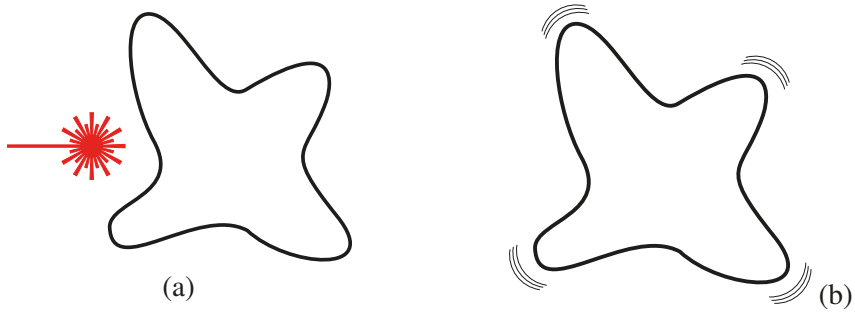


*Figure 1.5.* Illustration of photoacoustic imaging. (a) Laser pulse delivered to object. (b) Thermoelastic expansion creating outward propagating waves.

# 2. Basic Approach and Boundary Conditions

Assume that we want to solve a PDE on the domain $\Omega_1$, shown in Figure 1.1. Often one solves for the degrees of freedom of the elements illustrated in Figure 2.1. This is the smallest set of elements that covers $\Omega_1$. This approach is used in the majority of the papers in this thesis.

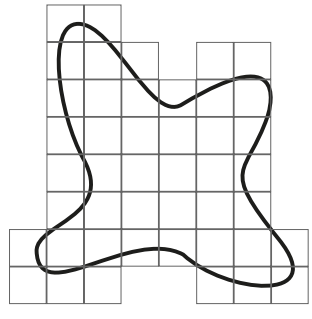

*Figure 2.1.* Smallest set of elements covering the domain $\Omega_1$.

A finite element method always starts from a weak formulation, consisting of various quantities integrated over the domain, $\Omega_1$, and its boundary, $\Gamma$. This integration is typically done element by element. Thus, the basic component of all immersed finite element methods is the following. On the elements intersected by the boundary, we are required to integrate over just a part of the element, as illustrated in Figure 2.2. That is, we need a quadrature rule for the part of $\Omega_1$, and the part of $\Gamma$, that falls within the element.

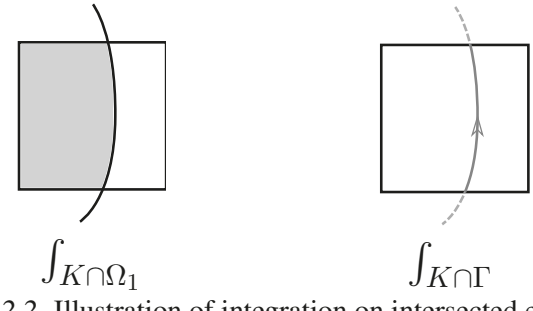


*Figure 2.2.* Illustration of integration on intersected elements.

There are a few different approaches for applying boundary conditions in immersed finite element methods. In order to have a model problem to discuss, consider the Poisson equation:

$$-\nabla^2 u = f(x), \quad x \in \Omega_1, \tag{2.1}$$

$$u = g_D(x), \quad x \in \Gamma_D, \tag{2.2}$$

$$\frac{\partial u}{\partial n} = g_N(x), \quad x \in \Gamma_N, \tag{2.3}$$

with $\Gamma_D \cup \Gamma_N = \Gamma$ and $\Gamma_D \neq \emptyset$. In the following, we shall use the standard inner products:

$$(u, v)_\Omega = \int_\Omega uv d\Omega, \quad \langle u, v \rangle_\Gamma = \int_\Gamma uv d\Gamma, \tag{2.4}$$

where the subscript indicates over which domain the integration takes part. Note, that $\langle \cdot, \cdot \rangle_\Gamma$ denotes integration over a curve in 2D (surface in 3D).

Let $V_h$ denote a finite element space. Different choices for constructing $V_h$ will be discussed in Chapter 3. Unless specified otherwise $V_h$ denotes a continuous space so that $V_h \subset H^1(\Omega_1)$. We shall discuss some aspects related to discontinuous Galerkin (dG) (see e.g. [9]) when $V_h$ is discontinuous between elements and $V_h \not\subset H^1(\Omega)$. However, letting $V_h$ be continuous shall for most of the time be sufficient to illustrate the point.

Multiplying (2.1) with a test function, $v \in H^1(\Omega_1)$, and integrating by parts leads to

$$(\nabla u, \nabla v)_{\Omega_1} - \left\langle \frac{\partial u}{\partial n}, v \right\rangle_{\Gamma_D \cup \Gamma_N} = (f, v)_{\Omega_1}. \tag{2.5}$$

Now the Neumann boundary condition (2.3) is consistent with

$$\left\langle \frac{\partial u}{\partial n}, v \right\rangle_{\Gamma_N} = \langle g_N, v \rangle_{\Gamma_N}, \tag{2.6}$$

so adding this leads to

$$(\nabla u, \nabla v)_{\Omega_1} - \left\langle \frac{\partial u}{\partial n}, v \right\rangle_{\Gamma_D} = (f, v)_{\Omega_1} + \langle g_N, v \rangle_{\Gamma_N}. \tag{2.7}$$

Note that the Neumann boundary condition is now enforced weakly by the new term appearing in the right-hand side. In two dimensions this term is a line integral that cuts through the elements, as in Figure 2. Now (2.7) is the starting point for a number of approaches for enforcing the Dirichlet boundary condition. In the following, we discuss a few different ones.

## 2.1 Nitsche's Method

Perhaps the most popular way to enforce boundary conditions is to use Nitsche's method [10]. Here, the Dirichlet boundary condition is enforced weakly, in the same way as with the Neumann condition in (2.7). Note that the Dirichlet boundary condition is consistent with the following two equations

$$\frac{\alpha}{h}\langle u, v\rangle_{\Gamma_D} = \frac{\alpha}{h}\langle g_D, v\rangle_{\Gamma_D}, \tag{2.8}$$

$$\pm\left\langle u, \frac{\partial v}{\partial n}\right\rangle_{\Gamma_D} = \pm\left\langle g_D, \frac{\partial v}{\partial n}\right\rangle_{\Gamma_D}, \tag{2.9}$$

which are also line integrals in 2D. Here, $\alpha \geq 0$ is a constant and $h$ denotes some measure of the grid size. Adding (2.8) and (2.9) to (2.7) gives the following weak formulation: Find $u_h \in V_h$ such that

$$a^{\pm}(u_h, v_h) = L^{\pm}(v_h), \quad \forall v_h \in V_h, \tag{2.10}$$

where

$$a^{\pm}(u_h, v_h) = (\nabla u_h, \nabla v_h)_{\Omega_1} - \left\langle \frac{\partial u_h}{\partial n}, v_h\right\rangle_{\Gamma_D} \pm \left\langle u_h, \frac{\partial v_h}{\partial n}\right\rangle_{\Gamma_D} + \frac{\alpha}{h}\langle u_h, v_h\rangle_{\Gamma_D}, \tag{2.11}$$

$$L^{\pm}(v_h) = (f, v_h)_{\Omega_1} + \langle g_N, v_h\rangle_{\Gamma_N} + \left\langle g_D, \frac{\alpha}{h}v_h \pm \frac{\partial v_h}{\partial n}\right\rangle_{\Gamma_D}. \tag{2.12}$$

We are here free to choose the formulation with either sign. Both versions have advantages. In order for a formulation to be well-posed we need the bilinear form $a^{\pm}$ to be coercive,[1] or fulfill the so-called inf-sup condition. Coercivity states that there needs to exist a constant $C > 0$ such that

$$C\|v_h\|_h^2 \leq a^{\pm}(v_h, v_h), \quad \forall v_h \in V_h, \tag{2.13}$$

in some discrete norm $\|\cdot\|_h$. Inf-sup stability can be seen as a slightly weaker condition. For the non-immersed case it turns out that the bilinear form $a^+(\cdot,\cdot)$ is coercive [12] for any $\alpha > 0$ and inf-sup stable [13] even for $\alpha = 0$. Thus, choosing the bilinear form $a^+$ results in a parameter-free method. The disadvantage of the bilinear form $a^+$ is that it is not symmetric:

$$\exists u_h, v_h \in V_h \quad \text{s.t.} \quad a^+(u_h, v_h) \neq a^+(v_h, u_h). \tag{2.14}$$

That the bilinear form discretizes to a symmetric stiffness matrix is quite an attractive property. This is an advantage of using the symmetric bilinear form $a^-$ and the symmetric version of Nitsche's method is used in the majority of the

[1] This is a requirement in the Lax-Milgram lemma, see e.g. [11] page 83.

papers in this thesis. However, as shown in the original paper by Nitsche [10] $a^-$ is only coercive given that one chooses the parameter $\alpha$ to be sufficiently large: $\alpha > C_\alpha$. This is a disadvantage since the smallest constant $C_\alpha$ is usually not known. In practice, $\alpha$ is usually chosen by trial and error. Recently there has been some work [14] on how to avoid the free parameter and still maintain the symmetry in the weak formulation.

In a similar approach as the one above, Nitsche's method can be used to enforce interface conditions. See for example [15, 16].

## 2.2 Lagrange Multipliers

Lagrange multipliers is a general technique for finding the minimum or maximum of a function subject to a constraint. For example, say that we would like to find a point, $(x, y)$, minimizing the function $j(x, y) : \mathbb{R}^2 \to \mathbb{R}$ along a curve $c(x, y) = 0$. In order to solve this we introduce an additional scalar, $\lambda$, (the Lagrange multiplier) and instead find a stationary point $(x, y, \lambda) \in \mathbb{R}^3$ to the function

$$F(x, y, \lambda) = j(x, y) + \lambda c(x, y). \tag{2.15}$$

The same technique can be used to apply Dirichlet boundary conditions in an immersed method. This is done by rewriting (2.1)–(2.3) as an optimization problem. If we look for the solution $u$ in the space

$$H^1_{g_D}(\Omega_1) = \{v \in H^1(\Omega_1) : v|_{\Gamma_D} = g_D\}, \tag{2.16}$$

(2.7) reduces to: Find $u \in H^1_{g_D}(\Omega_1)$ such that

$$(\nabla u, \nabla v)_{\Omega_1} = (f, v)_{\Omega_1} + \langle g_N, v\rangle_{\Gamma_N}, \quad \forall v \in H^1_0(\Omega_1). \tag{2.17}$$

This is now equivalent to the minimization problem[2]: Find $u \in H^1_{g_D}(\Omega_1)$ such that

$$J(u) = \min_{v \in H^1_{g_D}(\Omega_1)} J(v), \tag{2.18}$$

where

$$J(v) := \frac{1}{2}\|\nabla v\|^2_{\Omega_1} - (f, v)_{\Omega_1} - \langle g_N, v\rangle_{\Gamma_N}. \tag{2.19}$$

However, if we introduce a multiplier, $\lambda(x)$, for each point, $x$, on $\Gamma_D$, we can enforce the boundary condition as a constraint:

$$C(u, \lambda) = \int_{\Gamma_D} \lambda(x)(u - g_D)d\Gamma = 0. \tag{2.20}$$

This enables us to look for the solution, $u$, in the space $H^1(\Omega_1)$ instead of $H^1_{g_D}(\Omega_1)$ if we at the same time solve for the multiplier $\lambda(x) \in H^{-\frac{1}{2}}(\Gamma_D)$.

[2] See for example [17] page 189.

Thus, we end up with the problem of finding a stationary point $(u, \lambda)$ to the functional

$$F(v,\xi) = J(v,\xi) + C(v,\xi), \tag{2.21}$$

with $(v,\xi) \in H^1(\Omega_1) \oplus H^{-\frac{1}{2}}(\Gamma_D)$. The problem (2.21) can finally be converted to a weak formulation and solved using finite elements.

There are some disadvantages of using this method. First of all, the multiplier $\lambda(x)$ needs be a member of a finite element space covering $\Gamma_D$. This is illustrated in Figure 2.3, assuming that $\Gamma_D = \Gamma$. This will increase the degrees of freedom of the problem. Moreover, as stated in [18] the formulation does not have very good properties. The resulting system is a saddle point problem. In order to guarantee inf-sup stability, this requires a careful choice of the combination of subspaces, $V_U^h \oplus V_\Lambda^h$, where we look for the discrete solution, $(u_h, \lambda_h) \in V_U^h \oplus V_\Lambda^h$.

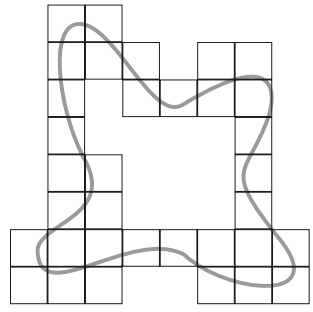

*Figure 2.3.* Illustration of the finite element space for the Lagrange multiplier.

## 2.3 Strong Enforcement

Another approach for imposing the boundary condition is to do it strongly. Here, we search for the solution in a space fulfilling the Dirichlet boundary condition. That is, instead of trying to find the solution in the space $V_h$, we search for it in the space

$$V_h^{g_D} = \{v \in V_h : v|_{\Gamma_D} = g_D\}. \tag{2.22}$$

The construction of this space is non-trivial in an immersed setting. As was discussed in [19], this may also lead to too many constraints imposed on $V_h$ close to the boundary, which in turn results in non-optimal convergence rates. In [19], this problem was solved by exchanging the elements intersected by the boundary to discontinuous Galerkin elements, thus introducing more degrees of freedom near the boundary. This resulted in second order convergence in $L_2$-norm, with piecewise linear basis functions.

Strong enforcement was also used for the hybridizable dG method in Paper V. There, additional degrees of freedom were introduced on the immersed boundary, which were strongly enforced to the boundary value. However, Paper V only considered discretization in 2D and this procedure appears non-trivial to generalize to 3D.

# 3. Degrees of Freedom and Ill-Conditioning

Even if the choice in Figure 2.1 is perhaps the most common way to distribute degrees of freedom, there is one problem with this approach. Consider the case illustrated in Figure 3.1. Here, the element $T$ has a very small intersection with the domain. This turns out to make the resulting matrices in the system of linear equations very ill-conditioned. Since the domain floats on top of the background mesh, the intersection between an element and the domain can be arbitrarily small. Because of this, the smallest eigenvalues of the matrices in the system cannot be bounded from below, and thus the matrices can be arbitrarily ill-conditioned.

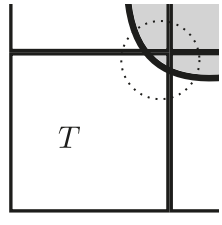


*Figure 3.1.* An element having a small intersection with the immersed domain, $\Omega_1$.

The ill-conditioning is perhaps easiest to understand for the case of symmetric positive definite matrices, which are common for many problems discretized with finite elements. Say that we would like to $L_2$-project a given function, $u$, to the domain, $\Omega_1$, in Figure 1.1. This problem reads: Find $u_h \in V_h$ such that

$$(u_h, v_h)_{\Omega_1} = (u, v_h)_{\Omega_1}, \quad \forall v_h \in V_h. \tag{3.1}$$

If we try to solve this problem naively this will result in a mass matrix, $\mathcal{M} \in \mathbb{R}^{N \times N}$, such that

$$\mathcal{M}_{ij} = (\phi_i, \phi_j)_{\Omega_1}. \tag{3.2}$$

The smallest eigenvalue, $\lambda_{\min}$, can be expressed in terms of the minimum of the Rayleigh quotient:

$$\lambda_{\min} = \min_{z \in \mathbb{R}^N : z \neq 0} \frac{z^T \mathcal{M} z}{z^T z}. \tag{3.3}$$

By fixing $j \in \{1, \ldots, N\}$ and choosing $z$ as $z_i = \delta_{ij}$ we obtain that each eigenvalue is smaller than each diagonal entry:

$$\lambda_{\min} \leq \mathcal{M}_{jj}, \quad j = 1, \ldots, N. \tag{3.4}$$

Since the cut with the background mesh can be arbitrarily small, some diagonal entries can also be arbitrarily small. Thus, the eigenvalues can be arbitrarily close to zero, which will lead to stability problems unless special measures are taken.

The property that we want an immersed method to have, is that the stability and accuracy of the method should be independent of how the intersection with the mesh occurs. There are several ways of dealing with ill-conditioning. Here, we briefly describe the main approaches for trying to overcome this problem.

## 3.1 Preconditioning

A fairly common way to try to remedy problems with ill-conditioning is by preconditioning. Advanced preconditioners have been designed, as in for example [20]. For an interface problem, Reusken [21] proved that for $\mathcal{P}_1$-elements simply scaling the mass matrix, $\mathcal{M}$, by its diagonal, $\mathcal{D}_\mathcal{M}$, makes the condition number of the mass matrix independent of the position of the cut and the grid size. In particular, it was shown that there exist constants, $c_1$ and $c_2$, independent of how the interface cuts the mesh, such that

$$c_1 z^T \mathcal{D}_\mathcal{M} z \leq z^T \mathcal{M} z \leq c_2 z^T \mathcal{D}_\mathcal{M} z, \quad \forall z \in \mathbb{R}^N. \tag{3.5}$$

Diagonal scaling was also used in [22] together with implicit time stepping for time-dependent problems. When using explicit time stepping a desirable property is that the time step restriction should depend on the grid size only, and be independent of how $\Gamma$ cuts the mesh. That is, if $h_{\text{cut}}$ denotes the smallest cut in the mesh then we want to avoid that the time step, $\tau$, must be chosen as

$$\tau \leq c h_{\text{cut}}. \tag{3.6}$$

Unfortunately, this is not true if only preconditioning is used. This restricts the use of preconditioning to implicit time stepping, but depending on the problem under consideration this may be acceptable.

## 3.2 Modifying the Finite Element Space

One strategy to resolve problems with small cuts is to remove or in some way modify basis functions with small support. This can be done in several different ways. Consider, for example, the one-dimensional case in Figure 3.2. Here, the domain is $\Omega_1 = [\Gamma_L, \Gamma_R]$, where the two boundary points $\Gamma_L$ and $\Gamma_R$ are located in between the grid points. Figure 3.2a corresponds to Figure 2.1.

Perhaps the simplest approach to deal with ill-conditioning would be to only solve for those degrees of freedom that lie inside the domain, as illustrated

in Figure 3.2b. Here, each basis function has a large portion of its support inside the domain. Unfortunately, removing basis functions in this way will have a large effect on the error close to the boundary, which will degrade the convergence of the method. In general, the main difficulty when modifying the basis is how to do it without destroying the approximation properties of the finite element space.

A third option is to include those basis functions which have sufficiently large support. In Figure 3.2c the leftmost grid point, $x_0$, falls close to the left boundary, $\Gamma_L$, and the corresponding basis function is included. But the rightmost grid point, $x_{N+1}$, falls far from $\Gamma_R$, and its basis function is excluded. This approach has been used in several papers. In [21], for example, this strategy was used for an interface problem. There, basis functions were removed if they fulfilled

$$\frac{\|\phi\|_{H^l(T\cap\Omega)}}{\|\phi\|_{H^l(T)}} \leq ch_T^\beta, \tag{3.7}$$

where $c$, $\beta$ and $l$ are some user-defined parameters.

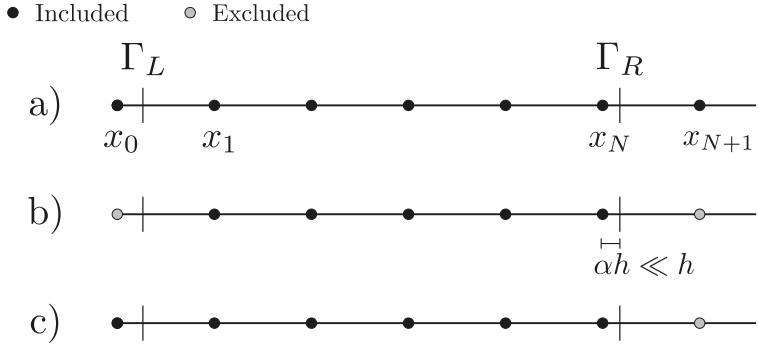


*Figure 3.2.* Three different options for distributing degrees of freedom.

Note that, in all alternatives in Figure 3.2, the basis function associated with each degree of freedom kept its original support. Another alternative is illustrated in Figure 3.3. In this procedure, the element furthest to the right has no basis functions on its own but "borrows" the basis functions from a neighboring element by extrapolating. This is the procedure used in Paper V. A related alternative is illustrated in Figure 3.4. Here the rightmost element has been merged with its neighbor, to form a larger element. These types of strategies are typically referred to as agglomeration and have been used in several papers, such as [23, 24, 25].

Typically, an intersected element is agglomerated to another element if the cut is smaller than some threshold. Often the element to agglomerate to is chosen as the neighbor with the largest intersection. However, since intersected elements in 2D or 3D have more than one neighbor there is more than one choice. It is sometimes not obvious how to choose how the intersected

elements should agglomerate. Whatever method is used to choose such an agglomeration-configuration needs to be flexible enough to handle situations such as those illustrated in Figure 3.5. As indicated by the arrows, in Figure 3.5a an element "wants to" agglomerate to an element, which in turn agglomerates to another element. In Figure 3.5b two elements "wants to" agglomerate to the same element. Generally, it is non-trivial to prove that a good agglomeration-configuration always exists.

One implementational issue with the concept of modifying the finite element space is that we, prior to assembling, need to have some measure of the support of each basis function. This knowledge is needed so that matrices and vectors of appropriate sizes can be allocated.

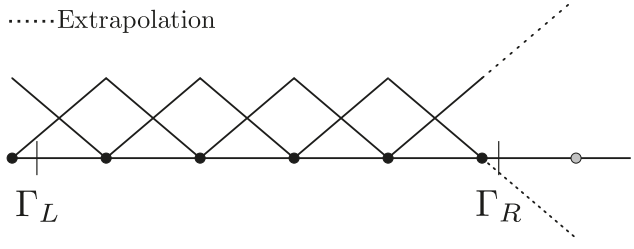


*Figure 3.3.* Extrapolation of basis functions to a neighboring element.

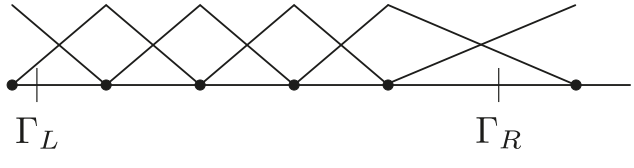


*Figure 3.4.* An element merged with its neighbor to form a larger element.

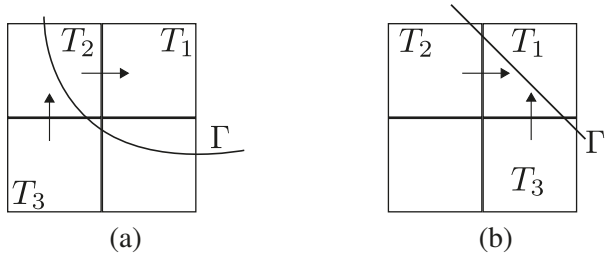


*Figure 3.5.* (a) An element, $T_3$, agglomerating to $T_2$, which in turn agglomerates to $T_1$. (b) Two elements, $T_2$, $T_3$, trying to agglomerate to the same element, $T_1$.

## 3.3 Stabilizing the Weak Formulation

Another approach to overcoming problems with ill-conditioning is to add stabilizing penalty terms to the weak formulation. The idea is that we add a term, $j$, to (3.1) so that the problem instead reads: Find $u_h \in V_h$ such that

$$M(u_h, v_h) = (u, v_h)_{\Omega_1}, \quad \forall v_h \in V_h. \tag{3.8}$$

Here, $M$ is a bilinear form corresponding to the scalar product, but with added stabilization:

$$M(u_h, v_h) = (u_h, v_h)_{\Omega_1} + \gamma_M j(u_h, v_h), \tag{3.9}$$

where $\gamma_M \in \mathbb{R}^+$ is a penalty parameter, and $j$ is defined as

$$j(u, v) = \sum_{F \in \mathcal{F}_\Gamma} \sum_{k=0}^{p} h^{2p+1} \left\langle [\partial_n^k u], [\partial_n^k v] \right\rangle_F. \tag{3.10}$$

In (3.10) $[v]$ denotes the jump over the face, $F$:

$$[v] = v|_{F+} - v|_{F-}, \tag{3.11}$$

and $\partial_n^k u$ denotes the $k$th derivative in the direction of the face normal $n$. $\mathcal{F}_\Gamma$ denotes the internal faces of the elements intersected by the boundary, as illustrated in Figure 3.6. In this way, the stabilization acts in the zone where the problems with the small cuts occur.

With this added stabilization one can show that the bilinear form $M$ is norm-equivalent to the $L_2$-norm on the background mesh. That is, let $\Omega_\mathcal{T}$ denote the domain that is covered by all elements in the background mesh:

$$\Omega_\mathcal{T} = \bigcup_{T \in \mathcal{T}} T. \tag{3.12}$$

then there exist constants, $C_1, C_2 > 0$, that are independent of how the boundary cuts the mesh, such that

$$C_1 \|v_h\|^2_{\Omega_\mathcal{T}} \leq M(v_h, v_h) \leq C_2 \|v_h\|^2_{\Omega_\mathcal{T}}, \quad \forall v_h \in V_h. \tag{3.13}$$

Given this property, the condition number, $\kappa$, of the mass matrix corresponding to the bilinear form $M$ will be bounded by the condition number of the mass matrix over the whole background mesh. That is, if we define the mass matrices as

$$(\mathcal{M}_s)_{ij} = M(\phi_i, \phi_j), \tag{3.14}$$

$$(\mathcal{M}_\mathcal{T})_{ij} = (\phi_i, \phi_j)_{\Omega_\mathcal{T}}. \tag{3.15}$$

then we have that

$$\kappa(\mathcal{M}_s) \leq \frac{C_2}{C_1} \kappa(\mathcal{M}_\mathcal{T}). \tag{3.16}$$

Note that $\kappa(\mathcal{M})$ is now bounded independently of how the boundary cuts the mesh, since the right-hand side in (3.16) does not depend on the cut.

Since the terms in (3.10) are consistent with

$$\partial_n^k u\Big|_{F^+} = \partial_n^k u\Big|_{F^-}, \quad k = 0, \ldots, p \tag{3.17}$$

one can think of the stabilization as penalizing discontinuity. The stabilization in (3.10) was first suggested in [26] and used in [27] for solving the Poisson equation using piecewise linear elements. It has also been used for a number of other problems, as in for example [28, 29, 30, 31]. This stabilization is used in Paper I, II, III and IV.

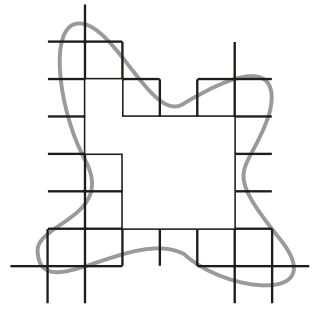

*Figure 3.6.* Faces, $\mathcal{F}_\Gamma$, that the stabilization term act on.

## 3.4 Avoiding Ill-Conditioning in dG Methods

In wave propagation problems discontinuous Galerkin methods are quite popular. One advantage of dG is that the mass matrix becomes block diagonal:

$$\mathcal{M} = \begin{bmatrix} \mathcal{M}_1 & 0 & \ldots & 0 \\ 0 & \mathcal{M}_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \ldots & 0 & \mathcal{M}_N \end{bmatrix}, \tag{3.18}$$

where each block is the local mass matrix for an element in the mesh. This structure makes the system easy to solve since each block can be inverted independently. The jump-stabilization from Section 3.3, unfortunately, destroys this property. Adding jump-stabilization creates couplings between elements that are connected over the faces $\mathcal{F}_\Gamma$ in Figure 3.6. This couples the corresponding blocks and destroys one of the benefits of dG. For this reason, other approaches like agglomeration might be better suited for dG methods. This is pursued in for example [24, 25] and is also used in the dG method in Paper V.

## 3.5 A Comparison of Spaces in XFEM and Cut-FEM

Immersed methods based on finite elements come under a variety of names, such as GFEM [32], XFEM [33], Cut-FEM [34], Finite Cell [35], and PUM [36]. Not all of these methods are specifically immersed methods, but lead to an immersed method by some choice. As pointed out in [33], many of these are very similar. XFEM is one of the most commonly used methods, especially in the applied mechanics community. Since the title of the present thesis includes Cut-FEM and not XFEM we would here like to highlight the difference between the two. The main difference lies in how we look at the subspaces used in the discretization.

In XFEM, the point of view is that one modifies the finite element basis. The idea is to introduce some extra "enriched" degrees of freedom, $a_i$, into our solution ansatz, which will now look like

$$u_h = \sum_{i \in I_O} u_i \phi_i(x) + \sum_{i \in I_E} a_i E(x) \phi_i(x). \tag{3.19}$$

Here, $E(x)$ is the so-called enrichment function. This is chosen depending on the application, but in order to construct an immersed method, it is common to choose the enrichment function to be the Heaviside function. In order to have an example to discuss, assume that we have an interface problem, as illustrated in Figure 3.7. Here, the solution has a discontinuity at the point $\Gamma$.

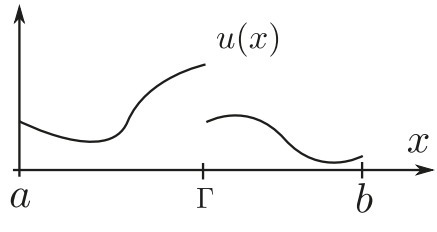


*Figure 3.7.* A 1D function with a discontinuity at the point $\Gamma$.

If we use the standard piecewise linear basis functions over the whole interval $[a, b]$ (as illustrated in Figure 3.8a), the convergence will be suboptimal due to the discontinuity. One way to try to capture this discontinuity is to keep the standard space from Figure 3.8a and add the basis functions illustrated in Figure 3.8b. Here, the basis functions of the element intersected by $\Gamma$ have been enriched by the Heaviside function. This creates the following finite element space

$$V_h^X = \text{span}\{\phi_1, \phi_2, \ldots, \phi_N, H(x-\Gamma)\phi_i(x), H(x-\Gamma)\phi_{i+1}(x)\}. \tag{3.20}$$

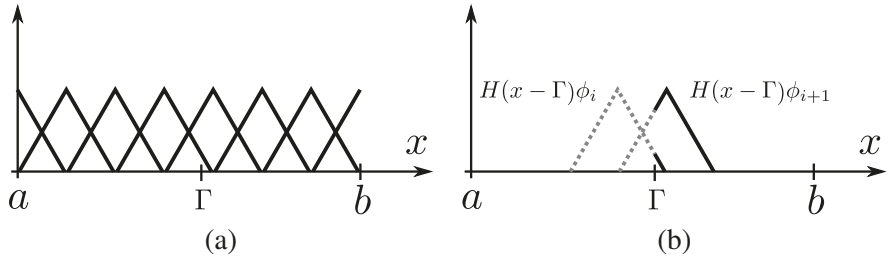


*Figure 3.8.* (a) Standard piecewise linear finite element space. (b) Added enrichment functions.

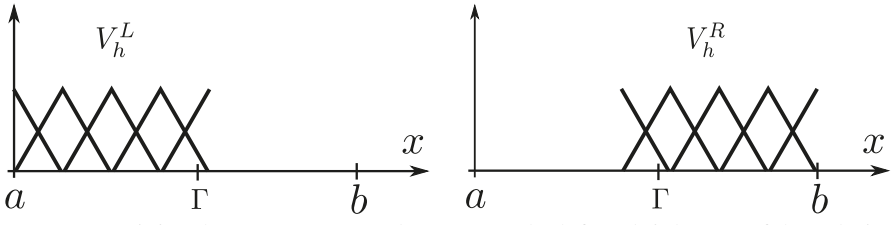


*Figure 3.9.* Finite element spaces used to capture the left and right part of the solution.

The choice of space in Figure 3.8 is perhaps the most natural in the XFEM-framework and was discussed in [33]. Another option, which is more natural in Cut-FEM, is to create a subspace for each of the intervals $[a, \Gamma]$ and $[\Gamma, b]$. As illustrated in Figure 3.9, we could consider each restriction separately:

$$u^L = u|_{x<\Gamma}\,, \tag{3.21}$$

$$u^R = u|_{\Gamma<x}\,, \tag{3.22}$$

and let each be approximated in its own subspace:

$$u_h^L \in V_h^L, \tag{3.23}$$

$$u_h^R \in V_h^R. \tag{3.24}$$

Here, $V_h^L$ and $V_h^R$ are the spaces of piecewise linear functions over all elements that have at least some part inside the intervals $[a, \Gamma]$ and $[\Gamma, b]$. This has been used in several papers, for example in [15, 16].

As mentioned in [37], the approach in Figure 3.8 might be preferable in two or three dimensions in problems where the interface does not split the domain in two but ends inside it. This kind of problem appears for example in fracture mechanics, when the interface describes a crack in a solid. On the other hand, it might be easier to implement the approach in Figure 3.9.

# 4. Implementational Aspects

Implementing an immersed method requires some special techniques. In particular, representing the geometry, categorizing elements and creating quadrature rules. In the following, we will discuss these aspects.

## 4.1 Representing Immersed Geometries

Since the geometry of the domain is no longer described by the mesh, we need some other way to represent it. One approach is to use an additional mesh just for this purpose, as depicted in Figure 4.1. This is usually referred to as overlapping meshes. This can be advantageous if the immersed domain is not deforming, but is only moving by a translation or a rotation. Another example is when the domain is very complicated but can be decomposed into several less complicated subdomains. The subdomains can, for example, be different components in a car engine. Typically, it is easier to mesh each subdomain individually than to mesh the complete domain. When a mesh exists for each subdomain, the domains can be coupled by overlapping the meshes.

Another way of describing the geometry is by a level set function.[1] This is perhaps the most common method and is also used in several of the papers in this thesis. As depicted in Figure 4.2, $\Gamma$ is here described as the zero contour of a scalar function, $\psi : \mathbb{R}^d \to \mathbb{R}$. The different sets in Figure 1.1 are identified in terms of the sign of $\psi$, as

$$\Omega_1 = \{x \in \mathbb{R}^d : \psi(x) < 0\}, \tag{4.1}$$

$$\Gamma = \{x \in \mathbb{R}^d : \psi(x) = 0\}, \tag{4.2}$$

$$\Omega_2 = \{x \in \mathbb{R}^d : \psi(x) > 0\}. \tag{4.3}$$

There are a few benefits of using a level set function. First of all, the description of the domain is conceptually the same regardless of the number of spatial dimensions. Moreover, normals, $n$, and curvatures, $\kappa$, can easily be computed along $\Gamma$, as functions of the gradient and Hessian of the level set function:

$$n = \frac{\nabla\psi}{\|\nabla\psi\|}, \tag{4.4}$$

$$\kappa = \kappa(\nabla\psi, \nabla\nabla\psi). \tag{4.5}$$

[1] For a comprehensive overview see [38] or [39].

In some applications, an important feature is also that topological changes can be handled quite easily. That is, it is possible to handle events such as when two fluid droplets merge into one, or two biological cells break apart.

There are, of course, disadvantages of using a level set function. The level set function is a member of a finite element space, typically over the whole background mesh. This requires a lot of degrees of freedom to represent the geometry. However, this can be made more efficient by the so-called narrow band level set method, where one only keeps a description of the geometry for the elements in a narrow region around the immersed interface. Furthermore, sharp features cannot be represented. Even if the analytic level set function, $\psi$, has corners, the approximation of the level set function, $\psi_h$, will be smooth as soon as it is transferred to the finite element space: $\psi \approx \psi_h \in V_h$.

There are also methods, such as the volume of fluid and the phase field method, which are conceptually similar to the level set method.

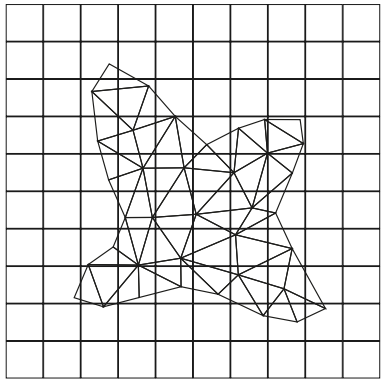

*Figure 4.1.* Two overlapping meshes.

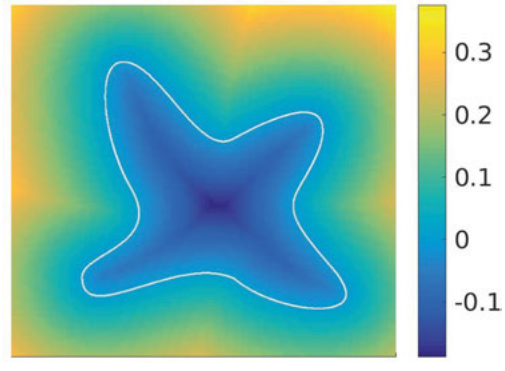


*Figure 4.2.* A level set function describing the geometry. Zero contour in white.

Another option for keeping track of the geometry is so-called marker points. This is illustrated in Figure 4.3. In 2D the approximation of the boundary, $\Gamma_h$, is represented as a sequence of ordered points, $\{x_i\}_{i=1}^{N}$. One problem with marker points is that they can start to cluster if they are advected with time.

In many industrial applications, design of various components in CAD software is often done using NURBS (Non-Uniform Rational B-Spline). A time-consuming problem is then to go from the CAD geometry to a mesh that conforms to the geometry. This is the main motivation for the isogeometric analysis method [40]. For this reason, NURBS is also a useful way to represent an immersed geometry.

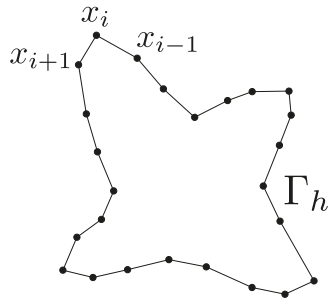


*Figure 4.3.* Illustration of marker points.

## 4.2 Categorizing Elements

Even if the aim of immersed methods is to avoid meshing the domain, there are still procedures that need to be done that relate the domain to the background mesh. Consider again Figure 1.1 and 1.3. As illustrated in Figure 4.4, we need to be able to determine which elements in the mesh lie completely in $\Omega_1$, which are intersected by $\Gamma$, and which ones lie completely in $\Omega_2$. This information is required in order to allocate matrices and vectors of appropriate sizes so that we in the next step can assemble the system. If a level set function is used to describe the geometry of the domain, this categorization can be done easily from the signs of the level set function on each element.

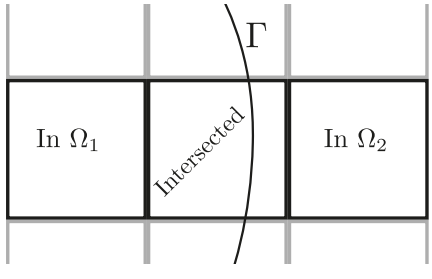


*Figure 4.4.* Elements divided into categories depending on their relation to the immersed geometry.

## 4.3 Quadrature

During the assembly process, we are in need of some algorithm to perform quadrature over the different parts of an element, as in Figure 2.2. One difficulty with high order immersed finite elements is that the order of the quadrature over the intersected elements needs to be sufficiently high. We could, for

example, create a quadrature rule very simply by making a straight line approximation of the boundary, as in Figure 4.6. However, this will not be good enough to obtain a high order of convergence. The quadrature needs to take curved boundaries into account. Performing this higher order quadrature is one of the main difficulties when going to high order immersed methods.

It is not uncommon to actually create a sub-triangulation over a part of the element, as illustrated in Figure 4.5, merely in order to perform the quadrature. This might seem strange when the point of immersed methods is to avoid triangulating the domain. One argument in favor of sub-triangulation is that the triangulation does not need to be of high quality since it is only used to distribute quadrature points over $K \cap \Omega_1$ and $K \cap \Gamma$.

Another way to create quadrature rules is moment fitting, as discussed in for example [41]. Here, the positions of the quadrature points are typically taken to be the same as standard quadrature rules. For each intersected element, a linear system is created that needs to be solved in order to obtain the quadrature weights. A common criticism of moment fitting is that it can lead to both positive and negative weights. Typically, positive weights are preferred since this guarantees that the computed mass matrix is positive definite.

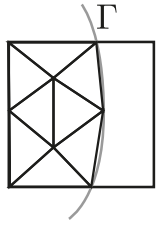


*Figure 4.5.* A sub-triangulation over a part of an element.

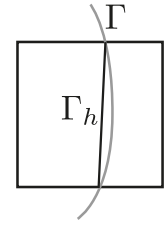


*Figure 4.6.* A straight line approximation, $\Gamma_h$, of $\Gamma$.

What algorithm is used in order to perform quadrature does, of course, depend on how the immersed geometry is represented. Recently there has been a number of articles published on how to generate quadrature rules, given that the domain is described by a level set function [42, 43].

In most of the papers in this thesis, we use the algorithm from [43], which works on hyperrectangles (rectangles in 2D and bricks in 3D). This algorithm is illustrated in Figure 4.7 and is based on finding height functions. On the element $K$ in Figure 4.7 we can express the $x$-coordinate on $\Gamma \cap K$ as a height function, $h(y)$, of the $y$-coordinate:

$$\Gamma \cap K = \{(y, h(y)) : y \in [y_1, y_2]\}. \tag{4.6}$$

Starting from the $y$-direction we find roots along the left and right faces to find the points (here only $y_1$) where the level set function $\psi = 0$ . We then place points in the $y$-direction according to a 1D quadrature rule. From each of these points, we find the root of $\psi$ in the $x$-direction to find points on the

surface $\Gamma$. These will be the quadrature points for the quadrature rule over $\Gamma \cap K$. Quadrature points for the bulk, $\Omega \cap K$, are then placed along each line in the $x$-direction according to the same 1D quadrature rule. Finding a height direction relies on the implicit function theorem, which states that if there is a point $\mathrm{x}_c$ such that

$$\frac{\partial \psi}{\partial x_k}(\mathrm{x}_c) \neq 0, \tag{4.7}$$

we can express $x_k$ as a function of the remaining coordinates in a neighborhood around $\mathrm{x}_c$. The condition (4.7) is checked by calculating the gradient and the Hessian of $\psi$ at the center of the element and approximating $\frac{\partial \psi}{\partial x_k}$ as a linear function. The same algorithm can be used in three dimensions by recursion over the dimension, going from element to faces to lines.

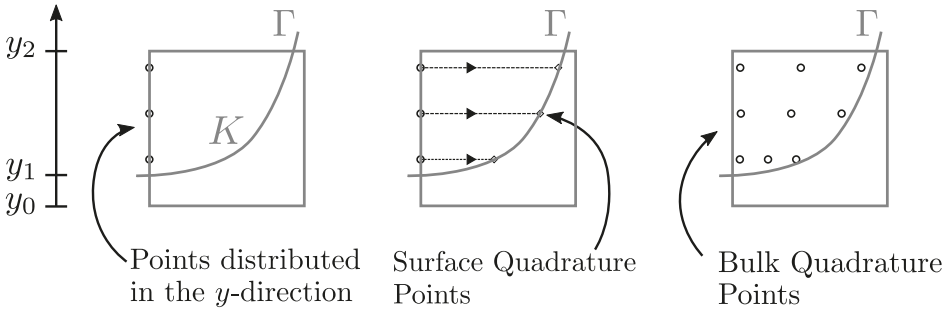


*Figure 4.7.* Illustration of the algorithm from [43].

# 5. Properties of Wave Equations

Since the present thesis is concerned with wave equations we will here discuss some aspects which are important for this type of equation. Consider the acoustic wave equation. This can be written either in second order form:

$$\frac{\partial^2 u}{\partial t^2} = \nabla^2 u + f, \quad x \in \Omega_1, \tag{5.1}$$

$$u = g_D, \qquad x \in \Gamma_D, \tag{5.2}$$

$$\frac{\partial u}{\partial n} = 0, \qquad x \in \Gamma_N, \tag{5.3}$$

or in first order form:

$$\frac{\partial u}{\partial t} = -\nabla \cdot v + F, \quad x \in \Omega_1, \tag{5.4}$$

$$\frac{\partial v}{\partial t} = -\nabla u, \qquad x \in \Omega_1, \tag{5.5}$$

$$u = g_D, \qquad x \in \Gamma_D, \tag{5.6}$$

$$v \cdot n = 0, \qquad x \in \Gamma_N. \tag{5.7}$$

If $\Omega_1 \subset \mathbb{R}^d$ we here have that $u : \Omega_1 \to \mathbb{R}$ and $v : \Omega_1 \to \mathbb{R}^d$. Both formulations should, of course, be supplemented with initial conditions. One should note that the same type of formulation (first and second order form) exists also for the elastic wave equation. One advantage of the second order form is that fewer degrees of freedom are used. Assume that each component in the numerical solution uses $N$ degrees of freedom. The second order form typically uses $2N$ degrees of freedom to store the solution ($N$ for $u_h$ and $N$ for $\frac{\partial u_h}{\partial t}$), while the first order form uses $(1+d)N$. In most of the papers in this thesis, the second order form is used. For some methods based on discontinuous Galerkin, the first order formulation may appear more natural. In for example the hybridizable dG method in Paper V, the first order formulation is used.

Typically high order methods are attributed to being more efficient for wave propagation problems. This is largely motivated by the results in [6]. There the error in phase of finite difference schemes of different orders, $q$, were compared when solving the advection equation. It was concluded that when solving to an end time, $t$, the number of grid points per wavelength, $N_q$, needed to make the error less than $\epsilon$ was

$$N_q \approx C_q \left(\frac{t}{\epsilon}\right)^{1/q}. \tag{5.8}$$

Here, $C_q$ is a constant which only varies slightly with $q$. If we compute for a long time or if we need the error to be small, $t/\epsilon$ is going to be large. Due to the $1/q$ factor in the exponent in (5.8) one can now draw the conclusion that we need much fewer points per wavelength for a high order method than for a low order method. That is, the grid needs to be a lot finer for a lower order method. So for applications where waves propagate over large distances and for a long time higher order methods are going to be more efficient. This is, for example, the case when we have elastic waves in the earth's crust as a result of earthquakes.

When solving time-dependent problems numerically an essential property of a method is stability with respect to time. The most common way to guarantee time stability for hyperbolic problems is to show that there exists a discrete energy, $E_h$, with bounded growth. That is, there exists a bound such that

$$E_h(t) \leq C(f, g_D, t), \tag{5.9}$$

where $C$ is some constant which depends on the problem data. In the continuous setting a similar energy,[1] $E$, is often used. This energy is used as a tool to show properties of the equation (see e.g. [44]), for example, whether it has a unique solution or not. For the acoustic wave equation, the energy is usually chosen as

$$E(t) = \frac{1}{2}\left(\|u_t\|^2_{\Omega_1} + \|\nabla u\|^2_{\Omega_1}\right), \tag{5.10}$$

but the energy is different depending on which equation is considered. The discrete energy is typically similar to the continuous energy but may include additional terms appearing from the discretization. However, a requirement is that given (5.9) we should be able to bound the growth of the numerical solution, $u_h$. In the context of immersed methods, we want to be able to bound $u_h$ over the whole background mesh:

$$\|u_h(t)\|_{\Omega_\mathcal{T}} \leq C(f, g_D, t), \tag{5.11}$$

where $\Omega_\mathcal{T}$ was defined in (3.12). Estimates like the ones in (5.9) and (5.11) are shown for the method in Paper I and II using a similar energy as the one in (5.10).

[1] Note that $E$ does not need to correspond to an energy in the physical sense.

# Sammanfattning på svenska

Vågor uppträder i många olika former i naturen. Ljud är vågor i luften, en jordbävning skapar vågor i marken, och vi skickar information runt hela världen med hjälp av elektromagnetiska vågor. I flera tillämpningar vill man veta hur vågor propagerar. När vi har vågor i fasta material är vi intresserade av förskjutningar. Om man betraktar en given punkt i materialet så kommer punkten flytta sig lite (förskjutas) när vågen träffar den. Problemet man försöker lösa består i att ta reda på hur varje punkt i ett material förskjuts under en given tid. Hur förskjutningarna uppträder beskrivs matematiskt med en ekvation i klassen partiella differentialekvationer. Dessa går i allmänhet inte att lösa exakt (det vill säga genom att använda papper och penna). Däremot kan man få ut lösningen approximativt med hjälp av så kallade numeriska metoder. Dessa formuleras för hand men implementeras och löses med hjälp av datorer. Området för denna avhandling är beräkningsvetenskap. Detta område handlar om hur man konstruerar och implementerar numeriska metoder för att lösa matematiska problem. Det finns flera olika numeriska metoder för att lösa partiella differentialekvationer. Varje metod har egna för- och nackdelar. Denna avhandling undersöker en speciell klass av så kallade finita elementmetoder.

Numeriska metoder använder i allmänhet någon typ av beräkningsnät. Nät består oftast av trianglar eller fyrhörningar som sitter ihop på ett sätt så att de tillsammans bildar formen för det område man vill räkna på. Antag att vi är intresserade av att lösa ett problem på området $\Omega_1$ i Figur 1.1 på sidan 8. Ett exempel på ett nät för detta område visas i Figur 1.2. Förenklat kan man säga att när man löser ekvationen med en numerisk metod får man ut hur hörnen på varje triangel förskjuts när vågen kommer.

Den klass av finita elementmetoder som denna avhandling diskuterar är metoder med så kallade inbäddade ränder. Detta illustreras i Figur 1.3. Alla fyrhörningar i nätet har nu samma storlek och form, och geometrin "flyter" ovanpå nätet. Metoder av denna typ kallas även skurna metoder, eftersom geometrins rand (kant) skär genom beräkningsnätet.

Varför man skulle vilja lösa vågutbredningsproblem med skurna metoder är inte uppenbart. Motiveringen för detta kommer i stor utsträckning från problem där man på förhand inte vet vilken form ett visst objekt har. Betrakta exemplet i Figur 1.5 på sidan 9. Vi har ett objekt med okänd form som vi vill bestämma. Det kan till exempel röra sig om vävnad i en medicinsk tillämpning. För att bestämma formen skickar vi en laserpuls mot objektet, vilket gör att det värms upp och expanderar. Expansionen skapar akustiska vågor som propagerar från objektet. Hur dessa akustiska vågor ser ut kommer bero på

objektets form, och genom att mäta dessa vågor kan man bestämma denna. Detta kräver dock att man upprepade gånger provar olika form på objektet.

Om man successivt ändrar form på området man räknar på och använder ett traditionellt nät (Figur 1.2) så kan trianglarna i nätet bli väldigt förvrängda. Detta illustreras i Figur 1.4. Många av trianglarna har blivit avlånga och har två sidor som är betydligt längre än den tredje. Detta kommer göra att den numeriska lösningen får större fel än om alla trianglar i nätet hade varit ungefär liksidiga. Ett sätt att komma runt detta problem är att använda inbäddade ränder. Om geometrin flyter ovanpå nätet på samma sätt som i Figur 1.3 så kan nätet inte bli förvrängt.

Genom att använda inbäddade ränder har man därmed löst ett problem men istället skapat nya problem. Betrakta återigen Figur 1.3. Ett problem är att det finns fyrhörningar som ligger helt utanför området $\Omega_1$. Detta kan man dock lösa på det sätt som illustreras i Figur 2.1 på sidan 10. Här används enbart det minsta antal fyrhörningar som krävs för att helt täcka området $\Omega_1$.

Geometrins rand kan förstås skära genom nätet på ett godtyckligt sätt. Detta gör att vissa av de fyrhörningar som är skurna enbart har en mycket liten del av sig som ligger inuti $\Omega_1$. Detta illustreras i Figur 3.1 på sidan 16. På grund av detta kommer de hörn som ligger utanför $\Omega_1$ vara dåligt bestämda. Detta betyder att de hörn som ligger utanför $\Omega_1$ kan förskjutas mycket utan att det har en stor påverkan på lösningen inuti $\Omega_1$. Hur man kommer runt detta problem (och vilka nya problem som uppstår när man gör det) diskuteras i detalj i Manuskript II i denna avhandling.

Som tidigare nämndes är den numeriska lösningen till ekvationen approximativ och den kommer därmed ha ett fel i sig. Givetvis vill man att detta fel ska vara så litet som möjligt. Ju finare nät (det vill säga ju fler trianglar eller fyrhörningar) man använder desto mindre fel får man i lösningen. Men ju finare nätet är desto mer tid och energi behöver en dator för att lösa problemet. Med en bra numerisk metod minskar felet snabbt när man gör nätet finare. Ju snabbare felet minskar desto högre noggrannhetsordning säger man att metoden har. När man är intresserad av vågutbredning visar det sig att det är mer effektivt att använda numeriska metoder med hög ordning. Om man alltså vill att felet ska vara mindre än någon given tolerans kommer man få fram lösningen på kortare tid om man använder en metod med hög ordning istället för en metod med låg ordning. Av denna anledning handlar denna avhandling om skurna finita elementmetoder med hög noggrannhetsordning.

# Acknowledgements

First of all, I want to thank my advisor Gunilla Kreiss, for introducing me to the field of immersed methods, for your encouragement, and for all the advice. Your never failing enthusiasm and (sometimes ridiculously) positive attitude is probably good for me. Further, I would like to thank all of my international coauthors: Svenja Schoeder, Martin Kronbichler, Kyle Steffen, Qing Xia and Yekaterina Epshteyn. Thank you also, to all my current and old colleagues at TDB, for creating such a great place to work.

I would like to thank the Swedish Research Council for providing funding for the research in this thesis, through grant 2014-6088. I am also grateful for travel grants from Sederholms Nordiska, ÅForsk, Liljewalchs, and STINT.

Thank you, to all my friends that have supported me during this time. You all mean a lot. To the WP-group: Martin Almquist, Siyang Wang, Jonatan Werpers, Gustav Ludvigsson and Ylva Rydin, thank you for all the discussions, and for together creating a context where research is fun. I am lucky to have friends that can also tell me which of my 3rd derivatives are discontinuous. Also, special thanks to my "offie"-mates: Hanna Holmgren, Slobodan Milovanović and Kristiina Ausmees; and to Timofey Mukha, Fredrik Hellman and Tilda Thöresson. Thank you all for being great friends and support.

Sist men inte minst, tack till min familj: Mamma, Pappa, Joel och Victor. Tack för att ni finns.

# References


[1] S. Sticko and G. Kreiss. A Stabilized Nitsche Cut Element Method for the Wave Equation. *Computer Methods in Applied Mechanics and Engineering*, 309:364–387, September 2016. ISSN: 00457825.

[2] S. Sticko and G. Kreiss. Higher Order Cut Finite Elements for the Wave Equation. *ArXiv e-prints: 1608.03107v2*, 2018. (Submitted).

[3] S. Sticko, G. Ludvigsson, and G. Kreiss. High Order Cut Finite Elements for the Elastic Wave Equation. *ArXiv e-prints: 1804.00332*, 2018. (Submitted).

[4] G. Ludvigsson, K. R. Steffen, S. Sticko, S. Wang, Q. Xia, Y. Epshteyn, and G. Kreiss. High-Order Numerical Methods for 2D Parabolic Problems in Single and Composite Domains. *Journal of Scientific Computing*:1–36, January 2018. ISSN: 0885-7474, 1573-7691.

[5] S. Schoeder, S. Sticko, G. Kreiss, and M. Kronbichler. High Order Cut Discontinuous Galerkin Methods with Local Time Stepping for Acoustics, 2017. (In revision).

[6] H.-O. Kreiss and J. Oliger. Comparison of Accurate Methods for the Integration of Hyperbolic Equations. *Tellus*, 24(3):199–215, 1972. ISSN: 2153-3490.

[7] C. S. Peskin. Numerical Analysis of Blood Flow in the Heart. *Journal of Computational Physics*, 25(3):220–252, November 1977. ISSN: 00219991.

[8] C. S. Peskin. The Immersed Boundary Method. *Acta Numerica*, 11, January 2002. ISSN: 0962-4929, 1474-0508.

[9] D. A. Di Pietro and A. Ern. *Mathematical Aspects of Discontinuous Galerkin Methods*. (69) of *Mathématiques et Applications*. Springer, Berlin, Heidelberg, 2012. ISBN: 978-3-642-22979-4.

[10] J. Nitsche. Über ein Variationsprinzip zur Lösung von Dirichlet-Problemen bei Verwendung von Teilräumen, die keinen Randbedingungen unterworfen sind. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 36:9–15, 1971. ISSN: 1865-8784.

[11] A. Ern and J.-L. Guermond. *Theory and Practice of Finite Elements*. Volume 159 of *Applied Mathematical Sciences*. Springer, New York, 2004. ISBN: 978-1-4419-1918-2, 978-1-4757-4355-5.

[12] J. Freund and R. Stenberg. On Weakly Imposed Boundary Conditions for Second Order Problems. In *Proceedings of the Ninth International Conference on Finite Elements in Fluids*. Università di Padova, 1995, pages 327–336.

[13] E. Burman. A Penalty-Free Nonsymmetric Nitsche-Type Method for the Weak Imposition of Boundary Conditions. *SIAM Journal on Numerical Analysis*, 50(4):1959–1981, January 2012. ISSN: 0036-1429, 1095-7170.

[14] C. Lehrenfeld. Removing the Stabilization Parameter in Fitted and Unfitted Symmetric Nitsche Formulations. In *Proceedings of the VII European Congress on Computational Methods in Applied Sciences and Engineering*. National Technical University of Athens, 2016, pages 373–383. ISBN: 978-618-82844-0-1.

[15] A. Hansbo and P. Hansbo. An Unfitted Finite Element Method, Based on Nitsche's Method, for Elliptic Interface Problems. *Computer Methods in Applied Mechanics and Engineering*, 191(47):5537–5552, 2002. ISSN: 00457825.

[16] P. Hansbo, M. G. Larson, and S. Zahedi. A Cut Finite Element Method for a Stokes Interface Problem. *Applied Numerical Mathematics*, 85:90–114, November 2014. ISSN: 01689274.

[17] M. G. Larson and F. Bengzon. *The Finite Element Method: Theory, Implementation, and Applications*. Volume 10 of *Texts in Computational Science and Engineering*. Springer, Berlin, Heidelberg, 2013. ISBN: 978-3-642-33286-9, 978-3-642-33287-6.

[18] S. Fernández-Méndez and A. Huerta. Imposing Essential Boundary Conditions in Mesh-Free Methods. *Computer Methods in Applied Mechanics and Engineering*, 193(12-14):1257–1275, March 2004. ISSN: 00457825.

[19] A. J. Lew and G. C. Buscaglia. A Discontinuous-Galerkin-Based Immersed Boundary Method. *International Journal for Numerical Methods in Engineering*, 76(4):427–454, October 2008. ISSN: 00295981.

[20] C. Lehrenfeld and A. Reusken. Optimal Preconditioners for Nitsche-XFEM Discretizations of Interface Problems. *Numerische Mathematik*, March 2016. ISSN: 0029-599X, 0945-3245.

[21] A. Reusken. Analysis of an Extended Pressure Finite Element Space for Two-Phase Incompressible Flows. *Computing and Visualization in Science*, 11(4-6):293–305, September 2008. ISSN: 1432-9360, 1433-0369.

[22] E. Burman and M. A. Fernández. An Unfitted Nitsche Method for Incompressible Fluid–Structure Interaction Using Overlapping Meshes. *Computer Methods in Applied Mechanics and Engineering*, 279:497–514, September 2014. ISSN: 00457825.

[23] A. Johansson and M. G. Larson. A High Order Discontinuous Galerkin Nitsche Method for Elliptic Problems with Fictitious Boundary. *Numerische Mathematik*, 123(4):607–628, April 2013. ISSN: 0029-599X, 0945-3245.

[24] F. Kummer. Extended Discontinuous Galerkin Methods for Two-Phase Flows: The Spatial Discretization. *International Journal for Numerical Methods in Engineering*, 109(2):259–289, January 2017. ISSN: 00295981.

[25] B. Müller, S. Krämer-Eis, F. Kummer, and M. Oberlack. A High-Order Discontinuous Galerkin Method for Compressible Flows with Immersed Boundaries. *International Journal for Numerical Methods in Engineering*, 2016. ISSN: 00295981.

[26] E. Burman. Ghost Penalty. *Comptes Rendus Mathematique*, 348(21-22):1217–1220, November 2010. ISSN: 1631073X.

[27] E. Burman and P. Hansbo. Fictitious Domain Finite Element Methods Using Cut Elements: II. A Stabilized Nitsche Method. *Applied Numerical Mathematics*, 62(4):328–341, April 2012. ISSN: 01689274.

[28] A. Massing, M. G. Larson, A. Logg, and M. E. Rognes. A Stabilized Nitsche Fictitious Domain Method for the Stokes Problem. *Journal of Scientific Computing*, 61(3):604–628, December 2014. ISSN: 0885-7474, 1573-7691.

[29] E. Burman, P. Hansbo, M. G. Larson, and S. Zahedi. Cut Finite Element Methods for Coupled Bulk–Surface Problems. *Numerische Mathematik*, 133(2):203–231, June 2016. ISSN: 0029-599X, 0945-3245.

[30] P. Hansbo, M. G. Larson, and S. Zahedi. A Cut Finite Element Method for a Stokes Interface Problem. *Applied Numerical Mathematics*, 85:90–114, November 2014. ISSN: 01689274.

[31] P. Hansbo, M. G. Larson, and S. Zahedi. Characteristic Cut Finite Element Methods for Convection–Diffusion Problems on Time Dependent Surfaces. *Computer Methods in Applied Mechanics and Engineering*, 293:431–461, August 2015. ISSN: 00457825.

[32] T. Strouboulis, K. Copps, and I. Babuška. The Generalized Finite Element Method. *Computer Methods in Applied Mechanics and Engineering*, 190(32-33):4081–4193, May 2001. ISSN: 00457825.

[33] T.-P. Fries and T. Belytschko. The Extended/Generalized Finite Element Method: An Overview of the Method and its Applications. *International Journal for Numerical Methods in Engineering*, August 2010. ISSN: 00295981.

[34] E. Burman, S. Claus, P. Hansbo, M. G. Larson, and A. Massing. CutFEM: Discretizing Geometry and Partial Differential Equations. *International Journal for Numerical Methods in Engineering*, 104(7):472–501, November 2015. ISSN: 00295981.

[35] M. Dauge, A. Düster, and E. Rank. Theoretical and Numerical Investigation of the Finite Cell Method. *Journal of Scientific Computing*, 65(3):1039–1064, December 2015. ISSN: 0885-7474, 1573-7691.

[36] I. Babuška and J. M. Melenk. The Partition of Unity Method. *International Journal for Numerical Methods in Engineering*, 40(4):727–758, February 1997. ISSN: 0029-5981, 1097-0207.

[37] P. M. Areias and T. Belytschko. A Comment on the Article "A Finite Element Method for Simulation of Strong and Weak Discontinuities in Solid Mechanics" by A. Hansbo and P. Hansbo [Comput. Methods Appl. Mech. Engrg. 193 (2004) 3523–3540]. *Computer Methods in Applied Mechanics and Engineering*, 195(9-12):1275–1276, February 2006. ISSN: 00457825.

[38] J. A. Sethian. *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science*. (3) of *Cambridge Monographs on Applied and Computational Mathematics*. Cambridge University Press, New York, 2nd edition, 1999. ISBN: 978-0-521-64204-0, 978-0-521-64557-7.

[39] S. Osher and R. P. Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*. (v. 153) of *Applied Mathematical Sciences*. Springer, New York, 2003. ISBN: 978-0-387-95482-0.

[40] T. J. R. Hughes, J. A. Cottrell, and Y. Bazilevs. Isogeometric Analysis: CAD, Finite Elements, NURBS, Exact Geometry and Mesh Refinement. *Computer Methods in Applied Mechanics and Engineering*, 194(39–41):4135–4195, October 2005. ISSN: 0045-7825.

[41] B. Müller, F. Kummer, and M. Oberlack. Highly Accurate Surface and Volume Integration on Implicit Domains by Means of Moment-Fitting. *International Journal for Numerical Methods in Engineering*, 96(8):512–528, November 2013. ISSN: 00295981.

[42] C. Lehrenfeld. High Order Unfitted Finite Element Methods on Level Set Domains Using Isoparametric Mappings. *Computer Methods in Applied Mechanics and Engineering*, 300:716–733, March 2016. ISSN: 00457825.

[43] R. I. Saye. High-Order Quadrature Methods for Implicitly Defined Surfaces and Volumes in Hyperrectangles. *SIAM Journal on Scientific Computing*, 37(2):A993–A1019, January 2015. ISSN: 1064-8275, 1095-7197.

[44] Y. Pinchover and J. Rubinstein. *Introduction to Partial Differential Equations*. Cambridge University Press, New York, 2005. ISBN: 978-0-521-84886-2, 978-0-521-61323-1.

Acta Universitatis Upsaliensis

*Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology* 1656

Editor: The Dean of the Faculty of Science and Technology

A doctoral dissertation from the Faculty of Science and Technology, Uppsala University, is usually a summary of a number of papers. A few copies of the complete dissertation are kept at major Swedish research libraries, while the summary alone is distributed internationally through the series Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology. (Prior to January, 2005, the series was published under the title "Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology".)

Distribution: publications.uu.se
urn:nbn:se:uu:diva-347439

ACTA
UNIVERSITATIS
UPSALIENSIS
UPPSALA
2018