

CutFEM: Discretizing geometry and partial differential equations

Erik Burman^{1,*}, Susanne Claus¹, Peter Hansbo², Mats G. Larson³ and André Massing⁴

¹*Department of Mathematics, University College London, London, UK-WC1E 6BT, UK*

²*Department of Mechanical Engineering, Jönköping University, SE-55111 Jönköping, Sweden*

³*Department of Mathematics and Mathematical Statistics, Umeå University, SE-901 87 Umeå, Sweden*

⁴*Center for Biomedical Computing, Simula Research Laboratory, PO Box 134, NO-1325 Lysaker, Norway*

SUMMARY

We discuss recent advances on robust unfitted finite element methods on cut meshes. These methods are designed to facilitate computations on complex geometries obtained, for example, from computer-aided design or image data from applied sciences. Both the treatment of boundaries and interfaces and the discretization of PDEs on surfaces are discussed and illustrated numerically. © 2014 The Authors. *International Journal for Numerical Methods in Engineering* published by John Wiley & Sons Ltd.

Received 14 March 2014; Revised 4 September 2014; Accepted 29 September 2014

KEY WORDS: extended finite element method; unfitted methods; finite element methods; meshfree methods; Galerkin; level sets; stability

1. INTRODUCTION

Research in numerical methods for solving problems in science and engineering has mainly focused on techniques for approximating models described by partial differential equations (PDEs), while the important coupling to the geometrical description of the domain has been largely overlooked. In recent years, the need for a unified approach has been recognized, and this area is today receiving rapidly increasing interest. This interest is motivated by the demand for efficient and robust techniques for solving problems involving complex and evolving geometries. The use of geometric descriptions in the computational method, that are closely linked to the geometric data acquisition, can dramatically reduce the computational cost of preprocessing, or transformation, of acquired geometry descriptions into representations suitable for the computational method at hand.

For instance, the simulation of blood flow dynamics in vessel geometries requires a series of highly non-trivial steps to generate a high quality, full 3D finite element mesh from biomedical image data [1]. Similar challenging and computationally costly preprocessing steps are required to transform geological image data into conforming domain discretizations, which respect complex structures such as faults and large scale networks of fractures; see, for instance, [2].

An example of a successful paradigm for the integration of geometry and computation is given by the isogeometric analysis pioneered by Hughes and co-workers [3]. Here, the merging of computation and geometry is obtained by adopting the functions used for geometry representation as basis for the computational method leading to new approaches in the discretization of PDEs.

The idea behind CutFEM is to make the discretization as independent as possible of the geometric description and minimize the complexity of mesh generation, while retaining the accuracy and

*Correspondence to: Erik Burman, Department of Mathematics, University College London, London, UK-WC1E 6BT, UK.

†E-mail: e.burman@ucl.ac.uk

This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.

robustness of a standard finite element method. In particular, we will show later how recent stabilization techniques can be applied to make both the accuracy of the approximation and the system condition number independent of the mesh/boundary intersection and physical parameters. Thanks to this robustness of the discretization, powerful linear algebra techniques developed for finite element methods can be made to bear on the solution of the linear systems obtained by the CutFEM discretization.

In the CutFEM approach, the boundary of a given domain is represented on a background grid, for instance, using a level set function. The background grid is then also used to represent the approximate solution of the governing PDEs. CutFEM builds on a general finite element formulation for the approximation of PDEs, in the bulk and on surfaces, that can handle elements of complex shape and where boundary and interface conditions are built into the discrete formulation. This way, CutFEM can ease the burden of mesh generation by requiring only a low-quality and even non-conform surface mesh representation of the computational geometry. The integration of the geometry in the discrete formulation leads to a method that can be applied equally well to CAD generated geometries and to geometries obtained from biomedical or geological image data.

In this paper, we give some examples of how CutFEM combined with Nitsche's method [4] is implemented for a range of problems with increasing complexity. The use of Nitsche's method for unfitted interface problems and fictitious domain methods has been developed in [5–13, 28, 29, 48]. Other related approaches based on Lagrange multipliers or discontinuous Galerkin methods have been suggested in the following works [14–20].

The paper is organized as follows. In Section 2, we give an overview of some archetypal problems with associated CutFEM discretizations. Then we will discuss the crucial question of robustness in Section 3. The representation of the geometry using level sets is discussed in Section 4 and implementation issues are reviewed in Section 5. A series of numerical illustrations for different coupling problems on non-trivial geometries are presented in Section 6.

2. NITSCHÉ'S METHOD FOR INTERFACE AND DIRICHLET BOUNDARY VALUE PROBLEMS

2.1. A Poisson model problem

Let Ω be a bounded domain in two or three space dimensions, with an interface Γ dividing Ω into two non-overlapping subdomains Ω_1 and Ω_2 , so that $\Omega := \Omega_1 \cup \Omega_2 \cup \Gamma$, with the interface defined by $\Gamma = \overline{\Omega_1} \cap \overline{\Omega_2}$. For simplicity, we assume that the subdomains are polyhedral (or polygonal in \mathbb{R}^2) and that Γ is polygonal (or a broken line).

For any sufficiently regular function u in $\Omega_1 \cup \Omega_2$, we define the jump of u on Γ by $\llbracket u \rrbracket := u_1|_{\Gamma} - u_2|_{\Gamma}$, where $u_i = u|_{\Omega_i}$ is the restriction of u to Ω_i . Conversely, for u_i defined in Ω_i , we identify the pair (u_1, u_2) with the function u , which equals u_i on Ω_i . For definiteness, we define \mathbf{n} as the outward pointing unit normal to Ω_1 on Γ . We shall consider a problem with piecewise constant data α so that $\alpha_i = \alpha|_{\Omega_i}$, and a reaction coefficient $c(x) \geq 0$ that may be discontinuous across Γ . Our first model problem is the following variant of Poisson's equation:

$$\begin{aligned} -\nabla \cdot (\alpha \nabla u) + cu &= f \quad \text{in } \Omega_1 \cup \Omega_2, \\ u &= 0 \quad \text{on } \partial\Omega, \\ \llbracket u \rrbracket &= 0 \quad \text{on } \Gamma, \\ \llbracket \mathbf{q} \cdot \mathbf{n} \rrbracket &= 0 \quad \text{on } \Gamma, \end{aligned} \tag{1}$$

where we used the flux vectors $\mathbf{q}_i := -\alpha_i \nabla u_i$.

In a standard finite element method, the jump in the normal derivative resulting from the continuity of the flux $\mathbf{q} := -\alpha \nabla u$, when $\alpha_1 \neq \alpha_2$, can be taken into account by letting Γ coincide with mesh lines. In [5, 7], another approach was taken in that (1) was solved approximately using piecewise polynomial finite elements on a family of conforming shape regular triangulations \mathcal{T}_h of

Ω that were independent of the location of the interface Γ . The approximation was then allowed to be discontinuous inside elements, which intersected the interface.

To recall this method, we will use the following notation for mesh related quantities. For any element K , let $K_i = K \cap \Omega_i$ denote the part of K in Ω_i . By $\mathcal{G}_h^* := \{K \in \mathcal{T}_h : K \cap \Gamma \neq \emptyset\}$, we here denote the set of elements that are intersected by the interface. For an element $K \in \mathcal{G}_h^*$, let $\Gamma_K := \Gamma \cap K$ be the part of Γ in K .

We shall seek a discrete solution $U = (U_1, U_2)$ in the space $V^h = V_1^h \times V_2^h$, where

$$V_i^h = \{v_i \in H^1(\Omega_i) : v_i|_{K_i} \text{ is linear, } v_i|_{\partial\Omega} = 0\}.$$

As Γ may intersect two edges of a triangle arbitrarily, the size of the parts K_i is not fully characterized by the meshsize parameters. Thus, to guarantee stability of this method using elements with *internal* discontinuities, further conditions on the combinations of numerical fluxes (co-normal derivatives) must be imposed by choosing appropriate mesh and geometry dependent weights κ . The approach suggested in [5, 7] was to choose the numerical fluxes by introducing weights

$$\kappa_i|_K = \frac{\text{meas}(K_i)}{\text{meas}(K)}, \quad (2)$$

where $\text{meas}(K)$ denotes the size (area or volume) of K , and to use a weighted mean

$$\langle \alpha \partial_n v \rangle := (\alpha_1 \kappa_1 \partial_n v_1 + \alpha_2 \kappa_2 \partial_n v_2)|_{\Gamma}, \quad (3)$$

where $\partial_n := \mathbf{n} \cdot \nabla$, as the flux on Γ (note that $\kappa_1 + \kappa_2 = 1$). Thus, for an intersected element, a mean numerical flux that takes the different meshsizes into account was computed. However, for problems in which there is a very large difference between the parameters α_i , this approach is not robust. To improve robustness, the weighted average must also take into account the parameters: in (3), we must change the definitions of the κ_i to

$$\kappa_1|_K := \frac{\alpha_2 \text{meas}(K_1)}{\alpha_2 \text{meas}(K_1) + \alpha_1 \text{meas}(K_2)}, \quad \kappa_2|_K := \frac{\alpha_1 \text{meas}(K_2)}{\alpha_2 \text{meas}(K_1) + \alpha_1 \text{meas}(K_2)} \quad (4)$$

cf. [21, 22].

The method is defined by the variational problem of finding $U \in V^h$ such that

$$a_h(U, v) = L(v), \quad \forall v \in V^h, \quad (5)$$

where

$$\begin{aligned} a_h(U, v) := & \sum_i (\alpha_i \nabla U_i, \nabla v_i)_{\Omega_i} - (\llbracket U \rrbracket, \langle \alpha \partial_n v \rangle)_{\Gamma} \\ & - (\langle \alpha \partial_n U \rangle, \llbracket v \rrbracket)_{\Gamma} + (\gamma h^{-1} \llbracket U \rrbracket, \llbracket v \rrbracket)_{\Gamma} \end{aligned}$$

with $\gamma \in \mathbb{R}^+$ and

$$L(v) := \sum_i (f, v_i)_{\Omega_i}.$$

For stability, one must take γ sufficiently large. Two choices of γ have been proposed in the literature for these weights. In [21], it was suggested to take

$$\gamma|_K = \hat{\gamma} h_K \frac{m_{K\Gamma}}{m_{K\Omega_1}/\alpha_1 + m_{K\Omega_2}/\alpha_2} \quad (6)$$

and in [22], the choice

$$\gamma|_K = \hat{\gamma} h_K \max \left\{ \alpha_1 \frac{m_{K_{\Omega_1}}}{m_K}, \alpha_2 \frac{m_{K_{\Omega_2}}}{m_K} \right\} \frac{m_{K_\Gamma}}{m_K}, \quad (7)$$

was advocated. Above $\hat{\gamma}$ is a mesh and parameter independent constant and

$$m_{K_\Gamma} := \text{meas}_{d-1}(K \cap \Gamma), \quad m_{K_{\Omega_i}} := \text{meas}_d(K \cap \Omega_i), \quad i = 1, 2 \quad m_K := \text{meas}_d(K).$$

With these definitions, the discrete problem (5) is consistent in the sense that, for u solving (1),

$$a_h(u, v) = L(v), \quad \forall v \in V^h. \quad (8)$$

In Section 3, we will discuss the consequences of the different choices of the penalty parameter and an alternative route to robustness inspired by fictitious domain methods.

A FE basis for V^h is easily obtained from a standard FE basis on the mesh by the introduction of new basis functions for the elements that intersect Γ . Thus, we replace each standard basis function living on an element that intersects the interface by two new basis functions, namely its restrictions to Ω_1 and Ω_2 , respectively. The collection of basis functions with support in Ω_i is then clearly a basis for V_i^h , and hence, we obtain a basis for V^h by the identification $\psi = (\psi|_{\Omega_1}, \psi|_{\Omega_2})$. If the interface coincides exactly with an element edge, no new basis functions are introduced on these elements, but the approximating functions may still be discontinuous over such an edge. As a consequence, there are six non-zero basis functions on each element that properly intersects Γ . Perhaps this process is most easily seen as creating two new separate meshes with doubling of the elements crossed by the interface (Figure 1).

We now want to show that the approximation property of V^h is optimal in the following mesh dependent norm:

$$\|v\|_h^2 := \sum_i \|\nabla v_i\|_{L_2(\Omega_i)}^2 + \sum_{K \in \mathcal{G}_h^*} h_K \|\langle \partial_n v \rangle\|_{L_2(\Gamma_K)}^2 + \sum_{K \in \mathcal{G}_h^*} h_K^{-1} \|\llbracket v \rrbracket\|_{L_2(\Gamma_K)}^2.$$

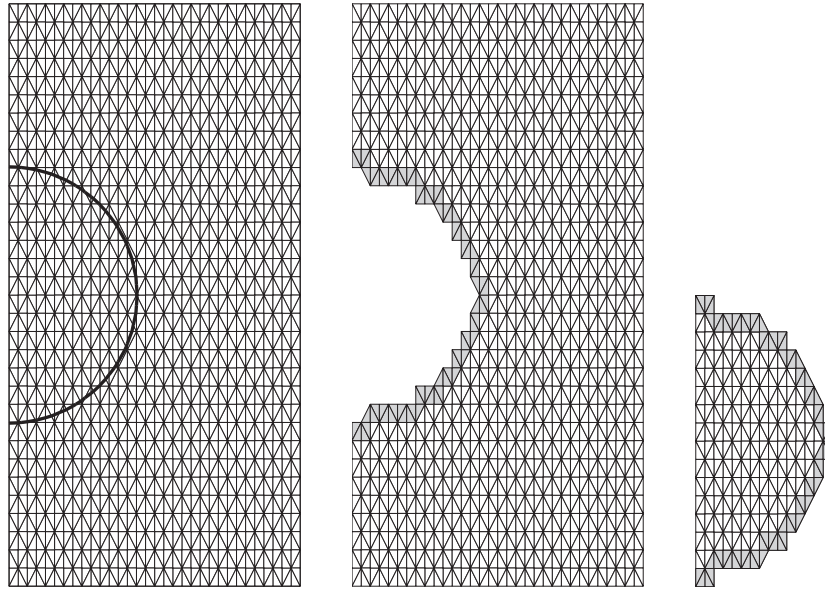


Figure 1. A mesh with the interface indicated is being divided into two new meshes. The doubled elements are shaded.

We thus wish to show that functions in V^h approximate functions $v \in H_0^1(\Omega) \cap H^2(\Omega_1 \cup \Omega_2)$ to the order h in the norm $|||\cdot|||_h$. For this purpose, we construct an interpolant of v by nodal interpolants of H^2 -extensions of v_1 and v_2 as follows. Choose extension operators $E_i : H^2(\Omega_i) \rightarrow H^2(\Omega)$ such that $(E_i w)|_{\Omega_i} = w$ and

$$\|E_i w\|_{s,\Omega} \leq C \|w\|_{s,\Omega_i} \quad \forall w \in H^s(\Omega_i), \quad s = 0, 1, 2. \quad (9)$$

Let I_h be the standard nodal interpolation operator and define

$$I_h^* v := \left(I_{h,1}^* v_1, I_{h,2}^* v_2 \right) \quad , \text{ where } I_{h,i}^* v_i := (I_h E_i v_i)|_{\Omega_i}. \quad (10)$$

We then have the following result. Let I_h^* be an interpolation operator defined as in (10). Then

$$|||v - I_h^* v|||_h \leq C h_{\max} \sum_i \|v\|_{H^2(\Omega_i)}, \quad \forall v \in H_0^1(\Omega) \cap H^2(\Omega_1 \cup \Omega_2). \quad (11)$$

In the proof of this result, we need to estimate the interpolation error at the interface. To that end, the following trace inequality is necessary: under reasonable mesh assumptions [5–7], there exists a constant C_T , depending on Γ but independent of the mesh, such that

$$\|w\|_{L_2(\Gamma_K)}^2 \leq C_T \left(h_K^{-1} \|w\|_{L_2(K)}^2 + h_K \|\nabla w\|_{L_2(K)}^2 \right), \quad \forall w \in H^1(K). \quad (12)$$

The crucial fact is that the constant in this inequality is independent of the location of the interface relative to the mesh. Optimal interpolation estimates follow, as does optimal convergence of the method irrespective of the location of the interface relative to the mesh. The key elements of the analysis are the robust coercivity of $a_h(\cdot, \cdot)$ with respect to the norm $|||\cdot|||_h$, the consistency property (8), and the approximability (11). For details, see [5].

2.2. The case of Dirichlet boundaries

We now consider the boundary value problem

$$\begin{aligned} -\nabla \cdot (\alpha \nabla u) &= f & \text{in } \Omega, \\ u &= g_D & \text{on } \Gamma_D, \\ \alpha \partial_n u &= g_N & \text{on } \Gamma_N, \end{aligned} \quad (13)$$

where $\Gamma = \Gamma_D \cup \Gamma_N$ denotes the boundary of the domain Ω , discretized on a mesh \mathcal{T}_h that contains Ω but is not fitted to the domain boundary. Denoting the mesh domain $\Omega_{\mathcal{T}}$, we consider the following finite element space:

$$W^h = \{v \in C^0(\bar{\Omega}_{\mathcal{T}}) : v|_K \in P^1(K), \forall K \in \mathcal{T}_h\}.$$

By $\mathcal{G}_h^* := \{K \in \mathcal{T}_h : K \cap \Gamma \neq \emptyset\}$, we denote the set of elements that are intersected by the interface. For an element $K \in \mathcal{G}_h^*$, let $\Gamma_K := \Gamma \cap K$ be the part of Γ in K . We also introduce the set of element faces \mathcal{F}_G associated with \mathcal{G}_h^* , defined as follows: for each face $F \in \mathcal{F}_G$, there exist two simplices K and K' such that $F = K \cap K'$ and at least one of the two is a member of \mathcal{G}_h^* . This means in particular that the boundary faces of the mesh \mathcal{T}_h are excluded from \mathcal{F}_G . On a face F such that $F = K \cap K'$, define the jump of the gradient of $v \in W^h$ by $[[\partial_n v]] = n_F \cdot \nabla v|_K - n_F \cdot \nabla v|_{K'}$, where n_F denotes the unit normal to F , the orientation is chosen arbitrarily. The problem that arises when applying Nitsche's method in this framework is that the inverse inequality corresponding to (12) cannot be robust when the right hand side must be controlled by norms over the physical domain

alone, as the cut elements can be arbitrarily small. We thus need to further stabilize the problem. The finite element discretization takes the form: find $U \in W^h$, such that

$$A_h(U, v) = L(v) \quad \forall v \in W^h, \quad (14)$$

where

$$\begin{aligned} L(v) := & (f, v)_\Omega + (g_D, \gamma_D h^{-1} v - \alpha \partial_n v)_{\Gamma_D} \\ & + (g_N, v + \gamma_N h \alpha \partial_n v)_{\Gamma_N} \end{aligned}$$

and

$$A_h(U, v) := a_h(U, v) + j(U, v)$$

with

$$\begin{aligned} a_h(U, v) := & a(U, v) - (\alpha \partial_n U, v)_{\Gamma_D} - (\alpha \partial_n v, U)_{\Gamma_D} \\ & + (\gamma_D h^{-1} U, v)_{\Gamma_D} + (\gamma_N h \alpha \partial_n U, \alpha \partial_n v)_{\Gamma_N}, \quad a(U, v) := (\alpha \nabla U, \nabla v)_\Omega \end{aligned} \quad (15)$$

and

$$j(U, v) = \sum_{F \in \mathcal{F}_G} (\gamma_1 h [\![\partial_{n_F} U]\!], [\![\partial_{n_F} v]\!])_F. \quad (16)$$

Here, γ_D, γ_N , and γ_1 are positive penalty parameters; cf. [12]. The rationale for the stabilization term (16) is that it extends the coercivity from the physical domain Ω to the mesh domain $\Omega_{\mathcal{T}}$. Details are given in Section 3.

2.3. Other interface conditions of interest

The interface conditions may vary depending on the application and may pose more discretization challenges than the aforementioned model problems. In this section, we mention a few alternatives that can still be handled in the same framework of Nitsche's method.

- Heat release at the interface leads to

$$\begin{aligned} [\![u]\!] &= 0 \quad \text{on } \Gamma, \\ [\![\mathbf{q} \cdot \mathbf{n}]\!] &= g \quad \text{on } \Gamma, \end{aligned} \quad (17)$$

where g is a heat source term. This source leads to a modified right-hand side in (8) so that

$$L(v) := \sum_i (f, v_i)_{\Omega_i} + (\kappa_2 g, v_1)_\Gamma + (\kappa_1 g, v_2)_\Gamma,$$

see [5].

- Spring-type boundary conditions common in solid mechanics can be modeled by

$$\begin{aligned} [\![u]\!] &= -k \mathbf{q} \cdot \mathbf{n} \quad \text{on } \Gamma, \\ [\![\mathbf{q} \cdot \mathbf{n}]\!] &= 0 \quad \text{on } \Gamma. \end{aligned} \quad (18)$$

Here, k denotes the compliance of distributed springs on the interface. These conditions can be modeled by Nitsche's approach as follows. Let $s_h = 1/(h/\gamma + k)$ and modify the bilinear form to

$$\begin{aligned}
a_h(U, v) := & \sum_i (\alpha_i \nabla U_i, \nabla v_i)_{\Omega_i} - (\llbracket U \rrbracket + k \langle \alpha \partial_n U \rangle, \langle \alpha \partial_n v \rangle)_{\Gamma} \\
& - (\langle \alpha \partial_n U \rangle, \llbracket v \rrbracket + k \langle \alpha \partial_n v \rangle)_{\Gamma} \\
& + (\langle \alpha \partial_n U \rangle, k \langle \alpha \partial_n v \rangle)_{\Gamma} + (s_h(\llbracket U \rrbracket + k \langle \alpha \partial_n U \rangle), \llbracket v \rrbracket + k \langle \alpha \partial_n v \rangle)_{\Gamma}.
\end{aligned}$$

The analysis of this method follows the same lines as the one for the original method; cf. [7, 23].

- Transport also on the interface can be modeled by

$$\begin{aligned}
\llbracket u \rrbracket &= 0 \quad \text{on } \Gamma, \\
\llbracket \mathbf{q} \cdot \mathbf{n} \rrbracket - \nabla_{\Gamma} \cdot (\alpha_{\Gamma} \nabla_{\Gamma} u) &= 0 \quad \text{on } \Gamma,
\end{aligned} \tag{19}$$

where ∇_{Γ} is the tangential derivative on Γ ,

$$\nabla_{\Gamma} := \nabla - \mathbf{n} \partial_n.$$

These conditions model, for example, porous media flow in a medium with a crack represented by Γ ; cf. [24]. Here, α_{Γ} represents a porosity along the crack. The bilinear form must now be modified to take the differential equation on Γ into account: formally replacing g in (17) by $\nabla_{\Gamma} \cdot (\alpha_{\Gamma} \nabla_{\Gamma} u)$, we see that a consistent bilinear form is given by

$$\begin{aligned}
a_h(U, v) := & \sum_i (\alpha_i \nabla U_i, \nabla v_i)_{\Omega_i} - (\llbracket U \rrbracket, \langle \alpha \partial_n v \rangle)_{\Gamma} \\
& - (\langle \alpha \partial_n U \rangle, \llbracket v \rrbracket)_{\Gamma} + (\gamma h^{-1} \llbracket U \rrbracket, \llbracket v \rrbracket)_{\Gamma} \\
& + (\alpha_{\Gamma} \{ \nabla_{\Gamma} U \}, \{ \nabla_{\Gamma} v \})_{\Gamma},
\end{aligned}$$

where $\{a\} := k_2 a_1 + k_1 a_2$, with k_1 and k_2 positive weights. This method requires additional stabilization in general; cf. the numerical example in Section 6.4.

- Alternative surface transport conditions are given by seeking $u : \Omega \rightarrow R$ and $u_{\Gamma} : \Gamma \rightarrow R$, where Γ denotes the boundary of Ω , such that

$$\begin{aligned}
\beta u - \beta_{\Gamma} u_{\Gamma} + \alpha \partial_n u &= 0 \quad \text{on } \Gamma, \\
\alpha \partial_n u - \nabla_{\Gamma} \cdot (\alpha_{\Gamma} \nabla_{\Gamma} u_{\Gamma}) &= g \quad \text{on } \Gamma.
\end{aligned} \tag{20}$$

Here, u_{Γ} is a concentration on the surface, which is independent of the concentration inside Ω . Applications are found, for example, in cell membrane transport; cf. [25]. Here, we no longer have a distinct side condition and can dispense with Nitsche's method. However, with cut elements, we now need a way to define the discrete approximation of u_{Γ} , which is different from the previous case, where the trace spaces on the cut elements were used to compute $\nabla_{\Gamma} U$. An obvious idea is to keep on using the trace spaces on a higher dimensional mesh as suggested by Olshanskii *et al.* [26]. Thus, we let W_{Γ}^h denote the space of continuous piecewise linear polynomials defined on \mathcal{G}_h^* and seek $U \in W^h$ and $U_{\Gamma} \in W_{\Gamma}^h$ such that

$$(\nabla U, \nabla v)_{\Omega} + ((\beta U - \beta_{\Gamma} U_{\Gamma}), v)_{\Gamma} = (f, v)_{\Omega} \quad \forall v \in W^h \tag{21}$$

and

$$(\nabla_{\Gamma} U_{\Gamma}, \nabla_{\Gamma} v_{\Gamma})_{\Gamma} - ((\beta U - \beta_{\Gamma} U_{\Gamma}), v_{\Gamma})_{\Gamma} = (g, v_{\Gamma})_{\Gamma} \quad \forall v_{\Gamma} \in W_{\Gamma}^h \tag{22}$$

or the symmetrized version

$$\begin{aligned}
& \beta(\nabla U, \nabla v)_{\Omega} + \beta_{\Gamma}(\nabla_{\Gamma} U_{\Gamma}, \nabla_{\Gamma} v_{\Gamma})_{\Gamma} \\
& + (\beta u - \beta_{\Gamma} U_{\Gamma}, \beta v - \beta_{\Gamma} v_{\Gamma})_{\Gamma} = \beta(f, v)_{\Omega} + \beta_{\Gamma}(g, v_{\Gamma})_{\Gamma}
\end{aligned} \tag{23}$$

for all $(v, v_\Gamma) \in W^h \times W_\Gamma^h$. The discretization of the equation on the interface can now be stabilized by adding

$$j_\Gamma(U_\Gamma, v_\Gamma) := \sum_{F \in \mathcal{F}_G} (\gamma_\Gamma [\partial_{n_F} U_\Gamma], [\partial_{n_F} v_\Gamma])_F$$

related to the stabilization in the Dirichlet case but with another scaling because of dimensionality. The bulk variable can be stabilized as in the Dirichlet case.

3. ENHANCING ROBUSTNESS: GHOST PENALTY

Cutting the mesh can result in boundary elements with very small intersections with the physical domain, or for PDEs on embedded surfaces, bulk elements with very small intersection with the surface domain. This may lead to a poorly conditioned system matrix or failure of stability of the discrete scheme.

Situations that are particularly sensitive are the imposition of Dirichlet boundary conditions [12, 27], or domain decomposition on unfitted meshes, where an inf-sup condition has to be satisfied, such as for incompressible elasticity [9, 28–31]. In these cases, one cannot choose the weights in (3) in a robust way. There are also situations where the weights are already prescribed by other concerns. This is the case in fluid–structure interaction [32, 33], where the elastodynamic system has no dissipation by which one can absorb the contribution of the boundary stress term and therefore only the fluid stresses are considered on the boundary. If independent adaptive mesh refinement is performed in the two subdomains of (5), this also imposes a certain choice of weights to ensure robustness, both with respect to large contrast in the physical parameter and in the mesh parameter [34].

For cases such as this, a useful trick [12, 27] is to add a penalty term in the interface zone that extends the coercivity to the whole mesh domain, that is, in the $O(h)$ zone of the mesh domain of each subdomain that does not intersect the associated physical domain. This penalty term must be carefully designed to add sufficient stability, while remaining weakly consistent for smooth solutions. To illustrate this idea, we consider the fictitious domain method for the Poisson problem (13) with $\alpha = 1$. In the next section, we demonstrate how these arguments can be extended to the coupled problem (1) and how the two formulations are related.

We observe that by taking $v = U$ in the bilinear form $a(U, v)$, we have the coercivity

$$\|\nabla U\|_{L^2(\Omega)}^2 \leq a(U, U).$$

However, to obtain coercivity of the form $a_h(U, v)$ using this stability and the boundary penalty term, the penalty parameter will depend on the cut, as by the Cauchy–Schwarz inequality

$$a_h(U, U) \geq \|\nabla U\|_{L^2(\Omega)}^2 + \left\| \gamma^{\frac{1}{2}} h^{-\frac{1}{2}} U \right\|_{L^2(\Gamma)}^2 - 2 \sum_{K \in \mathcal{G}_h^*} \|\nabla U\|_{L^2(\Gamma \cap K)} \|U\|_{L^2(\Gamma \cap K)}. \quad (24)$$

By the definition of the norm and because ∇U is piecewise constant, we have, using (12),

$$\|\nabla U\|_{L^2(\Gamma \cap K)} = \frac{m_{K_\Gamma}}{m_{K_\Omega}} \|\nabla U\|_{L^2(K_\Omega)},$$

where we recall the notation $K_\Omega = K \cap \Omega$, $K_\Gamma = K \cap \Gamma$, $m_{K_\Omega} := \text{meas}_d(K_\Omega)$ and $m_{K_\Gamma} := \text{meas}_{d-1}(K_\Gamma)$. It follows that in principle we obtain coercivity by choosing $\gamma|_K > 2h_K \left(\frac{m_{K_\Gamma}}{m_{K_\Omega}} \right)^2$, since by an arithmetic–geometric inequality, we have

$$\begin{aligned}
a_h(U, U) &\geq \|\nabla U\|_{L^2(\Omega)}^2 + \left\| \gamma^{\frac{1}{2}} h^{-\frac{1}{2}} U \right\|_{L^2(\Gamma)}^2 - \frac{1}{2} \|\nabla U\|_{L^2(\Omega)}^2 - \sum_{K \in \mathcal{G}_h^*} 2h_K \left(\frac{m_{K_\Gamma}}{m_{K_\Omega}} \right)^2 \left\| h^{-\frac{1}{2}} U \right\|_{L^2(K \cap \Gamma)}^2 \\
&\geq \frac{1}{2} \|\nabla U\|_{L^2(\Omega)}^2 + \left\| \left(\gamma - 2h_K \left(\frac{m_{K_\Gamma}}{m_{K_\Omega}} \right)^2 \right)^{\frac{1}{2}} h^{-\frac{1}{2}} U \right\|_{L^2(\Gamma)}^2.
\end{aligned}$$

Unfortunately, this makes γ strongly dependent on the cut, because for $m_{K_\Gamma} = O(h_K)$, the volume measure m_{K_Ω} can be arbitrarily small.

When the penalty parameter γ becomes strongly dependent on the mesh/boundary intersection, one may encounter problems with both conditioning and accuracy. A solution to this problem is to add the ghost penalty term of (16), denoted by $j(\cdot, \cdot)$, to the form $a_h(\cdot, \cdot)$ as in the formulation (14). The role of this term is to extend the coercivity from the physical domain Ω to the mesh domain $\Omega_{\mathcal{T}} := \Omega \cup \mathcal{G}_h^*$. Indeed, one may show the following inequality:

$$c_G \|\nabla U\|_{\Omega_{\mathcal{T}}}^2 \leq \|\nabla U\|_{\Omega}^2 + j(U, U), \quad (25)$$

where $c_G > 0$ is bounded away from zero independent of the mesh/boundary intersection for positive ghost penalty stabilization parameter γ_1 . The following weak consistency property can also be shown to hold:

$$j(I_h^* u, I_h^* u)^{\frac{1}{2}} \leq C h |u|_{H^2(\Omega)}, \quad (26)$$

where the constant C is independent of the mesh/boundary intersection. Coercivity now follows from (24) and (25) as follows:

$$\begin{aligned}
A_h(U, v) &\geq \|\nabla U\|_{L^2(\Omega)}^2 + \gamma \left\| h^{-\frac{1}{2}} U \right\|_{L^2(\Gamma)}^2 - 2C_T \|\nabla U\|_{L^2(\mathcal{G}_h^*)} \left\| h^{-\frac{1}{2}} U \right\|_{L^2(\Gamma)} + j(U, U) \\
&\geq c_G \|\nabla U\|_{\Omega_{\mathcal{T}}}^2 + \gamma \left\| h^{-\frac{1}{2}} U \right\|_{L^2(\Gamma)}^2 - 2C_T \|\nabla U\|_{L^2(\Omega_{\mathcal{T}})} \left\| h^{-\frac{1}{2}} U \right\|_{L^2(\Gamma)} \\
&\geq \frac{c_G}{2} \|\nabla U\|_{\Omega_{\mathcal{T}}}^2 + (\gamma - 2C_T^2 c_G^{-1}) \left\| h^{-\frac{1}{2}} U \right\|_{L^2(\Gamma)}^2.
\end{aligned} \quad (27)$$

Here, C_T is the constant of the trace inequality (12) and c_G is the coercivity constant of the stability estimate (25). We conclude by choosing $\gamma > 2C_T^2 c_G^{-1}$, where the lower bound is independent of the mesh/boundary intersection, but not of the penalty parameter γ_1 in $j(\cdot, \cdot)$. Error estimates now follow in a similar fashion as for the standard Nitsche's method, using (27) and (26). Indeed, provided the solution is smooth enough, there holds

$$\|U - u\|_{L^2(\Omega)} + h \|\nabla(U - u)\|_{L^2(\Omega)} \leq h^2 |u|_{H^2(\Omega)},$$

where the hidden constant is independent of the mesh/boundary intersection. One may also show that the conditioning of the system matrix is bounded independently of the mesh/boundary intersection. For further details, see [12].

3.1. Example: perfect conductor, the limit of infinite diffusion

As an example of how the improved robustness works, we will consider the problem (1), with Ω_2 completely enclosed by Ω_1 such that $\partial\Omega \subset \partial\Omega_1$ and $\Gamma := \partial\Omega_2$ (Figure 2). It is well known that in the limit $\alpha_2 \rightarrow \infty$, the solution u_2 becomes constant in Ω_2 and can therefore be exactly represented by one degree of freedom. We will give two formulations of this problem, one similar to (5), and the other using the reduced model in a framework similar to the fictitious domain approach (14) using only one degree of freedom to represent u_2 . In both formulations, we will use the ghost penalty

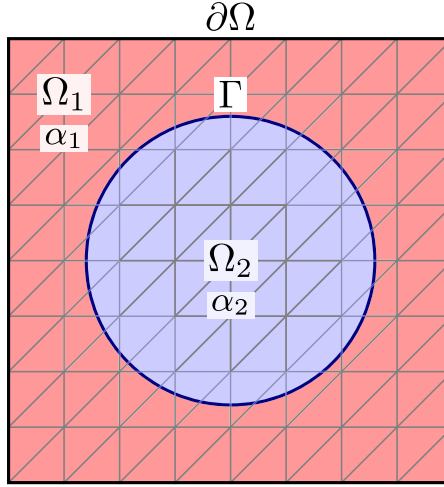


Figure 2. Illustration of the problem domain.

method so that the weights may be depending only on the diffusivities. This allows us to show that the solution of the domain decomposition approach converges to that of the fictitious domain approach in the limit as $\alpha_2 \rightarrow \infty$. We will then compare this to what is obtained if the weights are chosen as in equation (4), with the penalty defined in equation (6) or (7).

First, consider the formulation [35], and $U \in V^h$ such that

$$a_h(U, v) + \sum_{i=1}^2 \alpha_i j_i(U_i, v_i) = L(v), \quad \forall v \in V^h \quad (28)$$

with $a_h(U, v)$ and $L(v)$ as defined in (5), but for simplicity with $f_2 \equiv 0$ and using the weights

$$\kappa_1 := \frac{\alpha_2}{\alpha_1 + \alpha_2}, \quad \kappa_2 := \frac{\alpha_1}{\alpha_1 + \alpha_2} \quad (29)$$

and the penalty parameter

$$\gamma := 4 \left(\frac{\alpha_1 \alpha_2}{\alpha_1 + \alpha_2} \right) C_T^2 c_G^{-1}. \quad (30)$$

This choice of weights was first considered in the context of discontinuous Galerkin methods in [36] and then for Nitsche's method in [37]. The ghost penalty terms are designed similarly to the formulation (14). However, here, $j_i(U_i, v_i)$ is acting in the boundary zone of the mesh domain $\Omega_{\mathcal{T}}^i := \Omega_i \cup \mathcal{G}_h^*$. It is straightforward to prove coercivity of this formulation using the arguments of the previous section, indeed,

$$\begin{aligned} a_h(U, U) + \sum_{i=1}^2 \alpha_i j_i(U_i, U_i) &\geq c_G \sum_{i=1}^2 \alpha_i \|\nabla U_i\|_{\Omega_{\mathcal{T}}^i}^2 + \gamma \|h^{-\frac{1}{2}} \llbracket U \rrbracket\|_{L^2(\Gamma)}^2 - 2 \sum_{i=1}^2 (\kappa_i \alpha_i \partial_n U_i, \llbracket U \rrbracket)_{\Gamma} \\ &\geq \frac{c_G}{2} \sum_{i=1}^2 \alpha_i \|\nabla U_i\|_{\Omega_{\mathcal{T}}^i}^2 + \left(\gamma - 2 \left(\frac{\alpha_1 \alpha_2}{\alpha_1 + \alpha_2} \right) C_T^2 c_G^{-1} \right) \|h^{-\frac{1}{2}} \llbracket U \rrbracket\|_{L^2(\Gamma)}^2 \\ &= \frac{1}{2} \left(c_G \sum_{i=1}^2 \alpha_i \|\nabla U_i\|_{\Omega_{\mathcal{T}}^i}^2 + \gamma \|h^{-\frac{1}{2}} \llbracket U \rrbracket\|_{L^2(\Gamma)}^2 \right). \end{aligned} \quad (31)$$

Here, we used the inequality

$$\kappa_i \alpha_i^{\frac{1}{2}} \leq \left(\frac{\alpha_1 \alpha_2}{\alpha_1 + \alpha_2} \right)^{\frac{1}{2}}, \quad i = 1, 2$$

and the stability (25) in both subdomains.

If we formally let $\alpha_2 \rightarrow \infty$ and replace U_2 and v_2 by constant functions, we find the following formulation: find $\tilde{U}_1 \times \tilde{U}_2 \in V_1^h \times \mathbb{R}$ such that for all $v_1 \times v_2 \in V_1^h \times \mathbb{R}$, there holds

$$\begin{aligned} A_h(\tilde{U}, v) &:= (\alpha_1 \nabla \tilde{U}_1, \nabla v_1) - ([\tilde{U}], \alpha_1 \partial_n v_1)_\Gamma - (\alpha_1 \partial_n \tilde{U}_1, [v])_\Gamma \\ &+ (\gamma h^{-1} [\tilde{U}], [v])_\Gamma + \alpha_1 j_1(\tilde{U}_1, v_1) = L(v), \end{aligned} \quad (32)$$

where $\gamma = 4\alpha_1 C_T^2 c_G^{-1}$. It follows by inspection that this formulation is equivalent to (14), with the only difference that in this case the Dirichlet value set on Γ is an unknown constant value. The coercivity and hence the discrete wellposedness can be shown here exactly using the arguments of (27) together with a triangle inequality and a trace inequality to get control of \tilde{U}_2 . First note that by a triangular inequality, a trace inequality on the whole domain Ω_1 , and a Poincaré inequality, we have

$$\|\tilde{U}_2\|_{L^2(\Gamma)} \leq \|[\tilde{U}]\|_{L^2(\Gamma)} + \|\tilde{U}_1\|_{L^2(\Gamma)} \leq \|[\tilde{U}]\|_{L^2(\Gamma)} + C \|\nabla \tilde{U}_1\|_{L^2(\Omega_1)}.$$

Therefore, by the same arguments as above, there exists a constant $c > 0$ such that

$$c\alpha_1 \left(\|\tilde{U}_2\|_{L^2(\Gamma)}^2 + \|h^{-\frac{1}{2}} [\tilde{U}]\|_{L^2(\Gamma)}^2 + \|\nabla \tilde{U}_1\|_{\Omega_1^1}^2 \right) \leq A_h(\tilde{U}, \tilde{U}) = L(\tilde{U}_1) \leq \|f_1\|_{L^2(\Omega_1)} \|\nabla \tilde{U}_1\|_{\Omega_1^1}. \quad (33)$$

Consequently, both (28) and (32) are coercive, and with some additional effort, one may prove that the approximations indeed have optimal convergence in the H^1 and L^2 -norm provided the exact solution is sufficiently smooth when restricted to each subdomain.

Here, we will instead consider the robustness of the formulation (28). A natural question to ask, as (32) was obtained by taking the formal limit $\alpha_2 \rightarrow \infty$ in (28), is if the solutions of the two formulations also coincide in the limit. We will show now that this is the case. Let $U = (U_1, U_2)$ denote the solution of (28) and $\tilde{U} = (\tilde{U}_1, \tilde{U}_2)$ denote the solution of (32). If we set $\mathcal{E} = \tilde{U} - U$ and note that $\mathcal{E} \in V^h$, we may use the coercivity proven in (31):

$$\frac{1}{2} \left(c_G \sum_{i=1}^2 \alpha_i \|\nabla \mathcal{E}_i\|_{\Omega_i^1}^2 + \gamma \|h^{-\frac{1}{2}} [\mathcal{E}]\|_{L^2(\Gamma)}^2 \right) \leq a_h(\mathcal{E}, \mathcal{E}) + \sum_{i=1}^2 \alpha_i j_i(\mathcal{E}_i, \mathcal{E}_i).$$

Using the formulation (28) and once again that $\mathcal{E} \in V^h$, we see that

$$a_h(\mathcal{E}, \mathcal{E}) + \sum_{i=1}^2 j_i(\mathcal{E}_i, \mathcal{E}_i) = a_h(\tilde{U}, \mathcal{E}) + \alpha_1 j_1(\tilde{U}_1, \mathcal{E}_1) - L(\mathcal{E}).$$

Now, we apply the formulation (32) on the form $A_h(\tilde{U}, \mathcal{E}) = L(\mathcal{E})$ to write

$$\begin{aligned} a_h(\tilde{U}, \mathcal{E}) + \alpha_1 j_1(\tilde{U}_1, \mathcal{E}_1) - A_h(\tilde{U}, \mathcal{E}) &= -([\tilde{U}], \langle \alpha \partial_n \mathcal{E} \rangle - \alpha_1 \partial_n \mathcal{E})_\Gamma \\ &\quad - \left(\left(\left(\frac{\alpha_1 \alpha_2}{\alpha_1 + \alpha_2} \right) - \alpha_1 \right) \partial_n \tilde{U}_1, [\mathcal{E}] \right)_\Gamma \\ &= \left([\tilde{U}], \frac{\alpha_1^2}{\alpha_1 + \alpha_2} \partial_n \mathcal{E}_1 - \kappa_2 \alpha_2 \partial_n \mathcal{E}_2 \right)_\Gamma \\ &\quad + \left(\left(\frac{\alpha_1^2}{\alpha_1 + \alpha_2} \right) \partial_n \tilde{U}_1, [\mathcal{E}] \right)_\Gamma. \end{aligned} \quad (34)$$

By applying Cauchy–Schwarz inequalities and trace inequalities and using the definition of γ in the right hand side, we may then obtain

$$\begin{aligned} & \frac{1}{2} \left(c_G \sum_{i=1}^2 \alpha_i \|\nabla \mathcal{E}_i\|_{\Omega_{\mathcal{T}}^i}^2 + \|\gamma^{\frac{1}{2}} h^{-\frac{1}{2}} \llbracket \mathcal{E} \rrbracket\|_{L^2(\Gamma)}^2 \right) \\ & \leq C \left(\frac{\alpha_1}{\alpha_2} \right)^{\frac{1}{2}} \left(\|\gamma^{\frac{1}{2}} h^{-\frac{1}{2}} \llbracket \tilde{U} \rrbracket\|_{\Gamma}^2 + \|\alpha_1^{\frac{1}{2}} \nabla \tilde{U}_1\|_{\Omega_{\mathcal{T}}^1}^2 \right)^{\frac{1}{2}} \left(\sum_{i=1}^2 \alpha_i \|\nabla \mathcal{E}_i\|_{\Omega_{\mathcal{T}}^i}^2 + \|\gamma^{\frac{1}{2}} h^{-\frac{1}{2}} \llbracket \mathcal{E} \rrbracket\|_{L^2(\Gamma)}^2 \right)^{\frac{1}{2}}. \end{aligned}$$

Using (33) and $c\alpha_1 \leq \gamma \leq C\alpha_1$ when $\alpha_1 \leq \alpha_2$, it follows that

$$\left(\sum_{i=1}^2 \alpha_i \|\nabla \mathcal{E}_i\|_{\Omega_{\mathcal{T}}^i}^2 + \alpha_1^{\frac{1}{2}} \|\gamma^{\frac{1}{2}} h^{-\frac{1}{2}} \llbracket \mathcal{E} \rrbracket\|_{L^2(\Gamma)}^2 \right)^{\frac{1}{2}} \leq C \alpha_2^{-\frac{1}{2}} \|f_1\|_{L^2(\Omega_1)},$$

and we conclude that $\mathcal{E} \rightarrow 0$ as $\alpha_2 \rightarrow \infty$ meaning that the asymptotic limit of the solution of (28) coincides with that of (32).

Another important observation is that for the discretization using ghost penalty and weights depending only on the diffusion, preconditioning the system matrix using diagonal scaling with α_1, α_2 leads to a system whose condition number is independent of both the mesh/boundary intersection and the contrast in the diffusion (for details, see [35]).

Note that the use of the weights (4) and the penalty parameters (6) or (7) do not allow a similar robust limit formulation. This can be seen in the following way. Consider the penalty parameter γ for the choice (6) proposed in [21]. Then the limit satisfies

$$\lim_{\alpha_2 \rightarrow \infty} \gamma|_K = \alpha_1 h_K \frac{m_{K_\Gamma}}{m_{K_{\Omega_1}}},$$

which is robust in the diffusion parameter but degenerates into the choice of weights for the fictitious domain method that depends on the element cuts. Hence, for the weights (6), preconditioning using diagonal scaling will make the system robust for large contrast, but unfortunate cuts will still result in a poorly conditioned system matrix. This disadvantage motivated the introduction of the ghost penalty operator in the previous section.

The choice (7) on the other hand results in the limit behavior,

$$\lim_{\alpha_2 \rightarrow \infty} \gamma|_K = \infty.$$

This implies that $\llbracket U \rrbracket = 0$ across the boundary, which may not always be compatible with optimal accuracy. Observe also that the penalty parameter is present in the system matrices corresponding to the bulk discretization in both subdomains. As γ becomes big, with increasing α_2 , it will destroy the conditioning of the matrix corresponding to subdomain Ω_1 .

3.2. A numerical illustration

Robustness issues may also appear in the limit of vanishing diffusion in a subdomain. We consider a configuration similar to that of Figure 2, with the square domain $\Omega := [-1, 1]^2$. Let Ω_2 be a disc with radius 0.75 centered in the origin and $\Omega_1 := \Omega \setminus \Omega_2$. In this configuration, we solve (1), with $u|_{\partial\Omega} = 0$, $\alpha_1 = 1$, $f = 1$, $c|_{\Omega_1} = 0$, $c|_{\Omega_2} = 1$ and with $\alpha_2 \in \{10^{-4}, 10^{-6}, 10^{-9}\}$. First, we apply the formulation (5) with weights given by (3). The elevations of the solutions are presented in Figure 3. We then solve the same problem using the method (28) with (29)–(30) and give the corresponding elevations in Figure 4. Observe the relatively strong, but local, spurious overshoots that are present close to the layer in Figure 3. The combination of the ghost penalty and

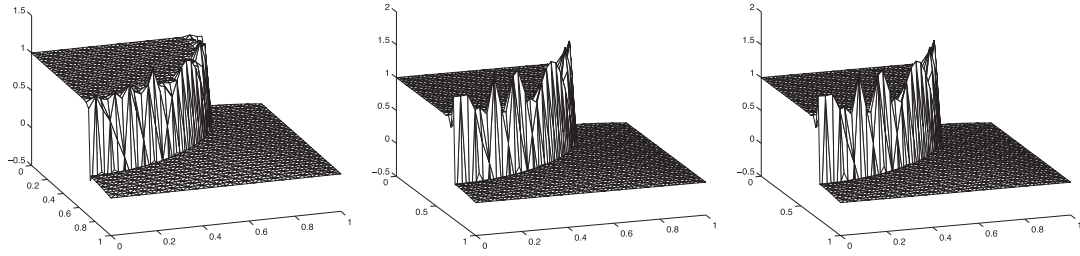


Figure 3. Computation using the weights (3). From left to right, $\alpha_2 = 10^{-4}$, 10^{-6} , and 10^{-9} .

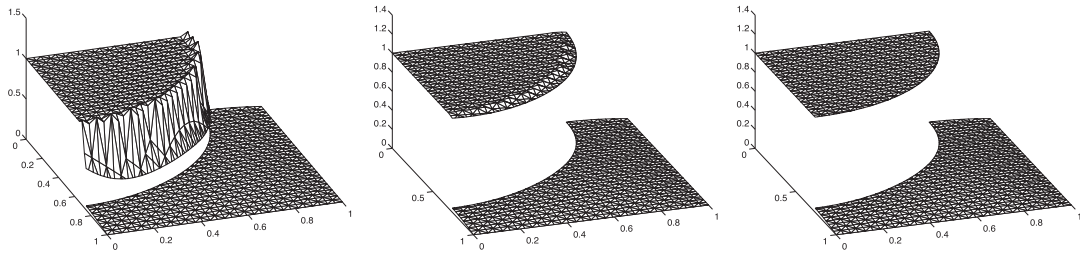


Figure 4. Computation using ghost penalty and the weights (29)–(30). From left to right, $\alpha_2 = 10^{-4}$, 10^{-6} , 10^{-9} .

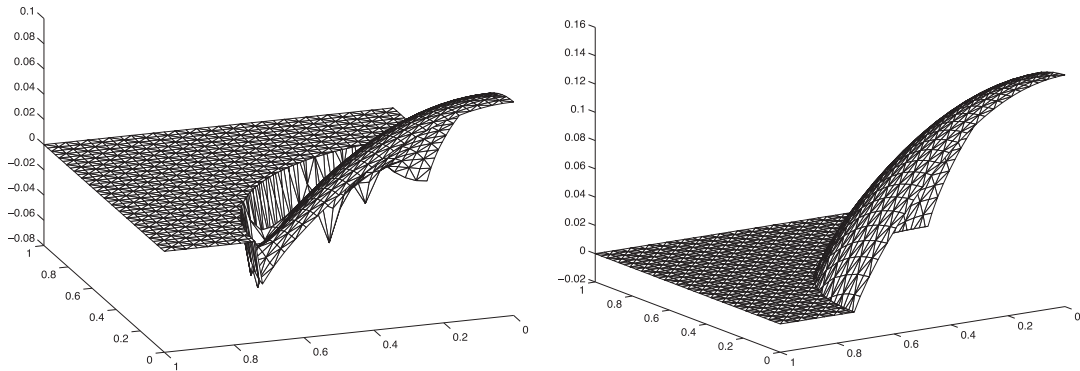


Figure 5. Limit of infinite diffusion. Left: method (5) with weights (3). Right: method (28) with (29)–(30).

the parameters (29)–(30) eliminates the oscillations by relaxing the continuity constraint in the limit. Indeed, the sharp layer is represented as a discontinuity when it is under resolved.

Now, we consider the case where $c|_{\Omega_1} = c|_{\Omega_2} = 0$, $\alpha_2 = 1$ and choose $\alpha_1 = 10^{20}$, to illustrate the effect in the limit $\alpha_1 \rightarrow \infty$. This corresponds to a situation similar to that explored in Example 3.1, but with the diffusivity going to infinity in the outer domain. In Figure 5, we compare the results of the method (5) with the weights (3) and of (28) with (29)–(30). We see that for this large contrast, the method using the weights (3) exhibits some spurious oscillations close to the boundary, probably owing to an incompatibility between the constraint $\llbracket U \rrbracket = 0$ and the weakly imposed condition on the gradient. In other words, the finite element space defined on the cut mesh does not allow for a H^1 -conforming interpolant that also can represent the jump in the gradient. This effect is reminiscent of locking but only present in the vicinity of the interface. The method (28) on the other hand converges to the fictitious domain solution of (14) in the limit for which optimal error estimates exist.

3.3. Ghost penalty for surface partial differential equations

Let us consider the Laplace–Beltrami problem: find $u_\Gamma : \Gamma \rightarrow \mathbf{R}$ such that

$$-\Delta_\Gamma u_\Gamma + u_\Gamma = f \quad \text{on } \Gamma, \quad (35)$$

where $\Delta_\Gamma = \nabla_\Gamma \cdot \nabla_\Gamma$ is the Laplace–Beltrami operator. The finite element method takes the form: find $U_\Gamma \in W_\Gamma^h$ such that

$$(\nabla_\Gamma U_\Gamma, \nabla_\Gamma v)_\Gamma + (U_\Gamma, v)_\Gamma + j_\Gamma(U_\Gamma, v) = (f, v)_\Gamma \quad \forall v \in W_\Gamma^h, \quad (36)$$

and we recall that W_Γ^h is the space of continuous piecewise linears on \mathcal{G}_h^* , the union of elements intersecting Γ .

In this case, the ghost penalty term provides additional stability to the discrete problem, which may be arbitrary ill conditioned. The proof of the optimal scaling estimate of the condition number, that is,

$$\text{cond}(A) \leq Ch^{-2}, \quad (37)$$

basically relies on the Poincaré estimate

$$\|v\|_{\mathcal{G}_h^*}^2 \leq Ch (\|\nabla_\Gamma v\|_\Gamma^2 + \|v\|_\Gamma^2 + j_\Gamma(v, v)) \quad \forall v \in W_h \quad (38)$$

and the inverse estimate

$$\|\nabla_\Gamma v\|_\Gamma^2 + \|v\|_\Gamma^2 + j_\Gamma(v, v) \leq Ch^{-3} \|v\|_{\mathcal{G}_h^*}^2 \quad \forall v \in W_h. \quad (39)$$

Note that, in these inequalities, the ghost penalty term plays a crucial role. Using the Poincaré and inverse estimate together with the coercivity of the bilinear form and the standard scaled equivalence,

$$C_1 h^d |\hat{v}|_{\mathbf{R}^N}^2 \leq \|v\|_{\mathcal{G}_h^*}^2 \leq C_2 h^d |\hat{v}|_{\mathbf{R}^N}^2 \quad \forall v \in W_h \quad (40)$$

between the Euclidian norm on the nodal values $\hat{v} \in \mathbf{R}^N$ of $v \in W_h$ and the L^2 -norm on \mathcal{G}_h^* , we may prove (37); see [38] for details.

Furthermore, we note that the ghost penalty term scales in the same way as the bilinear form $(\nabla_\Gamma v, \nabla_\Gamma w)_\Gamma$ and that we have the interpolation error estimate

$$\|\nabla_\Gamma (u_\Gamma - I_h^* u_\Gamma)\|_\Gamma^2 + \|u_\Gamma - I_h^* u_\Gamma\|_\Gamma^2 + j_\Gamma(u_\Gamma - I_h^* u_\Gamma, u_\Gamma - I_h^* u_\Gamma) \leq Ch^2 \|u_\Gamma\|_{H^2(\Gamma)}^2. \quad (41)$$

Using the Galerkin orthogonality, we obtain an optimal error estimate in the energy norm and by duality an L^2 -error estimate,

$$h \|\nabla_\Gamma (u_\Gamma - U_\Gamma)\|_\Gamma + \|u_\Gamma - U_\Gamma\|_\Gamma \leq Ch^2 \|u_\Gamma\|_{H^2(\Gamma)}. \quad (42)$$

In order to deal with the effect of approximating the boundary Γ , a more elaborate analysis is needed, and we refer to [38] for further details. For a numerical example of the solution of the Laplace–Beltrami equation on a non-trivial surface, see Section 6.3.

4. DISCRETIZING GEOMETRY IN CutFEM

The starting point for the discretization of the geometry in CutFEM is to immerse an arbitrary geometric description in a background mesh. This mesh is typically chosen structured, to facilitate the handling of data structures, communication, and hierarchic mesh adaption. The discretization space and the variational formulation are then adapted to the geometry so that suitable boundary

and interface conditions are imposed weakly as described in Section 2. This approach leads to some challenging implementation issues that we will describe later. It is advantageous to choose one particular geometry description that is versatile and simple for the construction of the discretization. Other geometrical descriptions can then be either included using modules that translate different formats to the one chosen to interface with the code or provided with their own CutFEM modules.

In our work, later, we focus on the level-set method [39] for its use both as a description of stationary boundaries and of interfaces evolved by computation. Here, the location of the boundary is given by the zero level set of a function $\phi : \mathbb{R}^d \rightarrow \mathbb{R}$. More precisely, a given domain $\Omega \subset \mathbb{R}^d$ can be decomposed into an inner part Ω_1 , an outer part Ω_1^c , and their common interface Γ by requiring that $\forall \mathbf{x} \in \Omega$

$$\begin{cases} \phi(\mathbf{x}) < 0 \Leftrightarrow \mathbf{x} \in \Omega_1, \\ \phi(\mathbf{x}) = 0 \Leftrightarrow \mathbf{x} \in \Gamma, \\ \phi(\mathbf{x}) > 0 \Leftrightarrow \mathbf{x} \in \Omega_1^c, \end{cases} \quad (43)$$

where we define Ω_1^c as the (open) complement of Ω_1 in Ω .

To give a few non-trivial examples of analytically given level-set based surface descriptions, we introduce

Doughnut

$$\phi(x, y, z) = \left(R - \sqrt{x^2 + y^2}\right)^2 + z^2 - r^2. \quad (44)$$

In the following, we choose $R = 1.2$, $r = 0.3$.

Popcorn ([21, 40])

$$\phi(x, y, z) = \sqrt{x^2 + y^2 + z^2} - r_0 - \sum_{k=0}^{11} A \exp^{-((x-x_k)^2 + (y-y_k)^2 + (z-z_k)^2)/\sigma^2}, \quad (45)$$

where

$$\begin{aligned} (x_k, y_k, z_k) &= \frac{r_0}{\sqrt{5}} \left(2 \cos\left(\frac{2k\pi}{5}\right), 2 \sin\left(\frac{2k\pi}{5}\right), 1 \right), \quad 0 \leq k \leq 4, \\ (x_k, y_k, z_k) &= \frac{r_0}{\sqrt{5}} \left(2 \cos\left(\frac{(2(k-5)-1)\pi}{5}\right), 2 \sin\left(\frac{(2(k-5)-1)\pi}{5}\right), -1 \right), \quad 5 \leq k \leq 9, \\ (x_k, y_k, z_k) &= (0, 0, r_0), \quad k = 10, \\ (x_k, y_k, z_k) &= (0, 0, -r_0), \quad k = 11. \end{aligned}$$

In the following, we choose $r_0 = 0.6$, $\sigma = 0.2$, $A = 2$.

Swiss Cheese Block

$$\begin{aligned} \phi(x, y, z) &= (x^2 + y^2 - 4)^2 + (z^2 - 1)^2 + (y^2 + z^2 - 4)^2 + (x^2 - 1)^2 \\ &\quad + (z^2 + x^2 - 4)^2 + (y^2 - 1)^2 - 15. \end{aligned} \quad (46)$$

The corresponding surfaces are depicted in Figure 6.

Using a level-set description, complex domains can easily be constructed by translating Boolean set operations and geometric transformations into simple manipulations of level-set representations. For instance, given two level set functions ϕ_1 and ϕ_2 representing the domains Ω_1 and Ω_2 , respectively, the level-set function representing the result of a standard Boolean set operation can be constructed according to the following table:

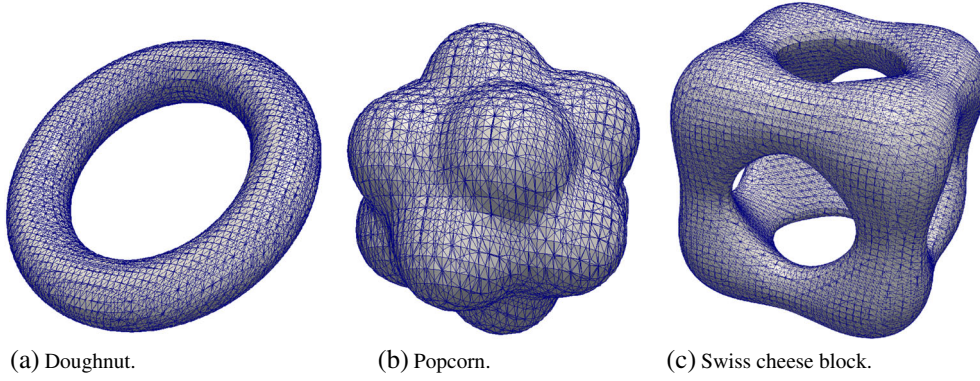


Figure 6. Examples of level-set based surface descriptions. (a) Doughnut. (b) Popcorn. (c) Swiss cheese block.

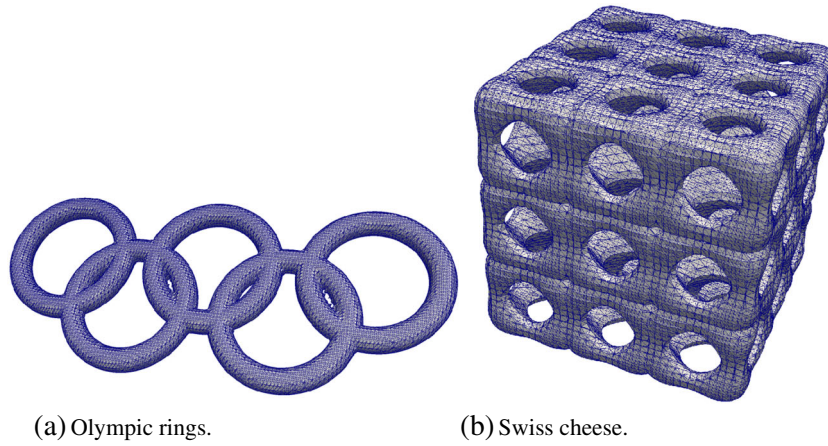


Figure 7. Examples of complex geometries generated by taking the union of properly translated domains. (a) Olympic rings. (b) Swiss cheese.

Ω_1^c	$\phi = -\phi_1$
$\Omega_1 \cup \Omega_2$	$\phi = \min(\phi_1, \phi_2)$
$\Omega_1 \cap \Omega_2$	$\phi = \max(\phi_1, \phi_2)$
$\Omega_1 \setminus \Omega_2$	$\phi = \max(\phi_1, -\phi_2)$
$\Omega_1 \triangle \Omega_2$	$\phi = \min(\max(\phi_1, -\phi_2), \max(\phi_2, -\phi_1))$

Moreover, for any diffeomorphic mapping $T : \mathbb{R}^d \rightarrow \mathbb{R}^d$, the mapped domain $\tilde{\Omega}_1 = T(\Omega_1)$ is represented by $\tilde{\phi}_1 = \phi_1 \circ T^{-1}$. Figure 7 illustrates how complex surfaces can be generated by combining these operations.

5. IMPLEMENTATIONAL ASPECTS OF A LEVEL-SET BASED CutFEM METHOD

We now briefly review some of the data structures and algorithms required to efficiently compute a discrete representation of surfaces implicitly given by a level-set and how to evaluate the variational formulation on this discrete geometry. An important step here is the introduction of a subtriangulation of cut elements to facilitate integration, which we will describe in more detail below. This subtriangulation is used only for integration so that the resulting subtriangles are not constrained by the conformity requirements of the finite element space. In addition, their aspect ratio does not impact the approximation properties of the discretization, because they are never used in the analysis.

In general, the mesh subdivision can be characterized as follows. Referring to the notation from Section 2, we recall that for a tessellation \mathcal{T}_h of Ω , the (unfitted) surface Γ leads to natural subdivision $\mathcal{T}_h = \mathcal{T}_h^1 \cup \mathcal{T}_h^2 \cup \mathcal{G}_h^*$, where $\mathcal{T}_h^i = \{K \in \mathcal{T}_h : K \subset \Omega_i\}$. Note that, to apply the finite element method to a pure surface problem, we only need to compute a tessellation of Γ with respect to \mathcal{G}_h^* . For a fictitious domain problem, we require a subtessellation for the inner part of the cut elements $\mathcal{G}_{h,1}^* = \{K \cap \Omega_1 : K \in \mathcal{G}_h^*\}$ and for an interface problem, we need to subtriangulate both the inner and outer parts, that is, also $\mathcal{G}_{h,2}^* = \{K \cap \Omega_2 : K \in \mathcal{G}_h^*\}$. However, as the following section will explain, all these sub-triangulations can be provided simultaneously by using the well-known marching tetrahedra algorithm [39, 41].

In addition to this subtessellation algorithm, routines for the computation of integrals over cut cell parts have to be provided. Indeed, for each intersected element $K \in \mathcal{G}_h^*$, volume integrals have to be evaluated over the parts of K that are covered by the two physical subdomains Ω_1 , $K_1 = K \cap \Omega_1$ and Ω_2 , $K_2 = K \cap \Omega_2$. To construct the matrix contributions that impose interface and boundary conditions, surface integrals over interface segments, $\Gamma_K = \Gamma \cap K$, that lie within the element $K \in \mathcal{G}_h^*$, also have to be computed.

In the following, we will first detail our implementation of the classical tetrahedra algorithm [39, 41] and then give some further details on how to evaluate integrals over cell parts.

5.1. Interface approximation and sub-triangulation of cut elements

In the first step of the subtessellation algorithm, the values of the level-set function in the element nodes are used to distinguish between elements that are fully contained in Ω_1 ($\phi < 0$ for all nodes) or Ω_2 ($\phi > 0$ for all nodes) and the elements that are intersected by the interface Γ ($\phi > 0$ and $\phi < 0$ in some nodes) (Figures 8 and 9).

For elements that are intersected by the interface, we perform a subtriangulation of the element allowing us to apply standard quadrature rules to the subtriangles for the integration over the parts K_1 , K_2 , and Γ_K . Here, we only consider linear intersections of the zero level-set with the elements

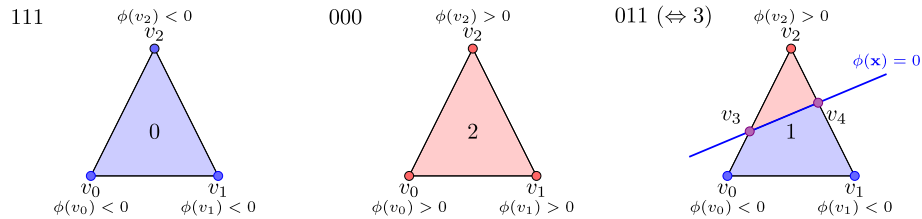


Figure 8. In a first step, a loop over all values of the level-set function in element nodes is performed, and all cells that are fully contained in Ω_1 are marked with 0, all cells that are fully contained in Ω_2 are marked with 2, and all cells that are intersected are marked with 1.

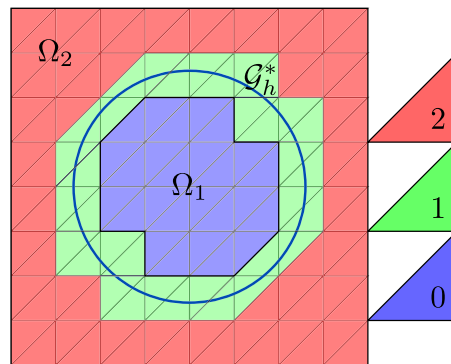


Figure 9. Example for resulting cell marker for level-set function $\phi(x) = x^2 + y^2 - 1$ of circular domain Ω_1 .

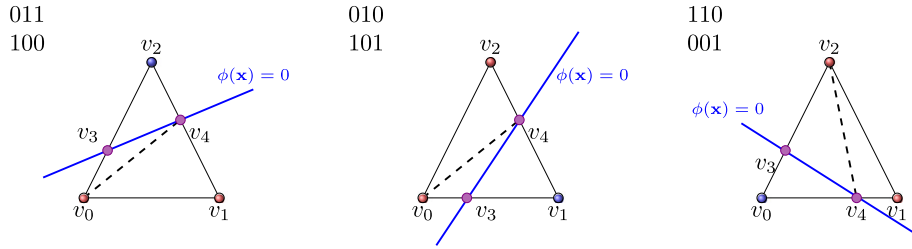


Figure 10. Straight intersection cases in 2d, their corresponding cell flags, and sub-triangulation.

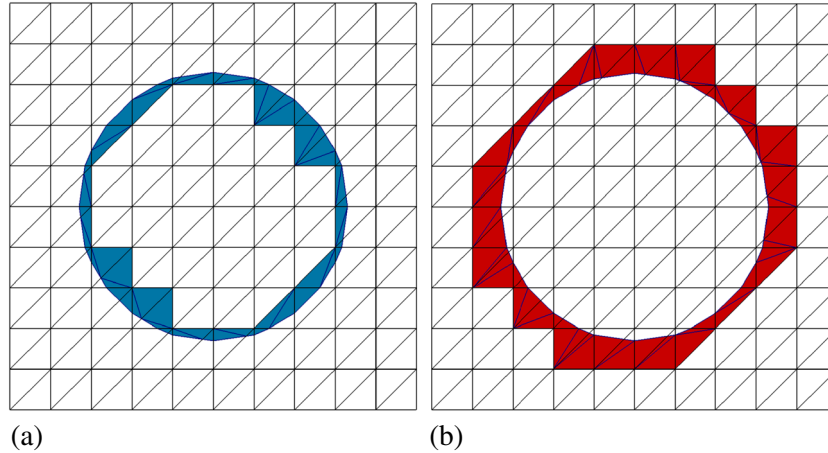


Figure 11. Cell subtriangulation for (a) Ω_1 and (b) Ω_2 for circular domain Ω_1 .

that are represented by one straight line segment per element in 2d or one plane in 3d. In more detail, the subtriangulation algorithm consists of the following steps.

For all elements that are intersected by the interface,

- (1) Flag cells according to which nodes of the element have a negative level-set value (flag with '1') and which nodes have a positive level-set value (flag with '0').
- (2) Use this flag in order to compute the intersection points of the zero level-set with element facets via linear interpolation between the level set values in the nodes connected to the facet. Note that an intersected facet is characterized by the fact that the values of the level-set function in the nodes connected to that facet have positive and negative values of the level-set function.
- (3) Build a subtriangulation of each cut cell part, that is, K_1 , K_2 and Γ_K . The subtriangulation of K_i , $i = 1, 2$, and the sub-triangulation for Γ_K are then added to the subtessellation for the inner part of the cut elements $\mathcal{G}_{h,1}^*$, the outer part of the cut elements $\mathcal{G}_{h,2}^*$, or the tessellation of Γ , respectively (Figure 11), together with a map between the cell parts and the corresponding parent cell.

In two space dimensions, the cut cell parts, K_1 and K_2 , are either triangular or quadrilateral (Figure 10). The triangular part can be added directly to the subtriangulation of the corresponding domain. The quadrilateral part is subdivided into two triangles first and then added to the corresponding subtriangulation. The interface segment Γ_K is represented by straight line segment connecting both intersection points. These interface line segments are stored in a separate mesh object with topological dimension 1. The resulting sub-triangulations for Ω_1 and Ω_2 for a circular domain Ω_1 are shown in Figure 11.

In three space dimensions, there is a much wider range of cut cases to consider. We can distinguish 14 cases, among which eight have a triangular interface plane cut of the zero level set with the tetrahedron and six with a quadrilateral interface plane cut (Figure 12). The triangular plane cut can be added directly to the subtriangulation of the two-dimensional

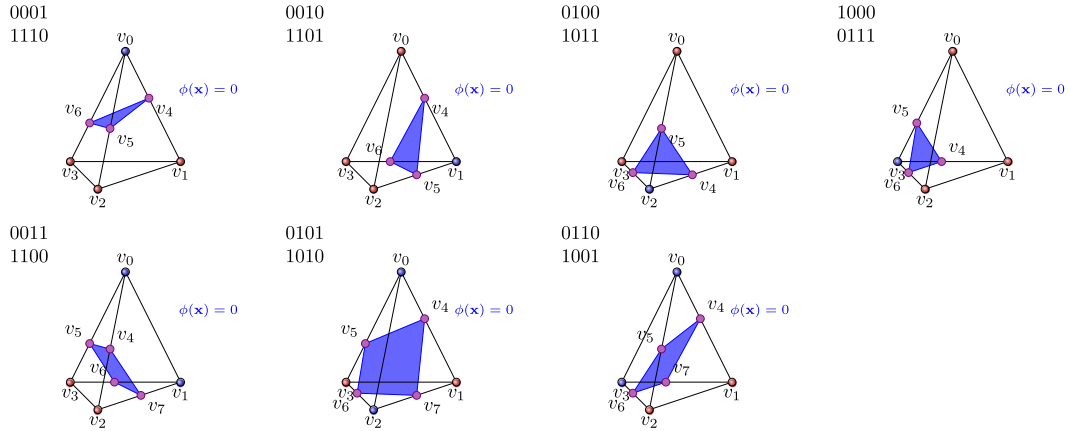


Figure 12. Intersection cases in 3d and their corresponding flag.

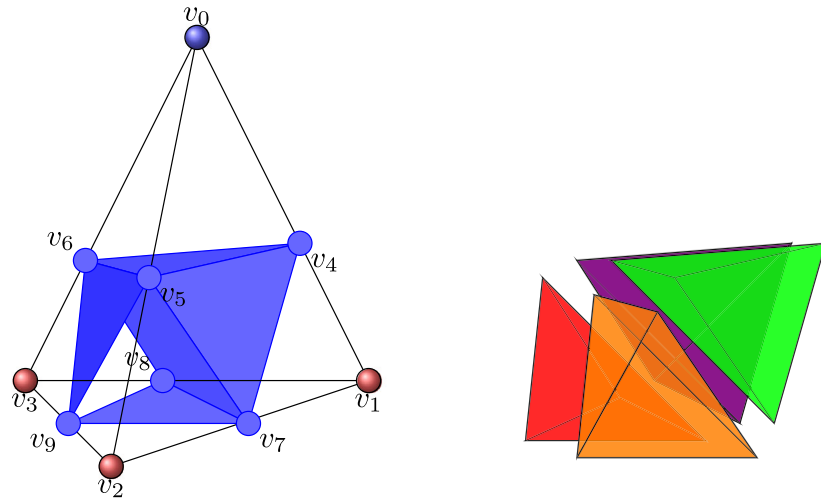


Figure 13. Subtriangulation of a tetrahedron into eight subtetrahedra.

zero level set interface representation, and the quadrilateral planes are subdivided into two triangular parts. For the volume subtriangulation, we decompose the tetrahedron into eight subtetrahedra (Figure 13) and add the subtetrahedra to the corresponding subtessellations of Ω_1 and Ω_2 depending on the cell flag, that is, depending on whether they lie in Ω_1 or Ω_2 .

5.2. Integration over subtriangulation

For the evaluation of integrals over the subtriangulation, we require two mappings $\chi_w^{-1} \circ \chi_p$. Here, the linear affine mapping, χ_p , between the reference element and the cell part, transforms a quadrature rule defined on the reference element in terms of quadrature points ξ_i and weights w_i into a quadrature rule on the subtriangle cell part (Figure 14). The mapping χ_w between the reference element and the whole ‘parent’ cell is used to map the quadrature points defined on the physical subtriangle to their location in the reference element of the whole parent cell.

More precisely, the quadrature rule over the cell part is given in terms of quadrature points in physical space $\mathbf{x}_i^p = \chi_p(\xi_i)$ and quadrature weights $w_i^p = w_i |\det(J_p)|$. Here, J_p is the Jacobian of the mapping χ_p . These points and weights can then be used to perform the numerical quadrature over the given cell part, for example, for a mass matrix, by

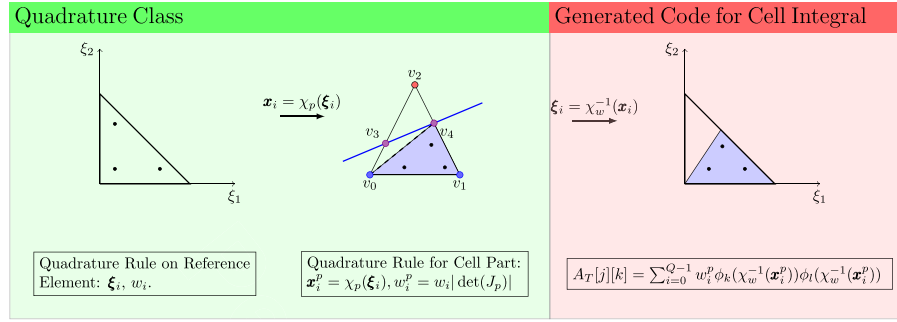


Figure 14. Schematics of integration over the subtriangulation involving two coordinate transformations χ_p between the reference element and the sub-triangle cell part and χ_w between the reference element and the whole cell.

$$A_T[j][k] = \sum_{i=0}^{Q-1} w_i^p \phi_k(\chi_w^{-1}(\mathbf{x}_i^p)) \phi_l(\chi_w^{-1}(\mathbf{x}_i^p)). \quad (47)$$

Here, $A_T[j][k]$ denotes the matrix entry for the cell integral over the degrees of freedom j and k . Note here that we have to evaluate basis functions and/or their derivatives at arbitrary points on the reference element $\xi_i = \chi_w^{-1}(\mathbf{x}_i)$ that result from the backward transformation of the quadrature rule over the cell part onto the reference element of the whole ‘parent’ element.

5.3. Software for CutFEM-type method

The technology required for the automated assembly of general cut finite element based variational forms over fictitious domains and embedded surfaces has been implemented as part of the software library libCutFEM. libCutFEM is open source and freely available from <http://www.cutfem.org>. This software library builds on the finite element library DOLFIN, which is part of the FEniCS project [42] for automated scientific computing. The main feature of FEniCS is the automated treatment of finite element variational problems, based on automated generation of highly efficient C++ code from abstract high-level descriptions of finite element variational problems expressed in near-mathematical notation [43].

libCutFEM makes use of this automatization technology in the following way. We specify the variational problem in near-mathematical notation in the domain specific Uniform Form Language [43] (for an example, see Figure 17). Then the code generation components of FEniCS (UFL+FFC+UFC [42]) generate code containing information about the cell integrals of the given forms and information about the elements such as the degrees of freedom maps. In particular, libCutFEM uses an extension developed in earlier works of the components FFC [44, 45] and UFC [46]. In these works, FFC and UFC were extended [33, 47, 48] to provide generated code for the cell integration over cut cell parts (see right box in Figure 14). These extensions provide the fundamental infrastructure for the evaluation of cut finite element based variational forms defined on fictitious domains and embedded surfaces. Consequently, given a high-level description of the variational formulation, low-level C++ code can be automatically generated for the evaluation of the cut element and surface integrals, in addition to the evaluation of integrals over the standard (non-cut) mesh entities. The generated code takes as input appropriate quadrature points and weights for each cut entity, which are provided by the libCutFEM library. The resulting cell integral matrix contributions are then assembled to the global matrix by routines provided in libCutFEM.

In addition to these quadrature and assembly routines, libCutFEM provides functionality for computing topological and geometric descriptions of the embedded surface and cut elements. Currently, libCutFEM mainly supports the level set based cutting algorithm as described in detail in the previous section. However, alternative geometrical algorithms can be integrated easily thanks to the modular structure of the library and FEniCS, which decouples the variational formulation, the quadrature and assembly routines, and the geometrical and mesh description.

In summary, in libCutFEM, one may specify variational forms defined over finite element spaces on fictitious domains in high-level UFL notation [43], define the background mesh \mathcal{T}_h , and give a description of the surface Γ , and then invoke the functionality provided by the libCutFEM library to automatically assemble the corresponding stiffness matrix. In particular, the numerical experiments presented in Sections 6.1–6.3, corresponding to the variational formulation defined by (50), (58) and (60), have been carried out using this technology. We also present the UFL scripts that were used to obtain the numerical results illustrating the simplicity by which the end user can access the CutFEM paradigm (Figures 17, 18, and 21). Other software projects including CutFEM style technology include GetFem++ [49], LifeV [50], and DUNE-UDG [51].

6. NUMERICAL RESULTS

6.1. Stabilized Nitsche fictitious domain method for the Poisson problem

Consider the following Poisson problem on the domain Ω :

$$\begin{aligned} -\Delta u &= f \text{ in } \Omega, \\ u &= 0 \text{ on } \Gamma, \end{aligned} \quad (48)$$

where Γ denotes the boundary of the domain Ω . Following [12], we consider the finite element space:

$$W^h = \{v \in C^0(\Omega_{\mathcal{T}}) : v|_K \in P^1(K), \forall K \in \mathcal{T}_h\} \quad (49)$$

and seek $U \in W^h$ for the stabilized Nitsche finite element formulation

$$A_h(U, v) = L(v), \quad \forall v \in W^h, \quad (50)$$

where

$$L(v) = (f, v)_{\Omega} \quad (51)$$

and

$$A_h(U, v) = a_h(U, v) + j(U, v) \quad (52)$$

with

$$a_h(U, v_h) = (\nabla U, \nabla v)_{\Omega} - (\partial_n U, v)_{\Gamma} - (\partial_n v, U)_{\Gamma} + (\gamma h^{-1} U, v)_{\Gamma} \quad (53)$$

and

$$j(U, v) = \sum_{F \in \mathcal{F}_G} (\gamma_1 h [\![\partial_{n_F} U]\!], [\![\partial_{n_F} v]\!])_F. \quad (54)$$

Here, the outward unit normal \mathbf{n}_{Γ} is determined by the level set through the formula

$$\mathbf{n}_{\Gamma} = \frac{\nabla \phi}{\|\nabla \phi\|_0}, \quad (55)$$

where ϕ is the level set function. In particular, we seek $\mathbf{n}_{\Gamma}^h \in [W^h]^d$ such that

$$\left(\|\nabla \phi_h\|_0 \mathbf{n}_{\Gamma}^h, v \right)_{\Omega_{\mathcal{T}}} = (\nabla \phi_h, v)_{\Omega_{\mathcal{T}}}, \quad \forall v \in [W^h]^d \quad (56)$$

We consider the solution of Equation (50) in two complex domains: the popcorn (45) and the olympic rings, which are constructed by taking the minimum over five tori specified in Equation (44). We set $f = 1$, $\gamma = 10.0$, and $\gamma_1 = 0.1$.

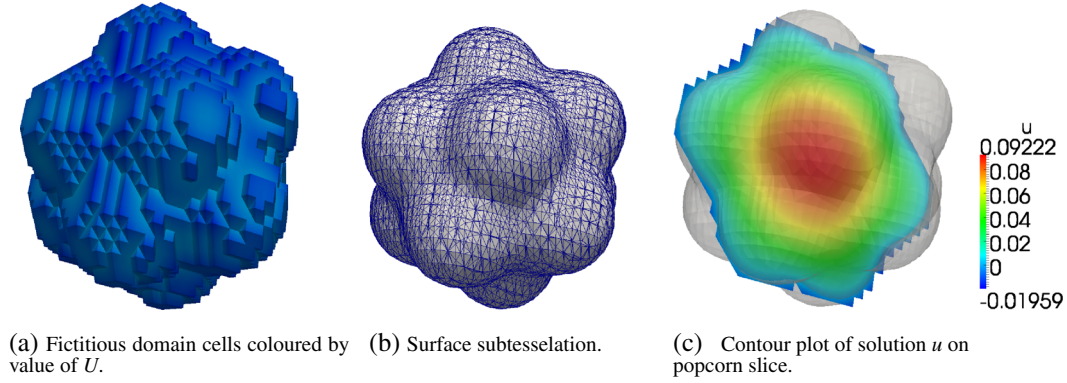


Figure 15. Solution of Poisson equation in popcorn geometry. (a) Fictitious domain cells coloured by value of U . (b) Surface subtessellation. (c) Contour plot of solution U on popcorn slice.

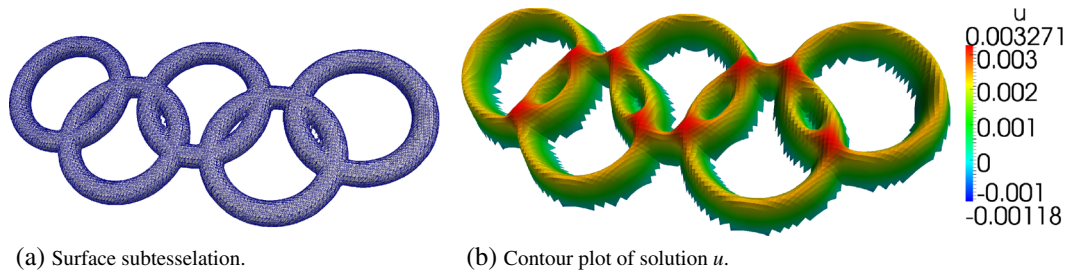


Figure 16. Solution of Poisson equation in olympic rings geometry. (a) Surface subtessellation. (b) Contour plot of solution U .

The solution u in the popcorn geometry is depicted in Figure 15. Figure 15a shows the fictitious domain mesh of the popcorn geometry colored by the value of U . Figure 15b displays the subtessellation of the popcorn surface, and Figure 15c shows a contour plot of U in a slice through the middle of the popcorn.

Solving the Poisson Equation (50) in the olympic ring geometry gives the result depicted in Figure 16, where Figure 16a shows the surface subtessellation, and Figure 16b shows the solution U in a slice through the middle of the olympic rings. The UFL input files to specify Equations (50 and (56) in the libCutFEM/FEniCS framework are displayed in Figures 17 and 18, respectively.

6.2. Stabilized Nitsche fictitious domain method for the Stokes' problem

Following [29, 30], we now employ the ghost penalty technique to solve Stokes' problem

$$\begin{aligned} -\Delta \mathbf{u} + \nabla p &= \mathbf{f} \quad \text{in } \Omega, \\ \nabla \cdot \mathbf{u} &= 0 \quad \text{in } \Omega, \\ \mathbf{u} &= \mathbf{g} \quad \text{on } \Gamma \end{aligned}$$

using a Nitsche fictitious domain method. We let the discrete velocity space V^h be the space of continuous, piecewise linear, \mathbb{R}^d -vector fields and let the discrete pressure space Q^h consist of either piecewise constant or continuous piecewise linear elements, denoted by $P_h^{0,dc}$ and P_h^1 , respectively. To formulate our CutFEM method for the Stokes problem, we introduce the forms

$$\begin{aligned} a_h(\mathbf{U}, \mathbf{v}) &= (\nabla \mathbf{U}, \nabla \mathbf{v})_\Omega - (\partial_n \mathbf{U}, \mathbf{v})_\Gamma - (\partial_n \mathbf{v}, \mathbf{U})_\Gamma + \gamma (h^{-1} \mathbf{U}, \mathbf{v})_\Gamma \\ b_h(\mathbf{v}, P) &= (-\nabla \cdot \mathbf{v})_\Omega + (\mathbf{n} \cdot \mathbf{v}, P)_\Gamma. \end{aligned}$$

Python code

```

cell = triangle
V = FiniteElement("CG", cell, 1)

u = TrialFunction(V)
v = TestFunction(V)

f = Coefficient(V)
g = Coefficient(V)

# Penalty parameter
gamma = Constant(cell)
beta = Constant(cell)

# Geometry
h = 2 * Circumradius(cell)
h_F = avg(h)
n = FacetNormal(cell)

# Create a measure for physical representation
physical = {"representation": "physical"}
dQ = dx(metadata=physical)

# Forms for standard cells
a = dot(grad(u), grad(v)) * dx
j = avg(beta) * h_F * dot(jump(grad(u), n), jump(grad(v), n)) * dS(1)
L = f * v * dx
a_std = a + j
L_std = L

# Cut-cell form
a_cut = dot(grad(u), grad(v)) * dQ
L_cut = f * v * dQ

# Surface form
N = VectorElement("DG", cell, 0)
n_s = Coefficient(N)
a_s = - dot(grad(u), n_s) * v * dQ - dot(grad(v), n_s) * u * dQ +
      gamma * 1.0 / h * u * v * dQ
L_s = - dot(grad(v), n_s) * g * dQ + gamma * 1.0 / h * g * v * dQ

forms = [a_std, L_std, a_cut, L_cut, a_s, L_s]

```

Figure 17. Input UFL file for the stabilized Nitsche fictitious domain method for the Poisson problem.

Python code

```

V = VectorElement("CG", cell, 1)
F = FiniteElement("CG", cell, 2)
n = TrialFunction(V)
v = TestFunction(V)

ls = Coefficient(F)

# Forms for standard cells
norm = sqrt(dot(grad(ls), grad(ls)))
a = norm * dot(n, v) * dx
L = dot(grad(ls), v) * dx

# Compile standard forms
forms = [a, L]

```

Figure 18. Input UFL file for the computation of the outside normal of the zero level set.

As explained in Section 3, we extend the coercivity of the bilinear form $a_h(\cdot, \cdot)$ from the physical to fictitious domain using the ghost penalty

$$j_u(U, \mathbf{v}) = \alpha_u \sum_{F \in \mathcal{F}_G} h_F (\llbracket \partial_{n_F} U \rrbracket, \llbracket \partial_{n_F} \mathbf{v} \rrbracket)_F.$$

As the mixed spaces $V^h \times P_h^{0,dc}$ and $V^h \times P_h^1$ are known to violate the Babuška–Brezzi conditions, we stabilize our scheme by the following symmetric pressure penalties:

$$j_p(P, q_h) = \begin{cases} \alpha_p \sum_{F \in \mathcal{F}_i} h_F (\llbracket P \rrbracket, \llbracket q \rrbracket)_F & \text{if } Q^h = P_h^{0,dc}, \\ \beta_p \sum_{F \in \mathcal{F}_i} h_F^3 (\llbracket \partial_{n_F} P \rrbracket, \llbracket \partial_{n_F} q \rrbracket)_F & \text{if } Q^h = P_h^1, \end{cases} \quad (57)$$

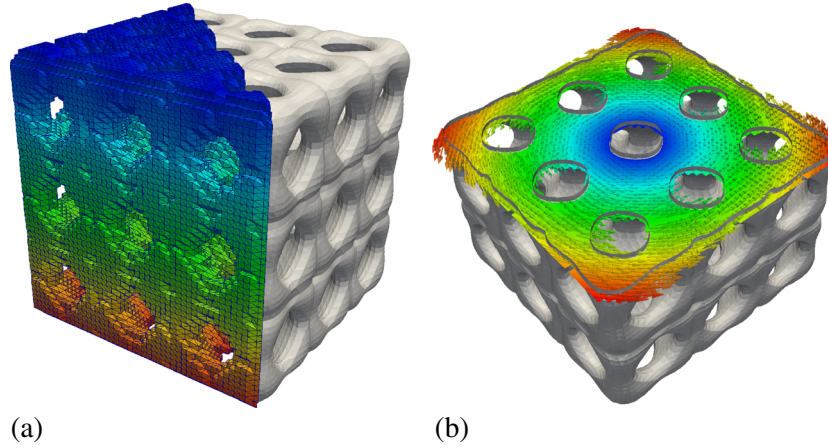


Figure 19. Stokes flow in a rotating Swiss cheese. (a) Pressure approximation. (b) Velocity approximation in a cross-section through the xy -plane.

where \mathcal{F}_i denotes all the interior faces in the mesh domain $\Omega_{\mathcal{T}}$. Note that the integral contributions in (57) are always evaluated on the *entire* face F , even for faces that are intersected by the boundary Γ . Again, such a ghost penalty ensures the robustness and optimal convergence properties of the stabilized scheme irrespective of the location of the boundary; see [29, 30] for detailed proofs.

Combining these forms, the stabilized Nitsche fictitious domain method for Stokes' problem reads: find $(U, P) \in V^h \times Q^h$ such that for all $(\mathbf{v}, q) \in V^h \times Q^h$

$$A_h(U, P; \mathbf{v}, q) + J_h(U, P; \mathbf{v}, q) = L_h(\mathbf{v}, q), \quad (58)$$

where

$$\begin{aligned} A_h(U, P; \mathbf{v}, q) &= a_h(U, \mathbf{v}) + b_h(\mathbf{v}, P) + b_h(U, q), \\ J_h(U, P; \mathbf{v}, q) &= j_u(U, \mathbf{v}) - j_p(P, q) \end{aligned}$$

and

$$L_h(\mathbf{v}, q) = (\mathbf{f}, \mathbf{v})_{\Omega} + (\gamma h^{-1} \mathbf{v} - \partial_n \mathbf{v} + q \mathbf{n}, \mathbf{g})_{\Gamma}.$$

We present two numerical examples. First, we compute the Stokes flow in the Swiss cheese domain while the domain is rotating along the z -axis at constant angle velocity. As volume force, we take $\mathbf{f} = (0, 0, -1)$. Figure 19 shows both the pressure approximation and the computed velocity. In the second example, we consider the flow through a blood vessel bifurcation with an aneurysm developed at the bifurcation; see Figure 20. Here, the vessel boundary is given as a surface mesh generated from biomedical image data, demonstrating the great potential of CutFEM technologies also to applications where only non-smooth boundary descriptions can be obtained.

6.3. Laplace–Beltrami problem on a surface using a bulk-mesh

We consider the Laplace–Beltrami problem for a surface Γ : find $u : \Gamma \rightarrow \mathbb{R}$ solving

$$-\Delta_{\Gamma} u = f \text{ on } \Gamma, \quad (59)$$

where Γ denotes the popcorn surface described by the level-set function (45). Then, the cut finite element method for the Laplace–Beltrami problem is to seek $U \in W_{\Gamma}^h$ such that

$$A(U, v) = (f, v)_{\Gamma} \quad \forall v \in W_{\Gamma}^h, \quad (60)$$

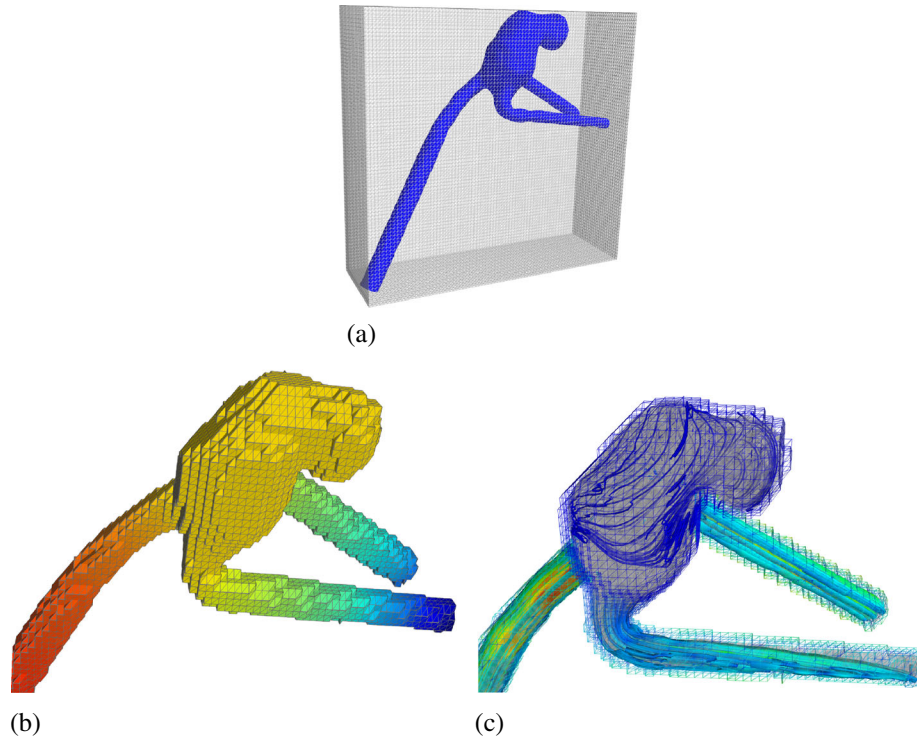


Figure 20. Stokes flow in an aneurysm. (a) Aneurysm embedded in a structured background mesh. (b) Pressure approximation on the 'reduced' background mesh. (c) Original aneurysm embedded in the background mesh shown with streamlines computed from the velocity approximation.

Python code

```
V = FiniteElement("CG", cell, 1)
F = FiniteElement("CG", cell, 2)
u = TrialFunction(V)
v = TestFunction(V)

f = Coefficient(F)
g = Coefficient(F)

# Penalty parameter
beta = Constant(cell)

# Geometry
h = 2*CurvatureRadius(cell)
h_F = avg(h)
n = FacetNormal(cell)

# Create a measure for physical representation
physical = {"representation": "physical"}
dQ = dx(metadata=physical)

j = avg(beta)*dot(jump(grad(u), n), jump(grad(v), n))*dS(1)
L = f*v*dx(0)

# Compile surface form
N = VectorElement("DG", cell, 0)
n_s = Coefficient(N)

def grad_pr(u,n):
    return grad(u) - inner(grad(u),n)*n

a_s = inner(grad_pr(u,n_s), grad_pr(v,n_s))*dQ + u*v*dQ
L_s = g*v*dQ

forms = [j, L, a_s, L_s]
```

Figure 21. Input UFL file for Laplace–Beltrami problem on a surface.

where the bilinear form $A(\cdot, \cdot)$ is defined by

$$A(U, v) = (\nabla_\Gamma U, \nabla_\Gamma v)_\Gamma + j(U, v) \quad (61)$$

with

$$j(U, v) = \sum_{F \in \mathcal{F}_i} (\llbracket \partial_{n_F} U \rrbracket, \llbracket \partial_{n_F} v \rrbracket)_F. \quad (62)$$

The right-hand side f is set to $f(x, y, z) = x + y + z$, satisfying the compatibility condition $(f, 1)_\Gamma = 0$. For the bulk mesh, the mesh size is approximately $h \approx 0.04$. Figure 22 depicts the solution U on the popcorn surface (45). Figure 21 shows the UFL input file used in libCutFEM to specify the Laplace–Beltrami problem.

6.4. Porous media flow in a domain with cracks

In this section, we give some preliminary numerical results obtained when solving (19) using the stabilized bilinear form

$$\begin{aligned} a_h^{\text{stab}}(U, v) := & \sum_i (\alpha_i \nabla U_i, \nabla v_i)_{\Omega_i} - (\llbracket U \rrbracket, \langle \alpha \partial_n v \rangle)_\Gamma \\ & - (\langle \alpha \partial_n U \rangle, \llbracket v \rrbracket)_\Gamma + (\gamma h^{-1} \llbracket U \rrbracket, \llbracket v \rrbracket)_\Gamma \\ & + (\alpha_\Gamma \{\nabla_\Gamma U\}, \{\nabla_\Gamma v\})_\Gamma + j_\Gamma(U, v). \end{aligned}$$

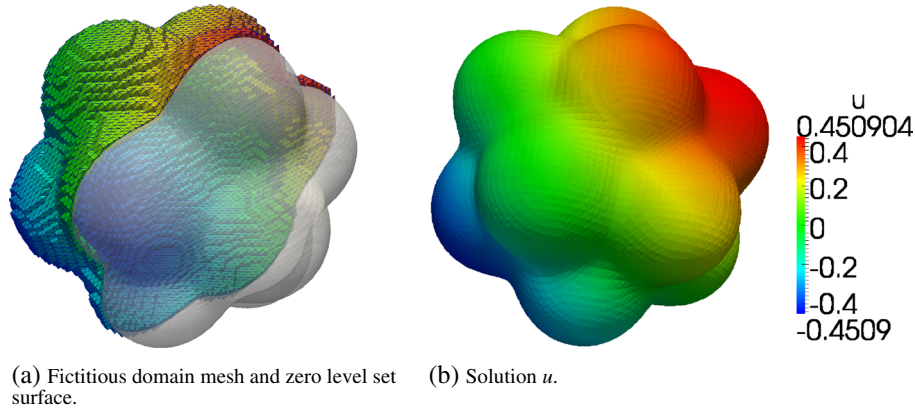


Figure 22. Solution u of Laplace–Beltrami problem on a popcorn surface. (a) Fictitious domain mesh and zero level set surface. (b) Solution u .

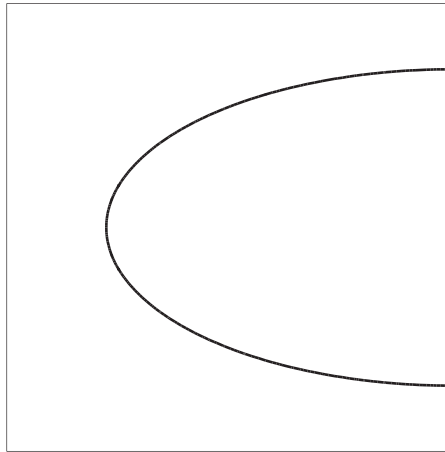


Figure 23. Crack with higher permeability in the upper arc.

The stabilization term $j_\Gamma(\cdot, \cdot)$ is necessary in the case when α_Γ/α becomes large, as we are then basically solving a perturbed Laplace–Beltrami problem.

We consider the following model problem in \mathbb{R}^2 : a domain $(0, 1) \times (0, 1)$ with Dirichlet boundary conditions $u = 1$ at $x = 0$ and $u = 0$ at $x = 1$, zero Neumann conditions elsewhere. Choosing $\alpha = 1$ in Ω and an elliptically shaped crack shown in Figure 23 and setting $\gamma_\Gamma = 10$, we obtain the following results when setting $\alpha_\Gamma = 0$ on the lower arc of the crack and $\alpha_\Gamma \in \{1, 2, 4\}$ in the upper arc. A close-up of the computed velocities $\alpha \nabla U$ close to the crack, using the computational mesh of Figure 24, is shown in Figure 25, together with a corresponding zoom of the mesh in Figure 26, and isolate plots of the pressures U are given in Figure 27. We note that even a small increase in permeability increases the velocity quite noticeably in the crack. Further developments for system of cracks are under way.

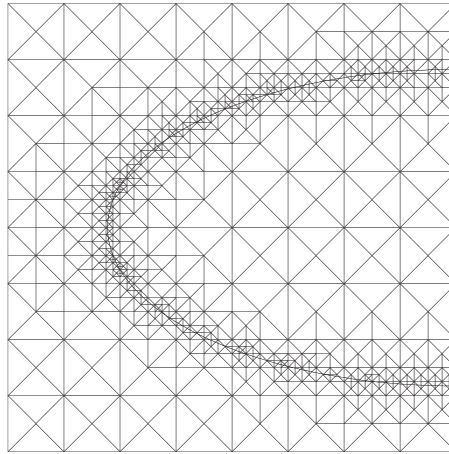


Figure 24. Computational mesh.

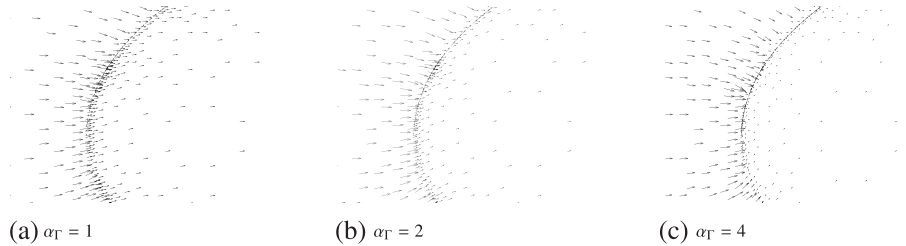


Figure 25. Computed velocities close to the crack. (a) $\alpha_\Gamma = 1$. (b) $\alpha_\Gamma = 2$. (c) $\alpha_\Gamma = 4$.

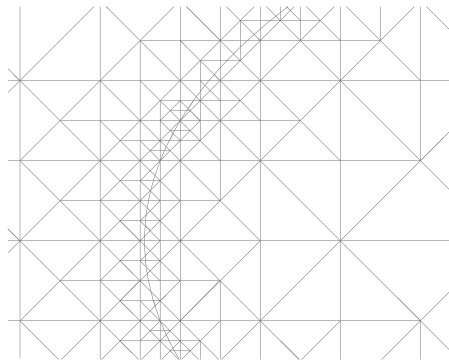
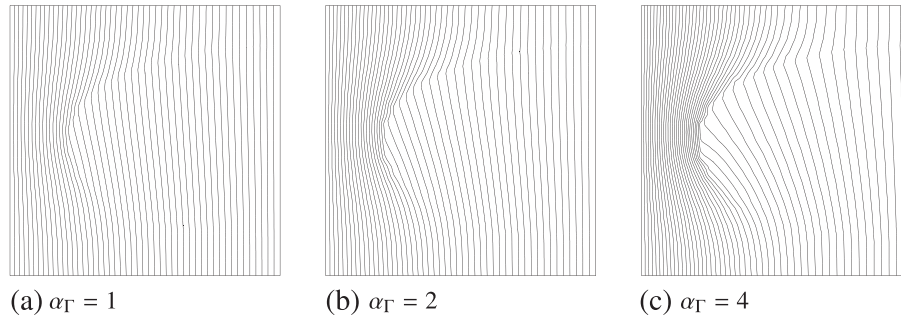


Figure 26. Zoom of the computational mesh where velocities are shown.

Figure 27. Pressure isolines. (a) $\alpha_\Gamma = 1$. (b) $\alpha_\Gamma = 2$. (c) $\alpha_\Gamma = 4$.

7. CONCLUDING REMARKS

In this paper, we have given an exposition of recent results on CutFEM combined with Nitsche's method and ghost penalty stabilization. The main theoretical ideas have been discussed, but emphasis has been put on implementation issues in the setting of the FEniCS software project.

We have endeavored to show the versatility of the CutFEM method as a method for fictitious domain computations, overlapping mesh methods, multiphysics coupling between a bulk domain and its surfaces or embedded interfaces, and model coupling problems. In particular, we have pointed out the possibility of posing and solving PDEs on interfaces/surfaces as well as in the bulk.

In conclusion, the results reported here (and in the references) show that CutFEM holds great promise as a versatile and powerful mesh-free method *posed on a mesh*. Challenges currently being investigated include optimization techniques using surface sensitivities, and moving interfaces, for example, for free surface problems.

ACKNOWLEDGEMENTS

This research was supported in part by EPSRC (first and second authors, Grant No. EP/J002312/2), in part by the Swedish Foundation for Strategic Research (second and third author, Grant No. AM13-0029) and the Swedish Research Council (second and third author, Grants No. 2011-4992 (PH) and No. 2013-4708 (MGL)). The work for this article was also supported by a Center of Excellence grant from the Research Council of Norway to the Center for Biomedical Computing at Simula Research Laboratory.

REFERENCES

1. Antiga L, Peiró J, Steinman DA. From image data to computational domains. In *MS&A*, Vol. 1, Formaggia L, Quarteroni A, Veneziani A (eds). Springer, 2009; 123–175.
2. Dassi F, Perotto S, Formaggia L, Ruffo P. Efficient geometric reconstruction of complex geological structures. *Mathematics and Computers in Simulation*, posted on 2014. DOI: 10.1016/j.matcom.2014.01.005, (to appear in print).
3. Hughes T JR, Cottrell JA, Bazilevs Y. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Computer Methods in Applied Mechanics and Engineering* 2005; **194**(39-41):4135–4195.
4. Nitsche J. Über ein Variationsprinzip zur Lösung Dirichlet-Problemen bei Verwendung von Teilräumen, die keinen Randbedingungen unterworfen sind. *Abhandlungen aus dem Abhandlungen Mathematische Seminar der Universität Hamburg* 1971; **36**:9–15.
5. Hansbo A, Hansbo P. An unfitted finite element method, based on Nitsche's method, for elliptic interface problems. *Computer Methods in Applied Mechanics and Engineering* 2002; **191**(47-48):5537–5552.
6. Hansbo A, Hansbo P, Larson MG. A finite element method on composite grids based on Nitsche's method. *ESAIM Mathematical Modelling and Numerical Analysis* 2003; **37**(3):495–514.
7. Hansbo A, Hansbo P. A finite element method for the simulation of strong and weak discontinuities in solid mechanics. *Computer Methods in Applied Mechanics and Engineering* 2004; **193**(33-35):3523–3540.
8. Hansbo P. Nitsche's method for interface problems in computational mechanics. *GAMM-Mitteilungen* 2005; **28**(2):183–206.

9. Becker R, Burman E, Hansbo P. A Nitsche extended finite element method for incompressible elasticity with discontinuous modulus of elasticity. *Computer Methods in Applied Mechanics and Engineering* 2009; **198**(41–44): 3352–3360.
10. Dolbow J, Harari I. An efficient finite element method for embedded interface problems. *International Journal for Numerical Methods in Engineering* 2009; **78**(2):229–252.
11. Harari I, Dolbow J. Analysis of an efficient finite element method for embedded interface problems. *Computational Mechanics* 2010; **46**(1):205–211.
12. Burman E, Hansbo P. Fictitious domain finite element methods using cut elements: II. A stabilized Nitsche method. *Applied Numerical Mathematics* 2011; **64**(4):328–341.
13. Schott B, Wall WA. A new face-oriented stabilized XFEM approach for 2D and 3D incompressible Navier–Stokes equations. *Computer Methods in Applied Mechanics and Engineering* 2014; **276**:233–265.
14. Lew AJ, Buscaglia GC. A discontinuous-Galerkin-based immersed boundary method. *International Journal for Numerical Methods in Engineering* 2008; **76**(4):427–454.
15. Rangarajan R, Lew A, Buscaglia GC. A discontinuous-Galerkin-based immersed boundary method with non-homogeneous boundary conditions and its application to elasticity. *Computer Methods in Applied Mechanics and Engineering* 2009; **198**(17–20):1513–1534.
16. Johansson A, Larson MG. A high order discontinuous Galerkin Nitsche method for elliptic problems with fictitious boundary. *Numerical Mathematics* 2013; **123**(4):607–628.
17. Haslinger J, Renard Y. A new fictitious domain approach inspired by the extended finite element method. *SIAM Journal on Numerical Analysis* 2009; **47**(2):1474–1499.
18. Burman E, Hansbo P. Fictitious domain finite element methods using cut elements: I. A stabilized Lagrange multiplier method. *Computer Methods in Applied Mechanics and Engineering* 2010; **199**(41–44):2680–2686.
19. Nicaise S, Renard Y, Chahine E. Optimal convergence analysis for the extended finite element method. *International Journal for Numerical Methods in Engineering* 2011; **86**(4–5):528–548.
20. Tur M, Albelda J, Nadal E, Ródenas JJ. Imposing Dirichlet boundary conditions in hierarchical cartesian meshes by means of stabilized Lagrange multipliers. *International Journal for Numerical Methods in Engineering* 2014; **98**(6):399–417.
21. Annavarapu C, Hautefeuille M, Dolbow JE. A robust Nitsche’s formulation for interface problems. *Computer Methods in Applied Mechanics and Engineering* 2012; **225–228**:44–54.
22. Barrau N, Becker R, Dubach E, Luce R. A robust variant of NXFEM for the interface problem. *Comptes Rendus Mathématique* 2012; **350**(15–16):789–792.
23. Juntunen M, Stenberg R. Nitsche’s method for general boundary conditions. *Mathematics of Computation* 2009; **78**(267):1353–1374.
24. Barrau N. Généralisation de la méthode NXFEM pour la discrétisation de problèmes d’interface elliptiques. *Ph.D. Thesis*, 2013.
25. Elliott CM, Ranner T. Finite element analysis for a coupled bulk–surface partial differential equation. *IMA Journal of Numerical Analysis* 2013; **33**(2):377–402.
26. Olshanskii MA, Reusken A, Grande J. A finite element method for elliptic equations on surfaces. *SIAM Journal on Numerical Analysis* 2009; **47**(5):3339–3358.
27. Burman E. Ghost penalty. *Comptes Rendus Mathématique* 2010; **348**(21–22):1217–1220.
28. Hansbo P, Larson MG, Zahedi S. A cut finite element method for a Stokes interface problem. *Applied Numerical Mathematics* 2014; **85**:90–114.
29. Massing A, Larson MG, Logg A, Rognes ME. A stabilized Nitsche fictitious domain method for the Stokes problem. *Journal of Scientific Computing* 2014; **61**(3):604–628.
30. Burman E, Hansbo P. Fictitious domain methods using cut elements: III. A stabilized Nitsche method for Stokes’ problem. *ESAIM Mathematical Modelling and Numerical Analysis* 2014; **48**(3):859–874.
31. Cattaneo L, Formaggia L, Iori G, Scotti A, Zunino P. Stabilized extended finite elements for the approximation of saddle point problems with unfitted interfaces. *Calcolo*, posted on 2014. DOI: 10.1007/s10092-014-0109-9, (to appear in print).
32. Burman E, Fernández MA. An unfitted Nitsche method for incompressible fluid–structure interaction using overlapping meshes. *Computer Methods in Applied Mechanics and Engineering* 2014; **279**(1):479–514.
33. Massing A, Larson MG, Logg A, Rognes ME. An overlapping mesh finite element method for a fluid–structure interaction problem. *ArXiv e-prints*, Unknown Month 11.
34. Juntunen M. A posteriori estimates for Nitsche’s method with discontinuous material parameters. *Presentation at ENUMATH* 2013.
35. Burman E, Zunino P. Numerical approximation of large contrast problems with the unfitted Nitsche method. In *Frontiers in Numerical Analysis—Durham 2010*, vol. 85, Lect. Notes Comput. Sci. Eng. Springer: Heidelberg, 2012; 227–282.
36. Dryja M. On discontinuous Galerkin methods for elliptic problems with discontinuous coefficients. *Computational Methods in Applied Mathematics* 2003; **3**(1):76–85.
37. Burman E, Zunino P. A domain decomposition method based on weighted interior penalties for advection–diffusion–reaction problems. *SIAM Journal on Numerical Analysis* 2006; **44**(4):1612–1638.
38. Burman E, Hansbo P, Larson MG. A stable cut finite element method for partial differential equations on surfaces: the Laplace–Beltrami operator. *ArXiv e-prints*, Unknown Month 12.

39. Sethian JA. *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science* (2nd ed.) Cambridge University Press: Cambridge, 1999.
40. Chern IL, Shu YC. A coupling interface method for elliptic interface problems. *Journal of Computational Physics* 2007; **225**(2):2138–2174.
41. Akio D, Koide A. An efficient method of triangulating equi-valued surfaces by using tetrahedral cells. *IEICE Transactions on Information and Systems* 1991; **74**(1):214–224.
42. Logg A, Mardal KA, Wells GN. (eds.) *Automated Solution of Differential Equations by the Finite Element Method: The FEniCS Book*. Springer: Berlin, 2012.
43. Alnæs MS, Logg A, Ølgaard KB, Rognes ME, Wells GN. Unified form Language: a domain-specific language for weak formulations of partial differential equations. *ACM Transactions on Mathematical Software* 2014; **40**(2): 9:1–9:37.
44. Kirby RC, Logg A. A compiler for variational forms. *ACM Transactions on Mathematical Software* 2006; **32**(3): 417–444.
45. Logg A, Mardal KA, Wells G. FFC: The FEniCS Form Compiler. In *Automated Solution of Differential Equations by the Finite Element Method*, vol. 84, Lecture Notes in Computational Science and Engineering. Springer Berlin Heidelberg, 2012; 227–238.
46. Alnæs MS, Logg A, Mardal KA. UFC: a finite element code generation interface. In *Automated Solution of Differential Equations by the Finite Element Method* vol. 84, Logg A, Mardal KA, Wells G (eds.), Lecture Notes in Computational Science and Engineering. Springer Berlin Heidelberg, 2012; 283–302.
47. Massing A, Larson MG, Logg A. Efficient implementation of finite element methods on non-matching and overlapping meshes in three dimensions. *SIAM Journal on Scientific Computing* 2013; **35**(1):C23–C47.
48. Massing A, Larson MG, Logg A, Rognes M. A stabilized Nitsche overlapping mesh method for the Stokes problem. *Numerical Mathematics* 2014; **128**(1):73–101.
49. GETFEM++, software package. (Available from: <http://download.gna.org/getfem/html/homepage/>). [Accessed on 27 October 2014].
50. LIFEV, software package. (Available from: <http://www.lifev.org/>). [Accessed on 27 October 2014].
51. Engwer C, Heimann F. Dune-udg: a cut-cell framework for unfitted discontinuous galerkin methods. In *Advances in Dune*, Dedner A, Flemisch B, Klöforn R (eds). Springer: Berlin, 2012; 89–100.