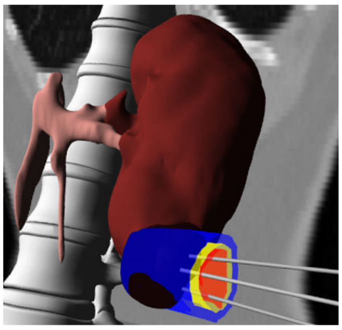
PROJECT REPORT V2

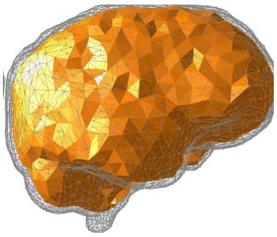
Simulation of soft tissues using innovative non-conforming Finite Element Methods

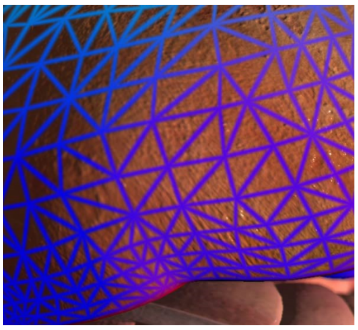
Supervisors
Michel DUPREZ
Stéphane COTIN

Student
Desmond Roussel NZOYEM

Referent teacher
Christophe PRUD'HOMME







This project was completed for the Master 2 CSMI,

from October 14, 2020, to January 26, 2021.

It was initiated by team MIMESIS

at Inria.

Academic year 2020 - 2021

January 28, 2021

Acknowledgements

I would like to thank Mr. DUPREZ for his uncountable advice, and all the time he spent helping get my code working. It would have been significantly harder (likely impossible) were it not for his indispensable help and assistance.

Contents

Chapter 1

Introduction

1.1 Environment

Inria is the National Institute for Research in Digital Science and Technology; **digital health** is one of their main research topics, and several teams are mobilized to face its challenges. At Inria, the **MIMESIS** team focuses on **real-time simulations for per-operative guidance**. The team develops **numerical techniques for real-time computation** and **data-driven simulations dedicated to patient-specific modeling**. Its global objective is to create a synergy between clinicians and scientists in order to create new technologies capable of redefining healthcare, with a strong emphasis on clinical translation (MIMESIS, 2020).

1.2 Context

The context of this project is the **development of a Finite Elements Method (FEM) adapted to computer-assisted surgery**. In recent years, numerical models using FEM to simulate the soft-tissue mechanisms of the human body have attracted a great interest from the scientific community. Models by FEM are among others, tools that contribute to the development of medical devices such as prosthesis and orthosis. They have the potential to improve strategies in planning and surgical assistance. In the context of computer-assisted surgery, it is essential that the FEM used is (Duprez, 2020b):

- **Quick**: given the context of real-time simulations ;
- **Precise**: in order to accurately guide the surgeon ;
- **Patient-specific**: involving complex body organ geometries.

1.3 Objectives

In the case of quasi-incompressibility for example, it is necessary to use hexahedral meshes[1] in order to avoid **locking** phenomena. However, there is no 3D mesh generator capable of meshing any geometry in an hexahedral manner. In order to completely avoid the creation of meshes, it is possible to perform numerical simulations on a **non-conforming** mesh[2], and to manage the boundary conditions by **penalization**, or using **Lagrange multipliers**. Such methods already exists (e.g.

[1] A hexahedral mesh can be defined as a geometric cell complex composed of 0-dimensional nodes, 1-dimensional edges, 2-dimensional quadrilaterals (residing in R^3), and 3-dimensional hexahedra (cubes, parallelepipeds, etc.)

[2] A non-conforming mesh is one which does not coincide with the boundary of the domain.

XFEM, CutFEM, SBM)[3], but their implementation generates other difficulties, among other things, the quadrature (computation of the integrals appearing in the FEM formulation).

In a preliminary study (Duprez and Lozinski, 2020; Duprez et al., 2020)[4], a new approach overcoming the afore-mentioned difficulty was developed. This method, called ϕ-FEM, uses a **Level Set** function that cancels itself at the edges of the domain. The main objective of this project is to **develop a ϕ-FEM technique for the dynamic of soft tissues, and potentially provide a mathematical proof of the method's convergence**. In order to achieve our main objective, we have divided the project into multiple milestones whose (intermediate) objectives are:

1. Understand the ϕ-FEM technique in question.
2. Reproduce the results from (Duprez and Lozinski, 2020) for the Poisson equation.
3. Develop a ϕ-FEM technique for the linear elasticity equation.
4. Perform simulations on body organ geometries.

1.4 Tools Needed

A mastery of the following tools and technologies is required to accurately complete the wide range of tasks in this project:

- **Python:** basic knowledge in Python is required in order to write scripts.
- **FEniCS:** this python library offers a relatively easy interface to implement weak formulations for PDEs.
- **Docker:** in order to set up the FEniCS environment.
- **Sympy:** to perform symbolic derivatives, needed to test our programs.

1.5 Mathematical fields involved

As for the mathematical aspect of the project, knowledge in several fields is needed. Those are:

- **Partial differential equations (PDE):** elasticity equations, biomechanics, etc.;
- **Scientific computing:** building and implementation of numerical schemes based on FEM;
- **Numerical analysis:** studying the convergence of a mathematical model.

1.6 Deliverables

At the end of the project, two deliverables were submitted as promised, all available on this GitHub repository :

- **A typewritten report** that can be found under `docs/pdfs/report-v2.pdf`.
- **A Python code base** corresponding to every problem we have solved. In each script, the variable `cvgStudy` (in the main section) has to be changed to `False` or `True` to indicate whether we want to run a single simulation, or perform a convergence study. The main files are:
  - **Poisson** Problem using the **classic FEM** formulation : `src/ClassicFEM/Poisson.py`.

[3] References for these techniques can be respectively found at (Fries, n.d.), (Burman et al., 2015), and (Atallah et al., 2020).

[4] Henceforth, these two papers will simply be referred to as "the papers".

- **Poisson** Problem using the ϕ-**FEM** formulation : `src/PhiFEM/Poisson.py`.
- **Elasticity** equation using the **classic FEM** formulation : `src/ClassicFEM/Elasticity2D.py`.
- **Elasticity** equation using the ϕ-**FEM** formulation : `src/PhiFEM/Elasticity2D.py`.

Chapter 2

Results

2.1 Presentation of the ϕ-FEM technique

2.1.1 The Classic FEM approach

As indicated by its name, the ϕ-FEM technique is based on the classic FEM technique (Ern and Guermond, 2013, p.111). Let Ω be a bounded domain in $\mathbb{R}^d$ with a regular boundary $\partial\Omega$. Let $\mathcal{L}$ be a scalar elliptic operator (Evans, 1998). Given a function $f \in L^2(\Omega)$, and another function $g \in L^2(\partial\Omega)$, the problem is to solve

$$\begin{cases} \mathcal{L}u = f & \text{in } \Omega \\ u = g & \text{on } \partial\Omega\,. \end{cases} \tag{2.1}$$

Except for the non-homogeneous Dirichlet case [1], problems of the form (2.1) can be transformed into a weaker form called variational formulation :

$$\text{Seek } u \in V \text{ such that} \quad a(u,v) = f(v), \quad \forall v \in V \tag{2.2}$$

where V is a Hilbert space satisfying

$$H_0^1(\Omega) \subset V \subset H^1(\Omega)\,.$$

Moreover, a is a bilinear form defined on $V \times V$, and f is a linear form defined on V.

The Poisson problem ($\mathcal{L} := -\Delta$) with homogeneous boundary condition is a classic example of an elliptical PDE that can be formulated as in (2.2). It is defined as

$$\begin{cases} -\Delta u = f & \text{in } \Omega \\ u = 0 & \text{on } \partial\Omega \end{cases} \tag{2.3}$$

and its weak formulation

$$\text{Seek } u \in H_0^1(\Omega) \text{ such that} \quad a(u,v) = f(v), \quad \forall v \in H_0^1(\Omega). \tag{2.4}$$

Once the variational form (2.2) has been obtained, it has to be approximated [2]. Assuming the boundary $\partial\Omega = \Gamma$ is sufficiently smooth, let $\mathcal{T}_h$ be a quasi-uniform simplicial mesh on Ω of mesh size h. The domain occupied by the mesh elements is denoted as $\Omega_h = (\cup_{T\in\mathcal{T}_h} T)^o$. Let's consider,

[1] The non-homogeneous Dirichlet case can be brought back to the 2.2 by writing $u = u_g + \Phi$, where u_g is a lifting function, and Φ satisfies 2.2 where $V = H_0^1(\Omega)$

[2] In order to have uniform notations everywhere in this section, let's use the notations from (Duprez and Lozinski, 2020).

for an integer $k \geq 1$, the finite element space

$$V_h^{(k)} = \{v_h \in H_0^1(\Omega_h) : v_h|_T \in \mathbb{P}_k(T) \,\forall\, T \in \mathcal{T}_h\} ,$$

where $\mathbb{P}_k(T)$ stands for the space of polynomials in d variables of degree $\leq k$ viewed as functions on T. The FEM approximation can now be derived from (2.2) as

$$\text{Seek } u_h \in V_h^{(k)} \text{ such that } \quad a_h(u_h, v_h) = l_h(v_h), \quad \forall v_h \in V_h^{(k)} \tag{2.5}$$

where the bilinear form a_h and the linear form l_h are defined on $V_h^{(k)} \times V_h^{(k)}$ and $V_h^{(k)}$ respectively. In the case of the Poisson problem (2.3), a_h and l_h are defined as:

$$\begin{aligned} a_h(u,v) &= \int_{\Omega_h} \nabla u \cdot \nabla v \,, \\ l_h(v) &= \int_{\Omega_h} f v \,. \end{aligned} \tag{2.6}$$

2.1.2 XFEM, CutFEM, and SBM

XFEM, CutFEM, SBM, and ϕ-FEM are all part of a family of numerical techniques called **immersed boundary methods**. These techniques use a non-body conforming grid. In general, the body does not align with the grid (meaning its boundaries do not match the mesh). So, the computational cells will have to be adapted (cut, shifted, etc.). The advantages of using an immersed boundary method is that grid generation is much easier[3]. Another benefit is that grid complexity and quality are not significantly affected by the complexity of the geometry when carrying out a simulation. Also, an immersed boundary method can handle moving boundaries, due to the stationary non-deforming grid (Bandringa, 2010).

The Extended Finite Element Method (**XFEM**) (Moës et al., 1999) is specifically designed to treat discontinuities. It is a numerical method that enables a local enrichment of the approximation space. The enrichment is realized through the partition of unity (PUM) concept. The method is particularly useful for the approximation of solutions with pronounced non-smooth characteristics in small parts of the computational domain, for example near discontinuities and singularities (see fig. 2.1).

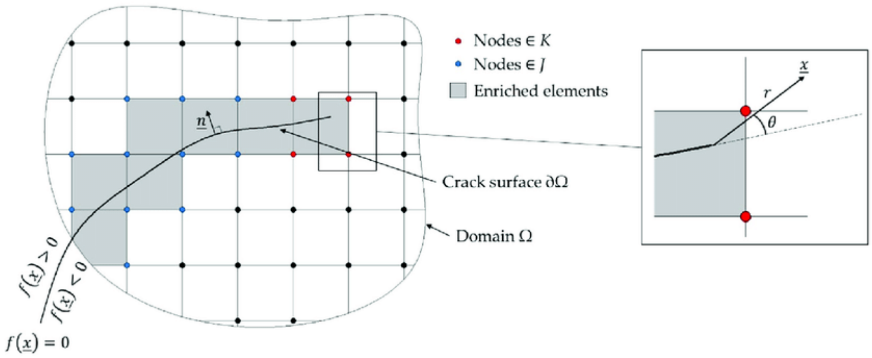


FIGURE 2.1: Illustration of the extended finite element method (XFEM) approach. The mesh is locally enriched as the crack propagates, in order to avoid remeshing (De Cicco and Taheri, 2018).

[3] This is because the body does not necessarily have to fit the grid.

In the **CutFEM** approach (Burman and Hansbo, 2010), the boundary of a given domain is represented on a background grid. CutFEM partitions domains into Ω_1 (exterior) and Ω_2 (interior); separates solutions for Ω_1 and Ω_2; captures jumps; integrates over intersections of each element with subdomains; and finally, it weakly imposes boundary and jump conditions (fig. 2.2).

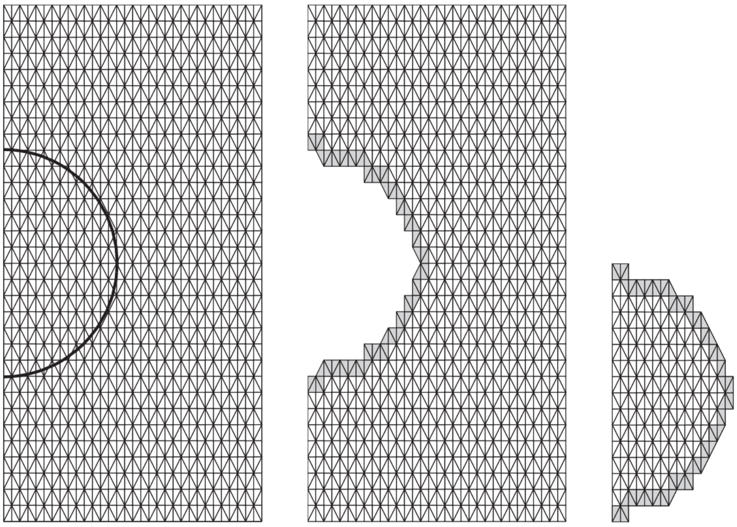


FIGURE 2.2: Illustration of the CutFEM approach. A mesh with the interface indicated is being divided into two new meshes. Ω_1 is the exterior (in the middle) and Ω_2 is the interior (on the right). The doubled elements (in both sub-meshes) are shaded (Burman et al., 2015).

SBM stands for Shifted Boundary Method (Main and Scovazzi, 2018). In the SBM, the location where boundary conditions are applied is shifted from the true boundary to an approximate (surrogate) boundary; and, at the same time, modified (shifted) boundary conditions are applied in order to avoid a reduction of the convergence rates of the overall formulation (fig. 2.3).

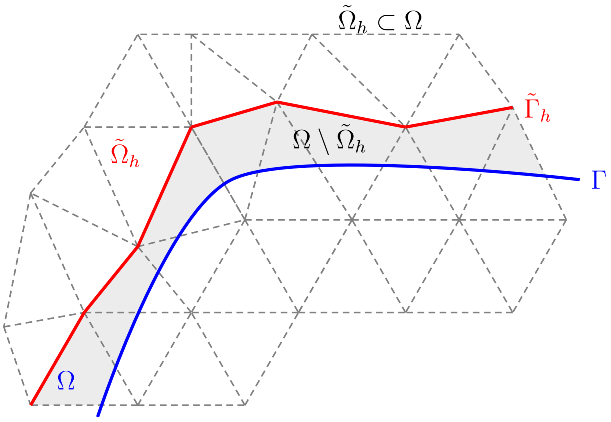

$\tilde{\Omega}_h \subset \Omega$
$\tilde{\Gamma}_h$
$\Omega \setminus \tilde{\Omega}_h$
$\tilde{\Omega}_h$
Γ
Ω

FIGURE 2.3: Illustration of the SBM (Shifted Boundary Method). The original boundary is in blue, while the shifted boundary is in red (Atallah et al., 2020).

While these methods (and others based on them) are effective, the integrals over Ω are kept in the discretization; which in practice, is cumbersome since one needs to implement integrations on the boundary Γ, and on parts of mesh elements cut by the boundary. Multiple attempts have been made to alleviate this practical difficulty with methods that do not require to perform the integration on the cut elements, but still need the integration on Γ. ϕ-FEM avoids any integration whatsoever on Γ.

2.1.3 Formal presentation of the ϕ-FEM technique

As of January 28, 2021, ϕ-FEM has only been developed and tested on the Poisson equation. In the coming paragraphs, we will present the general idea behind the technique, for any common elliptical equation (2.1).

In ϕ-FEM, the boundary condition is carried by the level-set function; and as hinted, all the integrations in ϕ-FEM are performed on the whole mesh elements, and there are no integrals on Γ (even if the mesh element in question is cut by the boundary Γ). As stated in the introduction, the domain's boundary is the region in space where the level-set function ϕ vanishes. This level-set function is chosen such that it remains strictly negative in the interior of the domain. The level-set function is assumed to be smooth, and to behave near Γ as the signed distance to Γ. As an indication, the following problem [4]

$$\exists\,?\,u \text{ such that } \begin{cases} \mathcal{L}u = f & \text{in } \Omega \\ u = 0 & \text{on } \partial\Omega \end{cases}$$

is reformulated in ϕ-FEM as (see fig. 2.4)

$$\exists\,?\,w \text{ such that } \begin{cases} \mathcal{L}(\phi w) = f & \text{in } \Omega \\ \text{where } \Omega = \{\phi < 0\}, & u = \phi w\,. \end{cases}$$

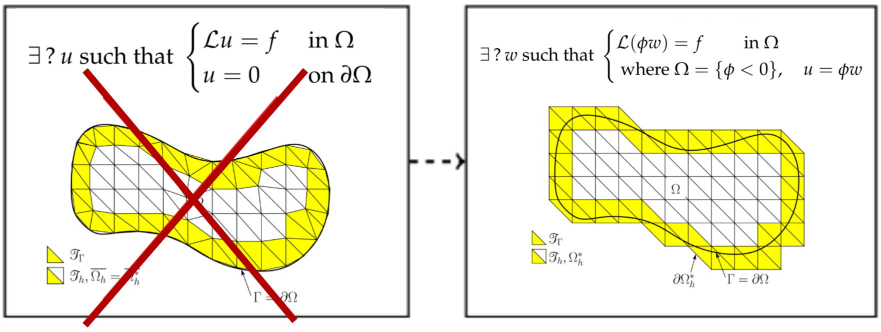

$\exists\,?\,u$ such that $\begin{cases} \mathcal{L}u = f & \text{in } \Omega \\ u = 0 & \text{on } \partial\Omega \end{cases}$
$\mathcal{T}_\Gamma$
$\mathcal{T}_h, \overline{\Omega_h} = \mathcal{T}_h^*$
$\Gamma = \partial\Omega$
$\exists\,?\,w$ such that $\begin{cases} \mathcal{L}(\phi w) = f & \text{in } \Omega \\ \text{where } \Omega = \{\phi < 0\}, & u = \phi w \end{cases}$
Ω
$\mathcal{T}_\Gamma$
$\mathcal{T}_h, \Omega_h^*$
$\partial\Omega_h^*$
$\Gamma = \partial\Omega$

FIGURE 2.4: Conversion of a problem from Classic FEM to ϕ-FEM in order to avoid complications due to meshing. Ω and $\partial\Omega$ are defined by the smooth level-set function ϕ. $\partial\Omega^* = \Gamma^*$ is a fictitious boundary defined by all the cells cut by ϕ. It is on this boundary that integrations will be performed (Duprez, 2020b).

As indicated in fig. 2.4, the bounded domain's boundary $\partial\Omega = \Gamma$ is assumed to be smooth and given by a level-set function ϕ. As stated in the introduction, the domain's boundary will often be the region in space where the level-set function ϕ vanishes. This level-set function is chosen such that it remains strictly negative in the interior of the domain:

$$\Omega := \{\phi < 0\} \quad \text{and} \quad \Gamma := \{\phi = 0\}.$$

The level-set method allows for treatment of internal boundaries and interfaces without any explicit treatment of the interface geometry. Such a representation is typical in **immersed boundary**

[4] This is eq. (2.1) with a homogeneous Dirichlet boundary condition.

techniques, or **fictitious domain methods**. It is important for dealing with complex boundaries and problems with evolving surfaces or interfaces. This provides a convenient and an appealing mean for tracking moving interfaces, such as the ones of living organs.

Now let's provide a precise formulation for the Poisson equation (2.3) using ϕ-FEM. As Ω is a bounded domain, we can write $\Omega \subset \mathcal{O} \subset \mathbb{R}^d$ ($d = 2,3$). Let $\mathcal{T}_h^{\mathcal{O}}$ be a quasi-uniform simplicial mesh on $\mathcal{O}$ of mesh size h. Let's consider, for an integer $l \geq 1$, the finite element space

$$V_{h,\mathcal{O}}^{(l)} = \{v_h \in H^1(\mathcal{O}) : v_h|_T \in \mathbb{P}_l(T)\ \forall T \in \mathcal{T}_h^{\mathcal{O}}\},$$

where $\mathbb{P}_l(T)$ stands for the space of polynomials in d variables of degree $\leq l$ viewed as functions on T. We define ϕ_h as

$$\phi_h := I_{h,\mathcal{O}}^{(l)}(\phi),$$

where $I_{h,\mathcal{O}}^{(l)}$ is the standard Lagrange interpolation operator on $V_{h,\mathcal{O}}^{(l)}$. Henceforth, this function will be used to approximate the physical domain and its boundary. Next we introduce the computational mesh $\mathcal{T}_h$ as the subset of $\mathcal{T}_h^{\mathcal{O}}$ composed of the triangles/tetrahedrons having a non-empty intersection with the approximate domain $\{\phi_h < 0\}$. We note the domain occupied by $\mathcal{T}_h$ as Ω_h i.e.

$$\mathcal{T}_h := \{T \in \mathcal{T}_h^{\mathcal{O}} : T \cap \{\phi_h < 0\} \neq \emptyset\} \quad \text{and} \quad \Omega_h = (\cup_{T\in\mathcal{T}_h} T)^o.$$

Now the function space will be defined as

$$V_h^{(k)} = \{v_h \in H^1(\Omega_h) : v_h|_T \in \mathbb{P}_k(T)\ \forall\ T \in \mathcal{T}_h\}.$$

The ϕ-FEM approximation is introduced as follows: find $w_h \in V_h^{(k)}$ such that:

$$a_h(w_h, v_h) = l_h(v_h) \text{ for all } v_h \in V_h^{(k)}, \tag{2.7}$$

where the bilinear form a_h and the linear form l_h are defined by

$$a_h(w,v) := \int_{\Omega_h} \nabla(\phi_h w) \cdot \nabla(\phi_h v) - \int_{\partial\Omega_h} \frac{\partial}{\partial n}(\phi_h w)\phi_h v + G_h(w,v), \tag{2.8}$$

and

$$l_h(v) := \int_{\Omega_h} f\phi_h v + G_h^{rhs}(v),$$

with G_h and G_h^{rhs} standing for

$$G_h(w,v) := \sigma h \sum_{E\in\mathcal{F}_h^\Gamma} \int_E \left[\frac{\partial}{\partial n}(\phi_h w)\right]\left[\frac{\partial}{\partial n}(\phi_h v)\right] + \sigma h^2 \sum_{T\in\mathcal{T}_h^\Gamma} \int_T \Delta(\phi_h w)\Delta(\phi_h v),$$

$$G_h^{rhs}(v) := -\sigma h^2 \sum_{T\in\mathcal{T}_h^\Gamma} \int_T f\Delta(\phi_h v),$$

where $\sigma > 0$ is an h-independent stabilization parameter, $\mathcal{T}_h^\Gamma \subset \mathcal{T}_h$ contains the mesh elements cut by the approximate boundary $\Gamma_h = \{\phi_h = 0\}$, *i.e.*

$$\mathcal{T}_h^\Gamma = \{T \in \mathcal{T}_h : T \cap \Gamma_h \neq \emptyset\}, \quad \Omega_h^\Gamma := \left(\cup_{T\in\mathcal{T}_h^\Gamma} T\right)^o;$$

and $\mathcal{F}_h^\Gamma$ collects the interior facets of the mesh $\mathcal{T}_h$ either cut by Γ_h or belonging to a cut mesh element

$$\mathcal{F}_h^\Gamma = \{E \text{ (an internal facet of } \mathcal{T}_h) \text{ such that } \exists T \in \mathcal{T}_h : T \cap \Gamma_h \neq \emptyset \text{ and } E \in \partial T\}.$$

The brackets inside the integral over $E \in \mathcal{F}_h^\Gamma$ in the formula for G_h stand for the jump over the facet E. The first part in G_h actually coincides with the ghost penalty as introduced in (Burman, 2010) for P_1 finite elements.

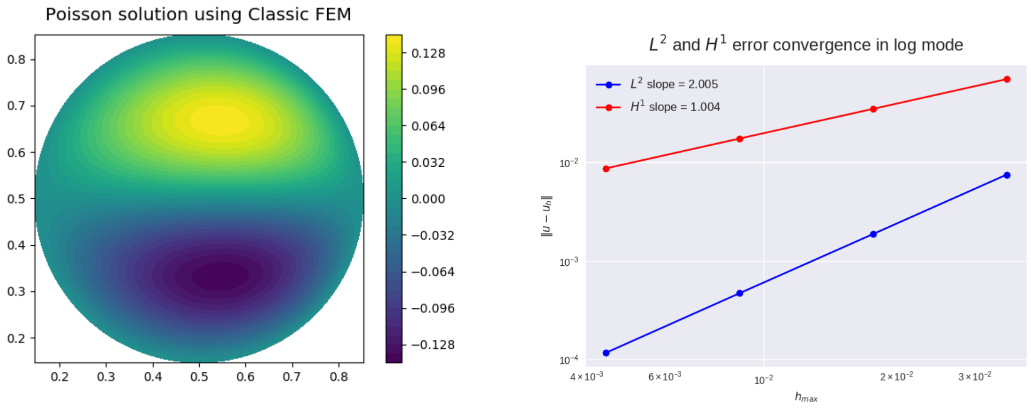
2.2 The Poisson problem

2.2.1 Using classic FEM

The basis for the weak formulation has been laid down in (2.6). The values for f, and the domain Ω need to be defined in order to run the test cases. Moreover, the exact solution u has to be known in order to perform a convergence study.

The classic FEM technique has been tested on a particular case. The case in question is the **test case 1** from Duprez and Lozinski, 2020, p.15. The corresponding parameters are presented below [5], and the result is shown in fig. 2.5.

$$\begin{cases} \Omega = \left\{(x,y) \in \mathbb{R}^2 : \left(x - \frac{1}{2}\right)^2 + \left(y - \frac{1}{2}\right)^2 < \frac{1}{8}\right\} \\ u(x,y) = -\left(\frac{1}{8} - \left(x - \frac{1}{2}\right)^2 - \left(y - \frac{1}{2}\right)^2\right) \exp(x) \sin(2\pi y) \\ f(x,y) = -\frac{\partial^2 u}{\partial x^2}(x,y) - \frac{\partial^2 u}{\partial y^2}(x,y) \end{cases} \tag{2.9}$$



(A) Solution

(B) Convergence study

FIGURE 2.5: Results obtained when applying the classic FEM technique to the Poisson equation (2.12). The H^1 norm presented in (B) is the norm $\|u - u_h\|_{1,\Omega}$. These results confirm what we expect since and will serve as a reference for the following results. The red and blue plots in (B) indicate the relative error.

The results we obtained (fig. 2.5b) fully match the expected theoretical framework. For the Poisson problem, the expected convergence is described below (Ern and Guermond, 2013, p.121).

[5] Notice that f is very cumbersome to compute manually ; that is why we will be using Sympy for such tasks.

As $u \in H^1(\Omega)$[6], we know that the error converges according to the inequalities [7]:

$$\|u - u_h\|_{L^2(\Omega)} \leq Ch^2 |u|_{H^2(\Omega)}\,, \tag{2.10}$$

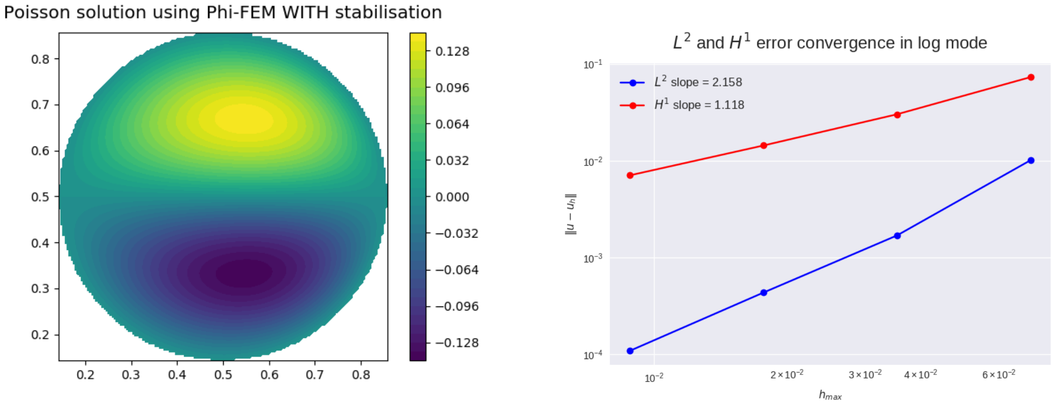
$$\|u - u_h\|_{H^1(\Omega)} \leq Ch^1 |u|_{H^2(\Omega)}\,. \tag{2.11}$$

2.2.2 Using ϕ-FEM

The basis for the Poisson problem in ϕ-FEM have been laid in (2.8). Multiple parameters must be defined in order to have a proper test case.

Let's repeat the same test case we did in classic FEM. The test case (defined in (2.9)), is redefined below. The result is presented at fig. 2.6.

$$\begin{cases} \mathcal{O} = [0,1] \times [0,1] \\ \phi(x,y) = -\frac{1}{8} + \left(x - \frac{1}{2}\right)^2 + \left(y - \frac{1}{2}\right)^2 \\ u(x,y) = \phi(x,y) \times \exp(x) \times \sin(2\pi y) \\ f(x,y) = -\frac{\partial^2 u}{\partial x^2}(x,y) - \frac{\partial^2 u}{\partial y^2}(x,y) \\ \sigma = 20 \end{cases} \tag{2.12}$$



(A) Solution

(B) Convergence study

FIGURE 2.6: Results obtained when applying the ϕ-FEM technique with stabilization to the Poisson equation. The H^1 norm presented in (B) is the norm $\|u - u_h\|_{1,\Omega\cap\Omega_h}$, for better comparison with the classic FEM case. These results confirm what we expected, with better slopes compared to fig. 2.5.

These results were expected. In fact, they are better than expected, and much better than classic FEM. When the right-hand side of the Poisson equation f belongs to $H^k(\Omega_h \cup \Omega)$, the convergence rates should be (Duprez and Lozinski, 2020, p.4)[8]:

$$|u - u_h|_{1,\Omega\cap\Omega_h} \leq Ch^k \|f\|_{k,\Omega\cup\Omega_h}\,, \tag{2.13}$$

$$\|u - u_h\|_{0,\Omega} \leq Ch^{k+1/2} \|f\|_{k,\Omega_h}\,. \tag{2.14}$$

[6] We can also prove that $u \in H^2(\Omega)$ using the elliptic regularity theorem.

[7] u is the exact solution interpolated in the variational space, and u_h is our FEM approximation.

[8] A number of assumptions had to be made. Those are not repeated here.

Concerning the fact that the solution (on the domain's boundary) in fig. 2.6a doesn't look as smooth as in fig. 2.5a, we can say that it was perfectly anticipated due to the way the meshes are generated in FEniCS. The reason is that: using triangles, a 2D disk is very easily meshable, which is what is done in Classic FEM; thus the smoothness on the boundary. The ϕ-FEM approach will come in handy when the geometry will not be meshable, and will thus have to be immersed in a larger mesh.

2.3 The elasticity equation

2.3.1 Using Classic FEM

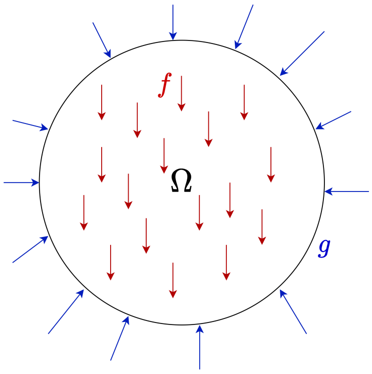


FIGURE 2.7: Illustration of a linear elasticity problem (with small deformations). The domain Ω is a disk. The vertical force f can be associated to the gravity. The displacement vector g on the boundary is fixed.

With $u : \Omega \mapsto \mathbb{R}^d$, $\sigma(u)$ and $\varepsilon(u) = \frac{1}{2}(\nabla u + \nabla u^T)$ representing (respectively) the displacement, the stress and the strain tensors, the model problem (illustrated at fig. 2.7) is as follows (Ern and Guermond, 2013, p.153) :

$$\begin{cases} \nabla \cdot \sigma(u) + f = 0 & \text{in } \Omega \\ \sigma(u) = \lambda \operatorname{tr}(\varepsilon(u))\mathcal{I} + 2\mu\varepsilon(u) & \text{in } \Omega \\ u = g & \text{on } \partial\Omega \end{cases} \tag{2.15}$$

where λ and μ are the Lamé coefficients. With $d = 2, 3$, the test functions are taken in the functional space

$$V = \left[H_0^1(\Omega)\right]^d .$$

We have

$$-\int_\Omega \nabla \cdot \sigma(u) \cdot v = \int_\Omega f \cdot v \qquad \forall v \in V$$

equivalent to (using Green's formula)

$$\int_\Omega \sigma(u) : \nabla v - \int_{\partial\Omega} (\sigma(u) \cdot n) \cdot v = \int_\Omega f \cdot v . \tag{$\star$}$$

Lets remember that the matrix dot product (double dots ":" operation) of a symmetric and a skew-symmetric matrix is zero. We know that $\sigma(u)$ is symmetric according to its definition in (2.15). We then decompose ∇v into the sum of a symmetric part $\nabla^s v = \frac{1}{2}\left(\nabla v + \nabla v^T\right)$ and a skew-symmetric part $\nabla^a v = \frac{1}{2}\left(\nabla v - \nabla v^T\right)$. Since $\sigma(u) : \nabla^a v = 0$, we get (by also taking $u = 0$ on the boundary) :

$$\int_\Omega \sigma(u) : \nabla^s v = \int_\Omega f \cdot v\,,$$

that is

$$\int_\Omega \sigma(u) : \varepsilon(v) = \int_\Omega f \cdot v\,.$$

This leads to the following weak formulation in classic FEM

$$\text{Seek } u \in G + V \text{ such that} \quad a(u,v) = l(v), \quad \forall v \in V\,, \tag{2.16}$$

where G as is a set of functions such that its elements' "trace" on the boundary $\partial\Omega$ is equal to g. The functionals a and l are defined as

$$a(u,v) = \int_\Omega \sigma(u) : \varepsilon(v)$$

and

$$l(v) = \int_\Omega f \cdot v\,.$$

Let's look at the a priori error estimation (Ern and Guermond, 2013, p.159). As $u \in V = \left[H^1(\Omega)\right]^d$[9], we know that the errors converge according to the inequalities below[10]:

$$\|u - u_h\|_{L^2(\Omega)^d} \leq Ch^2 |u|_{H^2(\Omega)^d} \tag{2.17}$$
$$\|u - u_h\|_{H^1(\Omega)^d} \leq Ch^1 |u|_{H^2(\Omega)^d} \tag{2.18}$$

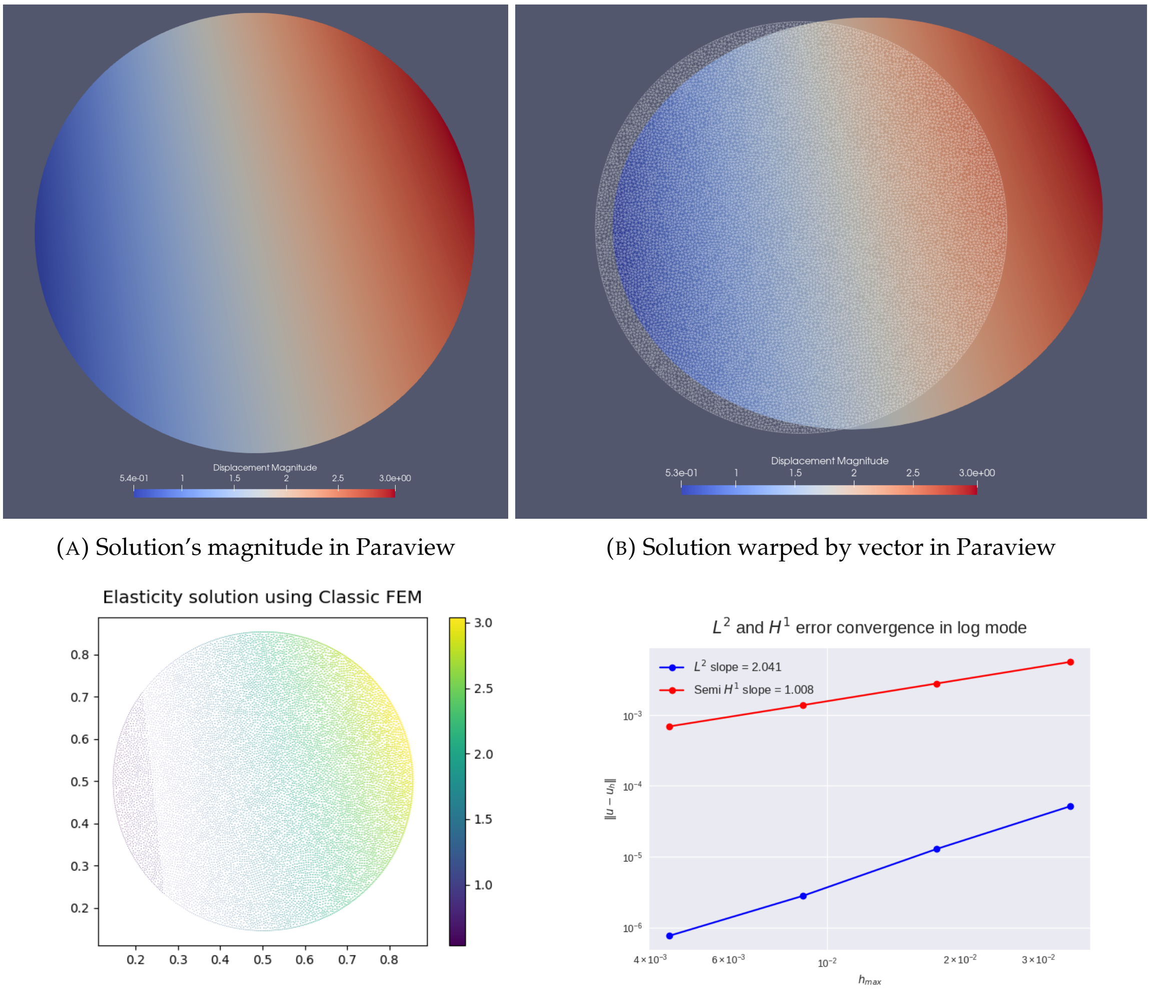
The classic FEM technique has been tested in FEniCS for the elasticity equation on a particular case[11]. The results, implemented in FEniCS in a 2D variational space, are presented in fig. 2.8.

$$\begin{cases} \Omega = \left\{(x,y) \in \mathbb{R}^2 : \left(x - \frac{1}{2}\right)^2 + \left(y - \frac{1}{2}\right)^2 < \frac{1}{8}\right\} \\ u(x,y) = \begin{pmatrix} 2x + \sin(x)\exp(y) \\ \frac{x}{2} + \cos(x) - 1 \end{pmatrix} \\ f(x,y) = -\nabla \cdot \sigma(u(x,y)) \\ g(x,y) = u(x,y) \end{cases} \tag{2.19}$$

[9] It is important to note that $u \in \left[H^2(\Omega)\right]^d$ according to the elliptic regularity theorem.

[10] As with the Poisson problem, u is the exact solution interpolated in the variational space, and u_h is our finite element approximation.

[11] Let's note that $f(x,y)$ was computed using the Sympy library ; the result is too long to the printed here.



(A) Solution's magnitude in Paraview

(B) Solution warped by vector in Paraview

(C) Solution's components (vector) in Matplotlib

(D) Convergence study

FIGURE 2.8: Results obtained when applying the classic FEM technique to the elasticity equation (2.15). The warp by vector (B) shows that the elastic material is expanding to its right, but contracting to its left. The convergence study (D) in done with relative errors. The H^1 slope indicated in (D) is the H^1 semi-norm $|u - u_h|_{1,\Omega}$.

2.3.2 Using ϕ-FEM

Let's derive the weak formulation based on (2.15)[12]. Since our whole boundary will consist of Dirichlet conditions, we go from eq. ($\star$) to write $u = \phi w + g$, and v becomes ϕv[13]. We use Green's formula just as in the classic FEM, and we get

$$\int_\Omega \sigma(\phi w + g) : \varepsilon(\phi v) - \int_{\partial\Omega} (\sigma(\phi w + g) \cdot n) \cdot \phi v = \int_\Omega f \cdot \phi v \,.$$

This gives us the "early" form of the weak formulation

$$\int_\Omega \sigma(\phi w) : \varepsilon(\phi v) - \int_{\partial\Omega} (\sigma(\phi w) \cdot n) \cdot \phi v = \int_\Omega f \cdot \phi v - \int_\Omega \sigma(g) : \varepsilon(\phi v) + \int_{\partial\Omega} (\sigma(g) \cdot n) \cdot \phi v \,. \tag{2.20}$$

[12] Note that we won't include the subscript h as we did for the Poisson problem's ϕ-FEM formulation; because what we are giving next is just a general framework for the technique, not its FEM approximation. By doing so, our notations are simpler, and without ambiguity.

[13] We turn v into ϕv for ease of use. As v is just a test function, we could have written $v = \phi v'$ instead, and continued with v'.

Now let's consider the ghost cells $\mathcal{T}_h^\Gamma$ [14] and edges $\mathcal{F}_h^\Gamma$, and apply stabilization terms[15] as customary in ϕ-FEM :

- First, the PDE for the elasticity equation $-\nabla \cdot \sigma(u) = f$ must be verified in each ghost cell. So, we perform the dot product on both sides of the PDE with $\nabla \cdot \sigma(\phi v)$[16] and we integrate over the concerned cell (let's call it E) to obtain:

$$-\int_E (\nabla \cdot \sigma(\phi w + g)) \cdot (\nabla \cdot \sigma(\phi v)) = \int_E f \cdot (\nabla \cdot \sigma(\phi v)) \ .$$

  Thus the quantity

$$\int_E (\nabla \cdot \sigma(\phi w)) \cdot (\nabla \cdot \sigma(\phi v)) + \int_E (f + \nabla \cdot \sigma(g)) \cdot (\nabla \cdot \sigma(\phi v)) \tag{2.21}$$

  most be close to 0.

- Second, the "gradient"[17] $\sigma(\phi w) \cdot n$ cannot behave arbitrarily at the ghost cell's interface (let's call it F). So, the quantity[18]

$$\int_F [\sigma(\phi w + g) \cdot n] \cdot [\sigma(\phi v) \cdot n] \ ,$$

  decomposed as

$$\int_F [\sigma(\phi w) \cdot n] \cdot [\sigma(\phi v) \cdot n] + \int_F [\sigma(g) \cdot n] \cdot [\sigma(\phi v) \cdot n] \tag{2.22}$$

  most be close to 0.

With $V = \left[H^1(\Omega)\right]^d$ as the variational space, we can now derive our ϕ-FEM formulation using (2.20), (2.21), and (2.22) :

$$\text{Seek } w \in V \text{ such that} \quad a(w,v) = l(v), \quad \forall v \in V \tag{2.23}$$

where a and l are defined as

$$a(w,v) = \int_\Omega \sigma(\phi w) : \varepsilon(\phi v) - \int_{\partial\Omega} (\sigma(\phi w) \cdot n) \cdot (\phi v) + G(w,v)$$

and

$$l(v) = \int_\Omega f \cdot \phi v - \int_\Omega \sigma(g) : \varepsilon(\phi v) + \int_{\partial\Omega} (\sigma(g) \cdot n) \cdot (\phi v) + G_{rhs}(v)$$

with the stabilization terms defined using a penalizing constant σ_{pen}, and the cell diameter h for a coherent dimensional analysis :

$$G(w,v) = \sigma_{pen} h^2 \sum_{E \in \mathcal{T}_h^\Gamma} \int_E (\nabla \cdot \sigma(\phi w)) \cdot (\nabla \cdot \sigma(\phi v)) + \sigma_{pen} h \sum_{F \in \mathcal{F}_h^\Gamma} \int_F [\sigma(\phi w) \cdot n] \cdot [\sigma(\phi v) \cdot n]$$

[14] These are the macro mesh's cells that are cut by the level-set function ϕ. The same goes for the ghost facets (edges), except we should include all the edges of a given cut cell.

[15] These terms will later be penalized by a constant σ_{pen} and multiplied by the cell diameter h (or h^2) in order to preserve dimensions.

[16] We could have performed the dot product with another quantity. The goal is simply to obtain a bounded symmetric coercive bilinear functional on the left, and bounded linear functional on the right hand side.

[17] This quantity matches the gradient of the displacement u. Generally in ϕ-FEM , the jump of the quantity of interest's gradient will be penalized at this point.

[18] The square bracket $[\cdot]$ is meant to indicate a jump over the cell's interface.

and

$$G_{rhs}(v) = -\sigma_{pen} h^2 \sum_{E \in \mathcal{T}_h^\Gamma} \int_E (f + \nabla \cdot \sigma(g)) \cdot (\nabla \cdot \sigma(\phi v)) - \sigma_{pen} h \sum_{F \in \mathcal{F}_h^\Gamma} \int_F [\sigma(g) \cdot n] \cdot [\sigma(\phi v) \cdot n] \ .$$

Using this framework, we can easily perform the FEM formulation and extract $u_h = \phi_h w_h + g$[19]. An error estimation wasn't performed for the elasticity equation in (Duprez and Lozinski, 2020). However, when $f \in H^k(\Omega_h \cup \Omega)$ (and given the same assumptions), we can expect the same error rates as with the Poisson problem :

$$|u - u_h|_{1,\Omega \cap \Omega_h} \leq C h^k \|f\|_{k,\Omega \cup \Omega_h} \ , \tag{2.24}$$

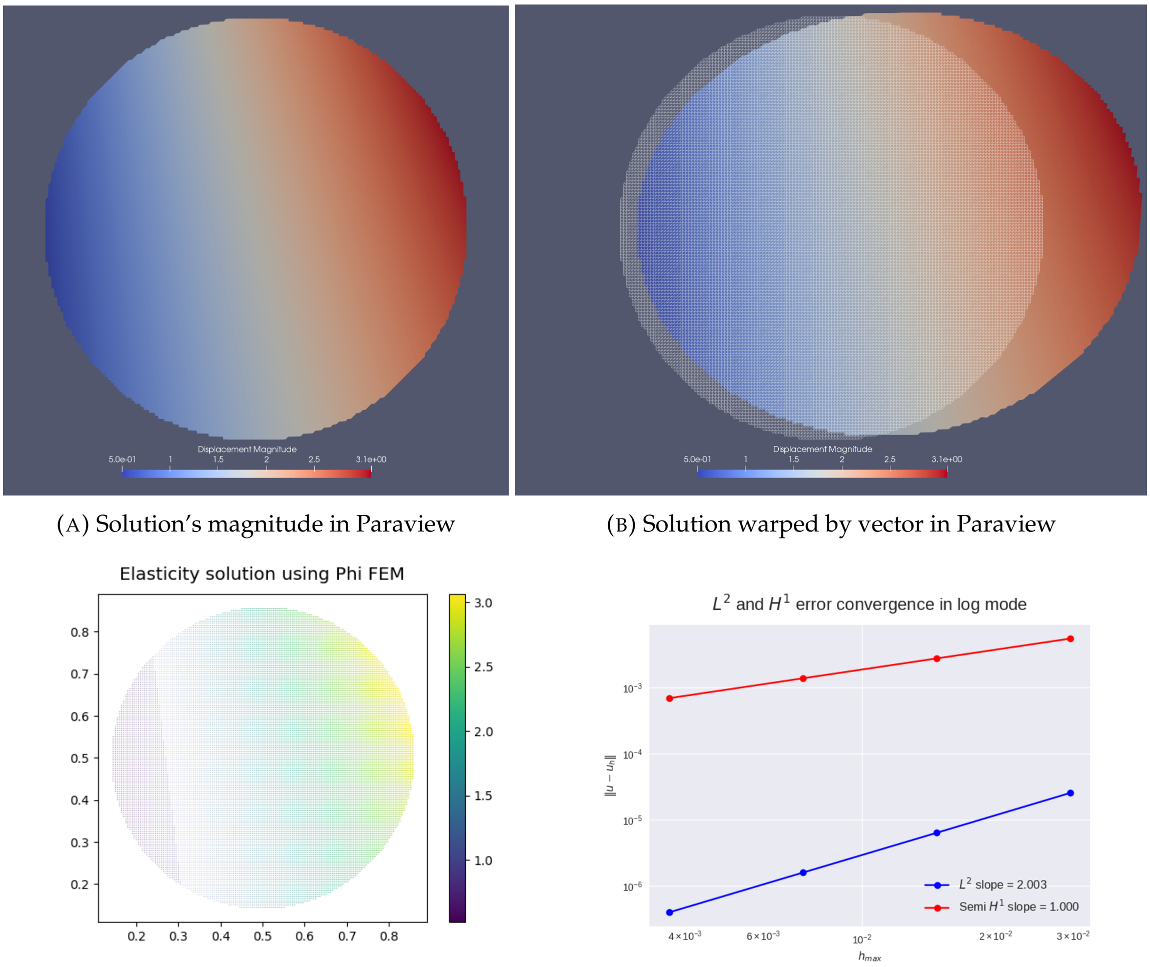
$$\|u - u_h\|_{0,\Omega} \leq C h^{k+1/2} \|f\|_{k,\Omega_h} \ . \tag{2.25}$$

Let's repeat the same simulation we performed in classic FEM. The test case [20], defined in classic FEM as (2.19), is redefined below. The result is presented at fig. 2.9. Once again, we get a much faster convergence when using ϕ-FEM .

$$\begin{cases} \mathcal{O} = [0,1] \times [0,1] \\ \phi(x,y) = -\frac{1}{8} + \left(x - \frac{1}{2}\right)^2 + \left(y - \frac{1}{2}\right)^2 \\ u(x,y) = \begin{pmatrix} 2x + \sin(x)\exp(y) \\ \frac{x}{2} + \cos(x) - 1 \end{pmatrix} \\ f(x,y) = -\nabla \cdot \sigma(u(x,y)) \\ g(x,y) = u(x,y) + \phi(x,y) \begin{pmatrix} \sin(x) \\ \exp(y) \end{pmatrix} \\ \sigma_{pen} = 20 \end{cases} \tag{2.26}$$

[19] The reader is referred to the Poisson problem's ϕ-FEM formulation on details on the FEM formulation.

[20] $\mathcal{O}$ is the macro domain, in which our mesh will be immersed. Note that g is not defined to be exactly equal to u on the boundary; this introduces a perturbation that will incite our stabilization terms to act by penalizing the result.



(A) Solution's magnitude in Paraview

(B) Solution warped by vector in Paraview

(C) Solution's components (vector) in Matplotlib

(D) Convergence study

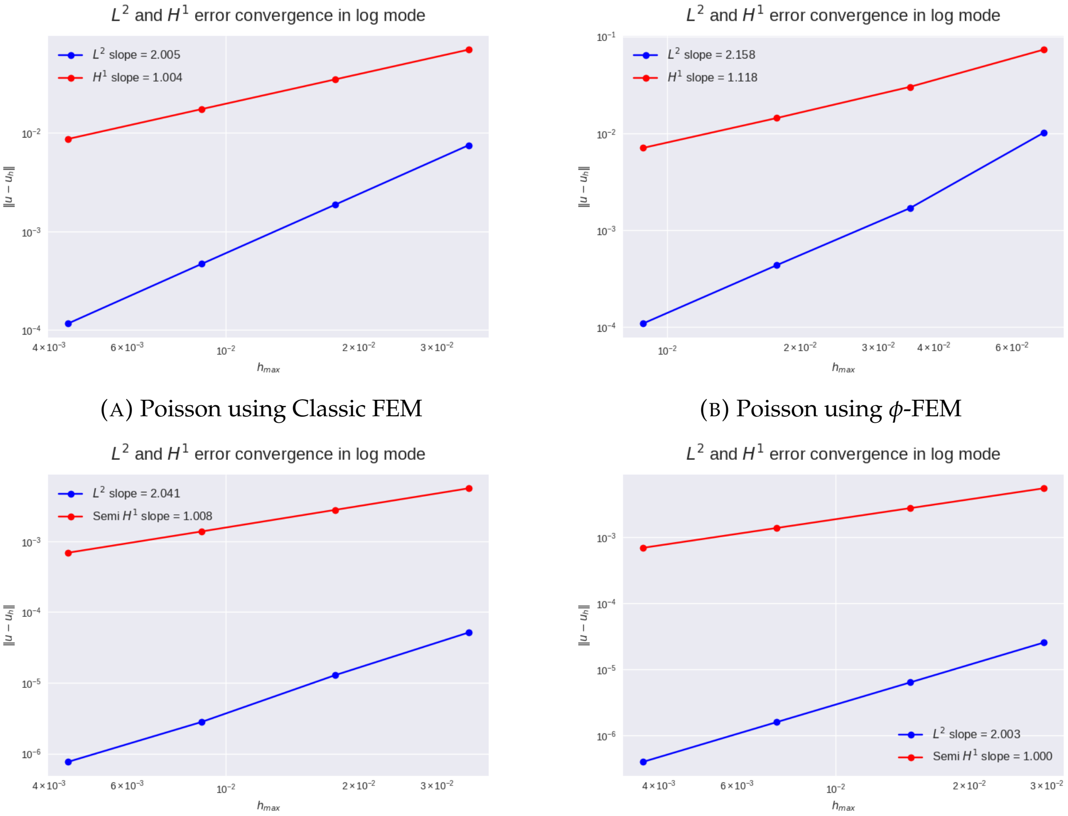
FIGURE 2.9: Results obtained when applying the ϕ-FEM technique to the elasticity equation (2.15). The convergence study in done with relative errors. The H^1 slope indicated in (D) is the H^1 semi-norm $|u - u_h|_{1,\Omega\cap\Omega_h}$.

2.4 Results summary

The table 2.1 and the figure 2.10 below summarize the results we obtained. All the errors that are presented are relative errors. We used the H^1 norm for the Poisson problem, and the semi-H^1 norm for the elasticity equation. We can see that for each of the problems we solved, the ϕ-FEM approach was better than the standard FEM approach.

Problem	**Technique**	L^2 **slope**	H^1 **slope**
Poisson	Classic FEM ϕ-FEM	2.005 2.158	1.004 1.118
Elasticity	Classic FEM ϕ-FEM	2.041 2.003	1.008 1.000

TABLE 2.1: Slopes summarizing the convergence results (Duprez, 2020a).



(A) Poisson using Classic FEM

(B) Poisson using ϕ-FEM

(C) Elasticity using Classic FEM

(D) Elasticity using ϕ-FEM

FIGURE 2.10: Comparison of the convergence results we obtained in the previous sections.

Chapter 3

Milestones

Milestones	Steps	Tools involved	Deadline	Estimated number of hours	Effective number of hours
Understand the ϕ-FEM technique	**1. Read the documents related to ϕ-FEM** • Read the introductory paper • Read the Neumann boundary case		03/11/2020	20	10
The Poisson equation	**1. Install FEniCS using Docker** • Install the most recent version • Test the installation with the demo case provided **2. Solve the Poisson equation using the classic FEM technique** • Use a simple domain (a unit disk) • Validate this step by differentiating a known solution and verifying the results **3. Perform the convergence study in norms L^2 and H^1** • According to the theory, the slopes must be respectively close to 2 and 1 **4. Solve the Poisson equation using the ϕ-FEM technique, without stabilising terms. Compare the results with the classic FEM technique** • Validate this step by comparison with the test cases in the paper **5. Repeat the preceding test, while applying stabilizing terms** • Validate this step by comparison with the paper • Repeat the exact test cases in the paper if necessary	Docker FEniCS	10/11/2020	25	50
The elasticity equation	**1. Reformulate the elasticity equation using ϕ-FEM** • Take inspiration from the Poisson formulation **2. Solve the equation using FEniCS** • The method can be validated using academic cases as done in the papers • The method can also be validated on classical solid mechanics cases such as beams	Docker FEniCS	19/01/2021	25	30
Simulations on organ geometries	**3. Find the geometries** **4. Integrate the results into SOFA**	Docker FEniCS SOFA	19/01/2021	25	0

The page above summarizes the most important steps we completed, along with their deadlines, the tools needed, the estimated/effective number of hours spent on each objective. We will now be presenting a more detailed report on the conduct of the project, week after week. Each week amounts to at least 8 hours of work ; the early ones were the most consuming.

- **Week #1** - (14/10/2020 - 21/10/2020) :
  - Install the most recent version of FEniCS through Docker.
  - Then test the installation by solving the Poisson problem using the classic FEM formulation.
  - Afterwards, perform a convergence study.
  - Begin reading the paper[1].
- **Weeks #2 and #3** - (21/10/2020 - 03/11/2020) : The required work was to solve the Poisson problem, now using the ϕ-FEM formulation:
  - First without stabilizing terms ;
  - Then with stabilizing terms.
- **Week #4** - (3/11/2020 - 10/11/2020) : Following the unconvincing results from the previous week,
  - Verify that the correct cells are being selected when performing ϕ-FEM .
  - Perform the first test case from the paper, using Sympy.
  - Finish reading the paper.
- **Week #5** - (10/11/2020 - 17/11/2020) :
  - Put more details about CutFEM and XFEM in the report version 0.
  - Correct the code for the Poisson PhiFEM.
- **Week #6** - (17/11/2020 - 24/11/2020) :
  - Implement the elasticity equation in classic FEM.
  - Start the Elasticity equation's implementation in ϕ-FEM by writing the variational form.
  - Finish wrting the report version 0.
- **Week #7** - (24/11/2020 - 01/12/2020) :
  - Perform a test case with the boundary clamped.
  - Add convergence details in the report version 0.
- **Week #8** - (01/12/2020 - 08/12/2020) :
  - Finish the elasticity equation's ϕ-FEM implementation.
- **Week #9** - (08/12/2020 - 15/12/2020) :
  - Correct the bug in the convergence study.
  - Add boundary terms to the formulation.

[1] The document known as "the paper" is the main publication describing ϕ-FEM (Duprez and Lozinski, 2020).

- **Weeks #10 to #13** - (15/12/2020 - 12/01/2021) : No additional work was done in this time period.
- **Week #14** - (12/01/2021 - 19/01/2021) :
  - Write the report version 1.
  - Prepare the slides for the presentation.
- **Week #15** - (19/01/2021 - 26/01/2021) :
  - Add corrections for the report version 2.
  - Polish the slides.

Chapter 4

Future works

This project has taught me a lot on the Finite Element Method, and the many ways to make it better. Having tested the ϕ-FEM technique on the Poisson and the elasticity equations, hence giving us a glimpse into its effectiveness on solid dynamics, it remains essential to complete the next steps in order to apply the technique to the real-time yet precise simulation of human organs. These steps could be :

1. Test the technique on complex geometries
2. Benchmark the technique for speed efficiency
3. Deploy the numerical implementation into the SOFA software

We should note that these steps would likely not be implemented in FEniCS due to limitations it faces on such problems. Also, a priority should be on implementing the ϕ-FEM technique using the Neumann conditions[1]. This is especially important for the elasticity equations, where the Neumann condition ultimately comes back to applying a certain force on the boundary of the domain. This results in a more realistic simulation of an organ's displacement.

[1] The reader is referred to Duprez et al., 2020 for the theoretical basis for the Neumann implementation.

Bibliography

Atallah, Nabil M et al. (2020). "Analysis of the Shifted Boundary Method for the Poisson Problem in General Domains". In: ***arXiv preprint arXiv:2006.00872***.

Bandringa, Henry (2010). "Immersed boundary methods". PhD thesis. Faculty of Science and Engineering.

Burman, Erik (2010). "Ghost penalty". In: ***Comptes Rendus Mathematique*** 348.21-22, pp. 1217–1220.

Burman, Erik and Peter Hansbo (2010). "Fictitious domain finite element methods using cut elements: I. A stabilized Lagrange multiplier method". In: ***Computer Methods in Applied Mechanics and Engineering*** 199.41-44, pp. 2680–2686.

Burman, Erik et al. (2015). "CutFEM: discretizing geometry and partial differential equations". In: ***International Journal for Numerical Methods in Engineering*** 104.7, pp. 472–501.

De Cicco, Davide and Farid Taheri (2018). "Delamination buckling and crack propagation simulations in fiber-metal laminates using xFEM and cohesive elements". In: ***Applied Sciences*** 8.12, p. 2440.

Duprez, M. (2020a). "Personal notes from M. Duprez on January 28, 2020". In.

– (Oct. 2020b). "Proposition de mini-projet". In: ***MIMESIS***. URL: `http://mduprez.perso.math.cnrs.fr/Recherches/projet_csmi_m2_2021_phifem.pdf`.

Duprez, M. et al. (Mar. 2020). "phi-FEM, a finite element method on domains defined by level-sets: the Neumann boundary case". In: ***arXiv***. URL: `https://arxiv.org/abs/2003.11733`.

Duprez, Michel and Alexei Lozinski (2020). "ϕ-FEM: a finite element method on domains defined by level-sets". In: ***SIAM Journal on Numerical Analysis*** 58.2, pp. 1008–1028.

Ern, Alexandre and Jean-Luc Guermond (2013). ***Theory and practice of finite elements***. Vol. 159. Springer Science & Business Media.

Evans, Lawrence C. (1998). ***Partial Differential Equations***.

Fries, Thomas-Peter (n.d.). "The Extended Finite Element Method". In: ***XFEM***. URL: `http://www.xfem.rwth-aachen.de/Background/Introduction/XFEM_Introduction.php`.

Main, Alex and Guglielmo Scovazzi (2018). "The shifted boundary method for embedded domain computations. Part I: Poisson and Stokes problems". In: ***Journal of Computational Physics*** 372, pp. 972–995.

MIMESIS (Nov. 2020). "ABOUT US". In: ***MIMESIS***. URL: `https://mimesis.inria.fr/`.

Moës, Nicolas et al. (1999). "A finite element method for crack growth without remeshing". In: ***International journal for numerical methods in engineering*** 46.1, pp. 131–150.