Research Internship (PRe)

Major: Applied Mathematics
School year: 2024-2025

Contribution to the development of Psydac, a finite element library.

Implementation of solid mechanics equations using Psydac and simulation on a supercomputer.

Non-confidential report and Can be published on the Internet

Author : Arasu Candassamy

Year Group: 2026

ENSTA Supervisor:
Sonia Fliss
sonia.fliss@ensta.fr

Host Organization Supervisor:
Martin Campos-Pinto
martin.campos-pinto@ipp.mpg.de

Internship from 26/05/2025 to 22/08/2025
Name of the host organization: Max-Planck-Institute Für Plasmaphysik
Address: Boltzmannstrasse 2, 85748 Garching bei München

Abstract

This report presents the work that I conducted during a research internship at the Max-Planck Institute for Plasma Physics in the Numerical Methods for Plasma Physics department. This report is focusing on the implementation and validation of solid mechanics equations using `Psydac`, a finite element library in Python.

The primary objective of my internship was to explore and extend Psydac's capabilities to solve partial differential equations from solid mechanics, which had not been previously tested in this library. The work focused on two main areas: linear elasticity and stationary thermoelasticity problems and their execution on a supercomputer. For linear elasticity, two formulations, pure displacement and mixed displacement-pressure formulations, were implemented and analyzed.

The thermoelasticity study involved solving coupled thermo-mechanical problems on composite materials with spatially varying properties. This work required the development of enhanced discretization capabilities, including the implementation of custom grid discretization features to handle material property discontinuities. Additionally, a multipatch framework was explored to address complex geometries with holes, utilizing Nitsche's method for interface coupling.

All simulations were performed on the supercomputer Raven, which is maintained by the Max Planck Computing and Data Facility. These results have been used to verify the implementation and the effectiveness of Psydac to solve solid mechanics problems.

Keywords: Finite element method (FEM), Isogeometric analysis (IGA), B-splines, Solid mechanics, Linear elasticity, Thermoelasticity, Multipatch, Psydac, Python, High performance computing (HPC), Supercomputers.

Résumé

Ce rapport présente mes travaux menés lors d'un stage de recherche à l'Institut Max-Planck de Physique des Plasmas dans le département des Méthodes Numériques pour la Physique des Plasmas. Ce rapport se concentre sur l'implémentation et la validation d'équations de mécanique des solides en utilisant la bibliothèque d'éléments finis `Psydac` sur Python.

L'objectif principal de mon stage était d'explorer les capacités de Psydac pour résoudre des équations aux dérivées partielles issues de la mécanique des solides, qui n'avaient pas été testées auparavant dans cette bibliothèque. Le travail s'est concentré sur deux domaines principaux : l'élasticité linéaire et les problèmes de thermoélasticité, ainsi que leur exécution sur un supercalculateur. Pour l'élasticité linéaire, deux formulations différentes ont été implémentées et analysées : la formulation en déplacement pur et la formulation mixte déplacement-pression.

L'étude de thermoélasticité a impliqué la résolution de problèmes thermo-mécaniques couplés sur des matériaux composites avec des propriétés variant spatialement. Ce travail a nécessité le développement de fonctionnalités de discrétisation sur une grille personnalisée pour gérer les discontinuités de propriétés matérielles. De plus, une implémentation utilisant un domaine *multipatch* a été testée pour traiter une géométrie complexe. Cette implémentation a fait appel à la méthode de Nitsche pour assurer la continuité aux interfaces des patches.

Toutes les simulations ont été testées sur le supercalculateur Raven, qui est maintenu par l'équipe du Max Planck Computing and Data Facility. Ces résultats ont été utilisés pour valider l'implémentation et l'efficacité de Psydac pour résoudre des problèmes issus de la mécanique des solides.

Mots-clés: Méthode des éléments finis, Analyse isogéométrique, B-splines, Mécanique des solides, Élasticité linéaire, Thermoélasticité, Multipatch, Psydac, Python, Calcul haute performance, Supercalculateurs.

Acknowledgement

I would like to express my sincere gratitude to my supervisor, Pr. Dr. Martin Campos Pinto, for his invaluable guidance, expertise, and constant support throughout this internship.

I am particularly thankful to Dr. Yaman Güçlü, my co-supervisor and head developer of Psydac, for his technical expertise and support. I am also grateful to the entire PSYDAC team for their collaboration and assistance during this project (Ms. Elena Moral Sanchez, Mr. Julian Owezarek, and Mr. Frederik Schnack).

I want to thank Ms. Svenja Eichler for her help in facilitating my integration into the institute and for her support with administrative matters.

I thank the entire Numerical Method for Plasma Physics (NMPP) team for their warm welcome and support throughout my internship.

Finally, I thank my flatmates, Mr. Thomas Lefert and Mr. Louis Marchal, for their companionship and for making my time in Germany both productive and memorable.

Contents

List of Figures

Chapter 1

Introduction

The Max Planck Institute of Plasma Physics (IPP) is one of the leading research institutions dedicated to fusion energy research, with two main sites in Garching near Munich and Greifswald. The institute focuses on developing the scientific and technological foundations for fusion power plants, conducting experiments on major facilities like ASDEX Upgrade and Wendelstein 7-X stellarator.

At IPP, the Numerical Methods for Plasma Physics (NMPP) department is developing advanced computational tools for plasma physics simulations. The department is responsible for creating and maintaining several high-performance computing libraries that support the institute's research activities. Among these tools, `GEMPIC` (Geometric Electromagnetic Particle-In-Cell) provides sophisticated particle simulation capabilities for kinetic plasma modelling, while `Struphy` offers specialized structure-preserving algorithms for long-term plasma evolution studies. These libraries collectively form a comprehensive computational ecosystem that enables researchers to tackle the challenging multi-physics problems inherent in fusion plasma research.

`Psydac` is a finite element library developed by NMPP. It implements isogeometric analysis (IGA) using B-splines and NURBS to solve partial differential equations from plasma physics. `Psydac` leverages the power of spline-based discretizations to achieve high-order accuracy while maintaining geometric flexibility. The library incorporates advanced features such as multipatch domains, parallel computing capabilities, and integration with high-performance computing environments. Its modular design, built upon `SymPDE` for symbolic problem formulation and `Pyccel` for code acceleration, makes it particularly well-suited for solving complex partial differential equations across various physics domains.

The primary objective of this internship was to explore `Psydac`'s capabilities to solve equations from solid mechanics. This work involved implementing and validating solutions for linear elasticity and stationary thermoelasticity problems, areas that had not been previously tested within the library's framework. The research encompassed two main directions: developing pure displacement and mixed displacement-pressure formulations for linear elasticity problems, and addressing thermoelasticity challenges involving composite materials with spatially varying properties.

This report is structured into two main chapters that document the progression of this work. Chapter 2 focuses on linear elasticity, beginning with the mathematical foundations and its well-posedness, followed by extensive numerical simulations using `Psydac` on the Raven supercomputer. Special attention is given to the comparison between two formulations, particularly in the context of nearly incompressible materials where volumetric locking becomes a significant concern. Chapter 3 addresses the stationary thermoelasticity problem. This problem is inspired by a real application presented at the IPP Days (two days of presentations and discussions on the latest advancements in

plasma physics and fusion research). It starts with a numerical solution on a simplified geometry, before exploring a more complex configuration with a multipatch domain. This chapter will briefly present a feature that I have implemented to handle discretisation on a custom grid.

Through this comprehensive study, the internship shows `Psydac`'s potential for broader engineering applications while contributing new capabilities to the library's functionality. The results validate the effectiveness of isogeometric analysis for solid mechanics problems and highlight both the advantages and challenges of extending specialized plasma physics tools to other domains.

Chapter 2

Linear Elasticity

2.1 Physical Model and Strong Formulation

The following presentation of the context of work and physical background comes from [Gou13] and [EG04]. Let's denote by $\Omega \subset \mathbb{R}^3$ a deformable medium characterized by λ and μ Lamé's coefficients ($\lambda, \mu > 0$). Ω is a bounded open domain with lipschitzian boundary. λ is called Lamé's first coefficient, μ is the shear modulus. These coefficients are related to the Young's modulus and Poisson's ratio of the medium. The deformable medium is under an external load $\vec{f} : \Omega \to \mathbb{R}^3$. Then, we are looking for $\vec{u} : \Omega \to \mathbb{R}^3$ the displacement field, when the equilibrium is reached.

The material verifies the following equilibrium condition :

$$\boldsymbol{\nabla} \cdot \boldsymbol{\sigma}(\vec{u}) + \vec{f} = \vec{0} \text{ in } \Omega, \tag{2.1}$$

where $\boldsymbol{\sigma}(\vec{u})$ is the Cauchy Stress Tensor, defined by $\boldsymbol{\sigma}(\vec{u}) = C : \boldsymbol{\varepsilon}(\vec{u})$. C is the fourth-order stiffness tensor and $\boldsymbol{\varepsilon}(\vec{u}) := \frac{1}{2}(\boldsymbol{\nabla}\vec{u} + (\boldsymbol{\nabla}\vec{u})^T)$ is the strain tensor. In the context of linear elasticity, the relation between $\boldsymbol{\varepsilon}(\vec{u})$ and $\boldsymbol{\sigma}(\vec{u})$ is simpler :

$$\boldsymbol{\sigma}(\vec{u}) = \lambda \operatorname{Tr}(\boldsymbol{\varepsilon}) I_3 + 2\mu\boldsymbol{\varepsilon} = \lambda(\boldsymbol{\nabla} \cdot \vec{u}) I_3 + \mu(\boldsymbol{\nabla}\vec{u} + (\boldsymbol{\nabla}\vec{u})^T) \text{ in } \Omega. \tag{2.2}$$

For boundary conditions, there are two types of boundary conditions : Dirichlet conditions on the displacement field $\vec{u}$ and Traction conditions on the Cauchy Stress Tensor $\boldsymbol{\sigma}(\vec{u})$.

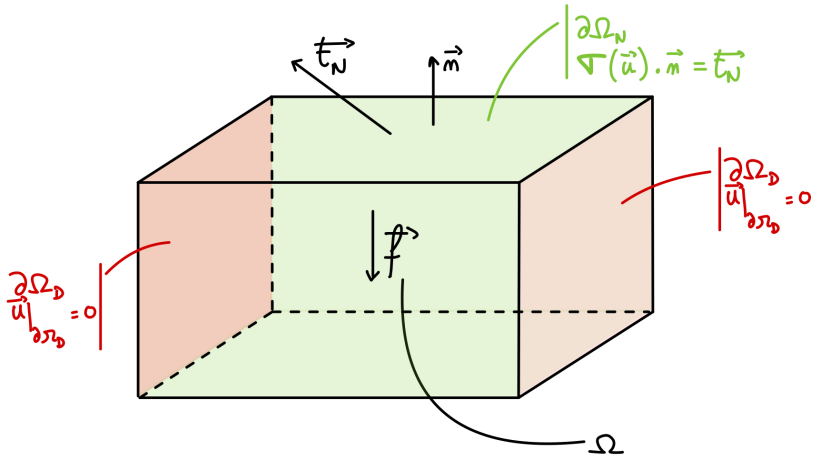


Figure 2.1: Representation of a domain Ω

Let's decompose $\partial\Omega$ as $\partial\Omega_D$ and $\partial\Omega_N$ such that $\partial\Omega_D \cup \partial\Omega_N = \partial\Omega$, $\partial\Omega_D \cap \partial\Omega_N = \emptyset$. Let's denote by $\vec{t}_N : \partial\Omega_N \to \mathbb{R}^3$ the normal traction force and $\vec{n}$ the normal vector of $\partial\Omega$ outside-oriented. We can consider the boundary displacement on $\partial\Omega_D$ as equal to $\vec{0}$.

In case of non-homogeneous boundary conditions $\vec{g} : \partial\Omega_D \to \mathbb{R}^3$, we can solve the problem for $\vec{u} - \vec{g}$ by modifying the source term. Then, the solution of linear elasticity problem should verify :
$\begin{cases} \vec{u} = \vec{0} & \partial\Omega_D \\ \boldsymbol{\sigma}(\vec{u}) \cdot \vec{n} = \vec{t}_N & \partial\Omega_N \end{cases}$

Physically, $\vec{f}$ is a "body force" on Ω and $\vec{t}_N$ is a "surface traction". The strong formulation of this problem is :
For a given $\vec{f} \in \boldsymbol{L}^2(\Omega)$, $\vec{t}_N \in \boldsymbol{L}^2(\partial\Omega_N)$, find $\vec{u} \in \boldsymbol{H}^1(\Omega)$ such that :

$$\begin{cases} -\boldsymbol{\nabla} \cdot \boldsymbol{\sigma}(\vec{u}) = \vec{f} & \text{in } \Omega \quad \text{(Equilibrium)} \\ \boldsymbol{\sigma}(\vec{u}) = \lambda(\boldsymbol{\nabla} \cdot \vec{u}) I_3 + \mu(\boldsymbol{\nabla}\vec{u} + (\boldsymbol{\nabla}\vec{u})^T) & \text{in } \Omega \quad \text{(Hooke's law)} \\ \vec{u} = \vec{0} & \text{in } \partial\Omega_D \\ \boldsymbol{\sigma}(\vec{u}) \cdot \vec{n} = \vec{t}_N & \text{in } \partial\Omega_N \end{cases} \tag{2.3}$$

where $\boldsymbol{L}^2(\Omega) = (L^2(\Omega))^3$, $\boldsymbol{L}^2(\partial\Omega_N) = (L^2(\partial\Omega_N))^3$, $\boldsymbol{H}^1(\Omega) = (H^1(\Omega))^3$,

2.2 Pure displacement weak-formulation

To derive the weak formulation from (2.3), let's introduce the following functional space : $\boldsymbol{H}^1_{\vec{0},D}(\Omega) = \left\{ \vec{v} \in \boldsymbol{H}^1(\Omega) \middle| \vec{v}_{|\partial\Omega_D} = \vec{0} \right\}$, which is a sub-Hilbert space of $\boldsymbol{H}^1(\Omega)$ equipped with the same inner product (as a closed space of a Hilbert space). By taking the scalar product of the equilibrium equation from (2.3) with $\vec{v} \in \left(H^1_{\vec{0},D}(\Omega)\right)^3$ and applying a Green's formula, one obtains : $\int_\Omega \boldsymbol{\sigma}(\vec{u}) : (\boldsymbol{\nabla}\vec{v})\mathrm{d}\Omega - \int_{\partial\Omega} (\boldsymbol{\sigma}(\vec{u}).\vec{n}) .\vec{v}\mathrm{d}\Gamma = \int_\Omega \vec{f}.\vec{v}\mathrm{d}\Omega$

One can remark that with (2.2), the tensor $\boldsymbol{\sigma}(\vec{u})$ is symmetric. It implies that : $\boldsymbol{\sigma}(\vec{u}) : \boldsymbol{\nabla}\vec{v} = \boldsymbol{\sigma}(\vec{u}) : \boldsymbol{\varepsilon}(\vec{v})$ Moreover, $\vec{v} \in \left(H^1_{\vec{0},D}(\Omega)\right)^3$, then $\vec{v}_{|\partial\Omega_D} = \vec{0}$. Thus, the weak formulation can be simplified to :

$$\begin{cases} \text{Find } \vec{u} \in \boldsymbol{H}^1_{\vec{0},D}(\Omega) \text{ such that :} \\ a(\vec{u},\vec{v}) = l(\vec{v}) \quad \forall \vec{v} \in \boldsymbol{H}^1_{\vec{0},D}(\Omega) \end{cases} \tag{2.4}$$

With :

$$a : \begin{cases} \boldsymbol{H}^1_{\vec{0},D}(\Omega) \times \boldsymbol{H}^1_{\vec{0},D}(\Omega) \to \mathbb{R} \\ (\vec{u},\vec{v}) \longmapsto \int_\Omega \boldsymbol{\sigma}(\vec{u}) : \boldsymbol{\varepsilon}(\vec{v})\mathrm{d}\Omega = \int_\Omega \lambda(\boldsymbol{\nabla} \cdot \vec{u})(\boldsymbol{\nabla} \cdot \vec{v})\mathrm{d}\Omega + \int_\Omega 2\mu\boldsymbol{\varepsilon}(\vec{u}) : \boldsymbol{\varepsilon}(\vec{v})\mathrm{d}\Omega \end{cases}$$
$$l : \begin{cases} \boldsymbol{H}^1_{\vec{0},D}(\Omega) \to \mathbb{R} \\ \vec{v} \longmapsto \int_\Omega \vec{f}.\vec{v}\mathrm{d}\Omega + \int_{\partial\Omega_N} \vec{t}_N.\vec{v}\mathrm{d}\Gamma \end{cases}$$

In continuum mechanics, (2.4) is known as the Principle of Virtual Work. Let's look at the well-posedness of (2.4). Before starting, let's recall some notations and lemmas.

$\|.\|_{\boldsymbol{H}^1(\Omega)}$ is defined as : $\forall \vec{v} \in \boldsymbol{H}^1(\Omega), \|\vec{v}\|_{\boldsymbol{H}^1(\Omega)} = \left(\sum_{i=1}^3 \|v\|^2_{H^1(\Omega)}\right)^{1/2}$.

Lemma 2.2.1. *1.* $\forall \vec{u} \in \boldsymbol{H}^1(\Omega)$, $\int_\Omega (\boldsymbol{\nabla} \cdot \vec{u})^2\mathrm{d}\Omega \leq 3\|\vec{u}\|^2_{\boldsymbol{H}^1(\Omega)}$

2. $\forall \vec{u} \in \boldsymbol{H}^1(\Omega)$,$\boldsymbol{\varepsilon}(\vec{u}) : \boldsymbol{\varepsilon}(\vec{u}) \leq \boldsymbol{\nabla}\vec{u} : \boldsymbol{\nabla}\vec{u}$ *Then :* $\int_\Omega \boldsymbol{\varepsilon}(\vec{u}) : \boldsymbol{\varepsilon}(\vec{u})\mathrm{d}\Omega \leq \int_\Omega \boldsymbol{\nabla}\vec{u} : \boldsymbol{\nabla}\vec{u}\mathrm{d}\Omega \leq \|\vec{u}\|^2_{\boldsymbol{H}^1(\Omega)}$

Proof. By direct computation, or a detailed version in [Cin18] □

Lemma 2.2.2. *(First Korn's inequality)*
Let $\Omega \subset \mathbb{R}^3$ a domain and denotes by $\|\boldsymbol{\varepsilon}(\vec{u})\|_{\boldsymbol{L}^2(\Omega)} := \left(\int_\Omega \boldsymbol{\varepsilon}(\vec{u}) : \boldsymbol{\varepsilon}(\vec{u})\right)^{1/2}$. Then, there exists $C > 0$ such that: $\forall \vec{v} \in \boldsymbol{H}^1_{\vec{0}}(\Omega), \quad C\|\vec{v}\|_{\boldsymbol{H}^1(\Omega)} \leq \|\boldsymbol{\varepsilon}(\vec{v})\|_{\boldsymbol{L}^2(\Omega)}$

Proof. A proof can be found in [BS08] □

Lemma 2.2.3. *(Second Korn's inequality)*
Let $\Omega \subset \mathbb{R}^3$ a domain. Then, there exists $C > 0$ such that: $\forall \vec{v} \in \boldsymbol{H}^1(\Omega), \quad C\|\vec{v}\|_{\boldsymbol{H}^1(\Omega)} \leq \|\boldsymbol{\varepsilon}(\vec{v})\|_{\boldsymbol{L}^2(\Omega)} + \|\vec{v}\|_{\boldsymbol{L}^2(\Omega)}$ By Rellich theorem, $\boldsymbol{H}^1(\Omega)$ is compactly embedded in $\boldsymbol{L}^2(\Omega)$ as Ω is a open bounded domain . Then, one can show that there exists $C > 0$ such that: $\forall \vec{v} \in \boldsymbol{H}^1(\Omega), \quad C\|\vec{v}\|_{\boldsymbol{H}^1(\Omega)} \leq \|\boldsymbol{\varepsilon}(\vec{v})\|_{\boldsymbol{L}^2(\Omega)}$

Proof. A proof can be found in [BS08] □

Now, we can prove the well-posedness of (2.4) when $|\Omega_D| > 0$.

- $\left(\boldsymbol{H}^1_{\vec{0},D}(\Omega), \|.\|_{\boldsymbol{H}^1(\Omega)}\right)$ is a Hilbert space.
- $a(.,.)$ is a bilinear form on $\boldsymbol{H}^1_{\vec{0},D}(\Omega) \times \boldsymbol{H}^1_{\vec{0},D}(\Omega)$ and $l(.)$ a linear form on $\boldsymbol{H}^1_{\vec{0},D}(\Omega)$.
- $l(.)$ is continuous :

$$\forall \vec{v} \in \boldsymbol{H}^1_{\vec{0},D}(\Omega), |l(\vec{v})| \leq \left|\left(\vec{f},\vec{v}\right)_{\boldsymbol{L}^2(\Omega)} + \left(\vec{t}_N,\vec{v}\right)_{\boldsymbol{L}^2(\partial\Omega_N)}\right|$$
$$|l(\vec{v})| \leq \left(\left\|\vec{f}\right\|_{\boldsymbol{L}^2(\Omega)} + \|\gamma_0\|\left\|\vec{t}_N\right\|_{\boldsymbol{L}^2(\partial\Omega_N)}\right)\|\vec{v}\|_{\boldsymbol{H}^1(\Omega)}$$

  Where γ_0 is the trace operator.

- $a(.,.)$ is continuous:

  $\forall \vec{u},\vec{v} \in \boldsymbol{H}^1_{\vec{0},D}(\Omega), |a(\vec{u},\vec{v})| \leq |a(\vec{u},\vec{u})|^{1/2}\,|a(\vec{v},\vec{v})|^{1/2}$ By the Cauchy-Schwarz Inequality as $a(.,.)$ i

  a symmetric and positive bilinear forn

  $$|a(\vec{u},\vec{v})| \leq (3\lambda + 2\mu)\|\vec{u}\|_{\boldsymbol{H}^1(\Omega)}\|\vec{v}\|_{\boldsymbol{H}^1(\Omega)}.$$

- $a(.,.)$ is coercive:
  - If $\Omega_D = \Omega$, then $\boldsymbol{H}^1_{\vec{0},D}(\Omega) = \boldsymbol{H}^1_{\vec{0}}(\Omega)$ and we can apply **Lemma 2.2.2** : there exists $C > 0$ such that:

$$\forall \vec{u} \in \boldsymbol{H}^1_{\vec{0},D}(\Omega), \quad a(\vec{u},\vec{u}) = \int_\Omega \lambda(\boldsymbol{\nabla}\cdot\vec{u})^2 \mathrm{d}\Omega + \int_\Omega 2\mu\boldsymbol{\varepsilon}(\vec{u}) : \boldsymbol{\varepsilon}(\vec{u})\mathrm{d}\Omega$$
$$\geq 2\mu\|\boldsymbol{\varepsilon}(\vec{u})\|_{\boldsymbol{L}^2(\Omega)}$$
$$a(\vec{u},\vec{u}) \geq 2\mu C\|\vec{u}\|^2_{\boldsymbol{H}^1(\Omega)}$$

  - If $\Omega_D \subsetneq \Omega$, we can now apply **Lemma 2.2.3** : there exists $C > 0$ such that: $\forall \vec{u} \in \boldsymbol{H}^1_{\vec{0},D}(\Omega), \quad a(\vec{u},\vec{u}) \geq 2\mu C\|\vec{u}\|^2_{\boldsymbol{H}^1(\Omega)}$

Then, by Lax-Milgram theorem, the problem (2.4) is well-posed when $|\Omega_D| > 0$.
When, $|\Omega_D| = 0$, (2.4) turns into the **Pure Traction Problem** :

$$\begin{cases} \text{Find } \vec{u} \in \boldsymbol{H}^1(\Omega) \text{ such that :} \\ a(\vec{u},\vec{v}) = \int_\Omega \vec{f}.\vec{v}\mathrm{d}\Omega + \int_{\partial\Omega} \vec{t}_N.\vec{v}\mathrm{d}\Gamma =: l(\vec{v}) \quad \forall \vec{v} \in \boldsymbol{H}^1(\Omega) \end{cases} \tag{2.5}$$

This problem is not truly well-posed. Regarding the Rigid Displacement Field (translations and rotations) $\boldsymbol{\mathcal{R}} = \left\{\vec{v} \in \boldsymbol{H}^1(\Omega) \,|\exists a, b \in \mathbb{R}^3, \vec{v}(x) = a + b \times x\right\}$, $\forall \vec{v} \in \boldsymbol{\mathcal{R}}$, $\boldsymbol{\nabla} \cdot v = 0$ and $\boldsymbol{\varepsilon}(\vec{v}) = 0$. Then : $\forall \vec{u} \in \boldsymbol{H}^1(\Omega), \forall \vec{v} \in \boldsymbol{\mathcal{R}}$, $a(\vec{u},\vec{v}) = 0$. So, the weak formulation (2.5) is solvable iff the following compatibility condition holds : $\forall \vec{v} \in \boldsymbol{\mathcal{R}}$, $\int_\Omega \vec{f}.\vec{v}\mathrm{d}\Omega + \int_{\partial\Omega} \vec{t}_N.\vec{v}\mathrm{d}\Gamma = 0$, and in this case, there exists an unique solution $\vec{u} \in \hat{\boldsymbol{H}}^1(\Omega) := \left\{\vec{v} \in \boldsymbol{H}^1(\Omega) \,\middle|\, \int_\Omega \vec{v}\mathrm{d}\Omega = \int_\Omega \boldsymbol{\nabla} \times \vec{v}\mathrm{d}\Omega = \vec{0}\right\}$ In case of **Pure Traction Problem**, we are looking for a solution modulo a rigid displacement field. *A more rigorous proof of the equivalence can be found in [BS08] and [Che24].*

2.3 Numerical simulation using PSYDAC on the supercomputer Raven

PSYDAC is a high-level programming IDE and very intuitive to use. It uses SymPDE and Pyccel. SymPDE implements in Python abstract objects (functional space, differential operators, linear and bilinear forms...). Pyccel is a transpiler, used to convert a Python code into a C or Fortran code, in order to take advantage of code acceleration and parallelization. Figure 2.2 summarizes the dependency between PSYDAC, SymPDE and Pyccel. A more detailed presentation of PSYDAC can be found in [GHR22].

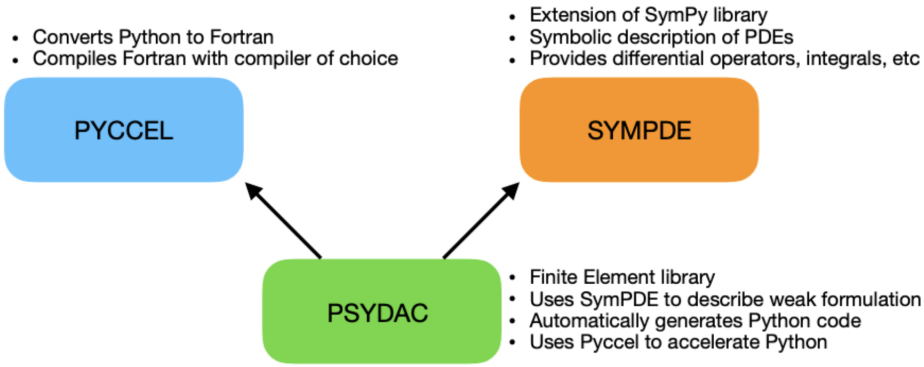


Figure 2.2: Dependency graph of PSYDAC, from [GHR22]

Raven is a supercomputer located at the Max-Planck Institute for Plasma Physics. It is part of the High Performance Computing (HPC) infrastructure and is used for large-scale simulations and data analysis in plasma physics and related fields. I had the opportunity to run my simulations on Raven, which provided me with access to powerful computational resources and parallel processing capabilities.

2.3.1 Dirichlet homogeneous boundary conditions on $\partial\Omega_D = \partial\Omega$

At first, to get familiar with PSYDAC and the HPC, I considered the case $\Omega = [0,1]^3$, $\partial\Omega_D = \partial\Omega$ and $\partial\Omega_N = \emptyset$. To verify if the numerical solution of PSYDAC of (2.4) is correct, I used the Method of

Manufactured Solutions. The concept is quite simple: we assume that a function $\vec{u}_e \in \boldsymbol{H}_{\vec{0}}^1(\Omega)$ verifies the strong problem (2.3), then we compute the source term $\vec{f}$ and we solve (2.4) with $\vec{f}$ as a source term and we look at the error between the numerical solution and the theoretical one. I considered $\vec{u}_e$ defined as: $\vec{u}_e = \begin{pmatrix} 0 \\ 0 \\ \sin(\pi x)\sin(\pi y)\sin(\pi z) \end{pmatrix} \in \boldsymbol{H}_{\vec{0}}^1(\Omega)$. Then, we can compute manually the source term $\vec{f}$: $\vec{f} = \begin{pmatrix} -\pi^2(\lambda+\mu)\cos(\pi x)\sin(\pi y)\cos(\pi z) \\ -\pi^2(\lambda+\mu)\sin(\pi x)\cos(\pi y)\cos(\pi z) \\ \pi^2(\lambda+4\mu)\sin(\pi x)\sin(\pi y)\sin(\pi z) \end{pmatrix} \in \boldsymbol{L}^2(\Omega)$.

To make the numerical simulation with `PSYDAC`, we have to choose two important parameters: d the maximum degree of B-spline functions that will approximate the solution and n_{cells} is the number of cells in each dimension of Ω for the finite dimensional space. $h \propto \dfrac{1}{n_{\text{cells}}}$, a usual discretization parameter for finite dimensional spaces and d will control the convergence speed of the numerical solution. The code in Section 6.1 presents the implementation details of the linear elasticity problem with Dirichlet boundary conditions. By solving the problem with Dirichlet homogeneous boundary conditions on the whole boundary, one can observe the following plots of the simulated $\vec{u}_{e,3}^h$ for different planes z. It is also possible to plot the error $\vec{u}_{e,3} - \vec{u}_{e,3}^h$.

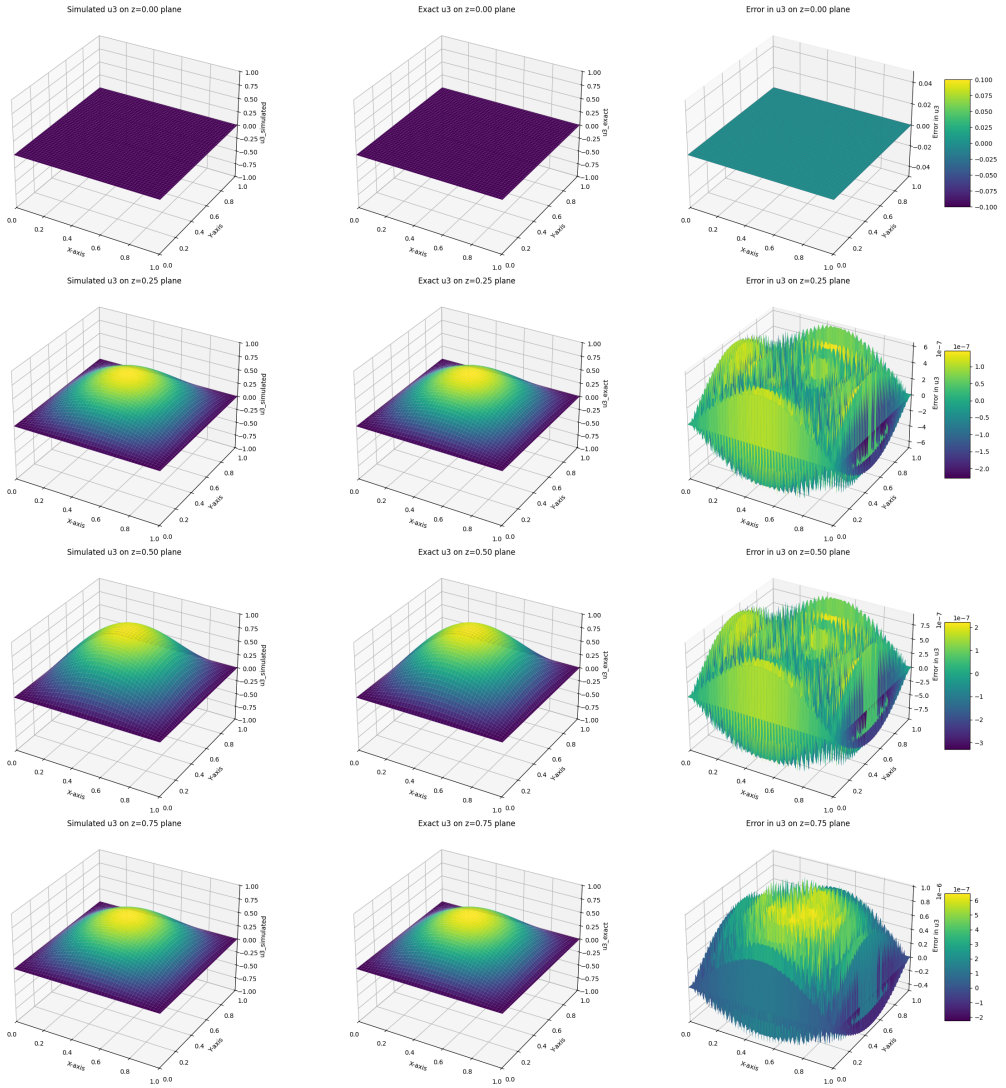


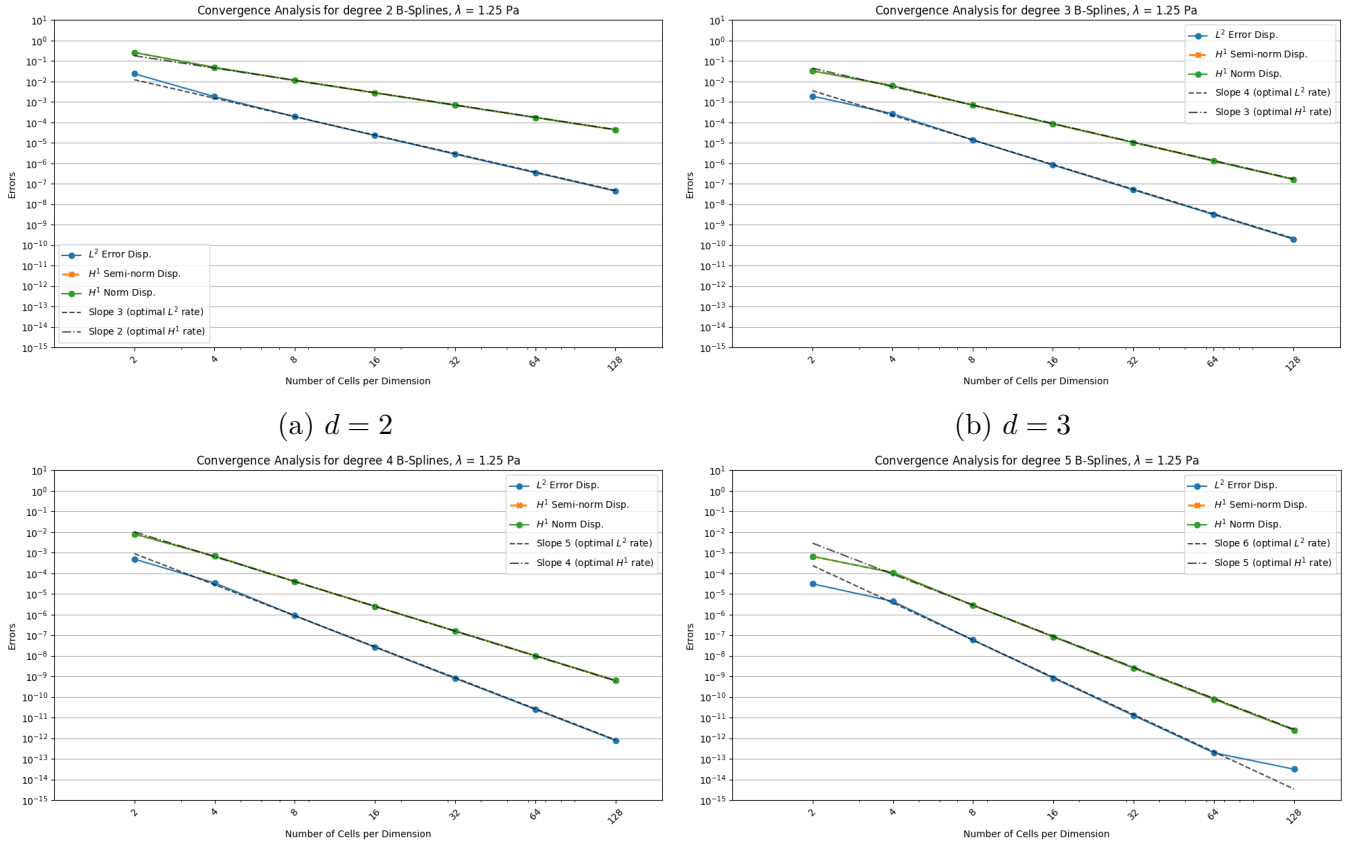
Figure 2.3: Plots comparison between the simulated and real $u_{e,3}$ with the error, for different planes z, with $d = 2$ and $n_{\text{cell}} = 64$

We can observe that the numerical solution is quite close to the real solution. To get a better estimation of the convergence rate, let's look at Céa's lemma in this case.

Let $\boldsymbol{V}^h \subset \boldsymbol{H}^1(\Omega)$ be the discrete finite dimensional space, $\vec{u} \in \boldsymbol{H}^{s+1}(\Omega) \subset \boldsymbol{H}^1(\Omega), 1 \leq s \leq d+1$ the real solution and $\vec{u}^h \in \boldsymbol{V}^h$ the discrete solution. Then, Céa's lemma gives the following estimates: $\left\|\vec{u} - \vec{u}^h\right\|_{\boldsymbol{H}^1(\Omega)} \leq \frac{3\lambda+2\mu}{2\mu C} \inf_{\vec{v}^h \in \boldsymbol{H}^1(\Omega)} \left\|\vec{u} - \vec{v}^h\right\|_{\boldsymbol{H}^1(\Omega)}$ *where* $C > 0$ *is a constant from Korn's inequality (independent of* h*).* By applying **Corollary 4.21, 4.27**, **Theorem 6.2** from [DV+14] and the Aubin-Nitsche theorem, we can get the following estimates:

$$\begin{aligned} \left\|\vec{u} - \vec{u}^h\right\|_{\boldsymbol{H}^1(\Omega)} &\lesssim \frac{3\lambda + 2\mu}{2\mu C} h^q \|\vec{u}\|_{\boldsymbol{H}^{q+1}(\Omega)} \\ \left\|\vec{u} - \vec{u}^h\right\|_{\boldsymbol{L}^2(\Omega)} &\lesssim (3\lambda + 2\mu) h^{q+1} \|\vec{u}\|_{\boldsymbol{H}^{q+1}(\Omega)} \end{aligned} \tag{2.6}$$

where $q = \min(s, d)$. In this case, u_e is an infinitely smooth solution, then $q = d$.



(a) $d = 2$ (b) $d = 3$

(c) $d = 4$ (d) $d = 5$

Figure 2.4: Four plots of absolute errors ($\left\|\vec{u} - \vec{u}^h\right\|_{\boldsymbol{H}^1(\Omega)}$ and $\left\|\vec{u} - \vec{u}^h\right\|_{\boldsymbol{L}^2(\Omega)}$) with different degree B-splines (with $\lambda = 1.25$ and $\mu = 1$)

We can see that the numerical solution converges with the expected order to the real solution: the more the mesh is refined, the lower the error is, and the higher the spline degree is, the higher the convergence rate is. For this numerical solution, I used the **GMRES** method to solve the linear system at the end of the discretization. This required taking care of the solver tolerance, because with a fixed tolerance, if d and n_{cells} are "too high", the convergence rate cannot hold. The following figure illustrates this case, when $d = 5$ and $n_{\text{cells}} = 16$.

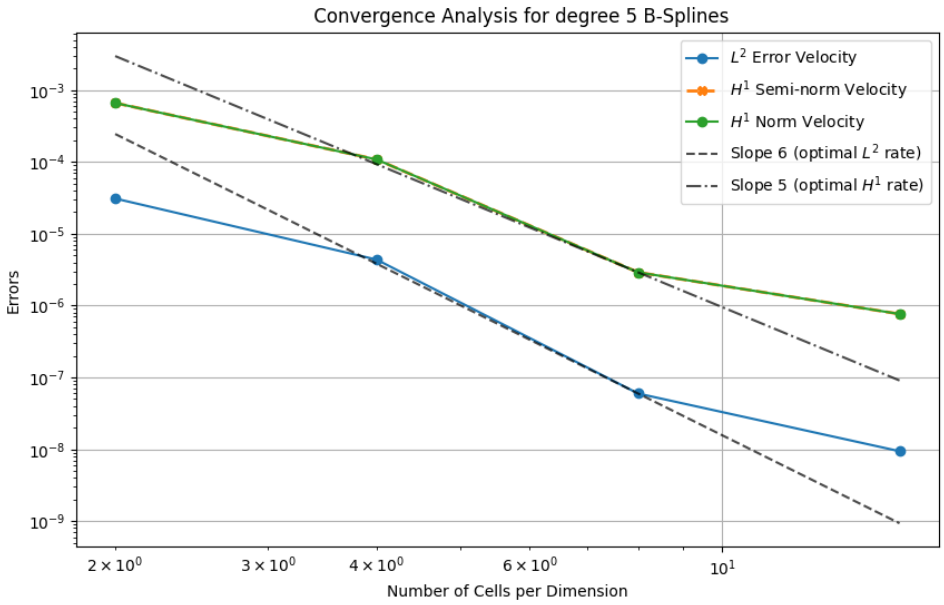


Figure 2.5: Error plot between exact and numerical solution with degree 5 B-splines with "high" tolerance for the solver

But even with a lower tolerance for the solver (10^{-15} for figure 2.4d), the error reaches a plateau, which is not the case for lower degree B-splines (at least for the tested configurations). This plateau can be explained by the matrix of the bilinear form $a(.,.)$ which is ill-conditioned. Moreover, the bad conditioning can become worse with more refined mesh and higher degree B-splines (as the convergence is faster). Consequently, the iterative solver stops when the maximum number of iterations is reached (the default value is 3000 iterations).

2.3.2 Mixed boundary conditions

Then, I made a simulation with mixed boundary conditions, as introduced in (2.3), with also non-homogeneous Dirichlet boundary conditions.

Again, $\Omega = [0,1]^3$, with:

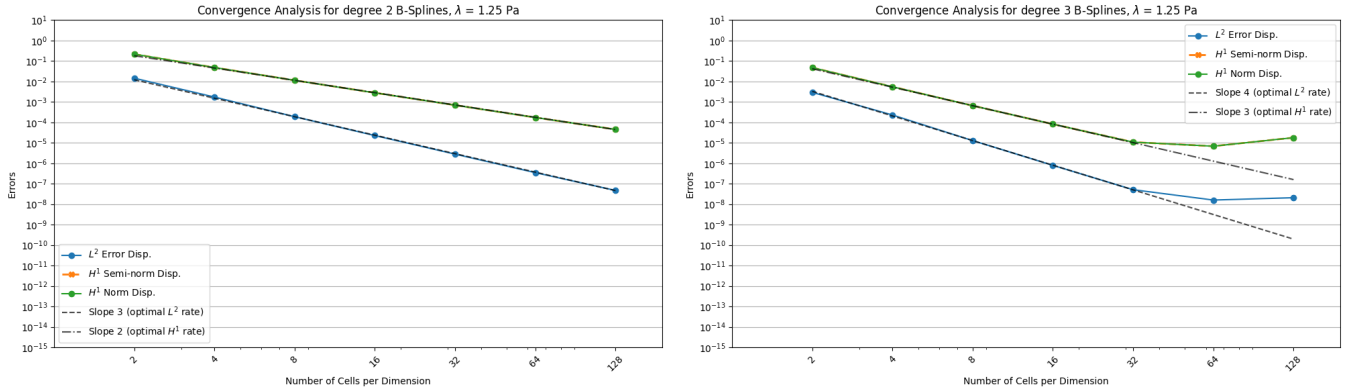
$$\partial\Omega_{\text{DH}} = \{(x,y,z) \in \partial\Omega \,|\, z = 0 \text{ or } z = 1\} \qquad \partial\Omega_{\text{DNH}} = \{(x,y,z) \in \partial\Omega \,|\, x = 0 \text{ or } x = 1\}$$
$$\partial\Omega_{\text{N}} = \{(x,y,z) \in \partial\Omega \,|\, y = 0 \text{ or } y = 1\}$$

For that purpose, $\vec{u}_e, \vec{f}, \vec{t}_N$ and $\vec{g}$ are now defined as:

$$\vec{u}_e = \begin{pmatrix} 0 \\ 0 \\ \sin(\pi z)\cos(\pi x)\cos(\pi y) \end{pmatrix}, \vec{f} = \begin{pmatrix} \pi^2(\lambda+\mu)\sin(\pi x)\cos(\pi y)\cos(\pi z) \\ \pi^2(\lambda+\mu)\sin(\pi y)\cos(\pi x)\cos(\pi z) \\ \pi^2(\lambda+4\mu)\sin(\pi z)\cos(\pi x)\cos(\pi y) \end{pmatrix}$$
$$\vec{t}_N = \begin{pmatrix} 0 \\ -\lambda\pi\cos(\pi x)\cos(\pi z) \\ 0 \end{pmatrix}, \vec{g} = \begin{pmatrix} 0 \\ 0 \\ \pm\sin(\pi z)\cos(\pi y) \end{pmatrix}$$

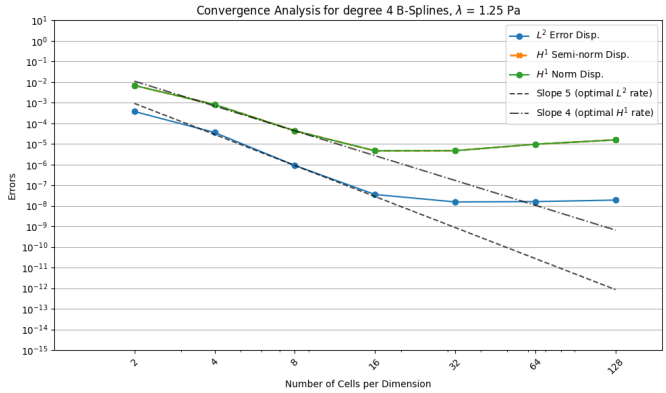
Then, we can get the same plots and interpretation as before in figure 2.6.

For fixed values $\lambda = 1.25$ and $\mu = 1$, the code seems to give a numerical solution that converges to the theoretical one with the expected order. We can notice that the plateau effect comes back and appears faster than in the case of Dirichlet homogeneous boundary conditions. The L^2 error seems to converge to a value slightly higher than 10^{-8}, while the H^1 error converges to a value slightly higher than 10^{-5}, and this phenomenon is more pronounced for higher degree B-splines. The

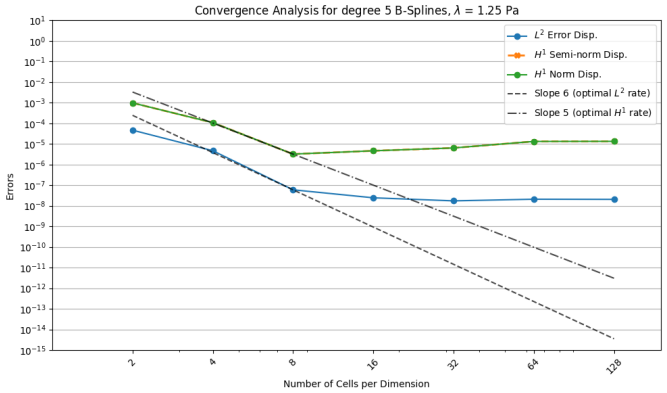


(a) $d = 2$

(b) $d = 3$



(c) $d = 4$



(d) $d = 5$

Figure 2.6: Four plots of absolute errors ($\left\|\vec{u} - \vec{u}^h\right\|_{\boldsymbol{H}^1(\Omega)}$ and $\left\|\vec{u} - \vec{u}^h\right\|_{\boldsymbol{L}^2(\Omega)}$) with different degree B-splines (with $\lambda = 1.25$ Pa and $\mu = 1$ Pa)

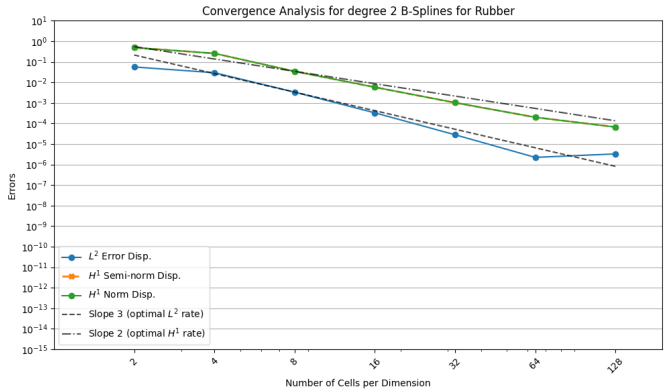
limit value of each error seems to be the same for all degree B-splines, when the L^2 error becomes lower than 10^{-7} and the H^1 error becomes lower than 10^{-5}. So, this plateau effect is not due to the degree of B-splines, but rather to the ill-conditioning of the matrix of the bilinear form $a(.,.)$. This ill-conditioning becomes worse with the refinement of the mesh and the increase of the degree of B-splines. A way to verify this hypothesis is to solve the linear system with a preconditioner applied to the matrices before the solution and see if the maximum number of iterations is reached.

These simulations have been made with $\lambda = 1.25$ and $\mu = 1$. We can now look at them for more realistic values of λ and μ. The following table 2.1 gives the Lamé coefficients for different materials.

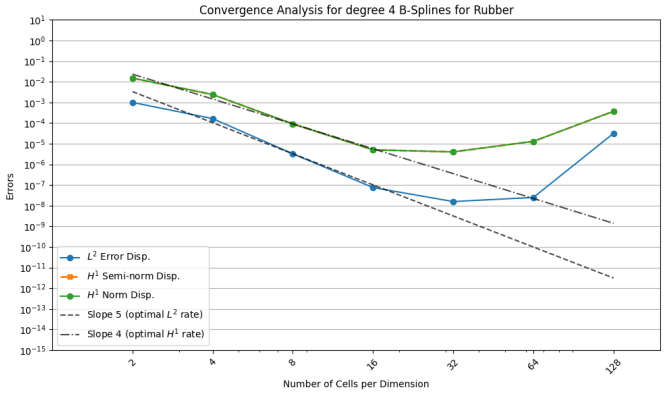
Material	λ (GPa)	μ (GPa)
Steel	120	80
Concrete	17	14
Rubber	0.16	0.00033

Table 2.1: Lamé coefficients for different materials

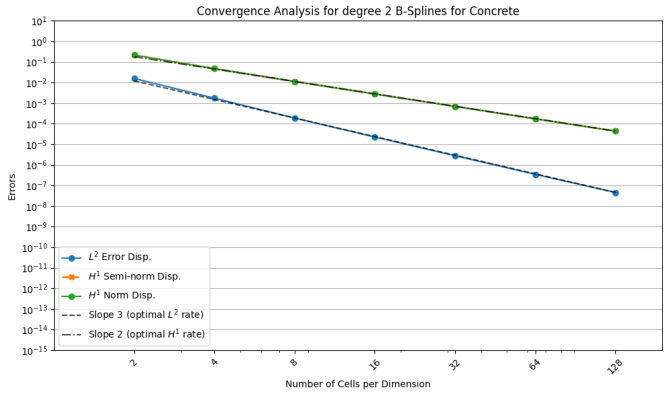
As we can see in figure 2.7, in the case of concrete and steel, the errors behave as expected: good convergence with expected order and the plateau effect appears after a certain value of the error. But for rubber, the convergence order is not really respected, the error decreases faster than expected and the plateau degenerates... To explain this, we can look at the estimates (2.6): $\left\|\vec{u} - \vec{u}^h\right\|_{\boldsymbol{H}^1(\Omega)} \lesssim \frac{3\lambda+2\mu}{2\mu C} h^d \|\vec{u}\|_{\boldsymbol{H}^{d+1}(\Omega)}$ As λ is very high compared to μ, the estimates are not really valid anymore, as $\frac{3\lambda+2\mu}{2\mu C} \to +\infty$ when $\lambda \to +\infty$. This is the case for incompressible materials, like



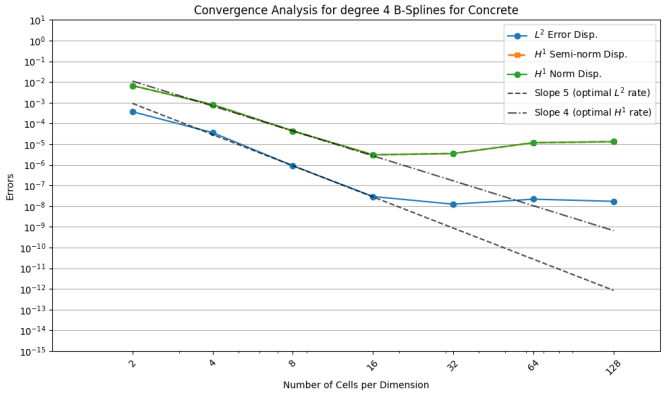
(a) Rubber with degree 2 B-splines



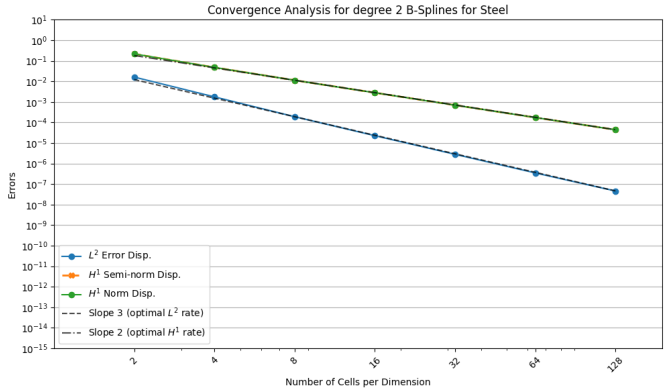
(b) Rubber with degree 4 B-splines



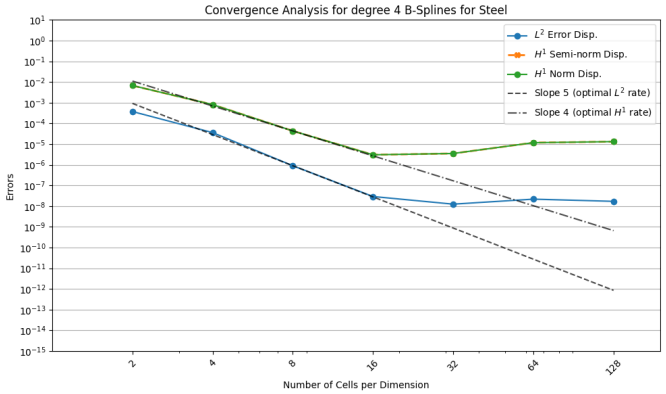
(c) Concrete with degree 2 B-splines



(d) Concrete with degree 4 B-splines



(e) Steel with degree 2 B-splines



(f) Steel with degree 4 B-splines

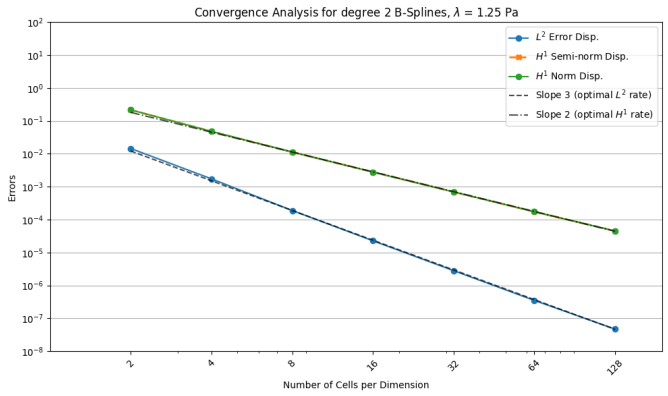
Figure 2.7: Error plots between exact solution and numerical solution with realistic values of Lamé coefficients with degree 2 and 4 B-splines

rubber, where λ is very high compared to μ. To get a better idea of the convergence, we can look at the error plots for different values of λ with $\mu = 1$.

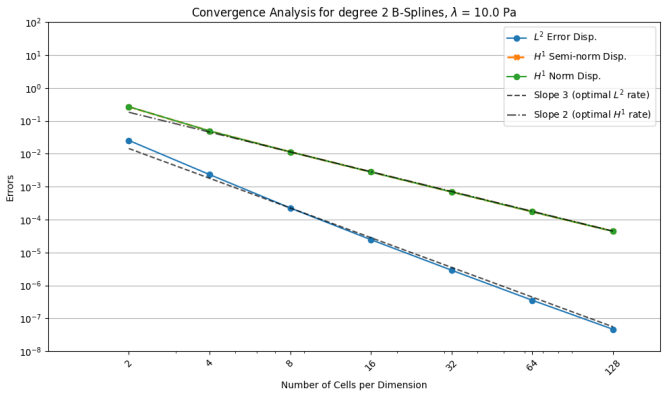
As we can see in figure 2.8, the convergence order is not respected anymore when λ is too high compared to μ. We can observe that when $\frac{\lambda}{\mu} = 10^4$, the error behaves like the error curves for rubber with degree 2 B-splines in figure 2.7a $\left(\frac{\lambda_{\text{rubber}}}{\mu_{\text{rubber}}} \approx 500\right)$. Then, for values of λ such that $\frac{\lambda}{\mu} \geq 10^4$, the error is not even decreasing anymore.

Two hypotheses can be made to explain this phenomenon:

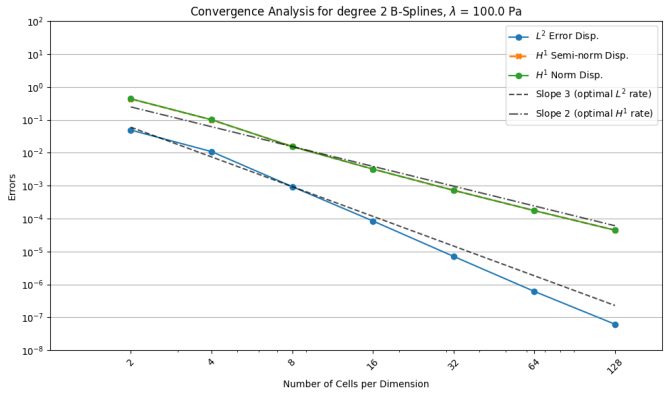
- The estimates (2.6) are not valid anymore, as λ is too high compared to μ.
- The bilinear form $a(.,.)$ is ill-conditioned, leading to numerical instability when λ is "too high"



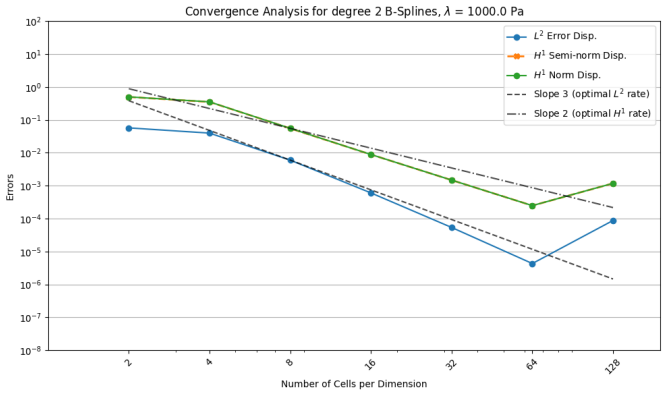
(a) $\lambda = 1.25$ Pa



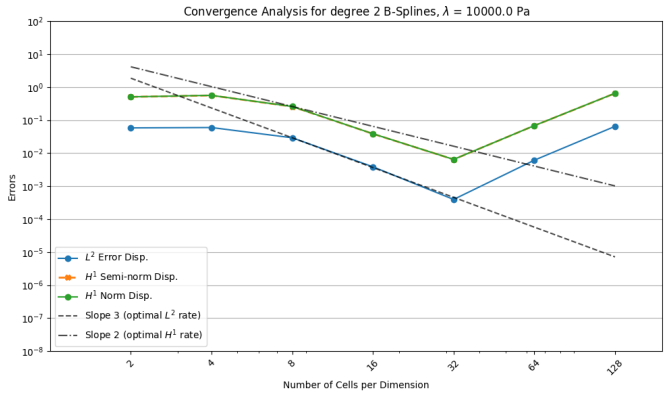
(b) $\lambda = 10$ Pa



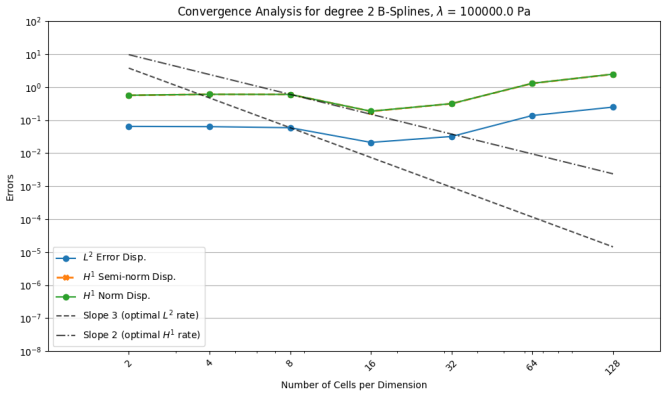
(c) $\lambda = 100$ Pa



(d) $\lambda = 1000$ Pa



(e) $\lambda = 10000$ Pa



(f) $\lambda = 100000$ Pa

Figure 2.8: Error plots between exact solution and numerical solution with degree 2 B-splines with increasing values of λ with $\mu = 1$ Pa

compared to μ.

One can try to increase the degree of B-splines to see if the convergence is reinforced as the power of h in the estimates is higher.

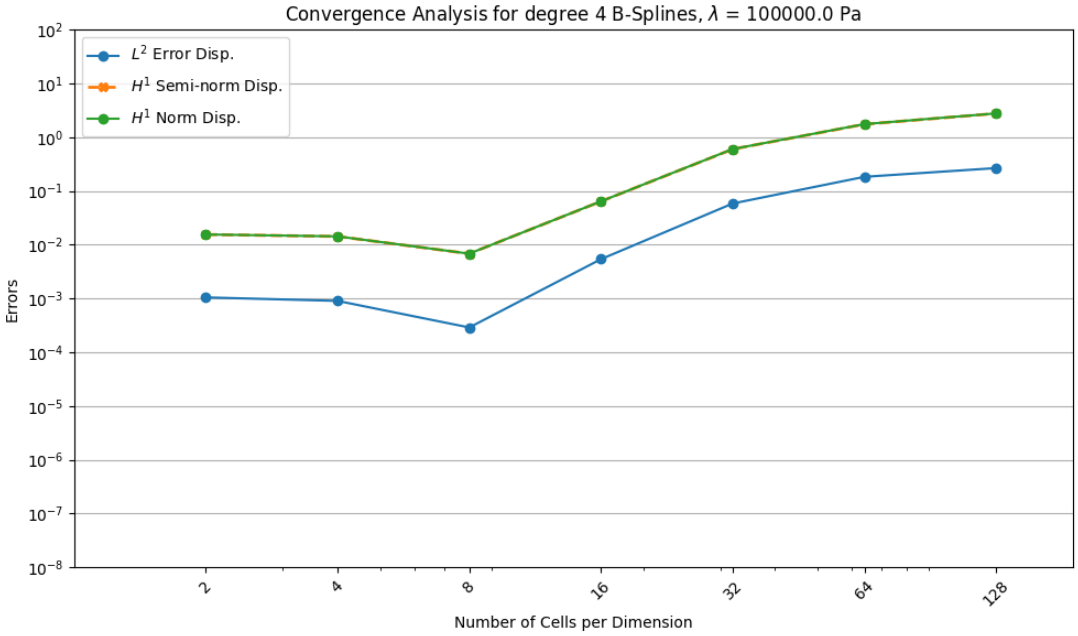


Figure 2.9: Error plot between exact solution and numerical solution with degree 4 B-splines with $\lambda = 100000$ Pa and $\mu = 1$ Pa

As we can see in figure 2.9, the convergence is not reinforced, the error is still not decreasing. Actually, this phenomenon is called **locking** and it is a well-known issue in FEM for incompressible materials. The locking occurs when the numerical solution is too stiff and does not converge to the real solution for high values of λ compared to μ. One way to avoid this issue is to use a mixed formulation, where the pressure field is introduced as an additional variable. This will be presented in the next section.

2.4 Mixed displacement-pressure formulation

To introduce this formulation, let's consider again the strong formulation from (2.3). Then, denote by p the pressure field, defined by $p = -\lambda \boldsymbol{\nabla} \cdot \vec{u}$. As we are seeking $\vec{u} \in \boldsymbol{H}^1(\Omega)$, p is naturally living in $L^2(\Omega)$. Then Hooke's law can be rewritten as: $\boldsymbol{\sigma}(\vec{u}, p) = -pI_3 + 2\mu\boldsymbol{\varepsilon}(\vec{\boldsymbol{u}})$ in Ω
So the strong formulation can be rewritten as: For a given $\vec{f} \in \boldsymbol{L}^2(\Omega)$, $\vec{t}_N \in \boldsymbol{L}^2(\partial\Omega_N)$, find $(\vec{u}, p) \in \boldsymbol{H}^1(\Omega) \times L^2(\Omega)$ such that:

$$\begin{cases} -\boldsymbol{\nabla} \cdot \boldsymbol{\sigma}(\vec{u}, p) = \vec{f} & \text{in } \Omega \quad \text{(Equilibrium)} \\ \boldsymbol{\nabla} \cdot \vec{u} + \dfrac{1}{\lambda} p = 0 & \text{in } \Omega \quad \text{(Pressure field definition)} \\ \vec{u} = \vec{0} & \text{on } \partial\Omega_D \\ \boldsymbol{\sigma}(\vec{u}) \cdot \vec{n} = \vec{t}_N & \text{on } \partial\Omega_N \end{cases} \tag{2.7}$$

Now, to derive the weak formulation, first we multiply the equilibrium equation by $v \in \boldsymbol{H}^1_{\vec{0},D}(\Omega)$ and integrate by parts on Ω, then we multiply the pressure field equation by $q \in L^2(\Omega)$ and integrate over Ω. Then we simply add both integrals. The weak formulation is then:

$$\begin{cases} \text{Find } (\vec{u},p) \in \boldsymbol{H}^1_{\vec{0},D}(\Omega) \times L^2(\Omega) \text{ such that:} \\ \tilde{a}((\vec{u},p),(\vec{v},q)) = l(\vec{v},q) \quad \forall (\vec{v},q) \in \boldsymbol{H}^1_{\vec{0},D}(\Omega) \times L^2(\Omega) \end{cases} \tag{2.8}$$

With:

$$\tilde{a} : \begin{cases} \left(\boldsymbol{H}^1_{\vec{0},D}(\Omega) \times L^2(\Omega)\right)^2 \to \mathbb{R} \\ ((\vec{u},p),(\vec{v},q)) \longmapsto \displaystyle\int_\Omega \left(2\mu\boldsymbol{\varepsilon}(\vec{u}) : \boldsymbol{\varepsilon}(\vec{v}) - p(\boldsymbol{\nabla}\cdot\vec{v}) + (\boldsymbol{\nabla}\cdot\vec{u})q + \frac{1}{\lambda}pq\right) \mathrm{d}\Omega \end{cases}$$

$$l : \begin{cases} \boldsymbol{H}^1_{\vec{0},D}(\Omega) \to \mathbb{R} \\ \vec{v} \longmapsto \displaystyle\int_\Omega \vec{f}\cdot\vec{v}\mathrm{d}\Omega + \int_{\partial\Omega_N} \vec{t}_N \cdot \vec{v}\mathrm{d}\Gamma \end{cases}$$

2.4.1 Well-posedness of the mixed displacement–pressure formulation

Let

$$V := \boldsymbol{H}^1_{\vec{0},D}(\Omega), \qquad Q := L^2(\Omega), \qquad X := V \times Q,$$

with the product norm

$$\|(\vec{u},p)\|_X^2 := \|\vec{u}\|^2_{\boldsymbol{H}^1(\Omega)} + \|p\|^2_{L^2(\Omega)}.$$

Now, we can prove the well-posedness of (2.8) **for every finite** $\lambda > 0, \mu > 0$ and $\Omega_D \neq \emptyset$.

- $(X, \|\cdot\|_X)$ is a Hilbert space.
- $\tilde{a}(\cdot,\cdot)$ is a bilinear form on $X \times X$ and $l(\cdot)$ is linear on X.
- $l(\cdot)$ is continuous: for all $(\vec{v},q) \in X$,

$$\begin{aligned} |l(\vec{v},q)| &\le |(\vec{f},\vec{v})_{\boldsymbol{L}^2(\Omega)}| + |(\vec{t}_N,\vec{v})_{\boldsymbol{L}^2(\partial\Omega_N)}| \\ &\le \left(\|\vec{f}\|_{\boldsymbol{L}^2(\Omega)} + \|\gamma_0\|\|\vec{t}_N\|_{\boldsymbol{L}^2(\partial\Omega_N)}\right)\|\vec{v}\|_{H^1(\Omega)} \\ |l(\vec{v},q)| &\le C\left(\|\vec{f}\|_{\boldsymbol{L}^2(\Omega)} + \|\vec{t}_N\|_{\boldsymbol{L}^2(\partial\Omega_N)}\right)\|(\vec{v},q)\|_X, \end{aligned}$$

  where γ_0 is the trace operator and C is a generic constant.

- $\tilde{a}(\cdot,\cdot)$ is continuous: for all $(\vec{u},p),(\vec{v},q) \in X$,

$$\begin{aligned} |\tilde{a}((\vec{u},p),(\vec{v},q))| &\le 2\mu\|\boldsymbol{\varepsilon}(\vec{u})\|_{L^2}\|\boldsymbol{\varepsilon}(\vec{v})\|_{L^2} + \|p\|_{L^2}\|\boldsymbol{\nabla}\cdot\vec{v}\|_{L^2} + \|\boldsymbol{\nabla}\cdot\vec{u}\|_{L^2}\|q\|_{L^2} + \tfrac{1}{\lambda}\|p\|_{L^2}\|q\|_{L^2} \\ &\le 2\mu\|\vec{u}\|_{\boldsymbol{H}^1}\|\vec{v}\|_{\boldsymbol{H}^1} + 3\|p\|_{L^2}\|\vec{v}\|_{\boldsymbol{H}^1} + 3\|\vec{u}\|_{\boldsymbol{H}^1}\|q\|_{L^2} + \frac{1}{\lambda}\|p\|_{L^2}\|q\|_{L^2} \\ &\le 2\mu\|(\vec{u},p)\|_X\|(\vec{v},q)\|_X + 3\|(\vec{u},p)\|_X\|(\vec{v},q)\|_X + 3\|(\vec{u},p)\|_X\|(\vec{v},q)\|_X \\ &\quad + \frac{1}{\lambda}\|(\vec{u},p)\|_X\|(\vec{v},q)\|_X \\ |\tilde{a}((\vec{u},p),(\vec{v},q))| &\le (2\mu + 6 + \frac{1}{\lambda})\|(\vec{u},p)\|_X\|(\vec{v},q)\|_X. \end{aligned}$$

  using $\|\boldsymbol{\nabla}\cdot\vec{w}\|_{L^2} \le 3\|\vec{w}\|_{H^1}$, $\|p\|_{L^2} \le \|(\vec{u},p)\|_X$ and $\|\vec{u}\|_{H^1} \le \|(\vec{u},q)\|_X$.

- $\tilde{a}(\cdot,\cdot)$ is coercive on X: for all $(\vec{u},p) \in X$,

$$\tilde{a}((\vec{u},p),(\vec{u},p)) = 2\mu\|\boldsymbol{\varepsilon}(\vec{u})\|^2_{L^2} + \tfrac{1}{\lambda}\|p\|^2_{L^2} \ge 2\mu C_K^2\|\vec{u}\|^2_{H^1} + \tfrac{1}{\lambda}\|p\|^2_{L^2} \ge \alpha\|(\vec{u},p)\|^2_X,$$

  with $\alpha := \min\{2\mu C_K^2, 1/\lambda\} > 0$ (for fixed finite λ).

Then, by the Lax–Milgram theorem, the mixed problem (2.8) admits a unique solution $(\vec{u}, p) \in X$ and the stability estimate

$$\|\vec{u}\|_{H^1(\Omega)} + \|p\|_{L^2(\Omega)} \leq C\Big(\|\vec{f}\|_{\boldsymbol{L}^2(\Omega)} + \|\vec{t}_N\|_{\boldsymbol{L}^2(\partial\Omega_N)}\Big)$$

holds with C independent of $(\vec{f}, \vec{t}_N)$.

Remark. As $\lambda \to \infty$ (near-incompressible limit), the coercivity constant degenerates on the pressure component; the formulation approaches a saddle-point (Stokes-like) problem requiring an inf-sup (Ladyzhenskaya–Babuška–Brezzi) condition on the pair (V, Q) to remain stable.

A way to ensure the inf-sup condition is to impose a zero mean value for the pressure field, i.e., $p \in L^2_0(\Omega)$, which is a common practice in numerical simulations of incompressible flows.

Then, the following inf-sup condition holds:

$$\inf_{q\in Q} \sup_{\vec{v}\in V} \frac{\tilde{a}((\vec{v}, q), (\vec{u}, p))}{\|\vec{v}\|_{H^1(\Omega)}\|q\|_{L^2(\Omega)}} \geq \beta > 0,$$

for some $\beta > 0$ independent of λ and μ. This condition ensures the stability and the well-posedness of the mixed formulation with $V = H^1_0(\Omega)$ and $Q = L^2_0(\Omega)$ [Bof+08].

If $\Omega_D = \emptyset$, the Dirichlet boundary condition is not imposed, and the problem becomes a pure Stokes problem. In this case, the uniqueness of the solution is guaranteed by the compatibility condition on the external loads, as discussed in Section 2.2 modulo a rigid body motion.

2.4.2 Isogeometric discretization and a priori error estimate

Let $(V_h, Q_h) \subset V \times Q$ be the tensor–product B-spline spaces of (vector) degree d (for the displacement) and (scalar) degree d (for the pressure) with mesh size h, and assume that the discrete inf–sup condition holds with a constant $\beta > 0$ independent of h, λ and μ (locking–free setting) [Bof+08]. Then, for the exact solution $(\vec{u}_e, p_e)$ of (2.8) and the discrete solution $(\vec{u}_h, p_h) \in V_h \times Q_h$, continuity plus coercivity on the kernel give the Céa-type bound

$$\|\vec{u}_e - \vec{u}_h\|_{\boldsymbol{H}^1(\Omega)} + \|p_e - p_h\|_{L^2(\Omega)} \;\leq\; C\left(\inf_{\vec{w}_h\in V_h} \|\vec{u}_e - \vec{w}_h\|_{\boldsymbol{H}^1(\Omega)} + \inf_{q_h\in Q_h} \|p_e - q_h\|_{L^2(\Omega)}\right),$$

with $C > 0$, and *a priori* $C \propto \frac{2\mu+6+1/\lambda}{\min(2\mu C_K^2, 1/\lambda)}$. For $(\vec{u}, p)$ sufficiently smooth, namely $\vec{u} \in \boldsymbol{H}^{d+1}(\Omega)$ and $p \in H^d(\Omega)$, spline approximation estimates yield

$$\|\vec{u}_e - \vec{u}_h\|_{\boldsymbol{H}^1(\Omega)} + \|p_e - p_h\|_{L^2(\Omega)} \;\leq\; Ch^d\Big(\|\vec{u}_e\|_{\boldsymbol{H}^{d+1}(\Omega)} + \|p_e\|_{H^d(\Omega)}\Big),$$

Then, while $2\mu C_K^2 > \frac{1}{\lambda}$, the coercivity constant α is bounded away from zero, and the estimate holds uniformly in λ and μ. But when $\lambda \to \infty$, the coercivity constant α tends to zero, leading to a loss of stability in the numerical method. Now we can look at the convergence estimates for the mixed displacement-pressure formulation to see if this formulation can solve for higher values of $\frac{\lambda}{\mu}$ than the pure-displacement formulation.

2.4.3 Example of simulation using `PSYDAC`

In this section, we will present the results of a simulation using the mixed displacement-pressure formulation with B-splines.

For the manufactured solution, we will use the same one as in Section 2.3.1. For reference, the exact solution is given by:

$$\vec{u}_e(x, y, z) = \begin{pmatrix} 0 \\ 0 \\ \sin(\pi x)\sin(\pi y)\sin(\pi z) \end{pmatrix}, \qquad p_e(x, y, z) = -\lambda\, \boldsymbol{\nabla}\cdot\vec{u}_e = -\lambda\pi \sin(\pi x)\sin(\pi y)\cos(\pi z).$$

$$\vec{f} = -\boldsymbol{\nabla} \cdot \boldsymbol{\sigma}(\vec{u}_e, p_e) = \begin{pmatrix} -\pi^2(\lambda+\mu)\cos(\pi x)\sin(\pi y)\cos(\pi z) \\ -\pi^2(\lambda+\mu)\sin(\pi x)\cos(\pi y)\cos(\pi z) \\ \pi^2(\lambda+4\mu)\sin(\pi x)\sin(\pi y)\sin(\pi z) \end{pmatrix}.$$

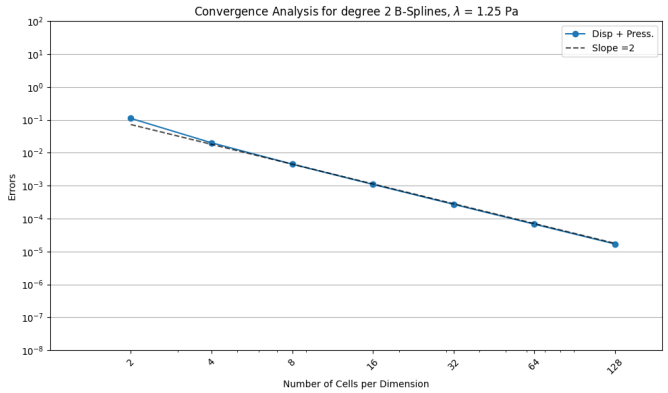
And the boundary conditions are:

$$\vec{u}_e|_{\partial\Omega} = 0,$$
$$p_e|_{\partial\Omega} = \begin{cases} -\lambda\pi\sin(\pi x)\sin(\pi y) & \text{if } z = 0 \\ \lambda\pi\sin(\pi x)\sin(\pi y) & \text{if } z = 1 \end{cases}$$

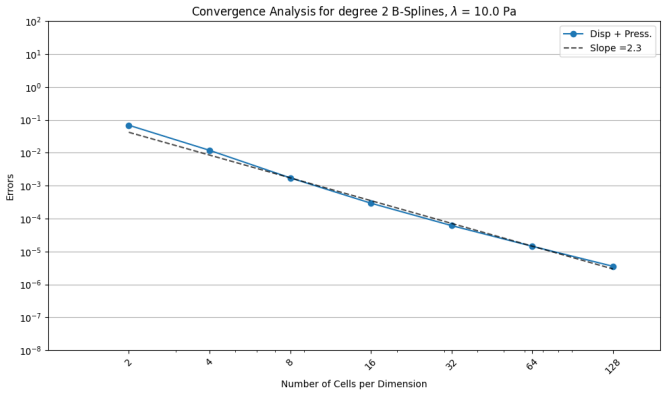
Since the exact solution now depends on λ, we will look at the relative error between exact and numerical solutions for different values of λ. (This wasn't necessary before as the exact solution was independent of λ, so the norm of the exact solution was the same for all values of λ). To do so, we can look at the relative error between exact and numerical solution, defined by:

$$E_{rel} = \frac{\|\vec{u}_e - \vec{u}_h\|_{H^1} + \|p_e - p_h\|_{L^2}}{\|\vec{u}_e\|_{H^1} + \|p_e\|_{L^2}}.$$

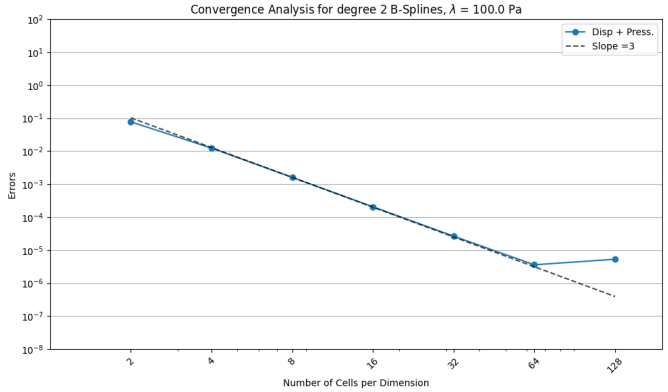
and its variation across the number of cells for different values of λ.



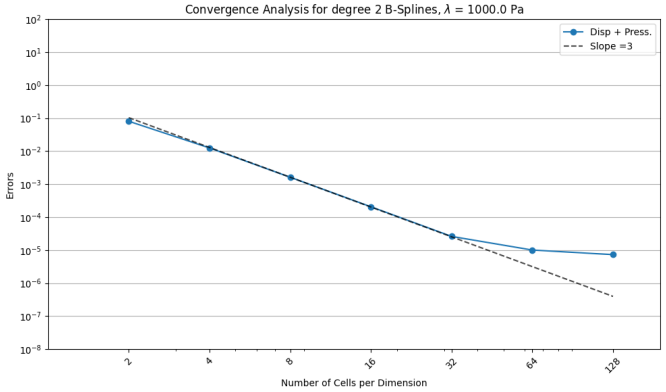
(a) $E_{rel}(n_{\text{cells}})$ with $\lambda = 1.25$ Pa and $\frac{\lambda}{\mu} = 1.25$



(b) $E_{rel}(n_{\text{cells}})$ with $\lambda = 10$ Pa and $\frac{\lambda}{\mu} = 10$



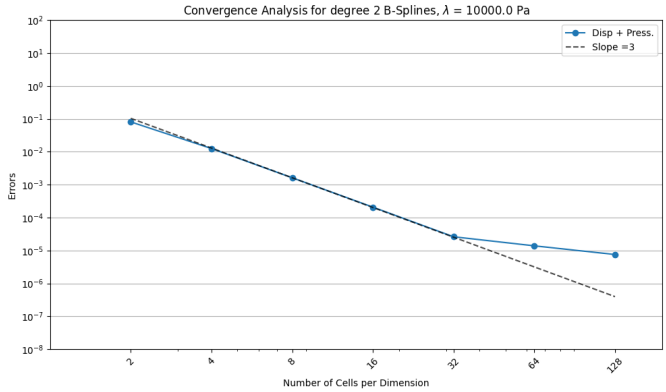
(c) $E_{rel}(n_{\text{cells}})$ with $\lambda = 100$ Pa and $\frac{\lambda}{\mu} = 100$



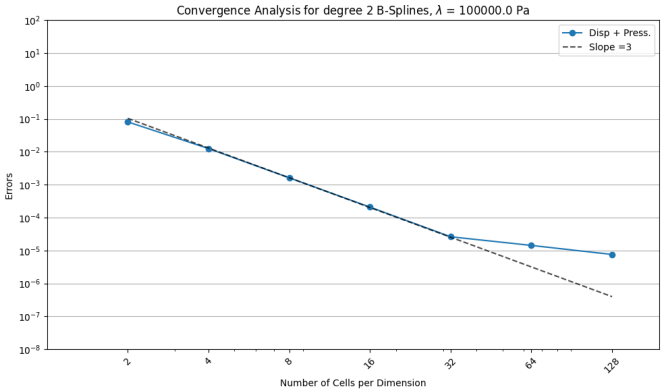
(d) $E_{rel}(n_{\text{cells}})$ with $\lambda = 10^3$ Pa and $\frac{\lambda}{\mu} = 1000$

The convergence analysis presented in Figure 2.10 reveals several important characteristics of the mixed displacement-pressure formulation across different Lamé parameter values. This comprehensive study examines the behavior of the numerical scheme for values of λ ranging from 1.25 Pa to 10^{16} Pa, corresponding to materials from highly compressible to nearly incompressible regimes.

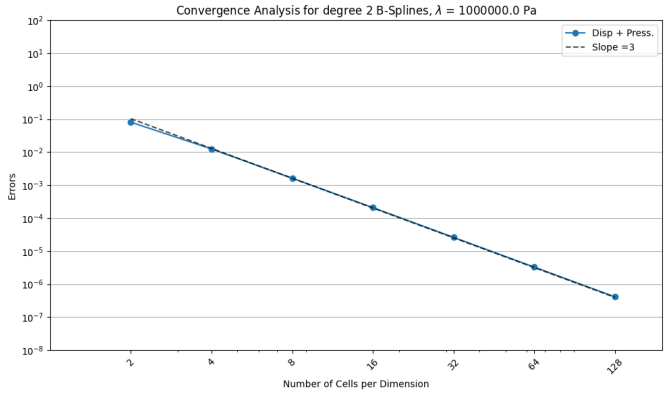
We can observe that from $\frac{\lambda}{\mu} = 100$, the convergence behavior is robust and follows a -3 slope. This indicates that the mixed formulation maintains optimal convergence properties even as the



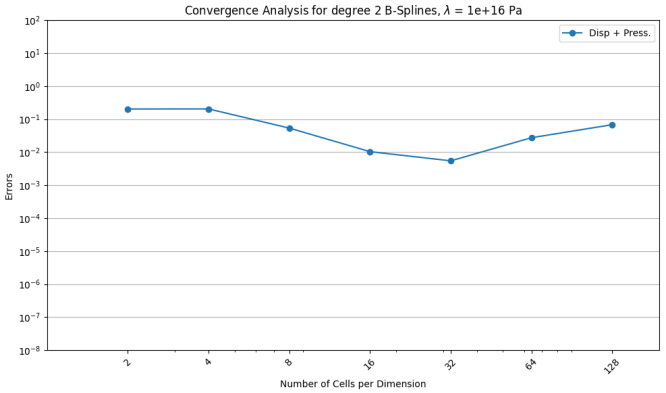
(e) $E_{rel}(n_{\text{cells}})$ with $\lambda = 10^4$ Pa and $\frac{\lambda}{\mu} = 10000$



(f) $E_{rel}(n_{\text{cells}})$ with $\lambda = 10^5$ Pa and $\frac{\lambda}{\mu} = 100000$



(g) $E_{rel}(n_{\text{cells}})$ with $\lambda = 10^6$ Pa and $\frac{\lambda}{\mu} = 1000000$



(h) $E_{rel}(n_{\text{cells}})$ with $\lambda = 10^{16}$ Pa and $\frac{\lambda}{\mu} = 10^{16}$

Figure 2.10: Relative error between exact and numerical solution with degree 2 B-splines for different values of λ

material approaches incompressibility. The observed slope is higher than expected. This could be due to the fact that as λ increases, the pressure variable becomes more dominant in the formulation and influences the overall convergence behavior. This is only an intuition and an interpretation of the error plots, and further analysis would be needed to confirm this hypothesis. That has not been done yet. The plateau phenomenon is observed again when the error reaches a certain threshold and no longer decreases significantly with mesh refinement.

However, when $\frac{\lambda}{\mu} \gg 1$, so here $\frac{\lambda}{\mu} = 10^{16}$, the same phenomenon as in the pure displacement formulation occurs, where the error does not decrease anymore. This is likely due to the fact that the pressure variable is not able to compensate for the locking phenomenon that occurs when λ is too high compared to μ. We can suppose that the coercivity constant α has reached its limit and now $\alpha = \frac{1}{\lambda}$.

When we look at the continuity and coercivity constants of the bilinear form a, one can also think that the more μ becomes higher, the more this loss of coercivity will appear faster when $\lambda \to \infty$.

For example, if we consider steel's Lamé parameters, which are typically around $\lambda = 120$ GPa and $\mu = 80$ GPa, we find that $\frac{\lambda}{\mu} = 1.5$. As we can see in the following error plot, the locking phenomenon appears.

The same behavior is actually observed for rubber and concrete, even if the ratio $\frac{\lambda}{\mu}$ is not as high as in 2.10h. This is due to the fact that with the mixed formulation, when μ increases, the coercivity constant α becomes smaller faster when λ increases.

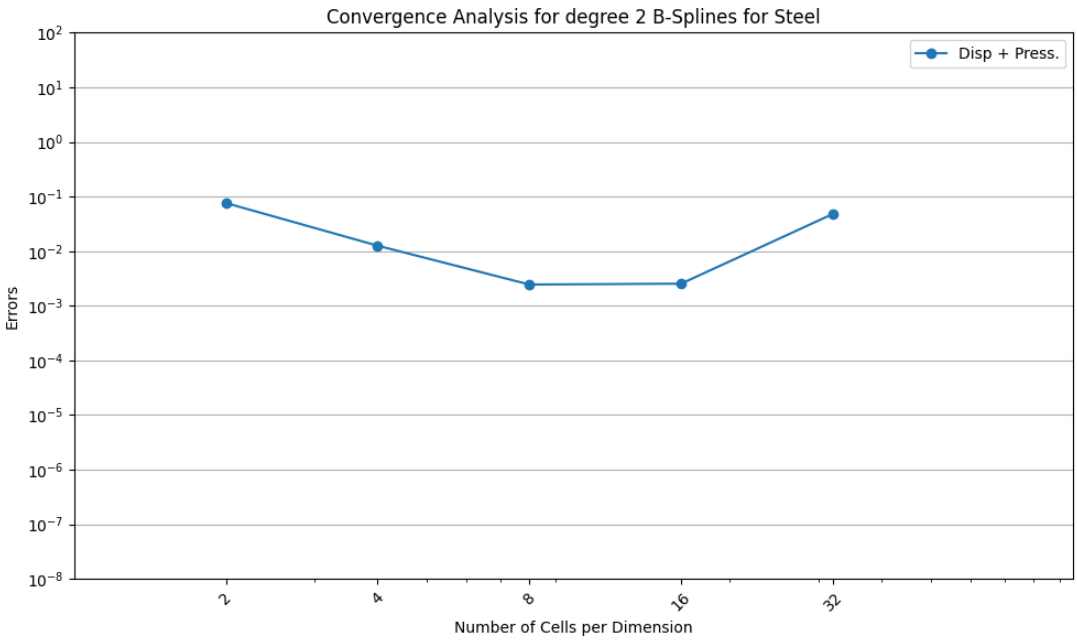


Figure 2.11: Relative error between exact and numerical solution with degree 2 B-splines for steel's Lamé parameters

2.5 Conclusion

This chapter presented a study of linear elasticity formulations implemented with `PSYDAC`, demonstrating the library's capability to handle solid mechanics problems beyond its original plasma physics applications. Two distinct formulations were investigated: the pure displacement formulation and the mixed displacement-pressure formulation, each offering unique advantages for different material regimes.

The pure displacement formulation proved effective for materials with moderate Lamé parameter ratios, achieving optimal convergence rates as predicted by theoretical estimates. However, the study revealed significant limitations when dealing with nearly incompressible materials, where the ratio $\frac{\lambda}{\mu}$ becomes large. The locking phenomenon observed for values of $\frac{\lambda}{\mu} \geq 10^4$ demonstrates the practical limitations of this approach for such materials, with convergence rates deteriorating significantly or even failing entirely.

The mixed displacement-pressure formulation successfully addressed many of the incompressibility issues inherent in the pure displacement approach. By introducing pressure as an additional variable, this formulation maintained robust convergence behavior for much higher values of $\frac{\lambda}{\mu}$, effectively extending the applicable range to materials approaching the incompressible limit.

Both formulations were thoroughly validated using the method of manufactured solutions across various material parameters, mesh refinements, and B-spline degrees. The convergence studies confirmed the expected optimal rates, with the occasional plateau effects attributed to matrix ill-conditioning at very fine mesh resolutions and a limited number of solver iterations. The successful implementation on the Raven supercomputer demonstrated `PSYDAC`'s scalability and performance capabilities for large-scale solid mechanics simulations.

The comprehensive comparison between these formulations provides valuable insights for practitioners choosing appropriate numerical methods based on material properties. For standard engineering materials like steel and concrete, where $\frac{\lambda}{\mu}$ remains moderate, the pure displacement formulation offers computational efficiency and straightforward implementation. For applications involving nearly incompressible materials or when robustness across a wide range of material parameters is required, the mixed formulation provides superior stability and accuracy.

This work has contributed to expanding `PSYDAC`'s application domain while maintaining its high-order accuracy and geometric flexibility. The implementations developed during this study are available as interactive Jupyter notebooks demonstrating both formulations, accessible through the IGA-Python repository: https://github.com/pyccel/IGA-Python/pull/21. These resources provide practical examples and can serve as starting points for further developments in computational solid mechanics using isogeometric analysis.

Chapter 3

Thermoelasticity

3.1 Introduction

This chapter is dedicated to the study of the thermoelasticity problem, which is a coupled problem between the mechanical and thermal fields. The aim is to solve the coupled system of equations that governs the behavior of a thermoelastic material under thermal and mechanical loads. This work was initiated to solve a problem presented during the IPP Programme Days 2025 by Prof. Dr. Rudolf Neu. The goal is to understand the deformation of a medium under the influence of thermal loads and mechanical forces. The medium is a composite material, composed of copper and tungsten. The geometry is presented in the following figure.

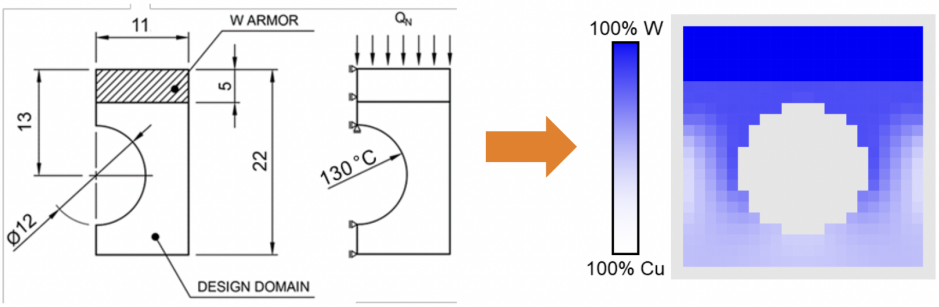


Figure 3.1: Geometry of the thermoelasticity problem from the IPP Programme Days 2025

3.2 Simplified problem - Physical model and equations

Before solving this problem, I want to simplify it a bit by considering a slightly different geometry, which is a square without the hole in the middle. The goal is to understand the behavior of the medium under thermal and mechanical loads, to see how the deformation of the medium is affected by the thermal loads and to see if PSYDAC can effectively solve this simplified problem. The geometry of the simplified problem is presented in figure 3.2.

3.2.1 Problem setup and parameters

The problem can be parameterized as follows with other boundary conditions and parameters:

- The domain Ω is a square of size 22: $\Omega = [0, 22]^2$.

- $\partial\Omega_{\text{heat}} = \{(x, y) \in \partial\Omega \,|\, x = 0 \text{ and } x = 22\}$ is the left and right edges of the square, where heat Q is applied. $\partial\Omega_{\text{temp}} = \{(x, y) \in \partial\Omega \,|\, y = 0 \text{ and } y = 22\}$ is the top and bottom edges of the square, where the temperature is fixed to T_0.

- $\partial\Omega_D = \{(x, y) \in \partial\Omega \,|\, x = 0 \text{ and } x = 22\}$ are the left and right edges of the square, where the Dirichlet boundary conditions are applied and the displacement is fixed. $\partial\Omega_N = \{(x, y) \in \partial\Omega \,|\, y = 0 \text{ and } y = 22\}$ are the top and bottom edges of the square, where the Neumann boundary conditions are applied and the traction is fixed.

- For $y \in (0, 17)$, the material is composite and properties are given by an affine interpolation between copper and tungsten properties. For $y \in (17, 22)$, the material is pure tungsten. For example, λ is defined as:

$$\lambda(y) = \begin{cases} \lambda_{\text{Cu}} + \frac{y}{17}(\lambda_{\text{W}} - \lambda_{\text{Cu}}) & \text{if } y \in (0, 17) \\ \lambda_{\text{W}} & \text{if } y \in (17, 22) \end{cases}$$

- The material properties are given by:

Property	**Cu**	**W**
Lamé's first constant λ [Pa]	1.1×10^{11}	2.5×10^{11}
Shear modulus μ [Pa]	4.1×10^{10}	1.6×10^{11}
Thermal expansion α [1/K]	1.7×10^{-5}	4.5×10^{-6}
Thermal conductivity κ [W/(m·K)]	401	174

Table 3.1: Material properties of Copper (Cu) and Tungsten (W).

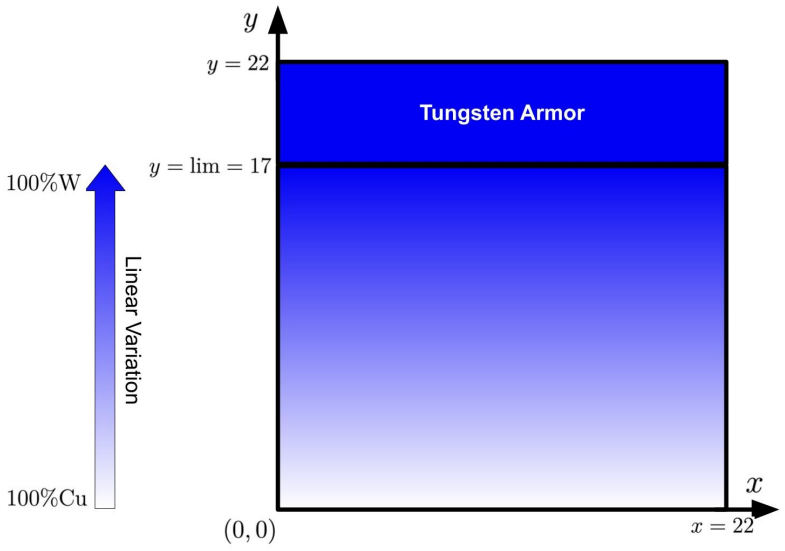


Figure 3.2: Geometry of the simplified thermoelasticity problem

3.2.2 Strong Formulation

The presentation of the strong formulation of the thermoelasticity problem is based on the principles of continuum mechanics and thermodynamics taken from [MS15] and [Per16]. The thermoelasticity problem is a coupled problem between the mechanical and thermal fields. The equations

that govern the behavior of a thermoelastic material are given by the coupling of the heat equation and the elasticity equation.

The full time-dependent thermoelasticity equations are:

$$\begin{cases} \rho(y)c_p(y)\dfrac{\partial T}{\partial t} + \boldsymbol{\nabla}\cdot(\kappa(y)\boldsymbol{\nabla}T) = Q - (3\lambda(y)+2\mu(y))\alpha(y)T_0\dfrac{\partial}{\partial t}(\boldsymbol{\nabla}\cdot\vec{u}) \\ \rho(y)\dfrac{\partial^2\vec{u}}{\partial t^2} - \boldsymbol{\nabla}\cdot\boldsymbol{\sigma}(\vec{u}) = \vec{f} \\ \boldsymbol{\sigma}(\vec{u}) = \lambda(y)(\boldsymbol{\nabla}\cdot\vec{u})I_3 + 2\mu(y)\boldsymbol{\varepsilon}(\vec{u}) - (3\lambda(y)+2\mu(y))\alpha(y)TI_3 \end{cases} \tag{3.1}$$

where $\rho(y)$ is the density, $c_p(y)$ is the specific heat capacity at constant pressure, T is the temperature, $\kappa(y)$ is the thermal conductivity, Q is the heat source, $\vec{u}$ is the displacement field, $\vec{f}$ is the body force, $\boldsymbol{\sigma}$ is the stress tensor, $\lambda(y)$ and $\mu(y)$ are the Lamé coefficients, $\boldsymbol{\varepsilon}$ is the strain tensor, $\alpha(y)$ is the thermal expansion coefficient and T_0 is the reference temperature.

For this study, we will consider the stationary case, where the time derivatives are zero ($\frac{\partial T}{\partial t} = 0$ and $\frac{\partial^2\vec{u}}{\partial t^2} = 0$). This simplifies the system to:

$$\begin{cases} -\boldsymbol{\nabla}\cdot(\kappa(y)\boldsymbol{\nabla}T) = Q \\ -\boldsymbol{\nabla}\cdot\boldsymbol{\sigma}(\vec{u}) = \vec{f} \\ \boldsymbol{\sigma}(\vec{u}) = \lambda(y)(\boldsymbol{\nabla}\cdot\vec{u})I_3 + 2\mu(y)\boldsymbol{\varepsilon}(\vec{u}) - (3\lambda(y)+2\mu(y))\alpha(y)TI_3 \end{cases}$$

These stationary equations represent the equilibrium state of the thermoelastic material under constant thermal and mechanical loads.

To complete the problem formulation, we must specify the boundary conditions. Based on the problem description, we have the following conditions on the boundary $\partial\Omega$:

For the thermal problem:

$$T = T_0 \qquad \text{on } \partial\Omega_{\text{temp}} \tag{3.2}$$

$$-\kappa\boldsymbol{\nabla}T\cdot\vec{n} = q_S \qquad \text{on } \partial\Omega_{\text{heat}} \tag{3.3}$$

where T_0 is the reference temperature and q_S is the applied heat flux on the boundary.

For the mechanical problem:

$$\vec{u} = \vec{u}_D \qquad \text{on } \partial\Omega_D \tag{3.4}$$

$$\boldsymbol{\sigma}(\vec{u})\cdot\vec{n} = \vec{t}_N \qquad \text{on } \partial\Omega_N \tag{3.5}$$

where $\vec{u}_D$ is the prescribed displacement and $\vec{t}_N$ is the prescribed traction. To prove the well-posedness of the problem, we will limit the analysis to $T_0 = 0$ and $\vec{u}_D = \vec{0}$, otherwise the problem can be rewritten with homogeneous boundary conditions with a change of variables.

For a given $Q \in L^2(\Omega)$, $q_S \in L^2(\partial\Omega_{\text{heat}})$, $\vec{f} \in \boldsymbol{L}^2(\Omega)$, $\vec{t}_N \in \boldsymbol{L}^2(\partial\Omega_N)$, the entire strong problem is:

Find $(T, \vec{u}) \in H^1(\Omega) \times \boldsymbol{H}^1_{\vec{0},D}(\Omega)$ such that:

$$-\boldsymbol{\nabla}\cdot(\kappa(y)\boldsymbol{\nabla}T) = Q \qquad \text{in } \Omega \tag{3.6}$$

$$-\boldsymbol{\nabla}\cdot\boldsymbol{\sigma}(\vec{u}) = \vec{f} \qquad \text{in } \Omega \tag{3.7}$$

$$\boldsymbol{\sigma}(\vec{u}) = \lambda(y)(\boldsymbol{\nabla}\cdot\vec{u})I_3 + 2\mu(y)\boldsymbol{\varepsilon}(\vec{u}) - (3\lambda(y)+2\mu(y))\alpha(y)TI_3 \qquad \text{in } \Omega \tag{3.8}$$

$$\vec{u} = \vec{0} \qquad \text{on } \partial\Omega_D \tag{3.9}$$

$$\boldsymbol{\sigma}(\vec{u})\cdot\vec{n} = \vec{t}_N \qquad \text{on } \partial\Omega_N \tag{3.10}$$

$$T = 0 \qquad \text{on } \partial\Omega_{\text{temp}} \tag{3.11}$$

$$-\kappa(y)\boldsymbol{\nabla}T\cdot\vec{n} = q_S \qquad \text{on } \partial\Omega_{\text{heat}} \tag{3.12}$$

This problem can be seen as a coupled thermo-mechanical problem, where the temperature field T affects the mechanical behavior of the material through the thermal expansion term in the stress tensor $\boldsymbol{\sigma}(\vec{u})$, but the mechanical displacement $\vec{u}$ does not directly affect the temperature field T. So, to solve this problem, one can first solve the thermal problem to obtain the temperature field T, and then use this temperature field to solve the mechanical problem for the displacement $\vec{u}$ with a modified source term that includes the thermal expansion effects. As $\forall y \in [0, 22], \dfrac{\lambda(y)}{\mu(y)} \leq 50$, according to results from Subsection 2.3.2, we can use the pure displacement formulation to solve the mechanical problem.

3.2.3 Weak Formulation

The corresponding weak formulation is obtained by multiplying the PDEs by suitable test functions, integrating over the domain Ω, and applying integration by parts.

Let the solution space for temperature be $H^1_{0,\text{temp}} = \{\phi \in H^1(\Omega) \mid \phi = 0 \text{ on } \partial\Omega_{\text{temp}}\}$ and for displacement be $\boldsymbol{H}^1_{\vec{0},D}(\Omega)$. The weak problem is:

Find $(T, \vec{u}) \in H^1_{0,\text{temp}} \times \boldsymbol{H}^1_{\vec{0},D}(\Omega)$ such that for all test functions $(\theta, \vec{v}) \in H^1_{0,\text{temp}} \times \boldsymbol{H}^1_{\vec{0},D}(\Omega)$:

$$\begin{cases} b(T, \theta) = l_T(\theta) \\ a(\vec{u}, \vec{v}) = l_U(\vec{v}) \end{cases} \tag{3.13}$$

With

$$b : \begin{cases} H^1_{0,\text{temp}} \times H^1_{0,\text{temp}} \longrightarrow \mathbb{R} \\ (T, \theta) \mapsto \displaystyle\int_\Omega \kappa(y) \boldsymbol{\nabla} T \cdot \boldsymbol{\nabla} \theta \, d\Omega \end{cases}$$

$$l_T : \begin{cases} H^1_{0,\text{temp}} \longrightarrow \mathbb{R} \\ \theta \mapsto \displaystyle\int_\Omega Q\theta \, d\Omega - \int_{\partial\Omega_{\text{heat}}} q_S \theta \, dS \end{cases}$$

$$a : \begin{cases} \boldsymbol{H}^1_{\vec{0},D}(\Omega) \times \boldsymbol{H}^1_{\vec{0},D}(\Omega) \longrightarrow \mathbb{R} \\ (\vec{u}, \vec{v}) \mapsto \displaystyle\int_\Omega \Big(\lambda(y)(\boldsymbol{\nabla} \cdot \vec{u})(\boldsymbol{\nabla} \cdot \vec{v}) + 2\mu(y)\boldsymbol{\varepsilon}(\vec{u}) : \boldsymbol{\varepsilon}(\vec{v}) \Big) \, d\Omega \end{cases}$$

$$l_U : \begin{cases} \boldsymbol{H}^1_{\vec{0},D}(\Omega) \longrightarrow \mathbb{R} \\ \vec{v} \mapsto \displaystyle\int_\Omega \vec{f} \cdot \vec{v} \, d\Omega + \int_{\partial\Omega_N} \vec{t}_N \cdot \vec{v} \, dS + \int_\Omega (3\lambda(y) + 2\mu(y))\alpha(y) T (\boldsymbol{\nabla} \cdot \vec{v}) \, d\Omega \end{cases}$$

$\exists \kappa_-, \kappa_+, \lambda_-, \lambda_+, \mu_-, \mu_+, \alpha_-, \alpha_+ > 0$ such that $\forall y \in [0, 22]$,

$0 < \kappa_- \leq \kappa(y) \leq \kappa_+ < +\infty \qquad 0 < \lambda_- \leq \lambda(y) \leq \lambda_+ < +\infty$

$0 < \mu_- \leq \mu(y) \leq \mu_+ < +\infty \qquad 0 < \alpha_- \leq \alpha(y) \leq \alpha_+ < +\infty.$

Now, we prove the well-posedness of the thermal problem.

- $(H^1_{0,\text{heat}}(\Omega), \|.\|_{H^1(\Omega)})$ is a Hilbert space.
- $b(\cdot, \cdot)$ is a bilinear form on $H^1_{0,\text{heat}}(\Omega) \times H^1_{0,\text{heat}}(\Omega)$ and ℓ_T is a linear form on $H^1_{0,\text{heat}}(\Omega)$.
- $\ell_T(\cdot)$ is continuous: by Cauchy–Schwarz and the trace inequality,

$$\begin{aligned} \forall \theta \in H^1_{0,\text{heat}}(\Omega), \qquad |\ell_T(\theta)| &\leq \|Q\|_{L^2(\Omega)} \|\theta\|_{L^2(\Omega)} + \|q_S\|_{L^2(\partial\Omega_{\text{heat}})} \|\theta\|_{L^2(\partial\Omega_{\text{heat}})} \\ &\leq \Big(\|Q\|_{L^2(\Omega)} + \|\gamma_0\| \, \|q_S\|_{L^2(\partial\Omega_{\text{heat}})} \Big) \|\theta\|_{H^1(\Omega)}, \end{aligned}$$

where $\gamma_0 : H^1(\Omega) \to L^2(\partial\Omega_{\text{heat}})$ is the trace operator.

- $b(\cdot,\cdot)$ is continuous:

$$\forall T, \theta \in H^1_{0,\text{heat}}(\Omega), \qquad |b(T,\theta)| \leq \kappa_+ \|\nabla T\|_{L^2(\Omega)} \|\nabla\theta\|_{L^2(\Omega)} \leq \kappa_+ \|T\|_{H^1(\Omega)} \|\theta\|_{H^1(\Omega)}.$$

- $b(\cdot,\cdot)$ is coercive: $\forall T \in H^1_{0,\text{heat}}(\Omega)$,

$$b(T,T) = \int_\Omega \kappa(y) |\nabla T|^2 \,\mathrm{d}x \geq \kappa_{\min} \|\nabla T\|^2_{L^2(\Omega)} \geq \kappa_- \left(\frac{1}{2} \|\nabla T\|^2_{L^2(\Omega)} + \frac{1}{2} \|\nabla T\|^2_{L^2(\Omega)} \right).$$

Since functions in $H^1_{0,\text{heat}}(\Omega)$ vanish on $\partial\Omega_{\text{temp}}$ and a Poincaré inequality holds on $H^1_{0,\text{heat}}(\Omega)$, there exists $C_P > 0$ such that $\|T\|_{L^2(\Omega)} \leq C_P \|\nabla T\|_{L^2(\Omega)}$.

$$b(T,T) \geq \kappa_- \left(\frac{1}{2} \|\nabla T\|^2_{L^2(\Omega)} + \frac{1}{2C_P^2} \|T\|^2_{L^2(\Omega)} \right) \geq \alpha \|T\|^2_{H^1(\Omega)}$$

$$\text{With } \alpha = \min\left(\frac{\kappa_-}{2}, \frac{\kappa_-}{2C_P^2} \right) > 0.$$

Therefore $b(\cdot,\cdot)$ is continuous and coercive on the Hilbert space $H^1_{0,\text{heat}}(\Omega)$, and $\ell_T \in H^1_{0,\text{heat}}(\Omega)'$. By the Lax–Milgram theorem there exists a unique $T \in H^1_{0,\text{heat}}(\Omega)$ such that

$$b(T,\theta) = \ell_T(\theta) \qquad \forall \theta \in H^1_{0,\text{heat}}(\Omega).$$

The well-posedness of the displacement problem is similar to Section 2.2. Here, we can add that ℓ_U is continuous, a is continuous and coercive with new constants:

- $\forall v \in \boldsymbol{H}^1_{\vec{0},D}(\Omega)$,

$$|\ell_U(v)| \leq \left(\left\| \vec{f} \right\|_{\boldsymbol{L}^2(\Omega)} + \|\gamma_0\| \left\| \vec{t}_N \right\|_{\boldsymbol{L}^2(\partial\Omega_N)} \right) \|\vec{v}\|_{\boldsymbol{H}^1(\Omega)} + (3\lambda_+ + 2\mu_+)\alpha_+ \|T\|_{L^2(\Omega)} \|\boldsymbol{\nabla} \cdot \vec{v}\|_{L^2(\Omega)}$$
$$\leq \left(\left\| \vec{f} \right\|_{\boldsymbol{L}^2(\Omega)} + \|\gamma_0\| \left\| \vec{t}_N \right\|_{\boldsymbol{L}^2(\partial\Omega_N)} \right) \|\vec{v}\|_{\boldsymbol{H}^1(\Omega)} + 3(3\lambda_+ + 2\mu_+)\alpha_+ \|T\|_{L^2(\Omega)} \|\vec{v}\|_{\boldsymbol{H}^1(\Omega)}.$$
$$|\ell_U(v)| \leq \left(\left\| \vec{f} \right\|_{\boldsymbol{L}^2(\Omega)} + \|\gamma_0\| \left\| \vec{t}_N \right\|_{\boldsymbol{L}^2(\partial\Omega_N)} + 3(3\lambda_+ + 2\mu_+)\alpha_+ \|T\|_{L^2(\Omega)} \right) \|\vec{v}\|_{\boldsymbol{H}^1(\Omega)}.$$

- $\forall \vec{u}, \vec{v} \in \boldsymbol{H}^1_{\vec{0},D}(\Omega)$,

$$|a(\vec{u},\vec{v})| \leq (3\lambda_+ + 2\mu_+) \|\vec{u}\|_{\boldsymbol{H}^1(\Omega)} \|\vec{v}\|_{\boldsymbol{H}^1(\Omega)}$$

- $\forall \vec{u} \in \boldsymbol{H}^1_{\vec{0},D}(\Omega)$,

$$|a(\vec{u},\vec{u})| \geq 2\mu_- C \|\vec{u}\|^2_{\boldsymbol{H}^1(\Omega)}$$

With $C > 0$ a constant from Korn's inequality.

3.2.4 Numerical simulation

Denote by $V^h \subset H^1_{0,\text{heat}}(\Omega)$ a finite-dimensional subspace of test functions for temperature, $\boldsymbol{V}^h \subset \boldsymbol{H}^1_{\vec{0},D}(\Omega)$ a finite-dimensional subspace of test functions for displacement. d is the degree of the B-spline spaces used to approximate the solution. Let's suppose that $(T,\vec{u}) \in H^{s+1}(\Omega) \times \boldsymbol{H}^{s+1}(\Omega)$

for some $s \in [1, d+1]$ is solving (3.2.3) and let's denote the corresponding approximations by $(T^h, \vec{u}^h) \in V^h \times \boldsymbol{V}^h$. The best approximation estimates for the B-spline spaces V^h and $\boldsymbol{V}^h$ are:

$$\begin{aligned} \|T - T^h\|_{H^1(\Omega)} &\leq C_1 h^q \|T\|_{H^{q+1}(\Omega)}, \\ \|T - T^h\|_{L^2(\Omega)} &\leq C_2 h^{q+1} \|T\|_{H^{q+1}(\Omega)}, \\ \|\vec{u} - \vec{u}^h\|_{\boldsymbol{H}^1(\Omega)} &\leq C_3 h^q \|\vec{u}\|_{\boldsymbol{H}^{q+1}(\Omega)}, \\ \|\vec{u} - \vec{u}^h\|_{\boldsymbol{L}^2(\Omega)} &\leq C_4 h^{q+1} \|\vec{u}\|_{\boldsymbol{H}^{q+1}(\Omega)}. \end{aligned} \tag{3.14}$$

where $C_1, C_2, C_3, C_4 > 0$ are constants independent of h, and $q = \min(d, s)$ [DV+14]. To evaluate the accuracy of the numerical solution, we can use these estimates to bound the error between the exact solution and the numerical approximation and compare it with the numerical results obtained from the simulation. Again, we use the manufactured solutions approach to evaluate the performance of the numerical method.

For that purpose, we can consider T_e and $\vec{u}_e$ defined as:

$$T_e = \cos\left(\frac{\pi x}{22}\right)\cos\left(\frac{\pi y}{22}\right) \qquad \vec{u}_e = \begin{pmatrix} 0 \\ \cos\left(\frac{\pi x}{22}\right)\cos\left(\frac{\pi y}{22}\right) \end{pmatrix}$$

and the corresponding source terms:

$$\begin{aligned} Q &= -\boldsymbol{\nabla} \cdot (\kappa(y) \boldsymbol{\nabla} T_e), \qquad & q_S &= -\kappa(y) \boldsymbol{\nabla} T_e \cdot \vec{n}, \\ \vec{f} &= -\boldsymbol{\nabla} \cdot \boldsymbol{\sigma}(\vec{u}_e), \qquad & \vec{t}_N &= \boldsymbol{\sigma}(\vec{u}_e) \cdot \vec{n}. \end{aligned}$$

The first result I obtained was the following plot:

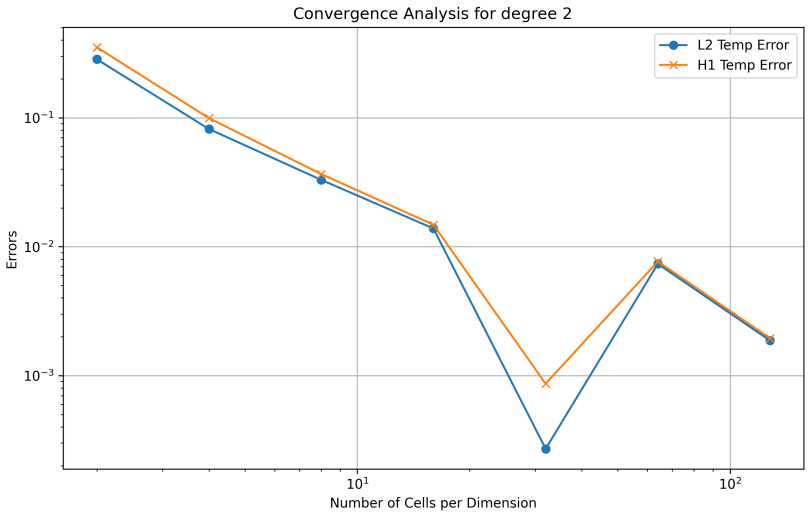


Figure 3.3: Error plot of the temperature field ($\|T - T^h\|_{L^2(\Omega)}$ and $\|T - T^h\|_{H^1(\Omega)}$) with $d = 2$ B-splines and lim = 17.0

In this figure, we can observe that the error in the temperature field is decreasing with the number of cells n_{cells} until $n_{\text{cells}} = 32$. After that, the error is increasing and decreases again. This behavior is likely due to the interaction between the mesh refinement, the B-spline approximation properties and the properties of the medium. When the degree of the B-splines is set to 3, we can see in figure 3.4 that the hollow moves to a lower number of cells.

The same phenomenon is observed for the displacement field. One reason that can explain this phenomenon is the placement of the B-spline knots and the corresponding basis functions. As the mesh is refined, the B-spline basis functions become more localized, which can lead to better

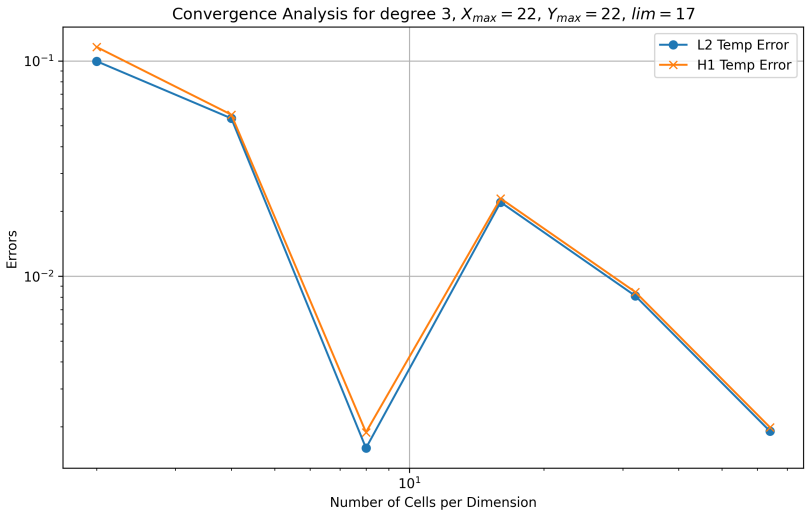
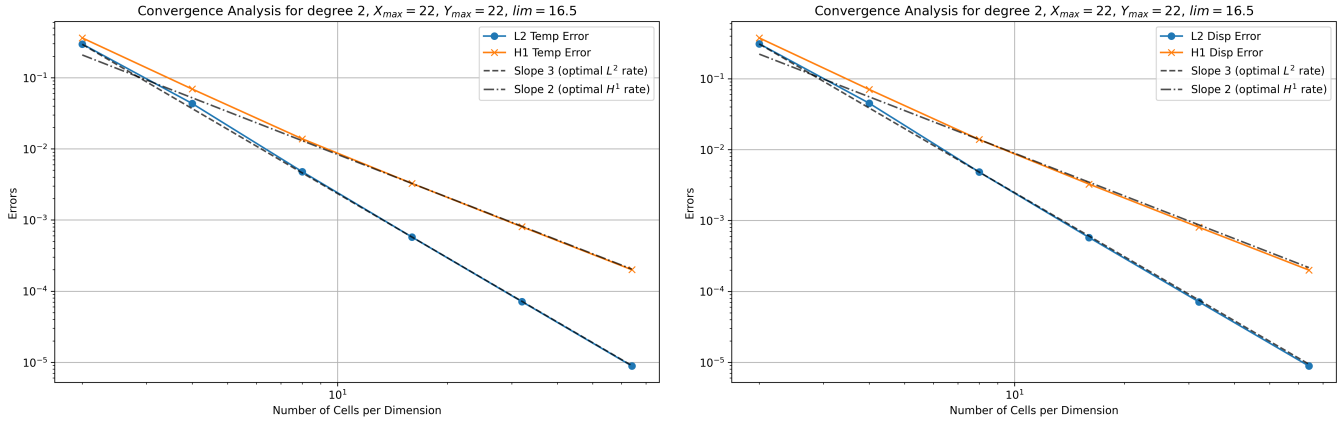


Figure 3.4: Error plot of the temperature field ($\|T - T^h\|_{L^2(\Omega)}$ and $\|T - T^h\|_{H^1(\Omega)}$) with $d = 3$ B-splines and lim = 17.0

approximation properties in certain regions of the domain. In the medium that we are considering, the line $y = \text{lim} = 17.0$ plays a crucial role in the behavior of the solution because it corresponds to a significant change in the material properties. Moreover, `PSYDAC` cannot handle coefficients defined piecewise so it is necessary to use an approximation of Heaviside functions to define coefficients. For example, $\lambda(y)$ is defined by:

$$\lambda(y) = \Big((\lambda_W - \lambda_{Cu})\frac{x}{\text{lim}} + \lambda_{Cu}\Big)\text{H}(\text{lim} - x) + \lambda_W \text{H}(x - \text{lim}) \qquad H \text{ is the Heaviside function.}$$

One hypothesis that can be made to verify the role of the line $y = \text{lim}$ is to move its value up and down and observe the effect on the solution. Especially, it's possible to move the value of lim to 16.5. This value corresponds to lines with breakpoints when the domain Ω is discretized into different cells.



(a) Error plot of the temperature field (b) Error plot of the displacement field

Figure 3.5: Error plots with lim = 16.5 and degree 2 B-splines

In this case, we can observe the optimal convergence behavior of the error as predicted by the theory. So the position of the line $y = \text{lim}$ is crucial for the accuracy of the solution and the simulation. To make the simulation possible in the case described in subsection 3.2.1, we need to get a simulation setup that solves the problem when lim = 17. For the moment, the function which discretizes the domain and creates the finite dimensional spline space takes only the number of cells

per direction as argument. I have implemented a new feature to the library which allows the user to provide a custom discretization grid. This feature required adding new functions to the `Geometry` class and modifying the previous `discretize` function in `PSYDAC`. To ensure the correct working of this new feature, I have also implemented a set of unit tests. This new feature allows the user to

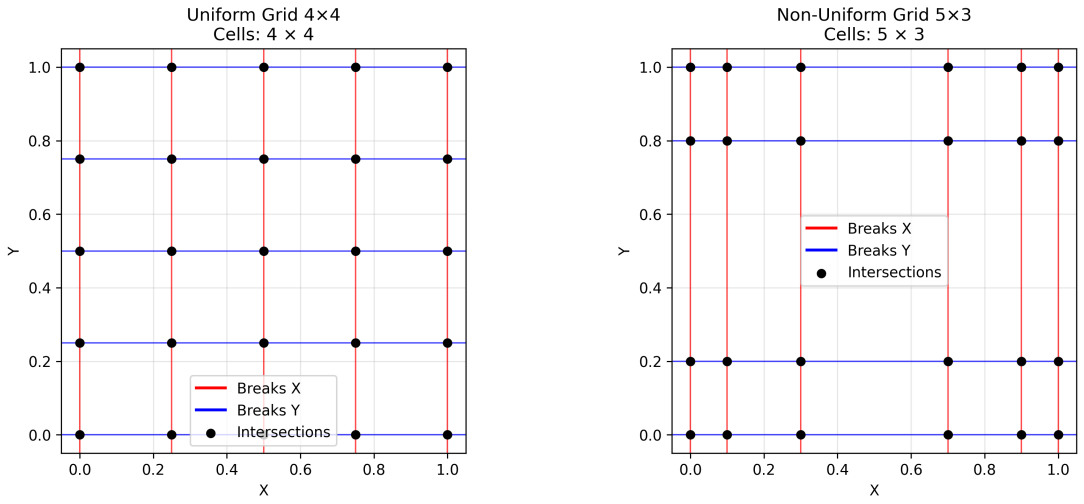
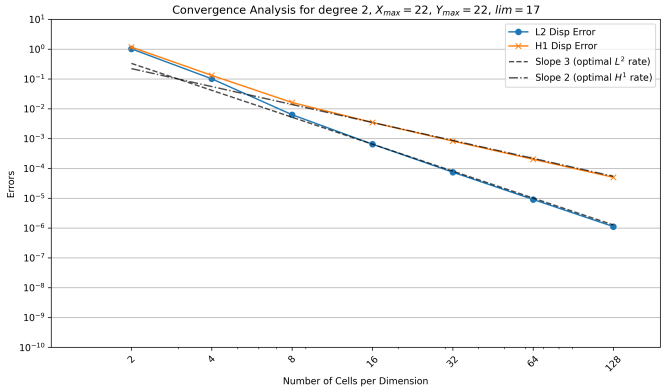


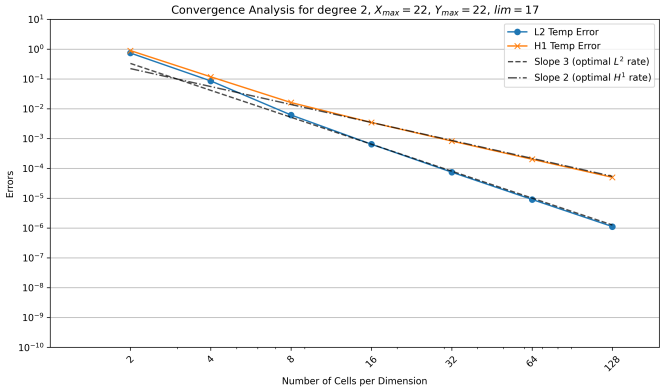
Figure 3.6: Example of a custom discretization grid

define a custom grid for the discretization of the domain. A complete description of this feature is available on GitHub: https://github.com/pyccel/psydac/pull/518

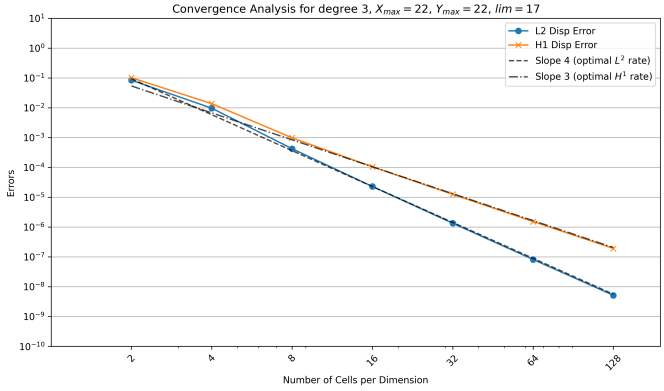
With this feature, we can now simulate correctly the problem when lim = 17 as it was described in the simplified problem setup. The results are shown in the figures below.



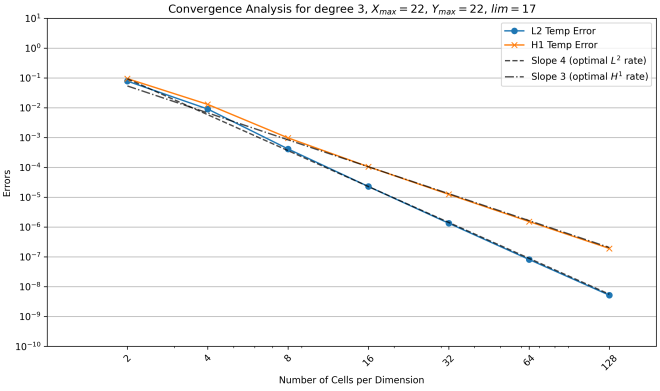
(a) Displacement field for degree 2 B-splines



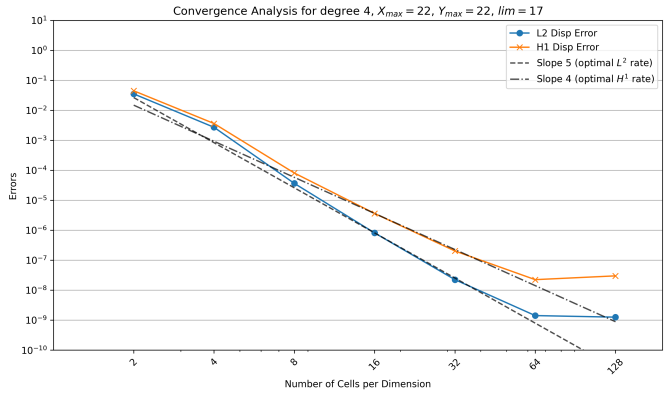
(b) Temperature field for degree 2 B-splines



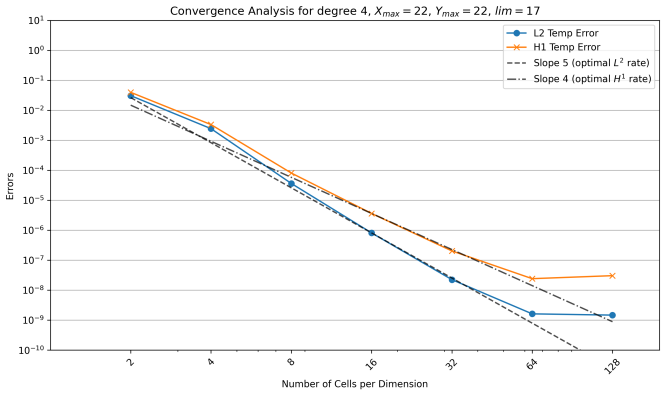
(c) Displacement field for degree 3 B-splines



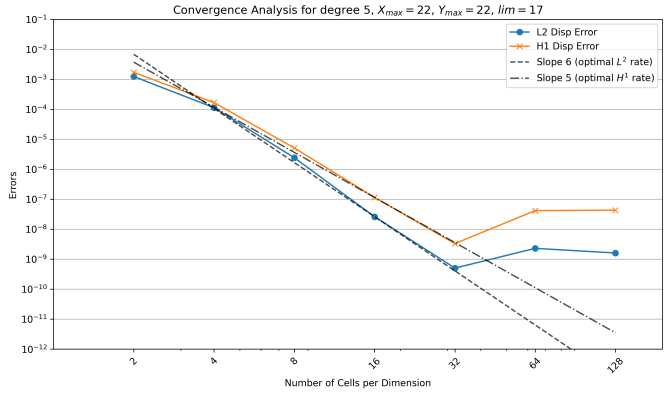
(d) Temperature field for degree 3 B-splines



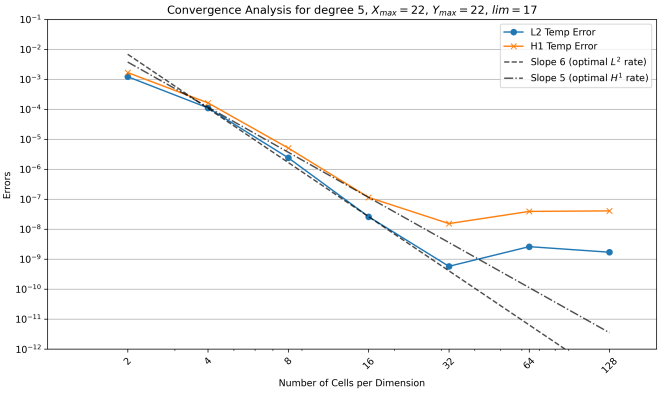
(e) Displacement field for degree 4 B-splines



(f) Temperature field for degree 4 B-splines



(g) Displacement field for degree 5 B-splines



(h) Temperature field for degree 5 B-splines

Figure 3.7: Error plots of thermoelasticity with different degree B-splines using custom grid discretization with lim = 17

From the results presented in the previous figures, we can observe that the error curves validate the theoretical estimates established in equation (3.14). The convergence behavior follows the expected order for both the temperature and displacement fields across all tested degrees of B-splines. The numerical solution demonstrates optimal convergence rates consistent with the theoretical predictions for smooth solutions.

However, we can identify the plateau phenomenon when the error reaches certain threshold values, specifically around 10^{-9} for the L^2 norm and between 10^{-7} and 10^{-8} for the H^1 norm. This behavior mirrors exactly what was observed in Chapter 2, suggesting that the underlying causes remain the same. The plateau effect is primarily attributed to the ill-conditioning of the system matrix, which becomes more pronounced as the mesh is refined and the degree of B-splines increases.

The error magnitudes are nearly identical for both the displacement and temperature fields. This similarity can be explained by the structure of our manufactured solutions: the exact temperature field T_e is identical to the second component of the displacement field $\vec{u}_e$, while the first component of the displacement field is zero and T_e appears in the source term when the displacement problem is solved. This deliberate choice in our test case creates a natural correspondence between the two fields, resulting in comparable approximation errors for both problems.

The successful implementation of the custom grid discretization feature has proven essential for handling the thermoelasticity problem with material property discontinuities at $y = 17$. This enhancement to `PSYDAC` enables accurate simulations in scenarios where material interfaces do not align with uniform grid boundaries, which is crucial for realistic engineering applications involving composite materials. It could be possible to use another main functionality of `PSYDAC`: calculation on a multipatch domain. This approach will be tackled in the next section.

3.3 Initial problem - Solution on a multipatch domain

In this section, we will consider the initial problem described in figure 3.1. To model this medium and simulate the thermoelastic behavior, we will use the multipatch framework provided by `PSYDAC`.

3.3.1 Quick overview of the multipatch framework

This section presents a quick overview of the multipatch framework in `PSYDAC`. For a more detailed description of the multipatch framework, please refer t [GHR22] and [GHP23]

The multipatch framework in `PSYDAC` allows for the simulation of problems defined on non-conforming grids, which is particularly useful for complex geometries and material interfaces. This framework enables the coupling of different patches, each potentially using different discretization strategies of mapping.

In `Psydac`, the *multipatch framework* is designed to represent complex physical domains by decomposing them into a collection of simpler *patches*. Each patch is the image of a logical domain (typically $[0,1]^d$) under a smooth mapping F_k. This approach allows the exact representation of curved geometries while maintaining a structured, tensor-product B-spline discretization within each patch.

Mathematically, each patch Ω_k supports a local finite element space $V_h^\ell(\Omega_k)$, built from tensor-product B-splines or NURBS of chosen degree and regularity. These local spaces form a discrete de Rham sequence

$$V_h^0(\Omega_k) \xrightarrow{\nabla} V_h^1(\Omega_k) \xrightarrow{\nabla\times} V_h^2(\Omega_k) \xrightarrow{\nabla\cdot} V_h^3(\Omega_k),$$

ensuring that key identities such as $\nabla \times (\nabla\varphi) = 0$ and $\nabla \cdot (\nabla \times \mathbf{u}) = 0$ hold at the discrete level.

The domain Ω is the union $\Omega = \bigcup_{k=1}^{K} \Omega_k$, and `Psydac` constructs the *broken space*

$$V_h^\ell := \left\{ v \in L^2(\Omega) \;\middle|\; v|_{\Omega_k} \in V_h^\ell(\Omega_k) \right\},$$

which does not enforce conformity across patch interfaces. This choice yields block-diagonal mass matrices (one block per patch), enabling local L^2 projections and efficient parallelization. To recover the necessary continuity when required, `Psydac` applies *conforming projection operators* P_h^ℓ that average interface degrees of freedom, defining discrete operators

$$d_h^\ell = d^\ell \, P_h^\ell,$$

where d^ℓ is the continuous differential operator. This guarantees the *commuting diagram property* of Finite Element Exterior Calculus (FEEC), thus preserving stability, exactness of sequences, and the structure of discrete Hodge–Helmholtz decompositions.

This multipatch combined with the broken-FEEC approach provides:

1. **Geometric flexibility**: each patch can have its own parametrization, allowing the exact modeling of complex CAD geometries;

2. **Computational locality**: operators and projections involve only neighboring patches, enhancing scalability for large-scale parallel simulations.

For example, `PSYDAC` can reproduce a Pretzel geometry by defining multiple patches that conform to the intricate shape of the object. Each patch can be individually parameterized to capture the local geometry accurately, while the multipatch framework ensures that the overall simulation remains consistent and stable. Then, it's possible to solve Poisson's equation or Time-dependent Maxwell equations on a Pretzel geometry.

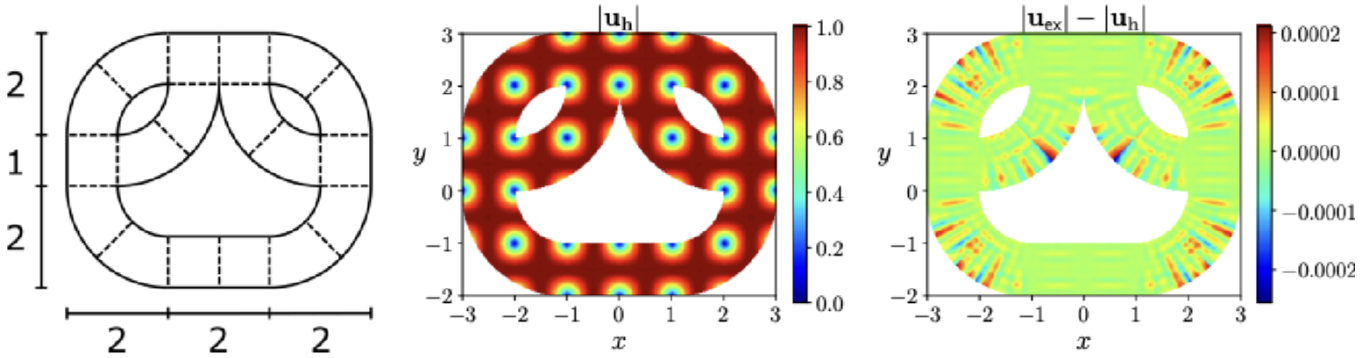


Figure 3.8: Time harmonic Maxwell's equation on a pretzel domain with inhomogeneous essential boundary conditions. Multi-patch domain (left), numerical amplitude $|u_h|$ (center) and amplitude error $|u_{ex}| - |u_h|$ (right) obtained with polynomial degree $p = 3$ and $N = 1$ from [GHR22]

In this context, the multipatch framework is particularly advantageous for handling the thermoelasticity problem with this unusual geometry.

3.3.2 A simple multipatch: square hole

To get familiar with the multipatch framework, we start with a simple example: a square domain with a square hole in the center. This geometry can be easily represented using a multipatch approach with 8 square patches. Actually, the domain can be described with fewer patches but to define correctly the boundary conditions, we need to define the patches in a way that allows us to apply the Dirichlet boundary conditions on the entire boundary of the domain.

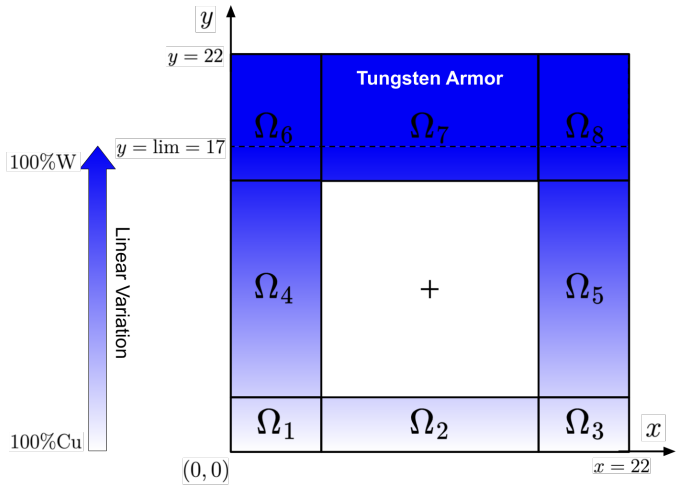


Figure 3.9: Multipatch representation of a square domain with a square hole in the center.

In this case, the hole is represented by a square $[5, 17] \times [3, 15]$ and is surrounded by 8 patches. Each patch is obtained by applying a transformation to the reference square $[0, 1]^2$. The patches are defined as follows: $\forall(\hat{x}, \hat{y}) \in [0, 1]^2$,

- Patch 1: $\Omega_1 := [0, 5] \times [0, 3]$ with $F_1(\hat{x}, \hat{y}) = (5\hat{x}, 3\hat{y})$
- Patch 2: $\Omega_2 := [5, 17] \times [0, 3]$ with $F_2(\hat{x}, \hat{y}) = (12\hat{x} + 5, 3\hat{y})$
- Patch 3: $\Omega_3 := [17, 22] \times [0, 3]$ with $F_3(\hat{x}, \hat{y}) = (5\hat{x} + 17, 3\hat{y})$
- Patch 4: $\Omega_4 := [0, 5] \times [3, 15]$ with $F_4(\hat{x}, \hat{y}) = (5\hat{x}, 12\hat{y} + 3)$

- Patch 5: $\Omega_5 := [17, 22] \times [3, 15]$ with $F_5(\hat{x}, \hat{y}) = (5\hat{x} + 17, 12\hat{y} + 3)$
- Patch 6: $\Omega_6 := [0, 5] \times [15, 22]$ with $F_6(\hat{x}, \hat{y}) = (5\hat{x}, 7\hat{y} + 15)$
- Patch 7: $\Omega_7 := [5, 17] \times [15, 22]$ with $F_7(\hat{x}, \hat{y}) = (12\hat{x} + 5, 7\hat{y} + 15)$
- Patch 8: $\Omega_8 := [17, 22] \times [15, 22]$ with $F_8(\hat{x}, \hat{y}) = (5\hat{x} + 17, 7\hat{y} + 15)$

The physical domain is then defined as the union of these patches:

$$\Omega = \bigcup_{i=1}^{8} \Omega_i = ([0, 22] \times [0, 22]) \setminus ([5, 17] \times [3, 15])$$

. The code to define this multipatch domain in `PSYDAC` is given in Section 6.2.

The use of a multipatch domain needs to ensure that the patches are properly aligned and that the transformations between them are smooth. Moreover, bilinear forms must be defined consistently across patch boundaries to ensure the stability and accuracy of the numerical methods employed. To do so, we call Σ_{ij} the interface between patches Ω_i and Ω_j, such that $\Sigma_{ij} = \partial\Omega_i \cap \partial\Omega_j$.

Then, bilinear forms are penalized across the interfaces to ensure continuity and stability: this is called Nitsche's method. This method consists of adding a penalty term to the bilinear forms that involves the traces of the functions on the interface Σ_{ij}, a symmetrization term and a penalty term that ensures the continuity of the traces across the interface [FHW04] [HHL03]. Consequently, the bilinear forms are redefined as follows:

$$a(\vec{u}, \vec{v}) = a_\Omega(\vec{u}, \vec{v}) + \sum_{i,j} a_{ij}(\vec{u}, \vec{v})$$
$$b(T, \Theta) = b_\Omega(T, \Theta) + \sum_{i,j} b_{ij}(T, \Theta)$$

With:

$$a_\Omega : \begin{cases} \boldsymbol{H}^1_{\vec{0},D}(\Omega) \times \boldsymbol{H}^1_{\vec{0},D}(\Omega) \longrightarrow \mathbb{R} \\ (\vec{u}, \vec{v}) \mapsto \displaystyle\int_\Omega \Big(\lambda(y)(\boldsymbol{\nabla} \cdot \vec{u})(\boldsymbol{\nabla} \cdot \vec{v}) + 2\mu(y)\boldsymbol{\varepsilon}(\vec{u}) : \boldsymbol{\varepsilon}(\vec{v}) \Big)\, d\Omega \end{cases}$$

$$a_{ij} : \begin{cases} \boldsymbol{H}^1_{\vec{0},D}(\Omega) \longrightarrow \mathbb{R} \\ (\vec{u}, \vec{v}) \mapsto \displaystyle\int_{\Sigma_{ij}} -\{\boldsymbol{\sigma}(\vec{u}) \cdot \vec{n}\} \cdot [[\vec{v}]] - \{\boldsymbol{\sigma}(\vec{v}) \cdot \vec{n}\} \cdot [[\vec{u}]] + \gamma_u \frac{\{2\mu + \lambda\}}{h} [[\vec{u}]] \cdot [[\vec{v}]]\, d\Gamma \end{cases}$$

$$b_\Omega : \begin{cases} H^1_{0,\text{temp}}(\Omega) \times H^1_{0,\text{temp}}(\Omega) \longrightarrow \mathbb{R} \\ (T, \theta) \mapsto \displaystyle\int_\Omega \kappa(y) \boldsymbol{\nabla} T \cdot \boldsymbol{\nabla}\theta\, d\Omega \end{cases}$$

$$b_{ij} : \begin{cases} H^1_{0,\text{temp}}(\Omega) \times H^1_{0,\text{temp}}(\Omega) \longrightarrow \mathbb{R} \\ (T, \theta) \mapsto \displaystyle\int_{\Sigma_{ij}} -\{\kappa(y)\boldsymbol{\nabla} T \cdot \vec{n}\} [[\theta]] - \{\kappa(y)\boldsymbol{\nabla}\theta \cdot \vec{n}\} [[T]] + \gamma_T \frac{\{\kappa\}}{h} [[T]] [[\theta]]\, d\Gamma \end{cases}$$

Where $\{\cdot\}$ denotes the average across the interface Σ_{ij}, $[[\cdot]]$ denotes the jump and $\gamma_u, \gamma_T > 0$ are penalty parameters.

With these modifications, the complete weak formulation for the thermoelasticity problem on the multipatch domain becomes:

Find $(T, \vec{u}) \in H^1(\Omega) \times \boldsymbol{H}^1(\Omega)$ such that for all test functions $(\theta, \vec{v}) \in H^1(\Omega) \times \boldsymbol{H}^1(\Omega)$:

$$\begin{cases} b_\Omega(T, \theta) + \sum_{i \neq j} b_{ij}(T, \theta) = l_T(\theta) \\ a_\Omega(\vec{u}, \vec{v}) + \sum_{i \neq j} a_{ij}(\vec{u}, \vec{v}) = l_U(\vec{v}) \end{cases} \tag{3.15}$$

The penalty parameters γ_u and γ_T must be chosen sufficiently large to ensure stability while maintaining accuracy. Typically, values in the range $[10, 100]$ work well in practice, though the optimal choice may depend on the specific problem parameters and mesh size.

This multipatch formulation with Nitsche's method allows for flexible handling of complex geometries while maintaining the stability and convergence properties of the underlying finite element discretization. The approach is particularly well-suited for problems involving material interfaces, complex boundaries, or domains that cannot be easily represented by a single structured mesh.

To test the convergence of the simulation on the multipatch domain, we use the same exact functions T_e and $\vec{u}_e$ as defined in the previous section:

$$T_e = \sin\left(\frac{\pi x}{22}\right) \sin\left(\frac{\pi y}{22}\right) \qquad \vec{u}_e = \begin{pmatrix} 0 \\ \sin\left(\frac{\pi x}{22}\right) \sin\left(\frac{\pi y}{22}\right) \end{pmatrix}$$

For the multipatch convergence test, we impose Dirichlet boundary conditions on all boundaries of the global domain $\partial\Omega$. This means that both the temperature and displacement fields are prescribed on the entire outer boundary of the domain (excluding the hole). The boundary conditions are therefore:

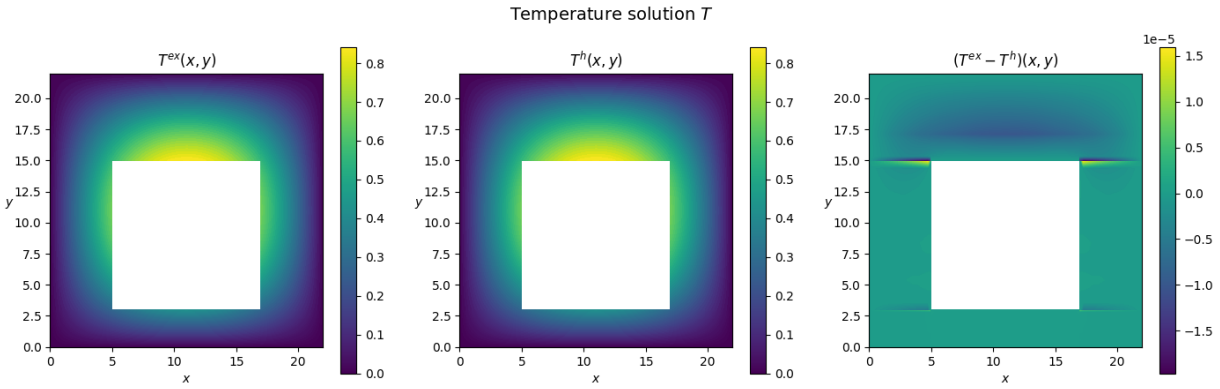
$$T = T_e \qquad \text{on } \partial\Omega \tag{3.16}$$

$$\vec{u} = \vec{u}_e \qquad \text{on } \partial\Omega \tag{3.17}$$

With these boundary conditions, the corresponding source terms are computed as:

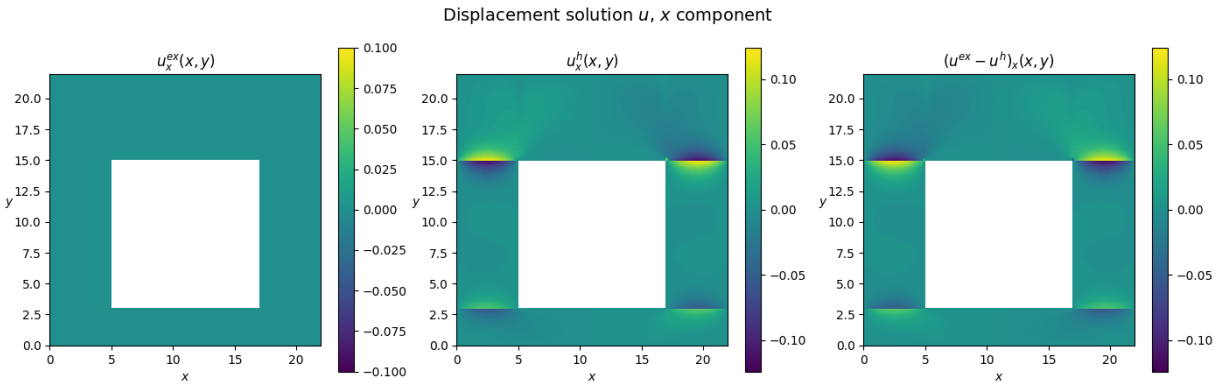
$$Q = -\boldsymbol{\nabla} \cdot (\kappa(y) \boldsymbol{\nabla} T_e),$$
$$\vec{f} = -\boldsymbol{\nabla} \cdot \boldsymbol{\sigma}(\vec{u}_e)$$

This test configuration allows us to verify that the multipatch implementation correctly handles the coupling between patches while maintaining the expected convergence rates for both the temperature and displacement fields.

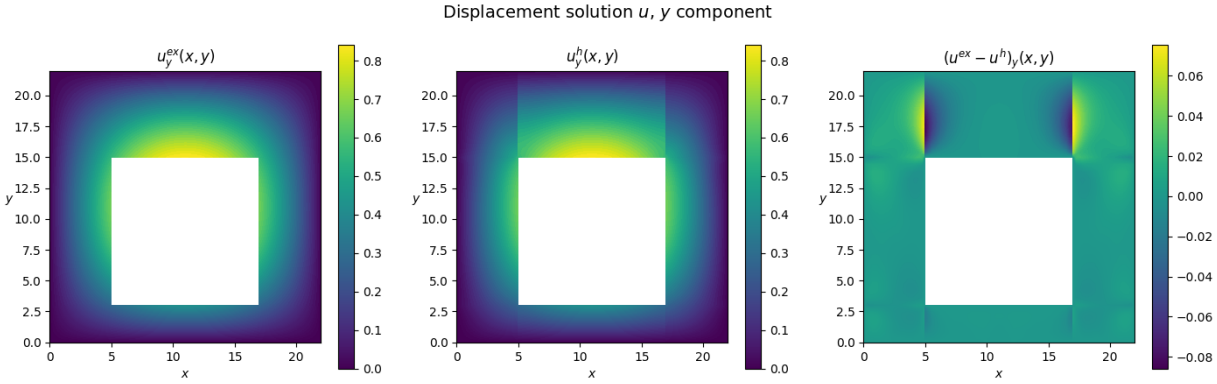


(a) Temperature field: T_e (left) - T_h (center) - Error (right)

Unfortunately, my implementation of the multipatch method is not yet complete. The simulation is running, but the displacement problem doesn't converge as expected. The displacement numerical



(b) Displacement field on x-axis: $u_{e,x}$ (left) - $u_{h,x}$ (center) - Error (right)

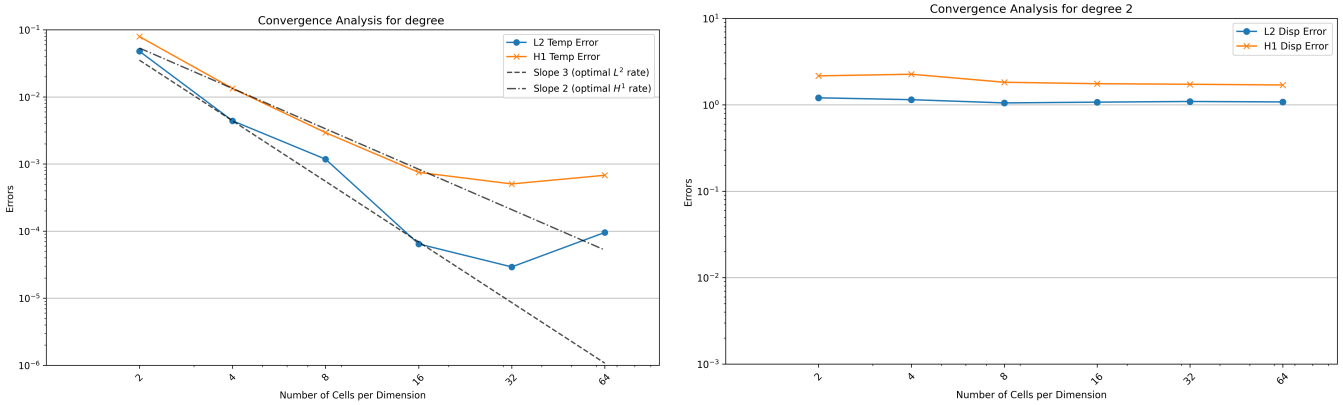


(c) Displacement field on y-axis: $u_{e,y}$ (left) - $u_{h,y}$ (center) - Error (right)

Figure 3.10: Simulation plots on a multipatch domain with $n_{cells} = 64$ per direction and domain with degree $p = 2$ B-Splines

solution seems to have some continuity problems between interfaces, while the thermal simulation seems to be working correctly.

As we can see in figures 3.10b and 3.10c, the displacement field on the x-axis is not continuous across the interfaces, which leads to non-convergence of the numerical solution. The error is not decreasing as expected, and the solution is not smooth across the interfaces. This issue is likely due to the Nitsche's method implementation or the way the bilinear forms are defined across the interfaces. Even by changing the values of constant γ_u to different values, the solution is not converging as expected...



(a) Error on the temperature field

(b) Error on the displacement field

Figure 3.11: Error plots on a multipatch domain with degree $p = 2$ B-Splines

Looking at the error plots across the number of cells, the error on the displacement field seems to be constant, while the error on the temperature field is decreasing. To a certain extent, the error on temperature is decreasing as expected... One can suppose that the Nitsche's method is not enforcing the continuity of the displacement field across the interfaces properly, leading to the observed constant error. However, this problem needs further investigation and potentially a different approach to ensure the continuity of the displacement field across the interfaces.

3.4 Conclusion

This chapter has explored the application of thermoelasticity problems within the `PSYDAC` framework, demonstrating the library's capability to handle coupled thermo-mechanical systems with varying material properties. The study encompassed two main configurations: a simplified square domain and a more complex geometry with a hole requiring multipatch treatment.

The simplified problem provided valuable insights into the behavior of coupled thermoelasticity systems under thermal and mechanical loads. The theoretical foundation, based on the stationary coupling between heat conduction and linear elasticity equations, was successfully implemented and validated using manufactured solutions. The decoupled approach, where temperature is solved first and then used as a source term in the mechanical problem, proved effective for the material parameters considered.

A significant achievement of this work was the development and implementation of custom grid discretization capabilities in `PSYDAC`. The custom grid feature enables practitioners to strategically place mesh boundaries at material interfaces, ensuring that the B-spline basis functions properly capture the discontinuous nature of material properties. This capability is essential for realistic engineering applications involving composite materials, layered structures, or any scenario where material properties vary abruptly within the computational domain.

The exploration of multipatch methods for complex geometries represents an ambitious extension of the work, though it revealed implementation challenges that require further development. While the thermal problem showed encouraging results on the multipatch domain, the displacement field exhibited continuity issues across patch interfaces. This suggests that the Nitsche method implementation for the elasticity equations needs a closer investigation to properly enforce interface conditions for vector-valued problems.

Chapter 4

Conclusion

This research internship at the Max Planck Institute for Plasma Physics has been a great experience that successfully achieved its primary objectives while providing invaluable insights into the computational mechanics domain. The work has demonstrated that `PSYDAC`, originally designed for plasma physics applications, can be effectively extended to solve solid mechanics problems, particularly linear elasticity and thermoelasticity equations.

Scientific Achievements: This internship extended `PSYDAC` from plasma physics to solid mechanics, implementing two formulations for linear elasticity, developing custom grid discretization for thermoelasticity problems with material discontinuities, and exploring multipatch methods for complex geometries. The work validated theoretical convergence rates, identified volumetric locking limitations in incompressible materials, and contributed essential capabilities for handling composite materials and coupled multiphysics simulations within the IGA framework.

Methodological Insights and Personal Growth: This internship provided me with a profound understanding of the research methodology, particularly the systematic decomposition of complex problems into manageable steps. Working with `PSYDAC` on the Raven supercomputer taught me that effective research progress comes through patient iteration and careful validation. The experience enhanced my computational skills while developing a holistic perspective on the interplay between theoretical foundations, practical implementation constraints, and high-performance computing resource management, insights that will prove invaluable for future scientific endeavors. I've also come to understand the high-performance tools at my disposal and how to leverage them effectively in my research to avoid excessive resource consumption.

Collaborative Outcomes and Future Prospects: This internship will contribute directly to a presentation at the International Conference on Isogeometric Analysis (IGA 2025) in the Netherlands in September 2025. Moreover, my work will be part of a collaborative publication with Dr. Yaman Güçlü.

Personal Reflection: Beyond the technical achievements, this internship has been profoundly enriching on a personal level. Working within an international research environment at IPP exposed me to diverse perspectives and collaborative approaches that broadened my understanding of scientific research culture. The experience of presenting complex technical concepts and contributing to open-source projects enhanced both my technical skills and professional confidence.

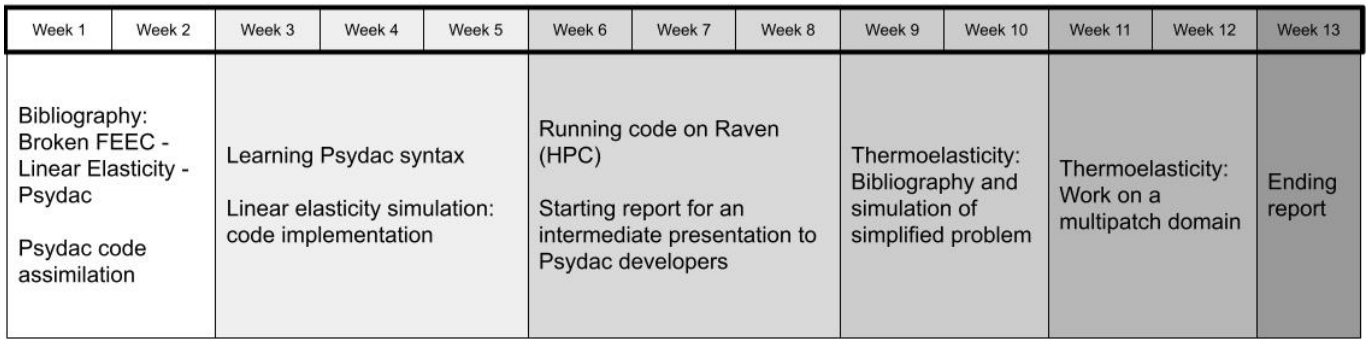


Figure 4.1: Internship Timeline

Chapter 5

References

[HHL03] Anita Hansbo, Peter Hansbo, and Mats G. Larson. "A Finite Element Method on Composite Grids Based on Nitsche's Method". In: *ESAIM: Mathematical Modelling and Numerical Analysis* 37.3 (2003). Received November 25, 2002; Revised March 6, 2003, pp. 495–514. DOI: 10.1051/m2an:2003039. URL: https://doi.org/10.1051/m2an:2003039.

[EG04] Alexandre Ern and Jean-Luc Guermond. *Theory and Practice of Finite Elements*. en. Ed. by S. S. Antman, J. E. Marsden, and L. Sirovich. Vol. 159. Applied Mathematical Sciences. New York, NY: Springer New York, 2004. ISBN: 978-1-4419-1918-2, 978-1-4757-4355-5. DOI: 10.1007/978-1-4757-4355-5. URL: http://link.springer.com/10.1007/978-1-4757-4355-5.

[FHW04] A. Fritz, S. Hüeber, and B. Wohlmuth. "A comparison of mortar and Nitsche techniques for linear elasticity". en. In: *Calcolo* 41 (2004). Received 01 May 2003; Accepted 01 June 2004, pp. 115–137. DOI: 10.1007/s10092-004-0087-4. URL: https://doi.org/10.1007/s10092-004-0087-4.

[Bof+08] Daniele Boffi et al. *Mixed Finite Elements, Compatibility Conditions, and Applications: Lectures given at the C.I.M.E. Summer School held in Cetraro, Italy, June 26–July 1, 2006*. en. Ed. by Daniele Boffi et al. Vol. 1939. Lecture Notes in Mathematics. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. ISBN: 978-3-540-78314-5, 978-3-540-78319-0. DOI: 10.1007/978-3-540-78319-0. URL: http://link.springer.com/10.1007/978-3-540-78319-0.

[BS08] Susanne C. Brenner and L. Ridgway Scott. *The Mathematical Theory of Finite Element Methods*. en. Ed. by J. E. Marsden, L. Sirovich, and S. S. Antman. Vol. 15. Texts in Applied Mathematics. New York, NY: Springer New York, 2008. ISBN: 978-0-387-75933-3, 978-0-387-75934-0. DOI: 10.1007/978-0-387-75934-0. URL: http://link.springer.com/10.1007/978-0-387-75934-0.

[Gou13] Phillip L. Gould. *Introduction to Linear Elasticity*. en. New York, NY: Springer New York, 2013. ISBN: 978-1-4614-4832-7, 978-1-4614-4833-4. DOI: 10.1007/978-1-4614-4833-4. URL: https://link.springer.com/10.1007/978-1-4614-4833-4.

[DV+14] L. Beirão Da Veiga et al. "Mathematical Analysis of Variational Isogeometric Methods". en. In: *Acta Numerica* 23 (2014), pp. 157–287. ISSN: 0962-4929, 1474-0508. DOI: 10.1017/S096249291400004X. URL: https://www.cambridge.org/core/product/identifier/S096249291400004X/type/journal_article.

[MS15] Arnd Meyer and Rolf Springer. *Basics of Linear Thermoelasticity*. en. Preprint 15-03. First published on 06.02.2015. Technische Universität Chemnitz, 2015. URL: https://nbn-resolving.org/urn:nbn:de:bsz:ch1-qucosa-160178.

[Per16] Anna Persson. “A Generalized Finite Element Method for Linear Thermoelasticity”. en. Licentiate Thesis. Gothenburg, Sweden: Chalmers University of Technology and University of Gothenburg, 2016.

[Cin18] Emma Cinatl. “Finite Element Discretizations for Linear Elasticity”. en. PhD thesis. Clemson University, 2018.

[GHR22] Y. Güçlü, S. Hadjout, and A. Ratnani. “PSYDAC: A High-Performance IGA Library in Python”. en. In: *8th European Congress on Computational Methods in Applied Sciences and Engineering*. CIMNE, 2022. DOI: 10.23967/eccomas.2022.227. URL: https://www.scipedia.com/public/Guclu_et_al_2022a.

[GHP23] Yaman Güçlü, Said Hadjout, and Martin Campos Pinto. “A Broken FEEC Framework for Electromagnetic Problems on Mapped Multipatch Domains”. In: *Journal of Scientific Computing* 97.52 (2023). Correction published on 10 November 2023. DOI: 10.1007/s10915-023-02351-x. URL: https://link.springer.com/article/10.1007/s10915-023-02351-x.

[Che24] Long Chen. “Variational Formulation of Linear Elasticity”. en. In: (2024).

Chapter 6

Appendices

6.1 Simulation code for linear elasticity with Dirichlet Homogeneous Boundary Conditions

```
from mpi4py import MPI
from sympde.expr import BilinearForm, LinearForm, integral
from sympde.expr     import find, EssentialBC, Norm, SemiNorm
from sympde.topology import VectorFunctionSpace, Cube, element_of, Union,
    NormalVector
from sympde.calculus import grad, inner, div, Transpose
from sympde.core import Constant
from sympde.core import Matrix

from sympy import Identity, Tuple, sin, pi, cos, diff

import matplotlib.pyplot as plt
import numpy as np
import time

from psydac.api.settings import PSYDAC_BACKENDS


from psydac.api.discretization import discretize
import os
import numpy as np
import matplotlib.pyplot as plt

def run_lin_elas_non_mixed(ncells, degree, lambda_val, mu_val, comm=None, backend
    =None, solver_tol=1e-8):

    #++++++++++++++++++++++++++++++++
    # 1. Abstract model
    #++++++++++++++++++++++++++++++++

    domain = Cube()

    timing   = {}
    t0 = time.time()

    domain = Cube()

    nn = NormalVector(domain.boundary)
```

```
lambda_ = Constant('lambda_', is_real=True, real=True)
mu   = Constant('mu',   is_real=True, real=True)

# Symbolic symmetric strain tensor
def epsilon_symb(w):
    return 0.5 * (grad(w) + Transpose(grad(w)))

# Symbolic stress tensor
def sigma_symb(w):
    return lambda_ * div(w) * Identity(3) + 2 * mu * epsilon_symb(w)

# Define the symmetric strain tensor (for to compute the source term)
def epsilon(w):
    grad_w = Matrix([
        [diff(w[0], x), diff(w[0], y), diff(w[0], z)],
        [diff(w[1], x), diff(w[1], y), diff(w[1], z)],
        [diff(w[2], x), diff(w[2], y), diff(w[2], z)]
    ])
    return 0.5 * (grad_w + grad_w.T)

# Define the stress tensor (to compute the source terms)
def sigma(w):
    eps = epsilon(w)
    div_w = diff(w[0], x) + diff(w[1], y) + diff(w[2], z)
    C_eps = lambda_ * div_w * Identity(3) + 2 * mu * eps
    return C_eps

V = VectorFunctionSpace('V', domain)

x,y,z = domain.coordinates

u,v = [element_of(V, name=i) for i in ['u', 'v']]

# Bilinear form
a = BilinearForm((u,v), integral(domain , inner(sigma_symb(u), epsilon_symb(v
))))


# Exact solution
ue1 = 0
ue2 = 0
ue3 = sin(pi*x)*sin(pi*y)*sin(pi*z)
ue = Tuple(ue1, ue2, ue3)

# Second member
sigma_ue = sigma(ue)
f1 = diff(sigma_ue[0, 0], x) + diff(sigma_ue[0, 1], y) + diff(sigma_ue[0, 2],
z)
f2 = diff(sigma_ue[1, 0], x) + diff(sigma_ue[1, 1], y) + diff(sigma_ue[1, 2],
z)
f3 = diff(sigma_ue[2, 0], x) + diff(sigma_ue[2, 1], y) + diff(sigma_ue[2, 2],
z)
f = Tuple(-f1, -f2, -f3)

# Linear forms
l = LinearForm(v, integral(domain, inner(f, v)))

# Dirichlet boundary conditions
bc = [EssentialBC(u, 0, domain.boundary)]
```

```
    # Variational problem
    equation = find(u, forall=v, lhs=a(u, v), rhs=l(v), bc=bc)

    error = Matrix([u[0]-ue[0], u[1]-ue[1], u[2]-ue[2]])

    l2norm = Norm(error, domain, kind='l2')
    h1norm = Norm(error, domain, kind='h1')
    h1_seminorm = SemiNorm(error, domain, kind='h1')

    t1 = time.time()
    timing["Abstract model"] = t1-t0

    #+++++++++++++++++++++++++++++++
    # 2. Discretization
    #+++++++++++++++++++++++++++++++

    t0 = time.time()
    domain_h = discretize(domain, ncells=ncells, comm=comm)
    Vh       = discretize(V, domain_h, degree=degree)

    equation_h = discretize(equation, domain_h, [Vh, Vh], backend=backend)

    l2norm_h = discretize(l2norm, domain_h, Vh, backend=backend)
    h1norm_h = discretize(h1norm, domain_h, Vh, backend=backend)
    h1_seminorm_h = discretize(h1_seminorm, domain_h, Vh, backend=backend)

    #+++++++++++++++++++++++++++++++
    # 3. Solver & Errors
    #+++++++++++++++++++++++++++++++

    equation_h.set_solver('gmres', info=True, tol=solver_tol)

    t1 = time.time()
    timing["Discretization"] = t1-t0

    comm.Barrier()
    t0       = time.time()
    uh, info = equation_h.solve(mu=mu_val, lambda_=lambda_val)
    t1       = time.time()
    timing['solution'] = t1-t0

    comm.Barrier()
    t0 = time.time()
    l2_error = l2norm_h.assemble(u=uh)
    h1_error = h1norm_h.assemble(u=uh)
    h1_seminorm_error = h1_seminorm_h.assemble(u=uh)
    t1       = time.time()

    timing['diagnostics'] = t1-t0
    timing["Abstract model"] = comm.reduce(timing["Abstract model"], op=MPI.MAX)
    timing["Discretization"] = comm.reduce(timing["Discretization"], op=MPI.MAX)
    timing['solution']       = comm.reduce(timing['solution'], op=MPI.MAX)
    timing['diagnostics']    = comm.reduce(timing['diagnostics'], op=MPI.MAX)

    return uh, info, timing, l2_error, h1_error, h1_seminorm_error
```

Listing 6.1: Simulation code for the linear elasticity problem with Dirichlet boundary conditions.

6.2 Multipatch domain decomposition with PSYDAC

```
from psydac.feec.multipatch.multipatch_domain_utilities import sympde_Domain_join
from sympde.topology import Square
from sympde.topology import IdentityMapping

def build_domain_hole(domain_name="domain"):
    OmegaLog1 = Square('OmegaLog1', bounds1=(0., 5), bounds2=(0., 3))
    mapping_1 = IdentityMapping('M1', 2)
    domain_1 = mapping_1(OmegaLog1)

    OmegaLog2 = Square('OmegaLog2', bounds1=(5, 17), bounds2=(0., 3))
    mapping_2 = IdentityMapping('M2', 2)
    domain_2 = mapping_2(OmegaLog2)

    OmegaLog3 = Square('OmegaLog3', bounds1=(17, 22), bounds2=(0., 3))
    mapping_3 = IdentityMapping('M3', 2)
    domain_3 = mapping_3(OmegaLog3)

    OmegaLog4 = Square('OmegaLog4', bounds1=(0, 5), bounds2=(3, 15))
    mapping_4 = IdentityMapping('M4', 2)
    domain_4 = mapping_4(OmegaLog4)

    OmegaLog6 = Square('OmegaLog6', bounds1=(17, 22), bounds2=(3, 15))
    mapping_6 = IdentityMapping('M6', 2)
    domain_6 = mapping_6(OmegaLog6)

    OmegaLog7 = Square('OmegaLog7', bounds1=(0, 5), bounds2=(15, 22))
    mapping_7 = IdentityMapping('M7', 2)
    domain_7 = mapping_7(OmegaLog7)

    OmegaLog8 = Square('OmegaLog8', bounds1=(5, 17), bounds2=(15, 22))
    mapping_8 = IdentityMapping('M8', 2)
    domain_8 = mapping_8(OmegaLog8)

    OmegaLog9 = Square('OmegaLog9', bounds1=(17, 22), bounds2=(15, 22))
    mapping_9 = IdentityMapping('M9', 2)
    domain_9 = mapping_9(OmegaLog9)

    # 7 8 9
    # 4 * 6
    # 1 2 3
    patches = [domain_1, domain_2, domain_3, domain_4, domain_6, domain_7,
   domain_8, domain_9]
    axis_0, axis_1 = 0, 1
    ext_0, ext_1 = -1, +1

    connectivity = [
        [(domain_1, axis_0, ext_1), (domain_2, axis_0, ext_0), 1],
        [(domain_2, axis_0, ext_1), (domain_3, axis_0, ext_0), 1],
        [(domain_7, axis_0, ext_1), (domain_8, axis_0, ext_0), 1],
        [(domain_8, axis_0, ext_1), (domain_9, axis_0, ext_0), 1],

        [(domain_1, axis_1, ext_1), (domain_4, axis_1, ext_0), 1],
        [(domain_4, axis_1, ext_1), (domain_7, axis_1, ext_0), 1],
        [(domain_3, axis_1, ext_1), (domain_6, axis_1, ext_0), 1],
        [(domain_6, axis_1, ext_1), (domain_9, axis_1, ext_0), 1],
    ]
    return sympde_Domain_join(patches, connectivity=connectivity, name=
   domain_name)
```

Listing 6.2: Multipatch domain decomposition with PSYDAC.