SISSA
Scuola
Internazionale
Superiore di
Studi Avanzati

Mathematics Area – PhD course in

Mathematical Analysis, Modelling, and Applications

Coupling methods for non-matching meshes through distributed Lagrange multipliers

Candidate:

Giovanni Alzetta

Advisor:

Prof. Luca Heltai

Academic Year 2018-19

Il presente lavoro costituisce la tesi presentata da Giovanni Alzetta, sotto la direzione del Prof. Luca Heltai, al fine di ottenere l'attestato di ricerca post-universitaria *Doctor Philosophiæ* presso la SISSA, Curriculum in Analisi Matematica, Modelli e Applicazioni. Ai sensi dell'art. 1, comma 4, dello Statuto della SISSA pubblicato sulla G.U. no. 36 del 13.02.2012, il predetto attestato è equipollente al titolo di *Dottore di Ricerca in Matematica*.

Trieste, Anno Accademico 2018/2019

In memoria di mia sorella Caterina.

Ringraziamenti

Desidero iniziare questa tesi, frutto del lavoro di tanti anni, con un piccolo gesto di riconoscimento per tutte le persone che mi hanno accompagnato; non riuscirò mai a ringraziare tutti, o ringraziare abbastanza, ma posso almeno fare un timido tentativo.

Ringrazio di cuore il Prof. Luca Heltai, che mi ha spinto a intraprendere la strada del dottorato, e mi ha guidato e supportato su questo lungo percorso: grazie per i consigli, la pazienza, le energie e gli insegnamenti di questi anni! Ringrazio sentitamente anche la Prof.ssa Lucia Gastaldi, controrelatrice di questa tesi, che ha fornito numerosi spunti, correzioni e preziosi suggerimenti sulla tesi ed il lavoro sui Fiber Reinforced Materials, aumentando la qualità del lavoro. Ringrazio anche tutti i professori e post-doc che mi hanno accompagnato, come il Prof. Giovanni Noselli per i suoi consigli in meccanica dei continui, il Dr. Nicola Giuliani, che mi ha sempre aiutato nei momenti difficili (computazionalmente parlando), il Dr. Alberto Sartori, per la sua disponibilità ed amicizia, e il Dr. Nicola Cavallini, capo della "truppa" del MHPC. Ringrazio anche il Dr. Alberto Maspero e il Dr. Francesco Ballarin, per i loro consigli e le piacevoli chiacchierate.

I would to thank the deal.II community for all the suggestions, time and energies they devoted to me, teaching me a lot about numerical analysis and programming. I want to thank, in particular, Prof. Daniel Arndt, Prof. Wolfgang Bangerth, Prof. Denis Davydov, Prof. Jean-Paul Pelteret, Prof. Timo Heister. and Prof. Bruno Turcksin.

I would also like to thank Dr. Wenyu Lei, for our short but fruitful collaboration.

Desidero ringraziare anche la SISSA e tutto il suo staff, soprattutto Riccardo Iancer: tante persone che hanno reso possibile il mio lavoro in questi anni, fornendo cibo e risolvendo problemi burocratici con disponibilità e gentilezza.

Un ringraziamento speciale va ai miei colleghi ed amici, soprattutto Maicol, che è sempre stato disponibile ad ascoltarmi per problemi di ogni natura (matematici e non), ed Emanuele, con cui ho condiviso gioie e dolori della vita di dottorato, oltre alle passeggiate post-prandiane nel parco della SISSA, che mi mancheranno moltissimo.

Desidero ringraziare anche tutti i ragazzi del mio anno di PhD: Ornela, Giulia, Noè, Zakia, Alaa. Ringrazio anche Giulio, con cui ho condiviso ormai sei anni di vita in SISSA, Matteo e Daniele. Ringrazio anche i miei vicini d'ufficio, Filippo, Flavia e Francesco, e tutte le altre persone che in questi anni hanno condiviso un pezzo della loro vita in SISSA con me, tra cui Giuseppe Puglisi, Carolina Biolo, Giulia Sormani, Elisa Paoli, ed Elena Tea Russo.

I would like to thank my friends from the MHPC master, with whom I shared a fruitful year of hardships: Anoop, Cristiano, Daniele, Paolo, James, Giangiacomo, Cosimo, Simone, Elliot, learning and studying with you was wonderful!

Desidero ringraziare tutti gli amici che mi hanno accompagnato in questi anni, chi dall'infanzia (grande Davide!), chi dalle superiori, come Sara e Fabrizio, chi dall'università

di Padova, come l'altro Davide, Sofia, Luigi, Marco, Andrey, Deniz, Paolo, chi dal mio arrivo a Trieste, come Alexander, Federica, Chiara, Federico e Maria.

Ringrazio tutti i miei ex coinquilini, che hanno dovuto sopportare le mie giornate storte: Giorgio, Marco, Mattia, Pierluigi, Andrea (Lattuada, Maioli e Battistoni), Laerte, ed Emanuele.

Desidero ringraziare le suorine di carità dell'Assunzione, soprattutto suor Carolina e suor Viviana, per la loro vicinanza e per l'avermi dato la possibilità di imparare tanto dai ragazzi del doposcuola.

Un grazie grandissimo va ai miei genitori, Michele e Gabriella, che mi hanno insegnato tanto e mi hanno sempre supportato, ai miei fratelli Giacomo, Pietro e Paolo, e a tutte le nonne, gli zii e i parenti lontani e vicini che mi accompagnano ogni giorno di questi quasi (sic!) trent'anni.

Infine ringrazio mia moglie Alice, che nonostante tutte le difficoltà mi è sempre stata vicina, ed i piccoli Elia e Daniele, che non finiscono di mostrarmi quanto sia incredibile e bello questo mondo.

Contents

Chapter 1

Introduction

Nature and engineering commonly present *multi-physics* problems, i.e., complex systems involving a number of mutually interacting subsystems that can be modelled by (non linearly coupled) Partial Differential Equations (PDEs). Examples range from fluid-structure interaction [52], to climate models [25], or thermomechanics [4]. The discretization of these systems of PDEs naturally leads to the use of multiple grids:

- Using different grids for independent problems joined at an interface, where the meshes need to be *coupled* and exchange information.
- Using different grids for different aspects of the physical problem on the same portion of space.

Whether it occurs in the bulk, or at the interface, coupling of different systems is a complex endeavour. From a modelling perspective, multiple, interacting components require additional effort. From a computational perspective, the need to accurately transfer information between meshes is an expensive operation. In addition, the implementation of this operation in a parallel, distributed environment poses a number of challenges connected with the lack of locality of the information.

When it is possible, the most natural way to model coupled system is through aligned interfaces and grids, implementing the coupling through boundary or transmission conditions. Although ideal, this situation is often impractical or computationally intractable.

In this work, we are interested in studying coupled problems through non-matching methods. Broadly speaking, non-matching methods are techniques that allow the coupling of different PDEs discretized on separate, independent, grids, or the solution to a PDE defined on a complex domain using a simpler domain (typically a structured grid). This strategy has been developed in a number of different methods and variations; here we report a simple description of some notable examples:

Penalty methods These methods use a "penalty" term to impose boundary conditions (see, e.g., [8]).

The general principle is summarized in [70] as follows: consider a minimization problem for the functional K over the domain B:

$$\min_{f \in W(B)} Kf,$$

where $W(B)$ is an opportune functional space. The penalty method can be thought of as the relaxation of the following problem. Given a domain Ω such that $\Omega \Subset B$, define the functional:

$$P_B(x) \coloneqq \begin{cases} 0 \text{ if } & x \in B \\ +\infty \text{ if } & x \in \Omega \setminus B \end{cases}.$$

We then consider the equivalent minimization problem:

$$\min_{f \in V(\Omega)} Kf + P_B,$$

where $V(\Omega)$ is an opportune functional space such that $W(B) \subset V(\Omega)$.

A simple implementation is obtained by considering a nonnegative function Ψ such that $B \coloneqq \{x \in \Omega \colon \Psi(x) = 0\}$, where B is the complex domain we are considering, embedded in the simpler domain Ω. Defining the following approximated functional:

$$K_\epsilon = K + \frac{1}{\epsilon}\Psi, \qquad \epsilon > 0,$$

we have the pointwise convergence:

$$\lim_{\epsilon \to 0} \frac{1}{\epsilon}\Psi(x) = P_B(x) \qquad \text{for every } x \in \Omega,$$

and we can expect that the minimizer u_ϵ of the approximated problem will approach the minimizer u of the original problem (assuming existence and uniqueness for the solution in both cases).

Composite finite element method This method was introduced in [42]: consider an immersed domain B into the bigger domain Ω. In this case, we modify the basis functions in the vicinity of the boundary to express the boundary conditions.

This strategy is suitable for complex geometries but requires an *ad-hoc* implementation of the finite element strategy.

XFem The eXtended Finite Element Method (XFEM) was originally developed for applications concerning fractures [15, 72], but has since been extended to several other applications for its ability to describe discontinuities, localized deformations, and complex geometries.

Unlike in the composite finite element method, this is not achieved through a modification of basis functions, but through a local enrichment of the approximation space, by adding to the finite elements space some extra functions that mimic the analytical behaviour of the solution.

The enrichment is based on the partition of unity concept: a global partition of unity on Ω, is a set of functions $\{\phi_i\}_{i=1}^M$ such that, for every $x \in \Omega$:

$$\sum_{i=1}^{M} \phi_i(x) = 1.$$

Let $\{u_j\}_{j=1}^N$ be the base of the standard finite element space; a generic function u_h of the enriched space is defined as:

$$u_h = \underbrace{\sum_{j=1}^{N} u_j c_j}_{\text{FEM}} + \underbrace{\sum_{i=1}^{M} \phi_i \Psi q_i}_{\text{enrichment}},$$

where c_j, q_i are the function coefficients, i.e., the unknowns describing the solution, and we call Ψ global enrichment function. The choice of Ψ depends on the sort of discontinuity we are expecting.

Immersed boundary methods Peskin introduced the immersed boundary (IB) method in the seventies, to simulate flow patterns in the heart. Since then, they have been applied successfully to a number of fluid-structure problems (see [80], and the references therein).

The IB method is both a mathematical formulation and a numerical scheme, that employs a mixture of Eulerian and Lagrangian variables; this idea is particularly effective in fluid-structure interaction problems, where the Eulerian framework, used to describe a fluid in terms of velocity and pressure fields, has to be coupled with the Lagrangian framework, used to describe the displacements of an elastic material.

The numerical scheme arising from the use of two frameworks for the mathematical formulation, employs a fixed Cartesian mesh for the Eulerian variables. The Lagrangian variables are then defined on a curvilinear mesh which can move freely through the fixed Cartesian mesh, without any constraint. A smooth approximation of the Dirac delta function is then used to pass information between the two frameworks.

Fictitious domain methods To solve an elliptic boundary value problem on a general domain B, these methods immerse it into a simpler domain Ω, the so-called *fictitious domain*, extending the right-hand side to Ω. This framework has two key advantages in constructing computational schemes [37]:

- The extended domain is geometrically simpler, admitting simpler meshes and, potentially, specialized fast solution methods, such as fast solvers for elliptic problems on rectangular domains.
- The extended domain might be time-independent, allowing the use of a fixed mesh even in time-dependent problems.

A Lagrange multiplier is then used to enforce the original boundary conditions, so that the solution to the extended problem matches the original one on B.

The extended problem is described by a symmetric saddle point problem, which can be studied using, for example, the well-known inf-sup conditions from Mixed Finite Elements [27]. Once the solution on the fictitious domain is found, it can be restricted to B, obtaining the solution to the original problem.

Even for smooth data, the solution to this saddle point problem is, in general, non smooth; this affects the convergence and optimality of the numerical methods used to study the problem. To solve these convergence problems, some adaptive refinement schemes have been proposed, see, e.g. [17, 16].

Fictitious-domain methods were first introduced by Hyman [57], and studied also by Saul'ev [93], who coined the term "fictitious domain". The pioneering work of Babuška [7] inspired the Lagrange multiplier approach, which was then introduced in [38, 36], for a Dirichlet problem. During the nineties, fictitious-domain methods became popular, and their use was extended to the study Navier-Stokes equations [39], fluid-structure interaction problems (see, e.g., [21]), particulate flows [37], and other problems.

As shown in [36], in the case of a particular finite element approximation, there are two main problems when considering this approach:

- compatibility conditions between the discretizations of B and Ω might impose some limitations, e.g., on the mesh size,
- the extension of the right hand side, from B to Ω, might affect the regularity of the solution over Ω. This problem is particularly apparent when considering a constrained solution, as in Chapter 2 and Chapter 4, where the right hand side is prescribed over the whole Ω.

The aim of this work is to study these methods, and extend their use to a continuum mechanics model of fiber reinforced materials. Our objective is to obtain a model which does not require aligned grids, and is computationally efficient.

1.1 Outline of the thesis

This thesis is divided into four chapters; in Chapter 2 we describe the fictitious domain method through an example model problem: a constrained Poisson equation, where the constraint is applied either to a codimension one domain or to a codimension zero domain. This allows us to introduce the basic mathematical tools needed for non-matching coupling, and show the numerical challenges connected to these problems. Chapter 3 deals with the technical problems related to the implementation of the coupled system, with a focus on the computation of coupling matrices, developing the first algorithm for the numerical coupling between arbitrarily distributed non-matching meshes. After describing the algorithms developed for the deal.II library, we show some benchmarks for both in serial and parallel examples. Finally, we show an application of these methods in Chapter 4, where we use the fictitious domain method to study composites materials, in particular fiber reinforced materials. After introducing the necessary tools of continuum mechanics and differential geometry, we develop a full three-dimensional model, where the effect of the fibers is imposed through a distributed Lagrange multiplier approach. We study our model using inf-sup conditions from Mixed Finite Elements (see, e.g., [19]). A numerical discretization of the fibers through three-dimensional meshes is expensive; thus, a one-dimensional approximation is often considered. In literature, to guarantee the existence of a coupling operator between the three-dimensional elastic matrix, and the one-dimensional fibers, Weighted Sobolev Spaces are used. As a new alternative, we propose a reduced model where the fibers are approximated with one-dimensional objects, and the coupling is obtained through an average operator. This method needs some additional modellistic hypotheses but does not require the use of some special functional spaces. Finally, we validate our model through some numerical simulations, where we compare the results of our coupled model with other solutions found in literature.

To conclude, Section 4.6 contains a summary of the main results, with some additional comments.

Chapter 2

Non-matching grid constraints

In this chapter, we introduce the problem of coupling multiple PDEs; after describing a general version of the problem, where a system of different, coupled, PDEs is introduced, we study the example of a constrained *Poisson Equation*.

The Poisson Equation, in its most basic form, reads: given a domain Ω, and a function f defined on Ω, find the function u such that

$$\begin{aligned} -\Delta u &= f \quad \text{on } \Omega \\ u &= 0 \quad \text{on } \partial\Omega. \end{aligned} \tag{2.1}$$

This equation is suitable to model a number of physical problems involving, for example, fluid mechanics, electromagnetism and heat transfer, and has numerous application in pure mathematical fields, such as probability and number theory [85, 34].

With some regularity conditions on the domain Ω, and on the function f, it is possible to prove existence and uniqueness of a solution for Equation 2.1, and in a number of cases it is also possible to find an explicit analytical solution.

One great difficulty remains when studying this problem for complex geometric shapes, which are typical of real-world problems. In these cases, the standard solving strategies involve the use of numerical methods; multiple difficulties are connected to the application of numerical methods on complex geometries: from the high computational costs to the very generation of meshes (which remains a challenge, especially in the three-dimensional case), to all problems connected with time evolution and large deformations of a mesh [64].

A number of solutions to these difficulties have been proposed and, among these, the fictitious domain method sees many successful applications. Variations of this technique are presented in the literature using also other names, such as the *immersed finite element method* or the *distributed Lagrange multiplier method*. The idea is to embed the complex geometry of the domain B in a larger, simpler domain $\Omega \supset B$ of the same dimension, where the PDE is solved. This allows to keep the discretization of the two grids, and their relative finite element spaces, completely independent, but requires the development of a method to "transfer" the original boundary equation from the complex geometry to the simple domain.

This technique is particularly efficient for the simulation of fluid-structure interaction problems, where the configuration of the embedded structure is part of the problem itself, and one solves a (possibly non-linear) elastic problem to determine the (time dependent) configuration of B, and a (possibly non-linear) flow problem in $\Omega \setminus B$, plus coupling conditions on the interface between the fluid and the solid.

This extension problem is closely related to *constrained problems*, where the value of the solution is prescribed on a certain subdomain $B \subset \Omega$.

To describe the constraint we consider a function g defined over B; then our constrained problems consists in looking for a solution u of the Poisson problem defined over Ω, and such that its restriction to B coincides with g.

A possible numerical strategy for the discretization of constrained problems involves the use of matching discretizations, where the alignment between the meshes of Γ and Ω is enforced; this approach is often not practical in applications requiring complex geometries and a time dependent domain [64]. As an alternative, we propose the use of Lagrange multipliers on non-matching grids. The arguments of this chapter are based on a tutorial of the deal.ii library: step 60 [62], written with prof. Luca Heltai, where it is shown how to solve the Poisson problem on Ω, with an additional constraint defined on a non-matching subdomain B, of codimension zero or one.

We discuss the constrained Problem following Glowinski [38], using this example to show the mathematical and numerical tools needed for non-matching coupling. To conclude we report some numerical results, and provide some details of the numerical implementation.

2.1 The model problem

Consider the n-dimensional domain $\Omega \subset \mathbb{R}^n$, let B be a subdomain of Ω such that $B \Subset \Omega$; then $B \cap \partial\Omega = \emptyset$. Define $\hat{\Omega} \coloneqq \Omega \setminus B$, we assume $\hat{\Omega}$ is also an n-dimensional domain. Let $V(\Omega)$ be a functional space defined over Ω, let $\hat{V}(\hat{\Omega})$ be a functional space defined over $\hat{\Omega}$, such that for every $v \in V(\Omega)$, $v\big|_{\hat{\Omega}} \in \hat{V}(\hat{\Omega})$. Let $W(B)$ be a functional space define over B. We assume that there exists a continuous operator

$$\gamma \colon V(\Omega) \to W(B),$$

which we call *coupling operator*. Given a function $u \in V(\Omega)$, we can think of γu as, essentially, the restriction of u over the domain B. This coupling operator allows to couple different PDEs, and to enforce certain conditions for the solution on B:

Problem 1 (Generic coupled problem)**.** *Consider two elliptic operators A_1 and A_2, defined respectively on $\hat{V}(\hat{\Omega})$ and $W(B)$. Then, through the coupling operator γ, we can define the following coupled problem: find $u \in V(\Omega)$ such that:*

$$\begin{aligned} A_1(u\big|_{\hat{\Omega}}) &= f_1 \text{ in } \hat{\Omega} \\ A_2(\gamma u) &= f_2 \text{ in } B \\ u &= 0 \text{ on } \partial\Omega, \end{aligned} \tag{2.2}$$

where f_1, f_2 are some functions defined on $\hat{\Omega}$ and B.

Equation 2.2 is a very generic way to interface two different PDEs, that is, two different physical problems. Indeed it may not have solution unless certain compatibility or transmission conditions are imposed. Assuming the existence and uniqueness of the solution, we now show how this problem is connected with an application of the fictitious domain method.

Assuming that operator A_1 can be extended to $V(\Omega)$, and that f_1 can be extended to f on Ω, we would like to modify Problem 1, considering only the operator A_1, and

replacing the problem on B by defining $g = A_2^{-1} f_2$. This is equivalent to controlling arbitrarily the value of the solution u on a part of the domain, imposing γu to be equal to the function g on B:

$$\begin{aligned} A_1 u &= f \text{ in } \Omega \\ \gamma u &= g \text{ in } B \\ u &= 0 \text{ on } \partial\Omega. \end{aligned} \tag{2.3}$$

As previously stated, Equation 2.3 has no solution unless certain compatibility conditions are satisfied; a strategy to modify the equation and impose these conditions is through the fictitious domain method with distributed Lagrange multipliers. Let $\gamma^T \colon W(B)' \to V(\Omega)'$ be the transpose of the coupling operator, Equation 2.3 can then be modified by imposing an arbitrary value for γu, and obtaining a Problem which, given some hypotheses on the domains and the boundary conditions, has a unique solution:

Problem 2 (Generic constrained problem)**.** *Consider the elliptic operator A_1 on $V(\Omega)$. Through the coupling operator γ, it is possible to define the following coupled problem: find $u \in V(\Omega)$ and λ, Lagrange multiplier defined over B, such that:*

$$\begin{aligned} A_1 u + \gamma^T \lambda &= f \ \textit{in } \Omega \\ \gamma u &= g \ \textit{in } B \\ u &= 0 \ \textit{on } \partial\Omega, \end{aligned} \tag{2.4}$$

where f, g are some functions defined on Ω and B respectively.

In general, the value of g may be given as an external datum, or it may be related to the solution to a coupled problem on B, i.e., $g = A_2^{-1} f_2$ (we provide an example in Chapter 4).

To fix ideas, in this Chapter we study the following prototype of a Poisson problem on Ω constrained over B:

Problem 3 (Model problem)**.** *Given a forcing term f defined on Ω, and a constraint g defined on B, find u satisfying:*

$$\begin{aligned} -\Delta u &= f \ \textit{in } \hat{\Omega} \\ \gamma u &= g \ \textit{in } B \\ u &= 0 \ \textit{on } \partial\Omega, \end{aligned} \tag{2.5}$$

where, for simplicity, we consider a homogeneous Dirichlet condition on $\partial\Omega$.

Equation 2.5 can be written in a distributional form over the entire domain Ω, by introducing a Lagrange multiplier λ.

Problem 4 (Distributional model problem)**.** *Given a forcing term f defined over Ω, a constraint g defined on B, find $u \in V(\Omega)$, and λ defined over B satisfying:*

$$\begin{aligned} -\Delta u + \gamma^T \lambda &= f \ \textit{in } \Omega \\ \gamma u &= g \ \textit{in } B \\ u &= 0 \ \textit{on } \partial\Omega. \end{aligned} \tag{2.6}$$

Where λ is a Lagrange multiplier that enforces the condition

$$\gamma u = g \quad \text{in } B,$$

in a variational form. As an example we use the duality product $W(B)' \times W(B)$ for the Lagrange multiplier λ (as done in, e.g., [21]). The weak formulation can then be found as a saddle point of the Lagrangian:

$$\mathcal{L}(u, \lambda) \coloneqq \frac{1}{2}(\nabla u, \nabla u)_\Omega - (w, f)_\Omega - \langle \lambda, \gamma u - g \rangle, \tag{2.7}$$

where $(\cdot, \cdot)_\Omega$ represents the L^2 scalar product over Ω, and $\langle \cdot, \cdot \rangle$ represents the duality product $W(B)' \times W(B)$. The weak variational formulation reads:

Problem 5 (Weak constrained formulation)**.** *Given a forcing term f defined over Ω, a constraint g defined on B, find $u \in V(\Omega)$, and $\lambda \in W(B)'$ satisfying:*

$$\begin{aligned} (\nabla u, \nabla v)_\Omega + \langle \lambda, \gamma v \rangle &= (f, v)_\Omega \qquad && \forall v \in V(\Omega) \\ \langle \gamma u, q \rangle &= \langle g, q \rangle \qquad && \forall q \in W(B)'. \end{aligned}$$

Assuming the restriction operator to be continuous, the second equation of Problem 5 can be rewritten as:

$$\langle q, \gamma u - g \rangle = 0 \qquad \forall q \in W(B)'.$$

2.1.1 The coupling operator

The coupling operator γ allows to use a function defined over B to "transmit information" to u, in this case constraining its value. The most obvious choice for this task is to consider the restriction of u over B; when considering continuous functions over a domain B, it is always possible to define the restriction operator:

$$\begin{aligned} \gamma_c : C^0(\bar{\Omega}) &\to C^0(B) \subset L^2(B) \\ u &\mapsto \gamma u = u\big|_B \in C^0(B). \end{aligned}$$

Our aim is to extend this operator to an operator $\gamma \colon V(\Omega) \to W(B)$.

We set Problem 5 in standard Sobolev Spaces; this has some consequence when considering the restriction of a function on B. To differentiate between codimension cases, instead of a generic B, we introduce the following domains:

- $\Omega \subset R^n$, a n dimensional subdomain of $\mathbb{R}^n$,
- $\Omega_a \Subset \Omega$, a subdomain of codimension 0,
- $\Gamma \coloneqq \partial\Omega_a$, a subdomain of codimension 1, which we assume to be the boundary of Ω_a,
- $\hat{\Omega} = \Omega \setminus \Omega_a$, a subdomain of codimension 0.

We assume all of these domains to be Lipschitz regular, and assume Ω, Ω_a, and Γ to be connected.

In the codimension 0 case, it is well known that the operator γ can be extended to a continuous operator on $H^1(\Omega)$, mapping functions in $H^1(\Omega)$ to functions in $H^1(\Omega_a)$.

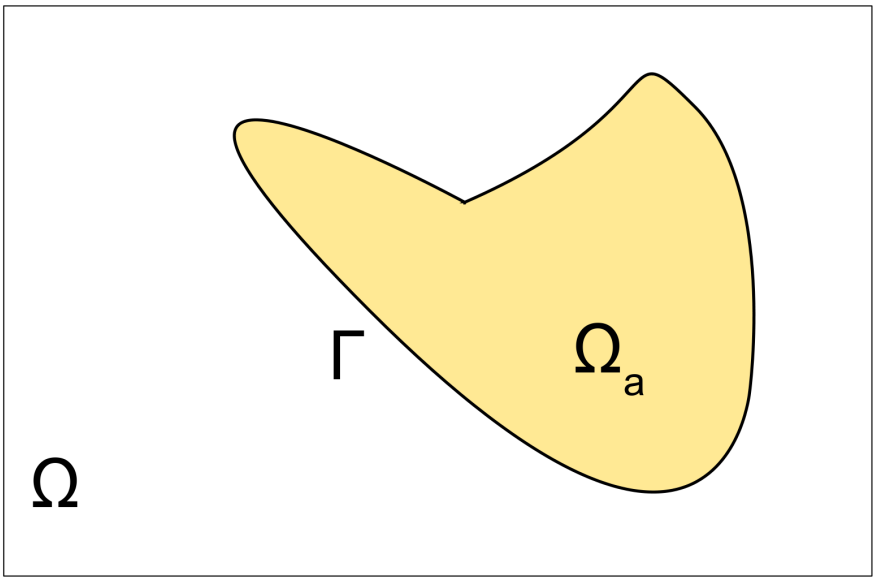


Figure 2.1: Example configuration for the domains: Ω contains Ω_a, of codimension 0, and its boundary Γ.

The codimension 1 case is more complicated, but if the domain Γ is Lipschitz and does not have a boundary, it is possible to extend γ to the so-called *trace operator*: a continuous operator mapping $H^1(\Omega)$ into $H^{1/2}(\Gamma)$, losing some regularity.

If B has codimension 2 or more, it is not possible, in general, to extend γ to $H^1(\Omega)$, and more involved approaches are necessary.

One possible solution for B of codimension 2, is to use weighted Sobolev spaces and graded meshes, as in [29]. Defining $L^2_\alpha(\Omega)$ as the Hilbert space of measurable functions u such that:

$$\int_\Omega u(x)^2 d^{2\alpha}(x)dx < \infty,$$

where d is the distance from x to B: $\mathrm{d}(x, B) \coloneqq \inf_{y\in B} \|y - x\|$, and the scalar product:

$$(u, v)_{L^2_\alpha} = \int_\Omega u(x)v(x)d^{2\alpha}(x)dx.$$

For $m \in \mathbb{N}$ we can then define the weighted sobolev space:

$$H^m_\alpha(\Omega) \coloneqq \{D^\beta u \in L^2_\alpha(\Omega), |\beta| \leq m\},$$

where β is a multi-index and D^β is the corresponding distributional partial derivative. Using the distance operator from B as a weight in the scalar product guarantees the existence a continuous restriction operator over B.

Another possibility is to substitute the "restriction operator", with some other operator which has the same purpose, e.g., considering the restriction of the elements of $V(\Omega)$ after a regularization through a convolution. For example, given $\eta \in C_c(\mathbb{R}^n)$, and extending u as 0 on Ω^c, we define $\eta * u$ for every $x \in \mathbb{R}^n$ as

$$\eta * u(x) \coloneqq \int_{\mathbb{R}^n} u(x - y)\eta(y)d\Omega_y.$$

Then $\eta * u \in C(\mathbb{R}^n)$, and we can consider its restriction [26].

The coupling operator might be defined in a number of other ways, for example, in Chapter 4 we study the codimension 2 case using an average operator, which approximates the restriction operator (in the case of continuous functions), and is well defined over the functional space $V(\Omega) = H_0^1(\Omega)$.

All these examples share two properties, which make coupling hard from a numerical point of view:

- Given a point $x \in B$, we need to evaluate functions defined over another grid, i.e. Ω, on x; the operation of transferring a point from one grid to the other is computationally intensive.
- The coupling at $x \in \mathbb{R}^n$ requires information about both grids in a neighbourhood of x. In a distributed setting, this information might be spread across multiple processes, making it not (easily) available.

For more details on these numerical challenges see Section 3.1.1.

2.1.2 Codimension one

Consider Ω, Ω_a, $\hat{\Omega}$ and Γ as above, define the following spaces:

$$\begin{aligned} V &\coloneqq H_0^1(\Omega) \\ W &\coloneqq H^{1/2}(\Gamma) \\ Q &\coloneqq L^2(\Gamma). \end{aligned}$$

Let $\gamma\colon V \to W$ be the trace operator, and $\gamma^T\colon W' \to V'$ be its transpose operator; identifying $L^2(\Gamma)$ with its dual, and since $L^2(\Gamma) \subset W'$, we can state the constrained problem in strong form as follows:

Problem 6 (Strong problem in codimension 1)**.** *Given a forcing term $f \in L^2(\Omega)$, and a constraint $g \in W$, find $(u, \lambda) \in V \times Q$ satisfying:*

$$\begin{aligned} -\Delta u + \gamma^T \lambda &= f \ \text{in } \Omega \setminus \Gamma \\ \gamma u &= g \ \text{in } \Gamma \\ u &= 0 \ \text{on } \partial\Omega, \end{aligned} \tag{2.8}$$

An equivalent formulation can be found as a critical point of the Lagrangian functional $\mathcal{L}\colon V \times L^2(\Gamma) \to \mathbb{R}$:

$$\mathcal{L}(v, \lambda) \coloneqq \frac{1}{2}(\nabla v, \nabla v)_\Omega - (v, f)_\Omega - (\lambda, v - g)_\Gamma. \tag{2.9}$$

Obtaining the following variational constrained problem:

Problem 7 (Weak constrained formulation)**.** *Given a forcing term $f \in L^2(\Omega)$, and a boundary term $g \in W$, find $(u, \lambda) \in V \times Q$ satisfying:*

$$\begin{aligned} (\nabla u, \nabla v)_\Omega + (\lambda, \gamma v)_\Gamma &= (f, v)_\Omega \qquad \forall v \in V \\ (\gamma u, q)_\Gamma &= (g, q)_\Gamma \qquad \forall q \in Q. \end{aligned} \tag{2.10}$$

The additional term $\gamma^T\lambda$ guarantees that the Laplacian is defined on the whole domain Ω. To better interpret the value of λ, we integrate by parts the first Equation of 2.10; assuming zero boundary conditions on $\partial\Omega$ and a sufficiently regular solution u:

$$(\lambda, \gamma v)_\Gamma = -(\Delta u, v)_\Omega + \int_\Gamma \frac{\partial u}{\partial \nu} v + (f, v)_\Omega$$

Testing with functions v such that $\gamma v = 0$ we obtain $f = -\Delta u$ almost everywhere on $\hat{\Omega}$. Then:

$$(\lambda, q) = (\frac{\partial u}{\partial \nu}, q) \qquad q \in L^2(\Gamma).$$

Using a similar procedure one can prove that λ is, in general, the L^2 projection of the jump of $\frac{\partial u}{\partial \nu}$ at Γ (see [38]).

In the article [38] Glowinski et. alia used as space $W(\Gamma)$ for the Lagrange multiplier λ, the space $H^{1/2}(\Gamma)$, and proved that the multiplier λ is the solution to a boundary equation involving a Steklov-Poincaré operator; the theory of these operators is fundamental in the study of boundary value problems [84].

Discrete formulation

To solve Problem 7 numerically we need to discretize the spaces V, W and Q; which we build using using Finite Elements:

$$\begin{aligned} V_h &= \text{span}\{v_i\}_{i=1}^N \subset V, \\ W_h &= \text{span}\{w_j\}_{j=1}^M \subset W, \\ Q_h &= \text{span}\{q_k\}_{k=1}^M \subset Q. \end{aligned}$$

respectively, where N is the dimension of V_h, and M the dimension of W_h and Q_h; we also assume:

$$W_h \subseteq Q_h.$$

Problem 8 (Discrete formulation)**.** *Given a forcing term $f \in L^2(\Omega)$, and a boundary term $g \in W$, find $(u_h, \lambda_h) \in V_h \times Q_h$ satisfying:*

$$\begin{aligned} (\nabla u_h, \nabla v_h)_\Omega + (\lambda_h, \gamma v_h)_\Gamma &= (f, v_h)_\Omega \qquad && \forall v_h \in V_h \\ (\gamma u_h, q_h)_\Gamma &= (g, q_h)_\Gamma \qquad && \forall q_h \in Q_h. \end{aligned}$$

Problem 8 can be represented with the following finite dimensional system of equations:

$$\begin{pmatrix} K & C^T \\ C & 0 \end{pmatrix} \begin{pmatrix} u \\ \lambda \end{pmatrix} = \begin{pmatrix} F \\ G \end{pmatrix}, \tag{2.11}$$

where

$$\begin{aligned} K_{ij} &\coloneqq (\nabla v_j, \nabla v_i)_\Omega && i, j = 1, \dots, N \\ C_{\alpha j} &\coloneqq (v_j, q_\alpha)_\Gamma && j = 1, \dots, N, \alpha = 1, \dots, M \\ F_i &\coloneqq (f, v_i)_\Omega && i = 1, \dots, N. \\ G_\alpha &\coloneqq (g, q_\alpha)_\Gamma && \alpha = 1, \dots, M. \end{aligned}$$

The matrix K is the standard stiffness matrix for the Poisson problem on Ω, the vectors F and G are the standard right-hand-side vectors for a finite element problem with forcing terms f and g defined, respectively, on Ω and Γ. The coupling between the two non-matching grids is handled by the matrices C and its transpose C^T, two non-standard matrices. Each entry of the matrix C is computed as:

$$C_{\alpha j} \coloneqq (\gamma v_j, q_\alpha)_\Gamma = \int_\Gamma v_j q_\alpha, \tag{2.12}$$

where v_j and q_α are the basis functions from the respective spaces. To compute the integral, using finite elements, we split it into contributions from all cells K of the triangulation used to discretize Γ; then the integration over K is transformed into an integral on the reference element $\hat{K}$, where we use a quadrature formula. Here F_K is the mapping from $\hat{K}$ to K:

$$\begin{aligned} C_{\alpha j} &\coloneqq (v_j, q_\alpha)_\Gamma \\ &= \sum_{K \in \Gamma} \int_{\hat{K}} \hat{q}_\alpha(\hat{x})(v_j \circ F_K)(\hat{x}) J_K(\hat{x}) \mathrm{d}\hat{x} \\ &= \sum_{K \in \Gamma} \sum_{i=1}^{n_q} \big(\hat{q}_\alpha(\hat{x}_i)(v_j \circ F_K)(\hat{x}_i) J_K(\hat{x}_i) w_i\big). \end{aligned}$$

Computing this sum is non-trivial because we have to evaluate $(v_j \circ F_K)(\hat{x}_i)$. In general, if Γ and Ω are not aligned, the point $F_K(\hat{x}_i)$ is completely arbitrary with respect to the grid of Ω. Moreover, when running the computation in parallel and using distributed meshes, it is not guaranteed that the process which is computing the integral over the cell $K \in \Gamma$ has the required information about the functions $v_j \in V_h$ which have $\mathrm{supp}(v_j) \cap K \neq \emptyset$; for more details on this problem see Section 3.1.1.

Once the matrices of Problem 2.11 are assembled, it is possible to solve the final saddle point problem using an interative solver, applied to the Schur complement $S \coloneqq CK^{-1}C^T$, obtaining:

$$\begin{pmatrix} \lambda \\ u \end{pmatrix} = \begin{pmatrix} S^{-1}(CK^{-1}F - G) \\ K^{-1}(F - C^T\lambda) \end{pmatrix}.$$

A sufficient condition for convergence as $h \to 0$ is the following inf-sup condition (see [27]): there exists a positive constant α such that, for every $\lambda_h \in Q_h$:

$$\sup_{v_h \in V_h} \frac{\int_\Gamma \lambda_h v_h}{\|\nabla v_h\|_\Omega} \geq \alpha \|\lambda_h\|_\Gamma.$$

It is difficult to prove this condition for two general, independent meshes; Glowinski proved the convergence directly in, e.g., [38] for a very specific configuration and assuming $g = 0$.

2.1.3 Codimension zero

Consider Ω, Ω_a, $\hat{\Omega}$ and Γ as above. For certain geometries, e.g., when solving a Problem on the $\hat{\Omega}$ shown in Figure 2.3a, it might be advantageous to consider the extended Problem on Ω, imposing a constraint for the solution on Ω_a. Similarly, it might be interesting to require a physical constraint to be satisfied over a codimension zero domain.

Consider g, defined over Ω_a, the constrained Poisson problem over Ω can be obtained through Lagrange multipliers, as in Subsection 2.1.2. Consider the spaces:

$$V \coloneqq H_0^1(\Omega),$$
$$W \coloneqq H^1(\Omega_a).$$

Let $\gamma\colon V \to W$ be the restriction from V to W, then we can describe the following constrained problems:

Problem 9 (Constrained distributional formulation on codimension 0)**.** *Given a forcing term $f \in L^2(\Omega)$, and a constraint term $g \in H^1(\Omega_a)$, find $(u, v) \in V \times L^2(\Omega_a)$ such that:*

$$\begin{aligned} -\Delta u + \gamma^T \lambda &= f \text{ in } \Omega \\ \gamma u &= g \text{ on } \Omega_a \\ u &= 0 \text{ on } \partial\Omega. \end{aligned} \tag{2.13}$$

Using similar procedures, and imposing the condition $\gamma u = g$ through a Lagrange multiplier in $L^2(\Omega_a)$, we obtain the equivalent weak formulation:

Problem 10 (Constrained weak formulation on codimension 0)**.** *Given a forcing term $f \in L^2(\Omega)$, and a constraint term $g \in H^1(\Omega_a)$, find $(u, \lambda) \in V \times L^2(\Omega_a)$ such that:*

$$\begin{aligned} (\nabla u, \nabla v)_\Omega + (\lambda, \gamma v)_{\Omega_a} &= (f, v)_\Omega && \forall v \in V \\ (\gamma u, q)_{\Omega_a} &= (g, q)_{\Omega_a} && \forall q \in L^2(\Omega_a). \end{aligned} \tag{2.14}$$

Existence and uniqueness of the solution for Problems 9 and 10 can be studied on the two subdomains $\hat{\Omega}$ and Ω_a. Existence and uniqueness on $\hat{\Omega}$ follows from the existence and uniqueness of a solution for the classic Poisson equation on Ω with forcing term $f - \chi_{\Omega_a}\lambda$ and 0 boundary condition, where χ_{Ω_a} is the characteristic function of the set Ω_a. Existence and uniqueness inside Ω_a is a result of the imposed constraint, which implies $u\big|_{\Omega_a} = g$.

Discrete formulation

As in Subsection 2.1.2, define V_h, W_h and Q_h finite dimensional subspaces of V, W, and $L^2(\Omega_a)$ respectively, with $W_h \subseteq Q_h$. The discretized form of Problem 10 reads:

Problem 11 (Constrained discrete formulation on codimension 0)**.** *Given a forcing term $f \in L^2(\Omega)$, and a constraint term $g \in H^1(\Omega_a)$, find $(u_h, \lambda_h) \in V_h \times W_h$ satisfying:*

$$(\nabla u_h, \nabla v_h)_\Omega + (\lambda_h, v_h)_{\Omega_a} = (f, v_h)_\Omega \qquad \forall v_h \in V_h \tag{2.15}$$
$$(u_h, q_h)_{\Omega_a} = (g, q_h)_{\Omega_a} \qquad \forall w_h \in Q_h. \tag{2.16}$$

If g is regular enough, it is possible to prove a general convergence result, with no particular dependence between the spaces V_h and W_h, see [38].

2.2 Numerical experiments

The results here reported were generated using the deal.II library; the Schur complement was solved using the conjugate gradient algorithm, with a tolerance of $1e^{-12}$.

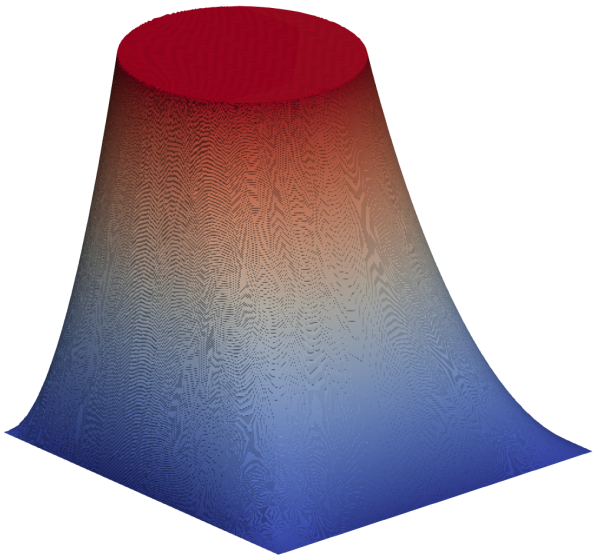


Figure 2.2: An example solution were u is not smooth.

Consider the following example problem: find (u, λ) such that:

$$\begin{aligned} -\Delta u + \gamma^T \lambda &= 0 \text{ in } \Omega \\ \gamma u &= 1 \text{ in } \Gamma \\ u &= 0 \text{ on } \partial\Omega, \end{aligned}$$

where $\Omega = [0,1]^2$, and Γ is the circumference of center $(0.4, 0.4)$ and radius 0.3.

A numerical solution can bee seen in Figure 2.2: even with very regular boundary conditions (in this case we are using constants); as expected, the gradient ∇u presents a discontinuity over Γ. This is one of the problems affecting the convergence rate for the discretized problem.

2.2.1 Codimension 1

As a first example we consider Ω to be the square $[0,1]^2$, let Ω_a be the circle of radius 0.3, with center in $(0.4, 0.4)$, and consider its circumference $\Gamma \coloneqq \partial\Omega_a$. Consider the following terms f and g on Ω:

$$\begin{aligned} f &\coloneqq 2\pi^2 \sin(\pi x) \sin(\pi y) \\ g &\coloneqq \sin(\pi x) \sin(\pi y). \end{aligned}$$

The solution u to the Poisson problem is in $C^\infty(\Omega)$:

$$u = \sin(\pi x) \sin(\pi y).$$

Let N be the number of degrees of freedom of V_h, M be the number of degrees of freedom of W_h; convergence Table 2.1 shows the error with respect of the analytic solution. Here h is the diameter of the cells of Ω's triangulation; the diameter h_g of the cells of Γ is $h_g \approx 1.3h$, and we are using linear elements for both meshes.

M	N	h	L2 error	rate	H1 error	rate	L^∞ error	rate
9	81	$\sqrt{2}/2^3$	0.0050	-	0.2578	-	0.0238	-
17	289	$\sqrt{2}/2^4$	0.0012	1.99	0.1262	1.03	0.0047	2.31
33	1089	$\sqrt{2}/2^5$	2.7e-4	2.22	0.0630	1.00	0.0011	2.13
65	4225	$\sqrt{2}/2^6$	6.7e-5	2.01	0.0314	1.00	2.8e-4	1.93
129	16641	$\sqrt{2}/2^7$	1.7e-5	1.98	0.0157	1.00	7.3e-5	1.95
257	66049	$\sqrt{2}/2^8$	4.2e-6	1.96	0.0078	1.00	1.9e-5	1.94
513	263169	$\sqrt{2}/2^9$	1.1e-6	2.03	0.0039	1.00	5.0e-6	1.92

Table 2.1: Convergence rate with linear elements, $u \in C^\infty(\Omega)$.

In this case the value of g coincides with the one of the solution for the Poisson problem without a constraint on Γ; thus we have the convergent rates of 2 and 1 for, respectively, L^2 and H^1 norms.

To obtain a less regular solution, we modify the function f as follows: let $r = (x-0.4)^2 + (y-0.4)^2$

$$\hat{f} \coloneqq \begin{cases} 2\pi^2 \sin(\pi x)\sin(\pi y) & \text{in } \hat{\Omega}, \\ 2\pi^2 \sin(\pi x)\sin(\pi y) + 0.03^2 - r^2 & \text{in } \Omega_a. \end{cases}$$

Using the same g function, we obtain the following solution on Ω:

$$\hat{u} \coloneqq \begin{cases} \sin(\pi x)\sin(\pi y) & \text{in } \hat{\Omega}, \\ \sin(\pi x)\sin(\pi y) - \frac{r^4}{16} + \frac{0.03^2 r^2}{4} & \text{in } \Omega_a. \end{cases}$$

The discontinuity of ∇u over Γ is small in norm, but as we can see in Table 2.2, this is sufficient to affect convergence rates.

What we can see from Table 2.2 (the parameters h and h_g are the same of the previous example), is that the H^1 error initially converges linearly, but the rate quickly drops. In the case of L^2 and L^∞ the convergence stops after reaching a certain threshold. A similar "stall" in the solution quality can be seen also using second order elements, or changing the relative mesh size: this effect is a consequence of the use of uniformly refined meshes.

Some solutions to this convergence problem can be found in literature: through adaptive refinement techniques it is possible to recover the convergence of the method and, in some cases, the optimality of the convergence (see, e.g., [13, 16]).

2.2.2 An example on a more complex geometry

Consider the curve Γ with the following parametrization:

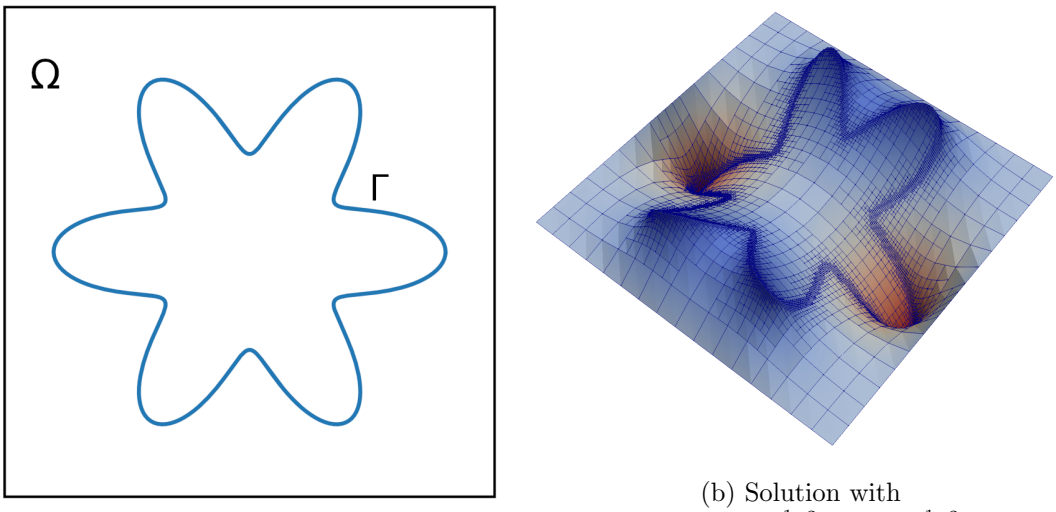
$$s \to (R + r\cos(w\pi s))\cos(2\pi s) + c_x; (R + r\cos(w\pi s))\sin(2\pi s) + c_y),$$

where $R = .3, c_x = .5, c_y = .5, r = .1, w = 12$, the domain is shown in Figure 2.3a.

This geometry is too complex for an uniform refinement strategy, but refining additionally cells of Ω which are close to the Γ boundary allows to better capture the geometry of Γ. Figure 2.3b shows the numerical solution with $f = 0$, $g = 2(x-0.5)^2 - 2(y-.5)^2$,

M	N	h	L2 error	rate	H1 error	rate	L^∞ error	rate
9	81	$\sqrt{2}/2^3$	0.0050	-	0.2578	-	0.0223	-
17	289	$\sqrt{2}/2^4$	0.0013	1.89	0.1263	1.02	0.0058	1.98
33	1089	$\sqrt{2}/2^5$	5.1e-4	1.37	0.0633	0.99	0.0023	1.31
65	4225	$\sqrt{2}/2^6$	4.7e-4	0.13	0.0322	0.97	0.0017	0.42
129	16641	$\sqrt{2}/2^7$	4.7e-4	0.00	0.0172	0.90	0.015	0.14
257	66049	$\sqrt{2}/2^8$	4.7e-4	0.00	0.0104	0.71	0.015	0.01
513	263169	$\sqrt{2}/2^9$	4.7e-4	0.00	0.0079	0.39	0.015	0.00

Table 2.2: Convergence rate with linear elements with a non-smooth solution.



(a) A complex example for the curve Γ

(b) Solution with $f = 0, g = 2(x - \frac{1}{2})^2 - 2(y - \frac{1}{2})^2$, and additional refinement close to Γ.

Figure 2.3: Solving on a complex geometry

with 4 uniform refinements of Ω and three additional refinements to the cells which are close to Γ.

2.2.3 Codimension 0

Consider Ω to be the square $[0, 1]^2$, and Ω_a be the circle of radius 0.3, with center in $(0.4, 0.4)$. We run our simulations with the same parameters of Subsection 2.2.1, and we first consider the case with solution $u = \sin(\pi x)\sin(\pi y)$ on Ω, and $g = u\big|_{\Omega_a}$. As we can see in Table 2.3, convergence rates for the H^1 norm are similar to the ones of the codimension 1 case, but the L^2 and L^∞ rates show some oscillations.

As a second numerical experiment we consider the functions:

$$f := 2\pi^2 \sin(\pi x)\sin(\pi y)$$
$$g := \sin(\pi x)\sin(\pi y) - x(0.3^2 - (x - 0.4)^2 - 2(y - 0.4)^2)^2.$$

M	N	h	L2 error	rate	H1 error	rate	L^∞ error	rate
25	289	$\sqrt{2}/2^3$	0.0011	-	0.1270	-	0.0047	-
81	1089	$\sqrt{2}/2^4$	3.2 e-4	1.79	0.0649	0.97	0.0031	0.61
289	4225	$\sqrt{2}/2^5$	8.11 e-5	1.98	0.0324	1.00	0.0011	1.42
1089	16641	$\sqrt{2}/2^6$	1.99 e-5	2.02	0.0161	1.00	6.4e-4	0.84
4225	66049	$\sqrt{2}/2^7$	4.19 e-6	2.25	0.0078	1.04	1.34 e-5	5.58
16641	263169	$\sqrt{2}/2^8$	1.85 e-6	1.17	0.0039	1.00	6.27 e-6	1.09

Table 2.3: Codimension 0: convergence rate with linear elements, $u \in C^\infty(\Omega)$.

M	N	h	L2 error	rate	H1 error	rate	L^∞ error	rate
25	289	$\sqrt{2}/2^3$	0.0011	-	0.1269	-	0.0047	-
81	1089	$\sqrt{2}/2^4$	3.3 e-4	1.8	0.0648	0.97	0.0031	0.61
289	4225	$\sqrt{2}/2^5$	8.15 e-5	2.03	0.0323	1.00	0.0011	1.42
1089	16641	$\sqrt{2}/2^6$	1.99 e-5	2.03	0.0161	1.00	6.2e-4	0.86
4225	66049	$\sqrt{2}/2^7$	4.85 e-6	2.03	0.0079	1.02	4.91 e-5	3.66
16641	263169	$\sqrt{2}/2^8$	2.38 e-6	1.02	0.0039	1.01	3.21 e-06	0.61

Table 2.4: Codimension 0: convergence rate with linear elements with a non-smooth solution.

The solution u to the Poisson problem is:

$$\hat{u} \coloneqq \begin{cases} \sin(\pi x)\sin(\pi y) & \text{in } \hat{\Omega}, \\ g & \text{in } \Omega_a, \end{cases}$$

which is non-smooth; as shown in Table 2.4, in this case the convergence rates do not stall, even though the solution is non-smooth.

Chapter 3

Numerical coupling

In Chapter 2 we introduced the theory behind the problem of numerical coupling for non-matching meshes through Lagrange multipliers. We now wish to solve the challenges which need to be addressed when implementing these methods, and show how to design algorithms which use efficiently modern hardware, with its characteristics and limitations.

The main objective of this chapter is the numerical evaluation of an example of the prototypical coupling problem: the coupling matrix C, used in the fictitious domain method. This task requires to efficiently transfer information between non-matching grids. Moreover, we are interested in extending these algorithms to the case of a distributed setting.

After describing the state of the art for numerical coupling, we analyse the algorithms present in deal.II 8.5.1 for this task, and then describe how to improve then, and how to adapt them for a distributed environment. The coupling algorithms we describe are quite generic, and can be used to solve a number of problems in finite elements, finite volumes, or finite differences, whenever a form of coupling is required.

The tools we developed include a Cache class, to memorize computationally intensive data, and a Parameter class, to better handle recurring parameters.

This chapter was developed as the final project for the MHPC (Master in High Performance Computing), and the code implemented was developed on top of the deal.II 8.5 library, and included in the deal.II library version 9.0 [5, 3].

3.1 Introduction

Computer clusters with thousands of cores are going to be the heart of most scientific computing, at least for the foreseeable future [9]; as a consequence, dedicated parallel codes have to be incorporated into scientific software.

As the number of cores increases, two common strategies used to facilitate the writing of parallel code become bottlenecks, hindering the code's performance:

- data replication across all processes,
- all-to-all communications.

The solution is apparently simple: using point-to-point communication, and distributed data structures [9]. Unfortunately these strategies are more complex, and become par-

ticularly difficult in simulations requiring coupling. Without expensive, dedicated algorithms, the parallel decompositions of geometric domains do not correlate and are independent [95]: while data distribution makes it possible to fit the simulation in the system, it inherently complicates the process of coupling, making it necessary to resort to complex communication patterns and strategies.

In Section 3.1.1, we formalize a general coupling problem and show the example of the coupling matrix C. After discussing some of the strategies found in the literature, we propose some improvements to the current implementations present in the deal.II library, version 8.5.1. After showing how, the absence of data locality makes this approach fail in a distributed setting, we propose and analyse a distributed algorithm, and study its performance. To conclude, we briefly report on furtherer improvements obtained though the use of trees data structures.

3.1.1 Generalized Coupling

Consider two physical problems defined on (possibly different) domains $A, B \subseteq \mathbb{R}^n$. Assume that $V(A)$ and $W(B)$ are some functional spaces on A and B respectively, a generic definition of coupling between fields defined on A and fields defined on B can be regarded as a continuous bi-linear operator in the form:

$$C : V(A) \times W(B) \mapsto \mathbb{R}. \tag{3.1}$$

Among all possible couplings, the computation of the coupling matrix is an example of a wide class of couplings, depending on a function in the form

$$K \colon A \times B \to \mathbb{R},$$

called *coupling kernel*. Then the coupling operator C takes the form:

$$C(v, w) := \int_A \int_B v(x) K(x, y) w(y) \mathrm{d}A_x \, \mathrm{d}B_y, \tag{3.2}$$

where $V(A)$ and $W(B)$ are some standard functional spaces, the above integrals are well defined, and the functional C is bounded.

As an example, consider the case where $B \subset A$, and fix $V(A)$ to be the space of continuous functions on A, i.e., $V(A) := C^0(A)$, and $W(B)$ the space of distributions on B, i.e., $W(B) := \mathcal{D}(B)$. Define the coupling kernel as $K(x, y) := \delta(x - y)$, where δ is the n-dimensional Dirac delta distribution defined by its action on continuous functions as

$$v(x) := \int_{\mathbb{R}^n} \delta(x - y) v(y) \mathrm{d}y, \qquad \forall v \in C^0(\mathbb{R}^n). \tag{3.3}$$

In this very particular case, if we pick a subset of points y_i contained in the domain $B \subset A$, and associate a Dirac delta to each of these points as an element of $W(B)$, say

$$w_i(y) := \delta(y - y_i), \tag{3.4}$$

the coupling defined above between an arbitrary function $v \in V(A)$ and w_i reduces simply to its point evaluation on y_i, i.e.:

$$C(v, w_i) := \int_A \int_B v(x) K(x, y) w_i(y) \mathrm{d}A_x \, \mathrm{d}B_y = \int_B v(y) w_i(y) \mathrm{d}B_y = v(y_i), \tag{3.5}$$

where the Kernel K removes the first integral in $\mathrm{d}A_x$ by the definition in (3.3), and the definition of the distributions w_i remove the second integral in $\mathrm{d}B_y$ using the same principle.

When the spaces and the Kernel are chosen as above, the coupling is simply called *interpolation* on the points y_i of the function v. The typical application of this *interpolation* operator is when the field $V(A)$ represents some physical quantity, for example a continuous temperature field, and y_i are the vertices of an embedded triangulation, not aligned with A, where we would like to evaluate this physical quantity, e.g., the temperature.

Different definitions of the spaces $V(A)$ and $W(B)$, and of the coupling kernel K lead to different coupling operators. In this chapter we will restrict our examples to the *interpolation* coupling above.

In the non-matching case, the domains A and B are discretised using independent triangulations, and are split on non-intersecting *cells*.

When physical fields are defined on one of these triangulations, say for example on A, they use the seperation on M cells to restrict the possible choice of functions to a *finite dimensional space* of dimension N, constructed using a linear combination of some *basis functions*, defined through the triangulation itself.

In general this construction is done in four steps:

- triangulate the domain A into a collection of $M \in \mathbb{N}$ cells

  $$A_h := \cup_{i=1}^{M}(K_i = F_i(\hat{K})),$$

  images under M (possibly non linear) isomorphisms of a reference cell $\hat{K}$;

- define a set of $nloc \in \mathbb{N}$ basis functions on the reference element $\hat{K}$, $\{\hat{v}_i\}_{i=1}^{nloc}$;

- define some global basis functions on A_h, as the push forward of $\hat{v}_j$ under F_i on $K = F_i(\hat{K})$, i.e.,

  $$\hat{v}_j(\hat{x}) \coloneqq v_l(F_i(\hat{x}))|_K,$$

  where l is an appropriate global numbering depending on j and i

- enumerate the global basis functions so that we have $V_h(A) \subset V(A)$ and $V_h(A) := \mathrm{span}\{v_i\}_{i=1}^{N}$, where $N \in \mathbb{N}$ is the dimension of the discrete space.

The construction above guarantees that any function $v_h \in V_h(A_h)$ can be expressed as

$$v_h(x) := \sum_{i=1}^{N} v^i v_i(x), \tag{3.6}$$

where the functions $v_i(x)$ are different from zero only on a limited number of cells K of A_h, namely $\mathrm{supp}(v_i) := \{K \in A_h \text{ s.t. } v_i|_K \neq 0\}$. Notice that, on any K, only $nloc$ global basis functions are different from zero. For the space $V_h(A_h)$, the *interpolation* coupling on a collection of points $y_\alpha \in B_h$, the triangulation of the domain B, can be expressed by the interpolation matrix C:

$$C_{\alpha i} := C(v_i, w_\alpha) := v_i(y_\alpha), \tag{3.7}$$

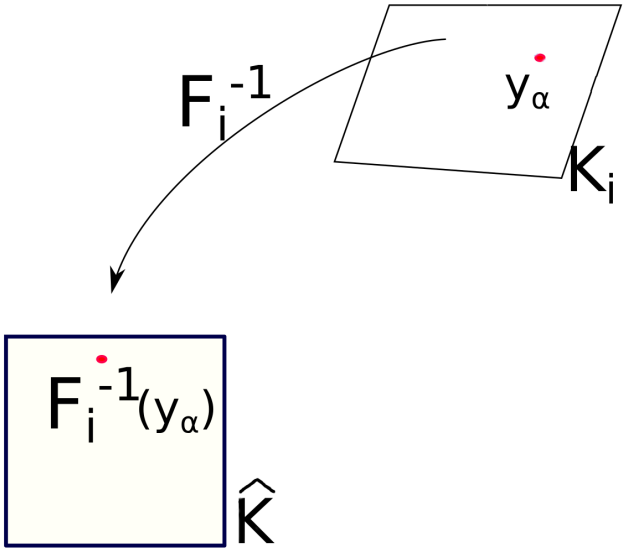


Figure 3.1: Interpolation Problem

such that, when it is multiplied with the vector of coefficients v^i of a generic function v_h, we obtain the *interpolation* of the function v_h on the points y_α:

$$\sum_{i=1}^{N} C_{\alpha i} v^i = C(v_h, w_\alpha) := \sum_{i=1}^{N} v^i v_i(y_\alpha) = v_h(y_\alpha). \tag{3.8}$$

Numerically only $\hat{v}_i$, F_i and F_i^{-1} can be computed directly: to evaluation of Equation 3.8 is achieved through F_i^{-1}:

$$v_i(y_\alpha) = \hat{v}_i \left(F_i^{-1}(y_\alpha) \right). \tag{3.9}$$

This translates in the following steps, which are illustrated in Figure 3.1:

1. Use the triangulation B_h to identify y_α in the real space.
2. Find the cell K_i of the triangulation of A for which $y_\alpha \in K_i$.
3. Use the triangulation of A to obtain F_i^{-1} and compute $F_i^{-1}(y_\alpha)$.
4. Evaluate the result on the basis functions v_i.

3.1.2 Existing Solutions

To fix the terminology: in a *distributed mesh* each process[1] "owns" a number of cells, of which it knowns all the variables and on which it runs computations; we call these cells

[1]Since this work is algorithmic, we use "process" as a generic term not making distinctions between processors, processor cores or MPI processes

locally owned. We shall use the term "own" to describe any object, such as cells and particles, for which the current process is the one knowing all information and running all computations.

Typically each process maintains the information of one or more layers of cells surrounding its locally owned cells, these belong to other processes and are called *ghost cells*; each process knowns all variables relative to ghost cells but can't modify their value. All remaining cells, which are not locally owned nor ghost, are called an *artificial cells*; their existence is purely instrumental to the chosen data structure, but has no participation in the solution of the actual physical problem

Particle-in-Cell Methods

The particle-in-cell (PIC) method is used to solve a certain class of partial differential equations (PDEs); the technique consist in tracking individual particles, which have a position and a number of individual properties, in a Lagrangian frame; at the same time moments of a distribution, e.g., such as densities of some physical fields, are computed on a stationary mesh. PIC methods have a long history, which sparked with applications in plasma physics [89].

Only recently PIC methods have been implemented in a fully distributed setting; Rene Gassmoeller et al. published a paper [35] describing how to implement PIC methods in a state-of-the art fluid dynamic solver, and implemented it using the ASPECT library [47, 10].

Our coupling problem has some similarities to the ones studied by Gassmoeller and, in fact, we benefited greatly from their work; but there are some important differences.

In a PIC method, each process owns a number of cells and generates particles inside it using an algorithm, e.g., randomly, or following particular patterns. After generating the particles, each process owns both the cells and the particles inside them; when working with independent meshes this is no longer the case (see the next paragraph on the Data Transfer Kit).

The method described by Gassmoeller requires a bound on the speed of each particle; this simplifies the algorithm because, if a particle exists the locally owned part of the mesh, it will most likely end up on a ghost cell, i.e., in which case the new owner is known to the current process, and it is possible to transfer the particle. If this is not possible, because the particle is lying on an artificial cell, the algorithm deletes it, approximating the result. As shown in [35], the bound on the particle's speed results in a low probability of a particle entering an artificial cell: given the small amount of deleted particles, the numerical effects are small.

While the strategy of communicating points lying on ghost cells clearly adapts to our cases, where coupling can be computed on ghost cells, eliminating a part of a mesh is not an option in a coupling problem. In general coupling problems we have little control on the topological distribution of the locally owned parts of the domains A and B: given a process, the spatial intersection of these parts might be small or even empty.

One solution to this problem, is to create a balanced configuration where the partition guarantees that, spatially, a process owning a cell K of A owns also the spatial cells of B which intersect K. This can be done, for example, with multi-constraint partitioning techniques [94].

Another approach, which is more flexible, is to develop a strategy to communicate points in all possible distributed settings. This would also allow the possibility to

change the distribution of each mesh during a simulation, allowing to re-initialize the mesh distribution even at running time, with a different number of processes and/or a different distribution scheme.

Data Transfer Kit

Many modern physical simulations make use of a partitioned approach: different kernels, handling different parts of the problem, work together to solve a complex problem. For example, in a fluid-structure interaction problem, a kernel might handle fluid equations and another might handle the structure; in this scenario accuracy and speed at data transfer is fundamental.

The Data Transfer Library (DTK) [96] makes this kind of transfer possible through rendezvous algorithms. These algorithms were developed by Plimpton et. al. [83], and allow to generate balanced meshes and transfer information between different triangulations, with some additional communication costs. This is how the algorithm is briefly described in [95]:

> for mesh-based data transfer generates the rendezvous decomposition which behaves as a hierarchical parallel and geometric search tree. Using this algorithm, a secondary decomposition of a subset of the source mesh that will participate in data transfer is generated, forming the rendezvous decomposition as described in the example above. The rendezvous decomposition is encapsulated as a separate entity from the original geometric description of the domain. It can be viewed as a temporary copy of the source mesh subset that intersects the target geometry. With the rendezvous decomposition, we effectively have a search structure that spans both parallel and physical space.

This "search structure that spans both parallel and physical space" is the missing step in the solution of our interpolation problem, which is absent in the PIC solution; this structure is implemented in DTK using trees of bounding boxes which are needed for the initial partition.

We shall create a similar structure, without constraints on the methods used for mesh partitioning.

3.2 DAG Structure

The structure of modern general purpose numerical libraries is organized to offer an abstraction to the user, while optimizing the object structure and the performance. This is often achieved structuring the code with a Directed Acyclic Graph (DAG) [18].

A DAG is a directed graph with topological ordering: each node structurally represents an object, and the directed edges stemming from it, represent how it can be used to generate new objects or data; a simple example is shown in Figure 3.2. This structure is efficient and has many benefits for a parallel code, but creates some "asymmetries", i.e., for certain operations it favours access along one direction, while making the inverse operation slower.

An example of this effect, in the deal.II library, is related to vertices and cells: finding the vertices of a cell is a very efficient operation but, given a vertex, it is inefficient to

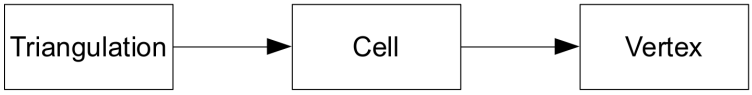


Figure 3.2: Example of the DAG structure in a numerical library: the hierarchy of the objects imposes a "preferred" way to access them.

find its neighbouring cells. For an algorithm requiring access to the neighbouring cells of a vector, this "inverse operation" is likely to be an important bottleneck; the only possible solution is to provide, with a new function or class, a new edge with the inverse direction in the DAG structure.

deal.II Utilities

The numerical simulations described in this thesis were made using and developing the deal.II library [11, 3]. The deal.II library is a modern example of a *state of the art* numerical library; as stated in the deal.II website (http://www.dealii.org/):

> deal.II is a C++ program library targeted at the computational solution of partial differential equations using adaptive finite elements. It uses state-of-the-art programming techniques to offer a modern interface to the complex data structures and algorithms required.
>
> The main aim of deal.II is to enable rapid development of modern finite element codes, using among other aspects adaptive meshes and a wide array of tools classes often used in finite element program. Writing such programs is a non-trivial task, and successful programs tend to become very large and complex. We believe that this is best done using a program library that takes care of the details of grid handling and refinement, handling of degrees of freedom, input of meshes and output of results in graphics formats, and the like. Likewise, support for several space dimensions at once is included in a way such that programs can be written independent of the space dimension without unreasonable penalties on run-time and memory consumption.

3.3 Serial Case

In Figure 3.3 we see a generic coupling example in two dimensions: two meshes intersect, and the point $y_i \in B$ needs to be evaluated by A.

Compute Point Locations

The main difficulties in computing $v_i(y_\alpha) = \hat{v}_i\left(F_i^{-1}(y_\alpha)\right)$ are:

- Find which cell K_i of A contains y_α.
- Use the triangulation of A to obtain F_i^{-1} and compute $F_i^{-1}(y_\alpha)$.

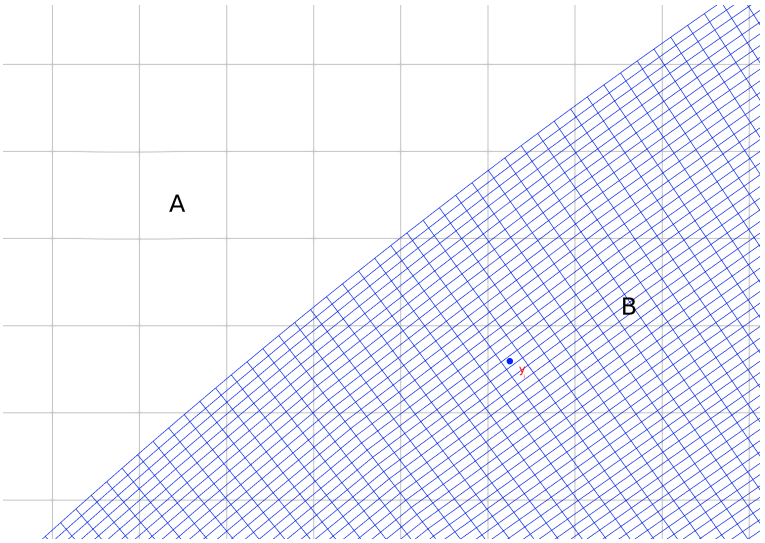


Figure 3.3: Coupling of two meshes, point y_i is known by B, and has to be communicated to A.

In a fully serial environment, all the information is present and, in the deal.II library version 8.5.1., these operations are coded in a function called *compute point locations*, which implements the following algorithm:

Input: A triangulation, a list of points

Output: cells containing points and their transformed

Initialization:
p = first point of the list;
Find the cell K surrounding p and compute $q = F_K^{-1}(p)$;
set p as found;
while *there are points not found* **do**
  for *p in points left* **do**
    compute $q = F_K^{-1}(p)$;
    if *q inside unit cell* **then**
      add p and q to the cell's list;
      set p as found;
    end
  end
  if *there are points not found* **then**
    p = first point not found;
    Find the cell K surrounding p and compute $q = F_K^{-1}(p)$;
    set p as found;
  end
end

Algorithm 1: Compute Point Locations in deal.II 8.5.1

The following should be noted on algorithm 1:

- If points are pre-ordered by cell the computation is faster: at every step the number of "not found" points decreases by the amount of points. This assumption is often

satisfied when working with meshes and their vertices.

- Using the inverse function F_K^{-1} to check if the points are inside the cell is inefficient.
- The cell surrounding p is found using *find active cell around point*, which has a high computational cost, as it does not follow the DAG structure (see Subsection 3.2).

This is how *find active cell around point* was implemented in deal.II 8.5.1.

Input: A triangulation, a point p

Output: cell containing the point and it's transformed

initialize the vector **distances**;
for *v in triangulation's vector* **do**
  compute the distance $||p - v||$;
  add it to **distances**;
end
Find the minimum in **distances**;
Save the corresponding vector v;
Initialize the cell's vector **neighbours**;
for *K in triangulation's cell* **do**
  if *v is a vertex of K* **then**
    Add K to **neighbours**;
  end
end
for *K in* ***neighbours*** **do**
  compute $q = F_K^{-1}(p)$;
  if *q inside unit cell* **then**
    return q and K;
  end
end

Algorithm 2: Find active cell around point in deal.II 8.5.1

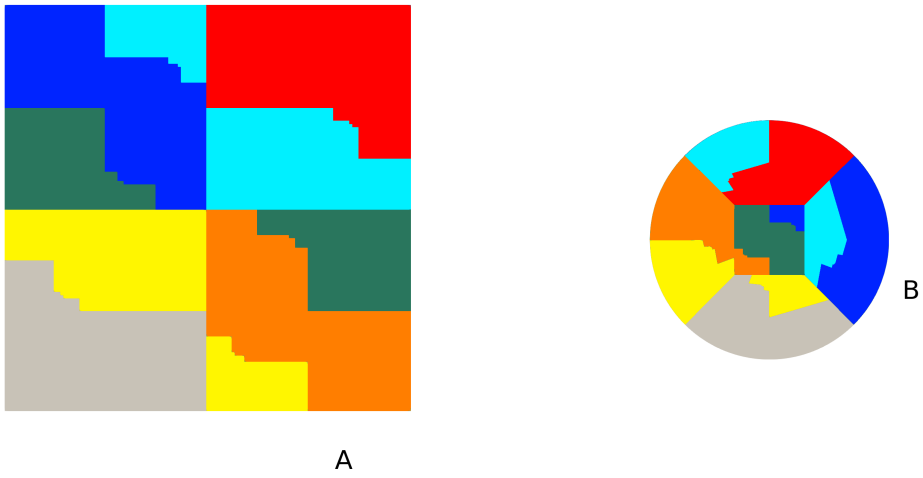
Algorithm 2 suffers from requiring the the relation from *vertex* to *neighbour cells*, and using the inverse function $F_K^{-1}(p)$, instead of a simpler method to guess which cell is most likely contain p.

The computational results of Subsection 3.5.1 confirm the poor performance of this approach.

3.4 Parallel Case

In addition to what stated in Subsection 3.1.2, we make the following general assumptions on distributed meshes (as reported in [9]):

- *Common coarse mesh*: all cells are derived by refinement from a common coarse mesh, which needs to capture the topology of the computational domain. Each cell is hierarchically refined into four (2d) or 8 (3d) children, which may be further refined, forming a forest.
- *Distributed storage*: Each processor in a parallel program may only store a part of the entire forest (the *locally owned* part).



(a) A partitioned square

(b) A partitioned ball

Figure 3.4: Partitioned meshes

As an example consider, in $\mathbb{R}^2$, the unitary square $[0,1]^2$ with its triangulation A, and the ball of radius 0.35 and center in $(0.4, 0.45)$ with its triangulation B. Figure 3.4 shows an example, obtained with deal.II, of a possible distribution for the two triangulations A and B: each color represents a different process. The two images are separated for better visualization, even though the ball is contained inside the square. To compute Equation 3.9, the process owning the point y_α which lies on B, must also own the cell K of A containing y_α; as the figures shows, in a distributed setting this condition is most likely **not** satisfied, and ghost cells are not enough to solve the problem.

Figure 3.5 shows a three-dimensional example: the mismatching between colors identifies situations where coupling is not possible, unless information is transferred between different processes.

The problem which needs to be solved has become an *ownership* problem: "which process owns the part of A in which p lies?" or "Is a part of B inside this portion of space?".

3.4.1 Bounding Boxes

An accurate description of the boundaries of each locally owned domain would answer to the problem, but this solution is not feasible, as it would require no data distribution. As a solution, we develop an approximated description, which we can find the process owning a point, possibly among few "guesses".

A new data structure has to be introduced; ideally this structure should be efficient and reliable while using as little memory and communication as possible. The natural solution is using simple containers, such as bounding spheres (BS) or axis-aligned bounding boxes (AABB). Both solutions are covered by a wide literature because of their applications ranging in the most diverse areas: from robotics to computer graph-

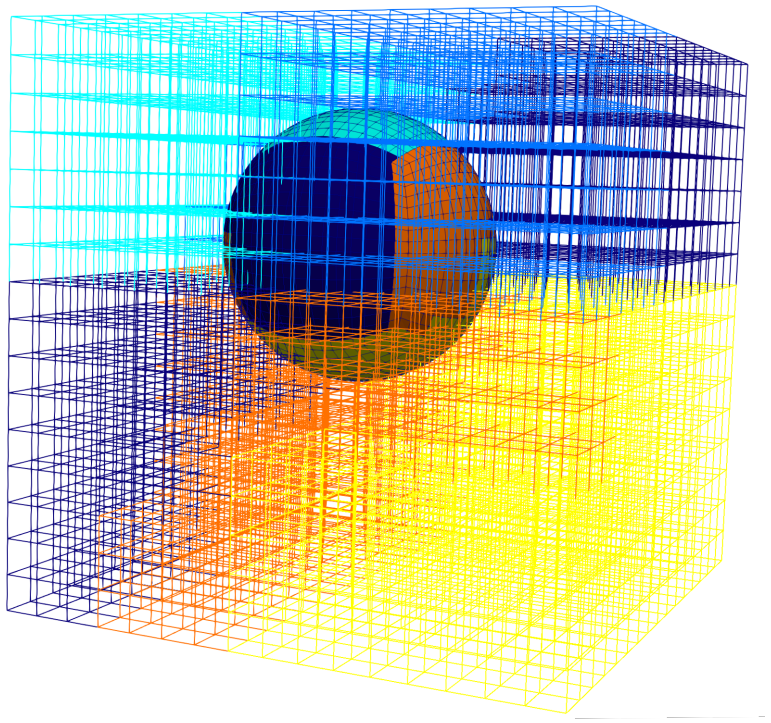


Figure 3.5: Distributed cube and sphere

ics (see, e.g., [33]). Our choice was to use AABB, as done in the DTK library, because of their best fitting abilities.

Bounding Boxes Computations

An AABB is described through a pair of points describing the bottom-left and top-right corners of the box. A number of standard functions to quickly check if a point is inside/outside a bounding box, computing the intersection and union of two bounding boxes etc. are well known in literature (see, e.g., [33]).

The intersection of two bounding boxes $b1$, and $b2$ coincides with their intersection as closed sets; but, if we define the operation $\sqcup$ as the union of two bounding boxes which, given two bounding boxes $b1$ and $b2$, returns the smallest bounding box contianing both $b1$ and $b2$, then $b1 \sqcup b2 \subseteq b1 \cup b2$ (see Figure 3.6). In the second example we see a case in which $b1 \sqcup b2 = b1 \cup b2$; we shall call such cases "mergeable". For the algorithm implemented it is of importance to understand if two bounding boxes are mergeable or not; this is computed with Algorithm 3.

Given a refinement level which has enough coarse cells to describe the space it is possible to create a description of the space occupied by the locally owned cells. This is a sketch of the algorithm used:

Algorithm 4 is guaranteed to return a collection of bounding boxes which contains all

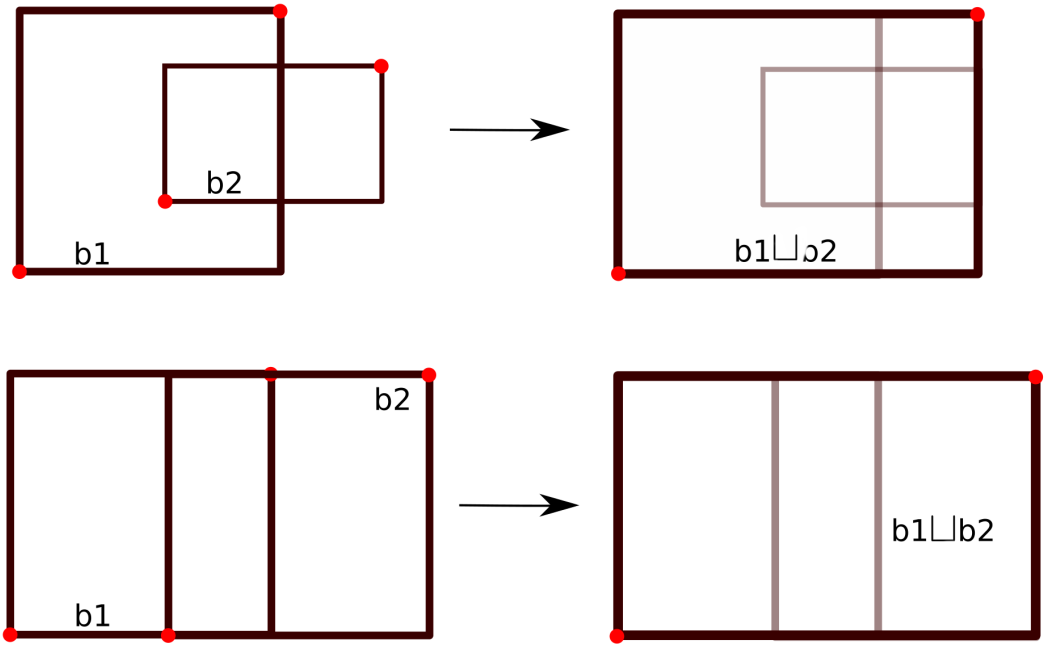


Figure 3.6: Non-mergeable and mergeable bounding boxes

Input: Two bounding boxes $b1$ **and** $b2$ **in the space of dimension** ***spacedim***

Output: True if the bounding boxes are mergeable

compute $b3 = b1 \cap b2$;
if $b3$ *is empty or has dimension* $\leq$ spacedim-*2* **then**
  return false;
else if $b1$ *and* $b2$ *are aligned or* $b1 \subset b2$ *or* $b2 \subseteq b1$ **then**
  return true;
end
return false;

Algorithm 3: Checking if bounding boxes are mergeable

Input: A triangulation, a refinement level

Output: A list of bounding boxes

initialize the list of bounding boxes b_list;
for K *in triangulation's cell of refinement level* **do**
  if *children of* K *are locally owned* **then**
    Create a bounding box *bb* surrounding all locally owned cells inside K;
    Add *bb* to b_list;
  end
end
Merging section:;
for $b1$ *in b_list;*
do
  merge_happened = false;
  for $b2$ *in b_list;*
  do
    if $b1 != b2$ *and* $b1, b2$ *are mergeable* **then**
      Add $b1 \sqcup b2$ to b_list;
      Remove $b1$ and $b2$ from b_list;
      merge_happened = true;
    end
    if *merge_happened* == *false* **then**
      return b_list;
    end
  end
end

Algorithm 4: Creating a description of the locally owned meshes

locally owned cells; but, because of the quadratic time required in the merging section of algorithm, it is suitable only for meshes obtained from a relatively small number of coarse cells. Another flaw is the need to choose a refinement level which works well with the current mesh. However, coupling complexity for the coupling is so high that Algorithm 4 suits our needs without using any relevant computing time. Moreover, in Section 3.6 we show how the use of the boost library recently offered an alternative solution, without the quadratic time limitation.

Guessing Ownership Once every process has a global description of the mesh, in terms of bounding boxes, a search through the bounding boxes allows to guess the possible "owners" of each point.

Bounding Boxes Exchange

It is important for the global description returned, for example, by of Algorithm 4, to contain few bounding boxes, as these boxes are communicated with a collective "allgather", and replicated on each process. This operation allows to build a container, *global bounding boxes*, containing an approximated description of the locally owned part of the whole mesh.

Given a point, a search through *global bounding boxes*, returns one (or few) processes owning the point. This allows to solve the coupling problem through one (or few) point-to-point communications (see Subsection 3.4.1).

Object Communication Sending multiple, different, objects using MPI is often difficult, complicated and impractical. A key ingredient to simplify our implementation, was the definition of two interfaces for an "all gather" and a "some to some" functions which can send arbitrary objects. This is made possible implementing a simple a serialize function for the Boost library [24].

This interface, allows to avoid the manual de-construction and reconstruction of objects, or the use of MPI Types, making the code simpler and cleaner, and saving time in both the writing and debugging part of the implementation.

Distributed Compute Point Locations

This function uses the structure we just described, to run the same computation as *compute point locations*, but in a distributed setting.

The function returns the output of *compute point location*, as if it was called on the points geometrically inside the locally owned domain, with the addition of a list of the points for which the output is presented, the rank of the process originally owning that point, and an id for each point .

The idea behind a "distributed compute point locations" is the following:

- Use the bounding boxes to guess where each point lies
- Use send and receive for points lying on parts of the domain which are not locally owned
- Use compute locations on all the points which lie on the locally owned part of the domain

Input: A triangulation, global bounding box description, a list of points

Output: cells containing points located inside the locally owned part of the mesh, their transformed, and unique point ids

Use *global bounding boxes* to guess the owners of each point;
Call *compute point location* on the points which are probably local;
Send and receive points which have a single owner;
Send the output of *compute point location* and relative points for what resulted to be in ghost cells;
Call *compute point location* on the points which received and owned;
Send and receive points with multiple possible owners;
Call *compute point location* on points which might be owned;
Build output from all computed data;

Algorithm 5: A scheme for distributed compute point locations

A sketch of the algorithm follows

The practical implementation has some complications, for example:

- The tasks are either *communication* intensive or *computation* intensive; task spawning was used in an attempt to minimize communication overhead
- Merging the output of multiple calls to *compute point location* has a non trivial computational cost

Final results for this function are presented in section 3.5.3.

3.5 Benchmarks

3.5.1 Serial Baseline

We shall call the version of *compute_point_locations* present in deal.II version 8.5.1 "version 1" (from now on: v1). The initial benchmark 3.7 was run with random points on a square grid.[2]

Even though the growth is linear, computational cost is extremely high: taking more than 20 seconds to find 4000 points inside a grid of 4000 cells. For a deeper analysis, *callgrind* and *kcachegrind* were used; results are reported in Table 3.2.

This preliminary analysis results can be summarized in the following:

- Function's performance is mediocre
- The biggest computational burden lies in *tranform_real_to_unit_cell* i.e. $F_K^{-1}(p)$, which is called 10176 to classify 200 points: this can probably be avoided
- The second big cost lies on *find_active_cell_around_point*

[2]This preliminary result was obtained on a laptop running ArchLinux on an Intel(R) Core(TM) i7-6700HQ CPU @ 2.60GHz; afterwards the benchmarks were run on SISSA's cluster Ulysses, which runs on Intel(R) Xeon(R) CPU E5-2640 0 @ 2.50GHz.

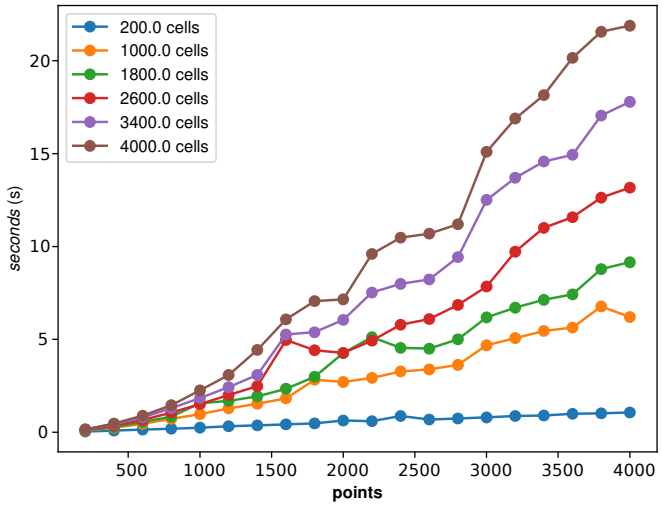


Figure 3.7: *compute_point_locations* v1: varying number of points

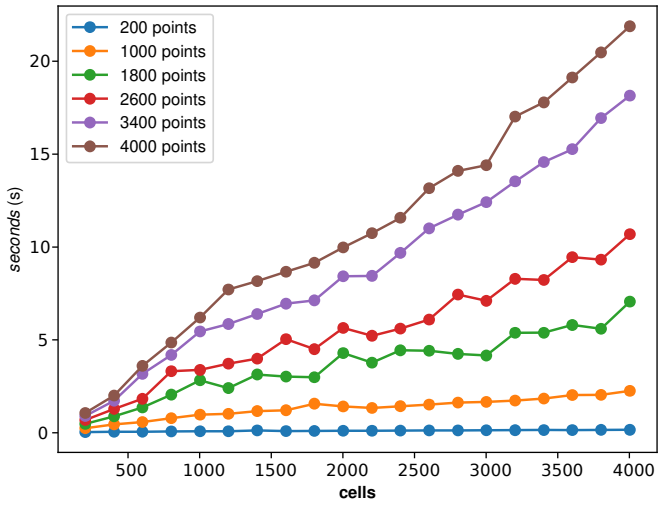


Figure 3.8: *compute_point_locations* v1: varying number of cells

Table 3.1: Kcachegrind analysis for v1 with 200 cells, 200 random points

CEst	CEst per call	Count	Callee
87.68	25372	10176	*tranform_real_to_unit_cell*
11.73	300388	115	*find_active_cell_around_point*
0.15	146937	3	dl runtime resolve avx
0.02	560	115	std::allocator⟨ std::vector⟨ Point⟨ 2 ⟩⟩⟩

Table 3.2: Kcachegrind analysis for v1 with 400 cells, 200 random points

CEst	CEst per call	Count	Callee
82.08	25499	13657	*tranform_real_to_unit_cell*
17.29	513026	143	*find_active_cell_around_point*
0.17	247442	3	dl runtime resolve avx
0.07	2135	143	std::allocator⟨ std::vector⟨ Point⟨ 2 ⟩⟩⟩

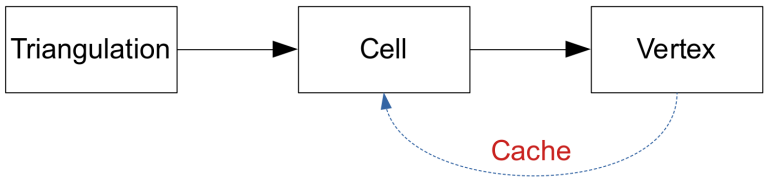


Figure 3.9: The DAG structure, with the additional direction provided by the Cache class.

Find active cell around point

As pointed out when describing Algorithm 2 the deal.II implementation of *find active cell around point* is quite slow but, because of its relevance to our problem we quickly report on the improvements brought by Dr. R. Gassmoeller and Prof. L. Heltai.

The initial improvements are due to Gassmoeller et al. [35], who implemented them in the ASPECT library [47, 10], and we ported then the code to the deal.II library:

- Adding a storing system, **GridTools::Cache**, to compute only once non DAG information
- Implementing an algorithm to guess which cell among the neighbours of v is most likely to be the one containing p.

There are many simple methods to guess which cell might contain a point, such as checking the distance from the point to the center. A better algorithm, based on the angle between the vector $p - v$ and the vectors going from each cell's center to v, was developed by Rene Gassmoeller et al. [35].

These improvements radically changed the performance of *find active cell around point*, with a great benefit for *compute point locations*.

The cache class Let c be a generic cell of a grid, let v be a vertex, we use the symbol $v \in c$ to state that v is one of the vertices of c. In a library with the classic DAG structure described in Figure 3.2, given the vertex v, the only way to find those cells c

for which $v \in c$, is to use a search on all cells. Running this operation, without further optimizations, results in two main drawbacks:

- for a simple operation on vertices the complexity $O(N)$, where N is the number of cells in the grid, is quite high,
- if the operation has to be repeated, e.g. when computing the coupling matrix C, it results in huge waste of computation as cells are visited multiple times, repeating the same operation with a waste of computational power.

As a solution we included a caching mechanism, in the class **GridTools::Cache**, where the association between vertices and cells is computed only once. While the initial operation of building the cache is slower than a single search, as memory has to be allocated and the whole mesh has to be traversed, the performance gain is usually apparent already at the second or third search.

3.5.2 Improvements for the Serial Code

The algorithm for compute point locations was improved in the following ways.

Calls to *tranform_real_to_unit_cell* The first problem is that *tranform_real_to_unit_cell* is **always** used to identify whether or not a point is inside a cell, even if it could be discarded using simpler methods, such as the distance from the cell's center.

In particular, given a cell K, let p_K be its center, and d_K the cell's diameter; then the following "distance check" can be used to avoid many calls to *tranform_real_to_unit_cell* when checking if the point p is inside. The *tranform_real_to_unit_cell* function is called only if:

$$||p - p_K|| < \frac{1}{2} d_K$$

This method reduces the number of calls to *tranform_real_to_unit_cell* cell by more than an order of magnitude, with a great impact on performance. We shall call the function with this improvement "version 2".

Looping on all the points The second bottleneck comes from the fact that the algorithm always loops on all remaining points: this number is potentially huge and, thus, even discarding them using the distance from the cell center is not sufficient.

Thanks to Subsection 3.5.1 the now improved speed of *find active cell around point*

makes it possible to use a completely different approach:

Data: A triangulation, a list of points
Result: cells containing points and their transformed

Initialize the vector with cells, points,qpoints;
for *p in points* **do**
  Find the active cell K around p and $q = F_K^{-1}(p)$;
  if *K in cells* **then**
    add p and q to the points in K;
  else
    Add K to cells;
    add p and q to the points in K;
  end
end
return cells, points, qpoints;

Algorithm 6: Compute Point Locations, version 3

This algorithm is much more elegant than the previous ones, and its cost lies almost entirely on *find active cell around point*, making it benefit greatly from the improvements of 3.5.1. Its performance is quite consistent and remains quite efficient even for unordered configurations of points.

Searching for K in the cell list has a high cost but, in many practical problems coming from mesh coupling, points are clustered, making the number of cell's they occupy low compared to the number of points.

Using different Containers In order to reduce the searching cost different containers were used instead of vectors; unordered maps and unordered multimaps were used, because of their constant $O(1)$ access time [97], depending on the hashing function; in this case the unique *active cell index* was used. The versions compared are:

- version 2
- version 3
- version 3 modified to check if the point is inside the last found cell (to take advantage of point's order, see Paragraph 3.5.2).
- version 3 which uses an unordered multimap and merges the different mapped values.
- version 3 which uses and unordered map and then creates an output vector.

As shown in 3.10 this results in a speed up of approximately 5%.

Mixing Both Approaches To improve the function's performance in the case of coupling, we make the assumption that p is probably in the last found cell, or close to it.

This leads to two algorithm changes:

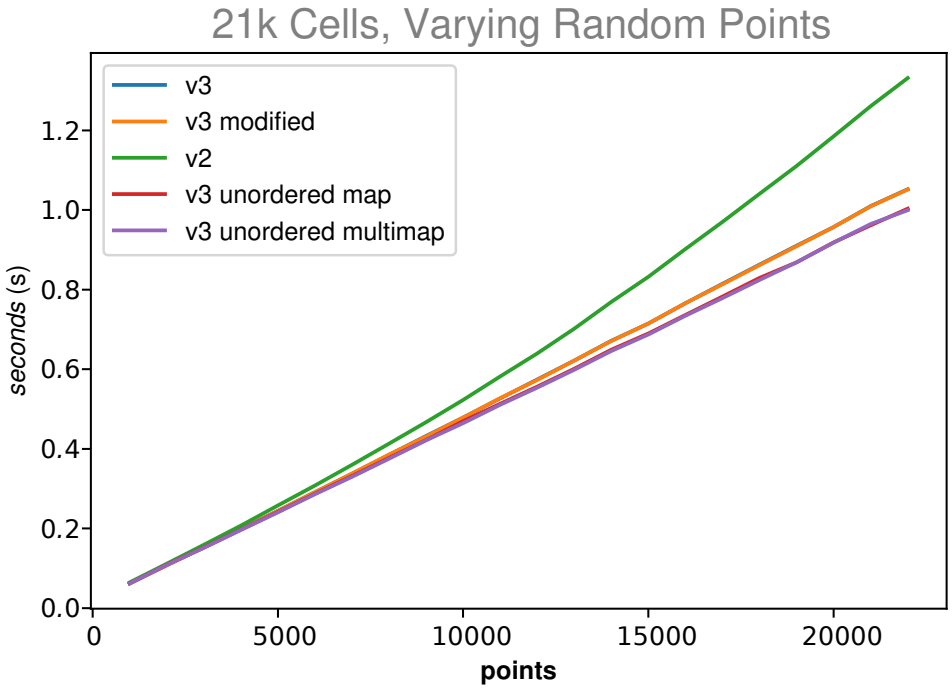


Figure 3.10: Container Comparison

1. Before calling the search function, check if the current cell coincides with the last one found.

2. Use the distance from the cell center, to evaluate if the last cell is a good hint cell for *find active cell around point*.

Passing a wrong hint cell to *find active cell around point* greatly reduces the performance, as the function looks for v, the closest vertex of the hint cell to p and checks if p is inside one v's neighbours; these checks waste computational power before calling the standard algorithm, slowing down the function. At the same time a simple test such as the distance test, represents only a small overhead with a random set of points, but it allows to safely use hint cells in any scenario.

2d Serial Benchmarks

Test in two dimensions were run with the following settings:

- a "Random Benchmark": random points, as shown in Figure 3.11a
- a "Clusterized Benchmark": points lying on a spiral with parametrization $p \to \frac{p}{2\pi}(\sin(p), \cos(p))$, with $0 < p < 2\pi$ (see Figure 3.11b)

If N_{cells} is the target number of cells and N_{child} the number of children obtained from the refinement of a single cell the cells were refined uniformly **floor** $\left(\log_{N_{child}}(N_{cells})\right)$

Data: A triangulation, a list of points
Result: cells containing points and their transformed

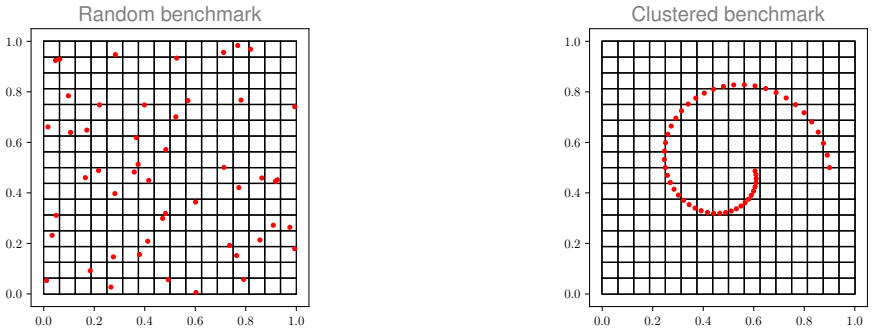
Initialize the vector with cells, points,qpoints;
Find the active cell K around p and $q = F_K^{-1}(p)$;
Compute the K's center p_K and diameter d_K;
for *p in points* **do**
  if $||p - p_K|| < d_K$ **then**
    Find the active cell with hint cell K;
  else
    Find the active cell;
  end
  if $K ==$ *last cell* **then**
    add p and q to the points in K;
  else
    Look for K in cells;
    Same as version 3;
  end
end
return cells, points, qpoints;

Algorithm 7: Compute Point Locations, version 4

times[3], and then further cells were refined reaching a total number of cells between N_{cells} and $N_{cells} + N_{child} - 1$.

To time the code the the timing tools offered by deal.II and deal2lkit [91] were used. Here we report the profiling for a clustered simulation on 100000 points and 50000 cells:

[3]It's the greatest number of possible uniform refinements without exceeding N_{cells}



(a) Random Benchmark (b) Clustered Benchmark

Figure 3.11: 2D benchmarks examples

```
   Version 2
   ------------------------------------------------------------
   Timer Name                 Global time (num calls)
   ------------------------------------------------------------
   Add new cell               0.04804 (399)
   add point to existing cell 0.5034 (9.96e+04)
   Bench v2                   114.1 (1)
   Loop on points of cell     106.9 (2.1e+07)
   Transform point            0.7632 (1.259e+05)
   find_active_cell           0.08292 (400)
   ============================================================
Version 3
------------------------------------------------------------
Timer Name                 Global time (num calls)
------------------------------------------------------------
Bench v3                   12.56 (1)
add new cell               0.002193 (400)
add point to existing cell 0.5158 (9.96e+04)
find if cell present       0.609 (1e+05)
find_active_cell           11.31 (1e+05)
============================================================
Version 4
------------------------------------------------------------
Timer Name                 Global time (num calls)
------------------------------------------------------------
Bench v4                   1.185 (1)
add new cell               0.002221 (399)
add point to existing cell 0.4949 (9.96e+04)
find if cell is present    0.002243 (399)
find_active_cell           0.03758 (4)
find_active_cell hint      0.5732 (1e+05)
============================================================
```

The time needed to add new cells and points it's the same among all versions, and this operation can not be optimized.

Remaining observations endorse what has been written in Subsection 3.5.2: a comparison between v3 and v4 shows that, using the distance to evaluate if the old cell is a reasonable hint, makes the finding process much faster while for v2 the cost of *find active cell around point* is low, but the time needed to loop on all points slows down the process.

For the random example, version 1 is used only in the simplest test, as it is extremely slow. The speedups depend on the problem complexity and, in our examples, we reached a speed-up of 236 in a clustered example, and one of 780 in an example with randomly distributed points.

Table 3.3: Clustered Benchmark: running times and speedups for different versions of compute point locations on 10^5 ordered points on a mesh with $5 \cdot 10^4$ cells

Points	Cells	v1	v2	speedup	v3	speedup	v4	speedup
10^5	$5 \cdot 10^4$	-	114.1	-	12.56	-	1.185	-
$5 \cdot 10^4$	$2 \cdot 10^4$	137.9	43.35	3.18	3.169	43.5	0.584	236
10^4	$2 \cdot 10^4$	26.96	7.67	3.51	0.6298	42.8	0.1233	219
10^4	$5 \cdot 10^3$	11.81	3.999	2.95	0.2818	41.9	0.1144	103

Table 3.4: Random Benchmark: running times and speedups for different versions of compute point locations on 10^5 random points on a mesh with $5 \cdot 10^4$ cells

Points	Cells	v1	v2	speedup	v3	speedup	v4	speedup
10^5	$5 \cdot 10^4$	-	8227	-	13.86	-	14.04	-
$5 \cdot 10^4$	$2 \cdot 10^4$	-	1879	-	3.451	-	3.58	-
10^4	$2 \cdot 10^4$	-	171.6	-	0.6645	-	0.6654	-
10^4	$5 \cdot 10^3$	231.6	86.06	2.69	0.2979	777	0.297	780

3D Serial Benchmarks

For 3D simulations, a spiral with parametrization $t \mapsto (0.4\cos(t) + 0.5, 0.4\sin(t) + 0.5, ht + 0.1)$ was used. Table 3.5 shows that v4 remains the is the best function in the clustered case, and its timing in the random case is comparable with v3, while v2 now becomes extremely slow with random points. Speed-ups are reported separately, as version 1 is too slow for the tests shown in Table 3.6.

In both clustered and random tests, the scaling is linear, but as shown in Figure 3.12, v3 is slow in comparison to the others. The speed ups reported in Table 3.6 show numbers reduced in comparison with the 2d case. In three dimensions the algorithm used in *find active cell around point* is slower because of the increased number of neighbouring cells.

Final 3D Benchmarks We have shown that different containers can improve the speed of about 5%. For this reason a new implementation of version 4, which uses an unordered map as container, was added. The tests were done in a 3D setting: the outer grid is a cube and the points are arranged in a spherical shape (see Figure 3.5).

Table 3.7 shows how, using unordered maps as containers gives a boost of about 5%

Table 3.5: Timings for random and clustered benchmarks of different versions of compute point locations on 10^5 random points on a three-dimensional mesh with $5 \cdot 10^4$ cells

Benchmark Type	v2 timing	v3 timing	v4 timing
Clustered	12.27	29.56	2.091
Random	8434	30.84	30.98

Table 3.6: Speed ups with respect of v1, for random and clustered benchmarks in 3D: different versions of compute point locations

Benchmark Type (points, cells)	v2 speedup	v3 speedup	v4 speedup
Clustered (10^4, 510^3)	-	1.77	**5.27**
Random (10^4, $5 \cdot 10^3$)	1.98	395	385

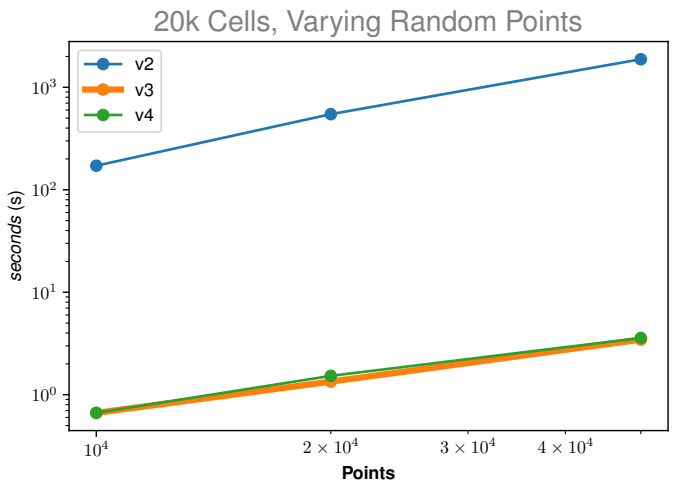


Figure 3.12: 3D Benchmark With Random Points

Table 3.7: v5 with unordered map comparison on random benchmark

(Points,Cells)	v5	v5 with unordered map	Speed Up
(3431, 4096)	0.1797	0.1427	1.26
(27967, 32768)	4.082	3.927	1.04
(226415, 32768)	33.53	32.27	1.04

in efficiency.

3.5.3 Parallel Results

In our initial test we used a cube and a sphere, as in Figure 3.5, partitioned among a varying number of processes.

```
================================================

      TimeMonitor results over 12 processors

Timer Name                          MeanOverProcs
------------------------------------------------
Compute and merge other points      0.1 (1)
Compute mesh predicate box          0.001423 (1)
Constructing points to be sent      0.0001599 (1)
Dcploc, cube 4 sphere: 3            0.2365 (1)
Merge ghost                         0.07784 (1)
Using BBoxes to guess owner         0.001361 (1)
all gather for bboxes               0.001553 (1)
some_to_some ghost part             0.01146 (1)
some_to_some other points           0.005509 (1)
some_to_some owned points           0.01564 (1)
================================================
```

The profiling on 12 processes shows how *Compute and merge other points* takes most of the time: this is the section of the code were compute point locations is run on points received from other processes and then added to the current output. This operation is slowed down because, after calling *compute point locations*, an additional search on cells is needed to merge the output.

Making a weak scaling test for this problems is complicated, as depending on the mesh distribution the number of points which needs to be computed or communicated varies. A preliminary scaling result is shown in Figure 3.13: in this case *compute point locations* was run on a total of $10k$ points; as we can see there are some scaling problems.

Using unordered maps To improve scaling we used another container, an unordered map, and of version 6 of *compute point locations*.

The test was run using a cube and a sphere inside it, as shown in figure 3.5. The following profiling results show that the algorithm is now well-balanced. Figure 3.14 shows the preliminary scaling results: clearly, for this particular use-case, unordered multimaps outperform vectors.

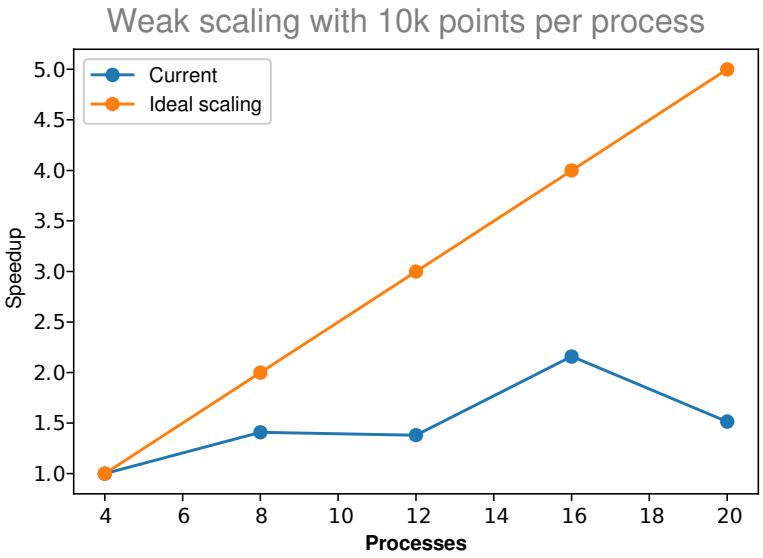


Figure 3.13: Weak Scaling test

```
TimeMonitor results over 20 processors

Timer Name                              MinOverProcs     MeanOverProcs
------------------------------------------------------------------------
Compute and merge other points          0.03702 (1)      0.03702 (1)
Compute mesh predicate box              0.00342 (1)      0.003455 (1)
Constructing points to be sent          0.0001049 (1)    0.0001074 (1)
Dcploc, cube 5 sphere: 4                0.07959 (1)      0.07962 (1)
Merge ghost                             0.000176 (1)     0.003586 (1)
Using BBoxes to guess owner             0.00106 (1)      0.001089 (1)
all gather for bboxes                   0.002365 (1)     0.002371 (1)
some_to_some ghost part                 0.000526 (1)     0.009646 (1)
some_to_some other points               0.001664 (1)     0.001669 (1)
some_to_some owned points               0.0008979 (1)    0.0009018 (1)
```

Notice that the distribution of the sphere cells among processes, and the points distribution, change with every new number of processes uses: his affects the performance of *compute point locations* and it's behind the apparent super-linear scaling. The plot clearly shows that the algorithm is scaling well at least with up to a few dozens of cores.

3.6 Further developments

A number of improvements were introduced thanks to efficient search trees algorithms implemented in the boost library [24]. Among the others, the following functions were added to **GridTools::Cache**:

1. **Cache.get_vertex_kdtree()**: generates a kdtree containing the vertices of the triangulation.

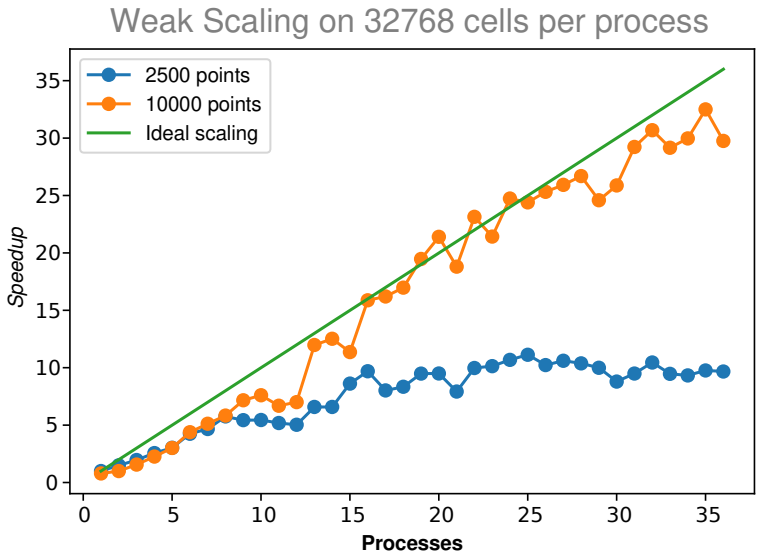


Figure 3.14: Weak Scaling test

2. **Cache.get_cell_bounding_boxes_rtree()**: generates an rtree of bounding boxes, where each bounding box covers a cell.

3. **Cache.get_covering_rtree()**: generates an rtree of bounding boxes which, given a point, allows to identify the owner or the possible owners of the cell containing the point.

These additions resulted in the following improvements:

- The kdtree spanning all vertices can be used to speed up the search for vertices in *compute point locations*, if no candidate is found.

- The function *compute point locations* benefits from a tree spanning the cells of the mesh because, instead of using the diameter, the bounding boxes can be used to test if a point belongs to a cell. This operation is faster, and usually returns less false positives.

- Through the intersection of trees spanning the meshes it is possible to generate more efficiently the coupling matrix.

- The function *distributed compute point locations* benefits from a tree describing the "ownership", implementing a fast search option to find processes which might own a point.

- The rtree describing the "ownership" can be used to generate a good local description of the mesh, without the need for a dedicated computation. The only drawback of this approach is that, currently, the only function implemented in the boost library returns a single, global bounding box. Multiple boxes can be obtained with some technical tricks, i.e., implementing a visitor which travels the tree and requesting all boxes after a certain number of splits.

Chapter 4

Fiber reinforced materials

In this chapter we show an application of non-matching coupling methods applied to a continuum mechanics problem: the study of fiber reinforced materials. We describe a full 3D model, where both the fibers and the underlying material are modelled as three-dimensional meshes, and then study a reduced model, where fibers are approximated with tubular neighbourhoods of one-dimensional meshes.

Composite materials are the prototypical example of problems requiring the coupling between multiple, complex geometries. During the past fifty years, the interest in composite materials flourished multiple times; at first it sparked for the applications of composites to new materials in aerospace engineering [45], civil engineering [71], materials science [14, 68], and other engineering fields.

During the nineties, the increasing importance of biomechanics in life sciences lead to the development of numerous models describing, e.g., arterial walls [55], soft tissues [53], and muscle fibers [56].

Recent years saw the rise of new application fields, such as the study of natural fiber composites [81], and engineering methods to accurately recover the three-dimensional structure of a material sample, e.g., [58, 59].

From the first studies on composites, it has been clear that their properties are strongly dependent on their internal structure: the volume ratio between each component, the orientation, the shape, all contribute substantially to the material's properties [45, 46]. One of the most wide-spread and significant example of composites is that of Fiber Reinforced Materials (FRMs), where thin, elongated structures (the fibers) are immersed in an underlying isotropic material (the elastic matrix).

We may separate the approaches used to study FRMs into two broad groups: i) "*homogenization methods*", which study a complex inhomogeneous body by approximating it with a fictitious homogeneous body that behaves *globally* in the same way [101], and ii) "*fully resolved*" methods, which use separate geometrical and constitutive descriptions for the elastic matrix and the individual fibers.

As examples of analytical "*homogenization methods*" we recall the rule of mixtures [44] and the *empirical* Halpin-Tsai equations [43], used to study a transversely isotropic unidirectional composite, where fibers are uniformly distributed and share the same orientation. The development of homogenization theory led, in recent years, to more complex models, e.g., [54, 12].

More intricate homogeneization approaches rely on numerical methods to provide a "cell" behaviour, which is then replicated using periodicity, using, e.g., the Finite Ele-

ment Method (FEM) [1, 76, 65], Fourier transforms [74, 75, 32], or Stochastic Methods [66].

The fundamental limit of all "*homogenization methods*" approaches is the impossibility of adapting them to study composites with little regularity. In these cases, the different phases are typically modeled separately, as a continuum. This approach began with Pipkin [82] on two dimensional membranes, and was then expanded to three-dimensional examples by others (see, e.g., [61] for a detailed bibliography).

Fully resolved methods allow richer structures, but require a high numerical resolution, especially when material phases have different scales. The complex meshing and coupling often result in an unbearable computational cost, limiting the use of these methods.

The purpose of this chapter is to introduce a new approach which is fit for materials that have *intermediate* properties, i.e., they posses no particular regularity, and are made by a relatively high number of fiber components. We propose an FRMs model inspired by the Immersed Boundary Method (IBM) [80], and by its variational counterparts [22, 48, 50, 88, 51], where the elastic matrix and the fibers are modeled independently, and coupled through a non-slip condition. We aim at providing an efficient numerical method for FRMs that allows the modeling of complex networks of fibers, where one may also be interested in the elastic properties of single fibers, without requiring the resolution of the single fibers in the background elastic matrix. From the computational point of view, this approach allows the use of two independent discretizations: one describing the fibers, and one describing the *whole domain*, i.e., both the elastic matrix and the fibers. A distributed Lagrange multiplier is used to couple the independent grids, following the same spirit of the finite element immersed boundary method [20, 22], separating the Cauchy stress of the whole material into a background uniform behavior and a *excess* elastic behavior on the fibers.

We begin this chapter introducing the tools needed for our Problem: in Section 4.1 we introduce the basic tools of continuum mechanics which are needed for the description of the composite materials, we then introduce hyperelastic materials in Subsection 4.1.2, and report some useful properties of linearized elastic equations in Subsection 4.1.3. In Section 4.2 we recall some notion of differential geometry, these tools are needed for Subsection 4.2.1, where we introduce the tubular neighbourhoods of curves, and Subsection 4.2.2, where we describe some useful coordinate systems for the description of the gradient on tubular neighbourhoods.

Section 4.3 introduces the classical fully resolved model of a collection of fibers immersed in an elastic matrix. For simplicity, we do not include dissipative terms, and restrict our study to linearly elastic materials. The problem is then reformulated exploiting classical results of mixed finite element methods (see, e.g., Chapter 4 of [19]), following ideas similar to those found in [21], proving that both the continuous and discrete formulations we propose are well-posed with a unique solution.

The use of a full three-dimensional model for the fibers still results in high computational costs; the obvious simplification is to approximate the fibers with one-dimensional structures. This approach is non-trivial because it is not possible to consider the restriction of a Sobolev function defined on a three-dimensional domain to a one-dimensional domain. A possible solution involves the use of weighted Sobolev spaces, combinded with graded meshes [30, 29] but, if the number of fibers is large, graded meshes may still be too computationally intensive. In Section 4.4, we propose and analyze an alternative solution, where additional modellistic assumptions enable a $3D-1D$ coupling that relies

on local averaging techniques. A similar procedure is used in [49] to model vascularized tissues. To conclude, we validate our thin fiber model in Section 4.5.

4.1 Basic notions of elasticity

Starting from some basic definitions, we now introduce kinematic description of the main balance laws needed for the mathematical description of linear elasticity.

4.1.1 Balance Equations

With $\mathrm{Lin}(n)$, where $n \in \mathbb{N}$, we denote the vector space of all *linear applications* of $\mathbb{R}^n$ in $\mathbb{R}^n$. With $\mathrm{Lin}^+(n)$ the subspace of $\mathrm{Lin}(n)$ of applications with positive determinant. We denote the subspace of all *symmetric applications* in $\mathbb{R}^n$ with $\mathrm{Sym}(n)$.

When the dimension n of the space is understood, we shall often omit the script "(n)".

An important subset of the linear transformations set is the *orthogonal group*:

$$\mathrm{O}(n) \coloneqq \{O \in \mathrm{Lin}(n) \colon OO^T = I(n)\},$$

where $I(n)$ is the *identity matrix* in $\mathbb{R}^n$.

In mechanics we are particularly interested to the subgroup of $\mathrm{O}(n)$ containing applications with positive determinant, as in dimensions $n = 2$ and $n = 3$ it coincides with rigid rotations; the *orthogonal group* is defined as:

$$\mathrm{SO}(n) \coloneqq \mathrm{O}(n) \cap \mathrm{Lin}^+(n).$$

Bodies have an intrinsic property: they occupy a region of space. This property allows to define a body *reference configuration* as a regular subset Ω of the Euclidean space $\mathbb{E}^3$.

To study the body's deformation we define a vector field ϕ, called *deformation*:

$$\phi \colon \Omega \to \mathbb{R}^3,$$

with the deformed configuration of the body being $\phi(\Omega)$.

Remark. *It is often necessary to include an explicit dependency on the time t for the deformation, which becomes*

$$\phi \colon [0, T] \times \Omega \to \mathbb{R}^3,$$

with T a positive constant, and $[0, T]$ a time interval.

A point can be described by its *Lagrangian coordinate* $p \in \Omega$, and by its *Eulerian coordinate* $x = \phi(p)$.

We indicate the *deformation gradient* with $F \coloneqq \nabla\phi$. This value describes locally the volume after deformation per unit original volume, and we assume $J \coloneqq \det F > 0$, i.e., $F(p) \in Lin^+$.

The *displacement vector* of a point $p \in \Omega$ is defined as:

$$u(p) \coloneqq \phi(p) - p,$$

implying

$$F = I + \nabla u.$$

Let ρ_0 be the mass density in the reference configuration, and let ρ be the mass density in the deformed configuration. Considering bodies with a continuous distribution of mass and imposing the *conservation of mass* for each (measurable) subset of Ω, leads to the equation:

$$\rho J = \rho_0.$$

Internal contact forces are described by the following axiom:

Axiom 1 (Cauchy's hypothesis). *Let $S \subset \Omega$ be an oriented surface inside Ω. There exists a surface force density $s(n, x)$, with positive unit normal n at $x \in S$, describing the force per unit area exerted across S upon the material on the negative side of S by the material on the positive side.*

This leads to the following statement for the conservation of linear momentum in Eulerian form: for every regular subset P of Ω:

$$\int_P \rho b + \int_{\delta P} s(n) = 0,$$

where b is the density of the external force per unit mass.

A fundamental milestone in continuum mechanics is Cauchy theorem [40], which states that $s(n, x)$ is *linear* in n.

Theorem 4.1.1. *Let (s, b) be a system of forces for a body Ω. Then a necessary and sufficient condition for the momentum balance laws to be satisfied is that there exists a spatial tensor field T, called the* Cauchy stress*, such that:*

- *for every unit normal vector n*
$$s(n) = Tn$$
- $T \in \mathrm{Sym}(n)$
- *T satisfies the conservation of linear momentum:*
$$\operatorname{div} T + \rho b = 0.$$

The Cauchy stress tensor describes the contact force per unit area in the deformed configuration and, in the case of elastic body, it is assumed to depend on the deformation gradient:

$$T = T(x, F(x)).$$

The same quantity can be represented in the reference configuration using the *first Piola-Kirchhoff stress tensor* P, defined as:

$$P = JF^{-1}T.$$

4.1.2 Hyperelasticity

A common strategy to simplify the study of elastic materials, is to assume the internal dissipation to be zero during any admissible motion: the second law of thermodynamics becomes an equality, and no heat is produced. This idea is formalized in the concept of perfect elasticity: a material is said to be perfectly elastic if it does not produce entropy

when deformed [100]. One such material is called hyperelastic if there exists a strain energy density, called Ψ, depending only on the deformation gradient F.

Starting from the second law of thermodynamics, it is possible to prove that the constitutive equation of a hyperelastic material can be expressed, at every point of the body, as:

$$P_{ij}(F) = \frac{\partial \Psi}{\partial F_{ij}}(F), \tag{4.1}$$

where P is the First Piola-Kirchhoff stress.

This simplified model is appropriate, for example, to describe the first soft material that has been extensively studied, especially by Treolar [99]: rubber. Rubber materials can maintain an elastic behaviour even when subjected to large deformations, e.g., an extension ratio of 3.0 as shown in Figure 4.1.

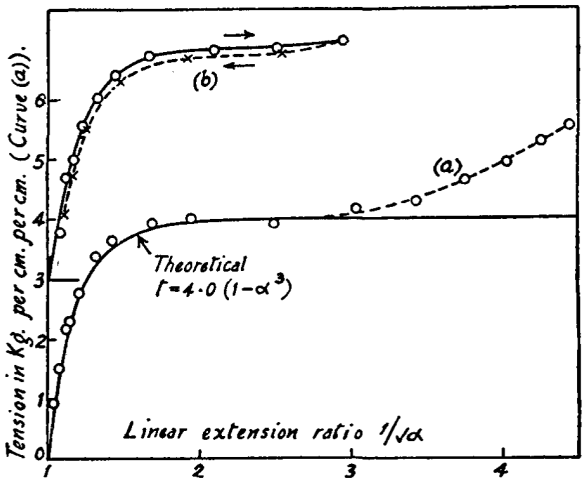


Figure 4.1: Two dimensional extension on S-rubber from the original article [99]; curve a) show the points obtained in the experiment, curve b) the agreement between the two curves, when increasing and reducing the extension.

Using the constitutive restrictions from continuum mechanics, it is possible to detail the dependence of Ψ on F. To avoid the possibility of an infinite deformation of the material with a finite amount of energy, Ψ should satisfy the *non-degeneracy axiom*:

$$\begin{cases} \Psi(F) \to +\infty & \text{for } \det(F) \to 0^+, \\ \Psi(F) \to +\infty & \text{for } \sqrt{\operatorname{tr}(F^T F)} + \sqrt{\operatorname{tr}(F^{-T} F^{-1})} \to +\infty. \end{cases}$$

A material is isotropic if its energy is independent of the reference frame:

$$\Psi(F) = \Psi(QF) \quad \text{for every } F \in Lin(3), \text{ and for every } Q \in SO(3),$$

Recall the right Cauchy-Green tensor is defined as:

$$C = F^T F.$$

It is possible to prove that the strain energy density can be expressed as a function of the principal invariants of C:

$$\Psi = \Psi(I_1, I_2, I_3),$$

where

$$I_1 = \operatorname{tr}(C) \qquad I_2 = \frac{(\operatorname{tr}(C))^2 - \operatorname{tr}(C^2)}{2} \qquad I_3 = \det(C).$$

By making additional hypotheses on the material, it is possible to find more precise formulations for Ψ. For example we can consider the Mooney-Rivlin model [73, 86], which describes an incompressible model, i.e., $\det F = 1$, in this case the Ψ can be expressed in the following form:

$$\Psi(F) = C_{10}(I_1 - 3) + C_{01}(I_2 - 3),$$

where C_{10} and C_{01} are two material parameters. For this model the shear modulus μ can be expressed as:

$$\mu = 2(C_{10} + C_{01}).$$

A special case of this model is the neo-Hookean model, in which $C_{01} = 0$, see [73].

4.1.3 Linear elasticity

The most widely used model for elasticity is the linear elastic model; this theory is appropriate when considering small deformations, and can be obtained linearizing the equations proposed in Section 4.1.2.

We now report some widely known results, for more details see [41] and Section 4.3 of [69].

When we consider Equation 4.1 near $F = I$, its value is governed by the linear transformation defined as:

$$\mathbb{C}_{ijkl} \coloneqq \frac{\partial}{\partial F_{ij}} P_{kl}(I) = \frac{\partial^2}{\partial F_{ij} \partial F_{kl}} \Psi(F) \bigg|_{F=I},$$

called the **elasticity tensor**.

The elasticity tensor $\mathbb{C}$ has a number of properties, such as being symmetric and invariant under the symmetry group of the material, or that $\mathbb{C}[W] = 0$ for every $W \in \operatorname{Skw}$, the subspace of skew matrices:

$$\operatorname{Skw} \coloneqq \{W \in \operatorname{Lin} \colon W = -W^T\}$$

Theorem 4.1.2. *Assume that the material is isotropic. Then there exists scalars λ and μ, called **Lamè moduli**, such that:*

$$\mathbb{C}[E] = 2\mu E + \lambda(\operatorname{tr}(E))I,$$

for every $E \in \operatorname{Sym}$

Theorem 4.1.2 is particularly important as, given the gradient ∇u of the displacement, it allows to write the constitutive law of the material as:

$$\mathbb{C}\nabla u := 2\mu Eu + \lambda(\operatorname{div} u)I,$$

where $Eu \coloneqq \frac{\nabla u + \nabla u^T}{2}$ is the symmetric part of the gradient.

An important property of elastic tensor is the **strong ellipticity**: there exists a positive constant $c > 0$ such that:

$$\mathbb{C}_{ijkl} v^i v^j w^k w^l \geq c\|v\|^2 \|w\|^2,$$

for all vectors $v, w \in \mathbb{R}^n$, where n is the dimension of the space we are considering (typically 3).

In the case of linear elasticity there exists a characterization of this property:

Theorem 4.1.3. *Let $\mathbb{C}$ be an isotropic an homogeneous elasticity tensor with Lamè moduli λ and μ. Then*

$$\mathbb{C} \text{ is strongly elliptic} \Leftrightarrow \mu > 0,\ 2\mu + \lambda > 0.$$

Strong ellipticity is a common and important property for elastic models; in this thesis we chose, for simplicity, the linear elastic model for our numerical simulations. Finally we describe **pointwise stability**, a stronger property than ellipticity: $\mathbb{C}$ is pointwise stable if, for every symmetric v_{ij}

$$\mathbb{C}_{ijkl} v^{ij} v^{kl} > 0.$$

It can be proven that a linear elastic operator is stable if and only if:

$$\mu > 0,\ \frac{2}{3}\mu + \lambda > 0.$$

4.2 Basic notions of differential geometry

The description of fiber reinforced materials, and their approximation as tubular neighbourhoods, requires some basic notions of differential geometry, which we briefly collect here; for a more detailed reference see, e.g., [60].

In the second part of this Section, we introduce the description of the Laplacian using coordinates on the curve, which is mostly used in physics-related fields for the description of particle trajectories.

Consider Γ, a one-dimensional curve immersed in $\mathbb{R}^3$; let $I \subset \mathbb{R}$ be a finite interval, and $\omega \colon I \to \Gamma$ be the arclength parametrization of Γ.

Assuming the curve is C^3, at each point $x \in \Gamma$, there exists an $s \in I$ such that $x = \omega(s)$; and the curve parametrization defines a base for $\mathbb{R}^3$, called the Frenet trihedron:

$$\begin{aligned} t(s) &= \frac{d\omega(s)}{ds} \\ n(s) &= \frac{dt(s)}{ds} \Big/ \left\| \frac{dt(s)}{ds} \right\| \\ b(s) &= t(s) \times n(s). \end{aligned}$$

Here t is the vector tangent to the curve at x, while n and b generate the plane orthogonal to the curve at x; for an example see Figure 4.2.

To describe the curve Γ it is useful to introduce the notions of *curvature*:

$$\kappa(s) \coloneqq \left\| \frac{d^2\omega(s)}{ds^2} \right\|,$$

and *torsion*:

$$\tau(s) \coloneqq -\left(\frac{d}{ds} b(s) \right) \cdot n(s).$$

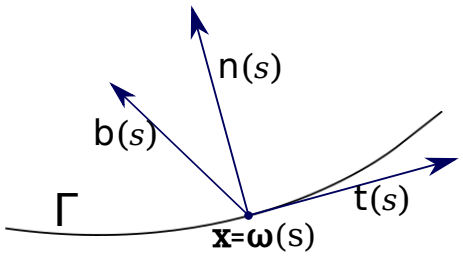


Figure 4.2: The Frenet trihedron.

A curve with torsion identically zero is called *planar curve*, as it can be embedded inside a plane.

To describe the evolution of curvature and torsion along the curve, we use the so-called *Frenet-Serret* formulas:

$$\begin{aligned} \frac{dt(s)}{ds} &= \kappa(s)n(s), \\ \frac{dn(s)}{ds} &= -\kappa(s)t(s) + \tau(s)b(s), \\ \frac{db(s)}{ds} &= -\tau(s)n(s). \end{aligned} \tag{4.2}$$

4.2.1 Tubular neighbourhoods

The concept of tubular neighbourhoods is one of the basic tools in differential geometry; the basic properties of a tubular neighbourhoods which we report here are often shown as examples of exercises in multidimensional real analysis and multivariable calculus books, e.g., [31].

Given a radius $a \in \mathbb{R},\ a > 0$, and a curve Γ in $\mathbb{R}^3$, we define the tubular neighbourhood of Γ of radius a, as the set of all points in $\mathbb{R}^3$ at distance at most a from the curve:

$$\Omega_a \coloneqq \{x \in \mathbb{R}^3 \colon \operatorname{dist}(x, \Gamma) \leq a\}.$$

If Γ is regular enough, and has an arclength parametrization ω, then Ω_a can also be defined as:

$$\Omega_a \coloneqq \{\omega(s) + n(s)\lambda_1 + b(s)\lambda_2 \colon s \in I; \lambda_1, \lambda_2 \in \mathbb{R} \ s.t.\ \sqrt{\lambda_1^2 + \lambda_2^2} \leq a\}.$$

We require $\partial\Omega_a$ to be non-intersecting. This hypothesis is satisfied locally if ω is either C^3 regular or a C^2 regular plane curve, and the radius is chosen such that

$$a < \max_{s \in I} \frac{1}{\kappa(s)}.$$

For a plane curve, the osculating circle has radius $\max_{s\in I} \frac{1}{\kappa(s)}$, and can be seen as the intersection of normals for infinitely closed points; because a is smaller than the radius, perpendicular disks do not intersect.

If the torsion τ is non-zero, then we can repeat the argument in space using the osculating sphere, which has radius:

$$R(s) = \sqrt{\frac{1}{\kappa(s)^2} + \frac{1}{(\kappa'(s)\tau(s))^2}},$$

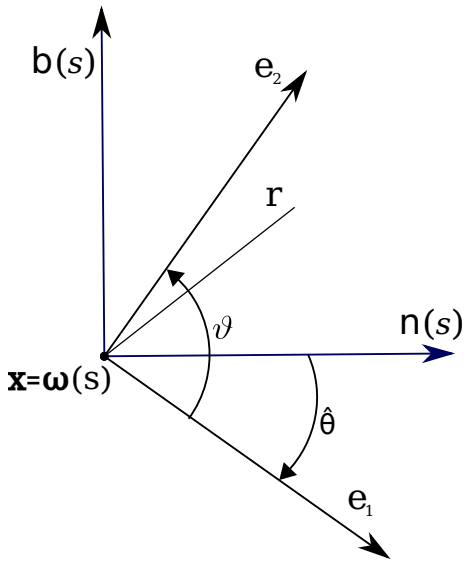


Figure 4.3: Frenet-Serret and the rotated coordinate systems.

after noticing that $R(s) \geq \frac{1}{\kappa(s)}$.

To describe the tubular neighbourhood, we introduce the following definition: for all $x = \omega(s) \in \Gamma$, we define $D_a(x)$ as the two-dimensional disk perpendicular to Γ, with radius a, centered at x:

$$D_a(x) \coloneqq \{\omega(s) + n(s)\lambda_1 + b(s)\lambda_2 \colon \lambda_1, \lambda_2 \in \mathbb{R}, \sqrt{\lambda_1^2 + \lambda_2^2} \leq a\}.$$

4.2.2 A reference system on one-dimensional fibers

The use of Frenet-Serret coordinates is used in physics to study wave propagation, optics and particle trajectories [63].

To simplify our computations we shall rotate the orthogonal vectors, as proposed by Tang [98], and then transform the plane $< n, b >$ using polar coordinates, as done in [90].

Recall that each point x of a the tubular neighbourhood Ω_a can be written as

$$x = \omega(s) + x_0 n(s) + y_0 b(s),$$

with $x_0, y_0 \in [-a, a]$, where x_0 and y_0 are the distances from x to the centerline Γ, along the normal and binormal directions. Using the Frenet-Serret formulas 4.2, we can compute the metric tensor:

$$dx \cdot dx = dx_0^2 + dy_0^2 + [(1 - \kappa x_0)^2 + (x_0^2 + y_0^2)\tau^2]ds^2 + 2\tau(x_0 dy_0 - y_0 dx_0)ds.$$

The metric is non-orthogonal because of the mixed term $2\tau(x_0 dy_0 - y_0 dx_0)ds$.

To obtain an orthogonal metric, define θ as an angle solving the following differential equation:

$$\begin{aligned} \frac{d\theta(s)}{ds} &= \tau(s), \\ \theta(0) &= \hat{\theta}. \end{aligned}$$

where the initial value $\hat{\theta}$ can be fixed arbitrarily. Define the rotated vectors:

$$e_1(s) = \cos(\theta(s))n(s) - \sin(\theta(s))n(s) \tag{4.3}$$
$$e_2(s) = \sin(\theta(s))n(s) + \cos(\theta(s))n(s), \tag{4.4}$$

each point $x \in \Omega_a$ can be written in the new system of reference as:

$$x = \omega(s) + x_1 e_1 + y_1 e_2.$$

The metric of the the coordinates (t, e_1, e_2) is orthogonal:

$$dx \cdot dx = dx_1^2 + dy_1^2 + [1 - \kappa(x_1 \cos(\theta) + y_1 \sin(\theta))]^2 ds^2.$$

We assume fibers to be cylindrical in their reference configuration; a natural developement for our model is for fibers to be axisymmetric. This suggests the use of a coordinate system where the orthogonal plane to a point x of the curve, is described through polar coordinates.

Consider the coordinate system (r, ϑ, s), where (r, ϑ) are defined so that:

$$x_1 = r\cos(\vartheta) \tag{4.5}$$
$$y_1 = r\sin(\vartheta). \tag{4.6}$$

This coordinate system is shown in Figure 4.3; the metric for the coordinate system (r, ϑ, s) is:

$$dx \cdot dx = dr^2 + r^2 d\vartheta^2 + (1 - \kappa r \cos(\vartheta - \hat{\theta}))^2 ds^2.$$

This allows to write the gradient in the coordinates (r, ϑ, s):

$$\nabla \cdot := \frac{d\,\cdot}{dr} e_1 + \frac{1}{r}\frac{d\,\cdot}{d\vartheta} e_2 + \frac{1}{1 - \kappa(s) r \cos(\vartheta - \hat{\theta})}\frac{d\,\cdot}{ds} t.$$

4.3 Three-dimensional model

Many bi-phasic materials present a relatively simple fiber structure but result in a very intricate elastic matrix. Consider, for example, Figure 4.4: constructing a discretization grid for the fibers themselves maybe simple enough, but building a fully resolved grid for the *surrounding* elastic matrix, in this case, may require eccessive resolution, and result in a computationally hard problem to solve. We wish to describe a new approach, where we substitute the complex mesh needed for the elastic matrix with a simple one describing the whole domain, and overlap the fiber structure independently with respect to the background grid, coupling the two systems via distributed Lagrange multipliers.

4.3.1 On the coupling bond

To introduce our model, it is useful to briefly report some well-known properties of FRMs (for more details see [28], Chapter 16).

The mechanical characteristics of a FRM do not depend only on the properties of the two phase materials, but also on the degree to which an applied load is transmitted from the elastic matrix to the fiber. The transmission depends on:

Figure 4.4: An example of a fiber structure for which the mesh generation for the fibers would be trivial, but the resulting three-dimensional elastic matrix would be much more expensive to resolve in full.

- The fiber-matrix bond strength τ_c, i.e., the degree to which an applied load is transmitted to the fibers through the interfacial bond between the fiber and the matrix phases

- The fiber length, because the matrix-fiber bond greatly reduces at the extremities; see Figure 4.5.

The fact that, at the edges of the fiber, the load transmittance reduces, leads to the definition of a **critical fiber length** l_c, dependent on both the fiber radius a and τ_c:

$$l_c \propto \frac{a}{\tau_c}.$$

If the fibers are shorter than this critical value l_c, the stress transference becomes negligible, while longer fibers generally result in better reinforcement for the material [28]. Fibers for which the length l is $l \gg l_c$ are called **continuous fibers**.

To simplify our model we consider continuous fibers with a "perfect bond" between the two phases, leading to the same deformation of both materials. Mathematically, the bond is imposed using a **non-slip** condition between the two phases.

For the simplified one-dimensional model, we introduce the average non-slip condition using the integral average; in the case of thin fibers, assuming the solution is piecewise continuous, this can be seen as an approximation of the classical non-slip condition.

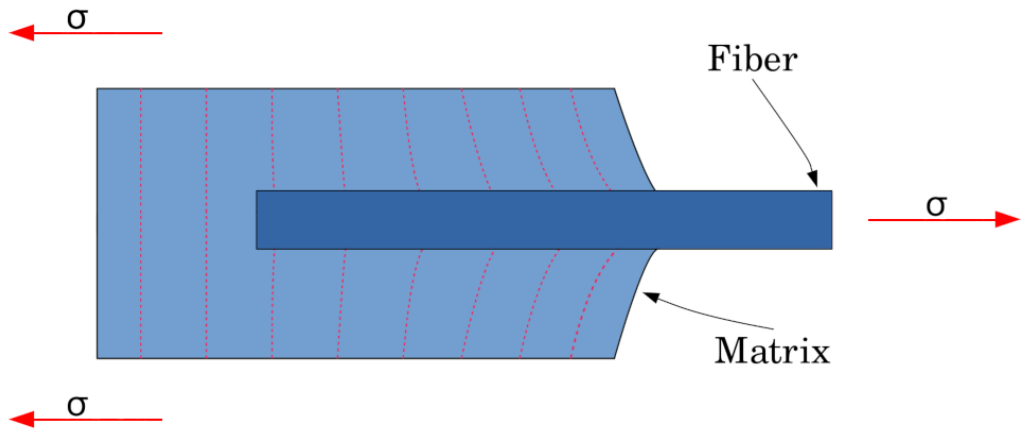
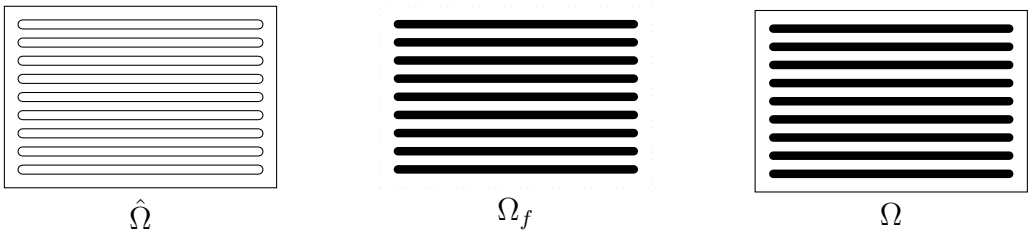


Figure 4.5: Deformation pattern around a fiber subjected to a load.

4.3.2 Problem formulation

As a model bi-phasic material, we consider a linearly elastic fiber reinforced material. To simplify the treatment of the problem, in this work we limit ourselves to the quasi-static small strain regime. The extension to finite strain elasticity and dynamic problems does not present additional difficulties and is going to be be the subject of a future work.

To describe the composite we use a connected, bounded, Lipschitz domain $\Omega \subset \mathbb{R}^d$ of dimension $d = 3$, composed of a fiber phase $\Omega_f \subset \Omega$, which we assume to be a Lipschitz domain, and an elastic matrix phase $\hat{\Omega} := \Omega \setminus \Omega_f \subset \mathbb{R}^d$. The fiber phase is obtained as the union of the $n_f \in \mathbb{N}$ fibers, each described through a connected, Lipschitz domain; for simplicity we shall fix $n_f = 1$. The results hereby reported can be easily extended to the case of a finite number of fibers.



Example of a two-dimensional section of an FRM with uniformly oriented fibers.

Remark. *The results of this section hold for a general domain Ω_f, union of multiple components with the required regularity. The property of the fibers of being thin, elongated, structures, is only needed for the model of Section 4.4, and plays no role in this section, where they could be considered "elastic inclusions".*

Given a displacement field $u \colon \Omega \to \mathbb{R}^d$, representing a deformation from the equilibrium configuration, the corresponding stress tensor on Ω can be expressed using the stress-strain law [40]:

$$S[u] = \mathbb{C}\nabla u,$$

where $\mathbb{C}$ is a symmetric 4^{th} order tensor that takes the form:

$$\mathbb{C} = \begin{cases} \mathbb{C}_\Omega & \text{in } \hat{\Omega}, \\ \mathbb{C}_f & \text{in } \Omega_f. \end{cases} \tag{4.7}$$

Here $\mathbb{C}_\Omega$ and $\mathbb{C}_f$ are assumed to be constant over their respective domains, and represent the elasticity tensors of the elastic matrix and of the fibers. The classical formulation of static linear elasticity can be thought of as a force balance equation (see, for example, [40]):

Problem 12 (Classic Strong Formulation)**.** *Given an external force density field b, find the displacement u such that*

$$\begin{aligned} -\operatorname{div}(\mathbb{C}u) &= b \qquad \text{in } \Omega, \\ u &= 0 \qquad \text{on } \partial\Omega. \end{aligned} \tag{4.8}$$

Due to the piecewise nature of $\mathbb{C}$, it is natural to reformulate Problem 12 into a variational or weak formulation. Given a subset D of $\partial\Omega$, we introduce the notation for the subspace of the Sobolev space $H^1(\Omega)$ with functions vanishing on a subset D of the boundary $\partial\Omega$:

$$H^1_{0,D}(\Omega)^d \coloneqq \{v \in H^1(\Omega)^d \colon v\big|_D = 0\},$$

with norm $\|\cdot\|_V = \|\cdot\|_{H^1(\Omega)} \coloneqq \|\cdot\|_\Omega + \|\nabla\cdot\|_\Omega$, where the symbol $\|\cdot\|_A$ represents the $L^2(A)$ norm over the measurable set $A \subset \Omega$, and $(\cdot,\cdot)_A$ represents the L^2 scalar product on the given domain A. We define the following space:

$$V \coloneqq \left(H^1_{0,\partial\Omega}(\Omega)\right)^d = \{v \in H^1(\Omega)^d \colon v\big|_{\partial\Omega} = 0\},$$

The standard weak formulation reads:

Problem 13 (Classic Weak Formulation)**.** *Given $b \in L^2(\Omega)^d$, find $u \in V$ such that:*

$$(\mathbb{C}\nabla u, \nabla v)_\Omega = (b, v)_\Omega \qquad \forall v \ \in V. \tag{4.9}$$

The main idea behind our reformulation is to rewrite Problem 13 into an equivalent form, where we define two independent functional spaces. The novelty we introduce is to define the functional spaces on Ω and Ω_f, and not on $\hat{\Omega}$ and Ω_f. To achieve this, we define two fictitious materials: one with the same properties of the elastic matrix, occupying the full space Ω, and one describing the "excess elasticity" of the fibers separately, defined on Ω_f only. The first step in this direction is to split the left-hand side of Equation 4.9 on the two domains:

$$\begin{aligned} (\mathbb{C}\nabla u, \nabla v)_\Omega &= (\mathbb{C}_f\nabla u, \nabla v)_{\Omega_f} + (\mathbb{C}_\Omega\nabla u, \nabla v)_{\hat{\Omega}} \\ &= (\mathbb{C}_f\nabla u, \nabla v)_{\Omega_f} + \underbrace{(\mathbb{C}_\Omega\nabla u, \nabla v)_{\hat{\Omega}} + (\mathbb{C}_\Omega\nabla u, \nabla v)_{\Omega_f}}_{\Omega} - (\mathbb{C}_\Omega\nabla u, \nabla v)_{\Omega_f} \\ &= (\mathbb{C}_\Omega\nabla u, \nabla v)_\Omega + (\delta\mathbb{C}_f\nabla u, \nabla v)_{\Omega_f}, \end{aligned}$$

where $\delta\mathbb{C}_f \coloneqq \mathbb{C}_f - (\mathbb{C}_\Omega)\big|_{\Omega_f}$.

For simplicity, we improperly use the expression "elastic matrix equation" and "fiber equation", even though they should be really considered as the "whole domain equation", and a "delta fiber equation".

This formal separation does not change the original variational problem, which can still be stated explicitly: given $b \in L^2(\Omega)^d$, find $u \in V$ such that:

$$(\mathbb{C}_\Omega \nabla u, \nabla v)_\Omega + (\delta \mathbb{C}_f \nabla u, \nabla v)_{\Omega_f} = (b, v)_\Omega \qquad \forall\ v\ \in V. \tag{4.10}$$

To simplify the coupling between the fibers and the elastic matrix, we need to split Equation 4.10 on two functional spaces, describing their boundary conditions.

The boundary of the domain Ω induces a natural splitting on the boundary of the fibers: we define the following partition of $\partial\Omega_f$:

$$B_i \coloneqq \partial\Omega_f \setminus \partial\Omega \tag{4.11}$$

$$B_e \coloneqq \partial\Omega \cap \partial\Omega_f, \tag{4.12}$$

where B_i is the **i**nterface between the fibers and the elastic matrix, and B_e is the interface between the fibers the **e**xterior part of Ω, that lies on the boundary $\partial\Omega$.

Next we define the restriction of $H_0^1(\Omega)$ on the fibers:

$$W \coloneqq \left(H^1_{0,B_e}(\Omega_f)\right)^d. \tag{4.13}$$

With the explicit introduction of the space W we can modify Problem 13 by separating the solution into two components, one describing the whole matrix, the other describing the fibers. To enforce the continuity at the boundary of the two components we impose an interface condition: the following **non-slip** constraint for the solution $(u, w) \in V \times W$:

$$u\big|_{\Omega_f} = w. \tag{4.14}$$

To easily the continuity at the boundary of the two components and Equation 4.14, one should choose matching grids for Ω and Γ. To avoid this inconvenience we follow the distributed Lagrange multiplier approach described in [20, 6, 23].

The modified problem can be described as a constrained minimization problem:

$$u, w = \arg \inf_{\substack{u \in V \\ w \in W \\ \text{subject to} \\ u\big|_{\Omega_f} = w}} \psi(u, w), \tag{4.15}$$

where we defined the total elastic energy of the system as

$$\psi(u, w) = \frac{1}{2}(\mathbb{C}_\Omega \nabla u, \nabla u)_\Omega + \frac{1}{2}(\delta \mathbb{C}_f \nabla w, \nabla w)_{\Omega_f} - (b, u)_\Omega. \tag{4.16}$$

To impose the non-slip constraint of Equation 4.14 in weak form, several choices are possible. One may use, for example, the scalar product of W, or the duality product $W' \times W$ as in [21]. In this work, we use the latter, let $Q \coloneqq H^{-1}_{0,B_e}(\Omega_f)^d = W'$, which enforces the transmission condition on the interface (see [6, 23]):

$$\langle q, u\big|_{\Omega_f} - w \rangle_{Q \times W} = 0 \qquad \forall q \in Q. \tag{4.17}$$

For simplicity we shall avoid the subscript $Q \times W$ from now on.

The constrained minimization expressed in Equation 4.15 is equivalent to the saddle point problem:

$$u, w, \lambda = \arg \inf_{\substack{u \in V \\ w \in W}} \left(\arg \sup_{\lambda \in Q} \psi(u, w, \lambda) \right), \tag{4.18}$$

where the constraint is imposed weakly as in 4.17 with a Lagrange multiplier:

$$\psi(u, w, \lambda) := \frac{1}{2}(\mathbb{C}_\Omega \nabla u, \nabla u)_\Omega + \frac{1}{2}(\delta \mathbb{C}_f \nabla w, \nabla w)_{\Omega_f} + \langle \lambda, u\big|_{\Omega_f} - w \rangle - (b, u)_\Omega.$$

A solution to Equation 4.18 is obtained by solving the Euler-Lagrange equation:

$$\langle D_u \psi, v \rangle + \langle D_w \psi, y \rangle + \langle D_\lambda \psi, q \rangle = 0 \qquad \forall\ v \in V,\ \forall\ y \in W, \forall\ q \in Q, \tag{4.19}$$

that is:

Problem 14 (Saddle Point Weak Formulation)**.** *Given $b \in L^2(\Omega)^d$, find $u \in V, w \in W, \lambda \in Q$ such that:*

$$(\mathbb{C}_\Omega \nabla u, \nabla v)_\Omega + \langle \lambda, v\big|_{\Omega_f} \rangle = (b, v)_\Omega \qquad \forall v \in V \tag{4.20}$$

$$(\delta \mathbb{C}_f \nabla w, \nabla y)_{\Omega_f} - \langle \lambda, y \rangle = 0 \qquad \forall y \in W \tag{4.21}$$

$$\langle q, u\big|_{\Omega_f} - w \rangle = 0 \qquad \forall q \in Q, \tag{4.22}$$

or, equivalently,

$$\begin{array}{rrrll} \mathcal{K}_\Omega u & & +\mathcal{B}^T \lambda & = (b, \cdot)_\Omega & \text{in } V' \\ & \mathcal{K}_f w & -\mathcal{M}^T \lambda & = 0 & \text{in } W' \\ \mathcal{B} u & -\mathcal{M} w & & = 0 & \text{in } Q', \end{array} \tag{4.23}$$

where

$$\begin{array}{ll} \mathcal{K}_\Omega \colon V \to V' & \langle \mathcal{K}_\Omega u, \cdot \rangle \coloneqq (\mathbb{C}_\Omega \nabla u, \nabla \cdot)_\Omega \\ \mathcal{K}_f \colon W \to W' & \langle \mathcal{K}_f w, \cdot \rangle \coloneqq (\delta \mathbb{C}_f \nabla w, \nabla \cdot)_{\Omega_f} \\ \mathcal{B} \colon V \to Q' & \langle \mathcal{B} u, \cdot \rangle \coloneqq \langle \cdot, u\big|_{\Omega_f} \rangle \\ \mathcal{M} \colon W \to Q' & \langle \mathcal{M} w \cdot \rangle \coloneqq \langle \cdot, w \rangle. \end{array} \tag{4.24}$$

4.3.3 Well-posedness, existence and uniqueness

The theory for saddle point structure problems is well known and can be found, for example, in [19]. To verify well-posedness, existence and uniqueness of the solution to such a problem, it is sufficient for certain $inf - sup$ and $ell - ker$ conditions to be satisfied.

To make our notation closer to the one used in [19], we introduce the following Hilbert space, with its norm:

$$\begin{aligned} \mathbb{V} &\coloneqq V \times W, \\ \|(u, w)\|_{\mathbb{V}}^2 &\coloneqq \|u\|_V^2 + \|w\|_W^2. \end{aligned}$$

We indicate with $\boldsymbol{u} \coloneqq (u, w), \boldsymbol{v} \coloneqq (v, y)$ the elements of $\mathbb{V}$, and define the following bilinear forms:

$$\begin{aligned}
&\mathbb{F}\colon \mathbb{V} \times \mathbb{V} \longrightarrow \mathbb{R} \\
&\quad (\boldsymbol{u}, \boldsymbol{v}) \longmapsto \langle \mathcal{K}_\Omega u, v\rangle + \langle \mathcal{K}_f w, y\rangle = (\mathbb{C}_\Omega \nabla u, \nabla v)_\Omega + (\delta \mathbb{C}_f \nabla w, \nabla y)_{\Omega_f}, \\
&\mathbb{E}\colon \mathbb{V} \times Q \longrightarrow \mathbb{R} \\
&\quad (\boldsymbol{u}, q) \longmapsto \langle \mathcal{B} u, q\rangle - \langle \mathcal{M} w, q\rangle = \langle q, v\big|_{\Omega_f} - w\rangle.
\end{aligned}$$

Summing the first two equations of Problem 14 and using the newly defined space and bilinear forms we can restate the problem as: find $\boldsymbol{u} \in \mathbb{V}, \lambda \in Q$ such that

$$\begin{aligned}
\mathbb{F}(\boldsymbol{u}, \boldsymbol{v}) + \mathbb{E}(\lambda, \boldsymbol{v}) &= (b, v)_\Omega \qquad && \forall\, \boldsymbol{v} \coloneqq (v, y) \in \mathbb{V}, \\
\mathbb{E}(\boldsymbol{u}, q) &= 0 && \forall\, q \in Q.
\end{aligned} \tag{4.25}$$

Following [19], to state the $inf - sup$ conditions we introduce the kernel

$$\ker \mathbb{E} \coloneqq \left\{ \boldsymbol{v} \coloneqq (v, w) \in \mathbb{V}\colon \langle q, v\big|_{\Omega_f} - w\rangle = 0 \ \forall q \in Q \right\}.$$

Since we can identify $L^2(\Omega_f)$ with its dual, and $L^2(\Omega_f) \subset W'$, we have that

$$\langle q, v\big|_{\Omega_f} - w\rangle = 0 \Rightarrow (q, v\big|_{\Omega_f} - w)_{\Omega_f} = 0 \Rightarrow v\big|_{\Omega_f} = w \ a.e. \tag{4.26}$$

A sufficient condition for the problem to be well-posed, and prove existence and uniqueness of the solution is the following: if there exist two positive constants $\alpha_1 > 0, \alpha_2 > 0$ such that

$$\inf_{q \in Q} \sup_{\boldsymbol{v} \in \mathbb{V}} \frac{\mathbb{E}(\boldsymbol{u}, q)}{\|\boldsymbol{u}\|_{\mathbb{V}}\ \|q\|_Q} \geq \alpha_1 \tag{4.27}$$

$$\inf_{\boldsymbol{u} \in \ker \mathbb{E}} \sup_{\boldsymbol{v} \in \ker \mathbb{E}} \frac{\mathbb{F}(\boldsymbol{u}, \boldsymbol{v})}{\|\boldsymbol{u}\|_{\mathbb{V}}\ \|\boldsymbol{v}\|_{\mathbb{V}}} \geq \alpha_2. \tag{4.28}$$

These two conditions can be proven with the following propositions:

Proposition 4.3.1. *There exists a constant $\alpha_1 > 0$ such that:*

$$\inf_{q \in Q} \sup_{(v,w) \in \mathbb{V}} \frac{\langle q, v\big|_{\Omega_f} - w\rangle}{\|(v, w)\|_{\mathbb{V}}\ \|q\|_Q} \geq \alpha_1.$$

Moreover $\alpha_1 = 1$.

The proof for this proposition, and its discrete version, are variations on the one found in [21].

Proof. The non slip condition is given by the duality pairing between $Q = W'$ and W; by definition of the norm in the dual space Q:

$$\begin{aligned}
\|q\|_Q &= \sup_{w \in W} \frac{\langle q, w\rangle}{\|w\|_Q} \\
&\leq \sup_{v \in V, w \in W} \frac{\langle q, v\big|_{\Omega_f} - w\rangle}{(\|w\|_Q^2 + \|v\|_V^2)^{\frac{1}{2}}},
\end{aligned}$$

where the last inequality can be proven fixing $v = 0$. The final statement is found dividing by $\|q\|_Q$ and taking the $\inf_{q \in Q}$. □

Proposition 4.3.1 proves Inequality 4.27. To prove Inequality 4.28 additional hypotheses are needed:

Proposition 4.3.2. *Assume $\mathbb{C}_\Omega$ and $\mathbb{C}_f$ to be strongly elliptical, with constants c_Ω and c_f respectively such that $c_f > c_\Omega > 0$; there exists a constant $\alpha_2 > 0$ such that:*

$$\inf_{(u,w)\in\ker\mathbb{E}} \sup_{(v,y)\in\ker\mathbb{E}} \frac{(\mathbb{C}_\Omega\nabla u,\nabla v)_\Omega + (\delta\mathbb{C}_f\nabla w,\nabla y)_{\Omega_f}}{\|(v,y)\|_{\mathbb{V}}\|(u,w)\|_{\mathbb{V}}} \geq \alpha_2.$$

Proof. An immediate consequence of the hypotheses is that $\delta\mathbb{C}_f$ is elliptic of constant $c_f - c_\Omega$. Following the proof for a similar statement found in [6, 23], given an element $(v,y) \in \ker(\mathbb{E})$, the fact that $v\big|_{\Omega_a} = y$ allows to use the Poincaré inequality on v to control the norm of y; for every $(u,w) \in \ker(\mathbb{E})$:

$$\begin{aligned}
\sup_{(v,y)\in\ker(\mathbb{E})} & \frac{(\mathbb{C}_\Omega\nabla u,\nabla v)_\Omega + (\delta\mathbb{C}_f\nabla w,\nabla y)_{\Omega_f}}{\|(v,y)\|_{\mathbb{V}}} \\
&\geq \frac{c_\Omega(\nabla u,\nabla u)_\Omega + (c_f - c_\Omega)(\nabla w,\nabla w)_{\Omega_f}}{\|(u,w)\|_{\mathbb{V}}} \\
&\geq \frac{c_p\min(c_\Omega, c_f - c_\Omega)}{2}\frac{(u,u)_{H^1(\Omega)} + (w,w)_{H^1(\Omega_f)}}{\|(u,w)\|_{\mathbb{V}}} \\
&\geq \alpha_2\|(u,w)\|_{\mathbb{V}},
\end{aligned}$$

where we used the Poincaré inequality with its positive constant c_p on $u \in V$, with $\alpha_2 \coloneqq \frac{c_p\min(c_\Omega, c_f - c_\Omega)}{2}$, and we used the scalar product of $H^1(\Omega)$:

$$(u,u)_{H^1(\Omega)} \coloneqq (u,v)_\Omega + (\nabla u,\nabla v)_\Omega,$$

and the analogous one for $H^1(\Omega_f)$. The final statement is obtained dividing by $\|(u,w)\|_{\mathbb{V}}$ and considering the $\inf_{(u,w)\in\ker\mathbb{E}}$. □

Remark. *This manuscript does not intend to focus on the* choice *of elastic tensors. As stated in Subsection 4.1.3, strong ellipticity is a common property among them, and holds in the case of linearly elastic materials with some hypotheses on μ and λ: let $u \in V$, $w \in W$*

$$\mathbb{C}_\Omega\nabla u \coloneqq 2\mu_\Omega Eu + \lambda_\Omega(tr\nabla u)I = 2\mu_\Omega Eu + \lambda_\Omega(\operatorname{div} u)I \tag{4.29}$$

$$\mathbb{C}_f\nabla w \coloneqq 2\mu_f Ew + \lambda_f(tr\nabla w)I = 2\mu_f Ew + \lambda_f(\operatorname{div} w)I \tag{4.30}$$

$$\begin{aligned}
\delta\mathbb{C}_f\nabla w &\coloneqq 2(\mu_f - \mu_\Omega)Ew + (\lambda_f - \lambda_\Omega)(tr\nabla w)I \\
&= 2\mu_\delta Ew + \lambda_\delta(\operatorname{div} w)I,
\end{aligned} \tag{4.31}$$

where $\mu_\delta \coloneqq \mu_f - \mu_\Omega$ and $\lambda_\delta \coloneqq \lambda_f - \lambda_\Omega$. For our numerical tests we shall use these equations, with coefficients satisfying the strongly ellipticity hypotheses of Theorem 4.1.3.

Proposition 4.3.2 implies Inequality 4.28, and we conclude that Problem 14 is well-posed, and has a unique solution.

In the next subsection 4.3.3, we prove Proposition 4.3.2 directly, in the case of a pointwise stable linear elastic operators.

Direct proof for the inf-sup

Assuming pointwise stability (see Subsection 4.1.3) for the constitutive laws, described in Equations 4.29 and 4.31, we can prove directly a modified version of Proposition 4.3.2:

Proposition 4.3.3. *Assume $\mathbb{C}_\Omega$ and $\mathbb{C}_f$ to be pointwise stable linear elastic equations, assume $\Omega_f \Subset \Omega$, there exists a constant $\alpha_2 > 0$ such that:*

$$\inf_{(u,w)\in\ker\mathbb{E}} \sup_{(v,y)\in\ker\mathbb{E}} \frac{(\mathbb{C}_\Omega\nabla u,\nabla v)_\Omega + (\delta\mathbb{C}_f\nabla w,\nabla y)_{\Omega_f}}{\|(v,y)\|_{\mathbb{V}}\|(u,w)\|_{\mathbb{V}}} \geq \alpha_2.$$

Idea of the proof: for elements of V we can use Korn and Poincare' inequalities; using the fact that $(u,w)\in\ker(\mathbb{E})$ this allows to control also w.

Proof. Define $\hat{\mu} \coloneqq \frac{1}{2}\min(\mu_\Omega,\mu_f) > 0$; if $\lambda_\Omega \geq 0$, using the definition of $\mathbb{C}_\Omega$:

$$(\mathbb{C}_{\hat{\Omega}}\nabla u,\nabla u)_{\hat{\Omega}} \geq 2\mu_\Omega\|Eu\|_{\hat{\Omega}} \geq 2\hat{\mu}\|Eu\|_{\hat{\Omega}}.$$

If $\lambda_\Omega < 0$, then the stability condition $2\mu_\Omega + 3\lambda_\Omega > 0 \Rightarrow \mu_\Omega + \lambda_\Omega > 0$. Now using the fact that $\|Eu\|_{\hat{\Omega}} \geq \|tr(Eu)\|_{\hat{\Omega}} = \|\operatorname{div}(u)\|_{\hat{\Omega}}$:

$$\begin{aligned}(\mathbb{C}_\Omega\nabla u,\nabla u)_{\hat{\Omega}} &= 2\mu_\Omega\|Eu\|_{\hat{\Omega}} + \lambda_\Omega\|\operatorname{div}(u)\|_{\hat{\Omega}} \\ &\geq \mu_\Omega\|Eu\|_{\hat{\Omega}} + \underbrace{(\mu_\Omega+\lambda_\Omega)}_{>0}\|\operatorname{div}(u)\|_{\hat{\Omega}} > \mu_\Omega\|Eu\|_{\hat{\Omega}} \geq 2\hat{\mu}\|Eu\|_{\hat{\Omega}}.\end{aligned}$$

Similarly:

$$(\mathbb{C}_f\nabla w,\nabla y)_{\Omega_f} \geq 2\hat{\mu}\|Eu\|_{\Omega_f}.$$

Splitting $\mathbb{F}$ over $\hat{\Omega}$ and Ω_f, and applying last inequality:

$$\begin{aligned}\sup_{(v,y)\in\ker(\mathbb{E})} (\mathbb{C}_\Omega\nabla u,\nabla v)_\Omega + (\delta\mathbb{C}_f\nabla w,\nabla y)_{\Omega_f} &= \sup_{(v,y)\in\ker(\mathbb{E})} (\mathbb{C}_\Omega\nabla u,\nabla v)_{\hat{\Omega}} + (\mathbb{C}_f\nabla w,\nabla y)_{\Omega_f} \\ &\geq \hat{\mu}(Eu,Eu)_{\hat{\Omega}} + 2\hat{\mu}(Ew,Ew)_{\Omega_f} \\ &= \hat{\mu}(Eu,Eu)_\Omega + \hat{\mu}(Ew,Ew)_{\Omega_f}.\end{aligned}$$

The last equality holds as $(u,w)\in\ker\mathbb{E}$. Then:

$$\sup_{(v,y)\in\ker(\mathbb{E})} \frac{(\mathbb{C}\nabla u,\nabla v)_\Omega + (\delta\mathbb{C}_f\nabla w,\nabla y)_{\Omega_f}}{\|(v,y)\|_{\mathbb{V}}} \geq \hat{\mu}\frac{\|Eu\|_\Omega^2 + \|Ew\|_{\Omega_f}^2}{\|(u,w)\|_{\mathbb{V}}} \tag{4.32}$$

As $u\in V = H_0^1(\Omega)$ we can use Korn's first inequality (see e.g. [77] or [78]): there is a constant C_K such that $C_K\|\nabla u\|_\Omega^2 \leq \|Eu\|_\Omega^2$. Using then Poincaré inequality with its constant C_Ω on $H_0^1(\Omega)$ we obtain:

$$\begin{aligned}(Eu,Eu)_\Omega + (Ew,Ew)_{\Omega_f} &\geq C_K\|\nabla u\|_\Omega^2 + \|Ew\|_{\Omega_f}^2 \\ &\geq C_K C_\Omega\|u\|_{H^1(\Omega)}^2 + \|Ew\|_{\Omega_f}^2 \\ &\geq C_1\left(\underbrace{\|\nabla u\|_\Omega^2}_{(a)} + \underbrace{\|u\|_\Omega^2 + \|Ew\|_{\Omega_f}^2}_{(b)}\right),\end{aligned}$$

where $C_1 = \min(1, C_\Omega C_K)$. Because $(u, w) \in \ker(\mathbb{E}) \Rightarrow u\big|_{\Omega_f} = w$ and $\|u\|^2_{\Omega_f} = \|w\|^2_{\Omega_f}$. The term (b) can thus be estimated using the second Korn inequality:

$$\|u\|^2_\Omega + \|Ew\|^2_{\Omega_f} \geq \|w\|^2_{\Omega_f} + \|Ew\|^2_{\Omega_f} \geq C_K \|\nabla w\|^2_{\Omega_f},$$

where we assume C_K is a constant satisfying both Korn inequalities. Applying Poincaré inequality on (a):

$$\begin{aligned}\|\nabla u\|^2_\Omega + \|u\|^2_\Omega + \|Ew\|^2_{\Omega_f} &\geq C_\Omega(\|u\|^2_\Omega + \|\nabla u\|^2_\Omega) + C_K\|\nabla w\|^2_{\Omega_f} \\ &\geq C_2 \left(\underbrace{\|\nabla u\|^2_\Omega}_{(a)} + \underbrace{\|u\|^2_{\Omega_f} + \|\nabla w\|^2_{\Omega_f}}_{(b)} \right),\end{aligned}$$

where $C_2 = \min(C_\Omega, C_K)$ and $(b) = \|w\|^2_{H^1(\Omega_f)}$. Apply again Poincaré inequality on (a):

$$(Eu, Eu)_\Omega + (Ew, Ew)_{\Omega_f} \geq \alpha_2 \left(\|u\|^2_{H^1(\Omega)} + \|w\|^2_{H^1(\Omega_f)} \right) = \|(u, w)\|^2_{\mathbb{V}}$$

with $\alpha_2 = \hat{\mu} C_1 C_2 C_\Omega$. Substituting in Equation 4.32:

$$\sup_{(v,y)\in\ker(\mathbb{E})} \frac{(\mathbb{C}\nabla v, \nabla u)_\Omega + (\mathbb{C}_f \nabla y, \nabla w)_{\Omega_f}}{\|(v, y)\|_{\mathbb{V}}} \geq \alpha_2 \frac{\|(u, w)\|^2_{\mathbb{V}}}{\|(u, w)\|_{\mathbb{V}}} = \alpha_2 \|(u, w)\|_{\mathbb{V}}$$

We conclude dividing by $\|(u, w)\|_{\mathbb{V}}$ and evaluating the inf over $(u, w) \in \ker(\mathbb{E})$. □

4.3.4 Finite element discretization

The formulation of Problem 14 makes it possible to consider independent, separate triangulations for its numerical solution. Consider the family $\mathcal{T}_h(\Omega)$ of regular meshes in Ω, and a family $\mathcal{T}_h(\Omega_f)$ of regular meshes in Ω_f, where we denote by h the maximum diameter of the elements of the two triangulations. We assume that no geometrical error is committed when meshing, i.e., $\Omega = \bigcup_{T_h \in \mathcal{T}_h(\Omega)} T_h$, and $\Omega_f = \bigcup_{S_h \in \mathcal{T}_h(\Omega_f)} S_h$. We consider two independent finite element spaces $V_h \subset V$, and $W_h \subset W$.

The duality paring between Q and W can be represented using the L^2 scalar product in Ω_a:

$$\langle q, w \rangle = (q, w)_{\Omega_f}.$$

Thus we discretize the duality pairing as the L^2 product, and consider $Q_h = W_h$. To handle the restriction of functions from V_h to W_h, we first introduce the L^2 projection:

$$P_W \colon W \to W_h \subset L^2(\Omega_f),$$

which is continuous: for every $u \in W$

$$\|P_W u\|_{L^2(\Omega_f)} \leq \|u\|_{L^2(\Omega_f)}.$$

This condition is too weak for our problem, which involves ∇w. We assume the projection P_W is H^1-stable, i.e., there exists a positive constant c, such that for all $w \in W$:

$$\|\nabla P_W w\|_{\Omega_f} \leq c \|\nabla w\|_{\Omega_f}. \tag{4.33}$$

Given a function $v \in V$, with a slight abuse of notation, we shall write $P_W v$ instead of $P_W(v|_{\Omega_a})$. We assume that the H^1-stability property holds uniformly on $\mathcal{T}_h(\Omega)$ and $\mathcal{T}_h(\Omega_f)$, provided that the mesh size h is comparable on the two domains.

To discretize Problem 14, we reformulate the weak non-slip condition of Equation 4.17: for every $v_h \in V_h$,$w_h \in W_h$,$q_h \in Q_h$:

$$\begin{aligned}(q_h, v_h|_{\Omega_f} - w_h)_{\Omega_f} &= (q_h, v_h|_{\Omega_f})_{\Omega_f} - (q_h, w_h)_{\Omega_f} \\ &= (q_h, P_W v_h)_{\Omega_f} - (q_h, w_h)_{\Omega_f} = (q_h, P_W v_h - w_h)_{\Omega_f}.\end{aligned}$$

Problem 15 (Discrete Weak Formulation)**.** *Given $b \in L^2(\Omega)^d$, find $u_h \in V_h, w_h \in W_h, \lambda_h \in Q_h$ such that:*

$$(\mathbb{C}_\Omega \nabla u_h, \nabla v_h)_\Omega + (\lambda_h, P_W v_h)_{\Omega_f} = (b, v_h)_\Omega \qquad \forall v_h \in V_h, \tag{4.34}$$

$$(\delta \mathbb{C}_f \nabla w_h, \nabla y_h)_{\Omega_f} - (\lambda_h, y_h)_{\Omega_f} = 0 \qquad \forall y_h \in W_h, \tag{4.35}$$

$$(q_h, P_W u_h - w_h)_{\Omega_f} = 0 \qquad \forall q_h \in Q_h. \tag{4.36}$$

As in the continuous case, we study the $inf - sup$ conditions after defining the following space:

$$\mathbb{V}_h \coloneqq V_h \times W_h,$$

equipped with the norm $\|\cdot\|_{\mathbb{V}}$, and the operators:

$$\begin{aligned}&\mathbb{F}_h \colon \mathbb{V}_h \times \mathbb{V}_h \longrightarrow \mathbb{R} \\ &\qquad (\boldsymbol{u}_h, \boldsymbol{v}_h) \longmapsto (\mathbb{C}_\Omega \nabla u_h, \nabla v_h)_\Omega + (\delta \mathbb{C}_f \nabla w_h, \nabla y_h)_{\Omega_f}, \\ &\mathbb{E}_h \colon \mathbb{V}_h \times Q_h \longrightarrow \mathbb{R} \\ &\qquad (\boldsymbol{u}_h, q_h) \longmapsto (q_h, P_W v_h - w_h)_{\Omega_f}.\end{aligned}$$

Equation 4.25 can now be discretized: find $\boldsymbol{u}_h \in \mathbb{V}_h, \lambda_h \in Q_h$ such that

$$\begin{aligned}\mathbb{F}_h(\boldsymbol{u}_h, \boldsymbol{v}_h) + \mathbb{E}_h(\boldsymbol{v}_h, \lambda_h) &= (b, v_h)_\Omega, \qquad &\forall\, \boldsymbol{v}_h \coloneqq (v_h, y_h) \in \mathbb{V}_h \\ \mathbb{E}_h(\boldsymbol{u}_h, q_h) &= 0 \qquad &\forall\, q_h \in Q_h.\end{aligned}$$

Following Subsection 4.3.3, Problem 15 is well-posed, and there exists a unique solution if there exist two positive constants $\alpha_3 > 0, \alpha_4 > 0$ such that

$$\inf_{q_h \in Q_h} \sup_{\boldsymbol{v}_h \in \mathbb{V}_h} \frac{\mathbb{E}(\boldsymbol{u}_h, q_h)}{\|\boldsymbol{u}_h\|_{\mathbb{V}}\ \|q_h\|_{\Omega_f}} \geq \alpha_3$$

$$\inf_{\boldsymbol{u}_h \in \ker \mathbb{E}_h} \sup_{\boldsymbol{v}_h \in \ker \mathbb{E}_h} \frac{\mathbb{F}(\boldsymbol{u}_h, \boldsymbol{v}_h)}{\|\boldsymbol{u}\|_{\mathbb{V}_h}\ \|\boldsymbol{v}_h\|_{\mathbb{V}_h}} \geq \alpha_4.$$

These conditions can be proved modifying Propositions 4.3.1 and 4.3.2:

Proposition 4.3.4. *There exists a constant $\alpha_3 > 0$, independent of h, such that:*

$$\inf_{q_h \in Q_h} \sup_{(v_h, w_h) \in \mathbb{V}_h} \frac{(q_h, P_W v_h - w_h)_{\Omega_f}}{\|(v_h, w_h)\|_{\mathbb{V}} \|q_h\|_{\Omega_f}} \geq \alpha_3.$$

Proof. The proof is similar to the one of Proposition 4.3.1. For every $q_h \in Q_h$, by definition of the Q norm, and using the representation of the duality paring as a scalar product in $L^2(\Omega_f)$, there exists $\hat{w} \in W$ such that:

$$\|q_h\|_Q = \sup_{w \in W} \frac{(q_h, w)_{\Omega_f}}{\|w\|_Q} = \frac{(q_h, \hat{w})_{\Omega_f}}{\|\hat{w}\|_Q} = \frac{(q_h, P_W \hat{w})_{\Omega_f}}{\|\hat{w}\|_Q}, \tag{4.37}$$

where the last equality holds for the choice of spaces. Using well-known properties of P_W, and the H^1 stability, we can prove there exists a c such that:

$$\|P_W \hat{w}\|_Q \leq C \|\hat{w}\|_Q.$$

Inserting this in Equation 4.37:

$$\begin{aligned}
\|q_h\|_Q &\leq \frac{(q_h, P_W \hat{w})_{\Omega_f}}{\|\hat{w}\|_Q} \leq C \frac{(q_h, P_W \hat{w})_{\Omega_f}}{\|P_W \hat{w}\|_Q} \\
&\leq C \sup_{w_h \in W_h} \frac{(q_h, w_h)_{\Omega_f}}{\|w_h\|_Q} \leq C \sup_{v_h \in V_h, w_h \in W_h} \frac{(q_h, v_h\big|_{\Omega_a} - w_h)_{\Omega_f}}{(\|w_h\|_Q^2 + \|v_h\|_V^2)^{\frac{1}{2}}},
\end{aligned}$$

and we conclude as in Proposition 4.3.1. □

To study the $ell - ker$ condition on $\mathbb{F}_h$ we define:

$$\begin{aligned}
\ker(\mathbb{E}_h) &:= \left\{ \boldsymbol{v}_h := (v_h, w_h) \in \mathbb{V}_h \colon (q_h, P_W v_h - w_h)_{\Omega_f} = 0 \qquad \forall q_h \in Q_h \right\} \\
&= \left\{ \boldsymbol{v}_h := (v_h, w_h) \in \mathbb{V}_h \colon (q_h, P_W v_h)_{\Omega_f} = (q_h, w_h)_{\Omega_f} \qquad \forall q_h \in Q_h \right\}.
\end{aligned}$$

Proposition 4.3.5. *Assume $\mathbb{C}_\Omega$ and $\mathbb{C}_f$ to be strongly elliptical with constants c_Ω and c_f respectively such that $c_f > c_\Omega > 0$; there exists a constant $\alpha_4 > 0$, independent of h, such that:*

$$\inf_{(u_h, w_h) \in \ker \mathbb{E}_h} \sup_{(v_h, y_h) \in \ker \mathbb{E}_h} \frac{(\mathbb{C}_\Omega \nabla v_h, \nabla u_h)_\Omega + (\delta \mathbb{C}_f \nabla y_h, \nabla w_h)_{\Omega_f}}{\|(v_h, y_h)\|_{\mathbb{V}} \|(u_h, w_h)\|_{\mathbb{V}}} \geq \alpha_4.$$

Proof. Mutatis mutandis, the proof follows the one of Proposition 4.3.2. □

Error estimate Proposition 4.3.1 and 4.3.2 allow us to apply the theory from Chapter 5 of [19], obtaining the following error estimate:

Theorem 4.3.1. *Consider $\mathbb{C}_\Omega$ and $\mathbb{C}_f$, elastic stress tensors satisfying the hypothesis of Proposition 4.3.2, the domains Ω and Ω_f with the regularity required in Section 4.3, and $b \in L^2(\Omega)^d$. Then the following error estimate holds for (u, w, λ), solution to Problem 14, and (u_h, w_h, λ_h), solution of Problem 15:*

$$\begin{aligned}
&\|u - u_h\|_V + \|w - w_h\|_W + \|\lambda - \lambda_h\|_Q \leq \\
&\qquad C_e \left(\inf_{v_h \in V_h} \|u - v_h\|_V + \inf_{y_h \in W_h} \|w - y_h\|_W + \inf_{q_h \in Q_h} \|\lambda - q_h\|_Q \right),
\end{aligned} \tag{4.38}$$

where $C_e > 0$, and depends on $\alpha_3, \alpha_4, c_\Omega, c_f$ and the norm of the operators $\mathcal{K}_\Omega$ and $\mathcal{K}_f$.

We remark how this constant C_e depends on α_3, which is affected by the coupling between the two meshes. As intuition suggests, the quality of the solution does not depend only the on the ability of V and W to individually describe it, but also on the coupling between them.

Non-matching meshes One of the basic assumptions made in the continuous case is that, for every element $v \in V$, we have that $v\big|_{\Omega_a} \in W$; similarly every element $w \in W$ can be extended to an element of V.

With an independent discretization of the two meshes the inclusion $W_h \subset V_h$ can not be guaranteed, leading to the use of the projection $P_W \colon V_h \to W_h$, which we require to be H^1-stable, i.e., that Equation 4.33 holds.

Since for every $\boldsymbol{v}_h = (v_h, w_h) \in \ker(\mathbb{E}_h) \Rightarrow P_W(v_h) = w_h \Rightarrow w_h \in P_W(V_h)$; if the set $P_W(V_h)$ is small, the $inf - sup$ constant can be negatively affected; the extreme case being $P_W = 0$, which results in $\ker(\mathbb{E}_h) = \{(0,0)\}$, and the $inf - sup$ condition for $\mathbb{F}_h$ not satisfied.

Under some simple construction hypotheses it is possible to guarantee that globally constant and linear functions are included in the kernel, ensuring that $\ker(\mathbb{E}_h) \neq \{(0,0)\}$.

4.4 Thin fibers

The computational cost of discretizing explicitly numerous three-dimensional fibers might render Problem 15 too computationally demanding: a possible simplification is suggested by the fiber shape, which can be approximated with a one-dimensional structure. Constructing this simplified model is a non-trivial task because the restriction (or trace) of a three-dimensional function to a one-dimensional domain is not well defined in Sobolev spaces.

Instead of resorting to weighted Sobolev spaces and graded meshes, as done in [30, 29], the solution we propose is to introduce additional modellistic hypotheses, that allow one to use averaging techniques instead of traces to render the problem well posed.

To simplify the exposition, we shall consider a single fiber; the same results hold with a finite collection of fibers.

We use the definitions of Section 4.2: let Γ be a one-dimensional connected domain, embedded in Ω, let $\omega \colon [0, L] \to \Gamma$ be its arclength parametrization of Γ. We assume the Frenet trihedron $(t(s), n(s), b(s))$ to be well defined for every $s \in [0, L]$, at every point $\omega(s) \in \Gamma$, and the function $s \mapsto (t(s), n(s), b(s))$ to be continuous.

We fix a constant radius $a \in \mathbb{R},\ a > 0$ for the physical fiber Ω_a, which is described by the tubular neighbourhood of Γ or radius a; we assume $\Omega_a \subset \Omega$.

We now modify the definitions of Section 4.3, for the boundaries of $\partial\Omega_a$:

$$
\begin{aligned}
B_i &\coloneqq \partial\Omega_a \setminus \partial\Omega, \\
B_e &\coloneqq \partial\Omega \cap \partial\Omega_a, \\
V &\coloneqq \left(H^1_{0,\partial\Omega}(\Omega)\right)^d, \\
W &\coloneqq \left(H^1_{0,\partial\Omega}(\Omega)\right)^d \Big|_{\Omega_a},
\end{aligned}
$$

with $d = 3$.

As a reference model, we consider small deformations of an FRM in which fibers are stiff compared to the underlying matrix. For this model, the fiber radius can be considered to be approximately constant.

Curved fiber In this section we call Γ be a one-dimensional straight line, immersed in $\mathbb{R}^3$ of finite length, and call Ω_a its tubular neighbourhood of radius a, which is a straight

cylinder. To represent a generic a one-dimensional connected domain we use the symbol $\tilde{\Gamma}$, and assume its tubular neighborhood can be obtained through the restriction of the diffeomorphism

$$\Phi\colon \Omega \to \widetilde{\Omega}, \tag{4.39}$$

satisfying the following hypotheses:

- It transforms the cylinder into the tubular neighbourhood of the curve $\widetilde{\Gamma}$ with constant radius a: $\widetilde{\Omega}_a = \Phi(\Omega_a)$
- It transforms the cylinder center line: $\widetilde{\Gamma} = \Phi(\Gamma)$
- It transforms orthogonal disks to Γ into orthogonal disks to $\widetilde{\Gamma}$: for every $x \in \Gamma$, $D_{a,\widetilde{\Gamma}}(\Phi(x)) = \Phi(D_{a,\Gamma}(x)) \Rightarrow \Phi^{-1}\left(D_{a,\widetilde{\Gamma}}(\Phi(x))\right) = D_{a,\Gamma}(x)$,

where we added the subscript to identify the disk as orthogonal to $\widetilde{\Gamma}$ and Γ respectively; $D_{a,\widetilde{\Gamma}}(\Phi(x)) \subset \widetilde{\Omega}_a$, and $D_{a,\Gamma}(\Phi(x)) \subset \Omega_a$.

Numerous transformations satisfy these properties, e.g., rotations and stretches, but also transformations modifying the curvature, see Section 4.2.

To simplify the notation we shall avoid making the explicit distinction between Γ, $\tilde{\Gamma}$ etc. in this manuscript, except for this paragraph and the next subsection, preferring then the symbol without tilde.

4.4.1 Regularity of the average

In this subsection we prove that, by considering a certain average for our functions, we obtain a sufficiently regular "restriction" operator which we can use for our fictitious domain approach.

To avoid confusion, in this subsection we shall avoid the symbols V, W, writing explicitly the spaces as $H^1(\Omega)$, $H^1(\widetilde{\Omega}_a)$, and so on.

In this Section, domains without tilde, such as Γ, refer to the case of cylinder aligned with the first axis; domains with tilde, such as $\widetilde{\Gamma}$, are used for other domains.

Let $w \in H^1(\widetilde{\Gamma})$ (or $w \in H^1(\Gamma)$), the surface gradient along the curve is defined as:

$$\nabla_{\widetilde{\Gamma}} w \coloneqq t \otimes t \nabla \bar{w}, \tag{4.40}$$

where $\bar{w}$ is a tubular extension of w on a neighbourhood of $\tilde{\Gamma}$ (or Γ), and we are using the unitary tangent vector to the curve $\widetilde{\Gamma}$ from the Frenet trihedron $(t(x), n(x), b(x))$.

We now want to prove that for every $u \in H^1(\widetilde{\Omega})$, its average on $\tilde{\Gamma}$ is also on $H^1(\tilde{\Gamma})$, i.e., $\bar{u} \in H^1(\widetilde{\Gamma})$, where $\bar{u}$ is defined for a.e. $x \in \Gamma$ as:

$$\bar{u}(x) \coloneqq \frac{1}{|D_a(0)|} \int_{D_a(x)} u(y) dD_y. \tag{4.41}$$

We first use cylinders aligned with the first axis, using the axis-aligned coordinates (x_1, x_2, x_3), and consider only the case $\Omega_a \Subset \Omega$ with functions defined on $H^1(\Omega_a) \coloneqq H^1(\Omega)\big|_{\Omega_a}$ (the extension to $H^1_0(\Omega)$ is trivial). The results hold also in the case $H^1_{B_e}(\Omega_a)$, which can be proven with few changes.

The following lemma is used to prove that, if u is C^∞, then $\bar{u} \in H^1(\Gamma)$; later we shall extend the result to H^1 using density.

Lemma 4.4.1. *Let $u \in C^\infty(\bar{\Omega}_a)$, then the following inequalities hold:*

$$\|\bar{u}\|_\Gamma \leq \frac{1}{\sqrt{\pi}a}\|u\|_{\Omega_a} \tag{4.42}$$

$$\|\nabla_\Gamma \bar{u}\|_\Gamma \leq \frac{1}{\sqrt{\pi}a^2}\|\nabla u\|_{\Omega_a} \tag{4.43}$$

Proof. Since we use C^∞, we can consider their restrictions: the first coordinate, x_1, shall be used for the cylinder axis, while x_2 and x_3 are normal to the axis. The first inequality can be proven directly, using either Jensen's or Hölder inequality:

$$\begin{aligned}\|\bar{u}\|_\Gamma^2 &= \frac{1}{(\pi a^2)^2}\int_\Gamma \left(\int_{D_a(x_1)} u(x_1,x_2,x_3)dx_2dx_3\right)^2 dx_1 \\ &\leq \frac{1}{\pi a^2}\int_\Gamma\int_{D_a(x_1)} u(x_1,x_2,x_3)^2 dx_2dx_3dx_1 = \frac{1}{\pi a^2}\|u\|_{\Omega_a}^2\end{aligned}$$

The second inequality can be proven in a similar fashion, after recognizing that, for every $x \in \Gamma$, the integral domain $D_a(x_1)$, used to compute $\bar{u}(x)$ does not depend on the variable x_1, and that $|\nabla_\Gamma \bar{u}(x)|^2 = |\partial_{x_1}\bar{u}(x)|^2$.

Define $D \coloneqq \{x_2, x_3 \text{ s.t. } x_2^2 + x_3^2 \leq a^2\}$, then:

$$\begin{aligned}\|\nabla_\Gamma \bar{u}\|_\Gamma^2 &= \frac{1}{(\pi a^2)^2}\int_\Gamma \left(\partial_{x_1}\int_{D_a(x_1)} u(x_1,x_2,x_3)dx_2dx_3\right)^2 dx_1 \\ &= \frac{1}{(\pi a^2)^2}\int_\Gamma \left(\partial_{x_1}\int_{(x_2,x_3)\in D} u(x_1,x_2,x_3)dx_2dx_3\right)^2 dx_1 \\ &= \frac{1}{(\pi a^2)^2}\int_\Gamma \left(\int_{(x_2,x_3)\in D} \partial_{x_1}u(x_1,x_2,x_3)dx_2dx_3\right)^2 dx_1 \\ &\leq \frac{1}{\pi a^2}\int_\Gamma\int_{(x_2,x_3)\in D} \partial_{x_1}u(x_1,x_2,x_3)^2 dx_2dx_3dx_1 \leq \frac{1}{\pi a^2}\|\nabla u\|_{\Omega_a}^2\end{aligned}$$

□

Proposition 4.4.1. *Consider a straight cylinder Ω_a; then*

$$u \in H^1(\Omega_a) \Rightarrow \bar{u} \in H^1(\Gamma).$$

Proof. We begin considering $u \in C^\infty(\bar{\Omega}_a)$. Using Lemma 4.4.1 we obtain:

$$\|\bar{u}\|_{H^1(\Gamma)} \leq \frac{1}{\sqrt{\pi}a}\|u\|_{H^1(\Omega_a)},$$

therefore $\bar{u} \in H^1(\Gamma)$.

Consider now a generic $u \in H^1(\Gamma)$, then there exists a sequence of functions $(u_n)_{n\in\mathbb{N}} \subset C^\infty(\bar{\Omega}_a)$ such that $u_n \xrightarrow{H^1} u$, and we conclude with a density argument. □

We now want to extend Proposition 4.4.1 to the case of a tubular neighborhood obtained through a diffeomorphism Φ satisfying the hypotheses at the beginning of this section.

To generalize our result on straight cylinders we use the following Lemma:

Lemma 4.4.2. *Given Φ diffeomorphism with the above hypotheses, for every $u \in H^1(\widetilde{\Omega}_a)$:*

$$\|u \circ \Phi\|_{H^1(\Omega_a)} \leq C_J^{1/2} C_\Phi \|u\|_{H^1(\widetilde{\Omega}_a)} \tag{4.44}$$

$$\|\bar{u}\|_{H^1(\widetilde{\Gamma})} \leq (C_J C_\Phi)^{3/2} \|\overline{u \circ \Phi}\|_{H^1(\Gamma)}, \tag{4.45}$$

where we defined the following positive constants:

$$C_J \coloneqq \sup_{z \in \widetilde{\Omega}_a} |J\Phi^{-1}(z)| \qquad C_\Phi \coloneqq \sup_{z \in \widetilde{\Omega}_a} |\nabla \Phi(z)|.$$

Proof. The proof for the two inequality is, essentially, a change of variables using Φ^{-1}. For the first inequality, we begin with the L^2 part of the norm:

$$\|u \circ \Phi\|^2_{\Omega_a} = \int_{\Omega_a} (u \circ \Phi)^2 d\Omega = \int_{\widetilde{\Omega}_a} u^2 |J\Phi^{-1}| d\widetilde{\Omega} \leq C_J \|u\|^2_{\widetilde{\Omega}_a}.$$

Similarly, for $\|\nabla(u \circ \Phi)\|^2_{\Omega_a}$:

$$\begin{aligned}
\|\nabla(u \circ \Phi)\|^2_{\Omega_a} &= \int_{\Omega_a} (\nabla(u \circ \Phi)(z))^2 d\Omega_z \\
&= \int_{\Omega_a} (\nabla\Phi(z)^T \nabla u(\Phi(z)))^2 d\Omega_z \leq C_\Phi^2 \int_{\Omega_a} (\nabla u(\Phi(z)))^2 d\Omega_z \\
&= \int_{\widetilde{\Omega}_a} (\nabla u(\tilde{z}))^2 \, |J\Phi^{-1}| d\widetilde{\Omega} \leq C_J C_\Phi^2 \|u\|^2_{\widetilde{\Omega}_a}.
\end{aligned}$$

To prove the second inequality we split the change of variables, considering the restrictions of Φ first to the disks, and then to the centerline of the tubular neighbourhood. The symbols we use for these restrictions are, respectively, Φ_D and Φ_Γ. With a slight abuse of notation we avoid to describe the exact disk used for the restriction.

$$\begin{aligned}
\|\bar{u}\|^2_{\widetilde{\Gamma}} = \int_{\widetilde{\Gamma}} \bar{u}(\tilde{x})^2 d\tilde{x} &= \int_{\widetilde{\Gamma}} \left(\frac{1}{\pi a^2} \int_{D_{a,\widetilde{\Gamma}}(\tilde{x})} u(\tilde{y}) dD\tilde{y} \right)^2 dx \\
&= \int_{\widetilde{\Gamma}} \left(\frac{1}{\pi a^2} \int_{\Phi^{-1}\left(D_{a,\widetilde{\Gamma}}(\tilde{x})\right)} u(\Phi(y)) |J\Phi_D| dDy \right)^2 d\tilde{x} \\
&= \int_{\Gamma} \left(\frac{1}{\pi a^2} \int_{\Phi^{-1}\left(D_{a,\widetilde{\Gamma}}(\Phi(x))\right)} u(\Phi(y)) |J\Phi_D| dDy \right)^2 |J\Phi_\Gamma| dx \\
&\leq C_J^3 \int_{\Gamma} \left(\frac{1}{\pi a^2} \int_{D_{a,\Gamma}(x)} u(\Phi(y)) dDy \right)^2 dx = C_J^3 \|\overline{u \circ \Phi}\|^2_{\Gamma},
\end{aligned}$$

where we used the fact that, by hypothesis, $\Phi^{-1}\left(D_{a,\widetilde{\Gamma}}(\Phi(x))\right) = u(\Phi(y))$. To conclude we apply the same procedure to the gradient norm.

□

Proposition 4.4.2. *Consider a cylinder $\widetilde{\Omega}_a$, and Φ with properties above; then:*

$$u \in H^1(\widetilde{\Omega}_a) \Rightarrow \bar{u} \in H^1(\widetilde{\Gamma}).$$

Proof. Notice that $u \circ \Phi$ is defined on a straight cylinder.

Combining Lemma 4.4.2 and Lemma 4.4.1:

$$\begin{aligned}\|\bar{u}\|_{H^1(\tilde{\Gamma})} \leq (C_J C_\Phi)^{3/2} \|\overline{u \circ \Phi}\|_{H^1(\Gamma)} &\leq \frac{(C_J C_\Phi)^{3/2}}{\sqrt{\pi} a} \|u \circ \Phi\|_{H^1(\Omega_a)} \\ &\leq \frac{C_J^2 C_\Phi^{5/2}}{\sqrt{\pi} a} \|u \circ \Phi\|_{H^1(\tilde{\Omega}_a)},\end{aligned}$$

which concludes the proof. □

4.4.2 Gradient equality

In the case of straight cylinders, after aligning the axis, it is possible to prove the following equation:

$$\frac{1}{2}(\delta \mathbb{C}_f \nabla Ew, \nabla Ew)_{\Omega_a} = \frac{c_\Gamma}{2}(\delta \mathbb{C}_f \nabla_\Gamma w, \nabla_\Gamma w)_\Gamma, \tag{4.46}$$

where $c_\Gamma \coloneqq \pi a^2$.

This is not true in general but, as we shall prove, it is possible to define a function g defined on Γ, which allows to reduce the integral to a one-dimensional one, by describing the geometry of the tubular neighbourhood.

Proposition 4.4.3. *For a general tubular neighborhood Ω_a of a curve Γ, for which curvature κ and torsion τ are defined a.e., there exists a function g defined on Γ satisfying:*

$$\frac{1}{2}(\delta \mathbb{C}_f \nabla Ew, \nabla Ew)_{\Omega_a} = \frac{c_\Gamma}{2}(g \delta \mathbb{C}_f \nabla_\Gamma w, \nabla_\Gamma w)_\Gamma. \tag{4.47}$$

Proof. Let $\omega \colon I \to \Gamma$ be the arclength parametrization of Γ; the proof is based on the idea of computing $(\delta \mathbb{C}_f \nabla Ew, \nabla Ew)_{\Omega_a}$ on the reference described in Subsection 4.2.2.

In this reference, consider an element $w \in H^1(\Gamma)$, and its extension as constant on each orthogonal disk Ew:

$$\begin{aligned}\nabla Ew &\coloneqq \frac{dEw}{dr}\boldsymbol{e}_r + \frac{1}{r}\frac{dEw}{d\theta}\boldsymbol{e}_\theta + \frac{1}{1 - \kappa(s) r \cos(\theta - \hat{\theta})}\frac{dEw}{ds}\boldsymbol{e}_s \\ &= \frac{1}{1 - \kappa(s) r \cos(\theta - \hat{\theta})}\frac{dEw}{ds}\boldsymbol{e}_s.\end{aligned}$$

For every vector component of the gradient (for simplicity we avoid adding a specific

index):

$$\begin{aligned}
(\delta\mathbb{C}_f\nabla Ew, \nabla Ew)_{\Omega_a} &= \int_{\Omega_a} \delta\mathbb{C}_f\nabla Ew\nabla Ew d\Omega \\
&= \int_\Gamma \left(\int_{D_a(s)} \delta\mathbb{C}_f \nabla_\Gamma w(s) \nabla_\Gamma w(s) dr d\theta \right) ds \\
&= \int_\Gamma \left(\int_{D_a(s)} \frac{1}{1-\kappa(s) r\cos(\theta-\hat{\theta})} \delta\mathbb{C}_f \frac{d}{ds} w(s) \frac{d}{ds} w(s) dr d\theta \right) ds \\
&= \int_\Gamma \delta\mathbb{C}_f \nabla_\Gamma w(s) \nabla_\Gamma w(s) \underbrace{\left(\int_{D_a(s)} \frac{1}{1-\kappa(s) r\cos(\theta-\hat{\theta})} dr d\theta \right)}_{=:\, g(s)\pi a^2} ds \\
&= (g\delta\mathbb{C}_f\nabla_\Gamma w, \nabla_\Gamma w)_\Gamma.
\end{aligned}$$

Where we define the function:

$$g(s) \coloneqq \frac{1}{\pi a^2} \int_{D_a(s)} \frac{1}{1-\kappa(s) r\cos(\theta-\hat{\theta})} dr d\theta.$$

□

Once g is computed, it is possible to compute the energy $(\delta\mathbb{C}_f\nabla Ew, \nabla Ew)_{\Omega_a}$ using a linear integration. In the case of a straight cylinder $\kappa(s) \equiv 0$, and we obtain again Equation 4.46.

The condition $r \leq a < \max_s 1/\kappa(s)$ is fundamental for the definition of g on the whole disk. Assume the curvature $\kappa(s)$ to be continuous, then $g(s)$ is continuous and there exist a constant $C_g > 0$ such that $g(s) \geq C_g > 0$.

To conclude, under the previous hypotheses on the domains, we introduce the extension operator:

$$\begin{aligned}
\bar{E} \colon H^1(\tilde{\Gamma}) &\to H^1(\tilde{\Omega}), \\
w &\to w \circ P_\Gamma,
\end{aligned}$$

where P_Γ is the geometric projection from the domain $\tilde{\Omega}$ to $\tilde{\Gamma}$; for every $y \in \tilde{\Omega}$, $P_\Gamma y$ is the element of $\tilde{\Gamma}$ such that:

$$\operatorname{dist}(y, P_\Gamma y) \leq \operatorname{dist}(y, x) \qquad \text{for every } x \in \tilde{\Gamma}.$$

As a consequence of our hypotheses on the fiber, for every $x \in \tilde{\Omega}$, we have $x \in D_a(P_\Gamma(x))$.

The operator $\bar{E}$ is clearly bounded in L^2, to prove it is bounded in H^1 we can use a strategy analogous to the one used in the proof of Proposition 4.4.3.

4.4.3 Modellistic Hypotheses

After proving that the average operator is continuous, we can finally describe the modellistic hypotheses we make on our model.

First modellistic hypothesis: fixed radius We assume the fiber's radius a to remain constant. This implies that the fiber displacement w is an element of $\hat{W}$, where:

$$\hat{W} \coloneqq \{w \in W\colon \text{ for a.e. } x \in \Gamma, \bar{E}\bar{w} = w \text{ a.e. on } D_a(x)\}. \tag{4.48}$$

It is useful to introduce the space of functions of V which, restricted to Ω_a, belong to $\hat{W}$:

$$\hat{V} \coloneqq \{v \in V\colon v\big|_{\Omega_a} \in \hat{W}\}.$$

Given $\hat{w} \in \hat{W}$, Equation 4.48 suggests the introduction of the following space:

$$W_\Gamma \coloneqq \left(H^1_{0,B_e}(\Gamma)\right)^d. \tag{4.49}$$

The following Proposition is an immediate consequence of the choice of our spaces:

Proposition 4.4.4. *The spaces W_Γ and $\hat{W}$ are isomorphic.*

Second modellistic hypothesis: average non-slip condition Consider $u \in V$ and $w \in W_\Gamma$, it is well known that in the case of thin fiber it is not possible to consider the classic non-slip condition:

$$u\big|_\Gamma = w.$$

With the results of Section 4.4.1 we can formulate following **average non-slip** condition:

$$\bar{u} = w. \tag{4.50}$$

As a second modellistic hypothesis we assume that the average non-slip condition can replace the non-slip condition.

As a justification, notice that when considering $w \in \hat{W}, u \in \hat{V}$, the classic non-slip condition, described in Equation 4.14, is equivalent to Equation 4.50. Moreover, with continuous function and a small radius a, the two conditions can be considered approximately the same.

4.4.4 Problem formulation

We assume the Modellistic Hypotheses described in the previous section hold:

- the fiber displacement w is contained in $\hat{W}$ (we shall use W_Γ, since it is isomorphic to $\hat{W}$),
- we can impose the coupling between fibers and elastic matrix through the average non-slip condition .

This allows us to modify Problem 14 for the $3D-1D$ case; the weak formulation of the average non-slip constraint on Γ is:

$$\langle q, \bar{u} - w\rangle = 0 \qquad \forall\ q \in Q_\Gamma,$$

with $Q_\Gamma \coloneqq H^{-1}(\Gamma)^d$.

For every $(u, w, q) \in V \times W_\Gamma \times Q_\Gamma$, we define the energy functional of our problem as:

$$\psi_T(u, w, \lambda) = \frac{1}{2}(\mathbb{C}_\Omega \nabla u, \nabla u)_\Omega + c_\Gamma(g\delta \mathbb{C}_f \nabla_\Gamma w, \nabla_\Gamma w)_\Gamma + (q, \bar{u} - w)_\Gamma - (b, u)_\Omega.$$

Then the saddle point problem becomes:

$$u, w, \lambda = \arg \inf_{\substack{u \in V \\ w \in W_\Gamma}} \left(\arg \sup_{\lambda \in Q_\Gamma} \psi_T(u, w, \lambda) \right). \tag{4.51}$$

Using the Euler-Lagrange equations as in Section 4.3.2, we obtain:

Problem 16 (1D-3D Weak Formulation)**.** *Given $b \in L^2(\Omega)^d$, find $u \in V, w \in W_\Gamma, \lambda \in Q_\Gamma$ such that:*

$$\begin{aligned} (\mathbb{C}_\Omega \nabla u, \nabla v)_\Omega + \langle \lambda, \bar{v} \rangle &= (b, v)_\Omega && \forall v \in V, \\ c_\Gamma(g\delta \mathbb{C}_f \nabla_\Gamma w, \nabla_\Gamma y)_\Gamma - \langle \lambda, y \rangle &= 0 && \forall y \in W_\Gamma, \\ \langle q, \bar{u} - w \rangle &= 0 && \forall q \in Q_\Gamma. \end{aligned}$$

We now re-define the space $\mathbb{V} \coloneqq V \times W_\Gamma$, and its norm: for every $(v, y) \in \mathbb{V}$, $\|(v, y)\|_\mathbb{V} \coloneqq \|\cdot\|_V + \|\cdot\|_{W_\Gamma}$. We define the operators for the tubular fiber Ω_a:

$$\begin{aligned} \mathbb{F}_T &\coloneqq (\mathbb{C}_\Omega \nabla u, \nabla v)_\Omega + c_\Gamma(g\delta \mathbb{C}_f \nabla_\Gamma w, \nabla_\Gamma w)_\Gamma, \\ \mathbb{E}_T &\coloneqq \langle q, \bar{u} - w \rangle, \end{aligned}$$

obtaining a new saddle-point problem, which we study using the $inf - sup$ conditions.

4.4.5 Well-posedness, existence and uniqueness

Similarly to the three-dimensional case, to prove the well-posedness and the existence and uniqueness of a solution we prove the following sufficient condition: there exist two positive constants $\alpha_5 > 0, \alpha_6 > 0$ such that

$$\begin{aligned} \inf_{q \in Q_\Gamma} \sup_{\boldsymbol{v} \in \mathbb{V}} \frac{\mathbb{E}(\boldsymbol{u}, q)}{\|\boldsymbol{u}\|_\mathbb{V} \, \|q\|_{Q_\Gamma}} &\geq \alpha_5 \\ \inf_{\boldsymbol{u} \in \ker \mathbb{E}} \sup_{\boldsymbol{v} \in \ker \mathbb{E}} \frac{\mathbb{F}(\boldsymbol{u}, \boldsymbol{v})}{\|\boldsymbol{u}\|_\mathbb{V} \, \|\boldsymbol{v}\|_\mathbb{V}} &\geq \alpha_6, \end{aligned}$$

Following a procedure similar to the one of Equation 4.26:

$$\begin{aligned} \ker \mathbb{E}_T &\coloneqq \{\boldsymbol{v} \coloneqq (v, w) \in \mathbb{V} \colon \langle q, \bar{v} - y \rangle = 0 \ \forall q \in Q_\Gamma\} \\ &= \{\boldsymbol{v} \coloneqq (v, w) \in \mathbb{V} \colon \bar{v} = w \ a.e. \ on \ \Omega_a\}. \end{aligned}$$

With a variation on the proof of Proposition 4.4.2, it is possible to prove that there exists a positive constant $c_b > 0$, such that:

$$\bar{v} = w \ a.e. \ on \ \Omega_a \Rightarrow \|w\|_\Gamma = \|\bar{v}\|_\Gamma \leq c_b \|v\|_{\Omega_a}$$

These conditions can be proved with the following Propositions:

Proposition 4.4.5. *There exists a constant $\alpha_5 > 0$ such that:*

$$\inf_{q\in Q_\Gamma}\sup_{(v,w)\in\mathbb{V}}\frac{\langle q,\bar{v}-w\rangle}{\|(v,w)\|_{\mathbb{V}}\|q\|_{Q_\Gamma}}\geq\alpha_5.$$

Moreover $\alpha_5 = 1$.

Proof. The non slip condition is given by the duality pairing between $Q_\Gamma = W_\Gamma'$ and W; by definition of the norm in the dual space Q_Γ:

$$\begin{aligned}\|q\|_{Q_\Gamma} &= \sup_{w\in W_\Gamma}\frac{\langle q,w\rangle}{\|w\|_W}\\ &\leq \sup_{v\in V,w\in W_\Gamma}\frac{\langle q,\bar{v}-w\rangle}{(\|w\|_W^2+\|v\|_V^2)^{\frac{1}{2}}},\end{aligned}$$

where the last inequality can be proven fixing $v = 0$.

The final statement is found dividing by $\|q\|_{Q_\Gamma}$ and taking the $\inf_{q\in Q}$. □

Proposition 4.4.6. *Assume $\mathbb{C}_\Omega$ and $\mathbb{C}_f$ to be strongly elliptical, with constants c_Ω and c_f respectively such that $c_\Omega > c_f > 0$; there exists a constant $\alpha_6 > 0$ such that:*

$$\inf_{(u,w)\in\ker\mathbb{E}_T}\sup_{(v,y)\in\ker\mathbb{E}_T}\frac{(\mathbb{C}_\Omega\nabla v,\nabla u)_\Omega+c_\Gamma(g\delta\mathbb{C}_f\nabla_\Gamma y,\nabla_\Gamma w)_\Gamma}{\|(v,y)\|_{\mathbb{V}}\|(u,w)\|_{\mathbb{V}}}\geq\alpha_6.$$

Proof. For every $(u, w) \in \ker(\mathbb{E}_T)$:

$$\begin{aligned}\sup_{(v,y)\in\ker(\mathbb{E}_T)}&\frac{(\mathbb{C}_\Omega\nabla u,\nabla v)_\Omega+c_\Gamma(g\delta\mathbb{C}_f\nabla_\Gamma w,\nabla_\Gamma y)_\Gamma}{\|(v,y)\|_{\mathbb{V}}}\\ &\geq\frac{c_\Omega(\nabla u,\nabla u)_\Omega+c_\Gamma C_g(c_f-c_\Omega)(\nabla_\Gamma w,\nabla_\Gamma w)_\Gamma}{\|(u,w)\|_{\mathbb{V}}}\\ &\geq\frac{c_p c_\Gamma c_b\min(c_\Omega,C_g(c_f-c_\Omega))}{2}\frac{(u,u)_{H^1(\Omega)}+(w,w)_{H^1(\Gamma)}}{\|(u,w)\|_{\mathbb{V}}}\\ &\geq\alpha_6\|(u,w)\|_{\mathbb{V}},\end{aligned}$$

with $\alpha_6 \coloneqq \frac{c_p c_\Gamma c_b\min(c_\Omega,C_g(c_f-c_\Omega))}{2}$.

The result is obtained dividing and considering the $\inf_{(u,w)\in\ker\mathbb{E}_T}$.

□

Using the saddle point theory we conclude that, under our modellistic assumptions, Problem 16 is well-posed, and has a unique solution.

Remark. *An analogous of Proposition 4.3.3 holds in the $3D-1D$ case; its proof uses the same arguments and Equality 4.4.3.*

4.4.6 Finite element discretization

The discretization of Problem 16 for thin fibers follows the steps of Section 4.3.4, on the domains Ω and Γ (the fiber's one-dimensional core). Consider two independent discretizations for these domains; the family $\mathcal{T}_h(\Omega)$ of regular meshes in Ω, and a family

$\mathcal{T}_h(\Gamma)$ of regular meshes in Γ. We assume no geometrical error is committed when meshing. We consider two independent finite element discretizations $V_h \subset V$, $W_h \subset W_\Gamma$.

Similarly to the three-dimensional case, the duality paring between Q_Γ and W_Γ can be represented using the L^2 scalar product in Γ:

$$\langle q, w \rangle = (q, w)_\Gamma.$$

Thus we discretize the duality pairing as the L^2 product, and consider $Q_h = W_h$. To handle the restriction of functions from V_h to W_h, we introduce the L^2 projection:

$$P_W \colon W_\Gamma \to W_h \subset L^2(\Gamma),$$

which is continuous: for every $w \in W_\Gamma$

$$\|P_W w\|_{L^2(\Gamma)} \leq \|w\|_\Gamma.$$

This condition is too weak for our Problem, which involves the gradient. We make the further assumption that the restriction $P_W\big|_W \neq 0$, and is H^1-stable, i.e., there exists a positive constant c, such that for all $w \in W_\Gamma$:

$$\|\nabla_\Gamma P_W w\|_\Gamma \leq c \|\nabla_\Gamma w\|_\Gamma. \tag{4.52}$$

We assume that the H^1-stability property holds uniformly on $\mathcal{T}_h(\Omega)$ and $\mathcal{T}_h(\Omega_f)$, provided that the mesh size h is comparable on the two domains.

We can now write the discretization of Problem 16:

Problem 17 (1D-3D Discretized Weak Formulation)**.** *Given $b \in L^2(\Omega)^d$, find $u_h \in V_h, w_h \in W_h, \lambda_h \in Q_h$ such that:*

$$(\mathbb{C}_\Omega \nabla u_h, \nabla v_h)_\Omega + (\lambda_h, \bar{v}_h)_\Gamma = (b, v_h)_\Omega \qquad \forall v_h \in V_h, \tag{4.53}$$

$$c_\Gamma (g\delta \mathbb{C}_f \nabla_\Gamma w_h, \nabla_\Gamma y_h)_\Gamma - (\lambda_h, y_h)_\Gamma = 0 \qquad \forall y_h \in W_h, \tag{4.54}$$

$$(q_h, \bar{u}_h - w_h)_\Gamma = 0 \qquad \forall q_h \in Q_h. \tag{4.55}$$

To study the saddle-point problem we define the Hilbert space $\mathbb{V}_h \coloneqq V_h \times W_h$, with its norm, and the operators

$$\begin{aligned}
\mathbb{F}_{T,h} &\colon \mathbb{V}_h \times \mathbb{V}_h \longrightarrow \mathbb{R} \\
&\qquad (\boldsymbol{u}_h, \boldsymbol{v}_h) \longmapsto (\mathbb{C}_\Omega \nabla u_h, \nabla v_h)_\Omega + c_\Gamma (g\delta \mathbb{C}_f \nabla_\Gamma w_h, \nabla_\Gamma y_h)_\Gamma, \\
\mathbb{E}_{T,h} &\colon \mathbb{V}_h \times Q_h \longrightarrow \mathbb{R} \\
&\qquad (\boldsymbol{u}_h, q_h) \longmapsto (q_h, \bar{v}_h - w_h)_\Gamma.
\end{aligned}$$

Then we can rewrite problem 17 in operatorial form, showing its saddle-point structure: find $\boldsymbol{u}_h \in \mathbb{V}_h, \lambda_h \in Q_h$ such that:

$$\begin{aligned}
\mathbb{F}_{T,h}(\boldsymbol{u}_h, \boldsymbol{v}_h) + \mathbb{E}_{T,h}(\boldsymbol{v}_h, \lambda_h) &= (b_h, v_h)_\Omega \qquad && \forall\, \boldsymbol{v}_h \coloneqq (v_h, y_h) \in \mathbb{V}_h \\
\mathbb{E}_{T,h}(q_h, \boldsymbol{u}_h) &= 0 && \forall\, q_h \in Q_h.
\end{aligned}$$

Proposition 4.4.7. *There exists a constant $\alpha_7 > 0$, independent of h, such that:*

$$\inf_{q_h \in W_h} \sup_{(v_h, w_h) \in \mathbb{V}_h} \frac{(q_h, \bar{v}_h - w_h)_\Gamma}{\|(v_h, w_h)\|_{\mathbb{V}} \|q_h\|_\Gamma} \geq \alpha_7.$$

Proof. The proof is similar to the one of Proposition 4.3.4. For every $q_h \in Q_h$, by definition of the Q norm, representing the duality product as a scalar product in L^2, there exists $\hat{w} \in W$ such that:

$$\|q_h\|_Q = \sup_{w \in W} \frac{(q_h, w)_\Gamma}{\|w\|_Q} = \frac{(q_h, \hat{w})_\Gamma}{\|\hat{w}\|_Q} = \frac{(q_h, P_W \hat{w})_\Gamma}{\|\hat{w}\|_Q}. \tag{4.56}$$

Using well-known properties of P_W, and the H^1 stability, we can prove there exists a c such that:

$$\|P_W \hat{w}\|_Q \leq c \|\hat{w}\|_Q.$$

Inserting this in Equation 4.56:

$$\begin{aligned}\|q_h\|_Q &\leq \frac{(q_h, P_W \hat{w})_\Gamma}{\|\hat{w}\|_Q} \leq c \frac{(q_h, P_W \hat{w})_\Gamma}{\|P_W \hat{w}\|_Q} \\ &\leq c \sup_{w_h \in W_h} \frac{(q_h, w_h)_\Gamma}{\|w_h\|_Q} \leq c \sup_{v_h \in V_h, w_h \in W_h} \frac{(q_h, v_h\big|_{\Omega_a} - w_h)_\Gamma}{(\|w_h\|_Q^2 + \|v_h\|_V^2)^{\frac{1}{2}}},\end{aligned}$$

and we conclude as in Proposition 4.3.1. □

To study the inf-sup condition for $\mathbb{F}_{T,h}$ define:

$$\begin{aligned}\ker(\mathbb{E}_{T,h}) &:= \{\boldsymbol{v}_h \in \mathbb{V}_h \colon (q_h, \bar{v}_h - w_h)_\Gamma = 0 \qquad \forall q_h \in Q_h\} \\ &= \{\boldsymbol{v} \in \mathbb{V}_h \colon (q_h, \bar{v}_h)_\Gamma = (q_h, w_h)_\Gamma \qquad \forall q_h \in Q_h\}.\end{aligned}$$

Proposition 4.4.8. *Assume $\mathbb{C}_\Omega$ and $\mathbb{C}_f$ to be strongly elliptical, with constants c_Ω and c_f respectively such that $c_f > c_\Omega > 0$; then there exists a constant $\alpha_8 > 0$, independent of h, such that:*

$$\inf_{(u_h, w_h) \in \ker \mathbb{E}_{T,h}} \sup_{(v_h, y_h) \in \ker \mathbb{E}_{T,h}} \frac{(\mathbb{C}_\Omega \nabla v_h, \nabla u_h)_\Omega + c_\Gamma (g\delta \mathbb{C}_f \nabla_\Gamma y_h, \nabla_\Gamma w_h)_\Gamma}{\|(v_h, y_h)\|_{\mathbb{V}} \|(u_h, w_h)\|_{\mathbb{V}}} \geq \alpha_8.$$

Proof. The proof is almost identical to the one of Proposition 4.4.6. □

One-Dimensional Approximation The approach of Section 4.4 hides a computational difficulty: given $v_h \in V_h$, the average $\bar{v}_h$ is obtained computing a number of two-dimensional integrals, nullifying the computational advantage of using one-dimensional fibers.

Since we are using finite element functions, there is a straight-forward solution to this problem: approximate $\bar{v}_h$ with the restriction $v_h\big|_\Gamma$. This approximation is exact for every $v_h \in \hat{V}$, and can be justified by the fact that for every $v_h \in V_h$:

$$\lim_{a \to 0} \bar{v}_h(x) = v_h(x) \qquad \text{for every } x \in \Gamma,$$

coherently with the initial choice of $\bar{v}_h$ as a "substitute" of v_h.

While this approach simplifies our computations, it makes difficult the derivation of a formula for the error committed. The theoretical basis for such an estimate have been described in this paper, the estimate itself shall be the subject of future work.

4.5 Numerical validation

The analytical solution of Problems 14 and 16, even for simple configurations, is non-trivial: we chose some FRMs structures which are studied in literature, and used the known approximated solutions as a comparison for our model.

Using the deal.II library [3, 11, 67, 92], and the deal.II step-60 tutorial [62] we developed a model for thin fibers proposed in Section 3, and compared it with the Rule of Mixtures and the Halpin-Tsai configurations in some pull and push tests.

Numerical Setting For our numerical solution, we now describe how to solve Problem 17 on a collection of fibers, while reducing the system size; we begin by redefining our spaces and meshes:

- Ω: the elastic matrix, on which we build the finite element space, of dimension $N \in \mathbb{N}$:
$$V_h = \operatorname{span}\{v_i\}_{i=1}^{N} \subset H^1(\Omega).$$
- $\Gamma \subset \Omega$: the collection of $n_f \in \mathbb{N}$ fibers, i.e., $\Gamma \coloneqq \bigcup_{k=1}^{n_f} \Gamma_k$, with the finite element discretization, of dimension $M \in \mathbb{N}$::
$$W_h = \operatorname{span}\{w_a\}_{i=a}^{M} \subset H^1(\Gamma).$$
- Q_h: the space of the Lagrange multiplier, discretized using the same base of W_h.

We use i, j as indices for the space V_h and a, b as indices for the space W_h, and assume all hypotheses on spaces and meshes of Subsection 4.4.6 are satisfied.

We assume that each fiber is parametrized by $X_k \colon I_k \to \Gamma_k$, where I_k is a finite interval in $\mathbb{R}$, and $1 \le k \le n_f$. We assume for any $1 \le j < k \le n_f$ we have $I_k \cap I_k = \emptyset$. Then we can define $X \colon \bigcup_{k=1}^{n_f} I_k \to \Gamma$, which parametrizes all fibers contained in Γ. For each fiber Γ_k we define its tubular neighborhood $\Omega_{a,k}$, and define

$$\Omega_f \coloneqq \bigcup_{k=1}^{n_f} \Omega_{a,k}.$$

We define the following sparse matrices:

$$\begin{aligned}
&A \colon V_h \to V_h' && A_{ij} \coloneqq (\mathbb{C}_\Omega \nabla v_i, \nabla v_j)_\Omega, \\
&K \colon W_h \to W_h' && K_{ab} \coloneqq c_\Gamma (g \delta \mathbb{C}_f \nabla_\Gamma w_a, \nabla_\Gamma w_b)_\Gamma, \\
&B \colon V_h \to Q_h' && B_{ia} \coloneqq (v_i\big|_\Gamma, w_a)_\Gamma = (v_i \circ X, w_a)_\Gamma, \\
&L \colon W_h \to Q_h' && L_{ab} \coloneqq (w_a, w_b)_\Gamma.
\end{aligned} \tag{4.57}$$

Here B is the coupling matrix from V_h to W_h, M the mass-matrix of W_h.

After defining the vector $g_i \coloneqq (b, v_i)_\Omega$, Problem 17 can be expressed in matrix form as: find $(u, w, \lambda_h) \in V_h \times W_h \times Q_h$ such that

$$\begin{pmatrix} A & 0 & B^T \\ 0 & K & -L^T \\ B & -L & 0 \end{pmatrix} \begin{pmatrix} u \\ w \\ \lambda \end{pmatrix} = \begin{pmatrix} g \\ 0 \\ 0 \end{pmatrix}. \tag{4.58}$$

This system has size NM^2: computationally it is convenient to reduce its size. From the second line of the Block Matrix 4.58:

$$Kw = L^T\lambda = L\lambda \Rightarrow \lambda = L^{-1}Kw.$$

System 4.58 becomes:

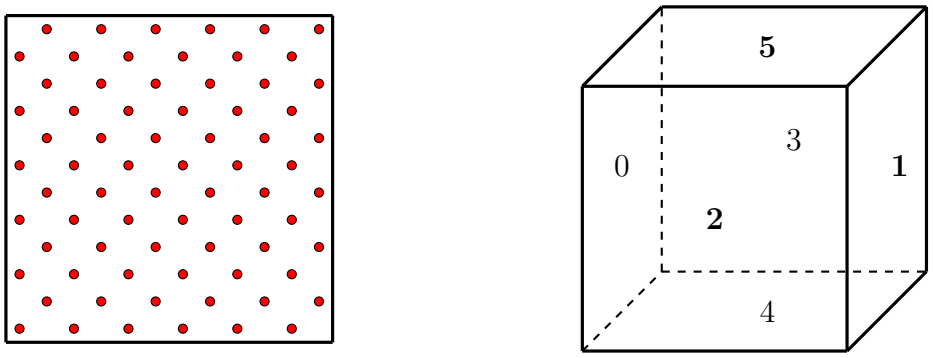
$$\begin{pmatrix} A & B^TL^{-1}K \\ B & -L \end{pmatrix} \begin{pmatrix} u \\ w \end{pmatrix} = \begin{pmatrix} g \\ 0 \end{pmatrix}. \tag{4.59}$$

We remove w using the equation $Bu = Lw \Rightarrow w = L^{-1}Bu$ and obtaining:

$$(A + B^TL^{-1}KL^{-1}B)u = (A + P_\Gamma^T K P_\Gamma)u = g, \tag{4.60}$$

where $P_\Gamma \coloneqq L^{-1}B$. Boundary conditions are imposed weakly, using Nitsche method (as in [87]).

4.5.1 Model description



(a) Section with homogeneous fibers

(b) Ω's faces numbering

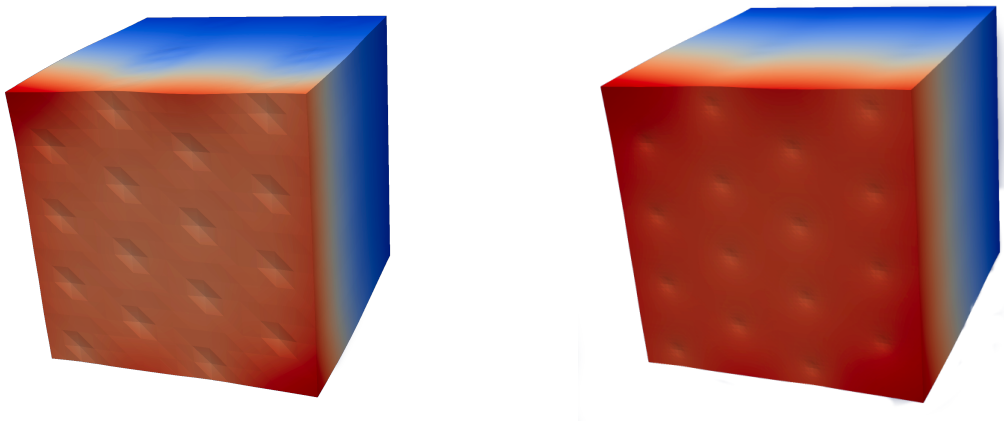
Figure 4.7: Elastic cube homogeneous structure and boundary description

The elastic matrix we consider in our tests is the unitary cube $\Omega \coloneqq [0,1]^3$. All considered meshes are only uniformly refined hexaedral meshes, and we use only piecewise bilinear finite elements.

The elastic tensors used are described in Equations 4.29 − 4.31; for the model description we use the following parameters:

- r_Ω: global refinements of the Ω mesh.
- r_Γ: global refinements of the Γ mesh.
- λ_Ω, μ_Ω: Lamé parameters for the elastic matrix.
- λ_f, μ_f: Lamé parameters for the fibers.
- β: fiber volume ratio or representative volume element (RVE), i.e., $\beta = |\Omega_f|/|\Omega|$.
- a: the radius of the fibers.

For the boundary conditions we refer to Figure 4.7b.



(a) Pull test example, with $r_\Omega = 4$

(b) Pull test example, with $r_\Omega = 6$

Figure 4.8: Pull test: comparison on the refinement level of the triangulation for Ω

4.5.2 Homogeneous fibers

In our first test we consider a unidirectional composite, where fibers are uniform in properties and diameter, continuous, and parallel throughout the composite Ω (see Figure 4.7a).

We compare the results obtained with our model with the ones obtained using the Rule of Mixtures [44, 2], which agrees with experimental tests especially for tensile loads, and when the fiber ratio β is small.

The composite stress-strain equation, under the condition $u\big|_{\Omega_f} = w$, is:

$$S[u, w] = \frac{1}{2}(\mathbb{C}_\Omega Eu, Eu)_\Omega + \frac{1}{2}(\delta\mathbb{C}_f Ew, Ew)_{\Omega_f}.$$

Using the Rule of Mixtures we can approximate the integral over Ω_a with one over Ω:

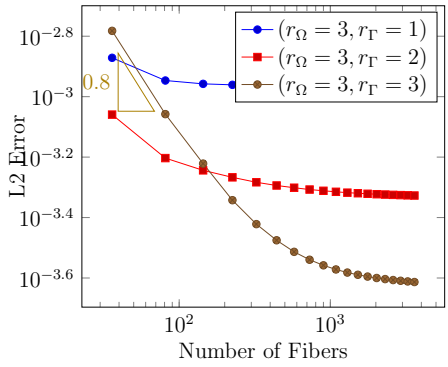
$$(\delta\mathbb{C}_f Ew, Ew)_{\Omega_f} \approx \beta(\delta\mathbb{C}_f Eu, Eu)_\Omega,$$

obtaining:

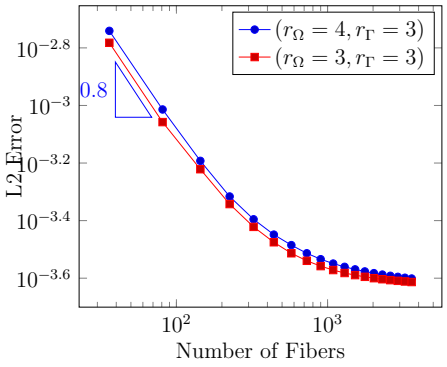
$$S[u] = \frac{1}{2}(\mathbb{C}_\Omega Eu, Eu)_\Omega + \beta\frac{1}{2}(\delta\mathbb{C}_f Eu, Eu)_\Omega. \tag{4.61}$$

Multiple tests were run, keeping β constant, while increasing the fiber density and reducing the fiber diameter; we expect this process to render the coupled model solution increasingly close to the homogenized one.

Comparing solutions Figures 4.8a and 4.8b illustrate the influence of Ω's refinement on the final result, when using few fibers on a pull test. Stiff fibers oppose being stretched, deforming the elastic matrix Ω through the non-slip condition: near each fiber, the deformation of Ω should be symmetrical, resembling a cone. This effect is better described in Figure 4.8b, where the higher value of r_Ω results in greater geometrical flexibility of the elastic matrix, allowing a better description of the effect of each fiber. Lower values of r_Ω result in a non symmetrical solution, as in Figure 4.8a. The lower geometrical flexibility results in an "averaged" solution which, in the case of few fibers,



(a) Pull Test, $\lambda = \lambda_F = 0.4$, $\mu = 1, \mu_F = 1000, \beta = 0.1$.



(b) Pull Test, $\lambda = \lambda_F = 0.4, \mu = 1$, $\mu_F = 1000, r_\Omega = 3, r_\Gamma = 3$.

Figure 4.9: Pull tests with varying refinements.

is closer to the homogenized model. To describe the slope in the plots, let $\mathbf{e}_1, \mathbf{e}_2$ be the L^2 errors and $n_{f,1}, n_{f,2}$ are the fiber numbers for two subsequent simulations; to compute the slope we then apply the formula:

$$\frac{\ln(\mathbf{e}_2/\mathbf{e}_1)}{\ln(n_{f,2}/n_{f,1})}.$$

Pull test along fibers Dirichlet homogeneous conditions is applied to face 0, Neumann homogeneous conditions is applied to faces $2, 3, 4, 5$. In the Push Test, the Neumann condition 0.05 is applied to face 1. For the Pull Test, the value -0.05 is applied to the same face. Boundary conditions are applied only to $\partial\Omega$; the fibers interact through the coupling with the elastic matrix.

We report here only data from pull tests, as push tests gave comparable results.

The use of the projection matrix $P_\Gamma \colon V_h \to W_h$, and the error estimate for the fully three-dimensional case (Inequality 4.38), both suggest that the solution quality on the elastic matrix depends on both V_h and W_h. This is apparent in Figure 4.9a, where for $r_\Gamma = 1$ the mesh of Γ is unable to describe the stretch of the material, resulting in the error remaining approximately constant after a certain fiber density is reached. A similar behaviour emerges in the case $r_\Gamma = 2$.

In a similar manner Figure 4.9b shows that refining only the elastic matrix does not improve the solution quality: as the number of fibers increases, the error converges to approximately the same value, which is limited by r_Γ.

Figure 4.10 shows an error comparison as the value of μ_F varies: as expected our model is better suited for stiff fibers.

4.5.3 Halpin-Tsai equations

For our random-test we compare our model to the Halpin-Tsai equations: and empirical set of equations introduced in [43]; we use the Halpin-Tsai equations for longitudinal moduli as described in [2]. The fibers have length l, diameter d, the fiber and the matrix Young moduli are E_f and E_m respectively, β is the volume fraction occupied by the fibers.

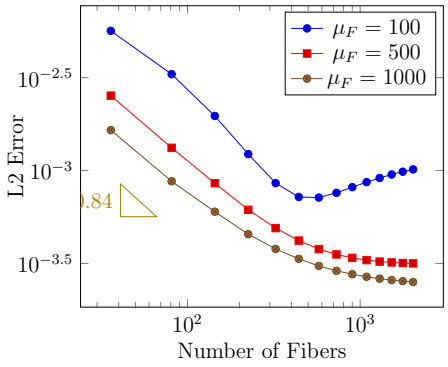


Figure 4.10: Pull Test with varying μ_F, $\lambda = \lambda_F = 0.4, \mu = 1, \beta = 0.1, r_\Omega = 3, r_\Gamma = 3$

We define two empirical constants:

$$\eta_L = \frac{(E_f/E_m) - 1}{(E_f/E_m) + (2l/d)}, \tag{4.62}$$

$$\eta_T = \frac{(E_f/E_m) - 1}{(E_f/E_m) + 2}. \tag{4.63}$$

This allows to compute the longitudinal and transverse moduli for aligned short fibers:

$$E_L = \frac{1 + (2l/d)\eta_L\beta}{1 - \eta_L\beta}, \tag{4.64}$$

$$E_T = \frac{1 + 2\eta_T\beta}{1 - \eta_T\beta}. \tag{4.65}$$

If fibers are randomly oriented in a plane the following equations can be used to predict the elastic modulus:

$$E_C = \frac{3}{8}E_L + \frac{5}{8}E_T, \tag{4.66}$$

$$\mu_C = \frac{1}{8}E_L + \frac{1}{4}E_T. \tag{4.67}$$

Since a random fiber composite is considered isotropic in its plane, the Poisson's ratio can be calculated as:

$$\nu_R = \frac{E_C}{2\mu_C} - 1. \tag{4.68}$$

The properties of this composite do not depend directly on the fiber length or radius, but on the aspect ratio l/d.

4.5.4 Random fibers

Our second test is a pull test on a more complex model: a random chopped fiber reinforced composite.

We distribute small fibers at a random point of Ω, with a random direction parallel to the $< x, y >$ plane; the fibers share the same size and properties. If a fiber surpasses the edge of Ω, it is cut.

Figure 4.11: Random fibers configuration inside the unitary cube.

Fiber length	0.6	0.4	0.3	0.25	0.225	0.2	0.18
Fiber radius	0.03	0.02	0.015	0.0125	0.01125	0.01	0.009
Number of Fibers	79	268	637	1100	1509	2149	2947

Table 4.1: Fiber Parameters

For more details on the algorithm used to distribute the fibers see the Random Sequential Adsorption algorithm [79]; our implementation generates only the plane angle, and does not implement an intersection-avoidance mechanism. As a comparison model, we estimate the material parameters using the empirical Halpin-Tsai equations.

Our test setting runs on the unitary cube, with a fiber ratio $\beta = 0.135$ and a fiber aspect ratio of $\frac{l}{2r} \approx 10$, where l is the fiber's length and r is the fiber's radius; the values used are described in Table 4.1.

We could not find an exact estimate of the error convergence, but we expect the solution to improve as the number of fibers increases because:

- the fiber radius a reduces, improving of the average non-slip condition,
- more fibers result in a more homogeneous material on the planes parallel to the $< x, y >$ plane.

Following [79], we consider a short fiber E-glass/urethane composite: the fiber and matrix Young's modulus are, respectively, $E_f = 70GPa$ and $E_m = 3GPa$, while the Poisson ratios are $\nu_f = 0.2$ and $\nu_m = 0.38$. These values were converted to the Lamé parameters using the classic formulas for hyper elastic materials.

Predicted parameters for the composite: $E_C = 2.20GPa$ and $\nu_C = 0.38GPa$; these are slightly different from [79] because, in the Halpin-Tsai equations, l/d was used instead of $2l/d$, see 4.5.3. The boundary conditions used for the pull tests are the same of Paragraph 4.5.2.

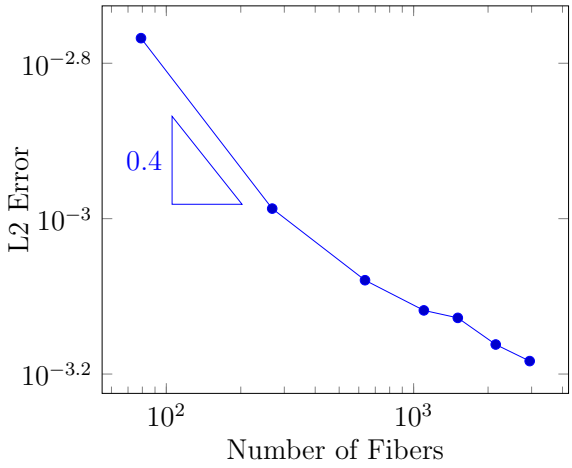


Figure 4.12: Random Pull Test with one-dimensional approximation; global refinement of Ω: 4, global refinement of Γ: 2.

We limit the global refinements of Γ, in order to obtain cells of approximately the same size on both Ω and the fibers. The results are shown in Figure 4.12: as the number of fiber increases the error reduces, but because the random fiber model is more complex than the homogeneous one, the final error achieved is higher than the one reached in the previous test.

4.6 Conclusions

This thesis deals with the mathematical and numerical modelling of non-matching coupling problems through distributed Lagrange multipliers. The results provide the numerical tools needed for fast and reliable coupling between meshes, and a reduced model for a fiber reinforced material, where fiber are approximated with one-dimensional meshes immersed inside a three-dimensional elastic matrix.

The research activities concerned both analytical and numerical tasks related to the coupling problem, with the following results:

- The implementation of a fast algorithm to compute the coupling between non-matching meshes in a serial environment.
- The first description of an algorithm for the coupling between arbitrarily distributed non-matching meshes.
- A new result proving the well posedness, existence, and uniqueness a solution for a reduced coupling problem between one-dimensional and three-dimensional meshes.

In Chapter 2 we studied a constrained Poisson equation, introducing the basic theory behind the fictitious domain method with distributed Lagrange multipliers. This example showed there are two main problems when using these methods:

- *data transfer*: all coupling algorithms require information to be accurately transferred between the two grids. This task is particularly complex in the case of independent meshes, as identifying points in the real space is an expensive operation.
- *data locality*: modern scientific simulations run on multiple processes. Data replication between a high number of processes is often infeasible, as it would require too much memory for the cluster; the typical solution to this memory problem is the use of distributed data structures. When meshes are distributed independently, a data locality problem arises, as the coupling might require information located on different processes.

To optimize the data transfer problem, we studied some search algorithms and proved their improved performance with respect to the original one implemented in deal.II version 8.5.1. To solve the data locality problem, we developed an algorithm where a local description of the mesh is generated through bounding boxes; this description is sufficiently small and efficient to be replicated on all processes through a single global communication. Once this building phases are concluded, the data locality problem can be solved through efficient one-to-one communications.

Our simulations show that our solution to the distributed problem scales well, at least when considering simulations with a few dozen cores.

In Chapter 4, we used the tools developed in the first two chapters to derive a model for a fiber reinforced material. Even with efficient coupling algorithms, a full three-dimensional discretization of fibers is often computationally too demanding; the typical solution consists in approximating fibers with one-dimensional meshes. This poses some mathematical challenges: the coupling between different meshes requires the existence of some "restriction" (or trace) operator. Since these PDEs are typically studied on Sobolev Spaces, the existence of such a trace operator is non-trivial when considering one-dimensional domains immersed in a three-dimensional one; current approaches define some ad-hoc Weighted Sobolev Spaces, in order to guarantee its existence.

We introduced a new solution to the problem, which relies on the continuity of an average operator on tubular neighbourhoods. With the introduction of some modelling hypotheses on the material, this allowed us to consider the coupling between one-dimensional and three-dimensional domains on standard Sobolev Spaces, proving well-posedness, existence, and uniqueness for the solution. To conclude we validated this approach through some numerical experiments.

Bibliography

[1] D. F. Adams and D. R. Doner. Transverse normal loading of a unidirectional composite. *Journal of composite Materials*, 1(2):152–164, 1967.

[2] B. D. Agarwal, L. J. Broutman, and K. Chandrashekhara. *Analysis and performance of fiber composites*. John Wiley & Sons, 2017.

[3] G. Alzetta, D. Arndt, W. Bangerth, V. Boddu, B. Brands, D. Davydov, R. Gassmoeller, T. Heister, L. Heltai, K. Kormann, M. Kronbichler, M. Maier, J.-P. Pelteret, B. Turcksin, and D. Wells. The `deal.II` library, version 9.0. *Journal of Numerical Mathematics*, 26(4):173–183, 2018.

[4] F. Armero and J. Simo. A new unconditionally stable fractional step method for non-linear coupled thermomechanical problems. *International Journal for numerical methods in Engineering*, 35(4):737–766, 1992.

[5] D. Arndt, W. Bangerth, D. Davydov, T. Heister, L. Heltai, M. Kronbichler, M. Maier, J.-P. Pelteret, B. Turcksin, and D. Wells. The `deal.II` library, version 8.5. *Journal of Numerical Mathematics*, 2017.

[6] F. Auricchio, D. Boffi, L. Gastaldi, A. Lefieux, and A. Reali. On a fictitious domain method with distributed lagrange multiplier for interface problems. *Applied Numerical Mathematics*, 95:36–50, 2015.

[7] I. Babuška. The finite element method with lagrangian multipliers. *Numerische Mathematik*, 20(3):179–192, 1973.

[8] I. Babuška. The finite element method with penalty. *Mathematics of computation*, 27(122):221–228, 1973.

[9] W. Bangerth, C. Burstedde, T. Heister, and M. Kronbichler. Algorithms and data structures for massively parallel generic adaptive finite element codes. *ACM Trans. Math. Softw.*, 38(2):14:1–14:28, Jan. 2012.

[10] W. Bangerth, J. Dannberg, R. Gassmöller, T. Heister, et al. ASPECT: Advanced Solver for Problems in Earth's ConvecTion, UserManual. 4 2017. doi:10.6084/m9.figshare.4865333.

[11] W. Bangerth, R. Hartmann, and G. Kanschat. deal.II – a general purpose object oriented finite element library. *ACM Trans. Math. Softw.*, 33(4):24/1–24/27, 2007.

[12] R. Barrage, S. Potapenko, and M. A. Polak. Modelling transversely isotropic fiber-reinforced composites with unidirectional fibers and microstructure. *Mathematics and Mechanics of Solids*, page 1081286519838603, 2019.

[13] R. Becker and S. Mao. An optimally convergent adaptive mixed finite element method. *Numerische Mathematik*, 111:35–54, Nov 2008.

[14] T. Behzad and M. Sain. Finite element modeling of polymer curing in natural fiber reinforced composites. *Composites Science and Technology*, 67(7-8):1666–1673, 2007.

[15] T. Belytschko and T. Black. Elastic crack growth in finite elements with minimal remeshing. *International journal for numerical methods in engineering*, 45(5):601–620, 1999.

[16] S. Berrone, A. Bonito, R. Stevenson, and M. Verani. An optimal adaptive fictitious domain method. *Mathematics of Computation*, 2019.

[17] S. Berrone, A. Bonito, and M. Verani. An adaptive fictitious domain method for elliptic problems. In *Advances in discretization methods*, pages 229–244. Springer, 2016.

[18] M. Berzins, Q. Meng, J. Schmidt, and J. C. Sutherland. Dag-based software frameworks for pdes. In *European Conference on Parallel Processing*, pages 324–333. Springer, 2011.

[19] D. Boffi, F. Brezzi, M. Fortin, et al. *Mixed finite element methods and applications*, volume 44. Springer, 2013.

[20] D. Boffi and L. Gastaldi. A finite element approach for the immersed boundary method. *Computers & structures*, 81(8-11):491–501, 2003.

[21] D. Boffi and L. Gastaldi. A fictitious domain approach with lagrange multiplier for fluid-structure interactions. *Numerische Mathematik*, 135(3):711–732, 2017.

[22] D. Boffi, L. Gastaldi, L. Heltai, and C. S. Peskin. On the hyper-elastic formulation of the immersed boundary method. *Computer Methods in Applied Mechanics and Engineering*, 197(25-28):2210–2231, 2008.

[23] D. Boffi, L. Gastaldi, and M. Ruggeri. Mixed formulation for interface problems with distributed lagrange multiplier. *Computers & Mathematics with Applications*, 68(12):2151–2166, 2014.

[24] Boost. Boost C++ Libraries. `http://www.boost.org/`, 2015. Last accessed 2015-06-30.

[25] B. A. Boville and P. R. Gent. The ncar climate system model, version one. *Journal of Climate*, 11(6):1115–1130, 1998.

[26] H. Brezis. *Functional analysis, Sobolev spaces and partial differential equations*. Springer Science & Business Media, 2010.

[27] F. Brezzi and M. Fortin. *Mixed and hybrid finite element methods*, volume 15. Springer Science & Business Media, 2012.

[28] W. D. Callister, D. G. Rethwisch, et al. *Materials science and engineering: an introduction*, volume 7. John Wiley & Sons New York, 2007.

[29] C. D'Angelo. Finite element approximation of elliptic problems with dirac measure terms in weighted spaces: applications to one-and three-dimensional coupled problems. *SIAM Journal on Numerical Analysis*, 50(1):194–215, 2012.

[30] C. D'Angelo and A. Quarteroni. On the coupling of 1d and 3d diffusion-reaction equations: application to tissue perfusion problems. *Mathematical Models and Methods in Applied Sciences*, 18(08):1481–1504, 2008.

[31] J. J. Duistermaat and J. A. Kolk. *Multidimensional real analysis I: differentiation*, volume 86. Cambridge University Press, 2004.

[32] K. S. Eloh, A. Jacques, and S. Berbenni. Development of a new consistent discrete green operator for fft-based methods to solve heterogeneous problems with eigenstrains. *International Journal of Plasticity*, 116:1–23, 2019.

[33] C. Ericson. *Real-Time Collision Detection*. The Morgan Kaufmann Series in Interactive 3d Technology. CRC Press, 2004.

[34] L. C. Evans. *Partial differential equations*. American Mathematical Society, 2010.

[35] R. Gassmöller, E. Heien, E. G. Puckett, and W. Bangerth. Flexible and scalable particle-in-cell methods for massively parallel computations. *arXiv preprint arXiv:1612.03369*, 2016.

[36] V. Girault and R. Glowinski. Error analysis of a fictitious domain method applied to a dirichlet problem. *Japan Journal of Industrial and Applied Mathematics*, 12(3):487, 1995.

[37] R. Glowinski, T.-W. Pan, T. I. Hesla, and D. D. Joseph. A distributed lagrange multiplier/fictitious domain method for particulate flows. *International Journal of Multiphase Flow*, 25(5):755–794, 1999.

[38] R. Glowinski, T.-W. Pan, and J. Periaux. A fictitious domain method for dirichlet problem and applications. *Computer Methods in Applied Mechanics and Engineering*, 111(3-4):283–303, 1994.

[39] R. Glowinski, T.-W. Pan, and J. Periaux. A fictitious domain method for external incompressible viscous flow modeled by navier-stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 112(1-4):133–148, 1994.

[40] M. E. Gurtin. *An Introduction to Continuum Mechanics*. Mathematics in Science and Engineering 158. Academic Press, 1981.

[41] M. E. Gurtin, E. Fried, and L. Anand. *The mechanics and thermodynamics of continua*. 2010.

[42] W. Hackbusch and S. A. Sauter. Composite finite elements for the approximation of pdes on domains with complicated micro-structures. *Numerische Mathematik*, 75(4):447–472, 1997.

[43] J. C. Halpin and J. Kardos. The halpin-tsai equations: a review. *Polymer engineering and science*, 16(5):344–352, 1976.

[44] Z. Hashin. On elastic behaviour of fibre reinforced materials of arbitrary transverse phase geometry. *Journal of the Mechanics and Physics of Solids*, 13(3):119–134, 1965.

[45] Z. Hashin. *Theory of fiber reinforced materials*. NASA, Washington, United States, 1972.

[46] Z. Hashin. Analysis of composite materials—a survey. *Journal of Applied Mechanics*, 50(3):481–505, 1983.

[47] T. Heister, J. Dannberg, R. Gassmöller, and W. Bangerth. High accuracy mantle convection simulation through modern numerical methods. II: Realistic models and problems. *Geophysical Journal International*, 210(2):833–851, 2017.

[48] L. Heltai. On the stability of the finite element immersed boundary method. *Computers & Structures*, 86(7-8):598–617, 2008.

[49] L. Heltai and A. Caiazzo. Multiscale modeling of vascularized tissues via non-matching immersed methods. *International Journal for Numerical Methods in Biomedical Engineering*, 2019. In press.

[50] L. Heltai and F. Costanzo. Variational implementation of immersed finite element methods. *Computer Methods in Applied Mechanics and Engineering*, 229-232, 2012.

[51] L. Heltai and N. Rotundo. Error estimates in weighted sobolev norms for finite element immersed interface methods. *Computers and Mathematics with Applications*, 2019. In press.

[52] L. Heltai, S. Roy, and F. Costanzo. A fully coupled immersed finite element method for fluid structure interaction via the deal. ii library. *arXiv preprint arXiv:1209.2811*, 2012.

[53] G. A. Holzapfel et al. Biomechanics of soft tissue. *The handbook of materials behavior models*, 3:1049–1063, 2001.

[54] G. A. Holzapfel and T. C. Gasser. A viscoelastic model for fiber-reinforced composites at finite strains: Continuum basis, computational aspects and applications. *Computer methods in applied mechanics and engineering*, 190(34):4379–4403, 2001.

[55] G. A. Holzapfel, T. C. Gasser, and R. W. Ogden. A new constitutive framework for arterial wall mechanics and a comparative study of material models. *Journal of elasticity and the physical science of solids*, 61(1-3):1–48, 2000.

[56] P. A. Huijing. Muscle as a collagen fiber reinforced composite: a review of force transmission in muscle and whole limb. *Journal of biomechanics*, 32(4):329–345, 1999.

[57] M. A. Hyman. Non-iterative numerical solution of boundary-value problems. *Applied Scientific Research, Section B*, 2(1):325–351, 1952.

[58] K. Iizuka, M. Ueda, T. Takahashi, A. Yoshimura, and M. Nakayama. Development of a three-dimensional finite element model for a unidirectional carbon fiber reinforced plastic based on x-ray computed tomography images and the numerical simulation on compression. *Advanced Composite Materials*, 28(1):73–85, 2019.

[59] T. Konopczyński, D. Rathore, J. Rathore, T. Kröger, L. Zheng, C. S. Garbe, S. Carmignato, and J. Hesser. Fully convolutional deep network architectures for automatic short glass fiber semantic segmentation from ct scans. *arXiv preprint arXiv:1901.01211*, 2019.

[60] E. Kreyszig. *Differential geometry*. University of Toronto Press, 1963.

[61] S. K. Kyriacou, J. D. Humphrey, and C. Schwab. Finite element analysis of nonlinear orthotropic hyperelastic membranes. *Computational Mechanics*, 18(4):269–278, Jul 1996.

[62] G. A. L. Heltai. The deal.ii tutorial step-60: non-matching grid constraints through distributed lagrange multipliers. 2018.

[63] S.-Y. Lee. *Accelerator physics*. World scientific publishing, 2018.

[64] X. Li, J. Lowengrub, A. Rätz, and A. Voigt. Solving pdes in complex geometries: a diffuse domain approach. *Communications in mathematical sciences*, 7(1):81, 2009.

[65] Z. Lu, Z. Yuan, and Q. Liu. 3d numerical simulation for the elastic properties of random fiber composites with a wide range of fiber aspect ratios. *Computational Materials Science*, 90:123–129, 2014.

[66] V. Lucas, J.-C. Golinval, S. Paquay, V.-D. Nguyen, L. Noels, and L. Wu. A stochastic computational multiscale approach; application to mems resonators. *Computer Methods in Applied Mechanics and Engineering*, 294:141–167, 2015.

[67] M. Maier, M. Bardelloni, and L. Heltai. `LinearOperator` – a generic, high-level expression syntax for linear algebra. *Computers and Mathematics with Applications*, 72(1):1–24, 2016.

[68] P. K. Mallick. *Fiber-reinforced composites: materials, manufacturing, and design*. CRC press, 2007.

[69] J. E. Marsden and T. J. Hughes. *Mathematical foundations of elasticity*. Courier Corporation, 1994.

[70] B. Maury. Numerical analysis of a finite element/volume penalty method. *SIAM Journal on Numerical Analysis*, 47(2):1126–1148, 2009.

[71] P. K. Mehta. *Concrete. Structure, properties and materials*. McGraw-Hill, 1986.

[72] N. Moës, J. Dolbow, and T. Belytschko. A finite element method for crack growth without remeshing. *International journal for numerical methods in engineering*, 46(1):131–150, 1999.

[73] M. Mooney. A theory of large elastic deformation. *Journal of applied physics*, 11(9):582–592, 1940.

[74] H. Moulinec and P. Suquet. A numerical method for computing the overall response of nonlinear composites with complex microstructure. *Computer methods in applied mechanics and engineering*, 157(1-2):69–94, 1998.

[75] H. Moulinec and P. Suquet. Comparison of fft-based methods for computing the response of composites with highly contrasted mechanical properties. *Physica B: Condensed Matter*, 338(1-4):58–60, 2003.

[76] T. Nakamura and S. Suresh. Effects of thermal residual stresses and fiber packing on deformation of metal-matrix composites. *Acta metallurgica et materialia*, 41(6):1665–1681, 1993.

[77] P. Neff, D. Pauly, and K.-J. Witsch. Poincaré meets korn via maxwell: extending korn's first inequality to incompatible tensor fields. *Journal of Differential Equations*, 258(4):1267–1302, 2015.

[78] J. A. Nitsche. On korn's second inequality. *RAIRO. Analyse numérique*, 15(3):237–248, 1981.

[79] Y. Pan, L. Iorga, and A. A. Pelegri. Analysis of 3d random chopped fiber reinforced composites using fem and random sequential adsorption. *Computational Materials Science*, 43(3):450–461, 2008.

[80] C. S. Peskin. The immersed boundary method. *Acta Numerica*, 11(1):479–517, jan 2002.

[81] K. L. Pickering, M. A. Efendy, and T. M. Le. A review of recent developments in natural fibre composites and their mechanical performance. *Composites Part A: Applied Science and Manufacturing*, 83:98–112, 2016.

[82] A. C. Pipkin. Integration of an equation in membrane theory. *Zeitschrift für angewandte Mathematik und Physik ZAMP*, 19(5):818–819, Sep 1968.

[83] S. J. Plimpton, B. Hendrickson, and J. R. Stewart. A parallel rendezvous algorithm for interpolation between multiple grids. *Journal of Parallel and Distributed Computing*, 64(2):266–276, 2004.

[84] A. Quarteroni and A. Valli. Theory and application of steklov-poincaré operators for boundary-value problems. In *Applied and Industrial Mathematics*, pages 179–203. Springer, 1991.

[85] S. S. Rao. *The Finite Element Method in Engineering*. Elsevier, 2013.

[86] R. Rivlin. Large elastic deformations of isotropic materials iv. further developments of the general theory. *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 241(835):379–397, 1948.

[87] N. Rotundo, T.-Y. Kim, W. Jiang, L. Heltai, and E. Fried. Error analysis of a b-spline based finite-element method for modeling wind-driven ocean circulation. *Journal of Scientific Computing*, 69(1):430–459, 2016.

[88] S. Roy, L. Heltai, and F. Costanzo. Benchmarking the immersed finite element method for fluid-structure interaction problems. *Computers and Mathematics with Applications*, 69:1167–1188, 2015.

[89] J. E. R.W Hockney. *Computer Simulation Using Particles*. CRC Press, 1988.

[90] E. Ryder. Nonlinear guided waves in fibre optics. 1993.

[91] A. Sartori, N. Giuliani, M. Bardelloni, and L. Heltai. deal2lkit: a toolkit library for deal.ii. Technical Report 57/2015/MATE, SISSA, 2015.

[92] A. Sartori, N. Giuliani, M. Bardelloni, and L. Heltai. deal2lkit: A toolkit library for high performance programming in deal.II. *SoftwareX*, 7:318–327, 2018.

[93] V. K. Saul'ev. On the solution of some boundary value problems on high performance computers by fictitious domain method. *Siberian Math. J*, 4(4):912–925, 1963.

[94] K. Schloegel, G. Karypis, and V. Kumar. Parallel static and dynamic multi-constraint graph partitioning. *Concurrency and Computation: Practice and Experience*, 14(3):219–240, 2002.

[95] S. Slattery, P. Wilson, and R. Pawlowski. The data transfer kit: a geometric rendezvous-based tool for multiphysics data transfer.

[96] S. R. Slattery. Mesh-free data transfer algorithms for partitioned multiphysics problems: Conservation, accuracy, and parallelism. *Journal of Computational Physics*, 307(Supplement C):164 – 188, 2016.

[97] B. Stroustrup. *The C++ Programming Language, 4th edition*. Addison-Wesley, 2013.

[98] C. Tang. An orthogonal coordinate system for curved pipes (correspondence). *IEEE Transactions on Microwave Theory and Techniques*, 18(1):69–69, 1970.

[99] L. R. G. Treloar. Stress-strain data for vulcanized rubber under various types of deformation. *Rubber Chemistry and Technology*, 17(4):813–825, 1944.

[100] C. Truesdell and W. Noll. The non-linear field theories of mechanics. In *The non-linear field theories of mechanics*, pages 1–579. Springer, 2004.

[101] A. Zaoui. Continuum micromechanics: survey. *Journal of Engineering Mechanics*, 128(8):808–816, 2002.